Large Systems

GCOS 8 OS
System Startup

Honeywell Bull

LARGE SYSTEMS

GCOS 8 OS SYSTEM STARTUP

SUBJECT

Initialization of Honeywell Bull Large Information Processing Systems Operating Under General Comprehensive Operating Supervisor 8 (GCOS 8) Software

SPECIAL INSTRUCTIONS

This revision supersedes DH18-02, dated December 1985. Change bars in the margins indicate technical additions and changes, and asterisks indicate deletions. Refer to the Software Release Bulletin (SRB) for this product release for a synopsis of technical changes.

SOFTWARE SUPPORTED

GCOS 8 OS Software Release 3000

ORDER NUMBER

DH18-03

August 1987

Honeywell Bull

PREFACE

The GCOS 8 OS System Startup manual describes software-related procedures for starting up a newly installed system and for restarting that system. This manual is directed at site personnel who must prepare the Startup job stream and at operations personnel who must interface with the Startup program via the system console.

Hardware operating procedures that are necessary to System Startup (e.g., mainframe and peripheral switch settings) are not included in this manual. Refer to the appropriate hardware reference manuals for description of switch settings.

In addition, this manual does not define Startup procedures for the Front-end Network Processors. Refer to the appropriate communications manuals for this information.

The major sections of this manual are:

- Section 1 presents an overview of the Startup program and the Startup job stream.
- Section 2 describes techniques for bootloading the system and the operator (console) interface with the Startup program.
- Section 3 describes procedures for recovery from error conditions that occur during the system startup process.
- Section 4 describes the procedure for loading Integrated Software and for modifying default parameters for Integrated Software.
- Section 5 describes the various statements that can be included in each section of the Startup job stream.

In this document, multiple vertical braces and brackets should be assumed to be a single brace or bracket; for example:

```
{   }                          [   ]
{   }   represent  {  }   and  [   ]   represent  [  ]
{   }                          [   ]
```

The information and specifications in this document are subject to change without notice. Consult your Honeywell Bull Marketing Representative for product or service availability.

 File No.: 1V13

CORRECTING MANUALS VIA PASS

Required corrections to this version of the manual will be entered on the Problem Analysis Solution System (PASS) data base. Query PASS periodically via the procedures described in the Problem Analysis Solution System (PASS) manual, Order Number DW95. Corrections documented on PASS, if applicable to the next release of the software, will be incorporated into the next update of the manual.

LISTING OF MANUALS

A list of large system software manuals is available to all Honeywell Bull users with access to an ASCII terminal. The manuals list is updated regularly to facilitate the ordering of manuals. Manual ordering instructions are printed on the listing. Use the following instructions to obtain a listing.

1. Dial the appropriate telephone number to connect your terminal to the Multics system in Phoenix.

	300-baud & 1200-baud	150-baud
(602)	249-5356 249-6430	249-7554

   Multics will respond with a computer system identification banner.

2. Enter the login command "login Sam" (without the quote marks). Press the carriage return key.

   Multics will respond with a request for the password.

3. Enter the password "Multics" (without the quote marks) and press the carriage return key.

   Multics will respond with a welcome message followed by a ready message.

   ```
   Example:  Welcome to the Multics system
             For services available online type:
                  :list
             r1111.7 Thu   (ready message)
   ```

4. Enter ":list" (without the quote marks) to obtain a list of commands. Press the carriage return key.

   Multics will list the commands available for specific topics and then display the ready message.

5. Enter the command selected and press the carriage return key.

6. Enter "logout" (without the quote marks) and press the carriage return key to log off the Multics system.

   Multics will respond with a logout message.

CONTENTS

CONTENTS (cont)

CONTENTS (cont)

CONTENTS (cont)

CONTENTS (cont)

CONTENTS (cont)

CONTENTS (cont)

CONTENTS (cont)

ILLUSTRATIONS

TABLES

SECTION 1

SYSTEM STARTUP OVERVIEW

The purpose of System Startup is to make operational a newly installed system or to restart a previously functioning system that was rendered inoperable by a malfunction. System Startup, also referred to as bootloading, can be divided into four main functions.

1. Bootloading firmware into the following input/output controllers:

   o IMU on DPS 8

   o IMX on DPS 8000

   o IOX on DPS 88

   o IOP on DPS 90

2. Bootloading of the Startup program

3. Establishing the system environment

4. Executing control statements within the Startup job stream (including bootloading firmware in all MPCs (Micro Programmable Controllers) on DPS 8 and DPS 8000 systems)

The Startup procedure is designed to give the site as much control as possible over its operating environment. Many of the events within the overall bootload process are operator-directed, while others are performed by the Startup program with no operator intervention required. In addition, a great deal of flexibility is allowed when constructing the Startup job stream, which defines the hardware and software environments that best meet the site's processing needs.

This section presents an overview of System Startup in terms of the preceding four functions. Detailed information is the subject of subsequent sections.

BOOTLOAD STARTUP PROGRAM

The Bootloader program, which loads the Startup program into main memory, in actuality, is the front-end of the Startup program.

The Bootloader program must be executed if the Startup program is to be loaded into main memory from magnetic tape. In the process of bootloading the Startup program, the Bootloader program also applies to Startup any existing MASK correction statements. Once Startup is in memory and placed in execution, the remainder of the System Startup procedure comes under its control.

Two additional procedures, neither utilizing the Bootloader program, are frequently used for loading the Startup program into main memory when the Startup program resides on mass storage. Under one procedure, the Startup program is loaded into main memory by the Master Mode Dump program (.MDUMP) from the AUTOLOAD file. Loading occurs as the result of the console entry AUTO or REPL in response to the question *BOOT SOURCE: TAPE, AUTO, OR REPL?.

*

The second procedure, sometimes called a "direct bootload", loads the Startup program from the AUTOLOAD file as the result of the console entry AUTO or REPL in response to the BOOT verb.

ESTABLISH SYSTEM ENVIRONMENT

Once loaded into main memory, the first function of the Startup program is to establish the system environment. This involves a sequence of events designed to reset memory locations to a known status and to establish the identity of the system. Startup performs the following functions at this time:

- Initialize all of main memory to zero
- Verify memory size
- Initialize certain communication region locations and apply default values
- Invoke the fast boot option (if warranted by conditions)
- Carry forward the bootload date and time (if the bootload is initiated by the Master Mode Dump program)
- Establish console tab settings

EXECUTE STARTUP JOB STREAM

After establishing the system environment, the Startup program processes the Startup job stream. This job stream, which comprises up to six separate sections, can reside on magnetic tape, or on the AUTOLOAD file on mass storage. These sections are $CONFIG, $INITIALIZE, $EDIT, $FILES, $PATCH, and $LOAD. The following briefly describes Startup's processing of each section. Section 5, "Startup Job Stream", describes the Startup job stream in detail.

$CONFIG Section

The $CONFIG section establishes system software and system hardware parameters. These parameters are contained in the communication region. As a result of $CONFIG section processing, Startup surveys and verifies processors and issues several questions to the system console. These questions are the first in a series that give the operator some control over the bootload process. Questions include *RESTART? *SYSOUT RECOVERY, *SCF continuation, *DATE?, and *TIME?

The site can use the $ ANSWER statement in this section to provide answers to questions that otherwise are issued to the system console by the Startup and Master Mode Dump (.MDUMP) programs.

$INITIALIZE Section

Device initialization, which is the re-establishment of tables and pointers on mass storage devices, is performed during processing of the $INITIALIZE section. This section is processed by Startup only if the operator makes a positive response (i.e., YES, TOTAL, or PARTIAL) to the *INITIALIZE? question issued to the console.

Optional functions performed only as a result of a positive response to the *INITIALIZE? question include:

- Formatting mass storage devices
- Defining mass storage defective space
- Creating the mass storage device label and allocation tables
- Bootloading Microprogrammable Peripheral Controllers (MPCs) (DPS 8 only)
- Creating the deckfile, which contains MPC firmware on mass storage
- Initializing the Startup Master Catalog

The Startup master catalog - this is not the System Master Catalog - contains information regarding all Startup-created files and resides on the device named in the $ GCOSFIL statement. Files that are defined on the Startup master catalog subsequently are recataloged within the File Management Supervisor (FMS) program's catalog and file structure under the SYS_CAT SMC entry.

Several functions are performed by Startup regardless of the operator's response to the *INITIALIZE? question. This portion of the Startup processing sequence is referred to as the DEVICE ROLLCALL. Startup functions include recovery or re-creation of mass storage device labels, verification of all permanent mass storage devices, surveying of magnetic tape handlers, surveying of unit record devices, and loading of printer character set and Vertical Format Control (VFC) images.

$EDIT Section

The $EDIT section establishes the identity of all GCOS system files and the device on which each file is to reside. Processing of the $EDIT section is dependent upon the operator's response to the *EDIT? question that is issued to the console. However, if the operator responded TOTAL to the *INITIALIZE? question, the *EDIT? question is automatically answered YES by Startup.

$FILES Section

The $FILES section selectively identifies those GCOS system files, defined in the $EDIT section, that the site intends to access during the system processing following this bootload. Consequently, only system programs residing on the files identified in this section can be used during system processing. System file and program data are written to main memory tables.

$PATCH Section

The $PATCH section is a means by which patch corrections (OCTAL statements and PATCH_Run_Unit directives) can be applied to GCOS system programs and to system programs that were edited by the Startup program onto the system. This includes Run Units installed through the boot process. Corrections can replace the content of single locations within a module or allow the site to select specific operating parameters for system software modules and programs. The second type of correction is referred to as a site-option patch.

Patch corrections are written to an in-memory table, from which they are applied to GCOS system programs/modules when the applicable programs/modules are called into memory.

In addition, the Startup program and job stream are written to the AUTOLOAD file on mass storage during processing of the $PATCH section.

$LOAD Section

The $LOAD section can be used to load modules into main memory. Any module loaded via the $LOAD section preempts the version of the module existing on mass storage (i.e., the module is copied to mass storage from the total system software tape).

Among the modules that can be loaded via this section are site-prepared modules that do not exist on the total system software tape. The content of this section is written to the load file on mass storage. Subsequently, modules can be loaded from the load file.

*

After loading modules from the $LOAD section into main memory, Startup loads required system modules from mass storage files. The Memory Allocator (.MPOPM) program is the last system module loaded from mass storage. The module initializes several additional tables required for system processing (e.g., device allocation tables). (Note that the .MPOPM module also can be loaded into memory via the $LOAD section. If loaded in this manner, the module is temporarily saved in a buffer. After all other modules are loaded into memory, the .MPOPM module is loaded.) During .MPOPM processing, the initial complement of system run unit software is installed into the system and made available for use.

Upon completion of $LOAD section processing, Startup passes control to .MPOPM, which completes the boot process and initiates the GCOS 8 system.

SECTION 2

BOOTLOADING THE SYSTEM

The term "system bootload" refers to the process of starting up (i.e., making operational) a system that is not operating. The system bootload comprises a series of integrated functions performed by the Startup program, using information in the Startup job stream, and which may require operator intervention via console entries in response to messages issued by the Startup program.

Several techniques can be employed to effect the system bootload. Preference of one technique over another is usually dictated by the circumstances precipitating the bootload and by the operating environment (i.e., hardware configuration and software capabilities) the site wants to achieve from the bootload. These procedures require varying degrees of operator intervention. In some cases, little or no operator-Startup program communication is required. In others, the operator must direct Startup to initiate or bypass specific processing functions.

This section defines and describes the various system bootloading techniques available for starting a system. The interface between the Startup program and the operator is described in detail, including the applicability of optional responses to questions issued during the bootload procedure.

Procedures for generating a site-specific boot tape online, starting from the Honeywell Bull-supplied site-customized boot tape, without a card reader, and for booting from that tape, are described in the Software Installation Bulletin.

DPS 88 AND DPS 90 FIRMWARE AND STARTUP BOOTLOADING

DPS 88 and DPS 90 firmware must be loaded before Startup is bootloaded. The procedures for bootloading firmware and the Startup program in these systems are defined in the DPS 88 Hardware Operations Manual and the DPS 90 IOP Startup Manual.

On the DPS 90, Startup does not allow the operator to edit the DECKFILE or load the firmware (i.e., does not allow the use of the $ MPCFIG statement).

After the Startup program has been placed in execution on the DPS 88 or the DPS 90, proceed to "Startup-Operator Interface" later in this section.

DPS 8 SYSTEMS WITH IMU CONTROLLERS

The IMU controllers need to have firmware loaded before the Startup program can be bootloaded. Refer to the IMU Hardware Operations Manual for the procedures. At this time, firmware is loaded into the IMU but not into the MPCs. Firmware is loaded into the MPCs by Startup in response to the *INITIALIZE? question, described later in this section.

BOOTLOADING

System startup/restart is performed by the Startup program, which bootloads the system in various ways, depending upon the state of system hardware, the status of software on mass storage, and parameters in the Startup job stream. In a specific situation, some parts of the bootload operation may not be necessary. Appropriate entries in the Startup job stream, or appropriate answers to questions displayed on the console cause unnecessary parts to be bypassed.

Bootload Operation	System Status And Functions Performed
Power-up	Power has just been applied to equipment that depends on firmware for operation; firmware must be loaded.
Cold boot	GCOS software on mass storage is defective or nonexistent; mass storage devices must be initialized and/or edited.
Warm boot	GCOS software on mass storage is valid; mass storage devices do not need to be initialized or edited. Startup questions need to be answered so that GCOS can be restarted.
Fast boot without FASTBT file	Same conditions as warm boot, except that the answers to all Startup questions are known in advance via $ ANSWER, and Startup job stream includes $ INFO FASTBT (but not $ FILCREA FASTBT).
Fast boot with FASTBT file	Same as fast boot without FASTBT file, except that the Startup job stream includes $ FILCREA FASTBT.

If the AUTOLOAD file already contains a valid, accessible copy of the Startup program and the Startup job stream, the AUTOLOAD file should be used as the input for the bootload. Otherwise, the site-specific boot tape should be used for input. Completion of the bootload requires less time when the Startup program and job stream are obtained from the AUTOLOAD file.

The AUTOLOAD file resides on the device defined via the $ AUTOLD statement in the $CONFIG section of the Startup job stream. If a specific device is not defined, the file is created on the first nondedicated mass storage device specified on the $ GCOSFIL statement in the $CONFIG section. (Refer to "$ AUTOLD Statement" and "$ GCOSFIL Statement" in Section 5 for detailed descriptions.)

Each time the system is bootloaded using a Startup program and job stream from magnetic tape, both the program and the job stream are copied to the AUTOLOAD file to overlay (replace) the version of the Startup program and job stream that previously resided on the AUTOLOAD file.

Bootloading clears in-memory system tables and uses recovery file data to reestablish system conditions as they existed at the time of the failure. System programs are restarted from a predefined point. User-level jobs are either reentered or restarted unless restart is not possible or desired.

During system operation, recovery data is saved at appropriate times to ensure that sufficient information is available for restart.

The Master Mode Dump (.MDUMP) program is invoked at the time of a system failure for the purpose of saving recovery data and failure evidence on mass storage files. The 64-word System Recovery Table, which resides on the mass storage device identified as ST1, contains 16 four-word entries defining the locations of the recovery files for privileged slave programs. (Refer to the GCOS 8 OS System Operating Techniques manual for a description of this table.)

Recovery data also is periodically written to mass storage files while the system is operational. The events at which data is written to the recovery files are referred to as cleanpoints and are controlled by system software. (The saving of cleanpoint data is enabled by including the $ INFO CLENPT statement in the $CONFIG section of the Startup job stream.) When a system failure occurs, the restart function utilizes data written during the most recent cleanpoint event for each process.

Power-up Operation

When electric power is removed, and then reapplied to an entire system, the mass storage devices are not accessible, even though they may contain valid and useful information (e.g., the Startup program and Startup job stream in the AUTOLOAD file, and the firmware in the DECKFILE). The mass storage devices do not become accessible until firmware has been loaded. That firmware must be read using a magnetic tape device, which also requires firmware. However, MPC-driven magnetic tape subsystems can be equipped with the Reset Out (RSO) hardware option, which provides primitive mode operations, sufficient to allow reading of the MPCB program and needed firmware from tape for loading into the mass storage controller and the magnetic tape controller. Under these conditions the sequence of statements in the $INITIALIZE section is critical, to ensure that all controllers have firmware loaded before accessing devices, etc. See "Importance Of Sequence In $INITIALIZE Section" in Section 5.

Cold Boot

When GCOS 8 is not functioning (i.e., is not in main memory and cannot be accessed on mass storage), a cold boot (answer TOTAL, YES, or PARTIAL to the *INITIALIZE? question, or answer YES or PARTIAL to the *EDIT? question) is required to rebuild system software in its entirety on mass storage files. There are several conditions under which the GCOS system can become inoperable (e.g., system files may be destroyed when a system failure occurs). A cold boot is also required when installing system software at a new site. In addition, for DPS 8 and DPS 8000, MPC firmware is loaded onto mass storage during a cold boot.

Four sources of input are required for a cold boot:

1. The total system software tapes (TSST), containing GCOS system software, which are mounted on the magnetic tape handlers defined by logical device name on $ FILDEF statements in the $EDIT section of the Startup job stream.

2. The tape containing the MPC firmware to be copied into the deckfile (DPS 8 and DPS 8000).

3. The Startup program, residing on either the AUTOLOAD file or the site-specific boot tape.

4. The Startup job stream, residing on either the AUTOLOAD file or the site-specific boot tape. For DPS 8 and DPS 8000 systems, the $INITIALIZE section of the Startup job stream must contain the MPC Bootload (MPCB) program deck image, and the two MPC firmware data deck images. MPCB loads firmware data into the MPCs; one data deck bootloads the magnetic tape controller and the other bootloads the mass storage controller for the AUTOLOAD device. On DPS 88 and DPS 90 systems, Startup issues a warning message and ignores any MPC bootload (MPCB) deck.

Warm Boot

The purpose of the warm boot is to restart the system without reloading or rebuilding system software on mass storage (answer NO to *EDIT? question). A warm boot can be performed whenever a valid version of GCOS software is accessible on mass storage, regardless of whether the GCOS system is currently functioning in main memory. The only inputs to a warm boot are the Startup program, residing on mass storage or magnetic tape, and the Startup job stream, also residing on mass storage or magnetic tape.

The warm boot allows the operator to control the restart procedure at several places where it questions the operator. Site management can reduce the time required for the warm boot by using $ ANSWER statements in the Startup job stream to provide answers to a majority of the Startup questions. Any question that is not answered on a $ ANSWER statement is directed to the console for operator response. (Refer to "Startup-Operator Interface", later in this section, and to "$ ANSWER Statement" in Section 5 for a discussion of Startup questions.)

Not all Startup questions can be answered via the $ ANSWER statement. The operator is required to respond to some questions with a console entry (e.g., the *CHANGE? question). In addition, a YES response to the *CHANGE? question causes the $ ANSWER statements to be ignored, forcing the operator to respond to each question during the bootload process. (Refer to "*CHANGE? Question", later in this section for additional information.)

A warm boot may be performed when the GCOS system is functioning in main memory. The need for such a bootload may occur under two conditions. Procedures for bootloading under these circumstances differ slightly.

1. The system malfunctions and the fast boot feature for automatic system restart is not in effect. Because the GCOS system is accessible on mass storage, the operator is only required to load the Startup job stream.

2. Startup job stream modifications are required while the system is operational. To halt the system, the operator enters BOOT via the system console. The changes can be made with the Startup Console Editor, as described later in this section, or the entire job stream or the modified section(s) therein can be loaded from punched cards via the card reader.

   The preceding procedure can also be used when the operator requests a dump of system software by entering DUMP via the system console.

Fast Boot

A fast boot (with or without the FASTBT file) is a warm boot that bypasses all operator intervention following a system failure. Because GCOS software is functional in memory and the Startup program can be accessed immediately on the AUTOLOAD file, system restart occurs within a few minutes following a failure.

The fast boot feature is enabled by including the $ INFO FASTBT statement in the $CONFIG section of the Startup job stream or via the console entry FASTBT ON after GCOS is in operation. Conversely, the fast boot feature can be deactivated and the warm boot procedure activated via the console entry FASTBT OFF. (Refer to "$ INFO Statement" in Section 5 and to the GCOS 8 OS System Operator Messages manual for a description of the FASTBT ON and FASTBT OFF entries.)

In addition to including the $ INFO FASTBT statement in the Startup job stream, three other conditions must be satisfied before the fast boot feature is invoked:

1. The Startup program must reside on the AUTOLOAD file.

2. Several questions generated during System Startup must be answered via the $ ANSWER statement. Failure to use $ ANSWER to answer a question causes issuance of that question to the system console, thereby requiring an operator response.

3. At least eight minutes must have elapsed since the most recent previous system abort.

Restart is attempted for all jobs that were in process at the time of the failure if the *RESTART? question is answered YES (via $ ANSWER RESTART/YES) and if the *SYSTEM SCHEDULER CLEAR? question is answered NO (via $ ANSWER SSCLEAR/NO). The Transaction Processing System (TPS) and the Time Sharing System, which otherwise require operator intervention to restart, also are automatically restarted if a $ INFO ACALL statement is included in the $CONFIG section of the Startup job stream.

The three primary reasons for a fast boot restarting the system in less time than a warm boot are:

- o All operator intervention is circumvented by using the $ ANSWER statement to provide answers to Startup questions.
- o Existing job-related system table information is used. This avoids the process of re-creating these tables, which requires additional time during the warm boot process.
- o MPC bootloading and printer character set loading are bypassed.

Several conditions may prevent restart via the fast boot procedure. These include destruction of one or more critical system tables or loss of a critical system hardware component. In addition, although it is a rarer occurrence than loss of system tables or system components, a user program can create a condition that causes a system failure. This could result in a situation whereby each time the fast boot recovers the system to the point of encountering the problematic job, another system failure occurs. To correct this situation, the operator can either delete the job from the system via the ABORT sssss console entry (where sssss = SNUMB) or inhibit the fast boot feature via the console entry FASTBT OFF.

If a system program causes the fast boot to abort, the message:

    ENTER [(O)PTION], (U)SE $ ANSWER OR (N)ONE?

is issued to the system console, requesting the operator to indicate the extent of the dump. Following the dump, system restart is via a warm boot.

Specification of the $ INFO ROLLCALL option interrupts the fast boot process, which otherwise bypasses the rollcall function.

Fast Boot With FASTBT File

Additional time can be saved during a fast boot if a $ FILCREA FASTBT statement is included in the $CONFIG section of the Startup job stream. This statement establishes the system FASTBT file which contains an image of the first portion (approximately 256K words) of memory as it is built by the Startup program. When the FASTBT file is present, the fast boot procedure reestablishes the contents of memory from the FASTBT file and bypasses most Startup job stream processing.

Bootload Procedures

This section describes procedures to be used when bootloading from mass storage, and from magnetic tape.

BOOTLOAD FROM MASS STORAGE

To do a bootload from mass storage, the AUTOLOAD file must be intact. The AUTOLOAD file resides on the device defined via the $ AUTOLD statement in the $CONFIG section of the Startup job stream. (Refer to "$ AUTOLD Statement" in Section 5 for a detailed description.)

The content of the AUTOLOAD file is produced by the bootload of the system from magnetic tape, in which the Startup program and job stream are copied to the AUTOLOAD file. When Startup creates or updates the AUTOLOAD file it creates a bootload record in sector 2, and puts a pointer in word 14 of the device label of the AUTOLOAD device (i.e., the mass storage device on which the AUTOLOAD file resides). This record permits access to the Startup program in the AUTOLOAD file.

Bootload from mass storage is supported by the MSU0400, MSU0402, MSU0451, MSU0500, MSU0501, MSU3380 and MSU3381 subsystems.

The DPS 88 SMAS command to boot from mass storage is

```
BOOT <os_name> -AUTO
```

where <os_name> is the logical name of the operating system (GCOS 8) as established by the site through the SMAS configuration facility (ICON).

The DPS 90 command to boot from mass storage is

```
BOOT OPSY AUTO.
```

Once the AUTOLOAD file has been established, the Startup program maintains a pointer to the AUTOLOAD file so that a disk boot is always possible, provided that the AUTOLOAD file is readable, and provided that the AUTOLOAD file represents the configuration and options that are desired. Several additional steps are necessary if a bootload is initiated via offline T&D. Those steps are described under "AUTOLOAD File Not On Default Device Number (DPS 8 And DPS 8000 Only)", which follows.

If a read error is encountered reading the AUTOLOAD file the system will halt with a DIS and the AUTOLOAD file will have to be rebuilt.

AUTOLOAD File Not On Default Device Number (DPS 8 And DPS 8000 Only)

The following procedure is required to establish information for using the AUTOLOAD file to bootload directly from mass storage only if the AUTOLOAD file is not on device number one (the default). Note that DSC500 firmware revision A1 (or later) must be loaded to perform this bootload.

1. Examine the $ AUTOLD statement to determine the logical device name of the mass storage device on which the AUTOLOAD file resides.

2. Examine the input/output controller statements. Determine the physical device number of the mass storage device on which the AUTOLOAD file resides. The device can be determined by matching the logical device name (from #1) with the physical device number.

   NOTE: The default addressing executed by firmware is DEVICE-1.

3. Use the same input/output controller statement (from #2) to determine the channel number over which the AUTOLOAD device is to be accessed.

4. Examine the $ MPC statements. Determine both the PSI number and the associated channel number over which the AUTOLOAD device is to be accessed.

5. Convert the device number (from #2) to hexadecimal.

6. Create RHEX correction statements that identify the channel and the device over which bootloading is to occur. The following depicts the information that must be included on these statements.

   MPC CORRECTIONS:

Columns 1-4	Columns 16-17	Columns 18-19
04FE	(0) dd	(1) dd
04FF	(2) dd	(3) dd

   where:

   dd - Device number
   0-3 - PSI numbers, which are defined by the columns in which the device number is specified

   DAU(MSP800) CORRECTIONS:

Columns 1-4	Columns 16-17
02B0	(0) dd
02C0	(1) dd
02D0	(2) dd
02E0	(3) dd

where:

dd - Device number
0-3 - PSI numbers, i.e., Patch 02B0 for PSI-0, 02C0 for PSI-1, etc.

7. Insert the RHEX correction statements into the $INITIALIZE section of the Startup job stream. The RHEX statement immediately precedes the $ DKEND statement of the associated mass storage MPC firmware deck.

8. Examine all $ XBAR statements and go back to step 3 for each of the XBAR channels that may access a different MPC or PSI connection.

9. Patch the firmware tape with the RHEX correction statements. Refer to the offline Test and Diagnostics listing for program 08F for patching procedures. This step is required so that the firmware on the AUTOLOAD file reflects the same bootload device and channel information as the bootload deck in the $INITIALIZE section of the Startup job stream. *

10. Bootload the system. The following must be performed during the bootload process: (1) reboot the MPC, (2) include a $ READIN statement in the $INITIALIZE section of the Startup job stream, and (3) respond YES to the CLEAR AND OVERWRITE EXISTING DECKFILE? question issued to the console.

11. Issue the console verb, TEST IFAD when the system becomes operational, to install the patched firmware in the online T&D extension file so that the correct device numbers will be restored if online ITRs are run and the firmware is rebooted online.

DPS 8 And DPS 8000 Operator Procedure For Disk Bootload

The procedure to boot GCOS from an IMU is described in the IMU Hardware Operations Manual.

The following operator procedures are required to execute a disk bootload to boot GCOS from an IOM, once the firmware is active in the controller.

1. Set the BOOTLOAD SOURCE TAPE/CARD switch to TAPE (even though disk will actually be used) on any IOM/IMU/IMX that can access the AUTOLOAD device. Also, set the channel number code switches to identify the channel connected to the AUTOLOAD device as the bootload channel.

2. Press, in sequence, the INITIALIZE and BOOTLOAD push buttons on the system console or the IOM/IMU/IMX.

3. At this point, the Startup program issues the *REPLACE? question to the first system console configured on the bootload IOM/IMU/IMX (as recorded on the AUTOLOAD file which was written on the previous boot). If no such console was configured, the default console (IMU channel 30 or IOM channel 31 of the prior boot IMU/IOM or as specified on a MASK for location 3) is used. This boot will define the console for the next boot. (Refer to "*REPLACE Question" in this section for operator responses to this question.)

BOOTLOAD FROM MAGNETIC TAPE ON DPS 8 OR DPS 8000

The Reset Out (RSO) hardware option must be included with MPC-driven magnetic tape subsystems to allow primitive mode operation for DPS 8 and DPS 8000 systems. In primitive mode, the subsystem can process a Read Tape Binary command, which is required for reading the first record on the boot tape. For MPC-driven magnetic tape subsystems, the MPC switches must be set as follows:

Switch	Setting
0	Down (reset)
2	Controls bootloading of the MPC firmware Up (set) - Bootload firmware via Line Adapter 1 Down (reset) - Bootload firmware via Line Adapter 0
5-7	MPC port number to which the magnetic tape controller is connected

Before initiating the bootload, set the BOOTLOAD SOURCE TAPE/CARD switch on the input/output controller to TAPE. MPC switches must reflect the density of the boot tape (556, 800, or 1600 bpi), the number of tracks (seven-track or nine-track tape handler), and the magnetic tape handler (1-15) on which the boot tape is mounted. If the tape device is controlled by an IMU or IMX, refer to the Hardware Operations Manual to:

1. Configure the MCA
2. Initialize the IMU by loading the IMU firmware
3. Start the GCOS bootload process.

After mounting the boot tape on the tape handler specified by the MPC switches, and, if a cold boot is necessary, after mounting the total system software tapes on the magnetic tape handlers defined by logical device name on $ FILDEF statements in the $EDIT section of the Startup job stream, press the INITIALIZE and BOOTLOAD push buttons on the system console in sequence to initiate the bootload process. If no changes are to be made to the content of the Startup job stream between the boot tape and the AUTOLOAD file, respond NO to the *REPLACE? question.

If changes are to be made to the content of the Startup job stream between the boot tape and the AUTOLOAD file, via the Startup Console Editor, respond PREPASS to the *REPLACE? question (see "*REPLACE? Question", later in this section, for details).

If changes are to be made to the content of the Startup job stream between the boot tape and the AUTOLOAD file, via the card reader, respond YES to the *REPLACE? question (see "Startup-Operator Interface", later in this section, for details). The Startup job stream is read from the boot tape through the section header (e.g., $EDIT section) for the section that is to be replaced. At that point, the cards in the card reader are read, and then used to overlay (replace) the existing section on the AUTOLOAD file. After skipping the replaced section on the boot tape, the procedure is repeated for any other section(s) to be replaced, and then the tape is read to completion.

Note that the preceding process modifies only the AUTOLOAD file. The boot tape is unaffected. (Refer to "Modifying Content Of Site-Specific Boot Tape", later in this section, for tape modification procedures.)

During the bootload process, the operator may be prompted to identify the card reader to be used for reading the changes via one of the messages:

```
*NEED READER PORT, IOM, CHANNEL PICCC
   or
*NEED READER IOM, CHANNEL ICCC
```

The operator response is in the form piccc or iccc, where p = SCU-0 port number to which the IOM or IMU initiating the bootload process is cabled, i = IOM or IMU number, and ccc = channel number. (Refer to "*REPLACE? Question" later in this section and to "*BOOT SOURCE: AUTO OR REPL? Question" in Section 3 for additional information.)

If the card reader becomes inoperable while reading cards, the operator can press the REQUEST push button and respond to the ??? prompt with

```
READER piccc    or READER iccc
```

to cause Startup to switch to an operable card reader.

If the bootload is successful, the boot tape is automatically rewound and programmatically dismounted. If the bootload is unsuccessful, the message *ABORT BOOT FROM TAPE is issued to the system console. Check all switch settings, the tape handler, and the tape during the error correction process. (Refer to the GCOS 8 OS System Operating Techniques manual for magnetic tape media-inspection procedures.)

Modification of Startup Job Stream

Execution of the Startup program causes the Startup job stream to be read from one of two sources. The job stream may be read from the boot tape, in which case it is copied to the AUTOLOAD file, or it may be read directly from the AUTOLOAD file. Techniques are available for indirectly modifying the contents of the AUTOLOAD file, and also for producing or modifying the site-specific boot tape.

MODIFYING CONTENT OF AUTOLOAD FILE

The content of the boot tape is written to the AUTOLOAD file when the tape is used to bootload the system, to ensure that the AUTOLOAD file always contains the current copy of the Startup job stream.

On an AUTOLOAD boot, the Startup program and card images are read from the AUTOLOAD file. At the end of processing each section after the $INITIALIZE section, the Startup program and current card images are re-written to the AUTOLOAD file.

The Startup Console Editor can be used to modify the contents of memory (and thus the AUTOLOAD file) while the Startup program is executing. Refer to "Startup Console Editor" later in this section.

The XCAF (Extract Card Images From AUTOLOAD File) program can be used to extract the various sections of the Startup job stream into user-defined files. Refer to the GCOS 8 OS Service Routines manual for a description of the XCAF program.

Another way of modifying the contents of the AUTOLOAD file is by responding YES or PREPASS to the *REPLACE? question. Refer to "*REPLACE? Question" later in this section.

MODIFYING CONTENT OF SITE-SPECIFIC BOOT TAPE

In the GCOS environment, the RSBT (Read Startup Boot Tape) program can be used to convert the site-specific boot tape into a permanent file in standard system format, for use as an updating master. The *C Editor (SCED) program can then be used for updating this permanent file, after which the permanent file can be converted back to the boot tape format via the WSBT (Write Startup Boot Tape) program. Refer to the GCOS 8 OS Service Routines manual for descriptions of the *C Editor, RSBT, and WSBT programs. Note that although Time Sharing techniques can be used instead of the *C Editor for updating the permanent file, the job stream may include binary records which must be preserved.

If the Startup Console Editor has been used to modify the AUTOLOAD file, a new site-specific boot tape can be built by using the XCAF (Extract Card Images From AUTOLOAD File) program. Refer to "Startup Console Editor" later in this section, and to the GCOS 8 OS Service Routines manual for a description of the XCAF program.

STARTUP RESILIENCE

This feature allows the system to be rebooted following an interruption requiring units to be deconfigured without changing switches and without altering the Startup job stream to reflect the deconfiguration. This is achieved by allowing Startup to release input/output controllers and by allowing Startup to operate with "holes" in memory.

With this feature, systems can be split and split systems can be recombined with minimal modification to the Startup job stream. (See the GCOS 8 OS System Operating Techniques manual for a description of System Resiliency, i.e., splitting systems.)

Areas of defective memory are marked, hence can be assigned later when the system is fully operational. This is preferable to releasing defective memory (and all memory above it), thereby rendering it useless to the system until the next boot.

The communication region word .CRIRL is used as follows:

Bit 0 = 1 A mismatch exists between the size of the configured and physical memory. POPM is required to perform a memory survey during rollcall.

Bits 1-29 Reserved

Bit 32 = 1 Input/output controller 3 released

Bit 33 = 1 Input/output controller 2 released

Bit 34 = 1 Input/output controller 1 released

Bit 35 = 1 Input/output controller 0 released

The following Startup messages are pertinent:

```
*Iyy-n NOT RESPONDING - YES TO CONTINUE OR R TO RETRY
where Iyy may be IMU, IMX, IOM, IOP, or IOX.
```

This message indicates that the input/output controller specified as "n" did not respond to a connect. An operator response of YES permits Startup to continue with the specified input/output controller released. A NO response aborts Startup. A RETRY response causes Startup to send another connect to the same input/output controller to see if it will respond. This allows for a delay in case firmware is not yet loaded into an IMU. After the IMU firmware loading process has completed, the RETRY should be made and then the IMU should respond. This message can be suppressed by using a $ ANSWER IOMERROR/NO statement or a $ ANSWER IOMERROR/YES statement. *

```
*MPC-n CHECK SWITCHES:  2{UP  }    5{UP  }
                         {DOWN}     {DOWN}
```

This message indicates that the PSI through which firmware for this MPC is normally booted is connected to a released input/output controller. The operator should confirm that the switches are in the correct positions to enable the MPC to be booted with firmware. This message does not apply to those MPCs having no switches; with these, the firmware handles the situation.

```
**XXXXXXXX.....XXX
```

This message is from either the $ CHAN or the $ UNIT statement. Messages of this form are site-specific usually indicative of the action that should be taken to move peripherals or media from a released input/output controller. The operator should follow the specific site procedure.

```
*DEPRESS EOM WHEN READY TO CONTINUE
```

This message indicates that the operator is allowed time to follow instructions given, then, upon completion, pressing EOM reenables activity.

*Iyy-n RELEASED
where Iyy may be IMU, IMX, IOM, IOP, or IOX.

This message indicates that Startup released the specified input/output controller.

* The following pertinent message is given by POPM:

MEMORY SURVEY: nnn PAGES FOUND DEFECTIVE

where nnn is the total number of configured pages found to be nonexistent during rollcall.

The control statements $ CHAN, $ UNIT, and $ BASE apply to Startup Resilience.

STARTUP-OPERATOR INTERFACE

The operator interface to Startup allows the site to select desired or desirable approaches to several processing functions. For example, jobs in process can be restarted or reentered in their entirety, statistical collection can be resumed at the point of interruption or can begin anew, or mass storage device content can be modified.

These alternatives are presented to the operator in the form of questions, which are directed to the system console by the Startup program. Each of the questions, the responses that can be made, and the ramifications of each alternative are discussed below.

Note that by the time the question-and-answer sequence has begun, Startup has read the $CONFIG section of the Startup job stream. Therefore, any questions that have been answered via the $ ANSWER statement do not require an operator response unless one is desired. (Refer to "$ ANSWER Statement" in Section 5 for a description of the $ ANSWER statement and a list of the Startup questions that can be answered.)

Startup Console Editor

The Startup console editor is an edit facility that allows the Startup job stream to be edited from the console. The edited job stream becomes the AUTOLOAD file, which may be used for subsequent boots. The console editor is called into execution during Startup as follows:

Enter the word TEXT in response to any question asked by Startup, or

Enter the word TEXT in response to the ??? prompt after pressing the REQUEST push button

The console edit function operates on the Startup BCD card images in memory. The following commands are available within the editor:

(F)ind:/string/	- Search the deck image in a cyclic fashion, starting with the current card image plus one and terminating on the current card image.
(F)orward	- Forward space one card image.
(F)orward;n	- Forward space n card images.
(F)orward;*	- Forward space to end of deck.
(P)rint	- Print the current card image.
(P)rint;n	- Print n card images starting at the current card image.
(P)rint;*	- Print all card images from the current card image to the end of the deck.
(P)rint:/string/	- Print the card image containing the specified string.
(I)nsert	- Insert card images(s) after current card image.
IB	- Insert card images(s) before current card image.
(D)elete	- Delete the current card image.
(D)elete;n	- Delete n card images starting at the current card image.
(D)elete;*	- Delete all card images from the current card image to the end of the deck.
(C)hange	- Change the current card image; delete current card image and insert new card image(s).

(C)hange:/string1/:/string2/ - Change string1 with string2.

(B)ackup	- Backup to the start of the deck.
(B)ackup;n	- Backup n card images.
(R)ead	- Reinstate Startup (including BCD card images) from AUTOLOAD file.
(Q)uit	- Exit from the editor.
PO	- Switch the output stream to a printer.
CO	- Switch the output stream back to the console.
(V)erify	- Verify editor operators. This is a toggle.

X	- Repeat the last command.
X;n	- Repeat the last command n times.
=	- Print the current card image number.
(T)ab x	- Set tab character to x (default is :).
(H)elp	- Print a summary of the editor commands.

Commands are denoted by their first letter with the exception of PO, CO, and IB.

Multiple commands are permitted on a line. A command is delimited by the next command or by a space. Leading spaces are tolerated. Any character is accepted as a string delimiter.

If an error is encountered, the rest of the command line is ignored.

On card input (insert/replace), colon is interpreted as a tab character. Tabs are set to: 8,16,32,73.

On exit from the editor, an AUTO boot is automatically performed.

If the output stream is a printer, the print command prefixes card images with their relative card number.

Binary card images (firmware) are treated in a totally transparent manner.

* Output from the Print command may be terminated by hitting the request key.

MSU3380 And MSU3381 Disk Device Survey Messages

The MSU3380 and MSU3381 disk device survey verifies that the sector size and device type of each MSU3380 and MSU3381 device configured in the Startup job stream agrees with the actual hardware configuration. Unlike the tape survey, the MSU3380 and MSU3381 disk survey is done in the $CONFIG section. The input/output controllers with MSU3380 or MSU3381 capability interrogate each device and pass this information to the Startup program for comparison. Errors are reported in the following mesages:

```
DEVICE TYPE MISMATCH ON i-ccc-dd
CONFIG = type, IPC = type
DEVICE RELEASED
```

where:

type = device type, such as MSC1A, MSC1B, MSD1A, or MSD1B or UNKNWN

IPC = Integrated Programmable Controller

This message appears if the $CONFIG device type doesn't match the IPC device type, for example,

CONFIG = MSC1A, IPC = MSC1B.

In this case, Startup aborts at the end of $CONFIG Section processing. If IPC = UNKNWN, Startup does not abort.

DEVICE FORMAT MISMATCH ON i-ccc-dd
CONFIG = type, IPC = UNFMTD
DEVICE RELEASED

This message appears for unformatted MSU3380 and MSU3381 devices. If the device is removable (RMVBL on the input/output controller statement), it will be marked as a stranger device. If it is not removable, the following message will appear and Startup will abort at the end of $CONFIG processing.

UNFORMATTED DEVICE MUST BE "RMVBL"

DEVICE STATUS MISMATCH ON i-ccc-dd
CONFIG = type, IPC = NOTONL
DEVICE RELEASED

This message appears if the device is not online, i.e., powered off, not ready, or not there.

Startup Questions

Questions that would otherwise be directed to the system console can be answered in two other ways. Questions can be answered by using the $ ANSWER statement in the $CONFIG section of the Startup job stream, as described in "$ ANSWER Statement" in Section 5. In addition, any question that may be issued several times during the bootload procedure can be answered once by the operator, by appending an asterisk (*) to the first answer given to the specific question. Subsequently, issuance of that question is bypassed and the answer given to the first question is applied. For example, if three MPCs are to be bootloaded, the operator can respond YES* to the first *xxxMPC ON iccc BOOTLOAD? question to answer this question for the second and third MPCs. (Refer to "*PROCESS $ READIN? Question" later in this section.)

While the sequence in which questions are issued to the console frequently is the same, some operator responses can cause the sequence to vary from bootload to bootload. Therefore, the sequence in which the Startup questions are described in this section may be different from that in which they are issued to the console. In addition, the following informational message may be issued prior to any of these questions being displayed:

* WARNING - SCU IN MANUAL MODE

The operator must decide whether to continue processing in this mode or to reset the SCU.

The general sequence of Startup questions is as follows:

*REPLACE?
*CHANGE?
*RETAIN CONFIGURATION?
*RESTART?
*SYSOUT RECOVERY?
*SCF CONTINUATION?
*DATE?
*TIME?
*DATE mmddyy TIME hh.mmm
*CHANGE SYSTEM CONTROLLER CLOCK?
*CHANGE DATE?
*CHANGE TIME?
*ENTER TIME OR ENTER A CORRECTION FACTOR?
*CHANGE > 5 MIN, YES TO ACCEPT?
*INITIALIZE?
*EDIT?
*SYSTEM SCHEDULER CLEAR?

One or more additional (secondary) questions may be issued during the initialization (*INITIALIZE?) and editing (*EDIT?) phases of System Startup. These questions are detailed below. (Note that use of the asterisk by the operator to answer subsequent questions is effective with these secondary questions.)

Throughout this section it must be understood that an end-of-message (EOM) response to a question is equivalent to a NO response.

It is recommended that the operator respond NO to a question during the Startup question-and-answer sequence when the correct answer to the question is not apparent. The effects of a NO response are less severe than those of a YES response, which may have undesirable results. For example, a YES response to the *INITIALIZE? question can result in a time-consuming re-editing process involving all configured mass storage devices.

*REPLACE? Question

The *REPLACE? question allows the operator to selectively replace one or more sections of the Startup job stream on the AUTOLOAD file.

A YES response results in the replacement of one or more sections via the card reader. If the replacement section(s) is (are) not in the card reader at the time the YES response is made, the console message:

*BOOTLOAD DEVICE ERROR icccdd: xxxxxx OPERATE WHEN READY

is issued (where i = IOM/IOX/IOP/IMU/IMX number, ccc = channel number, dd = device number, and xxxxxx = error status).

If more than one section is being replaced, the sections must be loaded by their order of appearance in the Startup job stream. For example, if the $CONFIG and $INITIALIZE sections are being replaced, the $INITIALIZE section cannot precede the $CONFIG section. If sections are nonsequential, the message:

```
*EXPECT s...s ccccc FIX TYPE GO
```

is issued to the console (where s...s = name of the section that is out of sequence and ccccc = image of the last statement read). The operator can enter BOOT via the system console to halt the bootload, place the Startup sections in the correct sequence, and initiate a warm boot.

The last two statements in the Startup job stream must be ***EOF statements (regardless of the number of sections being replaced). If the job stream does not conclude with two ***EOF statements, the message BOOTLOAD DEVICE ERROR is issued to the system console.

Note that the message:

```
*NEED READER PORT, Iyy, CHANNEL {PICC }
                                {PICCC}
```

may be issued during this procedure. An operator response is required to inform Startup which card reader is being used to load the new section(s). The response is in the form piccc (where p = IOM/IOX/IOP/IMU/IMX port number, i = IOM/IOX/IOP/IMU/IMX number, and ccc = channel number (0 - 127) or cc = channel number (0 - 63)).

A NO response to the *REPLACE? question indicates that no changes are desired to the Startup job stream that resides on the site-customized boot tape.

A PREPASS response to the *REPLACE? question permits the later use of the Startup Console Editor to make changes in the $CONFIG, $EDIT, $FILES, and $PATCH sections.

After the PREPASS response has been entered, the MASK section of the Startup job stream is read and processed normally. All of the $CONFIG statements are read from tape. After they have all been read from tape, they are processed as card images being read from memory. If an error is encountered, or if Startup aborts, the Startup Console Editor can be called and used to make corrections. See "Startup Console Editor", earlier in this section. Each time the Startup Console Editor is exited, the first section (MASK or $CONFIG) is reprocessed with the card images being made available from memory.

When the $CONFIG section is error free, Startup proceeds to read and process the $INITIALIZE section from tape in the normal manner. The $EDIT, $FILES, and $PATCH sections are read from the tape without being processed. After these sections have been read in, processing resumes at the $EDIT section with the card images being read from memory. If an error is encountered, all of the card images are available in memory and corrections can be made using the Startup Console Editor. After the $PATCH section has been successfully processed, card images are once again read and processed directly from the tape. If a $LOAD section is present on tape, it is read and processed directly from the tape without "prepass" intervention.

The replacement procedure can also be initiated via a REPL response to the *BOOT SOURCE: AUTO OR REPL? question.

*CHANGE? Question

If any $ ANSWER statements are included in the $CONFIG section of the Startup job stream, the *CHANGE? question is issued to allow the operator to override or accept all of the $ ANSWER statements.

A YES response indicates that the operator desires to override $ ANSWER statement content. All $ ANSWER statements are ignored for the duration of this bootload. (Note, however, that this override applies only to the current bootload and that the $ ANSWER statements are reinstated during the next bootload.) The operator then must answer all bootload questions via the system console, regardless of the bootload media (i.e., mass storage, magnetic tape, or punched cards).

A NO response indicates that all $ ANSWER statement content is acceptable. Startup executes the response(s) from the $ ANSWER statement(s).

*RETAIN CONFIGURATION? Question

The *RETAIN CONFIGURATION? question allows the operator to request that the system attempt to retain the status of the hardware configuration at the time the system was last interrupted.

A YES response causes the system to attempt to recover the "assigned/dedicated /released" state of each device.

A NO response causes each device to be returned to its initial state as defined in the Startup job stream. A NO response should be given if any Test and Diagnostic (T&D) jobs were running in the system prior to the last system interrupt. The T&D jobs may have changed the state of various hardware components. These states would be restored on the next boot and could cause problems in the system.

A NO response should also be given whenever a YES response was previously attempted and problems, especially hardware configuration problems, were encountered in the system.

*RESTART? Question

The operator's answer (YES or NO) to the *RESTART? question determines whether or not the system will attempt to restart - near the point of interruption - all jobs/activities that were in progress at the time of the system failure. A response of NO causes the system be cleared of all active jobs.

The response to the *RESTART? question has an impact on the response to the *SYSTEM SCHEDULER CLEAR? question. Startup does not permit a YES response to both questions. If the operator responds YES to *RESTART?, Startup automatically provides a NO response to the *SYSTEM SCHEDULER CLEAR? question. (The effect of YES responses to both questions would be to release job-related system files that are needed for restart of those jobs.)

A YES response to the *RESTART? question allows all jobs in progress at the time of the failure to be restarted from a known condition (i.e., a condition that existed prior to the failure). The following briefly identifies the restart process for jobs that are in the various processing phases at the time of restart:

- Jobs that were in allocation, execution, or termination will be restarted when the operator enters the RUN REST console verb. The SNUMBs of jobs that cannot be restarted are identified in a system console message.

- Jobs that were in the input or scheduling phases, but not yet cataloged on the System Scheduler's SSFILE file, may be automatically restarted if the jobs were saved in the SD.INJ table. Jobs that cannot be restarted are identified in a list of job numbers (SNUMBs) following the console message JOBS THAT SYS SCHED CANNOT RECOVER ARE.

- The system scheduling function will be resumed when the operator enters the JRUN REST console verb. Jobs that were cataloged on the SSFILE file, but not in allocation, execution, or termination, are restarted.

Once restart has been permitted at the system level, individual program restart procedures are invoked for each job at the user level.

A NO response to the *RESTART? question causes all jobs to be restarted from the catalog entry on the SSFILE file when the operator subsequently enters the JRUN REST console verb.

A NO response ensures that the *SYSTEM SCHEDULER CLEAR? question will be issued later during the question-and-answer sequence.

*SYSOUT RECOVERY? Question

The question *SYSOUT RECOVERY? permits system output to be recovered regardless of the answer to the *RESTART? question.

A YES response specifies that all job output and job output specials known to system output are to be recovered. Blinks marked in error are marked as unusable. A NO response specifies that none of the above output is to be recovered and that blinks marked in error are not to be marked as unusable.

NOTE: Blinks for jobs in execution are either kept or released depending on the response to the *RESTART question.

To increase system output reliability, the segments shown in the following list are cleanpointed if the $ INFO CLENPT statement is included in the $CONFIG section of the Startup job stream. The segments are:

DP.JOT - Job Output Table
DP.JOS - Job Output Specials
DP.SBT - SYSOUT Blink Table
DP.BDI - Backdoor Information

DP.JOT and DP.SBT cleanpoints are taken when a new job completes queue entry and blinks and jobs are released. DP.JOT and DP.BDI cleanpoints are taken when a backdoor job queue entry is processed and when backdoor files and jobs are released. DP.JOS cleanpoints are taken each time a special is added to or removed from that segment. Also, DP.SBT is cleanpointed in BRT6 upon activity termination if the job has caused modification to that segment.

*SCF CONTINUATION? Question

The *SCF CONTINUATION? question allows the operator to continue or restart data collection on the Statistical Collection File (SCF).

A YES response results in data collection resuming from the point of interruption.

A NO response results in SCF data collection beginning anew. If the SCF resides on mass storage, this response results in the clearing of all previously written information. This information cannot be recovered. If the SCF resides on magnetic tape, restart does not affect information written to the SCF. The tape containing the SCF must be rewound and a new tape mounted before collection is resumed. Note that rewinding of the tape is automatically effected if a site-option patch is applied to the .MPOPM module (Memory Allocation program). (Refer to "Memory Allocator Options" in Section 5 for the format of this site-option patch to .MPOPM.)

If a system failure prevents the Master Mode Dump (.MDUMP) program from closing the SCF, system restart will not proceed. The .MPOPM module aborts and the existing SCF is not closed. The system must be rebooted and the operator must respond NO to the *SCF CONTINUATION? message. Data collection then resumes on a new SCF.

***SYSTEM CONTROLLER CLOCK AND .CRDAT MISMATCH Message**

This message is issued when Startup detects a difference in the date calculated from the system controller clock and the date recovered from .CRDAT on an autoboot with CHANGE? NO. Startup also issues this message if a difference of one minute or more is determined when the time calculated from the system controller clock is compared to time calculated from .CRDAT+1 and .CRDAT+2. The *DATE question is output and the response values input are used to set the system controller clock and .CRDAT.

***DATE mmddyy TIME hh.mmm Information Message**

This message informs the operator of the date/time, and is issued when the "*CHANGE?" question is answered YES.

***CHANGE SYSTEM CONTROLLER CLOCK? Question**

The *CHANGE? question was answered YES. A YES response causes the *CHANGE DATE? and *CHANGE TIME? questions to be issued.

***CHANGE DATE? Question**

The *CHANGE? question was answered YES. A YES response causes the *DATE? question to be issued.

***CHANGE TIME? Question**

The *CHANGE? question was answered YES. A YES response causes the *ENTER TIME OR ENTER A CORRECTION FACTOR? question to be issued.

***ENTER TIME OR ENTER A CORRECTION FACTOR? Question**

If the *CHANGE TIME? question was answered YES, the operator can issue one of two responses:

1. Enter the time in one of the formats of the *TIME? question (e.g., 01:20).

2. Enter a time correction factor using the format of the *TIME? question, except that a plus (+) or minus (-) sign must precede the correction factor. This allows the operator to correct or change the time of day in increments (e.g., -01:00 would be entered to change a system time to standard time from daylight savings time).

*DATE? Question

Startup detected one of the following conditions:

1. The clock contains a value earlier than the Startup assembly date.

2. The clock contains a value greater than 100 years.

3. A YES response was given to the *CHANGE DATE? question.

Enter the current date (i.e., the date of this bootload) via the system console in the mmddyy format (where mm = month, dd = day, and yy = year). Startup validates the date entered (e.g., non-numeric date, invalid month or day). If the date entered is invalid, the *DATE? question is reissued.

*TIME? Question

Startup detected one of the following conditions:

1. The clock contains a value earlier than the Startup assembly date.

2. The clock contains a value greater than 100 years.

Enter the current time (i.e., the time of this bootload) via the system console. Startup validates the entry (e.g., the time cannot exceed 2400 hours or contain non-numeric characters). If the time is invalid, the *TIME? question is reissued.

The time can be entered in one of three formats:

- o hh:mm - where hh = 00-23 hours and mm = 00-59 minutes

- o hht or hh.t - where hh = 00-23 hours and t = tenths of one hour (i.e., 0-9)

*CHANGE > 5 MIN, YES TO ACCEPT? Question

The operator attempted to change the clock time by more than 5 minutes, thus, this message gives the operator the opportunity to reenter the time.

A YES response causes the system to accept the new time. A NO response causes the *ENTER TIME OR ENTER A CORRECTION FACTOR? to be reissued.

*INITIALIZE? Question

The *INITIALIZE? question allows mass storage devices to be reformatted or "initialized". The initialization process provides a clean basis for re-establishing tables, pointers, and files on the mass storage device. It is recommended that initialization be performed on each cold boot. (Refer to "Cold Boot" earlier in this section for additional information.)

Initialization destroys existing tables and pointers on the device prior to rebuilding them. The rebuilding process draws upon information in main memory and included in the $INITIALIZE section of the Startup job stream.

Among the tables rebuilt are those that compose the device header and define the physical attributes of the device in terms of usable space (i.e., that space which has been allocated or which is available for allocation) and defective space. (Refer to the GCOS 8 OS System Operating Techniques manual for device header information.)

A console message may be issued, warning of allocation table problems following the initialization process. The message:

```
ST1 AND SMCDUP INITED BUT NOT ALL OTHERS, FILE SPACE WILL BE LOST
```

is issued if device ST1, containing the System Master Catalog (SMC), and the device containing the duplicate SMC are initialized, but one or more other permanent mass storage devices are not initialized. (This condition may arise as the result of omitting $ INIT statements in the $INITIALIZE section for all permanent mass storage devices, or as the result of selective initialization via a PARTIAL response to the *INITIALIZE? question.)

While allocation tables on the devices that were not initialized continue to identify allocated space on those devices, the FMS catalog structure no longer reflects the assigned space. Therefore, the space is lost to the system (i.e., it is unassignable). To correct this condition, all permanent mass storage devices must be initialized.

While data files are not cleared from the initialized mass storage device, destruction (clearing) of the pointers and tables during the initialization process makes it impossible to subsequently access these files. Following device initialization, any write operation to that device overwrites the existing data. Therefore, any data files that are to be retained must be saved on magnetic tape prior to device initialization and then restored to the device following initialization. (Refer to the GCOS 8 OS File Management Supervisor manual for save and restore information.)

In addition to the reformatting process, Startup performs several other functions during initialization including:

1. Volume sets, and their characteristics, can be defined. Volume sets are multi-volume structured removable devices. Refer to "$ VSET Statement" and "$ VOPT Statement" in Section 5.

2. Mass storage, magnetic tape, and unit record MPCs are checked to determine if they have been bootloaded with firmware. If they have not, the operator is given the option of bootloading the controllers. MPC verification and bootloading is bypassed during a fast boot. (Refer to "MPC Bootloading" for additional information.)

3. Printer character set and VFC images are loaded from the Startup program's internal copy (i.e., buffer) of the PRINTIMAGE file for all qualifying, configured printers. (Note that if a print image is added to the PRINTIMAGE file on mass storage via the TVIM program, the internal Startup copy does not contain that image.) As with MPC bootloading, loading of character set and VFC images is bypassed during a fast boot. (Refer to "Character Set And VFC Image Loading" in this section for additional information.)

4. Permanent mass storage devices are verified by comparing each mass storage device label with information in the in-memory device name table. The device name table is created from $CONFIG section information and from the device directory, which occupies several contiguous llinks in the SMC on device ST1. (Refer to "Permanent Mass Storage Device Verification" later in this section for additional information.)

5. All controller-driven (such as an MPC) magnetic tape handlers are surveyed. The density and track-type attributes of each configured device are compared with information specified in the $CONFIG section. If the comparison fails, the device is unconditionally released (i.e., without querying the operator). (Refer to "Magnetic Tape Handler Survey" in this section for additional information.)

OPERATOR CONTROLS DURING INITIALIZATION

The initialization function is controlled to a great extent by the operator, who can enter one of several responses to the *INITIALIZE? question. Therefore, the operator must be aware of the consequences of each response. The following describes the ramifications of the NO, YES, TOTAL, and PARTIAL responses to the *INITIALIZE? question.

Response	Description of Initialization Process
NO	A response of NO indicates that initialization is not desired. All mass storage device information is to remain intact and mass storage information previously defined in the $INITIALIZE section of Startup still is valid. However, Startup performs all five other functions that accompany initialization (i.e., volume set definition, MPC verification and bootloading, character set and VFC image loading, mass storage device verification, and magnetic tape handler surveying).
YES or TOTAL	A response of YES or TOTAL initializes all configured mass storage devices. Information included on the $ INIT statements in the $INITIALIZE section of Startup is executed to recreate tables and pointers on mass storage devices. A response of YES or TOTAL automatically provides a YES answer to all MPC-related questions. (Refer to "MPC Bootloading" in this section for additional information.)

Response	Description of Initialization Process
	A response of TOTAL: (1) provides a YES answer to the *EDIT? question; (2) provides a NO answer to the *FORMAT/LABEL QUESTIONS? question; and (3) results in execution of $EDIT section content. Startup assumes that all mass storage devices were previously formatted. Under these circumstances, a $ INIT statement must be included in the $INITIALIZE section of Startup for each mass storage device. It is recommended that the bootload medium be magnetic tape when a YES or TOTAL response is made. This ensures that the AUTOLOAD file will contain current information.
PARTIAL	A response of PARTIAL allows the operator to selectively initialize mass storage devices on a device-by-device basis (as opposed to a YES or TOTAL response, which initializes all devices). This response also provides an audit trail of the initialization process, which can prove valuable when debugging initialization problems. If the operator responds PARTIAL, one of the following additional (secondary) questions is issued to the console. *INIT DEVICE ddd? *INIT RMVBL DEVICE ddd? *INIT SHARED DEVICE ddd? Each of these messages queries the operator as to whether mass storage device ddd (where ddd = logical device name) is to be initialized. In the message: *INIT RMVBL DEVICE ddd? device ddd is configured as a removable (RMVBL) device. In the message: *INIT SHARED DEVICE ddd? device ddd is configured as a shared device; i.e., shared with NPS. See "$ Shared Statement" in Section 5. A response of NO to these questions indicates that device ddd is not to be initialized (except for any automatic secondary initialization as described below). A response of YES indicates that device ddd is to be initialized. In addition, subsequent questions are issued regarding mass storage device formatting. (Refer to "*FORMAT/LABEL QUESTIONS? Question" in this section for additional information.)

Response	Description of Initialization Process

The system allows the operator to respond SECONDARY to one of the secondary questions following a PARTIAL response when circumstances may require an operator-initiated secondary initialization (e.g., the device encounters a temporary descriptor table (TDT) overflow condition). Otherwise, under normal operating conditions, secondary initialization is automatically performed as follows:

1. During System Startup, when the NIAST (No In-Memory Available Space Table) option is in effect, a secondary initialization is performed automatically for each permanent device whose last operating mode was not NIAST. This action establishes a current Available Space Table (AST) on the device. This automatic secondary initialization occurs for both shared and unshared permanent devices. It is not dependent on the response to the INITIALIZE? question or its subsidiary *INIT ... ddd? questions. (The AST resided in memory during non-NIAST operation; therefore, either no AST exists on the device, or it is obsolete.)

2. For removable mass storage devices in the NIAST mode, a secondary initialization is performed automatically whenever a structured disk pack is mounted at Startup. If it is mounted later, the AST rebuilding occurs during system operation. This action is not dependent on the response to the INITIALIZE? question or its subsidiary *INIT ... ddd? questions.

3. When processing D, O, A, L, and Y data (YES response to PROCESS DIRECTORY CARD FOR DEVICE ddd? message) on devices whose last operating mode was NIAST, at least a secondary initialization is required.

A SECONDARY response to one of the secondary questions following a PARTIAL response is logically ignored when NIAST mode is not in effect.

In contrast to an initialization, a secondary initialization releases no permanent file space and uses the device allocation unit table on the device to rebuild a current AST on the device. For permanent devices, the empty TDT is also rebuilt. This action synchronizes both tables containing mass storage space allocation information (i.e., the device allocation unit table and the AST) and, for permanent devices, returns all temporary file space to the system. Note that temporary files do not exist on removable devices.

NOTE: Recovery may not be performed on protected files if a secondary initialization is performed.

MPC BOOTLOADING (DPS 8 AND DPS 8000 ONLY)

MPC firmware must be functional to perform system startup. Separate firmware data decks are used to bootload the magnetic tape and mass storage controllers. These deck images can reside on mass storage, magnetic tape, or punched cards. In addition, the $INITIALIZE section must contain the MPC bootload (MPCB) program deck image, which permits loading of the data decks. (Refer to "$INITIALIZE Section" in Section 1 for additional information.)

Several console questions may be issued by the MPCB program to ensure that all MPC firmware is loaded and operational:

- Startup automatically provides a YES answer to all MPC-related questions if the operator responds YES or TOTAL to the *INITIALIZE? question. As a result, all MPCs are bootloaded from firmware on the deckfile.

  If firmware for a specific device type is absent from the deckfile, Startup bypasses bootloading of the MPC in question and completes the bootloading process for all other MPCs. All devices associated with the bypassed MPC are released during the device rollcall function. Messages are issued during the rollcall function to identify released devices. No message is issued when MPC bootloading is bypassed.

  To subsequently bootload the MPC while the system is operational, the operator can call the MOLTS subsystem of the Total Online Test System (TOLTS) via the console entry:

  ```
  TEST MPC icc
  ```

  (where i = IOM number and cc = channel number). The console entry ASGN iccdd then can be used to reassign each released device to the system (where i = IOM number, cc = channel number, and dd = device number).

  Use of the PARTIAL response to the *INITIALIZE? question during a cold boot can prevent the bypass of MPC bootloading. The PARTIAL response ensures that all bootloading occurs and that all configured devices are activated.

- Startup automatically provides a NO answer to all MPC-related questions if the operator responds NO to the *INITIALIZE? question and if Startup determines that all MPCs are loaded with firmware.

  The following question is issued to the console if the operator responds NO to the *INITIALIZE? question and if Startup determines that one or more MPCs are not loaded with firmware or are not operational.

  ```
  *xxxMPC ON icc NOT RESPONDING PROPERLY
  TYPE (B)OOT, (S)KIP, OR (R)ETRY
  ```

The following console entries can be made to correct or to circumvent the condition:

B - Load firmware into MPC xxx from the deckfile. This is the recommended response.

S - Bypass (skip) bootloading MPC xxx. All devices associated with this MPC are released during the device rollcall function. To subsequently bootload the MPC and reassign released devices, enter the TEST MPC icc and ASGN iccdd console entries as previously described.

R - Retry the MPC firmware verification function. If Startup again determines that the firmware is not loaded or that the MPC is not operational, the preceding message is reissued.

o If the operator responds PARTIAL to the *INITIALIZE? question, one or more of the following questions is issued to the console:

*MPC ON icc PRE-INITIALIZE?

*xxxMPC ON icc BOOTLOAD SYS ID NAME "y...y" REV.zz?

*xxxMPC ON icc APPEARS OK, BOOTLOAD?

The PRE-INITIALIZE? question is issued only if a DAU Disk Controller is recognized, and asks if the operator wishes to run concurrent self tests. Answering YES saves time (20 seconds per device), and also requires a YES answer to the BOOTLOAD? question because self tests destroy firmware. Answering NO to the PRE-INITIALIZE? question will add time to the bootloading of firmware if YES is entered in response to the BOOTLOAD? query later on.

The *xxxMPC on icc BOOTLOAD SYS ID NAME "y...y" REV.zz? question is issued for each configured MPC. A YES response results in bootloading of MPC xxx with firmware revision level zz from the deckfile. A NO response bypasses bootloading of MPC xxx. To subsequently bootload the MPC and reassign released devices, enter the TEST MPC icc and ASGN iccdd console entries as previously described.

Only the *xxxMPC ON icc APPEARS OK, BOOTLOAD? question is issued if firmware does not exist in nonshared MPC xxx. A YES response results in bootloading of MPC xxx from the deckfile. A NO response bypasses bootloading of MPC xxx. If the MPC is not bootloaded, all devices associated with the MPC are released during the device rollcall function. To subsequently bootload the MPC and reassign released devices, enter the TEST MPC icc and ASGN iccdd console entries as previously described.

PERMANENT MASS STORAGE DEVICE VERIFICATION

All permanent mass storage devices - especially device ST1 and the device that is to contain the (optional) duplicate System Master Catalog (SMC) - are verified to ensure that logical device labels are accurate. Any inconsistency between logical device label information and system control data in the device directory in the System Master Catalog causes a console message to be issued. The message will be:

```
*EXPECTED PACK NAME xxx FOUND yyy YES TO ACCEPT CHANGE
```

or:

```
*ddd EXPECTED PACK NUMBER xxx FOUND yyy YES TO ACCEPT CHANGE
```

where ddd = logical device name. (Refer to "Verification Of All Permanent Mass Storage Devices" in this section for a description of these messages and operator responses.)

Tables used during the verification process include the in-memory device name table, created from $CONFIG section data, and the device directory, which occupies several contiguous llinks in the SMC on device ST1.

Normally, this verification procedure is performed with no errors encountered and Startup continues on to the magnetic tape handler survey procedure.

ST1 And SMCDUP Device Name Verification

Startup determines whether the correct disk packs are mounted on device ST1 and on the device containing the duplicate SMC. If the comparison of logical device label and device name table information fails, the console message:

```
*ddd ST1/SMCDUP DEVICE WRONG PACK MOUNTED
```

is issued (where ddd = logical device name).

To correct the condition, power off the device and visually verify the pack serial number, which is stamped on the pack by the manufacturer. If the serial number does not identify the correct disk pack, mount the proper pack on device ddd. It is recommended that the system be restarted via a warm boot to ensure that any erroneous in-memory tables are corrected and rebuilt.

SMCDUP Device And Option Verification

Startup determines whether the device for which the SMCDUP option was specified actually contains the duplicate SMC. If the device does not contain the duplicate SMC, Startup aborts. The console message:

```
*ddd CAN'T CHANGE SMCDUP WITHOUT INIT
```

is issued (where ddd = logical device name). Remove the $ INFO...SMCDUP statement, which identifies device ddd, from the $CONFIG section of the Startup job stream and perform a warm boot to restart the system. (As a result of removing the $ INFO...SMCDUP statement, a duplicate SMC is not created. To provide a duplicate SMC, the device on which the duplicate is to reside must be defined on the $ INFO...SMCDUP statement and the operator must respond YES to the *INITIALIZE? and *EDIT? questions during the bootload.)

Verification Of All Permanent Mass Storage Devices

Following the preceding verification process, which specifically applies to device ST1 and to the device that contains the duplicate SMC, Startup performs a series of tests that applies to all permanent mass storage devices. As in the previous verification functions, logical device label information is compared with data contained in the device name table and in the device directory.

The following verification sequence is performed for each device:

1. If a Startup statement (e.g., $ FILDEF or $ GCOSFIL) includes a logical device name (ddd) not defined on a $ Iyy statement, the console message:

   *DEVICE ddd UNDEFINED

   is issued and Startup aborts. While this message also applies to magnetic tape and unit record devices, it is more often associated with mass storage devices.

2. If Startup determines that a device is offline and cannot be accessed, the following console message is issued:

   *DISK PACK ERROR icccdd s...s EOM TO CONTINUE -- 'RLSE' TO RELEASE THE DEVICE

   where:

   i - IOM/IOX/IOP/IMU/IMX number
   ccc - channel number
   dd - device number
   s...s - status message).

   The operator can (1) release the device via the console entry RLSE or (2) correct the error defined in the status message. The EOM push button on the system console is then pressed to retry the verification function. If the released device contains any system files, the system aborts.

   NOTE: If a $ INFO...RLSDSK statement is present in the $CONFIG section of the Startup job stream, the RLSE answer is automatically applied to the preceding message. The device is automatically released during a fast boot and no message is issued.

3. Startup compares the device name (i.e., pack name) contained in the logical device label with the device name contained on the $ Iyy statement for the device. If the comparison fails, the console message:

   *EXPECTED PACK NAME xxx FOUND yyy YES TO ACCEPT CHANGE

   is issued (where xxx = device name contained on the $ Iyy statement and yyy = device name contained in the logical device label).

   The operator has the following options:

   a. Power off the device and visually inspect the pack name, which normally is written by site operations personnel on an external pack label. If the wrong pack is mounted, mount the correct pack and perform a warm boot to restart the system.

   b. Respond YES to the message. As a result, Startup writes the pack name, contained in the logical device label (yyy), into the device name table and continues with the verification process.

   c. Respond NO to the message. Startup aborts. Rebuild the logical device label via the procedure specified under "2."

4. Startup compares the pack serial number contained in the logical device label with the pack serial number contained in the device directory. If the comparison fails, the console message:

   *ddd EXPECTED PACK NUMBER xxxxx FOUND yyyyy YES TO ACCEPT CHANGE

   is issued (where ddd = logical device name, xxxxx = pack serial number in the device directory, and yyyyy = pack serial number in the logical device label). The operator has the following options:

   a. Power off the device and visually inspect the pack serial number stamped on the pack by the manufacturer. If the pack serial number is not the one identified in the console message, mount the correct pack and perform a warm boot to restart the system.

   b. Respond YES to the message. As a result, Startup writes the pack serial number, contained in the logical device label (yyyyy), into the device directory and continues with the verification process.

   c. Respond NO to the message. Startup aborts. Rebuild the pack label via the procedure specified under "2."

5. If Startup encounters a device whose status has been changed from removable (RMVBL) to permanent (PERM), the console message:

   *ddd PRIOR DEVICE STATUS WAS RMVBL CHANGED TO PERM YES TO CONTINUE

   is issued to verify that the status change is correct (where ddd = logical device name). The following two console entries are valid:

YES - Startup changes the device status from removable to permanent on the device directory and continues the verification procedure. (Once the device status has been changed from removable to permanent, the only means of returning the status to removable is to initialize and edit the device during a bootload procedure.)

NO - Startup aborts. Modify the $ Iyy statement to reflect the correct status for device ddd. Perform a warm boot to restart the system.

6. Startup performs four tests, comparing the logical device label information to data in the device directory, to determine if:

   a. A $ SHARED statement indicates that the device is shared with NPS.

   b. A $ INIT...CAT statement enables FMS creation of catalogs on the device.

   c. A $ Iyy...NOFMS statement disables FMS creation of catalogs or files on the device.

   d. The device status was changed from permanent to removable via the $ Iyy statement.

   If any of the preceding tests fail, the following console message is issued and Startup aborts:

   *ddd CAN'T CHANGE DEVICE STATUS FROM xxx TO NONxxx

   where:

   ddd - logical device name
   xxx - NPS, FMS, CATALOGABLE, or PERM/RMVBL

   Correct the error condition by (1) inspecting the $ SHARED, $ INIT, and $ Iyy statements for accuracy, or (2) initializing device ddd or the entire mass storage subsystem during a cold boot.

7. Startup aborts if it does not encounter a $ INFO...SMCDUP statement, but does encounter a device whose label indicates that the device was initialized as the device on which the duplicate SMC is to reside. The following console message is issued:

   *ddd CAN'T CHANGE SMCDUP WITHOUT INIT

   where ddd = logical device name. Insert a $ INFO...SMCDUP statement, which identifies the correct device, into the $CONFIG section of the Startup job stream and perform a warm boot to restart the system.

MAGNETIC TAPE HANDLER SURVEY

The magnetic tape handler survey is designed to verify the density (bpi) and track type (seven or nine) attributes of each configured magnetic tape handler. Each magnetic-tape-oriented controller, such as an MPC or IPC, contains an internal table that reflects this information for each tape handler associated with the controller. Startup compares the controller table information with data specified in the $CONFIG section of the Startup job stream.

If controller table and $CONFIG section unit number information do not match, the tape handler is released by Startup. The console message:

    *UNIT NUMBER MISMATCH ON icccdd: CONFIG = nn, MPC = nn DEVICE RELEASED

is issued (where i = Iyy number, ccc = channel number, dd = device number, nn = conflicting unit numbers).

If controller table and $CONFIG section density information do not match, the tape unit will remain assigned and the following console message is issued:

    *DENSITY MISMATCH ON icccdd: CONFIG = <cdens>, MPC = <mdens>, ASSIGNED =
    <adens>

(where i = same as above, ccc = same as above, dd = same as above, cdens = configured density, mdens = MPC density, adens = assigned density). No operator action is required. The operator subsequently can reassign the released device by correcting the $CONFIG section data and performing a warm boot.

If an error is encountered during the magnetic tape handler survey while bootloading from magnetic tape, the console message:

    *TAPE ERROR, icccdd s...s MAY BE RECOVERABLE FIX, TYPE GO OR STOP OR TYPE
    CONT FOR CONTINUE

is issued (where s...s = error status condition). The operator can:

1. Correct the error and enter GO via the system console to resume the bootload;

2. Enter STOP via the system console to abort Startup; or

3. Enter CONT to release all magnetic tape handlers configured on this MPC subsystem and to resume the bootload. (To subsequently bootload the MPC and reassign released devices, enter TEST MPC iccc and ASGN icccdd as described under "MPC Bootloading" in this section.)

Processing is identical on a fast boot, but no message is issued to the console.

CHARACTER SET AND VFC IMAGE LOADING

Information from the $CONFIG section of the Startup job stream is compared with information contained in Startup's internal copy (i.e., buffer) of the PRINTIMAGE file during loading of the character set and Vertical Format Control (VFC) image. If $CONFIG section and PRINTIMAGE file information do not match, a console message is issued defining the problem.

If any of the following error conditions is encountered during a fast boot, the device is released and no console message is issued.

1. If Startup attempts to load a character set or VFC image into a printer that is not in a Ready status, the console message:

   *PRINTER ERROR icccdd s...s FIX TYPE GO OR STOP OR RLSE

   is issued (where i = Iyy number, ccc = channel number, dd = device number, and s...s = status message). The operator has three options:

   a. Place the device in a Ready status and enter GO via the system console. Startup attempts to load the character set or VFC image.

   b. Enter STOP via the system console, causing Startup to abort. Deconfigure the device, or bootload the MPC. Then perform a warm boot to restart the system.

   c. Enter RLSE via the system console to release the device. (If a $ INFO RLSPNT statement is included in the $CONFIG section of the Startup job stream, the RLSE answer is automatically applied to the preceding message.)

2. If Startup encounters an error condition (as defined in the status portion of the following message), the device is released. The console message:

   *PRINTER ERROR icccdd s...s DEVICE RELEASED

   is issued. The operator subsequently can correct the error condition while the system is inoperable and perform a warm boot, or correct the condition while the system is operational and reassign the device to the system via the console entry ASGN icccdd.

3. If Startup fails to find the character set defined in the $CONFIG section in its internal table, the device is released and the console message:

   *TRAIN ID UNKNOWN TO STARTUP DEVICE RELEASED

   is issued. The character set subsequently can be loaded onto the PRINTIMAGE file via the TVIM program while the system is operational. The device can be reassigned to the system via the console entry ASGN icccdd.

4. If Startup determines that printer paper is missing or requires realignment, the console message:

   *POSITION PAPER IN PRINTER icccdd FIX TYPE GO OR STOP OR RLSE

   is issued. After loading or realigning the paper, the operator can enter GO, STOP, or RLSE via the system console to get the same results as described under "1".

5. If Startup determines that a character set exists but is not defined in the $CONFIG section, the device is released and the console message:

   *MOUNTED TRAIN ID xxx REQUESTED TRAIN ID yyy DEVICE RELEASED

   is issued. The operator can:

   a. Correct the $CONFIG section and perform a warm boot to restart the system;

   b. Mount the correct print train or belt (yyy) on the printer and perform a warm boot to restart the system;

   c. Reassign the device via the ASGN icccdd console entry when the system is operational.

SUPPLEMENTAL QUESTIONS DURING INITIALIZATION

A series of supplemental (secondary) questions is issued regarding formatting of mass storage devices, clearing of defective space information from logical device labels, and re-editing the deckfile. The following describes the ramifications of responses to these questions.

Discrepancies During Mass Storage Device Verification

Two console messages are issued during the logical device initialization process if Startup determines that logical device label information differs from data contained on a $ Iyy statement. One message is issued if logical device names (i.e., pack names) differ, while the other message is issued if the pack serial numbers do not agree.

The console message:

   *icccdd xxx CHANGING TO yyy YES TO INITIALIZE PACK

is issued if Startup determines that the logical device name contained on the logical device label (xxx) is different from the logical device name on the $ Iyy statement (yyy). The operator can respond YES or NO to this message.

A YES response indicates that the correct disk pack is mounted, but the label contains erroneous information. Startup writes the logical device name on the $ Iyy statement to the logical device label and initialization continues.

A response of NO indicates that the wrong disk pack is mounted. As a result, Startup aborts. The operator can mount the correct disk pack and reboot the system.

If the logical device label was destroyed before Startup compared the logical device label with the $ Iyy statement, no console message is issued (even if the wrong disk pack is mounted).

The console message:

    *icccdd: ddd EXPECTED PACK NUMBER xxxxx FOUND yyyyy YES TO ACCEPT CHANGE

is issued if Startup determines that the pack serial number on the logical device label (yyyyy) is different from that on the $ Iyy statement (xxxxx). Note that this comparison is made only if the S-xxxxx option is included on the $ Iyy statement (where xxxxx = pack serial number).

The operator can respond YES or NO to this message.

A YES response indicates that Startup is to continue the device initialization process. The operator subsequently can change the pack serial number that is contained in the logical device label to match that on the $ Iyy statement. This modification can be effected during the pack serial number verification procedure that is performed if the operator responds YES to the *FORMAT/LABEL QUESTIONS? question.

A response of NO indicates that the wrong disk pack is mounted. As a result, Startup aborts. The operator can mount the correct disk pack and reboot the system.

FORMAT/LABEL QUESTIONS? Question

A YES response to the question *INIT DEVICE ddd causes Startup to initialize a device, and occurs when a $ DIRECT statement is encountered for device "ddd" or at the end-of-file for the $INITIALIZE section, whichever comes first. Startup begins initialization by reading the following disk sectors:

- Sector 0 - Label
- Sector 1 - Defective llink directory
- Sector 3 - Alternate track table (except MSU0500, MSU3380, and MSU3381)
- Sector 4 - Defective space history table (except MSU0500, MSU0501, MSU3380 and MSU3381)

MSU0500, MSU0501, MSU3380 and MSU3381 disks have neither a defective space history table nor an alternate track table; therefore, Startup does not attempt to read these sectors when processing such devices.

If no $ DIRECT ddd/CLEAR statement is processed and if one or more of the reads result in nonready status, incorrect checksum, or invalid table format, Startup issues the following console message:

    *ddd LLINK 0 BAD, MUST TEST/FORMAT/RTHDRS, YES TO CONTINUE

A NO response causes Startup to abort. A YES response causes Startup to continue the initialization process. At this point, two sets of snaps will have been written to the printer: one before and one after the label sector has been built. Snaps are taken of disk sectors 0-4, associated flags and counters, and the disk I/O trace buffer.

If a $ DIRECT ddd/CLEAR statement is present and was processed by responding YES to the following console message:

    PROCESS DIRECTORY CARD FOR DEVICE ddd?

Startup issues the following console message:

    *ddd LLINK 0 CLEARED, MUST TEST/FORMAT/RTHDRS, YES TO CONTINUE

A NO response causes Startup to abort. A YES response causes Startup to continue the initialization process. Again, if any of the sectors read were bad, printer snaps will have been written.

Startup then zeros the bad sectors (or clears sectors 1, 3, and 4 if using $ DIRECT ddd/CLEAR) and, if the defective space history table was cleared, builds an empty history table.

Startup now issues the following message:

    *FORMAT/LABEL QUESTIONS?

If the pack number is 00000, the following message is given regardless of the response to the message above:

    *icccdd: ENTER PACK #NNNNN?

The operator responds by entering the desired pack number.

If the response to the *FORMAT/LABEL QUESTIONS? message was YES, the following messages are issued:

    *icccdd: CHG PACK #nnnnn?

    *icccdd: FMT PACK #nnnnn? (except MSU0500, MSU0501, MSU3380 and MSU3381)

    *icccdd: TEST PACK #nnnnn? (MSU0500 and MSU0501 only)

The operator responds to the *icccdd: CHG PACK #nnnnn? message as follows:

NO - pack number is not changed

nnnnn - new pack number

The operator responds to the *icccdd: FMT PACK #nnnnn? message as follows:

NO - No formatting will be done

YES - Startup attempts to format (write track headers on) all tracks, except those in the defective space history table, as GOOD. Those tracks in the history table as well as those which cannot be formatted GOOD are formatted DEFECTIVE/NO ALTERNATE and entered into the defective llink directory.

RTHDRS - Startup issues a "read track header" command to all tracks on the disk. Tracks with track indicator (TI) bits specifying DEFECTIVE/NO ALTERNATE and tracks that cannot be successfully read are entered into the defective llink directory. Tracks with TI bits specifying DEFECTIVE/ALTERNATE ASSIGNED are entered into the alternate track table.

The operator responds to the *icccdd: TEST PACK #nnnnn? message as follows:

NO - No testing will be done

YES or RTHDRS - Startup issues a "read track header" command to all tracks on the disk. Tracks with track indicator (TI) bits specifying DEFECTIVE/NO ALTERNATE and tracks that cannot be successfully read are entered into the defective llink directory. Tracks with TI bits specifying DEFECTIVE/ALTERNATE ASSIGNED are entered into the alternate track table. MSU0500/0501 disks do not have alternate track tables; therefore, tracks marked DEFECTIVE/ALTERNATE ASSIGNED are treated as if they were marked GOOD.

NOTE: The FMT and TEST options do not apply to the MSU3380 and MSU3381. The FTAR program does the formatting of the devices on-line. In addition, if any defective tracks are encountered they must be assigned to an alternate track or be deallocated by using the $ DIRECT statement in the $INITIALIZE secton of the Startup job stream.

Note that Startup does not assign alternate tracks. If alternate track assignment is desired, the Removable Storage Initialization Program (RSIP) must be used. Refer to the <u>GCOS 8 OS System Operating Techniques</u> manual for RSIP information.

At the end of the format/test process, Startup equates the defective llink directory with the defective space history table (if one exists).

One result of this is that each entry in the defective llink directory defines a single track's worth of defective space. Startup continues with the initialization of the disk. Sectors 0-4 of the disk are written and the device allocation unit table is created. Defective space in the defective llink directory is marked unavailable in the device allocation unit table. Initialization is now complete.

If a $ DIRECT ddd JCL statement containing D, O, A, L, and Y data is processed (YES response to PROCESS DIRECTORY CARD FOR DEVICE ddd? message), Startup adds the D, O, A, L, and Y space to the defective llink directory, makes the defective llink directory agree with the history table, rewrites sectors 0-4, and marks defective space as unavailable in the device allocation unit table. Note that the $ DIRECT ddd is allowed even if the disk ddd is not being initialized. However, a secondary initialize is necessary when the disk ddd last mode of operation was NIAST.

Printer snaps of disk sectors 0-4 and associated flags and counters can be forced by inserting a $ BUGON statement in the $INITIALIZE section of the Startup job stream. A $ BUGOFF statement disables the forced printer snaps. Printer snaps of the Startup disk I/O interval trace table can be forced by inserting a $ DBGON statement in the $INITIALIZE section of the Startup job stream. A $ DBGOFF statement disables the forced printer snaps.

The following console messages may be issued prior to an abort of Startup during disk initialization:

*ddd LLINK ZERO BAD, MUST INIT

Startup detected invalid disk sectors 0, 1, and (or) 4 and one of the following conditions:

1. A NO response was made to the INIT DEVICE ddd? question;

2. No $ INIT ddd JCL statement was present;

3. $ DIRECT ddd D, O, A, L, and Y input was being processed.

The pack must be initialized.

*DISK PACK ERROR icccdd (status) TRACK ZERO BAD

Startup was unable to format track 0 GOOD. This pack cannot be used by GCOS unless this condition is due to a site error.

*DISK PACK ERROR icccdd PSEUDO ALT AREA ALL DEFECTIVE

The area to be used for pseudo alternate tracks (MSU3380 and MSU3381) is defective. The pack must be reformatted.

*ddd BLOCK 0 DEFECTIVE MUST REFORMAT DEVICE

Track zero on the disk couldn't be read successfully (channel ready status), or TI bits did not show a GOOD track. The disk must have track zero reformatted as GOOD and the pack must be reinitialized.

*ddd ERROR DETERMINING ALTERNATE TRACK MUST REFORMAT DEVICE

The track was found marked DEFECTIVE/ALTERNATE ASSIGNED and the alternate track was not in the alternate track table or it was already assigned to another defective track. The pack must be reformatted.

Some general guidelines for initializing/formatting/testing of mass storage devices are:

1. Use a $ DIRECT ddd/CLEAR statement whenever the status of the defective llink directory, alternate track table, and (or) the defective space history table sectors is unknown or questionable. Note that whenever a $ DIRECT ddd/CLEAR statement is processed, a YES response should be given to the *FORMAT/LABEL QUESTIONS? message and formatting or testing should be invoked to rebuild the alternate track table.

   This recommendation does not apply to MSU0500/0501 disks as they have no alternate track table. If formatting or testing is not done after a $ DIRECT ddd/CLEAR statement is processed, a $ DIRECT ddd (D, O, A, L, and Y) statement must be processed to withdraw all known defective space.

2. The only reason for reading track headers is to rebuild the alternate track table and to build a defective llink directory containing all tracks marked DEFECTIVE/NO ALTERNATE. Since reading track headers takes a relatively small amount of time (less than five minutes for MSU0451 and MSU0500 and about six minutes for an MSU0501), it is recommended that this be done each time a mass storage device is initialized. Reading of track headers is not necessary if the correct D, O, A, L, and Y statements for all tracks formatted DEFECTIVE/NO ALTERNATE are processed.

3. The recommended procedure for initializing mass storage devices is to answer "PARTIAL" to the "*INITIALIZE?" question and never use the "*" when answering any $INITIALIZE section questions. Use of the "*" not only eliminates the details of what has been done, but can lead to problems (e.g., "NO" to "FORMAT/LABEL QUESTIONS" prevents asking TEST/FMT questions when "DDD LLINK-0 BAD..." - if LLINK-0 is unexpectedly bad and needs TEST/FMT, Startup must be aborted and the "INITIALIZE?" redone).

   A further recommendation is to use a $ DIRECT ddd/CLEAR statement, answer YES or RTHDRS to the TEST/FMT question, and process a $ DIRECT ddd statement followed by D statements. These D statements should be a set defining all bad space that the pack manufacturer has identified, bad space found during RSIP/MTAR/FTAR formatting, and any bad space found during system operation.

4. Disk formatting during Startup is not a replacement for the disk test functions in RSIP and MTAR. Disk formatting during Startup must be complemented with a $ DIRECT ddd statement and the necessary D, O, A, L, and Y statements to identify all known marginal or defective space.

CLEAR AND OVERWRITE EXISTING DECKFILE? Question

The CLEAR AND OVERWRITE EXISTING DECKFILE? question introduces the first step in a two-part function that edits new firmware onto the deckfile. This function normally is requested by site techniques personnel when new firmware is to be edited onto the deckfile. The CLEAR AND OVERWRITE EXISTING DECKFILE? question is issued only if a deckfile exists (i.e., only if the file was not cleared earlier in the initialization sequence as a result of a YES or TOTAL response to the *INITIALIZE? question).

A NO response indicates that the file is to remain intact. A YES response clears the file preparatory to rebuilding it from data contained on another firmware tape/deck. This tape/deck is read during processing of the *PROCESS $ READIN? question.

NOTE: On DPS 88 and DPS 90 systems, Startup creates an empty deckfile to edit the IFAD tape. Therefore, Startup provides for clearing of the deckfile for DPS 88 and DPS 90 systems.

On DPS 8 and DPS 8000 systems, Startup is required to bootload MPC firmware. A copy of the MPCB program must be available. Since MPCB is not saved on the AUTOLOAD file, the deckfile is accessed during an AUTO boot to obtain the MPCB program deck. If a CLEAR AND OVERWRITE EXISTING DECKFILE is performed during an AUTO boot, Startup cannot load MPCB and a Startup abort occurs. Therefore, a TAPE boot or a REPLace of the $INITIALIZE section with a copy of the MPCB deck is required to perform a clear and overwrite of the deckfile.

***PROCESS $ READIN? Question (DPS 8 And DPS 8000 Only)**

The *PROCESS $ READIN? question signals the second step in editing new firmware onto the deckfile. This question permits the contents of a firmware tape/deck to be written to the deckfile. (This question is immediately followed by an image of the $ READIN statement.)

A NO response indicates there is no firmware tape/deck to be read (i.e., the operator responded NO to the CLEAR AND OVERWRITE EXISTING DECKFILE? question). A YES response indicates that a firmware tape/deck is to be read and its contents are to be written to the deckfile. (This firmware tape/deck normally is supplied by site techniques personnel or by the site's Honeywell Bull Customer Service Division (CSD) representative.)

One of two questions, which query the operator as to whether firmware is to be loaded, may be issued to the system console following a YES response to the *PROCESS $ READIN? question, or following a response of PARTIAL to the *INITIALIZE? question.

The questions are:

```
*xxxMPC ON iccc BOOTLOAD SYS ID NAME "y...y" REV.zz?
*xxxMPC ON iccc APPEARS OK, BOOTLOAD?
```

where:

xxx = subsystem type (e.g., 450)
i = input/output controller number
ccc = channel number
y...y = system name from $ SYID statement
zz = firmware revision level from the $ DKEND statement that delimits the firmware deck

- o If the data images reside on punched cards, one of the preceding questions is issued when the firmware deck image is encountered.

- o If the data images reside on magnetic tape, one of the preceding questions is issued when the $ READIN statement is encountered in the $INITIALIZE section of the Startup job stream. The $ READIN statement identifies the magnetic tape handler from which the firmware tape is to be read.

- o If the data images reside on mass storage, one of the preceding questions is issued when bootloading is to occur.

*EDIT? Question

The *EDIT? question allows GCOS system files to be edited to mass storage (i.e., either created for the first time or content restored following mass storage device initialization or following a severe system problem).

The *EDIT? question is issued only when a $EDIT section is encountered in the Startup job stream. With one exception, this question is independent of the *INITIALIZE? question. The exception occurs when the operator responds TOTAL to the *INITIALIZE? question. Under these conditions, the *EDIT? question is automatically answered YES by Startup and $EDIT section contents are executed.

The operator can respond NO, YES, or PARTIAL to the *EDIT? question.

Response	Description of Edit Process
NO	A response of NO is applicable when (1) the *INITIALIZE? question was answered NO, (2) no system files were affected during the initialization function, (3) no new system files are to be added, or (4) no existing system files are to be modified.
YES	A response of YES is required to initially create and/or to restore all system files.

Response	Description of Edit Process

If a single magnetic tape handler is being used to edit the system, Startup rewinds the edit tape whenever it encounters a valid tape name on a $ FILDEF statement. The operator must dismount the old edit tape and mount the new one before replying "YES" to the EDIT question.

When a YES response is entered, Startup expects the total system software tape(s) to already be mounted on one or more magnetic tape handlers. (The $ FILDEF statements in the $EDIT section define the device(s) on which the total system software tapes are to be mounted.) If the tapes are not already mounted, the message:

    *TAPE ERROR icccdd: ATT I ssssss OPERATE WHEN READY

is issued directing the operator to do so (where i = Iyy number, ccc = channel number, dd = device number, and ssssss = device standby code). (Refer to the GCOS 8 OS I/O Programming manual for standby code definitions.)

(Because several reels constitute the total system software tape, the editing process can be expedited by allocating two tape handlers and mounting two reels. When reading of one reel completes, reading of the second reel can begin while the first reel is automatically rewound. The operator then can mount another reel on the free tape handler.)

Once the total system software tape is mounted, Startup verifies the reel serial number and issues the informational message REEL NO. nnnnnn (where nnnnnn = reel serial number). The tape then is read. This verify-and-read sequence occurs for each tape reel. If the reel number in the message REEL NO. nnnnnn identifies the wrong reel, the operator can press the REQUEST push button on the console to halt the read function, enter BOOT via the console, mount the correct tape reel, and reboot the system.

If the file name on a $ FILDEF statement differs from that of the file being read from the total system software tape, the console message *FILE IS aaaaaa, SHOULD BE bbbbbb SCFR? is issued (where aaaaa = name of the file being read and bbbbbb = name of the file specified on the $ FILDEF statement that should be read). The operator can respond with one of the following console entries.

S - Abort the Startup program. The console message STARTUP ABORTED LOC mmmmmm PICK ONE DUMP:ALL,NONE,NORMAL is issued. The operator can elect to print a main memory dump or to bypass the dump. (Refer to "Initiating Memory Dump" in Section 3 for additional information.)

C - Accept the error condition and continue processing the current file (aaaaaa).

Response	Description of Edit Process
	F - Find the required file (bbbbbb) by searching the tape forward from the current position.
	R - Rewind the tape and search forward for the required file (bbbbbb).
	If (following a YES or PARTIAL response to the *EDIT? question) the end of the tape is reached without encountering the required file, the console message:
	*END OF TAPE ENTER FILE NAME
	is issued. In addition, names of all files on the tape also are generated at the system console. The operator can (1) enter via the console the name of another file to be searched for and then enter R, or (2) mount another tape reel and enter F to search for the required file.
PARTIAL	A response of PARTIAL is required to (1) selectively add new system files and (2) selectively modify the content of one or more existing system files. Startup queries the operator regarding the disposition of each file that is identified on a $ FILDEF statement. The query is via the *EDIT ffffff? question (where ffffff = file name).
	The *EDIT ffffff? question is issued when a $ FILDEF statement is encountered in the $EDIT section. A response of NO indicates that the file is not to be edited. A response of YES indicates that the file is to be modified or added using information from the $ FILDEF statements in the $EDIT section.
	If the file to be added or modified is not the first file that Startup encounters on the total system software tape, the console message *FILE IS aaaaaa, SHOULD BE bbbbbb SCFR? is issued. A response of F results in the forward search of the tape for the file.
	One additional question may be issued during the editing function that occurs following a response of PARTIAL. This question is *PROCESS SSFILE CARD AND SYSTEM SCHEDULER CLEAR?. A response of NO indicates that existing $ SSFILE statement parameters are still valid. A response of YES indicates that the existing $ SSFILE statement parameters are to be modified and that the new parameters exist either on the total system software tape or on a $ SSFILE statement in the $EDIT section. (A response of YES requires that the operator respond NO to the *RESTART? question.) As a result of the YES response, the informational message:

*SYSTEM SCHEDULER CLEAR FORCED DUE TO PRIOR $ SSFILE CARD PROCESS

is issued to the system console.

*SYSTEM SCHEDULER CLEAR? Question

The *SYSTEM SCHEDULER CLEAR? question allows the operator to attempt restart of those jobs that were cataloged on the System Scheduler program's SSFILE file (catalog). In general, however, clearing of the SSFILE file is not necessary because jobs in the scheduling phase have not been allocated system resources (i.e., memory or peripheral devices) and have not entered the execution phase.

A response of YES to the *SYSTEM SCHEDULER CLEAR? question removes all entries from the SSFILE file (including jobs that were scheduled for running on a cyclic basis). All jobs that were cataloged on the SSFILE file at the time the system became inoperable must be reentered.

As previously indicated, Startup does not permit a YES response to both the *RESTART? and *SYSTEM SCHEDULER CLEAR? questions. If the operator responds YES to *RESTART?, Startup pre-answers the *SYSTEM SCHEDULER CLEAR? question NO.

A response of NO to the *SYSTEM SCHEDULER CLEAR? question retains all jobs that were cataloged on the SSFILE file at the time the system became inoperable and allows restart to be attempted for those jobs. The operator must enter JRUN REST via the system console to activate the scheduling function following restart. (Refer to "*RESTART? Question" in this section for additional information.)

If a system malfunction occurs before System Startup is complete, clearing of the SSFILE file may be incomplete. To ensure that the file is cleared, respond YES to the *SYSTEM SCHEDULER CLEAR? question during the ensuing bootload.

*

SECTION 3

STARTUP RECOVERY PROCEDURES

The System Startup process can be interrupted by two types of error conditions. One type of error aborts the Startup process and requires that the operator reinitiate the entire bootload procedure. The second type of error results in temporary interruption of the bootload procedure and allows the operator to correct the problem and resume System Startup at the point of interruption.

In addition to these two types of error conditions, the operator can intentionally render the system inoperable and reinitiate a system bootload.

Each of these conditions, and techniques to recover from them, are discussed in this section. Also included are procedures for obtaining dumps following a Startup program abort and for patching the Startup program. Descriptions of system aborts that may occur after the system has been bootloaded and is operational are not included in this section.

BOOTLOADER PROGRAM ERRORS

The Bootloader program loads the Startup program into memory. The Bootloader program must be entered in its entirety before Startup program loading can begin.

Once the Bootloader program completes, the Startup program can be loaded into memory. Any error encountered during this loading process results in a console message. No dump can be taken when an error is detected before the Startup program is completely loaded.

Startup assumes the system console is configured on channel 31 of IOM-0, or channel 30 of IMU-0 (for DPS 8), or channel 30 of the boot IMX (for DPS 8000), and attempts to direct the error messages to a console at that location. If the system console is configured on any other input/output controller or channel, the following MASK correction statement must be inserted immediately following the last statement (a ***EOF statement) of the Bootloader program.

```
            1
1      8    3
000003 MASK 31cccct000ip
```

where:

```
31   - Console type (EMC/SCC)
cccc - PUB (channel) number times 4 (octal)
t    - Type of input/output controller - IOM = 0
                                         IMU/IMX = 4
i    - IOM/IMU/IMX number
p    - IOM/IMU/IMX port number
```

NOTE: On the DPS 88, the console emulator is logically connected to channel 31 of IOX-0. On the DPS 90, the console location is passed to Startup via reserved memory. Therefore, in both of these cases, the MASK statement will be rejected.

The type of controller specifies the console I/O protocol. The defaults will be determined as follows:

1. If there is no console MASK, the hardware-set flags (IOM/IMU/IMX) will determine the protocol.

2. If a MASK is used and specifies an IOM/IMU/IMX that is not the bootload IOM/IMU/IMX, the MASK specified protocol will be used.

3. If a MASK is used and specifies the bootload IOM/IMU/IMX, and if IOM protocol is specified (t=0) but the controller is an IMU/IMX (hardware flagged), then the IMU/IMX protocol will be used. However, if the bootload controller is an IOM but the MASK specifies IMU/IMX protocol (t=4), the MASK will be ignored and flagged as an error as follows:

   a. If the erroneous MASK was applied to the Startup program before the transfer record to cards or tape, the Startup program will halt with a DIS identified in the instruction as 400020616203.

   b. If the erroneous MASK was input via REPL (in response to BOOT SOURCE: AUTO OR REPL?), it will be identified in the following message:

      ILLEGAL MASK CARD BELOW IGNORED
      (mask card image)

Only one console message is issued if a Startup program error is encountered while bootloading the program from magnetic tape. This message is STATS/RESIDU = x...x ABORT BOOT FROM TAPE (where x...x = 12-digit (octal) magnetic tape error status). Correct the error and reinitiate the bootload by rewinding the tape, and pressing the INITIALIZE and BOOTLOAD push buttons on the DPS 8 system console, or by following the DPS 8000 IMX Service Processor procedures. (Refer to the GCOS 8 OS I/O Programming manual for a description of magnetic tape status codes.)

Several console messages are associated with Startup program errors encountered while bootloading the program from punched cards. These messages include the following:

Error Message	Description
STATS/RESIDU= x...x -FIX,EOM	A status error was detected. Correct the error and press the EOM push button on the system console to resume the bootload. (Refer to the GCOS 8 OS I/O Programming manual for a description of card reader status codes.)
ILLEGAL BIN. -FIX,EOM	The Startup program includes an invalid binary card. Remove the card and press the EOM push button on the system console to resume the bootload.
CRD CHECKSUM -FIX,EOM	The Startup program contains a statement with a checksum error. Correct the error and press the EOM push button on the system console to resume the bootload.
CARD CHECKSUM ERROR FIX TYPE GO OR TYPE DUMP	The Startup program contains a statement with a checksum error. Correct the error and enter GO, or enter DUMP and follow the standard dump procedure.
CRD SEQUENCE -FIX,EOM	A card is out of sequence in the Startup program. Resequence the cards and press the EOM push button on the system console to resume the bootload.
CARD SEQUENCE ERROR FIX TYPE GO OR TYPE DUMP	A card appeared out of sequence. Resequence the cards correctly and enter GO, or enter DUMP and follow the standard dump procedure.
MISSING/EXTRA CARDS -FIX,EOM	Either a required statement is missing from or an unrecognized statement is included in the Startup deck. Include the required statement or remove the invalid statement and press the EOM push button on the system console to resume the bootload.
ILLEGAL HOL. -FIX,EOM	The Startup program includes an invalid BCD card or an erroneous MASK correction statement. Correct the statement and press the EOM push button on the system console to resume the bootload.
ILLEGAL BCD CARD FIX TYPE GO OR TYPE DUMPGO	The Startup program includes an illegal BCD statement. Correct illegal statement and enter GO or enter DUMPGO.

Error Message	Description
DUPED MASKS -FIX,EOM	The Startup program includes duplicate MASK correction cards. Remove one of the cards and press the EOM push button on the system console to resume the bootload.
ILLEGAL OCT. -FIX,EOM	The Startup program includes a MASK correction card that contains non-octal data. Correct the card and press the EOM push button on the system console to resume the bootload.

RECOVERABLE STARTUP PROGRAM ERRORS

When the Startup program completes loading into memory, it begins processing Startup job stream contents. A variety of error conditions can be encountered during this processing phase. With few exceptions, it is possible for the operator to correct the error condition and continue the bootload.

Most of the errors are device-oriented and dependent upon the medium (i.e., magnetic tape or mass storage) on which the Startup job stream resides. The following defines several console messages that are commonly issued during this portion of the bootload. The messages are defined on a device basis. In each of the messages, i = I/O controller number, ccc = channel number, dd = device number, and s...s = status condition. (Refer to the GCOS 8 OS I/O Programming manual for a description of error statuses.)

The following console messages are issued when errors are encountered while processing a Startup job stream that resides on punched cards, which are being entered via the card reader:

Error Message	Description
*BOOTLOAD DEVICE ERROR, icccdd: s...s OPERATE WHEN READY	The card reader is not in a ready condition. The status of the device is identified via s...s. Correct the condition defined by s...s and resume input.
*BOOTLOAD DEVICE ERROR, icccdd: s...s BACKSPACE OPERATE WHEN READY	An error condition was encountered while reading the last card. This condition is defined by s...s. Clear any cards from the reader, backspace the reader, and resume input.
*CARD CHAR ILLEGAL, icccdd: BACKSPACE, OPERATE WHEN READY	The last card read contained an invalid character. Correct the card, clear any cards from the reader, backspace the reader, and resume input.

The following console messages are issued when errors are encountered while processing a mass storage device:

Error Message	Description
*DISK PACK ERROR, icccdd: s...s EOM WHEN READY	An error condition was encountered while reading from or writing to mass storage. This condition is defined by s...s. Correct the condition and press the EOM push button on the system console to resume the bootload.
*DISK PACK ERROR, icccdd: s...s FIX, TYPE GO OR STOP OR RLSE	An error condition was encountered while reading from or writing to mass storage. This condition is defined by s...s. Correct the condition and enter GO via the system console to resume the bootload or STOP to abort Startup. If STOP is entered, the messages *STARTUP ABORTED LOC mmmmmm and *PICK ONE DUMP: ALL,NONE,NORMAL are issued. (Refer to "Memory Dump Following Startup Abort" in this section for the dump procedure.) Enter RLSE to release the device and continue the bootload. The device subsequently can be reassigned via the console entry ASGN icccdd.

The following console message is issued when errors are encountered while processing a magnetic tape device:

Error Message	Description
*TAPE ERROR, icccdd: s...s MAY BE RECOVERABLE FIX, TYPE GO OR STOP OR TYPE CONT TO CONTINUE	An error condition defined by s...s was encountered. The operator can (1) correct the condition and enter GO via the system console to resume the bootload; (2) enter STOP via the system console to abort Startup; or (3) enter CONT via the system console to release all magnetic tape handlers configured on this MPC subsystem and, subsequently, to resume the bootload. If STOP is entered, the messages *STARTUP ABORTED LOC mmmmmm and *PICK ONE DUMP: ALL,NONE,NORMAL are issued. (Refer to "Memory Dump Following Startup Abort" in this section for the dump procedure.)

The following console messages are issued when printer-related errors are encountered while processing the Startup job stream. Refer to "Character Set And VFC Image Loading" in Section 2 for a description of the recovery procedure associated with each message.

*PRINTER ERROR icccdd s...s FIX TYPE GO OR STOP OR RLSE

*PRINTER ERROR icccdd s...s DEVICE RELEASED

*TRAIN ID UNKNOWN TO STARTUP DEVICE RELEASED

*POSITION PAPER IN PRINTER icccdd FIX TYPE GO OR STOP OR RLSE

*MOUNTED TRAIN ID xxx REQUESTED TRAIN ID yyy DEVICE RELEASED

STARTUP PROGRAM ABORTS

If the Startup program aborts during the bootload procedure, a console message is issued defining the reason for the abort. Following issuance of this message, an informational message is issued defining the main memory location at which the abort occurred. A third message then is issued giving the operator several dump-related options. (Refer to "Memory Dump Following Startup Abort" in this section for a discussion of the dump procedure.)

After the operator has responded to the dump options and after any dump output has been printed, the question *BOOT SOURCE: AUTO OR REPL? is issued to the console. This question gives the operator the opportunity to reinitiate the bootload procedure. (Refer to "*BOOT SOURCE: AUTO OR REPL? Question" in this section for a discussion of the various restart options.)

The following console messages are commonly issued when Startup aborts.

Error Message	Description
*DISK PACK ERROR, icccdd: ACCESS BEYOND FILE fffff CANNOT PROCEED	An attempt was made to access file fffff beyond the file boundary. Startup aborts under this condition. This message is followed by the messages *STARTUP ABORTED LOC mmmmm and *PICK ONE DUMP: ALL,NONE,NORMAL. (Refer to "Memory Dump Following Startup Abort" for the dump procedure.) Increase the file size on the $ FILDEF statement in the $EDIT section and reinitiate the bootload. If this does not correct the error, contact the site's CSD representative.

Error Message	Description
*DISK PACK ERROR, icccdd: s...s CANNOT PROCEED	An error condition defined by s...s was encountered. Startup aborts under this condition. This message is followed by the messages *STARTUP ABORTED LOC mmmmm and *PICK ONE DUMP: ALL,NONE,NORMAL. (Refer to "Memory Dump Following Startup Abort" for the dump procedure.) Correct the condition defined by s...s and reinitiate the bootload.
*FATAL I/O ERROR ON icccdd: s...s	An error defined by s...s was encountered. Startup aborts under this condition.
*TAPE ERROR, icccdd: s...s CANNOT PROCEED	An error condition defined by s...s was encountered. Startup aborts under this condition.

OPERATOR-INITIATED STARTUP AND SYSTEM ABORTS

The console entries BOOT and DUMP immediately render the system inoperable. These entries can be made in response to any Startup question during the bootload sequence. The operator also can press the REQUEST push button on the console while the system is operational and, in response to the console message ???, can enter either BOOT or DUMP.

In addition to the BOOT and DUMP entries, the operator can create an error condition and abort the system or the bootload procedure by placing the EXECUTE SWITCHES/EXECUTE FAULT switch on the processor maintenance panel in the EXECUTE FAULT position. (Refer to "Dumps Initiated From A VIP Maintenance Terminal" in this section for a discussion of this procedure.)

The BOOT and DUMP entries often are useful when Startup program/job stream or bootload changes are desirable. For example, BOOT can be entered to reinitiate the bootload sequence if the operator wishes to change a response to a Startup question (e.g., to change a response of YES to the *INITIALIZE? question).

Each of the preceding techniques results in the system becoming inoperable, in the saving of system restart data, and in the question *BOOT SOURCE: AUTO OR REPL? being issued to the console. In addition, the DUMP entry and use of the EXECUTE SWITCHES/EXECUTE FAULT switch result in a series of dump-related questions and messages being issued to the console prior to the *BOOT SOURCE: AUTO OR REPL? question. (Refer to "Memory Dump Following Startup Abort" in this section for a discussion of the dump process.)

As a result of the DUMP entry, portions or all of the Startup job stream are printed on a printer that is not dedicated.

- o If the system was being bootloaded from a Startup job stream residing on punched cards or magnetic tape, only the statement images that were read prior to the malfunction are printed.
- o If the system was being bootloaded from a Startup job stream residing on mass storage (i.e., the AUTOLOAD file), the entire Startup job stream is printed.

The contents of the $LOAD section are not printed in either of these cases. In addition, none of the Startup job stream is printed if the operator requests a dump or if Startup aborts while the $LOAD section is being loaded into main memory.

*BOOT SOURCE AUTO OR REPL? QUESTION

The *BOOT SOURCE: AUTO OR REPL? question is issued when a system failure occurs or when the operator deliberately renders the system inoperable (e.g., to perform system maintenance). As previously indicated, the system becomes inoperable when the operator enters BOOT or DUMP via the system console.

The following responses can be made to this question:

AUTO - The Startup program residing on the AUTOLOAD file is to be used to effect the bootload. This response is not acceptable if the AUTOLOAD file was destroyed by the system failure or is incomplete. Under these circumstances, the message MUST REPLACE FROM s...s is issued (where s...s = Startup section that must be replaced). The *BOOT SOURCE message is reissued. The operator must respond REPL and enter the required section via the card reader. Once the AUTOLOAD file is modified, file content is used to bootload the system.

REPL - The operator can selectively replace one or more sections of the Startup job stream which exists on the AUTOLOAD file. The section(s) must be entered in their entirety via the card reader. Selective replacement of individual statements is not permitted. The entire section is written to the AUTOLOAD file to overlay (replace) the existing section.

Only the affected sections require loading. For example, if the $INITIALIZE section is being replaced, the replacement section need not be preceded by the $CONFIG section or followed by the $EDIT and $FILES sections.

If more than one section is being replaced, the sections must be loaded in their order of appearance in the Startup job stream. For example, if the $CONFIG and $INITIALIZE sections are being replaced, the $INITIALIZE section cannot precede the $CONFIG section.

If sections are nonsequential, the message *EXPECT s...s ccccc FIX TYPE GO is issued to the console (where s...s = name of the section that is out of sequence and ccccc = image of the last card read). To correct a sequencing problem, the operator can enter BOOT via the system console to render the system inoperable. After placing the sections in the proper sequence, the operator can enter REPL or GO to resume the bootload.

When the operator responds AUTO to the *BOOT SOURCE question, the deckfile cannot be rebuilt because the $ READIN statement is excluded from the $INITIALIZE section existing on the AUTOLOAD file. To rebuild the deckfile via the $ READIN statement, the $INITIALIZE section can be entered via the card reader following a REPL response to the *BOOT SOURCE question.

Modules included in the $LOAD section are saved on the loadfile if a $ LOADFIL statement is included in the Startup job stream. Under these circumstances, these modules are automatically loaded when the system is restarted. If a loadfile is to be replaced, the $LOAD section can be entered via the card reader following a REPL response to the *BOOT SOURCE question.

PATCHING STARTUP PROGRAM

The Startup program can be patched via MASK correction statements. These statements are written to the AUTOLOAD file and are applied to the Startup program each time the system is bootloaded. A copy of the patched location's original content is saved on the file. If the applicable MASK statement is removed, the copy is applied on the next bootload to restore original data to the location.

MASK statements must immediately precede the transfer statement which is the last statement in the Startup program. (Exceptions to this are (1) the MASK statement that defines the input/output controller and channel on which the system console is configured and (2) MASK statements that are to be applied to the Bootloader program. These statements immediately follow the Bootloader program. Refer to "Bootloader Program Errors" in this section for a discussion of this statement.)

Modification, deletion, or addition of one or more MASK statements requires that all applicable MASK statements be loaded via the card reader during the bootload. The operator can respond using one of the following methods:

1. Respond REPL to the question:

   *BOOT SOURCE: AUTO OR REPL?

2. At the end of the MASK section, insert:

   ***EOF

   instead of a TRANSFER statement.

The format of the MASK correction statement follows. Fields beginning in columns 1, 8, and 13 must be completed. If they are not, bootloading halts and the console message ILLEGAL OCT. -FIX,EOM is issued. Correct the statement and press the EOM push button on the system console to resume the bootload.

```
            1          2                                             7
1     8     3          6                                             3
______________________________________________________________________

aaaaaa MASK dddddddddddd cccccc                                  .MINIT
```

where:

aaaaaa - Absolute address (octal) of the location to be patched in the Startup program. This address must contain six digits (i.e., leading zeros must be included if the address is fewer than six digits).

d...d - Patch correction data (octal) to be applied to location aaaaaa. This data must contain 12 digits (i.e., leading zeros must be included if the data is fewer than 12 digits).

cccccc - Patch-related comments optionally can be included in columns 26-72.

.MINIT - An optional Startup program identifier.

DEBUGGING AIDS

Three techniques can be employed to generate main memory and mass storage dumps during System Startup. These include an operator-initiated dump and use of $ SNAP and $ DUMP statements in the Startup job stream.

Memory Dump Following Startup Abort

There are several conditions under which a memory dump can be effected and which subsequently require the system to be bootloaded. Among these are the occurrence of an abort during System Startup, an operator-initiated malfunction via the console entry DUMP, occurrence of an unrecoverable system error, and setting of a switch on the processor maintenance panel.

Described below is the procedure to obtain a dump when the Startup program aborts during bootloading and how master mode dumps can be initiated from the processor maintenance panel. Other dump procedures are described in the GCOS 8 OS System Operating Techniques manual.

INITIATING MEMORY DUMP

When an abort occurs during System Startup, two messages are issued. One message defines the main memory location (mmmmmm) at which the abort occurred. The second message allows the operator to dump main memory, to dump only buffer areas and the communication region, or to forego any dump.

All dump output is directed to an undedicated printer (i.e., dumping to magnetic tape or to mass storage is not available during System Startup). If printer PR1 is available, the dump is directed to that device. If PR1 is not available, the dump is directed to any available undedicated printer.

```
*STARTUP ABORTED LOC mmmmmm

*PICK ONE DUMP: ALL,NONE,NORMAL
```

The following responses can be made via the system console:

NORMAL - Dump only Startup buffer areas and all of the communication region. Under most conditions (i.e., except those cited with the ALL response), this response provides adequate debugging information. The date nd time of the dump are printed in the dump banner and in a console message.

ALL - Dump all of main memory. This response is necessary only when illegal procedure, memory, or parity faults are encountered. The date and time of the dump are printed in the dump banner.

NONE - Do not take a dump.

If the Startup program aborts before the $CONFIG section of the Startup job stream has been processed, the printer to which the dump is to be directed (in response to an entry of NORMAL or ALL) is unknown to Startup (i.e., the printer is defined in the $CONFIG section). The console message:

```
*NEED PRINTER PORT,Iyy,CHAN,MPC/CPI,TRAIN(PICCCS [TTTT])
```

is issued. The operator must identify the printer via a console entry in the format piccestttt,

where:

p - input/output controller port number
i - input/output controller number
ccc - Channel number
s - Printer type:
  M - Printer is connected to an MPC
  C - Printer is not connected to an MPC
tttt - Character set identification number. Startup recognizes only the following character set identification numbers:

  764 - Standard BCD character set for PRU1200/1600 printers
  1130 - Standard ASCII character set PRU1200/1600 printers

If the printer is connected to an MPC that was not bootloaded, a dump cannot be taken. An informational message is issued to the console indicating a device power off status. The operator can load the firmware from punched cards, magnetic tape, or mass storage. (Refer to "MPC Bootloading (DPS 8 And DPS 8000 Only)" in Section 2 for additional information.)

DUMPS INITIATED FROM A VIP MAINTENANCE TERMINAL (DPS 8 ONLY)

Master mode dumps can be initiated from a VIP maintenance terminal using the following procedure:

1. Set the Maintenance Panel mode switch to TEST, thus enabling the Processor Maintenance Panel.

2. Press the RESET CONSOLE push button on the system console to reset the console (if necessary).

3. At the maintenance VIP terminal, enter EX4 <RETURN> to execute the fault.

If the preceding procedure does not generate the dump, perform the following:

1. Enter ST CU, ST VU <RETURN> to stop the processor (i.e., Control Unit and Virtual Unit).

2. Enter TXA 002000 <RETURN> to enter the transfer address (i.e., the address referenced by module .CRSBT from which the dump will execute).

3. Enter GO <RETURN> to execute the dump.

NATIVE STAND-ALONE DUMP PROGRAMS

There are three Native Stand-Alone (NSA) dump programs that can be used to obtain a physical memory dump when all normal methods of obtaining a system dump have been unsuccessful. NSAR is used for the DPS 8000, NSA8 for the DPS 88, and NSAD for the DPS 8. (No corresponding program is provided for the DPS 90 because the necessary functions are provided by MSOS.) These programs write the contents of the physical memory connected to their respective systems to a magnetic tape. The tape should then be processed by the DCPY (Dump CoPY) program described below.

For NSAD (DPS 8) this tape contains the binary image of the program (currently two binary card image records), followed by a two-record set for each 4K (4096) words of physical memory.

For NSA8 (DPS 88) this tape contains the binary image of the program (currently two binary card image records), followed by a 4K record for each 4K (4096) words of physical memory.

For NSAR (DPS 8000) this tape contains the binary image of the program (currently four binary card image records), followed by a two-record set for each 4K (4096) words of physical memory.

The first record of each set contains information used by the DCPY program to determine the physical memory address of the data in the second record of the set. If the NSA program encounters a tape write error, it retries the current write sequence until the write is successful or until the end of the magnetic tape has been reached. If the NSA program succeeds in writing all of the memory to the magnetic tape before reaching the end-of-tape marker, it writes an end-of-file (EOF) mark on the tape prior to issuing a rewind and standby (REWS) command. If the end of the tape is reached prior to reaching the end of memory, the NSA program writes an EOF mark, but does not issue any rewind command; the system operator must manually rewind the tape before resuming normal system operations.

The NSA programs can be booted into memory from either a non-FIPS or FIPS tape controller. If a non-FIPS tape controller is used, a 9-track tape written at 1600 BPI is recommended. (Note: A 7-track tape written at 800 BPI can be used; however, a significantly smaller amount of physical memory will be written to the tape.) For FIPS tape controllers, a 9-track tape written at either 1600 BPI or 6250 BPI can be used.

The NSA programs can write more than 8 MW (megawords) of physical memory to the dump tape at 1600 BPI. This amount of memory should be sufficient to allow a systems techniques person to determine the cause of the system failure.

Creating Bootable NSA Tapes

After the GCOS 8 system is operational, the following JCL can be used to create a bootable dump tape containing the appropriate NSA program.

```
               1
1      8       6
-----------------------------------------------------------------
$      IDENT   site option
$      FILEDIT SOURCE,NOBJECT
$      LIMITS  100,48K,,5000
$      TAPE    M*,X1D,,#####,,VS-PRIMARY
$      PRMFL   *7,W,S,source program file
$      DATA    *C,,COPY
$      LIST    xxxx,COMDK
$      ENDEDIT
$      ENDCOPY
$      GMAP    DECK,NSAF,ON5
$      LOWLOAD
$      PRMFL   G*,R,S,source program file
$      FILE    C*,X1S,25L
$      PROGRAM SCED
$      FILE    IN,X1D
$      FILE    OT,X2S,25L
$      DATA    A*,,COPY
$      CHANGE  1,1             See NOTE
$      CHANGE  5,5             See NOTE
$      ENDCOPY
$      PROGRAM WSBT
$      FILE    I1,X2D
$      TAPE9   OT,X3D,,,,RAWDUMP,,DEN16
$      ENDJOB
```

where xxxx on the $ LIST statement is

NSAR for the DPS 8000
NSA8 for the DPS 88
NSAD for the DPS 8

NOTE: For NSAR, the $ CHANGE statements should be

```
$      CHANGE  1,1
$      CHANGE  7,7
```

Dumps Initiated At The DPS 8 MCA

The following procedure can be used at the DPS 8 MCA (Maintenance Computer Adapter) to boot the NSAD program into DPS 8 memory.

1. Enter <Escape>#nn where nn is the MCA number (e.g., 01). This allows the next line of input to be processed by the MCA, and not by GCOS 8. The MCA will display the 'greater than' character ('>') to prompt for input.

2. The NSAD boot/dump tape should be mounted on the tape handler selected by the device switches of the Tape MPC. The Tape MPC does not allow writes to be issued to the dump tape if the RSO signal has been received.

3. Enter the IBOOT command to reinitialize the system hardware. The reinitialization of the hardware takes several minutes. When it completes, the two boot records from the NSAD boot/dump tape will be booted into memory and NSAD will write 4 MW (megawords) of memory per minute to the boot/dump tape. The configuration file used must have the RSO REQ variant set to N for the tape channel used to boot the NSAD dump tape.

Dumps Initiated At The IOM Maintenance Panel (DPS 8 Only)

The following procedure can be used at the IOM maintenance panel to boot the NSAD program into DPS 8 memory.

1. Press the SYSTEM INITIALIZE button on the IOM maintenance panel.

2. Wait for the console to become ready.

3. The NSAD boot/dump tape should be mounted on the tape handler selected by the device switches of the Tape MPC. The Tape MPC does not allow writes to be issued to the dump tape if the RSO signal has been received.

4. Push the RESET/BRANCH button on the Tape MPC being used. Ensure that the HEX dial is set 480 first.

5. Push the BOOT button on the IOM maintenance panel.

6. The two boot records from the NSAD boot/dump tape will be booted into memory and NSAD will write 4 MW (megawords) of memory per minute to the boot/dump tape.

Dumps Initiated At The Console (DPS 8 Only)

The following procedure can be used at the DPS 8 console to boot the NSAD program into DPS 8 memory.

1. Enter <Escape>I to initialize the hardware.

2. Wait for the console to become ready.

3. The NSAD boot/dump tape should be mounted on the tape handler selected by the device switches of the Tape MPC. The Tape MPC does not allow writes to be issued to the dump tape if the RSO signal has been received.

4. Push the RESET/BRANCH button on the Tape MPC being used. Ensure that the HEX dial is set 480 first.

5. Enter <Escape>B to boot the hardware.

6. The two boot records from the NSAD boot/dump tape will be booted into memory and NSAD will write 4 MW (megawords) of memory per minute to the boot/dump tape.

Dumps Initiated Via DPS 8000 Service Processor

In order to boot the NSAR program into DPS 8000 memory, the system operator must be able to communicate with the service processor attached to the system. At the service processor the operator should enter

```
<Escape>
#SP
```

to inform the service processor that the next input is to be processed by it and not GCOS 8. The service processor responds with a greater than ('>') character to prompt for input.

Sample Console Dialog For DPS 8000 Dump

Figure 3-1 shows a console log from a tandem DPS 8000 system. The system operator used the RLOAD command to inform the service processor that both IMUs configured were to be reloaded. IMU #0 was loaded before IMU #1. Approximately three minutes later, the operator entered the IBOOT command to boot the NSAR boot/dump tape into memory. Approximately two minutes after the IBOOT command was entered, the NSA tape moved off the load point. NSAR writes approximately 4 MW (megawords) of memory per minute to the boot/dump tape.

The configuration file used to boot the NSAR boot/dump tape must have the RSO REQ variant set to N in order for the NSAR program to function correctly. NSAR writes to the boot tape after all four of the binary card images have been loaded into memory and placed in execution. The tape MPC does not allow writes to be issued to the dump tape if the RSO signal is sent to the tape MPC.

The configuration file used in Figure 3-1 could be listed via the LISTCFIG command. To do this, the operator enters the pound sign character '#' followed by the characters '00'. The MCA responds with a greater than character '>'. The operator may now enter the LISTCFIG command to display the current configuration file.

```
#50>MCA 01 RLOAD IMU .Cfilename
#50>MCA 00 RLOAD IMU .Cfilename
 ...
# CONSOLE SELF TEST SUCCESSFUL #
# IPC CONSOLE READY (F/W TAB 003) #


# MULTIDROP ENABLED #

 ...

#00<STATE status
#00< ...
#00< ...
#00<Please set new date & time with the command:
#00<TIME mmddyy,hhmmss
 ...
#SP>IBOOT_

 ...

SCU 0 CONFIGURATION LOADED
 ...
SCU 1 CONFIGURATION LOADED
 ...
CPU 0 LOADED CONFIGURATION
 ...
CPU 1 LOADED CONFIGURATION
 ...
<<0000>> - LOAD APPLICATION - NORMAL TERMINATION
 ...
<<0000>> BOOT SENT - BOOT NORMAL TERMINATION

<<0000>> INIT - NORMAL TERMINATION

 ...

#00<MSG  MCA  SYSTEM BOOT IN PROGRESS:  CH 008  DEV 00
 ...
```

Figure 3-1. Sample DPS 8000 Console Log For Dump By NSAR

DCPY Description

The DCPY program copies dump tapes created by NSAR, NSAD, NSA8, or SMAS to a user specified disk dump file. The user must create the dump file as a random file with an initial size of 3300 llinks per megaword of physical memory prior to running DCPY. For example, the dump file size necessary for a 4 megaword system is 4 * 3300 llinks, or 13,200 llinks. This dump file can then be analyzed by PERSUE 8 or printed by .MMDMP. The dump file will not contain any of the following data:

- o Cache Memory
- o Associative Memory
- o Reserved Memory
- o EPILOG data -- the BCD card images from the Startup job stream
- o Summary data for each process in execution at the time of the system hang

SYSOUT report code 00 contains a list of seek addresses corresponding to each tape record processed. When DCPY terminates, it produces a few lines of summary data indicating how many tape reads and disk writes were done. Also, a list of tape read errors for the NSAD/NSAR dump tapes is given.

The following options may be specified with DCPY on the $ EXECUTE statement.

STARTUP
: This option indicates that the dump tape was produced when the GCOS 8 system was not fully operational. DCPY will copy the first megaword of physical memory to the dump file.

NEWNSAD
: This option must be used to process the dump tapes created by the NSAD or NSAR programs. DCPY reads data from the dump tape in two-record sets.

The following options are intended to be used for internal debugging of the DCPY program.

ZDEBUG
: This option allows the use of the master-mode debugger (.MDEBG) while DCPY is executing.

TESTn
: This option allows simulation of a limited amount of program flow without requiring an NSA program dump tape. The value of 'n' can be from 0 to 3:

0 - No testing
1 - Simulate dump tape created by SMAS
2 - Simulate dump tape created by NSAD
3 - Simulate dump tape created by NSA8

The following JCL illustrates the use of DCPY. The file USER/DCPY.OBJ contains an object deck produced by assembling the source for the DCPY program, which is packaged on the M* source tapes, plus any alters contained on the latest MRT or the PASS database on System T in Phoenix. The file USER/DCPYFILE must be created prior to running the JCL.

```
               1
1      8       6
-----------------------------------------------------------------
$      SNUMB   site option
$      IDENT   site option
$      LOWLOAD
$      SELECT  USER/DCPY.OBJ
$      EXECUTE DUMP,NEWNSAD
$      LIMITS  9,36K,,20K
$      TAPE9   IN,T1D,,#####,,NSAD,,DEN16
$      PRMFL   OT,W,R,USER/DCPYFILE
$      ENDJOB
```

After DCPY has written the contents of the dump tape to the specified file, analysis can begin using either PERSUE 8 or the system dump analyzer .MMDMP. To access the dump file from .MMDMP, use a $ PRMFL statement similar to:

```
$      PRMFL   IN,R,R,USER/DCPYFILE
```

Creating The DCPY Object Deck

The following JCL can be used to obtain the source file (USER/DCPY.SRC) and create an object file (USER/DCPY.OBJ) from the M* source library tapes. Alters from the latest MRT or the PASS database on System T in Phoenix can be applied using the procedure documented in the SIB.

```
               1
1      8       6
-----------------------------------------------------------------
$      SNUMB   site option
$      IDENT   site option
$      FILEDIT SOURCE,NOBJECT
$      LIMITS  100,48K,,5000
$      TAPE    M*,X1D,,#####,,VS-PRIMARY
$      PRMFL   *7,W,S,USER/DCPY.SRC
$      DATA    C*,COPY
$      LIST    DCPY,COMDK
$      ENDEDIT
$      ENDCOPY
$      GMAP    DECK,NSAF,ON5
$      PRMFL   G*,R,S,USER/DCPY.SRC
$      PRMFL   C*,W,S,USER/DCPY.OBJ
$      ENDJOB
```

UTILIZATION OF THE DIAGNOSTIC PROCESSOR UNIT

With the assistance of a Honeywell Bull Customer Services Division (CSD) representative, the Diagnostic Processor Unit (DPU) can be used for the following purposes:

- o System maintenance, when executing instruction (KWIK) tests on isolated procedures, or on IOMs and SCUs, once they are physically deallocated (via the RLSEP console verb) from the system.

- o Connecting to the system console for online/offline system testing and development.

- o Connecting to an individual line on a system, which allows for maintenance of normal time sharing connections.

- o Diagnosing DATANET 8s from a local or remote site.

To initiate a dump from the DPU, the site should contact the Honeywell Bull Response Center in Phoenix.

DUMP BANNER

A one-line banner is common to several types of Startup-generated printer output: Startup dumps, file dumps, system maps, and file maps. The format of this banner is as follows.

GCOS 8 p...p (aaaaaa) DATE bbbbbb TIME hh.ttt SYSTEM ID-nnnnnn SYSTEM NO.-0

where:

p...p - FILE MAP, identifies a map of the internal Startup file catalog.

SYSTEM MAP, identifies a map of the GCOS file structure.

FILE DUMP, identifies a dump of Startup-initiated files and which is initiated via the $ DUMP statement.

STARTUP DUMP, identifies a dump that was initiated via a console entry of NORMAL or ALL in response to the console message *PICK ONE DUMP: ALL,NONE,NORMAL.

aaaaaa - Date from the TTL statement in the Startup assembly job stream. The format of this date is yymmdd.

bbbbbb - Date of this output. The format of this date is mmddyy.

hh.ttt - Time of this output (where hh = hour 00-23 and ttt = thousandths of one hour).

nnnnnn - Identification of this system. This identification is specified on the $ SYID statement in the $CONFIG section.

STARTUP SNAPSHOT DUMP

A Startup-initiated snapshot dump is automatically directed to an undedicated printer whenever Startup aborts as the result of an error in the Startup file system. Dump content includes:

- o Processor register contents at the time of the abort, occupying the three lines immediately following the banner.

- o Content of the Startup file system buffers residing in main memory follow the processor register information.

After the snapshot dump is printed, the console message:

```
*PICK ONE DUMP: ALL,NONE,NORMAL
```

is issued. If an illegal procedure fault caused Startup to abort and if the operator responds NORMAL or ALL to the console message, history register or transfer register content is printed following the snapshot dump.

Dump Via $ SNAP Statement

The $ SNAP statement can be used to generate a dump of one or more locations in main memory. The dump is directed to an undedicated printer.

The $ SNAP statement is used frequently to monitor main memory that is affected by a module in the $LOAD section of the Startup job stream.

The $ SNAP statement can be included in any section of the Startup job stream. The format for this statement is:

```
               1
1      8       6
------------------------------------------------------------
$      SNAP    aaaa,nnnn
```

where:

aaaa - Main memory address (octal) at which the dump is to begin.

nnnn - Number of words of main memory to be dumped, beginning at location aaaa.

The following example indicates that the contents of 64 words of main memory are to be dumped beginning at location 2200 (octal).

```
               1
1      8       6
------------------------------------------------------------
$      SNAP    2200,64
```

Dump Via $ DUMP Statement

The $ DUMP statement can be used to dump the content of a specific file or of one or more contiguous llinks/links of a device. Once the debugging procedure is complete, it is recommended that the $ DUMP statement be removed from the Startup job stream to avoid dumping the same locations on a subsequent bootload.

A $ DUMP statement can be included in the $INITIALIZE, $EDIT, $FILES, or $PATCH sections of the Startup job stream. The statement is invalid if it is included in the $CONFIG or $LOAD sections. The console message:

```
*ILLEGAL xxxxxx CARD BELOW IGNORED
```

(where xxxxxx = $CONFIG or $LOAD) is issued. This message is followed by the $ DUMP statement image.

Because processing of the $INITIALIZE and $EDIT sections is dependent upon a positive response by the operator to the *INITIALIZE? and *EDIT? questions during the bootload, it is recommended that the $ DUMP statement(s) be included in the $FILES section, which is always processed.

Two $ DUMP statement formats can be used. In each case, dumping of noncontiguous areas on a device or within a file requires multiple $ DUMP statements.

DUMP LLINKS/LINKS

The following $ DUMP statement format allows dumping of one or more contiguous llinks or links from a device:

```
                1
1      8        6
_____________________________________________________

$      DUMP     ddd,{LLINK}{-}bbb{,}xxx
                    {LINK }{/}   {/}
```

where:

ddd - Logical device name of the mass storage device.

LLINK - One or more llinks (320 words) are to be dumped.

LINK - One or more links (12 llinks) are to be dumped.

bbb - Number of the llink or link at which dumping is to begin. If bbb is preceded by a minus sign (-) the number is decimal; if preceded by a slash (/) the number is octal.

xxx - Number of contiguous llinks or links to be dumped, beginning with llink/link number bbb. If xxx is preceded by a comma (,) the number is decimal; if preceded by a slash (/) the number is octal.

The following example indicates that 18 (decimal) contiguous llinks on mass storage device DP4 are to be dumped. The dump is to begin at llink 49 decimal.

```
               1
1     8        6
-----------------------------------------------------------------
$     DUMP     DP4,LLINK-49,18
```

The following example indicates that 22 (octal) contiguous llinks on mass storage device DP4 are to be dumped. The dump is to begin at llink 61 (octal).

```
               1
1     8        6
-----------------------------------------------------------------
$     DUMP     DP4,LLINK/61/22
```

DUMP FILE CONTENT

The following $ DUMP statement format allows a Startup-created file to be dumped:

```
               1
1     8        6
-----------------------------------------------------------------
$     DUMP     ddd,fffff
```

where:

ddd - Logical device name of the mass storage device on which file fffff resides. This is an optional field.

fffff - Name of the Startup-created file to be dumped. If the System Master Catalog (SMC) is to be dumped, enter SMC. If the duplicate SMC is to be dumped, enter SMCDUP.

The following example indicates that the SSFILE file, which resides on device DP3, is to be dumped:

```
               1
1     8        6
-----------------------------------------------------------------
$     DUMP     DP3,SSFILE
```

PROCESSOR RELEASE AND ASSIGNMENT AT STARTUP

All physical processor numbers must be defined either by the $ MCT or the $ CIU statement in the $CONFIG section of the Startup job stream. PROC NUMBER switches on each processor's configuration panel must reflect the same number as specified on the $ MCT or the $ CIU statement.

Because any processor can handle interrupts, there are no restrictions on the release of processors. Any processor can be released during the bootload sequence if the processor does not respond to a Startup query during the rollcall function. As a result, the console message:

*PROCESSOR ON PORT p DOESN'T ANSWER CONNECT, YES TO CONTINUE

is issued (where p = number of the SCU port to which the processor is connected). The operator can respond with YES or NO:

YES - Release the processor and continue the bootload. The released processor is identified via the console message *PROCESSOR n RELEASED (where n = physical processor number). The processor subsequently can be restored (assigned) to the system via the console entry ASGNP n while the system is operational.

NO - Abort the Startup program. Correct the problem and reinitiate the bootload.

*

This question can be answered via the $ ANSWER PROCERROR/YES statement. The processor subsequently can be restored to the system via the ASGNP console entry. The question cannot be answered PROCERROR/NO. (Refer to "$CONFIG Section" in Section 5 for a complete description of the $ ANSWER statement.)

An additional System Startup technique affecting the status of any processor is the /OFF option on the $ MCT statement. This option permits definition of a processor that is not currently assigned, but which may be assigned (activated) in the future.

SECTION 4

INTEGRATED SOFTWARE

Integrated Software provides generalized services to the GCOS 8 Operating System and to a site's application programs. It consists of shared domains as well as storage space for buffers and tables.

While the content of the Integrated Software package varies depending on the site's system configuration and selected options, the following domains constitute Integrated Software (the related module name is in parentheses):

- After Journal Management Domain (.MAJAC, .MAJIO)
- Automatic Workstation Enable Domain (.MWCMI, .MWCMS)
- Before Journal Management Domain (.MBJDO)
- Buffer Management Domain (.MBMGT)
- Checkpoint Journal Manager Domain (.MCKJM)
- Concurrency Control Domain (.MCC00)
- Data Base Procedure Domain (.MDZPC)
- Data Management Control System Domain (.MDMCS)
- Global Data Management Domain (.MDMGL)
- Historical Log Domain (.MHLOG)
- Integrated Software Termination Domain (.MUT01)
- Integrated Software Trace Domain (.MUTTR)
- Integrated Software Trace and Load Map Display Domain (.MUTTD)
- Integrated Software Output Report Domain (.MUTOR)
- Integrity Management Domain (.MIM00)
- Protected File I/O Domain (.MPFIO)
- Session Control Domain (.MMC01)
- Shared Data Base Control System Domain (.MDBCS)
- Shared Fault Handler Domain (.MUTFH)
- TP8 Command Executive Domain (.MCX00)
- Tenant Management Domain (.MTM00)
- Transaction Queuer Support Domain (.MTQ00)
- Workstation Management Domain (.MWMAN)

This section discusses:

1. The procedure to load Integrated Software into real memory for execution

2. Various site-modifiable parameters for Integrated Software

3. Real memory requirements for Integrated Software

4. Syntax and usage of Integrated Software Shared Domain Loader (SDLD) directives

INSTALLING INTEGRATED SOFTWARE

Integrated Software is delivered to a user site on the first reel of the stacked save tape under the SYS_SOFTWARE file structure. The save tape content must be installed (restored) and loaded into memory for execution under the snumb SHARD immediately after the system is bootloaded. Refer to the Software Installation Bulletin for this Software Release for instructions to install Integrated Software.

LOADING INTEGRATED SOFTWARE FOR EXECUTION

Integrated Software is loaded into memory by the privileged slave program SHARD, which is spawned during System Startup. No special Startup procedure is required. The following message is issued to the system console to indicate that Integrated Software has been successfully loaded.

```
*S#SHARD ****** SI4.2 INTEGRATED SOFTWARE LOADED SUCCESSFULLY ******
```

If errors were found, the following message is issued to the system console:

```
*S#SHARD ****** SI4.2 INTEGRATED SOFTWARE STATUS ******
*S#SHARD SHARED NAME= SYS_     LOADED SUCCESSFULLY
*S#SHARD # OF FATAL ERRORS - ff    # OF NON-FATAL ERRORS - nn
*S#SHARD # OF BUFFER POOL COMMANDS - mm     # OF B-F ERRORS - bb
*S#SHARD MAXIMUM WIRED PAGES ALLOWED(dddd) - ccc
*S#SHARD BACKDOOR LOAD-MAP - YES
```

where:

ff - Number of fatal errors.

nn - Number of non-fatal errors. The site should determine whether or not to continue processing after reviewing all of these errors.

mm - Number of buffer pool commands.

bb - Number of errors in the buffer pool commands.

ccc - Maximum number of pages that S#SHARD can wire. Default is 511. The site can change this via ASSIGN_SITE_PARAMETER -SET_MAX_WIRED_PAGES.

dddd - DEFAULT or REQUESTED.
REQUESTED will be used if ASSIGN_SITE_PARAMETER -SET_MAX_WIRED_PAGES is specified. Otherwise, DEFAULT will be used.

Integrated Software must be executing before any dependent processes can execute. The software remains in execution as long as GCOS 8 software is active or until Integrated Software aborts. In addition, Integrated Software automatically restarts each time GCOS 8 software is rebooted.

While the Shared Domain Loader (SDLD) automatically loads the Integrated Software program into execution, a spawn file can also call the program into execution (the spawn file must be named SHARD and must be stored under OPNSUTIL). Several seconds may elapse from the time of the *SRT message before the software is actually ready for execution. The issuance of the "SI4.2 INTEGRATED SOFTWARE LOADED SUCCESSFULLY" information message indicates that Integrated Software is available for reference.

Only one version of Integrated Software can be active at any single time. Consequently, special attention must be given to scheduling so that all system users (i.e., development, production, and test organizations) are aware of which version is active.

Integrated Software is loaded and placed in execution by inclusion of the following SPAWN directive in the job stream supplied to the System Spawn Facility (SYSPN) program:

```
SPAWN -SNUMB SHARD -PATHNAME SYS_SOFTWARE/yyyy/JCL/SHARD.SPWN
```

where: yyyy is the Software Release identifier (i.e., 3000)

SYSPN is automatically spawned during the system boot process. Refer to the GCOS 8 OS System Operating Techniques manual for further details.

To load Integrated Software from a spawn file:

1. Edit a $ SELECTD SYS_SOFTWARE/yyyy/JCL/SHARD.SPWN statement onto the OPNSUTIL/SHARD file (where yyyy = the Software Release identifier).

2. Enter SPAWN SHARD via the system console.

The following job stream is called into execution from the SHARD.SPWN file:

```
               1
1      8       6                                                    NOTES
------------------------------------------------------------------------------
$      IDENT   nnnnn                                                  1
$      RUN     RUFILE=SYSTEM,RUNAME=.VSDLD,OPTION=DUMP,SSDUMP         2
$      PRIVITY                                                        3
$      LIMITS  ,,,,,5K
$      RESOURC VSPACE=3800K,RSPACE=100K                               4
$      DATA    BP,,COPY                                               5
CREATE_BUFFER_POOL STD..POOL &
  -BUFFER_SIZE 2048 &
  -MIN_NUMBER_BUFFERS 0 &
  -MAX_NUMBER_BUFFERS 100 &
  -BI_SUPPRESSION_ENTRIES 0 &
  -TYPE_PUBLIC &
  -POOL_ALLOCATION IMMEDIATE
-- --------- END STANDARD BP ALLOCATION ------------
$      ENDCOPY
$      DATA    I*
START_DEFAULTS
-- NOTE - THESE ARE THE SDLD DEFAULT DIRECTIVES NEEDED TO
--        LOAD INTEGRATED-SOFTWARE
--
ALLOCATE -LIBRARY Q. -PATHNAME SYS_SOFWARE/<sti>/SYS -OPTIONAL     6,10
-- **** THE 'Q.' LIBRARY IS THE PRIMARY SEARCH FILE ****
-- ALLOCATE -LIBRARY .Q -PATHNAME USER-BACKUP/LIBRARY -OPTIONAL     7
-- **** THE '.Q' LIBRARY IS THE SECONDARY SEARCH FILE ****
.
.         (See NOTES 8 through 10)
.
END_DEFAULTS
ALLOCATE -DIRECTIVES -PATHNAME SYS_SOFTWARE/yyyy/DATA/SHARD.CHGS     10
```

NOTES:

1. Site-defined content (nnnnn) for the $ IDENT statement; replaces the default user identification provided in OPNSUTIL. If this statement is not included, system software provides a default $ IDENT statement.

2. This $ RUN statement points to a run unit that loads Integrated Software into memory for execution.

3. Required to execute this process.

4. Specifies the defaults for allocation of virtual memory space (VSPACE) and for initial allocation of real memory (RSPACE) for Integrated Software. These values can be increased to meet site requirements.

5. The $ DATA BP file contains SDLD directives to explicitly define a buffer pool (i.e., CREATE_BUFFER_POOL and ASSIGN_PRIVATE_BP_FILE directives). An ALLOCATE -BPDIRECTIVES directive may be used instead of the $ DATA BP file.

6. Identifies the primary search library (Q.) from which Integrated Software domains are loaded.

7. Identifies the secondary search library (.Q) from which Integrated Software domains are loaded if the primary search library is unavailable. To use, remove the initial characters ("-- ") and supply a pathname.

8. The following Integrated Software domains reside on the SHARD.SPWN file:

```
INCLUDE_DOMAIN .MUTTR -LIBRARY Q. -REQUIRED   Trace
INCLUDE_DOMAIN .MUTFH -LIBRARY Q. -REQUIRED   Shared Fault Handler
INCLUDE_DOMAIN .MUTOR -LIBRARY Q. -REQUIRED   Output Report
INCLUDE_DOMAIN .MUTTD -LIBRARY Q. -REQUIRED   Trace Display
INCLUDE_DOMAIN .MUT01 -LIBRARY Q. -REQUIRED   Termination Wrapup
INCLUDE_DOMAIN .MPFIO -LIBRARY Q. -REQUIRED   Protected File I/O
INCLUDE_DOMAIN .MBJDO -LIBRARY Q. -REQUIRED   Before Journal Management
INCLUDE_DOMAIN .MIM00 -LIBRARY Q. -REQUIRED   Integrity Management
INCLUDE_DOMAIN .MCC00 -LIBRARY Q. -REQUIRED   Concurrency Control
INCLUDE_DOMAIN .MHLOG -LIBRARY Q. -REQUIRED   Historical Log
INCLUDE_DOMAIN .MCKJM -LIBRARY Q. -REQUIRED   Checkpoint Journal Management
INCLUDE_DOMAIN .MBMGT -LIBRARY Q. -REQUIRED   Buffer Management
INCLUDE_DOMAIN .MDMGL -LIBRARY Q. -REQUIRED   Global Data Management
INCLUDE_DOMAIN .MAJAC -LIBRARY Q. -REQUIRED   After Journal Access and
                                              Reconstruction
INCLUDE_DOMAIN .MAJIO -LIBRARY Q. -REQUIRED   After Journal Shared I/O
                                              Support
INCLUDE_DOMAIN .MDMCS -LIBRARY Q. -REQUIRED   Data Management Control
                                              System
INCLUDE_DOMAIN .MDZPC -LIBRARY Q. -REQUIRED   Data Base Procedures
INCLUDE_DOMAIN .MMC01 -LIBRARY Q. -REQUIRED   Session Control
INCLUDE_DOMAIN .MWCMI -LIBRARY Q. -REQUIRED   Automatic Enable
INCLUDE_DOMAIN .MTM00 -LIBRARY Q. -REQUIRED   Tenant Management
INCLUDE_DOMAIN .MWMAN -LIBRARY Q. -REQUIRED   Workstation Management
```

9. The following default parameters are assembled in UTOR for Integrated Software and reside on the SHARD.SPWN file. For more information on these default parameters, refer to "Integrated Software Default Parameter Modification", later in this section, and to the GCOS 8 OS Protected Files Administrator's Guide and the GCOS 8 OS Integrity Control manual.

```
ASSIGN_SITE_PARAMETER -AJ_USERID USERID$PASSWD
ASSIGN_SITE_PARAMETER -AJ_SAVE_RESTORE_** FILE SYS_SOFTWARE/<sti>/SYS
                        (See NOTE 10)
ASSIGN_SITE_PARAMETER -MAX_NUMBER_JOURNALS 20
ASSIGN_SITE_PARAMETER -JOURNAL_DIRECTORY_CISIZE 2
ASSIGN_SITE_PARAMETER -AJ_NFIELDS 8
ASSIGN_SITE_PARAMETER -AJ_IDENT "ACCOUNT,GUESS-WHO"
ASSIGN_SITE_PARAMETER -BJ_CISIZE 2816
ASSIGN_SITE_PARAMETER -BJ_MIN_CI 1000
ASSIGN_SITE_PARAMETER -BJ_MAX_CI 0  --  ZERO MEANS UNLIMITED
ASSIGN_SITE_PARAMETER -BJ_HEADER_WRITE_PERIOD 30
ASSIGN_SITE_PARAMETER -RS_DUP NO
ASSIGN_SITE_PARAMETER -RS_DEVICE 000000  (See NOTE 10)
```

```
ASSIGN_SITE_PARAMETER -RS_DUP_DEVICE 000000  (See NOTE 10)
ASSIGN_SITE_PARAMETER -CC_PROCESS_PAGES 25900
ASSIGN_SITE_PARAMETER -BM_PARAMETER_CHECK YES
ASSIGN_SITE_PARAMETER -BM_COMPRESSED_JOURNAL YES
ASSIGN_SITE_PARAMETER -BI_SUPPRESSION_ENTRIES 0
ASSIGN_SITE_PARAMETER -MOVE_MODE_MAX_BUF 10
ASSIGN_SITE_PARAMETER -DEFAULT_BP_BUFFERS 40
ASSIGN_SITE_PARAMETER -PFIO_OPEN_BUFFERS 2
ASSIGN_SITE_PARAMETER -PFIO_INTERNAL_TRACE NO
ASSIGN_SITE_PARAMETER -WCC_1_STAR_FILE  SYS_SOFTWARE/yyyy/DATA/WCC-1STR
                      (See NOTE 10)
ASSIGN_SITE_PARAMETER -WCC_6_STAR_FILE  SYS_SOFTWARE/yyyy/DATA/WCC-6STR
                      (See NOTE 10)
ASSIGN_SITE_PARAMETER -WD_FILE          SYS_SOFTWARE/yyyy/DATA/WS-DB
                      (See NOTE 10)
ASSIGN_SITE_PARAMETER -SC_VIRTUAL_BUFFER_SIZE 1000
ASSIGN_SITE_PARAMETER -SC_INTERNAL_TRACE NO
ASSIGN_SITE_PARAMETER -TM_INTERNAL_TRACE NO
ASSIGN_SITE_PARAMETER -WM_INTERNAL_TRACE NO
ASSIGN_SITE_PARAMETER -HISTORICAL_LOG NO
ASSIGN_SITE_PARAMETER -HL_CISIZE 2048
ASSIGN_SITE_PARAMETER -HL_NUM_CI 1000
ASSIGN_SITE_PARAMETER -HL_DUP NO
ASSIGN_SITE_PARAMETER -HL_DEVICE 000000  (See NOTE 10)
ASSIGN_SITE_PARAMETER -HL_DUP_DEVICE 000000  (See NOTE 10)
ASSIGN_SITE-PARAMETER -CKPT_IDENT "ACCOUNT,GUESS-WHO"
ASSIGN_SITE_PARAMETER -CKPT_USERID USERID$PASSWD
ASSIGN_SITE_PARAMETER -CKPT_FILE_SIZE 40000
ASSIGN_SITE_PARAMETER -CKPT_FILE_DEVICE 000000  (See NOTE 10)
ASSIGN_SITE_PARAMETER -CKPT_DUMP_FILL 60
ASSIGN_SITE_PARAMETER -CKPT_DUMP_TIME 0  -- NO DEFAULT
ASSIGN_SITE_PARAMETER -CKPT_COLLECT ALL
ASSIGN_SITE_PARAMETER -CKPT_FILE_CONTROL NONE
ASSIGN_SITE_PARAMETER -CKPT_MAX_ACTIVITIES 200
ASSIGN_SITE_PARAMETER -CKPT_FILE_RMVBL 000000,000000   --  ANY DEVICE,PACK
ASSIGN_SITE_PARAMETER -ISSTAT_ENABLE NO
ASSIGN_SITE_PARAMETER -ISSTAT_ABNORMAL_TERM NO
ASSIGN_SITE_PARAMETER -ISSTAT_FILE_CODE J*
ASSIGN_SITE_PARAMETER -ISSTAT_REPORT_CODE 74
ASSIGN_SITE_PARAMETER -ISSTAT_ALL_SYS_LEVEL NO -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_ALL_PROC_LEVEL YES -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_BM_SYS_LEVEL NO -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_DM_SYS_LEVEL NO -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_AJ_SYS_LEVEL NO -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_CC_SYS_LEVEL NO -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_HL_SYS_LEVEL NO -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_PFIO_SYS_LEVEL NO -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_BM_PROC_LEVEL YES -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_DM_PROC_LEVEL YES -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_BJ_PROC_LEVEL YES -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_AJ_PROC_LEVEL YES -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_PFIO_PROC_LEVEL YES -PRINT_ZERO NO
ASSIGN_SITE_PARAMETER -ISSTAT_IDS2_PROC_LEVEL YES -PRINT_ZERO NO
```

```
ASSIGN_SITE_PARAMETER -ISSTAT_BATCH_PROCESSES YES
ASSIGN_SITE_PARAMETER -ISSTAT_WS_PROCESSES NO
ASSIGN_SITE_PARAMETER -ISSTAT_TP_PROCESSES NO
ASSIGN_SITE_PARAMETER -ISSTAT_TDS_PROCESSES NO
ASSIGN_SITE_PARAMETER -ISSTAT_TS8_PROCESSES NO
ASSIGN_SITE_PARAMETER -CKPT_FILE_DUP NO
ASSIGN_SITE_PARAMETER -CKPT_FILE_DUP_DEVICE 000000   -- NO DEFAULT
ASSIGN_SITE_PARAMETER -HL_IGNORE_ERRORS YES
ASSIGN_SITE_PARAMETER -HL_COMMITMENT_ENTRY YES
ASSIGN_SITE_PARAMETER -HL_FILE_OPEN_ENTRY YES
ASSIGN_SITE_PARAMETER -HL_FILE_CLOSE_ENTRY YES
ASSIGN_SITE_PARAMETER -TRACE_BUFFER_SIZE 1  -- ONE 4K BUFFER
```

10. 000000 (Six zeros) - Any device
    yyyy - Current Software Release
    sti - Software Technical Identifier (see the appropriate SRB for the STI name) *

INTEGRATED SOFTWARE DEFAULT PARAMETER MODIFICATION

Integrated Software is assembled with parameters that provide controls over specific processing conditions (see note 9 above for a list of parameters). Assembled parameter values are default values, which are defined in the following files (where yyyy = software release identifier, such as 3000) and described below.

- o SYS_SOFTWARE/yyyy/JCL/SHARD.SPWN
- o SYS_SOFTWARE/yyyy/DATA/SHARD.CHGS

Each default value can be modified to meet installation processing requirements. It is recommended that any modification be applied to the SHARD.CHGS file. This approach leaves the default parameters unchanged in the SHARD.SPWN file, which provides a point of reference for determining default values.

Modification of a parameter in either the SHARD.SPWN or SHARD.CHGS file applies changes to Integrated Software spawned either from the OPNSUTIL/SHARD file or during System Startup.

A two-step procedure can be used to apply parameter modifications to either the SHARD.SPWN file or the SHARD.CHGS file. First, use the time sharing system CONVER subsystem to copy the file (e.g., SHARD.CHGS) into the current file; then, use the time sharing system EDIT command to modify the value in question.

The following usage rules apply to SDLD directives.

1. If a syntax error is encountered in the parameter definition, either the default is applied or - if a previous assignment was made for the parameter - the existing value is used.

2. If an ASSIGN_SITE_PARAMETER directive is duplicated, the last iteration encountered is applied.

3. A literal string is permitted only in an argument field.

4. Quotation marks (") serve as beginning and ending delimiters of a literal string. Use of these delimiters permits embedded blanks within the string.

5. If a quotation mark is embedded within a literal string, it must be immediately preceded by another quotation mark. For example, "a""b" is interpreted as a"b.

6. If the directive is misspelled (e.g., ASSIGN-SITE-PARAMETER), SDLD cannot recognize the directive as valid, which causes SHARD to abort.

Site Parameters

This section describes the valid parameters that can be used with the SDLD directive ASSIGN_SITE_PARAMETER.

SITE PARAMETERS FOR ARCHIVED BEFORE/AFTER/ACCOUNTING JOURNALING

Maximum Number of Journals

The -MAX_NUMBER_JOURNALS parameter is used to specify the maximum number of journals:

```
ASSIGN_SITE_PARAMETER -MAX_NUMBER_JOURNALS n
```

where:

n - Maximum number of journals. This value may range from 1-20. The default value is 10.

Journal Directory Control Interval Size

The -JOURNAL_DIRECTORY_CISIZE parameter is used to specify the journal directory control interval size:

```
ASSIGN_SITE_PARAMETER -JOURNAL_DIRECTORY_CISIZE c
```

where:

c - Control interval size, specified in K (while other control interval size parameters are specified in bytes). This value may range from 1-16 (K). The default value is 2 (K).

SITE PARAMETERS FOR AFTER JOURNALING

After Journaling is a means by which data bases can be protected against media failure (usually disk) and erroneous data base modifications. Any data base file created or modified with the FMS RDERR/JOURNAL/ option is protected by After Journaling. The following parameters instruct the After Journaling software with the site's selection of options.

After Journal $ IDENT Statement

A $ IDENT statement is assembled into a job stream that is spawned by the After Journal domain to interface with the File Management Supervisor (FMS) save and/or restore process. Columns 16-80 must be redefined by each site via the -AJ_IDENT parameter:

```
ASSIGN_SITE_PARAMETER -AJ_IDENT x
```

where:

x - 1-65 alphanumeric characters to be placed in columns 16 to 80 of the $ IDENT statement. Blank is a valid character in this field, provided that the field is enclosed with quotes (e.g., "b c"). ACCT#,GUESS-WHO is used as the default.

After Journal $ USERID Statement

A $ USERID statement is assembled into a job stream that is spawned by the After Journal domain to interface with the File Management Supervisor (FMS) save and/or restore process. Columns 16-80 must be redefined by each site via the -AJ_USERID parameter:

```
ASSIGN_SITE_PARAMETER -AJ_USERID y
```

where:

y - 1-65 alphanumeric characters to be placed in columns 16 to 80 of the $ USERID statement. USERID$PASSWD is used as the default.

After Journal Save/Restore ** File Pathname

A default pathname is provided for this file. There may be some circumstances, however, under which pathname modification is desirable. For example, before installing a new version of a file, the system administrator may wish to test file content; by modifying the pathname of the new (test) version, the file can be accessed without interrupting system processing that uses the existing version. The pathname can be modified via the -AJ_SAVE_RESTORE_**_FILE parameter:

```
ASSIGN_SITE_PARAMETER -AJ_SAVE_RESTORE_**_FILE p
```

where:

p - File pathname. SYS_SOFTWARE/W2SI00842000/SYS is the default.

SITE PARAMETERS FOR BEFORE JOURNALING

Before Journaling provides a means by which files are protected from incomplete updating due to process abort or system failure. A Before Journal is created and maintained for each process accessing a file for which the FMS ABORT/ROLLBACK/ file option has been specified. The following parameters instruct the Before Journaling software with the site's selection of options.

Before Journal Control Interval Size

The -BJ_CISIZE parameter is used to specify the before journal control interval size:

```
ASSIGN_SITE_PARAMETER -BJ_CISIZE c
```

where:

c - Control interval size, in bytes, ranging from 256-66536. The default is 2816 bytes (704 words) which accommodates two 320-word control intervals for a file with ABORT/ROLLBACK protection and supporting header information.

Minimum Number Of Before Journal CIs Per Commitment Unit

The -BJ_MIN_CI parameter is used to specify the minimum number of before journal control intervals per commitment unit:

```
ASSIGN_SITE_PARAMETER -BJ_MIN_CI n
```

where:

n - Minimum number of before journal control intervals per commitment unit, ranging from 5-100000. The default is 1000.

Maximum Number Of Before Journal CIs Per Commitment Unit

The -BJ_MAX_CI parameter is used to specify the maximum number of before journal control intervals per commitment unit:

```
ASSIGN_SITE_PARAMETER -BJ_MAX_CI n
```

where:

n - Maximum number of before journal control intervals per commitment unit, ranging from 5-100000. The default is unlimited.

Before Journal Control Interval Writes Between Header Writes

A header record containing control information is periodically written to the before journal. The header record occupies one control interval and is always the first control interval in the file. The number of before journal control intervals written between header records can be modified via the -BJ_HEADER_WRITE_PERIOD parameter:

```
ASSIGN_SITE_PARAMETER -BJ_HEADER_WRITE_PERIOD n
```

where:

n - Number of before journal control intervals written between header records, ranging from 1-1000. The default value is 30. A larger value of n reduces the number of run-time I/O operations. However, a longer recovery period is required in the event of a system failure.

FMS Protection For Before Journal Recovery Structure

The -RS_DUP parameter is used to specify whether the RDERR/DUP/ FMS protection option should be applied to the recovery structure (before journals and PAT image files):

```
ASSIGN_SITE_PARAMETER -RS_DUP x
```

where:

x - 0 or NO or OFF applies RDERR/NONE/. This is the default.

1 or YES or ON applies RDERR/DUP/.

Before Journal Device Specification

The -RS_DEVICE parameter is used to assign a specific device or devices on which all before journals and PAT image files will be created:

```
ASSIGN_SITE_PARAMETER -RS_DEVICE ddd,dde,ddf,...ddm
```

where:

ddd,dde,ddf,...ddm - Device codes. 1-10 may be specified. If a specific device is not assigned, FMS will create the before journals and PAT image files on a device of its choosing. If more than one device is specified, the before journals and PAT image files will be placed on the devices in a round robin manner. Note that these devices <u>must</u> be configured as permanent FMS devices.

Before Journal Duplicate Device Specification

The -RS_DUP_DEVICE parameter assigns a specific device or devices (up to 10) on which all duplicate before journals and PAT image files will be created:

```
ASSIGN_SITE_PARAMETER -RS_DUP_DEVICE ddd,dde,ddf,...ddm
```

where:

ddd,dde,ddf,...ddm - Device codes. 1-10 may be specified. If a specific device is not assigned, FMS will create the duplicate before journals and PAT image files on a device of its choosing. If more than one device is specified, the duplicate files will be placed on the devices in a round robin manner. Note that these devices must be configured as permanent FMS devices. They must be different physical devices from the devices specified with the -RS_DEVICE parameter.

SITE PARAMETERS FOR CHECKPOINT JOURNALING

Checkpoint Journaling $ IDENT Statement

A $ IDENT statement is assembled into a job stream that is spawned by the Checkpoint Journal Manager domain to dump the checkpoint journal to the checkpoint journal archive (i.e., to tape). Columns 16-80 must be redefined by each site via the -CKPT_IDENT parameter:

```
ASSIGN_SITE_PARAMETER -CKPT_IDENT x
```

where:

x - 1-65 alphanumeric characters to be placed in columns 16 to 80 of the $ IDENT statement. Blank is a valid character in this field, provided that the field is enclosed with quotes (e.g., "b c"). ACCT#,GUESS-WHO is used as the default.

Checkpoint Journaling $ USERID Statement

A $ USERID statement is assembled into a job stream that is spawned by the Checkpoint Journal Manager domain to dump the checkpoint journal to the checkpoint journal archive (i.e., to tape). Columns 16-80 must be redefined by each site via the -CKPT_USERID parameter:

```
ASSIGN_SITE_PARAMETER -CKPT_USERID y
```

where:

y - 1-65 alphanumeric characters to be placed in columns 16 to 80 of the $ USERID statement. USERID$PASSWD is used as the default.

Default Checkpoint File For Checkpoint Journaling

The -CKPT_FILE_CONTROL parameter is considered the master checkpoint Journal parameter. It determines the type of file on which process snapshots are collected, unless an activity statement option is used to override this value. The file used for process snapshot collection may be either the system-wide checkpoint journal or a user-specific checkpoint file (e.g., QX file used for I-D-S/I and Indexed Sequential Processor (ISP) data base management programs). Values for this parameter are NONE, USER, JOURNAL, and SYSTEM.

The -CKPT_FILE_CONTROL parameter is specified as follows:

```
ASSIGN_SITE_PARAMETER -CKPT_FILE_CONTROL {NONE|USER|JOURNAL|SYSTEM}
```

where:

NONE - The checkpoint journal is not used. No process snapshots are collected on the checkpoint journal. User-specific checkpoint files may be used on a per-process basis to collect the process snapshot from the most recent commitment. This is the default.

USER - Process snapshots from the most recent commitment are maintained on user-specific checkpoint files. Activity statement options CHKPT/{ARCH|JRNL|DEFER} can be used to override the effects of this SDLD directive on a per-process basis.

JOURNAL - Process snapshots are maintained on the checkpoint journal, which is a cataloged, permanent file under the user identifier SYS_RECOVERY. Activity statement option CHKPT/USER/ can be used to override the effects of this SDLD directive on a per-process basis.

SYSTEM - Produces the same results as specifying JOURNAL (i.e., process snapshots are maintained on the checkpoint journal). In addition, SYSTEM imposes a security restriction by preventing the use of the CHKPT/USER/ activity statement option to override the collection of checkpoints to the checkpoint journal (i.e., use of the activity option CHKPT/USER is invalid).

Checkpoint Journaling Collection Of Historical Process Snapshots

The -CKPT_COLLECT parameter specifies the extent of checkpoint information to be collected when -CKPT_FILE_CONTROL JOURNAL or -CKPT_FILE_CONTROL SYSTEM has been specified:

```
ASSIGN_SITE_PARAMETER -CKPT_COLLECT {ALL|NONE}
```

where:

ALL - Specifies that all process snapshots for processes using protected files are to be collected. In addition, activity start and termination are registered for these processes. ALL is the default. For processes that do not use protected files, no checkpoint information is collected unless use of the checkpoint journal is requested via activity statement options, or required via -CKPT_FILE_CONTROL SYSTEM. Collection of all process snapshots can be selectively suppressed for an individual process using activity statement options. (Refer to the GCOS 8 OS Integrity Control manual for more information.)

NONE - Specifies that no process snapshots are to be collected for processes using protected files (i.e., only activity start and termination are registered). The two most recent snapshots are maintained on the checkpoint journal, but deleted if the activity terminates normally. The most recent checkpoint is maintained if the process terminates abnormally. For processes that do not use protected files, no checkpoint information is collected unless use of the checkpoint journal is requested via activity statement options, or required via -CKPT_FILE_CONTROL SYSTEM. Collection of all process snapshots can be selectively enabled for an individual process using activity statement options. (Refer to the GCOS 8 OS Integrity Control manual for more information.)

Checkpoint Journal Size

The -CKPT_FILE_SIZE parameter specifies the maximum size of the checkpoint journal:

```
ASSIGN_SITE_PARAMETER -CKPT_FILE_SIZE s
```

where:

s - Maximum size (in llinks) of the checkpoint journal. The valid range is 1000-131071 llinks. The default value is 40000.

Maximum Number Of Activities Using Checkpoint Journal

The -CKPT_MAX_ACTIVITIES parameter defines the maximum number of activities that will write process snapshots to the checkpoint journal:

```
ASSIGN_SITE_PARAMETER -CKPT_MAX_ACTIVITIES n
```

where:

n - Maximum number of activities. The range for this parameter is 5-2200, with a default value of 200. This value is used in a file control block to reserve space for the process snapshots. It can be determined by using the following formula:

nnnn = (Number of activities taking commitments)
* (No. of times activities are run before journal is dumped)

For example, if an installation has 20 activities taking commitments and each activity is run 5 times before the checkpoint journal is dumped to the checkpoint journal archive (i.e.,tape), then the -CKPT_MAX_ACTIVITIES parameter should be assigned a value of 100.

Checkpoint Journal Dump Frequency (Percent Filled)

The -CKPT_DUMP_FILL parameter specifies the percentage full requirement for dumping the checkpoint journal to archive (i.e., tape):

ASSIGN_SITE_PARAMETER -CKPT_DUMP_FILL p

where:

p - Percent (0-100). The default value is 60 percent. For example, if this value is 70, the checkpoint journal is dumped to archive when it has been filled to 70 percent of its capacity.

If this parameter is used in conjuntion with the -CKPT_DUMP_TIME parameter, then whichever condition is met first determines when the journal is dumped.

Checkpoint Journal Dump Frequency (Time)

The -CKPT_DUMP_TIME parameter indicates a specific time interval for dumping the checkpoint journal to the checkpoint archive (i.e., tape):

ASSIGN_SITE_PARAMETER -CKPT_DUMP_TIME t

where:

t - Time interval (0-1440) minutes. There is no default; a value must be supplied if the parameter is used.

Duplicate Checkpoint Journal

The -CKPT_FILE_DUP parameter is used to specify whether the protection option RDERR/DUP/ is applied to the checkpoint journal:

```
ASSIGN_SITE_PARAMETER -CKPT_FILE_DUP x
```

where:

x - 0 or NO or OFF applies RDERR/NONE/. This is the default.

1 or YES or ON applies RDERR/DUP/. This FMS protection option maintains a duplicate checkpoint journal, thus protecting against loss of the primary checkpoint journal.

Checkpoint Journal Device Specification

The -CKPT_FILE_DEVICE parameter is used to assign a specific device on which to create the checkpoint journal:

```
ASSIGN_SITE_PARAMETER -CKPT_FILE_DEVICE d
```

where:

d - Device code. If a specific device is not assigned, FMS will create the checkpoint journal file on a device of its choosing.

Duplicate Checkpoint Journal Device Specification

The -CKPT_FILE_DUP_DEVICE parameter is used to assign a specific device on which to create the duplicate checkpoint journal:

```
ASSIGN_SITE_PARAMETER -CKPT_FILE_DUP_DEVICE d
```

where:

d - Device code. If a specific device is not assigned, FMS will create the duplicate checkpoint journal file on a device of its choosing.

Removable Device For Checkpoint Journal

The -CKPT_FILE_RMVBL parameter is used to assign a removable device on which to create the checkpoint journal. If this parameter is not used, the file is created on PERM device.

```
ASSIGN_SITE_PARAMETER -CKPT_FILE_RMVBL t,x
```

where:

t - Device type. Valid values are 400PK, 450PK, 500PK, 501PK, C1APK, and C1BPK.

x - Pack serial number. Five alphanumeric characters.

SITE PARAMETERS FOR HISTORICAL LOG

The Historical Log is a chronological record of activity for processes accessing protected files and processes that use journals.

Create Historical Log

The -HISTORICAL_LOG parameter specifies whether or not the historical log is to be created:

```
ASSIGN_SITE_PARAMETER -HISTORICAL_LOG x
```

where:

x - 0 or NO or OFF. The historical log will not be created (default).

1 or YES or ON. The historical log will be created. Specifying YES is recommended if an installation is processing with protected files and is using the checkpoint journal service to collect all process snapshots during the life of a process.

Number Of Historical Log Control Intervals

The -HL_NUM_CI parameter defines the number of control intervals for the historical log:

```
ASSIGN_SITE_PARAMETER -HL_NUM_CI n
```

where:

n - Number of control intervals. The valid range for this value is 5-100000. The default value is 1000. Multiplying the value of this parameter by the value of the control interval size parameter (-HL_CISIZE) determines the size of the historical log.

Historical Log Control Interval Size

The -HL_CISIZE parameter defines the size (in bytes) of the control interval for the historical log:

ASSIGN_SITE_PARAMETER -HL_CISIZE c

where:

c - Control interval size, in bytes. The valid range is 256-65536. The default value is 2048 bytes (512 words).

Ignore Errors When Writing To Historical Log

The -HL_IGNORE_ERRORS parameter specifies whether errors that occur when a process is writing to the historical log are ignored:

ASSIGN-SITE-PARAMETER -HL_IGNORE_ERRORS x

where:

x - 0 or NO or OFF will cause processes to abort when they encounter an error writing to the historical log.

1 or YES or ON (default) will cause processes to ignore errors that occur when writing to the historical log.

Duplicate Historical Log

The -HL_DUP parameter specifies whether a duplicate historical log will be created:

ASSIGN_SITE_PARAMETER -HL_DUP x

where:

x - 0 or NO or OFF. The duplicate historical log <u>will not</u> be created (default).

1 or YES or ON. The duplicate historical log <u>will</u> be created.

Historical Log Device

The -HL_DEVICE parameter is used to define a specific device upon which to create the historical log:

ASSIGN_SITE_PARAMETER -HL_DEVICE ddd

where:

ddd - Device code. Any configured three-character logical device name is a valid argument. If a specific device is not assigned, FMS will create the historical log file on a device of its choosing.

Historical Log Duplicate Device

The -HL_DUP_DEVICE parameter defines a specific device upon which to create the duplicate historical log, if one was specified.

ASSIGN_SITE_PARAMETER -HL_DUP_DEVICE ddd

where:

ddd - Device code. Any configured three-character logical device name is a valid argument. If a specific device is not assigned, FMS will create the historical log file on a device of its choosing.

Historical Log Commitment Entry

The -HL_COMMITMENT_ENTRY parameter determines whether a historical log entry is made each time a commitment is taken:

ASSIGN_SITE_PARAMETER -HL_COMMITMENT_ENTRY a

where:

a - 0 or NO or OFF prevents a historical log entry from being made each time a commitment is taken.

1 or YES or ON (default) causes a historical log entry to be made each time a commitment is taken.

Historical Log Entry For Each Protected File Open

The -HL_FILE_OPEN_ENTRY parameter determines whether a historical log entry is made each time a protected file is opened:

ASSIGN_SITE_PARAMETER -HL_FILE_OPEN_ENTRY a

where:

a - 0 or NO or OFF prevents a historical log entry from being made each time a protected file is opened.

1 or YES or ON (default) causes a historical log entry to be made each time a protected file is opened.

Historical Log Entry For Each Protected File Close

The -HL_FILE_CLOSE_ENTRY parameter determines whether a historical log entry is made each time a protected file is closed:

ASSIGN_SITE_PARAMETER -HL_FILE_CLOSE_ENTRY a

where:

a - 0 or NO or OFF prevents a historical log entry from being made each time a protected file is closed.

1 or YES or ON (default) causes a historical log entry to be made each time a protected file is closed.

SITE PARAMETER FOR CONCURRENCY CONTROL

Maximum Reservations Retained For One Process

The -CC_PROCESS_PAGES parameter specifies the maximum number of concurrency control reservations (i.e., data base pages and files) retained for a single process:

ASSIGN_SITE_PARAMETER -CC_PROCESS_PAGES r

where:

r - Maximum number of reservations per process. The valid range for this value is 1-25900 reservations. The default is 25900. (One reservation requires five words of memory for each record and five words for each file.)

SITE PARAMETERS FOR BUFFERING

Maximum Number Of Buffers In Default Buffer Pools

The -DEFAULT_BP_BUFFERS parameter specifies the maximum number of buffers that can exist in each buffer pool:

ASSIGN_SITE_PARAMETER -DEFAULT_BP_BUFFERS b

where:

b - Maximum number of buffers in each buffer pool. The value can range from 1 to 512. By default, 40 buffers exist in each buffer pool.

This parameter does not necessarily specify the actual number of buffers assigned to each buffer pool. For example, if the buffer pool limit is 20 and a process requests 12 buffers, then 12 buffers are assigned. However, if the buffer pool limit is 20 and a process requests 24 buffers, an error status will be returned.

Note that the number of buffers in a buffer pool is reduced for each open file by the value of the PFIO_OPEN_BUFFERS parameter. For example, if the DEFAULT_BP_BUFFERS value is 40 and the PFIO_OPEN_BUFFERS value is 10, 30 buffers are available from the buffer pool after one file has been opened.

Buffers Allocated To Protected Files

The -PFIO_OPEN_BUFFERS parameter specifies the number of buffers allocated to each protected file at file-open time in a batch or TPE processing environment:

```
ASSIGN_SITE_PARAMETER -PFIO_OPEN_BUFFERS b
```

where:

b - Number of buffers allocated to each protected file at file-open time. This value can range from 1 to 512. The default is 2.

Buffer Management Parameter And Descriptor Consistency Checks

The -BM_PARAMETER_CHECK parameter specifies whether buffer management parameter and descriptor consistency checks are to be performed:

```
ASSIGN_SITE_PARAMETER -BM_PARAMETER_CHECK c
```

where:

c - 0 or NO or OFF causes the consistency checks to be bypassed, so that Buffer Management will execute faster.

1 or YES or ON (default) causes the consistency checks to be performed.

If a problem arises while the consistency check is bypassed, reinstate the check; then rerun the process to verify parameters and descriptors before attempting to analyze the problem.

Compressed And Full Control Interval Journalization

The -BM_COMPRESSED_JOURNAL parameter provides a choice between compressed journalization and full control interval journalization:

```
ASSIGN_SITE_PARAMETER -BM_COMPRESSED_JOURNAL c
```

where:

c - 0 or NO or OFF causes full control interval journalization, so that entire control intervals (i.e., the modified record(s) and all unmodified records for that control interval) are written to the appropriate journal(s).

1 or YES or ON (default) causes compressed journalization, so that when modifying a file with ABORT/ARCHIVE/, ABORT/ROLLBACK/, and/or RDERR/JOURNAL/ protection, only the modified control interval portions of the file are journalized. (An algorithm divides each file control interval into partitions containing a minimum of 64 words per partition. A maximum of 16 partitions is created.)

Number Of Modifications Before Control Interval Written

The -AJ_NFIELDS parameter defines the maximum number of times a specific control interval can be modified before the entire control interval is written to the after journal:

```
ASSIGN_SITE_PARAMETER -AJ_NFIELDS f
```

where:

f - Maximum number of times a specific control interval can be modified before the entire control interval is written to the after journal. The valid range for this value is 1-16. The default value is 8.

Minimize Before Image Journalization

The -BI_SUPPRESSION_ENTRIES parameter is used to modify the number of internal entries used to minimize before image journalization:

```
ASSIGN_SITE_PARAMETER -BI_SUPPRESSION_ENTRIES g
```

where:

g - Number of internal entries. The valid range for this value is 0-280. The default value is 0 (i.e., before image journalization is not minimized).

Maximize Buffers For Access Methods

The -MOVE_MODE_MAX_BUF parameter is used to modify the maximum number of buffers for each file open that a particular access method (e.g., Relational File Manager (RFM)) can allocate:

```
ASSIGN_SITE_PARAMETER -MOVE_MODE_MAX_BUF m
```

where:

m - Maximum number of buffers. The valid range for this value is 1-50. The default value is 10.

SITE PARAMETER FOR WIRED REAL MEMORY FOR INTEGRATED SOFTWARE

The -SET_MAX_WIRED_PAGES parameter allows the site to exercise some control over system performance by allowing the Shared Domain Loader (S#SHARD) to wire the desired number of pages:

```
ASSIGN_SITE_PARAMETER -SET_MAX_WIRED_PAGES n
```

where:

n - Maximum number of pages to be wired for Integrated Software. The value specified can range from 40 to 16384 pages (1K). If a value is not specified, the GCOS 8 default of 511 is applied.

If too many pages are wired, other jobs may swap in and out of memory, degrading system performance. If too few pages are wired, jobs that access protected files may abort. For more information about determining memory requirements, refer to "Determining Minimum Wired Real Memory Page Requirements" later in this section.

SITE PARAMETER FOR VIRTUAL MEMORY

The -SC_VIRTUAL_BUFFER_SIZE parameter is used to specify the size of the virtual memory buffer used by Session Control:

```
ASSIGN_SITE_PARAMETER -SC_VIRTUAL_BUFFER_SIZE m
```

where:

m - Number of pages. The valid range is 16-4080 pages (1K). The default is 1000. If m > 1000, the value of the VSPACE parameter on the $ RESOURC statement must be increased.

SITE PARAMETERS FOR TRACES

Integrated Software Trace Buffer Size

The -TRACE_BUFFER_SIZE parameter specifies the size of the buffer used for Integrated Software trace event information:

```
ASSIGN_SITE_PARAMETER -TRACE_BUFFER_SIZE n
```

where:

n - Number of buffers. The valid range is 1 to 4. Each buffer has a size of 4K words. Thus the total buffer size may be 4K, 8K, 12K, or 16K words.

Increasing the default size during debugging sessions may be useful. For more information on Integrated Software trace, refer to the GCOS 8 OS Protected File Administration Guide.

Internal Traces For Specific Domains

Site parameters are provided to cause generation of internal traces in the following specific domains, to collect trace-event information that might otherwise be overwritten in the Integrated Software trace:

- Session Control (SC)
- Tenant Management (TM)
- Workstation Management (WM)
- Protected File I/O (PFIO)

These parameters are:

```
ASSIGN_SITE_PARAMETER -SC_INTERNAL_TRACE t

ASSIGN_SITE_PARAMETER -TM_INTERNAL_TRACE t

ASSIGN_SITE_PARAMETER -WM_INTERNAL_TRACE t

ASSIGN_SITE_PARAMETER -PFIO_INTERNAL_TRACE t
```

where:

t - 0 or NO or OFF (default) turns off the internal trace in the specified domain.

1 or YES or ON enables the internal trace in the specified domain.

SITE PARAMETERS FOR INTEGRATED SOFTWARE STATISTICS

Statistical information provided by Integrated Software can be extremely useful to the site for system-tuning purposes. The printing of both process-level and system-level Integrated Software statistics can be controlled through use of the following parameters. The site can specify file code (J* or P*), report code, and whether or not each group of statistics should be printed. Process-level statistics are available for the Buffer Management, Global Data Management, Before Journaling, After Journaling, Protected File I/O, and I-D-S/II domains. System-level statistics are available for Buffer Management, Global Data Management, After Journaling, Concurrency Control, Historical Logging, and Protected File I/O domains. Refer to the GCOS 8 OS System Operating Techniques manual for more information on these parameters, as well as a description of their output.

Print Integrated Software Statistics (Normal Termination)

The -ISSTAT_ENABLE parameter determines whether Integrated Software statistics are to be printed if a process terminates normally:

```
ASSIGN_SITE_PARAMETER -ISSTAT_ENABLE x
```

where:

x - 0 or NO or OFF (default) inhibits printing of statistics if a process terminates normally.

1 or YES or ON enables printing of statistics if a process terminates normally.

Print Integrated Software Statistics (Abnormal Termination)

The -ISSTAT_ABNORMAL_TERM parameter determines whether Integrated Software statistics are to be printed if a process terminates abnormally:

```
ASSIGN_SITE_PARAMETER -ISSTAT_ABNORMAL_TERM x
```

where:

x - 0 or NO or OFF (default) inhibits printing of statistics if a process terminates abnormally.

1 or YES or ON enables printing of statistics if a process terminates abnormally.

Integrated Software Statistics Output File Code

The -ISSTAT_FILE_CODE parameter defines the output file code for Integrated Software statistics:

ASSIGN_SITE_PARAMETER -ISSTAT_FILE_CODE c

where:

c - Output file code. Valid file codes are J* (default) or P*. If the P* file code is selected, a report code must also be specified via the -ISSTAT_REPORT_CODE parameter.

Integrated Software Statistics Report Code

If the P* file code is selected via the -ISSTAT_FILE_CODE parameter, the -ISSTAT_REPORT_CODE parameter must be used to define a report code for Integrated Software statistics:

ASSIGN_SITE_PARAMETER -ISSTAT_REPORT_CODE r

where:

r - Report code. Valid report codes are 0-77 (octal). The default report code is 74.

Print All Integrated Software System-Level Statistics

The -ISSTAT_ALL_SYS_LEVEL parameter specifies whether all system-level statistics (i.e., Buffer Management, Global Data Management, After Journal, Concurrency Control, Historical Log, and Protected File I/O domains) are to be printed:

ASSIGN_SITE_PARAMETER -ISSTAT_ALL_SYS_LEVEL x -PRINT_ZERO y

where:

x - 0 or NO or OFF inhibits printing of all system-level statistics, regardless of the values of any of the individual system-level statistics parameters.

1 or YES or ON causes printing of all system-level statistics, regardless of the values of any of the individual system-level statistics parameters.

If a value is not specified for x, the printing of system-level statistics for one or more individual domains may be enabled by the value of the individual system-level statistics parameter.

y - 0 or NO or OFF suppresses printing of zero-valued system-level statistical fields, regardless of the value of any PRINT_ZERO parameter used with individual system-level statistics parameters.

1 or YES or ON causes printing of zero-valued system-level statistical fields, regardless of the value of any PRINT_ZERO parameter used with individual system-level statistics parameters.

If a value is not specified for y, the printing of zero-valued system-level statistical fields for individual domains may be controled by the value of the PRINT_ZERO parameter used with the individual system-level statistics parameter.

Print Buffer Management Domain System-Level Statistics

The -ISSTAT_BM_SYS_LEVEL parameter specifies whether system-level Buffer Management Domain statistics are to be printed. This parameter is ignored if a value has been specified (YES or NO) for the -ISSTAT_ALL_SYS_LEVEL parameter.

```
ASSIGN_SITE_PARAMETER -ISSTAT_BM_SYS_LEVEL xxx -PRINT_ZERO yyy
```

where:

x - 0 or NO or OFF inhibits printing of system-level Buffer Management Domain statistics (default).

1 or YES or ON causes printing of system-level Buffer Management Domain statistics.

y - 0 or NO or OFF suppresses printing of zero-valued fields (default).

1 or YES or ON causes printing of zero-valued fields.

Print Global Data Management Domain System-Level Statistics

The -ISSTAT_DM_SYS_LEVEL parameter specifies whether system-level Global Data Management Domain statistics are to be printed. This parameter is ignored if a value has been specified (YES or NO) for the -ISSTAT_ALL_SYS_LEVEL parameter.

```
ASSIGN_SITE_PARAMETER -ISSTAT_DM_SYS_LEVEL xxx -PRINT_ZERO yyy
```

where:

x - 0 or NO or OFF inhibits printing of system-level Global Data Management Domain statistics (default).

1 or YES or ON causes printing of system-level Global Data Management Domain statistics.

y - 0 or NO or OFF suppresses printing of zero-valued fields (default).

1 or YES or ON causes printing of zero-valued fields.

Print After Journal Domain System-Level Statistics

The -ISSTAT_AJ_SYS_LEVEL parameter specifies whether system-level After Journal Domain statistics are to be printed. This parameter is ignored if a value has been specified (YES or NO) for the -ISSTAT_ALL_SYS_LEVEL parameter.

```
ASSIGN_SITE_PARAMETER -ISSTAT_AJ_SYS_LEVEL xxx -PRINT_ZERO yyy
```

where:

x - 0 or NO or OFF inhibits printing of system-level After Journal Domain statistics (default).

1 or YES or ON causes printing of system-level After Journal Domain statistics.

y - 0 or NO or OFF suppresses printing of zero-valued fields (default).

1 or YES or ON causes printing of zero-valued fields.

Print Concurrency Control Domain System-Level Statistics

The -ISSTAT_CC_SYS_LEVEL parameter specifies whether system-level Concurrency Control Domain statistics are to be printed. This parameter is ignored if a value has been specified (YES or NO) for the -ISSTAT_ALL_SYS_LEVEL parameter.

```
ASSIGN_SITE_PARAMETER -ISSTAT_CC_SYS_LEVEL xxx -PRINT_ZERO yyy
```

where:

x - 0 or NO or OFF inhibits printing of system-level Concurrency Control Domain statistics (default).

1 or YES or ON causes printing of system-level Concurrency Control Domain statistics.

y - 0 or NO or OFF suppresses printing of zero-valued fields (default).

1 or YES or ON causes printing of zero-valued fields.

Print Historical Log Domain System-Level Statistics

The -ISSTAT_HL_SYS_LEVEL parameter specifies whether system-level Historical Log Domain statistics are to be printed. This parameter is ignored if a value has been specified (YES or NO) for the -ISSTAT_ALL_SYS_LEVEL parameter.

ASSIGN_SITE_PARAMETER -ISSTAT_HL_SYS_LEVEL xxx -PRINT_ZERO yyy

where:

x - 0 or NO or OFF inhibits printing of system-level Historical Log Domain statistics (default).

1 or YES or ON causes printing of system-level Historical Log Domain statistics.

y - 0 or NO or OFF suppresses printing of zero-valued fields (default).

1 or YES or ON causes printing of zero-valued fields.

Print Protected File I/O Domain System-Level Statistics

The -ISSTAT_PFIO_SYS_LEVEL parameter specifies whether system-level Protected File I/O Domain statistics are to be printed. This parameter is ignored if a value has been specified (YES or NO) for the -ISSTAT_ALL_SYS_LEVEL parameter.

ASSIGN_SITE_PARAMETER -ISSTAT_PFIO_SYS_LEVEL xxx -PRINT_ZERO yyy

where:

x - 0 or NO or OFF inhibits printing of system-level Protected File I/O Domain statistics (default).

1 or YES or ON causes printing of system-level Protected File I/O Domain statistics.

y - 0 or NO or OFF suppresses printing of zero-valued fields (default).

1 or YES or ON causes printing of zero-valued fields.

Print All Integrated Software Process-Level Statistics

The -ISSTAT_ALL_PROC_LEVEL parameter specifies whether all process-level statistics (i.e., Buffer Management, Global Data Management, Before Journal, After Journal, Protected File I/O, and I-D-S/II domains) are to be printed:

ASSIGN_SITE_PARAMETER -ISSTAT_ALL_PROC_LEVEL x -PRINT_ZERO y

where:

x - 0 or NO or OFF inhibits printing of all process-level statistics, regardless of the values of any of the individual process-level statistics parameters.

1 or YES or ON causes printing of all process-level statistics, regardless of the values of any of the individual process-level statistics parameters.

If a value is not specified for x, the printing of process-level statistics for one or more individual domains may be enabled by the value of the individual process-level statistics parameter.

y - 0 or NO or OFF suppresses printing of zero-valued process-level statistical fields, regardless of the value of any PRINT_ZERO parameter used with individual process-level statistics parameters.

1 or YES or ON causes printing of zero-valued process-level statistical fields, regardless of the value of any PRINT_ZERO parameter used with individual process-level statistics parameters.

If a value is not specified for y, the printing of zero-valued process-level statistical fields for individual domains may be controled by the value of the PRINT_ZERO parameter used with the individual process-level statistics parameter.

Print Buffer Management Domain Process-Level Statistics

The -ISSTAT_BM_PROC_LEVEL parameter specifies whether process-level Buffer Management Domain statistics are to be printed. This parameter is ignored if a value has been specified (YES or NO) for the -ISSTAT_ALL_PROC_LEVEL parameter.

```
ASSIGN_SITE_PARAMETER -ISSTAT_BM_PROC_LEVEL xxx -PRINT_ZERO yyy
```

where:

x - 0 or NO or OFF inhibits printing of process-level Buffer Management Domain statistics.

1 or YES or ON causes printing of process-level Buffer Management Domain statistics (default).

y - 0 or NO or OFF suppresses printing of zero-valued fields (default).

1 or YES or ON causes printing of zero-valued fields.

Print Global Data Management Domain Process-Level Statistics

The -ISSTAT_DM_PROC_LEVEL parameter specifies whether process-level Global Data Management Domain statistics are to be printed. This parameter is ignored if a value has been specified (YES or NO) for the -ISSTAT_ALL_PROC_LEVEL parameter.

```
ASSIGN_SITE_PARAMETER -ISSTAT_DM_PROC_LEVEL xxx -PRINT_ZERO yyy
```

where:

x - 0 or NO or OFF inhibits printing of process-level Global Data Management Domain statistics.

1 or YES or ON causes printing of process-level Global Data Management Domain statistics (default).

y - 0 or NO or OFF suppresses printing of zero-valued fields (default).

1 or YES or ON causes printing of zero-valued fields.

Print Before Journal Domain Process-Level Statistics

The -ISSTAT_BJ_PROC_LEVEL parameter specifies whether process-level Before Journal Domain statistics are to be printed. This parameter is ignored if a value has been specified (YES or NO) for the -ISSTAT_ALL_PROC_LEVEL parameter.

ASSIGN_SITE_PARAMETER -ISSTAT_BJ_PROC_LEVEL xxx -PRINT_ZERO yyy

where:

x - 0 or NO or OFF inhibits printing of process-level Before Journal Domain statistics.

1 or YES or ON causes printing of process-level Before Journal Domain statistics (default).

y - 0 or NO or OFF suppresses printing of zero-valued fields (default).

1 or YES or ON causes printing of zero-valued fields.

Print After Journal Domain Process-Level Statistics

The -ISSTAT_AJ_PROC_LEVEL parameter specifies whether process-level After Journal Domain statistics are to be printed. This parameter is ignored if a value has been specified (YES or NO) for the -ISSTAT_ALL_PROC_LEVEL parameter.

ASSIGN_SITE_PARAMETER -ISSTAT_AJ_PROC_LEVEL xxx -PRINT_ZERO yyy

where:

x - 0 or NO or OFF inhibits printing of process-level After Journal Domain statistics.

1 or YES or ON causes printing of process-level After Journal Domain statistics (default).

y - 0 or NO or OFF suppresses printing of zero-valued fields (default).

1 or YES or ON causes printing of zero-valued fields.

Print Protected File I/O Domain Process-Level Statistics

The -ISSTAT_PFIO_PROC_LEVEL parameter specifies whether process-level Protected File I/O Domain statistics are to be printed. This parameter is ignored if a value has been specified (YES or NO) for the -ISSTAT_ALL_PROC_LEVEL parameter.

```
ASSIGN_SITE_PARAMETER -ISSTAT_PFIO_PROC_LEVEL xxx -PRINT_ZERO yyy
```

where:

x - 0 or NO or OFF inhibits printing of process-level Protected File I/O Domain statistics.

1 or YES or ON causes printing of process-level Protected File I/O Domain statistics (default).

y - 0 or NO or OFF suppresses printing of zero-valued fields (default).

1 or YES or ON causes printing of zero-valued fields.

Print I-D-S/II Domain Process-Level Statistics

The -ISSTAT_IDS2_PROC_LEVEL parameter specifies whether process-level I-D-S/II Domain (Shared Data Base Control System (SDBCS)) statistics are to be printed. This parameter is ignored if a value has been specified (YES or NO) for the -ISSTAT_ALL_PROC_LEVEL parameter.

```
ASSIGN_SITE_PARAMETER -ISSTAT_IDS2_PROC_LEVEL xxx -PRINT_ZERO yyy
```

where:

x - 0 or NO or OFF inhibits printing of process-level I-D-S/II Domain statistics.

1 or YES or ON causes printing of process-level I-D-S/II Domain statistics (default).

y - 0 or NO or OFF suppresses printing of zero-valued fields (default).

1 or YES or ON causes printing of zero-valued fields.

Print Statistics For Batch Processes

The -ISSTAT_BATCH_PROCESS parameter specifies whether Integrated Software statistics for batch processes are to be printed:

```
ASSIGN_SITE_PARAMETER -ISSTAT_BATCH_PROCESS x
```

where:

x - 0 or NO or OFF inhibits printing of statistics for batch processes.

1 or YES or ON causes printing of statistics for batch processes (default).

2 or ISSTAT causes printing of statistics for batch processes that specified the ISSTAT option on the JCL activity definition statement.

Print Statistics For Workstation Processes

The -ISSTAT_WS_PROCESSES parameter specifies whether Integrated Software statistics for workstation processes are to be printed:

ASSIGN_SITE_PARAMETER -ISSTAT_WS_PROCESSES x

where:

x - 0 or NO or OFF inhibits printing of statistics for workstation processes (default).

1 or YES or ON causes printing of statistics for workstation processes.

2 or ISSTAT causes printing of statistics for workstation processes that specified the ISSTAT option on the JCL activity definition statement.

Print Statistics For DM-IV/TP Processes

The -ISSTAT_TP_PROCESSES parameter specifies whether Integrated Software statistics for DM-IV/TP processes are to be printed:

ASSIGN_SITE_PARAMETER -ISSTAT_TP_PROCESSES x

where:

x - 0 or NO or OFF inhibits printing of statistics for DM-IV/TP processes (default).

1 or YES or ON causes printing of statistics for DM-IV/TP processes.

2 or ISSTAT causes printing of statistics for DM-IV/TP processes that specified the ISSTAT option on the JCL activity definition statement.

Print Statistics For TDS Processes

The -ISSTAT_TDS_PROCESSES parameter specifies whether Integrated Software statistics for TDS processes are to be printed. Specifying a value of YES for the parameter indicates that the statistics are to be printed. Valid arguments are YES or NO. NO is the default value. The directive containing this parameter is as follows:

ASSIGN_SITE_PARAMETER -ISSTAT_TDS_PROCESSES x

where:

x - 0 or NO or OFF inhibits printing of statistics for TDS processes (default).

1 or YES or ON causes printing of statistics for TDS processes.

2 or ISSTAT causes printing of statistics for TDS processes that specified the ISSTAT option on the JCL activity definition statement.

Print Statistics For TS8 Processes

The -ISSTAT_TS8_PROCESSES parameter specifies whether Integrated Software statistics for TS8 processes are to be printed:

ASSIGN_SITE_PARAMETER -ISSTAT_TS8_PROCESSES x

where:

x - 0 or NO or OFF inhibits printing of statistics for TS8 processes.

1 or YES or ON causes printing of statistics for TS8 processes (default).

2 or ISSTAT causes printing of statistics for TS8 processes that specified the ISSTAT option on the JCL activity definition statement.

SITE PARAMETERS FOR WORKSTATIONS

Default pathnames for three Integrated Software files are provided. There may be some circumstances, however, under which pathname modification is desirable. For example, before installing a new version of a file, the system administrator may wish to test file content; by modifying the pathname of the new (test) version, the file can be accessed without interrupting system processing that uses the existing version.

Pathnames are provided for the following files:

- o Workstation definition data base (WS-DB)
- o Workstation management schema file (WCC-1STR)
- o Workstation management subschema file (WCC-6STR)

Pathnames for these files are defined via the directives:

ASSIGN_SITE_PARAMETER -WD_FILE p

where:

p - File pathname. The default is SYS_SOFTWARE/yyyy/DATA/WC-DB

yyyy - Software Release identifier, such as 3000

ASSIGN_SITE_PARAMETER -WCC_1_STAR_FILE p

where:

p - File pathname. The default is SYS_SOFTWARE/yyyy/DATA/WCC-1STR

yyyy - Software Release identifier, such as 3000

ASSIGN_SITE_PARAMETER -WCC_6_STAR_FILE p

where:

p - File pathname. The default is SYS_SOFTWARE/yyyy/DATA/WCC-6STR

yyyy - Software Release identifier, such as 3000

Buffer Pools

The CREATE_BUFFER_POOL directive creates buffer pools for protected files. Each buffer pool must be assigned a name and a control interval size using this directive. Other parameters that can be assigned specify:

- o Minimum and maximum number of buffers comprising the buffer pool
- o Number of internal table entries used to minimize before image journalization
- o Whether the buffer pool is public or private
- o When the buffer pool is created.
  - When Integrated Software is loaded
  - When protected files are accessed

If the CREATE_BUFFER_POOL directive declares a buffer pool private, the directive:

```
ASSIGN -PRIVATE_BUFFER_POOL_FILE
```

must be used to define the pathname (i.e., catalog/file string) of the file assigned to the pool and the buffer pool name when a process opens the file. The GCOS 8 OS Integrity Control manual provides a detailed discussion of buffer pools.

The following directives can be used to define buffer pools and to modify parameter values. The abbreviated format of the directive or keyword is in parentheses.

```
CREATE_BUFFER_POOL (CBP) aaaaa &
   -BUFFER_SIZE (-BS or -CI_SIZE) bbbbb &
   [option(s)]
```

where option(s) = one or more of the following keyword-argument fields:

```
[-TYPE ttttt]
[-MIN_NUMBER_BUFFERS (-MNNB) cccc]
[-MAX_NUMBER_BUFFERS (-MXNB) dddd]
[-POOL_ALLOCATION (-PA) eeeee]
[-BI_SUPPRESSION_ENTRIES (-BSE) fff]
```

```
ASSIGN -PRIVATE_BP_FILE &
   -BUFFER_POOL_NAME (-BPN) aaaaa &
   -PATH_NAME (-PN or -PATHNAME) ggggg &
   -NUMBER_BUFFERS (-NB) hhhhh
```

where:

& - Continuation character, which is required when a directive extends to more than one line.

aaaaa - Buffer pool name, which can contain from 1 to 12 alphanumeric characters. No blanks are permitted.

bbbbb - Size of the buffer or control interval, specified in bytes or words, as indicated by a "B" (default) or "W", respectively, following the value. The size can range from 256 to 65536 bytes, in increments of 256 bytes. Values that are not multiples of 256 bytes are rounded up to the next 256-byte increment.

cccc - Minimum number of buffers in the buffer pool. This value can range from 0 to 4096; the default is 0.

dddd - Maximum number of buffers in the buffer pool. This value can range from 1 to 4096; the default is 40 or the MIN_NUMBER_BUFFERS value if cccc is greater than 40.

ttttt - Literal identifying the buffer pool type:

PRIVATE = private buffer pool

PUBLIC = public buffer pool

eeeee - Literal identifying when the buffer pool is created:

IMMEDIATE = Created when Integrated Software is loaded

DEFERRED = Created when protected files are accessed

fff - Number of entries used to suppress before images. Range is 0-280, default is 0.

ggggg - Pathname of the protected file assigned to the private buffer pool bbbbb. The pathname cannot contain passwords.

hhhhh - Number of buffers to be assigned when a process opens the file (0-1024; default is 0). When this parameter has a value of 0, the value specified for ASSIGN_SITE_PARAMETER -PFIO_OPEN_BUFFERS (buffers allocated at file open) is used. When a value other than 0 is specified, it overrides the value specified for -PFIO_OPEN_BUFFERS.

TERMINATING INTEGRATED SOFTWARE

Integrated Software must be explicitly aborted to terminate execution. The operator must enter an ABORT SHARD console command. This makes Integrated Software unavailable for starting any new processes; however, no negative effects result on any processes currently executing, as enough SHARD facilities remain active to continue processing until all processing completes. Thus, any attempt to terminate SHARD prematurely is delayed.

If S#SHARD hangs in termination, use the following procedure:

1. Enter "WORKST LIST" at the console to determine the active workstations. Then abort each workstation by entering "WORKST ABORT xxxx" for each of the active workstations (where xxxx is the workstation id).

2. Run SHRNM to determine if jobs are still tied to SHARD, then abort them.

3. Wait until all abort messages for requested jobs have been typed on the console log. S#SHARD will then terminate.

*

To restart Integrated Software after it terminates, execute the privileged slave program that loads the software into memory. (Refer to "Loading Integrated Software For Execution" earlier in this section for additional information.)

ENSURING SUFFICIENT MEMORY FOR INTEGRATED SOFTWARE EXECUTION

Integrated Software memory requirements are of the following categories.

1. Virtual memory for all Integrated Software domains, Session Control buffers, and data base buffers. The $ RESOURC statement that is incorporated into the Honeywell Bull-supplied job stream which loads Integrated Software into memory requests 3800K of virtual memory. This value is normally sufficient for most sites.

2. Wired real memory pages for all buffers reserved for all batch and workstation processes executing portions of Integrated Software. The maximum number of real memory pages that Integrated Software can wire is defined via the -SET_MAX_WIRED_PAGES parameter.

   NOTES: a. Every 1000K of virtual memory requires 1K of table space in wired real memory pages. Therefore, if less than the default allocation of virtual memory (3800K) is required, some real memory space can be saved by reducing the default allocation via the $ RESOURC statement.

   b. Real memory requirements for Integrated Software are initially defined via the RSPACE filed on the $ RESOURC statement in the job stream that spawns Integrated Software into execution. By default, 100K of real memory is reserved for Integrated Software. However, if this value is understated or overstated, the GCOS 8 System Dynamic Memory Manager program dynamically modifies allocated real memory for Integrated Software to meet the immediate demand. Integrated Software's working set is dynamically increased to reduce page faults during peak periods of processing. Conversely, memory is returned to the pool of available space for other uses during low-processing periods.

If the $ RESOURC statement specifies insufficient space for the VSPACE, which Integrated Software is to load, Integrated Software aborts and an abort code of VF is issued.

The SDLD directive ASSIGN_SITE_PARAMETER -SET_MAX_WIRED_PAGES establishes the maximum number of real memory pages that Integrated Software may wire. The default for this parameter is 511(K). The range for the parameter value is 40-16384(K). If this parameter value is set too low, processes attempting to wire memory will abort. If the parameter value is set too high, performance of other applications run on the system may be impacted.

Processes encountering an Integrated Software memory allocation problem may abort with the error message:

```
UFAS ERR TYPE 8 - ERR # 015:
UNABLE TO RECOGNIZE IO STATUS 477702000000 ON FILE ff
   or:
IMG/WSM ERROR RETURN
```

issued to the Execution Report (where ff = file code).

Determining Minimum Wired Real Memory Page Requirements

Determining the number of wired real memory pages for Integrated Software may require some experimentation. The following formula yields a value that may be used as a starting point for determining a value for the -SET_MAX_WIRED_PAGES parameter (i.e., maximum number of real memory pages that Integrated Software may wire). This wired real memory is used for executing Integrated Software and for other job-related functions (e.g., journalizing after images). When using this formula, it is recommended that each variable value specified represent the maximum expected during the load of Integrated Software.

The recommended approach for setting a final value for -SET_MAX_WIRED_PAGES is to determine a value such that if all memory wired for Integrated Software (i.e., SNUMB of SHARD) were used, sufficient memory would still be available to support remaining system functions.

minimum value for -SET_MAX_WIRED_PAGES = S + P + B + J + A + H

Variable	Description
S	Integrated Software domain wired pages. 14 + (4 * n) where n is the value specified via the site parameter -TRACE_BUFFER_SIZE n which determines the number of 4K trace buffers.
P	Wired real memory pages for system-wide protected file services. This value is required any time one or more executing jobs access protected files (i.e., files created with the ACCESS/MONITOR/, ABORT/ROLLBACK/, ABORT/ARCHIVE/, ACCESS/CONCURRENT/, and/or RDERR/JOURNAL/ option specified). This value is determined via the formula (3 * F) + 5 where: F - Number of different control interval sizes for all jobs accessing protected files at any one time. For example, if a total of 10 jobs access protected files at one time or another but only 3 jobs access the files at any one time and if control interval sizes are 256 and 512 (words), specify 2. Consequently, the number of wired pages required for system-wide protected file service is (3 * 2) + 5 = 11 (K)

Variable	Description
B	Wired real memory pages for buffer space required for protected files accessed by executing jobs. Determine this value via the formula

F * C * N

where:

F - Number of allocations to protected files of the same control interval size (C) for all executing jobs concurrently accessing those files.

C - Control interval size (in words) of the protected files. This is the page size for the files.

N - Default value is 2. Other values may be specified via the site parameter:

-PFIO_OPEN_BUFFERS bbb.

Repeat this formula for each different control interval size and derive a cumulative RSPACE value from the totals; then divide by 1024 to get the number for wired pages. Round the result upward to the nearest integer.

For example, three executing jobs access nine protected files with varying control interval sizes. Four files have 256-word control interval sizes, four files have 512-word control interval sizes, and one file has a 1024-word control interval size. The value for B would be:

[(4*256*2) + (4*512*2) + (1*1024*2)]/1024 = 8(K)

J Wired real memory for journalizing before images. If any file accessed by a process was created with the ABORT/ROLLBACK/ option, this value is required. Determine the value via the formula

3 + (3 * C * P)/1024

where:

C - Value for parameter -BJ_CISIZE (in words). Default is 704 words.

P - Maximum number of processes using before journals at any one time. For example, if a total of four batch processes use before journals at one time or another, but only two processes are executing at any one time, specify 2. Consequently, the amount of wired pages required for before journals (assuming the default value of 704 words is in effect for -BJ_CISIZE) is:

Variable	Description

$$3 + \frac{3*704*2}{1024} = 8 \text{ (K)}$$

Refer to the GCOS 8 OS TP8 Administrator's Guide for information on the additional wired memory pages required for TP8 workstation processing.

A Wired real memory pages for system-level journals (i.e., archived before journals, after journals, and checkpoint journal). Specify 18K for the first (or only) journal implemented. For each additional journal created, add the following:

2 * B

where:

B - Control interval size (in K words) of the journal (i.e., value for BLKSIZ option of Create Journal (CRJR) command)

For example, assuming that an installation has one archived before journal and one after journal and that both journals are created with a BLKSIZ value of 2 (K), the value for A would be

18 + (2 * 2) = 22 (K)

H Wired real memory pages for the historical log. If this feature is implemented (i.e., SDLD parameter -HISTORICAL_LOG has a value of YES), this value is required. Determine the value via the formula

(C * 3)/1024

where:

C - Control interval size of historical log (SDLD parameter -HL_CISIZE) in words.

Round the result upward to the nearest integer.

For example, if -HL_CISIZE has a value of 2816 bytes (704 words) the value for H is

(704 * 3)/1024 = 3

The following example approximates the minimum wired real memory pages required at an installation where (1) -TRACE_BUFFER_SIZE parameter has a value of 1 (default); (2) there are two control interval sizes for protected files; (3) four files have a control interval size of 256-words and four files have a control interval size of 512 words; (4) a maximum of two processes journalize before images at any one time and the before journal has a control interval size of 2816 bytes (704 words); (5) one after journal and one archived before journal have been created; and (6) historical log is implemented with a control interval size of 28816 bytes (704 words).

S =	18 (K)	14 + (4 * 1)
P =	11 (K)	(3 * 2) + 5
B =	6 (K)	[(4 * 256 * 2) + (4 * 512 * 2)]/1024
J =	8 (K)	3 + (3 * 704 * 2)/1024
A =	22 (K)	18 + (2 * 2)
H =	3 (K)	(704 * 3)/1024
	68 (K)	

EXAMPLES APPROXIMATING WIRED REAL MEMORY PAGES REQUIREMENTS

The following examples suggest minimum wired real memory page requirements when batch processes access protected files. As previously indicated, these requirements represent a starting point for determining a value for the -SET_MAX_WIRED_PAGES parameter.

1. The following examples assume one 4K trace buffer. Add 4 for each additional 4K trace buffer.

2. The following examples assume PFIO_OPEN_BUFFERS is 2.

3. Each additional data base file having the same control interval size as those files already considered in the wired pages computation creates a requirement for additional wired real memory pages. The following formula needs to be evaluated for each such additional data base file:

   B = (F * C * N) / 1024

   where:

   B = the number of wired real memory pages required for this file
   F = the number of processes accessing this file concurrently
   C = control interval size, in words
   N = 2 open buffers

   If B is not an integer it should be increased to the next larger integer.

4. The value of B should be increased by 3 for each additional control interval size that is different from those already considered in the wired pages computation, to represent the overhead of each buffer pool.

Accessing One Protected File

NOTE: The value of B should be increased by 1 for each additional process accessing the data base file.

1. A minimum of 36 wired real memory pages is required when two batch processes access one data base file having the following characteristics:

   - o The file is allocated with Write/Concurrent permission.
   - o The file control interval size is 512 words.
   - o ACCESS/MONITOR/ and/or ABORT/ROLLBACK/ protection is specified.

2. A minimum of 54 wired real memory pages is required when two batch processes access one data base file having the following characteristics:

   - o The file is allocated with Write/Concurrent permission.
   - o The file control interval is 512 words.
   - o ACCESS/MONITOR/, ABORT/ROLLBACK/, and RDERR/JOURNAL/ permissions are specified.

*

Accessing Multiple Protected Files Of Same Control Interval Size

NOTE: The value of B should be increased by 2 for each additional process accessing the data base files.

1. A minimum of 38 wired real memory pages is required when two batch processes access two data base files, each of which has the following characteristics:

   - o The files are allocated with Write/Concurrent permission.
   - o The file control interval size is 512 words.
   - o ACCESS/MONITOR/ and/or ABORT/ROLLBACK/ protection is specified.

2. A minimum of 56 wired real memory pages is required when two batch processes access two data base files, each of which has the following characteristics:

   - The files are allocated with Write/Concurrent permission.
   - The file control interval size is 512 words.
   - ACCESS/MONITOR/, ABORT/ROLLBACK/, and RDERR/JOURNAL/ protections are specified.

*

Accessing Multiple Protected Files Of Different CI Size

NOTE: The value of B should be increased by 2 for each additional process accessing the data base files.

1. A minimum of 41 wired real memory pages is required when two batch processes access two data base files, which have the following characteristics:

   - The files are allocated with Write/Concurrent permission.
   - One file has a control interval size of 320 words; the other file has a control interval size of 512 words.
   - ACCESS/MONITOR/ and/or ABORT/ROLLBACK/ protection is specified for each file.

2. A minimum of 59 wired real memory pages is required when two batch processes access two data base files, which have the following characteristics:

   - The files are allocated with Write/Concurrent permission.
   - One file has a control interval size of 320 words; the other file has a control interval size of 512 words.
   - ACCESS/MONITOR/, ABORT/ROLLBACK/, and RDERR/JOURNAL/ protections are specified for each file.

*

Executing Only Unpurge

The minimum number of wired real memory pages required to execute only the Unpurge program (i.e., no other processing is occurring in the system) is 15. If the Unpurge program is executing with any other programs in the product set, use the preceding formula to determine the wired pages requirement.

INTEGRATED SOFTWARE SHARED DOMAIN LOADER (SDLD) DIRECTIVES

The SDLD directives specify configuration information for the load of the Integrated Software shared domains such as files to be allocated, domains to be included or excluded, domain patches, start and end of defaults or changes, comments, and the assignment of a shared space name.

The SDLD directives are initially read from the I* filecode. The "ALLOCATE -DIRECTIVES" diverts the input to the specified input file. When an EOF is found in the directives file, processing of input continues from I*.

SDLD Directives Syntax (General)

The SDLD directives syntax is a subset of Honeywell Bull Control Language (HCL) with the following restrictions:

CHARACTER-SET:	BCD for the I* file ($ DATA limitation); BCD or ASCII for "Allocate Directives" files.
DELIMITER:	SPACE is a field delimiter. Multiple or leading spaces are treated as one or ignored, except leading spaces cannot be used for continuation. The PATCH directive has a fixed format.
CONTINUATION:	Ampersand (&) specifies input continuation to the next input statement. DATA after the "&" is ignored (see NOTE).
comments/remarks:	The three-character escape sequence of "-- " (2 hyphens, space) indicates the beginning of a comment field. Data following the comment sequence is ignored (see NOTE).
line-numbers:	Line numbers are not allowed; use $$STRIP on ASCII files.

NOTE: The SDLD LOAD-MAP generator checks arguments for a valid domain name; if found, the input line is listed in the Load Map for the specified domain.

SDLD Directives Format

The format of SDLD directives is:

<command> <arg> <keyword-1 arg-1>...<keyword-n arg-n>

where:

<command>	= SDLD directive name
<arg>	= Positional argument, as required
<keyword-1 arg-1> <keyword-n arg-n>	= SDLD keyword and the keyword argument (as required). All keywords must begin with "-" (hyphen) (e.g., -LIBRARY, -PATHNAME).

NOTE: <Keyword arg> phrases may appear in any order. Abbreviated <command> and <keyword> forms are valid as described later, under "Directives Syntax (Specific)".

Shared Loader Domain Directives Description

The Shared Loader Domain directives specify the actions the SDLD performs and are described, alphabetically, as follows:

Directive	Description
ALLOCATE	Specifies a dynamically allocated file for SDLD execution. File options available are: -LIBRARY, for virtual domain Q* library -DIRECTIVES, for diverting the input stream to a directives file -BACKDOOR, for a backdoor print of LOAD-MAP -BPDIRECTIVES, for Buffer Pool directives file
ASSIGN_SITE_PARAMETER	Assigns site parameter values.
END_CHANGES	Specifies the end of the CHANGE directives.
END_DEFAULTS	Specifies the end of the DEFAULTS directives.
EXCLUDE_DOMAIN	Specifies a virtual domain not to be loaded (i.e., excludes a domain specified by the DEFAULT directives).
INCLUDE_DOMAIN	Specifies a virtual domain to load.
PATCH or OCTAL	Patches a virtual domain. Either format is fixed and consistent with the Startup PATCH directives format. The word PATCH is conventionally used for patches for Integrated Software.

Directive	Description
Remarks:	Remarks are specified by "-- " in columns 1-3. SDLD inserts "REMARKS" images in the Load Map when a valid domain-id is found in columns 73-78.
START_CHANGES	Specifies the beginning of the CHANGE directives.
START_DEFAULTS	Specifies the beginning of the DEFAULT directives.

NOTE: The START and END directives specify a block of directives with consistent usage (i.e., an END_DEFAULTS must follow the START_DEFAULTS and precede START_CHANGES). Also, MOST directives must be preceded by a START block directive. An exception is "ALLOCATE -DIRECTIVES", which may precede the START block directives.

Directives Syntax (Specific)

The general syntactical format for the directives was described earlier, under "SDLD Directives Syntax (General)". The specific format for each directive is described alphabetically below, with any valid abbreviations enclosed within parentheses "()".

ALLOCATE Directive (ALLO)

ALLOCATE <file-type> <path-name> <abt-ctl>

where:

<file-type> = -LIBRARY <lc> (-LIB <lc>)
To specify domain library allocation. <lc> is the library-code reference used on the INCLUDE_DOMAIN directive and is two alphanumeric characters. Library_code ".Q" is reserved for the DEFAULT search library.

-BACKDOOR <lc> (-BD <lc>)
To specify the backdoor load map print file. The default value for <lc> is ".B".

-DIRECTIVES <lc> (-DIRE <lc>)
To specify an input directives file. The default value for <lc> is ".D".

-BPDIRECTIVES <lc> (-BPDIRE <lc>)
To specify an input Buffer Pool directives file which will override the "$ DATA BP" data file specified in SHARD.SPWN JCL. The default value for <lc> is "BP".

<path-name> = -PATHNAME <cat-string> (-PN <cat-string>)
To specify the pathname of the file to be allocated. <cat-string> is the catalog file string (e.g., UMC/CAT1/CAT2/FILE).

<abt-ctl> = -REQUIRED (-R)
To specify SDLD fatal abort control if allocation fails. This is the default.

-OPTIONAL (-O)
To specify SDLD non-fatal abort control if allocation fails.

Examples:

```
ALLOCATE -LIBRARY AA -PATHNAME UMC/CAT1/CAT2/FILE -REQUIRED
or ALLO -LIB AA -PN UMC/CAT1/CAT2/FILE -R

ALLOCATE -BACKDOOR -PATHNAME UMC/LOAD-MAP -OPTIONAL
or ALLO -BD -PN UMC/LOAD-MAP -O

ALLOCATE -DIRECTIVES -PATHNAME UMC/CHANGES (-REQUIRED is default)
or ALLO -DIRE -PN UMC/CHANGES

ALLOCATE -BPDIRECTIVES -PATHNAME umc/BP-DIRE
or ALLO -BPDIRE -PN umc/BP-DIRE
```

ASSIGN_SITE_PARAMETER Directive

ASSIGN_SITE_PARAMETER <keyword-1><arg-1>

where:

<keyword> is one of the allowed site parameter keywords. See "Site Parameters", earlier in this section.

<arg> is the value associated with the keyword.

NOTE: Only one site parameter is allowed per ASSIGN_SITE_PARAMETER directive.

END CHANGES Directive

END_CHANGES

This directive specifies the end of the CHANGES directives and must appear as shown.

END_DEFAULTS Directive

END_DEFAULTS

This directive specifies the end of the DEFAULTS directives and must appear as shown.

EXCLUDE_DOMAIN Directive (EXCL)

EXCLUDE_DOMAIN <domain-id>

where: <domain-id> = Name of the domain to exclude.

Examples:

```
EXCLUDE_DOMAIN .MZZZZ
or EXCL .MZZZZ
```

INCLUDE_DOMAIN Directive (INCL)

INCLUDE_DOMAIN <domain-id> -LIBRARY <lc> <abt-ctl>

where:

<domain-id> = Name of the domain to be included.

-LIBRARY <lc> = <lc> specifies the library-code from which the domain is to be loaded (-LIB <lc>). If library <lc> is not located, library ".Q" is searched.

<abt-ctl> = -REQUIRED (-R)
To specify SDLD FATAL "ERROR" if the domain search fails. This is the default.

-OPTIONAL (-O)
To specify a SDLD NON-FATAL "ERROR" if the domain search fails.

Examples:

```
INCLUDE_DOMAIN .MZZZZ -LIBRARY XX -REQUIRED
or -INCL .MZZZZ -LIB XX -R
or -INCL .MZZZZ -LIB XX
```

PATCH Or OCTAL Directive

The PATCH and OCTAL directives have fixed formats as follows:

Position	Description
1-6	The 1-6 positions of the OCTAL location to patch.
8-12	"PATCH" or "OCTAL". "PATCH" is used by convention.
16-27	The 1-12 positions of the octal value for PATCH or OCTAL
n-72	Comments may be entered. There must be at least one space between a comment and the preceding field.
73-78	The domain-id to patch. This field is required.

Examples:

```
                1                                             7
1       8       6                                             3
--------------------------------------------------------------------
123     PATCH   1234 Set location 123=0...1234                .MZZZZ
000123 OCTAL    000000001234 Set location 123=00...1234       .MZZZZ
```

START_CHANGES Directive

START_CHANGES

This directive specifies the beginning of the CHANGES directives and must appear as shown. This directive must not precede the START_DEFAULTS ... END_DEFAULTS directives. See Figure 4-1, "Order Of SDLD Directives".

START_DEFAULTS Directive

START_DEFAULTS

This directive specifies the beginning of the DEFAULTS directives and must appear as shown. This directive must precede the START_CHANGES directive. See Figure 4-1, "Order Of SDLD Directives".

Remarks:

The SDLD remarks line is specified by "-- " in columns 1-3 of any directive and causes all data on that line to be ignored. However, if a valid domain-id is specified in columns 73-78, the remarks line is displayed in the Load Map domain-id section.

Examples:

```
              1                                    7
1     8       6                                    3
_____________________________________________________

-- THIS IS A "REMARKS" LINE WITH NO DOMAIN-ID
-- THIS IS A "REMARKS" LINE THAT WILL APPEAR IN DOMAIN  .MUTTD
-- 0123 PATCH 007777001001 THIS DELETES A PATCH FROM    .MZZZZ
```

NOTE: PATCH directives may be deleted by placing "-- " (dash, dash, space) in columns 1-3.

```
START_DEFAULTS
-- allocate all DEFAULT library files
ALLOCATE -LIBRARY AA -PATHNAME path-name
    .
    .
    .
ALLOCATE -LIBRARY ZZ -PATHNAME path-name
ALLOCATE -LIBRARY .Q -PATHNAME path-name
-- specify DEFAULT DOMAINS to be loaded
INCLUDE_DOMAIN .MUTTR -LIBRARY AA
    .
    .
    .
INCLUDE_DOMAIN .MZZZZ -LIBRARY ZZ
END_DEFAULTS
-- allocate CHANGE directive file
ALLOCATE -DIRECTIVES -PATHNAME change/path/name
-- directives read from "change/path/name"
START_CHANGES
ALLOCATE -LIBRARY AA -PATHNAME test/lib/aa
EXCLUDE_DOMAIN .MZZZZ -- don't load .MZZZZ
0123 PATCH 123412341234 patch loc 123       .MUTTR
ASSIGN_SHARED_NAME XYZ
ALLOCATE -BACKDOOR -PATHNAME test/backdoor/file
END_CHANGES
(EOF on directive file)
(EOF on I*)
```

Figure 4-1. Order Of SDLD Directives

Shared Domain Loader Configuration

The Shared Domain Loader (SDLD) configuration is controlled by the ALLOCATE directive, which dynamically allocates the SDLD input and output files. The dynamic allocation facility provides the ability to bypass the allocation of files that cannot be allocated when the -OPTIONAL control argument is specified. This allows non-fatal references to missing files (e.g., during cold boot).

The dynamic feature also provides configuration of Integrated Software options not ordered, without causing a fatal error during loading.

The ALLOCATE directive specifies the pathname which SDLD uses to attempt dynamic allocation. Allocation errors are controlled by SDLD instead of GEIN or PALC.

INPUT FILE ALLOCATION

Library Files

The SDLD Library files are searched to load specified domains using a library code specifying unique files. The default value is ".Q".

Directive Files

SDLD input Directive files can be dynamically allocated to allow for bypassing of CHANGE directive files during cold boot or any time a Directives file is missing.

OUTPUT FILE ALLOCATION

Backdoor File

Backdoor file allocation is done dynamically to prevent fatal aborts if the Backdoor file is unavailable; thus, preventing the listing of the Load Map via the Backdoor facility. A Backdoor file is not allocated in the DEFAULT directives, where, normally, no Backdoor Load Map is generated. The user may allocate the Backdoor file as desired.

JCL And Directives Usage

JCL REQUIREMENTS

SDLD is executed as a native mode batch process. The required virtual domains are loaded with the SNUMB "SHARD" and started during System Startup ROLLCALL. SDLD may be restarted with a set of JCL in a TSS JRN file or through the card reader. JCL requirements to load Integrated Software are:

```
              1
1      8      6
------------------------------------------------------------------

$      IDENT   <ident-image>
$      USERID  <user-id$password>
$      RUN     RUFILE=SYSTEM,RUNAME=.VSDLD,OPTION=DUMP    (1)
$      RESOURC VSPACE=3800K                               (2)
$      PRIVITY
$      DATA    I*
  - SDLD Directives (see Figure 4-1)
$      ENDJOB
```

NOTES: 1. The $ RUN JCL Statement invokes the SDLD loader from the installed set of system run units taken from the seed workspace image file.

2. The $ RESOURC JCL Statement has the virtual space limits for Integrated Software loading.

LOADING REQUIREMENTS FOR INTEGRATED SOFTWARE DOMAINS

The SDLD directives are processed by SDLD from the I* file, normally, by using the "$ DATA I*" JCL statement. The directives usually specify DEFAULT and, optionally, CHANGE directives.

The best method of configuring SDLD into the system uses the "ALLOCATE -DIRECTIVES" directive for DEFAULT and CHANGE file allocation. This method provides dynamic allocation of the directives files and documents the files used on the SDLD reports.

Example:

```
              1
1      8      6
------------------------------------------------------------------

$      DATA    I*
ALLOCATE -DIRECTIVES -PATHNAME UMC/DEFAULTS
ALLOCATE -DIRECTIVES -PATHNAME UMC/CHANGES
$      ENDJOB
```

The directives files may be allocated using GCOS allocation via a "$ SELECT" JCL statement. The previous example can thus be restated as:

```
               1
1      8       6
------------------------------------------------------------

$      DATA    I*
$      SELECT  UMC/DEFAULTS
$      SELECT  UMC/CHANGES
$      ENDJOB
```

SDLD Standard DEFAULTS File

Notes 8-10 of "Loading Integrated Software for Execution" (earlier in this section) illustrate a load of all available domains for standard DEFAULTS and CHANGES directives using the SI4.2 virtual domain.

The CHANGES directives file normally contains replacements or additions to the DEFAULT directives to form a loader task block used for domain loading. The CHANGES directives replace matching DEFAULTS directives or add the task block entries when a match is not found.

The resulting task block directives are used in the domain load. An internal line number is assigned to each CHANGE and DEFAULT directive. The Load Map report displays the internal line number for each task block directive to provide reference to the DEFAULTS or CHANGES line number.

OPNSUTIL/SHARD FILE REQUIREMENT FOR CONSOLE SPAWN OF "SHARD"

The OPNSUTIL/SHARD file needs the following JCL statement to support the "SPAWN SHARD" console request:

```
               1
1      8       6
------------------------------------------------------------

$      SELECTD SYS_SOFTWARE/yyyy/JCL/SHARD.SPWN
```

where yyyy is the Software Release identifier, such as 3000.

SDLD Reports

SDLD output is Report Code 01 on the SYSOUT file and contains four sections:

1. The directives
2. The virtual domain load-map
3. Buffer Pool directives
4. The loader summary

All sections have a page header line containing:

- o The SNUMB assigned to the SDLD activity
- o The date and time of the SDLD execution
- o The report title - "INTEGRATED SOFTWARE DOMAIN LOADER"
- o The report type - Directives, Load Map, Summary, or BPDirectives
- o SDLD version (8SI4.2) and the SDLD assembly date
- o Report/page number

DIRECTIVES LIST

The Directives list is divided into DEFAULT and CHANGE Sections, with the directives listed in each block as assigned by their internal line number, which is carried with the directive into the load task block. During the domain load, each directive used is displayed in the Load Map and the respective DEFAULT or CHANGE line number to ease cross-referencing to the input directive.

The Directives list shows the exact content of each input line and the line number assigned to that line. Syntax errors are shown as they occur.

LOAD MAP LIST

The Load Map list consists of a domain Load Map for each domain loaded. Each domain Load Map block contains:

- o The domain subtitle line with the domain-id
- o The INCLUDE_DOMAIN directive for the domain-id
- o The ALLOCATE -LIBRARY directive used to load the domain-id
- o The domain memory allocation values
    - SEG address
    - ISR descriptor
    - Initialization entry
    - Patch Table Space
    - STI offset
    - STI value
    - Software protection notice

o Module Information

  - Module name
  - Module number
  - Load offset
  - Assembly date
  - Title date
  - Transmittal logs

o Patch directives applied

o Remarks directives with domain-id in columns 73-78.

o Fatal or non-fatal error messages

LOAD MAP SUMMARY

The Load Map Summary Section is produced after the last domain has been loaded, and contains the status for the following items:

*

o Number of fatal errors

o Number of non-fatal errors

o Number of domains loaded

o Number of pages required to load shared software

o Number of buffer pool commands

o Number of buffer pool command errors

o Maximum wired pages allowed

o Shared software <loaded successfully> or <aborted>

o ASSIGN_SHARED_NAME <name>

BACKDOOR REPORTS

The "ALLOCATE -BACKDOOR" directive is required if the user wants a Backdoor Load Map report. This report duplicates the SDLD report code 01 and is released to SYSOUT at the end of the domain-load sequence. Report Code 01 is not available until the SDLD activity is aborted. SDLD normally does not terminate.

SDLD Error And Warning Messages

SDLD issues fatal and conditionally fatal (warning) error messages. Errors are controlled as follows:

- o When possible, the loading procedure continues until all directives are processed (the first fatal error does not stop the load process).
- o At the end of directive processing, if any fatal errors were encountered, processing terminates.
- o The Load Map Summary displays the count of fatal and non-fatal errors encountered.

FATAL ERRORS

MESSAGE: *** FATAL ERROR *** SYNTAX ERROR <reason>

CAUSE: Any directive syntax error is treated as fatal. The erroneous field (<reason>) indicates where the error occurred.

MESSAGE: *** FATAL ERROR *** SYNTAX ERROR -- DEFAULT/CHANGE CONFLICT

CAUSE: The START and END directives do not match. The correct sequence is:

```
START_DEFAULTS
    .
    .
END_DEFAULTS
START_CHANGES
    .
    .
END_CHANGES
```

NOTE: The CHANGE block is optional. The DEFAULT block is required.

MESSAGE: *** FATAL ERROR *** <PATCH statement processing>

CAUSE: Any PATCH statement error causes processing to abort; check all fields for:

- Invalid PATCH value
- Long field
- Non-octal characters (8 or 9)
- PATCH location out of bounds
- Invalid domain
- Non-applied patches; i.e., the domain needing to be patched was not loaded.

CONDITIONALLY FATAL ERRORS

The abort-control <abt-ctl> field determines if an error message is fatal or non-fatal by the use of the -REQUIRED/-OPTIONAL conditions. The following conditional errors are FATAL when issued in conjunction with a -REQUIRED option and NON-FATAL when issued in conjunction with an -OPTIONAL option:

MESSAGE: *** <error-type> ERROR *** <error message detected by file system>

CAUSE: Allocate directive condition incorrect.

MESSAGE: *** <error-type> error *** domain <DOMAIN-ID> NOT FOUND IN LIBRARY <lc>

CAUSE: INCLUDE_DOMAIN directive contains incorrect value(s).

NOTE: All error messages contain the term "FATAL" to provide an easy method to find loader errors using the TSS JOUT scan facility.

Example:

If the SDLD activity aborts, the scan output shows all errors and includes them in the Summary Report, as follows:

```
JOUT SNUMB
SCAN 01 D Y
PS:/FATAL/;*
```

SDLD Abort Codes

The SDLD abort codes are documented in the GCOS 8 OS Programmer Messages And Abort Codes manual.

MODIFYING INTEGRATED SOFTWARE ERROR MESSAGES

Integrated Software error messages that are issued to an Execution Report reside within the .MUTOR domain. These messages can be modified to revise message text and/or change the language in which message text is issued (e.g., from English to French).

Message modifications can be effected using the following procedure.

1. Review the existing messages. The following Time Sharing System command can be executed to generate a listing of all messages.

   ```
   JRN  SYS_SOFTWARE/yyyy/JCL/UTOR.CMPL
   ```

   where yyyy = Software Release identifier, such as 3000.

2. Create a file on which installation-specific message modifications will reside.

3. Prepare $ ALTER statements identifying all message modifications. Incorporate these statements into the file created in step 2.

   Note that the content of this alter file can be modified and applied to the .MUTOR domain when subsequent system software releases are installed.

4. Incorporate a $$SELECT statement that identifies the pathname for the alter file created in step 2 into the JCL residing on the file pointed to by pathname,

   SYS_SOFTWARE/yyyy/JCL/UTOR.CMPL

   where yyyy = Software Release identifier, such as 3000.

   Modify the $ IDENT statement to site specifications. Store the resulting JCL file at a site-created pathname.

5. Execute the following Time Sharing System command to rebuild the .MUTOR domain.

   JRN fffff

   where fffff is the pathname of the file created in the last step.

The next time GCOS 8 system software is booted with "RESTART NO", installation-specific message modifications will be issued to Execution Reports.

In conjunction with message modification, the following may prove beneficial:

- o Review the Load Map generated the next time Integrated Software (SNUMB of SHARD) executes. Verify that the UTOR module shows the assembly date on which the message modifications were bound in.

- o Retain the alter file prepared in steps 2 and 3 to use in rebuilding the UTOR module when a subsequent GRR is restored.

SECTION 5

STARTUP JOB STREAM

The Startup job stream comprises six sections in the following fixed sequence:

$CONFIG
$INITIALIZE
$EDIT
$FILES
$PATCH
$LOAD

Each of the six sections serves a distinct purpose. For example, one section defines the site's hardware configuration, while another section applies patch corrections to GCOS programs. The purpose of each section and the control statements required and permitted in each section are described below.

If a section is not in the proper position, the console message *EXPECT s...s FIX TYPE GO, OR TYPE STOP is issued (where s...s = name of the section that is out of sequence). The operator can enter BOOT via the system console to halt the bootload, place the section in the proper position, and initiate a warm boot.

Only two of the sections ($CONFIG and $FILES) are required to be processed. For various reasons, processing of the other sections may be bypassed. For example, processing of the $INITIALIZE and $EDIT sections can be bypassed if the operator enters NO in response to the *INITIALIZE? and *EDIT? questions when issued to the system console; the $PATCH section can be omitted if there are no patches to apply via this section; and the $LOAD section is required only to load qualifying modules into main memory.

Each section is delimited by a section identification statement and a ***EOF statement. The section identification statement specifies the section name, which begins in column 1 (e.g., $CONFIG). Following the ***EOF statement delimiting the last section in the Startup job stream, another ***EOF statement must be included to denote the end of the Startup job stream.

A section identification statement and ***EOF statement are not required if a section is not included in the Startup job stream.

Duplicate statements are invalid and can cause the Startup program to abort. In other instances, the duplicate statements are ignored by Startup and identified via a console message.

COMMENTS IN STARTUP JOB STREAM

Comments can be included on any statements, including section identification statements, in the Startup job stream. Any comment included on a ***EOF statement is ignored. At least one blank space must separate a comment from Startup data. Comments cannot be continued to another statement. A statement with ** in character positions 1 and 2 contains nothing but a comment.

Comments are printed on a dump report when a dump is requested. In addition, quotation marks (" ") cause the enclosed comment to be printed at the system console. For example, the comment on each of the following statements is printed at the console when the statement is processed:

```
               1
1      8       6
------------------------------------------------------------
$CONFIG  "DEFINING MS0450 SYSTEM "
  ...
$       INIT    ST1,DS1  "DESTROY PERM FILES?"
```

An exception to printing comments at the console is when the operator responds NO to either the *INITIALIZE? or the *EDIT? question. If the operator responds NO to *INITIALIZE?, only comments on the $INITIALIZE section identification statement are printed. Comments on all other statements in the $INITIALIZE section are ignored. If the operator responds NO to *EDIT?, only comments on the $EDIT statement are printed. Comments on all other statements in the $EDIT section are ignored.

RULES FOR NOTATION

The following rules apply to the notations used in the descriptions of the Startup statements:

1. Parameters enclosed in brackets, [ ], are optional and may be included or omitted, as required by the user. Optional fields can appear in any sequence following the required fields unless a requirement for a specific sequence is stated in the description of the statement.

2. When parameters are enclosed in braces, { }, one, and only one, of the enclosed parameters <u>must</u> be chosen.

3. An ellipsis, ... , indicates that the preceding parameter may be repeated.

4. The vertical bar "|" indicates that a choice among the parameters separated by vertical bars must be made.

5. All text printed entirely in uppercase letters must be typed as is, unless the portion of the general form containing it is itself optional.

6. Any spaces, semicolons, commas, dashes, parentheses, etc., shown in the general form are required punctuation.

$ ETC STATEMENT

In most cases, data can be continued from a Startup statement onto a $ ETC statement. (Those instances in which a $ ETC statement cannot be used are identified in the descriptions of the statements, later in this section.) A comma must terminate the statement preceding each $ ETC statement. The $ ETC statement must reflect the format of the statement that it is continuing. For example, a $ IOX statement might be continued as follows:

```
                1
1     8         6
-------------------------------------------------
$     IOX       CH-14,MS0450,UNITS-2,
$     ETC       UNIT-1,MS1,
$     ETC       UNIT-2,MS2
```

NOTE: In this example, the use of "CH-14" to assign channel 14 is equivalent to "PUB-14". Both are valid designations and are interchangeable.

$CONFIG SECTION

The $CONFIG section defines system software and system hardware parameters. The $CONFIG section is one of two sections required in the Startup job stream (the other is the $FILES section). Control statements in this section define device connections, device characteristics, shared mass storage devices, and the maximum hardware configuration. This section is preceded by MASK statements and followed by the $INITIALIZE section in the Startup job stream.

Internal configuration and device control tables are derived from $CONFIG section information. When a device rollcall is performed during the initialization portion of the bootload, connections are verified and the status of peripheral devices is determined. Inoperable devices at System Startup time are released, but the entries remain within internal tables so that the devices may be restored (i.e., assigned) to the system at a later time.

Startup also performs a survey of processors and initializes .CRCMC for each processor during $CONFIG section processing. (Refer to "Processor Release And Assignment At Startup" in Section 3 for additional information and to the GCOS 8 OS System Tables manual for the .CRCMC format.)

Startup issues the following questions, described in Section 2 under "Startup-Operator Interface", to the system console during $CONFIG section processing:

- *RETAIN CONFIGURATION?
- *RESTART?
- *SCF CONTINUATION?
- *DATE?
- *TIME?
- *DATE mmddyy TIME hh.mmm
- *CHANGE DATE?
- *CHANGE TIME?
- *ENTER TIME OR ENTER A CORRECTION FACTOR?
- *CHANGE > 5 MIN. YES TO ACCEPT?

System configurations vary greatly depending upon the choice of configuration options in the $CONFIG section. Peripheral device options permit the system to be tailored to the needs of the site. For this reason, it is recommended that preparing the $CONFIG section be closely coordinated, so that the configuration described in the Startup job stream accurately reflects the physical configuration of devices.

The following are requirements for the $CONFIG section control statement sequence:

1. Several options on the $ INFO statement (e.g., MIXED, RLSDSK, RLSPNT) and the $ BASE statement must precede the $ MCT and any input/output controller (i.e., $ IOM/IOX/IOP/IMU/IMX) statements.

2. The $ MCT/$ CIU statement must precede all input/output controller statements.

3. The input/output controller statements must precede $ XBAR statements, which, in turn, must precede the $ MPC statement.

4. The $ IMAGE statement(s) must precede any $ TRAINS statement(s).

5. The $ GCOSFIL statement must follow the mass storage $ Iyy statement(s).

The following statement types, described later in this section, are accepted by the $CONFIG section of Startup for DPS 8, DPS 8000, DPS 88, and DPS 90 systems, unless a system type (DPS xx) follows the statement type. In that case, the statement is valid only in the system(s) indicated.

$ ANSWER	$ IMAGE	$ MPC
$ AUTOLD	$ IMU (DPS 8)	$ MPCFIG
$ BASE	$ IMX (DPS 8000)	$ SECURE
$ CHAN	$ INFO	$ SHARED
$ CIU (DPS 88, DPS 8000)	$ IOM (DPS 8)	$ SYID
$ DECKFIL	$ IOP (DPS 90)	$ TRACE
$ DUP	$ IOX (DPS 88)	$ TRAINS
$ FILCREA	$ LOADFIL	$ UNIT
$ GCOSFIL	$ MCT (DPS 8, DPS 90)	$ URP
$ GROUP		$ XBAR

$ ANSWER Statement

The $ ANSWER statement allows site personnel to provide answers, in the Startup job stream, to questions that otherwise are directed to the system console during System Startup or dump operations. Any System Startup or dump questions with answers defined via the $ ANSWER statement are not issued to the console. Conversely, questions without predefined answers are issued to the console and require operator response via a console entry.

The $ ANSWER statement normally is used to expedite warm and fast boots (i.e., to circumvent console questions requiring operator responses).

Answers to System Startup and dump questions are specified in the format qqqqqqq/aaaaa (where qqqqqqq = parameter denoting a specific question, and aaaaa = answer). The question parameters and related answers can be specified in any sequence (e.g., the SSCLEAR parameter and answer can precede the RESTART parameter and answer).

Answers to multiple questions can be defined on one $ ANSWER statement, using the comma as a separator. If one statement does not accommodate all responses, information can be continued onto additional $ ANSWER statements or onto $ ETC statements. Some sites have found that one question parameter and its related answer per $ ANSWER statement simplifies maintenance of the Startup job stream when only one answer is to be changed or removed.

If at least one $ ANSWER statement is encountered, Startup issues the *CHANGE? question to the console during the bootload sequence. If the operator responds YES, the $ ANSWER statements are ignored for that bootload.

The format for the $ ANSWER statement is:

```
              1
1      8      6
--------------------------------------------------------------
$      ANSWER qqqqqqq/aaaaa,...
```

Tables 5-1, 5-2, and 5-3 indicate which System Startup and dump questions can be answered on the $ ANSWER statement, along with the format for the parameter denoting the question and the format for the answer. (Refer to "Startup-Operator Interface" in Section 2 for additional information on system Startup questions.) Table 5-2 indicates additional Startup questions for which answers can be defined; however, because these questions are issued only under special circumstances, $ ANSWER should not be used for these questions.

It is not possible to guarantee that a mass storage device will be fully initialized when using $ ANSWER statements to answer Startup questions.

Table 5-1. Frequently Used $ ANSWER Responses To Startup Questions

Startup Question To Be Answered	$ ANSWER Parameter (qqqqqqq/aaaaa)	
*RETAIN CONFIGURATION?	RETCONFIG/YES RETCONFIG/NO	
*RESTART?	RESTART/YES RESTART/NO	(See note 1.)
*SCF CONTINUATION?	SCF/YES SCF/NO	
*TIME?	TIME/NO	(See note 2.)
*INITIALIZE?	INITIALIZE/YES INITIALIZE/TOTAL INITIALIZE/PARTIAL INITIALIZE/NO	
*EDIT?	EDIT/YES EDIT/PARTIAL EDIT/NO	
*SYSTEM SCHEDULER CLEAR?	SSCLEAR/YES SSCLEAR/NO	(See note 1.)
*PROCESSOR ON PORT p DOESN'T ANSWER CONNECT	PROCERROR/YES	(See note 3.)
*SYSOUT RECOVERY?	SYSOTRCVY/YES SYSOTRCVY/NO	(See note 4.)

NOTES:

1. If the *RESTART? question is answered RESTART/YES on the $ ANSWER statement, Startup automatically answers the *SYSTEM SCHEDULER CLEAR? question NO.

2. Once the time has been established via an operator entry, Startup continually updates the time. Use of this option ensures that the most recent time saved by Startup is used to answer the *TIME? question.

3. A PROCERROR/YES response indicates that a warm or fast boot is desired following release of the slave processor. A PROCERROR/NO response aborts the Startup program and is invalid on the $ ANSWER statement. (Refer to "Processor Release And Assignment At Startup", in Section 3 for additional information.)

4. A SYSOTRCVY/YES answer allows for recovery of system output regardless of answers to RESTART questions.

Table 5-2. Special $ ANSWER Responses To Startup Questions

Startup Question To Be Answered	$ ANSWER Parameter (qqqqqqq/aaaaa)
*FORMAT/LABEL QUESTIONS?	LABEL/YES LABEL/NO
*FORMAT/LABEL QUESTIONS?	FORMAT/YES FORMAT/NO
*INIT DEVICE ddd?	INIT/YES INIT/SECONDARY INIT/NO
*INIT SHARED DEVICE ddd?	SINIT/YES SINIT/SECONDARY SINIT/NO
*INIT RMVBL DEVICE ddd?	RINIT/YES RINIT/SECONDARY RINIT/NO
MPC ON icc PRE-INITIALIZE?	INITDAU/YES INITDAU/NO
SHARED MPC ON icc PRE-INITIALIZE?	INITSHRDAU/YES INITSHRDAU/NO
PROCESS DIRECTORY CARD FOR DEVICE ddd?	DIRECT/YES DIRECT/NO
PROCESS DIRECTORY CARD FOR SHARED DEVICE ddd?	SDIRECT/YES SDIRECT/NO
*SHARED MPC on iccc APPEARS OK, BOOTLOAD?	SBOOTLOAD/YES SBOOTLOAD/NO
*xxxMPC ON iccc BOOTLOAD SYS ID NAME "y...y" REV.zz?	BOOTLOAD/YES BOOTLOAD/NO
*xxxMPC ON iccc APPEARS OK, BOOTLOAD?	BOOTLOAD/YES BOOTLOAD/NO
*CLEAR AND OVERWRITE EXISTING DECKFILE? (FIRMWARE)	DECKCLEAR/YES DECKCLEAR/NO
*Iyy-n NOT RESPONDING - YES TO CONTINUE OR R TO RETRY	IOMERROR/YES IOMERROR/NO

NOTE: The SYSTEMMAP/NO response inhibits printing of a system map and a file map. There is no console message offering this option.

Table 5-3. $ ANSWER Responses To Dump Questions

Dump Question To Be Answered	$ ANSWER Parameter (qqqqqqq/aaaaa)
SELECT DEVICE: (PRINT), (DISK), PRINTER NAME (PR1), (DDD/TAPE#,DENSITY), (NONE)	DUMPON/PRINT DUMPON/(ttt/rrrr,dddd) DUMPON/DISK DUMPON/NONE
ADDITIONAL OPTIONS: ALL,HCM,SOFT,SLV,W/XX, P/XX,REAL,NPURE,NONE,DEVICE,DUMPO,(-)SD.XXX, .MXXXX,PH.XXX,PSH,SSF,HIS,TRC,COM,PTW,ONLY, EXEC,FLTKPX,CACHE,SDC,AUTO,PAED,RES,END	DUMP/ALL DUMP/HCM DUMP/SOFT DUMP/SLV DUMP/Wxxx-yyy (W/xxx-yyy) DUMP/ppp (P/ppp) DUMP/REAL DUMP/NPURE DUMP/NON DUMP/OPTS DUMP/SD.sss DUMP/.Mmmmm DUMP/PH.sss DUMP/PSH DUMP/SSF DUMP/HIS DUMP/TRC DUMP/COM DUMP/PTW DUMP/ONLY DUMP/SDC DUMP/EXEC DUMP/FLTKPX DUMP/AUTO DUMP/PAED DUMP/RES DUMP/CACHE
*BOOT SOURCE: AUTO OR REPL?	BOOT/AUTO BOOT/REPL

*

where:

ttt - Logical device name of the magnetic tape handler
rrrr - Tape reel number
dddd - Tape density
xxx-yyy - Dump working spaces
ppp - Dump known process index
sss - Segment identification
mmmm - Module identifier

The following example specifies that job restart (RESTART/YES) is desired, that system restart is to occur from the Startup job stream residing on the AUTOLOAD file (BOOT/AUTO), that statistical collection is to resume at the point of interruption (SCF/YES), that neither the initialization nor the edit function is desired (INITIALIZE/NO and EDIT/NO), that the SSFILE is not to be cleared (SSCLEAR/NO), that a warm or fast boot is desired following release of the slave processor (PROCERROR/YES), and to use the most recent time saved by Startup to answer the *TIME? question (TIME/NO).

```
               1
1      8       6
------------------------------------------------------------------
$      ANSWER  RESTART/YES,BOOT/AUTO,SCF/YES
$      ANSWER  INITIALIZE/NO,EDIT/NO,SSCLEAR/NO,PROCERROR/YES,TIME/NO
```

$ AUTOLD Statement

The $ AUTOLD statement specifies the permanent mass storage device on which the AUTOLOAD file is to reside and defines the file size. (This device is referred to as the AUTOLOAD device.) The Startup job stream is written to the AUTOLOAD file, and used during subsequent warm and fast boots.

If the $ AUTOLD statement does not specify a device, the AUTOLOAD file is written to the first nondedicated mass storage device that is defined on a $ GCOSFIL statement. Note that Startup also writes the content of the file specified on the $ DECKFIL statement to the AUTOLOAD device.

The format for the $ AUTOLD statement is:

```
               1
1      8       6
------------------------------------------------------------------
$      AUTOLD  ddd,n
```

where:

ddd - Logical device name of the mass storage device on which the AUTOLOAD file is to reside. For device types requiring unit numbers of the form UNIT-uu.s, where s is the subunit number, the AUTOLOAD file must be assigned to a logical device for which s is zero.

n - Size (in llinks) of the AUTOLOAD file. The default size and minimum size is 600 llinks.

The following example shows the AUTOLOAD file specified to reside on mass storage device DS3, with the default size of 600 llinks.

```
               1
1      8       6
------------------------------------------------------------------
$      AUTOLD  DS3
```

$ BASE Statement

The $ BASE statement defines the maximum site mainframe configuration and, thus, the amount of memory to be allocated in memory-resident tables for input/output controllers and network processors.

The configuration panel switches or configuration file must be set to reflect the number of input/output controllers and network processors specified on the $ BASE statement. These settings apply regardless of the number of input/output controllers, network processors, and CPUs configured on the system. This allows mainframe components to be added or deleted without changes to either the Startup job stream or mailbox switches.

The $ BASE statement, when supplied, must precede the $ MCT/$ CIU statement. If it is not supplied, memory is allocated in memory-resident tables for the number of input/output controllers and network processors described in the $CONFIG section of the Startup job stream.

The format for the $ BASE statement is:

```
               1
1      8       6
-----------------------------------------------------------------
$      BASE    {IMU-x},DN-y
               {IMX-x}
               {IOM-x}
               {IOP-x}
               {IOX-x}
```

where:

x - Maximum number of input/output controllers used by the site.

y - Maximum number of network processors for DNET/ROUT mode plus any configured On Line Page Processing Systems used by the site (exclude Network Processors for CXI mode). The value y can range from 0-16 and should be equal to n in the FNP/n parameter of the $ INFO statement. There is no default value.

The following example specifies that memory is to be allocated in memory-resident tables for four IOMs and four network processors. Note that the same type of statement can be used in a DPS 88 environment by specifying IOX-2 (maximum of 2 IOXs) instead of IOM-4, and in a DPS 90 environment by specifying IOP-4 instead of IOM-4.

```
               1
1      8       6
-----------------------------------------------------------------
$      BASE    IOM-4,DN-4
```

$ CHAN Statement

The $ CHAN statement permits peripheral devices to be switched from one channel to another during Startup by specifying another channel for the device. Any number of $ CHAN statements can be included in the Startup job stream.

The format for the $ CHAN statement is:

```
               1
1      8       6
------------------------------------------------------------------------
$      CHAN    Iyy-a,{PUB-b},Iyy-c,{PUB-d},message
                     {CH-b }       {CH-d }
```

where:

```
Iyy-a,{PUB-b} - A channel previously configured on a $ Iyy or $ XBAR
      {CH-b }   statement (Iyy = IMU, IMX, IOM, IOP, or IOX).

Iyy-c,{PUB-d} - A channel not previously configured.
      {CH-d }
```

If Iyy-a has been released by Startup, all references to Iyy-a,PUB/CH-b are changed to reference Iyy-c,PUB/CH-d (assuming they have not been released). The system operates as though the devices configured on Iyy-a,PUB/CH-b have been configured on Iyy-c,PUB/CH-d.

The user specified "message" in the control statement is issued to the system console if Iyy-a has been released and Iyy-c has not been released. If Iyy-a has not been released or if both Iyys have been released, the control statement is ignored.

The example below indicates that PUB-20 on IOM-2 has been previously configured and that PUB-16 on IOM-3 has not been previously configured.

```
               1
1      8       6
------------------------------------------------------------------------
$      CHAN    IOM-2,PUB-20,IOM-3,PUB-16,NOTIFY SITE OPS
```

If IOM-2 is released during Startup, the following message is displayed on the system console:

```
NOTIFY SITE OPS
```

$ CIU Statement

For the DPS 88 the $ CIU statement defines the memory size, number of processors, and IOXs connected to the CIU. For the DPS 8000 the $ CIU statement defines the memory size, number of processors, and IMXs connected to the SCU. The $ CIU statement must precede all $ IOX or $ IMX statements. Only one $ CIU statement may be specified. For DPS 8 and DPS 90, see "$ MCT Statement", later in this section. The $ MCT statement cannot be used for DPS 88 and DPS 8000.

The fields of the $ CIU statement can be arranged in any order; however, CPUs and IOXs or IMXs must be configured contiguously, beginning with 0.

```
                 1
1       8        6
_____________________________________________________________

$       CIU     [MEM/sss,]PRO-n,{IMX-m|IOX-m}
```

where:

MEM/sss - The number of blocks of 1K words of memory configured into the system. The maximum value is the total available memory size; however, it need not be set to that value.

If this field is omitted, the available memory is determined via a survey of reserved memory by the SSF or service processor.

PRO-n - Required for every CPU configured into the system (maximum = 4). CPUs must be configured contiguously, beginning with 0.

IMX-m
IOX-m - Required for every IMX or IOX configured into the system. IMXs or IOXs must be configured contiguously, beginning with 0.

The following example defines a system with 2048 1K blocks of memory, two CPUs, and two IOXs:

```
                 1
1       8        6
_____________________________________________________________

$       CIU      PRO-0,PRO-1,IOX-0,IOX-1,MEM/2048
```

The minimum memory size required to support GCOS 8 is 4MB. If less than this is specified, Startup will issue the following error message and abort:

GCOS-8 REQUIRES AT LEAST 1024K OF MEMORY

$ DECKFIL Statement

The $ DECKFIL statement specifies the name of the file to which the MPC Bootload (MPCB) and firmware data decks are written. The file also contains the MPCC deck (table), which is used to pass data to the MOLTS subsystem of the Total Online Test System (TOLTS).

The deckfile is created on the same permanent mass storage device that contains the AUTOLOAD file (refer to "$ AUTOLD Statement", earlier in this section). The size of the deckfile, which also is specified on this statement, is dependent upon the size of the MPCB, firmware data, and MPCC decks. For this reason, it is recommended that site personnel verify the file size with the site's Customer Service Division (CSD) representative.

If the $ DECKFIL statement is not included in the Startup job stream, a file with the name DECKFILE is created by default.

The deckfile can be rebuilt during a warm or cold boot. The operator must respond YES or PARTIAL to the *INITIALIZE? question and YES to the CLEAR AND OVERWRITE EXISTING DECKFILE? question during the bootload procedure. (Refer to "*INITIALIZE? Question" and to "CLEAR AND OVERWRITE EXISTING DECKFILE? Question" in Section 2 for additional information.)

The format for the $ DECKFIL statement is:

```
              1
1      8      6
_____________________________________________________________________

$      DECKFIL fffff,n[/O]
```

where:

fffff - Name of the deckfile (maximum of 12 characters), the first four characters of which must be "DECK".

n - Size (in llinks) of the deckfile. There is no default size; therefore, this field must be defined.

/O - Overwrite any existing deckfile (letter O).

The following example allocates the deckfile named DECKFILE 250 llinks of space on the permanent mass storage device containing the AUTOLOAD file. Any existing deckfile contents are to be overwritten (/O).

```
              1
1      8      6
_____________________________________________________________________

$      DECKFIL DECKFILE,250/O
```

NOTE: To install the Integrated Firmware And Diagnostic (IFAD) tape, a corresponding DECKFILE Extension File must be created, and a $ PFILES statement must be included in the $EDIT and $FILES sections of the Startup job stream. The DECKFILE Extension File uses the same name as the DECKFILE, except replace "DECK" with "TAND" (e.g., "DECKFILE" becomes "TANDFILE").

IFAD - integrated firmware & diagnostic

$ DUP Statement

The $ DUP statement specifies two removable disk devices which are to be associated with each other as primary and secondary mirrored disk devices. The $ DUP statement has the following format:

```
              1
1     8       6
-------------------------------------------------------------

$     DUP     ddd,Iyy-m,{CH-p },{UNIT-uu   }
                        {PUB-p} {UNIT-uu.s}
```

where:

ddd - Logical device name of the primary device.

m - Number of the Iyy to which the secondary device and channel (CH/PUB) are connected. The value of m can be 0-3. Iyy = IMU, IMX, IOM, IOP, or IOX.

p - Iyy channel (CH/PUB number to which the secondary device is connected).

uu
uu.s - Logical device identification number of the secondary device. The subunit portion (.s) of this field is only valid for some device types. (Refer to Table 5-4, later in this section, for the valid form for each device type.)

Both devices (primary and secondary) of a pair must be defined via $ Iyy statements before using the $ DUP statement to establish the pairing. The secondary device must be configured with the device name of SPARE.

$ FILCREA Statement

The $ FILCREA statement creates the FASTBT file used by the fast boot feature. The $ FILCREA statement has the following format:

```
              1
1     8       6
-------------------------------------------------------------

$     FILCREA FASTBT,fffff
```

where:

fffff - Name of the fast boot disk reload file (maximum of 12 characters).

$ GCOSFIL Statement

The $ GCOSFIL statement identifies all devices on which files, identified in the $CONFIG section and the $EDIT section, can reside. When searching for $CONFIG section and $EDIT section files, Startup checks only devices identified on the $ GCOSFIL statement(s).

The format for the $ GCOSFIL statement follows.

```
                 1
1      8         6
-------------------------------------------------------------
$      GCOSFIL   ddd[,ddd]...
```

where:

ddd - Logical device name. A maximum of 63 devices can be identified.

The following example indicates that system files are permitted on devices DP3, DP4, DP5, and DP6. Other devices in the configuration (e.g., DP2, DP7, and DP8) are reserved for other uses. For example, one device may be reserved for recovery purposes, another may be defined as removable (RMVBL) - system files cannot reside on removable devices - and another may be reserved for user data base purposes.

```
                 1
1      8         6
-------------------------------------------------------------
$      GCOSFIL   ST1,DP3,DP4,DP5,DP6
```

$ GROUP Statement

The $ GROUP statement identifies the physical channels of the IMU/IMX/IOP/IOX to which an MSU3380, MSU3381, MSS8080, or MTS8200 subsystem is physically connected. The $ GROUP statement also identifies the logical channels associated with those physical channels. The $ GROUP statement is required for MSU3380, MSU3381, MSS8080, and MTS8200 subsystems and is optional for other devices. A separate $ GROUP statement is required for each subsystem. The $ GROUP statement has the following format:

```
                 1
1      8         6
-------------------------------------------------------------
$      GROUP     Iyy-m,{PUB-p-q}[,{PUB-p-q}]...[,
                       {CH-p-q }[ {CH-p-q }]   [

$      ETC       Iyy-n,{PUB-r-s}[,{PUB-r-s}]...]...
                       {CH-r-s }[ {CH-r-s }]   ]
```

where:

Iyy - IMU, IMX, IOP, or IOX.

m,n - Number (0-3) of the Iyy to which the MSU3380, MSU3381, MSS8080, or MTS8200 subsystem is connected.

p,q,r,s - Iyy channel (CH/PUB) number. Permissible values are:

IMU	8-63
IMX	0-127
IOP	8-63
IOX	0-127

Channel number p (and r) must be the primary channel. Channel number q (and s) must be the last logical channel number in the group to which the subsystem is connected. Logical channels for the same physical channel are specified using the hyphen (-) as a separator indicating consecutive channels. Channels on a different subsystem are specified using another $ GROUP statement. Physical channels to a subsystem are specified using a comma as a separator on the $ GROUP statement. Each physical channel number on $ Iyy and $ XBAR statements for MSU3380, MSU3381, MSS8080, and MTS8200 must be specified on a $ GROUP statement.

Example:

```
              1
1      8      6
_________________________________________________________
$      GROUP  IMU-1,CH-40-43,CH-50-53
```

Channels 40 and 50 are physical channels; channels 41, 42, 43, 51, 52, and 53 are logical channels.

$ IMAGE Statement

The $ IMAGE statements provide a logical representation of the characters on a physical belt. Each position in the image corresponds to the graphic character on the belt at that position. Placement of a particular internal code in that position on the image causes that internal code to print as the graphic character at that position on the belt. The image defined on $ IMAGE statements replaces the standard ASCII or BCD image. The $ IMAGE statement defines the image of a printer belt to replace the default image associated with a specific printer type. The default images are defined in an internal buffer of the Startup program.

In addition, the first six 64-word sectors of the PRINTIMAGE file, containing the print train catalog and the VFC catalog, are overwritten with information from the $ IMAGE statement. Overwriting of the catalogs on the PRINTIMAGE file destroys existing file content.

If Startup detects a difference in the content of the PRINTIMAGE file and the $ IMAGE statements, the following error message will result:

```
*CHECKSUM ERROR, FILE PRINTIMAGE
* ANSWER YES TO IGNORE ERROR AND CONTINUE
* ANSWER NO TO ABORT, & THEN EDIT PRINTIMAGE.
```

If the file contained any images other than standard printer belt and vertical format control (VFC) images, the TVIM program must be used to rebuild the images on the file. (Refer to "Printer Belt Characters/Codes" in the GCOS 8 OS System Operating Techniques manual for PRINTIMAGE file information and for procedures to use the TVIM program.)

Default printer belt images can be restored to both the internal Startup buffer and the PRINTIMAGE file during a bootload from punched cards or magnetic tape. Re-edit the PRINTIMAGE file by defining the file on a $ FILDEF statement in the $EDIT section of the Startup job stream and responding YES to the *EDIT? question during the bootload procedure.

All $ IMAGE statements must precede the $ TRAINS statement (or its related $ ETC statement) in the Startup job stream. If the $ IMAGE statements follow the $ TRAINS statement, Startup aborts. (Refer to the $ TRAINS, $ Iyy, and $ FILDEF statements in this section for additional printer-related information.)

Each printer belt has 240 character positions to be defined. Multiple $ IMAGE statements are used (15 for ASCII belts, 5 for BCD belts) for this purpose. Continue the printer belt definition from one $ IMAGE statement to another. Do not use a $ ETC statement to continue the definition. Startup aborts if a $ ETC statement is used for this purpose.

Startup also aborts if information in the printer belt identification number (nnnn) or $ IMAGE statement sequence number (s) field is incorrect.

The format for the $ IMAGE statement is:

```
               1
1      8       6
________________________________________

$      IMAGE   ppp,ccc,nnnn,s,x...x
```

where:

ppp - Printer type number:

4nn

where nn = any numerical value

ccc - Printer belt being replaced:

BCD - Replacing standard BCD printer belt
ASC - Replacing standard ASCII printer belt

nnnn - Printer belt identification number (octal) of the standard belt being replaced. The software identification of any belt is obtained by converting the belt number (given in decimal) into its octal equivalent (e.g., for the PRB0500 belt, 0500 (decimal) = 764 (octal), so the software identification is 764). For PR54 printer belts the first digit (3) should be dropped before conversion (e.g., PRB3600 is converted by dropping the 3, and then converting 600 (decimal) to 1130 (octal)).

PRB Number (decimal)	(octal)
0500 or 3500	764
0600 or 3600	1130

s - $ IMAGE statement sequence number, one or two digits, starting with 1 (or 01) and incrementing by 1, up to 15 for ASCII belts or 5 for BCD belts. Within a range of sequence numbers (1-15 or 1-5), Startup will abort if the ppp, ccc, and nnnn fields do not remain constant.

x...x - When ccc = ASC this field defines a sequence of 16 of the 240 character positions of the printer belt replacing the standard printer belt for this device. Each position is specified by three octal digits.

When ccc = BCD this field defines a sequence of 48 of the 240 character positions of the printer belt replacing the standard printer belt for this device. Each position is specified by one BCD character.

The following example shows the fifteen statements required to define a nonstandard ASCII printer belt for PRU1200/1600 printers:

```
              1          2                                             7
1     8       6          9                                             9
------------------------------------------------------------------------
$     IMAGE   400,ASC,1130,01,102104106107110112147123060101103105111116117122
$     IMAGE   400,ASC,1130,02,114124141142143144145146113150151152153154155115
$     IMAGE   400,ASC,1130,03,060120121125126127130131132072073077041045067170
$     IMAGE   400,ASC,1130,04,171156157160162163164167060172061062063064065066
$     IMAGE   400,ASC,1130,05,043070071056054050051100044075046057047134136137
$     IMAGE   400,ASC,1130,06,060042074076133135147123176101103105111116117122
$     IMAGE   400,ASC,1130,07,173124141142143144145146060150151152153154155175
$     IMAGE   400,ASC,1130,08,174140055052053161165166102104106107110112067170
$     IMAGE   400,ASC,1130,09,060156157160162163164167114172061062063064065066
$     IMAGE   400,ASC,1130,10,113070071056054050051115060120121125126127130131
$     IMAGE   400,ASC,1130,11,132072073077041045147123171101103105111116117122
$     IMAGE   400,ASC,1130,12,060124141142143144145146043150151152153154155100
$     IMAGE   400,ASC,1130,13,044075046057047134136137060042074076133135067170
$     IMAGE   400,ASC,1130,14,176156157160162163164167173172061062063064065066
$     IMAGE   400,ASC,1130,15,060070071056054050051175174140055052053161165166
```

The following example shows the five statements necessary to define the nonstandard BCD print belt, PRB 3513/0513. (Refer to the GCOS 8 OS System Operating Techniques manual).

```
                  1           2                                      7
1     8           6           9                                      9
-----------------------------------------------------------------------
$     IMAGE       400,BCD,0764,01,=5678GHIJKL\0.-*/MNOPQR:&(9),STUVWX[01234YZ$@#+!
$     IMAGE       400,BCD,0764,02,?5678ABCDEF^0.-+/GHIJKL<>(9),MNOPQR[01234STUVWX;
$     IMAGE       400,BCD,0764,03,'5678YZ$@#%_0.-*/ABCDEF"=(9),GHIJKL\01234MNOPQR:
$     IMAGE       400,BCD,0764,04,&5678STUVWX[0.-*/YZ$@#+!?(9),ABCDEF^01234GHIJKL<
$     IMAGE       400,BCD,0764,05,>5678MNOPQR[0.-*/STUVWX;'(9),YZ$@#%_01234ABCDEF"
```

$ INFO Statement

The $ INFO statement defines a variety of operating system parameters. Multiple parameters can be included on one $ INFO statement, or they can be continued onto as many $ INFO or $ ETC statements as required.

The format for the $ INFO statement is:

```
                  1
1     8           6
------------------------------------------------------------
$     INFO        parameter[,parameter]...
```

The following parameters can be included on the $ INFO statement(s):

$ INFO Parameter	Description
ACALL/ {nn } {TSS} {TPE} {TSx}	Automatically execute an ACALL instruction following a restart via fast boot. The associated parameters indicate that the specified network processor is to be reconnected (where nn = processor number 0-16) and/or the Time Sharing System (TSS parameter) or the Transaction Processing System Executive (TPE parameter) is to be restarted (if either was executing at the time the system malfunctioned) upon system restart. The TSx parameter (where x = 1-4) identifies the copy of Time Sharing that is to be restarted in a multicopy configuration. For example, TS2 indicates that only the second copy of Time Sharing is to be restarted if a system failure occurs. More than one TSx parameter can be specified; each must be preceded by ACALL/ (e.g., ACALL/TS1, ACALL/TS2). If more than one parameter is specified, ACALL must be included for each (e.g., $ INFO ACALL/2,ACALL/TSS).

$ INFO Parameter	Description
ASCII	A printer with ASCII capabilities (i.e., with an ASCII printer belt) is configured and can be allocated to print ASCII output. NOTE: This is not a SYSOUT parameter. If online ASCII printer output is collected by SYSOUT, periodic attempts are made to allocate an ASCII printer (MGEOT INFO (025) -- ASCII PRINTER NEEDED). This continues until an ASCII printer becomes available, or until the output is disposed of via REDRC (REDRC APRONL BPRONL), NOPRO or other SYSOUT console verb.
CATDUP	Duplicate all File Management Supervisor (FMS) catalogs. File content is not duplicated. This parameter ensures file availability in the event of a hardware failure on the device containing the original catalog structure. The following conditions apply to this parameter: 1. Original and duplicate catalogs are written to mass storage devices for which the CAT parameter is specified on the $ INIT statement. If the CAT parameter is not specified, Startup assigns the original and duplicate FMS catalogs to the same devices on which the original and duplicate System Master Catalogs (SMCs) reside. Startup ensures that the original and duplicate catalogs are not on the same device. 2. The CATDUP parameter is not activated unless the SMCDUP parameter also is specified on the $ INFO statement. If the CATDUP parameter is specified and the SMCDUP parameter is not, Startup aborts.
CATFNC/n	Startup defined files will be allocated at or above llink number "n"(decimal) on devices containing FILSYS catalogs. The purpose is to preserve the lower address range for FILSYS catalog growth. The minimum value of n = 0; the maximum value of n = 51200 (decimal); the default value of n = 30000. A message: *UPPER LIMIT FOR CATALOG FENCE IS 51200 is issued indicating that the value for n entered exceeded the allowable maximum for the $ INFO CATFNC/n statement.

$ INFO Parameter	Description
CCACHE/nnn	Main memory buffers are reserved in support of the FMS catalog cache feature. The value nnn can range from 0-400 (0 is equivalent to not specifying the parameter). Each buffer is 320 words. NOTE: This is a Priced Software Product that requires KEY3 to be loaded.
CHKSUM	Checksum all modules during the module-load process. If not specified, modules are not checksummed.
CLENPT	Enable the cleanpoint function for system restart. The cleanpoint facility attempts to restart jobs which were in process at the time of system failure at a predefined point (i.e., at a cleanpoint). If cleanpoint recovery is unsuccessful, restart is from the beginning of each job that was in process. Recovery/restart is vitally important to system output. For this reason, and to increase reliability, the segments shown in the following list are cleanpointed if the $ INFO CLENPT statement is included in the $CONFIG section. The segments are: DP.JOT - Job Output Table DP.JOS - Job Output Specials DP.SBT - SYSOUT Blink Table DP.BDI - Backdoor Information DP.JOT and DP.SBT cleanpoints are taken after a new job, which has completed queue entry, is processed and when blinks and jobs are released. DP.JOT and DP.BDI cleanpoints are taken when a backdoor job queue entry is processed and when backdoor files and jobs are released. DP.JOS cleanpoints are taken each time a special is added to or removed from the current segment. Also, DP.SBT is cleanpointed in BRT6 upon activity termination if the job modified that segment. NOTES: Cleanpoint cannot be used on a single pack system. If a $ INFO SMCDUP is also specified, the cleanpoint control file is also duplicated on another device to be used in the event the primary control file on ST1 is destroyed.
CNTDCW	This parameter causes the number of words transferred by an activity when doing disk I/O to be counted and collected with other accounting information. Enabling of this option may adversely affect I/O overhead constraints.

$ INFO Parameter	Description
DENxx/DENyy	The default high and low densities for the site's MPC-driven magnetic tape subsystem are specified by the values xx and yy, respectively. If the option is not specified, default high and low densities for NRZI devices are 800 bpi and 556 bpi, respectively. Note that if a tape density is not specified on the $ ACCOUNT or $ FILDEF statements, the default high density of 800 bpi is used. The following density specifications are valid: DEN2 - 200 bpi DEN5 - 556 bpi DEN8 - 800 bpi DEN16 - 1600 bpi DEN62 - 6250 bpi If PE (1600 bpi) and GCR (6250 bpi) subsystems/devices are configured, a $ INFO statement must be included in the Startup job stream to activate the default-density feature. Both parts of the option contain the same numeric identifier in these cases (e.g., DEN16/DEN16 for PE devices). If default densities are not specified and if no other subsystem/device is configured with the standard (800/556) default density capabilities, Startup aborts. The console message NO HANDLERS CAPABLE OF DEFAULT HIGH/LOW DENSITY is issued.
DPSE	This parameter must be present if a DPS 8/50 or 8/70 processor is configured. It allows the operator to dynamically assign (i.e., place online) the processor, if it is currently offline, via the ASGNP console entry.
ENCRYP	Activate password encryption so that SMC passwords entered via FILSYS directives CRMAST and MODMAS will be encoded. This security feature protects against unauthorized examination and disclosure of passwords. Immediately before the first installation of password encryption, the site must run the encryption utility program ENCRYP. (Refer to GCOS 8 OS System Operating Techniques manual.) As soon as password encryption has been installed for the first time, the MODMAS file created by the ENCRYP utility must be entered into the system via a $ FILSYS activity. Once the ENCRYP option has been activated, the $ INFO ENCRYP statement must be included in the $CONFIG section for all subsequent Startups.

$ INFO Parameter	Description
EXTTDS/xxx,yyy	This parameter defines the amount of memory available to the Transaction Driven System (TDS) or to DM-IV/TP when Extended Memory is utilized. The value xxx specifies the number of auxiliary instruction segments assigned to TDS and the value yyy specifies the maximum number of pages assigned. The maximum limit on xxx is 61. There is no maximum limit on yyy. There is no default for xxx. The default for yyy is 64.
EXTTSS/xxx,yyy[,zzz]	This parameter defines the amount of memory available to the Time Sharing System. The value xxx specifies the number of auxiliary instruction segments assigned to Time Sharing and the value yyy specifies the maximum number of pages assigned. The maximum limit on xxx is 61 (61 is the maximum capacity of a page table). There is no maximum limit on yyy. There is no default for xxx. The default for yyy is 64. (The value zzz is a literal assigned to the value of the work space.) THIS IS A REQUIRED STATEMENT -- SYSTEM WILL ABORT IF THE $ INFO EXTTSS STATEMENT IS NOT INCLUDED AND TIME SHARING IS CONFIGURED ON THE SYSTEM.
FASTBT	Enable the fast boot function for automatic system restart. This parameter is equivalent to the FASTBT ON console entry. If the FASTBT option is specified, rollcall messages are suppressed. However, operator interface messages are suppressed during the Startup question-and-answer sequence only if $ ANSWER statements are present in the Startup job stream.
FMSCT/n	The value n specifies the maximum percentage (%) of the FMS catalog area of a CAT DISK device that may be used for file content in FMS file create and restore. The value of n may range from 0 to 100. The default value is 100.
FMSDT/n	The value n specifies a descriptor threshold (maximum number of space descriptors) for a selected disk device in FMS file create and restore. When the threshold is exceeded, a different device is selected. The value of n may range from 0 to 127. A value of 0 indicates an unlimited number of descriptors. The default value is 8.
FMSST/n	The value n specifies a space threshold (% of size of the selected device) for file content in FMS file create and restore. When the threshold is exceeded a different device is selected. The value of n may range from 1 to 100. The default value is 50.

$ INFO Parameter	Description
FMSTAT	Accumulate statistics for File Management Supervisor (FMS) program functions (e.g., the number of times a function is used, the number of I/O operations for a function, and the average amount of time required to perform a function).
FNP/n	The value n identifies the number of network processors for DNET/ROUT mode (exclude network processors for CXI mode and include any configured On Line Page Processing Systems) configured on an input/output controller. The $ INFO...FNP/n statement directs Startup to establish the Network Processor mailbox descriptor and is required if any Network Processors for DNET/ROUT mode and/or On Line Page Processing Systems are configured. This statement must precede the $ MCT or $ CIU statement. The value n can range from 0-16 and should be equal to y in the DN-y parameter of the $ BASE statement. There is no default value.
GENSYS	Generalized Tape Management System (GTMS). The system is to execute with tape management enabled. Module .MKEY7 (SPS element), and the files 'GSYSDATABASE' and 'GSYSAJ' must be edited onto the system. If this parameter is not specified, GTMS cannot be invoked, even with the presence of the .MKEY7 module. This parameter sets bit 30 in .CROPT and may be referenced by the symbol .FGSYS.
GEOTRC[/n]	Enable GEOT trace for more thorough SYSOUT delivery trouble analysis. The optional octal mask, which limits enabling to selected traces, is normally omitted unless requested by a Honeywell Bull CSD Representative.
* HEX/ {ON } {OFF}	The HEX/ON option results in acceptance of MME GMODES requests for setting HEX floating point mode. This option is valid only if all online processors have hexadecimal capability. The console message HEX MODE ENABLED is issued if Startup encounters a $ INFO...HEX/ON statement and determines that all online processors have hexadecimal capability. The console message: *NO HEX CAPABILITY ON CPU-n HEX OPTION DISABLED is issued if a $ HEX/ON statement is included in the Startup job stream, but processor n is incapable of running in HEX mode. As a result, the HEX option is disabled while Startup continues processing. The HEX/OFF option results in denial of MME GMODES requests to enter HEX mode. HEX/OFF is the default.

$ INFO Parameter	Description
INTERRUPT/xxx	This parameter selects the interrupt processing type for multiprocessor systems. xxx - ALL allows all processors.to receive interrupts - ONE selects the control processor to receive interrupts (default for DPS 8 and DPS 88) For DPS 90 systems this parameter is not valid, and ONE is used. For DPS 8000 systems this parameter is not valid, and ALL is used. If this parameter is present on a DPS 90 or a DPS 8000 system, an error message is given and Startup attempts to process the remainder of the $CONFIG section, after which Startup aborts.

Native mode memory management parameters KLFTH, KLMIW, KMMWP, KLMWS, KLWPR, and KLWSW can be varied to obtain better performance for different situations. However, the system is effectively self-adjusting for a variation of load. Except for very extreme cases, optimal performance will be obtained without varying these parameters, because the adjustment mechanisms are sufficient to cope with considerable load variations. Refer to the GCOS 8 OS System Operating Techniques manual for more information.

KLFTH/n	This native mode memory management parameter designates the number of pages for the system free-chain low threshold. The value n is placed in the upper half of .KLFTH. The default value is 18.
KLMIW/ssss	This native mode memory management parameter designates minimum window time for a shared workspace in 16-microsecond units. The value ssss is placed in .KLMIW. The default value is 1875.
KLMWP/ssss	This native mode memory management parameter designates maximum window time for a process workspace in 16-microsecond units. The value ssss is placed in .KLMWP. The default value is 625000.
KLMWS/ssss	This native mode memory management parameter designates maximum window time for a shared workspace in 16-microsecond units. The value ssss is placed in .KLMWS. The default value is 2097152.
KLWPR/xxxx	This native mode memory management parameter designates the initial number of missing page faults which will constitute a window for a process workspace and can be considered as the "Page Fault Window Delimiter". The value xxxx will be placed in the upper half of .KLWPR. The default value is 35.

$ INFO Parameter	Description
KLWSW/xxxx	This native mode memory management parameter designates the initial number of missing page faults which will constitute a window for a shared workspace and can be considered as the "Page Fault Window Delimiter". The value xxxx will be placed in the upper half of .KLWSW. The default value is 35.
LIMITS	When a program exceeds its SYSOUT record or processor time limit, rather than aborting, the operator is asked: S#sssss aa SYSOUT LINES EXHAUSTED CONTINUE OR ABORT C/A? or S#sssss aa RUN TIME EXHAUSTED CONTINUE OR ABORT C/A? where: sssss - Snumb. aa - Activity number. When the operator response is "C", the job is granted 10,000 additional lines of SYSOUT, or another 5 minutes of processor time, and the job continues processing. When the operator response is A, the job is aborted with the appropriate abort code.
LINES/n	The value n establishes the maximum number of lines per page of output. This value must be a minimum of 6. The default number of lines per page is 55.
LOGON	The $ INFO LOGON parameter sets appropriate flags in system software to recognize that LOGON may be operational. If TS8 is configured (via the $ INFO TS8 statement), and this statement is not included, GCOS 8 system initialization proceeds. However, the following error message is displayed at the console when the $LOAD section is processed. (TS8 is also deconfigured as a result of this error.) TS8 - COMMON LOGON REQUIRED ($ INFO LOGON), CAN'T CONFIGURE TS8
MCPTSS/n	The value n specifies the configured number of Time Sharing System copies. This value can range from 1-4. The default value is 1.

$ INFO Parameter	Description
MEMORY/n	The value n specifies the sieve limit for the maximum memory size allowed for any one activity in a job. The number, n, is specified in 1024(1K) blocks. There is no maximum value for n (other than the constraint imposed by the system's memory size). The default value is 511K if this parameter is not specified. The operator can modify the value n via the console entry LIMIT SIZE nK.
MIGRAT	Specifies that Startup is to attempt to migrate SR 4/JS3 Startup-defined files for use with a GCOS 8 release. The following message is given on the system console to indicate successful migration of a file: 4JS3 FILE filename MIGRATED No message is given if a file cannot be migrated. Migrated files cannot be released or grown. NOTE: When executing 4/JS3-DPS1.3 software release and software release 2000, 2300, or 2500 mixed systems, individual disk packs containing Startup defined files must be initialized via 4/JS3-DPS1.3 .MINIT, because no procedure exists in software releases 2000, 2300, or 2500 to remove the space contained in the 4/JS3-DPS1.3 Startup master catalog. Initializing the disk under 4/JS3-DPS1.3 clears the pointer to the SR 2000/2300/2500 Startup master catalog. Examples of such mixed operating systems include: o Migration testing when the site is using unique files for each release o Production environments when the site is cycling between releases This restriction applies whether or not the $ INFO MIGRAT option or unique file names are used.
MIXED	CPL and NPL peripheral device types coexist within the same configuration for a single-system. If a device is unavailable for allocation, the MIXED parameter allows substituting its equivalent device type to satisfy the request (e.g., if a CPL PRT401 printer is unavailable for allocation, its NPL counterpart (PRU1200) could be allocated). * *

$ INFO Parameter	Description
MODULE/.mmmmm	One or more modules (.mmmmm) can be specified to reside in main memory, thereby avoiding the system overhead resulting from the module-load function. For example, several FMS modules are high-use modules and loading these modules into main memory via the MODULE/.mmmmm option can prove beneficial. Modules that are not reentrant are loaded in the Hard Core Monitor (HCM). A memory-to-memory transfer is then executed when the module is called. Multiple modules can be specified on one $ INFO statement by repeating the MODULE/.mmmmm option for each module; or each module can be identified on a separate $ INFO statement.
MODULE/.MSDCB	Same as above; required for use of Rapid Access Data System (RADS). This parameter activates all RADS functions. RADS remains dormant until the .MSDCC control program utilizes the cache memory and issues an enable request. If .MSDCB is unknown to the system, RADS is inoperable.
NDFMAP	The deckfile map is not to be printed. If this parameter is not specified, the deckfile map is printed.
NIAST/ {ALL/n } {RMVBL/n} RADS- Rapid Access Data System	No In-memory Available Space Table. For each removable (RMVBL) mass storage device on which a structured removable disk pack is mounted, and for each permanent device, several tables are used to maintain control of disk space allocation to files. The device allocation unit table maintains a current record of whether or not each allocation unit (AU) is allocated for permanent file usage. For each device, this table is maintained on the device itself. The system builds an available space table (AST) from the device allocation unit table information whenever the operator requests an initialization or secondary initialization. This same action is also triggered automatically whenever one of several circumstances occur (described under "Other NIAST Considerations" below). The system maintains the AST with a record of whether or not each AU is allocated for file use (including both permanent and temporary file use, without distinction). The temporary file space considered is only the space allocated since the AST was built.

$ INFO Parameter	Description

NOTE: The NIAST option is implied when the $ SHARED Startup statement is specified. Devices that are connected to shared mass storage controllers and not specified as removable can be shared with the Network Processing Supervisor (NPS).

The parameters available for use with the NIAST option are:

ALL - Specifies maintaining the AST on the device, rather than in memory, for each removable device on which a structured removable disk pack is mounted and for each permanent device.

RMVBL - Specifies maintaining the AST on the device for each removable mass storage device on which a structured removable disk pack is mounted. (Typically, removable devices have less space allocation activity than permanent devices. Therefore, for some sites, the best memory and I/O utilization may occur by using the RMVBL option. Resource trade-off is discussed in detail below.)

n - Specifies the number of I/O buffers required to read from or write to the ASTs. The default value reserves one buffer for every four NIAST devices.

Other NIAST Considerations

The AST resides on the device when NIAST is specified. (Although the maintained AST is on the device, the site sets the number of memory buffers (n) used to manage the entire group of ASTs for all devices operating in NIAST mode.) In addition to the AST and the device allocation unit table, a third table, the temporary descriptor table (TDT), resides on each permanent device when NIAST is specified. The TDT is rebuilt automatically whenever the AST is rebuilt. The availability of the ASTs and TDTs on the devices enables reallocation of a temporary file after a system bootload (e.g., for restart purposes), unless the device is initialized or given a secondary initialization during the bootload.

$ INFO Parameter	Description

Because the AST resides on the device when NIAST is specified, the AST can be read and modified by sources external to the system that built it. This mode is invoked automatically when the $ SHARED statement specifies that the device is shared with a network processor using the Network Processor Supervisor (NPS) software. The NIAST option enables the system and network processor to have direct access to a common, current source of space allocation information located on the device.

The AST resides in memory when NIAST is not specified or implied (i.e., neither $ SHARED nor $ INFO NIAST are specified in the Startup job stream). When the NIAST option is not in effect, the AST is rebuilt automatically from the device allocation unit table whenever the system is bootloaded. Thus, reallocation of a temporary file is not guaranteed after a system bootload for a device when NIAST is not specified. As a result, jobs may need to be reentered to reestablish file space and data previously resident on temporary files. (If a job is unsuccessful in its attempt at such a reallocation, it is deleted from the system. The operator is then notified.)

The following circumstances cause the system to rebuild the device's AST:

1. When NIAST is specified, the AST is automatically rebuilt on permanent mass storage devices whenever the system is bootloaded and the device was last operating without the NIAST option activated.

2. When NIAST is not specified, the AST is rebuilt automatically in memory for permanent mass storage devices whenever the system is bootloaded.

3. In either case, the AST is rebuilt automatically for removable mass storage devices whenever a structured disk pack is mounted at Startup, or if it is mounted later, the AST rebuilding occurs during system operation.

The site's decision to use the NIAST option involves the following trade-off considerations as to the availability of memory versus the need for I/O resources (except when $ SHARED is used, which automatically invokes the NIAST option).

1. The NIAST option requires less memory because there are fewer memory buffers than there are devices.

$ INFO Parameter	Description
	2. When NIAST is not specified, memory space is required for each AST. Also, increased Startup time for I/O activity results from generating the in-memory AST whenever the system is bootloaded. 3. The NIAST parameter requires more run-time I/O activity, because, unless it is already present in one of the I/O buffers, the AST must be read into memory each time it is modified. The "Mass Storage Operations" section in the GCOS 8 OS System Operating Techniques manual contains instructional information on this subject.
NPCHSZ/nnn	This parameter enables the application of virtual mode patches from the Startup job stream $PATCH section. The value nnn specifies the size (in words) of the system segment holding the patch card images. A value of 256 is recommended; this may be increased or decreased, depending on site patch requirements. There is no default for this parameter. If a size is not specified, a segment of zero size is provided. Virtual mode patches are in GCL-style format, which is a free form format beginning with a PATCH_RU directive and followed by a positional parameter and keyword parameters. If a virtual mode patch included in the $PATCH section of the Startup job stream is in error and is not applied, Startup gives an error message sequence to flag this condition. For example: erroneous patch statement image ** PATCH_RUN_UNIT COMMAND HAS INCORRECT COMMAND SYNTAX. erroneous patch statement image ** KEYWORDS ARE REQUIRED FOR THE PATCH COMMAND. erroneous patch statement image ** A PATCH VALUE IS REQUIRED FOR A PATCH DIRECTIVE. erroneous patch statement image ** THERE IS NOT ENOUGH INFORMATION TO PERFORM PATCH. Report the occurrence of any unapplied patches to the Honeywell Bull Response Center.
NPRINT	No printer is available to the Startup program. Any print routines encountered during System Startup are ignored and no output is generated. If a printer becomes inoperable during a dump operation, the operator can respond SKIP to the message DUMP nnn FROM ddd TO TAPE (where nnn = file name and ddd = logical device name of a mass storage device), which permits the system to be rebooted with the NPRINT option active.

$ INFO Parameter	Description
	This parameter prevents Startup from doing any printing, to prevent destroying preprinted forms which may be mounted in printers following a system crash. If Startup aborts, or is directed to abort via the DUMP command, and this option is in effect, Startup will ask for a printer via the NEED PRINTER message. If no printer is available, press the EOM push button on the system console and the dump will be skipped (lost).
NPUNCH	No card punch is configured for local punch output. If a CCU0401 card reader/card punch is configured, the NPUNCH option makes the card punch unusable but permits use of the card reader. If the NPUNCH parameter is specified, bit 2 is set in .CROPT to indicate that no card punch is configured. If an activity includes a $ PUNCH statement, the Peripheral Allocator program deletes the job. NPUNCH instructs SYSOUT to collect local card punch output (PNCONL), without delivering it. Such output remains in SYSOUT space until the system is rebooted with "SYSOUT RECOVERY=NO", or until one of the following console verbs is applied to it: NOPRO, PURGE, or REDRC (e.g., NOPRO PNCONL).
NSYASC	No SYOT ASCII Space Compression. ASCII print line records (media codes 7 and 15) are normally edited by SYOT to reduce disk storage requirements. Editing consists of replacing sequences of three or more spaces with an equivalent two-character, unit-separator control sequence. The NSYASC option disables ASCII space compression.
NSYBSC	No SYOT BCD Space Compression. BCD print line records (media codes 3 and 11) are similarly edited by replacing sequences of nine or more spaces with an equivalent two-character escape sequence. As with ASCII space compression, the escape sequence is recognized by the printer hardware (or network processor software). It is therefore unnecessary to expand compressed records at the time they are printed. The NSYBSC option disables BCD space compression.

$ INFO Parameter	Description
NSYSAV	No SYOT Seek Address Validation. SYOT normally validates the seek address for every write operation it initiates. This is accomplished by mapping the referenced link number for the write back to the blink table to ensure correct ownership of the blink. If an error is detected, the job aborts after taking the following SNAPS: 1 The .MSYOT module itself (with trace table) 2 Slave being serviced (or at least the GFRC buffer) 3 Slave's collection control segment (PH.SYT) 4 Slave's SSA segment (PH.SSA) 5 GCOS communication region segment (SD.CR) 6 SYSOUT blink table segment (SD.SBT) 7 SYSOUT collection area segment (SD.SCA) The NSYSAV parameter disables the seek address validation procedure.
NSYTRC	No SYOT TRaCe. SYOT normally maintains a circular trace of the last 100 key processing events which took place. The table is an invaluable aid for problem analysis purposes but, of necessity, requires some processor and memory resources to maintain. For the benefit of sites which feel this overhead is prohibitive, the NSYTRC parameter may be employed to disable the trace mechanism.
OWNID/xxxxxxxxxxxx	Default value for FIPS-79 Owner-id. Fills the communication region locations, beginning with .CROID, with up to 12 ASCII characters used by FIPS-79 UFAS tape labels. The first blank character encountered terminates the string.
PASSWD/nnn	Password expiration interval. The value nnn specifies the maximum number of days for which a user password can remain valid, and ranges from 1 through 365. This statement is optional; if it is included, it applies to the Time Sharing System (TSS) as well as TS8, and keeps the password expiration interval consistent between the two time sharing systems. When the interval is exceeded, a user trying to log on is prompted to change the password before being allowed to continue. If the value for nnn is zero, or if this statement is not included, there is no expiration interval for user passwords (i.e., passwords do not have to be changed).

$ INFO Parameter	Description
	If the value for nnn is nonnumeric, the following message is output to the system console, and Startup aborts:

```
*CONFIG CARD BELOW CONTAINS AN ERROR NEAR COLUMN 23.
$      INFO    PASSWD/xxx
```

where xxx is the nonnumeric value.

If nnn is greater than 365, the following message is output to the system console:

```
VALUE FOR PASSWORD EXPIRATION INTERVAL > MAXIMUM
```

Startup continues, using the maximum of 365 as a default.

PROEXT

Process Extension. Required to enable the Six-Processor functionality. When this statement is included, Startup expects the processor number to be in bits 9-11 of the Fault Base Switches. This parameter is not valid on DPS 8000, DPS 88, and DPS 90 systems.

PSUM/nnn

The value nnn specifies the maximum number of processes to be allowed (cataloged) in the system.

The value nnn must be in the range of 63 to 487. These values are applied by Startup if less than the minimum (e.g., 62) or more than the maximum (e.g., 488) is specified. The value nnn must be an odd-numbered value (e.g., 65 or 67). If it is not, the specified number plus 1 is substituted by Startup (e.g., 66 becomes 67). The default value of nnn is 127. A maximum of 487 processes, including the number of TS8 users who will be concurrently logged on, can be cataloged.

$ INFO Parameter	Description
PT1SIZ/nnnn	The value nnnn specifies the size of the page table for working space 1. The default size of this page table is 2048 words, which is sufficient for most software configurations. When the default is not sufficient, it will be detected and reported by POPM immediately after it receives control from Startup. When the condition is detected, the normal START ROLL CALL message to the operator's console will be followed by ***ERR ***ERR WORK SPACE 1 PAGE TABLE IS TOO SMALL. ***ERR USE $ INFO PT1SIZ/nnnn AND REBOOT. ***ERR where nnnn is the needed page table size, in words, as determined by POPM. The operator should do a warm boot using the AUTOLOAD file for input, and using the Startup console editor to add the $ INFO PT1SIZ/nnnn.
PTCHSZ/nnn	The value nnn specifies the maximum number of patches that can be included in the $PATCH section. The default value is 256. It is recommended that consideration be given to subsequent additions of patches via the system console when establishing the patch limit. If the number of patches included in the $PATCH section exceeds the number specified with the $ INFO PTCHSZ parameter, Startup accepts only the number of patches specified with the PTCHSZ parameter. All patches exceeding this number are ignored and the message: TOO MANY PATCHES USE $ INFO PTCHSZ/N is issued to the system console.

$ INFO Parameter	Description

RESVID/FORMS,id1[,id2,...,idn]/

The RESVID/FORMS parameter provides a means to reserve certain station ids for use as collection nodes for special forms. No remote connection will be allowed to these reserved ids, which will be class supported. At least one id must be specified if the parameter is used; a maximum of 512 ids may be declared. Each id must consist of a two-character identifier. The identifiers 00, space-space, and !! are not permitted. A space is not permitted as part of an identifier. Segment DP.RID is allocated for the reserved id list and is organized as follows:

```
Word 0:   TALLY 1,n (n = number of ids)
Word 1-n: VFD 2/1,22/0,H12/xx (xx=BCI id)
```

RESVID/{PPSOFL},id1[,id2,...,idn]/
{PPSONL}

The RESVID/PPSOFL and RESVID/PPSONL parameters provide a means to reserve certain station ids for use as collection nodes for the Page Processing System (PPS) in offline and online modes, respectively. No remote connection will be allowed to these reserved ids, which will be class supported. At least one id must be specified if the parameter is used; a maximum of 512 ids may be declared. Each id must consist of a two-character identifier. The identifiers 00, space-space, and !! are not permitted. A space is not permitted as part of an identifier.

PPSOFL - Required parameter to enable offline PPS functionality; either the offline or online PPS software must be available to the system when PPSOFL is specified. Any output for PPS reserved ids is spooled to tape for subsequent input to the Page Processor.

PPSONL - Required parameter to enable both online and offline functionality; the online PPS software must be available to the system when PPSONL is specified. User jobs may assign output to the reserved ids for subsequent transfer directly to the Page Processor, as well as spooling to a PPS tape.

Destination id(s) for the PPS correspond to remote station id(s) used for standard remote output. They are reserved id(s) and are included in the reserved id table within memory.

$ INFO Parameter	Description
	The PPSOFL and PPSONL parameters and reserved id(s) on the $ INFO RESVID statement do not represent a one-to-one correspondence with Page Processors actually configured on a system. Two or more Page Processors may utilize the same reserved id(s) and one system may be configured online while another is configured offline.
	Multiple $ INFO RESVID statements can be used to assign more reserved ids than can be contained on one statement; however, the same processing mode (i.e., PPSOFL or PPSONL) must be specified on each statement.
RLSDSK	Startup is to release any mass storage devices that are offline during System Startup. If the released device contains any GCOS system files, Startup aborts. If the device is online at System Startup, it is not released.
RLSPNT	Startup is to release any printer that is offline during System Startup. If the released printer is that which is used by the Startup program, the NPRINT parameter is automatically invoked. A released printer can be reassigned to the system via the console entry ASGN icccdd (where i = input/output controller number, ccc = channel number, and dd = device number). If the device is online at System Startup, it is not released.
ROLLCALL/ddd	Rollcall messages, which are issued during System Startup, are directed to printer ddd instead of to the system console (where ddd = logical device name of the printer). Use of this parameter interrupts a fast boot process, which otherwise bypasses the rollcall function.

$ INFO Parameter	Description
SAVDMP/list	The list defines the set of default options used to control what is dumped to system-controlled space (i.e., a save dump) at the time a process (slave) dump is taken. The options specified on this statement are augmented with options specified on the SAVDMP= option of the $ DMPOPT JCL statement associated with the activity in which the dump is taken. The options are specified after the "/", and are separated by commas. The underlined option is the default: ALL7 - Dump the entire contents of dynamically allocatable space behind WSR7. ANY - Utilize default options, which means to dump OS control segments behind WSR7. <u>NODUMP</u> - Produce no save dump of any kind. If the site desires that the activity definer SSDUMP option be effective, or allow the use of the SAVDMP option on the $ DMPOPT JCL statement to be effective, a $ INFO SAVDMP option must be used to specify a value other than NODUMP. PSDUMP or W4DUMP - Dump the entire dynamically allocatable contents of WSR4 (private shared space). SSDUMP or W5DUMP - Dump the entire dynamically allocatable contents of WSR5 (integrated software space). This is the same as specifying SSDUMP on the activity definer. WSDUMP or W6DUMP - Dump the entire dynamically allocatable contents of WSR6 (workstation space). C4DUMP - Dump the OS control segments behind WSR4. C5DUMP - Dump the OS control segments behind WSR5. C6DUMP - Dump the OS control segments behind WSR6.
SCFPRG	The SCF purge-to-disk option is enabled, so that SCF data will be collected on permanent files instead of on magnetic tape. Refer to the <u>GCOS 8 OS Accounting</u> manual for details.

$ INFO Parameter	Description
SCHSAV	All job input is saved. This feature protects against the loss of jobs following a system malfunction. The duration of the save period is dependent upon System Scheduler program parameters (i.e., the size of the class catalog and the amount of processing activity within the job class). All saved jobs are placed in a temporary status (i.e., "saved"). Rerunning a saved job is initiated via the console entry JRERUN sssss (where sssss = SNUMB). If this parameter is active, the 15 oldest saved jobs that have completed execution are removed from the System Scheduler's SSFILE file as space for a class approaches exhaustion (i.e., approaches the class maximum as specified on the $ SSFILE statement in the $EDIT section). Errors within the job stream (i.e., JCL statements), which cause fatal processing errors, override the save function and job input is not saved.
SDCOL/n	The value n specifies the number of additional pages to be allocated to the SD.COL segment, which is used by IOS and ECOL to store and retrieve type 3 GEPR records. The default size of SD.COL is two pages. If three pages are desired, specify the value of n as 1. If four pages are desired, specify the value of n as 2. If the value of n is not 0, 1, or 2, the following error message is issued to the console: CONFIG CARD BELOW CONTAINS AN ERROR NEAR cc $ INFO SDCOL/n where cc is the JCL statement column number. *
SLAVE/n	The value n specifies the maximum size of the slave Instruction Segment (i.e., the largest slave job in execution) in 1K blocks. The default value is 255. Any value exceeding the default is truncated to 255.
SLINKS/n	The value n specifies the sieve limit for the number of mass storage links requested for any temporary file. There is no default value set.
SLTAPE/n	The value n specifies the sieve limit for the number of magnetic tape files requested by any activity in a job. There is no default value set.

$ INFO Parameter	Description
SLTIME/hhmm	The value hhmm specifies the maximum amount (i.e., sieve limit) of processor time that can be allocated to one job (where hh = hours and mm = hundredths of one hour). The maximum value that can be specified is 9999. The default time limit is 9.99 hours. The time specified must not be less than 0.01 hours.
SMCDUP/ddd	The System Master Catalog (SMC) is to be duplicated on mass storage device ddd (where ddd = logical device name). Device ddd must not be device ST1, which contains the SMC. If ST1 is defined with the SMCDUP parameter, Startup aborts. SMCDUP status can be added or deleted only when the operator responds YES to both *INITIALIZE? and *EDIT? during a bootload. If an attempt is made to change the status by adding or deleting the parameter to/from the $ INFO statement, except during a bootload sequence, Startup aborts. Startup also aborts if an attempt is made to assign the SMC and the duplicate SMC to two logical devices on the same physical unit.

SPN,umc_1[/cat-or-file-1],umc_2[/cat-or-file-2]

The system path name (SPN) replacement parameter replaces the actual UMC or UMC/cat-or-file specified in the first argument (umc_1/cat-or-file-1) with that specified in the second argument (umc_2/cat-or-file-2) whenever FMS accesses umc_1/cat-or-file-1. The rules for replacement follow:

1. SYS_CAT cannot be specified as umc_1 or umc_2.
2. Commas must separate SPN, the first argument, and the second argument.
3. umc_1 must either begin with SYS_ or be OPNSUTIL.
4. umc_2 must begin with SYS_ if umc_1 begins with SYS_, or umc_2 must be a replacment for a umc_1 of OPNSUTIL.
5. Default UMCs do not exist (i.e., umc_1 and umc_2 cannot be null).
6. If a slash delimiter follows umc_1 or umc_2, it must follow both, and a cat-or-file (either catalog name or quick access file name) must follow both.

$ INFO Parameter	Description
	7. The UMC/cat-or-file arguments are limited, as the format shows, to two levels. 8. Multiple replacement sets are allowed. Each replacement set must be contained on one $ INFO SPN statement (i.e., $ ETC is not supported with the SPN parameter). The specified replacements are made for all FMS accesses (e.g., JCL statements (including $ USERID statements), Time Sharing accesses, and FILSYS directives (including USERID but excluding RESTORE, SAVE, RESTOREMAST, and SAVEMAST)). Error messages that apply to the SPN parameter are listed below. The rules violated are denoted in parentheses (and are not part of the message). * SPN ERR SYS_CAT CANNOT BE CHANGED (1) * SPN ERR EXPECT , OR / (2) * SPN ERR "OLD" FIELD NOT FOLLOWED BY COMMA (2) * SPN ERR "NEW" NOT SYS_ OR REPL FOR OPNSUTIL (4) * SPN ERR BLANK USERID FIELD INVALID (5) * SPN ERR BLANK CATALOG, FIELD INVALID (5) * SPN ERR ONLY ONE SET PER INFO CARD (8)
SSAOVL/n	The value n specifies the number of real pages allocated to Slave Service Area (SSA) cache. The default value is 8. The minimum value is 4 and the maximum value is 64.
SYBDI/n	Size of backdoor information segment. The default value of 1024 allows for approximately 50 backdoor jobs in the system.
SYBRT/n	GEOT blink release threshold. This parameter defaults to a value of 20. This parameter should be used by a site wanting to discard output more rapidly after delivery is complete.
SYJOS/n	Size of job output special segment. The default value of 64 allows for 15 output specials. Note: because of the reduced use of the specials table, this should probably be sufficient for most sites.
SYJOT/n	Size of job output table segment (SNUML). The default value of 1024 allows for 146 jobs with one output type and destination. A reasonable value for sites requiring space for 500 jobs with more than one output type or destination is 4096.

$ INFO Parameter	Description
SYSOUT/n	The value n specifies the sieve limit for the maximum number of SYSOUT records allowed for the job. Only the job $ LIMITS statement is considered if it is present; otherwise, the value considered is the sum of the limits specified for the individual activities by a $ LIMITS statement, if present; or by the default value for the activity type. There is no default value set.
SYSTEMMAP/NO	Do not print the map of the GCOS file structure. The default is for this map to be printed. Inclusion of this parameter reduces the time required for starting the system.
TAPDEN/xxxx	The codes specified as xxxx summarize the total density capabilities of all configured magnetic tape handlers. The xxxx field overrides the system default value used when processing the DEN9 field of the $ TAPE control statement (see the GCOS 8 OS Job Control Language manual for further details). The xxxx parameter can include the following codes:

Code	Description
ABCD	200, 556, 800, 1600 bpi
ABC	200, 556, 800 bpi
BCD	556, 800, 1600 bpi
BC	556, 800 bpi
CD	800, 1600 bpi
DE	1600, 6250 bpi
D	1600 bpi
E	6250 bpi

TIMEZ/zzz

Universal date/time is automatically set from the local date/time entry by the operator when the system is booted. This is done from a time zone entry which must be included on the $ INFO TIMEZ statement. The $ INFO TIMEZ must be included in the Startup job stream. If this statement is not included, Startup will abort with the following message:

*MANDATORY $ INFO TIMEZ CARD WAS NOT FOUND

The value zzz specifies the local time zone identifier. The following list defines the time zone identifiers recognized by Startup:

$ INFO Parameter	Description

ZONE IDENT	OFFSET	DESCRIPTION
NT	-11	NOME TIME
AHST	-10	ALASKA-HAWAII STANDARD TIME
YST	-9	YUKON STANDARD TIME
PST	-8	PACIFIC STANDARD TIME
MST	-7	MOUNTAIN STANDARD TIME
PDT	-7	PACIFIC DAYLIGHT TIME
CST	-6	CENTRAL STANDARD TIME
MDT	-6	MOUNTAIN DAYLIGHT TIME
EST	-5	EASTERN STANDARD TIME
CDT	-5	CENTRAL DAYLIGHT TIME
AST	-4	ATLANTIC STANDARD TIME
EDT	-4	EASTERN DAYLIGHT TIME
NST	-3:30	NEWFOUNDLAND STANDARD TIME
GST	-3	GREENLAND STANDARD TIME
ADT	-3	ATLANTIC DAYLIGHT TIME
AT	-2	AZORES TIME
WAT	-1	WEST AFRICA TIME
UT	+0	UNIVERSAL TIME
Z	+0	ZERO ("ZULU")
GMT	+0	GREENWICH MEAN TIME
CET	+1	CENTRAL EUROPEAN TIME
MET	+1	MIDDLE EUROPE TIME
BST	+1	BRITISH SUMMER TIME
SWT	+1	SWEDISH WINTER TIME
FWT	+1	FRENCH WINTER TIME
EET	+2	EASTERN EUROPEAN TIME
SST	+2	SWEDISH SUMMER TIME
FST	+2	FRENCH SUMMER TIME
BT	+3	BAGDAD TIME
ZP4	+4	Universal time +4 HOURS
ZP5	+5	Universal time +5 HOURS
IST	+5:30	INDIAN STANDARD TIME
ZP6	+6	Universal time +6 HOURS
ZP7	+7	Universal time +7 HOURS
JT	+7:30	JAVA TIME
CCT	+8	CHINA COAST TIME
JST	+9	JAPAN STANDARD TIME
SAST	+9:30	SOUTH AUSTRALIA STANDARD TIME
ZP10	+10	Universal time +10 HOURS
ZP11	+11	Universal time +11 HOURS
NZT	+12	NEW ZEALAND TIME

TS8

This statement requires that TS8 module .MTSVC be edited onto the system. If this module is not present, system initialization proceeds but the following error message is displayed when the $LOAD section is processed:

$ INFO Parameter | Description

TS8 - REQUIRED MODULE .MTSVC NOT FOUND, CAN'T CONFIGURE TS8

The presence of this message indicates that the required module was not part of the tapes comprising the Software Release media. The Honeywell Bull Response Center should be contacted immediately.

USRDMP/list

The list defines the set of default options used to control what is dumped to the destination specified for the process (by default to the SYSOUT P* file) at the time a process (slave) dump is taken. The options specified on this statement are augmented with options specified on the USRDMP= option of the $ DMPOPT JCL statement associated with the activity in which the dump is taken. The options are specified after the "/", and are separated by commas. The underlined option is the default. Those marked with asterisks (*) apply only to processes with privity.

ALL7 - Along with active domains, dump the entire contents of dynamically allocatable space behind WSR7.

ANY - Utilize default options, which means for all active domains to dump OS control segments behind WSR7 and writeable segments behind other workspace registers 2-6.

* C4DUMP - Dump the OS control segments behind WSR4.

* C5DUMP - Dump the OS control segments behind WSR5.

* C6DUMP - Dump the OS control segments behind WSR6.

* INHIBD - Include domains whose property keys say to inhibit dump as active domains.

MINI - Produce a dump including only summary information, history registers, and the dump causing the safe-store frame.

NODUMP - Produce no user dump of any kind.

NOSHR - Do not dump segments in shared workspaces, even if they exist in active domains.

OSEXT - Dump OS control segments not normally useful in analyzing process dump problems. This includes most notably SYSOUT control segments.

$ INFO Parameter	Description
	* PSDUMP or * W4DUMP - Dump the entire dynamically allocatable contents of WSR4 (private shared space).
	SHRRW - Dump all segments in active domains, even if they are read-only.
	* SSDUMP or * W5DUMP - Dump the entire dynamically allocatable contents of WSR5 (integrated software space).
	* STACKS - Dump the content, including residual, of stack segments PH.ADS, PH.AP, and PH.SS.
	* WSDUMP or * W6DUMP - Dump the entire dynamically allocatable contents of WSR6 (workstation space).
	* WSR01 - Dump segments behind WSR0 and WSR1 for active domains.
WSSIZE/2,s2,b2 WSSIZE/3,s3,b3	These statements can be used to set the virtual sizes and backing store file coverage of system shared workspaces. The s2 value is ignored. This field is reserved for future use. The b2 value is used to specify the percentage of virtual memory for which backing store file space is to be allocated for the standard WSR2 (Workspace 4). The value is specified in percent, with 20% being the default. The s3 value is used to establish the size of the standard WSR3 (Workspace 10) virtual memory size, in megawords. The default is 1M. The b3 value is used to specify the percentage of virtual memory for which backing store file space is to be allocated for the standard WSR3 (Workspace 10). The value is specified in percent, with 50% being the default. The virtual size for WSR3 is taken from the s3 parameter. The b3 value can be omitted, even if the s3 value is specified.

Input/Output Controller Statements

The input/output controller statements ($ IMU, $ IMX, $ IOM, $ IOP, or $ IOX) are used to define the site's peripheral devices. Each peripheral that is assigned to a specific channel (CH/PUB) of an input/output controller must be identified. If a device is not specified on such a statement, the Startup program does not consider that device a part of the system configuration, even though the device may be referred to on other statements in the Startup job stream.

Once the system has been initialized, the operator cannot deconfigure a permanent device via a console entry or by removing the $ Iyy statement from the Startup job stream. However, a permanent device can be released via the console entry RLSE icccdd (where i = input/output controller number, ccc = channel number, and dd = device number). If a device is released, the Startup or FMS program may abort if required files and/or catalogs were assigned to the device.

If the mass storage hardware configuration is to change, all jobs in the system must first run to termination before the modification can be effected. After all jobs have terminated, the $ Iyy statement (images) can be modified to reflect the new configuration. During the bootload sequence, the operator must respond NO to the *RESTART? question and YES to the *SYSTEM SCHEDULER CLEAR? question.

The DEDICATED option can be used to reserve (restrict) use of the specific device to only those programs that request the device by its logical device name (ddd field). If the device is not requested by its logical device name, it is not allocated - even if the device is not busy at allocation time. It is not a good practice to dedicate ST1.

This feature frequently is used to avoid repeated printer form or magnetic tape reel changes. For example, a job that requires a special form can have its output directed via the $ PRINT JCL statement to a dedicated printer, which contains the special form.

The DEDICATED option cannot apply to the system console.

A device also can be dynamically dedicated via the console entry DEDCAT. The dedicated status - either Startup or console assigned - can be dynamically removed via the console entry UNDED. (Refer to the GCOS 8 OS System Operator Messages manual for a description of the DEDCAT and UNDED entries.)

In addition to the DEDICATED option, the following comments apply to the logical device name (ddd) field. A logical device name comprises three
* alphanumeric characters. The first character must be nonzero, the second character must be alphabetic, and the third character can be alphabetic or numeric.

With the exception of mass storage devices, which can have only one logical device name, each device can be assigned multiple logical device names. Each peripheral must have at least one logical device name and each logical name must be unique. A maximum of 2048 logical device names can be assigned via the Startup job stream. (Two additional names are reserved for assignment via the system console.)

Logical device names also can be assigned via the console entry NAME icccdd nnn (where i = input/output controller number, ccc = channel number, dd = device number, and nnn = logical device name). Logical device names also can be removed from the name table via the console entry UNAME nnn (where nnn = logical device name). Note, however, that neither of these entries can be used in conjunction with mass storage devices.

Logical device names TY1-TY4 (console) and ST1 (mass storage) must be specified at System Startup. If device names ST1 and TY1 are not defined, the console message *DEVICE xxx UNDEFINED is issued and Startup aborts (where xxx = ST1 or TY1).

One $ Iyy statement is required for each console configured as a system control device, while a separate $ Iyy statement is required to configure a System Control Center (SCC) that is to be used by the VIDEO program.

NOTE: The choice of the input/output controller statement type ($ IMU, $ IMX, $ IOM, $ IOP, $ IOX) is dependent on the hardware platform (DPS 8, DPS 8000, DPS 88, or DPS 90) and the peripheral subsystem (in the DPS 8 platform). A DPS 8 system may use either $ IOM or $ IMU statements or both. A DPS 8000 system may use only $ IMX statements. A DPS 88 system may use only $ IOX statements. A DPS 90 system may use only $ IOP statements. Channels can be specified interchangeably either as 'CH' or 'PUB', regardless of the statement type.

Throughout the rest of this section, $ Iyy is used to represent $ IMU/$ IMX/ $ IOM/$ IOP/$ IOX in the text and examples. Substitution of the proper statement type should be based upon the rules previously outlined. Any additional restrictions are noted in reference to the affected statement type.

$ Iyy STATEMENT FOR SYSTEM CONTROL DEVICES

The four system console names TY1, TY2, TY3, and TY4 must be assigned via $ Iyy statements defining system control devices (i.e., consoles). (GCOS software issues system messages to the consoles on the basis of the four console names.) These four names must be assigned even if only one device is configured.

When multiple consoles are configured, the four required console names can be assigned to any of the consoles in any manner (e.g., one name can be assigned to one console and three names can be assigned to another console). However, all four names must be specified.

The format for $ Iyy statements defining system control devices is:

```
             1
1     8      6
_______________________________________________________________

$     Iyy-m  {PUB-p},tttttt,nnn,...,nnn
             {CH-p }
```

where:

m - Number of the Iyy to which the system control device (i.e., console) is connected. The value of m can be 0-3.

p - Iyy channel (CH/PUB) number. Permissible values are:

IMU	8-63
IMX	0-127
IOM	8-31
IOP	8-63
IOX	0-127

tttttt - Device type as follows:

Code	Description
CS66/RMC66	CSU6601 console
CS6001	CSU6001 console
CS6002	CSU6002 console
CS6602	CSU6602 console
CS6604	CSU6604 console
CS6605	CSU6605 console

nnn - Name of the system console assigned to this Iyy and channel. The console name can only be TY1, TY2, TY3, and TY4.

The following example shows the logical device names assigned to the CSU6601 console (CS66) configured on channel 31 (CH-31) of Iyy-0.

```
             1
1     8      6
_______________________________________________________________

$     Iyy-0  CH-31,CS66,TY1,TY2,TY3,TY4
```

The following example indicates use of two $ Iyy statements to configure logical devices TY1-TY4 on two CSU6601 consoles (CS66). Note that these devices may be connected to one or two Iyys and that the channel number must be different on each $ Iyy statement.

```
             1
1     8      6
_______________________________________________________________

$     Iyy-0  CH-30,CS66,TY1,TY2
$     Iyy-0  CH-31,CS66,TY3,TY4
```

$ Iyy STATEMENT FOR VIDEO DEVICE

The logical device name TY5 can be assigned to a CSU6001, CSU6601, or CSU6602 if reserving one of the devices for use as a VIDEO device. The reserved device (for VIDEO purposes) is a CRT output-only device, and cannot be used for general input purposes. The device can be used to enter VIDEO-related commands.

The format for the $ Iyy statements defining a VIDEO device follows. The variables m, p, and tttttt are the same as those for defining system control devices.

```
               1
1      8       6
______________________________________________________________

               {CH-p }
$      Iyy-m   {PUB-p},tttttt,TY5
```

VIDEO operations change when the Console Manager software option is to be used or when GCOS 8 is executing on the DPS 88 system. In both of these instances, the VIDEO facility must be run on a terminal configured on a Front-end Network Processor. Refer to the GCOS 8 OS VIDEO manual for instructions for using VIDEO on a terminal.

$ Iyy STATEMENT FOR REMOTE MAINTENANCE CONSOLE

The Remote Maintenance Console is used primarily for the support of Customer Service maintenance and is configured in the Startup job stream as follows:

```
               1
1      8       6
______________________________________________________________

               {CH-p }
$      Iyy-m   {PUB-p},RMC66,TR1
```

where:

RMC66 - The Remote Maintenance Console device mnemonic

TR1 - The Remote Console Device name

The variables m and p are the same as those defined under "$ Iyy Format For System Control Device" on the preceding pages in this section.

The RMC66 resides on the remote port of the CSU6601 console channel adapter when the adapter is configured for two channels. It consists of a Diagnostic Processor Unit (DPU) with its on site console and remote port which has dial-in capability for Honeywell Bull Response Center usage.

The CSU6601 console in this mode requires that the channel for the Master Console have an even number. The channel for the Remote Console must have a contiguous odd number.

A second RMC66 channel may be configured as the odd-numbered channel on another channel adapter board, but is not given a device name. Access through the second channel to the RMC66 is accomplished by use of the MOVE console verb.

The following is a recommended Local/Remote system initialization procedure using TY1 and the RMC66 where TY1 and the RMC66 are configured on the same CSU6601 channel adapter board. The TY1 operator enables the RMC66 at that point via the ENABLE verb, and either the TY1 or RMC66 operator enters the RELINQUISH console verb. Then, either the TY1 or RMC66 operator places their keyboard in control (by pushing the RETURN or End Of Message key) and responds to system initialization dialog.

$ IMU OR $ IMX STATEMENT FOR THE MAINTENANCE CONSOLE

Every IMU or IMX in a system configuration has a maintenance console adapter (MCA) channel. It is fixed on channel 3 traditionally in the overhead channel group. It is configured in the Startup job stream as follows:

```
               1
1      8       6
_____________________________________________________________

$      {IMU-m} {CH-p },MCA[,ddd][,DEDICATED][,RLSE]
       {IMX-m} {PUB-p}
```

where:

m - IMU or IMX #

p - Channel # (Must be equal to 3)

ddd - Logical device name. A logical device name comprises three alphanumeric characters. The first character must be nonzero, the second character must be alphabetic, and the third character can be alphabetic or numeric.

Example:

```
               1
1      8       6
_____________________________________________________________

$       IMU-1  CH-3,MCA,MC1
```

For each configured IMU or IMX, for which no MCA is configured in the Startup job stream, Startup builds an entry at the end of the device SCT area, as though the following statement were included in the Startup job stream:

```
               1
1      8       6
_____________________________________________________________

$      {IMU-m} CH-3,MCA
       {IMX-m}
```

where:

m - IMU or IMX #

$ Iyy STATEMENT FOR MASS STORAGE SUBSYSTEMS

The same basic $ Iyy statement format applies for all mass storage devices. The format for these $ Iyy statements follows. (If more than one device is defined on a single $ Iyy statement, the UNIT-u and ddd fields must be repeated for each mass storage device configured within the device type (tttttt field) specified.)

```
                1
1      8        6
-------------------------------------------------------------------

               {CH-p }
$      Iyy-m   {PUB-p},tttttt,UNITS-n[,NONSEQ],

               {UNIT-uu   } {ddd  }
$      ETC     {UNIT-uu.s},{SPARE}[,vvvvvv][,S-xxxxx][,AU-a][,RMVBL][,

$      ETC      DEDICATED][,NOFMS][,RLSE]
```

where:

m - Number of the Iyy to which this mass storage subsystem is connected. The value of m can be 0-3.

p - Iyy channel (CH/PUB) number. Permissible values are:

IMU	8-63
IMX	0-127
IOM	8-31
IOP	8-63
IOX	0-127

tttttt - Device type for all units except those for which vvvvvv is specified. Refer to Table 5-4 for valid device types.

NOTE: The MSU3380, MSU3381, and MSS8080 subsystems have the following configuration restrictions:

1. These subsystems may not be specified on a $ IOM statement.

2. These subsystems may not be used for NPS sharing environment. *

3. The MSS8080 may only be specified on a $ IMX statement.

Table 5-4. Mass Storage Subsystem Parameters

Device Type Code (tttttt)	Description	Maximum Physical Devices	Maximum Logical Devices (UNITS-n)	Logical Device Number (UNIT-uu)
MS0400	MSU0400	32	32	1 - 32
MS0402	MSU0402	32	32	1 - 32
MS0450	MSU0451	32	32	1 - 32
MS0500	MSU0500	63	63	1 - 63
MS0501	MSU0501	63	63	1 - 63
MSC1A	MSU3380 *	32	32	0 - 31
MSC1B	MSU3380 **	32	64	0.0 - 31.1
MSD1A	MSU3381 *	32	64	0.0 - 31.1
MSD1B	MSU3381 **	32	96	0.0 - 31.2
MSM1A	MSS8080 *	64	64	0 - 63
MSM1B	MSS8080 **	64	128	0.0 - 63.1
MSM1E	MSS8080 **	64	64	0 - 63

* 64-word sector format
** 512-word sector format

n - Number of logical devices on this channel (CH/PUB) for this device type (tttttt field). If the value n is not the same as the total number of logical devices defined by UNIT-u fields, Startup aborts. For example, if there are three UNIT-u fields, n must be 3 (UNITS-3). (Refer to Table 5-4 for the maximum number of logical devices that can be specified for each device type.)

When defining MSC1B, MSD1A, MSD1B. and MSM1B subsystems, UNIT applies to a logical device. One MSC1B, MSD1A, or MSM1B comprises two logical devices. One MSD1B comprises three logical devices. Each logical device must be separately defined by a UNIT-u field.

NONSEQ - The logical devices that are defined by UNIT-u fields are not necessarily in numeric sequence. For example, the devices may be defined in the sequence UNIT-2, UNIT-3, UNIT-1, and UNIT-4 rather than in the sequence UNIT-1, UNIT-2, UNIT-3, and UNIT-4. Because mass storage devices are allocated in the same sequence that devices are defined on $ Iyy statements, the NONSEQ parameter can be used to influence the order of allocation.

uu
uu.s - Logical device identification number. The logical device name (ddd field) immediately following this field is associated with this logical device identification number. The subunit portion (.s) of this field is only valid for the device types in Table 5-4 that show subunits in the Logical Device Number column. (Refer to Table 5-4 for the valid form for each device type.)

ddd - Logical device name, comprised of three alphanumeric characters. The first character must be nonzero, the second character must be alphabetic, and the third character can be alphabetic or numeric. One device on the system must be named ST1. If ST1 is not defined, the console message DEVICE ST1 UNDEFINED is issued and Startup aborts. For mirrored pairs of devices, "SPARE" must be used as the logical device name for the secondary device.

vvvvvv - Device type, for logical device ddd, immediately preceeding. Valid codes are the same as for tttttt. vvvvvv defaults to the value specified for tttttt.

xxxxx - Five-character alphanumeric identifier (device label) of the logical device named ddd.

a - Number of llinks per allocation unit (AU). If this AU size is not the same as that used to initialize the logical device (i.e., via the $ INIT statement in the Startup job stream, Removable Storage Initialization Program (RSIP), or Removable Disk Initialization Program (RDIP)) the device will be released when there is an attempt to allocate space on the device.

It is recommended that AU size always be specified in conjunction with a released or removable device. Startup passes this value to the Mass Storage Allocation program, which builds in-memory allocation tables.

If a device is to have an AU size other than the default value, the value must be specified in this field. Otherwise, Startup assigns the default value, which is one for all devices except the MSU0501, MSC1B, MSD1B, MSM1B, and MSM1E. Those devices have a default value of eight.

For any volume formatted in 512-word sectors, that is to be a member of a volume set, the AU size must be eight or a multiple of eight.

The AU size may also be specified via the $ INIT statement in the $INITIALIZE section. If the size specified on the $ Iyy statement and the size specified on the $ INIT statement are different, the smaller size is used.

In addition, if the $ SHARED statement defines a logical device as being shared with NPS, the AU size for that logical device must be 12. (NPS allocates mass storage space in one-link (12-llink) increments.) If the device does not have an AU size of 12, NPS aborts.

RMVBL - This logical device is a removable device. (System files cannot reside on a device defined as RMVBL.) Any device that is to be a member of a volume set must be declared as RMVBL.

DEDICATED - This is a dedicated device.

NOFMS - The FMS catalog and permanent file structure is not permitted on this device. All space on this mass storage device is temporary file space. NOFMS cannot be used with RMVBL. If NOFMS is specified for device ST1, the console message:

ST1/SMCDUP/GCOS FILE/CAT DEVICES CAN'T DEFINE FMS

is issued and Startup aborts.

Startup also aborts if the CAT option is specified on the $ INIT statement for a device for which a $ Iyy...NOFMS statement is included.

RLSE - This device is configured and released at Startup.

The following example indicates that four (UNITS-4) MSU0451 mass storage devices (MS0450) are configured on channel 14 (CH-14) of IOM-0. One device (UNIT-1, logical device name MS1) contains permanent space and has a default allocation unit size of 1. The second device (UNIT-2, logical device name MS2) is a removable (RMVBL) device with an allocation unit size of 1 (AU-1). The third device (UNIT-3, logical device name MS3) is a dedicated (DEDICATED) permanent mass storage device with a default allocation unit size of 1. Programs using UNIT-3 must request this device by its logical device name. The fourth device (UNIT-4, logical device name MS4) contains only temporary space (NOFMS) and has a default allocation unit size of 1.

```
                1
1       8       6
_____________________________________________________________

$       IOM-0   CH-14,MS0450,UNITS-4,
$       ETC     UNIT-1,MS1,
$       ETC     UNIT-2,MS2,RMVBL,AU-1,
$       ETC     UNIT-3,MS3,DEDICATED,
$       ETC     UNIT-4,MS4,NOFMS
```

The next example indicates that two physical MSU0500 mass storage devices are configured on channel 15 (CH-15) of IOX-0. The logical device definitions for each physical device must be in odd/even pairs. UNIT-1 and UNIT-2 refer to one physical device and have logical device names ST1 and DS1 respectively. UNIT-5 and UNIT-6 refer to a second physical device and have logical device names DS5 and DS6 respectively.

Space is allocated on logical device DS5 in two-llink allocation units (AU-2). AU size on all other logical devices defaults to one llink. Use of this field exemplifies that each logical device is independent of the others and that parameters can be assigned on an individual logical device basis.

```
                1
1      8        6
_______________________________________________________________

$      IOX-0    CH-15,MS0500,UNITS-4,
$      ETC      UNIT-1,ST1,
$      ETC      UNIT-2,DS1,
$      ETC      UNIT-5,DS5,AU-2,
$      ETC      UNIT-6,DS6
```

The next example indicates two physical MSC1B mass storage devices configured on channel 56 (CH-56) of IMU-2. The UNIT-00.0 and UNIT-00.1 logical device designators refer to the same physical device and have logical device names MSA and MSB. The UNIT-01.0 and UNIT-01.1 logical device designators refer to the second physical device and have logical device names MSC and MSD. (Refer to the GCOS 8 OS System Operating Techniques manual for information about physical and logical device characteristics for each device type.)

```
                1
1      8        6
_______________________________________________________________

$      IMU-2    CH-56,MSC1B,UNITS-4,
$      ETC      UNIT-00.0,MSA,
$      ETC      UNIT-01.0,MSC,
$      ETC      UNIT-00.1,MSB,
$      ETC      UNIT-01.1,MSD
```

The next example indicates two physical devices configured on channel 44 (CH-44) of IMU-0. The logical device designators for one physical device are UNIT-04.0 and UNIT-04.1, with logical device names of MS4 and MS6, respectively. The other physical device has only one logical device designator, UNIT-05, with a logical device name of MS5. Note that this device is an exception to the attributes defined for the subsystem since it has only one logical device name for the physical device.

```
                1
1      8        6
_______________________________________________________________

$      IMU-0    CH-44,MSC1B,UNITS-3,NONSEQ,
$      ETC      UNIT-04.0,MS4,
$      ETC      UNIT-05,MSC1A,MS5,
$      ETC      UNIT-04.1,MS6
```

The MSU0501, MSC1B, MSD1B, MSM1B, and MSM1E devices are formatted into 512-word sectors which are transparent to software.

Device assignment and release functions for Startup are on the basis of logical devices. When one logical device is released the other logical device on the same spindle may still be assigned (not released). The release/assign condition of the other logical device should be interrogated using the PSTATS command. For the PSTATS, ASGN, or RLSE verbs, the following message is output to indicate the device release/assignment condition:

```
DEV icccdd     { RLSE }
               { ASGN }
```

MSU0501, MSU3380, MSU3381, and MSS8080 logical devices cannot be exchanged via the EXCHG console verb or the X response to an Exception Processing message.

MSU0501 logical devices can be mixed with MSU0451 and MSU0500 logical devices on the same subsystem (subject to the Startup rules for mixed configurations).

$ Iyy STATEMENT FOR MAGNETIC TAPE SUBSYSTEMS

The same basic $ Iyy statement format can be used to define MTS0400/0500, MTS0600/0610, MTS500, and MTS8200 magnetic tape subsystems. The format for this $ Iyy statement follows. (If more than one device is defined on a single $ Iyy statement, the UNIT-u and ddd fields must be repeated for each configured magnetic tape device within the device type (tttttt field) specified.)

```
                1
1      8        6
________________________________________________________________

               {CH-P }
$      Iyy-m   {PUB-p},tttttt,aaaa,UNITS-n[,NONSEQ],UNIT-u,ddd[,
$      ETC      zzzzz][,CLASS-c][,RESERVED][,DEDICATED][,RLSE]
```

where:

m - Number (0-3) of the Iyy to which this magnetic tape subsystem is connected.

p - Iyy channel (CH/PUB) number. Permissable values are:

IMU	8-63
IMX	0-127
IOM	8-31
IOP	8-63
IOX	0-127

tttttt - Device type:

Code	Description
TAPE*500*7	MTS500 seven-track magnetic tape
TAPE*500*9	MTS500 nine-track magnetic tape
TAPE*500*7	MTS0400/0500 seven-track magnetic tape
TAPE*500*9	MTS0400/0500 nine-track magnetic tape
TAPE*600*7	MTS0600 seven-track magnetic tape
TAPE*600*9	MTS0600 nine-track magnetic tape
TAPE*610*7	MTS0610 seven-track magnetic tape
TAPE*610*9	MTS0610 nine-track magnetic tape
TAPE*FIPS*9	MTS8200 nine track magnetic tape

NOTE: TAPE*FIPS*9 may not be specified on a $ IOM statement.

aaaa - Density code. This code represents the density capabilities of most of the devices (ddd) that are specified for this device type (tttttt). Valid density codes are:

Code	Description
ABCD	200, 556, 800, and 1600 bpi
ABC	200, 556, and 800 bpi
BCD	556, 800, and 1600 bpi
BC	556 and 800 bpi
CD	800 and 1600 bpi
DE	1600 and 6250 bpi
D	1600 bpi
E	6250 bpi

For example, if the UNITS-n field specifies four units are being defined for this device type and three of the units have density capabilities of 200, 556, 800, and 1600 bpi, the aaaa field would contain ABCD.

Exceptions to the entry in the aaaa field (i.e., any device that has density capabilities that differ from those defined by the density code) are defined in the zzzzz field.

n - Number of magnetic tape units on this channel (CH/PUB) for this device type (tttttt field). (Refer to Table 5-5 for the maximum number of units that can be specified for each device type.) If the value n does not equal the combined number of magnetic tape units defined in the UNIT-u fields, Startup aborts. For example, if there are three UNIT-u fields, n must be 3 (UNITS-3).

NONSEQ - The units (devices) defined in the UNIT-u field are not necessarily in numeric sequence. For example, the devices may be defined in the sequence UNIT-2, UNIT-3, UNIT-1, and UNIT-4 rather than in the sequence UNIT-1, UNIT-2, UNIT-3, and UNIT-4. Because magnetic tape handlers are allocated in the same sequence that devices are defined on $ Iyy statements, the NONSEQ parameter can be used to influence the order of allocation.

u - Unit identification number. The logical device name (ddd field) that immediately follows the UNIT-u field is associated with this unit number. (Refer to Table 5-5 for valid unit numbers for each device type.) Note that FIPS devices can have UNIT-0 as a valid unit number.

ddd - Logical device name. A logical device name comprises three alphanumeric characters. The first character must be nonzero, the second character must be alphabetic, and the third character can be alphabetic or numeric.

zzzzz - Magnetic tape characteristics (i.e., tape density and number of tracks) for individual magnetic tape units that are exceptions to information specified in the tttttt and aaaa fields. For each exception device, the zzzzz field must immediately follow the UNIT-u,ddd fields for the exception device:

UNIT-u,ddd,zzzzz

The last character of the zzzzz field represents the magnetic tape track type (i.e., 7 or 9). The preceding characters represent the tape density capabilities. Valid exception codes are:

Code	Description
ABC7	Seven track, 200/556/800 bpi
BC7	Seven track, 556/800 bpi
ABCD9	Nine track, 200/556/800/1600 bpi
BCD9	Nine track, 556/800/1600 bpi
CD9	Nine track, 800/1600 bpi
DE9	Nine track, 1600/6250 bpi
D9	Nine track, 1600 bpi
E9	Nine track, 6250 bpi

For ease of reference, it is recommended that each exception device be defined on a separate $ ETC statement.

c - Allocation class for logical device ddd. This parameter allows the site to reserve devices for specific uses based on site-defined considerations. For example, speed and transfer rate are two device attributes that can be considered when defining magnetic tape allocation classes. Physical location of devices also may be a consideration. Valid class codes are:

Code	Description
3	Most preferred
2	Preferred
1	Ordinary
0	Subordinate (unclassified) default code

If no allocation class is specified, class code 0 is assigned by default.

Based on the site-defined classes, the user programmer can indicate on the $ TAPE JCL statement which device and/or class is required for a specific application. (Refer to the GCOS 8 OS Job Control Language manual for a description of the $ TAPE JCL statement.)

RESERVED - The designated tape unit may be allocated only when the associated class is requested via the $ TAPE JCL statement. If the class is not specified on the $ TAPE statement, the tape unit is not allocated.

DEDICATED - This is a dedicated device.

RLSE - This device is configured and released at Startup.

Table 5-5. Magnetic Tape Subsystem Parameters

Device Type Code (tttttt Field)	Maximum Number of Units (UNITS-n Field)	Unit Number (UNIT-v Field)
TAPE*500*7	16	1-16
TAPE*500*9	16	1-16
TAPE*600*7	16	1-16
TAPE*600*9	16	1-16
TAPE*610*7	16	1-16
TAPE*610*9	16	1-16
TAPE*FIPS*9	17	0-16

NOTE: The device type codes TAPE*500*7 and TAPE*500*9 apply to seven-track and nine-track MTS500 and MTU0400/0500 devices.

The following example indicates that six (UNITS-6) MTU0600 magnetic tape handlers (TAPE*600*9) are configured on logical channel 16 (CH-16) of IOX-0. A majority of the six tape units are nine-track (*9) handlers with density capabilities of 800 bpi and 1600 bpi (CD). All six units are defined on $ ETC statements. One unit (TB6) is an exception to the attributes defined for the subsystem (i.e., 800 bpi and 1600 bpi density capabilities and nine-track handlers). TB6 is a seven-track handler with 200, 556, and 800 bpi density capabilities (ABC7).

In addition, TB4 is assigned to class 2 (CLASS-2) and will be allocated (RESERVED) only when that class is specified on the $ TAPE JCL statement. TB5 is dedicated (DEDICATED) and will be allocated only when requested by its logical device name.

```
              1
1      8      6
------------------------------------------------------------
$      IOX-0  CH-16,TAPE*600*9,CD,UNITS-6,
$      ETC    UNIT-1,TB1,
$      ETC    UNIT-2,TB2,
$      ETC    UNIT-3,TB3,
$      ETC    UNIT-4,TB4,CLASS-2,RESERVED,
$      ETC    UNIT-5,TB5,DEDICATED,
$      ETC    UNIT-6,TB6,ABC7
```

An equivalent set of Startup statements can be used on a system using any of the other input/output controller types.

The next example indicates that six units (UNITS-6) of MTS8200 magnetic tape handlers (TAPE*FIPS*9) are configured on channel 40 (CH-40) of IMU-1. All units are defined on a separate $ ETC statement for clarity. UNIT-2 is an exception to the attributes defined for the subsystem since it has a density capability of 800/1600 bpi.

```
                1
1      8        6
-------------------------------------------------------------
$      IMU-1    CH-40,TAPE*FIPS*9,DE,UNITS-6,NONSEQ,
$      ETC      UNIT-0,F0,
$      ETC      UNIT-1,1F1,
$      ETC      UNIT-2,CD9,1F2,
$      ETC      UNIT-8,1F8,
$      ETC      UNIT-9,1F9,
$      ETC      UNIT-10,1FA
```

$ Iyy STATEMENT FOR PRINTERS

The following statement format is used to specify a printer. Note that printer belt and vertical format control (VFC) image files must be created and initialized via a $ FILDEF statement in the $EDIT section of the Startup job stream for all printers except the PRU1100. Only one printer is permitted per logical Iyy channel (i.e., per $ Iyy statement).

The format for the $ Iyy statements for defining printers is:

```
                1
1      8        6
_______________________________________________________________

                {CH-p }
$      Iyy-m    {PUB-p},tttttt,UNITS-1,UNIT-1,
$      ETC       TRAIN-nnnn,ccc,ddd[,DEDICATED][,RLSE]
```

where:

m - Number (0-3) of the Iyy to which this printer subsystem is connected.

p - Iyy channel (CH/PUB) number. Permissible values are:

IMU	8-63
IMX	0-127
IOM	8-31
IOP	8-63
IOX	0-127

tttttt - Device type:

Code				Description
DPS 8	DPS 8000	DPS 88	DPS 90	
PR1100	PR1100	PR1100	N/A	PRU1100 printer
PR0901	PR0901	PR0901	N/A	PRU0901 printer
PR0901	PR0901	PR0901	N/A	PRU0903 printer
N/A	N/A	N/A	PR1200	PRU0908 printer
PR1200	PR1200	PR1200	PR1200	PRU1200 printer
PR1201	PR1201	PR1201	N/A	PRU1201 printer
PR1201	PR1201	PR1201	N/A	PRU1203 printer
N/A	N/A	N/A	PR1600	PRU1208 printer
PR1600	PR1600	PR1600	PR1600	PRU1600 printer

NOTE: On a DPS 88, the configuration of the PRU0901, PRU1201, PRU0903, or PRU1203 may require an upgrade to the IOX for this capability. Contact your Honeywell Bull representative for more details concerning this upgrade.

UNITS-1 - Number of printers on this channel (CH/PUB) for this device type (tttttt). (Only one printer can be configured on each channel.)

UNIT-1 - Unit identification number.

nnnn - Printer belt code. The printer belt number is imprinted in decimal on the carriage and must be converted to octal for this field. This field does not apply to PRU1100 printers. Valid codes are:

Code	Description
1130	Standard ASCII printer belt
764	Standard BCD printer belt

ccc - Print line width. This field does not apply to PRU1100 printers. Valid entries in this field are:

136 - Standard 136-character print line
160 - Extended print line of 160 characters

ddd - Logical device name. A logical device name comprises three alphanumeric characters. The first character must be nonzero, the second character must be alphabetic, and the third character can be alphabetic or numeric.

DEDICATED - This is a dedicated device. The Startup program prints only on a printer that is not dedicated. If output is desired during System Startup, there must be at least one printer which is not dedicated.

RLSE - This device is configured and released at Startup.

$ Iyy STATEMENT FOR FRONT-END PROCESSORS

There are two methods of communicating with a DATANET 8 Front-End Processor running DNS. The first of these, called FEP Gateway (DNET/ROUT mode), is an interrupt driven interface using traditional GCOS remote communication protocol. The second method is standard DSA Common Exchange Interface (CXI), a time driven queued interface used only for native mode GCOS 8 communications.

There are three different $ Iyy formats to configure the DATANET 8 Front-End Processor, one for each of the above methods and one for a combination of both.

Network Processor Using FEP Gateway (DNET/ROUT Mode)

The $ Iyy statement defines the configuration of a DATANET Network Processor for DNET/ROUT mode on a logical channel of an input/output controller. Only one DATANET Processor can be defined per logical input/output controller channel (i.e., per $ Iyy statement). The format for the $ Iyy statement which defines a DATANET Processor for DNET/ROUT mode is:

```
                1
1       8       6
_______________________________________________________________

$       Iyy-m   {PUB-p},tttt-n,LINES-xxx
                {CH-p }
```

where:

m - Number (0-3) of the Iyy to which the DATANET Processor is connected.

p - Iyy channel (CH/PUB) number. Permissible values are:

IMU	8-63
IMX	0-127
IOM	8-31
IOP	8-63
IOX	0-127

tttt - DATANET Network Processor identification (i.e., 6616, 6624, 6632, 6670, FNP)

n - DATANET Network Processor number. Processors must be sequentially numbered. The value of the processor number can range from 0-7.

xxx - Maximum number of lines (i.e., terminals) configured on DATANET Network Processor tttt that can be connected to the central system at the same time. Only the number of lines specified in this field are available, even though additional lines may be configured. This is a required field.

If multiple $ Iyy statements are used to define several DATANET Processors, the statements must be in processor number sequence (i.e., the $ Iyy statement for DATANET Processor 1 must precede the $ Iyy statement for DATANET processor 2). If the statements are not in the correct sequence, the console message:

```
*CARD BELOW IS OUT OF ORDER
```

is issued and Startup aborts. This message is immediately followed by an image of the statement out of sequence.

The following example indicates that one DATANET 6616 Network Processor (DNET/ROUT interface) is configured on channel 24 (CH-24) of IOM-1. The configured processor is processor number 0. A maximum of 50 terminals (LINES-50) can be connected to the central system.

```
                1
1       8       6
_______________________________________________________________

$       IOM-1   CH-24,6616-0,LINES-50
```

An equivalent statement can be used for a system configured with IMUs, IMXs, IOPs, or IOXs.

DSA Common Exchange Interface (CXI Mode)

The DATANET 8 Network Processor in Common Exchange Interface (CXI) mode must be configured on an Input/Output Controller Direct Interface Adapter (DIA) port under a device type of UNCP. This configuration is reflected in the following $ Iyy statement in the GCOS 8 Startup job stream:

```
                1
1       8       6
_______________________________________________________________

               {CH-p }
$       Iyy-m  {PUB-p},tttt,ddd
```

where:

m - Number of the Iyy serving the DATANET 8 processor

p - Iyy channel (CH/PUB) number. Permissible values are:

IMU	8-63	IOP	8-63
IMX	0-127	IOX	0-127
IOM	8-31		

tttt - Device type

UNCP - DATANET 8 network processor
SSF - System Support Facility

ddd - Logical device name. A logical device name comprises three alphanumeric characters. The first character must be nonzero, the second character must be alphabetic, and the third character can be alphabetic or numeric. Recommended names are FE0, FE1, FE2,..., FE9, FEA, FEB,..., FEF.

The logical device names FE0, FE1, FE2, FE3, FE4, FE5, FE6, FE7, FE8, FE9, FEA, FEB, FEC, FED, FEE, and FEF and the default values for CXI communications are defined in the standard workstation definition data base (WD) file in the SYS_SOFTWARE structure. If other names or values are used, the site administrator must make the appropriate changes in the WD file. New names must be chosen using the device naming conventions discussed under the input/output controller statements earlier in this section.

Note that Network Processors for CXI mode must be configured independently of Network Processors for DNET/ROUT mode. The GCOS 8 Startup process sees the following differences between the configurations:

- o Device types for the two modes are different, even when the physical device is the same. For CXI mode, the device type is UNCP and the following field of the configuration statement contains the Network Processor name (e.g., FEn, where n = 0 - F). For DNET/ROUT mode, the device type is 66xx-n, where xx is the specific device type and n is the device number in the configuration.

- o Network Processors for CXI mode must be numbered starting from device 0 (e.g., FE0, FE1,...FE9 ,FEA ,FEB,...FEF) and need not be in any particular number sequence. Network Processors for DNET/ROUT mode must be numbered sequentially (e.g., 6670-0 - 6670-7) and must be defined in processor number order in the $CONFIG section of the Startup job stream. While the configuration statements may be intermingled in the Startup job stream, the two numbering sequences must not be mixed.

- o Network Processors for CXI mode are not included in the FNP count on the $ INFO FNP/n statement in the Startup job stream. The n value must include only the Network Processor configured for DNET/ROUT mode.

- o Mailbox address switches for all Network Processors in CXI mode are BF8 (hexadecimal). Mailbox addresses for Network Processors in DNET/ROUT mode are based on input/output controller mailbox addresses and vary with the number of input/output controllers and Network Processors in the configuration.

  For example, if location 3000 (octal) defines the beginning of the mailbox area for the (first and) last input/output controller configured, the mailbox area for the first DATANET Processor (Processor 0) begins at location 5400 (octal). The mailbox area for each additional DATANET processor is obtained by incrementing the mailbox address of the predecing DATANET Processor by 100 (octal).

  The need to change mailbox address switches when splitting a large system configuration into two smaller configurations can be avoided by using the $ BASE statement.

$ IOX Statement For DPS 88 System Support Facility

The DPS 88 System Support Facility is configured as a DATANET 8 Network Processor in Common Exchange Interface (CXI) mode. Refer to "DSA Common Exchange Interface (CXI Mode)" above, for the definition of the Startup job stream statement to configure the System Support Facility.

Both DNET/ROUT Mode And CXI Mode

This format is used for a DATANET 8 processor that is used for both DNET/ROUT mode and CXI mode over the same input/output controller channel. The format is as follows:

```
                 1
1      8         6
_______________________________________________________________
                 {CH-p }
$      Iyy-m     {PUB-p},DDIA-n,ddd,LINES-xxx
```

where:

m - Number (0-3) of the Iyy to which the DATANET processor is connected.

p - Iyy channel (CH/PUB) number. Permissible values are:

IMU	8-63
IMX	0-127
IOM	8-31
IOP	8-63
IOX	0-127

n - DATANET processor number. n must be sequentially consistent (0,1,2,3,...) with all other DATANET processors that use DNET/ROUT mode (including On-Line PPSs)

ddd - Logical device name. A logical device name comprises three alphanumeric characters. The first character must be nonzero, the second character must be alphabetic, and the third character can be alphabetic or numeric. Recommended names are FE0, FE1, FE2,..., FE9, FEA, FEB,..., FEF

xxx - Maximum number of lines that can be connected at the same time on this DATANET processor. This is a required parameter for this format. It has significance in the DNET/ROUT mode but not in the CXI mode.

The logical device names FE0, FE1, FE2, FE3, FE4, FE5, FE6, FE7, FE8, FE9, FEA, FEB, FEC, FED, FEE, and FEF and the default values for CXI communications are defined in the standard workstation definition data base (WD) file in the SYS_SOFTWARE structure. If other names or values are used, the site administrator must make the appropriate changes in the WD file. New names must be chosen using the device naming conventions discussed under the input/output controller statements earlier in this section.

The following example illustrates the configuration of a DATANET 8 processor with the capability of using both DNET/ROUT mode and CXI mode over the same channel.

```
                 1
1      8         6
_____________________________________________________

$      IOP-0     PUB-53,DDIA-0,FE0,LINES-200
```

This is functionally equivalent to the following example, which requires the DATANET 8 processor to be physically connected to two channels.

```
                 1
1      8         6
_____________________________________________________

$      IOP-0     PUB-53,UNCP,FE0
$      IOP-0     PUB-54,6632-0,LINES-200
```

Page Processing System

A Page Processing System directly connected to an Iyy is specified to the Startup program via a $ Iyy statement. The format is:

```
                1
1      8        6
______________________________________________________________________

$      Iyy-m    {PUB-p},PPS*FEP-n,LINES-1
                {CH-p }
```

where:

m - Number (0-3) of the Iyy to which the Page Processing System is connected.

p - Iyy channel number (CH/PUB).

PPS*FEP - PPS device type code.

n - Network Processor number.

LINES-1 - Number of lines which may be connected to the PPS. This value is always 1.

The PPS is configured as a network processor (for DNET/ROUT mode), and the parameters required are analogous to those used to configure a DATANET Network Processor (for DNET/ROUT mode). The network processor number must be sequentially consistent (e.g., 0,1,2,...) with other PPSs and DATANET Processors (for DNET/ROUT mode) included in the system configuration.

A Page Processing System which is connected remotely to a DATANET Processor does not require this statement.

Examples Of Front-End Processor Configuration Formats

```
                1
1      8        6
______________________________________________________________________

$      BASE     IOX-2,DN-3
$      INFO     FNP/3
$      IOX-0    PUB-60,SSF,FE0                 SYSTEM SUPPORT FACILITY
$      IOX-0    PUB-61,UNCP,FE1                CXI MODE
$      IOX-0    PUB-62,DDIA-0,FE2,LINES-100    CXI & DNET/ROUT
$      IOX-0    PUB-63,6632-1,LINES-50         DNET/ROUT MODE
$      IOX-0    PUB-64,PPS*FEP-2,LINES-1       ON-LINE PPS
```

NOTES: 1. With the $ BASE statement included, the $ INFO FNP statement is redundant; however, if the statements do not agree on the number of FNPs, Startup will use the larger of the two numbers.

2. The number of FNPs in the $ INFO FNP/n statement and the $ BASE statement includes only the FNPs that use DNET/ROUT mode, PUBs 62, 63, and 64.

3. The devices on the three CXI mode channels are assigned logical device names FE0, FE1, and FE2. Although the $ IOX statements were presented in logical device name sequence in this example, no particular sequence is required for CXI mode.

4. The Network Processor number for DNET/ROUT mode device types are sequentially numbered 0, 1, and 2. It is required that statements be presented in that sequence for DNET/ROUT mode.

$ Iyy STATEMENT FOR HYPERCHANNEL COMMUNICATIONS ADAPTER

One $ Iyy statement is required per HYPERchannel (HYPERchannel is a trademark of the Network Systems Corporation) adapter to define the primary channel for the device. Additional logical channels on the same Peripheral Subsystem Interface Adapter (PSIA) may be used and are defined via a $ XBAR statement. A logical device should be defined for each use of the physical adapter by different communications protocols (e.g., DSA, MASSNET). The logical device, in conjunction with the HYPERchannel adapter logical device capabilities, separates I/O using the HYPERchannel into "logical streams" containing different protocols. The format for these $ Iyy statements is shown below.

```
                1
1      8        6
_______________________________________________________________

               {CH-p }
$      Iyy-m   {PUB-p},HYPER,UNITS-n,UNIT-u,ddd[,DEDICATED][,RLSE]
```

where:

m - Number (0-3) of the Iyy to which this HYPERchannel adapter is connected.

p - Iyy channel (CH/PUB) number. Permissible values are:

IMU	8-63
IMX	0-127
IOM	8-31
IOP	8-63
IOX	0-127

HYPER - HYPERchannel device type.

n - Number of logical devices on this channel (CH/PUB) for this HYPERchannel adapter. If value n does not equal the number of logical devices defined in the UNIT-u fields, Startup aborts.

u - Logical device identification number. The logical device name (ddd field) that immediately follows the UNIT-u field is associated with this device number.

ddd - Logical device name. A logical device name comprises three alphanumeric characters. The first character must be nonzero, the second character must be alphabetic, and the third character can be alphabetic or numeric.

DEDICATED - Specifies a dedicated device.

RLSE - Specifies a device that is configured and released at Startup.

$ Iyy STATEMENT FOR DATABASE COMPUTER

The $ Iyy statement defines the configuration of channels between the main frame and the database computer. A database computer includes two or more Interface Processors (IFPs), each of which is configured as two devices (one for input, the other for output).

A maximum of eight IFPs, or 16 devices, can be configured per physical channel.

A separate $ Iyy statement is required for each physical channel. If there are multiple physical channels, they may be connected to one or more input/output controllers.

The format for the $ Iyy statement defining database computer connections is:

```
                 1
1      8         6
______________________________________________

                {PUB-p}
$      Iyy-m    {CH-p }, DBC, UNITS-n,
$      ETC       UNIT-0,ddd,
$      ETC       UNIT-1,ddd, ...
$      ETC       UNIT-n-1,ddd
```

where:

m - Number (0-3) of the Iyy to which the channel is connected.

p - Iyy channel (CH/PUB) number of the first of the four logical channels assigned to the FIPS channel connected to the database computer. Permissible values are:

IMU	8-63
IMX	0-127
IOM	8-31
IOP	8-63
IOX	0-127

DBC - Symbol for database channel

n - Number of devices (twice the number of IFPs configured on this channel)

ddd - Logical device name. A logical device name comprises three alphanumeric characters. The first character must be nonzero, the second character must be alphabetic, and the third character can be alphabetic or numeric. These may be DB0, DB1, ..., DB9, DBA, ..., DBF corresponding to UNIT-0, ..., UNIT-15, respectively.

A separate UNIT-u phrase is provided for each of the n devices configured on the channel. Each phrase gives the name for that device. One or more $ ETC statements may be used to contain these phrases. Only unit numbers from 0 through 15 may be used. Unit numbers must appear in pairs. The first number in each pair must be even, and the second number in each pair must be one greater than the first. Numbering must start with zero.

Each physical channel is usually configured with two or four logical channels. To indicate logical channel usage, a $ XBAR statement is required. See "$ XBAR Statement", later in this section. A $ GROUP statement must be used to show that the logical channels can be used interchangeably. See "$ GROUP Statement", earlier in this section.

In the following example, two physical channels are connected to the database computer, and each is connected to four IFPs or eight devices. Four logical channels are provided for each physical channel.

```
                1
1       8       6
-------------------------------------------------------------
$       IOX-0   CH-12,DBC,UNITS-8,
$       ETC     UNIT-0,DB0,UNIT-1,DB1,UNIT-2,DB2,UNIT-3,DB3,
$       ETC     UNIT-4,DB4,UNIT-5,DB5,UNIT-6,DB6,UNIT-7,DB7
$       XBAR    IOX-0,CH-12,CH-13,CH-14,CH-15
$       GROUP   IOX-0,CH-12-15
$       IOX-0   CH-16,DBC,UNITS-8,
$       ETC     UNIT-0,DB8,UNIT-1,DB9,UNIT-2,DBA,UNIT-3,DBB,
$       ETC     UNIT-4,DBC,UNIT-5,DBD,UNIT-6,DBE,UNIT-7,DBF
$       XBAR    IOX-0,CH-16,CH-17,CH-18,CH-19
$       GROUP   IOX-0,CH-16-19
```

$ Iyy STATEMENT FOR CARD READERS AND CARD PUNCHES

The same basic $ Iyy statement format can be used to define card readers and card punches. Only one device can be defined per logical Iyy channel (i.e., per $ Iyy statement). The format for the $ Iyy statement defining card readers and card punches is:

```
                1
1       8       6
-------------------------------------------------------------------------
                {CH-p }
$       Iyy-m   {PUB-p},tttttt,UNITS-1,UNIT-1,ddd[,DEDICATED][,51OPT][,RLSE]
```

p - Iyy channel (CH/PUB) number. Permissible values are:

IMU	8-63
IMX	0-127
IOM	8-31
IOP	8-63
IOX	0-127

tttttt - Device type:

Code	Description
READER*MPC31	CRU1050 card reader
CC0401	CCU0401 card reader/card punch
CR0501	CRU0501 card reader
CR1050	CRU1050 card reader
PUNCH*200	CPZ201 card punch
PUNCH*300	CPZ300 card punch
PUNCH*MPC30	CPZ300 or PCU0120/0121 card punch
PC0120	PCU0120 card punch
PC0121	PCU0121 card punch
PC0300	CPZ300 card punch

UNITS-1 - Number of card readers or card punches on this channel (CH/PUB) for this device type (tttttt). (Only one card reader or card punch can be configured on each channel.)

UNIT-1 - Unit identification number.

ddd - Logical device name.

DEDICATED - This is a dedicated device.

51OPT - This parameter applies only to CRU1050 card readers and indicates that the device defined in the logical device name (ddd) field has the capability of reading 51-column cards.

RLSE - This device is configured and released at Startup.

The following example defines one CRU1050 card reader (CR1050) with a logical device name of CR1. The device is configured on logical channel 10 (CH-10) of Iyy-0. This device can read 51-column cards (51OPT) and is dedicated (DEDICATED) to programs that request the device by its logical device name.

```
               1
1      8       6
------------------------------------------------------------
$      Iyy-0   CH-10,CR1050,CR1,51OPT,DEDICATED
```

$ LOADFIL Statement

The $ LOADFIL statement defines the mass storage device and the file onto which the $LOAD section of the Startup job stream is to be written. (The $LOAD section contains modules in object deck format that are to be loaded into main memory.) A response of AUTO or REPL to the Startup message *BOOT SOURCE: AUTO OR REPL? during the bootload sequence results in the $LOAD section being read from the mass storage file (as opposed to loading the section from punched cards or magnetic tape).

The format for the $ LOADFIL statement is:

```
                1
1      8        6
_____________________________________________________________

$      LOADFIL ddd,fffff,sss
```

where:

ddd - Logical device name of the mass storage device on which a file is created to contain the $LOAD section of the Startup job stream.

fffff - Name of the load file (maximum of 12 characters) to which the $LOAD section is to be written. The file name cannot contain an embedded hyphen ("-").

sss - Size (in llinks) of the load file. There is no default size; therefore, this field must be defined.

The following example indicates that file LOADFILE is to be allocated 500 llinks of space on mass storage device DS2.

```
                1
1      8        6
_____________________________________________________________

$      LOADFIL DS2,LOADFILE,500
```

$ MCT Statement

The $ MCT statement defines the physical attributes (i.e., memory size and port connections) of each System Control Unit (SCU) for DPS 8 and DPS 90 systems. See "$ CIU Statement", earlier in this section for DPS 88 and DPS 8000. This statement must precede all $ Iyy statements.

Below is the format for the $ MCT statement. The optional fields (i.e., those enclosed in brackets) can be repeated to define a maximum of eight SCU port connections.

```
                1
1      8        6
_____________________________________________________________

$      MCT-n    ssss[,PORT-p,mmmmm/OFF]
```

where:

n - SCU number.

ssss - Memory size of this SCU. This value is specified as the number of 1K (1024 words) blocks. For DPS 90 the entire SCU configuration should be described on one $ MCT statement.

p - SCU port number (0-7).

mmmmm - Type code of the hardware module that is connected to the specified SCU port (p). The following codes are valid:

Code	Module
Iyy-x	Input/output controller (IMU/IOM/IOP) number x
PRO-x	Physical processor number x (x=0-5) (DPS 8 only)
EPU-x	Physical processor number x (x=0-3) (DPS 90 only)

NOTES: 1. Whichever device is configured here must be used throughout the Startup job stream or an error message will be displayed.

2. Refer to the "$ CIU Statement", earlier in this section, for DPS 88 and DPS 8000 systems.

3. In a DPS 8 system with a mixture of IOM and IMU input/output controllers, the input/output controller number must be unique among both controller types (i.e., specifying IMU-0 and IOM-0 is not valid).

/OFF - Defines the processor not currently assigned, but which may be assigned (activated) in the future. (Refer to "Processor Release And Assignment At Startup" in Section 3 for additional information.)

The following example defines two ports (0 and 1) on SCU 0, which has a memory size of 1024K. Processor 0 is connected to port 0. IOM 0 is connected to port 1.

```
                 1
1       8        6
_____________________________________________________________
$       MCT-0    1024,PORT-1,IOM-0,PORT-0,PRO-0
```

The minimum memory size required to support GCOS 8 is 4MB. If less than this amount is specified, Startup will issue the following error message and abort:

GCOS-8 REQUIRES AT LEAST 1024K OF MEMORY

DPS 90 CONSIDERATIONS

Figure 5-1 shows the hardware (physical) port assignments for a DPS 90 system.

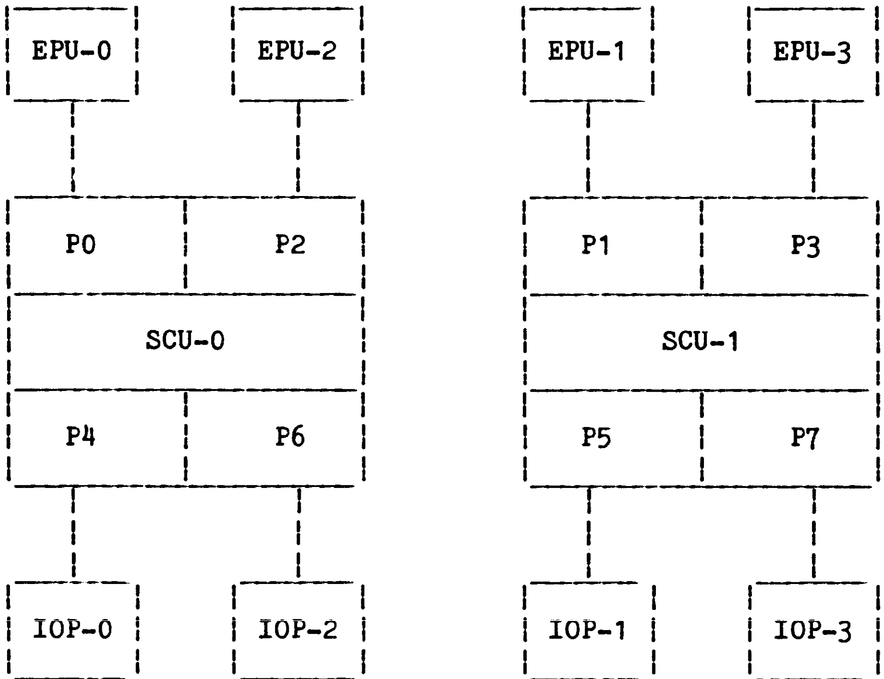

Figure 5-1. DPS 90 Hardware Port Assignments

Only one $ MCT statement should be used, even though two SCUs are physically present. The port assignments on the $ MCT statement must be made as shown under the appropriate columnar heading for "TWO SCUs" or "ONE SCU".

	TWO SCUs	ONE SCU
Port-0	EPU-0	EPU-0
Port-1	EPU-1	
Port-2	EPU-2	EPU-1
Port-3	EPU-3	
Port-4	IOP-0	IOP-0
Port-5	IOP-1	IOP-1
Port-6	IOP-2	
Port-7	IOP-3	

In order to avoid configuring more memory than is available, the entire SCU configuration should be described on a single $ MCT statement (with appropriate $ ETC statements).

Example 1

The logical EPU/IOP number is derived from the $ MCT statement. For a dual SCU system the logical EPU/IOP number can be the same as the physical EPU/IOP number. The following control statements define the maximum DPS 90 configuration available on a two-SCU system (i.e., 4 EPUs and 4 IOPs):

```
                 1
1      8         6
$      MCT-0     ssss,PORT-0,EPU-0,PORT-1,EPU-1,PORT-2,EPU-2,PORT-3,EPU-3,
$      ETC       PORT-4,IOP-0,PORT-5,IOP-1,PORT-6,IOP-2,PORT-7,IOP-3
```

where:

ssss - The total amount of memory configured to the system.

Example 2

However, on a single SCU, dual IOP, dual EPU system, the logical EPU/IOP numbers cannot be the same as the physical EPU/IOP numbers. In this case the following control statement should be used to define the configuration:

```
                 1
1      8         6
$      MCT-0     ssss,PORT-0,EPU-0,PORT-2,EPU-1,PORT-4,IOP-0,PORT-5,IOP-1
```

where:

ssss - The total amount of memory configured to the system.

In example 2, physical IOP number 2 is logical IOP number 1, and physical EPU number 2 is logical EPU number 1. It should also be noted in this example that logical IOP-1 is defined on PORT-5, even though PORT-5 is not physically present on a single SCU system. This is due to the fact that the port numbers specified on the $ MCT statement for IOPs are used in the following manner:

> For every logical IOP defined in the IOPMPS configuration, MSOS builds a connect table entry in Reserved Memory Space (RMS). If two logical IOPs are defined, then two connect entries will be built. These entries contain the true SCU port number to which the IOPs are connected. The port number specified on the $ MCT statement is used as an index into the RMS connect table as follows:
>
> connect table index = MCT port number -4
>
> (The first entry in the table is accessed with an index of zero.)

So, in example 2, IOP-1 is defined on PORT-5 on the $ MCT statement in order to obtain the correct index value of 1. The connect table entry at position 1 will contain the true SCU port number of 6.

$ MPC Statement

The $ MPC statement defines each configured Microprogrammable Peripheral Controller (MPC), URP0600 Unit Record Processor, or PPU and all active Peripheral Subsystem Interface (PSI) logical channels on that MPC or PPU. Hereafter, except for the SIZE parameter, MPC is used for either an MPC, URP0600, or PPU. Definition of the PSI logical channel includes identifying each associated input/output controller (i.e., IMU/IMX/IOM/IOP/IOX) and each logical channel (CH) on the input/output controller.

The format for the $ MPC statement is:

```
               1
1      8       6
________________________________________________

$      MPC-m   parameter-1,parameter-2,...,parameter-n
```

where:

m - MPC number (0-63). Each configured MPC must be assigned a logical number and defined via the $ MPC statement. If an MPC is not defined on this statement, the Startup program aborts or the MPC and its associated PSI configuration is ignored.

The following parameters can be included on the $ MPC statement(s):

$ MPC Parameter	Description
SIZE-s	The value s (default s = 4) specifies the size of the MPC read/write memory in 1K (1024 words) blocks. The value may be: MPC 4 PPU 4 (TAPE) PPU 16 (DISK)
TAPE*URC	Required only when a magnetic tape-unit record MPC is configured.
PSI-p periph subsystem interface	PSI logical channel number (0-3). The PSI that is used to bootload the MPC must be the first defined on the $ MPC statement. Thereafter, PSIs can be defined in any sequence (e.g., 0, 1, 3, 2).
SDB	The SDB (shared data base) parameter indicates that this PSI is being shared with NPS. This option is required to prevent Startup from bootloading the MPC firmware each time the system is bootloaded.

$ MPC Parameter	Description
BOOT	The use of the BOOT parameter following the PSI-p parameter allows the site to specify an alternate PSI logical channel over which bootloading can be performed. If BOOT is specified, the MPC Bootload program (MPCB) determines which PSI channel to use as the primary PSI channel, and which to use as the alternate channel. The MPCB program is not supported on the DPS 88 and DPS 90 systems. The following example specifies that bootloading should occur over PSI logical channel 0 (primary) or PSI logical channel 1 (alternate).

```
                1
1       8       6
_____________________________________________

$       MPC-0   SIZE-4,PSI-0,IOM-0,CH-8,IOM-0,CH-9,
$       ETC     PSI-1,IOM-1,BOOT,CH-10,CH-11
```

It is recommended that the BOOT parameter be used only with MTS0610 magnetic tape subsystems. If BOOT is specified for any other subsystem, bootloading problems may occur.

PORT-u	Unit record MPC controller port number associated with the preceding channel number. Port and channel numbers are assigned on a one-to-one basis. Startup assigns port numbers by default (1) if they are not specified on the $ MPC statement; or (2) if they are not specified sequentially beginning with PORT-0. Port numbers are not applicable on the DPS 88 system. On TAPE*URC controllers only ports 4-7 may be used, if specified.
Iyy-i	(Iyy = IMU, IMX, IOM, IOP, or IOX). The value assigned to i ranges from 0 to 3 and corresponds to the Iyy to which the PSI is connected.
CH-c	Base Iyy channel (CH/PUB) number.

The base channel number is the lowest channel number in a particular application. Due to a hardware requirement, the base channel number must be defined with a modulo 2, 4, or 8 value, as follows:

Number of Channels In Application	Legal Base Channel Numbers
1	8 to 31 ...
2	8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30 ...
3-4	8, 12, 16, 20, 24, 28 ...
5-8	8, 16, 24 ...

$ MPC Parameter	Description
	Logical channels for unit record MPCs must be sequentially defined in one-digit increments (e.g., 8, 9, 10 and not 8, 10, 12). Each Iyy channel number that is included on a $ Iyy statement must be defined on $ MPC statements. Permissible values are: IMU 8-63 IMX 0-127 IOM 8-31 IOP 8-63 IOX 0-127

The following briefly depicts the hierarchical structure that must be defined on $ MPC statements and associated $ ETC statements:

MPC-m — Defines the first MPC configured. After all subordinate PSI-p, Iyy-i, and CH-n have been defined, this is followed by other MPCs.

PSI-p — Defines the first PSI to which this MPC is connected. After all subordinate Iyy-i and CH-n have been defined, this is followed by other PSIs on this same MPC.

Iyy-i — Defines the input/output controller to which this PSI is connected.

CH-n — Defines the lowest logical channel number on the input/output controller to which this PSI is connected, followed in numerical order by other logical channel numbers on the input/output controller to which this PSI is connected.

This sequence is repeated until (1) all logical channels (CHs) are defined for the input/output controller, (2) the input/output controller is defined for its corresponding PSI, (3) all PSIs are defined for the MPC, and (4) all configured MPCs are defined. The MPCs can be defined in any sequence; however, for ease of reference, they should be defined sequentially.

EXAMPLES

MPC Examples

The following example defines three PSI logical channels (0-2) on MPC-0, which has 4K of read/write memory. PSI-0 is connected to Iyy-0, which has four logical channels (CH-8 through CH-11). PSI-1 is shared (SDB) with NPS. PSI-2 is connected to four logical channels (CH-16 through CH-19) on Iyy-1.

```
              1
1      8      6
------------------------------------------------------------
$      MPC-0  SIZE-4,PSI-0,Iyy-0,CH-8,CH-9,CH-10,CH-11,
$      ETC    PSI-1,SDB,PSI-2,Iyy-1,CH-16,CH-17,CH-18,CH-19
```

PPU Example

The following example defines two PPU logical channels (0 and 3) on MPC-1, which has 16K of memory. PSI-0 is connected to Iyy-0, channels 14 and 15. PSI-3 is connected to Iyy-1, channels 42 and 43.

```
              1
1      8      6
------------------------------------------------------------
$      MPC-1  SIZE-16,PSI-0,Iyy-0,PUB-14,PUB-15,PSI-3,Iyy-1,PUB-42,PUB-43
```

$ MPCFIG Statement

The $ MPCFIG statement permits identification of the firmware revision level to bootload to a specific Microprogrammable Peripheral Controller (MPC). This information is written to a table on the deckfile. If a firmware revision level is not specified for an MPC, Startup bootloads the MPC with the first firmware that it encounters on the deckfile. To ensure that all MPCs are bootloaded with the correct firmware, it is recommended that a $ MPCFIG statement be included for each configured MPC.

It is required that only one logical channel be defined per $ MPCFIG statement. If multiple $ MPCFIG statements are required, it is recommended that the statements be grouped at the end of the $CONFIG section to facilitate scanning by the MPC Bootload (MPCB) program.

The format for the $ MPCFIG statement is:

```
              1
1      8      6
------------------------------------------------------------
$      MPCFIG icc,pppp,REV.rr[,U-nndnn]
```

where:

i - Input/output controller number (0-3).

cc - Logical channel number that is used to bootload the MPC.

pppp - Four-character object deck (program) identification from columns 73-76 of the $ OBJECT statement in the firmware deck.

rr - Two-character revision level from columns 71 and 72 of the $ DKEND statement in the firmware deck.

nn - Device codes. These codes are used in conjunction with Microcoded Device Routines (MDRs) for precise definition of the devices for which MDRs are to be executed. It is recommended that the site's Customer Services Division (CSD) representative be consulted when defining the parameters in this field. Depending upon the device, the following codes apply:

- 00 - All devices are affected
- 01-32 - Mass storage or magnetic tape devices
- 00-07 - Controller ports (or devices) for unit record MPC
- 01-08 - Same as 00-07
- 01-02 - Link Adapter or Controller Adapter 0 and 1, respectively

d - Delimiters for the nn field

- Blank - End of field
- Comma (,) - Continuation of the field (e.g., 01, 03)
- Hyphen (-) - Separates a string of devices (e.g., 01-03)

The following example indicates that the MPC connected to Iyy-0 and to logical channel 16 is to be bootloaded with revision E1 of firmware deck M601.

```
              1
1      8      6
------------------------------------------------------------------------
$      MPCFIG 016,M601,REV.E1
```

$ SECURE Statement

The GCOS 8 Multi-Level Security Manager (Security) provides several levels of system security. A site may choose the options which provide the level of security required. The $ SECURE statement is used to specify which Security options are to be activated. The format for the $ SECURE statement is:

```
              1
1      8      6
------------------------------------------------------------------------
$      SECURE [option[ -arg]...[;option[ -arg]...]...]
```

The valid options and their respective valid arguments are:

ACCESS -TRYS n1
-COM a1
The ACCESS option activates the security authentication process, which requires the user to specify a valid userid and password for all remote system accesses. The ACCESS option also allows the system console operator to use the SECUR console verb to manually lock out a registered userid. The effect of the ACCESS option is modified by the PID option and the ALL option.

SCC -TRYS n1
-COM a1
The SCC option turns on the ACCESS option and, in addition, allows classification of userids and files by means of Security Classification Codes (SCCs), so that access to files can be limited based on the SCC. The SCC option also activates security breach processing, so that occurrence of a security breach locks the userid out of the system. The effect of the SCC option is modified by the PID option and the ALL option.

SYSTEM -TRYS n1
-COM a1
-MINLVL x
-MAXLVL y
The SYSTEM option turns on the ACCESS and SCC options, and also provides the capability to establish the minimum and maximum security levels for the system. The minimum security level defines the lowest security level allowed to access the system, and can be specified to disallow non-classified users and work units. The maximum security level defines the highest security level allowed to access the system.

PID -TRYS n1
-COM a1
-RANGE n2 n3
The PID option turns on the ACCESS option, and requires all remote users to use a person_id and a Personal Identification Code (PIC) instead of a userid and password in the logon sequence for the functions of the ACCESS option. When the PID option is active, the term "userid" is not used, replaced by "project name", and the password for the project is not used. Only person_ids that are registered on a project are allowed to gain access to that project. The PID option also allows the system console operator to use the SECUR console verb to manually lock out the Person Profile corresponding to a registered person_id. The PID option causes the person_id to be used in place of the userid for the functions of the SCC option if the SCC option is active.

PIC -TRYS n1
-COM a1
-RANGE n2 n3
The PIC option turns on the PID and ACCESS options, and has the same arguments as the PID option. The PIC option requires a Personal Identification Code (PIC) on the $ SECURTY (JCL) statement, along with the person_id, for all card reader, IMCV, and remote batch jobs.

ALL -TRYS n1
-COM a1
-MINLVL x
-MAXLVL y
-RANGE n2 n3
The ALL option provides a convenient method of turning on all of the Security options.

where:

n1 - The number of unsuccessful logon attempts allowed before the remote line is disconnected and the project profile or person profile is locked. (0 to 63, default = 2)

n2 - The minimum number of characters for PIC generation (4 to 11, default = 4).

n3 - The maximum number of characters for PIC generation (4 to 11, default = 4).

x - One character representing the minimum security level allowed to access the system (0 to 9, A to Z, no default).

y - One character representing the maximum security level allowed to access the system (0 to 9, A to Z, no default).

a1 - Comment string to be displayed by Startup. This string must be a protected string delimited by either quotes (") or apostrophes ('), with a maximum length of 55 characters, including delimiters. The null string is the default. If a delimiter must be displayed in the message, two adjacent delimiters will yield one delimiter in the output. For example,

```
-COM "AB""CD"   yields AB"CD
-COM "A""BCD""" yields A"BCD"
```

Another way to do the same thing is to delimit the protected strings using apostrophes: 'AB"CD' or 'A"BCD"'.

A $ SECURE statement option and associated argument(s) are separated from other options and associated argument(s) by a semicolon (;). For example:

```
                1
1      8        6
-----------------------------------------------------------------------
$      SECURE   SCC -TRYS 4 -COM "ALLOW 4 LOGON ATTEMPTS"; PID -RANGE 5 6
```

The $ SECURE statement can be continued to a $ ETC statement by using an ampersand (&) at the end of a line to indicate the continuation. For example:

```
                1
1      8        6
-----------------------------------------------------------------------
$      SECURE   SCC -TRYS 4 &
$      ETC      -COM "ALLOW 4 LOGON ATTEMPTS";&
$      ETC      PID -RANGE 5 6
```

$ SHARED Statement

The $ SHARED statement allows the site to define permanent mass storage devices to be shared with the Network Processor Supervisor (NPS).

If one or more $ SHARED statements are included with the Startup job stream, the No In-memory Available Space Table (NIAST) option is activated for all mass storage devices configured on that system. (Refer to the NIAST parameter under "$ INFO Statement" in this section.)

Devices described as removable (i.e., RMVBL) on the $ INFO NIAST statement cannot be shared. *

When identifying one or more permanent mass storage devices on a central system as common with NPS, the $ SHARED statement defines the amount of mass storage space to initially reserve on each device for NPS use. The format for the $ SHARED statement is:

```
                1
1      8        6
-----------------------------------------------------------------------
$      SHARED   MPC-m/PRIMARY,ddd/LIMIT-xxxxx
```

where:

m - Number (0-63) of the MPC controlling the common storage devices. Only one MPC can be shared with NPS.

PRIMARY - Must be specified to indicate mass storage devices are shared with NPS.

ddd - Logical device name of the permanent mass storage device on MPC-m shared with NPS. If multiple devices are shared, the ddd/LIMIT-xxxxx fields must be specified for each device. A maximum of two devices can be shared with NPS and all must be configured on the same MPC.

xxxxx - Number of llinks initially reserved on device ddd for GCOS 8 software allocation. As previously stated, if more than one device is specified by the ddd field, the /LIMIT-xxxxx field must be repeated for each device. Note that this value defines the amount of space initially allocated for GCOS 8 software. Additional space can be requested by and allocated to NPS as required for processing by the system.

The following example illustrates reserving a total of 6000 llinks of mass storage space for GCOS 8 use on permanent mass storage devices NP1 and NP2 (i.e., 3000 llinks are reserved for each device). Both devices are connected to MPC-0.

```
               1
1      8       6
______________________________________________________________

$      SHARED  MPC-0/PRIMARY,NP1/LIMIT-3000,NP2/LIMIT-3000
```

*

$ SYID Statement

The $ SYID statement identifies the system being bootloaded. This statement frequently identifies the Software Release on which the system is operating. Data included on the $ SYID statement is printed in the banner of all dumps and job output. (Refer to "Dump Banner" in Section 3.)

The format for the $ SYID statement is:

```
               1
1      8       6
______________________________________________________________

$      SYID    nnnnnn
```

where:

nnnnnn - A maximum of six alphanumeric characters identifying the system being bootloaded. The name must be acceptable to the File Management System (FMS), as the Error Logging and Analysis System (ELAN) uses the system-id as part of its collection file name.

*

$ TRACE Statement

The $ TRACE statement specifies which internal system trace entries to activate for the purpose of assisting in system failure analysis. The trace function provides a history of system operating events (that precede a malfunction). These events are recorded in a trace table, which is maintained as a circular list (i.e., the most recent trace entries overlay the oldest entries when the table fills). Table content can be written to magnetic tape and then printed to assist in debugging efforts.

Each bit position in the four control words (a...a, b...b, c...c, d...d) corresponds to a trace type. Setting a bit (=1) indicates that the corresponding trace function is not desired. (A common error is to set the bit to indicate that the trace is desired.) Each value specified in columns 16-64 of the $ TRACE statement is an octal number representing the bit configuration for the desired traces. For example, the octal number 7 (binary 111) indicates that none of the three traces represented by the bits is desired; the octal number 4 indicates that only the trace type represented by the first bit (1) is not desired. A $ TRACE statement containing all 7s in the four control words bypasses the trace routine.

The console entries TRACE ON and TRACE OFF override $ TRACE statement content for the duration of the current bootload. $ TRACE statement definitions are reinstated with the next bootload. The TRACE ON entry activates all system traces, while the entry TRACE OFF deactivates all system traces. Note that the TRACE ON entry is not effective unless at least one trace type was activated via the $ TRACE statement at System Startup.

The $ TRACE statement has no impact on Time Sharing System traces. All Time Sharing System traces are active when received with the Software Release unless changed via a site-option patch to the .MTIMS module at System Startup.

The format for the $ TRACE statement is:

```
                1
1       8       6
_______________________________________________________________

$       TRACE   aaaaaaaaaaaa,bbbbbbbbbbbb,cccccccccccc,dddddddddddd
```

where:

a...a - First control word, which defines trace types 0-35. At least one and as many as twelve octal characters must be specified.

b...b - Second control word, which defines trace types 36-71. At least one and as many as twelve octal characters must be specified.

c...c - Third control word, which defines trace types 72-107. At least one and as many as twelve octal characters must be specified.

d...d - Fourth control word, which defines trace types 108-143. At least one and as many as twelve octal characters must be specified.

NOTE: Each of the control words is zero-filled (i.e., if a control word contains a single zero, all trace types for that control word are desired).

The $ TRACE statement in the following example activates trace types 0-5 (represented by the two leading zeros in the first word), 36-71 (represented by all zeros in the second word), and 76 (represented by the octal number 6 in the third word).

```
                1
1      8        6
-----------------------------------------------------------------

$      TRACE    007777777777,0,767777777777,777777777777
```

Table 5-6 defines each trace type and its associated trace number. The module that creates the trace entry is identified in parentheses.

Table 5-6. System Trace Types

Octal Trace Number	Trace Type Description
000	Exit a module and return through a 64-word safe-store frame. (.MDISP)
001	Call a module via .CALLX or .CALLY. (.MDISP)
002	Call a module from SSA/PRC via .CALL or .CALLA. (.MDISP)
003	Process switch and dispatch to a process. (.MDISP)
004	Not used. Was end of courtesy call.
005	Processing GPR, SWAP, or ABORT. (.MDISP)
006	Enable execution of a process. (.MDISP)
007	Exit a module and return to SSA or PRC. (.MDISP)
010	Occurrence of a fault other than a MME. (.MFALT)
011	Transfer (GOTO) to a module. (.MDISP)
012	The processor is idle. (.MDISP)
013	Master Mode Entry (MME) fault occurred. (.MFALT)
014	Memory parity error occurred. (.MFALT)
015	Privileged MME (PMME) fault request. (.MPMME)
016	Marks start of courtesy call. (.MDISP)
017	Identifies the start of a system event for exception processing, process swap, or process abort. (.MDISP)
020	Marks end of activity for a process or end of job. (.MBRT6)
021	T&D trace entry (COLTS on/off trace or TOLTS termination trace. (T&D)
022	Call an HCM module. (.MDISP)
023	Exit a module and return to HCM. (.MDISP)
024	Entry to domain exception processing. (.MFALT)
025	OCLIMB from domain exception processing. (.MFALT)
026-027	Not used.
030	OCLIMB type dispatch. (.MDISP)
031	LDP/TRA type dispatch. (.MDISP)
032-043	Not used.

Table 5-6 (cont). System Trace Types

Octal Trace Number	Trace Type Description
044	Execute CIOC to start I/O. (.MIOSO)
045	Connect to a specified Front-end Network Processor (FNP). (.MDNET)
046	A call was made to entry point 2 of the specified channel module. (.MIOSO)
047	The specified I/O status was sent to the Exception Processor. (.MGEPR)
050	Link a request in channel queue. (.MIOSO)
051	IOM fault occurred on a channel. (.MIOSO)
052	Not used. Was accept marker.
053	Interrupt other than a marker interrupt, special interrupt, or terminate interrupt was processed. (.MIOSO)
054	Process special interrupt. (.MIOSO)
055	Terminate interrupt or initiate interrupt was processed. (.MIOSO)
056	Relinquish broken for a process. (.MIOSO)
057	Roadblock broken for a process. (.MIOSO)
060	Accept special interrupt. (.MIOSO)
061	Start status return action. (.MIOSO)
062	A fault or marker interrupt occurred. (.MIOSO)
063	Terminate interrupt or initiate interrupt accepted. (.MIOSO)
064	MPC controller special interrupt. (.MIOSO)
065	An IOX interrupt (DPS 88) was accepted. (.MIOSO)
066	A Rapid Access Data System (RADS) data move occurred. (.MSDCB)
067-107	Not used.
110	Specified activity sent to memory allocator. (.MALC1)
111	Specified peripheral device could not be allocated. (.MALC1)
112	Completion of SYSOUT printing for specified job. (.MGEOT)
113	Start of SYSOUT printing for specified job. (.MGEOT)
114	Completion of SYSOUT punching for specified job. (.MGEOT)
115	Start of SYSOUT punching for specified job. (.MGEOT)
116	Job to Peripheral Allocator. (.MSCHD)
117	Process number assigned. (.MSCHD)
120	Start of FLINKER activity. (.MFLKR)
121	Start of GEIN. (.MGEIN)
122	End of FLINKER activity. (.MFLKR)
123	SYSOUT notified of end-of-job. (.MSYOT)
124	SCT type specific allocation. (.MALC5)
125	DEVICE type specific allocation. (.MALC6)
126	DEVICE space deallocation. (.MALC9)
127	PERM file space allocation. (.MASO6)
130	User trace entry.
131-153	Not used.
154	Start of swap operation. (.MPOP4)
155	Start a new process. (.MPOP5)
156	Process swap complete. (.MPOP3)
157-177	Not used.

$ TRAINS Statement

The $ TRAINS statement defines the site's standard printer belts (i.e., those that support the GCOS system character set, but which may differ in character sequence). The Startup program uses these parameters to initialize the Table of Print Train Standards, which resides in main memory and which is a catalog into the PRINTIMAGE file. (Refer to the GCOS 8 OS System Operating Techniques manual for character set and PRINTIMAGE file information.)

Multiple character sets can be defined on one $ TRAINS statement or they can be continued on one or more $ ETC statements. Note that the printer type, character set identification number, and character set name fields are considered a single entity and are separated by slashes. These three fields are separated from the next trio of character set parameters by a comma.

ASCII character set definitions must be preceded by the word LONG. Any ASCII character set definition must follow the last BCD character set definition.

If information provided in the three fields is incompatible (e.g., the character set identification number and the printer type are inconsistent), the printer is released and the console message:

```
TRAIN ID UNKNOWN TO STARTUP DEVICE icccdd RELEASED
```

is issued (where i = input/output controller number, ccc = channel number, and dd = device number).

The format for the $ TRAINS statement is:

```
                1
1      8        6
-----------------------------------------------------------------
$      TRAINS   ppp/nnnn/tttttt,...,ppp/nnnn/tttttt,
$      ETC      LONG,ppp/nnnn/tttttt,...,ppp/nnnn/tttttt
```

where:

ppp - Printer type number:

4nn

where nn = any numerical value

nnnn - Standard print train identification number:

1130 - Standard ASCII character set
764 - Standard BCD character set

tttttt - Character set name:

GCOS - BCD character set (764)
ASCII4 - ASCII character set (1130)

LONG - Must precede definitions of long-character (i.e., nine-bit character) character sets to include in the Table of Print Train Standards. While this identifier can be appended to the BCD character set definitions on the $ TRAINS statement, it often is less confusing to define ASCII character sets on a separate $ ETC statement.

The following example defines one BCD (i.e., short-character) character set and one ASCII (i.e., long-character) character set.

```
              1
1      8      6
-----------------------------------------------------------------
$      TRAINS 402/764/GCOS,LONG,402/1130/ASCII4
```

$ UNIT Statement

The $ UNIT statement permits a switchable device or either of two separate devices to be referenced by the same logical name during Startup regardless of the controller in use. Any number of $ UNIT statements can be included in the Startup job stream.

The format for the $ UNIT statement is:

```
              1
1      8      6
-----------------------------------------------------------------
$      UNIT   aaa,bbb,message
```

where:

aaa - A previously configured logical device name.

bbb - A previously configured logical device name on an input/output controller other than the one to which aaa is assigned.

message - Any user-specified message not exceeding 24 characters.

If device aaa is on a released input/output controller and device bbb is on an input/output controller that has not been released, the references to the two devices are interchanged and the "message" specified by the user in the control statement is given on the system console.

If device aaa is on an input/output controller that has not been released or if both devices aaa and bbb are on released input/output controllers, the control statement is ignored.

The following example indicates that devices TB2 and TB6 have been previously configured and that TB6 will assume the device name TB2 if the input/output controller for TB2 is released during Startup. The message TB6 NOW TB2 will be given on the system console when the input/output controller for TB2 is released.

```
                1
1       8       6
_______________________________________________________________

$       UNIT    TB2,TB6,TB6 NOW TB2
```

$ URP Statement

The $ URP statement defines each URP800n Unit Record Processor (URP0600 must be defined on the $ MPC statement) and all active unit record peripherals controlled by that URP800n.

The format for the $ URP statement is:

```
                1
1       8       6
_______________________________________________________________

$       URP-m   Iyy-i,[PORTS-p],{PUB-n}
                                {CH-n }
```

where:

m - MPC/URP number (0-63). Each configured URP800n must be assigned a logical number and specified on the $ URP statement. See the $ MPC statement for specifications for the value of m.

i - The number (0-3) of the input/output controller to which the URP800n is attached.

p - The number of ports the URP800n controls. The value of p may be between 1 and 4.

n - The input/output controller channel (CH/PUB) number of Port-0. A device must be configured on this channel.

The following example indicates that logical Unit Record Processor (URP) number 5 is configured in IOM-1, and its four ports, Port-0 is connected to IOM channel 24.

```
                1
1       8       6
_______________________________________________________________

$       URP-5   IOM-1,PORTS-4,CH-24
```

The following example is identical to the preceding example, except it is used with a system configured with an IOX:

```
                1
1       8       6
_______________________________________________________________

$       URP-5   IOX-1,PORTS-4,CH-24
```

$ XBAR Statement

Component redundancy allows a site to employ the crossbarring technique, which is the physical and logical connecting of system components so that data can flow across channels through the system even though one or more components become inoperable. A channel is a data path between the peripheral device and the processor. Crossbarring configures data channels to provide two distinct paths. These pathways allow system software to route data along whichever channel is available without requiring reconfiguration of system components.

While hardware crossbarring considerations center on the physical interconnection (i.e., cabling) of components, software relationships between crossbarred input/output controllers and data channels (CH/PUBs) are defined via the $ XBAR statement.

The $ XBAR statement has two formats for each type of input/output controller. The first of the following formats is used to crossbar logical channels on the same input/output controller. The second format is used to crossbar logical channels from one input/output controller to another.

To crossbar logical channels on the same input/output controller:

```
               1
1      8       6
______________________________________________________________________

                     {PUB-a} {PUB-b}
$      XBAR    Iyy-m,{CH-a },{CH-b }
```

where:

Iyy - IMU, IMX, IOM, IOP, or IOX.

m - Iyy number (0-3)

a,b - Channel number. Permissible values are:

IMU	8-63
IMX	0-127
IOM	8-31
IOP	8-63
IOX	0-127

To crossbar logical channels from one input/output controller to another:

```
               1
1      8       6
______________________________________________________________________
$      XBAR    Iyy-m,CH-a,Iyy-n,CH-b
```

where:

Iyy - IMU, IMX, IOM, IOP, or IOX.

m,n - Iyy number (0-3).

a - Channel number of Iyy-m. Permissible values are the same as a,b, above.

b - Channel number of Iyy-n. Permissible values are the same as a,b, above.

In each of the preceding formats, the first channel (CH) defined is the primary channel and the channel number following an Iyy field applies to the preceding Iyy number. (The $ XBAR statement must follow the $ Iyy statement describing the primary channel identified on the $ XBAR statement.) A channel number cannot be defined on more than one $ XBAR statement.

Multiple logical channels (CHs/PUBs) can be crossbarred on the same $ XBAR statement. The sequence in which the channels are identified on the $ XBAR statement reflects the sequence in which the channels will be serviced. For optimum service, logical channel identifications should be alternated from one input/output controller to the other. If two channels from the same input/output controller are identified in sequence, the second channel is not serviced until I/O has completed on the first channel.

The following examples indicate a variety of crossbarring techniques within one subsystem. (Two or more subsystems cannot be crossbarred to each other.)

1. This example defines a single cable (one PSIA) with two logical channels crossbarred.

```
                1
1      8        6
------------------------------------------------------------

$      XBAR     Iyy-0,PUB-12,PUB-13
or
$      XBAR     Iyy-0,CH-12,CH-13
```

2. This example defines dual cables (two PSIAs) with logical channels crossbarred. Logical channel identifications are alternated between the PSIAs to optimize service. Logical channels 12 and 13 are on one PSIA and 20 and 21 are on another PSIA.

```
                1
1      8        6
------------------------------------------------------------

$      XBAR     Iyy-0,PUB-12,PUB-20,
$      ETC      Iyy-0,PUB-13,PUB-21
or
$      XBAR     Iyy-0,CH-12,CH-20
$      ETC      Iyy-0,CH-13,CH-21
```

3. This example expands example 2 to include two input/output controllers.

```
                1
1     8         6
-------------------------------------------------------------
$     XBAR      Iyy-0,PUB-12,Iyy-1,PUB-20,
$     ETC       Iyy-0,PUB-13,Iyy-1,PUB-21,
$     ETC       Iyy-0,PUB-14,Iyy-1,PUB-22,
$     ETC       Iyy-0,PUB-15,Iyy-1,PUB-23
or
$     XBAR      Iyy-0,CH-12,Iyy-1,CH-20,
$     ETC       Iyy-0,CH-13,Iyy-1,CH-21,
$     ETC       Iyy-0,CH-14,Iyy-1,CH-22,
$     ETC       Iyy-0,CH-15,Iyy-1,CH-23
```

NOTE: XBARed devices cannot be channelled across two different subsystems.

$INITIALIZE SECTION

The $INITIALIZE section has four essential functions in the System Startup process. Each of these functions is peripheral device-oriented. (Refer to "*INITIALIZE? Question" in Section 2 for discussions of each of these functions.)

First, the $INITIALIZE section identifies, via $ INIT statements, the mass storage devices to be initialized. To accomplish the initialization, the operator must make a positive response to the *INITIALIZE? question during the bootload procedure. The first $ INIT statement should name the device on which the AUTOLOAD file will reside.

Defective space existing on configured mass storage devices is identified in this section, via $ DIRECT statements and related D, O, A, L, and Y statements. This information is written to each applicable disk or fixed device during the initialization process.

If a device is not defined on a $ INIT statement, the device is not initialized. If the operator responds NO to the *INITIALIZE? question during the bootload procedure, the entire $INITIALIZE section is ignored and no mass storage devices are initialized.

Second, the $INITIALIZE section provides a facility for defining volume sets and their associated characteristics.

Third, the $INITIALIZE section performs a rollcall function during which the peripheral device configuration is confirmed. Mass storage devices are verified; magnetic tape handlers are surveyed; character set images and VFC images are loaded.

Fourth, the $INITIALIZE section is used to write firmware to the deckfile and to read firmware from that file for DPS 8 and DPS 8000 systems.

Importance Of Sequence In $INITIALIZE Section

The following statements can be included in the $INITIALIZE section:

$ DEKSAV
$ DIRECT
$ DKEND
$ INIT
$ OBJECT
$ READIN
$ VOPT
$ VSET

Although the following descriptions are presented in alphabetic order, the actual sequence of statements in the $INITIALIZE section of the Startup job stream must be as shown in Table 5-7 to ensure that it will be possible to start up a system

- o from a power-up condition (where no controller has firmware), or
- o with an INIT of the AUTOLOAD device, or
- o with the change of the AUTOLOAD device (causing the deckfile to become undefined), or
- o with the deckfile cleared to install a new revision of the Integrated Firmware And Diagnostic (IFAD) tape.

Any other sequence may lead to failure under some conditions.

With a production system bootload tape set up as defined in Table 5-7, successful controller initialization is always possible, except for hardware failure of a controller. A $INITIALIZE section without the firmware decks can then be used to selectively initialize devices, or to add $ DIRECT or D, O, A, L, or Y statements following a bootload with a REPL response to the *EDIT? question.

Table 5-7. Required Sequence In $INITIALIZE Section

1	8	16	
$INITIALIZE			
$	INIT		All $ INIT statements to be processed.
$	DIRECT		With related D, O, A, L, and Y statements to define defective space for the device on which the AUTOLOAD file is configured.
$	OBJECT		MPCB program
$	DEKSAV	OFF	To prevent writing of firmware to the deckfile before the firmware to support that operation has been loaded.
$	OBJECT		AUTOLOAD device controller firmware
		Caution:	If the system configuration includes two or more mass storage controllers that require different firmware, each of those firmware decks should be put here to ensure that there will be firmware for the AUTOLOAD device, no matter which device is configured as the AUTOLOAD device.
$	OBJECT		$ READIN device controller firmware
		Caution:	If the system configuration includes two or more tape controllers that require different firmware, each of those firmware decks should be put here to ensure that there will be firmware for the $ READIN device, no matter which device is configured as the $ READIN device.
$	DEKSAV	ON	To allow writing of subsequent firmware to the deckfile after firmware to support that operation has been loaded.
$	READIN		
$	DIRECT		With related D, O, A, L, and Y statements to define defective space for all remaining mass storage devices. The $ DIRECT statements must be in the same sequence as the devices are configured on $ Iyy statements in the $CONFIG section.
$	VSET		} For each volume set.
$	VOPT		}
***EOF			

$ DEKSAV Statement

The $ DEKSAV statement allows an MPC firmware job stream to be saved on the job stream file, which resides on the AUTOLOAD device and which is created via a $ DECKFIL statement. Two options are valid with the $ DEKSAV statement:

OFF - The firmware job stream(s) following this $ DEKSAV OFF statement in the card reader is not to be written to the deckfile.

ON - The firmware job stream(s) following this $ DEKSAV ON statement in the card reader is to be written to the deckfile.

The OFF option remains in effect until a $ DEKSAV ON statement is encountered. A $ DEKSAV ON statement is needed only to turn off (reset) the OFF option and allow subsequent firmware job streams to be written to the job stream file. Therefore, two $ DEKSAV statements must enclose a firmware job stream that is not to be saved.

The following example indicates that the first group of firmware deck images is not to be written to the deckfile, while the second group of firmware deck images is to be saved.

```
                  1
1      8          6
---------------------------------------------------------------
$      DEKSAV     OFF
.
.                 Firmware deck images that are not to be written to the
.                 deckfile.

$      DEKSAV     ON
.
.                 Firmware deck images that are to be written to the deckfile.
```

$ DIRECT Statement

The $ DIRECT statement can be used for two purposes. Each function is related to disk pack defective space information contained in the defective llink directory and the defective space history table. Both the directory and the table reside in the pack header.

1. The $ DIRECT statement identifies a mass storage device (ddd) and introduces a series of space-definition statements containing defective space information applicable to the device. Information from these statements, referred to as the D, O, A, L, and Y card images, is written to the defective llink directory and the defective space history table on device ddd.

2. The /CLEAR parameter can be included on the $ DIRECT statement to clear the defective llink directory and the defective space history table of information.

Defective space, which is not available for allocation, normally is defined on a list provided by the manufacturer of the pack. This data must then be written to the defective llink directory and the defective space history table via the D, O, A, L, and Y card images before the pack is used.

As a result of encountering a $ DIRECT.../CLEAR statement during disk pack initialization (after pack formatting has occurred), Startup issues the console message PROCESS DIRECTORY CARD FOR DEVICE ddd?, allowing the operator to clear or retain defective space in the pack header. If the operator responds YES, the defective llink directory and defective space history table are cleared.

To avoid accidental clearing of defective space information from a pack header, it is recommended that, following the initial system bootload, the $ DIRECT and D, O, A, L, and Y card images be removed from the Startup job stream.

When a disk pack is moved from one device to another, a $ DIRECT statement and its related space-definition statements must reflect this change. For example, if a disk pack is moved from device DS1 to device DS5, the $ DIRECT statement must indicate that the pack (i.e., the defective space) is on DS5.

The $ DIRECT statement for the AUTOLOAD device must always - regardless of the options included on the statement - precede the $ OBJECT statement. In addition, the $ DIRECT statements for all other devices must be in the same sequence that the devices are defined on $ Iyy statements.

The format for the $ DIRECT statement is:

```
              1
1      8      6
______________________________________________________________

$      DIRECT ddd[/CLEAR]
```

where:

ddd - Logical device name of the mass storage device on which the pack containing defective space resides.

/CLEAR - The pack header is cleared of defective space data, thereby indicating that all space on the pack is available for allocation. Defective space data can be reestablished via D, O, A, L, and Y card images.

If using the $ DIRECT ddd/CLEAR option, a $ DIRECT ddd statement specifying the same device must be included as the next statement, preceding any D, O, A, L, or Y card images of that device.

The D, O, A, L, and Y card images must immediately follow the $ DIRECT ddd statement identifying the mass storage device for which they are defining defective space. (The space-definition statements must not immediately follow a $ DIRECT.../CLEAR statement.) Each statement defines defective space in different quantities: cylinder/head, sectors, llinks, and links. The statement defining the contiguous defective space in the most efficient manner can be used. For example, if defective space occupies three contiguous llinks, the Y statement best defines the space.

The Defective Llink Directory Print (DLDP) program can be used to punch D, O, A, L, and Y card images from the defective llink directory for a specific device. Subsequently, these statements can be used to reestablish defective space information in the pack label. A listing of all tracks composing the defective llinks also is printed. (Refer to "Executing RSIP" in the GCOS 8 OS System Operating Techniques manual for a description of this program.)

The formats for the D, O, A, L, and Y card images are:

```
1
_______________________________________________

D0aaa00bb,ppppp,tttttt[,mmddyy][,zzzzz]

000cccccc-00dddddd,ppppp,tttttt[,mmddyy][,zzzzz]

Aeeeeeeee-ffffffff,ppppp,tttttt[,mmddyy][,zzzzz]

Lgggggggg-hhhhhhhh,ppppp,tttttt[,mmddyy][,zzzzz]

Yiiiiiii-jjjjjjj,ppppp,tttttt[,mmddyy][,zzzzz]
```

where:

aaa - Cylinder number (decimal).

bb - Head number (decimal).

c...d - Beginning (c) and ending (d) sector number (octal) in a series of contiguous defective sectors. If only one sector contains defective space, enter only 00cccccc.

e...f - Beginning (e) and ending (f) word numbers (octal) of defective space within a sector or within a series of contiguous defective sectors.

g...h - Beginning (g) link number (octal) and number (h) of links (octal) in a series of contiguous defective links. If only one link contains defective space, enter only gggggggg.

i...j - Beginning (i) llink number (octal) and number (j) of llinks (octal) in a series of contiguous defective llinks. If only one llink contains defective space, enter only iiiiiii.

ppppp - Pack serial number of the pack on which the defective space resides. (Logical device identifier if this is an MSU0500, MSU0501, MSU3380, MSU3381, or MSS8080 device.)

tttttt - Device type:

Code	Description	Code	Description
MS0400	MSU0400	MSC1B	MSU3380
MS0402	MSU0402	MSD1A	MSU3381
MS0450	MSU0451	MSD1B	MSU3381
MS0500	MSU0500	MSM1A	MSS8080
MS0501	MSU0501	MSM1B	MSS8080
MSC1A	MSU3380	MSM1E	MSS8080

mmddyy - Optional field specifying today's date in the format month (mm), day (dd), and year (yy).

z...z - Optional field containing any desired comments.

The following example causes the defective llink directory and the defective space history table for the disk pack (pack serial number A1665) residing on MSU0451 mass storage device DS3 to be cleared of information (/CLEAR). Subsequently, the directory and table entries that define defective space in sectors 1253-1255 and 1271-1278 and in llinks 3560-3575 of pack A1665 are to be recreated from information included on the O and Y space-definition card images and written to the defective llink directory and defective space history table on device DS3.

```
               1
1      8       6
______________________________________________________

$      DIRECT  DS3/CLEAR

$      DIRECT  DS3
000001253-00001255,A1665,MS0450
000001271-00001278,A1665,MS0450
Y00003560-00000015,A1665,MS0450
```

$ INIT Statement

The $ INIT statements identify the mass storage devices to be initialized during a bootload. (The statement does not apply to magnetic tape or unit record devices.) Any positive response (i.e., YES, TOTAL, or PARTIAL) to the *INITIALIZE? question during the Startup question-and-answer sequence may affect each device specified on a $ INIT statement. (Refer to "*INITIALIZE? Question" in Section 2 for additional information.)

If a device is not defined on a $ INIT statement or if the $ INIT statement has been removed, a positive response to the *INITIALIZE? question has no effect on the device. Unless a purge of all files on all devices is intended, it is recommended that all $ INIT statements be removed from the Startup job stream following the initial bootload (i.e., after the system has been installed).

It is recommended that all mass storage devices be defined on $ INIT statements and initialized when performing a cold boot. This technique helps ensure that required pointers and tables are reestablished on each device.

Finally, it is recommended that all permanent mass storage devices be initialized if the device named ST1 and/or the device for which the SMCDUP parameter applies are initialized. (The SMCDUP parameter is specified via the $ INFO statement.) Initializing in this manner also helps ensure that pointers and tables on all devices are synchronized. Failure to perform this type of initialization results in the warning message *ST1 AND SMCDUP INITED BUT NOT ALL OTHERS, FILE SPACE WILL BE LOST being issued to the system console. However, Startup does not abort.

All $ INIT statements must immediately follow the section-identifier statement ($INITIALIZE) and precede all other statements and job streams in this section. When Startup encounters a $ INIT statement, allocation control information is written to the pack label portion of the device (ddd) defined on the statement.

The format for the $ INIT statement is:

```
              1
1      8      6
_____________________________________________________
$      INIT   ddd[,AU-a][,CAT]
```

where:

ddd - Logical device name of the mass storage device to be initialized.

AU-a - Number of llinks (a) per allocation unit (AU) for this device. This value can be 1, 2, 4, 8, 12, 24, 36, 48, or 60.

NOTE: If device ddd is shared with NPS, the AU value must be 12, otherwise, NPS aborts.

If the AU size is not specified on the $ INIT statement, the size specified on the $ Iyy statement or the default size for this device is written to the Available Space Table (AST) in main memory. If both statements define an AU size and the sizes differ, the smaller value is used. An error condition could result on a subsequent warm boot. The message:

```
CANNOT BUILD IN-CORE TABLE ON ddd DUE TO
TBLSIZ s...s DEVICE RELEASED
```

is issued (where ddd = logical device name of the mass storage device on which the AST resides and s...s = status word). (Refer to the GCOS 8 OS System Operating Techniques manual for available space values for each mass storage device type. Refer to the $ INFO statement in this section for a description of the No In-memory Available Space Table (NIAST) option.)

The $ INIT statement initializes disk packs. Startup uses the AU value of the initialized pack to establish the size of allocation tables in memory. If a disk pack is not available at System Startup, Startup uses the AU size from the $ Iyy statement to establish allocation table size. To help prevent overlap problems between disk pack and in-memory tables whereby the in-memory tables are overwritten with and destroyed by information from the device tables, either the AU values on the two statements must be the same or the AU size must be greater on the $ INIT statement.

For any volume that is to be a member of a volume set, the AU size must be a multiple of eight if the volume is formatted in 512-word sectors.

CAT - Allows FMS to create catalogs on this device. This parameter can be specified for a maximum of 20 mass storage devices. If the parameter applies to more than 20 devices, the console message:

*ONLY 20 CATALOGABLE/SMC/SMCDUP DEVICES ALLOWED

is issued and Startup aborts.

If the CAT parameter is not specified for any device, Startup assigns the parameter to device ST1.

If the CATDUP parameter is specified on the $ INFO statement, but the CAT parameter is not specified on the $ INIT statement for the device, Startup assigns the duplicate catalogs to the device containing the duplicate SMC.

Startup aborts if the CAT parameter is specified on the $ INIT statement for a device for which the NOFMS parameter was specified on the $ Iyy statement. The following console message is issued:

CAN'T BE CAT AND NOFMS

The CAT option should not be specified for any volume that is configured as RMVBL. Catalog volumes for volume sets are specified via the $ VOPT statement; a Structured RMVBL volume is always a catalog volume.

The following example indicates that FMS can create catalogs (CAT) on mass storage device DS5. Space is allocated on the device in one-llink increments (AU-1).

```
                1
1      8        6
_____________________________________________

$      INIT     DS5,AU-1,CAT
```

Firmware Deck Images

As previously indicated, the $INITIALIZE section for DPS 8 and DPS 8000 systems must contain the MPC Bootload (MPCB) job stream and the firmware data images for mass storage and magnetic tape MPCs. These are in object format and are provided as a package with each Software Release.

The MPCB and firmware data images are written to the deckfile, which resides on the same device as the AUTOLOAD file. Unless overridden by a bootload using punched cards or a firmware tape, MPC firmware is loaded from the job stream file during subsequent warm boots. (If an MPC firmware job stream is not to be written to the deckfile, a $ DEKSAV statement containing the OFF parameter must precede the set of images. Refer to the $ DEKSAV statement description in this section for additional information.)

Only one MPCB job stream, which permits loading of the firmware data decks, is required regardless of the number of firmware data images included in the $INITIALIZE section.

One set of data images contains the firmware for the mass storage subsystem controller (MPC). A second set of data images contains the firmware for the magnetic tape subsystem controller (only required if processing a $ READIN for tape). Neither the mass storage subsystem nor the magnetic tape subsystem is operational until the applicable MPC is bootloaded via MPCB and a set of data images.

If more than one type of mass storage subsystem is configured (e.g., an MSU0500 subsystem and an MSU0451 subsystem), the only required firmware data images are those for the subsystem on which the AUTOLOAD device is configured. Firmware for the remaining MPCs is loaded from the deckfile.

The format for each firmware deck is:

Content	Description
$ OBJECT	
Preface statement	This statement is in standard relocatable format, with BOOT and DECK the primary SYMDEFs at location 0.
Data images	These images are in standard relocatable format and include MPC micro-instructions and data.
Identification block statement	This statement contains the following 10 words of data in standard relocatable format. This is the last binary data image. Word 0 - Name identifying type of firmware (e.g., unit record or mass storage firmware) 1 - Revision level 2 - Purpose of job stream 3 - Zeros 4 - Bits 0-17 contain the size of the first part of the job stream, which is the control store; bits 18-35 are unused. 5-8 - Zeros 9 - BCI constant MPCBOT
$ DKEND	

$ OBJECT AND $ DKEND STATEMENTS

Each firmware data deck is delimited by $ OBJECT and $ DKEND statements. While the data decks and the delimiter statements are provided with the Software Release, the formats for the $ OBJECT and $ DKEND statements are repeated here in case the card images are lost or damaged and must be re-created.

1	8	16	35	67	73
$	OBJECT	HMPCJ1	DAT	REV.rr	nnnn0000
$	DKEND			REV.rr	nnnnssss

where:

rr - Revision level of the firmware, which differentiates between different versions of firmware. This level also must be defined on the $ MPCFIG statement in the $CONFIG section of the Startup job stream. The revision level is saved from the $ MPCFIG statement in a change table on the deckfile and defines the firmware that is to bootload each MPC. It must also match the ID block data word 1.

nnnn - Level of the job stream

ssss - Deck sequence number. The $ DKEND statement sequence should be the number of the last binary statement plus 1.

PATCHING MPCB AND FIRMWARE DATA IMAGES

The MPCB job stream and the firmware data images can be modified or extended via patching procedures. The following briefly defines these processes.

- o The MPCB job stream can be modified via OCTAL correction statements, which must immediately precede the $ DKEND statement in the MPCB job stream. (Refer to "$PATCH Section", later in this section, for the format of the OCTAL correction statement.)

  NOTE: Patches to .MMPCB must be applied in this manner. They should not be included in the $PATCH section or patch edited.

- o The firmware data images can be modified via CHEX (control store) and RHEX (read/write memory) correction statements, which must immediately precede the $ DKEND statement in the firmware job stream. It is recommended that corrections to firmware data images be made under the direction of the site's CSD representative.

In each of the preceding cases, the operator must respond YES to the CLEAR AND OVERWRITE EXISTING DECKFILE? question to rebuild the MPCB or firmware data images on the deckfile for TOLTS subsystem usage. The entire MPCB job stream and firmware data job streams must be loaded via the card reader and the firmware tape must be mounted on the magnetic tape handler identified on the $ READIN statement. (Refer to "CLEAR AND OVERWRITE EXISTING DECKFILE? Question" elsewhere in this manual for additional information.)

$ READIN Statement (DPS 8 And DPS 8000 Only)

The $ READIN statement directs the Startup program to read MPC firmware from the firmware tape. The tape and its contents are cataloged on the deckfile, residing on the AUTOLOAD device.

It is recommended that the $ READIN statement be included in the Startup job stream only when a cold boot is performed, when the AUTOLOAD device is initialized, and when new firmware is written to the deckfile. When these functions are completed, remove the $ READIN statement from the job stream. If the statement is left in the deck and the operator responds YES to the *INITIALIZE? question, Startup unsuccessfully attempts to read the firmware. To circumvent the problem, the operator must cause Startup to fail (i.e., enter BOOT via the system console) and remove the $ READIN statement.

NOTE: The $ READIN statement is not recognized on DPS 88 Systems. The firmware is loaded by SMAS. Refer to the DPS 88 System Maintainability Availability Software (SMAS) manual for SMAS procedures.

The format for the $ READIN statement is:

```
              1
1      8      6
-----------------------------------------------------------------------
$      READIN ddd[,,,,tttt]
```

where:

ddd - Logical device name of the magnetic tape handler on which the firmware tape is mounted.

tttt - Density of the firmware tape:

DEN2 - 200 bpi	DEN16 - 1600 bpi
DEN5 - 556 bpi	DEN62 - 6250 bpi
DEN8 - 800 bpi	

This parameter is required only if the density of the firmware tape is different from the site's default high density, which is specified on the $ INFO DENxx/DENyy statement. If no default high density has been specified, DEN16 is the default value.

The following example indicates that the firmware tape is mounted on magnetic tape handler 1T7 with a density of 556 bpi.

```
              1
1      8      6
-----------------------------------------------------------------------
$      READIN 1T7,,,,DEN5
```

$ VOPT Statement

The $ VOPT statement is used to specify the characteristics and options for a volume set. The $ VOPT statement must immediately follow the $ VSET statement which defines the volume set. Exactly one $ VOPT statement is required for every $ VSET statement.

The format for the $ VOPT statement is:

```
              1
1      8      6
-----------------------------------------------------------------------
$      VOPT   vs_name,VSMC/pack1[,pack2]/[,CAT/packx[,packy[,...]]/][,
$      ETC    FDUP|NOFDUP]
```

where:

vs_name Volume set name. Must be the same as the vs_name on the immediately preceding $ VSET statement.

VSMC Volume Set Master Catalog. The VSMC specification is required. One or two pack numbers (device labels) must be provided. The volumes must have been defined as members of the volume set via the preceding $ VSET statement. If two volumes are specified, they must be on different physical units. The first will be used for the original VSMC; the second will be used for the duplicate VSMC.

If catalog duplication is elected (i.e., two volumes are specified with VSMC), the VSMC and all catalogs on the volume set will be duplicated. If one volume is specified, neither will be duplicated.

CAT The CAT option specifies the volumes on which catalogs are to be placed. The logical devices must be members of the volume set. If the CAT option is not specified, the VSMC volume(s) are used by default.

If catalog duplication is elected (i.e., two volumes are specified with VSMC), the number of catalog volumes must be evenly divisible by two and the list of catalog volumes must include more than one physical unit. (If this requirement is not met, then any attempt to create a catalog on the volume set will return the error message "NO SPACE FOR CATALOG ON DEVICE".)

A volume may appear only once in the list of CAT volumes. A VSMC volume may also be a CAT volume.

FDUP The files on the volume set may be duplicated. FDUP may be specified only if the volume set includes at least two physical units.

NOFDUP The files on the volume set may not be duplicated.

$ VSET Statement

The $ VSET statement is used to identify the members of a volume set. The $ VSET statement must be immediately followed by the $ VOPT statement for the same volume set. The volumes specified must all be mass storage, but may be of mixed types. The minimum number of volumes in a volume set is one; the maximum number is 100. All volumes in a volume set must be configured as RMVBL (see "$ Iyy Format For Mass Storage Subsystems" earlier in this section).

The format for the $ VSET statement is:

```
               1
1      8       6
------------------------------------------------------------
$      VSET    vs_name,pack1[,pack2[,pack3...]]
```

where:

vs_name Volume set name, 1-12 characters. Allowable characters are A-Z, 0-9, dash (-) and underscore (_). The first character must be from the set A-Z. The volume set name must be unique.

pack1 (etc.) The alphanumeric device label of a member of the volume set. The order of the device labels is not important. A volume may not be a member of more than one volume set.

$EDIT SECTION

The $EDIT section establishes GCOS system files on the mass storage subsystem and is closely affiliated with the $FILES and $INITIALIZE sections.

- o The $EDIT section establishes the content of all system files that the site may wish to use during the current and subsequent bootloads. The $FILES section selectively identifies which of these files the site intends to use during the current bootload.

- o The $INITIALIZE section recreates tables and pointers on each disk drive (i.e., disk pack) involved in the initialization procedure. (Initialization destroys existing tables and pointers on a disk pack, thereby rendering associated files inaccessible.) The $EDIT section, along with the total system software tapes, permits restoration of the files that were destroyed during the initialization procedure.

The $EDIT section also allows:

- o Creation of new system files. In addition, this section is the means by which these files are released and the space they occupied returned to the pool of available space.

- o Sites to edit data files for both site-options and JCL, rather than hard-coding them into memory, and provides default workstation JCL via the $ DTFILE/$ ENDTF statements.

If the content of a mass storage device that was initialized (i.e., the operator made a positive response to the *INITIALIZE? question) is to be restored, (1) the $EDIT section must contain information pertaining to that file, (2) the total system software tape containing file data must be mounted, and (3) the operator must respond YES or PARTIAL to the *EDIT? question during the bootload procedure. (Refer to "*INITIALIZE? Question" and "*EDIT? Question" in Section 2 for additional information.)

If the operator responds TOTAL to the *INITIALIZE? question, Startup forces a YES answer to the *EDIT? question. No console entry is required. The $EDIT section must be loaded via the card reader or magnetic tape and the total system software tapes must be mounted to effect the restoration.

If a file is not defined on a $ FILDEF or $ SSFILE statement, file content is not restored. If the operator responds NO to the *EDIT? question during the bootload procedure, the entire $EDIT section is ignored and no files are restored.

Files established within the $EDIT section cannot be dynamically modified via console entries. Modification must be performed during subsequent bootloads (e.g., manually changing statement content or adding and removing statements).

The following statements can be included in the $EDIT section:

$ DTFILE
$ ENDTF
$ FILDEF
$ SSFILE

The $ FILDEF and $ SSFILE statements can be in any sequence, while the $ DTFILE statement must always precede the $ ENDTF statement, when used.

$ DTFILE And $ ENDTF Statements

The $ DTFILE Startup statement is used with a $ ENDTF Startup statement. Every card image enclosed between these statements is written to the Startup-defined file ffffff on device ddd. This file is a system standard format card image file with block control words (BCWs) and record control words (RCWs).

The advantages of using the $ DTFILE Startup statement are:

- o Users are given the ability to edit data files for both site options and JCL, rather than hard-coding them into memory.
- o Default workstation initiation JCL is provided (under SYS_SOFTWARE).

The formats for the $ DTFILE and $ ENDTF statements are:

```
              1
1      8      6
-------------------------------------------------------------

$      DTFILE ddd,ffffff,sss[/O]
$      ENDTF  [ffffff]
```

where:

ddd - The device name of the mass storage device, named on the $ GCOSFIL Startup statement, where the file is to be created and written.

ffffff - File name (e.g., SYSPN). A maximum of 12 non-blank, alphanumeric characters.

sss - Size, in llinks. (Code places 25 card images per llink.)

/O - Overwrite option recommended (letter O).

The following sample Startup job stream uses the $ DTFILE and $ ENDTF statements with sample SYSPN JCL for SHARD and CWS.

```
                1
1      8        6
--------------------------------------------------------------------------

$      DTFILE   DT1,SYSPN,12/0
$      RUN      RUFILE=SYSTEM,RUNAME=SYSPN,OPTION=DUMP,NJREST
$      RESOURC  VSPACE=100K,SSDUMP
$      PRIVITY
$      DATA     I*
SPAWN -PATHNAME SYS_SOFTWARE/nnnn/JCL/SHARD.SPWN -SNUMB SHARD
SPAWN -PATHNAME SYS_SOFTWARE/nnnn/JCL/CMGR00.SPWN -SNUMB CMGRO
       .
       .
       .
$      ENDJOB
$      ENDTF    SYSPN
```

where: nnnn is the software release (i.e., 3000)

NOTE: For details on SYSPN and SYS_SOFTWARE refer to the GCOS 8 OS System Operating Techniques manual.

$ FILDEF Statement

The $ FILDEF statement defines a GCOS system file and assigns it to a specific mass storage device. All GCOS system files must be defined via $ FILDEF statements. The $ FILDEF statement also defines the magnetic tape device from which the file is to be copied. A separate $ FILDEF statement is required for each file.

The $ FILDEF statement gives the site the opportunity to balance system files across permanent mass storage devices. Often, different combinations must be tried to determine which best fits the site's operating environment. The purpose of this effort is to balance assignment of high-use and low-use system files across all configured devices so that files can be accessed faster and more efficiently.

The System Master Catalog (SMC) always resides on device ST1. Because these are high-use system files, it is recommended that several low-use system files be assigned to ST1 and that other high-use system files be assigned to other devices (thereby reducing device contention on ST1). Among the low-use system files are System Scheduler files, the Time Sharing Subsystem Library, the AUTOLOAD file, the deckfile, the load file, the DUMP0 and DUMP1-9 files, and the Integrated Firmware and Diagnostics (IFAD) file.

If the following files are not defined on a $ FILDEF statement, Startup creates the files and assigns them to ST1. The file sizes can be increased via a $ FILDEF statement and a device other than the default system device can be specified.

File Name	Default Size (llinks)	Description
DUMPO	100	Used by the Master Mode Dump (.MDUMP) program for storing the contents of the memory area used during swap operation. To enable .MDUMP program snaps, the size should be set to 200 llinks.
LUMP	24	Contains dump information and pushdown space.

The format for the $ FILDEF statement is:

```
1      8      16
$      FILDEF ddd,fffff,sss[/O][,xxx][,ttt][,LABEL/nnnnn][,yyyy]
```

where:

ddd - Logical device name of the mass storage device on which to create
* file fffff. This device must be defined on a $ Iyy statement in the $CONFIG section and must be a permanent mass storage device. If this device is a magnetic tape unit or is defined as a NOFMS or RMVBL device, Startup aborts and the console message:

*ddd fffff CAN'T BE xxxxx

is issued (where xxxxx = NOFMS, REMOVABLE, or NON-STORAGE). A maximum of 78 Startup files can be assigned to one device.

fffff - File name (1-12 alphanumeric characters). Note that use of the dump-to-disk feature requires that one or more files (DUMP1-DUMP9) be specified.

sss - File size (in llinks). If the file is not created large enough to accommodate the file that is being copied from magnetic tape handler ttt, Startup aborts. To correct the problem, re-edit the file from magnetic tape.

Special consideration must be given to the specification of the file size for .CRACF (accounting file), which must not exceed 16,383 llinks.

It is recommended that the DUMP1-DUMP9 files be created with an initial file size of 3500 llinks and then re-edited when dump requirements are known. Actual space used on these files is identified in the console message WROTE xxxxx LLINKS, which is issued following a dump. (Refer to the <u>GCOS 8 OS System Operating Techniques</u> manual for additional information on dumping to disk.)

/O - Filegrow (overwrite) option (letter O). If a file was defined for a previous boot and that file has since grown, the use of a new $ FILDEF statement that includes the /O option and the new filesize requirement replaces the old file to accommodate the new, larger file.

xxx - Code indicating the format for writing file fffff on device ddd.

Code	Description
SYS	Copy the file in system-loadable format. The file must be defined on a $ SYSTEM statement in the $FILES section.
RDM	Copy the file in random format (i.e., not in system-loadable format). The file must be defined on a $ LIBRARY statement in the $FILES section.
RUP	Copy the file in run unit update file format. The file must be defined on a $ SYSTEM statement in the $FILES section.
SWI	Copy the file in Seed Workspace Image format. At least one such file must be defined on a $ SYSTEM statement in the $FILES section.

ttt - Logical device name of the magnetic tape handler from which file fffff is to be copied. If ttt is not specified, file fffff will be created, but no data will be written to that file. When a magnetic tape device is specified on a preceding $ FILDEF statement, that same device can be referred to in the field via an asterisk (*).

nnnnn - This field is required for tape label verification and must contain the name of the file being copied from magnetic tape to file fffff (if file name fffff is different from the file name in the tape label).

yyyy - Density of the MPC-driven magnetic tape handler ttt from which file fffff is being edited. This field is required if the density of device ttt is other than 800 bpi. DEN8 is the default density. (If this field is included for a device that is not MPC driven, Startup aborts.)

Code	Description
DEN2	200 bpi
DEN5	556 bpi
DEN8	800 bpi
DEN16	1600 bpi
DEN62	6250 bpi

The following example shows a 2000-llink file (DUMP1) being created on device ST1. No data is being written to the file from an existing file on magnetic tape. Instead, information is written by the system after file creation.

```
                1
1       8       6
______________________________________________________________

$       FILDEF  ST1,DUMP1,2000
```

The following example shows a 1400-llink file (GC1) being created on device ST1. This file overwrites (/O) any existing file named GC1 on device ST1. Information is being written to this file in system-loadable format (SYS) from a file named VS-PRIMARY (LABEL/VS-PRIMARY), which exists on the 1600-bpi (DEN16) total system software tape that is mounted on magnetic tape handler 1T2.

```
                1
1       8       6
______________________________________________________________

$       FILDEF  ST1,GC1,1400/O,SYS,1T2,LABEL/VS-PRIMARY,DEN16
```

$ SSFILE Statement

The $ SSFILE statement defines the size of each job class from which the System Scheduler program assigns work to the system. In addition, the $ SSFILE statement performs two functions with respect to the System Scheduler program's SSFILE file that are similar to those performed by the $ FILDEF statement for other system files.

- o The sum of the job class sizes specified on the $ SSFILE statement is the amount of space that Startup assigns to the SSFILE file. Each job is cataloged in the SSFILE file by the System Scheduler.

- o The device on which the SSFILE file resides must be defined on the $ SSFILE statement.

Definition of class sizes on the $ SSFILE statement - as well as the parameters specified on the $ SSLOAD statement in the $FILES section - indicates the relative importance of each job class to the System Scheduler program. The number of llinks specified for each job class defines the number of jobs within the class that can be waiting for scheduling. Each llink of SSFILE file space can accommodate 39 jobs. (Each job requires an eight-word entry on the SSFILE.) Therefore, the number of jobs that can be cataloged for one class is derived by multiplying the number of llinks specified times 39.

Each job that is saved following execution (see $ INFO SCHSAV) also occupies eight words of SSFILE file space. It is recommended that this be taken into account when defining the size of each class.

The $ SSFILE statement must be present in the Startup job stream. If it is not present, Startup aborts. However, only one $ SSFILE statement is permitted.

Three job classes must be defined. The required classes are .EXPRS, .HOLD, and .user. The site can specify any name for the .user class. (Frequently, .user is defined as .USER.) If these job classes are not specified, Startup aborts.

Based on any parameter that the site selects, the .user class can be expanded to multiple classes. A maximum of 49 (02-50) .user classes can be specified in the optional .userx field(s). Note that definition of more than one .user class requires a .MSCAN module to evaluate the site-specified parameters and to direct jobs into the various classes.

The site can specify any names for the optional .userx classes. All names assigned must be defined on both the $ SSFILE and $ SSLOAD statements. The names assigned to optional .userx classes have no significance in their sequence of processing (e.g., class .USER04 is not necessarily processed before class .USER05). Priority processing by class is controlled by the maximum number of jobs within a class that can be scheduled for processing at one time, which is specified on the $ SSLOAD statement.

The special job classes .TASK and .TRANS should not be specified on the $ SSFILE statement, even though they may be specified on the $SSLOAD statement. Specification of these classes on the $ SSFILE statement may cause erroneous incrementing of class numbers, resulting in processing problems. However, all other job classes specified on the $ SSLOAD statement must also be specified on the $ SSFILE statement.

The format for the $ SSFILE statement follows. All classes must be defined in the specified sequence; otherwise, Startup aborts.

1	8	16
$	SSFILE	ddd,.EXPRS/n,.HOLD/n,.user/n[,.userx/n]

where:

ddd - Logical device name of the mass storage device on which the SSFILE file resides.

n - Number of llinks assigned to this class.

.userx - Optional site-defined classes. A maximum of 49 classes can be specified.

The following example indicates that the SSFILE file resides on device ST1. The .EXPRS class is assigned seven llinks, the .HOLD class is assigned five llinks, and the required .user class (.USER1) is assigned three llinks. Five optional site-defined classes also are defined. These include .USER2 (three llinks), .USER3 (four llinks), .USER4 (one llink), .USER5 (one llink), and .USER6 (one llink). The .TS8 class is assigned two llinks. Based on these definitions, Startup will assign 30 llinks of space on ST1 to the SSFILE file, assuming ST1 has one llink per allocation unit (AU). (That is, five llinks more than the sum of the llinks specified for each job class. If the AU size is greater than 1, the SSFILE size is further adjusted upward to be evenly divisible by the AU size.)

```
                1
1       8       6
--------------------------------------------------------------------------
$       SSFILE  ST1,.EXPRS/7,.HOLD/5,.USER1/3,.USER2/3,.USER3/4,.USER4/1,
$       ETC     .USER5/1,.USER6/1,.TS8/2
```

$FILES SECTION

The $FILES section identifies the system files to use throughout the duration of the current bootload. The $FILES section is one of two sections required in the Startup job stream. Only system programs residing on files defined in this section are identified in main memory tables. Therefore, a system program is not available during system operation if the file on which it resides is not identified in this section.

Each file identified in the $FILES section must be defined in the $EDIT section. Identification of a file in the $FILES section is a selective process. Some files identified in the $EDIT section may not be required and, therefore, can be omitted from the $FILES section.

For example, a total of 25 system files may be defined in the $EDIT section and created on mass storage devices. However, if the site requires only 20 files, only those 20 need identification in the $FILES section.

Two errors are commonly associated with the $FILES section.

1. If a file is identified in the $FILES section but not in the $EDIT section, the console message *FILE fffff UNDEFINED is issued and Startup aborts (where fffff = file name).

2. Any references to system programs residing on files not defined in the $FILES section result either in an error condition or in the request being ignored.

The following statements can be included in the $FILES section. These statements can be in any sequence:

$ ACCBUF
$ ACCOUNT
$ DFILES
$ LIBRARY
$ PFILES
$ SAVE
$ SCFBUF
$ SCFDSP
$ SSLOAD
$ SYSOUT
$ SYSTEM

$ ACCBUF Statement

The $ ACCBUF statement is used to:

- o Specify those record types buffered before being written to the Statistical Collection File (SCF).
- o Specify those record types whose buffering disposition (i.e., buffer or delete) can be changed by operator request.
- o Identify the maximum record type buffered.

If the $ ACCBUF statement is not present, record types 1-23 are buffered (by default) and can have their dispositions changed.

The format for the $ ACCBUF statement is:

```
               1
1     8        6
______________________________________________________________

$     ACCBUF   xxxxxxxxxxxx,yyyyyyyyyyyy,nn
```

where:

x...x - Twelve-character (octal) field identifying the record types to be buffered. Each bit corresponds to a record type (i.e., bit 0 defines record type 1, bit 1 defines record type 2, ... bit 22 defines record type 23). If a bit = 0, the record is buffered; if a bit = 1, the record is deleted.

The operator subsequently can enter ACCEPT nn to specify that record type nn is buffered and written to the Statistical Collection File (SCF). Conversely, the entry IGNORE nn specifies that record type nn is deleted and not written to the SCF.

y...y - Twelve-character (octal) field identifying the record types that cannot have their dispositions changed by operator request. If the bit is set (=1), the record's disposition cannot be changed. If the bit is not set (=0), the record's disposition can be changed.

nn - One-digit or two-digit number (0-36) specifying the maximum record type that can be buffered. The default value is 23. Any record specified in the x...x and y...y fields that exceeds the value nn is deleted and cannot have its disposition changed.

The following example indicates that all record types are to be buffered and that all record types can have their buffering dispositions changed. By default, record type 23 is the maximum record type that can be buffered (i.e., the nn field is null).

```
               1
1      8       6
______________________________________________________________

$      ACCBUF  0,0
```

When more than 36 accounting types are required, the $ SCFBUF and the $ SCFDSP statements may be used in place of the $ ACCBUF statement.

$ ACCOUNT Statement

The $ ACCOUNT statement defines the device on which the Statistical Collection File (SCF) is to reside.

```
               1
1      8       6
______________________________________________________________

$      ACCOUNT {RDM,fffff,ggggg}
               {RMV,ddd[,DENxx]}[,IDS

$      ETC     [,BUFSIZ/sss][,RETENT/rrr][,CONCUR][,JMAP]]
```

where:

RDM - The SCF resides on a mass storage device. Note that the maximum accounting file size is 16,383 llinks. (Refer to the "$ FILDEF Statement" definition, earlier in this section).

fffff - File name of the SCF.

ggggg - File name of the alternate SCF, used for data overflow.

RMV - The SCF resides on a magnetic tape device (ddd).

ddd - Logical device name of the magnetic tape unit on which the SCF resides.

DENxx - Tape density for magnetic tape units.

```
DEN2  -  200 bpi
DEN5  -  556 bpi
DEN8  -  800 bpi
DEN16 - 1600 bpi
DEN62 - 6250 bpi
```

The density must be the same as the density specified on the $ Iyy statement for the device defined in the ddd field.

If a density is not specified, the system default high density is assumed (as specified on the $ INFO DENxx/DENyy statement in the $CONFIG section).

IDS - I-D-S/I journal is to be configured.

sss - Buffer size. The minimum and maximum sizes are 320 and 1604 words, respectively. The default is 320 words.

rrr - Retention period (in days) for SCF records. The maximum retention period value is 999, indicating permanent retention. The default value is 0. The retention value is specified in decimal.

CONCUR - Specified only if the SCF is to be concurrently accessed.

JMAP - Specified only if a data base execution report is desired.

The following example shows the SCF residing on magnetic tape utilizing the maximum available buffer space available:

```
              1
1     8       6
________________________________________________________________

$     ACCOUNT RMV,1T1,DEN16,IDS,BUFSIZ/1604
```

The following example indicates that the files STATS and STATS1 are the SCF files, which reside on mass storage.

```
              1
1     8       6
________________________________________________________________

$     ACCOUNT RDM,STATS,STATS1
```

$ DFILES Statement

The $ DFILES statement causes the referenced file to be catalogued as a linked sequential file under SYS_CAT. It allows GEIN to read $ DTFILE related JCL statements, as described in "$ DTFILE/$ ENDTF Statements" earlier in this section.

The format for the $ DFILES statement is:

```
              1
1     8       6
________________________________________________________________

$     DFILES  ffffff
```

~e:

References the file name of the DTFILE described in the $EDIT ~tion (e.g., SYSPN).

$ LIBRARY Statement

The $ LIBRARY statement identifies one or two files to use as system libraries. They are searched by the General Loader program during program loading. One file defines the System Subroutine Library (L*); the second file is optional and defines the User Subroutine Library (*L), also referred to as the Secondary Subroutine Library.

The L* file contains high-use subroutines required for execution of major software packages. The *L file usually contains smaller, less frequently used subroutines. Consequently, the L* file normally is configured on a faster device than the *L file. If configured, the *L file is searched before the L* file.

File sizes and the device on which each file resides must be specified on the $ FILDEF statement in the $EDIT section. If the L* and *L files are configured, they must reside on the same type of storage medium (i.e., both files must reside on mass storage or both must reside on magnetic tape).

The format for the $ LIBRARY statement is:

```
              1
1      8      6
______________________________________________________________________

$      LIBRARY RDM,fffff[,RDM,ggggg]
```

where:

fffff - File name of the System Subroutine Library (L* file).

ggggg - File name of the optional User Subroutine Library (*L file).

RDM - Must precede the file name (fffff and ggggg) to identify the file as a random file.

The following example indicates that a one-llink system subroutine library named SYSTEMLIB is to be created on mass storage device DS5. In addition, a one-llink user library named USERLIB is created on mass storage device DS6.

```
              1
1      8      6
_______________________________________________________________________

.
.
.
$EDIT
$      FILDEF  DS5,SYSTEMLIB,1,RDM
$      FILDEF  DS6,USERLIB,1,RDM
.
.
.
$FILES
$      LIBRARY RDM,SYSTEMLIB,RDM,USERLIB
.
.
.
```

$ MODOPT Statement

The $ MODOPT statement provides a means for specifying parameters for initialization of GCOS modules. The parameters from the $ MODOPT statement(s) are passed to each designated module at module initialization time so that the module can verify the parameters and perform initialization according to the parameters.

The format for the $ MODOPT statement is:

```
              1
1      8      6
-----------------------------------------------------------------------
$      MODOPT .Mxxxx{,keyword[={value       }]   }
                   {        [ {(value,...)}]...}
                   {/{type       }              }
                   { {(type,name)}              }
```

where:

.Mxxxx - The GECALL name of the module, used to generate the module number. The symbol .Mxxxx must have been previously defined by a system macro such as .MDDEF.

keyword - Keyword identifying a parameter to be initialized. The valid keywords, and associated values for each module are listed below.

value - May have one of the following forms, and may be further limited by the module being initialized.

string - Any sequence of characters except space, quote ("), equal, left parenthesis, right parenthesis, or comma. Note: slash is allowed.

"D_Q_string" - Any sequence of characters. A quote (") may be contained within D_Q_string by using two successive quotes ("").

```
type     - { 000000                                }
           { [R][P]{HSSA|FSSA|SSSA|GHCM|SHCM} }
           { {NULL|R|P|RP|PR|APRG|NPRG|SPRG}    }
```

000000 - Octal value for module entry information, SD.MDD bits 0-17

R - Required

P - Privileged

NULL - To reset required and privileged modes

HSSA - Half SSA (512 words)

FSSA - Full SSA (1024 words)

SSSA - Shared SSA

GHCM - Grouped HCM

SHCM - Segmenting HCM

APRG - GCOS-type program

NPRG - RU-type program

SPRG - Shared program

name - A new GECALL name to put in the GECALL name table instead of the macro generated name. If this option is used, no patch card or later $ MODOPT references can be made to the old name; it will cease to exist as far as the system is concerned for accesses by module name. Note that any macro generated calls will have been assembled and will be executed by the module number and not the GECALL name.

INITIALIZATION PARAMETERS FOR .MDNET

```
                1
1      8        6
-----------------------------------------------------------------------
$      MODOPT   .MDNET,{TRACE_TYPES|TRACE}={OFF|ON|bbbbb|({OFF|ON}[,n...])}
```

where:

OFF - No trace. Memory for trace is released.

ON - Enable all trace types. Default.

bbbbb - Disable trace types 1 thru 13 (i.e., 47204 bit mapped equals 100111010000100 which would disable trace types 1,4,5,6,8,13). (Format of the old patch card).

OFF,n[,n...] - Turn off trace type n, where n = 1 thru 13.

ON,n[,n...] - Turn on trace type n, where n = 1 thru 13.

```
$       MODOPT   .MDNET,{NUMBER_TRACE_ENTRIES | TRCSZ}=nnnn
```

where:

nnnn - Specifies the number of entries (decimal) in the trace table. Default/Minimum = 80; Maximum = 1023. (If TRACE=OFF is used, there will be no trace table.)

```
$       MODOPT   .MDNET,{MAX_CONCURRENT_REMOTE_INQUIRIES | RMIQT}=nnn
```

where:

nnn - Specifies the number of entries (decimal) in the Remote Inquiry Name Table (N.AME). Default/Minimum = 40; Maximum = 255.

```
$       MODOPT   .MDNET,{MAX_GENERAL_STATUS_PROGRAMS | GENSTS}=nnn
```

where:

nnn - Specifies the number of entries (decimal) in the General Status Inquiry Table (GE.STS). Default = 40; Minimum = 0; Maximum = 255.

```
$       MODOPT   .MDNET,{RESERVE_MAILBOX_7 | RSMB7}={YES|NO}
```

where:

YES - Reserve mailbox #7 for Front-end Network Processor (FNP).

NO - Default.

```
$       MODOPT   .MDNET,{RECONNECT_TSS_AFTER_SLAVE_DISCONNECT |
                         RECDIS}={YES|NO}
```

where:

YES - Indicates that the automatic reconnect to TSS upon Disconnect Request (MME GEROUT 17) from a DAC process is desired. If the line originated from a TSS copy, an attempt will be made to return the line to that originating TSS copy and UST for continued service. If the originating copy is not available or if the line did not originate from TSS, the line will be disconnected.

NO - Disconnect the line when a DAC program issues a GEROUT 17. Default.

```
$       MODOPT   .MDNET,{TSS_LOAD_FACTOR | TSSLF}=n
```

where:

n - Specifies a value from 0-9 (decimal) representing the load factor that will cause additional users (i.e., logons) to be assigned to another executing copy of Time Sharing. The default value is 6.

A value of 0 indicates that users are to be assigned to all executing copies of Time Sharing on a round robin basis.

```
$       MODOPT   .MDNET,{CHECK_FOR_DNET_BUSY | CKBSY}={ON|OFF}
```

where:

ON - Enables scan of the line table for an FNP at the end of interrupt processing. Search is made for GS.DBS (DNET Busy Bit) left on any line on the FNP. If found, system will abort. USED FOR DIAGNOSTIC PURPOSES ONLY.

OFF - Default.

```
$       MODOPT   .MDNET,{I/O_TRACE | IOTRC}={OFF|ON}
```

where:

ON - Enables the trace of changes in .SRQCT, which is a count of I/O requests and requests in transmission for a process. USED FOR DIAGNOSTIC PURPOSES ONLY.

OFF - Default.

INITIALIZATION PARAMETERS FOR .MDSX6

```
                1
1       8       6
_______________________________________________________________
$       MODOPT  .MDSX6[,TMPFIL=(xxx,xxx[,xxx]...)[,SYSFIL=(xxx,xxx[,xxx]...)
```

where:

xxx - Each element in the lists is a logical device name. Each device name may appear in one or both lists. All devices specified must be mass storage devices configured as PERM. None of the devices may be configured as RMVBL. At least two physical units must be included in each list.

TMPFIL - The devices in the TMPFIL list are used to satisfy requests for disk space for temporary files ($ FILE or MME GEMORE), Test Mode collection space, Remote Job Input (RGIN) space, backing store files, swap files, and pushdown ($S) files. If the TMPFIL list is specified, it must include at least two physical units. If the TMPFIL list is not specified, the TMPFIL pool will consist of all PERM devices.

SYSFIL - The devices in the SYSFIL list are used to satisfy requests for J* and *J files and for the GEIN $ SELECT pushdown file (i.e., any request for temp space that originates from .MGEIN) plus the J$ and $J files, system recovery and cleanpoint files and System Scheduler Unpurge (SSUP) work space.

The devices in the SYSFIL list are also used during file grow. If a MME GEMORE to grow a permanent file requires crossing to another device (because the current device has insufficient unused llinks), AS06 allocates space from a device in the SYSFIL list selected by a round robin algorithm. SD.MSD contains an index to the next device to be tried for allocation. After each try, whether successful or not, the index is advanced to point to the next device in the list.

The catalog thresholds ($ INFO FMSST, FMSCT, and FMSDT) are in effect only when a file is created or restored. During file grow these thresholds are ignored.

It is recommended that the devices in the SYSFIL list not be designated as $ INFO CAT so that FMS will not build catalogs on those devices. The $ IOM/IOX NOFMS option may be specified for devices in the SYSFIL list; thus those devices would contain temporary file space, but would not contain FMS catalog or permanent file structure.

If the SYSFIL list is specified, it must include at least two physical units. If the SYSFIL list is not specified, the SYSFIL pool will consist of all TMPFIL volumes.

The TMPFIL and SYSFIL pools may overlap; a PERM device may be in the TMPFIL pool, the SYSFIL pool, both, or neither.

A temporary file may be placed on a PERM device not in the TMPFIL pool by requesting that specific device on the $ FILE JCL record.

$ PFILES Statement

The $ PFILES statement identifies Startup-created files to be cataloged by the File Management Supervisor (FMS) for subsequent access by users. The files are cataloged under the System Master Catalog (SMC) entry SYS_CAT. Unless Startup-created files are identified on the $ PFILES statement, they cannot be accessed. (Exceptions are all DUMP Files, which are automatically cataloged.)

A maximum of 79 files can be created by Startup via $ PFILES statements. If an attempt is made to create more than 79 files, Startup aborts. The message:

```
*UNABLE TO PFILE (file name)
*MORE THAN 79 PFILES AND DFILES
```

or

```
*UNABLE TO AUTOMATICALLY PFILE DUMPn
*MORE THAN 79 PFILES AND DFILES
```

will be output when the main PFILE routine detects an overflow condition. The message:

```
FATAL ERROR IN $FILES SECTION CARD BELOW CONTAINS AN ERROR NEAR COLUMN xx
```

is issued to the console (where xx = statement column number).

The format for the $ PFILES statement is:

```
              1
1      8      6
________________________________________________________________

$      PFILES ffffff
```

where:

ffffff - File name.

The following example identifies dump files to be used in dump-to-disk operations.

```
              1
1      8      6
________________________________________________________________

$      PFILES DUMP1,DUMP2
```

$ SAVE Statement

The $ SAVE statement specifies the name of the file used by the GCOS system for dump control and pushdown purposes. This file frequently is referred to as the "save file," however, any file name can be assigned.

The logical device name of the mass storage device on which this file is to reside and the file size may be specified on the $ FILDEF statement. This file must be allocated a minimum of 24 llinks.

The format for the $ SAVE statement is:

```
                1
1       8       6
--------------------------------------------------------------------
$       SAVE    ffffff
```

where:

ffffff - File name (one-six characters).

The $ SAVE statement is needed only if the site wishes to move the save file from ST1 or increase the file size to more than 24 llinks.

$ SCFBUF Statement

The $ SCFBUF statement is used in lieu of the $ ACCBUF statement and is found in the $FILES section; this provides extended accounting capability. It defines which accounting record types to buffer. The $ SCFBUF statement is followed by the $ SCFDSP statement.

When operating under the Tape Management System (GTMS), up to 72 accounting type records may be specified, including tape mount trace records written to the SCF. This record is produced only when the privileged slave program $GSYS is executing in debug mode. The record type created is 28 (octal 34).

The $ SCFBUF statement format is:

```
                1
1       8       6
--------------------------------------------------------------------
$       SCFBUF  nnn,ALL
or
$       SCFBUF  nnn,REC/X(1),X(2) - X(n)
```

where:

nnn - Three-digit number defining the maximum accounting record type for which buffering is done. The range of nnn is from 1-288.

ALL - Buffering for all record types (0-max.) has been selected.

REC - Selected accounting record types are to be provided buffering. X(1),...X(n) are the record types which have been selected and are to be stated in decimal. For example:

```
              1
1      8      6
------------------------------------------------------------
$      SCFBUF 36,ALL
$      SCFBUF 36,REC/1-10,12,14,17-19,21,30-36
$      SCFBUF 288,ALL
```

$ SCFDSP Statement

The $ SCFDSP statement defines the accounting record types which can have their status changed from the console. The $ SCFDSP statement should be immediately preceded by the $ SCFBUF statement.

The formats for the $ SCFDSP statement are:

```
              1
1      8      6
------------------------------------------------------------
$      SCFDSP nnn,ALL
or
$      SCFDSP nnn,REC/X(1),X(2), - X(n)
```

where:

nnn - Three-digit number defining the maximum accounting record type whose status can be changed from the console. This must match nnn of the $ SCFBUF statement.

ALL - All accounting record types can have their status changed via the console.

REC - Accounting record types X(1),...X(n) have been selected for console status change capability. X(1), etc., are decimal values. For example:

```
              1
1      8      6
------------------------------------------------------------
$      SCFDSP 36,ALL
$      SCFDSP 36,REC/2-8,14,17,21-30,32-36
$      SCFDSP 288,ALL
```

$ SSLOAD Statement

The $ SSLOAD statement further defines the various System Scheduler program job classes initially introduced in the $EDIT section on the $ SSFILE statement. The $ SSLOAD statement specifies the maximum number of jobs that can be scheduled from each class at one time. This statement also specifies the maximum and minimum number of jobs from all job classes that can compete for system resources (i.e., peripheral devices, memory, and processor time) at any one time.

The site can influence its processing workload and job scheduling through judicious use of $ SSLOAD statement parameters. Values specified for these parameters are unique to each site. No one value can be suggested as a norm for each parameter and no default values are assigned by Startup. Two recommendations can help establish parameters that best apply to each site.

1. Site techniques personnel must have a good working knowledge of the site's processing profile.

2. Site techniques personnel can experiment with the three types of parameters (i.e., system maximum, system minimum, and class maximums) to determine the combination of values that provides optimum throughput. Manipulation of these values - especially manipulation of the system maximum value - can produce favorable results.

The format for the $ SSLOAD statement follows. Note that definition of parameters is not required for the .HOLD class.

```
               1
1      8       6
-----------------------------------------------------------------
$      SSLOAD  MAX/xx,MIN/yy[,.TASK/nn][,.TRANS/nn],.EXPRS/nn,
$      ETC     .user/nn[,.cccc/nn],.TS8/nn
```

where:

MAX/xx - Maximum number (xx) of program numbers available for processing jobs from all classes at one time. This value specifies the number of jobs the System Scheduler program will pass to the Peripheral Allocator program to compete for resources at any one time.

MIN/yy - Minimum number (yy) of jobs that can be active before the System Scheduler program invokes individual class maximum values in the scheduling algorithm. If the total number of active jobs is less than this value, scheduling is based on the following:

1. Schedule any available job in the express class.

2. If there is no available express class job, schedule a job - regardless of its user class - on a first-in, first-out basis.

3. Return to the express class to schedule any available job.

When establishing this value (yy), consider the following:

1. Do not set the value so high that if the number of large jobs available is equal to the system minimum value, all resources would be assigned to those jobs (thereby denying resources to any other jobs entering the system).

2. If the value is 0, the class maximums for all jobs are strictly enforced. There is no override of class maximums and only the number of jobs specified by the class maximum values are run.

.TASK/nn - Maximum number (nn) of jobs that can be assigned at any one time via DRL TASK directives in the Time Sharing System. This field is required only if the site has Time Sharing System capabilities.

.TRANS/nn - Maximum number (nn) of Transaction Processing Applications Programs (TPAPs) that the Transaction Processing Executive program can have spawned at any one time. This field is required only if the site has Transaction Processing capabilities.

.EXPRS/nn - Maximum number (nn) of jobs that can be active at any one time from the express class.

.user/nn - Maximum number (nn) of jobs that can be active at any one time from the required site-defined class (.user).

.cccc/nn - Maximum number (nn) of jobs that can be active at any one time from the optional site-defined class (.cccc). A maximum of 49 optional classes can be defined (on the $ SSFILE statement). This field must be specified for each class defined on the $ SSFILE statement.

.TS8/nn - Maximum number of jobs that can be active at any one time from the .TS8 class. The value nn should be the same as the value specified for the TS8 Initialization file directive MAXIMUM USERS.

The following example indicates that a maximum of 10 jobs (MAX/10) from all defined job classes can be scheduled (i.e., compete for system resources) at any one time. A minimum of five jobs (MIN/5) from all defined job classes can be active at any one time before the System Scheduler program invokes individual class maximum values in the scheduling algorithm. The maximum number of jobs that can be assigned from the various job classes are: .TASK (10); .TRANS (10); .EXPRS (5); required user class .USER1 (1); and optional user classes .USER2 (3), .USER3 (1), .USER4 (1), .USER5 (1), and .USER6 (1).

```
                1
1       8       6
-------------------------------------------------------------------
$       SSLOAD  MAX/10,MIN/5,.TASK/10,.TRANS/10,.EXPRS/5,
$       ETC     .USER1/1,.USER2/3,.USER3/1,.USER4/1,.USER5/1,.USER6/1
```

$ SYSOUT Statement

The $ SYSOUT statement specifies the names of files used for the collection (i.e., spooling) of system output on mass storage. A maximum of 64 files can be specified.

All files should be created the same size. The logical device name of the mass storage device on which the file is to reside and the file size are defined via the $ FILDEF statement. If the sizes of all files named in the $ SYSOUT statement are not the same, the smallest value specified on a $ FILDEF statement is allocated on each device. For example, if 100 llinks are specified for all devices except one on which 50 llinks are specified, only 50 llinks are used on all devices.

Specification of SYSOUT collection space requirements allows site management to define its particular needs according to the site's processing environment.

The format for the $ SYSOUT statement follows:

```
                1
1       8       6
-------------------------------------------------------------------
$       SYSOUT  ffffff,ffffff, ... ,ffffff
```

where:

ffffff - File name (one-six characters). A maximum of 64 files can be identified for SYSOUT collection purposes. These are all defined in one $ SYSOUT statement or a $ ETC continuation of that statement.

$ SYSTEM Statement

The $ SYSTEM statement identifies each file containing system modules and system-shared software necessary for site operations. These file names are written to in-memory tables. The size of each file defined on the $ SYSTEM statement and the device on which the file resides must be specified on a $ FILDEF statement in the $EDIT section. The RUP, SWI, or SYS parameter must be included on the $ FILDEF statement for each of these system or shared software library files.

A maximum of 32 system files can be defined. If more than 32 files are specified, Startup aborts. While the $ SYSTEM statement normally identifies the names of all files specified on a $ FILDEF...SYS statement, this is not always the case. Some sites may have more than one version of system software. In these instances, $ SYSTEM statement definitions may not include all files specified in the $EDIT section.

Multiple file names can be specified on one $ SYSTEM statement. The format for the $ SYSTEM statement is:

```
                1
1      8        6
_______________________________________________________

$      SYSTEM   fffff
```

where:

fffff - System file name.

The $ SYSTEM statement can also be used either to replace existing system modules, or to add system modules to the content of the existing system file by employing a technique called an insert-edit. This technique expediently updates one or more software modules resident on the system files and, because it is an alter only update, eliminates the need to re-edit the entire system. Although the number of program executions is no fewer than when performing a total edit, the selectivity of the insert-edit reduces the scope of the edit by minimizing the number of modules involved in the editing process.

A replacement file can have a unique file name, or it can retain the name of the file being replaced or modified. In addition, all other files containing system programs and remaining in effect must also be specified on the $ SYSTEM statement for the insert-edit.

The sequence in which the new or replacement files and the existing files are defined on the $ SYSTEM statement(s) is important so that Startup can identify and replace an existing file with the replacement file, or insert a new file. Systems statements defining new system files should appear first. Replacement file names should immediately precede the preexisting system file name being replaced. This allows site personnel to easily identify the system files involved in modification.

If it appears that the Startup program has not inserted a new file or replaced an old file following an insert-edit, check the order of appearance of system file names on the output list of the $ SYSTEM statement(s). If improperly sequenced, rearrange them as required to produce the desired results and reboot.

In the following example, replacement file NEW-SOFT contains system programs that are to replace programs residing on existing file, VS-PRIMARY. Other files containing system programs that are not involved in the replacement procedure are identified on $ ETC statements.

```
1       8       1
                6
$       SYSTEM  NEW-SOFT
$       SYSTEM  VS-PRIMARY
$       SYSTEM  TSS-SUB-SYS
$       SYSTEM  SOFTW-PART1
```

$PATCH SECTION

The $PATCH section provides a means by which corrections can be applied to GCOS programs and to software programs that were edited by Startup onto the system. Corrections cannot be applied via the $PATCH section to programs added to the system via a File Management Supervisor (FMS) restore function. (Refer to the GCOS 8 OS File Management Supervisor manual for a description of restore procedures.)

Four types of patch corrections can be applied via the $PATCH section:

- OCTAL correction statements to replace the contents of specific module locations
- OCTAL correction statements to add to the contents of a module
- OCTAL correction statements which are site-option patches. Site-option patches allow the site to select specific operating parameters for its system software modules and programs. Each of the valid site-option patches is described later in this section, under "Site-Option Patches".
- PATCH_RUN_UNIT statements which are applied as corrections to run units that are part of the Seed Workspace Image (SWI)

Corrections applied via the $PATCH section do not affect the content of the total system software tape. Patches can be edited onto an unpatched total system software tape via the Patch Editor (PAED) program. (Refer to the GCOS 8 OS Service Routines manual for PAED program procedures.)

Corrections to the Startup program and the Startup job stream content are not applied via the $PATCH section. Among these types of corrections are:

- MASK statements that correct the Startup module (.MINIT). (Refer to "Patching Startup Program" in Section 3)
- OCTAL statements that correct the MPC Bootload module (.MMPCB). (Refer to "$INITIALIZE Section" earlier in this section.)
- HEX statements that correct firmware data decks. These statements are CHEX (control store) and RHEX (read/write memory) statements. (Refer to "$INITIALIZE Section" earlier in this section.)
- OCTAL statements that correct modules loaded via the $LOAD section of the Startup job stream. (Refer to "$LOAD Section" later in this section.)

Application of Patches

Patch corrections included in the $PATCH section are written to a main memory table during system bootloading. This table is accessed each time a GCOS or software program/module, for which patches exist, is loaded into memory from system storage. All applicable patches are applied to the program/module while it is in memory.

Programs/modules eligible for patching via the $PATCH section are identified in either the Module Directory Table or the Module Name Table. (These tables are pointed to by the SD.MDD and the SD.MNT segment descriptors, respectively.) Table information includes the address of the program/module loaded from system storage and an indicator denoting whether any patches exist in the patch table for the program/module.

The patch table is initialized each time the system is bootloaded. Only the content of the $PATCH section for that bootload is written to the table. Therefore, only current corrections exist in the table. Also, any OCTAL correction statements or PATCH_RUN_UNIT statements that were removed from the $PATCH section are eliminated from the table.

There is no limitation to the number of patches that this table can accommodate. However, it is recommended that the site consider the following when determining the number of patches to include in the $PATCH section (i.e., in the patch table):

1. The entire patch table is scanned each time a system program is called into memory. An abundance of patches could impact system performance if a significant amount of time is required to complete this scan.

2. Each patch correction occupies two words of main memory.

Slave Service Area (SSA) modules are limited in size. This limitation must be considered when applying add-on patches that expand the size of the module. Each main memory (HCM) module contains space for patches and patches must not overflow this space, or an error condition may result (e.g., the module may not execute properly).

Note that if the job calling for a program/module includes a $ EXTEND control statement in its job control language, patches from an E* file also must be applied while the program/module is in main memory. If patches for the same program/module location exist in both the $PATCH section and on the E* file, the E* file patches override $PATCH section corrections for that location. In addition, programs on a ** file can be patched only via the E* file. (Refer to the GCOS 8 OS Service Routines manual for procedures to create an E* file and for the E* file format.)

OCTAL Statement Format

The name of the program/module to be patched, the location at which to apply the correction, and the correcting data are specified on an OCTAL correction statement. Only the fields containing this information - as well as the field containing the OCTAL identifier - are recognized by Startup. While other information can be included on the statement, it is for documentation purposes only, and ignored by Startup.

The format for the OCTAL statement is:

```
                1              3                                 7
1     8         6              2                                 3
_________________________________________________________________

aaaaaa OCTAL    vvvvvvvvvvvv   cccccc                            nnnnnn
```

where:

aaaaaa - Module/program address (octal) at which to apply this correction. This value must be left-justified and cannot exceed six digits. If more than six digits are specified, patch data may be applied to the wrong location.

Patches must be floatable. The address in field aaaaaa (as well as an address in field v...v) must not require relocation prior to execution of the module. For example, if patching location 12122 (the offset begins at location 12000), the address specified must be 12122 and not relative address 122.

v...v - Patch correction data (octal) to be applied at module/program location aaaaaa. A maximum of 12 digits can be specified. If more than 12 digits are specified, erroneous data may be applied at location aaaaaa.

A single OCTAL correction statement can contain more than one patch correction. However, each correction on the statement is counted as one patch occupying two words in main memory.

Inclusion of leading zeros in the above fields is optional.

cccccc - Patch-related comments. These comments can extend from column 32 through column 72. Information often specified in this area includes comments describing the code or patch function, the number of the SER describing the error, the library number, the edit name of the module being corrected, the TTL date, and the number of the transmittal describing the correction.

nnnnnn - Program/module name. If this is a GCOS program/module, nnnnnn is the BCD catalog name in the format .Mmmmm (where mmmm = edit name of the module). For example, .MALC1 defines the ALC1 (Peripheral Allocator program) module.

If this is not a GCOS program/module, nnnnnn is the MME GECALL name (e.g., a System program/module such as COBOL, GELOAD). The maximum length of this name is six characters.

In both cases, the name nnnnnn must be the same as the name used when the file containing the program/module was created in system-loadable format. (This name is specified in the CATALOG= field of the $ SYSLD JCL statement for the System Library Editor to use when editing the program/module onto system storage.)

The following OCTAL correction statement defines patch information that is to be applied to location 23562 of the Peripheral Allocator program (.MALC1 module).

```
                 1                  3                 5              7
1      8         6                  2                 4              3
-------------------------------------------------------------------------
23562  OCTAL     10604004           TMI TRYCH1,$      101071ALC140918122E.MALC1

Octal            Octal              Patch-Related     SER#  Edit TTL    Catalog
Address          Data               Comment                 Name Date   Name

                                                        Library   Transmittal
                                                        Number    Log Number
```

Four errors are commonly associated with use of the OCTAL correction statement:

1. Attempting to apply two OCTAL statements containing different patch information to the same program/module location. Startup ignores the second OCTAL statement encountered and issues the informational message:

   *OVER-PATCH CARD BELOW IGNORED

   to the system console. An image of the ignored statement follows the message.

2. Attempting to apply two OCTAL statements containing the same patch information to the same program/module location. Startup ignores the second OCTAL statement encountered and issues the informational message:

   *DUPLICATED PATCH CARD BELOW IGNORED

   to the system console. An image of the ignored statement follows the message.

3. Specifying a non-existing module name in either the Module Directory Table or in the Module Name Table. Startup ignores the statement and issues the informational message:

   *PATCH CARD BELOW CONTAINS UNDEFINED PROGRAM NAME to the system console. An image of the ignored statement follows the message.

4. Failing to begin the octal address or patch data in columns 1 and 16, respectively, or including non-octal data in these fields. Startup ignores the statement and issues the informational message:

   *PATCH CARD BELOW CONTAINS ERROR NEAR COLUMN nn

   to the system console (where nn = column in which the error begins). An image of the ignored statement follows the message.

Upon completion of $PATCH section processing by Startup, if any errors are encountered, the operator is given the opportunity to either continue with or abort the bootload. The message:

*ANSWER YES TO CONTINUE WITHOUT ABOVE PATCHES

is issued to the console. If the operator responds YES, Startup ignores the statements in question and continues the bootload. If the operator responds NO, Startup aborts. Remove or correct the statements in question before reinitiating the bootload.

Another format of the OCTAL statement allows a contiguous sequence of data-words to be given on one statement; i.e.,

```
22     OCTAL   0,0,0,0
```

means patch four contiguous words starting at address 22.

PATCH_RUN_UNIT Statements

PATCH_RUN_UNIT (or PATCH_RU) statements in the Startup job stream are only valid if

1. The run unit to be patched is part of the Seed Workspace Image (SWI); and

2. A non-empty segment to contain these patches is provided via $ INFO NPCHSZ/nnn.

Advantages And Disadvantages To $PATCH Section Use

In addition to the $PATCH section, patch corrections can be applied to GCOS programs and software programs via the Patch Editor (PAED) program. As previously indicated, the PAED program applies the patches to the total system software tape. The following summarizes some of the advantages and disadvantages to using the $PATCH section rather than the PAED program. (Refer to the GCOS 8 OS Service Routines manual for PAED program procedures.)

ADVANTAGES OF USING $PATCH SECTION

The advantages of applying patches via the $PATCH section include its flexibility and compatibility features.

1. Patch corrections are easy to insert into and remove from the Startup job stream. For example, applicable patches can be manually inserted into or removed from the $PATCH section to meet various configuration and operational requirements. Corrections that are applied via the PAED program are of a more permanent nature and, once applied, lose their immediate visibility.

   Because patch corrections are easily inserted and removed, the $PATCH section approach is a good method for validating new patches. Once validation is complete, the patches can be removed from the $PATCH section and applied via the PAED program.

   Temporary patches that are applied only once or for a short period of time usually are applied via the $PATCH section. It also is recommended that local patches be applied via the $PATCH section. Local patches are patches generated by a site and are neither Honeywell Bull-generated site-options nor officially released corrections.

2. All patches generated for use in the $PATCH section are fully compatible with PAED program requirements (i.e., the patches are interchangeable). With few exceptions, the patching results are the same under each approach. (These exceptions are the specific programs/modules that cannot be patched via the PAED program. If an attempt to do so is made, the patches are rejected and the programs/modules are identified on a report generated by the PAED program. The patches must be removed from the PAED job stream and inserted into the $PATCH section.)

3. Some patches, primarily those that cannot be edited onto the total system software tape via the PAED program, must be applied via the $PATCH section. In addition, the $PATCH section must be used to apply patches if the total system software tape has been patched. (The total system software tape can be patched only once and cannot be used as input to another PAED program edit.)

   Also included among the patches that cannot be applied via the PAED program are add-on patches, which are applied below the Lower Address Limit (LAL) of the module being patched, and patches applied from a common patch area. Patches included in a common patch area are below the LAL of the individual modules.

   Note that use of common patch space is more economical if the module defines one or more common patch areas in the main overlay at the lower-address end of memory. This common patch area can be in addition to patch space within each overlay. All other overlays also can use this space for needed add-on patches.

DISADVANTAGES OF USING $PATCH SECTION

The disadvantages of applying patches via the $PATCH section include the fact that they encourage oversights, are time consuming, and corrections may be misplaced.

1. Because this approach is easy to use, patch corrections often are not well-planned. Inefficiencies or errors can result.

2. Each time the affected program/module is loaded into main memory for execution, a series of events must occur before patches are applied. (The Module Directory Table and the Module Name Table are searched, a determination is made as to whether there are applicable patches, the patch table is searched for the applicable patches, and patches are applied to the program/module.) This consumes resources and increases load time.

3. Because patch corrections are incorporated into a card deck that is frequently used, cards often are lost, damaged, or inadvertently moved outside of the $PATCH section. A patch residing outside the $PATCH section may result in an error condition.

Changing PAED-Applied Patches Via $PATCH Section

The $PATCH section can be used to modify/add/delete an existing patch that was applied via the PAED program. It is recommended that this approach be used only as an emergency procedure and for testing purposes. If the $PATCH section is used for this purpose, conflicts may arise between the patches in the $PATCH section and in the PAED job stream. Therefore, tight control must be maintained and the replacement patches must be clearly defined.

The *C Editor (SCED) program provides an alternative to use of the $PATCH section for the purpose of modifying/adding/deleting PAED-applied patches. (Refer to the GCOS 8 OS Service Routines manual for *C Editor program procedures.)

- o To modify an existing patch via the $PATCH section, create an OCTAL correction statement containing the correct information and insert the statement into the $PATCH section. The correcting patch will override the erroneous patch that was applied via the PAED program.

- o To delete an existing patch via the $PATCH section, prepare an OCTAL correction statement for the affected location. (The original image can be obtained from the ORIG. CONTENTS field of the most recent PAED run report.) This statement resets the contents of the location to their value prior to the PAED run. Then, insert the OCTAL correction statement into the $PATCH section.

- o To add a new patch via the $PATCH section, create an OCTAL correction statement containing the desired patch information and insert the statement into the $PATCH section.

Site-Option Patches

A site can selectively apply special operating system parameters via OCTAL correction statements. These statements and parameters are referred to as site-option patches and are applied to GCOS system programs (modules) via the $PATCH section. Applicable OCTAL correction (patch) statements for site-option patches are defined below.

DISPATCHER OPTIONS

The site can manipulate certain parameters (factors) within the Dispatcher algorithm to help meet processing objectives. The following site-option patch, applied to location 100 of the .MDISP module, allows the site to adjust the job urgency and I/O priority parameters.

```
                 1                                          7
1       8        6                                          3
-------------------------------------------------------------------
100     OCTAL    w000xxy0000z                               .MDISP
```

where:

w - Octal value that expands to:

Bit 0 =1, I/O ratio priority enabled
    1 =1, Urgency priority control enabled

xx - Bits 14-17 specify the consecutive subdispatch count

y - Octal value that expands to:

Bit 18 =1, Class B priority enabled
    19 =1, Class A priority enabled

z - Reserve an entry in the Dispatcher's queue (i.e., job priority table) for a job that is to be given Class A priority via the console entry APRIOR sssss (where sssss = SNUMB). The value z is the priority level that is to be assigned to job sssss and can range from 1-5. If the value specified is 0, the operator cannot assign Class A priority via the console.

Variations in the Dispatcher algorithm are:

1. Bits 0 and 1 can be varied to control dispatching.

0 1	Description
0 0	Dispatch according to the order in which programs are placed in the Dispatcher queue (i.e., first-in, first-out basis).
0 1	Dispatch according to the adjusted urgency value for each program, found in the process status (PST) segment.

0 1	Description
1 0	Dispatch according to the ratio of channel time to processor time.
1 1	Dispatch according to the adjusted urgency value, found in the process status (PST) segment, weighted by ratio of channel time to processor time.

2. If bit 18 =1, Class B priority programs receive preference in main level dispatches, subject only to the number of nonpriority dispatches specified.

3. If bit 19 =1, Class A priority programs are dispatched according to their priority level. The priority level ranges from 1-5 (1 is the highest). Class A programs are given preference in courtesy calls and main level dispatches. Total system throughput may be significantly slowed in that all other processing halts when this option is in effect. For this reason, it is recommended that Class A priority be used only for programs of extreme urgency.

Applying Class A Priority

A maximum of three Class A priority entries can be reserved in the Dispatcher queue for specific SNUMBs. To reserve an entry in this priority table, each program must be identified via separate patches to locations 104, 105, and 106, respectively, of the .MDISP module. The format for these patches is:

1	8	16	73
aaa	OCTAL	sssssssss0p	.MDISP

where:

aaa - 104, 105, or 106.

s...s - SNUMB of the program being given Class A priority.

p - Priority level (1-5).

Inclusion of a SNUMB in locations 4-6 (OCTAL) of .MDISP does not necessarily invoke Priority A. The presence of a SNUMB only causes DISP initialization to include the job in the priority job table. Its position in the table is determined by the value of bits 30-35.

To assign Priority A, call "DISP,22" either through program initiation via .CALL, or by the operator issuing an APRIOR console verb. The APRIOR verb can only be used, however, if location 0 of .MDISP contains 01-05 (OCTAL) in bits 30-35. These bits should differ from the value of bits 30-35 in words 4-6, as this level indicates the position of the operator entry into the priority table, of which there is only one.

If a job with a SNUMB in locations 4-6 does not generate a .CALL DISP,22, an APRIOR console verb must be used to initiate Priority A, as APRIOR uses the reserved operator entry. Inclusion of a SNUMB in locations 4-6 is only necessary with TSS or TDS, neither of which call DISP,22.

NOTE: The APRIOR DELETE console verb only affects those jobs executing with A Priority as a result of the operator entering an APRIOR console verb. A job which calls DISP,22 cannot be removed via the APRIOR console verb. The call must be made within the job.

Applying Class B Priority

A maximum of three programs, which can include Transaction Driven System (TDS), Transaction Processing System (TPS), and Time Sharing System programs, can be given Class B priority. (The patch format for TSS and TDS applications is given below under "Assigning Class B Priority To Time Sharing Copies And TDS.") Each of these programs must be identified via separate patches to locations 101-103, respectively, of the .MDISP module. The format for the TPS patch is:

```
                1                                             7
1      8        6                                             3
-----------------------------------------------------------------
aaa    OCTAL    ssssssssssdt                                  .MDISP
```

where:

aaa - 101, 102, or 103.

s...s - SNUMB of the program being given Class B priority.

d - Number of dispatches permitted to jobs without Class B status (between dispatches to jobs with Class B status). The default value is 7.

t - Time quantum (i.e., time slice), in 32-millisecond units, for each job with Class B status. If this value is 0, a standard time quantum is applied. The default value is 92 milliseconds.

Class B priority is given to a program only when that program calls entry point 21 of .MDISP. Time Sharing System, TPS, and TDS programs issue this call during their initialization.

Assigning Class B Priority To Time Sharing Copies And TDS

The following site-option patch can be used to assign Class B priority:

1. To all configured copies of Time Sharing and/or
2. To all Transaction Driven System (TDS) and DM-IV Transaction Processing programs. Separate patches are required if Class B priority is applied to Time Sharing as well as TDS and DM-IV Transaction Processing.

If multicopy Time Sharing is configured, this patch must be applied. Class B priority is assigned to all copies.

1	8	16	73
aaa	OCTAL	xxxxxx000000	.MDISP

where:

aaa - 101, 102, or 103.

xxxxxx - 632462 for TDS and 636262 for TSS.

WRAPUP CONTROL

The following site-option patch to the .MBRT1 module can be used to enable/disable a wrapup request after an abort caused by an SSA module:

1	8	16	73
44	OCTAL	n	.MBRT1

where:

n = Zero - Disable wrapup request if abort was caused by SSA (default).

n = Nonzero - Enable wrapup request if abort was caused by SSA.

NOTE: Enabling the wrapup request may make file-system-related sensitive information such as userids, passwords, etc., available to unauthorized slave processes.

PERIPHERAL ALLOCATOR OPTIONS

A site can alter the disposition of several Peripheral Allocator functions by applying the following site-option patch to the .MALC1 module:

```
              1                                         7
1      8      6                                         3
---------------------------------------------------------
100    OCTAL  nnnnnnnnnnnn                          .MALC1
```

The nnnnnnnnnnnn field is a 12-character octal code defining the following functions. If a bit is set (=1), the corresponding function is enabled:

Bits	Function
0	No GET message is issued to the system console for magnetic tapes.
1	No GET message is issued to the system console for local batch jobs.
2	No HOLD/LIMBO/REST/SIEVE messages are issued to the system console.
3	No OVERDUE message is issued to the system console.
4	No GET message is issued to the system console for a print train.
5	No activity restart is performed.
6	Nine-track magnetic tapes are standard.
7	No circular allocation of magnetic tape handlers is to be performed.
8	No job sieve limits are to be applied.
9	No BMC spinoff is to occur; BMC activities are run in sequence.
10	No print banners are to be printed on output generated on the device defined on a $ PRINT statement.
11	Peripheral Allocator program debug mode. If the Peripheral Allocator initiates a fault, a system failure occurs and a master mode dump is taken.
12	No GET messages are issued to the system console for disk packs.
13	No automatic restart from checkpoints is to be performed.
14	No "GRANT" command required for privity job.
15	Used by GCOS for $ SHRNM.
16	User wants "-n" appended to SYSPN SELECT filename, where n = system number.
17-25	Reserved for GCOS.
26-29	This field is used for information only. It contains the system high-density code for magnetic tape, obtained from .CROPT, which is initialized at System Startup (from $ INFO DENxx). Any data included on the site-option patch in this field is ignored.
30-31	Reserved for GCOS.
32-35	This field is used for information only. It contains the system low-density code for magnetic tape, obtained from .CROPT, which is initialized at System Startup (from $ INFO DENyy). Any data included on the site-option patch in this field is ignored.

The system density codes for magnetic tape are:

01 - 200 bpi (bits per inch)	11 - 1600 bpi
02 - 556 bpi	14 - 6250 bpi
04 - 800 bpi	

SUPPRESS CONSOLE MESSAGES AT REEL SWITCH TIME

If a multireel tape file is being written via GFRC and an alternate tape drive is in use, this option can suppress the "LOCATE SCRATCH..." and "VERIFY SCRATCH" messages, provided that the tape mounted on the alternate drive has a proper header label.

The format for this site-option patch is:

1	8	16	73
1	OCTAL	777777625012	.GLLMO

NOTE: The use of this option will remove all operator interaction at reel switch time, provided that the tape mounted on the alternate drive has a readable header label. If the wrong tape was accidently mounted, it will be overwritten unless it generates an "OHLR..." message for unexpired retention, etc.

RELEASE DISK SPACE

The site can check the Master Available Space Table (BIT MAP) for temporary space to be released or to release space on any descriptor, AST, or BIT MAP error with the following patch:

1	8	16	73
7	OCTAL	uuuuuulllll	.MALC9

where:

uuuuuu - ≠ 0. Requests temporary BITMAP check. Temporary space is to be released providing it is not marked as permanent space in the Master Available Space Table (BIT MAP).

llllll - ≠ 0. ZOP any descriptor error, AST error, or BITMAP error.

FMS TABLE SPACE FOR DUPLICATED FILE OR TEST MODE ALLOCATIONS

The site can modify the default size (5K) of the table that is used for duplicate file control or test mode allocations by applying the following patch.

1	8	16	73
36	OCTAL	0000nn000000	.MFSEX

where:

nn - Number of 1K (1024 word) blocks of memory to be reserved. Requests for greater than 31K will be reduced to that value. In general, eight allocations can be made in each 1K block.

36 - Symbolic location CMBLK.

MEMORY ALLOCATOR OPTIONS

Several site-option patches can be applied to the .MPOPM module to control such functions as magnetic tape positioning, trace table size, mass storage device preference, or change the $ IDENT, $ SELECT, or $ USERID images for console spawned jobs. These functions and the patch card formats are described below.

Magnetic Tape Disposition At .MPOPM Rollcall

The site can control the positioning of magnetic tapes during the .MPOPM module's rollcall function. If no site-option patch is applied, tapes are rewound and placed in a ready condition. If the following site-option patch is applied to location 27 of the .MPOPM module, tapes are rewound and placed in a standby condition:

1	8	16	73
27	OCTAL	720000020001	.MPOPM

Statistical Collection Tape Disposition At .MPOPM Rollcall

The site can control the positioning of the statistical collection tape (SCT) during the .MPOPM module's rollcall function. SCT positioning also may be dependent upon the operator's response to the *SCF CONTINUATION? question during Startup's question-and-answer sequence. The following optional approaches are available:

1. A response of NO to the *SCF CONTINUATION? question overrides any site-option patch to location 26 of .MPOPM. The SCT is rewound and left in a standby condition.

2. A response of YES to the *SCF CONTINUATION? question and application of the following site-option patch to location 26 of .MPOPM results in the SCT being left in a ready condition and available to continue statistical collection. (Note that the tape is not rewound.)

1	8	16	73
26	OCTAL	400000020001	.MPOPM

3. A response of NO to *SCF CONTINUATION? and no site-option patch to location 26 of .MPOPM results in the SCT being rewound during the .MPOPM rollcall function. The tape is left in a ready condition and is repositioned to continue statistical collection.

Trace Tables

A site can maintain both a primary and a secondary trace table. Each trace entry in these tables comprises four words.

The primary trace table by default accommodates 511 entries. The following site-option patch can be applied to modify the number of trace entries the table can accommodate:

1	8	16	73
24	OCTAL	0000000nnnnn	.MPOPM

where:

nnnnn - Number of entries the primary trace table can accommodate. A maximum of 4096 (10000 octal) entries can be specified.

A secondary trace table can be specified when trace entries are written to magnetic tape. By configuring two trace tables, a double-buffer effect results. While a trace entry is being written into one of the tables, another trace entry is being read from the other table and written to magnetic tape. Under these circumstances, both the primary and secondary tables must accommodate 1024 entries (i.e., a total of 2048 entries).

1	8	16	73
25	OCTAL	0000000nnnnn	.MPOPM

where:

nnnnn - Number of entries that the secondary trace table can accommodate. A maximum of 4096 (10000 octal) entries can be specified. The default value is 0 (zero).

Mass Storage Device Type Preference List

The following site-option patch, when applied to location 31 of the .MPOPM module, permits a site with multiple types of mass storage devices to specify its own preference list of device types for device allocation. The patch contains a preference-ordered list for allocating the different types of mass storage devices. (This list is similar to a user's device type request on a $ FILE statement.)

The variable field can specify up to six device codes. The position of a device code in the variable field determines the order of preference (i.e., the most desirable device type is specified in field aa, followed by device types bb, cc, dd, ee, and ff). If fewer than six fields are used, unused fields (i.e., low-order fields) must be zero-filled.

1	8	16	73
31	OCTAL	aabbccddeeff	.MPOPM

where:

aa, bb, cc, dd, ee, ff - Device codes (octal).

Device Code	Device Type
64	MSU0400 or MSU0402
65	MSU0451
66	MSU0500
67	MSU0501
73	MSC1A (64-word sector format)
53	MSC1B (512-word sector format)
75	MSD1A (64-word sector format)
55	MSD1B (512-word sector format)
42	MSM1A (64-word sector format)
41	MSM1B (512-word sector format)
74	MSM1E (512-word sector format)

This patch to .MPOPM does not take effect unless round robin device allocation is bypassed via the following patch to .MALC6:

1	8	16	73
16	OCTAL	1	.MALC6

IDENT Image For Console Spawned Jobs

If the site needs to change the $ IDENT image for console spawned jobs, patches are required to the location(s) described below, in the following format:

```
                1                                       7
1       8       6                                       3
-------------------------------------------------------------
aa      OCTAL   bbccddeeffgg                            .MPOPM
```

where:

aa - 33-45 (OCTAL). At least one, and as many of these locations as required, may be specified.

bb...gg - The octal values representing the codes for the BCD characters of the $ IDENT image desired, beginning with the data for card column 16. Note that the last statement should end with the octal value "20" (blank), as this position is ignored.

Maximum Number Of DRL TASK Jobs Allowed

The maximum number of DRL TASK jobs allowed by default is 10 (decimal). This site-option patch can change that default value.

```
                1                                       7
1       8       6                                       3
-------------------------------------------------------------
47      OCTAL   00000000nnnn                            .MPOPM
```

where:

nnnn - The maximum number (octal) of DRL TASK jobs permitted.

USERID Image For Console Spawned Jobs

If the site needs to change the $ USERID image for console spawned jobs, patches are required to the location(s) described below, in the following format:

```
                1                                         7
1      8        6                                         3
-----------------------------------------------------------
aa     OCTAL    bbccddeeffgg                          .MPOPM
```

where:

aa - 50-62 (octal). At least one, and as many of these locations as required, may be specified.

bb...gg - The octal values representing the codes for the BCD characters of the $ USERID image desired, beginning with that data for card column 16. The image must be in the form: userid$password. Note that the last statement should end with the octal value "20" (blank), as this position is ignored.

USERID Image Of Select File For Console Spawned Jobs

If the site needs to change the $ SELECT image for console spawned jobs, patches are required to the location(s) described below, in the following format:

```
                1                                         7
1      8        6                                         3
-----------------------------------------------------------
aa     OCTAL    bbccddeeffgg                          .MPOPM
```

where:

aa - 64-76 (octal) at least one, and as many of these locations as required, may be specified.

bb...gg - The octal values representing the codes for the BCD characters of the $ SELECT image desired, beginning with the data for card column 16. The image must be in the format: userid.

Note that the last statement should end with the octal value "6120" (slash blank).

Bypass Post-Mortem Data

The site can elect to bypass the recovery of Post-Mortem data in module .MPOPM for Restart, SYSOUT Recovery, and Phase-1 Recovery processing. Cleanpoint data must be used when this patch is applied to ensure that Restart, SYSOUT Recovery, and Phase-1 Recovery processing do not fail.

The format for this patch is:

```
               1                                          7
1      8       6                                          3
__________________________________________________________
101    OCTAL   n                                     .MPOPM
```

where:

n - Any non-zero octal value.

Urgency Assignment

This site-option patch can be applied to define the maximum urgency available to an operator through the URGC verb. The maximum urgency permitted is 63 (77 octal). The default is 40 (50 octal).

```
               1                                          7
1      8       6                                          3
__________________________________________________________
20     OCTAL   nn                                    .MPOPN
```

where:

nn - Any non-zero octal value up to 77.

MEMORY RELEASE OPTION

This site-option patch is applicable to DPS 8 and DPS 8000 only. It specifies the limit for the number of MEMREL spawns when attempting to release memory. If the requested release has not been accomplished when this limit is reached, POR5 is notified to shut down the memory release process and to return any memory already released back to the system.

```
               1                                          7
1      8       6                                          3
__________________________________________________________
121    OCTAL   00000n000000                          .MRASE
```

where:

n - Retry value (default = 3).

VIRTUAL MODE MEMORY MANAGEMENT PARAMETER

This virtual mode memory management parameter designates the initial value for the target page fault rate. It is used in conjunction with the other native mode memory management parameters supplied via $ INFO statements. Refer to the GCOS 8 OS System Operating Techniques manual for more information about native mode memory management parameters. The default value is 31 (25 decimal).

1	8	16	73
23	OCTAL	nn	.MDMM1

FAULT PROCESSING OPTIONS

The site can control several fault processing functions by applying the following site-option patches to the .MFALT module. A patch containing all zeros is ignored; only a patch that contains a nonzero value is accepted.

Abort On Master Mode Faults

The following site-option patch causes a system abort on all master mode faults except MME, Timer Runout, Connect, Overflow, and Divide Check.

1	8	16	73
100	OCTAL	7	.MFALT

Disable Instruction Retry

The following site-option patch disables the Instruction Retry function and releases the memory (approximately 1/2 K) allocated to this feature.

1	8	16	73
107	OCTAL	7	.MFALT

Abort On Faults Outside MME GELBAR Limits

The following site-option patch causes a system abort if any system program (program numbers 1-10) faults in slave mode outside of MME GELBAR limits (except MME, Timer Runout, Connect, Overflow, and Divide Check faults).

1	8	16	73
102	OCTAL	7	.MFALT

Processor Error Record Buffers

The following site-option patch specifies the number of processor error record buffers to allocate:

```
              1                                           7
1      8      6                                           3
---------------------------------------------------------------
103    OCTAL  n                                       .MFALT
```

where:

n - Number of processor error record buffers desired. The default value is 2 or the number of processors configured plus 1 (whichever value is greater). If the value specified on this statement is less than the default value, the default is applied. A maximum of seven buffers can be specified.

One buffer is required for processing each fault that locks a history register.

System Lockup Fault Control Value

The following site-option patch can be applied to override the standard system lockup fault control value. The value must be specified in bits 34 and 35.

```
              1                                           7
1      8      6                                           3
---------------------------------------------------------------
106    OCTAL  n                                       .MFALT
```

where:

n - Fault control value (1, 2, 3, or 4). The value 4 indicates that 0 is desired (a zero value is rejected). The default value is 3.

EXCEPTION PROCESSING STATISTICS COLLECTION

Inhibit Automatic Start Of ECOL

The following site-option patch inhibits the automatic start of ECOL during Startup.

```
              1         3                                 7
1      8      6         2                                 3
---------------------------------------------------------------
121    OCTAL  0         DON'T START ECOL              .MSCHD
```

ECOL Communications Words

The default parameters in the ECOL communications words can be modified by applying the following site-option patches to the .MFALT module:

```
                 1                                            7
1       8        6                                            3
---------------------------------------------------------------
111     OCTAL    0tttttti000cc                                .MFALT
```

where:

ttttt - MOS error message time threshold (in minutes, expressed in octal). Right-justify this value in bits 4-17. The default is 10 minutes.

i - Bits used to permit/inhibit fault logging and reporting functions.

BIT

18 must be zero

19 = 1, inhibit logging of memory faults. The default is to permit this function (i.e., bit 19 =0).

20 = 1, inhibit logging of illegal procedure (IPR) faults. The default permits this function (i.e., bit 20 =0).

cc - MOS error message count threshold. Right-justify this value in bits 31-35. The default value is 25.

```
                 1                                            7
1       8        6                                            3
---------------------------------------------------------------
112     OCTAL    ppppeeee000t                                 .MFALT
```

where:

pppp - MPC sample period in 0.1-minute increments (expressed in octal). This value can range from 0-4095. If the value is 0, sampling is disabled. The default value is 600 (60 minutes). Right-justify this value in bits 0-11.

eeee - MPC error threshold in 0.01 percent increments (expressed in octal). This value can range from 1-4095. The default value is 1 (0.01 percent). If the value is 0, the error threshold message reporting function is disabled. Right-justify this value in bits 12-23.

t - If bit 33 =1, inhibit monitoring of the magnetic tape MPC threshold. The default permits monitoring (i.e., bit 33 =0).

```
                1                                      7
1      8        6                                      3
_________________________________________________________

113    OCTAL    000000ggss00                      .MFALT
```

where:

gg - ECOL MME GEWAKE time in 0.1-minute increments. This value can range from 0-77 octal (0-6.3 minutes). The default value is 50 decimal (5 minutes). Right-justify this value in bits 18-23.

ss - Automatic retry summary logging period in 0.1-hour increments. This value can range from 0-77 octal. If the value is 0, automatic retry summary logging is inhibited. The default value is 10 decimal (1.0 hour). Right-justify this value in bits 24-29.

```
                1                                      7
1      8        6                                      3
_________________________________________________________

114    OCTAL    1                                 .MFALT
```

The patch above disables the logging of exception processing records. The default is to log these records.

CACHE MEMORY CONTROL OPTIONS

By default, the ECOL module assumes control of cache memory when it is enabled, unless the following site-option patch is used to place cache memory under operating system control. (Details on cache control and any associated console verbs are described in the GCOS 8 OS Distributed Maintenance Services Tools Reference Manual.)

```
                1                                      7
1      8        6                                      3
_________________________________________________________

105    OCTAL    1                                 .MFALT
```

The following patch, optionally, directs GCOS to control cache memory until ECOL is enabled. If the ECOL program is subsequently enabled, it assumes control of cache memory.

```
                1                                      7
1      8        6                                      3
_________________________________________________________

105    OCTAL    2                                 .MFALT
```

ECFILE Creation Patch

The Error Collection File (ECFILE) size is set to 1200 llinks for both minimum and default when created by the ECOL module, unless a file size value is entered at site-patch location 257 (octal) of the ECOL program.

The file size range that can be requested is 1200 to 262143 llinks (i.e., 2260 - 777777 octal). A request outside of this range results in the display of one of the following messages upon the console:

*FYI ECOL - FILE SIZE REQUEST > 262143, WILL ATTEMPT TO GET 1200 LLINKS

*FYI ECOL - FILE SIZE REQUEST WAS nnnn LLINKS, WILL ATTEMPT TO USE 1200 LLINKS

A request for file space in excess of 12000 llinks results in issuance of the following message:

*FYI ECOL - REQUESTED COLLECTOR FILE SIZE OF nnnnnn LLINKS IS VERY LARGE

The requested file space (either by request or by default) can be denied by the system if not enough space is available. When this occurs, ECOL generates a request for 100 llinks of space less than the requested amount, and continues to decrement the file space request by 100 llinks until either the file space request is accepted by the system, or the file space requested is smaller than 800 llinks, which causes ECOL to abort with an HU MME GEBORT. One of the following messages is issued in either case:

*FYI ECOL - COULD NOT OBTAIN AT LEAST 800 LLINKS PERM FILE SPACE, PLEASE RESTART ECOL WHEN ADEQUATE SPACE IS AVAILABLE

*FYI ECOL - PERM FILE SPACE NEEDED WAS nnnnnn LLINKS, SPACE OBTAINED WAS nnnnnn LLINKS

The file is created as a sequential file with read and write concurrency permitted. Specific read permission is given to SY_ELAN userid for ELAN report production. The format for this patch is:

```
              1                                                 7
1      8      6                                                 3
------------------------------------------------------------------
257    OCTAL  000000nnnnnn                                  .MECOL
```

where:

nnnnnn - 2260-777777 (octal). This value specifies the number of llinks requested to execute ECOL. This value must be right-justified.

Shared Controller/Device I/O Statistics Collection

Because of the differences in the collection processes of controller or device I/O statistics between ECOL and its predecessor, the HEALS program, ELAN I/O methods will not accurately collect statistics when devices are shared between two systems with different releases, unless the older HEALS module does the collection. The site option patch, described below, defines these circumstances for ECOL and allows the HEALS module to assume the task of statistical collection for shared devices between systems. Non-standard device statistics collection is performed normally for each of the other devices on each of the systems involved.

Site-option patch location 261 (octal) of .MECOL (bit 31) is used to control access of shared device statistics. Setting bit 31 to 1 in this location initializes bit 31 of .CREC1. (The default value is zero for both bits.) After initializing .CREC1, ECOL checks for one of the following conditions:

1. If bit 19 of the channel data within that entry of the Controller Information Table (pointed to by .CRCST) is 1, then the devices on this channel are considered shared. (Details of the Controller Information Table are described in the GCOS 8 OS System Tables manual.)

2. If bit 19 is set (as in step 1) and bit 31 of .CREC1 is 1 (per ECOL initialization, above), ECOL collects statistics for all devices on this channel.

3. If bit 19 is set (as in step 1) and bit 31 of .CREC1 = 0 (per ECOL initialization, above), ECOL does not collect statistics for any device on this channel.

4. If bit 19 is not set (as in step 1), this channel is considered unshared and no consideration is given to the content of bit 31 of .CREC1. Statistics are collected for all devices on this channel.

The format for this patch is:

1	8	16	73
261	OCTAL	0000000000n0	.MECOL

where:

n - 2 or 0 (the default is 0). The status of n determines the control access to shared device statistics and is checked by ECOL. If n = 2, a bit is set in a communication region cell for the MPCD program to check. MPCD uses the same criteria used by ECOL to determine whether to read firmware statistics or not.

NOTE: MPCD also checks bit 31 of .CREC1 to determine the status of ECOL statistics collection on the system shared devices. If bit 31 = 1, ECOL and MPCD assume that ECOL will collect statistics for all configured shared (and non-shared) devices. If bit 31 = 0, any statistical collecting between shared devices on one channel is done by the other system, and any statistical collecting between unshared devices or channels is done by the ECOL module in this system.

ECS Error Thresholds

The site-option patch for location 267 (octal) can be used to set ECS processor time and/or ECS error count thresholds. The format for this patch is:

```
              1                                      7
1      8      6                                      3
------------------------------------------------------
267    OCTAL  Ommmmm000ccc                      .MECOL
```

where:

mmmmm - 0 to 37777 (octal). This is the ECS error threshold time interval. The default value is 12 (octal).

ccc - 0 to 777 (octal). This is the ECS error count threshold. The default value is 31 (octal).

SYSTEM INPUT OPTIONS

Site-option patches can be applied to the .MGEIN module to modify and/or control several System Input program functions. The patches described below can be applied to the locations specified.

J And J File Space Allocation

The following patch can be applied to the .MGEIN module to modify the number of 320-word llinks initially allocated for *J subfiles and the J* file.

```
              1                                      7
1      8      6                                      3
------------------------------------------------------
114    OCTAL  xxxxxxyyyyyy                      .MGEIN
```

where:

xxxxxx - Number of llinks, expressed in octal, initially requested for *J subfiles. The default initial allocation size is 144 llinks.

yyyyyy - Number of llinks (modulo 12), expressed in octal, initially requested for the J* file for each job. The default initial allocation size is 12 llinks.

JCL Statement Examination And Modification

Several patches can be applied to the .MGEIN module to define the extent of the .MSCN1 module's responsibilities when examining and/or modifying JCL statements encountered during program execution. The .MGEIN module can be patched to specify examination and/or modification of JCL statements as defined by Figure 5-2. One bit is set for each JCL statement that the .MSCN1 module is to examine and/or modify.

For Resource Control Words statement types:

```
                1                                          7
1      8        6                                          3
------------------------------------------------------------------
116    OCTAL    nnnnnnnnnnnn                               .MGEIN
117    OCTAL    nnnnnnnnnnnn                               .MGEIN
```

where:

n...n - Octal value to expand to a binary value. A set bit (=1) specifies examining and/or modifying the corresponding JCL statement type from the Resource Control Words 1 and 2 list by .MSCN1.

For Activity Control Word 1 statement types:

```
                1                                          7
1      8        6                                          3
------------------------------------------------------------------
120    OCTAL    nnnnnnnnnnnn                               .MGEIN
```

where:

n...n - Octal value to expand to a binary value. A set bit (=1) specifies examine and/or modify the corresponding JCL statement type from the Activity Control Word 1 statement list by .MSCN1.

For Activity Control Word 2 statement types:

```
                1                                          7
1      8        6                                          3
------------------------------------------------------------------
121    OCTAL    nnnnnnnnnnn                                .MGEIN
```

Same as Activity Control Word 1 except Activity Control Word 2 list is applicable.

For Activity Control Word 3 statement types:

```
                1                                      7
1     8         6                                      3
--------------------------------------------------------
123   OCTAL     nnnnnnnnnnnn                      .MGEIN
```

Same as Activity Control Word 1 except Activity Control Word 3 list is applicable.

Stranger Statement Examination And Modification

The following site-option patch can be applied to the .MGEIN module to examine and/or modify JCL statements other than those listed in Figure 5-2. Each of a maximum of ten locations of the .MGEIN module can be patched with a six-character BCD representation of the statement type to be examined or modified and can be applied to either standard or site-unique statements. One card type can be identified in each of these locations. Further discussion of stranger options is included in the GCOS 8 OS System Programming Guide.

```
                1                                      7
1     8         6                                      3
--------------------------------------------------------
aaa   OCTAL     xxxxxxxxxxxx                      .MGEIN
```

where:

aaa - Location (124-135 octal) in the System Input program to be patched. Addresses cannot be duplicated.

x...x - 12-character octal representation of the BCD statement type.

Bit Definitions for:

Resource Control Word 1 (RESOR1)		Resource Control Word 2 (RESOR2)	
Patch Location 116		Patch Location 117	
Bit(s)	JCL Statement	Bit(s)	JCL Statement
0	Reserved for GCOS	36	$ DBASE
1	$ FORM	37	Reserved for GCOS
2	$ FFILE	38	Reserved for GCOS
3	$ FILE	39	Reserved for GCOS
4	$ PPS	40	Reserved for GCOS
5	$ PRINT	41	Reserved for GCOS
6	$ DATA	42	Reserved for GCOS
7	$ PUNCH	43	Reserved for GCOS
8	$ READ	44	Reserved for GCOS
9	$ CONSL	45	Reserved for GCOS
10	$ CONSLA	46	Reserved for GCOS
11	$ TYPE	47	Reserved for GCOS
12	$ SYSOUT	48	Reserved for GCOS
13	$ REMOTE	49	Reserved for GCOS
14	$ REPORT	50	Reserved for GCOS
15	$ REPTL	51	Reserved for GCOS
16	$ REPTR	52	Reserved for GCOS
17	$ PRMFL	53	Reserved for GCOS
18	$ DAC	54	Reserved for GCOS
19	$ TAPE	55	Reserved for GCOS
20	$ TAPE9	56	Reserved for GCOS
21	$ NTAPE	57	Reserved for GCOS
22	$ A1APK	58	Reserved for GCOS
23	$ A2APK	59	Reserved for GCOS
24	$ A1BPK	60	Reserved for GCOS
25	$ A2BPK	61	Reserved for GCOS
26	$ FILGP	62	Reserved for GCOS
27	$ GENFIL	63	Reserved for GCOS
28	$ 190PK	64	Reserved for GCOS
29	$ TAPE7	65	Reserved for GCOS
30	$ 191PK	66	Reserved for GCOS
31	Reserved for GCOS	67	Reserved for GCOS
32	$ 400PK	68	Reserved for GCOS
33	$ TAPE27	69	Reserved for GCOS
34	$ TAPE29	70	Reserved for GCOS
35	$ 450PK	71	Reserved for GCOS

Figure 5-2. .MSCN1 Function Words

Bit Definitions for:

Activity Control Word 1 (ACTIV1)		Activity Control Word 2 (ACTIV2)	
Patch Location 120		Patch Location 121	
Bit(s)	JCL Statement	Bit(s)	JCL Statement
0	Reserved for GCOS	36	$ FORT77
1	$ FORTY	37	$ F77V
2	$ GMAP	38	$ GMAP66
3	$ COBOL	39	$ FORTV
4	$ ADA	40	$ LKED
5	$ PL6	41	$ SHRUE
6	$ IDS	42	$ UTL8
7	$ PSM	43	Reserved for GCOS
8	$ CONVER	44	Reserved for GCOS
9	$ UTILITY	45	Reserved for GCOS
10	$ RUN	46	Reserved for GCOS
11	$ RERUN	47	Reserved for GCOS
12	$ SYSEDIT	48	Reserved for GCOS
13	$ EXTEDIT	49	Reserved for GCOS
14	$ S2PROG	50	Reserved for GCOS
15	$ FILEDIT	51	Reserved for GCOS
16	$ FILSYS	52	Reserved for GCOS
17	$ GMAPV	53	Reserved for GCOS
18	$ ADLIB	54	Reserved for GCOS
19	$ 355MAP	55	Reserved for GCOS
20	$ 355SIM	56	Reserved for GCOS
21	$ PROGRAM	57	Reserved for GCOS
22	$ EXECUTE	58	Reserved for GCOS
23	$ PL1	59	Reserved for GCOS
24	$ CBL74	60	Reserved for GCOS
25	$ LINKER	61	Reserved for GCOS
26	$ GENSYS	62	Reserved for GCOS
27	$ UTL2	63	Reserved for GCOS
28	$ CBL68	64	Reserved for GCOS
29	$ IDS2	65	Reserved for GCOS
30	$ CBL85	66	Reserved for GCOS
31	$ SRCLIB	67	Reserved for GCOS
32	$ LODLIB	68	Reserved for GCOS
33	$ OBJLIB	69	Reserved for GCOS
34	$ RPG2	70	Reserved for GCOS
35	$ FORTRA	71	Reserved for GCOS

Figure 5-2 (cont). .MSCN1 Function Words

Activity Control Word 3 (ACTIV3)

Patch Location 123

Bit(s)	JCL Statement	Bit(s)	JCL Statement
0	$ IDENT	28	$ STRANGER #3
1	$ SNUMB	29	$ STRANGER #4
2	$ LIMITS	30	$ STRANGER #5
3	$ NEED	31	$ STRANGER #6
4	$ PRIVITY	32	$ STRANGER #7
5	$ RESOURC	33	$ STRANGER #8
6-25	Reserved for GCOS	34	$ STRANGER #9
26	$ STRANGER #1	35	$ STRANGER $10
27	$ STRANGER #2		

Figure 5-2 (cont). .MSCN1 Function Words

Execution Report Message File

Site-option patches can be applied to the .MGEIN module to help control access to and security for the site-defined Execution Report message file. (Unless modified, the name of this file is MSG.)

The following patches can be applied to .MGEIN to change the default user identification.

```
                1                                               7
1      8        6                                               3
-----------------------------------------------------------------------
136    OCTAL    xxxxxxxxxxxxx                                   .MGEIN
137    OCTAL    xxxxxxxxxxxxx                                   .MGEIN
```

where:

x...x - Octal codes for six BCD characters of the user identification (i.e., catalog name) for the MSG file. A maximum of 12 BCD characters can be specified (i.e., if the identifier is more than six characters, the excess is specified in the second location). The default user identification is OPNSUTIL.

The following patch can be applied to .MGEIN to define a new name for the Execution Report message file.

```
                1                                                         7
1      8        6                                                         3
--------------------------------------------------------------------------------
140    OCTAL    yyyyyyyyyyyy                                         .MGEIN
```

where:

y...y - Octal codes for six BCD characters of the name of the file that contains the Execution Report message. A maximum of six BCD characters can be specified. The default file name is MSG.

The following patch can be applied to .MGEIN to suppress the GEIN SPAWN... console message.

```
                1                                                         7
1      8        6                                                         3
--------------------------------------------------------------------------------
141    OCTAL    n                                                    .MGEIN
```

where:

n - Any non-zero octal value

DEBUG OPTIONS AND SPECIAL ACTIONS

Location 144 of .MGEIN is reserved for special actions, and is patched as follows:

```
                1                                                         7
1      8        6                                                         3
--------------------------------------------------------------------------------
144    OCTAL    n                                                    .MGEIN
```

where:

n - A binary value such that a set bit (=1) implies a special action is to take place.

```
Bits 0-33 = currently unused
Bit    34 = force scan of all jobs
Bit    35 = abort the system
```

FLUSHING SYSOUT COLLECTION SPACE

Refer to $ INFO SYBRT/n parameter for flushing of SYSOUT collection space.

INPUT/OUTPUT SUPERVISOR (IOS) OPTION

The following site-option patch can be applied to significantly reduce the overhead associated with I/O interrupt processing. The patch ensures a return to an interrupted program whenever an I/O interrupt occurs (unless the terminating I/O operation requested a courtesy call and the interrupted program is executing at main level). Without this patch, the interrupted program is removed from execution to allow redispatching of the processor following interrupt processing.

1	8	16	73
60	OCTAL	0	.MIOS

TEMPORARY DISK FILE GROWTH

The amount of disk space is calculated by the system for a GROW request for a temporary file if the amount of space requested is 0(zero). The calculation is made as follows:

Growth amount = (current file size / ddd) + aaa

where ddd is a constant divisor and aaa is a constant adder, both expressed in octal. The default value for ddd is 4 and for aaa is 8. The default values may be changed via the following site-option patches:

1	8	16	73
12	OCTAL	ddd	.MMORE
13	OCTAL	aaa	.MMORE
14	OCTAL	fff	.MMORE

If fff is non-zero, the number of links in the GROW request is ignored and the system calculates the growth amount as above. The default value for fff is zero.

MME GEMORE RETRIES

If a MME GEMORE request is denied for an internal GCOS Operating System reason, i.e., .RBRT or some other appropriate bit is set in the .SRQST or .STATE words, this site-option patch allows the site to determine the number of times the request is to be retried (default is 20) before returning the denial status code to the process.

```
                1                                       7
1      8        6                                       3
---------------------------------------------------------
50     OCTAL    nnnnnn000000                       .MPMME
```

where:

nnnnnn - Octal value of the number of retries to be attempted before taking the denial return.

NOTE: Each retry results in a .CALLX to .MDISP entry point 16, which takes the process out of execution and puts it back in the dispatcher queue to wait for the next dispatch. On a busy system this could result in enough delay (even with the default value of 20) to allow the condition to be cleared by GCOS and allow the MME GEMORE request to continue. During periods when the system is relatively idle, each retry will be almost instantaneous. Thus the Site Administrator should use discretion in choosing the value for nnnnnn.

MME GESNUM AND PMME GENSNB OPTIONS

This site-option patch allows the range of SNUMBs generated by MME GESNUM and PMME GENSNB to be extended by using a selected sequence of characters of the English alphabet (excluding A, E, I, O, U, Y, and non-alphanumeric characters) in character position 1 and/or in character positions 2 - 4 of the generated SNUMB.

```
                1                                       7
1      8        6                                       3
---------------------------------------------------------
47     OCTAL    0000mm0000nn                       .MPMME
```

where:

mm - Octal value in the range 01 - 36, representing (see below) the last character to be used in character position 1 of the generated SNUMB (default mm = 12).

nn - Octal value in the range 01 - 36, representing (see below) the last character to be used in character positions 2 - 4 of the generated SNUMB (default nn = 12).

01	02	03	04	05	06	07	10	11	12	13	14	15	16	17	20	21	22	23	24	25	26	27	30	31	32	33	34	35	36
0	1	2	3	4	5	6	7	8	9	B	C	D	F	G	H	J	K	L	M	N	P	Q	R	S	T	V	W	X	Z

For example, 000012000013 would allow a B to be generated in character positions 2 - 4 of the generated SNUMBs in addition to the normal numerics.

NOTE: SNUMBs of site-spawned jobs should be considered when setting the above values, since there is a possibility of duplicate SNUMBs.

TIME SHARING SYSTEM OPTIONS

Several Time Sharing System functions can be controlled via site-option patches to the Time Sharing Executive program (.MTIMS). These functions and their patch formats are defined below. Note that the absolute location specified in the format is equated to a symbolic location.

Processor Time For DRL TASK Job

The following patch specifies the maximum allowable time for DRL TASK jobs:

```
                1                                    7
1       8       6                                    3
-----------------------------------------------------------
113
OCTAL
nnnnnnnnnnnn
.MTIMS
```

where:

n...n - Maximum processor time (octal), in increments of 1/64 millisecond, for a job using DRL TASK

113 - Symbolic location .TTPRC.

Maximum Number Of Keyboard-Display Terminals

The following site-option patch specifies the maximum number of keyboard display terminals (VIP type) permitted on the Time Sharing System at any one time.

```
                1                                    7
1       8       6                                    3
-----------------------------------------------------------
116     OCTAL   000mmm00nnnn                         .MTIMS
```

where:

mmm - Maximum number (octal) of keyboard-display terminals permitted at any one time. This value cannot exceed the number of Time Sharing System users. The default value is 0.

nnnn - The size of the input buffer which a VIP type terminal is allocated. The default value is 512 (1000 octal) words.

116 - Symbolic location .T760

Maximum Number Of Terminals

The following site-option patch specifies the maximum number of terminals allowed on the system (including keyboard-display terminals and the statistical record (TSRI) accounting file):

```
                1                                                         7
1       8       6                                                         3
_______________________________________________________________________________
117     OCTAL   000000000nnn                                              .MTIMS
```

where:

nnn - Maximum number of terminals permitted (octal value, right-justified). The default value is 18 (decimal).

117 - Symbolic location .TFMAX.

Time Interval For TSRI Data Issue To SCF

The following patch changes the time interval for writing Time Sharing accounting data to the Statistical Collection File (SCF). If the Time Sharing System Statistical Record (TSRI) accounting records are required, this patch must be applied. If there is no patch to this location, no TSRI data is written to the SCF.

```
                1                                                         7
1       8       6                                                         3
_______________________________________________________________________________
120     OCTAL   000000nnnnnn                                              .MTIMS
```

where:

nnnnnn - Number of seconds (octal) in the interval between data issues to the SCF. This value is right-justified in the variable field.

120 - Symbolic location .TSTAT.

Memory Allocation For DRL TASK Job

The following patch specifies the amount of memory that can be allocated to a batch job using a DRL TASK.

```
                1                                                    7
1      8        6                                                    3
______________________________________________________________________

127    OCTAL    nnnnnn000000                                    .MTIMS
```

where:

nnnnnn - Number (octal) of words of memory desired plus one additional word. The default value is (64*1024)+1.

127 - Symbolic location .TTCOR.

Urgency For DRL TASK Job

The following patch specifies the maximum urgency permitted for batch jobs using a DRL TASK:

```
                1                                                    7
1      8        6                                                    3
______________________________________________________________________

130    OCTAL    000000nn0000                                    .MTIMS
```

where:

nn - Maximum urgency (octal) for a job using a DRL TASK. The default value is 51 (octal).

130 - Symbolic location .TTURG.

DRL T.CMOV Users

The following patch identifies the type of Time Sharing System users that can examine portions of GCOS memory via a DRL T.CMOV.

```
                1                                                    7
1      8        6                                                    3
______________________________________________________________________

131    OCTAL    00000n000000                                    .MTIMS
```

where:

n - Numeric code identifying the category of users that can examine portions of GCOS memory. The following codes are valid:

1 - All users
2 - Master users (the default)
3 - Master users or users with LODS permission

131 - Symbolic location .TCMOV.

Permanent File User Identifier

The following patches specify the user identifier employed when allocating a permanent deferred queue file, primary or secondary file or swap file:

1	8	16	73
132	OCTAL	xxxxxxxxxxxx	.MTIMS
133	OCTAL	xxxxxxxxxxxx	.MTIMS

where:

x...x - User identifier (octal code for six BCD characters). A maximum of 12 characters can be specified (left-justified with trailing blanks). The default user identifier is SYS_CAT

132 - Symbolic location .TUSER.

133 - Symbolic location .TUSER+1.

Master User Identification

The following patches identify the userid for master users:

1	8	16	73
134	OCTAL	xxxxxxxxxxxx	.MTIMS
135	OCTAL	xxxxxxxxxxxx	.MTIMS

where:

x...x - Master user identifier (octal code for six BCD characters). The default identifier is MASTER. A maximum of 12 characters can be specified (left-justified with trailing blanks).

134 - Symbolic location .TMAST.

135 - Symbolic location .TMAST+1.

Status Message Time Interval

The following patch changes the frequency of Time Sharing System status messages issued to the system console:

```
                1                                          7
1      8        6                                          3
--------------------------------------------------------------
160    OCTAL    nnnnnnnnnnnn                          .MTIMS
```

where:

n...n - Time interval, in 1/64-millisecond increments (octal) between status message outputs. The default value is 15 minutes.

160 - Symbolic location .TATYI.

Class B Priority Dispatches

Use of the Class B priority feature (GCOS priority dispatching and control) by the Time Sharing System permits the system to control the number of dispatches permitted to jobs without Class B priority between dispatches to Class B priority jobs. The following patch allows the site to control the frequency of these dispatches as well as their duration. (Refer to "Dispatcher Options" in this section for a discussion of Class B priority and for a discussion of Dispatcher options that apply to multicopy Time Sharing operations.)

```
                1                                          7
1      8        6                                          3
--------------------------------------------------------------
176    OCTAL    0000xx0000yy                          .MTIMS
```

where:

xx - Number (octal) of 32-millisecond quantums per dispatch. The default value is 4.

yy - Number (octal) of non-Time Sharing System dispatches between dispatches to the Time Sharing System. The default value is 1.

176 - Symbolic location .TAGPP.

Maximum Memory Size

The following patch can be applied to modify the default maximum amount of memory allocated to the Time Sharing System. If this size must be changed while the system is operational, the TSS SIZE nnn console entry can be used.

1	8	16	73
200	OCTAL	000nnn000000	.MTIMS

where:

nnn - Maximum number (octal) of 1K blocks of memory to be allocated to the Time Sharing System (1K = 1024 words). The default value is 80K.

200 - Symbolic location .TAMMS.

Minimum Memory Size

When the Time Sharing System is lightly loaded and no longer needs all of the memory allocated to it for loading subsystems, memory can be released to GCOS until the default minimum of 40K is reached. The following patch can be applied to modify this default minimum:

1	8	16	73
201	OCTAL	nnnnnn000000	.MTIMS

where:

n...n - Minimum amount of memory allocated to Time Sharing for loading time sharing subsystems. This value (octal) is specified in pages (1K increments).

201 - Symbolic location .TASMS.

Memory Allocation Delays

Large jobs, whose memory requirements exceed the default limitation of 36K, frequently must wait four times as long as smaller jobs for memory allocation. The following patch modifies this default priority threshold value:

```
                1                                                       7
1       8       6                                                       3
---------------------------------------------------------------------------
202     OCTAL   000nnn000000                                       .MTIMS
```

where:

nnn - Memory-requirement level at which a job is assigned a low-priority classification by the Time Sharing System. This value (octal) is expressed in terms of 1K blocks.

202 - Symbolic location .TAMIS.

Request And Release Memory

The Time Sharing System automatically requests and releases memory in minimum increments of 7K and 5K, respectively. The following patches adjust these default values.

```
                1                                                       7
1       8       6                                                       3
---------------------------------------------------------------------------
203     OCTAL   0000xx000000                                       .MTIMS
204     OCTAL   0000yy000000                                       .MTIMS
```

where:

xx - Minimum number (octal) of 1K pages of memory to be requested. The default value is 7.

yy - Minimum number (octal) of pages of memory to be released. The default value is 5.

203 - Symbolic location .TAMII.

204 - Symbolic location .TASRI.

Memory Reserved for Urgent Jobs

When the Time Sharing System is heavily loaded, the delay in memory allocation for a job may exceed the time limit parameter. As a result, the job is given an "urgent" status. When urgent jobs exist, the memory allocation algorithm favors those jobs that have awaited allocation for the longest time by reserving a portion of available memory for exclusive use by the jobs. The following patch controls the amount of memory reserved for exclusive use by urgent jobs.

1	8	16	73
213	OCTAL	00000000000n	.MTIMS

where:

n - Reserve the amount of memory equal to the result of the algorithm x/y (where x = total space available and y = a value greater than 1). The default value for n is 2. If n is specified as 0, only memory required by the urgent job(s) that has waited the longest for allocation is reserved.

213 - Symbolic location A.URMD.

Deferred Processing Parameters

The following three patches control the maximum number of scheduled and active DRUNs, processor time limit, and preferred execution time for deferred processing.

Maximum scheduled and active DRUNs:

1	8	16	73
271	OCTAL	xxxxxxyyyyyy	.MTIMS

where:

xxxxxx - Maximum number (octal) of deferred sessions that can be scheduled at any one time. When this value is zero (0) or exceeds the limit imposed by the size of the deferred queue file, DRUN permits scheduling of as many sessions as possible.

yyyyyy - Maximum number (octal) of combined online and deferred-session users that can be active at any one time. When this maximum is reached, no additional deferred sessions are initiated. When this field is zero, deferred processing is disabled. The default value is 264 (octal).

271 - Symbolic location .TSDMX

Deferred processing time limit:

```
                 1                                         7
1   8            6                                         3
-------------------------------------------------------------
272  OCTAL   nnnnnnnnnnnn                              .MTIMS
```

where:

n...n - Octal value specifying maximum amount of processor time (in 1/64-millisecond increments) allowed for a single deferred session. The default value is zero, which indicates there is no time limit.

272 - Symbolic location .TSDPT.

Preferred time-of-day to run deferred job:

```
                 1                                         7
1   8            6                                         3
-------------------------------------------------------------
273  OCTAL   nnnnnnnnnnnn                              .MTIMS
```

where:

n...n - Octal value specifying the preferred time-of-day (in 1/64-millisecond increments past midnight) to initiate deferred sessions that do not request a specific start time. The default value is zero, which indicates that these sessions will be initiated as soon as possible.

273 - Symbolic location .TSDDT.

Line-Hold Interval Following Disconnect

When a terminal disconnects for any reason other than the BYE sequence, the Time Sharing System holds that line for a short time to give the terminal an opportunity to reconnect. The following patch modifies the hold interval:

```
                 1                                         7
1   8            6                                         3
-------------------------------------------------------------
326  OCTAL   000nnnnnnnnn                              .MTIMS
```

where:

n...n - Octal value specifying the amount of time (in 1/64-millisecond intervals) that a line is held before it is released following a terminal disconnect. The default value is equivalent to two minutes.

326 - Symbolic location .TIMER.

Memory Limit

A maximum memory limitation can be established for Time Sharing subsystems via the following site-option patch. If the object program exceeds this limitation, the subsystem aborts.

```
                  1                                       7
1      8          6                                       3
-----------------------------------------------------------
367    OCTAL      nnnnnn                                  .MTIMS
```

where:

nnnnnn - Memory limitation in pages (octal, left-justified). There is no default value.

367 - Symbolic location .TASSZ.

Processor Time Limit

A maximum processor time limit can be established for Time Sharing subsystems via the following site-option patch. If the object program exceeds this limitation, the subsystem aborts.

```
                  1                                       7
1      8          6                                       3
-----------------------------------------------------------
370    OCTAL      nnnn                                    .MTIMS
```

where:

nnnn - Processor time limit in 1/64-millisecond increments (octal). There is no default value.

370 - Symbolic location .TASTM.

Enable Measurement Record

The following site-option patch enables the Time Sharing System measurement record (SCF record type 19), used primarily in system tests and measurements.

```
                  1                                       7
1      8          6                                       3
-----------------------------------------------------------
377    OCTAL      00000000000n                            .MTIMS
```

where:

n - Enable the measurement record if this value is nonzero.

377 - Symbolic location .TSSAS.

Swap File Parameters

A series of patches can modify or establish parameters for Time Sharing System swap files. The following three patches can be used to:

- Modify the default number of swap files allocated at System Startup.
- Identify the growth factor to use when the swap files are created too small or too large. File sizes are dynamically adjusted by this factor.
- Define the size of the swap files.

Minimum swap file size:

```
                1                                        7
1     8         6                                        3
-----------------------------------------------------------
402   OCTAL     000000nnnnnn                        .MTIMS
```

where:

nnnnnn - Swap file size (octal) in llinks. To determine this size, multiply the growth factor from the patch to symbolic location .TSGRW by 48.

402 - Symbolic location .TSFS.

Swap file growth factor:

```
                1                                        7
1     8         6                                        3
-----------------------------------------------------------
403   OCTAL     nnnnnn000000                        .MTIMS
```

where:

nnnnnn - Minimum number (octal) of llinks to be requested or released to adjust the file size. This value can be determined via the formula (u*3s)/f (where u = maximum number of users configured, s = file size (in llinks), and f = number of swap files configured).

403 - Symbolic location .TSGRW.

Number of active swap files:

```
                1                                        7
1     8         6                                        3
-----------------------------------------------------------
404   OCTAL     00000000000n                        .MTIMS
```

where:

n - Number (octal) of swap files to be active. The maximum number of swap files is 16. Default allocation values are:

Number of users	Number of files
1-31	2
32-47	3
48-180	4

404 - Symbolic location .TSSF.

Permanent File Identification

The following patches can be applied for two purposes.

- o The site can define file names for the deferred queue file, primary and secondary program files, and swap files (if the swap files are permanent files). In this case, the specified file is created on a system-assigned mass storage device.
- o The site can selectively assign these files to specific mass storage devices. Because these files are heavily used by the Time Sharing System, system performance is often improved via the device-assignment technique.

```
                1                                        7
1      8        6                                        3
---------------------------------------------------------------
aaa    OCTAL    b00000nnnnnn                              .MTIMS
```

where:

aaa - Absolute octal address of the location in .MTIMS containing either the file name or the device name specified in the nnnnnn field. The addresses, corresponding symbolic identifications, and file identifications are described in Table 5-8.

b - Flag indicating whether nnnnnn is a file name or a logical device name. If bit 0 in this field =0, field nnnnnn contains the logical device name (in BCD) of the mass storage unit on which the file resides. If bit 0 in this field =1, field nnnnnn contains the file name (in BCD) of the permanent file defined at this location.

nnnnnn - Logical device name of the mass storage device on which this file resides, or the file name (1-3 characters) of this file. (nnnnnn is the octal code for three BCD characters.)

NOTE: If bit 17 is set in the #D allocation patch, and multicopy Time Sharing is configured, a common deferred queue file is used.

Table 5-8. Permanent File Address Descriptions

Octal Address (aaa)	Symbolic Identification	File Code	File Description
405	.TSFDV	#D	Deferred Queue
406	.TSFDV+1	#P	Primary Program
407	.TSFDV+2	#Q	Secondary Program
412	.TSFDV+5	#S	Swap
413	.TSFDV+6	#T	Swap
414	.TSFDV+7	#U	Swap
415	.TSFDV+8	#V	Swap
416	.TSFDV+9	#W	Swap
417	.TSFDV+10	#X	Swap
420	.TSFDV+11	#Y	Swap
421	.TSFDV+12	#Z	Swap
422	.TSFDV+13	+S	Swap
423	.TSFDV+14	+T	Swap
424	.TSFDV+15	+U	Swap
425	.TSFDV+16	+V	Swap
426	.TSFDV+17	+W	Swap
427	.TSFDV+18	+X	Swap
430	.TSFDV+19	+Y	Swap
431	.TSFDV+20	+Z	Swap

Program Logic Error Processing

The following patch can be applied to control the manner in which the Time Sharing System performs program logic error processing. If bits 0-17 contain a nonzero value, Time Sharing error recovery is attempted; if bits 0-17 are zero, error recovery is not attempted. If recovery is not attempted or is unsuccessful, bits 18-35 are inspected to determine how to terminate the Time Sharing System. If bits 18-35 are zero, the GCOS system aborts; if bits 18-35 contain a nonzero value, only the Time Sharing System aborts.

1	8	16	73
1124	OCTAL	00000x00000y	.MTIMS

where:

x - If zero, error recovery is not attempted. If nonzero, error recovery is attempted on the basis of error counts, timer data, and macro expansions.

y = If zero, the GCOS system aborts when recovery is not attempted or fails. If nonzero, the Time Sharing System aborts with a reason code of 74.

1124 - Symbolic location .TPLES

Time Sharing Processors

The following patch specifies the number of processors that can execute subdispatch processes.

```
                1                                         7
1      8        6                                         3
____________________________________________________________________
1744   OCTAL    00000n000000                              .MTIMS
```

where:

n - Number of processors on which Time Sharing subsystems can execute. The default value is 6.

1744 - Symbolic location .QPROC.

Dispatching Time Slice

The following patch specifies the dispatching quantum (i.e., time slice) for one Time Sharing subsystem subdispatch.

```
                1                                         7
1      8        6                                         3
____________________________________________________________________
1746   OCTAL    000000nnnnnn                              .MTIMS
```

where:

nnnnnn - Dispatch quantum expressed in 1/64-millisecond increments. The default value is 20 milliseconds (002400 octal).

1746 - Symbolic location .QQTM

Additional PAT Space For Time Sharing

If the number of Time Sharing System users exceeds 90, the .MPOPA module must be patched to provide additional Peripheral Assignment Table (PAT) space for the Time Sharing System. One additional page is required for every 18 users over 90 (e.g., 105 users requires one additional page). Apply one patch for each copy of Time Sharing in a multicopy Time Sharing environment. The format for this patch is:

```
                1                                         7
1      8        6                                         3
____________________________________________________________________
aa     OCTAL    562xxx756yyy                              .MPOPA
```

where:

aa - Address to apply this patch. The following addresses are associated with the various copies of Time Sharing:

25 - The first, or only, copy of Time Sharing (TS1)
30 - The second copy of Time Sharing (TS2)
33 - The third copy of Time Sharing (TS3)
36 - The fourth copy of Time Sharing (TS4)

562 - Module number of .MTIMS, containing related Time Sharing System patches.

xxx - Number (octal) of 1024-word pages allocated for the Time Sharing Executive program. The default is 69 decimal (105 octal) pages.

75 - Urgency level (octal) of the Time Sharing Executive program.

6 - The Time Sharing System is a privileged slave program and is in accommodation mode.

yyy - Peripheral Assignment Table (PAT) segment size (in 512-word pages, expressed in octal). The default is 6.

.MDNET OPTIONS

Use of octal patches as the means for selecting site options in .MDNET has been eliminated. See "$ MODOPT Statement" in the $FILES Section.

*

MULTICOPY TIME SHARING CLASS B PRIORITY

A site-option patch must be applied to the .MDISP module to activate Class B priority in a multicopy Time Sharing environment. Refer to "Apply Class A and Class B Priorities" for a discussion of this patch.

TIME SHARING LOGON MESSAGE

Three site-option patches can be applied to the .TSNEW subsystem of the Time Sharing System to provide system resource information in the logon message.

Display System Identifier

The following patch includes the system identifier in the logon message. (The system identifier is specified on a $ SYID statement in the $CONFIG section.)

1	8	16	73
152	OCTAL	n	.TSNEW

where:

n - Any nonzero value places the system identifier at the beginning of the logon message. The default value is 0.

152 - Symbolic location NEWSID.

Identify Remaining File Space

The following site-option patch issues a logon message identifying the amount of file space remaining for the user.

1	8	16	73
153	OCTAL	n	.TSNEW

where:

n - Any positive, nonzero value activates the message; any negative, nonzero value suppresses the message. The default value is 0, which results in issuance of the message:

    nn LLINKS AVAILABLE

when more than 85% of the user's maximum file space is occupied (where nn = number of llinks available).

153 - Symbolic location NEWBLK

Activate Logon LOAD Message

The following site-option patch activates the logon LOAD message, which displays the current number of users, the amount of memory allocated to the Time Sharing System, the percentage of Time Sharing System memory in use, the number of users waiting for memory allocation, and the total amount of memory required by all waiting users.

```
                1                                        7
1      8        6                                        3
__________________________________________________________

154    OCTAL    n                                   .TSNEW
```

where:

n - Any nonzero value activates the LOAD message. The default value is 0.

154 - Symbolic location NEWLOD.

Logon Security

This site-option patch can invoke an optional security feature, whereby, password changes are required the first time a user logs on following implementation of this option and, thereafter, every n days from the last change date.

```
                1                                        7
1      8        6                                        3
__________________________________________________________

303    OCTAL    yyyddd000nnn                        .TSNEW
```

where:

yyy - The three-digit octal equivalent of the year in which the password change becomes a requirement.

ddd - The three-digit octal equivalent of the day of the year in which the password becomes a requirement.

nnn - The three-digit octal equivalent of the number of days between required password changes (must be less than 365 decimal).

TIME SHARING ACCOUNTING

A site can specify unique accounting parameters by applying site-option patches to locations 176-210 (octal) of the .TSLOG subsystem. Note that the values included on patches to locations 176-206 can be specified as floating-point numbers.

Elapsed Time Charge Rate

The following patch changes the hourly charge rate for elapsed time:

```
                1                                           7
1     8         6                                           3
-------------------------------------------------------------------
176   OCTAL     000000nnnnnn                                .TSLOG
```

where:

nnnnnn - Elapsed time hourly rate. This value represents cents per hour and is expressed in octal. The default value is 600 (octal).

176 - Symbolic location ELPPR.

Memory-Usage Rate

The following patch changes the hourly rate per 512-word block of memory used:

```
                1                                           7
1     8         6                                           3
-------------------------------------------------------------------
177   OCTAL     000000nnnnnn                                .TSLOG
```

where:

nnnnnn - Hourly rate for each 512-word block of memory used. This value represents cents per block per hour and is expressed in octal. The default value is 1000.

177 - Symbolic location PPH

Mass Storage File I/O Rate

The following patch modifies the charge rate for mass storage file I/O:

```
                1                                        7
1      8        6                                        3
-----------------------------------------------------------

200    OCTAL    000000nnnnnn                        .TSLOG
```

where:

nnnnnn - Hourly rate for mass storage file I/O. This value represents cents and is expressed in octal. The default value is 2400.

200 - Symbolic location DISCPR.

Keyboard I/O Rate

The following patch modifies the charge rate per page for keyboard I/O (typewriter or keyboard-display terminals).

```
                1                                        7
1      8        6                                        3
-----------------------------------------------------------

201    OCTAL    0000000000nn                        .TSLOG
```

where:

nn - Cents per page for mass storage file I/O. This value is expressed in octal. The default value is 02.

201 - Symbolic location PPP

Service Rate Factor

The following patch changes the multiplication factor that varies the rate for services:

```
                1                                        7
1      8        6                                        3
-----------------------------------------------------------

202    OCTAL    0000000000nn                        .TSLOG
```

where:

nn - Multiplication factor (octal). The default value is 1.

202 - Symbolic location K5

Average Disk I/O Time Value

The following patch changes the average disk I/O time value used for calculating disk I/O charges.

```
                    1                                              7
1       8           6                                              3
--------------------------------------------------------------------------
203     OCTAL       00000000000n                                   .TSLOG
```

where:

n - Average disk I/O time (octal) in 1/10-second increments. The default value is 3.

203 - Symbolic location AVDIO

Keyboard I/O Characters Per Page

The following patch changes the average number of characters per page of keyboard I/O:

```
                    1                                              7
1       8           6                                              3
--------------------------------------------------------------------------
204     OCTAL       000000nnnnnn                                   .TSLOG
```

where:

nnnnnn - Average number (octal) of characters per page of keyboard I/O. The default value is 3720.

204 - Symbolic location AVNOCR.

Deferred Session Rate

In some cases, deferred sessions may be charged at a reduced rate when the sessions begin within a specified time frame. The following patch can be applied to specify the reduced rate, which is expressed as a percentage of the normal rate.

```
                     1                                      7
1       8            6                                      3
-----------------------------------------------------------------
205     OCTAL        000000000nnn                       .TSLOG
```

where:

nnn - Percentage (octal) of the normal rate that defines the reduced rate. For example, 120 defines 80% of the normal rate. The default value is 144, which is 100% of the normal rate.

205 - Symbolic location RR

Deferred Session Elapsed Time Rate

The following patch specifies the hourly rate for elapsed time of deferred sessions:

```
                     1                                      7
1       8            6                                      3
-----------------------------------------------------------------
206     OCTAL        000000nnnnnn                       .TSLOG
```

where:

nnnnnn - Hourly rate for elapsed time of a deferred session. This value represents cents per hour and is expressed in octal. The default value is 600.

206 - Symbolic location DELPR

Starting And Ending Times For Reduced Rates

The following patches specify the times of day at which the reduced rates for deferred sessions begin and end.

```
                1                                              7
1       8       6                                              3
-----------------------------------------------------------------

207     OCTAL   nnnnnnnnnnnn                              .TSLOG

210     OCTAL   mmmmmmmmmmmm                              .TSLOG
```

where:

n...n - Starting time (in 1/64-millisecond increments, expressed in octal) of the reduced-rate period. The default value is -1, which indicates there is no reduced rate.

m...m - Ending time (in 1/64-millisecond increments, expressed in octal) of the reduced-rate period. The default value is 0, which indicates there is no reduced rate.

207 - Symbolic location RRLL

210 - Symbolic location RRUP

Additional Control Over Save Dumps (.MBRT8)

The site's use of $ INFO SAVDMP and/or the use of the SAVDMP= option on the user's $ DMPOPT JCL statement may cause save dumps (information dumped to system-controlled space when a user dump is taken) to be placed on the SCF. Though not recommended (for security reasons), it is possible for the site to cause save dumps to go to the user's P* file instead of the SCF.

The format for this site-option patch is:

```
                1                                              7
1       8       6                                              3
-----------------------------------------------------------------

6       OCTAL   n                                         .MBRT8
```

where:

n = 0 - Direct Save Dumps to the SCF when specified in JCL (by $ DMPOPT SAVDMP= option) or by $ INFO SAVDMP option.

2 - Direct Save Dumps to P* when specified in JCL (by $ DMPOPT SAVDMP= option) or by $ INFO SAVDMP option.

Time Sharing Initialization File

The Time Sharing initialization file is described in detail in the GCOS 8 OS Time Sharing Administration Guide. Following is a general description to state its purpose.

A site can establish parameter data on an initialization file during Time Sharing initialization in lieu of applying site-option patches to the Time Sharing System. Use of this technique is optional. However, if the initialization file is allocated, the following requirements apply:

- o One initialization file must be created for each copy of Time Sharing. The file name must be that of the corresponding SNUMB (i.e., TS1-TS4) so that the file can be attached.

- o The file is cataloged as a quick-access file with no password and is subordinate to the user identification specified in .TUSER cells within the TSSA communication region.

- o The file may be created in either ASCII or BCD format. Several different subsystems can be used to create the file (e.g., EDITOR, $CONVER, or TSCONV).

- o The file is composed of logical records. Each logical record comprises one parameter.

Parameters (i.e., logical records) are defined in either the $INFO section or the $PATCH section of the Time Sharing initialization file. These sections perform essentially the same functions as their counterparts (of the same names) in the GCOS Startup deck. The following describes use of the two sections.

$INFO SECTION

The purpose of the $INFO section of the Time Sharing initialization file is to eliminate the need for a site to use site-option patches to invoke various time sharing options. The $INFO section makes it much easier for site personnel to activate the options. Because of the impact of file-activated options, it is recommended that this feature be used only by site personnel who are familiar with the format and content of Time Sharing site-option patches.

$PATCH SECTION

The $PATCH section of the Time Sharing initialization file can include Time Sharing System Executive or subsystem patches for application during Time Sharing initialization. With few exceptions, the $PATCH section of the initialization file is the same as the $PATCH section of the GCOS Startup job stream. However, the initialization file section allows a site to modify the Time Sharing System without rebooting GCOS.

$LOAD SECTION

The $LOAD section of the Time Sharing initialization file provides a mechanism for loading site modified TSS subsystems and editing the subsystems into the program descriptors, command list, and primitives. The user subsystems must be built in H* format and be loaded for access by the executive in one of two places:

1. Subordinate to the Master userid or

2. Subordinate to the CMDLIB userid

$LOAD SECTION

The $LOAD section of the Startup job stream can be used to load modules into main memory for execution without affecting total system software tape contents. Any module loaded via this section overrides (until the next bootload) the version of the module existing on the total system software tape.

The $LOAD section can prove beneficial under several circumstances. For example, it provides a way to:

- o Load modules not contained on the total system software tape into memory.

- o Test new or corrected modules - especially those with extensive modifications - before they are edited onto the total system software tape.

- o Load object decks of any Slave Service Area (SSA) modules that are site-prepared. Among these modules are .MSCAN, .MGNAT, and .MSCN1. (These types of modules can also be loaded via an insert edit.)

Use of the $LOAD section is optional. If there are no modules requiring loading in this manner, the $LOAD section can be excluded from the Startup job stream. If the $LOAD section is not included, insert the ***EOF statement that marks the end of the Startup job stream after the ***EOF statement that marks the end of the $PATCH section.

Each module included in the $LOAD section must be an object deck in binary format and must be delimited by $ OBJECT and $ DKEND control statements. Any relocation references within the code are ignored. Each module must specify a primary SYMDEF. If a primary SYMDEF is not included, the module is ignored and not written to the $LOAD file. In addition, if more than one copy of a module is included in the $LOAD section, only the first copy encountered is written to the $LOAD file; all other copies are ignored.

NOTE: Some modules cannot be loaded via the $LOAD section. Among these are non-system software (e.g., applications programs) and GCOS system modules that are assembled as programs (e.g., the System Input program).

As the $LOAD section is entered via magnetic tape during a cold boot, the Startup program writes each module included in the section to the load file on mass storage. This file and the mass storage device on which it resides must be defined on the $ LOADFIL statement in the $CONFIG section.

Any change in $LOAD section content (i.e., modifying existing module content, adding to an existing module, deleting an existing module, or adding an entire new module) requires that the entire $LOAD section be entered via the card reader. The operator must respond YES to the *REPLACE? question or REPL to the *BOOT SOURCE: AUTO OR REPL? question during the bootload sequence. Existing load file contents are replaced by the current $LOAD section.

If patched modules are loaded via the $LOAD section, all OCTAL correction statements must immediately precede the $ DKEND statement of the module. Patches contained in the $PATCH section of the Startup job stream are not applied to modules included in the $LOAD section.

Modules modified via alter corrections can be loaded via the $LOAD section by (1) generating a compressed deck (COMDK) of the module; (2) using the COMDK and all applicable alters to compile the module; and (3) inserting the resulting binary deck (delimited by $ OBJECT and $ DKEND statements) into the $LOAD section.

Because $LOAD section modules are resident in main memory during system operation until the next bootload, it is recommended that discretion be used when specifying a module to be included in this section. A block of memory equal to the size of the module is dedicated to module storage.

$ GHCM And $ SHCM Statements

Most GCOS 8 system modules are shared modules and are loaded into the Segmented Hard Core Monitor (SHCM) portion of memory. Each shared module is assigned its own segment in the HCM.

Shared modules include those which are identified as Segmented HCM (SHCM) and Shared Slave Service Area (SSSA) modules. When loading SHCM and SSSA modules via the $LOAD section, the modules must be preceded by a $ SHCM statement.

The format for the $ SHCM statement is:

```
1      8
$      SHCM
```

Some system modules are not shared modules and must co-exist with other modules. These modules must be loaded into that portion of memory known as the Grouped Hard Core Monitor (GHCM).

Nonshared modules include those which are categorized as Grouped HCM (GHCM), Full SSA (FSSA), and Half SSA (HSSA). When loading GHCM, FSSA, and HSSA modules via the $LOAD section, the modules must be preceded by a $ GHCM statement.

The format for the $ GHCM statement is:

```
1      8
$      GHCM
```

If Startup encounters a module erroneously situated following a $ SHCM or $ GHCM statement, the message:

*OBJECT DECK CARD nnnnmmmm IS NOT hhhh TYPE, DECK SKIPPED

(where nnnn = statement number, mmmm = module name, and hhhh = SHCM or GHCM) is issued.

NOTE: Modules loaded into GHCM via the $LOAD section are loaded and initialized in the order in which they appear in the $LOAD section, and before any other modules that would normally reside in GHCM. DNET and ROUT both reside in GHCM, and it is a system requirement that DNET be initialized before ROUT. Therefore, ROUT must not be placed in the $LOAD section unless DNET is also placed there, and DNET must come first. If only ROUT is to be replaced, use the $ FILDEF and $ SYSTEM statements, not the $LOAD section.

APPENDIX A

STARTUP JOB STREAM EXAMPLE

The Startup job stream in this appendix is for example purposes only and is not to be construed as a working job stream. Comments regarding statement formats and specific data on individual statements are minimized. Refer to Section 5 for a complete description of the statement formats and for annotated examples.

```
               1
1      8       6
_________________________________________________________________________

$CONFIG  "SYS-C1 SR3000 OSI TEST SYSTEM"
$      SYID    SYC1
$      BASE    IOM-4,DN-8
$      ANSWER  RETCONFIG/NO
$      ANSWER  INITIALIZE/NO,EDIT/NO,SSCLEAR/YES
$      ANSWER  RESTART/NO,SYSOTRCVY/NO,SCF/NO,TIME/NO
$      ANSWER  DUMPON/TAPE,DUMP/ALL
$      ANSWER  PROCERROR/YES
$      INFO    SCHSAV
$      INFO    LOGON
$      INFO    TS8
$      INFO    SAVDMP/ANY
$      INFO    FNP/2
$      INFO    RLSPNT
$      INFO    ACALL/1
$      INFO    ACALL/2
$      INFO    ASCII
$      INFO    ACALL/TS1
$      INFO    CATFNC/5000
$      INFO    CATDUP,SMCDUP/ST2
$      INFO    CCACHE/30
$      INFO    CHKSUM
$      INFO    CLENPT
$      INFO    DEN16/DEN16
$      INFO    DPSE
$      INFO    EXTTDS/12,230
$      INFO    EXTTSS/61,512,1
$      INFO    EXTTSS/61,512,2
$      INFO    EXTTSS/61,512,3
$      INFO    EXTTSS/61,512,4
$      INFO    PSUM/487
$      INFO    FMSCT/100
$      INFO    FMSTAT
$      INFO    GEOTRC/777777
```

```
                1
1      8        6
------------------------------------------------------------------------
$      INFO     HEX/ON
$      INFO     INTERRUPT/ALL
$      INFO     LIMITS
$      INFO     MCPTSS/4
$      INFO     MEMORY/240
$      INFO     MODULE/.MSDCB
$      INFO     MIXED
$      INFO     NDFMAP
$      INFO     NIAST/RMVBL
$      INFO     PSHWSN/15
$      INFO     PT1SIZ/2048
$      INFO     RESVID/PPSONL,P1,P2,P3/
$      INFO     RESVID/PPSONL,P4,P5,P6,P7,P8,P9/
$      INFO     RESVID/FORMS,F1,F2,F3,F4,F5,F6,F7,F8,F9/
$      INFO     SLINKS/4000
$      INFO     SSAOVL/16
$      INFO     SYBRT/200
$      INFO     SYJOT/5000
$      INFO     SYSOUT/50000
$      INFO     SYSTEMMAP/NO
$      INFO     TAPDEN/DE
$      INFO     TIMEZ/MST
$      TRACE    0,0,0,0
$      MCT-0    4096,PORT-3,IOM-0,PORT-2,IMU-1,PORT-4,PRO-0,PORT-5,PRO-1
$      IOM-0    PUB-30,CS66,TY1,TY2,TY3,TY4
$      IMU-1    CH-30,CS66,TY5
$      IOM-0    PUB-31,RMC66,TR1
$      IOM-0    PUB-12,MS0450,UNITS-08,NONSEQ,
$      ETC      UNIT-11,ST1,S-H304,AU-1,
$      ETC      UNIT-12,ST2,S-C413,AU-1,
$      ETC      UNIT-13,ST3,S-C443,AU-1,
$      ETC      UNIT-14,ST4,S-C730,AU-1,
$      ETC      UNIT-15,ST5,S-C885,AU-1,NOFMS,
$      ETC      UNIT-16,RP1,S-C544,RMVBL,
$      ETC      UNIT-17,RP2,RMVBL,
$      ETC      UNIT-18,RP3,RMVBL
$      IOM-0    PUB-14,MS0501,UNITS-04,NONSEQ,
$      ETC      UNIT-57,NM1,S-S543,
$      ETC      UNIT-58,NM2,S-S544,
$      ETC      UNIT-59,NM3,NOFMS,S-S545,
$      ETC      UNIT-60,NM4,NOFMS,S-S546
$      IMU-1    CH-44,MSC1B,UNITS-10,NONSEQ,
$      ETC      UNIT-1.0,DP1,AU-1,
$      ETC      UNIT-1.1,DP2,NOFMS,AU-1,
$      ETC      UNIT-2.0,DP3,
$      ETC      UNIT-2.1,DP4,
$      ETC      UNIT-16.0,ET1,MSD1B,
$      ETC      UNIT-16.1,ET2,MSD1B,NOFMS,
$      ETC      UNIT-16.2,ET3,MSD1B,
$      ETC      UNIT-17.0,ET4,MSD1B,
```

```
                1
1      8        6
$      ETC      UNIT-17.1,ET5,MSD1B,RMVBL,
$      ETC      UNIT-17.2,ET6,MSD1B,RMVBL
$      IOM-0    PUB-18,TAPE*610*9,CD,S2000,UNITS-08,NONSEQ,
$      ETC      UNIT-01,DE9,1T1,CLASS-3,
$      ETC      UNIT-02,DE9,1T2,CLASS-3,
$      ETC      UNIT-03,DE9,1T3,CLASS-3,
$      ETC      UNIT-04,CD9,1T4,CLASS-2,
$      ETC      UNIT-05,BCD9,1T5,
$      ETC      UNIT-06,BC7,1T6,
$      ETC      UNIT-07,BC7,1T7,
$      ETC      UNIT-08,DE9,1T8,CLASS-3
$      IOM-0    PUB-24,PR1600,UNITS-1,UNIT-1,TRAIN-0764,136,PR3,DEDICATED
$      IMU-1    PUB-24,CR1050,UNITS-1,UNIT-1,CR2,510PT
$      IMU-1    PUB-25,PC0120,UNITS-1,UNIT-1,PU2
$      IOM-0    PUB-28,6632-0,LINES-200       DN8#9911 DIOC00  NODENAME=SYC1
$      IOM-0    PUB-29,6632-1,LINES-200       DN8#0365 DIOA80  NODENAME=SC1S
$      IMU-1    PUB-29,UNCP,FE2               DN8#0365 DIOC00  NODENAME=SC1X
$      IMU-1    PUB-16,PR1600,UNITS-1,UNIT-1,TRAIN-1130,136,PR1
$      IMU-1    CH-32,PR1201,UNITS-1,UNIT-1,TRAIN-1130,136,PR2,DEDICATED
$      IMU-1    PUB-52,TAPE*FIPS*9,DE,UNITS-6,NONSEQ,
$      ETC      UNIT-00,DE9,2T0,CLASS-3,
$      ETC      UNIT-01,DE9,2T1,CLASS-3,
$      ETC      UNIT-02,CD9,2T2,CLASS-2,
$      ETC      UNIT-03,DE9,2T3,CLASS-2,
$      ETC      UNIT-04,CD9,2T4,CLASS-2,
$      ETC      UNIT-05,DE9,2T5,CLASS-2
$      XBAR     IOM-0,PUB-14,PUB-15
$      XBAR     IOM-0,PUB-12,IMU-1,PUB-40,PUB-41,IOM-0,PUB-13
$      XBAR     IOM-0,PUB-18,IMU-1,PUB-20,PUB-21,IOM-0,PUB-19
$      XBAR     IMU-1,CH-44,CH-45,CH-46,CH-47
$      GROUP    IMU-1,CH-44-47
$      XBAR     IMU-1,CH-52,CH-53,CH-54,CH-55
$      GROUP    IMU-1,CH-52-55
$      MPC-0    SIZE-4,PSI-0,IOM-0,PUB-12,PUB-13,
$      ETC      PSI-2,IMU-1,PUB-40,PUB-41
$      MPC-1    SIZE-16,PSI-0,IOM-0,PUB-14,PUB-15
$      MPC-2    SIZE-4,PSI-0,IOM-0,PUB-18,PUB-19,             MTP610 3252
$      ETC      PSI-3,IMU-1,PUB-20,PUB-21
$      URP-4    IMU-1,PORTS-2,PUB-16
$      URP-5    IOM-0,PORTS-2,PUB-24
$      URP-7    IMU-1,PORTS-2,CH-32
$      URP-8    IMU-1,PORTS-2,CH-24
$      TRAINS   402/764/GCOS,LONG,402/1130/ASCII4
$      FILCREA  FASTBT,ST2,FASTBOOT
$      AUTOLD   ST1,800 LLINKS
$      LOADFIL  ST3,SR2LOAD,500
$      DECKFIL  DECKFILE,1900/0
$      GCOSFIL  ST1,ST2,ST3,ST4,NM1,ET1,DP1
***EOF
```

```
               1
1       8      6
$INITIALIZE      "SR3000 INITIALIZE"
$       INIT    ST1,CAT,ST2,CAT,ST3,CAT,ST4,CAT,ST5
$       INIT    NM1,CAT,NM2,CAT,NM3,NM4
$       INIT    ET1,CAT,ET2,ET3,CAT,ET4,ET5,ET6
$       INIT    DP1,CAT,DP2,DP3,DP4
$       INIT    RP1
$       OBJECT  HTND  DU MPCB      CON                          G      861028MPCB0000
$       DKEND                                                   15.330102886MPCB0107
$       READIN  2T0,,,,DEN16
***EOF
$EDIT      "SR3000 S8.19 / G8.20 / N8.17"
$       FILDEF  ST2,SYSLIB,1500/0,RDM,1T2,DEN16,LABEL/SOFTW-SYSLIB
$       FILDEF  ST3,VS-PRIMARY,8000/0,SYS,*,LABEL/VS-PRIMARY
$       FILDEF  ST4,TSS,1200/0,SYS,*,LABEL/TSS-SUB-SYS
$       FILDEF  NM1,SOFTW-P1,2000/0,SYS,*,LABEL/SOFTW-PART1
$       FILDEF  ET1,SOFTW-P2,2800/0,SYS,*,LABEL/SOFTW-PART2
$       FILDEF  DP1,SOFTW-P3,2100/0,SYS,*,LABEL/SOFTW-PART3
$       FILDEF  NM1,SWI-30,4700/0,SWI,1T3,DEN16,LABEL/SWI-3000
$       DTFILE  ST1,SYSPN_3000,15/0
$ *M*     SYSPNC.SPWN    ***SYSTEM SPAWN FILE FOR - SYSPN (COLD BOOT)
$ *C*     MODULE:  SYSPNC.SPWN
$       FILSYS  DUMP
$       PRIVITY
USERID SYS_SOFTWARE
RESTORE SYS_SOFTWARE,RESET/DEVICE/,RESET/RFILES/,RESET/DENIED/
CM SYS_SOFTWARE,MODIFY/OPNSUTIL,SYS_CAT/
$       SYSOUT  ER
$       TAPE9   PR,P1D,,99999,,SYS_SAVES
$       MSG2    1,PLEASE MOUNT THE LATEST SYS_SOFTWARE SAVE TAPE.
$       BREAK
$       UTILITY
$       FUTIL   IN,OT,REW/IN,OT/,COPY/1F/
$       PRMFL   IN,Q,L,SYS_SOFTWARE/3000USERDATA/SYSPN.PRE
$       PRMFL   OT,W,L,SYS_SOFTWARE/3000USERDATA/SYSPN.DIRECT
$       RUN     RUFILE=SYSTEM,RUNAME=SYSPN,OPTIONS=DUMP,SSDUMP,NJREST
$       PRIVITY
$       LIMITS  10
$       RESOURC RSPACE=100K
$       PRMFL   I*,R,S,SYS_SOFTWARE/3000/DATA/SYSPN.COLD
$       ENDJOB
$       ENDTF
$       DTFILE  ST2,SYS_MEMREL,2/0
$       IDENT   P23GFA,MCDONALD,STA-AX
$       PROGRAM MEMREL,DUMP,NJREST
$       LIMITS  10,100K
$       PRIVITY
$       ENDJOB
$       ENDTF
```

```
                1
1      8        6
------------------------------------------------------------------------
$      FILDEF   DP1,BRT7-3000,20/0,SYS,1T3,DEN16,LABEL/BRT7-3000
$      FILDEF   ST3,ASC-SOFTW,2700/0,SYS,1T1,DEN16                  "56702"
$      FILDEF   ST1,COBOL-74,1100/0,SYS,1T8,DEN62      42701
$      FILDEF   ST4,PDQ,10/0,SYS,*
$      FILDEF   NM1,IDS2,3500/0,SYS,*
$      FILDEF   ST1,8DS2.2-SOFTW,300/0,SYS,*
$      FILDEF   ST4,8FT2.1-SOFTW,700/0,SYS,*
$      FILDEF   ST4,8F77-SOFTW,1000/0,SYS,*
$      FILDEF   ST2,DMS-SOFTW,1800/0,SYS,*
$      FILDEF   ST2,MENU,50/0,SYS,*
$      FILDEF   ST4,PCF-SOFTW,200/0,SYS,*
$      FILDEF   DP1,8PL3.3,1500/0,SYS,*
$      FILDEF   ST3,PDQ-SYS,5300/0,SYS,*
$      FILDEF   ST1,RPG2,200/0,SYS,*
$      FILDEF   ST1,PSPSW-SYSLIB,1300/0,RDM,*
$      FILDEF   ST2,IFAD,1000/0,SYS,1T4,DEN16,LABEL/30-IFAD-QS06
$      FILDEF   ST2,ELAN,1200/0,SYS,1T2,DEN16,LABEL/W2ELN0830040
$      FILDEF   ST3,8FM3.0,400/0,SYS,1T3,LABEL/8FM3.0        TAPE# 37390
$      FILDEF   NM1,ACCNT0,1200
$      FILDEF   ET1,ACCNT1,1200
$      FILDEF   ST2,LUMP0,150
$      FILDEF   ST1,DUMP0,1000/0
$      FILDEF   ET1,DUMP1,26500/0
$      FILDEF   ET1,PRINTIMAGE,85
$      FILDEF   ET1,SYSOUT0,19000
$      FILDEF   DP1,SYSOUT1,19000
$      FILDEF   ET1,SYSOUT2,19000
$      FILDEF   DP1,SYSOUT3,19000
$      FILDEF   ST2,TANDFILE,7000
$      SSFILE   ST1,.EXPRS/100,.HOLD/09,.NORM/100,
$      ETC      .IMCV/100,.NNNNT/100,.NNNNZ/100
***EOF
$FILES  "SR3000 FILES"
$      SYSTEM   SWI-30
$      PFILES   SWI-30
$      DFILES   SYSPN_3000
$      DFILES   SYS_MEMREL
$      LIBRARY  RDM,SYSLIB,RDM,PSPSW-SYSLIB
$      SYSTEM   ELAN
$      PFILES   ELAN
$      SYSTEM   VS-PRIMARY
$      PFILES   VS-PRIMARY
$      SYSTEM   TSS
$      PFILES   TSS
$      SYSTEM   SOFTW-P1
$      PFILES   SOFTW-P1
$      SYSTEM   SOFTW-P2
$      PFILES   SOFTW-P2
$      SYSTEM   SOFTW-P3
$      PFILES   SOFTW-P3
```

```
                1
1      8        6
________________________________________________________________
$      SYSTEM   BRT7-3000
$      SYSTEM   IFAD
$      PFILES   IFAD
$      SYSTEM   8FM3.0
$      SYSTEM   PCF-SOFTW
$      SYSTEM   PDQ-SYS
$      SYSTEM   COBOL-74
$      SYSTEM   8PL3.3
$      SYSTEM   PDQ
$      SYSTEM   IDS2
$      SYSTEM   8DS2.2-SOFTW
$      SYSTEM   8FT2.1-SOFTW
$      SYSTEM   8F77-SOFTW
$      SYSTEM   ASC-SOFTW
$      SYSTEM   DMS-SOFTW
$      SYSTEM   MENU
$      SYSTEM   RPG2
$      PFILES   AUTOLOAD
$      PFILES   DECKFILE
$      PFILES   TANDFILE
$      SAVE     LUMP0
$      PFILES   DUMP0
$      PFILES   DUMP1
$      PFILES   PRINTIMAGE
$      ACCOUNT  RDM,ACCNT0,ACCNT1,IDS
$      SCFDSP   288,ALL
$      SCFBUF   288,ALL
$      SYSOUT   SYSOUT0,SYSOUT1,SYSOUT2,SYSOUT3
$      SSLOAD   MAX/400,MIN/5,.TASK/10,.EXPRS/50,.NORM/40,.IMCV/5,
$      ETC      .NNNNT/20,.NNNNZ/20
***EOF
$PATCH           "PATCH SECTION"
.
.               OCTAL patch corrections
.
***EOF
$LOAD   "SR3000 LOAD SECTION"
$      SHCM
.
.               Modules that are loaded into Shared HCM
.
$      GHCM
.
.               Modules that are loaded into Grouped HCM
***EOF
***EOF
```

APPENDIX B

FUNCTIONAL CHECKLISTS

The purpose of this appendix is to provide a series of quick-reference, functional checklists that should be reviewed when installing a new system or whenever Startup job stream modification is required. For example, the checklist of statement and operator procedures should be reviewed when re-creating the AUTOLOAD file.

So that the checklist or "cookbook" concept is maintained, detailed statement formats and operator procedures are not included in this appendix. Refer to Section 5 for statement formats and for descriptions of individual statements and to Section 2 for detailed operator procedures.

STARTUP JOB STREAM - NEW INSTALLATION

The following discussion concentrates on statements that either are required in or should be considered for inclusion in the Startup job stream when installing a new system.

Startup Statements	Discussion
$CONFIG Section	Reflects the site's gross software and hardware configuration.
$ SYID	Can be used to specify the system name that is to print in the dump banner. This is helpful if the site has multiple systems. In addition, identification of the Software Release is useful when the site is communicating with offsite Honeywell Bull personnel.
$ TRACE	All system traces normally are active when installing a new system. This assists in isolating initialization problems. However, this statement can be used to selectively deactivate traces that are not to be used.

Startup Statements	Discussion
$ ANSWER	Messages that otherwise are issued to the system console and that require an operator response can be pre-answered. It is recommended that the following responses be included on the $ ANSWER statement at this time. Additional responses (e.g., dump-related responses) can be included at site discretion. TIME/NO EDIT/NO INITIALIZE/NO SCF/YES SSCLEAR/NO RESTART/YES It is recommended that a system map be printed during system installation to assist in problem analysis. Thereafter, the SYSTEMMAP/NO response can be included on the $ ANSWER statement to forego printing of the system map.
$ INFO	It is recommended that the following parameters be specified. Additional parameters can be included at site discretion. SMCDUP - Provides a duplicate System Master Catalog (SMC). DENxx/DENyy - Defines the default high and low densities of an MPC-driven magnetic tape subsystem. It is recommended that the NDFMAP parameter not be included during initial installation procedures (i.e., that a deckfile map be printed to assist in problem analysis). Thereafter, the NDFMAP parameter can be included on the $ INFO statement to forego printing of the deckfile map. The FNP/n parameter or the $ BASE statement is required if any Network Processors (DNET/ROUT mode) are configured. The GENSYS parameter is required if the system is to execute the Generalized Tape Management System. * The MIGRAT parameter is required to migrate a 4/JS3 system to execute under GCOS 8. When a Page Processing System is configured into the system, either the RESVID/PPSOFL,id1[id2...]/ or the RESVID/PPSONL,id1[,id2...]/ parameter must be specified - but not both. A maximum of 64 ids can be specified. The NIAST ALL option is implied for all shared systems.

Startup Statements	Discussion
	The EXTTSS/xxx,yyy,zzz option must be specified for all Time Sharing systems. The OWNID/xxxxxxxxxxxx option must be specified when processing FIPS-79 tape labels. PROEXT must be specified when the six processor functionality is to be utilized.
$ BASE	Can define the maximum site mainframe configuration (i.e., number of input/output controllers, network processors (DNET/ROUT mode)) and the maximum amount of space allocated to memory-resident tables (e.g., ASTs). The $ BASE statement must precede any $ Iyy or $ MCT statements.
$ MCT	Defines the memory-processor-input/output controller configuration and connectability relationships for DPS 8 and DPS 90.
$ CIU	Defines the memory-processor-input/output controller configuration and connectability relationships for DPS 88 and DPS 8000.
$ Iyy	A series of input/output controller statements must be included to define the system's peripheral and communications configuration: mass storage subsystem, magnetic tape subsystem, unit record devices, consoles, and remote processors.
$ URP	Define each Unit Record Processor (URP800n) and all active unit record peripherals controlled by that URP800n. Unit record devices can also be specified on $ Iyy statements.
$ MPC	Defines the physical configuration of the MPC (i.e., the physical connection of the MPC and PSI channel to the input/output controller (Iyy) channel). Consult with the site's CSD representative for the exact physical relationship of these components. There is a direct relationship between the $ Iyy and $ MPC statements. Each input/output controller channel (CH/PUB) number specified on $ Iyy statements must also be identified on $ MPC statements. The following example defines IOM channel 8.

Startup Statements	Discussion
	<pre> 1 1 8 6 $ IOM-0 PUB-8,MS0450,UNITS-4 . . . $ MPC-1 SIZE-4,PSI-0,IOM-0,PUB-8</pre>
$ CHAN	A tool which allows channel switching for devices during the Startup process. No limit is set on the number of $ CHAN statements that can be used in a Startup job stream.
$ UNIT	Allows a switchable device or either of two separate devices to be referenced by the same logical name during startup. Any number of $ UNIT statements are permissible.
$ GCOSFIL	Identifies devices on which files that are defined in the $EDIT section can reside.
$ XBAR	Defines the crossbar configuration (i.e., multichannel access) of the mass storage and/or magnetic tape subsystems.
$ LOADFIL	Identifies the mass storage device on which the $LOAD section of the Startup job stream (i.e., the load file) is to reside.
$ IMAGE	Defines the character set to replace the default character set. If used, the $ IMAGE statement must precede the $ TRAINS statement and must be continued by using successive $ IMAGE statements (i.e., $ ETC statements are not permitted). If the conditions are not adhered to, Startup aborts.
$ TRAINS	Defines the character sets that are available on high-speed printers.
$ SHARED	Defines permanent mass storage devices on a specific system to be shared with the Network Processor Supervisor (NPS). Removable devices cannot be shared. The NIAST option of the $ INFO statement is automatically activated when $ SHARED is specified.
$ AUTOLD	Defines the size of the AUTOLOAD file, which accommodates the Startup program and job stream and is used when rebooting the system, and the mass storage device on which the file resides.

Startup Statements	Discussion
$ DECKFIL	Identifies the size of the deckfile (which resides on the AUTOLOAD device).
$ MPCFIG	Should be included to identify the firmware revision level that is to be bootloaded to a specific MPC. Consult with the site's CSD representative for the correct firmware revision identifier.
$INITIALIZE Section	Provides the basis for mass storage device initialization and is the means by which firmware is written to and read from the deckfile.
$ INIT	Identifies each mass storage device to be initialized (i.e., existing pointers and tables on the device are cleared and rebuilt). Any special parameters associated with a device (e.g., allocation unit size or device availability for FMS catalogs) also must be included. It is recommended that $ INIT statements be removed from the Startup job stream after the system is initialized. Removal prevents accidental initialization of mass storage devices (i.e., an operator response of YES to the *INITIALIZE? question). The statements can be replaced in the Startup job stream when initialization is desired.
MPCB Deck	Bootloads the MPC firmware.
Mass Storage Firmware	Bootloads the mass storage controller. (This deck enables mass storage initialization.)
Magnetic Tape Firmware	Bootloads the magnetic tape controller. (This deck enables reading of the tape that contains MPC firmware.)
$ READIN	Defines the device on which the firmware tape is mounted.
D, O, A, L, Y	Space-definition statements that are used to identify defective space on mass storage devices. If these statements are included in the Startup job stream, also include $ DIRECT.../CLEAR and $ DIRECT statements. D, O, A, L, and Y statements define defective space on the following bases. D - Cylinder and head numbers O - Beginning and ending sector numbers (octal) A - Beginning and ending word numbers (octal) L - Beginning link and number of links (octal) Y - Beginning llink and number of llinks (octal)

Startup Statements	Discussion
$EDIT Section	Each device identified in this section also must be defined on a $ Iyy statement in the $CONFIG section.
$ FILDEF	Defines all system files and the devices on which the files are to reside. Some specific files must be created. (Note that the files for which the RDM option is included must be identified on a $ LIBRARY statement in the $FILES section and files for which the SYS option is included must be identified on a $ SYSTEM statement in the $FILES section.) DUMP0 - This file is used by .MDUMP to store overlays and swapped memory. If DUMP0 is 200 llinks it will also be used to store an error snap of .MDUMP. This file can be created on a specific device or can be assigned to a device by Startup. DUMP1-DUMP9 - Used to store dump information. From one to nine files can be defined. SYSOUT Files - Files to which system output is spooled. The device on which each file resides must also be specified. It is recommended that each file be assigned to a different device, and - to avoid further contention problems - that no SYSOUT files be assigned to device ST1. All SYSOUT files must be the same size. If the files have different sizes, the smallest size specified is assigned by default to all files. All SYSOUT files also must be identified on $ SYSOUT statements in the $FILES section. PRINTIMAGE - Startup writes all printer character set and VFC images to this file, which is used to reload and re-create these images during subsequent bootloads. SCF - Define the mass storage device to which the Statistical Collection File (SCF) is assigned. The file also must be defined on the $ ACCOUNT statement in the $FILES section.

Startup Statements	Discussion
$ SSFILE	Defines System Scheduler job classes and class sizes. Additional parameters are specified on the $ SSLOAD statement in the $FILES section.
$ DTFILE/$ ENDTF	The $ DTFILE cues Startup that the statements following it are user-oriented directives. The directives that follow give users the ability to edit files for both site options and JCL, rather than hard-coding them into memory. This feature also provides default workstation initiation JCL for SYS_SOFTWARE. The $ ENDTF statement is required to terminate this function.
$FILES Section	Identifies files that are to be used for the duration of the current bootload. Each of these files also must be identified in the $EDIT section.
$ ACCOUNT	Identifies the device on which the Statistical Collection File (SCF) resides and specifies file-related characteristics (e.g., buffer size).
$ ACCBUF	Specifies record types buffered before being written to the SCF, whose buffering disposition can be changed via an operator request, and identifies the maximum record type buffered. The $ ACCBUF statement cannot be used when more than 36 accounting record type are required (use $ SCFBUF instead).
$ SCFBUF	Provides extended processing capabilities for a maximum of 288 accounting record types (compare to $ ACCBUF statement). This statement must be used when operating under the Tape Management System (GTMS) and must precede any $ SCFDSP statement.
$ SCFDSP	Defines the accounting record types which can have their status changed from the console. The $ SCFDSP statement must be preceded by a $ SCFBUF statement.
$ SYSTEM	Identifies files containing system programs that are necessary to site operations. A maximum of 32 files can be defined. File sizes and the device on which each file resides must be specified on $ FILDEF statements in the $EDIT section. (The $ FILDEF statements must include the SYS option.)
$ LIBRARY	Identifies a maximum of two random program files that also were specified on $ FILDEF statements in the $EDIT section. (The $ FILDEF statements must include the RDM option.)

Startup Statements	Discussion
$ SSLOAD	Defines System Scheduler job class parameters (e.g., maximum number of jobs that can be scheduled from one job class).
$ SAVE	Defines the Save file (i.e., a file named LUMP).
$ SYSOUT	Defines all SYSOUT collection files.
$ DFILES	Causes the referenced file to be cataloged as a linked sequential file under SYS_CAT and allows GEIN to read $ DTFILE related JCL statements.
$ PFILES	Identifies Startup-created files that are to be cataloged by FMS for subsequent access by users.
$PATCH Section	Contains OCTAL correction statements that are to be applied to GCOS programs and to software programs that were edited by Startup onto the system.
$LOAD Section	Allows object deck images of modules to be loaded directly into main memory. Each module included in this section also is written to the load file on mass storage. This section often is used to add new or to replace existing operating system programs.

Refer to "Bootload Phases" in Section 2 for operator procedures to bootload the system.

MASS STORAGE SUBSYSTEM - INITIAL CONFIGURATION

The following statements require special attention when initially configuring the site's mass storage subsystem.

Startup Statements	Discussion
$ IMU, $ IMX, $ IOM, $ IOP, or $ IOX	Defines each mass storage device and its physical and logical relationship to other components in the subsystem. The following fields deserve particular attention. UNITS - The number of units specified in this field must equal the combined number of mass storage units defined in the UNIT fields. AU - The default AU value is 1 for all devices except MSU0501, MSC1B, and MSD1B, which have a default value of 8. RMVBL - There are two advantages to defining several mass storage devices as removable (RMVBL). First, only permanent (PERM) devices must be initialized during a total system initialization. Second, the procedure for redefining a mass storage device is much easier when changing the definition from RMVBL to PERM than from PERM to RMVBL. (Refer to "Redefining A Mass Storage Device" in this appendix for additional information.) The RMVBL parameter cannot be used for a device defined as shared. Logical Device Name - Each device must be uniquely identified by a logical device name. One mass storage device must be named ST1. CH/PUB - The CH/PUB number must agree on $ Iyy, $ MPC, and $ XBAR statements. If multiple device types (e.g., MSU0402 and MSU0451) are configured on the same PSI, each CH/PUB (string) is defined on a separate $ Iyy statement.
$ MPC	All CH/PUB numbers that are included on $ Iyy statements must be defined on $ MPC statements. The first PSI specified on this statement is the PSI over which bootloading occurs.
$ XBAR	If the mass storage subsystem is crossbarred, all Iyy channel (CH/PUB) numbers that are identified on $ MPC statements must be included on $ XBAR statements. The PUB or CH number included on the $ Iyy statement must be the first defined on the associated $ XBAR statement.

Startup Statements	Discussion
	A CH/PUB number cannot be defined on more than one $ XBAR statement. The following example reflects the relationship among $ IOM, $ MPC, and $ XBAR statements.

```
                  1
1       8         6
------------------------------------------------------------
$       IOM-0     PUB-8,MS0400, ...
$       IOM-0     PUB-9,MS0450, ...
$       MPC       SIZE-4,IOM-0,PUB-8,PUB-9,PUB-10,PUB-11
$       XBAR      IOM-0,PUB-8,PUB-10
$       XBAR      IOM-0,PUB-9,PUB-11
```

$ INIT	A $ INIT statement must be included in the Startup job stream for each configured mass storage device during initial installation. The following fields deserve particular attention. CAT - FMS catalogs can be created on this device. It is recommended that several devices be made available for this purpose. This approach allows expansion of the FMS catalog structure. In addition, if FMS catalogs are not allowed on any devices, catalogs are restricted to device ST1. This practice can impact system efficiency. AU - The default AU value is 1 for all devices except MSU0501, MSC1B, and MSD1B, which have a default value of 8.
$ INFO	A configured mass storage device can be released via the RLSDSK option (e.g., the device must be released for maintenance).

MAGNETIC TAPE SUBSYSTEM - INITIAL CONFIGURATION

The following statements require special attention when initially configuring the site's magnetic tape subsystem.

Startup Statements	Discussion
$ IMU, $ IMX, $ IOM, $ IOP, or $ IOX	Define each magnetic tape device and its physical and logical relationship to other components in the subsystem. The following fields deserve particular attention. CH/PUB - The Iyy channel number must agree on $ Iyy, $ MPC, and $ XBAR statements. Logical Device Name - Each device must be uniquely identified by a logical device name. UNITS - The number of units specified in this field must equal the combined number of magnetic tape units defined in the UNIT fields. Subsystem Component Characteristics - Three fields (device type, density code, and magnetic tape characteristics) combine to define the physical characteristics of magnetic tape subsystem components. The device type and density fields define characteristics of most of the devices in the subsystem. The magnetic tape characteristics field defines individual magnetic tape units that are exceptions to information specified in the other two fields. For example, if six MTS0500 units have nine-track and 800/1600 bpi capabilities and two MTS0500 units have seven-track and 556/800 bpi capabilities, the following fields must be included on a $ Iyy statement. TAPE*500*9,CD UNIT-5,BC7 UNIT-6,BC7
$ MPC	All CH/PUB numbers included on $ Iyy statements must be defined on $ MPC statements.
$ XBAR	If the magnetic tape subsystem is crossbarred, all Iyy channel (CH/PUB) numbers that are identified on $ MPC statements must be included on $ XBAR statements. The CH/PUB number included on the $ Iyy statement must be the first defined on the associated $ XBAR statement. A CH/PUB number cannot be defined on more than one $ XBAR statement.

UNIT RECORD DEVICES - INITIAL CONFIGURATION

The following statements require special attention when initially configuring the site's unit record devices.

Startup Statements	Discussion
$ IMU, $ IMX, $ IOM, $ IOP, or $ IOX	Defines each unit record device (i.e., printers, card punches, and card readers) and its physical and logical relationship to other components in the configuration. One $ Iyy statement is required for each device. The following fields deserve particular attention. Logical Device Name - Each device must be uniquely identified by a logical device name. CH/PUB - The Iyy channel number must agree on $ Iyy and $ MPC statements. TRAIN - When defining a printer, the character set code must be identified in this field and on a $ TRAINS statement. No more than three device types (with differing characteristics) can be defined for one MPC. A maximum of eight devices can be defined. The following combination of devices is valid for one MPC. One CRU1050 card reader One PCU0120 card punch Two PRU1100 printers However, the following combination of devices is invalid for one MPC because there are more than three device types with differing characteristics. One CRU1050 card reader One PCU0120 card punch One PRU1100 printer One PRU1600 printer
$ MPC	Iyy channel (CH/PUB) numbers must be sequentially identified (e.g., PUB-21, PUB-22, PUB-23, PUB-24 and not PUB-21, PUB-22, PUB-24).

Startup Statements	Discussion
$ TRAINS	This is a list of all print trains that embrace the site's standard printed character set. The sequence of the characters on the printer belt is not significant as long as it contains all the characters in the set. Any of the BCD belts listed on this statement may be substituted for the Startup defined BCD standard belt when the only printer allocation requirement is for a BCD standard printer. Standard ASCII belts may also be substituted for the Startup-defined ASCII standard belt.
$ XBAR	Crossbarring of unit record devices is not permitted.

DATANET NETWORK PROCESSORS - INITIAL CONFIGURATION

The following input/output controller statement considerations require special attention when initially configuring the site's DATANET Network Processors.

1. DATANET Network Processors must be numbered sequentially (i.e., 0, 1, 2 and not 0, 2, 3).

2. $ Iyy statements must sequentially define DATANET Network Processors in the Startup job stream. For example,

```
               1
1      8       6
------------------------------------------------------------------
$      IOM-1   PUB-p,6620-0
$      IOM-1   PUB-p,6620-1
$      IOM-1   PUB-p,6620-2
```

   and not

```
               1
1      8       6
------------------------------------------------------------------
$      IOM-1   PUB-p,6620-2
$      IOM-1   PUB-p,6620-0
$      IOM-1   PUB-p,6620-1
```

   However, channel (PUB-p) numbers do not have to be in sequence.

3. The value specified in the LINES-x field defines the maximum number of users that can be connected at any one time.

   Therefore, if the value x is less than the number of lines configured and available, the undefined lines are not available for use. For example, if 50 lines are configured but LINES-20 is specified on the $ Iyy statement, only 20 users can be connected at one time.

If the value x is greater than the number of lines configured and available, no more lines than are configured can be used. For example, if 50 lines are configured but LINES-60 is specified on the $ Iyy statement, only 50 users can be connected at one time.

MODIFY/ADD SYSTEM FILE

The following statements should be reviewed and operator procedures followed when modifying an existing system file (e.g., increasing the file size) or when adding a new system file.

Startup Statements	Discussion
$ FILDEF	If a new file is being added, the file name and size and the device on which the file is to reside must be identified. If the size of an existing file is being increased, it is recommended that a new file name be assigned and that the file size be increased. (In addition, the appropriate changes on the $ SYSTEM and $ LIBRARY statements must be made. Note that the replaced file continues to exist after the new file is created.) If the file merely is being assigned to a different device, identify the device in the logical device name field. Retain the same file size and name.
$ GCOSFIL	Identify the device to which the file is being assigned. Startup searches devices in the sequence in which they appear on the $ GCOSFIL statement.
$ IMU, $ IMX, $ IOM, $ IOP, or $ IOX	Verify that the mass storage device to which the file is assigned is defined.
$ SYSTEM	If a new file is being added, the file must be identified on a $ SYSTEM statement. Files are searched in the order in which $ SYSTEM statements are encountered. Therefore, if the file is to override the content of an existing file, the $ SYSTEM statement that identifies the new file must precede the $ SYSTEM statement that identifies the existing file.

The operator must edit the system file. Respond PARTIAL to the *EDIT? question that is issued to the system console and respond YES to the *EDIT fffff? question (where fffff = name of the system file).

AUTOLOAD FILE AND DECKFILE MODIFICATION

The following statements should be reviewed and operator procedures followed when modifying either the AUTOLOAD file or the deckfile (e.g., increasing the size of the deckfile).

Although the AUTOLOAD file and deckfile are not directly related in function, the deckfile is always created on the same device as the AUTOLOAD file. Therefore, moving either of the files from one device to another affects both files.

Startup Statements	Discussion
$ AUTOLD	The size of the AUTOLOAD file normally will not require modification (i.e., the default size of 600 llinks is sufficient). However, moving the AUTOLOAD file from one device to another automatically moves the deckfile from one device to another. The new device must be identified on the $ DECKFIL statement.
$ DECKFIL	Changing the size of the deckfile requires that either the file be assigned to another device or the file remain on the same device and be renamed. If the deckfile is moved to another device, the AUTOLOAD file also must be assigned to the new device. As a result of redefining the two files, space that previously was assigned to the files is lost to the system.
$ GCOSFIL	Identifies the device to which the files are being assigned.
$ IMU, $ IMX, $ IOM, $ IOP, or $ IOX	Verifies that the mass storage device to which the files are assigned is defined.
$ READIN	To re-edit the deckfile, a $ READIN statement is included in the $INITIALIZE section of the Startup job stream.

The operator must re-edit the deckfile. Respond PARTIAL to the *INITIALIZE? question that is issued to the system console. Then, respond YES to the CLEAR AND OVERWRITE EXISTING DECKFILE? question and YES to the *PROCESS $ READIN? question.

LOAD FILE MODIFICATION

The following statements should be reviewed when modifying the load file (e.g., moving the file to another device). No re-editing is required. Therefore, operator intervention is not necessary.

Startup Statements	Discussion
$ LOADFIL	Changing the size of the load file requires that either the file be assigned to another device or the file remain on the same device and be renamed. The device on the $ LOADFIL statement must be specified. As a result of redefining the file, space that previously was assigned to the file is lost to the system.
$ GCOSFIL	Identifies the device to which the file is being assigned.
$ IMU, $ IMX, $ IOM, $ IOP, or $ IOX	Verifies that the mass storage device to which the file is assigned is defined.

SAVE FILE MODIFICATION

The following statements should be reviewed and operator procedures followed when modifying the Save file (e.g., increasing the file size). (The Save file frequently is referred to as the LUMP file. However, any file name can be assigned to the file.)

Startup Statements	Discussion
$ FILDEF	If the size of the existing file is being increased, the file (with the same file name) should be assigned to a different device. The new device on the $ GCOSFIL statement must be identified before identifying the device on which the file currently resides. This creates a new file of the same name. However, the space that is assigned to the current file is lost to the system. If the file merely is being assigned to a different device, the device is identified in the logical device name field. The same file size and name is retained.
$ GCOSFIL	Identifies the device to which the file is being assigned.
$ IMU, $ IMX, $ IOM, $ IOP, or $ IOX	Verifies that the mass storage device to which the file is assigned is defined.
$ SAVE	If the file name is being changed, the file is identified on the $ SAVE statement.

The operator must edit the save file. Respond PARTIAL to the *EDIT? question that is issued to the system console and respond YES to the *EDIT fffff? question (where fffff = name of the save file).

MODIFY/ADD SYSOUT FILE

The following statements should be reviewed and operator procedures followed when modifying an existing SYSOUT file (e.g., increasing the file size) or when adding another SYSOUT file.

Startup Statements	Discussion
$ FILDEF	If a new file is being added, the file name and size and the device on which the file is to reside is identified. To avoid device contention, it is recommended that SYSOUT files be assigned to separate devices. If the size of an existing file is being increased, the file (with the same file name) should be assigned to a different device. The new device on the $ GCOSFIL statement must be identified before identifying the device on which the file currently resides. This creates a new file of the same name. However, the space that is assigned to the current file is lost to the system. If one SYSOUT file size is changed, all SYSOUT files should be changed. If SYSOUT files are not the same size, the usable space of all SYSOUT files is reduced to the size of the smallest. If the file merely is being assigned to a different device, the device in the logical device name field. The same file size and name is retained.
$ GCOSFIL	Identifies the device to which the file is being assigned.
$ IMU, $ IMX, $ IOM, $ IOP, or $ IOX	Verifies that the mass storage device to which the file is assigned is defined.
$ SYSOUT	If a new file is being added, the file must be identified on a $ SYSOUT statement.

The operator must edit the SYSOUT file. Respond PARTIAL to the *EDIT? question that is issued to the system console and respond YES to the *EDIT fffff? question (where fffff = name of the SYSOUT file). In addition, respond NO to the *RESTART? question and YES to the *SYSTEM SCHEDULER CLEAR? question.

CONFIGURING A SYSTEM COMPONENT

The following procedure is recommended when adding a new hardware component (e.g., mass storage device, magnetic tape unit, processor, or input/output controller) to the system configuration.

1. Complete all jobs in process. Adding a new hardware component to the system configuration modifies the Hard Core Monitor (HCM) layout. Therefore, all existing references to jobs in process would change.

2. Enter BOOT or DUMP via the system console to render the system inoperable.

3. Insert a $ Iyy statement into the Startup job stream to define the component that is being added to the configuration.

4. If the new component is involved in a crossbarring scheme, modify $ XBAR statements to reflect the new component.

5. Review the $ MPC statements to ensure that all existing Iyy channel (CH/PUB) numbers are defined.

6. Review the $ MCT or $ CIU statement (as applicable) if either a processor or an input/output controller is being added to the configuration to ensure that the new component is identified.

When rebooting the system, the operator must respond NO to the *RESTART? question and YES to the *SYSTEM SCHEDULER CLEAR? question.

DECONFIGURING A SYSTEM COMPONENT

The following procedure is recommended when deleting a hardware component (e.g., magnetic tape unit, processor, or input/output controller) from the system configuration.

1. Complete all jobs in process. Deleting a component from the system configuration modifies the Hard Core Monitor (HCM) layout. Therefore, all existing references to jobs in process would change.

2. Remove from the Startup job stream the $ Iyy statement defining the component being deleted from the system configuration.

   Note, however, that this must not be done to deconfigure a mass storage device. The device must remain configured, but can be logically released from the system via the following procedure.

   The device can be physically disconnected from the system, but the $ Iyy statement defining the device must be included in the Startup job stream. During Startup's mass storage device verification procedure, the console message:

   ```
   *DISK PACK ERROR icccdd EOM TO CONTINUE--
    'RLSE' TO RELEASE THE DEVICE
   ```

   is issued when Startup determines that the device is not physically connected. The operator then can logically release the device from the configuration via the console entry RLSE.

   Alternatively, the preceding console message can be pre-answered by including the RLSDSK option on the $ INFO statement.

3. If the deconfigured component is involved in a crossbar scheme, modify $ XBAR statements to reflect the deletion.

4. Review the $ MPC statements to ensure that only existing Iyy channel (CH/PUB) numbers are defined.

5. Review the $ MCT or $ CIU statement (as applicable) if either a processor or an input/output controller is being deleted from the configuration to ensure that the component is not identified.

During the bootload process, the operator must respond NO to the *RESTART? question and YES to the *SYSTEM SCHEDULER CLEAR? question.

REDEFINING A MASS STORAGE DEVICE

The following procedure must be used to change the definition of a mass storage device from removable (RMVBL) to permanent (PERM).

1. Remove the RMVBL parameter from the applicable $ Iyy statement.

2. Respond YES to the console message *ddd PRIOR DEVICE STATUS WAS RMVBL CHANGED TO PERM YES TO CONTINUE.

The following procedure can be used to change the definition of a mass storage device from permanent (PERM) to removable (RMVBL).

1. Include the RMVBL parameter on the applicable $ Iyy statement.

2. Respond YES or TOTAL to the *INITIALIZE? question that is issued to the system console and YES to the *EDIT? question.

MODIFY/ADD PRINTER CHARACTER SET

The following statements should be reviewed and operator procedures followed when modifying an existing printer character set or when adding a new character set.

Startup Statements	Discussion
$ IMAGE	Defines the character set. All $ IMAGE statements must precede the $ TRAINS statements.
$ TRAINS	Identifies the character set.
$ IMU, $ IMX, $ IOM, $ IOP, or $ IOX	Determines whether the name (nnnnn) in the TRAIN-nnnnn field is to be modified.

The operator must re-edit the PRINTIMAGE file. Include a $ FILDEF statement in the Startup job stream to identify the PRINTIMAGE file. Respond PARTIAL to the *EDIT? question that is issued to the system console and respond YES to the *EDIT PRINTIMAGE? question.

REDEFINING SYSTEM SCHEDULER PARAMETERS

The following statements should be reviewed and operator procedures followed when moving the SSFILE file from one device to another or when changing System Scheduler job class parameters.

Startup Statements	Discussion
$ SSFILE	If the file is being assigned to a different device, the device is identified in the logical device name field. If job class parameters are being changed, only those parameters to be changed need modification.
$ SSLOAD	Modifies job class parameters as required or adds/deletes job classes.
$ GCOSFIL	If the file is being assigned to a different device, the device to which the file is being assigned is identified. The new device name must appear in the $ GCOSFIL card before the old device name.
$ IMU, $ IMX, $ IOM, $ IOP, or $ IOX	Verifies that the mass storage device to which the file is assigned is defined.

The operator must edit the SSFILE file. Respond PARTIAL to the *EDIT? question that is issued to the system console. Then, respond YES to the question *PROCESS SSFILE CARD AND SYSTEM SCHEDULER CLEAR? so that the $ SSFILE statement is read.

APPENDIX C

SUMMARY OF $ INFO PARAMETERS

See Section 5 for detailed descriptions.

$ INFO Parameter	Brief Description
ACALL/ {nn } {TSS} {TPE} {TSx}	Automatically execute ACALL following fast boot restart. Reconnect network processor nn, and/or restart TPE or TSS.
ASCII	An ASCII printer is configured.
CATDUP	Duplicate all File Management Supervisor (FMS) catalogs.
CATFNC/n	Catalog fence. Allocate Startup files above llink n.
CCACHE/nnn	Memory buffers are reserved for FMS catalog cache.
CHKSUM	Checksum all modules during the module-load process.
CLENPT	Enable the cleanpoint function for system restart.
CNTDCW	Include # of words transferred via disk I/O in SCF data.
DENxx/DENyy	Default high and low densities for MPC tapes.
DPSE	Must be present if DPS 8/50 or 8/70 is configured.
ENCRYP	Encrypt SMC passwords.
EXTTDS/xxx,yyy	Amount of memory available to TDS or DM-IV/TP.
EXTTSS/xxx,yyy[,zzz]	Amount of memory available to TSS.
FASTBT	Enable fast boot for automatic system restart.
FMSCT/n	Max % of FMS catalog area that may be used for files.
FMSDT/n	Max number of descriptors for selected disk device.
FMSST/n	Max % of space on selected device.
FMSTAT	Accumulate statistics for FMS.

$ INFO Parameter	Brief Description
FNP/n	Number of network processors for DNET/ROUT.
GENSYS	Enable Generalized Tape Management System (GTMS).
GEOTRC[/n]	Enable GEOT trace for SYSOUT delivery trouble analysis.
HEX/ {ON } {OFF}	Controls acceptance of MME GMODES requests for setting HEX floating point mode.
INTERRUPT/xxx	Type of interrupt processing for multiprocessor systems.
KLFTH/n	Native mode memory management parameter.
KLMIW/ssss	Native mode memory management parameter.
KLMWP/ssss	Native mode memory management parameter.
KLMWS/ssss	Native mode memory management parameter.
KLWPR/xxxx	Native mode memory management parameter.
KLWSW/xxxx	Native mode memory management parameter.
LIMITS	Operator control when program exceeds SYSOUT or time limit.
LINES/n	The maximum number of lines per page of printer output.
LOGON	Sets flags to recognize that LOGON may be operational.
MCPTSS/n	The number of Time Sharing copies.
MEMORY/n	Sieve limit for maximum memory size.
MIGRAT	Migrate SR 4/JS3 Startup-defined files to GCOS 8.
MIXED	CPL and NPL peripheral device types coexist.
MODULE/.mmmmm	Modules specified to reside in main memory.
NDFMAP	The deckfile map is not to be printed.
NIAST/ {ALL/n } {RMVBL/n}	No In-memory Available Space Table.
NPCHSZ/nnn	Enables the application of virtual mode patches.
NPRINT	No printer is available to the Startup program.
NPUNCH	No card punch is configured for local punch output.

$ INFO Parameter	Brief Description
NSYASC	No SYOT ASCII Space Compression.
NSYBSC	No SYOT BCD Space Compression.
NSYSAV	No SYOT Seek Address Validation.
NSYTRC	No SYOT TRaCe.
OWNID/xxxxxxxxxxxx	Default owner-id for FIPS-79 UFAS tape labels.
PASSWD/nnn	Password expiration interval.
PROEXT	Enable the six-processor functionality.
PSUM/nnn	The maximum number of processes allowed in the system.
PT1SIZ/nnnn	The size of the page table for working space 1.
PTCHSZ/nnn	The maximum number of patches allowed in $PATCH section.
RESVID/{FORMS }{PPSOFL}{PPSONL},id1[,id2,...,idn]/	Reserve station ids for collection of special output.
RLSDSK	Startup is to release offline mass storage devices.
RLSPNT	Startup is to release offline printers.
ROLLCALL/ddd	Direct Startup rollcall messages to printer.
SAVDMP/list	Defines default options for save dumps.
SCFPRG	The SCF purge-to-disk option is enabled.
SCHSAV	Save all job input for System Scheduler.
SDCOL/n	Number of additional pages for type 3 GEPR records.
SLAVE/n	Maximum size of slave instruction segment.
SLINKS/n	Sieve limit for temporary file mass storage llinks.
SLTAPE/n	Sieve limit for number of magnetic tape files.
SLTIME/hhmm	Sieve limit for processor time.
SMCDUP/ddd	Duplicate System Master Catalog (SMC).
SPN,umc_1[/cat-or-file-1],umc_2[/cat-or-file-2]	System path name replacement.
SSAOVL/n	Number of real pages allocated to SSA cache.

$ INFO Parameter	Brief Description
SYBDI/n	Size of backdoor information segment.
SYBRT/n	GEOT blink release threshold.
SYJOS/n	Size of job output special segment.
SYJOT/n	Size of job output table segment (SNUML).
SYSOUT/n	Sieve limit for maximum number of SYSOUT records.
SYSTEMMAP/NO	Do not print the map of the GCOS file structure.
TAPDEN/xxxx	Identifies the configured tape densities.
TIMEZ/zzz	Specifies local time zone.
TS8	Requires that TS8 module .MTSVC be edited.
USRDMP/list	Default options for user dumps.
WSSIZE/2,s2,b2 WSSIZE/3,s3,b3	Set virtual sizes and backing store file coverage of system shared workspaces.

INDEX

IONEYWELL BULL
echnical Publications Remarks Form

ITLE	LARGE SYSTEMS GCOS 8 OS SYSTEM STARTUP	ORDER NO.	DH18-03
		DATED	AUGUST 1987

RRORS IN PUBLICATION

JGGESTIONS FOR IMPROVEMENT TO PUBLICATION

Your comments will be investigated by appropriate technical personnel and action will be taken as required. Receipt of all forms will be acknowledged; however, if you require a detailed reply, check here. ☐

PLEASE FILL IN COMPLETE ADDRESS BELOW.

IOM: NAME ______________________ DATE __________

TITLE ______________________

COMPANY ______________________

ADDRESS ______________________

____

PLEASE FOLD AND TAPE–
NOTE: U.S. Postal Service will not deliver stapled forms

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 39531 WALTHAM, MA

POSTAGE WILL BE PAID BY ADDRESSEE

Honeywell Bull Inc.
200 Smith Street MS486
P.O. Box 9199
Waltham, Massachusetts, 02254-9832

Honeywell Bull

Honeywell Bull

Corporate Headquarters:
3800 West 80th St., Minneapolis, MN 55431
U.S.A.: 200 Smith St., MS 486, Waltham, MA 02154
Canada: 155 Gordon Baker Rd., North York, ON M2H 3P9
Mexico: Av. Constituyentes 900, 11950 Mexico, D.F. Mexico
U.K.: Great West Rd., Brentford, Middlesex TW8 9DH **Italy:** 32 Via Pirelli, 20124 Milano
Australia: 124 Walker St., North Sydney, N.S.W. 2060 **S.E. Asia:** Mandarin Plaza, Tsimshatsui East, H.K.

49660, 7.5C1087, Printed in U.S.A.