File No. S360-31
Form C28-6554-3

OS

IBM Systems Reference Library

IBM System/360 Operating System

System Generation

System generation is a process that generates an IBM System/360 Operating System adapted to both the machine configuration and the data processing requirements of an installation. The system generation process is performed under the control of an existing IBM System/360 Operating System. This publication provides information on the machine and operating system requirements for system generation, the initialization of system volumes and data sets, the macro-instructions used in specifying system generation, the methods of including user-written programs in the operating system, restart procedures, and the sample programs used to test the new system.

IBM provides a starter operating system that can be used for the first system generation. The procedures required to initialize the starter system are also described in this publication.

PREFACE

This publication provides system programmers with information about the system generation process provided with IBM System/360 Operating System. The reader must be familiar with the various facilities and options offered within IBM System/360 Operating System, and must have already decided upon the operating system configuration to be generated.

This publication tells him how to generate the operating system configuration he has selected. The following IBM System/360 Operating System publications are required to understand and select the operating system to be generated:

IBM System/360 Operating System: Introduction, Form C28-6534

IBM System/360 Operating System: Concepts and Facilities, Form C28-6535

IBM System/360 Operating System: Storage Estimates, Form C28-6551

IBM System/360 Operating System: System Programmer's Guide, Form C28-6550

The following publications are required for a better understanding of the system generation process:

IBM System/360 Operating System: Job Control Language, Form C28-6539

IBM System/360 Operating System: Assembler Language, Form C28-6514

Reference is made throughout this publication to several utility programs. The reader must be familiar with the requirements and facilities they provide. The utility programs are described in the publication:

IBM System/360 Operating System: Utilities, Form C28-6586

Fourth Edition (August 1967)

This edition corresponds to Release 13 and is a major revision of Form C28-6554-2 and obsoletes it. In addition to incorporating information released in Technical Newsletters N28-2230 and N28-2252, significant changes have been made. All sections, except "Specifying the System," are either new or have been completely rewritten and should be reviewed in their entirety. Changes to the section "Specifying the System" are indicated by a vertical line to the left of the change; revised illustrations are denoted by the symbol • to the left of the caption.

The IFCDIP00 and IEHIOSUP utility programs that were previously described in this publication, are now described in IBM System/360 Operating System: Utilities, Form C28-6586.

This publication also incorporates the information in IBM System/360 Operating System: Starter Operating System Guide, Form C28-6630-1 and in Technical Newsletter N28-2254, which are now obsolete.

Specifications contained herein are subject to change from time to time. Any such change will be reported in subsequent revisions or Technical Newsletters.

This publication was prepared for production using an IBM computer to update the text and to control the page and line format. Page impressions for photo-offset printing were obtained from an IBM 1403 Printer using a special print chain.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form is provided at the back of this publication for reader's comments. If the form has been removed, comments may be addressed to IBM Corporation, Programming Systems Publications, Department D58, PO Box 390, Poughkeepsie, N. Y. 12602

FIGURES

TABLES

IBM System/360 Operating System is composed of modules that can be united in a variety of combinations to meet the given requirements of a particular installation. The user selects the programming options that meet his data processing requirements and that conform to the processing, storage, and input/output facilities of his machine configuration. System generation is the process of interpreting the user's selection and combining the operating system modules into the system data sets that form the installation's new operating system.

The new operating system is composed of the standard features incorporated in every operating system, those optional features selected from the distributed modules, and any additional features provided by the user. System generation also provides facilities for adding more features to the operating system after it has been generated.

The system generation process, the system data sets, and the types of system generation are discussed in the following sections.

THE SYSTEM GENERATION PROCESS

System generation is a process that generates an IBM System/360 Operating System adapted to both the machine configuration and the data processing requirements of an installation. The desired operating system is specified by the user through system generation macro-instructions. During the system generation process, several operating system programs are used to build a new operating system tailored according to the specifications in the macro-instructions. These programs are executed under the control of an existing operating system (also called "generating operating system" or simply "generating system" throughout this publication).

An operating system is generated in two stages (see Figure 1). During Stage I, user-supplied macro-instructions that describe both the installation's machine configuration and the programming options desired are analyzed and used to generate a job stream. In Stage II, this job stream is processed to generate the libraries of modules that form the user's operating system. These libraries contain modules supplied by IBM and, optionally, modules supplied by the user.

STAGE I: PRODUCING THE JOB STREAM

Stage I consists of two phases. During the first phase all the macro-instructions supplied by the user are analyzed for errors. Error messages are written for each error found. If errors are not found in any of the macro-instructions, a job stream is produced during the second phase. If errors are found, the second phase is bypassed and the job stream is not produced.

STAGE II: PROCESSING THE JOB STREAM

During Stage II the job stream is processed by the assembler, by the linkage editor, and by utilities. The following functions are performed:

- Selected modules are assembled.
- The linkage editor combines the modules selected for inclusion in the resident portion of the control program (nucleus).
- The linkage editor processes those selected modules to construct members of the new operating system libraries.
- Utility programs complete the construction and initialization of the libraries selected for the new operating system.

The generated operating system is then ready for use.

SYSTEM DATA SETS

The modules that comprise the operating system are contained in system data sets. The system data sets are the system catalog, the system libraries, and data sets such as SYS1.SYSJOBQE. The system libraries are partitioned data sets. All other system data sets are sequential data sets.

The following system data sets are required for every operating system:

- SYSCTLG (System Catalog) -- The system catalog contains pointers to all cataloged data sets.
- SYS1.NUCLEUS (Nucleus library) -- This library usually contains only one member, the resident portion (nucleus) of the control program. To include additional members, see the description of the GENERATE macro-instruction in the section entitled "Specifying the System."
- SYS1.SVCLIB (SVC library) -- The members of the SVC library are the nonresident SVC routines, the data management access methods, and the system's standard error recovery routines.
- SYS1.LOGREC -- This data set is used to record statistical data about machine errors.
- SYS1.LINKLIB (Link library) -- The members of the link library are programs and routines that can be referred to by XCTL, ATTACH, LINK, or LOAD macro-instructions, or by EXEC statements. Nonresident operating system programs, e.g., the COBOL compiler, are contained in this library.
- SYS1.PROCLIB (Procedure library) -- The members of the procedure library include those cataloged procedures used to perform certain system functions, e.g., compile-linkage edit-go.
- SYS1.SYSJOBQE -- This data set is used as a work area by the job scheduler.

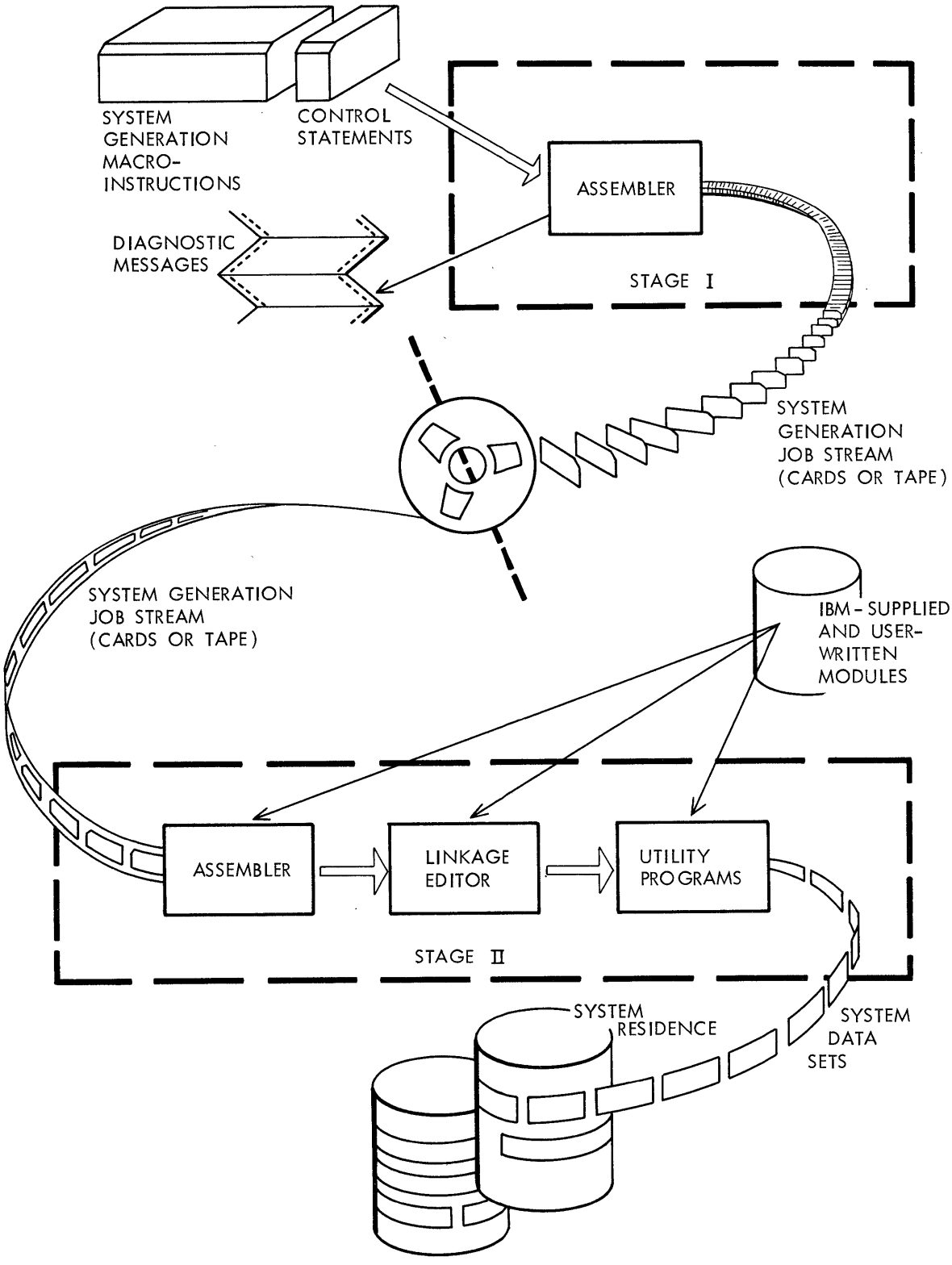


Figure 1. The System Generation Process

The following system data sets are optional:

- SYS1.MACLIB (Macro library) -- The members of the macro library include the macro-definitions for the system macro-instructions.

- SYS1.SORTLIB (Sort library) -- The members of the sort library are the load modules from which the system produces a sort/merge program at execution time.

- SYS1.ALGLIB (ALGOL library) -- The members of the ALGOL library are load modules (ALGOL subroutines).

- SYS1.COBLIB (COBOL library) -- The members of the COBOL library are load modules (COBOL subroutines).

- SYS1.FORTLIB (FORTRAN library) -- The members of the FORTRAN library are load modules (FORTRAN subprograms).

- SYS1.PL1LIB (PL/I library) -- The members of the PL/I library are load modules (PL/I subprograms).

- SYS1.TELCMLIB (Telecommunications library) -- The members of the telecommunications library are load modules (telecommunications subroutines).

- SYS1.SYSVLOGX and SYS1.SYSVLOGY (System log data sets) -- These data sets are used to record write-to-log (WTL) messages before they are printed on the system output unit. These two data sets are used with multiprogramming with a variable number of tasks (MVT) only.

The following system data sets are required for system generation only:

- SYS1.GENLIB (Generation library) -- This library contains the macro-definitions of the system generation macro-instructions.

- SYS1.MODLIB (Module library) -- The members of the module library are the load modules from which an operating system is generated.

The following system data set is originally distributed with the starter operating system, and is usually kept in card decks by the installation.

- SYS1.SAMPLIB (Sample library) -- The members of the sample library are the sample programs used to test operating system components (described in the section "Testing the New System"), the independent utility programs (IBCDASDI, IBCDMPRS, and IBCRCVRP), the IPL program (IEAIPL00), a dump program (PROCDUMP), and an example of writing an accounting routine (SAMACTRT).

TYPES OF GENERATION

During each system generation process the user can specify one of three types of generation:

- Complete Operating System generation.
- Nucleus generation.
- Processor and Library generation.

In the first type, the user specifies the generation of an operating system consisting of either a control program only, or a control program and language processors and their associated libraries. The control program specified can be the primary control program (PCP), multiprogramming with a fixed number of tasks (MFT), or multiprogramming with a

variable number of tasks (MVT). Primary data management routines, and system utilities are always provided with this type of generation. These standard features and the optional features specified are adapted to the installation's machine configuration during the generation process. An Operating System generation must be performed whenever the installation's machine configuration is modified, or whenever changes are to be made to the system generation options for the control program. (An Operating System generation may not be needed if only changes to the nucleus of the control program are to be made. In this case, a new nucleus can be added to the operating system through a Nucleus generation.)

In the other types of generation, the user specifies that a nucleus, or that language processors and/or their associated libraries are to be added to the operating system. During the preparation for a Processor and Library generation, the user must allocate space for, and, if desired, catalog, any new system data set to be added to the operating system. If any of the existing system data sets are to be modified, the user must have allocated sufficient space to those data sets when they were initially generated or, where permitted, provided for multiple extents in those data sets. (SYS1.NUCLEUS is the only system data set affected by a Nucleus generation.)

During a Nucleus generation or a Processor and Library generation, the generating system may also be the system being modified. That is, the generating system can add a new nucleus or processors and libraries to itself. To faciliate the description of system generation, the terms "new operating system" or "new system" are used throughout this publication to mean either the operating system generated in a Complete generation, or the operating system modified by a Nucleus or a Processor and Library generation.

Figure 2 illustrates the three types of system generation. A complete generation is performed first to produce the installation's new operating system. A Processor and Library generation is used to add the PL/I processor and SYS1.PL1LIB to the new operating system. A nucleus generation adds a second nucleus to SYS1.NUCLEUS.

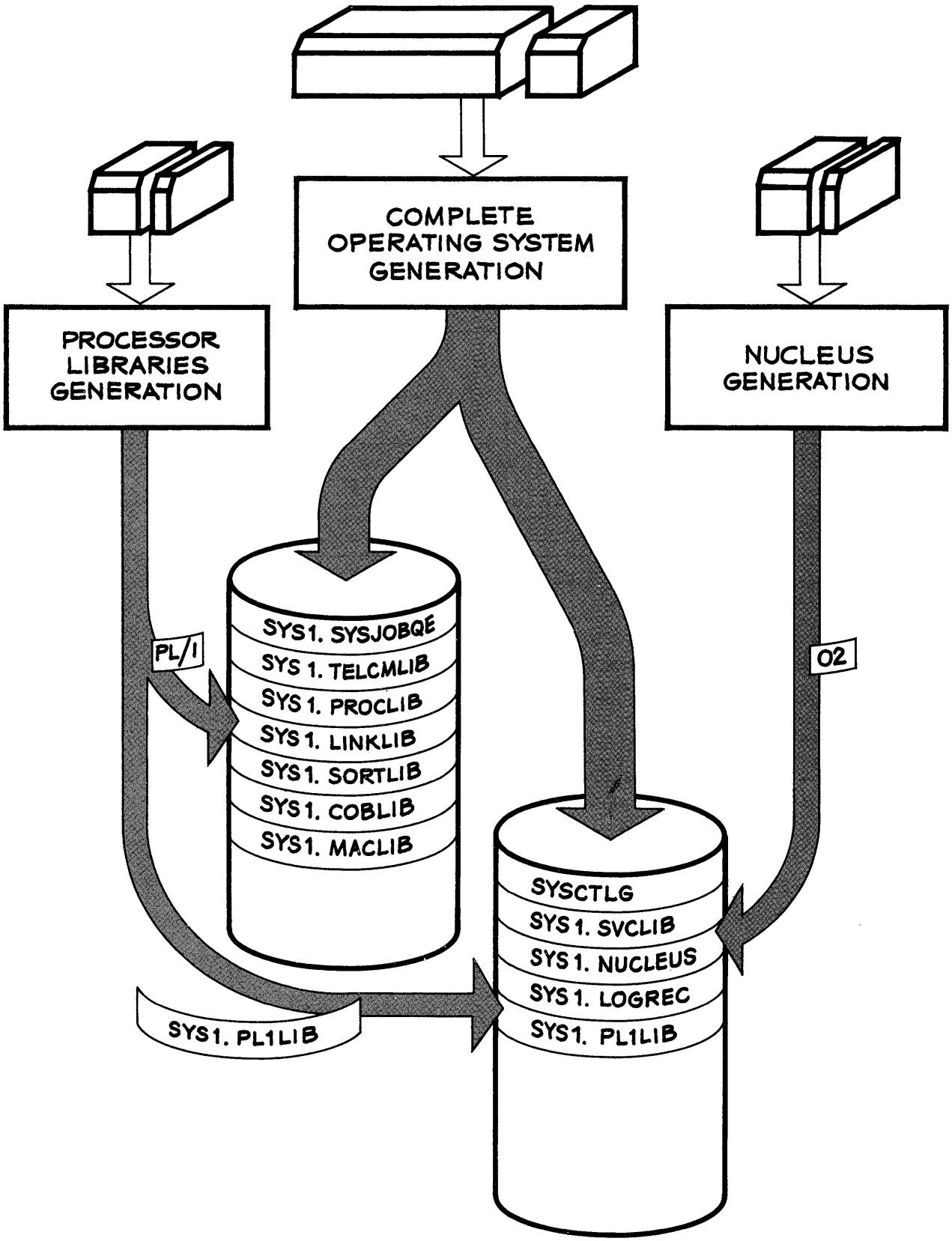


Figure 2. Types of System Generation

To generate an operating system, the installation's machine configuration and its current operating system must fulfill certain minimum requirements. These requirements are described in the following text.

MACHINE REQUIREMENTS

The following minimum System/360 configuration is required to generate an operating system:

- 64K bytes of main storage
- One console device
- Two direct-access drives (one drive must be either a 2311 or a 2314)
- One card reader or one magnetic tape drive (input)
- One card punch or one magnetic tape drive (intermediate output)
- One printer or one magnetic tape drive (print output)

OPERATING SYSTEM REQUIREMENTS

System generation is performed under the control of an existing PCP or MFT operating system and is executed as any other job. The operating system used to perform a system generation process must have system data sets, utility data sets, and, if desired, data sets that contain user-written programs to be included in the new system.

IBM provides a starter operating system that can be used for the first system generation. The starter system must be initialized and made operational by following the procedures described in Appendix E. Once initialized, the starter system meets the system data set requirements listed below and can be used as a generating system.

SYSTEM DATA SETS

The generating system must have the following system data sets:

- SYSCTLG (System Catalog)
- SYS1.NUCLEUS (Nucleus library)
- SYS1.SVCLIB (SVC library)
- SYS1.LOGREC
- SYS1.LINKLIB (Link library)
- SYS1.PROCLIB (Procedure library)
- SYS1.SYSJOBQE
- SYS1.MACLIB (Macro library)
- SYS1.MODLIB (Module library)
- SYS1.GENLIB (Generation library)

The link library (SYS1.LINKLIB) must include an assembler language processor (with the alias ASMBLR), a linkage editor (with the alias IEWL), and the IEBCOPY, IEHMOVE, IEHLIST, IEHIOSUP, and IFCDIP00 utilities. SYS1.MACLIB, SYS1.MODLIB, and SYS1.GENLIB must be cataloged in the generating system. SYS1.PROCLIB need not be cataloged unless the PROCLIB macro-instruction is used during system generation.

SYS1.GENLIB and SYS1.MODLIB must correspond to the release of the operating system to be generated. In certain cases, IEHIOSUP and IFCDIP00 must be of the same release as SYS1.GENLIB and SYS1.MODLIB.

This information is stated in the Prose material distributed with each release of the operating system.

If the E-design-level assembler language processor is to be used for system generation, both the macro library and the generation library must be unblocked. SYS1.MACLIB and SYS1.GENLIB can be unblocked by the IEBCOPY utility program by specifying DCB=(RECFM=FB,BLKSIZE=80,LRECL=80) in the SYSUT2 DD statement. (For a description of IEBCOPY and of the control statements it requires, refer to the publication IBM System/360 Operating System: Utilities.)

If the new operating system is to have ALGOL, COBOL, FORTRAN, PL/I, or sort/merge processors, the necessary processor modules must be included in the module library (SYS1.MODLIB). However, the generating system need not have the actual processor to generate it for the new system. For example, to generate a system that contains a FORTRAN compiler, the generating system need not have a FORTRAN compiler, but the FORTRAN compiler modules must be in SYS1.MODLIB.

If SYS1.COBLIB (for COBOL E), SYS1.FORTLIB, SYS1.PL1LIB, or SYS1.SORTLIB is to be included in the new system, that library must exist as a cataloged partitioned data set in the generating operating system. SYS1.ALGLIB, SYS1.COBLIB (for COBOL F), and SYS1.TELCMLIB need not exist as cataloged partitioned data sets in the generating system, but their modules must be included in SYS1.MODLIB. The modules for SYS1.FORTLIB must also be included in SYS1.MODLIB.

If the installation's operating system does not have SYS1.GENLIB and SYS1.MODLIB, they can be obtained from any generating system (either from the starter system or from any other operating system that meets these system data set requirements). If enough drives are available in the system, SYS1.GENLIB and SYS1.MODLIB need only be cataloged in the generating system using the IEHPROGM utility program. If there are not enough drives available, either SYS1.GENLIB or SYS1.MODLIB, or both, must be copied onto a generating system volume using the IEHMOVE utility program, and then cataloged. An example of obtaining SYS1.GENLIB and SYS1.MODLIB is in the section "Examples."

The arrangement and contents of the generating system data sets determine the performance of the system generation process. In general, any factors that contribute to faster assemblies and link edits will also contribute to faster system generations. For example, having resident supervisor functions such as BLDL and RAM; or having SYS1.LINKLIB reside on a volume other than the system residence volume.

UTILITY DATA SETS

In addition to the system data sets, four utility data sets must be allocated space and cataloged in the generating system for use during the system generation process. Each of these data sets must be cataloged as SYS1.name, where the value of name must not exceed eight alphameric characters, the first of which must be alphabetic. Three of these data sets must be sequential data sets; the fourth must be a partitioned data set. One of the sequential data sets and the partitioned data set must reside on a direct-access volume. The assembler, linkage editor, and utilities use the three sequential data sets during system generation. The partitioned data set is used for the storage of object modules assembled during system generation. (See "Input Deck for System Generation" in the section "Specifying the System" for further details on the utility data sets.)

USER-WRITTEN PROGRAMS

During the system generation process, the user can include his own routines in the nucleus, SVC library, and/or link library of the new operating system. The routines to be included in each of these libraries must be members of partitioned data sets cataloged in the generating system. The names under which they are cataloged are of the form SYS1.name. There must be only one partitioned data set for each of the libraries to be modified. However, if more than one library is to be modified, all routines can be included in one partitioned data set.

The user specifies the inclusion of his routines in the appropriate libraries by using the RESMODS, SVCTABLE, SVCLIB, and/or LINKLIB macro instructions. (See the section "Adding User-Written Functions.")

Before an operating system can be generated, a new system residence volume, and any other direct-access volumes required, must be initialized. The highest level of the system catalog must be built on the new system residence volume. Space must be allocated for the appropriate system data sets in the new operating system, and the appropriate data sets must be cataloged in the new system catalog. By definition, the system residence volume is that volume on which the nucleus library, the SVC library, the IPL program, the SYS1.LOGREC data set, and the volume index of the catalog are located.

Volume initialization is performed by the IBCDASDI independent utility program. The IEHPROGM system utility program is used to build the volume index of the system catalog, to allocate space for system data sets, and to catalog data sets. This system utility can be executed at any time after the system residence volume is initialized.

The following paragraphs describe the initialization of the system residence volume and other required direct-access volumes, the initialization of the system data sets, and considerations on allocating space for these data sets. Detailed descriptions of the utility programs and of the control statements they require are found in the publication IBM System/360 Operating System: Utilities.

INITIALIZING DIRECT-ACCESS VOLUMES

Initialization is the process of writing home addresses, a volume label, and a volume table of contents (VTOC) on direct-access volumes. In addition, the initial program load (IPL) program must be written on the direct-access volume that is to become the system residence volume. These functions are accomplished by the direct-access device initialization (IBCDASDI) utility program. This utility program is self-loading and operates independently of the operating system. In addition to its initialization functions, this utility program checks for defective tracks and, if any are found, assigns alternative tracks and issues appropriate messages. Alternative tracks are not accepted for track 0 of the system residence volume. (Track 0 is required for the IPL program.) If track 0 is found defective, another volume must be initialized for system residence.

The IPL program (IEAIPL00) is originally distributed with the starter operating system. Appendix E describes the procedure used to punch the IPL program cards from the distribution libraries. These cards can then be inserted in the IBCDASDI input deck whenever a system residence volume is initialized.

Figure 3 is an example of an input deck for system residence volume initialization. In this example, the volume to be initialized resides on an IBM 2311 Disk Storage Drive. The volume serial number to be written in the label is 111111. The volume table of contents (VTOC) starts at track 2, and is eight tracks long.

Note: If the direct-access volume is being initialized for the first time, the parameter FLAGTEST=NO should be included in the DADEF statement.

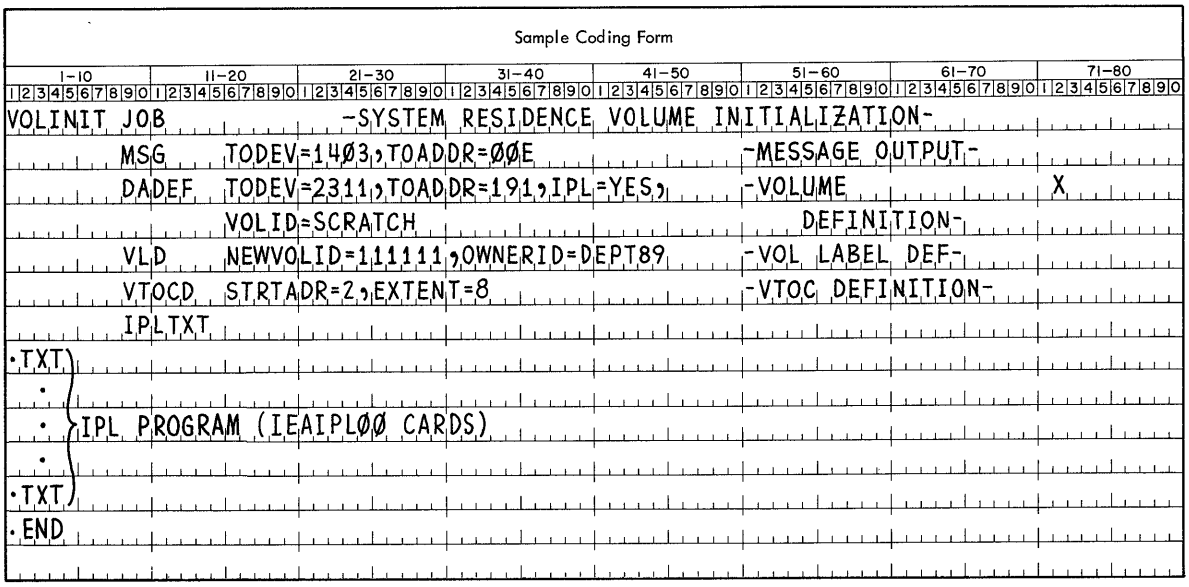


Figure 3. Initializing the System Residence Volume

INITIALIZING SYSTEM DATA SETS

The initialization of system data sets is the process of allocating space to the system data sets, building the volume index of the system catalog, and cataloging system data sets in the system catalog. The contents of the system libraries are placed in the allocated space during system generation. The contents of the other data sets are placed in the allocated space during job execution time in the generated operating system.

The volume index of the system catalog is built on the new system residence volume by the IEHPROGM system utility program. This index points to the system data sets that will form the new operating system. These data sets can be cataloged by the same utility program. Space should be allocated for these data sets (except for SYS1.LOGREC) by DD statements included in the input to this system utility program. The amount of space to be allocated to each system data set is shown in the publication IBM System/360 Operating System: Storage Estimates. The following sections describe the data sets for the new system, the input deck for initialization, and guidelines for the allocation of space to the system data sets.

DATA SETS FOR THE NEW SYSTEM

The following system data sets are required, and must have space allocated on the system residence volume. These data sets need not be cataloged. However, it is recommended that at least one of these data sets be cataloged to permit references to the system residence volume without knowing its serial number. (These references can be made by specifying VOLUME=REF=dsname in a DD statement, where dsname is the name of the cataloged data set.)

- SYSCTLG (System Catalog) -- Only the volume index of the system catalog need reside on the system residence volume.
- SYS1.NUCLEUS (Nucleus library)

- SYS1.SVCLIB (SVC library) -- It is recommended that this data set be cataloged because some cataloged procedures use its name to refer to the system residence volume.

- SYS1.LOGREC -- Space must not be allocated for this data set by the user. (SYS1.LOGREC is allocated space and initialized during the system generation process. Information on this data set is supplied in a message during Stage II of the system generation process. If SYS1.LOGREC must be reinitialized after system generation, refer to the description of the IFCDIP00 utility program in the publication IBM System/360 Operating System: Utilities.)

The following system data sets are required, and must have space allocated on a direct-access volume. They need not reside on the system residence volume.

- SYS1.LINKLIB (Link library) -- This data set must be cataloged.

- SYS1.PROCLIB (Procedure library) -- It is recommended that this data set be cataloged. It must be cataloged if any of the residency options or link pack area options are to be used. These options are described in the section "Residency Options and the Link Pack Area" of the publication IBM System/360 Operating System: System Programmer's Guide.

- SYS1.SYSJOBQE -- This data set need not be cataloged.

The operating system can function without the following system data sets. If the user wishes to make use of the optional facilities they provide, they can be included in the new operating system. Space must be allocated on a direct-access volume for the optional data sets desired. They need not reside on the system residence volume. It is recommended that the data sets to be included be cataloged.

- SYS1.MACLIB (Macro library)

- SYS1.SORTLIB (Sort library)

- SYS1.ALGLIB (ALGOL library)

- SYS1.COBLIB (COBOL library)

- SYS1.FORTLIB (FORTRAN library)

- SYS1.PL1LIB (PL/I library)

- SYS1.TELCMLIB (Telecommunications library)

- SYS1.SYSVLOGX and SYS1.SYSVLOGY (MVT system log data sets) -- These data sets must be cataloged

INPUT DECK FOR INITIALIZATION

The system data sets are allocated space and cataloged by the IEHPROGM utility program. Detailed descriptions of the control cards and functions of IEHPROGM can be found in the publication IBM System/360 Operating System: Utilities. The following text only describes the use of IEHPROGM for initializing the system data sets.

The input deck for IEHPROGM must contain the following:

- JOB statement with any parameters required by the installation.

- EXEC statement with the PGM=IEHPROGM parameter.
- A DD statement for the message output data set (SYSPRINT).
- A DD statement for each of the new system data sets (except for SYS1.LOGREC). These DD statements have the following format:

```
//ddname  DD  DSNAME=dsname,VOLUME=(,RETAIN,SER=serial),          X
//              UNIT=unit,LABEL=EXPDT=99350,SPACE=(allocation),   X
//              DISP=(,KEEP),DCB=(see Table 1)
```

- A DD * statement (SYSIN).
- A CATLG statement for each new system data set to be cataloged. Each CATLG statement must have the following format:

```
CATLG   DSNAME=dsname,CVOL=unit=serial,VOL=unit=serial
```

The DD statements and CATLG statements are described below. A section with examples of allocation is also included. Table 1 lists each system data set and summarizes allocation and cataloging requirements. For more information on the coding of these parameters refer to the publication IBM System/360 Operating System: Job Control Language.

Table 1. System Data Sets

System Data Set	Required	System Residence	Secondary Allocation Allowed	DCB Subparameters	Cataloged
SYSCTLG	Yes	Yes	Yes	N/A	N/A
SYS1.NUCLEUS	Yes	Yes	No	N/A	Optional
SYS1.SVCLIB	Yes	Yes	No	DSORG=POU,RECFM=U,BLKSIZE=1024	Recommended
SYS1.LOGREC[1]	Yes	Yes	N/A	N/A	N/A
SYS1.LINKLIB	Yes	Optional	No	RECFM=U,BLKSIZE=3625[2]	Yes
SYS1.PROCLIB	Yes	Optional	Yes	RECFM=F,BLKSIZE=80	Recommended
SYS1.SYSJOBQE	Yes	Optional	No	N/A	Optional
SYS1.MACLIB	No	Optional	Yes	RECFM=FB,LRECL=80,BLKSIZE=3360[3]	Recommended
SYS1.SORTLIB	No	Optional	Yes	RECFM=U,BLKSIZE=3625[2]	Recommended
SYS1.ALGLIB	No	Optional	Yes	RECFM=U,BLKSIZE=3625[2]	Recommended
SYS1.COBLIB	No	Optional	Yes	RECFM=U,BLKSIZE=3625[2]	Recommended
SYS1.FORTLIB	No	Optional	Yes	RECFM=U,BLKSIZE=3625[2]	Recommended
SYS1.PL1LIB	No	Optional	Yes	RECFM=U,BLKSIZE=3625[2]	Recommended
SYS1.TELCMLIB	No	Optional	Yes	RECFM=U,BLKSIZE=3625[2]	Recommended
SYS1.SYSVLOGX	No	Optional	No	N/A	Yes
SYS1.SYSVLOGY	No	Optional	No	N/A	Yes

[1]Space must not be allocated for SYS1.LOGREC by the user.
[2]BLKSIZE=3625 applies if the system data set resides on a 2311. BLKSIZE=20843 must be specified for 2301, BLKSIZE=4892 for 2303, and BLKSIZE=7294 for 2314.
[3]The value of BLKSIZE for SYS1.MACLIB must be a multiple of 80 which is less than, or equal to, 3600 for a 2311, 20480 for a 2301, 4880 for a 2303, or 7280 for a 2314. If the E-design-level assembler language processor is to be included in the new system, BLKSIZE=80 must be specified.

DD Statements

The DD statements in the input deck for initializing the system data sets have the following parameters:

ddname
name of the DD statement.

DSNAME=dsname
name of the system data set. There must be a DD statement for each required system data set (except for SYS1.LOGREC), and for each optional system data set selected.

SYS1.SYSJOBQE, SYS1.SYSVLOGX, and SYS1.SYSVLOGY are not affected by system generation. Therefore, they need not be allocated space and cataloged until before the initial program load (IPL) of the new system. For convenience, it is recommended that they be allocated space and cataloged together with the other system data sets.

During the preparation for processor and libraries generation only those optional system data sets that are to be added to the system must be cataloged and allocated space.

VOLUME=(,RETAIN,SER=serial)
serial number of the direct-access volume on which the system data set is to reside. The serial number of the new system residence volume must be specified for SYSCTLG, SYS1.NUCLEUS, and SYS1.SVCLIB. The volume indicated with this parameter determines the corresponding volume specification for the system data set in the system generation macro-instructions. (See the descriptions of the GENERATE, PROCLIB, TELCMLIB, MACLIB, ALGLIB, SORTLIB, COBLIB, FORTLIB, and PL1LIB macro-instructions in "Specifying the System.") The section "Location of System Data Sets" provides guidelines for the allocation of system data sets on available volumes.

UNIT=unit
the name of the direct-access device that can be allocated to the system data set. It is recommended that the value given to UNIT be a generic unit name. (Generic unit names are listed in Appendix C.)

LABEL=EXPDT=99350
the expiration date for all data sets (except for SYS1.SYSJOBQE) is chosen to prevent accidental deletion. The DD statement for SYS1.SYSJOBQE should not have an expiration date because this data set must have an expiration date lower than the date entered during IPL for the new system.

The data set protection provided by the expiration date requires additional action by the user. If the current date is set in the generating system during system generation, the operator is required to override this current date each time a protected data set is opened. Another, more convenient method can be used. For the system generation job (or any time the data sets are to be modified), the current date may be set at a higher value than the expiration date specified for the protected data sets. (In this case the set date becomes the new expiration date.) This may be done by the operator from the console, or by a card in the job stream. In either case, the current date should be reset in the generating system immediately after the completion of the system generation process.

SPACE

the amount of auxiliary storage to be allocated to the system data set can be obtained from the publication IBM System/360 Operating System: Storage Estimates. The directory quantities for the system libraries are also shown in that publication.

The user must also allow space for the inclusion of his own programs into the system data set during (or after) system generation. Some system data sets, such as SYS1.LINKLIB, may not have a secondary allocation and must be contiguous. Thus, sufficient space must be initially allocated to them if user modules are to be included. Table 1 indicates which system data sets may not have a secondary allocation. To insure that these data sets are also contiguous, the SPACE parameter may be coded as follows:

SPACE=(units(quantities),,CONTIG)

Only the volume index of SYSCTLG must reside on the system residence volume. SYS1.NUCLEUS and SYS1.SVCLIB must be allocated space entirely on the system residence volume. SYS1.SVCLIB may not occupy more than 1023 tracks on the system residence volume. (The size of SYS1.LOGREC is determined during system generation and is allocated on the system residence volume.) The maximum space that can be allocated to the remaining system data sets is one volume, except for SYS1.SYSJOBQE which may not occupy more than 1250 tracks on a 2314. Alternative track assignment is accepted for the system data sets. However, to achieve maximum efficiency in the new system, alternative tracks should not be used for SYS1.SVCLIB, SYS1.LINKLIB, and SYS1.SYSJOBQE.

Space must be allocated for SYS1.PROCLIB in the new system. If neither IBM-supplied nor user-written cataloged procedures are to be used with PCP or MFT, a null allocation can be made for SYS1.PROCLIB; i.e., the SPACE parameter is specified as SPACE=(TRK,(0)). A null allocation is not permitted if MVT is used in the new system, because a minimum of three cataloged procedures are required for MVT. (The three cataloged procedures are named IEEVMPCR, IEEVRPCR, and IEEVWPCR.)

Unless sufficient space is allocated to the system data sets, the system generation process cannot be completed successfully.

DISP=(,KEEP)

this parameter must be coded as shown. DISP=(,CATLG) cannot be used because the data set would be cataloged in the generating system rather than in the catalog of the new system residence volume.

DCB

certain system data sets require a DCB parameter. The values to be given to this parameter are shown in Table 1 for the appropriate system data sets.

CATLG Statements

The CATLG statements in the input deck for initializing system data sets have the following parameters:

DSNAME=dsname

the name of the system data set to be cataloged. Table 1 shows whether a system data set should be cataloged.

CVOL=unit=serial
specifies the unit name and serial number of the new system residence volume. (The values must be the same as specified in the DD statement for SYS1.NUCLEUS.)

VOL=unit=serial
the unit name and serial number of the volume on which the system data set resides. These values must be the same as specified in the corresponding DD statement for the system data set.

Sample Data Set Initializations

This section contains two examples of initialization of system data sets. In the first example, all the system data sets required by a new system reside on one volume. In the second example, the system data sets of another new system are arranged on two volumes. The numbers chosen for space allocation in these two examples are for illustrative purposes only. Space requirements for the system data sets are determined by several factors, especially the type of device used and the characteristics of the system to be generated. For example, the processors chosen for the new system affect the size of SYS1.LINKLIB. Exact auxiliary storage requirements on various types of direct-access devices can be obtained from the publication IBM System/360 Operating System: Storage Estimates.

Figure 4 is an example of an input deck for building the system catalog and for allocating space to the system data sets on one volume. It is assumed that the system residence volume was initialized as shown in Figure 3. The new system requires SYS1.MACLIB, SYS1.SORTLIB, SYS1.COBLIB, and SYS1.FORTLIB. These optional data sets, and SYS1.SVCLIB, SYS1.PROCLIB and SYS1.LINKLIB are to be cataloged in the new system. The unit for the new system residence volume is a 2311. The serial number of the system residence volume is 111111.

Figure 5 is another example of an input deck for building the system catalog and for allocating space to the system data sets. The system data sets are to reside on two volumes. The unit for the system residence volume is a 2301 and its serial number is AAA111. The unit for the second volume is a 2311 and its serial number is AAA112. It is assumed that both volumes were previously initialized. The new system requires all the optional system data sets. These optional data sets, SYS1.SVCLIB, SYS1.PROCLIB, and SYS1.LINKLIB, are to be cataloged in the new system. All system data sets, except SYS1.LINKLIB, SYS1.MACLIB, SYS1.SYSVLOGX, and SYS1.SYSVLOGY, reside on the system residence volume. SYS1.LINKLIB, SYS1.MACLIB, SYS1.SYSVLOGX, and SYS1.SYSVLOGY reside on the second volume (AAA112).

LOCATION OF SYSTEM DATA SETS

This section provides guidelines for allocating space to the data sets required by the new system, and to the four utility data sets required during system generation. (Two of the utility data sets must reside on direct-access volumes; the other two may reside on either magnetic tape or direct-access volumes.) The location of the new system data sets is determined by:

- The size of the system data sets (given in the publication IBM System/360 Operating System: Storage Estimates).
- The size of the utility data sets required for system generation (given in the section "Input Deck for System Generation.")
- The machine configuration, in particular the number and type of devices available in the generating system.

Sample Coding Form

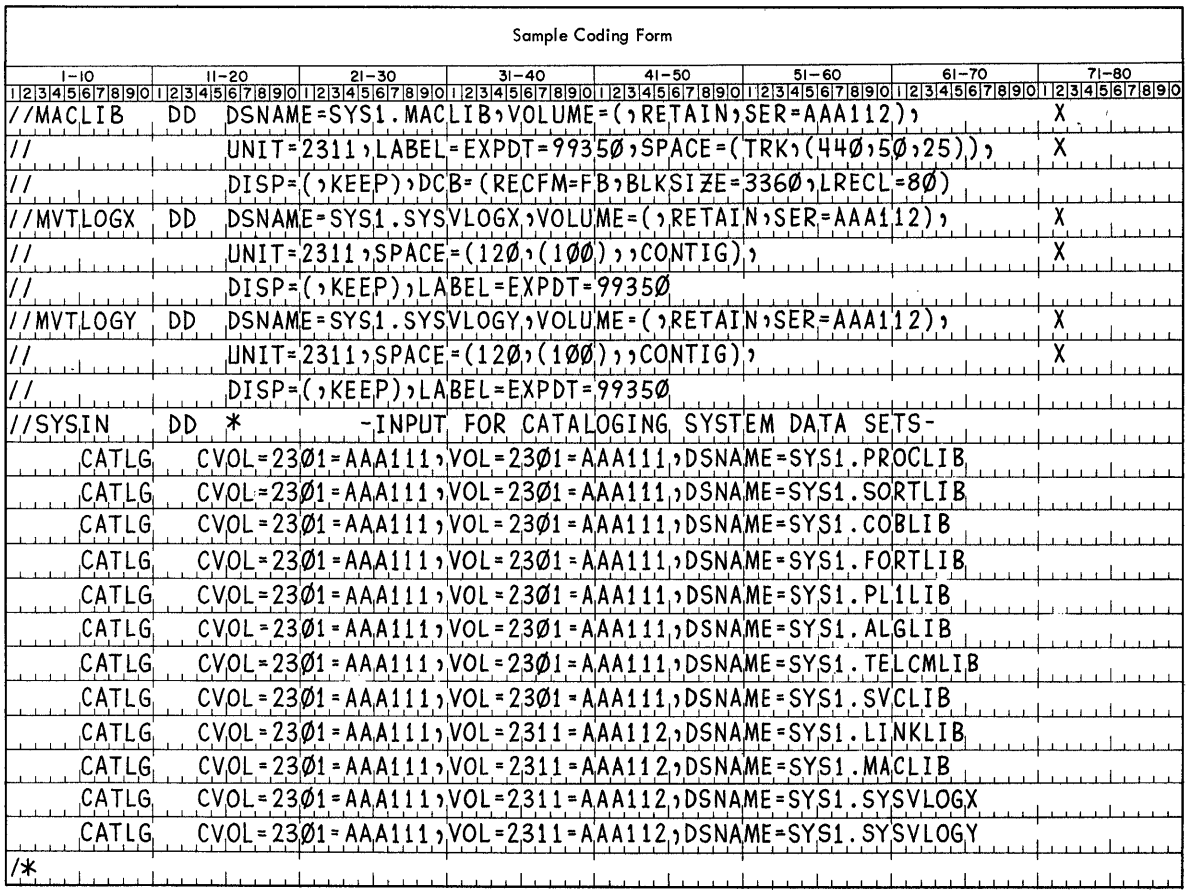
```
1-10      11-20     21-30     31-40     41-50     51-60     61-70     71-80
//SYSGEN   JOB  MSGLEVEL=1       -ALLOCATE ON ONE VOLUME-
//STEPØ    EXEC PGM=IEHPROGM
//SYSPRINT DD   SYSOUT=A
//JOBQE    DD   DSNAME=SYS1.SYSJOBQE,VOLUME=(,RETAIN,SER=111111),                X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(12Ø),,CONTIG)
//SVCLIB   DD   DSNAME=SYS1.SVCLIB,VOLUME=(,RETAIN,SER=111111),                 X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(15Ø,,75),,CONTIG),            X
//             LABEL=EXPDT=99350,DCB=(DSORG=POU,RECFM=U,BLKSIZE=1Ø24)
//LINKLIB  DD   DSNAME=SYS1.LINKLIB,VOLUME=(,RETAIN,SER=111111),                X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(97Ø,,1ØØ),,CONTIG),           X
//             LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=3625)
//CATALOG  DD   DSNAME=SYSCTLG,VOLUME=(,RETAIN,SER=111111),                     X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(1Ø,1)),                       X
//             LABEL=EXPDT=99350
//PROCLIB  DD   DSNAME=SYS1.PROCLIB,VOLUME=(,RETAIN,SER=111111),                X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(3Ø,1Ø,9)),                    X
//             LABEL=EXPDT=99350,DCB=(RECFM=F,BLKSIZE=8Ø)
//SORTLIB  DD   DSNAME=SYS1.SORTLIB,VOLUME=(,RETAIN,SER=111111),                X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(6Ø,2,4Ø)),                    X
//             LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=3625)
//COBLIB   DD   DSNAME=SYS1.COBLIB,VOLUME=(,RETAIN,SER=111111),                 X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(35,2,3Ø)),                    X
//             LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=3625)
//FORTLIB  DD   DSNAME=SYS1.FORTLIB,VOLUME=(,RETAIN,SER=111111),                X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(5Ø,2,4Ø)),                    X
//             LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=3625)
//MACLIB   DD   DSNAME=SYS1.MACLIB,VOLUME=(,RETAIN,SER=111111),                 X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(46Ø,3Ø,25)),                  X
//             LABEL=EXPDT=99350,DCB=(RECFM=FB,BLKSIZE=336Ø,LRECL=8Ø)
//NUCLEUS  DD   DSNAME=SYS1.NUCLEUS,VOLUME=(,RETAIN,SER=111111),                X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(3Ø,,1),,CONTIG),              X
//             LABEL=EXPDT=99350
//SYSIN    DD   *          -INPUT FOR CATALOGING SYSTEM DATA SETS-
     CATLG   CVOL=2311=111111,VOL=2311=111111,DSNAME=SYS1.LINKLIB
     CATLG   CVOL=2311=111111,VOL=2311=111111,DSNAME=SYS1.PROCLIB
     CATLG   CVOL=2311=111111,VOL=2311=111111,DSNAME=SYS1.MACLIB
     CATLG   CVOL=2311=111111,VOL=2311=111111,DSNAME=SYS1.SORTLIB
     CATLG   CVOL=2311=111111,VOL=2311=111111,DSNAME=SYS1.COBLIB
     CATLG   CVOL=2311=111111,VOL=2311=111111,DSNAME=SYS1.FORTLIB
     CATLG   CVOL=2311=111111,VOL=2311=111111,DSNAME=SYS1.SVCLIB
/*
```

•Figure 4. Initializing the System Data Sets -- One Volume Residence

Sample Coding Form

```
1-10      11-20     21-30     31-40     41-50     51-60     61-70     71-80
//SYSGEN  JOB  MSGLEVEL=1        -ALLOCATE ON TWO VOLUMES-
//ONE     EXEC PGM=IEHPROGM
//SYSPRINT DD  SYSOUT=A
//JOBQE   DD   DSNAME=SYS1.SYSJOBQE,VOLUME=(,RETAIN,SER=AAA111),                X
//            UNIT=2301,DISP=(,KEEP),SPACE=(TRK,(30),,CONTIG)
//SVCLIB  DD   DSNAME=SYS1.SVCLIB,VOLUME=(,RETAIN,SER=AAA111),                 X
//            UNIT=2301,DISP=(,KEEP),SPACE=(TRK,(40,,75),,CONTIG),             X
//            LABEL=EXPDT=99350,DCB=(DSORG=POU,RECFM=U,BLKSIZE=1024)
//CATALOG DD   DSNAME=SYSCTLG,VOLUME=(,RETAIN,SER=AAA111),                     X
//            UNIT=2301,LABEL=EXPDT=99350,SPACE=(TRK,(2,1)),                   X
/           DISP=(,KEEP)
//PROCLIB DD   DSNAME=SYS1.PROCLIB,VOLUME=(,RETAIN,SER=AAA111),                X
//            UNIT=2301,LABEL=EXPDT=99350,SPACE=(TRK,(8,2,9)),                 X
//            DISP=(,KEEP),DCB=(RECFM=F,BLKSIZE=80)
//SORTLIB DD   DSNAME=SYS1.SORTLIB,VOLUME=(,RETAIN,SER=AAA111),                X
//            UNIT=2301,LABEL=EXPDT=99350,SPACE=(TRK,(15,2,40)),               X
//            DISP=(,KEEP),DCB=(RECFM=U,BLKSIZE=20843)
//COBLIB  DD   DSNAME=SYS1.COBLIB,VOLUME=(,RETAIN,SER=AAA111),                 X
//            UNIT=2301,LABEL=EXPDT=99350,SPACE=(TRK,(10,1,30)),               X
//            DISP=(,KEEP),DCB=(RECFM=U,BLKSIZE=20843)
//FORTLIB DD   DSNAME=SYS1.FORTLIB,VOLUME=(,RETAIN,SER=AAA111),                X
//            UNIT=2301,LABEL=EXPDT=99350,SPACE=(TRK,(12,2,40)),               X
//            DISP=(,KEEP),DCB=(RECFM=U,BLKSIZE=20843)
//PL1LIB  DD   DSNAME=SYS1.PL1LIB,VOLUME=(,RETAIN,SER=AAA111),                 X
//            UNIT=2301,LABEL=EXPDT=99350,SPACE=(TRK,(20,10,65)),              X
//            DISP=(,KEEP),DCB=(RECFM=U,BLKSIZE=20843)
//ALGLIB  DD   DSNAME=SYS1.ALGLIB,VOLUME=(,RETAIN,SER=AAA111),                 X
//            UNIT=2301,LABEL=EXPDT=99350,SPACE=(TRK,(10,2,15)),               X
//            DISP=(,KEEP),DCB=(RECFM=U,BLKSIZE=20843)
//TELCLIB DD   DSNAME=SYS1.TELCMLIB,VOLUME=(,RETAIN,SER=AAA111),               X
//            UNIT=2301,LABEL=EXPDT=99350,SPACE=(TRK,(10,1,10)),               X
//            DISP=(,KEEP),DCB=(RECFM=U,BLKSIZE=20843)
//NUCLEUS DD   DSNAME=SYS1.NUCLEUS,VOLUME=(,RETAIN,SER=AAA111),                X
//            UNIT=2301,SPACE=(TRK,(10,,1),,CONTIG),                           X
//            DISP=(,KEEP),LABEL=EXPDT=99350
//LINKLIB DD   DSNAME=SYS1.LINKLIB,VOLUME=(,RETAIN,SER=AAA112),                X
//            UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(1250),,100),,CONTIG),         X
//            LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=3625)
```

Figure 5. Initializing the System Data Sets -- Two Volumes Residence (Part 1 of 2)



```
Sample Coding Form
1-10  11-20  21-30  31-40  41-50  51-60  61-70  71-80
//MACLIB   DD  DSNAME=SYS1.MACLIB,VOLUME=(,RETAIN,SER=AAA112),                X
//           UNIT=2311,LABEL=EXPDT=99350,SPACE=(TRK,(440,50,25)),             X
//           DISP=(,KEEP),DCB=(RECFM=FB,BLKSIZE=3360,LRECL=80)
//MVTLOGX  DD  DSNAME=SYS1.SYSVLOGX,VOLUME=(,RETAIN,SER=AAA112),              X
//           UNIT=2311,SPACE=(120,(100),,CONTIG),                             X
//           DISP=(,KEEP),LABEL=EXPDT=99350
//MVTLOGY  DD  DSNAME=SYS1.SYSVLOGY,VOLUME=(,RETAIN,SER=AAA112),              X
//           UNIT=2311,SPACE=(120,(100),,CONTIG),                             X
//           DISP=(,KEEP),LABEL=EXPDT=99350
//SYSIN    DD  *          -INPUT FOR CATALOGING SYSTEM DATA SETS-
     CATLG   CVOL=2301=AAA111,VOL=2301=AAA111,DSNAME=SYS1.PROCLIB
     CATLG   CVOL=2301=AAA111,VOL=2301=AAA111,DSNAME=SYS1.SORTLIB
     CATLG   CVOL=2301=AAA111,VOL=2301=AAA111,DSNAME=SYS1.COBLIB
     CATLG   CVOL=2301=AAA111,VOL=2301=AAA111,DSNAME=SYS1.FORTLIB
     CATLG   CVOL=2301=AAA111,VOL=2301=AAA111,DSNAME=SYS1.PL1LIB
     CATLG   CVOL=2301=AAA111,VOL=2301=AAA111,DSNAME=SYS1.ALGLIB
     CATLG   CVOL=2301=AAA111,VOL=2301=AAA111,DSNAME=SYS1.TELCMLIB
     CATLG   CVOL=2301=AAA111,VOL=2301=AAA111,DSNAME=SYS1.SVCLIB
     CATLG   CVOL=2301=AAA111,VOL=2311=AAA112,DSNAME=SYS1.LINKLIB
     CATLG   CVOL=2301=AAA111,VOL=2311=AAA112,DSNAME=SYS1.MACLIB
     CATLG   CVOL=2301=AAA111,VOL=2311=AAA112,DSNAME=SYS1.SYSVLOGX
     CATLG   CVOL=2301=AAA111,VOL=2311=AAA112,DSNAME=SYS1.SYSVLOGY
/*
```

Figure 5. Initializing the System Data Sets -- Two Volumes Residence (Part 2 of 2)

- The use of the generating system data sets, new system data sets, and utility data sets during system generation. (The generating system data sets and utility data sets are described in the section "System/360 Requirements.")

The following rule may be used to allocate the system data sets on the available volumes:

- During Stage I and the beginning (assembly steps) of Stage II of system generation, the volumes that contain the utility data sets for system generation, and the generating system data sets, except for SYS1.MODLIB, must be mounted at the same time.

- After the assembly steps of Stage II, SYS1.GENLIB and two of the utility data sets (those that need not reside on direct-access volumes) are no longer needed and may be removed. SYS1.MACLIB of the generating system may also be removed at this point if the MACLIB macro-instruction is not used during system generation. (See the description of the MACLIB macro-instruction in the section "Specifying the System.")

- The volumes that contain SYS1.MODLIB, the new system data sets, and the volumes that contain user-written modules to be added to the new system must be mounted after the Stage II assemblies.

If there are enough drives available, all generating system data sets, new system data sets, and system generation utility data sets should be mounted at the same time. If there are not enough drives, the volumes that contain data sets not required for a given step must be

dismounted and the volumes that contain the data sets required for that step must be mounted. The scheduler indicates to the operator which volume should be mounted.

The new system data sets can be arranged in different ways on one or more volumes. For best performance, it is desirable to place the system data sets on more than one volume. Whenever possible, frequently used data sets should be located on a volume other than the system residence volume. System data sets on the same volume should be arranged according to the interaction between them: the more interaction, the closer they should be placed. For example, for best performance SYS1.LINKLIB should be placed on a volume other than the system residence volume. However, if it must be on the system residence volume, it should be adjacent to SYS1.SVCLIB because SYS1.LINKLIB and SYS1.SVCLIB are frequently used data sets and are closely related. In this same example, space for SYS1.NUCLEUS should be allocated last, because this data set is referred to only by the IPL program. (Data sets are allocated in the order in which they are defined by DD statements in the input deck.)

If there are not enough drives available to achieve the desired distribution of new system data sets, the IEHMOVE utility program can be used to distribute the data sets after system generation. If SYS1.SVCLIB is changed, replaced, or moved after system generation, the IEHIOSUP utility program must be executed. If the MACLIB macro instruction was not used during system generation, the IEBCOPY utility program can be used to copy SYS1.MACLIB from the generating system to the new system.

The following pages contain examples of allocation on configurations with two, three, four, and five direct-access devices. In all these examples, the four utility data sets required for system generation are called SYS1.UT1, SYS1.UT2, SYS1.UT3, and SYS1.OBJMOD. SYS1.UT1 and SYS1.UT2 may reside on magnetic tape.

Generation on Two Drives

Figure 6 shows the distribution of data sets using two 2311 Disk Storage Drives and three volumes. The generating system data sets reside on volumes #1 and #2. All new system data sets reside on volume #3. SYS1.UT1 and SYS1.UT2 reside on magnetic tape. The scheduler, when required, automatically requests the dismounting of volume #2 and the mounting of volume #3 on drive #2. The generating system is a customized version (see Appendix E) of the starter operating system.

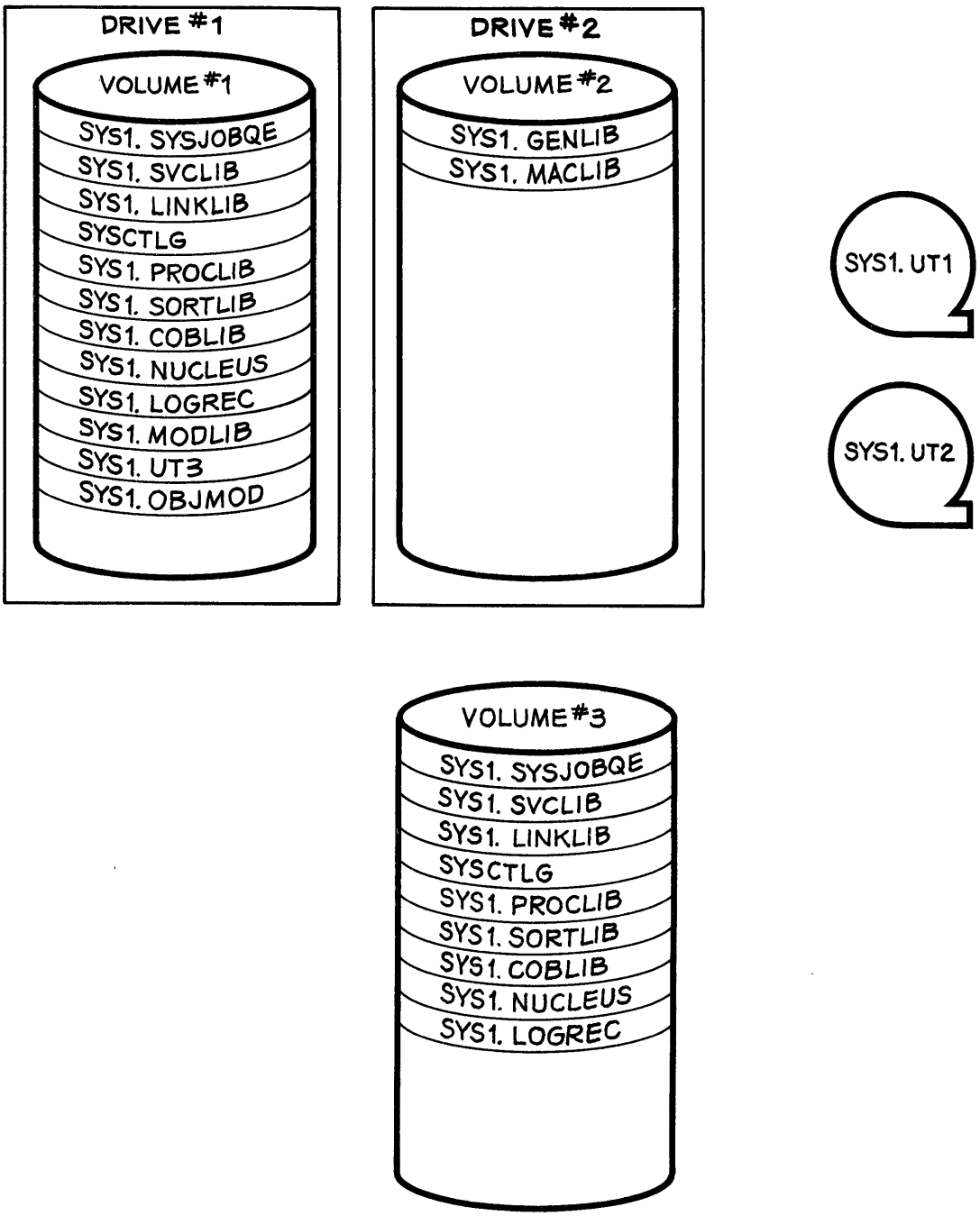


Figure 6. Generation on Two Drives

Generation on Three Drives

Figure 7 shows the distribution of data sets using three 2311 Disk Storage Drives and four volumes. The generating system data sets reside on volumes #1, #2, and 3. All new system data sets reside on volume #4. The scheduler, when required, requests the dismounting of volume #2 and the mounting of volume #4 on drive #2. Note that volume #2 contains all the generating system and utility data sets that are no longer needed after the Stage II assemblies.

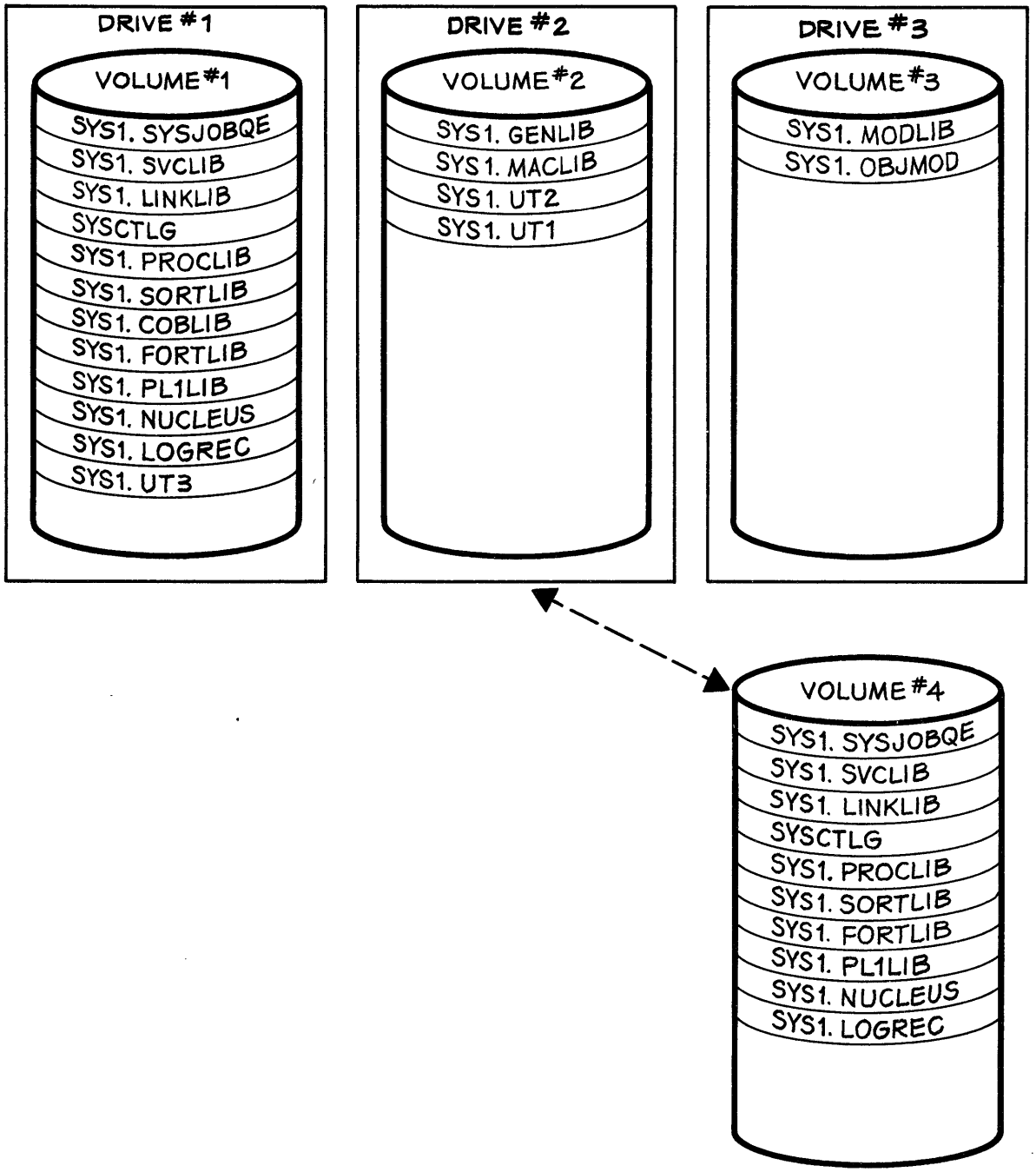


Figure 7. Generation on Three Drives

Generation on Four Drives

Figure 8 shows the distribution of data sets using four 2311 Disk Storage Drives and four volumes. All new system data sets reside on volume #4. No dismounting and mounting of volumes is required during this generation.

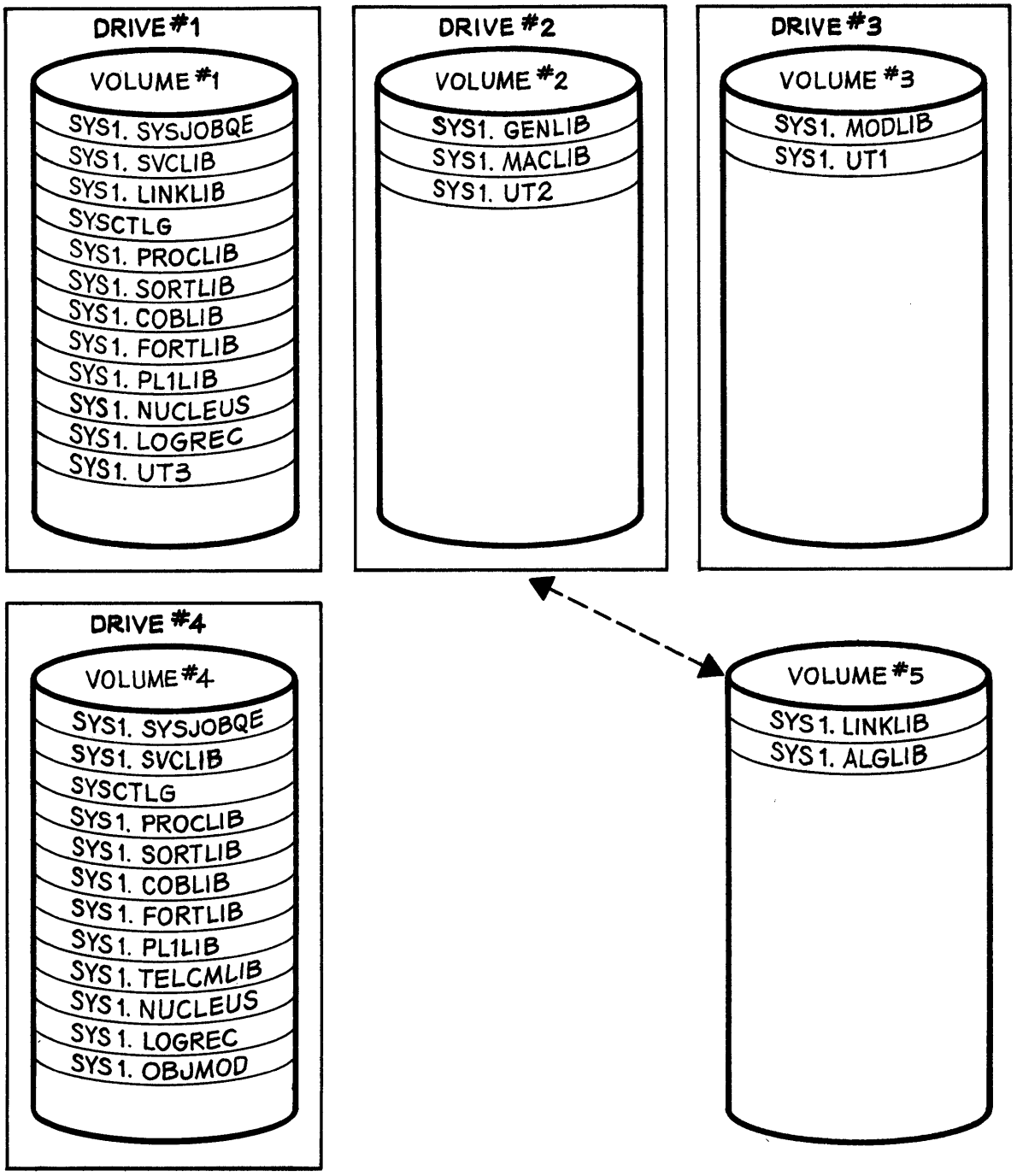


Figure 8. Generation on Four Drives

Generation on Four Drives With Two Volumes for New System

Figure 9 shows the distribution of data sets using four 2311 Disk Storage Drives and five volumes. The generated SYS1.LINKLIB and SYS1.ALGLIB are to be placed on the volume marked #5. All other new system data sets are to be placed in volume #4. The scheduler, when required, requests the dismounting of volume #2 and the mounting of volume #5 on drive #2.

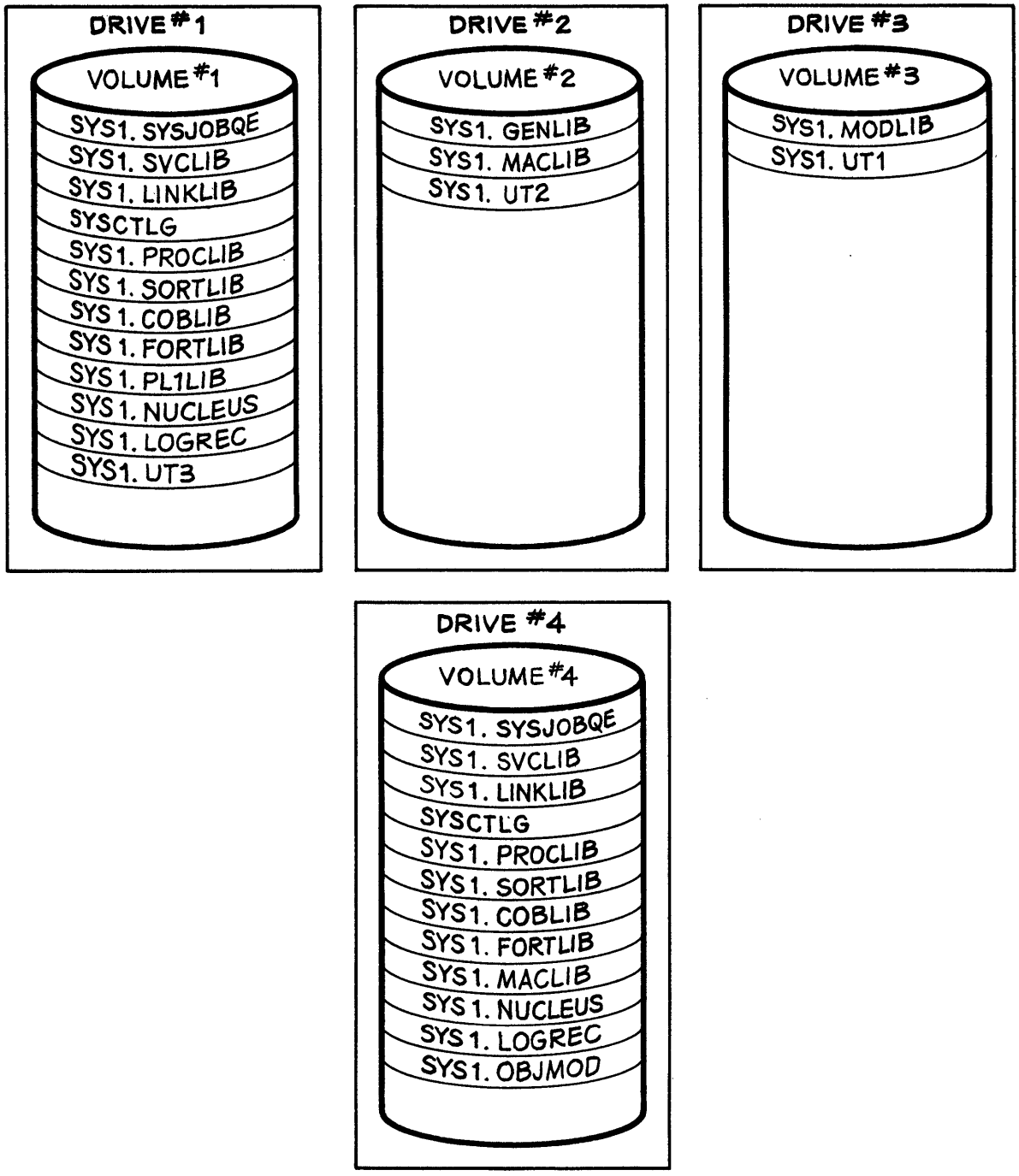


Figure 9. Generation on Four Drives With Multiple Volumes

Generation on Five Drives With Two Volumes for New System

Figure 10 shows the distribution of data sets using four 2311 Disk Storage Drives, one 2301 drum storage drive, and five volumes. Volume #5 is the new system residence volume. SYS1.MACLIB resides on volume #5. All other new system data sets not required to be on the system residence volume are located on volume #4.

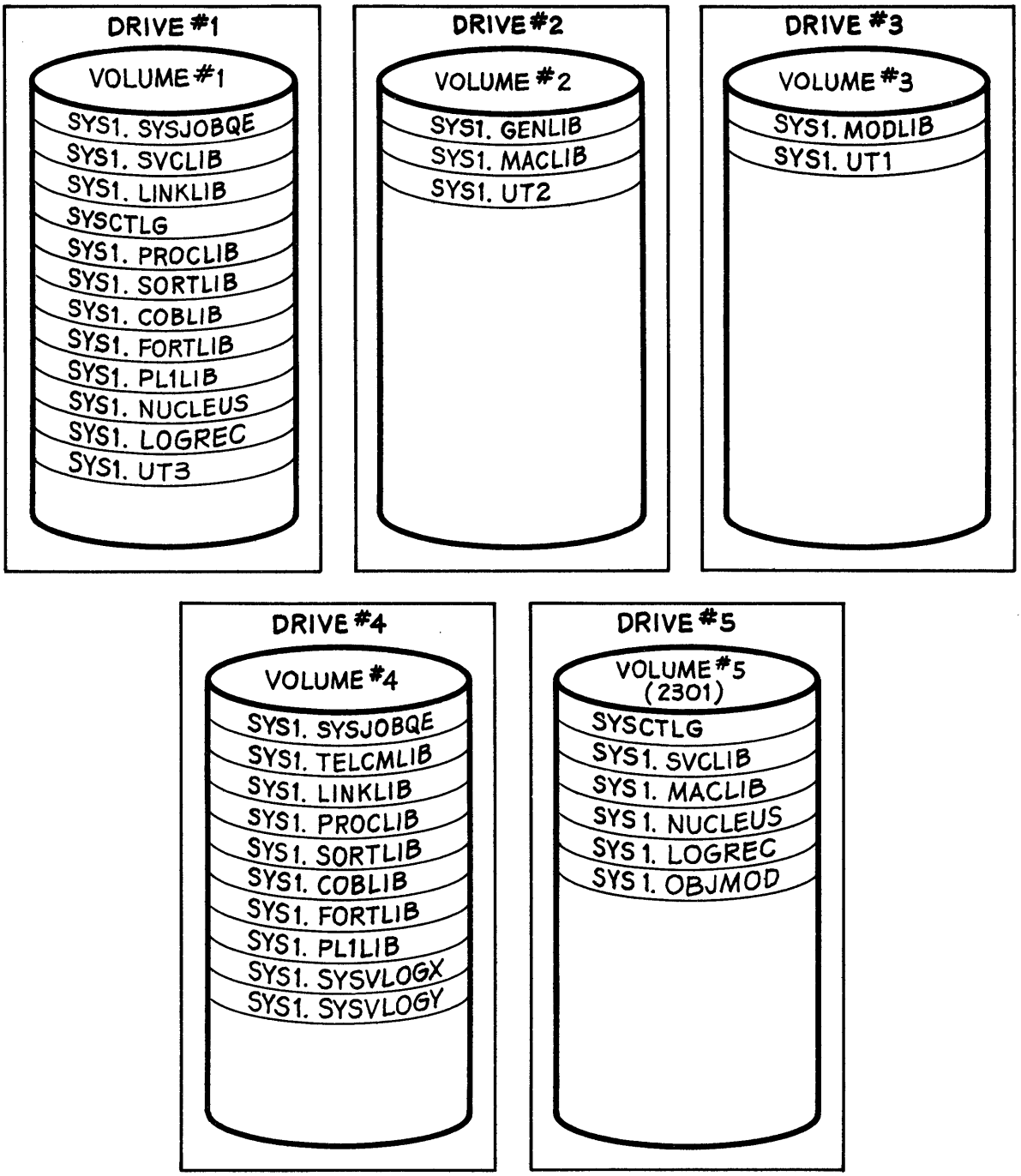


Figure 10. Generation on Four Drives With Multiple Volumes

Generation on Four 2314 Drives

Figure 11 shows the distribution of data sets using four 2314 Direct-Access modules. All generating system data sets reside on volume #1. Volume #2 is the new system residence volume. The new SYS1. LINKLIB, SYS1.MACLIB, and SYS1.ALGLIB are to be placed on volume #3.

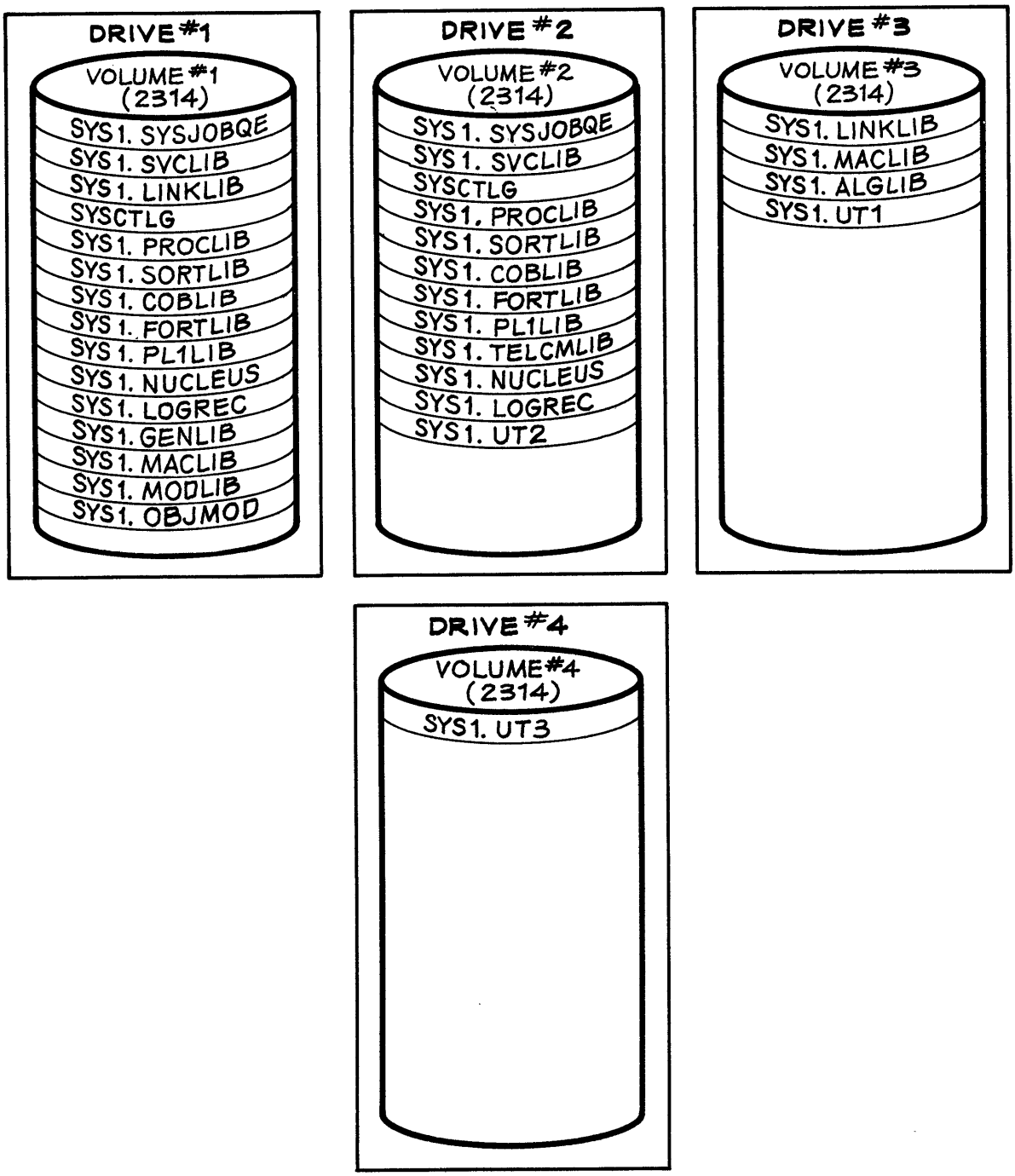


Figure 11. Generation on Four 2314 Volumes

SPECIFYING THE SYSTEM

The system to be generated is specified by the user. Specifications are of two types: job control language statements and system generation macro-instructions. The job control language statements are those required to perform an assembly. The system generation macro-instructions describe the machine configuration of the installation, those options the user desires, and the type of generation to be performed. The programs and routines generated as a result of these specifications are placed into the appropriate operating system libraries during the system generation process.

The following sections describe the input deck required for system generation, and the conventions used to code system generation macro-instructions. A detailed description of each system generation macro-instruction and a table showing their relationship to each type of system generation are also included.

From the specifications supplied to the system generation macro-instructions, parameters are created for the job control language statements produced from Stage I and included in the job stream to Stage II. Samples of these statements are shown in the section "Restart Procedures."

Note: The specifications for a system generation must be compatible with the cataloging and space allocation performed during the preparation for that generation.

INPUT DECK FOR SYSTEM GENERATION

The input deck required for Stage I of system generation consists of job control language statements and system generation macro-instructions. The sequence of the deck and the job control language statements are shown in Figure 12. This figure represents a three-drive system generation. For other generations, only the underlined values need vary. (In Figure 12, the generating system resides on a volume whose serial number is DLIB01; SYS1.MACLIB and SYS1.GENLIB reside on a volume whose serial number is DLIB02; SYS1.MODLIB resides on a volume whose serial number is DLIB03; and unit 182 is a 9-track magnetic tape.) It is recommended that the values given to the UNIT keywords of the DD statements be generic unit names. (See Appendix C.)

The first statement of the deck is an EXEC statement indicating that the generation process immediately follows the catalog building step described in the previous section. Alternatively, system generation can be defined as an independent job.

The four DD statements named OBJPDS, SYSUT1, SYSUT2, and SYSUT3, respectively, are used to allocate space to the four utility data sets required for the system generation process. These data sets are cataloged as SYS1.name in the generating system, where the value of name cannot exceed eight alphameric characters, the first of which must be alphabetic. These names (SYS1.name) must also be specified as the value of the corresponding keywords (OBJPDS, UT1SDS, UT2SDS, and UT3SDS) in the GENERATE macro-instruction. The data set defined by the OBJPDS DD statement must be a partitioned data set. The other three data sets must be sequential data sets, of which the one specified by the SYSUT3 DD statement must reside on a direct-access volume. Table 2 shows the values to be given to the SPACE keyword of each of the DD statements for the utility data sets according to the type of direct-access device on

which they may reside. To determine if there is enough space available in the direct-access volume, list its volume table of contents (VTOC) before Stage I using the IEHLIST utility program.

```
//STEP1     EXEC PGM=ASMBLR        -STAGE I INPUT DECK-
//SYSLIB    DD  DSNAME=SYS1.GENLIB,DISP=OLD
//OBJPDS    DD  DSNAME=SYS1.OBJMOD,VOLUME=(,RETAIN,SER=DLIB01),        X
//              DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(40,20,8))
//SYSUT1    DD  DSNAME=SYS1.UT1,VOLUME=(,RETAIN,SER=DLIB01),           X
//              DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(240,20))
//SYSUT2    DD  DSNAME=SYS1.UT2,VOLUME=(,RETAIN,SER=DLIB02),           X
//              DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(240,20))
//SYSUT3    DD  DSNAME=SYS1.UT3,VOLUME=(,RETAIN,SER=DLIB03),           X
//              DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(300,20))
//DUMMY     DD  VOLUME=(,RETAIN,REF=*.SYSUT3),SPACE=(TRK,(80))
//SYSPUNCH DD   UNIT=182,LABEL=(,NL)
//SYSPRINT DD   SYSOUT=A
//SYSIN     DD  *
            .
            .
            .   System generation macro-instructions
            .
            .
            END
/*
//          START  RDR,182  (Optional statement; see text)
```

Note: Underlined values represent variables. All other values must be coded as shown. The continuation characters are in column 72.

Figure 12. Input Deck Organization for System Generation

If magnetic tape drives are available, the data sets specified by the SYSUT1 and SYSUT2 DD statements can be assigned to 9-track magnetic tape. If only one utility data set is to reside on magnetic tape, the one specified by the SYSUT1 DD statement should be chosen. If the data sets defined by the SYSUT1 or SYSUT2 DD statements reside on unlabeled magnetic tape, LABEL=(,NL) must be specified in the corresponding DD statement, and NL must also be specified in the UT1SDS or UT2SDS keyword of the GENERATE macro-instruction.

The DD statement named DUMMY is used to reserve space on the volume that contains the utility data set defined by the SYSUT3 DD statement for the IEHMOVE utility program. IEHMOVE allocates its own space on that volume during Stage II of system generation. The DUMMY DD statement is optional, but it should be used to ensure that the required number of tracks will be available during Stage II. (The data set

created by the DUMMY DD statement is deleted at the end of Stage I.) The value to be given to the SPACE keyword of the DUMMY DD statement is determined by the type of device on which the data set specified by the SYSUT3 DD statement resides. The appropriate values are shown in Table 2.

Table 2. Space Allocation for Utility Data Sets

Device Type	DD Statement				
	OBJPDS	SYSUT1	SYSUT2	SYSUT3	DUMMY
2311	(40,20,8)	(240,20)	(240,20)	(300,20)	(80)
2301	(10,2,8)	(45,4)	(45,4)	(60,4)	(14)
2303	(40,8,8)	(180,16)	(180,16)	(240,16)	(80)
2314	(32,10,8)	(160,10)	(160,10)	(200,10)	(60)

The DD statement named SYSPUNCH defines the data set that is to contain the job stream produced during Stage I of system generation. If any error messages (see Appendix A) occur during system generation, the job stream is not produced. After a successful completion of Stage I, the job stream produced becomes the input to Stage II. (The stage I output should be saved after system generation for maintenance purposes.) If the device defined by the SYSPUNCH DD statement is a card punch, the operator is required to place the cards in an input device and to issue a START RDR command for that device. However, operator intervention can be eliminated by making the output (SYSPUNCH) of Stage I become the input reader to Stage II. If the value given to UNIT in the SYSPUNCH DD statement is the specific unit name of a magnetic tape drive, this can be accomplished by inserting a // START RDR,xxx statement after the /* card of the input deck (see Figure 12). xxx is the specific unit name given to UNIT in the SYSPUNCH DD statement. If there were any errors during Stage I, there will be an end-of-file condition detected and the reader will be closed. The section "Operating Considerations" indicates some of the conditions that require operator intervention, and shows sample console listings from system generation processes.

Note: The input deck described is the input to Stage I of the system generation process. During Stage I a job stream is produced which serves as the input for Stage II. After a satisfactory completion of Stage I, the beginning of Stage II is a logical restart point. If any errors are made that require restarting from the beginning of Stage I, the data sets specified by the OBJPDS, SYSUT1, SYSUT2, and SYSUT3 DD statements must be scratched and uncataloged before restarting. For more information refer to the section "Restart Procedures."

CONVENTIONS

This section describes the conventions used to code system generation macro-instructions and the notation used in this publication to describe system generation macro-instructions.

CODING MACRO-INSTRUCTIONS

The rules for coding system generation macro-instructions are those of the assembler language. The following paragraphs are a summary of these rules as stated in the publication IBM System/360 Operating System: Assembler Language.

System generation macro-instructions have the following standard format:

Name	Operation	Operand
Symbolic name	Macro-instruction type	Optional and required parameters

The name symbolically identifies the macro-instruction. If included, it can contain from one through eight alphameric characters, the first of which must be alphabetic. The name must begin in the first position of the macro-instruction and must be followed by one or more blanks. Unless otherwise indicated in the description of individual macro-instructions, the name field of a system generation macro-instruction is ignored during system generation.

The operation identifies the macro-instruction. It must be preceded and followed by one or more blanks.

The operand contains parameters coded in any order and separated by commas. The operand field ends with one or more blanks placed after the last parameter. In most system generation macro-instructions, keyword parameters are used in the operand field. A keyword parameter consists of a keyword followed by an equal sign (=) and the keyword value. The keyword value must be a single value or a list of values; in the latter case, the values must be separated by commas and the list enclosed in parentheses.

Comments can be written in a system generation macro-instruction, but they must be separated from the last parameter of the operand field by one or more blanks. An entire card may be used for a comment by placing an asterisk in the first column. Extensive comments may be written by using a series of cards with an asterisk in the first column of each card. A macro-instruction that has no parameters can not have comments.

A typical system generation macro-instruction might appear as:

```
NAME OPERATION KEY1=value,KEY4=(value1,value2),KEY3=value,...
```

System generation macro-instructions are coded in columns 1 through 71 of a card. A macro-instruction that exceeds column 71 can be continued onto one or more additional cards; a nonblank character is placed in column 72 to indicate continuation. The macro-instruction can be interrupted either in column 71 or after any comma that separates parameters. The continued portion must begin in column 16 of the following card. Comments can be coded through column 71, and, if continued, must begin in column 16 of the following card. In addition, comments may appear on every card of a continued statement. Columns 73 through 80 can be used to code identification and/or statement sequence characters.

DESCRIBING MACRO-INSTRUCTIONS

The following conventions are used in this publication to illustrate the format and coding of system generation macro-instructions:

- Upper case letters, numbers, and punctuation marks must be coded by the programmer exactly as shown. Exceptions to this convention are brackets, [ ]; braces, { }; ellipses, ...; and subscripts. These are never coded.

- Lower case letters and words represent variables for which the programmer must substitute specific information or specific values.

- Items or groups of items within brackets, [ ], are optional. They may be omitted at the programmer's discretion. Conversely, the lack of brackets indicates that an item or group of items must be coded.

- Braces, { }, group related items.

- Stacked items, enclosed in either brackets or braces, represent alternative items. Only one of the stacked items should be coded.

- If an alternative item is underlined, that item is implied; that is, the operating system will automatically assume it is the programmer's choice when none of the items are coded.

- An ellipsis (...) indicates that the preceding item or group of items can be coded more than once in succession.

SYSTEM GENERATION MACRO-INSTRUCTIONS

The content of the new operating system is specified through system generation macro-instructions. Not all system generation macro-instructions are required for the system generation process. Table 3 lists the system generation macro-instructions for each type of system generation, indicating whether they are required or optional. If neither required nor optional is indicated, that macro-instruction does not apply to that type of system generation.

The type of system generation must be specified in the GENERATE macro-instruction. Table 3 also shows which macro-instructions can be issued more than once during a system generation process. All UNITNAME macro-instructions having the same NAME value must appear together in the input deck. Each IOCONTRL macro-instruction must precede in the input deck to system generation those IODEVICE macro-instructions that define devices attached to that control unit. All other system generation macro-instructions, with the exception of GENERATE, can be issued in any order. The GENERATE macro-instruction must be the last macro-instruction in the input deck for the system generation process.

Dependencies between the parameters of a macro-instruction are illustrated by the macro-instruction format and, when necessary, by tables within each macro-instruction description. Dependencies between macro-instructions are stated in the descriptions of each macro-instruction and summarized in Appendix B. Appendix B also lists the references made in the macro-instruction to other publications.

Several keywords in the system macro-instructions request the specification of the unit name of a device. For example, UNIT=name in the FORTLIB macro-instruction and RESNAME=name in the GENERATE macro-instruction. The unit name of a device can be one of the following:

- specific unit name, or address, such as 192.
- generic unit name, such as 2311. (Generic unit names are listed in Appendix C.)
- user-specified unit name, or name of a collection of devices, such as TAPE or SYSDA.

It is recommended that generic unit names be used, although all three types are equally valid.

Primary data management routines and system utilities are included in every operating system generated; therefore, their inclusion is not specified in any of the system generation macro-instructions.

Note: The region size for a processing program operating under MVT must be greater than the value given to the SIZE parameter of the macro-instruction that specifies that processing program. For further information refer to the Programmer's Guide associated with the processing program. Region sizes are given in the publication IBM System/360 Operating System: Storage Estimates.

Table 3. System Generation Macro-Instructions

Macro-Instruction	Type of System Generation[1]		
	Operating System	Nucleus	Processor/Library
CENPROCS	Required	Required	Required
CHANNEL[2]	Required	Required	-----
IOCONTRL[2]	Required	Required	-----
IODEVICE[2]	Required	Required	-----
UNITNAME[2]	Optional	-----	-----
CTRLPROG	Required	Required	-----
SCHEDULR	Required	Required	-----
SUPRVSOR	Optional	Optional	Optional[3]
SVCTABLE[2]	Optional	Optional	-----
RESMODS	Optional	Optional	-----
SVCLIB	Optional	-----	Optional
PROCLIB	Optional	-----	Optional
LINKLIB	Optional	-----	Optional
DATAMGT	Optional	Optional	-----
TELCMLIB	Optional	-----	-----
GRAPHICS	Optional	Optional	-----
SYSUTILS	Optional	-----	-----
EDITOR[2]	Optional	-----	Optional
ASSEMBLR[2]	Optional	-----	Optional
TESTRAN	Optional	Optional	-----
MACLIB	Optional	-----	Optional
CHKPOINT	Optional	Optional	-----
SORTMERG	Optional	-----	Optional
SORTLIB	Optional	-----	Optional
ALGOL	Optional	-----	Optional
ALGLIB	Optional	-----	Optional
COBOL[2]	Optional	-----	Optional
COBLIB[2]	Optional	-----	Optional
FORTRAN[2]	Optional	-----	Optional
FORTLIB	Optional	-----	Optional
PL1	Optional	-----	Optional
PL1LIB	Optional	-----	Optional
RPG	Optional	-----	Optional
GENERATE	Required	Required	Required

[1]The type of system generation is specified with the GENTYPE parameter of the GENERATE macro-instruction. GENTYPE=ALL specifies the generation of an operating system. GENTYPE=NUCLEUS specifies the generation of a nucleus to be included in the operating system. (No libraries, except SYS1.NUCLEUS, are affected by this type of system generation.) GENTYPE=PROCESSOR specifies the generation of processors and/or their associated libraries to be added to the operating system.

[2]This macro-instruction can be issued more than once during a system generation process.

[3]This macro-instruction is used during a processor/library generation if, and only if, the PL1 macro-instruction is used.

CENPROCS

The CENPROCS macro-instruction is used to describe the central processing unit. This macro-instruction is required.

Name	Operation	Operand
	CENPROCS	[MODEL={30|40|50|65|75}] [STORAGE={E|F|G|H|I|J}] [INSTSET={STD|COMM|SCNTF|UNIV}] [FEATURE=(feature[,feature])]

Operand Field:

MODEL=model
specifies the model of the central processing unit.

STORAGE
specifies the size of main storage as one of the following:

Value	Storage Size (in bytes)
E	32K
F	64K
G	128K
H	256K
I	512K
J	1024K

Note: The MODEL and STORAGE parameters are used for diagnostic purposes only. Therefore, an operating system generated for a CPU could operate on another one.

INSTSET
specifies the instruction set available in the central processing unit as one of the following:

Value	Instruction Set
STD	Standard
COMM	Commercial (Standard instruction set with decimal feature)
SCNTF	Scientific (Standard instruction set with floating point feature)
UNIV	Universal (Standard instruction set with decimal, floating point and storage protection features)

Note: This parameter need not be specified for model 50 or above; UNIV is always assigned in this case.

FEATURE=feature$_n$
specifies the optional features installed in the central processing unit as one or more of the following values. These values can be written in any order.

Value	Feature
TIMER	Interval timer on model 30
PROTECT	Storage protection

Example: In the following example, a CENPROCS macro-instruction is used to describe a model 40 central processing unit with a main storage size of 64K bytes. The commercial instruction set is used.

```
CENPROCS MODEL=40,STORAGE=F,INSTSET=COMM
```

CHANNEL

The CHANNEL macro-instruction is used to describe channel characteristics. A CHANNEL macro-instruction is required for each channel of an installation's computing system.

Name	Operation	Operand
[name]	CHANNEL	ADDRESS=address TYPE={SELECTOR / MULTIPLEXOR / HISPEEDMULTIPLEXOR}

Name Field:

name
is used in system generation error messages (see Appendix A) to identify the CHANNEL macro-instruction that produced an error. If no name is entered, the macro-instruction supplies sequential identification numbers to the CHANNEL macro-instructions in the same order in which these macro-instructions are introduced in the user's input stream. These numbers are used for identification purposes instead of names. For example, if the name is omitted from the third CHANNEL macro-instruction in the input stream, the name CHAN#3 is supplied in each diagnostic message resulting from an error encountered in the macro-instruction.

Operand Field:

ADDRESS=address
specifies the address of the channel as one of the following: 0, 1, 2, 3, 4, 5, or 6.

TYPE
specifies the type of channel as one of the following:

Value	Channel
SELECTOR	Selector
MULTIPLEXOR	Multiplexor
HISPEEDMULTIPLEXOR	2870 Multiplexor

Note: If a 2870 multiplexor channel is specified, no burst devices should be attached to the multiplexor portion of the channel.

Example: The following example illustrates the use of the CHANNEL macro-instruction to describe a multiplexor channel whose address is 0.

```
MPX CHANNEL ADDRESS=0,TYPE=MULTIPLEXOR
```

IOCONTRL

The IOCONTRL macro-instruction is used to describe a control unit and its operating system requirements. An IOCONTRL macro-instruction is required for each control unit (listed in Table 4) in the user's computing system. A maximum of 40 IOCONTRL macro-instructions may be issued during a system generation. If more are required see Appendix D for the procedure to be followed. For assistance in choosing valid combinations of values for the UNIT, MODEL, and FEATURE keywords, refer to Table 4.

Name	Operation	Operand
[name]	IOCONTRL	UNIT=unit ADDRESS=address [MODEL=model] [FEATURE=(feature[,feature]...)] For UNIT=2821 Only: TRNMODE={BYTE / BURST} For UNIT=2840 Only: [EXPBFR=number]

Name Field:

name
is used in system generation error messages (see Appendix A) to identify the IOCONTRL macro-instruction that produced an error. If no name is entered, the macro-instruction supplies sequential identification numbers to the IOCONTRL macro-instructions in the same order in which these macro-instructions are introduced in the user's input stream. These numbers are used for identification purposes instead of names. For example, if the name is omitted from the sixth IOCONTRL macro-instruction in the input stream, the name UNIT#6 is supplied in each diagnostic message resulting from an error encountered in the macro-instruction.

Operand Field:

UNIT=unit
specifies the control unit number as one of the following: 1051, 2403, 2404, 2415, 2701, 2702, 2703, 2803, 2804, 2820, 2821, 2822, 2840, 2841, or 2848.

Notes:
- The IBM 1052 printer-keyboard is attached to model 40 or above through an IBM 1052 adapter, and not through a control unit. Thus, an IOCONTRL macro-instruction is not needed in this case.
- 2816 is implied through the specification of the OPTCHAN parameter in the IODEVICE macro-instruction, and need not be specified in any macro-instruction.
- If a 2260 is attached to a 2701 through a 2848; the 2848 is implied and must not be specified.

ADDRESS=address
specifies the address of the control unit. The address value consists of two hexadecimal digits whose valid range of values is

00 to 6F. This value normally corresponds to the two high order digits of the addresses of the devices attached to the control unit. If the high order digits of the device addresses differ, the lowest value must be used. For example, if the addresses of the devices attached to the control unit are 00E and 010, the value given to the ADDRESS keyword of the IOCONTRL macro-instruction must be 00.

Note: There must be only one IOCONTRL macro-instruction for each control unit. The value given to address must be the lowest address of the control unit. The other addresses must not be specified.

MODEL=model
specifies the model, if any, of the control unit as one of the following: 1, 2, 3, 4, 5, 6, 21, 22, or N1.

FEATURE=feature$_n$
specifies the optional features that are present on the control unit as one of the following values. These values can be written in any order. Features enclosed in braces { } are mutually exclusive.

Value	Feature
ABSLTVEC	Absolute vector control on 2840, model 1
{ADSTORAG}	Additional storage on 2841
{2-CHANSW}	2-channel switch on 2841
BUFFER8K	Additional 8K buffer on 2840, model 1
COLBNRY	Column binary on 2821, model 1, 4, or 5
DATACONV	Data conversion on 2403, 2404, 2803, or 2804
LINEADDR	Line addressing on 2848
NODESCUR	Non-destructive cursor on 2848
RECOFLO	Record overflow on 2841
7-TRACK	7-track compatibility on 2403, 2404, 2803, or 2804
16-DRIVE	16-drive addressing on 2403 or 2803
800BPI	800 BPI on model 4, 5, or 6 of 2403 or 2415, or on model 2 of 2803 or 2804

TRNMODE
specifies the data transmission mode for an IBM 2821 Control Unit attached to a multiplexor channel. The value selected, either BYTE or BURST, must correspond to the setting of the mode switch on the 2821 CE panel. This parameter must be specified if, and only if, the value of UNIT is 2821.

EXPBFR=number
specifies that programs written for a 2250, model 1, not using the DCB operand GTYPE=BASIC can be used with a 2250, models 2 or 3, attached to a 2840. The value specified is the amount of buffer space, in bytes, required by the programs. The value must be an integer of from 1 to 8192. If this parameter is omitted, a value of 4096 is assumed.

Example: The following example illustrates the use of the IOCONTRL macro-instruction to describe an IBM 2821 control unit, model 4, with the column binary feature. The control unit operates in the byte mode, and its address is 05.

```
IOCONTRL UNIT=2821,MODEL=4,ADDRESS=05,FEATURE=COLBNRY,TRNMODE=BYTE
```

Table 4. Keyword Values for IOCONTRL Macro-Instruction (Part 1 of 2)

UNIT	MODEL	FEATURE
1051	1 or N1	
2403	1, 2, or 3	DATACONV 7-TRACK 16-DRIVE
	4, 5, or 6	DATACONV 7-TRACK 16-DRIVE 800BPI
2404	1, 2, or 3	DATACONV 7-TRACK
2415	1, 2, or 3	DATACONV 7-TRACK
	4, 5, or 6	DATACONV 7-TRACK 800BPI
2701		
2702		
2703		
2803	1	DATACONV 7-TRACK 16-DRIVE
	2	DATACONV 7-TRACK 16-DRIVE 800BPI
2804	1	DATACONV 7-TRACK
	2	DATACONV 7-TRACK 800BPI
2820		
2821	1, 4, or 5	COLBNRY
	2, or 3	
2822		

(Continued)

Table 4. Keyword Values for IOCONTRL Macro-Instruction (Part 2 of 2)

UNIT	MODEL	FEATURE
2840	1	BUFFER8K ABSLTVEC
	2	
2841		ADSTORAG or 2-CHANSW RECOFLO
2848[1]	1, 2, 3, 21, or 22	LINEADDR NODESCUR

[1]A 2260, model 1, cannot be attached to a 2848 model 1, 2, 21, or 22.

IODEVICE

The IODEVICE macro-instruction is used to describe the characteristics of an input/output device and its operating system requirements. An IODEVICE macro-instruction is required for each uniquely addressable input/output device in the system. A maximum of 96 IODEVICE macro-instructions may be issued during a system generation. If more are required see Appendix D for the procedure to be followed. In the input deck for system generation, each IOCONTRL macro-instruction must be immediately followed by the IODEVICE macro-instructions that define devices attached to that control unit. For assistance in choosing valid combinations of values for the UNIT, MODEL and FEATURE keywords, refer to Table 5.

In the case of telecommunications devices, there must be one IODEVICE macro-instruction for each telecommunications line. That is, the IODEVICE macro-instruction applies to telecommunications lines, not to terminal devices. All terminals on a line must be of the same type with the same features. The type of terminal on the line is used to identify the line in the UNIT keyword.

The value given to the ADDRESS keyword becomes the specific unit name of the device. Specific unit names are automatically assigned to the devices during system generation. Generic unit names are also provided in every operating system for each type of device specified by the UNIT parameter of an IODEVICE macro-instruction. Generic unit names are described in Appendix C. User-specified unit names can be assigned to a device, or to a collection of devices, with the UNITNAME macro-instruction.

Name	Operation	Operand
[name]	IODEVICE	UNIT={unit / DUMMY} ADDRESS=address [MODEL=model] [FEATURE=(feature[,feature]...)] [ERRTAB=nnn] [DEVTYPE=type] For UNIT=2302,2303,2311,2314,2401,2402, 2403 or 2404 Only: [OPTCHAN=(address[,address]...)] For UNIT=1443 Only: [TRNMODE={BURST / BYTE}] For UNIT=2250 (models 2 or 3),2280,or 2282 Only: [NUMSECT=number] For Telecommunications Lines Only: ADAPTER=adapter [SETADDR=value]

Name Field:

name
is used in system generation error messages (see Appendix A) to identify the IODEVICE macro-instruction that produced an error. If no name is entered, the macro-instruction supplies sequential identification numbers to the IODEVICE macro-instructions in the same order in which these macro-instructions are introduced in the user's input stream. These numbers are used for identification purposes instead of names. For example, if the name is omitted from the eleventh IODEVICE macro-instruction in the input stream, the name DEV#11 is supplied in each diagnostic message resulting from an error encountered in the macro-instruction.

Operand Field:

UNIT=unit
specifies the unit number of the device as one of the following: 1030, 1050, 1052, 1053, 1060, 1130, 1403, 1442, 1443, 2250, 2260, 2280, 2282, 2301, 2302, 2303, 2311, 2314, 2321, 2401, 2402, 2403, 2404, 2415, 2501, 2520, 2540R, 2540P, 2671, 2740, 115A, 83B3, S360, or TWX. (In the case of telecommunications lines, UNIT specifies the device type that is in the telecommunications line.)

DUMMY specifies an unsupported device. A 32-byte UCB with all its standard fields is generated. It is assumed that the user provides his own I/O support routines for the device. Specific unit names are not generated for unsupported devices. If desired, they can be generated by the UNITNAME macro instruction. (See the examples following the description of the UNITNAME macro instruction.)

Notes:
- 2540R and 2540P refer to the same IBM 2540 card read punch, but, because they are uniquely addressable, the read (2540R) and punch (2540P) functions must be specified as two different units in separate IODEVICE macro-instructions.
- S360 refers to a remote System/360 attached to the channel through a 2701 or 2703 control unit. For the purposes of system generation, this remote System/360 is considered to be a telecommunications line, and must be defined as such with an IODEVICE macro instruction.
- TWX refers to either the teletype model 33 or 35.

ADDRESS=address
specifies the address of the device or telecommunications line. The address value consists of three hexadecimal digits whose valid range of values is 000 to 6FF. The high order digit of the address value is the address of the channel (specified in the CHANNEL macro-instruction) to which the device is attached. For example, if the address of the device is 190, the address of the channel is 1.

Note: There must be only one IODEVICE macro-instruction for each 2314. The value given to ADDRESS must be the lowest address of the 2314. The remaining seven addresses are automatically generated. For example, if the addresses of the 2314 are 130 through 137, only ADDRESS=130 must be specified. Any other device that has more than one address, such as the 2302, must be considered as a separate device for each address. For example, there must be two IODEVICE macro-instructions for each 2302 that has two addresses.

MODEL=model
specifies the model number, if any, of the device as one of the following: 1, 2, 3, 4, 5, 6, 7, B1, B2, B3, N1, or N2. This parameter must be specified if the unit has a model number (see Table 5).

Table 5. Keyword Values for the IODEVICE Macro-Instruction (Part 1 of 3)

UNIT	MODEL	FEATURE
1030		AUTOPOLL
1050[1]		AUTOANSR AUTOCALL AUTOPOLL
1052[2]	5, 6, or 7	
1053	4	
1060		AUTOPOLL
1130[1]		AUTOANSR AUTOCALL AUTOPOLL DUALCODE DUALCOMM
1403[3]	2, 3, or N1	UNVCHSET
	7	
1442	N1 or N2	CARDIMAGE
1443	N1	SELCHSET 24ADDPOS
2250	1	ABSLTVEC ALKYB2250 BUFFER4K or BUFFER8K CHARGNTR DESIGNFEAT LIGHTPEN OPPAN1 or OPPAN2 PRGMKYBD
	2	ALKYB2250 LIGHTPEN PRGMKYBD
	3	ALKYB2250 PRGMKYBD
2260	1 or 2	ALKYB2260 or NMKYB2260 or DEKYB2260
2280		
2282		
2301		
2302	3 or 4	
2303		
2311		

(Continued)

Table 5. Keyword Values for the IODEVICE Macro-Instruction (Part 2 of 3)

UNIT	MODEL	FEATURE
2314		2-CHANSW
2321		
2401	1, 2, or 3	READWRITE 7-TRACK or 9-TRACK MDECOMPAT
	4, 5, or 6	READWRITE 7-TRACK or 9-TRACK DUALDENS
2402	1, 2, or 3	READWRITE 7-TRACK or 9-TRACK MDECOMPAT
	4, 5, or 6	READWRITE 7-TRACK or 9-TRACK DUALDENS
2403	1, 2, or 3	7-TRACK or 9-TRACK MDECOMPAT
	4, 5, or 6	7-TRACK or 9-TRACK DUALDENS
2404[4]	1, 2, or 3	7-TRACK or 9-TRACK
2415	1, 2, 3, 4, 5, or 6	7-TRACK or 9-TRACK
2501	B1 or B2	CARDIMAGE
2520	B1, B2, or B3	CARDIMAGE
2540R	1	

(Continued)

Table 5. Keyword Values for the IODEVICE Macro-Instruction (Part 3 of 3)

UNIT	MODEL	FEATURE
2540P	1	
2671	1	
2740[1]		AUTOANSR AUTOCALL AUTOPOLL CHECKING SCONTROL or XCONTROL
115A		
83B3		
S360		AUTOANSR AUTOCALL DUALCODE DUALCOMM
TWX		AUTOANSR AUTOCALL

[1]AUTOPOLL cannot be specified if either AUTOANSR or AUTOCALL (or both) is specified.
[2]A 2150 used to connect a 1052 is addressed as the 1052, and may not be specified.
[3]A 1404 printer is supported only as a continuous form printer, and must be specified as a 1403, model 2.
[4]The READWRITE feature is implicit for the 2404 and must not be specified.

FEATURE=feature$_n$
specifies the optional features that are present on the device as one or more of the following values. These values can be written in any order. Features enclosed in braces { } are mutually exclusive.

Value	Feature
ABSLTVEC	Absolute vector control on 2250, model 1
ALKYB2250	Alphameric keyboard on 2250
{ALKYB2260 / DEKYB2260 / NMKYB2260}	Alphameric keyboard on 2260 / Data entry keyboard on 2260 / Numeric keyboard on 2260
AUTOANSR	Automatic answering capability for 1050, 1130, 2740, S360, or TWX
AUTOCALL	Automatic calling feature for 1050, 1130, 2740, S360, or TWX
AUTOPOLL	Automatic polling feature for 1030, 1050, 1060, 1130, or 2740. If AUTOPOLL is specified for 2740, SCONTROL must also be specified.
{BUFFER4K / BUFFER8K}	4096-byte buffer storage on 2250, model 1 / 8192-byte buffer storage on 2250, model 1
CARDIMAGE	Card image on 1442, 2501, or 2520
CHARGNTR	Character generator on 2250, model 1
CHECKING	VRC/LRC checking on 2740
DESIGNFEAT	Graphic design feature on 2250, model 1
DUALCODE	Decode and dual code feature for 1130, or S360
DUALCOMM	Dual communication interface for 1130, or S360
DUALDENS	Dual density on model 4, 5, or 6 of 2401, 2402, or 2403
LIGHTPEN	Light pen detect on 2250, model 1 or 2
MDECOMPAT	Mode compatibility on model 1, 2, or 3 of 2401, 2402, or 2403
{OPPAN1 / OPPAN2}	First operator control panel on 2250, model 1 / First and second operator control panel on 2250, model 1
PRGMKYBD	Programmed function keyboard on 2250
READWRITE	Simultaneous reading and writing on 2401, or 2402
{SCONTROL / XCONTROL}	Station control feature on 2740. SCONTROL may not be specified if AUTOANSR or AUTOCALL are specified. / Transmit control feature on 2740. XCONTROL requires AUTOANSR, AUTOCALL or both.
SELCHSET	Selective character set on 1443
UNVCHSET	Universal character set on 1403, model 2, 3, or N1
2-CHANSW	2-channel switch on 2314
{7-TRACK / 9-TRACK}	7-track head on 2401, 2402, 2403, 2404, or 2415 / 9-track head on 2401, 2402, 2403, 2404, or 2415
24ADDPOS	24 additional print positions on 1443

Notes:
- Either 7-TRACK or 9-TRACK must be specified when the value of the UNIT keyword is 2401, 2402, 2403, 2404, or 2415.
- If AUTOPOLL is specified for the telecommunications line, neither AUTOCALL nor AUTOANSR can be specified for that line.
- If DUALDENS is specified, the generic unit names 2400 and 2400-3 are generated in addition to 2400-4.

ERRTAB=nnn
specifies that an error routine, other than its standard error routine is to be used for the device. Either an IBM-supplied or a user-written routine may be specified. IBM error routines have the values 000 through 219, and 230 through 254. User-written routines can have the values 220 through 229. This value is the suffix of the name IGE00 under which the error routine is contained in SYS1.SVCLIB. (This parameter should be specified if UNIT=DUMMY is specified.)

DEVTYPE=type

specifies any additional characteristics of the device. The value specified must not exceed eight hexadecimal characters. (This parameter should be specified if UNIT=DUMMY is specified.) For further information on this parameter, refer to the publication IBM System/360 Operating System: System Programmer's Guide, and to the description of the UCB in IBM System/360 Operating System: System Control Blocks, Form C28-6628.

OPTCHAN=address$_n$

specifies the alternative channels through which a 2302, 2303, 2311, 2314, 2401, 2402, 2403, or 2404 may be addressed. Each value specified is the address of an alternative channel as specified in the CHANNEL macro-instruction. These values must be greater than the high order digit of the value of the ADDRESS keyword. (If the magnetic tape drive is attached to a 2403 or 2803 control unit, a 2816 is required in order to have alternative channel addressing. In this case, the presence of the 2816 is implied when specifying the OPTCHAN parameter and it must not be specified elsewhere.) Only one alternative channel may be specified for 2302, 2303, 2311 or 2314. A maximum of three alternative channels can be specified for magnetic tape drives. However, if the READWRITE feature is specified for the device, one and only one alternative channel may be specified. The addresses of alternative channels can be written in any order.

Note: There must be only one IODEVICE macro-instruction for each I/O device, regardless of the number of alternative addresses given to the device. For example, if the primary address of a device is 181, and if it can also be addressed through channels 2, 3, and 4, there must not be separate IODEVICE macro-instructions defining the address of the device as either 281, 381, or 481. The primary address of the device, that is, the one with the lowest channel address, must be specified in the ADDRESS keyword. The other channel addresses must be specified with the OPTCHAN keyword. In this example, the macro-instruction for the device must contain the parameters ADDRESS=181 and OPTCHAN=(2,3,4).

TRNMODE

specifies the data transmission mode for the device. BURST specifies the burst mode of data transmission. BYTE specifies the byte mode, i.e., multiplex mode of data transmission. This parameter must be specified if, and only if, the device is on a multiplexor channel and the device specified is an IBM 1443 printer, Model N1.

NUMSECT=number

specifies the number of 256-byte buffer sections in the 2840 control unit to be guaranteed available (that is, that can only be used by the device) to the 2250 (model 2 or 3), 2280, or 2282. The value specified may range from 1 to a maximum value which is computed as follows:

$$\frac{A}{256} - B + 1$$

where:

A is the size of the buffer.
B is the number of devices attached to the 2840.

The total amount of buffer sections guaranteed to the devices attached to a 2840 must not exceed the number of sections in the buffer of that 2840. All sections not guaranteed to a device are available to all devices attached to the 2840. If this parameter is omitted, the device uses the sections not guaranteed to other devices; however, there must always be at least one section available (guaranteed or not) for assignment to each device. Note

that the assignment of guaranteed sections limits the amount of available contiguous sections. For further information, see the publication IBM System/360 Operating System: Graphic Programming Services for IBM 2250 Display Unit, Form C27-6909.

ADAPTER=adapter
specifies the terminal control or transmission adapter used to connect a telecommunications I/O device to its control unit.

Value	Adapter or Control
BSCA	IBM Binary Synchronous Terminal Adapter Type II attaching an 1130, or S360 to a 2701, or IBM Binary Synchronous Terminal Control Type II attaching an 1130, or S360 to a 2703.
IBM1	IBM Terminal Adapter Type I attaching a 1050, 1060, or 2740 to a 2701, or IBM Terminal Control Type I attaching a 1050, 1060, or 2740 to a 2702 or 2703
IBM2	IBM Terminal Adapter Type II attaching a 1030 to a 2701, or IBM Terminal Control Type II attaching a 1030 to a 2702 or 2703
IBM3	IBM Terminal Adapter Type III attaching a 2848/2260 to a 2701
IBMT	IBM Telegraph Adapter attaching a 1050 to a 2701, or IBM Terminal Control Type I and a Telegraph Line Adapter attaching a 1050 to a 2702 or 2703
TELE1	Telegraph Adapter Type I attaching a 115A or 83B3 to a 2701, or Telegraph Terminal Control Type I attaching a 115A or 83B3 to a 2702 or 2703
TELE2	Telegraph Adapter Type II attaching a TWX to a 2701, or Telegraph Terminal Control Type II attaching a TWX to a 2702 or 2703

SETADDR=value
specifies the set address (SAD) command to be issued for a telecommunications line attached to a 2702 or 2703 control unit. This parameter is required if the device is attached to a 2702.

Value	Command
0	SADZERO
1	SADONE
2	SADTWO
3	SADTHREE

Examples: The following example illustrates the use of the IODEVICE macro-instruction to describe an IBM 1404 printer, model 2. The address of the device is 20E.

```
PRINTER2 IODEVICE UNIT=1403,MODEL=2,ADDRESS=20E
```

The following example illustrates the use of the IODEVICE macro-instruction to describe an IBM 2401 magnetic tape drive, model 3, with a 9-track head. The address of the device is 181. This device can be addressed alternatively through channel 2.

```
TAPE1 IODEVICE UNIT=2401,ADDRESS=181,OPTCHAN=2,MODEL=3,FEATURE=9-TRACK
```

UNITNAME

The UNITNAME macro-instruction is used to name a collection of I/O devices. A UNITNAME macro-instruction is required for each named collection of I/O devices in the system, except for generic unit names (see Appendix C). A maximum of 50 uniquely named collections can be specified for the system. If more are required, see Appendix D for the procedure to be followed. This macro-instruction is optional. If selected, all UNITNAME macro-instructions having the same NAME value must appear together in the input stream.

Name	Operation	Operand
	UNITNAME	UNIT=(address[,address]...) NAME=name

Operand Field:

UNIT=address$_n$
specifies the addresses of the I/O devices to be included in the collection. The addresses must be the same as those specified in the IODEVICE macro-instructions for these devices. The only combination of unlike device types permitted in a collection is magnetic tape and direct-access devices. The maximum number of devices that can be included in collections is determined by the following formula:

$$A=510-N$$

where:

A is the maximum number of devices (a device may belong to more than one collection, but it must be counted as a separate device for each of the collections).

N is the number of uniquely named collections. (The maximum value of N is 50.)

For example, if there are 40 collections, a maximum of 470 devices can be distributed among those collections.

NAME=name
specifies the name to be given to the collection of devices. The name may be from one to eight alphameric characters.

Note: If the user intends to use IBM-supplied cataloged procedures, he must assign, using UNITNAME macro-instructions, certain names to collections of I/O devices. These names will be used by the IBM-supplied cataloged procedures (contained in the procedure library) to specify the I/O devices required. The names to be specified follow.

Name	Types of I/O Devices in the Collection
SYSSQ	magnetic tape, direct-access
SYSDA	direct-access
SYSCP	card punch

It is recommended that 2321 addresses not be included in the collections specified for SYSDA and SYSSQ because some of the processors (COBOL E, sort/merge, and RPG) do not support intermediate work data sets on 2321 volumes. It is also recommended that a collection of devices named SYSOUT be defined for intermediate system output data sets in MVT. The devices in the SYSOUT collection should be a subset of the devices in the SYSDA collection. (The SYSOUT unit name should also be

defined as the default value in the reader cataloged procedure for MVT.) For further information on the SYSSQ, SYSDA, SYSCP, and SYSOUT unit names see the section "System Reader and Writer Cataloged Procedures" in the publication IBM System/360 Operating System: System Programmer's Guide.

Examples: The following example illustrates the use of the UNITNAME macro-instruction to assign the name TAPE to the devices located at 180, 181, 182, and 183.

```
UNITNAME UNIT=(180,181,182,183),NAME=TAPE
```

The following example illustrates the use of the UNITNAME macro instruction to assign a specific unit name to an unsupported I/O device. The unsupported device is located at address 167 (specified as UNIT=DUMMY, ADDRESS=167 with an IODEVICE macro instruction).

```
UNITNAME NAME=167,UNIT=167
```

CTRLPROG

The CTRLPROG macro-instruction is used to specify control program options. This macro-instruction is required.

Name	Operation	Operand
	CTRLPROG	[TYPE={PCP / MFT / MVT}] MAXIO=number For TYPE=PCP or MFT only: [OVERLAY={BASIC / ADVANCED}] [FETCH={STD / PCI}] For TYPE=MFT Only: HITASK=size LOWTASK=(size$_1$[,size$_2$][,size$_3$]) For TYPE=MVT only: [QSPACE=number] [ADDTRAN=number] [OVERLAY=ASYCHRON] [FETCH=PCI]

Operand Field:

TYPE
specifies the type of control program. PCP specifies the primary control program. MFT specifies multiprogramming with a fixed number of tasks. MVT specifies multiprogramming with a variable number of tasks.

Note: MVT can only operate on a Model 40, or above, central processing unit that has at least 256K bytes of main storage. The universal instruction set and the storage protection feature are also required. (See the CENPROCS macro-instruction.)

MAXIO=number
specifies the maximum number of I/O operations that can be simultaneously processed by the new operating system. This number is the sum of those I/O operations that are being executed simultaneously and those that are concurrently queued but are not being executed. (A recommended minimum value for MAXIO is the number of uniquely addressable I/O devices in the new system.)

Note: This number limits the maximum number of channel programs that can be started when using access methods or graphic programming services.

OVERLAY
specifies overlay supervisor options. BASIC specifies synchronous overlay without exclusive call checking. ADVANCED specifies synchronous overlay with error checking for invalid SEGWT instructions.

Note: If ADVANCED is specified, STORAGE=E must not be specified in the CENPROCS macro-instruction.

FETCH
specifies the type of program fetch. STD specifies standard fetch. PCI specifies the use of Program Controlled Interrupt while fetching a program into storage.

HITASK=size
specifies the size, in bytes, of the highest priority partition. The value specified must be a decimal integer, and it must also be a multiple of 8 that is equal to, or greater than, 6144.

LOWTASK=$size_n$
specifies the sizes, in bytes, of one, two, or three low priority partitions. The value specified must be a decimal integer, and it must also be a multiple of 8 that is equal to, or greater than, 6144. The last value specified must be at least as large as the scheduler design level specified with the DESIGN keyword of the SCHEDULR macro-instruction.

Note: If PROTECT is specified as a value of the OPTIONS keyword of the SUPRVSOR macro-instruction, the values given to HITASK and LOWTASK must be multiples of 2048.

QSPACE=number
specifies the number of 2048-byte blocks required for the system queue area. If this parameter is omitted, a value of 10 is assumed. The value specified can be increased at IPL time if OPTION=COMM is specified in the SUPRVSOR macro-instruction. Refer to IBM System/360 Operating System: Storage Estimates for information on what value to specify for this parameter.

ADDTRAN=number
specifies the number, if any, of additional pairs of transient areas. Each area is 1024 bytes.

OVERLAY=ASYCHRON
specifies the asynchronous overlay supervisor for MVT.

FETCH=PCI
specifies PCI fetch for MVT.

Examples: The following example illustrates the use of the CTRLPROG macro-instruction to specify the primary control program. The maximum number of I/O operations that can be processed simultaneously is 20. The basic overlay supervisor and standard fetch are included.

```
CTRLPROG MAXIO=20
```

The following example illustrates the use of the CTRLPROG macro-instruction to specify multiprogramming with a variable number of tasks. The maximum number of I/O operations that can be processed simultaneously is 30. The asynchronous overlay supervisor and PCI fetch are assumed. Fifteen 2K blocks are specified for the system queue area. There will be a total of six transient areas: the original pair plus two additional pairs.

```
CTRLPROG   TYPE=MVT,MAXIO=30,QSPACE=15,ADDTRAN=2
```

SCHEDULR

The SCHEDULR macro-instruction is used to specify job scheduler options. This macro-instruction is required.

Name	Operation	Operand
	SCHEDULR	[TYPE={SEQUENTIAL / PRIORITY}] CONSOLE={address / (I-address,O-address)} [ALTCONS={address / (I-address,O-address)}] [OPTIONS=BYLABEL] [STARTR=(A-address[,V-serial] [,D-dsname])] [STARTW=(A-address[,V-serial] [,D-dsname])] [ACCTRTN={NOTSUPPLIED / SUPPLIED}] [WTOBFRS=number] [REPLY=number] For TYPE=SEQUENTIAL Only: [DESIGN={18K / 44K / 100K}] [RESJOBQ=number] [CANCEL=(condition$_1$[,condition$_2$])] [OPSEPAR=(class[,class])] [TSYSIN={200 / 556 / 800}] [TSYSOUT={200 / 556 / 800}{,1600 / ,800}] [VLMOUNT=AVR] [TAVR={200 / 556 / 800}] For TYPE=PRIORITY Only: [STARTI={MANUAL / AUTO}] [WTLCLSS=classname] [WTLBFRS=number] [PROCRES=address] [JOBQRES=address} [JOBQFMT=number] [JOBQLMT=number] [JOBQTMT=number] [INITQBF=number] [MINPART=number]

Operand Field:

TYPE
specifies the type of job scheduler as either SEQUENTIAL or PRIORITY. SEQUENTIAL must be specified if TYPE=PCP or TYPE=MFT is specified in the CTRLPROG macro-instruction. PRIORITY must be specified if TYPE=MVT is specified in the CTRLPROG macro-instruction.

CONSOLE
specifies the address or addresses of the primary console device. If a composite console is used (e.g., a card reader and a printer) as the primary console, I-address specifies the address of the input device and O-address specifies the address of the output device. The address value(s) must be the same as that specified for the device(s) in the IODEVICE macro-instruction(s).

ALTCONS=address
specifies the address of an alternative console device. If a composite console is used as an alternative console, I-address specifies the address of the input device and O-address specifies the address of the output device. If the primary console is a composite console, an alternative console cannot be specified for PCP. An alternative composite console can always be specified for MFT and MVT. The address value must be the same as that specified for the device in the IODEVICE macro-instruction.

OPTIONS=BYLABEL
specifies that all label processing will be bypassed when the BLP parameter is specified in a DD statement. If OPTIONS=BYLABEL is not specified, the BLP parameter will be processed in the same manner as NL. (For further information on this parameter, refer to the publication IBM System/360 Operating System: Job Control Language.)

STARTR
specifies that a START RDR command is to be executed automatically each time the new operating system is loaded into main storage after IPL. A-address specifies the address of the I/O device to be started. The address value must be the same as that specified for the device in the IODEVICE macro-instruction. V-serial specifies the serial number of the labeled volume associated with the I/O device. D-dsname specifies the name of the data set associated with the magnetic tape drive to be started. (D-dsname can only be specified if TYPE=PRIORITY is specified.)

STARTW
specifies that a START WTR command is to be executed automatically each time the new operating system is loaded into main storage after IPL. A-address specifies the address of the device to be started. The address value must be the same as that specified for the device in the IODEVICE macro-instruction. V-serial specifies the serial number of the labeled volume associated with the device. D-dsname specifies the name of the data set associated with the magnetic tape drive to be started. (D-dsname can only be specified if TYPE=PRIORITY is specified.)

ACCTRTN
specifies whether the user supplies an accounting routine. NOTSUPPLIED specifies that the user does not intend to provide an accounting routine. SUPPLIED specifies that the user supplies (or will supply after system generation) an accounting routine. (For further information on accounting routines, refer to the publication IBM System/360 Operating System: System Programmer's Guide.)

WTOBFRS=number
specifies the number of buffers to be used by the WTO routines. Each write buffer is 144 bytes. If this parameter is omitted, a value of 5 is assumed for MFT, or of 100 for MVT.

REPLY=number
specifies the number of reply queue elements to be used by the WTOR routines. Each reply queue element is 24 bytes. If this parameter is omitted, a value of 5 is assumed for MFT, or of 100 for MVT.

Note: WTOBFRS and REPLY can be specified if, and only if, TYPE=MFT or MVT is specified in the CTRLPROG macro-instruction.

DESIGN
specifies the design level (in bytes) of the sequential job scheduler.

RESJOBQ=number
specifies the inclusion of portions of SYS1.SYSJOBQE in the nucleus. The value specified is the number of 176-byte records to be resident.

CANCEL
specifies conditions under which a job is to be canceled without being executed. NONAME specifies that a job is to be canceled if the programmer's name field is omitted from the JOB statement. NOACCNUM specifies that a job is to be canceled if the account number field is omitted from the JOB statement.

Note: The conditions for canceling jobs for the priority scheduler are specified in the PARM field of the EXEC statement of the START RDR procedure.

OPSEPAR
specifies the inclusion of IBM-supplied separator routines. A specifies that the routines are to be used for system output class A; B that the routines are to be used for system output class B.

TSYSIN
specifies the standard magnetic tape density for the system input unit in characters per inch. This parameter applies only to 7-track magnetic tape.

TSYSOUT
specifies the standard magnetic tape density for the system output unit in characters per inch. The first positional parameter refers to 7-track magnetic tape. The second positional parameter refers to dual density magnetic tape.

Note: If either system input or output is on 7-track magnetic tape, the drives must have the data conversion feature. The standard magnetic tape densities for the priority scheduler are specified by DD statements in the START RDR and START WTR procedures.

VLMOUNT=AVR
specifies automatic volume recognition for the volume mounting procedures.

TAVR
specifies the standard density for magnetic tapes used with automatic volume recognition. (This parameter applies only to 7-track magnetic tape.) This parameter may be specified only if VLMOUNT=AVR is specified.

STARTI
specifies whether a START INIT command is to be executed automatically each time the new operating system is loaded into main storage after IPL. AUTO specifies that the command is to be executed automatically; MANUAL specifies that it is not to be executed automatically.

WTLCLSS=classname
specifies the classname to be used as a default for SYSOUT for write-to-log (WTL) messages. The value specified must be a letter from A through Z, or a number from 0 through 9. If this parameter is omitted, L is assumed. (If A is specified, SYSOUT and WTL messages will be interspersed.)

WTLBFRS=number
specifies the maximum number of internal buffers to be used for WTL messages if neither SYS1.SYSVLOGX nor SYS1.SYSVLOGY is available. (If the internal buffers specified are exhausted during execution, the oldest WTL message in the buffers will be written on the console, and the latest message will be written in the buffer.) The value specified must be an integer greater than or equal to 0. If this parameter is omitted, a value of 2 is assumed. If 0 is specified all messages are written on the console.

PROCRES=address
specifies the address of the device on which SYS1.PROCLIB resides. If this parameter is omitted, the address of the system residence device is assumed. The address must be the same as that specified for the device in the IODEVICE macro-instruction. The address specified can be changed when the scheduler is made ready after IPL.

JOBQRES=address
specifies the address of the device on which SYS1.SYSJOBQE resides. If this parameter is omitted, the address of the system residence device is assumed. The address must be the same as that specified for the device in the IODEVICE macro-instruction. The address specified can be changed when the scheduler is made ready after IPL.

JOBQFMT=number
specifies the format of SYS1.SYSJOBQE, the value is the number of 176-byte records to be included in a logical track of SYS1. SYSJOBQE. The value specified must be an integer from 10 to 255. If this parameter is omitted, a value of 12 is assumed. The value specified can be changed when the scheduler is made ready after IPL. Refer to IBM System/360 Operating System: System Programmer's Guide for information on how to specify a value for this parameter.

JOBQLMT=number
specifies the number of 176-byte records in SYS1.SYSJOBQE to be reserved for each initiator started. The value specified must be an integer less than or equal to 9999. If this parameter is omitted, a value of 60 is assumed. The value specified can be changed when the scheduler is made ready after IPL. Refer to IBM System/360 Operating System: System Programmer's Guide for information on how to specify a value for this parameter.

Note: A job is terminated if the number of records required to initiate it exceeds the value specified. It is recommended that the value specified for JOBQLMT be a multiple of the value given to JOBQFMT.

JOBQTMT=number
specifies the number of 176-byte records in SYS1.SYSJOBQE to be reserved for the termination of jobs that require more records for initiation than those specified in JOBQLMT. The value specified must be an integer less than or equal to 9999. If this parameter is omitted, a value of 60 is assumed. The value specified can be changed when the scheduler is made ready after IPL. Refer to IBM System/360 Operating System: System Programmer's Guide for information on how to specify a value for this parameter.

INITQBF=number
specifies the number of 1024-byte buffers, if any, to be used for maintaining portions of SYS1.SYSJOBQE in main storage. The value specified must be an integer from 0 to 255. The value given must be great enough to hold one logical track (specified with JOBQFMT); otherwise, no buffering will occur. This parameter can also be specified, or changed at IPL time if OPTIONS=COMM is specified in the SUPRVSOR macro-instruction. Refer to IBM System/360 Operating System: Storage Estimates for information on how to specify a value for this parameter.

MINPART=blocks
specifies the number of 1024-byte blocks of main storage required for the minimum region for a job. The value specified must be an integer equal to or greater than 52 plus the value of INITQBF. If this parameter is omitted, a value of 52 is assumed. The value specified can be changed at IPL time if OPTIONS=COMM is specified in the SUPRVSOR macro-instruction. Refer to IBM System/360 Operating System: Storage Estimates for information on how to specify a value for this parameter.

Examples: The following example illustrates the use of the SCHEDULR macro-instruction to specify the 18K design level of the sequential job scheduler. The address of the console device is 01F. The address of an alternative console is 21F. The START RDR command is to be executed after the system is loaded into main storage. The device to be started is located at 181. The standard magnetic tape density for system input and output is 200 characters per inch. The user will supply an accounting routine. In addition, the macro-instruction is used to specify the cancellation of all jobs whose account number is omitted.

```
SCHEDULR CONSOLE=01F,ALTCONS=21F,STARTR=A-181,ACCTRTN=SUPPLIED,
                                                CANCEL=NOACCNUM
```

The following example illustrates the use of the SCHEDULR macro-instruction to specify the priority job scheduler required for multiprogramming with a variable number of tasks (MVT). The address of the primary console is 01A. A composite console is used as an alternative console; its input address is 00C and its output address is 00D. An accounting routine will not be supplied. START RDR and START WTR commands are to be executed automatically after IPL. The devices to be started are located at 00E and 00F, respectively. 75 buffers are to be used by WTO routines. 75 reply queue elements are to be used by WTOR routines. A START INIT command is to be executed automatically after IPL. The classname for WTL messages is L. A maximum of 20 buffers are to be used for WTL messages. SYS1.SYSJOBQE and SYS1.PROCLIB are located on the system residence device. The format of SYS1.SYSJOBQE will be the following: twenty 176-byte records for each logical track. Eighty 176-byte records (4 logical tracks) will be reserved for each initiator started, and eighty 176-byte records will be reserved for the termination of jobs that require more than 80 records for initiation. Fifteen 1024-byte buffers are requested for SYS1.SYSJOBQE. Sixty-seven 1024-byte blocks are required to process a job (52 plus 15 INITQBF buffers).

```
SCHEDULR TYPE=PRIORITY,CONSOLE=01A,ALTCONS=(I-00C,O-00D),
         STARTR=A-00E,STARTW=A-00F,WTOBFRS=75,REPLY=75,
         STARTI=AUTO,WTLBFRS=20,JOBQFMT=20,JOBQLMT=80,
         JOBQTMT=80,INITQBF=15,MINPART=67
```

SUPRVSOR

The SUPRVSOR macro-instruction is used to select task supervisor options. This macro-instruction is optional.

For assistance in choosing valid task supervisor options for PCP, MFT, and MVT, refer to Table 6.

Name	Operation	Operand
	SUPRVSOR	[RESIDNT=(function[,function]...)] [OPTIONS=(option[,option]...)] [WAIT={SINGLE / MULTIPLE}] [TIMER={TIME / INTERVAL / JOBSTEP}] [TRACE=number] [SER={SER0 / SER1}]

Operand Field:

RESIDNT=function$_n$
specifies that one or more of the following routines, normally executed from the transient area, are to be included in the resident portion of the control program. The values are ATTACH, EXTRACT, IDENTIFY, and SPIE. The following values can also be specified; they refer to functions that can be made resident at IPL time:

Value	IPL Time Function
BLDLTAB	Directory entries for selected SYS1.LINKLIB modules are to be resident.
ACSMETH	Access method modules to be loaded and made part of the nucleus. This value must not be specified if TYPE=MVT is specified in the CTRLPROG macro-instruction.
RENTCODE	Any module in SYS1.LINKLIB or SYS1.SVCLIB can be made resident. This value can only be specified if TYPE=MVT is specified in the CTRLPROG macro-instruction.
TRSVC	Type 3 and 4 SVC modules are to be loaded and made resident.

The use of these options is discussed in the publication IBM System/360 Operating System: System Programmer's Guide.

Note: ATTACH, EXTRACT, IDENTIFY, and SPIE are assumed if TYPE=MVT is specified in the CTRLPROG macro-instruction.

OPTIONS=option$_n$

specifies task supervisor options as one or more of the following values. These values may be listed in any order.

Value	Option
IDENTIFY	The IDENTIFY function is to be included. If IDENTIFY is specified as a value of the RESIDNT keyword, it need not be specified as a value of the OPTIONS keyword.
TRSVCTBL	A table containing the relative track addresses of all transient SVCs is to be stored in the resident portion of the control program. This value must be specified if TRSVC is specified in the RESIDNT keyword.
PROTECT	Specifies the inclusion of the protect function when the protect feature is part of the central processing unit. This value must be specified for MVT. (PROTECT includes the VALIDCHK option if TYPE=PCP or MFT are specified in the CTRLPROG macro-instruction.)
VALIDCHK	The WAIT, POST, and GETMAIN/FREEMAIN modules are to contain extra validity checking to determine whether addresses are located within proper boundaries. The validity checking for WAIT also checks for the number of events. VALIDCHK should not be specified if PROTECT is specified.
COMM	Specifies communication with the operator at IPL time for the purpose of changing certain system generation options. The options that can be changed at IPL time are BLDLTAB, ACSMETH, RENTCODE, and TRSVC from the SUPRVSOR macro-instruction; MINPART and INITQBF from the SCHEDULR macro-instruction; and QSPACE from the CTRLPROG macro-instruction. (BLDLTAB, ACSMETH, RENTCODE, and TRSVC can only be changed if they are specified in the RESIDNT keyword.) The communications procedure is described in the publication IBM System/360 Operating System: Operator's Guide under the discussion of NIP messages.
ONLNTEST	specifies the inclusion of the On-Line Test system function. This function allows the running of I/O device tests under the operating system as a job step. (The on-line tests are contained in a separate data set, and must be obtained from the IBM Branch Office representative.)

Note: IDENTIFY and TRSVCTBL are assumed if TYPE=MVT is specified in the CTRLPROG macro-instruction; VALIDCHK is invalid; PROTECT is required.

WAIT

specifies, as either SINGLE or MULTIPLE, the number of events that can be specified in a WAIT macro-instruction. MULTIPLE is assumed if TYPE=MVT is specified in the CTRLPROG macro-instruction; SINGLE is invalid.

TIMER

specifies the inclusion of the timer function when the timer feature is part of the central processing unit. TIME provides the ability to request date plus time of day in various units of measurement. INTERVAL provides the same functions, plus the ability to request, check, and cancel intervals of time. JOBSTEP provides timing of each job step and enforcement of job step time limits.

Note: Either INTERVAL or JOBSTEP must be specified if TYPE=MVT is specified in the CTRLPROG macro-instruction; JOBSTEP is invalid if TYPE=PCP or MFT is specified.

TRACE=number
specifies the inclusion of an optional trace table. The value is the number of entries in the table. (See the publication IBM System/360 Operating System: System Programmer's Guide for a description of the trace table.)

SER
specifies the system environment recording (SER) level desired. Either SER0 or SER1 may be specified. If this parameter is included, the nucleus generated is CPU dependent, that is, it can only be used in the CPU model specified in the CENPROCS macro-instruction. (This parameter is valid only on models 40, 50, 65, and 75.) For further information on this parameter, refer to the publication IBM System/360 Operating System: Storage Estimates.

Table 6. SUPRVSOR Macro-Instruction Values for PCP, MFT, and MVT

Keyword	Value	PCP	MFT	MVT
RESIDNT	ATTACH	Optional	Optional	Assumed
	EXTRACT	Optional	Optional	Assumed
	IDENTIFY	Optional	Optional	Assumed
	SPIE	Optional	Optional	Assumed
	BLDLTAB	Optional	Optional	Optional
	ACSMETH	Optional	Optional	N/A
	RENTCODE	Invalid	Invalid	Optional
	TRSVC	Optional	Optional	Optional
OPTIONS	IDENTIFY	Optional	Optional	Assumed
	TRSVCTBL	Optional	Optional	Assumed
	PROTECT	Optional	Optional	Required
	VALIDCHK	Optional	Optional	Invalid
	COMM	Optional	Optional	Optional
	ONLNTEST	Optional	Optional	Optional
WAIT	SINGLE	Optional	Optional	Invalid
	MULTIPLE	Optional	Optional	Assumed
TIMER	TIME	Optional	Optional	Invalid
	INTERVAL	Optional	Optional	Optional[1]
	JOBSTEP	Invalid	Invalid	Optional[1]
TRACE	number	Optional	Optional	Optional
SER	SER0	Optional	Optional	Optional
	SER1	Optional	Optional	Optional

[1]Either INTERVAL or JOBSTEP must be specified for MVT.

Examples: The following example illustrates the use of the SUPRVSOR macro-instruction to specify task supervisor options for the primary control program (PCP) or for multiprogramming with a fixed number of tasks (MFT). A table containing the relative track addresses of transient SVCs is to be stored in the resident portion of the control program. The IDENTIFY and the ATTACH functions are to be included in the resident portion of the control program. Multiple events can be specified in a WAIT macro-instruction. There are 100 entries in the trace table.

```
SUPRVSOR OPTIONS=TRSVCTBL,RESIDNT=(IDENTIFY,ATTACH),WAIT=MULTIPLE,
                                                    TRACE=100
```

The following example illustrates the use of the SUPRVSOR macro-instruction to specify task supervisor options for multiprogramming with a variable number of tasks (MVT). It is assumed that ATTACH, EXTRACT, IDENTIFY, and SPIE are to be made resident. Resident type 3 and 4 SVC routines are to be loaded and made resident at IPL time. The IDENTIFY function and a table of the relative track addresses of all transient SVCs are assumed. The protect function is required. The operator may change TRSVC, MINPART, INITQBF, and QSPACE, but not RENTCODE and BLDLTAB. MULTIPLE events are assumed for the WAIT macro-instruction. Timing of each job step is selected. There are 150 entries in the trace table. SER1 is to be used.

```
SUPRVSOR RESIDNT=TRSVC,TIMER=JOBSTEP,TRACE=150,SER=SER1,
         OPTIONS=(PROTECT,COMM)
```

SVCTABLE

The SVCTABLE macro-instruction is used to specify the number, type, and SVRB extended save area of the user-written supervisor call (SVC) routines that are to be added to the new operating system. This macro-instruction is optional.

Name	Operation	Operand
	SVCTABLE	operand[,operand]...

Operand Field: Each operand must be written in the following format:

SVC-nnn-Ta-Sb

Upper case letters and hyphens (-) must be written exactly as shown.

nnn
specifies the SVC number as a decimal integer. The highest SVC number that may be assigned is 255. The user must assign unique numbers to his SVC routines, and should assign them in descending order starting with 255 and ending with 200 to avoid conflict with the numbers assigned to IBM-written SVC routines.

a
specifies the SVC type as either 1, 2, 3, or 4.

b
specifies the size of the extended save area of the SVRB associated with the SVC routine. The value indicates the number of double words by which the SVRB is to be extended. A type 1 SVC must have a value of 0. Types 2, 3, and 4 can have a value of from 0 to 6.

Note: For each type 1 or type 2 SVC, there should be a corresponding module specified in the RESMODS macro-instruction; one module may contain more than one resident SVC routine. For each type 3 SVC, there should be a corresponding module specified in the SVCLIB macro-instruction; each module may contain only one transient SVC routine. For each type 4 SVC, there should be one corresponding module specified in the SVCLIB macro-instruction for each load module of the SVC routine. (For further information on user-written SVC routines refer to the publication IBM System/360 Operating System: System Programmer's Guide.)

Example: The following example illustrates the use of the SVCTABLE macro-instruction to specify that four user-written SVCs are to be added to the operating system to be generated.

```
SVCTABLE SVC-255-T4-S5,SVC-254-T2-S3,SVC-253-T3-S1,SVC-252-T1-S0
```

RESMODS

The RESMODS macro-instruction is used to add user-written routines, in load module form, to the nucleus library (SYS1.NUCLEUS) to be generated. Before these load modules can be included in the nucleus library, they must be members of a partitioned data set. This data set must have been cataloged, as SYS1.name, in the generating system. This macro-instruction is optional.

Name	Operation	Operand
	RESMODS	PDS=SYS1.name MEMBERS=(name[,name]...)

Operand Field:

PDS=SYS1.name
specifies the name of the partitioned data set that contains the load modules to be included. The value of name cannot exceed eight alphameric characters, the first of which must be alphabetic.

MEMBERS=$name_n$
specifies the simple names of the members to be included. The value of name cannot exceed eight alphameric characters, the first of which must be alphabetic. A maximum of ten load modules can be included in the nucleus library.

Note: If resident SVC routines are being included, each load module can contain more than one SVC routine. The type, number, and SVRB extended save area of each of the resident SVC routines to be included must be specified in the SVCTABLE macro-instruction. (For further information on user-written SVC routines, refer to the publication IBM System/360 Operating System: System Programmer's Guide.)

Example: This example illustrates the use of the RESMODS macro-instruction to include the load modules CONTROL and IORTN in the nucleus library. These modules are members of the SYS1.NEW partitioned data set.

```
RESMODS PDS=SYS1.NEW,MEMBERS=(CONTROL,IORTN)
```

SVCLIB

The SVCLIB macro-instruction is used to add transient user-written routines, in load module form, to the SVC library (SYS1.SVCLIB) during system generation. Before these routines can be included in the SVC library they must be members of a partitioned data set. This data set must have been cataloged, as SYS1.name, in the generating system. The number, type, and SVRB extended save area of each of the SVC routines to be added must be specified in the SVCTABLE macro-instruction. The SVCLIB macro-instruction is optional.

Name	Operation	Operand
	SVCLIB	PDS=SYS1.name MEMBERS=(name[,name]...)

Operand Field:

PDS=SYS1.name
specifies the name of the partitioned data set that contains the routines to be added. The value of name cannot exceed eight alphameric characters, the first of which must be alphabetic.

MEMBERS=$name_n$
specifies the names of the members to be added. Each name (unless referring to nonstandard label routines) must be of the form:

IGCssnnn

where:

ss
is the number of the load module minus one, e.g., the second load module has a value of 01. The value of ss is always 00 for type 3 SVC routines.

nnn
is an SVC number. nnn must be a signed decimal integer (e.g., 242=24B) if the routine is called directly by SVC.

The names of nonstandard label routines must begin with NS and can not exceed eight alphameric characters. With this macro-instruction, a maximum of 50 members may be added to the SVC library. (For further information on user-written SVC routines and nonstandard label routines refer to the publication IBM System/360 Operating System: System Programmer's Guide.)

Example: The following example illustrates the use of the SVCLIB macro-instruction to add the routines named IGC0025E, IGC0025D, IGC0025C, and IGC0025B to the SVC library. These routines are members of the SYS1.USERSVC partitioned data set, and are called directly by an SVC.

```
SVCLIB PDS=SYS1.USERSVC,MEMBERS=(IGC0025E,IGC0025D,IGC0025C,
                                                IGC0025B)
```

PROCLIB

The PROCLIB macro-instruction is used to specify the inclusion of the procedure library (SYS1.PROCLIB) in the new operating system. SYS1.PROCLIB must be allocated space and, if desired, cataloged in the new system during the preparation for system generation. This library must also exist as a cataloged partitioned data set (SYS1.PROCLIB) in the generating system. This macro-instruction is optional. If not chosen, a null data set must be made available to the generated system. (See "Input Deck for Initialization" in the section "Preparation for System Generation.") If IBM-supplied cataloged procedures are to be used, certain names must be assigned to collections of devices. Refer to the description of the UNITNAME macro-instruction for a list of these names.

Name	Operation	Operand
	PROCLIB	[UNIT=name] [VOLNO=serial]

Operand Field:

UNIT=name
specifies the unit name of a direct-access device in the generating system. The volume on this device is to contain the procedure library being generated.

VOLNO=serial
specifies the serial number of the volume that is to contain the procedure library. The value specified must be the same as the value given to the VOLUME keyword of the DD statement used to allocate space for SYS1.PROCLIB during the preparation for system generation.

Note: If these parameters are omitted, the procedure library is placed on the new system residence volume. (If one of these parameters is specified, both must be specified.)

Example: This example illustrates the use of the PROCLIB macro-instruction to specify the inclusion of the procedure library in the operating system to be generated. The unit name is 2311. The volume serial number is 909090.

```
PROCLIB UNIT=2311,VOLNO=909090
```

LINKLIB

The LINKLIB macro-instruction is used to add user-written routines, in load module form, to the link library (SYS1.LINKLIB) during system generation. Before these routines can be included in the link library, they must be members of a partitioned data set. This data set must have been cataloged, as SYS1.name, in the generating system. This macro-instruction is optional.

Name	Operation	Operand
	LINKLIB	PDS=SYS1.name MEMBERS=(name[,name]...)

Operand Field:

PDS=SYS1.name
specifies the name of the partitioned data set that contains the routines to be added. The value of name cannot exceed eight alphameric characters, the first of which must be alphabetic.

MEMBERS=$name_n$
specifies the names of the members to be added. The value of name cannot exceed eight alphameric characters, the first of which must be alphabetic. With this macro-instruction, a maximum of 20 routines may be included in the link library during one system generation process.

Example: The following example illustrates the use of the LINKLIB macro-instruction to add the routines PAYROLL, COMPILER, and MULT to the link library. These routines are members of the SYS1.USER partitioned data set.

```
LINKLIB PDS=SYS1.USER,MEMBERS=(PAYROLL,COMPILER,MULT)
```

DATAMGT

The DATAMGT macro-instruction allows the user to specify optional access methods. This macro-instruction is optional.

Name	Operation	Operand
	DATAMGT	ACSMETH=(method[,method]...)

Operand Field:

ACSMETH=method$_n$
specifies the optional access methods to be included as one or more of the following values. These values may be listed in any order.

Value	Access Method
BDAM	Basic direct access method (BDAM) and routines for creating a direct data set.
ISAM	Queued and basic index sequential access methods (QISAM and BISAM).
BTAM	Basic telecommunications access method (BTAM).
QTAM	Queued telecommunications access method (QTAM).

Note: BTAM and QTAM require the telecommunications library. The generation of SYS1.TELCMLIB is specified with the TELCMLIB macro-instruction. If QTAM is specified, TYPE=MFT or MVT must be specified in the CTRLPROG macro-instruction, and WAIT=MULTIPLE and TIME=INTERVAL must be specified in the SUPRVSOR macro-instruction.

Example: The following example illustrates the use of the DATAMGT macro-instruction to specify that the basic direct access method is to be included in the operating system to be generated.

```
   DATAMGT ACSMETH=BDAM
```

TELCMLIB

The TELCMLIB macro-instruction is used to specify the inclusion of the telecommunications subroutine library in the new operating system. SYS1.TELCMLIB must be allocated space and, if desired, cataloged in the new system during the preparation for system generation. This macro-instruction is optional.

Name	Operation	Operand
	TELCMLIB	[UNIT=name] [VOLNO=serial]

Operand Field:

UNIT=name
specifies the unit name of a direct-access device in the generating system. The volume on this device is to contain the telecommunications subroutine library being generated.

VOLNO=serial
specifies the serial number of the volume that is to contain the telecommunication subroutine library. The value specified must be the same as the value given to the VOLUME keyword of the DD statement used to allocate space for SYS1.TELCMLIB during the preparation for system generation.

Note: If these parameters are omitted, the telecommunications subroutine library is placed on the new system residence volume. (If one of these parameters is specified, both must be specified.)

Example: This example illustrates the use of the TELCMLIB macro-instruction to specify the inclusion of the telecommunications subroutine library in the operating system to be generated. SYS1.TELCMLIB is to reside on the new system residence volume.

```
   TELCMLIB
```

GRAPHICS

The GRAPHICS macro-instruction is used to specify the inclusion of Graphic Programming Services. This macro-instruction is optional.

Name	Operation	Operand
	GRAPHICS	[PORRTNS={INCLUDE / EXCLUDE}] [GSP={EXCLUDE / INCLUDE}]

Operand Field:

PORRTNS
specifies the inclusion of problem oriented routines (PORs) in SYS1.LINKLIB. INCLUDE specifies that the PORs are to be included; EXCLUDE, that PORs are not to be included.

GSP
specifies the inclusion of the FORTRAN IV Graphic Subroutine Package (GSP) in SYS1.LINKLIB and SYS1.FORTLIB. INCLUDE specifies that GSP is to be included; EXCLUDE that GSP is not to be included.

Example: The following example illustrates the use the GRAPHICS macro-instruction to specify the inclusion of Graphic Programming Services in the operating system to be generated. Problem oriented routines are to be included in SYS1.LINKLIB. GSP is not to be included.

```
GRAPHICS
```

SYSUTILS

The SYSUTILS macro-instruction is used to specify the amount of main storage available to the system and data set utilities. System and data set utilities are generated with every operating system and they will operate in 15K bytes of main storage unless this macro-instruction is used to specify a larger amount. This macro-instruction is optional.

Name	Operation	Operand
	SYSUTILS	SIZE=size

Operand Field:

SIZE=size
specifies the amount of main storage, in bytes, available to the system and data set utilities. The value specified must be an integer of from 15360 to 999999, or it may be of the form nnnnK where nnnn is an integer of from 15 to 9999 and K represents 1024 bytes. If this parameter is omitted, a value of 15360 is assumed.

Example: The following example illustrates the use of the SYSUTILS macro-instruction to specify that there are 44K bytes of main storage available to the system and data set utilities.

```
SYSUTILS SIZE=44K
```

EDITOR

The EDITOR macro-instruction is used to specify the inclusion of the linkage editor. This macro-instruction is optional. This macro-instruction must be issued once for each design level to be generated.

IEWLE150, IEWLE180, and IEWLE440 are the names of the 15K, 18K, and 44K E-design-level linkage editors. The alias IEWL (used for cataloged procedures) and the alias LINKEDIT (used for supervisor assisted linkages) are given to the linkage editor chosen or, if more than one is chosen, it is given to the largest.

The cataloged procedures for MVT assume the E44 linkage editor. If the E44 linkage editor is not chosen, the REGION parameter of the cataloged procedures should be changed using the IEBUPDTE utility program described in the publication IBM System/360 Operating System: Utilities. Region sizes are given in the publication IBM System/360 Operating System: Storage Estimates.

Name	Operation	Operand
	EDITOR	DESIGN={E15 / E18 / E44}

Operand Field:

DESIGN
specifies the design level of the linkage editor to be included. E15 specifies the E-design-level linkage editor that operates in 15K bytes of main storage. E18 specifies the E-design-level linkage editor that operates in 18K bytes of main storage. E44 specifies the E-design-level linkage editor that operates in 44K bytes (or more) of main storage.

Example: The following example illustrates the use of the EDITOR macro-instruction to specify the E-design-level linkage editor that operates in 18K bytes of main storage.

```
EDITOR DESIGN=E18
```

ASSEMBLR

The ASSEMBLR macro-instruction is used to specify the inclusion of the assembler language processor. This macro-instruction is optional. However, it must be issued once for each design level to be generated.

IETASM and IEUASM are the names of the assembler E and assembler F, respectively. The alias ASMBLR is given to the assembler chosen or, if both are chosen, it is given to assembler F.

The cataloged procedures for MVT assume the assembler F processor. If assembler F is not chosen, the REGION parameter of the cataloged procedures should be changed using the IEBUPDTE utility program described in the publication IBM System/360 Operating System: Utilities. Region sizes are given in the publication IBM System/360 Operating System: Storage Estimates.

Name	Operation	Operand
	ASSEMBLR	DESIGN={E / F}

Operand Field:

DESIGN
specifies the design level of the assembler language processor as E or F.

Note: The E-design-level of the assembler language processor requires an unblocked macro library (SYS1.MACLIB). If the E-design-level assembler language processor is to be used for future system generations, SYS1.GENLIB must also be unblocked.

Example: The following example illustrates the use of the ASSEMBLR macro-instruction to specify the E design level of the assembler language processor.

```
ASSEMBLR DESIGN=E
```

TESTRAN

The TESTRAN macro-instruction is used to specify the inclusion of test translator options in the new operating system. This macro-instruction is optional.

Name	Operation	Operand
	TESTRAN	PHASES=(phase[,phase]) [MODE={NOTRACE / TRACE}] [PAGES=numbers] [EXEC=statements]

Operand Field:

PHASES=phase$_n$
specifies the test translator components to be included as one or both of the following values.

Value	Component
INTER	The TESTRAN interpreter facility is to be included.
EDITOR	The TESTRAN editor facility is to be included.

MODE
specifies the inclusion of the tracing facilities of the test translator. TRACE specifies that the tracing facilities are to be included. NOTRACE specifies that the tracing facilities are not to be included. This parameter is meaningful only if INTER is specified in the PHASES parameter.

PAGES=number
specifies the maximum number of pages to be produced during one execution of the test translator. The value specified must be a positive decimal integer smaller than 999. This parameter is required if and only if INTER is specified in the PHASES parameter.

EXEC=statements
specifies the maximum number of test translator statements that are to be interpreted during one execution of the test translator. The value specified must be a positive decimal integer smaller than 65535. This parameter is required if and only if INTER is specified in the PHASES parameter.

Example: The following example illustrates the use of the TESTRAN macro-instruction to specify a test translator with tracing facilities. The test translator interpreter is specified. During one execution of the test translator the maximum number of pages to be written is 50, and the maximum number of test translator statements to be interpreted is 200.

```
TESTRAN MODE=TRACE,PHASES=INTER,PAGES=50,EXEC=200
```

MACLIB

The MACLIB macro-instruction is used to specify the inclusion of the macro library (SYS1.MACLIB) in the new operating system. SYS1.MACLIB must be allocated space and, if desired, cataloged in the new system during the preparation for system generation. This library must also exist as a cataloged partitioned data set (SYS1.MACLIB) in the generating system. This macro-instruction is optional.

If the new system is to contain the E-design-level assembler language processor, SYS1.MACLIB must be unblocked. Unblocking is accomplished by allocating space to SYS1.MACLIB during the preparation for system generation with a BLKSIZE value of 80 rather than 3360 (see Table 1).

This macro-instruction may not be used if the volume that contains SYS1.MACLIB of the generating system cannot remain mounted throughout Stages I and II of system generation. SYS1.MACLIB may be included after system generation in the new system with the IEBCOPY utility program.

Name	Operation	Operand
	MACLIB	[UNIT=name] [VOLNO=serial]

Operand Field:

UNIT=name
specifies the unit name of a direct-access device in the generating system. The volume on this device is to contain the macro library being generated.

VOLNO=serial
specifies the serial number of the volume that is to contain the macro library. The value specified must be the same as the value given to the VOLUME keyword of the DD statement used to allocate space for SYS1.MACLIB during the preparation for system generation.

Note: If these parameters are omitted, the macro library is placed on the new system residence volume. (If one of these parameters is specified, both must be specified.)

Example: This example illustrates the use of the MACLIB macro-instruction to specify the inclusion of the macro library in the operating system to be generated. The unit name is 2311. The volume serial number is 003475.

```
MACLIB UNIT=2311,VOLNO=003475
```

CHKPOINT

The CHKPOINT macro-instruction is used to include the checkpoint/restart facility. This macro-instruction is optional. It cannot be used if TYPE=MVT is specified in the CTRLPROG macro-instruction.

Name	Operation	Operand
	CHKPOINT	

Operand Field: The operand field must be left blank.

Example: The following example illustrates the use of the CHKPOINT macro-instruction to specify checkpoint/restart.

```
CHKPOINT
```

SORTMERG

The SORTMERG macro-instruction is used to specify the inclusion of sorting and merging functions in the new operating system. Either all sort/merge functions or selected sort/merge functions can be specified for inclusion. However, any function not specified through the SORTMERG macro-instruction must not be specified on sort/merge control cards at execution time. For example, if the sorting of only fixed length records is specified during system generation, the sorting of variable length records must not be requested at sort/merge execution time. If such sorting were specified, the sort/merge would be terminated because the programs for sorting variable length records would not be in the generated operating system. This macro-instruction is optional. If it is used, the EDITOR and the SORTLIB macro-instructions must also be specified.

Name	Operation	Operand
	SORTMERG	[SIZE=size] [SORTOPT=FULLIB]
		If SORTOPT=FULLIB is not chosen:
		RECTYPE=(rectype[,rectype]...) SORTDEV=(device[,device]...) CNTLFLD=(cntlfld[,cntlfld]) [MERGE=MERGONLY] [MESSAGE=(msgopt[,msgopt])] [SORTOPT=MODPRGM]

Operand Field:

SIZE=size
specifies, as a positive decimal integer, the maximum amount of main storage, in bytes, that can be used for sorting. This amount of main storage is used only for sorting and does not include the space required for data management functions. If this parameter is omitted, a value of 12000 is assumed. This parameter can be changed at execution time. For further information on this parameter refer to the publication IBM System/360 Operating System: Sort/Merge, Form C28-6543.

Note: The sort/merge program operates in 15500 bytes of main storage. Of these 15500 bytes, 12000 (minimum value that can be given to SIZE) are used for sorting. The user can specify in the SIZE parameter that a larger amount of main storage be used for sorting. The sort/merge program issues a variable GETMAIN macro-instruction to request storage in which to operate. The main storage thus made available to the sort/merge program will vary from 12000 bytes to any other amount specified by the user in the SIZE parameter.

The maximum value that can be specified for SIZE is the difference between the total amount of main storage available, and the amount required for data management routines. The following formula can be used to determine the value of SIZE:

size=A-24N-Y-5000

where:

A
is the total amount of main storage available for execution. (The maximum amount is the number of bytes of main storage as specified in the CENPROCS macro-instruction, minus the bytes required for the nucleus of the operating system minus, if sort/merge is called by another program, the bytes occupied by other programs.)

N
is the maximum number of DD statements to be used in any user's sort/merge program.

Y
is a constant with a value of 1500 which must be entered into the formula if any messages are to be written on SYSOUT. Otherwise, the value of Y is 0.

SORTOPT=FULLIB
specifies that all sort/merge functions be included in the user's operating system. If this parameter is written, the RECTYPE, SORTDEV, CNTLFLD, MERGE, MESSAGE, and SORTOPT=MODPRGM parameters must be omitted. (The values of CONSOLE and ALL are assumed for the MESSAGE keyword.)

RECTYPE=$rectype_n$
specifies the type and length of records to be sorted or merged as one or more of the following values:

Value	Records
VAR	Variable length records.
FIXED	Fixed length records.
LONG	Records longer than 256 bytes.

Note: VAR or FIXED or both must be specified.

SORTDEV=$device_n$
specifies the device(s) to be used for sorting or merging as one or more of the following values: 2301, 2311, 2314, 2400.

Note: The value of 2400 stands for 2401, 2402, 2403, 2404 and 2415.

CNTLFLD=$cntlfld_n$
specifies control field requirement(s) for sorting and/or merging. One or both of the following values must be specified:

Value	Control Field
SINGLE	Single control fields
MULTIPLE	Multiple control fields

MERGE=MERGONLY
specifies that the merge routines of the sort/merge processor can be executed independently from the sort routines for a merging application.

MESSAGE=mesgopt$_n$

specifies the I/O device on which sort/merge messages are to be printed, as well as the type of messages to be produced. If the MESSAGE operand is omitted, no messages are printed during a sorting or merging operation. The values included in braces { } are mutually exclusive. Acceptable values for this operand are:

Value	Meaning
{PRINTER}	Sort/merge messages are to be printed on a printer.
{CONSOLE}	Sort/merge messages are to be printed on a console typewriter.
{ALL}	All sort/merge messages are to be printed.
{CRITICAL}	Only serious diagnostic sort/merge messages are to be printed.

Note: This parameter can be changed at execution time. For further information refer to the publication IBM System/360 Operating System: Sort/Merge.

SORTOPT=MODPRGM

specifies the inclusion, at sort/merge execution time, of user-written modification programs.

Example: The following example illustrates the use of the SORTMERG macro-instruction to specify the use of fixed length records, single control fields, and IBM 2311 Disk Storage Drives. The merge functions are to be included. The maximum amount of main storage to be used for sorting is 12000 bytes.

```
SORTMERG RECTYPE=FIXED,CNTLFLD=SINGLE,SORTDEV=2311,MERGE=MERGONLY
```

SORTLIB

The SORTLIB macro-instruction is used to specify the inclusion of the sort/merge subroutine library (SYS1.SORTLIB) in the new operating system. SYS1.SORTLIB must be allocated space and, if desired, cataloged in the new system during the preparation for system generation. This library must also exist as a cataloged partitioned data set (SYS1.SORTLIB) in the generating system. This macro-instruction is optional. If it is used, the SORTMERG macro-instruction must also be specified.

Name	Operation	Operand
	SORTLIB	[UNIT=name] [VOLNO=serial]

Operand Field:

UNIT=name
specifies the unit name of a direct-access device in the generating system. The volume on this device is to contain the sort subroutine library being generated.

VOLNO=serial
specifies the serial number of the volume that is to contain the sort subroutine library. The value specified must be the same as the value given to the VOLUME keyword of the DD statement used to allocate space for SYS1.SORTLIB during the preparation for system generation.

Note: If these parameters are omitted, the sort subroutine library is placed on the new system residence volume. (If one of these parameters is specified, both must be specified.)

Example: This example illustrates the use of the SORTLIB macro-instruction to specify the inclusion of the sort subroutine library in the operating system to be generated. The unit name is 2311. The volume serial number is 654321.

```
SORTLIB UNIT=2311,VOLNO=654321
```

ALGOL

The ALGOL macro-instruction is used to specify the inclusion of the ALGOL compiler. This macro-instruction is optional. If it is used, either SCNTF or UNIV must be specified as the value of the INSTSET keyword of the CENPROCS macro-instruction. (The scientific or universal instruction set is required for ALGOL compilations and executions.) Also, STORAGE=E may not be specified in the CENPROCS macro-instruction. If the ALGOL macro-instruction is included, the ALGLIB macro-instruction must also be included.

Name	Operation	Operand
	ALGOL	[SIZE=size] [PUNCH={NODECK | DECK}] [TYPERUN={LOAD | NOLOAD}] [SORCODE={EBCDIC | ISO}] [SORLIST={SOURCE | NOSOURCE}] [PRECISN={SHORT | LONG}]

Operand Field: The parameters of the ALGOL macro-instruction specify the setting of default options at compilation time. Default options are the options that are assumed if the corresponding values of the PARM keyword are omitted from an EXEC statement in an ALGOL compilation.

SIZE=size
: specifies the default option at compilation time for the maximum number of bytes of main storage available to the ALGOL compiler. The value specified must be an integer of from 45056 to 999999. If this parameter is omitted, a value of 45056 is assumed.

PUNCH
: specifies the default option at compilation time for the production of a punched deck of the object program. DECK specifies that a punched deck is to be produced; NODECK, that a punched deck is not to be produced.

TYPERUN
: specifies the default option at compilation time for the production of input to the linkage editor from the program being compiled. LOAD specifies that the program is to be processed by the linkage editor after compilation. NOLOAD specifies that the program is only to be compiled.

SORCODE
: specifies the default option at compilation time that indicates the character set used to keypunch the source programs to be compiled. EBCDIC specifies the EBCDIC character set. ISO specifies the standard 46 character set in BCD established by the International Standards Organization (ISO) for ALGOL.

SORLIST
: specifies the default option at compilation time for the production of a listing of the ALGOL source program and identifier table. SOURCE specifies that the listing is to be produced; NOSOURCE, that the listing is not to be produced.

PRECISN
specifies the default option at compilation time for the internal representation of real values. SHORT specifies fullwords. LONG specifies doublewords.

Example: This example illustrates the use of the ALGOL macro-instruction to specify the inclusion of the ALGOL compiler. Unless otherwise specified at compilation time the compiler will use 90112 bytes of main storage and will accept source programs written in the standard 46 character set in BCD. Also, a source program listing is to be produced, internal real values are to be represented in doublewords, and compiled source programs are to be processed by the linkage editor; a punched deck is not to be produced.

```
ALGOL   SIZE=90112,PRECISN=LONG,SORCODE=ISO
```

ALGLIB

The ALGLIB macro-instruction is used to specify the inclusion of the ALGOL subroutine library (SYS1.ALGLIB) in the new operating system. SYS1.ALGLIB must be allocated space and, if desired, cataloged in the new system during the preparation for system generation. This macro-instruction is optional.

Name	Operation	Operand
	ALGLIB	[UNIT=name] [VOLNO=serial]

Operand Field:

UNIT=name
specifies the unit name of a direct-access device in the generating system. The volume on this device is to contain the ALGOL subroutine library being generated.

VOLNO=serial
specifies the serial number of the volume that is to contain the ALGOL subroutine library. The value specified must be the same as the value given to the VOLUME keyword of the DD statement used to allocate space for SYS1.ALGLIB during the preparation for system generation.

Note: If these parameters are omitted, the ALGOL subroutine library is placed on the new system residence volume. (If one of these parameters is specified, both must be specified.)

Example: This example illustrates the use of the ALGLIB macro-instruction to specify the inclusion of the ALGOL subroutine library in the operating system to be generated. The unit name is 2301. The volume serial number is 333777.

```
   ALGLIB  UNIT=2301,VOLNO=333777
```

COBOL

The COBOL macro-instruction is used to specify the inclusion of the COBOL compiler. This macro-instruction is optional. If it is used, either COMM or UNIV must be specified as the value of the INSTSET keyword in the CENPROCS macro-instruction. (The commercial or the universal instruction set is required for COBOL compilations and executions.) UNIV is required if either floating point literals are used at compilation time, or if exponentiation to a non-integer power or floating point numbers are used at object time. If the COBOL macro-instruction is included, the COBLIB macro-instruction must also be included. This macro-instruction must be issued once for each design level to be generated.

The cataloged procedures for MVT assume the COBOL F compiler. If COBOL F is not chosen, the REGION parameter of the cataloged procedures should be changed using the IEBUPDTE utility program described in the publication IBM System/360 Operating System: Utilities. Region sizes are given in the publication IBM System/360 Operating System: Storage Estimates.

Name	Operation	Operand
	COBOL	DESIGN={E|F} [MSGLEV={FLAGW|FLAGE}] [LINECNT=lines] For COBOL E Only: [DATAMAP={DMAP|NODMAP}] [PROCMAP={PMAP|NOPMAP}] [DISPCHK={DISPCK|NODISPCK}] [BUFSIZE=number] [EDIT={REGED|INVED}] For COBOL F Only: [SIZE=size] [BUF=number] [SORLIST={SOURCE|NOSOURCE}] [STORMAP={NOMAP|MAP}] [PUNCH={NODECK|DECK}] [TYPERUN={LOAD|NOLOAD}] [SEQCHK={SEQ|NOSEQ}] [BASIS={BAS|NOBAS}] [COPY={COPY|NOCOPY}] [SPACE={SPACE1|SPACE2|SPACE3}]

Operand Field: Many of the parameters of the COBOL macro-instruction specify the setting of default options at compilation time. Default options are the options that are assumed if the corresponding values of the PARM keyword are omitted from an EXEC statement in a COBOL compilation.

DESIGN
specifies the design level of the COBOL compiler as either E or F.

MSGLEV
specifies the default option at compilation time for the type of compilation error messages to be printed. FLAGW specifies that all warning and error messages are to be printed. FLAGE specifies that warning messages are not to be printed.

LINECNT=lines
specifies the default option at compilation time for the number of lines to be printed on each page of the COBOL compiler output listing. The value specified must be a two-digit integer of from 10 to 99. If this parameter is omitted, a value of 60 is assumed.

DATAMAP
specifies the default option at compilation time for the production of a listing of the data-names and their addresses either relative to load point for the Working Storage Section or relative to the record addresses for the File or Linkage Sections. DMAP specifies that a listing is to be produced; NODMAP specifies that the listing is not to be produced.

PROCMAP
specifies the default option at compilation time for the production of a listing of the generated instructions for each statement in the Procedure Division. PMAP specifies that the listing is to be produced; NOPMAP specifies that the listing is not to be produced.

DISPCHK
specifies the default option at compilation time for the generation of object code which determines if a field to be displayed exceeds the record length of the device on which it is to be written. DISPCK specifies that a check is to be made; NODISPCK specifies that no check is required.

BUFSIZE=number
specifies the default option at compilation time for the size, in bytes, of each of the six work buffers used during a COBOL compilation. The valid range of values for magnetic tape is 180 to 32000; for volumes on 2311 Disk Storage drives, 180 to 3600; for volumes on 2301 Drum Storage drives, 180 to 20000. (The maximum size is an object time option and it is not checked during system generation.) If this value is omitted, a value of 180 is assumed. The following formula can be used as a guide to determine the maximum value that can be specified to optimize the allocation of available storage for the data-name table and work buffers. (Any remainder should be ignored.)

$$number=\frac{M-30000-[(13+L)(N)]}{6}$$

where:

number
is the size of each work buffer. If the result is less than 180, 180 must be specified.

M
is the size (in bytes) of main storage. This value should correspond to the size specified in the CENPROCS macro-instruction.

L
is the length of the average data-name.

N
is the number of data-names.

EDIT
specifies the default option at compilation time for the editing function to be used by the compiler. REGED specifies that the standard monetary editing function will be used. INVED specifies that the inverted monetary editing function will be used.

SIZE=size
specifies the default option at compilation time for the number of bytes of main storage available to the COBOL F compiler. The value specified must be an integer of from 81920 to 9999999. If this parameter is omitted, a value of 81920 is assumed. For further information on this parameter, refer to the publication IBM System/360 Operating System: COBOL (F) Programmers Guide, Form C28-6380.

BUF=number
specifies the default option at compilation time for the number of bytes of main storage to be used for buffer allocation by the COBOL F compiler. The value specified must be an integer of from 2762 to 99999. This value must be included in the value given to the SIZE parameter. If BUF is omitted and SIZE is specified, the value of BUF is calculated as:

$$\frac{\text{SIZE-81920}}{4} + 2762$$

If both BUF and SIZE are omitted, a value of 2762 is assumed for BUF.

SORLIST
specifies the default option at compilation time for the production of a listing of the COBOL source program. SOURCE specifies that the listing is to be produced; NOSOURCE specifies that the listing is not to be produced.

STORMAP
specifies the default option at compilation time for the production of a listing of the data-names and their internal attributes in the Data Division and the generated instructions for the statements in the Procedure Division. MAP specifies that a listing is to be produced; NOMAP specifies that a listing is not to be produced.

PUNCH
specifies the default option at compilation time for the production of a punched deck of the object program. DECK specifies that a punched deck is to be produced; NODECK specifies that a punched deck is not to be produced.

TYPERUN
specifies the default option at compilation time for the production of input to the linkage editor from the program being compiled. LOAD specifies that the program is to be processed by the linkage editor after compilation; NOLOAD, specifies that the program is to be compiled only.

SEQCHK
specifies the default option at compilation time for the checking of the source program card sequence numbers. SEQ specifies that the source program card sequence numbers are to be checked; NOSEQ specifies that the source program card sequence numbers are not to be checked.

BASIS
specifies the default option at compilation time for the possible presence of a BASIS card in the source program. BAS specifies that a BASIS card may be present in the source program; NOBAS specifies that a BASIS card is not present in the source program. (This parameter is used to optimize buffer allocation at compilation time.)

COPY
specifies the default option at compilation time for the possible presence of a COPY and/or INCLUDE clause in the source program. COPY specifies that a COPY and/or INCLUDE clause may be present in the source program; NOCOPY specifies that neither a COPY nor an INCLUDE clause is present in the source program. (This parameter is used to optimize buffer allocation at compilation time.)

SPACE
specifies the default option at compilation time for the line spacing on the listing obtained when the SOURCE and/or MAP options are specified. SPACE1 specifies single spacing; SPACE2 specifies double spacing, and SPACE3 specifies triple spacing.

Example: The following example illustrates the use of the COBOL macro-instruction to specify an E-design-level COBOL compiler. The number of lines to be printed in each compiler output listing is 55. Listings of data-names and their address, and listings of the generated instructions for each statement in the Procedure Division are to be produced. All warning and error messages are to be printed. The generation of object code to determine the length of fields to be displayed is not required. The size of each of the six work buffers used during a COBOL compilation is 5708. The standard monetary editing function will be used.

The formula used to compute the BUFSIZE value is as follows:

$$\frac{65536-30000-[(13+10)(56)]}{6}=5708$$

where the main storage specified is 64K, the length of the average data-name is 10, and the number of data names is 56.

```
COBOL DESIGN=E,LINECNT=55,DISPCHK=NODISPCK,BUFSIZE=5708
```

COBLIB

The COBLIB macro-instruction is used to specify the inclusion of the COBOL subroutine library (SYS1.COBLIB) in the new operating system. SYS1.COBLIB must be allocated space and, if desired, cataloged in the new system during the preparation for system generation. If the E-design-level library is specified, it must also exist as a cataloged partitioned data set (SYS1.COBLIB) in the generating system. The modules for the F-design-level library are generated from SYS1.MODLIB. This macro-instruction is optional. If a combined E and F design level subroutine library is required, two COBLIB macro-instructions must be issued: one specifying DESIGN=E and one specifying DESIGN=F.

Name	Operation	Operand
	COBLIB	DESIGN={E / F}
		[UNIT=name]
		[VOLNO=serial]

Operand Field:

DESIGN
specifies the design level of the subroutine library as either E or F.

UNIT=name
specifies the unit name of a direct-access device in the generating system. The volume on this device is to contain the COBOL subroutine library being generated.

VOLNO=serial
specifies the serial number of the volume that is to contain the COBOL subroutine library. The value specified must be the same as the value given to the VOLUME keyword of the DD statement used to allocate space for SYS1.COBLIB during the preparation for system generation.

Note: If the UNIT and VOLNO parameters are omitted, the COBOL subroutine library is placed on the new system residence volume. (However, if one of these parameters is specified, both must be specified.) If a combined E- and F-design-level subroutine library is to be included and these parameters are specified, they should be specified identically in both COBLIB macro-instructions. If they are specified differently or they are omitted from one of the macro-instructions, the second macro-instruction processed determines where the subroutine library is to reside.

Example: This example illustrates the use of the COBLIB macro-instruction to specify the inclusion of the F-design-level COBOL subroutine library in the operating system to be generated. SYS1.COBLIB is to reside on the new system residence volume.

```
   COBLIB DESIGN=F
```

FORTRAN

The FORTRAN macro-instruction is used to specify the inclusion of the FORTRAN compiler. This macro-instruction is optional. If it is used, either SCNTF or UNIV must be specified as the value of the INSTSET keyword in the CENPROCS macro-instruction. (The scientific or the universal instruction set is required for FORTRAN compilations and executions.) This macro-instruction must be issued once for each design level to be generated.

The cataloged procedures for MVT assume the FORTRAN H compiler. If FORTRAN H is not chosen, the REGION parameter of the cataloged procedures should be changed using the IEBUPDTE utility program described in the publication IBM System/360 Operating System: Utilities. Region sizes are given in the publication IBM System/360 Operating System: Storage Estimates.

Name	Operation	Operand
	FORTRAN	DESIGN={E|G|H} [PUNCH={NODECK|DECK}] [SORLIST={SOURCE|NOSOURCE}] [STORMAP={NOMAP|MAP}] [OBJPROG={LOAD|NOLOAD}] [SORCODE={EBCDIC|BCD}] For FORTRAN E only: [LINELNG=length] [STORAGE={SPACE|PRFRM}] [SCAN={NOADJUST|ADJUST}] For FORTRAN E and FORTRAN H only: [SIZE=size] For FORTRAN G and FORTRAN H only: [OBJLIST={NOLIST|LIST}] [LINECNT=lines] [OBJID={NOID|ID}] For FORTRAN H only: [OPT={0|1|2}] [SOREDIT={NOEDIT|EDIT}] [SORXREF={NOXREF|XREF}]

Operand Field: Many of the parameters of the FORTRAN macro-instruction specify the setting of default options at compilation time. Default options are the options that are assumed if the corresponding values of the PARM keyword are omitted from an EXEC statement in a FORTRAN compilation.

DESIGN
specifies the design level of the FORTRAN compiler as E, G, or H.

Note: The FORTRAN H compiler can only operate on a Model 40, or above, central processing unit that has at least 256K bytes of main storage. (See the CENPROCS macro-instruction.)

PUNCH
specifies the default option at compilation time for the production of a punched deck of the object program. DECK specifies that a punched deck is to be produced; NODECK, that a punched deck is not to be produced.

SORLIST
specifies the default option at compilation time for the production of a listing of the FORTRAN source program. SOURCE specifies that the listing is to be produced; NOSOURCE, that the listing is not to be produced.

STORMAP
specifies the default option at compilation time for the production of a map showing the relative location of variables, constants, etc., in the source program. MAP specifies that the map is to be produced; NOMAP, that the map is not to be produced.

OBJPROG
specifies the default option at compilation time for the production of input to the linkage editor from the program being compiled. LOAD specifies that the source program is to be processed by the linkage editor after compilation. NOLOAD specifies that the source program is only to be compiled.

SORCODE
specifies the default option at compilation time that indicates the character set used to keypunch the source programs to be compiled. BCD specifies the BCD character set. EBCDIC specifies the EBCDIC character set.

LINELNG=length
specifies the default option at compilation time for the maximum print line length at object time. At compilation time, if a FORMAT statement indicates a record larger than the length specified, a warning message is issued. The value specified must be a three-digit integer of from 001 to 255. If this parameter is omitted, a value of 132 is assumed.

STORAGE
specifies the default option at compilation time for the use of main storage during a FORTRAN compilation. SPACE specifies that 15360 bytes of main storage are used for compilations and any amount specified in the SIZE keyword in excess of 15360 bytes is used for the dictionary, overflow table, and I/O buffers. PRFRM specifies that 19456 bytes of main storage are used for compilations and any amount specified in the SIZE keyword in excess of 19456 bytes is used for the dictionary, overflow table, and I/O buffers. (If PRFRM is specified, blocked input to and output from the compiler is allowed.) PRFRM must be specified if TYPE=MVT is specified in the CTRLPROG macro-instruction.

Note: If PRFRM is specified, 19456 is the smallest value that can be specified in the SIZE keyword.

SCAN

specifies the default option at compilation time that indicates whether the source program to be compiled is to be prescanned. The source program would be prescanned for meaningful blanks, embedded blanks, and reserved words and put into a form acceptable to the compiler. NOADJUST specifies that the source program is not to be scanned. ADJUST specifies that the source program is to be scanned.

SIZE=size (For FORTRAN E)

specifies the default option at compilation time for the maximum number of bytes of main storage available to the FORTRAN E compiler. The value specified must be an integer of from 15360 to 9999999, or it may be of the form nnnnK where nnnn is an integer of from 15 to 9999 and K represents 1024 bytes. If 9999999 or 9999K is specified, all available main storage is to be used by the FORTRAN compiler. If this parameter is omitted, a value of 15360 is assumed. For further information on this parameter, refer to the publication IBM System/360 Operating System: FORTRAN IV E Programmer's Guide, Form C28-6603.

SIZE=size (For FORTRAN H)

specifies the amount of main storage available to the FORTRAN H compiler. The value specified must be an integer of from 153600 to nnnnnnn, or it may be of the form nnnnK, where nnnn is an integer of from 150 to nnnn and K represents 1024 bytes. The values nnnnnnn or nnnnK represent the size of the user's problem program area. If this parameter is omitted, a value of 204800 is assumed. The size specified cannot be changed at compilation time.

OBJLIST

specifies the default option at compilation time for the production of pseudo-assembly listing of the direct program. LIST specifies that the listing is to be produced; NOLIST that the listing is not to be produced.

LINECNT=lines

specifies the default option at compilation time for the number of lines to be printed on each page of the FORTRAN G or H output listing. The value specified must be a two-digit integer of from 01 to 99. If this parameter is omitted, a value of 50 is assumed.

OBJID

specifies the default option at compilation time for the assignment of internal statement numbers to calls and function references. ID specifies that statement numbers are to be assigned; NOID that statement numbers are not to be assigned.

OPT

specifies the default option at compilation time to optimize the execution time of the object modules produced by the FORTRAN H compiler. 0 specifies that the object module is not to be optimized. 1 specifies that it is to receive full register assignment and basic program optimization; 2 that it is to receive full register assignment and complete program optimization.

SOREDIT
: specifies the default option at compilation time for the production of a structured source listing in the data set defined by the SYSPRINT DD statement. This listing indicates the loop structure and the logical continuity of the source program. EDIT specifies that the listing is to be produced; NOEDIT that the listing is not to be produced.

SORXREF
: specifies the default option at compilation time for the production of a cross-reference listing in the data set defined by the SYSPRINT DD statement. XREF specifies that the listing is to be produced; NOXREF, that the listing is not to be produced.

Example: The following example illustrates the use of the FORTRAN macro-instruction to specify an E-design-level FORTRAN compiler that operates in 20480 bytes of main storage. Any storage in excess of 15360 bytes but less than 20480 bytes is used for the dictionary, overflow table, and I/O buffers. The BCD character set is to be the default character set option at compilation time. Unless otherwise specified at compilation time, a FORTRAN source program listing is to be produced, and compiled source programs are to be processed by the linkage editor. Also, punched decks and source program maps of variables, constants, etc., are not to be produced. The source program is not to be prescanned. A default option of 132 is assumed as the maximum line length.

```
FORTRAN DESIGN=E,SIZE=20480,STORAGE=SPACE,SORCODE=BCD
```

FORTLIB

The FORTLIB macro-instruction is used to specify the inclusion of the FORTRAN subroutine library (SYS1.FORTLIB) in the new operating system. SYS1.FORTLIB must be allocated space and, if desired, cataloged in the new system during the preparation for system generation. This library must also exist as a cataloged partitioned data set (SYS1.FORTLIB) in the generating system. This macro-instruction is optional.

All library members of SYS1.FORTLIB that have a non-IBM name, are copied intact into the SYS1.FORTLIB of the new system. All IBM-supplied members are copied from SYS1.MODLIB.

The graphic subroutine package (GSP) for FORTRAN IV is specified through the GRAPHICS macro-instruction.

Only one design level of SYS1.FORTLIB may be generated during a system generation process. The E-design-level library contains subroutines for programs compiled on the FORTRAN E compiler. The G- and H-design-level libraries support programs compiled on any design level of the FORTRAN compiler. (The subroutines in SYS1.FORTLIB may also be used by any operating system program.) If the G- or the H-design-level library is to be used by FORTRAN E programs, the FORTRAN E cataloged procedures must be modified as described in the publication IBM System/360 Operating System: FORTRAN IV E Programmer's Guide, Form C28-6603.

Name	Operation	Operand
	FORTLIB	[DESIGN={E|G|H}] [UNIT=name] [VOLNO=serial] [UNTABLE=number] [OBJERR=unit] For G and H Libraries only: [ONLNRD=unit] [ONLNPCH=unit] [BOUNDRY={ALIGN|NOALIGN}]

Operand Field:

DESIGN
specifies the design level of the FORTRAN subroutine library as E, G, or H.

UNIT=name
specifies the unit name of a direct-access device in the generating system. The volume on this device is to contain the FORTRAN subroutine library being generated.

VOLNO=serial
specifies the serial number of the volume that is to contain the FORTRAN subroutine library. The value specified must be the same as the value given to the VOLUME keyword of the DD statement used to allocate space for SYS1.FORTLIB during the preparation for system generation.

Note: If the UNIT and VOLNO parameters are omitted, the FORTRAN subroutine library is placed on the new system residence volume. (If one of these parameters is specified, both must be specified.)

UNTABLE=number
specifies the number of FORTRAN logical I/O units to be used at object time. This number does not necessarily correspond to the number of I/O devices in the installation's System/360 Computing System. The value specified must be a two-digit integer of from 08 to 99. If this parameter is omitted, a value of 08 is assumed.

OBJERR=unit
specifies which FORTRAN logical I/O unit is to be used for object time error messages and FORTRAN dumps. The value specified must be a two-digit integer that does not exceed the value given to UNTABLE, and cannot be the same as the value given to ONLNRD or to ONLNPCH. If the OBJERR parameter is omitted, a value of 06 is assumed.

Note: FORTRAN E cataloged procedures assume logical I/O unit 03 as the OBJERR unit. If the value given to the OBJERR parameter is not 03, the cataloged procedures must be modified as described in the publication IBM System/360 Operating System: FORTRAN IV E Programmer's Guide. (It is recommended that when using the E-design-level library, OBJERR=03 be specified to avoid the allocation of an additional output device.)

ONLNRD=unit
specifies which FORTRAN logical I/O unit is to be used when the READ (on-line) statement is encountered in a FORTRAN G or H source program. The value specified must be a two-digit integer that does not exceed the value given to UNTABLE, and cannot be the same as the value given to OBJERR or to ONLNPCH. If the ONLNRD parameter is omitted, a value of 05 is assumed.

ONLNPCH=unit
specifies which FORTRAN logical I/O unit is to be used when the PUNCH (on-line) statement is encountered in a FORTRAN G or H source program. The value specified must be a two-digit integer that does not exceed the value given to UNTABLE, and cannot be the same as the value given to OBJERR or to ONLNRD. If the ONLNPCH parameter is omitted, a value of 07 is assumed.

Note: FORTRAN G and H cataloged procedures assume logical I/O units 06, 05, and 07 as the OBJERR, ONLNRD, and ONLNPCH units, respectively. If a different value is given to any of those parameters, the cataloged procedures must be modified as described in the publications IBM System/360 Operating System: FORTRAN IV G Programmer's Guide, Form C28-6639, and IBM System/360 Operating System: FORTRAN IV H Programmer's Guide, Form C28-6602.

BOUNDRY
specifies the inclusion of the execution time boundary alignment routine in SYS1.LINKLIB. ALIGN specifies that the routine is provided; NOALIGN that the routine is not provided.

Example: The following example illustrates the use of the FORTLIB macro-instruction to specify the inclusion of the FORTRAN E subroutine library in the operating system to be generated. The unit name is 2301. The volume serial number is 333555. Thirty-two logical units are to be used by the object time load modules. The third unit is to be used for error messages and FORTRAN dumps.

```
FORTLIB UNIT=2301,VOLNO=333555,UNTABLE=32,OBJERR=03,DESIGN=E
```

PL1

The PL1 macro-instruction is used to specify the inclusion of the PL/I compiler. This macro-instruction is optional. If it is used, UNIV must be specified as the value of the INSTSET keyword in the CENPROCS macro-instruction. (The universal instruction set is required for PL/I compilations and executions.) The multiple wait option (specified in the SUPRVSOR macro-instruction) is required for the execution of PL/I object programs that contain WAIT statements with multiple arguments. If this macro-instruction is included, the PL1LIB macro-instruction must also be included. If the PL1 macro-instruction is used during a Processor/Library generation, the same SUPRVSOR macro-instruction specified for the operating system being modified must also be included.

Name	Operation	Operand
	PL1	DESIGN=F [PUNCH={NODECK / DECK}] [TYPERUN={LOAD / NOLOAD}] [SORCODE={EBCDIC / BCD}] [SIZE=size] [OBJLIST={NOLIST / LIST}] [MSGLEV={FLAGW / FLAGE / FLAGS}] [OPT={1 / 0}] [SORLIST={SOURCE / NOSOURCE}] [CHARSET={CHAR60 / CHAR48}] [EXTLIST={NOEXTREF / EXTREF}] [ATRLIST={NOATR / ATR}] [REFLIST={NOXREF / XREF}] [SORMGIN=(m,n)] [LINECNT=number] [CMPTIME={NOMACRO / MACRO}] [MACLIST={SOURCE2 / NOSOURCE2}] [COMPILE={COMP / NOCOMP}] [STMDIAG={NOSTMT / STMT}] [DELETE=(item[,item]...)]

Operand Field: Many of the parameters of the PL1 macro-instruction specify the setting of default options at compilation time. Default options are the options that are assumed if the corresponding values of the PARM keyword are omitted from an EXEC statement in a PL/I compilation. The DELETE parameter specifies a list of values that cannot be used as values of the PARM keyword.

DESIGN
specifies the F design level of the PL/I compiler.

PUNCH
specifies the default option at compilation time for the production of a punched deck of the object program. DECK specifies that a punched deck is to be produced. NODECK specifies that a punched deck is not to be produced.

TYPERUN
specifies the default option at compilation time for the production of input to the linkage editor from the program being compiled. LOAD specifies that the program is to be processed by the linkage editor after compilation. NOLOAD specifies that the source program is only to be compiled.

SORCODE
specifies the default option at compilation time that indicates the character set used to keypunch the source programs to be compiled. BCD specifies the BCD character set. EBCDIC specifies the EBCDIC character set.

SIZE=size
specifies the maximum number of bytes of main storage available to the PL/I compiler at compilation time. The value specified must be an integer of from 45056 to 999999. If this parameter is omitted, a value of 45056 is assumed. For further information on this parameter refer to the publication IBM System/360 Operating System: PL/I (F) Programmer's Guide, Form C28-6594.

OBJLIST
specifies the default option at compilation time for the production of a listing of the object program. LIST specifies that a listing is to be produced; NOLIST, that a listing is not to be produced.

MSGLEV
specifies the default option at compilation time for the type of compilation error messages to be printed. FLAGW specifies that warning messages, error messages, and severe error messages are to be printed. FLAGE specifies that only error messages and severe error messages are to be printed. FLAGS specifies that only severe error messages are to be printed.

OPT
specifies the default option at compilation time to optimize the execution time of the object program produced by the compiler. 1 specifies that it is to be optimized even at the expense of object time storage requirements; 0 that it is to be optimized unless it increases object time storage requirements.

SORLIST
specifies the default option at compilation time for the production of a printed listing of the PL/I source program. SOURCE specifies that a listing of the source text is to be produced. NOSOURCE specifies that a listing is not to be produced.

CHARSET
specifies the number of characters in the character set used to write the source program to be compiled. CHAR60 specifies a character set with 60 characters. CHAR48 specifies a character set with 48 characters.

EXTLIST
specifies the default option at compilation time for the production of a listing of all external data, external entries, and files.

EXTREF specifies that a listing is to be produced. NOEXTREF specifies that a listing is not to be produced.

ATRLIST
specifies the default option at compilation time for the production of a listing for each identifier, the identifier with full qualification, the statement number declaring the identifier, and a list of attributes pertaining to the identifier. ATR specifies that a listing is to be produced. NOATR specifies that a listing is not to be produced.

REFLIST
specifies the default option at compilation time for the production of a listing for each identifier, the identifier with full qualification, the statement number declaring the identifier and a list of all statements in which a reference is made to the identifier. XREF specifies that a listing is to be produced. NOXREF specifies that a listing is not to be produced.

SORMGIN=(m,n)
specifies the default option at compilation time for the margins for scanning the source statements. The value m specifies the beginning margin and n specifies the end margin. If the source statement input to the compiler is from the system input stream (i.e., following a DD* statement), the condition $2 \geq m \geq n \geq 100$ must be valid. If the input is not from the system input stream, the condition $1 \geq m \geq n \geq 100$ must be valid. If this operand is omitted, a value of 2 is assumed for m, and a value of 72 is assumed for n.

LINECNT=number
specifies the default option at compilation time for the maximum number of lines to be printed in each page of a PL/I compiler output listing. The value specified must be an integer of from 10 to 99. If this parameter is omitted, a value of 50 is assumed.

CMPTIME
specifies the default option at compilation time for the compile-time processor. MACRO specifies that compile-time processing is required; NOMACRO, that compile-time processing is not required.

MACLIST
specifies the default option at compilation time for the listing of the input to the compile-time processor. SOURCE2 specifies that a listing is to be produced; NOSOURCE2 that the listing is not to be produced.

COMPILE
specifies the default option at compilation time for compilation to proceed after the compile-time processor has been used. COMP specifies that compilation is required; NOCOMP, that compilation is not required.

STMDIAG
specifies the default option at compilation time for the contents of diagnostic messages printed during execution of the compiled source program. STMT specifies that the messages are to contain source program statement numbers; NOSTMT specifies that the messages are not to contain source program statement numbers. Offsets from PL/I entry points are included in the messages in both cases.

DELETE=item$_n$
specifies that the keyword values or keywords in the value list cannot be used at compilation time in the PARM field of the EXEC statement. The following values can be specified; each has been described above.

DECK	FLAGS	NOXREF
NODECK	OPT	SORMGIN
LOAD	SOURCE	LINECNT
NOLOAD	NOSOURCE	MACRO
EBCDIC	CHAR60	NOMACRO
BCD	CHAR48	SOURCE2
SIZE	EXTREF	NOSOURCE2
LIST	NOEXTREF	COMP
NOLIST	ATR	NOCOMP
FLAGW	NOATR	STMT
FLAGE	XREF	NOSTMT

Example: The following example illustrates the use of the PL1 macro-instruction to specify an F design-level PL/I compiler that operates in 56320 bytes of main storage. The compiled source programs are to be processed by the linkage editor unless otherwise specified at compilation time. It is assumed that all warning and error messages are to be printed, that the EBCDIC character set is used to punch the source programs, that the character set used to write the source programs has 60 characters, and that the execution time of the object programs is not to be optimized. The default options for scanning the source statements are 2 and 72, and for the number of lines in each printed page is 50. The following keywords, and keyword values cannot be used at compilation time in the PARM field of the EXEC statement: CHAR48, BCD, FLAGE, FLAGS, OPT, LINECNT, and SORMGIN. It is assumed that, unless otherwise specified at compilation time, listings of the source text are to be produced; and that a punched deck of the object program, listings of the object program, a listing of external data, entries, and files, listings of identifiers and their attributes, numbers, qualifications, and references are not to be produced. It is also assumed that, unless specified at compilation time, compile-time processing is not required, and that the diagnostic messages are not to contain source program statement numbers.

```
PL1 DESIGN=F,TYPERUN=LOAD,SIZE=56320,
    DELETE=(CHAR48,BCD,FLAGE,FLAGS,OPT,LINECNT,SORMGIN)
```

PL1LIB

The PL1LIB macro-instruction is used to specify the inclusion of the PL/I subroutine library (SYS1.PL1LIB) in the new operating system. SYS1.PL1LIB must be allocated space and, if desired, cataloged in the new system during the preparation for system generation. This library must exist as a cataloged partitioned data set (SYS1.PL1LIB) in the generating system. This macro-instruction is optional.

Name	Operation	Operand
	PL1LIB	[UNIT=name] [VOLNO=serial] [LIBFCNS={REAL | COMPLEX}]

Operand Field:

UNIT=name
specifies the unit name of a direct-access device in the generating system. The volume on this device is to contain the PL/I subroutine library being generated.

VOLNO=serial
specifies the serial number of the volume that is to contain the PL/I subroutine library. The value specified must be the same as the value given to the VOLUME keyword of the DD statement used to allocate space for SYS1.PL1LIB during the preparation for system generation.

Note: If the UNIT and VOLNO parameters are omitted, the PL/I subroutine library is placed on the new system residence volume. If one of these parameters is specified, both must be specified.)

LIBFCNS
specifies the inclusion of complex object time functions in SYS1.PL1LIB. REAL specifies that the complex functions are not to be included. COMPLEX specifies that the complex object time functions are to be included.

Example: The following example illustrates the use of the PL1LIB macro-instruction to specify the inclusion of the PL/I subroutine library in the operating system to be generated. Complex object time functions are not to be included. This library is to be placed on the new system residence volume.

```
PL1LIB
```

RPG

The RPG macro-instruction is used to specify the inclusion of the report program generator (RPG) language processor. This macro-instruction is optional. IESRPG and RPG are the name and alias, respectively, of the RPG language processor.

Name	Operation	Operand
	RPG	

Operand Field: The operand field must be left blank.

Example: The following example illustrates the use of the RPG macro-instruction to specify the RPG language processor.

```
RPG
```

GENERATE

The GENERATE macro-instruction is used to specify the data sets, volumes, and I/O devices required for the system generation process, the system generation output options, and the type of generation being done.

The GENERATE macro-instruction must be the last system generation macro-instruction in the user's input deck. The GENERATE macro-instruction must immediately be followed by an assembler END statement. This macro-instruction is required.

Name	Operation	Operand
	GENERATE	[GENTYPE={ALL / (NUCLEUS,n) / PROCESSOR}] UT1SDS=(SYS1.name{,SL / ,NL}) UT2SDS=(SYS1.name{,SL / ,NL}) UT3SDS=SYS1.name OBJPDS=SYS1.name RESNAME=name RESVOL=serial [RESTYPE={2311 / 2301 / 2303 / 2314}] [LNKNAME=name] [LNKVOL=serial] [LBMAINT=E] [ASMPRT={OFF / ON}] [LEPRT=(option[,option])] [DIRDATA={CATALOG / VTOC / PDS}]

Operand Field:

GENTYPE
specifies the type of system generation. (See Table 3 in the section "Specifying the System.") This may be one of the following:

Value	Generation Type
ALL	An operating system is to be generated. This is the standard type of generation and is assumed if the GENTYPE keyword is omitted.
NUCLEUS	Only a nucleus is to be generated.
PROCESSOR	Only language processors and libraries are to be generated.

When NUCLEUS is specified, n, a decimal integer of from 1 to 9, must also be specified. This identifies the member of the nucleus library (SYS1.NUCLEUS) that is to contain the nucleus to be generated. (The value 1 specifies the member IEANUC01, which is the primary nucleus loaded by the normal IPL procedure.) For further information, refer to the publication IBM System/360 Operating System: Operator's Guide, Form C28-6540.

If NUCLEUS or PROCESSOR is specified, the RESNAME, RESVOL, and RESTYPE parameters refer to the system residence volume of the

operating system being modified, and the LNKNAME, LNKVOL, and LBMAINT parameters refer to the link library of the operating system being modified.

If GENTYPE=ALL, the RESVOL value may not be equal to the serial number of the system residence volume of the generating system; and the LNKVOL value may not be equal to the serial number of the volume that contains the SYS1.LINKLIB of the generating system. If sufficient space is available, and if GENTYPE equals NUCLEUS or PROCESSOR, members may be added to the libraries of the generating system.

Note: If a NUCLEUS generation is specified, the same machine configuration specified for the operating system to be modified must be specified in this generation. Also, a previously resident function cannot be made transient, because no libraries except SYS1.NUCLEUS are affected by this type of generation. However, a previously transient function can be made resident.

UT1SDS
UT2SDS
UT3SDS
specify the names of the sequential data sets to be used during system generation by the assembler, linkage editor and utilities. These data sets must exist as cataloged data sets in the generating system. The data set specified by UT3SDS is used by the linkage editor and must reside on a direct-access volume. If the data sets specified by UT1SDS or UT2SDS reside on magnetic tape, either standard labels (SL) or no labels (NL) must be specified. (See "Input Deck For System Generation" in the section "Specifying the System.")

OBJPDS=SYS1.name
specifies the name of the partitioned data set to be used for the storage of object modules assembled during system generation. This data set must exist as a cataloged partitioned data set in the generating system. (See "Input Deck for System Generation" in the section "Specifying the System.")

RESNAME=name
specifies the unit name of a direct-access device in the generating system. The volume on this device is to become the new system residence volume.

RESVOL=serial
specifies the serial number of the new system residence volume. The value specified must be the same as the value given to the VOLUME keyword of the DD statement used to allocate space for SYS1.NUCLEUS during the preparation for system generation.

RESTYPE
specifies the unit number of the new system residence device as 2311, 2301, 2303, or 2314.

LNKNAME=name
specifies the unit name of a direct-access device in the generating system. The volume on this device is to contain the link library being generated.

LNKVOL=serial
specifies the serial number of the volume that is to contain the link library. The value specified must be the same as the value given to the VOLUME keyword of the DD statement used to allocate space for SYS1.LINKLIB during the preparation for system generation.

Note: If LNKNAME and LNKVOL are omitted, the link library is placed on the new system residence volume. (If one of these parameters is specified, both must be specified.)

LBMAINT=E
specifies that the size of the load modules in the SYS1.LINKLIB and SYS1.FORTLIB being generated is 1024 bytes. The omission of this parameter specifies larger load modules. If this parameter is omitted, the E44 linkage editor must be used to maintain the new system.

ASMPRT
specifies whether assembly listings are to be produced for the modules assembled during system generation. ON specifies that assembly listings are to be generated; OFF that assembly listings are not to be generated.

LEPRT=option$_n$
specifies linkage editor print options as one or two of the following values. The values included in braces {} are mutually exclusive.

Value	Print Option
LIST	List of control statements in card-image format
{MAP}	Module map
{XREF}	Cross-reference table (XREF includes the MAP option)

If this parameter is omitted, only linkage editor error messages, if any, are printed. For a more detailed description of these options see the publication IBM System/360 Operating System: Linkage Editor, Form C28-6538.

DIRDATA
specifies the system directory data to be printed during system generation as one of the following:

Value	System Directory Data
CATALOG	The catalog of the new system is to be printed.
VTOC	The volume table of contents (VTOC) of each volume in the new system is to be printed. The catalog is also to be printed.
PDS	The directories of all partitioned data sets in the new system are to be printed. The VTOCs and the catalog are also to be printed.

If the DIRDATA parameter is omitted, no system directory data is printed.

Example: The following example illustrates the use of the GENERATE macro-instruction to specify the generation of an operating system. The sequential data sets named SYS1.UTIL1, SYS1.UTIL2, and SYS1.UTIL3 will be used during system generation by the assembler, linkage editor, and utilities. SYS1.UTIL3 resides on a direct-access volume. SYS1.OBJECT is the name of the partitioned data set to be used for the storage of load modules assembled during system generation. The unit name of the new system residence device is 190, the device type is 2311, and the serial number of the system residence volume is SYSTEM. Assembly listings, linkage editor printed output, and system directory data are not to be produced. The link library is to be placed on the system residence volume. The size of the load modules in the new SYS1.LINKLIB is to be 1024 bytes.

```
GENERATE UT1SDS=SYS1.UTIL1,UT2SDS=SYS1.UTIL2,UT3SDS=SYS1.UTIL3,
         OBJPDS=SYS1.OBJECT,RESNAME=190,RESTYPE=2311,RESVOL=SYSTEM,
         LBMAINT=E,GENTYPE=ALL
```

The system generation process provides facilities for adding user-written functions to the new SYS1.NUCLEUS, SYS1.SVCLIB, and SYS1.LINKLIB. These user-written functions must be load modules residing in a cataloged partitioned data set in the generating system. That is, the user-written function must be compiled, link edited, and placed in a cataloged partitioned data set before system generation. (Each load module must be a member of the data set). The name of the partitioned data set must be of the form SYS1.name. The value of name cannot exceed eight alphameric characters, the first of which must be alphabetic. The name of the partitioned data set and of the members that contain user-written functions are specified with system generation macro-instructions.

The RESMODS macro-instruction specifies load modules to be added to SYS1.NUCLEUS. For example, Type 1 and 2 SVC routines.

The LINKLIB macro-instruction specifies load modules to be added to SYS1.LINKLIB. These load modules become members of SYS1.LINKLIB. For example, accounting routines.

The SVCLIB macro-instruction specifies load modules to be added to SYS1.SVCLIB. These load modules become members of SYS1.SVCLIB. For example, Type 3 and 4 SVC routines and nonstandard label routines.

If SVCs are added to SYS1.NUCLEUS or SYS1.SVCLIB, the SVCTABLE macro-instruction must also be used. This macro-instruction adds an entry to the SVC table that specifies the characteristics of each SVC added.

The RESMODS, LINKLIB, SVCLIB, and SVCTABLE macro-instructions are described in the section "Specifying the System." Accounting routines, SVCs, and nonstandard label routines are described in the publication IBM System/360 Operating System: System Programmer's Guide.

In the example shown in Figure 13, a user-written function is added to the new operating system. In this example, a CSECT is to be added to SYS1.NUCLEUS. This CSECT consists of a series of constants that describe the nucleus to be generated. During the first step, the CSECT is assembled and placed in a temporary data set (DSNAME=&LOADSET). The CSECT is link edited during the second step and the resulting module becomes member ID of the SYS1.USER data set. SYS1.USER is a partitioned data set residing on volume 333333; it is cataloged in the generating system. During system generation, the CSECT will be included in the nucleus by the following macro-instruction:

```
RESMODS   PDS=SYS1.USER,MEMBERS=ID
```

For a more detailed description of the control statements required by the assembler and linkage editor, refer to the publication IBM System/360 Operating System: Assembler (F) Programmer's Guide, Form C26-3756.

```
//USER      JOB MSGLEVEL=1
//STEP1    EXEC PGM=ASMBLR,PARM='NODECK,LOAD,LIST,NOTEST,NOXREF,NORENT'
//SYSLIB    DD  DSNAME=SYS1.MACLIB,DISP=OLD
//SYSUT1    DD  UNIT=SYSSQ,SPACE=(1700,(400,50))
//SYSUT2    DD  UNIT=SYSSQ,SPACE=(1700,(400,50))
//SYSUT3    DD  UNIT=(SYSSQ,SEP=(SYSUT1,SYSUT2,SYSLIB)),                X
//              SPACE=(1700,(400,50))
//SYSPRINT DD   SYSOUT=A
//SYSGO     DD  DSNAME=&LOADSET,UNIT=SYSSQ,SPACE=(80,(200,50)),         X
//              DISP=(MOD,PASS)
//SYSIN     DD  *
ID          CSECT
            DC  C'XXXXXXXXXX-NUCLEUS ID CSECT-XXXXXXXXXX'
            DC  C'OPERATING SYSTEM GENERATED--5/20/67'
            DC  C'OWNER--DEPT. D58'
            DC  C'SUPPORTS--NFT-ALL ACCESS METHODS'
            DC  C'NUCLEUS--01'
            DC  C'XXXXXXXXXX-END ID CSECT-XXXXXXXXXX'
            END
/*
//STEP2    EXEC PGM=IEWL,PARM=(XREF,LIST,NCAL)
//SYSLIN    DD  DSNAME=&LOADSET,DISP=(OLD,DELETE)
//SYSLMOD   DD  DSNAME=SYS1.USER(ID),UNIT=2311,DISP=(,CATLG),          X
//              VOLUME=(,RETAIN.SER=333333),                           X
//              SPACE=(1024,(50,20,5))
//SYSUT1    DD  UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),                     X
//              SPACE=(1024,(50,20,5))
//SYSPRINT DD   SYSOUT=A
/*
//
```

Figure 13. Preparing a User-Written Load Module

Operator intervention may be required during the system generation process. If the output of Stage I is punched cards, and Stage I has been successfully completed, the operator is required to place those cards on an input device and to issue a START RDR command for that device.

Operator intervention is also required if system generation is performed with only two direct-access devices. During Stage II of a two-drive generation, a message is issued to the operator requesting him to remove the volume that contains SYS1.GENLIB and to mount the volume that is to contain the new system. (The MOUNT command must not be issued by the operator during a two-drive generation, because it would prevent this demounting and mounting of volumes.)

At start of Stage II the date in the SET DATE command should be higher than any expiration date for the system data sets being generated. Otherwise the operator must type in "REPLY 00,'U'" every time a data set with a higher expiration date is to be modified.

The completion of Stage II is indicated by a reader closed message. (After Stage II is terminated, system utility programs may be used to scratch the data sets indicated by UT1SDS, UT2SDS, and UT3SDS.)

If the new system is to be used in the future as a generating system, SYS1.GENLIB and SYS1.MODLIB should be saved.

It is recommended that a back-up copy of the generated operating system be made using the IBCDMPRS utility program. A detailed description of this program can be found in IBM System/360 Operating System: Utilities.

For maintenance purposes the job stream (SYSPUNCH) and the object modules assembled during Stage II (OBJPDS) should be saved after the completion of system generation.

The console sheets produced during system generation should also be saved. The IFC001I message contains information required to reinitialize the SYS1.LOGREC data set.

Figure 14 lists sample console sheets for a two-drive and a three-drive generation. The lines prefixed with an asterisk indicate commands typed by the operator. The asterisks and the comments (preceded by ..) do not appear on the console sheets. These messages are explained in detail in the publication IBM System/360 Operating System: Operator's Guide.

Two-Drive Generation

```
  IEE007A READY
* SET DATE=99.360
  START RDR,00C
  START WTR,00E
* START
  IEF236I ALLOCATION FOR SYSGEN STEP0..... initialize new SYSRES
  IEF237I SYSIN  ON 00C
  IEF280I K 191,111111,SYSGEN
  IEF233A M 191,DLIB02,SYSGEN.............. mount for GENLIB+MACLIB
  IEF233A M 182,SCRTCH,SYSGEN.................. mount for SYSPUNCH
  IEF236I ALLOCATION FOR SYSGEN STEP1.................. start Stage I
  IEF237I SYSIN  ON  00C
  IEF280I  K 191,DLIB02,SYSGEN
  // START RDR,182
  IEF101I RDR CLOSED.................................... end Stage I
  IEF223A M 191,DLIB02,SYSGEN.......................... start Stage II
  IEF234A R 191,DLIB02.......................... remove GENLIB+MACLIB
  IEF233A M 191,111111,SYSGEN....................... mount new SYSRES
  IFC001I D=2311 N=021 F=00070001 L=00070005 S=0007000200 .. SYS1.LOGREC
  IEF280I K 191,111111,SYSGEN....................... save new SYSRES
  IEC202I K 182,1G1001.............................. save job stream
  IEF101I RDR CLOSED.................................... end Stage II
  IEE007A READY
```

Three-Drive Generation

```
  IEE007A READY
* SET DATE=99.360
  START RDR,00C
  START WTR,00E
* START
  IEF233A M 192,111111,SYSGEN
  IEF236I ALLOCATION FOR SYSGEN STEP0..... initialize new SYSRES
  IEF237I SYSIN  ON 00C
  IEF280I K 192,111111,SYSGEN
  IEF233A M 192,111111,SYSGEN....................... mount new SYSRES
  IEF233A M 191,DLIB02,SYSGEN..................... mount GENLIB+MACLIB
  IEF233A M 182,SCRTCH,SYSGEN.................. mount for SYSPUNCH
  IEF236I ALLOCATION FOR SYSGEN STEP1.................. start Stage I
  IEF237I SYSIN  ON 00C
  IEF280I K 192,111111,SYSGEN
  IEF280I K 191,DLIB02,SYSGEN
  // START RDR,182
  IEF101I RDR CLOSED.................................... end Stage I
  IEF233A M 191,DLIB02,SYSGEN....................... start Stage II
  IEF233A M 192,111111,SYSGEN
  IFC001I D=2311 N=021 F=00070001 I=00070005 S=0007000200 .. SYS1.LOGREC
  IEF280I K 192,111111,SYSGEN....................... save new SYSRES
  IEF280I K 191,DLIB02,SYSGEN..................... save GENLIB+MACLIB
  IEC202I K 182,LGL001.............................. save job stream
  IEF101I RDR CLOSED.................................... end Stage II
  IEE007A READY
```

Figure 14. Sample Console Sheets

The system generation process may come to an unsatisfactory completion due to errors during Stage I or Stage II. The procedures to restart Stage I and Stage II are described below.

RESTARTING STAGE I

The most common causes of error during Stage I are:

- Faulty allocation of the utility data sets for system generation.
- Key punching errors in the input deck.
- Contradictory or invalid specifications in the system generation macro instructions.

Faulty allocation of the utility data sets usually causes an abnormal-end-of-task (ABEND) termination. Keypunching errors and invalid specifications are indicated with the system generation error messages (see Appendix A) printed in the SYSPRINT data set. If any errors are found during Stage I, the job stream is not produced.

Stage I consists of only one job step (execution of the assembler) and it must be restarted from the beginning. To restart Stage I follow these procedures:

1. Correct the input deck for system generation.
2. Scratch and uncatalog the utility data sets (specified by the OBJPDS, SYSUT1, SYSUT2, and SYSUT3 DD statements in the input deck).

The utility data sets can be scratched and uncataloged by inserting the following statements in the input deck for system generation. These statements must precede the EXEC PGM=ASMBLR statement of the input deck, as shown in Figure 15.

```
//SCRATCH  EXEC   PGM=IEHPROGM  -RESTART DECK-
//SYSPRINT DD     SYSOUT=A
//OBJPDS   DD     DSNAME=SYS1.name,DISP=(OLD,DELETE)
//SYSUT1   DD     DSNAME=SYS1.name,DISP=(OLD,DELETE)
//SYSUT2   DD     DSNAME=SYS1.name,DISP=(OLD,DELETE)
//SYSUT3   DD     DSNAME=SYS1.name,DISP=(OLD,DELETE)
//SYSIN    DD     DUMMY
```

In the preceding statements, the values given to the DSNAME parameters must be the same as the values given to DSNAME in the corresponding DD statements in the input deck for Stage I. These statements represent a dummy execution of the IEHPROGM utility program. The utility data sets are scratched and uncataloged through the facilities of the job control language. Alternatively, the utility data sets can be scratched and uncataloged through the SCRATCH and UNCATLG statements of IEHPROGM as described in IBM System/360 Operating System: Utilities.

Figure 15 shows an input deck for restarting Stage I. It is assumed that all errors in the input deck have been corrected. The four utility data sets are named SYS1.OBJECT, SYS1.ONE, SYS1.TWO, and SYS1.THREE. SYS1.ONE resides on an unlabeled 9-track magnetic tape volume whose serial number is 000167. SYS1.OBJECT, SYS1.TWO, and SYS1.THREE reside on 2311 volumes whose serial numbers are SYSTEM, GENVOL, and MODVOL,

respectively. The job stream is to be punched by the device designated by the SYSCP unit name.

```
//SYSGEN   JOB MSGLEVEL=1   -SYSTEM GENERATION-
//SCRATCH  EXEC PGM=IEHPROGM  -RESTART DECK-
//SYSPRINT DD  SYSOUT=A
//OBJPDS   DD  DSNAME=SYS1.OBJECT,DISP=(OLD,DELETE)
//SYSUT1   DD  DSNAME=SYS1.ONE,DISP=(OLD,DELETE)
//SYSUT2   DD  DSNAME=SYS1.TWO,DISP=(OLD,DELETE)
//SYSUT3   DD  DSNAME=SYS1.THREE,DISP=(OLD,DELETE)
//SYSIN    DD  DUMMY
//STEP1    EXEC PGM=ASMBLR    -STAGE I INPUT DECK-
//SYSLIB   DD  DSNAME=SYS1.GENLIB,DISP=OLD
//OBJPDS   DD  DSNAME=SYS1.OBJECT,VOLUME=(,RETAIN,SER=SYSTEM),        X
//            DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(40,20,8))
//SYSUT1   DD  DSNAME=SYS1.ONE,VOLUME=(RETAIN,SER=000167),           X
//            DISP=(,CATLG),UNIT=2400,LABEL=(,NL)
//SYSUT2   DD  DSNAME=SYS1.TWO,VOLUME=(,RETAIN,SER=GENVOL),          X
//            DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(240,20))
//SYSUT3   DD  DSNAME=SYS1.THREE,VOLUME=(,RETAIN,SER=MODVOL),        X
//            DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(300,20))
//DUMMY    DD  VOLUME=(,RETAIN,REF=*.SYSUT3),SPACE=(TRK,(80))
//SYSPUNCH DD  UNIT=SYSCP
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
            .
            .
            .   System Generation macro-instructions
            .
            .
            END
/*
//
```

Figure 15. Restarting Stage I

RESTARTING STAGE II

The most common causes of error during Stage II are:

- Machine interruptions and noncontinuous machine time.
- Faulty space allocation of the system data sets during the preparation for system generation, especially the primary allocation and the directory quantity.
- Errors in the input deck that cannot be detected during Stage I. For example, if SYS1.NUCLEUS was allocated space on volume 111111 during the preparation of system generation, and if RESVOL=A11111 is specified in the GENERATE macro-instruction, an error would occur.
- The job stream is too large for the generating SYS1.SYSJOBQE.
- Procedural errors, e.g., volume mounting problems.

Stage II consists of the execution of the job stream produced during Stage I. This job stream is one job that has several job steps. Restarting can be accomplished by starting at the beginning of any step in the job stream, providing all previous steps have been properly executed. Before restarting, some other operations may be required depending upon the reason for restarting and the number of steps already executed. The following topics describe the job stream, discuss restart techniques, give guidelines

for restarting Stage II, describe the reallocation of data sets, and give guidelines for changing the size of SYS1.SYSJOBQE.

THE JOB STREAM

If no error messages are printed during Stage I, the job stream is produced on the SYSPUNCH data set. The job stream contains one JOB statement followed by many EXEC statements (see Figure 16). Each EXEC statement is followed by its associated DD statements and other data required to execute the assembler, linkage editor, and utility programs during Stage II.

The format of the JOB statement is:

```
//SYSGEN JOB 1,"SYSTEM GENERATION"
```

The format of the EXEC statement is:

```
//SGXX EXEC PGM=program[,COND=condition][,PARM=value]
```

where

SGXX
: is the step name. XX represents sequential identification numbers supplied by the system generation process. The stepname is printed in the IEF236I allocation message while the step is being initiated. For example, the message

  IEF236I ALLOC. FOR SYSGEN SG7

  indicates that the seventh step is being executed.

PGM
: indicates the name of the program being executed. The names are ASMBLR, IEWL, IEHMOVE, IEBCOPY, IEHIOSUP, IFCDIP00, and IEHLIST, and the programs are executed in this order. As shown in Figure 16, the assembler (ASMBLR) and the linkage editor (IEWL) are executed several times. The first IEWL step builds the new SYS1.NUCLEUS.

COND
: The ASSMBLR, IEWL, and IEHMOVE steps have a COND parameter that enables them to test whether the previous step was successfully completed. If the previous step was unsuccessful, the remaining steps are bypassed and Stage II is terminated.

PARM
: This parameter is supplied for job steps that require PARM information.

During the ASMBLR steps, selected modules are assembled and stored in the utility data set defined by the OBJPDS DD statement in Stage I. These modules and other modules from SYS1.MODLIB are processed during the IEWL steps to form load modules. These load modules are placed in the new SYS1.NUCLEUS, SYS1.SVCLIB, SYS1.LINKLIB, SYS1.ALGLIB, SYS1.COBLIB, SYS1.FORTLIB, and SYS1.TELCMLIB.

The IEHMOVE utility program copies load modules to the new SYS1.SVCLIB, SYS1.LINKLIB, SYS1.COBLIB, SYS1.FORTLIB, SYS1.PL1LIB, and SYS1.SORTLIB.

The IEBCOPY utility program copies SYS1.MACLIB to the new system. The IEBCOPY step is produced only if the MACLIB macro instruction was specified in Stage I.

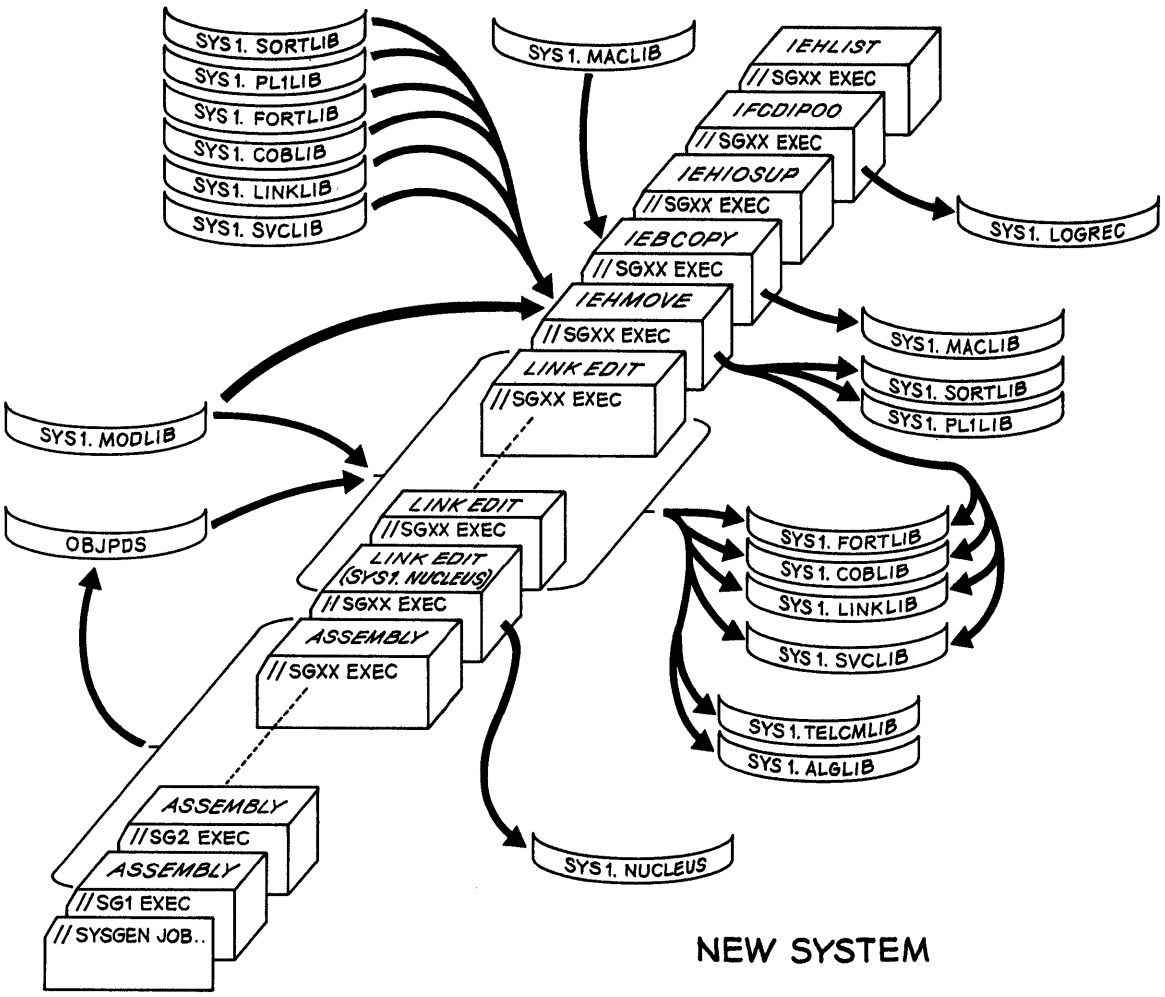


Figure 16. The Job Stream

The IEHIOSUP utility program builds the XCTL tables for type 4 SVCs in SYS1.SVCLIB. The IFCDIP00 utility program initializes SYS1.LOGREC. The IEHIOSUP and IFCDIP00 steps are produced only for complete Operating System generations.

The IEHLIST utility program lists the catalog of the new system residence volume and any other data specified with the DIRDATA parameter of the GENERATE macro instruction. The IEHLIST step is not produced during a NUCLEUS generation.

Figure 17 shows sample job control language statements for each type of step in the job stream. The values selected for the parameter result from the specifications in the system generation macro-instructions. In Figure 17, the underlined macro-instruction keywords are used to show where the value indicated by those keywords is placed. These keywords are from the GENERATE macro-instruction unless otherwise indicated by a comment. Comments (preceded by ...) do not appear in the statements.

```
                              Assembler
//SGXX EXEC PGM=ASMBLR,COND=(4,LT)
//SYSLIB DD DSNAME=SYS1.GENLIB,DISP=(OLD,PASS)
//     DD DSNAME=SYS1.MACLIB,DISP=OLD,VOLUME=(,RETAIN)
//SYSUT1 DD DISP=OLD,VOLUME=(,RETAIN),LABEL=(,UT1SDS),DSNAME=UT1SDS
//SYSUT2 DD DISP=OLD,VOLUME=(,RETAIN),LABEL=(,UT2SDS),DSNAME=UT2SDS
//SYSUT3 DD DISP=OLD,VOLUME=(,RETAIN),DSNAME=UT3SDS
//SYSPRINT DD SYSOUT=A
//SYSPUNCH DD DISP=OLD,VOLUME=(,RETAIN),DSNAME=OBJPDS(member)
//SYSIN DD *
      PRINT ON,NODATA
                            Linkage Editor
//SGXX EXEC PGM=IEWL,PARM='NCAL,XREF,LIST,LET',COND=(8,LT)
//SYSUT1 DD DISP=OLD,VOLUME=(,RETAIN),DSNAME=UT3SDS
//SYSPRINT DD SYSOUT=A
//SYSLMOD DD DISP=OLD,UNIT=RESNAME,VOLUME=SER=RESVOL,                  X
//           DSNAME=SYS1.name(member)
//MODLIB DD DISP=OLD,DSNAME=SYS1.MODLIB,VOLUME=(,RETAIN)
//SYSPUNCH DD DISP=OLD,VOLUME=(,RETAIN),DCB=(,RECFM=F,BLKSIZE=80),     X
//           DSNAME=OBJPDS
//RESLIB DD DISP=OLD,VOLUME=(,RETAIN),DSNAME=PDS .....RESMODS macro
//SYSLIN DD *
                              IEHMOVE
//SGXX EXEC PGM=IEHMOVE,PARM='POWER=1',COND=(8,LT)
//SYSUT1 DD DISP=OLD,VOLUME=(,RETAIN),DSNAME=UT3SDS
//SYSPRINT DD SYSOUT=A
//FROMLIB DD DISP=OLD,DSNAME=SYS1.MODLIB
//TOLIB1 DD DISP=OLD,VOLUME=(,RETAIN,SER=RESVOL),                      X
//           UNIT=RESNAME
//TOLIB2 DD DISP=OLD,VOLUME=(,RETAIN,SER=LNKVOL),                      X
//           UNIT=LNKNAME
//FRLNK DD DISP=OLD,VOLUME=RETAIN,DSNAME=PDS .........LINKLIB macro
//FRSVC DD DISP=OLD,VOLUME=(,RETAIN),DSNAME=PDS .........SVCLIB macro
//SYSIN DD *
                              IEHIOSUP
//SGXX EXEC PGM=IEHIOSUP
//SYSUT1 DD DSNAME=SYS1.SVCLIB,DISP=(OLD,KEEP),                        X
//           UNIT=RESNAME,VOLUME=SER=RESVOL
//SYSPRINT DD SYSOUT=A
                              IFCDIP00
//SGXX EXEC PGM=IFCDIP00,PARM=value
//SERERDS DD DSNAME=SYS1.LOGREC,UNIT=RESNAME,DISP=(NEW,KEEP),          X
//             VOLUME=SER=RESVOL,SPACE=(allocation)
                              IEHLIST
//SGXX EXEC PGM=IEHLIST
//LINK DD DISP=OLD,VOLUME(,RETAIN,SER=LNKVOL),UNIT=LNKNAME
//SYSRES DD DISP=OLD,VOLUME=(,RETAIN,SER=RESVOL),UNIT=RESNAME
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
```

Figure 17. Sample Steps in the Job Stream

RESTART TECHNIQUES

Stage II can be restarted at the beginning of any job step. If any statements in the job stream are to be changed, the job stream must be in cards. If no statements are to be changed, the IEBEDIT utility program can be used to restart a job stream on tape. Certain operations may have to be performed before restarting Stage II. (These operations are discussed in the section "Guidelines for Restarting Stage II.") This section discusses the techniques used for restarting the job stream after any other necessary operations have been performed. The following topics describe restarting from cards, punching the job stream, and restarting from tape.

Restarting From Cards

If the job stream is on cards, a job step can be restarted by placing a JOB card ahead of the step's EXEC card. A START RDR command must then be issued for the card reader.

Punching the Job Stream

If the unit (SYSPUNCH) specified for the job stream during Stage I was not a card punch, the IEBPTPCH utility program can be used to punch the job stream. IEBPTPCH requires the following statements:

```
//PUNCH    JOB                -PUNCH JOB STREAM-
//         EXEC PGM=IEBPTPCH
//SYSUT1   DD   (Parameters of SYSPUNCH DD statement for Stage I)
//SYSUT2   DD   (Parameters designating a card punch)
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
      PUNCH    TYPORG=PS,MAXFLDS=1
      RECORD   FIELD=80
/*
```

where:

- The parameters of the SYSUT1 DD statement must be the same as those of the SYSPUNCH DD statement of Stage I.
- Sequence numbers can be specified for the punched cards using the CDSEQ and CDINCR parameters in the PUNCH utility statement.

The assembly listing produced (SYSPRINT) at the end of Stage I contains a series of PUNCH statements. The operands of these PUNCH statements are the cards of the job stream. If a separate listing of the job stream is desired, the IEBPTPCH utility program can be used to print the job stream.

Figure 17.1 shows a series of statements that can be used to punch the job stream on any 2540 card read punch. The SYSPUNCH DD statement in the Stage I input deck was: //SYSPUNCH DD UNIT=182,LABEL=(,NL)

```
//PUNCH    JOB                -PUNCH JOB STREAM-
//         EXEC PGM=IEBPTPCH
//SYSUT1   DD   UNIT=182,LABEL=(,NL)
//SYSUT2   DD   UNIT=2540-2
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
      PUNCH    TYPORG=PS,MAXFLDS=1
      RECORD   FIELD=80
/*
```

Figure 17.1 Punching the Job Stream

Restarting From Tape

The IEBEDIT utility program can be used to restart Stage II when the job stream is on tape. IEBEDIT should be used to restart from any job step after the first. To restart from the first step, issue a START RDR command for the tape drive that contains the job stream.

IEBEDIT can be used to create a new job stream. The new job stream contains one of the following arrangements:

1. the job step specified by the user and all the steps that follow it; or
2. only those job steps specified by the user; or
3. all job steps except those specified by the user.

IEBEDIT requires the following control statements:

```
//RESTART  JOB                -RESTART STAGE II-
//         EXEC PGM=IEBEDIT
//SYSPRINT DD  SYSOUT=A
//SYSUT1   DD  (parameters of SYSPUNCH DD statement for Stage I)
//SYSUT2   DD  UNIT=XXX,LABEL=(,NL)
//SYSIN    DD  *
   EDIT    START=SYSGEN,STEPNAME=SGXX[,NOPRINT]
or EDIT    START=SYSGEN,TYPE=INCLUDE,STEPNAME=(SGXX[,SGXX]...)[,NOPRINT]
or EDIT    START=SYSGEN,TYPE=EXCLUDE,STEPNAME=(SGXX[,SGXX]...)[,NOPRINT]
/*
```

where:

- The parameters of the SYSUT1 DD statements must be those of the SYSPUNCH DD statement in the input deck for Stage I.
- The value of the UNIT parameter of the SYSUT2 DD statement is the specific unit name (address) of a magnetic tape drive.
- Only one EDIT statement must be used.
- If the TYPE parameter is omitted, STEPNAME specifies the first job step to be placed in the new job stream.
- If TYPE=INCLUDE or TYPE=EXCLUDE is specified, STEPNAME specifies the job steps to be included or excluded, respectively, from the new job stream. Individual job steps and sequences of job steps can be specified for inclusion or exclusion. For example:

  ```
  STEPNAME=(SG20,SG32-SG37,SG50)
  ```

  indicates that steps SG20, SG32 through and including SG37, and SG50 are to be included or excluded from the operation.
- NOPRINT must be included if a listing of the new job stream is not desired. After the new job stream is created, a START RDR command must be issued for the magnetic tape drive designated by the SYSUT2 DD statement.

Refer to the publication IBM System/360 Operating System: Utilities for a more detailed description of IEBEDIT.

Figure 17.2 shows an IEBEDIT input deck for restarting Stage II. The job stream resides on unit 182. The new job stream will reside on unit 282. All jobsteps after and including SG7 are to be restarted. A

listing of the new job stream is not desired. The START RDR command to start the new job stream is included with the IEBEDIT deck. The appropriate operations described in "Guidelines for Restarting Stage II" must have been performed.

```
//RESTART  JOB           -RESTART STAGE II-
//         EXEC PGM=IEBEDIT
//SYSPRINT DD  SYSOUT=A
//SYSUT1   DD  UNIT=182,LABEL=(,NL)
//SYSUT2   DD  UNIT=282,LABEL=(,NL)
//SYSIN    DD  *
      EDIT     START=SYSGEN,STEPNAME=SG7,NOPRINT
/*
//    START    RDR,282
```

Figure 17.2 Input deck for IEBEDIT

GUIDELINES FOR RESTARTING STAGE II

The following topics give guidelines for restarting during Stage II. Restarting may require the scratching and reallocation of data sets. When this is necessary, reference is made to the section "Reallocating Data Sets" for the procedure to be followed. After these operations have been performed, the actual restarting of Stage II can be accomplished by one of the methods described in "Restart Techniques."

Restarting During Assemblies

To restart at the beginning of any assembly, place the JOB card in front of the EXEC statement of that assembly. If more than one ASMBLR step has been executed, and if it is desired to restart at the beginning of Stage II (SG1 step), the data set defined by the OBJPDS DD statement must be reallocated (see "Reallocating Data Sets").

All assemblies must be satisfactorily executed before the link edit (IEWL) steps can be executed.

Restarting Link Edit Steps

The first IEWL step builds the new SYS1.NUCLEUS. Subsequent steps produce load modules for the new SYS1.SVCLIB, SYS1.LINKLIB, and, if specified for the new system, SYS1.ALGLIB, SYS1.COBLIB, SYS1.FORTLIB, and SYS1.TELCMLIB.

If only the first IEWL step was executed, SYS1.NUCLEUS must be reallocated (see "Reallocating Data Sets") before restarting the step.

To restart at the beginning of the second IEWL step, any of the following libraries that contain system data must be reallocated: SYS1.SVCLIB, SYS1.LINKLIB, SYS1.ALGLIB, SYS1.COBLIB, SYS1.FORTLIB, and SYS1.TELCMLIB. If more than one IEWL step was executed, and if it is desired to restart with the first IEWL step, SYS1.NUCLEUS must also be reallocated.

If reallocation is not performed, the space already used in the new system libraries is not available, and therefore there may not be enough space available for these data sets. This is particularly important for those system data sets that cannot have a secondary allocation. If more than sufficient space was allocated to a data set during the preparation for system generation, the IEWL step that failed can be restarted without reallocation. It is recommended that the reallocations indicated above be performed when restarting from the beginning of the first or second IEWL steps. To restart from the beginning of Stage II, the data set defined by the OBJPDS DD statement must also be reallocated.

Restarting IEHMOVE Step

The IEHMOVE utility program copies members to the new SYS1.SVCLIB, SYS1.LINKLIB, SYS1.COBLIB, SYS1.FORTLIB, SYS1.PL1LIB, and SYS1.SORTLIB. No reallocation is needed when restarting this step, unless the step failed because there was not enough space in a system data set. Only the system data set that did not have enough space must be reallocated (see "Reallocating Data Sets"), because any member that existed in the new system data set will not be moved into it again when the step is restarted.

If the data set that did not have enough space was SYS1.SVCLIB, SYS1.LINKLIB, SYS1.COBLIB, or SYS1.FORTLIB, one of the two following procedures must be performed.

- Follow the procedures for restarting with the second IEWL step as described in "Restarting Link Edit Steps," or
- Copy the data set to a scratch volume using the IEBCOPY utility program. Scratch and uncatalog the space for the data set on its original volume with IEHPROGM. Allocate and catalog the required space for the data set on its original volume. Use IEBCOPY to copy the data set from the scratch volume to its original volume. Restart Stage II at the beginning of the IEHMOVE step.

Restarting IEBCOPY Step

This step copies SYS1.MACLIB to the new system. To restart at the beginning of this step, the new SYS1.MACLIB must be reallocated (see "Reallocating Data Sets"). To restart at any preceding step, follow the instructions in the preceding topics.

Restarting IEHIOSUP, IFCDIP00, and IEHLIST

These three steps can be executed in any order. IEHIOSUP and IEHLIST can be restarted any number of times without need of reallocation of system data sets. Before restarting IFCDIP00, the DISP=(NEW,KEEP) parameter of its DD statement for SYS1.LOGREC must be changed to DISP=(OLD,KEEP), or SYS1.LOGREC must be scratched (the IFCDIP00 step does its own allocation for SYS1.LOGREC).

The ASMBLR, IEWL, IEHMOVE, and IEBCOPY steps must have been executed without error before the IEHIOSUP step is executed.

REALLOCATING DATA SETS

Reallocating data sets includes the following operations:

- Scratching and uncataloging the space allocated to the data set during the preparation for system generation.
- Allocating new space to the data set and cataloging it.

The following topics discuss the reallocation of the utility data set defined by the OBJPDS DD statement during Stage I, reallocating new system data sets using the same amount of space, and reallocating system data sets using more space.

The reallocations described in this section may alter the order in which the new system data sets were originally allocated. If this order is important, the IEHMOVE utility program can be used after system generation to rearrange the new system data sets.

Reallocation of OBJPDS

Use the following statements to reallocate the utility data set specified by the OBJPDS DD statement during Stage I.

```
//OBJPDS     JOB
//STEP1      EXEC   PGM=IEHPROGM -SCRATCH OBJPDS-
//OBJPDS     DD     DSNAME=SYS1.name,DISP=(OLD,DELETE)
//SYSPRINT   DD     SYSOUT=A
//SYSIN      DD     DUMMY
//STEP2      EXEC   PGM=IEHPROGM - REALLOCATE OBJPDS-
//OBJPDS     DD     (Parameters for OBJPDS in Stage I input deck)
//SYSPRINT   DD     SYSOUT=A
//SYSIN      DD     DUMMY
//
```

In the preceding statements:

- The DSNAME parameter of the first OBJPDS DD statement must contain the name given to the utility data set in the input deck to Stage I.
- The second OBJPDS DD statement must contain the same parameters that it contains in the Stage I input deck.

Reallocating on the Same Space

The input deck for scratching and reallocating space to the new system data sets must contain the following statements. The statements must be in the order shown.

1. JOB statement.
2. EXEC statement with the PGM=IEHPROGM parameter.
3. A SYSPRINT DD statement defining the system output unit.
4. A DD statement defining the unit and serial number of the generating system residence volume:

   ```
   //GENRES DD UNIT=unit,VOLUME=SER=serial,DISP=OLD
   ```

5. A DD statement defining any other permanent volume on which data sets to be reallocated reside:

   ```
   //ddname DD UNIT=unit,VOLUME=SER=serial,DISP=OLD
   ```

6. A DD statement for each type of removable device on which data sets to be reallocated reside:

```
//ddname DD UNIT=(unit,,DEFER),VOLUME=PRIVATE,DISP=OLD
```

7. A DD * statement (SYSIN).

8. A SCRATCH and an UNCATLG statement for each new system data set to be reallocated. The SCRATCH and UNCATLG statements must have the following format:

```
SCRATCH  DSNAME=dsname,VOL=device=serial,PURGE
UNCATLG  DSNAME=dsname,CVOL=device=serial
```

   where CVOL designates the new system residence volume.

9. A /* statement.

10. EXEC statement with the PGM=IEHPROGM parameter.

11. A GENRES DD statement (described above).

12. A DD statement for each permanent device (described above).

13. A DD statement for each type of removable device (described above).

14. A SYSPRINT DD statement defining the system output unit.

15. A DD statement for each of the new system data sets to be reallocated. This DD statement must be the same used in the input deck for the preparation for system generation.

16. A DD * statement (SYSIN).

17. A CATLG statement for each new system data set to be reallocated. The CATLG statement must be the same used in the input deck for the preparation for system generation.

18. A /* statement.

For example, SYS1.SVCLIB and SYS1.LINKLIB must be reallocated. The following DD statements defined these two libraries during the preparation for system generation:

```
//SVCLIB   DD  DSNAME=SYS1.SVCLIB,VOLUME=(,RETAIN,SER=AAA111),          X
//             UNIT=2301,DISP=(,KEEP),SPACE=(TRK,(40,,75),,CONTIG),     X
//             LABEL=EXPDT=99350,DCB=(DSORG=POU,RECFM=U,BLKSIZE=1024)

//LINKLIB  DD  DSNAME=SYS1.LINKLIB,VOLUME=(,RETAIN,SER=AAA112),         X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(1250,,100),,CONTIG), X
//             LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=3625)
```

The generating system residence volume is a 2311 volume whose serial number is 111111. Figure 18 shows the input deck for this reallocation.

```
//SAME     JOB                          -REALLOCATE ON SAME SPACE-
//STEP1    EXEC PGM=IEHPROGM                 -SCRATCH-
//GENRES   DD   UNIT=2311,VOLUME=SER=111111,DISP=OLD
//NEWRES   DD   UNIT=2301,VOLUME=SER=AAA111,DISP=OLD
//LINVOL   DD   UNIT=(2311,,DEFER),VOLUME=PRIVATE,DISP=OLD
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
      SCRATCH  DSNAME=SYS1.SVCLIB,VOL=2301=AAA111
      UNCATLG  DSNAME=SYS1.SVCLIB,CVOL=2301=AAA111
      SCRATCH  DSNAME=SYS1.LINKLIB,VOL=2311=AAA112
      UNCATLG  DSNAME=SYS1.LINKLIB,CVOL=2301=AAA111
/*
//STEP2    EXEC PGM=IEHPROGM                 -ALLOCATE-
//GENRES   DD   UNIT=2311,VOLUME=SER=111111,DISP=OLD
//NEWRES   DD   UNIT=2301,VOLUME=SER=AAA111,DISP=OLD
//LINVOL   DD   UNIT=(2311,,DEFER),VOLUME=PRIVATE,DISP=OLD
//SYSPRINT DD   SYSOUT=A
//SVCLIB   DD   DSNAME=SYS1.SVCLIB,VOLUME=(,RETAIN,SER=AAA111),        X
//             UNIT=2301,DISP=(,KEEP),SPACE=(TRK,(40,,75),,CONTIG),   X
//             LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=1024,DSORG=POU)
//LINKLIB  DD   DSNAME=SYS1.LINKLIB,VOLUME=(,RETAIN,SER=AAA112),       X
//             UNIT=2311,DISP=(,KEEP),                                X
//             SPACE=(TRK,(1250,,100),,CONTIG),LABEL=EXPDT=99350,     X
//             DCB=(RECFM=U,BLKSIZE=3625)
//SYSIN    DD   *
      CATLG    DSNAME=SYS1.SVCLIB,CVOL=2301=AAA111,VOL=2301=AAA111
      CATLG    DSNAME=SYS1.LINKLIB,CVOL=2301=AAA111,VOL=2311=AAA112
/*
//
```

Figure 18. Reallocating on Same Space

Reallocating With More Space

The method for reallocating with more space depends on whether the space for the data set must be contiguous. If the data set need not be contiguous, follow the procedure described in "Reallocating on the Same Space" changing the SPACE parameter of the DD statement (statement 15) for the new system data set in the second step. This same procedure can be followed for a contiguous data set if there is enough contiguous space for the new allocation elsewhere on the volume.

The second case can best be illustrated with the following example. One of the new system volumes is organized as shown in Figure 19.

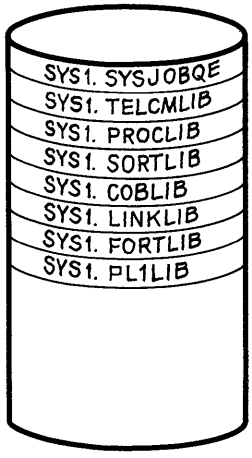


Figure 19. Reallocation on Same Volume

Enough space was not allocated for SYS1.LINKLIB and there is not enough contiguous space after SYS1.PL1LIB for a new allocation of SYS1.LINKLIB. One of two procedures can be followed to free space for the new allocation for SYS1.LINKLIB.

1. If there is no system data in SYS1.LINKLIB, SYS1.FORTLIB, and SYS1.PL1LIB, follow the procedures described in "Reallocating in the Same Space" to reallocate SYS1.LINKLIB, SYS1.FORTLIB, and SYS1.PL1LIB. Correct the SPACE parameter in the DD statement for SYS1.LINKLIB.

2. If there is system data in SYS1.LINKLIB, SYS1.FORTLIB, or SYS1.PL1LIB follow these procedures:

   a. Copy the library that contains system data onto a scratch volume using the IEBCOPY utility program.
   b. Follow the procedures described in "Reallocating in the Same Space" to scratch SYS1.LINKLIB, SYS1.FORTLIB, and SYS1.PL1LIB, correcting the SPACE parameter in the DD statement for SYS1.LINKLIB.
   c. Copy the library that contains system data from the scratch volume to its original volume.

If a system data set (contiguous or not) contains system data, and if there is enough space elsewhere in the volume, the following procedure can be used:

1. Rename the system data set.

2. Allocate and catalog space for the system data set (with its correct name) on the same volume.

3. Copy the data in the renamed data set onto the newly allocated data set.

4. Scratch and uncatalog the renamed data set.

Figure 20 is an example of reallocation on the same volume of a system data set that contains system data. The system data set to be reallocated is SYS1.LINKLIB. It was allocated space during the preparation for system generation with the following DD statement:

```
//LINKLIB   DD  DSNAME=SYS1.LINKLIB,VOLUME=(,RETAIN,SER=LINVOL),        X
//              UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(500,,50),,CONTIG),  X
//              LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=3625)
```

The new system residence volume is a 2301 volume whose serial number is SYSTEM. The renamed SYS1.LINKLIB will be called SYS1.HARRY. The generating system residence volume is a 2311 volume whose serial number is 111111.

```
//MOVE     JOB                          -REALLOCATE WITH DATA-
//STEP1    EXEC PGM=IEHPROGM                 -RENAME-
//SYSPRINT DD   SYSOUT=A
//GENRES   DD   UNIT=2311,VOLUME=SER=111111,DISP=OLD
//NEWRES   DD   UNIT=2301,VOLUME=SER=SYSTEM,DISP=OLD
//LINVOL   DD   UNIT=(2311,,DEFER),VOLUME=PRIVATE,DISP=OLD
//SYSIN    DD   *
       RENAME   DSNAME=SYS1.LINKLIB,VOL=2311=LINVOL,                    X
                NEWNAME=SYS1.HARRY
       UNCATLG  DSNAME=SYS1.LINKLIB,CVOL=2301=SYSTEM
       CATLG    DSNAME=SYS1.HARRY,CVOL=2301=SYSTEM,VOL=2311=LINVOL
/*
//STEP2    EXEC PGM=IEHPROGM,                -REALLOCATE-
//SYSPRINT DD   SYSOUT=A
//GENRES   DD   UNIT=2311,VOLUME=SER=111111,DISP=OLD
//NEWRES   DD   UNIT=2301,VOLUME=SER=SYSTEM,DISP=OLD
//LINVOL   DD   UNIT=(2311,,DEFER),VOLUME=PRIVATE,DISP=OLD
//LINKLIB  DD   DSNAME=SYS1.LINKLIB,VOLUME=(,RETAIN,SER=LINVOL),       X
//             UNIT=2311,DISP=(,KEEP),                                 X
//             SPACE=(TRK,(1200,,100),,CONTIG),LABEL=EXPDT=99350,      X
//             DCB=(RECFM=U,BLKSIZE=3625),
//SYSIN    DD   *
       CATLG    DSNAME=SYS1.LINKLIB,CVOL=2301=SYSTEM,VOL=2311=LINVOL
/*
//STEP3    EXEC PGM=IEBCOPY                  -COPY-
//SYSPRINT DD   SYSOUT=A
//SYSUT1   DD   DSNAME=SYS1.HARRY,VOLUME=SER=LINVOL,UNIT=2311,         X
//             DCB=(RECFM=U,BLKSIZE=3625)
//SYSUT2   DD   DSNAME=SYS1.LINKLIB,VOLUME=SER=LINVOL,UNIT=2311,       X
//             DCB=(RECFM=U,BLKSIZE=3625)
//SYSIN    DD   DUMMY
//STEP4    EXEC PGM=IEHPROGM                 -SCRATCH-
//SYSPRINT DD   SYSOUT=A
//GENRES   DD   UNIT=2311,VOLUME=SER=111111,DISP=OLD
//NEWRES   DD   UNIT=2301,VOLUME=SER=SYSTEM,DISP=OLD
//LINVOL   DD   UNIT=(2311,,DEFER),VOLUME=PRIVATE,DISP=OLD
//SYSIN    DD   *
       SCRATCH  DSNAME=SYS1.HARRY,VOL=2311=LINVOL
       UNCATLG  DSNAME=SYS1.HARRY,VOL=2301=SYSTEM
/*
//
```

Figure 20. Reallocate Data Set With System Data

REALLOCATING SYS1.SYSJOBQE

If the job stream is too large for the SYS1.SYSJOBQE of the generating system one of the following procedures can be followed:

- Converting the job stream into two or more jobs
- Enlarging SYS1.SYSJOBQE

The job stream can be converted into two or more jobs by placing additional JOB cards throughout the job stream. For example, inserting a JOB card in front of the first IEWL step. This method requires that the job stream be in cards.

The size of SYS1.SYSJOBQE can be changed following this procedure:

1. Determine the appropriate size of SYS1.SYSJOBQE using the formula given in the publication IBM System/360 Operating System: Storage Estimates.

2. Run the following job:

```
//LASTJOB  JOB
//STEP     EXEC  PGM=IEHPROGM
//DUMMY     DD   DSNAME=SYS1.DUMMY,VOLUME=(,RETAIN,SER=serial),          X
//              UNIT=unit,DISP=(,CATLG),SPACE=(TRK,(xxx),,CONTIG)
//GENRES    DD   UNIT=unit,VOLUME=SER=serial,DISP=OLD
//SYSPRINT DD    SYSOUT=A
//SYSIN     DD   *
      SCRATCH    DSNAME=SYS1.SYSJOBQE,VOL=unit=serial
      UNCATLG    DSNAME=SYS1.SYSJOBQE
      RENAME     DSNAME=SYS1.DUMMY,VOL=unit=serial,                    X
                 NEWNAME=SYS1.SYSJOBQE
      CATLG      DSNAME=SYS1.SYSJOBQE,VOL=unit=serial
/*
//
```

In the preceding statements,

- SYS1.DUMMY is allocated space on the same volume on which the generating SYS1.SYSJOBQE resides. The amount of space allocated for SYS1.DUMMY must be the appropriate size for SYS1.SYSJOBQE.
- The GENRES DD statement defines the generating system residence volume.
- The UNCATLG and CATLG statements should be omitted if SYS1.SYSJOBQE is not cataloged in the generating system.
- The VOL parameter of the utility statements defines the volume on which SYS1.SYSJOBQE resides.

3. Stop the system and restart (IPL again).

TESTING THE NEW SYSTEM

This section describes the sample programs provided by IBM to test the functioning of various components of the new system after system generation. The sample programs are contained in SYS1.SAMPLIB of the starter operating system package. Appendix E of this publication describes the procedure used to punch sample program cards from SYS1.SAMPLIB. These card decks can then be used whenever an operating system component is to be tested.

The following list shows the names of the sample programs provided in SYS1.SAMPLIB and the components they test:

Program	Component
IEXSAMP	ALGOL
IETESP	Asembler E and F
IEPSAMP	COBOL E
IEQSAMP	COBOL F
IEJESP	FORTRAN E
IEYSP	FORTRAN G and H
SAMP2250	Graphics - 2250 Display Units
SAMP2260	Graphics - 2260 Display Stations
GSPSAMP	Graphic Subroutine Package for FORTRAN IV
IEMSP2	PL/I F
RPGSMPL	RPG
IERSP	Sort/merge
IHGSAMP	Update Analysis Program

Each of the following pages contains the description of a sample program, operating instructions, and a description of the program execution results. More detailed operating instructions can be found in the publication IBM System/360 Operating System: Operator's Guide.

Note: The DD statements of the sample programs specify UNIT=2311. If the system data sets affected by the sample program do not reside on a 2311, the UNIT parameter should be changed accordingly. If the system data set is cataloged, the UNIT parameter of the corresponding DD statement can be deleted rather than changed.

ALGOL SAMPLE PROGRAM (IEXSAMP)

The IEXSAMP card deck (punched from SYS1.SAMPLIB) consists of:

1. Job control language statements for an ALGOL compilation, a linkage edit, and execution.

2. ALGOL sample program source statements.

The sample program generates the first twenty lines of Pascal's Triangle. Comments included in the program may be used for checking the results. (The sample program is included as example 3 in IBM System/360 Operating System: ALGOL Language, Form C28-6615.)

OPERATING INSTRUCTIONS

1. Mount the operating system and an initialized scratch pack.

2. Set the load address switches and press the Initial Program Load key to load the operating system.

3. Place the IEXSAMP card deck in the card reader, ready the reader, and press the End of File key.

4. Ready the printer.

5. Execute the job.

OUTPUT

The normal output from the compiler will be:

1. A list of all job control language statements that apply to the compiler.

2. A list of all source statements supplemented by a count of the semicolons occurring in the program.

3. The identifier table in symbolic form, giving details of all identifiers used in the program.

4. Information on main storage requirements at execution time.

The normal output from the linkage editor will be:

1. A list of all job control language statements which apply to the linkage editor.

2. A cross-reference table of the load module.

The normal output at execution time will be:

1. A list of all job control language statements which apply to the execution.

2. The first twenty lines of Pascal's triangle, corresponding to the comments included in the source statements.

A more detailed description of the system output is contained in IBM System/360 Operating System: ALGOL Programmer's Guide, Form C33-4000.

ASSEMBLER E & F SAMPLE PROGRAM (IETESP)

The IETESP card deck (punched from SYS1.SAMPLIB) is the sample program source deck.

The sample problem demonstrates the use of the assembler and serves as a minimal test of the functioning of the assembler. In addition, it provides sample coding that demonstrate the definition and use of user written macro-instructions, the calling of system macro-instructions, and the proper method of saving and restoring registers upon entry and exit from a problem program.

The input is assembled into the program in the form of a TABLE and a LIST of entries which are to be passed against the table. Each item in the table contains an argument name such as ALPHA and space in which information concerning that name is to be placed. Each entry in the LIST contains an argument name and function values. The formats of the TABLE entries and the LIST entries are different, and both formats are described by means of DSECTs. The program searches the TABLE for an argument name in the list. If a match is found, the function values are reformatted and moved to the appropriate TABLE entry. If an argument name in the LIST cannot be found in the TABLE, a switch is set in the LIST entry. After all LIST entries have been processed, both the LIST and TABLE areas are compared with TESTTABL which contains the predefined results. If the comparison is equal, the routine executed properly and a message is written to indicate this.

Use of the program for Assembler E may be achieved by calling the IBM supplied cataloged procedure ASMECLG in the following manner:

```
//jobname     JOB
//stepname    EXEC    PROC=ASMECLG
//ASM.SYSIN   DD      *
Sample Program Source Deck
/*
```

The program for Assembler F may be called by substituting the following execute card in the above deck:

```
//stepname    EXEC    PROC=ASMFCLG
```

This procedure calls for an assembly, link edit, and execution of the sample program. A more detailed explanation of the contents of the procedure and use of the assembly program is given in IBM System/360 Operating System: Assembler (E) Programmers Guide, Form C28-6595 and in IBM System/360 Operating System: Assembler (F) Programmers Guide, Form C26-3756.

OPERATING INSTRUCTIONS

1. Mount the operating system and an initialized scratch pack.
2. Set the load address switches and press the Initial Program Load key to load the operating system.
3. Place the sample program source deck in card reader, ready reader, and press the End of File key.
4. Ready the printer.
5. Execute the job.

OUTPUT

The check for the successful execution of the sample program is a message on the operator's console. The message says either, "ASSEMBLER SAMPLE PROGRAM SUCCESSFUL" or "ASSEMBLER SAMPLE PROGRAM UNSUCCESSFUL." The program gives a normal return to the operating system with a return code of zero.

COBOL E SAMPLE PROGRAM (IEPSAMP)

The IEPSAMP card deck (punched from SYS1.SAMPLIB) consists of:

1. Job control language statements to execute the COBOL E cataloged procedure COBECLG to compile, link edit, and execute.

2. COBOL sample program source statements.

3. DD statements for tape data sets required at execution time.

The sample program tests the COBOL compiler's ability to WRITE to and READ from tape, checking proper data alignment where data is a mixture of DISPLAY COMPUTATIONAL and COMPUTATIONAL-3.

The verbs exercised in this program are OPEN, CLOSE, READ, WRITE, PERFORM, IF, MOVE, GO TO, DISPLAY, ADD. There are 575 source statements.

OPERATING INSTRUCTIONS

1. Mount the operating system and an initialized scratch pack.

2. Mount unlabeled tapes on 182, 183, and 282.

3. Set the load address switches and press the Initial Program Load key to load the operating system.

4. Place the IEPSAMP card deck in the card reader, ready reader, and press End of File key.

5. Ready the printer.

6. Execute the job.

OUTPUT

1. The scheduler will read, process, and print (on the device specified as SYSOUT by the operator) all job control language statements.

2. The compiler will read and list the source program on the device specified as SYSOUT by the operator.

3. The compiler will prepare and list (on SYSOUT)

   a. Heading including date and level
   b. The source program
   c. A data map of the Data Division
   d. An object code listing (PMAP) of the Procedure Division

4. The linkage editor will prepare and list (on SYSOUT).

   a. A module map
   b. A cross-reference list

5. The COBOL load module will list (on SYSOUT) the following:

   GROUP B LEVEL P TEST CASE 1
   END OF PROGRAM

COBOL F SAMPLE PROGRAM (IEJSP)

The IEQSAMP card deck (punched from SYS1.SAMPLIB) consists of

1. Job control language to call a cataloged procedure to compile, linkage edit, and execute the sample program.

2. COBOL sample program source statements.

The sample program tests the operation of the COBOL F compiler; it generates an output data set on tape and then reads, processes, and exhibits this data set. The program uses the COBOL verbs, IF, OPEN, READ, WRITE, CLOSE, DISPLAY, MOVE, ADD, STOP, GO TO, PERFORM, NOTE, TRACE, and EXHIBIT. A complete output listing of the program may be found in the publication IBM System/360 Operating System: COBOL (F) Programmer's Guide, Form C28-6380.

OPERATING INSTRUCTIONS

1. Mount the operating system.

2. Set the load address switches and press the Initial Program Load key to load the operating system.

3. Place the IEQSAMP deck in the card reader, ready reader, and press End of File key.

4. Ready the printer.

5. Execute the job.

OUTPUT

1. Listing of all job control language statements.

2. Listing of the source program.

3. Listing of
   a. heading including date and level
   b. source program
   c. data map of the Data Division
   d. object code listing of Procedure Division

4. Module map and cross-reference list.

FORTRAN E SAMPLE PROGRAM (IEJSP)

The IEJESP card deck (punched from SYS1.SAMPLIB) consists of:

1. Job control language statements for a FORTRAN E compilation.

2. FORTRAN sample program source statements.

3. Job control language statements for a link edit and execution of the sample program.

4. Data deck for FORTRAN sample program execution.

The sample program is a simultaneous equations routine which consists of 104 FORTRAN source statements:

```
 1    Specification
14    Format
17    Read/Write
27    Control
16    Arithmetic
29    Comments
```

The program processes eight data cards as its input. The comments cards show a complete list of the expected output from the execution of the program. This list may be used for checking output.

OPERATING INSTRUCTIONS

1. Mount the operating system and an initialized scratch pack.

2. Set the load address switches and press the Initial Program Load key to load the operating system.

3. Place the IEJESP card deck in the card reader, ready reader, and press the End of File key.

4. Ready the printer.

5. Execute the job.

OUTPUT

1. The scheduler will read, process, and print (on the device specified as SYSOUT by the operator) all job control statements.

2. The compiler will read and list the source program on the device specified as SYSOUT by the operator.

3. The compiler will prepare and list (on SYSOUT) the following:

   a. Message showing compiler options in effect
   b. Heading including date and level
   c. The source program
   d. Storage map including size of COMMON and size of program
   e. Message showing "END OF COMPILATION program-name"

4. The linkage editor will prepare and list (on SYSOUT) a storage map including the relative address of each external reference.

5. The FORTRAN load module will list (on SYSOUT) the results of execution. The results should correspond to the comments in the source program.

FORTRAN G AND H SAMPLE PROGRAM (IEYSP)

The IEYSP card deck (punched from SYS1.SAMPLIB) consists of

1. Job control language statements for a FORTRAN G or H compilation.
2. FORTRAN sample program source statements.
3. Job control language statements for a linkage edit and execution of the sample program.
4. Data card for FORTRAN sample program execution.

The sample program consists of one main program and one function subprogram, which together compute and print out binomial coefficients. The deck is composed of the following source statements:

	Main Program	Subprogram
Comments	42	19
Specification	1	1
Format	5	
Read/Write	5	
Control	8	13
Arithmetic	2	13
Total	63	46

The program processes one data card as its input. The comments cards show a complete list of the expected output from the execution of the program and may be used for checking the output.

OPERATING INSTRUCTIONS

1. Mount the operating system and an initialized scratch pack.
2. Set the load address switches and press the Initial Program Load key to load the operating system.
3. Place the IEYSP card deck in the card reader, ready reader, and press the End of File key.
4. Ready the printer.
5. Execute the job.

OUTPUT

1. The scheduler will read, process, and print (on the device specified as SYSOUT by the operator) all job control statements.
2. The compiler will read and list the source program on the device specified as SYSOUT by the operator.
3. The compiler will prepare and list (on SYSOUT) the following:
   a. Heading, including date and level
   b. The source program
   c. Storage map including size of COMMON and size of program
   d. The generated object code
4. The linkage editor will prepare and list (on SYSOUT) a storage map including the relative address of each external reference.
5. The FORTRAN load module will list (on SYSOUT) the results of execution. The results should correspond to the comments in the source program.

GRAPHICS SAMPLE PROGRAMS (SAMP2250 AND SAMP2260)

The SAMP2250 card deck (sample program to exercise the 2250 Display Unit) and the SAMP2260 card deck (sample program to exercise the 2260 Display Station, local) can be punched from SYS1.SAMPLIB.

Each card deck consists of

1. Appropriate job control language statements. Included is a cataloged procedure (ASMFCLG) to assemble, linkage edit, and execute the compiled program. Before compiling either deck, the four over-ride cards below must be added to the end of the deck after the assembler language END card.

   ```
   /*
   //LKED.SYSLIB DD DSNAME=SYS1.LINKLIB,DISP=OLD
   //GO.GRAPHIC DD UNIT=2250-1
   or //GO.GRAPHIC DD UNIT=(2260-1,2)
   /*
   ```

   In the preceding statements, the underlined values must be replaced with the appropriate generic unit name of the 2250 or 2260 being tested. (See Appendix C.) Note that the GO.GRAPHIC DD statement for the 2260 specifies a collection of two devices. (For further information on collections of devices, see the publication IBM System/360 Operating System: Job Control Language.)

2. Sample program input symbolic deck.

OPERATING INSTRUCTIONS

1. Insure that the graphic device is ON.
2. Place the sample program deck in the card reader.
3. Mount the operating system.
4. Set the load address switches and press the Initial Program Load key to load the system.
5. Perform assemble-link-edit-go.
6. Follow instructions which appear on the display screen.

OUTPUT

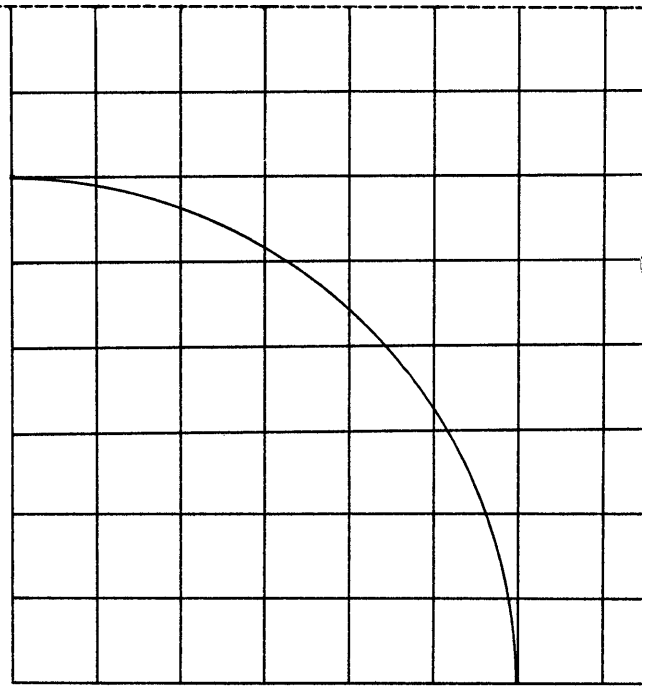
The displays which appear on the 2250 Display Unit are shown in Figure 21. The displays which appear on the 2260 Display Station are shown in Figure 22.

```
                    2250 SAMPLE PROGRAM - END ATTENTION TO BEGIN
```

a. First Display

```
THIS PROGRAM WILL PLOT A GRID WITH CORNERS AT 2400,2400-2400,4000-
4000,4000-4000,2400.  IT WILL PLOT 90 DEGREES OF ARC WHOSE RADIUS IS
1200 RASTER UNITS.  IT ILLUSTRATES THE USE OF ORDER MACROS TO CREATE
THIS DISPLAY AND PORS TO CREATE THE NEXT DISPLAY.

DEPRESS KEY D AND THEN THE END KEY ON THE ALPHAMERIC KEYBOARD TO
INITIATE THE NEXT DISPLAY.
```

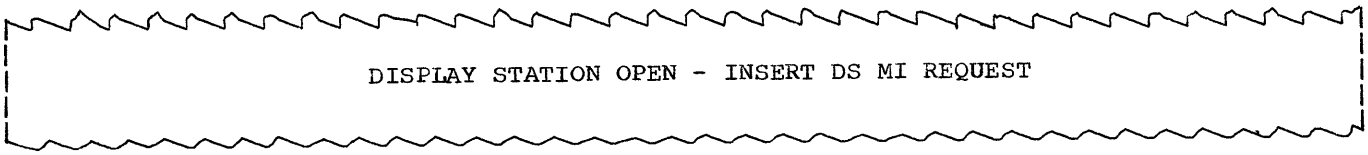
b. Second Display



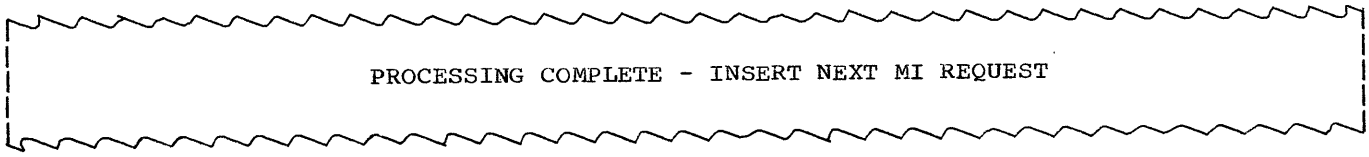
```
THIS DISPLAY ILLUSTRATES THE OUTPUT OF THE GCGRID, GARC,
AND GCPRNT PROBLEM ORIENTED ROUTINES.

DEPRESS KEY E AND THEN THE END KEY ON THE ALPHAMERIC KEYBOARD TO
TERMINATE THIS SAMPLE PROGRAM.
```

c. Third Display

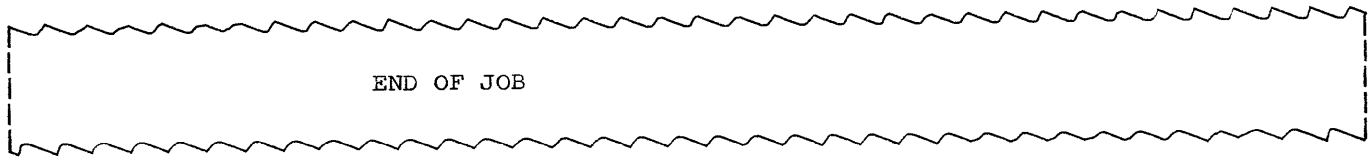
Figure 21. 2250 Displays



a. First display



b. Second display



c. Third display

Figure 22. 2260 Displays

GRAPHIC SUBROUTINE PACKAGE FOR FORTRAN IV SAMPLE PROGRAM (GSPSAMP)

The GSPSAMP card deck (punched from SYS1.SAMPLIB) consists of:

1. Appropriate job control language statements that will call a cataloged procedure (FORTxCLG), where x=E, G, or H, to compile, linkage edit, and execute the program. The following cards are required after the FORTRAN Language END card:

```
/*
//LKED.SYSIN DD *
    INCLUDE SYSLIB(IHCGSP03)
/*
//GO.SYSABEND DD SYSOUT=A
//GO.FT05F001 DD UNIT=AFF=FT01F001
//GO.FT10F001 DD UNIT=xxx (where xxx is the 2250 unit address)
/*
//SYSIN DD *
(Data Cards)
/*
```

   The //GO.FT05F001 DD statement must be omitted if FORTRAN G or H is used.

2. Sample program input symbolic deck.

OPERATING INSTRUCTIONS

1. Insure that the graphic device is on.

2. Place the sample program deck in the card reader.

3. Mount the operating system.

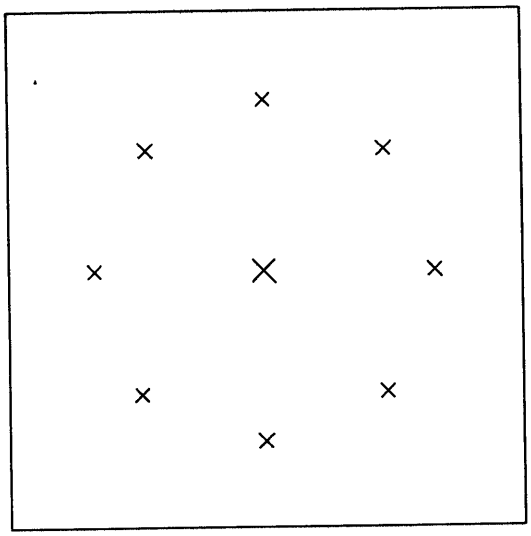
4. Set the load address switches and press the Initial Program Load key to load the system.

5. Perform compile-link edit-go procedure.

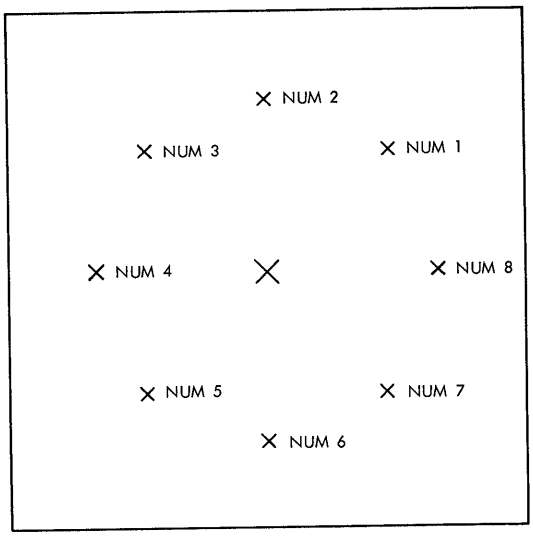
6. Follow instructions which appear with sample program in IBM System/360 Operating System: Graphic Programming Services for FORTRAN IV, Form C27-6932.

OUTPUT

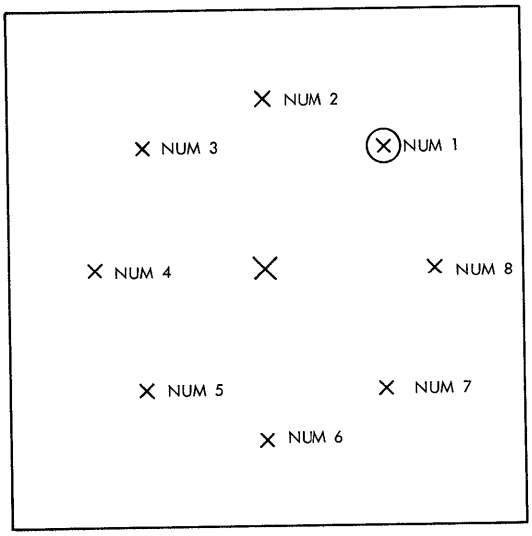
The displays which will appear on the 2250 Display Unit are shown in Figure 23.



A. Display Produces After Call to EXEC for IGDS1.



B. Display Produced After Light Pen Attention on Center X.



C. Display Produced After Light Pen Attention on X Labeled NUM1.

Figure 23. Display for GSP Sample Program

PL/I F SAMPLE PROGRAM (IEMSP2)

The IEMSP2 card deck (punched from SYS1.SAMPLIB) consists of

1. Job control language statements for a PL/I compilation, link edit, and execution of the compiled program. The cataloged procedure PL1LFCLG is used.

2. PL/I sample program source statements.

3. Input data.

The purpose of the sample program is to illustrate the use of record-oriented input/output, initialization of STATIC arrays, and tabulation of data-directed output. Statements used include PROCEDURE, DECLARE, BEGIN, END, ON, GOTO, READ, WRITE, PUT EDIT, and PUT DATA.

OPERATING PROCEDURES

1. Mount the operating system.

2. Set the load address switches and press the Initial Program Load key to load the operating system.

3. Place the IEMSP2 card deck in the card reader, ready the reader, and press End of File.

4. Ready printer.

5. Execute the job.

OUTPUT

1. Job control language statements in the cataloged procedure.

2. Normal compilation output:

   a. All source statements.

   b. An attribute and cross-reference listing.

   c. A list of errors found during the compilation.

3. Normal link edit step output.

4. The correct generated output is shown in Figure 24.

```
                    OUTPUT OF PL/1 SAMPLE PROGRAM

THIS CARD IGNORED BECAUSE TYPE IS NOT NUMERIC:=&   DELIBERATE DUD CARD

TYPE(0)= 1    TYPE(1)= 0      TYPE(2)= 4    TYPE(3)= 0   TYPE(4)= 1
TYPE(5)= 0    TYPE(6)= 10     TYPE(7)= 1    TYPE(8)= 1   TYPE(9)= 0;

                                    END OF SAMPLE OUTPUT
```

Figure 24. PL/I Generated Output

RPG SAMPLE PROGRAM (RPGSMPL)

The RPGSMPL card deck (punched from SYS1.SAMPLIB) consists of

1. Job control language statements to call a cataloged procedure for a compilation, linkage-edit, and execution of the sample program.
2. Report Program Generator source statements; there are 45 of these.
3. Data cards for the program to process; there are 13 of these, and no other input is necessary.

The program processes the 13 data cards which contain details of customer transactions and prepares the report shown in Figure 12. A detailed description of the source program appears in IBM System/360 Operating System: Report Program Generator Language, Form C24-3337.

OPERATING INSTRUCTIONS

1. Mount the operating system and an initialized scratch pack.
2. Set the load address switches and press the Initial Program Load key to load the operating system.
3. Place the RPGSMPL card deck in the card reader, ready the reader, and press the End of File key.
4. Ready the printer.
5. Execute the job.

OUTPUT

Program output will consist of a printed listing of the source program and the printed report shown in Figure 25.

A C C O U N T S R E C E I V A B L E R E G I S T E R

CUSTOMER NUMBER	CUSTOMER NAME	LOCATION STATE	LOCATION CITY	INVOICE NUMBER	INVOICE DATE NO.	INVOICE DATE DAY	INVOICE AMOUNT
10712	AMALGAMATED CORP	33	61	11603	11	10	$ 389.25
							$ 389.25*
11315	BROWN WHOLESALE	30	231	12324	12	28	$ 802.08
11315	BROWN WHOLESALE	30	231	99588	12	14	$ 261.17
							$ 1,063.25*
11897	FARM IMPLEMENTS	47	77	10901	10	18	$ 27.63
							$ 27.63*
18530	BLACK OIL	16	67	11509	11	8	$ 592.95
18530	BLACK OIL	16	67	12292	12	23	$ 950.97
							$ 1,543.92*
20716	LEATHER BELT CO	36	471	11511	11	8	$ 335.63
20716	LEATHER BELT CO	36	471	12263	12	17	$ 121.75
							$ 457.38*
29017	GENERAL MFG CO	6	63	11615	11	14	$ 440.12
29017	GENERAL MFG CO	6	63	11676	11	23	$ 722.22
							$ 1,162.34*
29054	A-B-C DIST CO	25	39	9689	9	11	$ 645.40
29054	A-B-C DIST CO	25	39	11605	11	11	$ 271.69
29054	A-B-C DIST CO	25	39	12234	12	14	$ 559.33
							$ 1,476.42*
							$ 6,120.19**

Figure 25. RPG Sample Program Printed Report

SORT/MERGE SAMPLE PROGRAM (IERSP)

The IERSP card deck (punched from SYS1.SAMPLIB) consists of:

1. Job control language statements for TAPESORT.
2. Job control language statements for DISKSORT.
3. Data to be sorted.

The data to be sorted consists of 500 80-character records, each containing a 6-digit sequence number and a 10-character control field. As provided, the data deck is in the 6-digit sequence number order. The SORT control card specifies that the data is to be sorted on the 10-character control field. The output of either a TAPESORT or a DISKSORT will be in 10-character control field sequence.

The TAPESORT job control language statements provided in IERSP call for four tape units (three for scratch and one for SORTOUT). To use more than three work tapes, additional cards may be added behind the SORTWK03 DD card. They should be identical to other SORTWK DD cards except for DD name which must be consecutive, i.e., SORTWK04, SORTWK05,...,SORTWK32.

The DISKSORT job control language statements provided in IERSP call for six work areas of 30 contiguous tracks each on a 2311.

OPERATING INSTRUCTIONS

1. Mount the operating system and an initialized scratch pack.
2. Set the load address switches and press the Initial Program Load key to load the operating system.
3. For a TAPESORT, the data deck should be placed on tape as one unblocked sequential data set. An operator message (during step #7 below) will give mounting instructions.

   For a DISKSORT, the data deck should be placed on the scratch pack (mounted in step #1 above) as one unblocked sequential data set and cataloged under the name TEST.
4. For a TAPESORT, place the TAPESORT job control language statements in the card reader.

   For a DISKSORT, place the DISKSORT job control language statements in the card reader.
5. Ready the reader and press the End of File key.
6. Ready the printer.
7. Execute the job.

OUTPUT

A message to the operator will indicate the unit assigned to SORTOUT (unit containing output). This output can be printed and reviewed. The records will be in 10-character control field sequence.

Output of TAPESORT will be on tape.

Output of DISKSORT will be on disk.

UPDATE ANALYSIS PROGRAM SAMPLE PROGRAM (IHGSAMP)

The IHGSAMP card deck (punched from SYS1.SAMPLIB) consists of

1. Job control language statements and data to accomplish Stage 1, steps 1, 1A, 2, 3, 4, 5, 6, and 7 described in "Output" below.

2. Job control language statements to accomplish Stage 2, steps 3, 4, 5, and 6.

This sample problem is a two-stage operation requiring two input card decks. The Stage 1 input deck is complete. However, the output created by Stage 1 must be placed in front of the Stage 2 input deck before Stage 2 can be executed (The output of Stage 1 is the job control language statements to accomplish Stage 2, steps 1 and 2.)

In order to demonstrate the functions of Update Analysis, it is necessary to create some data sets on a disk pack of the operating system. However, the sample problem has been constructed so that at its completion all data sets created by it are deleted. Therefore, the program can be run without leaving residue from its operations on the system disk packs. The printer output acts as a graphic demonstration of the changing programs.

Stage 1 builds two sample symbolic libraries and one sample change library. It then executes the Update Analysis Program which creates the job stream required to update the sample libraries. Stage 2 utilizes the output of the Update Analysis to update the two sample libraries, then proceeds to eliminate the data sets from the system. This is described in the section "Output."

OPERATING INSTRUCTIONS

1. Mount the operating system and initialized scratch pack.

2. Set the load address switches and press the Initial Program Load key to load the operating system.

3. Place Stage 1 card deck in reader and ready the reader, printer, and punch.

4. Execute the Stage 1 job stream.

5. At the end of the job, take the output of Stage 1 from the punch stacker and place in front of Stage 2 deck.

6. Place Stage 2 deck in reader and ready reader.

7. Issue commands:
   START RDR, 00C
   START
   to execute the Stage 2 job stream.

OUTPUT

Stage 1

Step 1 EXEC IEBUPDAT
This step creates a temporary data set that contains the SYSIN control cards for step 1A.

Step 1A EXEC IEBGENER
This step creates the PDS SAMPLCHG which is the change PDS

containing three change members to be applied to the two sample libraries.

Step 2 EXEC IEBUPDAT
This step creates the PDS SAMPLIB1 which is a sample system library containing two members: MEMBER1 and MEMBER2.

Step 3 EXEC IEBUPDAT
This step creates the PDS SAMPLIB2 which is a sample system library containing one member: MEMBER3.

Step 4 EXEC IEHLIST
This step lists the directories of the two sample libraries. The list will show the SSI bytes of the members prior to updating.

Step 5 EXEC IHGUAP
This step causes the SAMPLCHG PDS to be analyzed and creates the job stream which will update the sample system libraries. For the purpose of the sample problem, this output is temporarily stored on disk to be available for steps 6 and 7.

Step 6 EXEC IEBPTPCH
This will print the output of step 5.

Step 7 EXEC IEBPTPCH
This will punch the output of step 5.

Stage 2

Steps 1 and 2 (Using punched output from step 7 of Stage 1)
EXEC IEBUPDAT Update SAMPLIB1
EXEC IEBUPDAT Update SAMPLIB2

Step 3 EXEC IEHLIST
This step lists the directories of the two sample libraries. The list will show the SSI bytes of the members after they have been updated.

Step 4 EXEC IEBUPDAT
This reproduces SAMPLIB1 in its updated form. This operation creates a printed list which shows the effect of the change.

Step 5 EXEC IEBUPDAT
This reproduces SAMPLIB2 in its updated form. It creates a printed list which shows the effect of the change.

Step 6 EXEC IEHPROGM
This causes the deletion of data sets SAMPLCHG, SAMPLIB1, and SAMPLIB2.

This section contains two examples of system generation. Example 1 shows a complete system generation, a Processor/Libraries generation, and a Nucleus generation. Example 2 shows a complete generation using the Starter Operating System.

The machine configurations and operating systems shown in these examples are not meant to represent the needs of an average installation, but were chosen because of their value as examples.

EXAMPLE 1

This example consists of:

- Diagram of a machine configuration
- Diagram of the volumes that contain generating and new system data sets
- Deck for initializing new system data sets
- Deck for system generation
- Creating back up of new system
- Scratching utility data sets
- Using the generating system to add a new compiler and library to the new system
- Cataloging the generating SYS1.GENLIB and SYS1.MODLIB in the new system
- Using the new system to add a second nucleus to itself

Machine Configuration

Figure 26 shows the machine configuration used in this example.

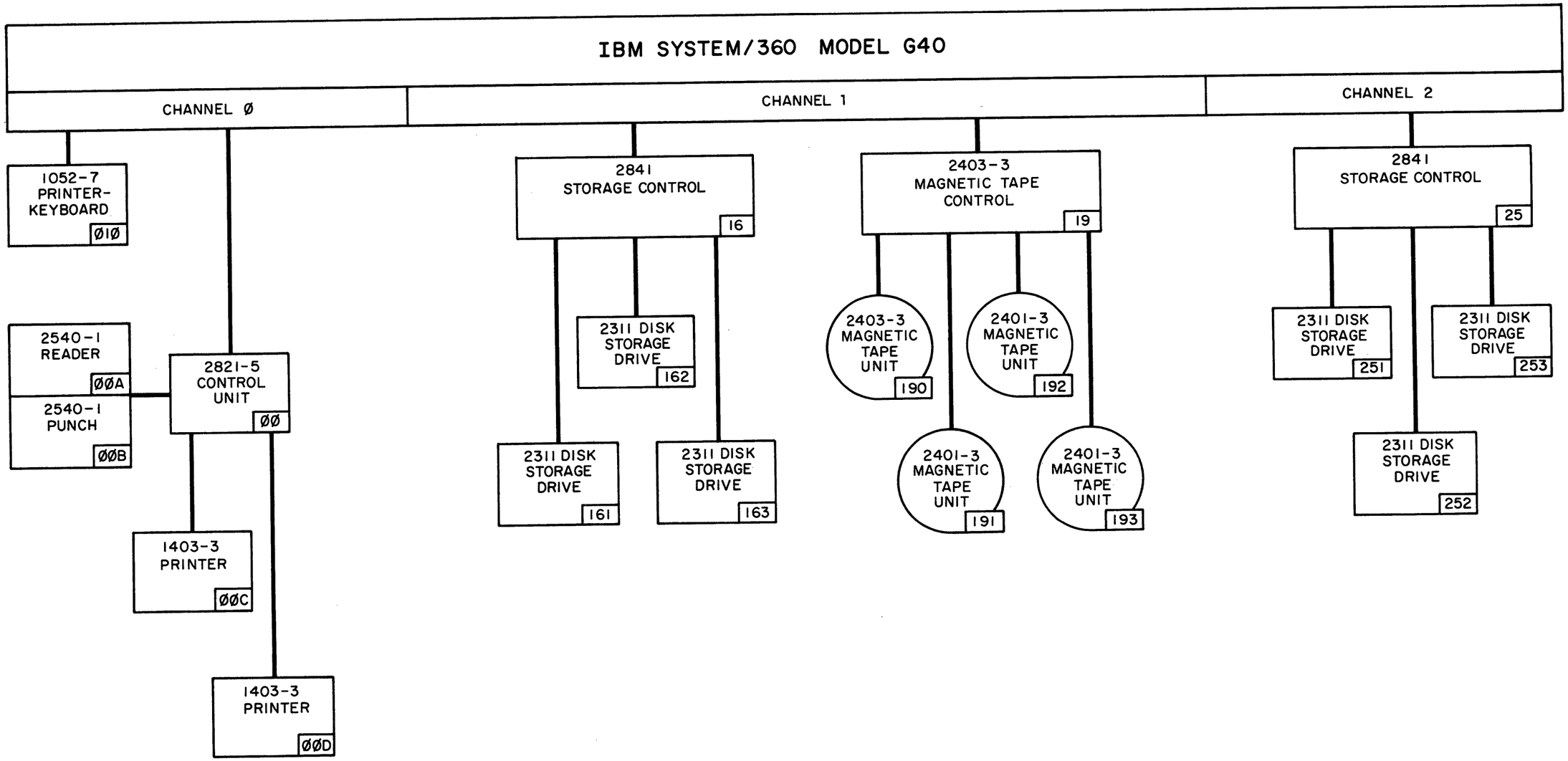


Figure 26. Example 1: Machine Configuration

Volumes Used for System Generation

Figure 27 shows the volumes that contain the generating and new system data sets. The serial number of the generating system residence volume is 111111. Volumes 222222 and 333333 contain the remaining generating system data sets. The serial number of the new system residence volume is SYSTEM. Volume LINVOL contains the remaining new system data sets. The four utility data sets used during system generation are named SYS1.MOD, SYS1.ONE, SYS1.TWO, and SYS1.THREE. The SYS1.USER data sets contain user-written routines for the new system.

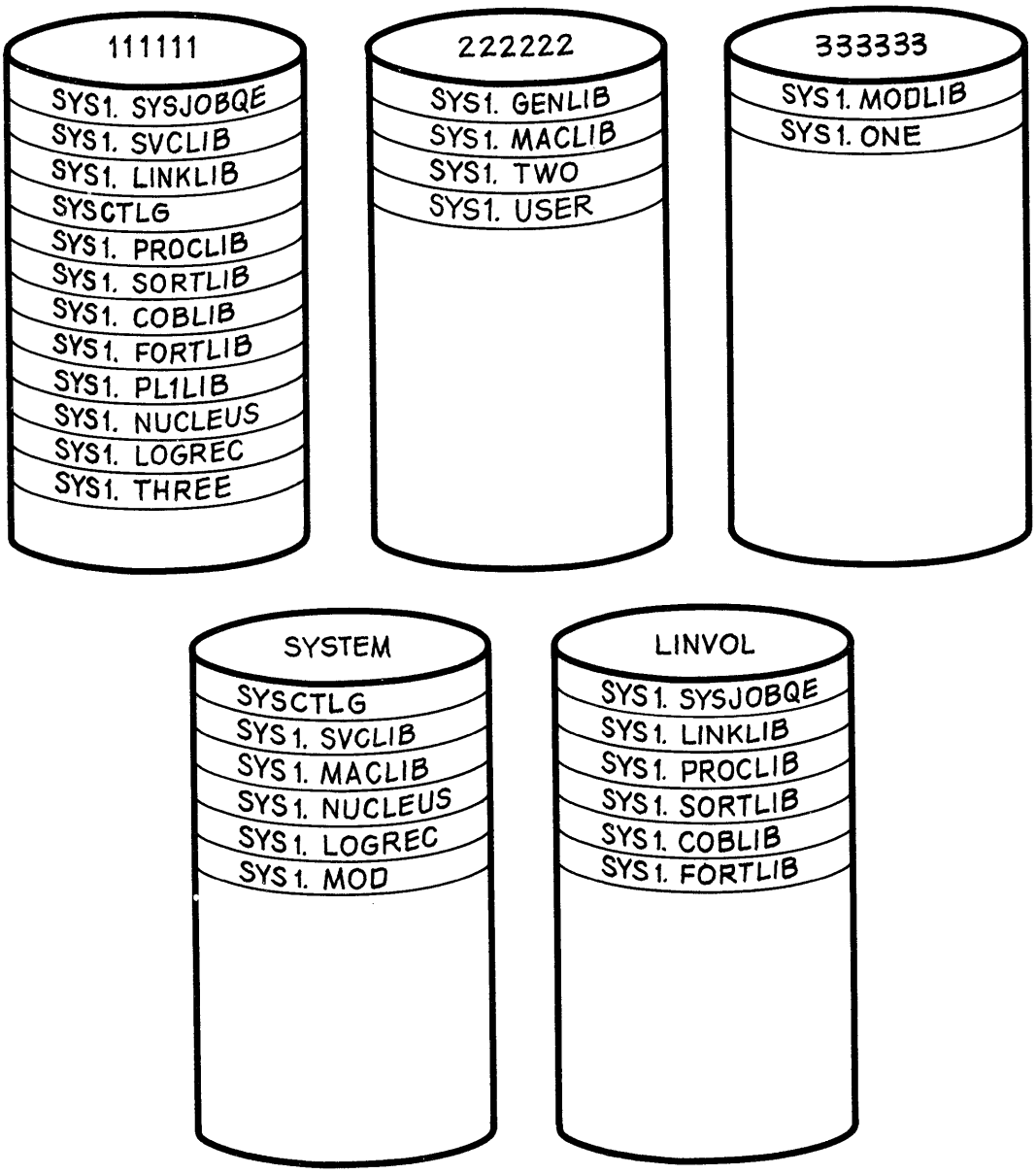


Figure 27. Example 1: Generating and New System Volumes

Deck for Initializing New System Data Sets

Figure 28 shows the input deck for initializing the new system data sets. During this initialization, volumes 111111, SYSTEM, and LINVOL must be mounted. The configuration used in this example (Figure 26) allows the simultaneous mounting of all five volumes (Figure 27) involved in this generation. Therefore, to simplify operating procedures, all five volumes should be mounted at this point. It is assumed that there is no dependency on device addresses, and that the volumes can be mounted on any 2311 drive. For illustrative purposes, assume that volumes 111111, 222222, 333333, SYSTEM, and LINVOL are mounted on drives 161, 162, 163, 251, and 252, respectively.

```
//SYSGEN   JOB  MSGLEVEL=1                    -EXAMPLE 1-
//STEP0    EXEC PGM=IEHPROGM               -ALLOCATE ON 2311-
//SYSPRINT DD  SYSOUT=A
//CATALOG  DD  DSNAME=SYSCTLG,VOLUME=(,RETAIN,SER=SYSTEM),                X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(10,1)),                 X
//             LABEL=EXPDT=99350
//SVCLIB   DD  DSNAME=SYS1.SVCLIB,VOLUME=(,RETAIN,SER=SYSTEM),            X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(150,,75),,CONTIG),      X
//             LABEL=EXPDT=99350,DCB=(DSORG=POU,RECFM=U,BLKSIZE=1024)
//MACLIB   DD  DSNAME=SYS1.MACLIB,VOLUME=(,RETAIN,SER=SYSTEM),            X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(440,50,25)),            X
//             LABEL=EXPDT=99350,DCB=(RECFM=FB,BLKSIZE=3360,LRECL=80)
//NUCLEUS  DD  DSNAME=SYS1.NUCLEUS,VOLUME=(,RETAIN,SER=SYSTEM),           X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(60,,2),,CONTIG),        X
//             LABEL=EXPDT=99350
//JOBQE    DD  DSNAME=SYS1.SYSJOBQE,VOLUME=(,RETAIN,SER=LINVOL),          X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(120),,CONTIG)
//LINKLIB  DD  DSNAME=SYS1.LINKLIB,VOLUME=(,RETAIN,SER=LINVOL),          X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(1100,,100),,CONTIG),   X
//             LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=3625)
//PROCLIB  DD  DSNAME=SYS1.PROCLIB,VOLUME=(,RETAIN,SER=LINVOL),          X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(30,10,9)),              X
//             LABEL=EXPDT=99350,DCB=(RECFM=F,BLKSIZE=80)
//SORTLIB  DD  DSNAME=SYS1.SORTLIB,VOLUME=(,RETAIN,SER=LINVOL),          X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(60,2,40)),              X
//             LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=3625)
//COBLIB   DD  DSNAME=SYS1.COBLIB,VOLUME=(,RETAIN,SER=LINVOL),           X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(35,2,30)),              X
//             LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=3625)
//FORTLIB  DD  DSNAME=SYS1.FORTLIB,VOLUME=(,RETAIN,SER=LINVOL),          X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(50,2,40)),              X
//             LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=3625)
//SYSIN    DD  *          -INPUT FOR CATALOGING SYSTEM DATA SETS-
     CATLG   CVOL=2311=SYSTEM,VOL=2311=LINVOL,DSNAME=SYS1.LINKLIB
     CATLG   CVOL=2311=SYSTEM,VOL=2311=LINVOL,DSNAME=SYS1.PROCLIB
     CATLG   CVOL=2311=SYSTEM,VOL=2311=SYSTEM,DSNAME=SYS1.MACLIB
     CATLG   CVOL=2311=SYSTEM,VOL=2311=LINVOL,DSNAME=SYS1.SORTLIB
     CATLG   CVOL=2311=SYSTEM,VOL=2311=LINVOL,DSNAME=SYS1.COBLIB
     CATLG   CVOL=2311=SYSTEM,VOL=2311=LINVOL,DSNAME=SYS1.FORTLIB
     CATLG   CVOL=2311=SYSTEM,VOL=2311=SYSTEM,DSNAME=SYS1.SVCLIB
/*
```

Figure 28. Example 1: Initializing New System Data Sets

Input Deck for Stage I

Figure 29 shows the input deck for system generation. The utility data sets are allocated space on the volumes indicated in Figure 27. The job stream will be written on an unlabeled magnetic tape that resides on drive 190 (see Figure 26). Stage II will be started automatically after Stage I is completed. Generic unit names are used. This input deck is the second step of the job defined in Figure 28.

The new system supports all devices in Figure 26. The primary control program (PCP) is used. BDAM, BISAM, QISAM, Assembler E, COBOL E, FORTRAN E, sort/merge, and RPG are included in the new system. The compilers have all standard default options for compilation time. The procedure library and the unit names it requires are also included. An user-written routine named NUCID is included in the nucleus. NUCID is a member of the SYS1.USER data set.

```
//STEP1     EXEC PGM=ASMBLR                -STAGE I INPUT-
//SYSLIB    DD  DSNAME=SYS1.GENLIB,DISP=OLD
//OBJPDS    DD  DSNAME=SYS1.MOD,VOLUME=(,RETAIN,SER=SYSTEM),                X
//              DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(40,20,8))
//SYSUT1    DD  DSNAME=SYS1.ONE,VOLUME=(,RETAIN,SER=333333),                X
//              DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(240,20))
//SYSUT2    DD  DSNAME=SYS1.TWO,VOLUME=(,RETAIN,SER=222222),                X
//              DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(240,20))
//SYSUT3    DD  DSNAME=SYS1.THREE,VOLUME=(,RETAIN,SER=111111),              X
//              DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(300,20))
//DUMMY     DD  VOLUME=(,RETAIN,REF=*.SYSUT3),SPACE=(TRK,(80))
//SYSPUNCH  DD  UNIT=190,LABEL=(,NL)
//SYSPRINT  DD  SYSOUT=A
//SYSIN     DD  *
          CENPROCS  MODEL=40,STORAGE=G,INSTSET=UNIV,FEATURE=PROTECT
CHAN0     CHANNEL   ADDRESS=0,TYPE=MULTIPLEXOR
CONSOLE   IODEVICE  UNIT=1052,MODEL=7,ADDRESS=010
CONT00    IOCONTRL  UNIT=2821,MODEL=5,ADDRESS=00,TRNMODE=BURST
READ00A   IODEVICE  UNIT=2540R,MODEL=1,ADDRESS=00A
PUNCH00B  IODEVICE  UNIT=2540P,MODEL=1,ADDRESS=00B
PRINT00C  IODEVICE  UNIT=1403,MODEL=3,ADDRESS=00C,FEATURE=UNVCHSET
PRINT00D  IODEVICE  UNIT=1403,MODEL=3,ADDRESS=00D,FEATURE=UNVCHSET
CHAN1     CHANNEL   ADDRESS=1,TYPE=SELECTOR
CONT16    IOCONTRL  UNIT=2841,ADDRESS=16
DISK161   IODEVICE  UNIT=2311,ADDRESS=161
DISK162   IODEVICE  UNIT=2311,ADDRESS=162
DISK163   IODEVICE  UNIT=2311,ADDRESS=163
CONT19    IOCONTRL  UNIT=2403,MODEL=3,ADDRESS=19
TAPE190   IODEVICE  UNIT=2403,MODEL=3,ADDRESS=190,FEATURE=9-TRACK
TAPE191   IODEVICE  UNIT=2401,MODEL=3,ADDRESS=191,FEATURE=9-TRACK
TAPE192   IODEVICE  UNIT=2401,MODEL=3,ADDRESS=192,FEATURE=9-TRACK
TAPE193   IODEVICE  UNIT=2401,MODEL=3,ADDRESS=193,FEATURE=9-TRACK
CHAN2     CHANNEL   ADDRESS=2,TYPE=SELECTOR
CONT25    IOCONTRL  UNIT=2841,ADDRESS=25
DISK251   IODEVICE  UNIT=2311,ADDRESS=251
DISK252   IODEVICE  UNIT=2311,ADDRESS=252
DISK253   IODEVICE  UNIT=2311,ADDRESS=253
```

Figure 29. Example 1: Stage I Input Deck (Part 1 of 2)

```
          UNITNAME  NAME=SYSSQ,UNIT=(161,162,163,190,191,192,193,253)
          UNITNAME  NAME=SYSDA,UNIT=(161,162,163,253)
          UNITNAME  NAME=SYSCP,UNIT=00B
          UNITNAME  NAME=TAPE,UNIT=(190,191,192,193)
          CTRLPROG  TYPE=PCP,MAXIO=15,OVERLAY=ADVANCED
          SCHEDULR  CONSOLE=010,ACCTRTN=SUPPLIED,                          X
               VLMOUNT=AVR,CANCEL=(NONAME,NOACCNUM)
          SUPRVSOR  RESIDNT=(ATTACH,EXTRACT,IDENTIFY),WAIT=MULTIPLE,       X
               OPTIONS=(TRSVCTBL,PROTECT),TRACE=100,SER=SER1
          PROCLIB   UNIT=2311,VOLNO=LINVOL
          RESMODS   PDS=SYS1.USER,MEMBERS=NUCID
          DATAMGT   ACSMETH=(BDAM,ISAM)
          EDITOR    DESIGN=E18
          EDITOR    DESIGN=E44
          ASSEMBLR  DESIGN=E
          TESTRAN   PHASES=(INTER,EDITOR),PAGES=50,EXEC=200,MODE=TRACE
          MACLIB
          CHKPOINT
          SORTMERG  SORTOPT=FULLIB,SIZE=51200
          SORTLIB   UNIT=2311,VOLNO=LINVOL
          COBOL     DESIGN=E
          COBLIB    DESIGN=E,UNIT=2311,VOLNO=LINVOL
          FORTRAN   DESIGN=E
          FORTLIB   DESIGN=E,UNIT=2311,VOLNO=LINVOL,OBJERR=03
          RPG
          GENERATE  UT1SDS=SYS1.ONE,UT2SDS=SYS1.TWO,UT3SDS=SYS1.THREE,     X
               OBJPDS=SYS1.MOD,RESNAME=2311,RESVOL=SYSTEM,RESTYPE=2311,    X
               LNKNAME=2311,LNKVOL=LINVOL,LBMAINT=E,ASMPRT=ON,             X
               LEPRT=(LIST,XREF),DIRDATA=PDS
          END
/*
//        START     RDR,190
```

Figure 29. Stage I Input Deck (Part 2 of 2)

Backup of New System

Figure 30 shows the input decks for the IBCDMPRS independent utility program for creating the backup of the new system. The backup copy is on magnetic tape. magnetic tape. It is assumed that SYSTEM is mounted on drive 251, and LINVOL on drive 252. After the two backup copies have been made, the two magnetic tapes on 191 and 192 should be removed and stored in the tape library.

```
DUMP1  JOB
       MSG   TODEV=1403,TOADDR=00C
       DUMP  FROMDEV=2311,FROMADDR=251,TODEV=2400,TOADDR=191
       END

DUMP2  JOB
       MSG   TODEV=1403,TOADDR=00C
       DUMP  FROMDEV=2311,FROMADDR=252,TODEV=2400,TOADDR=192
       END
```

Figure 30. Example 1: Creating Backup of SYSTEM and LINVOL

Scratching Utility Data Sets

After the new operating system is generated, SYS1.ONE, SYS1.TWO, and SYS1.THREE should be scratched and uncataloged. The job stream and SYS1.MOD should be saved. Figure 31 shows the input deck for scratching and uncataloging the three sequential data sets. The magnetic tape that contains the job stream can be removed from unit 190 and stored in the tape library. SYS1.MOD remains in the new system residence volume (SYSTEM) to be used later on in this example. The IEHPROGM utility program is executed under control of the generating system.

```
//SCRATCH  JOB    MSGLEVEL=1  -SCRATCH DECK-
//STEP0    EXEC   PGM=IEHPROGM
//SYSPRINT DD     SYSOUT=A
//SYSUT1   DD     DSNAME=SYS1.ONE,DISP=(OLD,DELETE)
//SYSUT2   DD     DSNAME=SYS1.TWO,DISP=(OLD,DELETE)
//SYSUT3   DD     DSNAME=SYS1.THREE,DISP=(OLD,DELETE)
//SYSIN    DD     DUMMY
```

Figure 31. Example 1: Scratching Utility Data Sets

Processor/Library Generation

After the new operating system has been generated, the PL/I compiler and SYS1.PL1LIB are to be added to it. This system generation will be performed under control of the generating system (volumes 111111, 222222, and 333333).

Figure 32 shows the input deck for this generation. STEP1 allocates space (on LINVOL) and catalogs the new SYS1.PL1LIB. STEP2 is the input deck for Stage I. Three sequential utility data sets are defined. Their names are SYS1.UT1, SYS1.UT2, and SYS1.UT3 and they reside on volumes 333333, 222222, and 111111, respectively. The SYS1.MOD partitioned data set used in the previous generation is used again. This can be done because only new components are added to the new system during this Processor/Library generation. The job stream will be written on an unlabeled magnetic tape that resides on drive 191.

After Stage II has terminated, SYS1.UT1, SYS1.UT2, and SYS1.UT3 can be scratched and uncataloged. The job stream tape should be saved. SYS1.MOD can be copied onto some removable volume and saved. The generating system can be removed. The new system is ready to operate after initial program load (IPL).

```
//SYSGEN   JOB MSGLEVEL=1      -PROCESSOR/LIBRARY GENERATION-
//STEP1    EXEC PGM=IEHPROGM          -ALLOCATE SYS1.PL1LIB-
//SYSPRINT DD  SYSOUT=A
//PL1LIB   DD  DSNAME=SYS1.PL1LIB,VOLUME=(,RETAIN,SER=LINVOL),          X
//             UNIT=2311,DISP=(,KEEP),SPACE=(TRK,(80,10,65)),           X
//             LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=3625)
//SYSIN    DD  *
     CATLG   CVOL=2311=SYSTEM,VOL=2311=LINVOL,DSNAME=SYS1.PL1LIB
/*
//STEP2    EXEC PGM=ASMBLR                -STAGE I INPUT-
//SYSLIB   DD  DSNAME=SYS1.GENLIB,DISP=OLD
//OBJPDS   DD  DSNAME=SYS1.MOD,DISP=OLD
//SYSUT1   DD  DSNAME=SYS1.UT1,VOLUME=(,RETAIN,SER=333333),            X
//             DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(240,20))
//SYSUT2   DD  DSNAME=SYS1.UT2,VOLUME=(,RETAIN,SER=222222),            X
//             DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(240,20))
//SYSUT3   DD  DSNAME=SYS1.UT3,VOLUME=(,RETAIN,SER=111111),            X
//             DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(300,20))
//DUMMY    DD  VOLUME=(,RETAIN,REF=*.SYSUT3),SPACE=(TRK,(80))
//SYSPUNCH DD  UNIT=191,LABEL=(,NL)
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
         CENPROCS  MODEL=40,STORAGE=G,INSTSET=UNIV,FEATURE=PROTECT
         SUPRVSOR  RESIDNT=(ATTACH,EXTRACT,IDENTIFY),WAIT=MULTIPLE,   X
             OPTIONS=(TRSVCTBL,PROTECT),TRACE=100,SER=SER1
         PL1   DESIGN=F
         PL1LIB    UNIT=2311,VOLNO=LINVOL,LIBFCNS=COMPLEX
         GENERATE  GENTYPE=PROCESSOR,UT1SDS=SYS1.UT1,UT2SDS=SYS1.UT2, X
             UT3SDS=SYS1.UT3,OBJPDS=SYS1.MOD,RESNAME=2311,            X
             RESVOL=SYSTEM,RESTYPE=2311,LNKNAME=2311,LNKVOL=LINVOL,   X
             LBMAINT=E,ASMPRT=ON,LEPRT=(LIST,XREF),DIRDATA=PDS
         END
/*
//         START     RDR,191
```

Figure 32. Example 1: Processor/Library Generation

Obtaining SYS1.GENLIB and SYS1.MODLIB

After the new operating system has been generated and PL/I has been included, a second nucleus is to be added to SYS1.NUCLEUS. This nucleus generation can be performed under control of the generating system (volumes 111111, 222222, and 333333). However, this example shows how the new system can be converted into a generating system, and how it can add a new nucleus to itself.

The new system (volumes SYSTEM and LINVOL) meets all the requirements for a generating system except that it does not have SYS1.GENLIB and SYS1.MODLIB. These two libraries can be copied from the generating system to the new system, and then cataloged. However, because there are enough drives to mount volumes SYSTEM, LINVOL, 222222, and 333333 simultaneously, SYS1.GENLIB and SYS1.MODLIB need only be cataloged in the new system. Figure 33 shows the input deck for cataloging SYS1.GENLIB and SYS1.MODLIB in the new system. The SYS1.USER data set that resides on volume 222222 is also cataloged in the new system. This cataloging is performed under control of the new system.

```
//GENMOD   JOB MSGLEVEL=1  -GENLIB&MODLIB-
//CATLG    EXEC PGM=IEHPROGM
//NEWRES   DD  UNIT=2311,VOLUME=SER=SYSTEM,DISP=OLD
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
     CATLG  DSNAME=SYS1.GENLIB,VOL=2311=222222
     CATLG  DSNAME=SYS1.MODLIB,VOL=2311=333333
     CATLG  DSNAME=SYS1.USER,VOL=2311=222222
/*
```

Figure 33. Example 1: Cataloging SYS1.GENLIB and SYS1.MODLIB

Nucleus Generation

A second nucleus (02) is added to the new system during this generation. The same CENPROCS, CHANNEL, IOCONTRL, and IODEVICE macro-instructions included in the generation of the new system (see Figure 29) must be included in this generation. The new nucleus will support the same access methods (BDAM, BISAM, and QISAM) TESTRAN options, and checkpoint/restart facility as the first nucleus. Each nucleus must support the same control program (PCP), because the resident portion of the control program (nucleus) must be compatible with the nonresident portion.

Figure 34 shows the volumes involved in this generation. The four system generation utility data sets are named SYS1.OBJECT, SYS1.UTIL1, SYS1.UTIL2, SYS1.UTIL3.

Figure 35 shows the input deck to system generation. There is no allocation step before Stage I because the only library affected by a Nucleus generation is SYS1.NUCLEUS. The job stream will be written on an unlabeled magnetic tape that resides on drive 190. A user-written routine named NUCID2 is included in the nucleus. NUCID2 is a member of the SYS1.USER data set.

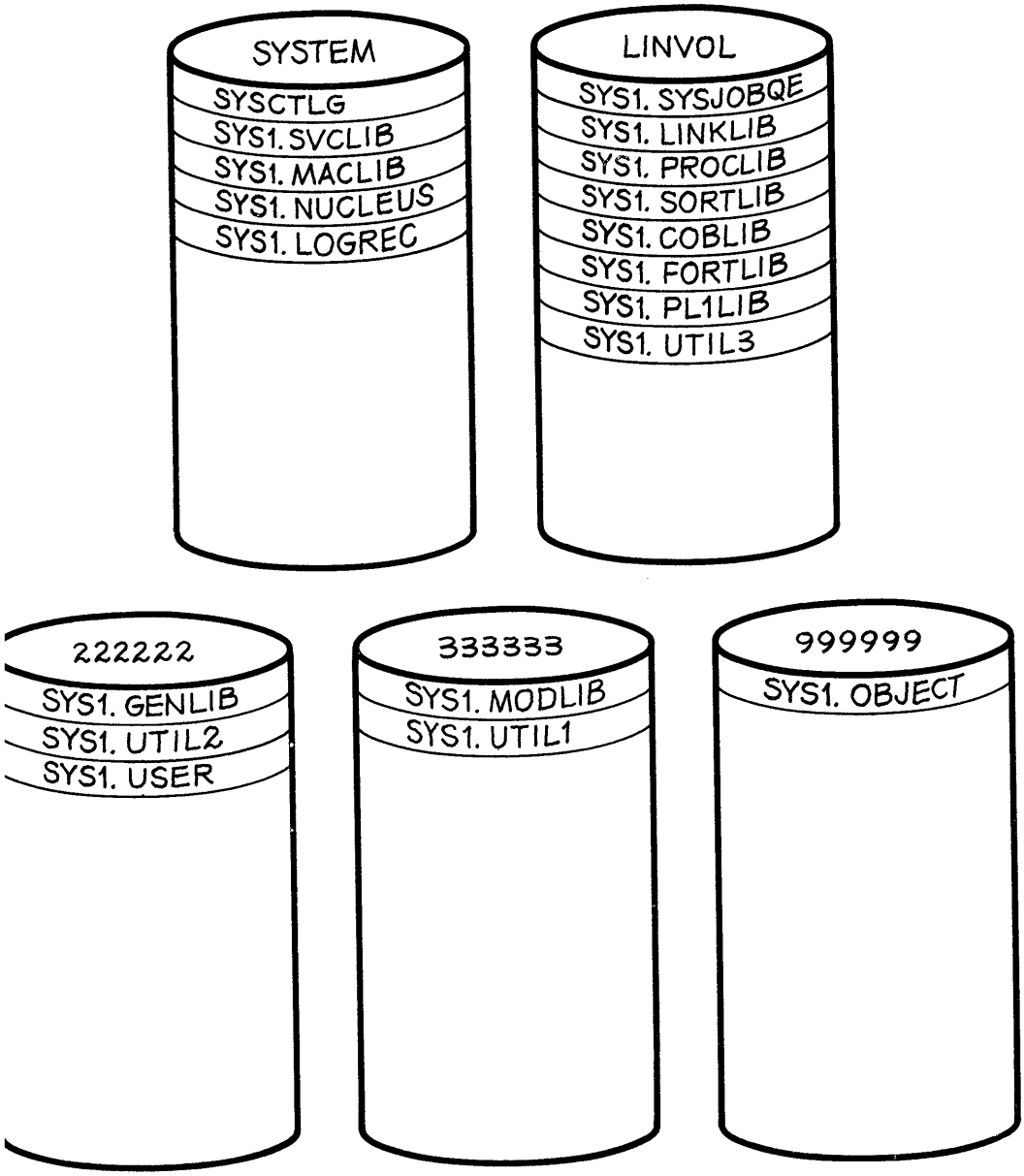


Figure 34. Example 1: Volumes for Nucleus Generation

```
//SYSGEN   JOB  MCR,67,MSGLEVEL=1        -NUCLEUS GENERATION-
//STEP     EXEC PGM=ASMBLR               -STAGE I INPUT-
//SYSLIB   DD  DSNAME=SYS1.GENLIB,DISP=OLD
//OBJPDS   DD  DSNAME=SYS1.OBJECT,VOLUME=(,RETAIN,SER=999999),          X
//             DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(40,20,8))
//SYSUT1   DD  DSNAME=SYS1.UTIL1,VOLUME=(,RETAIN,SER=333333),          X
//             DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(240,20))
//STSUT2   DD  DSNAME=SYS1.UTIL2,VOLUME=(,RETAIN,SER=222222),          X
//             DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(240,20))
//SYSUT3   DD  DSNAME=SYS1.UTIL3,VOLUME=(,RETAIN,SER=LINVOL),          X
//             DISP=(,CATLG),UNIT=2311,SPACE=(TRK,(300,20))
//DUMMY    DD  VOLUME=(,RETAIN,REF=*.SYSUT3),SPACE=(TRK,(80))
//SYSPUNCH DD  UNIT=190,LABEL=(,NL)
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
         CENPROCS  MODEL=40,STORAGE=G,INSTSET=UNIV,FEATURE=PROTECT
CHAN0    CHANNEL   ADDRESS=0,TYPE=MULTIPLEXOR
CONSOLE  IODEVICE  UNIT=1052,MODEL=7,ADDRESS=010
CONT00   IOCONTRL  UNIT=2821,MODEL=5,ADDRESS=00,TRNMODE=BURST
READ00A  IODEVICE  UNIT=2540R,MODEL=1,ADDRESS=00A
PUNCH00B IODEVICE  UNIT=2540P,MODEL=1,ADDRESS=00B
PRINT00C IODEVICE  UNIT=1403,MODEL=3,ADDRESS=00C,FEATURE=UNVCHSET
PRINT00D IODEVICE  UNIT=1403,MODEL=3,ADDRESS=00D,FEATURE=UNVCHSET
CHAN1    CHANNEL   ADDRESS=1,TYPE=SELECTOR
CONT16   IOCONTRL  UNIT=2841,ADDRESS=16
DISK161  IODEVICE  UNIT=2311,ADDRESS=161
DISK162  IODEVICE  UNIT=2311,ADDRESS=162
DISK163  IODEVICE  UNIT=2311,ADDRESS=163
CONT19   IOCONTRL  UNIT=2403,MODEL=3,ADDRESS=19
TAPE190  IODEVICE  UNIT=2403,MODEL=3,ADDRESS=190,FEATURE=9-TRACK
TAPE191  IODEVICE  UNIT=2401,MODEL=3,ADDRESS=191,FEATURE=9-TRACK
TAPE192  IODEVICE  UNIT=2401,MODEL=3,ADDRESS=192,FEATURE=9-TRACK
TAPE193  IODEVICE  UNIT=2401,MODEL=3,ADDRESS=193,FEATURE=9-TRACK
CHAN2    CHANNEL   ADDRESS=2,TYPE=SELECTOR
CONT25   IOCONTRL  UNIT=2841,ADDRESS=25
DISK251  IODEVICE  UNIT=2311,ADDRESS=251
DISK252  IODEVICE  UNIT=2311,ADDRESS=252
DISK253  IODEVICE  UNIT=2311,ADDRESS=253
         CTRLPROG  TYPE=PCP,MAXIO=15,FETCH=PCI
         SCHEDULR  CONSOLE=010,ALTCONS=(1-00A,0-00C),ACCTRTN=SUPPLIED,X
               DESIGN=44K,CANCEL=(NONAME,NOACCNUM)
         SUPRVSOR  RESIDNT=(ATTACH,EXTRACT,IDENTIFY,BLDLTAB,ACSMETH), X
               OPTIONS=(TRSVCTBL,PROTECT,COMM),WAIT=MULTIPLE,TRACE=100,X
               SER=SER1
         RESMODS   PDS=SYS1.USER,MEMBERS=NUCID2
         DATAMGT   ACSMETH=(BDAM,ISAM)
         TESTRAN   PHASES=(INTER,EDITOR),PAGES=50,EXEC=200,MODE=TRACE
         CHKPOINT
         GENERATE  GENTYPE=(NUCLEUS,2),UT1SDS=SYS1.UTIL1,              X
               UT2SDS=SYS1.UTIL2,UT3SDS=SYS1.UTIL3,OBJPDS=SYS1.OBJECT, X
               RESNAME=2311,RESVOL=SYSTEM,RESTYPE=2311,LBMAINT=E,       X
               ASMPRT=ON,LEPRT=(LIST,XREF),DIRDATA=PDS
         END
/*
//       START     RDR,190
```

Figure 35. Example 1: Input Deck for Nucleus Generation

EXAMPLE 2

This example consists of:

- Diagram of a machine configuration.
- Decks for initializing the starter operating system (2314 distribution).
- Decks for initializing direct-access volumes for the new system.
- Diagram of the volumes that contain generating and new system data sets.
- Deck for initializing new system data sets.
- Deck for system generation.
- Decks for system residence on 2303.

Machine Configuration

Figure 36 shows the machine configuration used in this example. Only those devices on shaded areas are supported by the starter operating system.

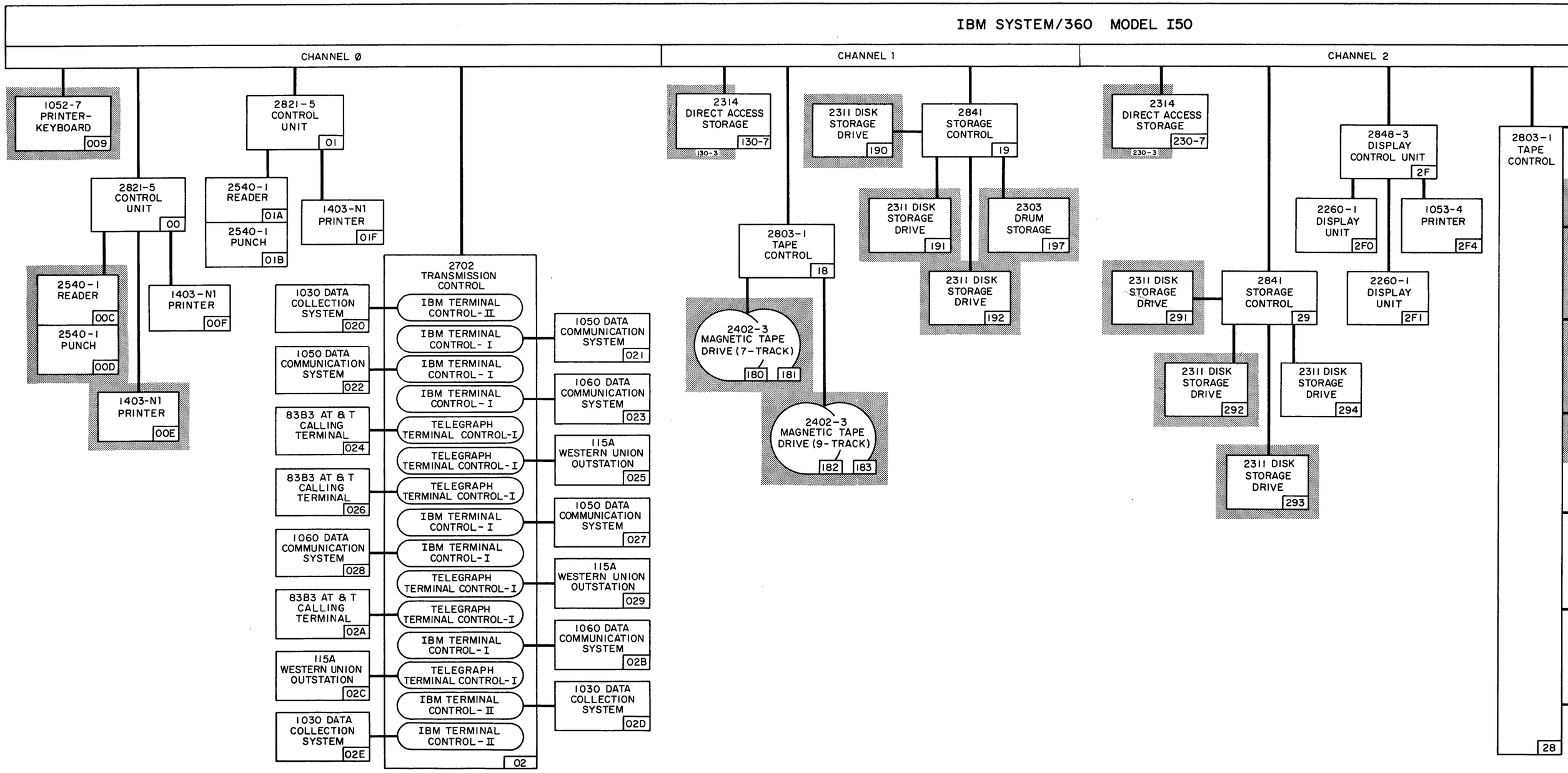
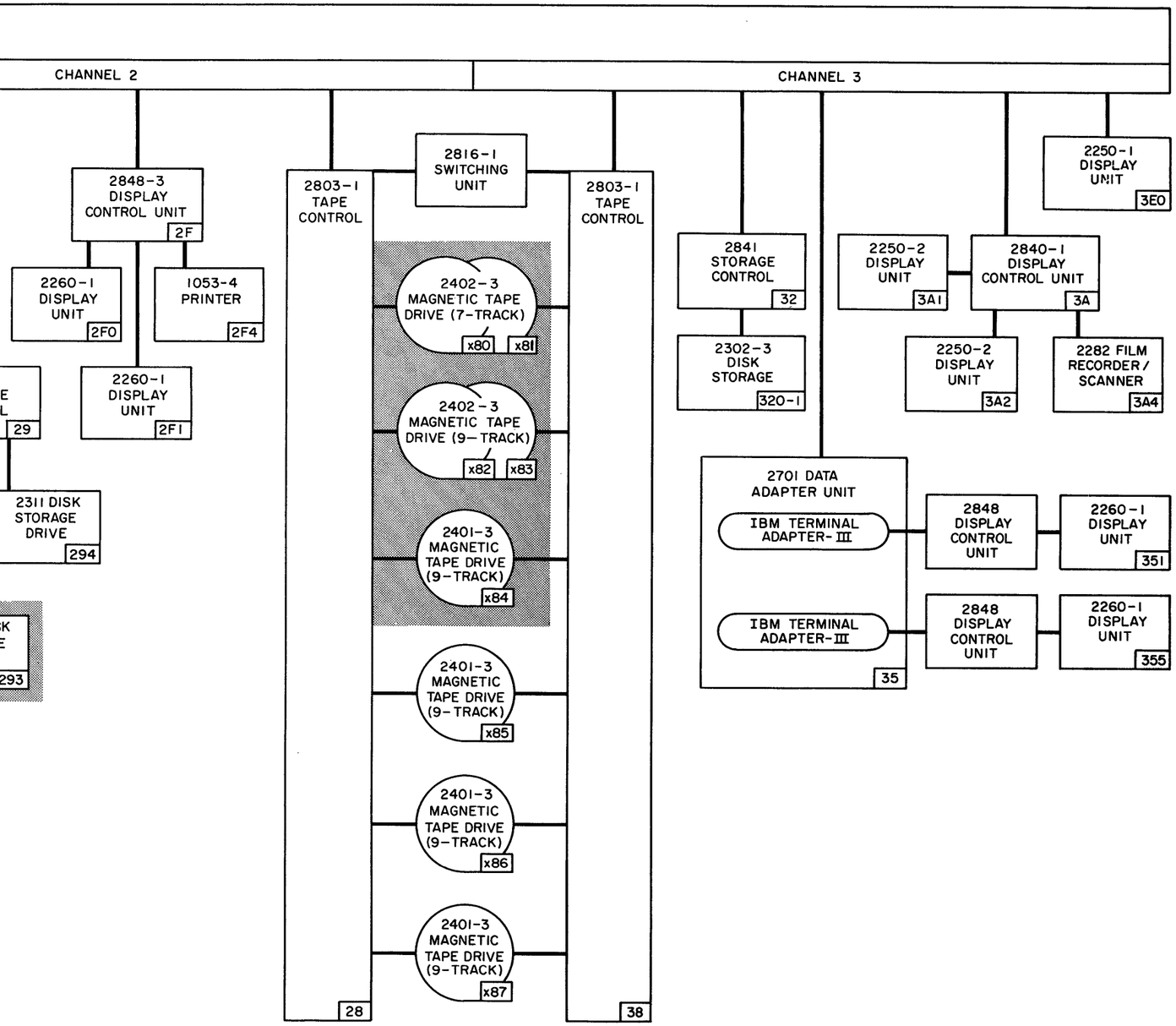


Figure 36. Example 2: Machine Configuration


CHANNEL 2
CHANNEL 3
2848-3 DISPLAY CONTROL UNIT 2F
2260-1 DISPLAY UNIT 2F0
1053-4 PRINTER 2F4
2260-1 DISPLAY UNIT 2F1
29
2311 DISK STORAGE DRIVE 294
293
2803-1 TAPE CONTROL
28
2816-1 SWITCHING UNIT
2402-3 MAGNETIC TAPE DRIVE (7-TRACK) x80 x81
2402-3 MAGNETIC TAPE DRIVE (9-TRACK) x82 x83
2401-3 MAGNETIC TAPE DRIVE (9-TRACK) x84
2401-3 MAGNETIC TAPE DRIVE (9-TRACK) x85
2401-3 MAGNETIC TAPE DRIVE (9-TRACK) x86
2401-3 MAGNETIC TAPE DRIVE (9-TRACK) x87
2803-1 TAPE CONTROL
38
2841 STORAGE CONTROL 32
2302-3 DISK STORAGE 320-1
2701 DATA ADAPTER UNIT
IBM TERMINAL ADAPTER-III
IBM TERMINAL ADAPTER-III
35
2848 DISPLAY CONTROL UNIT
2848 DISPLAY CONTROL UNIT
2260-1 DISPLAY UNIT 351
2260-1 DISPLAY UNIT 355
2250-2 DISPLAY UNIT 3A1
2250-2 DISPLAY UNIT 3A2
2840-1 DISPLAY CONTROL UNIT 3A
2282 FILM RECORDER/ SCANNER 3A4
2250-1 DISPLAY UNIT 3E0

Decks for Initializing the Starter Operating System

Four decks are required to initialize the 2314 distribution of the starter system.

Figure 37 shows the IBCDASDI deck for initializing the volume that is to contain DLIB01. The serial number of the volume is NNNNNN. This volume is being initialized for the first time. This volume resides on unit 130. The distribution tape (DLIB01) resides on unit 280. The DASDI deck is placed on the 2540-1 card reader at unit 00C. Messages will be printed on the 1403-N1 printer at unit 00E.

```
DLIB01   JOB
         MSG      TODEV=1403,TOADDR=00E
         DADEF    TODEV=2314,TOADDR=130,VOLID=SCRATCH,                X
                  FLAGTEST=NO
         VLD      NEWVOLID=NNNNNN,OWNERID=DEPTD58
         VTOCD    STRTADR=50,EXTENT=10
         END
```

Figure 37. Example 2: Initializing Volume for DLIB01

Figure 38 shows the IBCDMPRS deck for restoring DLIB01 to volume NNNNNN. This deck is placed on unit 00C.

```
RESTORE  JOB
         MSG      TODEV=1403,TOADDR=00E
         RESTORE  FROMDEV=2400,FROMADDR=280,TODEV=2314,               X
                  TOADDR=130,VOLID=NNNNNN
         END
```

Figure 38. Example 2: Restoring DLIB01

After executing the IBCDMPRS program, the volume on unit 130 is now DLIB01. The tape on unit 280 should be removed and stored in the tape library as a backup for the starter system.

After IPL, DLIB01 is ready for operations. The programs in Figure 39 and 40 operate under the starter system.

Figure 39 shows the deck for punching the contents of SYS1.SAMPLIB. In addition to the independent utility programs, the sample programs for FORTRAN G, PL/I, and sort/merge are punched.

```
//PUNCH    JOB MSGLEVEL=1  -PUNCH SYS1.SAMPLIB-
//        EXEC PGM=IEBPTPCH
//SYSUT1   DD  DSNAME=SYS1.SAMPLIB,DISP=(OLD,KEEP),                X
//            UNIT=2314,VOLUME=SER=DLIB01
//SYSUT2   DD  UNIT=2540-2
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
      PUNCH    TYPORG=PO,MAXNAME=7
      MEMBERNAME=IBCDMPRS
      MEMBERNAME=IBCDASDI
      MEMBERNAME=IBCRCVRP
      MEMBERNAME=IEAIPL00
      MEMBERNAME=IEYSP
      MEMBERNAME=IEMSP2
      MEMBERNAME=IERSP
/*
```

Figure 39. Example 2: Punching Members of SYS1.SAMPLIB

After these seven decks are punched, the first card (member name card) of each deck must be removed.

Figure 40 shows the deck for listing the system data describing the starter system's SYS1.NUCLEUS, SYS1.SAMPLIB, and SYS1.PROCLIB.

```
//LIST     JOB MSGLEVEL=1
//         EXEC PGM=IEHLIST
//SYSPRINT DD  SYSOUT=A
//DD1      DD  UNIT=2314,DISP=OLD,VOLUME=SER=DLIB01
//SYSIN    DD  *
      LISTCTLG
      LISTVTOC DUMP
      LISTPDS  DSNAME=(SYS1.NUCLEUS,SYS1.PROCLIB,                 X
               SYS1.SAMPLIB)
/*
```

Figure 40. Example 2: Listing Data in DLIB01

Initializing Volumes for New System

The volumes for the new system must be initialized before system generation. Five volumes (besides DLIB01) will be used. Their serial numbers will be MVT111, MVT222, MVT333, MVT444, and MVT555. These volumes are mounted on units 131, 230, 231, 232, and 132, respectively. MVT111 is to be the new system residence volume. It is assumed that all these volumes had been initialized before.

Figure 41 shows five IBCDASDI input decks for initializing the new volumes. The deck for MVT111 contains the IPL program (IEAIPL00). The IPL cards were obtained from SYS1.SAMPLIB (see Figure 39).

The IBCDASDI object program cards must be placed in the card reader on unit 00C and loaded into main storage. The input decks in Figure 41 are placed in the card reader immediately following the IBCDASDI object program deck. (For further details on the operating procedure for IBCDASDI, refer to the publication IBM System/360 Operating System: Utilities.)

```
MVT111 JOB
       MSG      TODEV=1403,TOADDR=00E
       DADEF    TODEV=2314,TOADDR=131,IPL=YES,VOLID=SCRATCH
       VLD      NEWVOLID=MVT111,OWNERID=DEPTD58
       VTOCD    STRTADR=2,EXTENT=8
       IPLTXT
   .
   .   IEAIPL00 cards
   .
.END
MVT222 JOB
       MSG      TODEV=1403,TOADDR=00E
       DADEF    TODEV=2314,TOADDR=230,VOLID=SCRATCH
       VLD      NEWVOLID=MVT222,OWNERID=DEPTD58
       VTOCD    STRTADR=2,EXTENT=8
       END
MVT333 JOB
       MSG      TODEV=1403,TOADDR=00E
       DADEF    TODEV=2314,TOADDR=231,VOLID=SCRATCH
       VLD      NEWVOLID=MVT333,OWNERID=DEPTD58
       VTOCD    STRTADR=2,EXTENT=8
       END
MVT444 JOB
       MSG      TODEV=1403,TOADDR=00E
       DADEF    TODEV=2314,TOADDR=232,VOLID=SCRATCH
       VLD      NEWVOLID=MVT444,OWNERID=DEPTD58
       VTOCD    STRTADR=2,EXTENT=8
       END
MVT555 JOB
       MSG      TODEV=1403,TOADDR=00E
       DADEF    TODEV=2314,TOADDR=132,VOLID=SCRATCH
       VLD      NEWVOLID=MVT555,OWNERID=DEPTD58
       VTOCD    STRTADR=2,EXTENT=8
       END
```

Figure 41. Example 2: Initializing New System Volumes

Volumes Used for System Generation

Figure 42 shows the volumes that contain the generating and new system data sets. The serial number of the generating system residence volume is DLIB01. The serial number of the new system residence volume is MVT111. Volumes MVT222, MVT333, and MVT444 contain the remaining new system data sets. The four utility data sets used during system generation are named SYS1.SGOBJ, SYS1.SG1, SYS1.SG2, and SYS1.SG3.

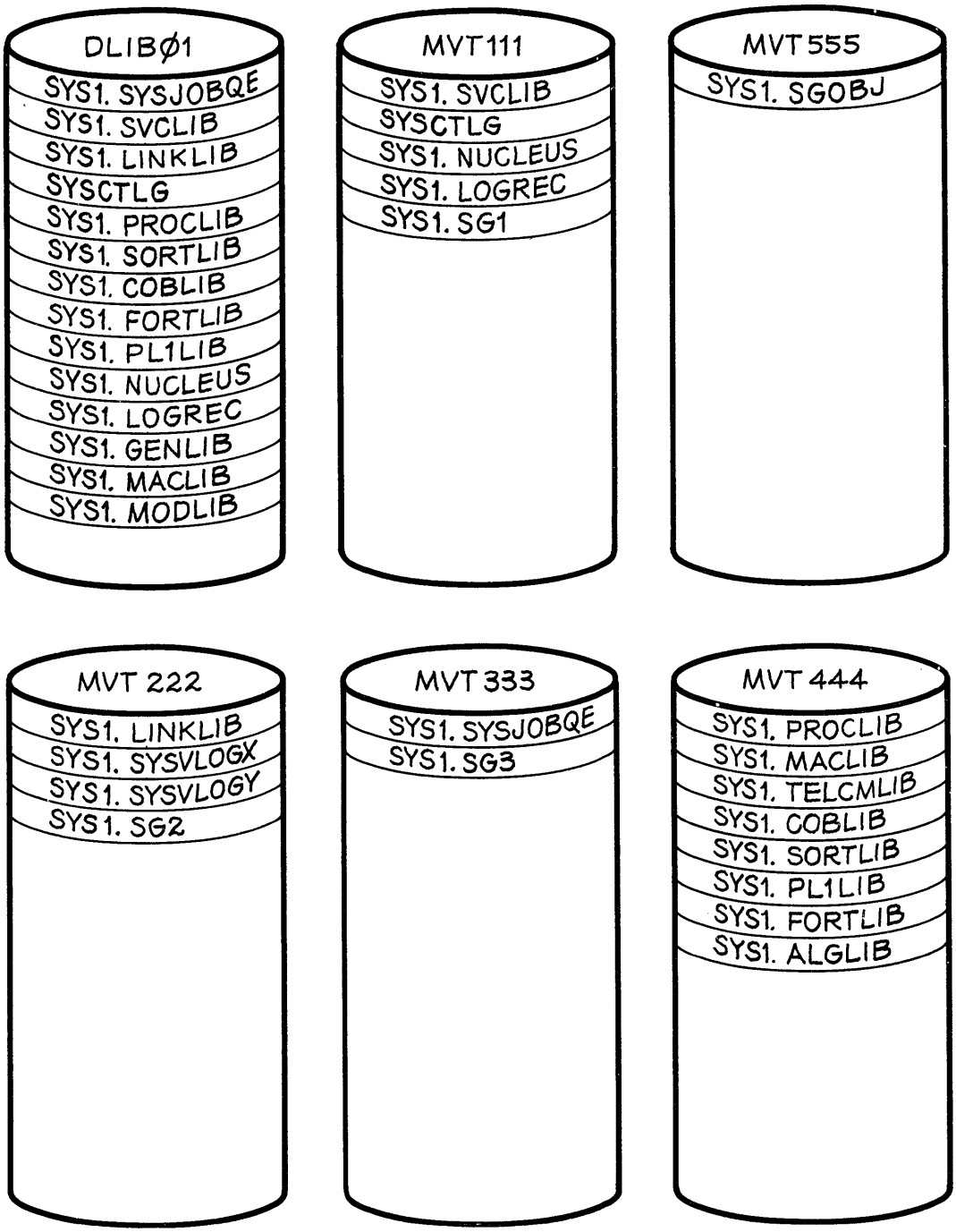


Figure 42. Example 2: Generating and New System Volumes (2314)

Initializing New System Data Sets

Figure 43 shows the input deck for initializing the new system data sets. During this initialization, volumes DLIB01, MVT111, MVT222, MVT333, and MVT444 must be mounted. The configuration used in this example (Figure 36) allows the simultaneous mounting of all six volumes (Figure 42) involved in this generation. Therefore, to simplify operating procedures, all six volumes should be mounted at this point. It is assumed that there is no dependency on device addresses, and that the volumes can be mounted on any 2314 drive. For illustrative purposes assume that volumes DLIB01, MVT111, and MVT555 are mounted on drives 130, 131, and 132; and that volumes MVT222, MVT333, and MVT444 are mounted on drive 230, 231, and 232, respectively.

```
//SYSGEN    JOB  MSGLEVEL=1                    -EXAMPLE 2-
//STEP      EXEC PGM=IEHPROGM               -ALLOCATE ON 2314-
//SYSPRINT  DD   SYSOUT=A
//SVCLIB    DD   DSNAME=SYS1.SVCLIB,VOLUME=(,RETAIN,SER=MVT111),          X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(130,,75),,CONTIG),    X
//               LABEL=EXPDT=99350,DCB=(DSORG=POU,RECFM=U,BLKSIZE=1024)
//CATALOG   DD   DSNAME=SYSCTLG,VOLUME=(,RETAIN,SER=MVT111),             X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(6,1)),                X
//               LABEL=EXPDT=99350
//NUCLEUS   DD   DSNAME=SYS1.NUCLEUS,VOLUME=(,RETAIN,SER=MVT111),        X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(20,,1),,CONTIG),      X
//               LABEL=EXPDT=99350
//LINKLIB   DD   DSNAME=SYS1.LINKLIB,VOLUME=(,RETAIN,SER=MVT222),        X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(850,,100),,CONTIG),   X
//               LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=7294)
//LOGX      DD   DSNAME=SYS1.SYSVLOGX,VOLUME=(,RETAIN,SER=MVT222),        X
//               UNIT=2314,DISP=(,KEEP),SPACE=(120,(100),,CONTIG),        X
//               LABEL=EXPDT=99350
//LOGY      DD   DSNAME=SYS1.SYSVLOGY,VOLUME=(,RETAIN,SER=MVT222),        X
//               UNIT=2314,DISP=(,KEEP),SPACE=(120,(100),,CONTIG),        X
//               LABEL=EXPDT=99350
//JOBQE     DD   DSNAME=SYS1.SYSJOBQE,VOLUME=(,RETAIN,SER=MVT333),        X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(200),,CONTIG)
//PROCLIB   DD   DSNAME=SYS1.PROCLIB,VOLUME=(,RETAIN,SER=MVT444),        X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(20,6,9)),             X
//               LABEL=EXPDT=99350,DCB=(RECFM=F,BLKSIZE=80)
//MACLIB    DD   DSNAME=SYS1.MACLIB,VOLUME=(,RETAIN,SER=MVT444),         X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(300,30,25)),          X
//               LABEL=EXPDT=99350,DCB=(RECFM=FB,BLKSIZE=7280,LRECL=80)
//TELCLIB   DD   DSNAME=SYS1.TELCMLIB,VOLUME=(,RETAIN,SER=MVT444),       X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(30,2,10)),            X
//               LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=7294)
//COBLIB    DD   DSNAME=SYS1.COBLIB,VOLUME=(,RETAIN,SER=MVT444),         X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(25,1,30)),            X
//               LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=7294)
//SORTLIB   DD   DSNAME=SYS1.SORTLIB,VOLUME=(,RETAIN,SER=MVT444),        X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(40,1,40)),            X
//               LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=7294)
//PL1LIB    DD   DSNAME=SYS1.PL1LIB,VOLUME=(,RETAIN,SER=MVT444),         X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(55,2,65)),            X
//               LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=7294)
//FORTLIB   DD   DSNAME=SYS1.FORTLIB,VOLUME=(,RETAIN,SER=MVT444),        X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(25,1,40)),            X
//               LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=7294)
//ALGLIB    DD   DSNAME=SYS1.ALGLIB,VOLUME=(,RETAIN,SER=MVT444),         X
//               UNIT=2314,DISP=(,KEEP),SPACE=(TRK,(26,2,15)),            X
//               LABEL=EXPDT=99350,DCB=(RECFM=U,BLKSIZE=7294)
//SYSIN     DD   *              -INPUT FOR CATALOGING SYSTEM DATA SETS-
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT222,DSNAME=SYS1.LINKLIB
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT111,DSNAME=SYS1.SVCLIB
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT333,DSNAME=SYS1.SYSJOBQE
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT444,DSNAME=SYS1.PROCLIB
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT444,DSNAME=SYS1.MACLIB
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT444,DSNAME=SYS1.TELCMLIB
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT444,DSNAME=SYS1.COBLIB
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT444,DSNAME=SYS1.SORTLIB
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT444,DSNAME=SYS1.PL1LIB
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT444,DSNAME=SYS1.FORTLIB
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT444,DSNAME=SYS1.ALGLIB
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT222,DSNAME=SYS1.SYSVLOGX
     CATLG   CVOL=2314=MVT111,VOL=2314=MVT222,DSNAME=SYS1.SYSVLOGY
/*
```

Figure 43. Example 2: Initializing New System Data Sets (2314)

Input Deck for Stage I

Figure 44 shows the input deck for system generation. The utility data sets are allocated space on the volumes indicated in Figure 42. The job stream will be written on an unlabeled magnetic tape that resides on drive 182 (see Figure 36). Stage II will be started automatically after Stage I is completed. Generic unit names are used.

The new system supports all devices in Figure 36. Multiprogramming with a variable number of tasks (MVT) is used. All access methods and optional system data sets are generated. Assembler F, linkage editor E44, TESTRAN, sort/merge, ALGOL, COBOL F, FORTRAN G, PL/I F, and RPG are included in the new system. All standard default options are selected for the processors.

After Stage II has terminated, SYS1.SG1, SYS1.SG2, and SYS1.SG3 can be scratched and uncataloged. The job stream and SYS1.SGOBJ should be saved. DLIB01 can be removed. A backup copy of the new system should be made. The new system is ready to operate after initial program load (IPL).

```
//STEP1   EXEC PGM=ASMBLR                    -STAGE I INPUT-
//SYSLIB   DD  DSNAME=SYS1.GENLIB,DISP=OLD
//OBJPDS   DD  DSNAME=SYS1.SGOBJ,VOLUME=(,RETAIN,SER=MVT555),          X
//             DISP=(,CATLG),UNIT=2314,SPACE=(TRK,(32,10,8))
//SYSUT1   DD  DSNAME=SYS1.SG1,VOLUME=(,RETAIN,SER=MVT111),            X
//             DISP=(,CATLG),UNIT=2314,SPACE=(TRK,(160,10))
//SYSUT2   DD  DSNAME=SYS1.SG2,VOLUME=(,RETAIN,SER=MVT222),            X
//             DISP=(,CATLG),UNIT=2314,SPACE=(TRK,(160,10))
//SYSUT3   DD  DSNAME=SYS1.SG3,VOLUME=(,RETAIN,SER=MVT333),            X
//             DISP=(,CATLG),UNIT=2314,SPACE=(TRK,(200,10))
//DUMMY    DD  VOLUME=(,RETAIN,REF=*.SYSUT3),SPACE=(TRK,(60))
//SYSPUNCH  DD  UNIT=182,LABEL=(,NL)
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
      CENPROCS  MODEL=50,STORAGE=I,FEATURE=PROTECT
MPX    CHANNEL   ADDRESS=0,TYPE=MULTIPLEXOR
CONSOLE   IODEVICE  UNIT=1052,MODEL=7,ADDRESS=009
CNT00   IOCONTRL    UNIT=2821,MODEL=5,ADDRESS=00,FEATURE=COLBNRY,     X
            TRNMODE=BYTE
DEV00C    IODEVICE  UNIT=2540R,MODEL=1,ADDRESS=00C
DEV00D    IODEVICE  UNIT=2540P,MODEL=1,ADDRESS=00D
DEV00E    IODEVICE  UNIT=1403,MODEL=N1,ADDRESS=00E
DEV00F    IODEVICE  UNIT=1403,MODEL=N1,ADDRESS=00F
CNT01   IOCONTRL    UNIT=2821,MODEL=5,ADDRESS=01,FEATURE=COLBNRY,     X
            TRNMODE=BYTE
DEV01A    IODEVICE  UNIT=2540R,MODEL=1,ADDRESS=01A
DEV01B    IODEVICE  UNIT=2540P,MODEL=1,ADDRESS=01B
DEV01F    IODEVICE  UNIT=1403,MODEL=N1,ADDRESS=01F
CNT02   IOCONTRL    UNIT=2702,ADDRESS=02
LINE020   IODEVICE  UNIT=1030,ADDRESS=020,ADAPTER=IBM2,SETADDR=2,     X
            FEATURE=AUTOPOLL
LINE021   IODEVICE  UNIT=1050,ADDRESS=021,ADAPTER=IBM1,SETADDR=1,     X
            FEATURE=AUTOPOLL
LINE022   IODEVICE  UNIT=1050,ADDRESS=022,ADAPTER=IBM1,SETADDR=1,     X
            FEATURE=AUTOPOLL
LINE023   IODEVICE  UNIT=1060,ADDRESS=023,ADAPTER=IBM1,SETADDR=1,     X
            FEATURE=AUTOPOLL
LINE024   IODEVICE  UNIT=83B3,ADDRESS=024,ADAPTER=TELE1,SETADDR=3
LINE025   IODEVICE  UNIT=115A,ADDRESS=025,ADAPTER=TELE1,SETADDR=3
LINE026   IODEVICE  UNIT=83B3,ADDRESS=026,ADAPTER=TELE1,SETADDR=3
LINE027   IODEVICE  UNIT=1050,ADDRESS=027,ADAPTER=IBM1,SETADDR=1,     X
            FEATURE=AUTOPOLL
LINE028   IODEVICE  UNIT=1060,ADDRESS=028,ADAPTER=IBM1,SETADDR=1,     X
            FEATURE=AUTOPOLL
LINE029   IODEVICE  UNIT=115A,ADDRESS=029,ADAPTER=TELE1,SETADDR=3
LINE02A   IODEVICE  UNIT=83B3,ADDRESS=02A,ADAPTER=TELE1,SETADDR=3
LINE02B   IODEVICE  UNIT=1060,ADDRESS=02B,ADAPTER=IBM1,SETADDR=1,     X
            FEATURE=AUTOPOLL
LINE02C   IODEVICE  UNIT=115A,ADDRESS=02C,ADAPTER=TELE1,SETADDR=3
LINE02D   IODEVICE  UNIT=1030,ADDRESS=02D,ADAPTER=IBM2,SETADDR=2,     X
            FEATURE=AUTOPOLL
LINE02E   IODEVICE  UNIT=1030,ADDRESS=02E,ADAPTER=IBM2,SETADDR=2,     X
            FEATURE=AUTOPOLL
CH1    CHANNEL    ADDRESS=1,TYPE=SELECTOR
DEV130    IODEVICE  UNIT=2314,ADDRESS=130
CNT18   IOCONTRL    UNIT=2803,MODEL=1,ADDRESS=18,                     X
            FEATURE=(DATACONV,7-TRACK)
DEV180    IODEVICE  UNIT=2402,MODEL=3,ADDRESS=180,FEATURE=7-TRACK
DEV181    IODEVICE  UNIT=2402,MODEL=3,ADDRESS=181,FEATURE=7-TRACK
DEV182    IODEVICE  UNIT=2402,MODEL=3,ADDRESS=182,FEATURE=9-TRACK
DEV183    IODEVICE  UNIT=2402,MODEL=3,ADDRESS=183,FEATURE=9-TRACK
```

Figure 44. Example 2: Input Deck for Stage I (Part 1 of 3)

```
CNT19   IOCONTRL     UNIT=2841,ADDRESS=19
DEV190    IODEVICE   UNIT=2311,ADDRESS=190
DEV191    IODEVICE   UNIT=2311,ADDRESS=191
DEV192    IODEVICE   UNIT=2311,ADDRESS=192
DEV193    IODEVICE   UNIT=2303,ADDRESS=193
CH2    CHANNEL    ADDRESS=2,TYPE=SELECTOR
DEV230    IODEVICE   UNIT=2314,ADDRESS=230
CNT28   IOCONTRL     UNIT=2803,MODEL=1,ADDRESS=28,                          X
              FEATURE=(DATACONV,7-TRACK)
DEVX80    IODEVICE   UNIT=2402,MODEL=3,ADDRESS=280,FEATURE=7-TRACK,        X
              OPTCHAN=3
DEVX81    IODEVICE   UNIT=2402,MODEL=3,ADDRESS=281,FEATURE=7-TRACK,        X
              OPTCHAN=3
DEVX82    IODEVICE   UNIT=2402,MODEL=3,ADDRESS=282,FEATURE=9-TRACK,        X
              OPTCHAN=3
DEVX83    IODEVICE   UNIT=2402,MODEL=3,ADDRESS=283,FEATURE=9-TRACK,        X
              OPTCHAN=3
DEVX84    IODEVICE   UNIT=2401,MODEL=3,ADDRESS=284,FEATURE=9-TRACK,        X
              OPTCHAN=3
DEVX85    IODEVICE   UNIT=2401,MODEL=3,ADDRESS=285,FEATURE=9-TRACK,        X
              OPTCHAN=3
DEVX86    IODEVICE   UNIT=2401,MODEL=3,ADDRESS=286,FEATURE=9-TRACK,        X
              OPTCHAN=3
DEVX87    IODEVICE   UNIT=2401,MODEL=3,ADDRESS=287,FEATURE=9-TRACK,        X
              OPTCHAN=3
CNT29   IOCONTRL     UNIT=2841,ADDRESS=29
CNT291    IODEVICE   UNIT=2311,ADDRESS=291
CNT292    IODEVICE   UNIT=2311,ADDRESS=292
CNT293    IODEVICE   UNIT=2311,ADDRESS=293
CNT294    IODEVICE   UNIT=2311,ADDRESS=294
CNT2F   IOCONTRL     UNIT=2848,MODEL=3,ADDRESS=2F,FEATURE=NODESCUR
DEV2F0    IODEVICE   UNIT=2260,MODEL=1,ADDRESS=2F0,FEATURE=ALKYB2260
DEV2F1    IODEVICE   UNIT=2260,MODEL=1,ADDRESS=2F1,FEATURE=NMKYB2260
DEV2F4    IODEVICE   UNIT=1053,MODEL=4,ADDRESS=2F4
CH3    CHANNEL    ADDRESS=3,TYPE=SELECTOR
CNT38   IOCONTRL     UNIT=2803,MODEL=1,ADDRESS=38,                          X
              FEATURE=(DATACONV,7-TRACK)
CNT32   IOCONTRL     UNIT=2841,ADDRESS=32
DEV320    IODEVICE   UNIT=2302,MODEL=3,ADDRESS=320
DEV321    IODEVICE   UNIT=2302,MODEL=3,ADDRESS=321
CNT35   IOCONTRL     UNIT=2701,ADDRESS=35
DEV351    IODEVICE   UNIT=2260,MODEL=1,FEATURE=ALKYB2260,ADAPTER=IBM3,    X
              ADDRESS=351
DEV355    IODEVICE   UNIT=2260,MODEL=1,FEATURE=ALKYB2260,ADAPTER=IBM3,    X
              ADDRESS=355
CNT3A   IOCONTRL     UNIT=2840,MODEL=1,ADDRESS=3A
DEV3A1    IODEVICE   UNIT=2250,MODEL=2,ADDRESS=3A1,NUMSECT=5,              X
              FEATURE=(ALKYB2250,LIGHTPEN,PRGMKYBD)
DEV3A2    IODEVICE   UNIT=2250,MODEL=2,ADDRESS=3A2,NUMSECT=5,              X
              FEATURE=(ALKYB2250,LIGHTPEN,PRGMKYBD)
DEV3A4    IODEVICE   UNIT=2282,ADDRESS=3A4,NUMSECT=5
DEV3E0    IODEVICE   UNIT=2250,MODEL=1,ADDRESS=2E0,                         X
              FEATURE=(ALKYB2250,BUFFER8K,LIGHTPEN,DESIGNFEAT)
          UNITNAME   UNIT=(180,181,182,183,280,281,282,283,291,292,293),X
              NAME=SYSSQ
          UNITNAME   UNIT=(190,191,192,193,291,292,293,294,235,236,237),X
              NAME=SYSDA
          UNITNAME   UNIT=COD,NAME=SYSCP
          UNITNAME   UNIT=(282,283,284,285,286,287),NAME=TAPE
          UNITNAME   UNIT=(320,321),NAME=RECORDS
          UNITNAME   UNIT=(134,135,136,137,234,235,236,237),NAME=FILE
```

Figure 44. Example 2: Input Deck for Stage I (Part 2 of 3)

```
          CTRLPROG  TYPE=MVT,MAXIO=61,QSPACE=20,ADDTRAN=4
          SCHEDULR  TYPE=PRIORITY,CONSOLE=009,ALTCONS=(I-00C,O-00F),    X
                OPTIONS=BYLABEL,STARTR=A-00D,STARTW=A-00E,              X
                ACCTRTN=SUPPLIED,STARTI=AUTO,WTLBFRS=10,PROCRES=232,    X
                JOBQRES=231,INITQBF=10,MINPART=54
          SUPRVSOR  RESIDNT=(BLDLTAB,RENTCODE,TRSVC),TRACE=100,         X
                OPTIONS=(PROTECT,COMM),TIMER=JOBSTEP,SER=SER1
          PROCLIB   UNIT=2314,VOLNO=MVT444
          DATAMGT   ACSMETH=(BDAM,ISAM,BTAM,QTAM)
          TELCMLIB  UNIT=2314,VOLNO=MVT444
          GRAPHICS
          EDITOR    DESIGN=E44
          ASSEMBLR  DESIGN=F
          TESTRAN   PHASES=(INTER,EDITOR),MODE=TRACE,PAGES=20,EXEC=200
          MACLIB    UNIT=2314,VOLNO=MVT444
          CHKPOINT
          SORTMERG  SIZE=24000,SORTOPT=FULLIB
          SORTLIB   UNIT=2314,VOLNO=MVT444
          ALGOL
          ALGLIB    UNIT=2314,VOLNO=MVT444
          COBOL     DESIGN=F
          COBLIB    DESIGN=F,UNIT=2314,VOLNO=MVT444
          FORTRAN   DESIGN=G
          FORTLIB   DESIGN=G,UNIT=2314,VOLNO=MVT444
          PL1       DESIGN=F
          PL1LIB    UNIT=2314,VOLNO=MVT444,LIBFCNS=COMPLEX
          RPG
          GENERATE  UT1SDS=SYS1.SG1,UT2SDS=SYS1.SG2,UT3SDS=SYS1.SG3,    X
                OBJPDS=SYS1.SGOBJ,RESNAME=2314,RESVOL=MVT111,           X
                RESTYPE=2314,LNKNAME=2314,LNKVOL=MVT222,LBMAINT=E,      X
                ASMPRT=ON,DIRDATA=PDS,LEPRT=(LIST,XREF)
          END
/*
//  START    RDR,182
```

Figure 44. Example 2: Input Deck for Stage I (Part 3 of 3)

Decks for System Residence on 2303

The 2303 unit at address 197 (see Figure 36) can be used for system residence instead of volume MVT111. The IBCDASDI deck in Figure 45 must replace the deck MVT111 in Figure 41. The serial number of the 2303 volume is SYSRES.

```
SYSRES JOB
       MSG      TODEV=1403,TOADDR=00E
       DADEF    TODEV=2303,TOADDR=197,IPL=YES,VOLID=SCRATCH
       VLD      NEWVOLID=SYSRES,OWNERID=DEPTD58
       VTOCD    STRTADR=2,EXTENT=8
       IPLTXT
   .
   .   IEAIPL00 cards
   .
.END
```

Figure 45. Example 2: IBCDASDI Deck for 2303

The DD statements in Figure 46 must replace the DD statements for SYS1.SVCLIB, SYSCTLG, and SYS1.NUCLEUS in Figure 43. The CATLG statements replace all the CATLG statements in Figure 43.

Only two changes must be made to the Stage I input deck in Figure 44:

- The SYS1.SG1 utility data set must be allocated space on SYSRES rather than on MVT111 as follows:

```
//SYSUT1  DD DSNAME=SYS1.SG1,VOLUME=(,RETAIN,SER=SYSRES),          X
//           DISP=(,CATLG),UNIT=2303,SPACE=(TRK,(180,16))
```

- The RESNAME, RESVOL, and RESTYPE parameters of the GENERATE macro instruction must be coded as follows:

```
RESNAME=2303,RESVOL=SYSRES,RESTYPE=2303
```

```
//SVCLIB   DD  DSNAME=SYS1.SVCLIB,VOLUME=(RETAIN,SER=SYSRES),          X
//             UNIT=2303,DISP=(,KEEP),SPACE=(TRK,(120,,75),,CONTIG),   X
//             LABEL=EXPDT=99350,DCB=(DSORG=POU,RECFM=U,BLKSIZE=1024)
//CATALOG  DD  DSNAME=SYSCTLG,VOLUME=(,RETAIN,SER=SYSRES),            X
//             UNIT=2303,DISP=(,KEEP),SPACE=(TRK,(8,1)),               X
//             LABEL=EXPDT=99350
//NUCLEUS  DD  DSNAME=SYS1.NUCLEUS,VOLUME=(,RETAIN,SER=SYSRES),       X
//             UNIT=2303,DISP=(,KEEP),SPACE=(TRK,(25,,1),,CONTIG),     X
//             LABEL=EXPDT=99350
       .
       .
       .
       .
       .
//SYSIN    DD   *              -INPUT FOR CATALOGING SYSTEM DATA SETS-
     CATLG   CVOL=2303=SYSRES,VOL=2314=MVT333,DSNAME=SYS1.SYSJOBQE
     CATLG   CVOL=2303=SYSRES,VOL=2314=MVT222,DSNAME=SYS1.LINKLIB
     CATLG   CVOL=2303=SYSRES,VOL=2303=SYSRES,DSNAME=SYS1.SVCLIB
     CATLG   CVOL=2303=SYSRES,VOL=2314=MVT444,DSNAME=SYS1.PROCLIB
     CATLG   CVOL=2303=SYSRES,VOL=2314=MVT444,DSNAME=SYS1.MACLIB
     CATLG   CVOL=2303=SYSRES,VOL=2314=MVT444,DSNAME=SYS1.TELCMLIB
     CATLG   CVOL=2303=SYSRES,VOL=2314=MVT444,DSNAME=SYS1.COBLIB
     CATLG   CVOL=2303=SYSRES,VOL=2314=MVT444,DSNAME=SYS1.SORTLIB
     CATLG   CVOL=2303=SYSRES,VOL=2314=MVT444,DSNAME=SYS1.PLILIB
     CATLG   CVOL=2303=SYSRES,VOL=2314=MVT444,DSNAME=SYS1.FORTLIB
     CATLG   CVOL=2303=SYSRES,VOL=2314=MVT444,DSNAME=SYS1.ALGLIB
     CATLG   CVOL=2303=SYSRES,VOL=2314=MVT222,DSNAME=SYS1.SYSVLOGX
     CATLG   CVOL=2303=SYSRES,VOL=2314=MVT222,DSNAME=SYS1.SYSVLOGY
/*
```

Figure 46. Example 2: Allocation on 2303

APPENDIX A: SYSTEM GENERATION MESSAGES

System generation messages are produced by the assembler program during the expansion of system generation macro-instructions. These messages are printed in the assembler listing in the SYSPRINT data set. Two types of messages are produced: error messages and informative messages.

ERROR MESSAGES

Table 7 shows the message code and format of system generation error messages. The messages follow.

IEIaaannn text

Explanation: The error indicated by the message text is a coding error in the system generation macro-instruction, aaa. The message serial number, nnn, identifies the message.

For the CHANNEL, IOCONTRL, and IODEVICE macro-instructions, the message text begins with either the name field of the macro-instruction or, if the name field was omitted, the sequential identification number provided by the system.

Examples of these messages are:

5,* * * IEICEN104 INSTSET VALUE NOT SPECIFIED

5,* * * IEICHA102 CHANNEL2-ADDRESS VALUE NOT SPECIFIED

5,* * * IEICHA102 CHAN#2-ADDRESS VALUE NOT SPECIFIED

The second example illustrates a message for a CHANNEL macro-instruction. "CHANNEL2" is the name field of the macro-instruction. The third example illustrates the same message, but in this case the name field of the macro-instruction was omitted and "CHAN#2" was supplied by the macro-instruction.

System Action: The assembler program did not produce a job stream in the SYSPUNCH data set. The program analyzed all remaining system generation macro-instructions and printed any other required messages. Either message IEIGEN113 or IEIGEN116 was printed, followed by the message: GENERATION TERMINATED. Then the system generation process was abnormally terminated.

Severity Code: 5

User Response: Correct the error or errors indicated and begin the system generation process from the start of Stage I.

IEIGEN113 QUIT SWITCH ON BEFORE GENERATE MACRO

Explanation: One or more errors, indicated by messages, were detected before the GENERATE macro-instruction was expanded.

Severity Code: 7

IEIGEN116 QUIT SWITCH SET IN GENERATE MACRO

Explanation: One or more errors were detected during expansion of the GENERATE macro-instruction.

Severity Code: 7

7, * * * GENERATION TERMINATED * * *

Explanation: The system generation process was abnormally terminated.

Severity Code: 7

INFORMATIVE MESSAGES

*, message text

Explanation: This type of message documents the options selected for the new system through the system generation macro-instructions. All options are described, whether the selection was explicit or implicit.

Table 7. System Generation Error Messages

Message Code	Code
IEI	s,* * * IEIaaannn text s = Severity code: 5 Error message; error in coding of a system generation macro-instruction. 7 Error message; message is produced by GENERATE macro-instruction. aaa = Indication of system generation macro-instruction at which error was detected: aaa — Macro-instruction ALG — ALGOL AGL — ALGLIB ASM — ASSEMBLR CEN — CENPROCS CHA — CHANNEL CHK — CHKPOINT COB — COBOL COL — COBLIB CTR — CTRLPROG DAT — DATAMGT EDI — EDITOR FOL — FORTLIB FTC — FORTRAN GEN — GENERATE GPH — GRAPHICS IOC — IOCONTRL IOD — IODEVICE LNK — LINKLIB MAL — MACLIB PLL — PL1LIB PL1 — PL1 PRL — PROCLIB RES — RESMODS RPG — RPG SCH — SCHEDULR SOL — SORTLIB SOR — SORTMERG SUP — SUPRVSOR SVC — SVCTABLE SVL — SVCLIB SYS — SYSUTILS TEL — TELCMLIB TES — TESTRAN UNI — UNITNAME nnn = Message serial number text= Message text

APPENDIX B: CROSS-REFERENCES IN MACRO-INSTRUCTIONS

This appendix shows in tabular form which macro-instructions and keywords refer to other macro-instructions or publications. Dependencies between the keywords of a macro-instruction are not shown because they are illustrated by the macro-instruction format and, when necessary, by tables within each macro-instruction description.

To facilitate reference to other publications, the titles are abbreviated as follows:

Abbreviation	Publication
	IBM System/360 Operating System:
COBOL F	COBOL (F) Programmer's Guide
Editor	Linkage Editor
FORTRAN E	FORTRAN IV (E) Programmer's Guide
FORTRAN G	FORTRAN IV (G) Programmer's Guide
FORTRAN H	FORTRAN IV (H) Programmer's Guide
JCL	Job Control Language
Operator	Operator's Guide
PL/I	PL/I (F) Programmer's Guide
SCB	System Control Blocks
Sort	Sort/Merge
SPG	System Programmer's Guide
Storage	Storage Estimates
Utilities	Utilities
2250	Graphic Programming Services for IBM 2250 Display Unit

Macro-Instruction	Keyword	Refers to	
		Macro-Instruction	Publication
IOCONTRL	UNIT	IODEVICE	
IODEVICE	All	IOCONTRL,UNITNAME	
	ADDRESS	CHANNEL	
	DEVTYPE		SPG,SCB
	OPTCHAN	CHANNEL	
	NUMSECT		2250
UNITNAME	UNIT	IODEVICE	
	NAME		SPG
CTRLPROG	TYPE	CENPROCS	
	OVERLAY	CENPROCS	
	LOWTASK	SCHEDULR,SUPRVSOR	
	HITASK	SUPRVSOR	
	QSPACE	SUPRVSOR	Storage

(Continued)

Macro-Instruction	Keyword	Refers to	
		Macro-Instruction	Publication
SCHEDULR	TYPE	CTRLPROG	
	CONSOLE	IODEVICE	
	ALTCONS	IODEVICE	
	OPTIONS		JCL
	STARTR	IODEVICE	
	STARTW	IODEVICE	
	ACCTRTN		SPG
	WTOBFRS	CTRLPROG	
	REPLY	CTRLPROG	
	PROCRES	IODEVICE	
	JOBQRES	IODEVICE	
	JOBQFMT		SPG
	JOBQLMT		SGP
	JOBQTMT		SPG
	INITQBUF	SUPRVSOR	Storage
	MINPART	SUPRVSOR	Storage
SUPRVSOR (For further details refer to Table 6)	RESIDENT	CTRLPROG	SPG
	OPTIONS	CTRLPTOG.SUPRVSOR	Operator
	WAIT	CTRLPROG	
	TIMER	CTRLPROG	
	TRACE		SPG
	SER	CENPROCS	Storage
SVCTABLE	All	RESMODS,SVCLIB	SPG
RESMODS	MEMBERS	SVCTABLE	SPG
SVCLIB	All	SVCTABLE	
	MEMBERS		SPG
PROCLIB	All	UNITNAME	
DATAMGT	ACSMETH	CTRLPROG SUPRVSOR TELCMLIB	

(Continued)

Macro-Instruction	Keyword	Refers to	
		Macro-Instruction	Publication
EDITOR	All		Utilities, Storage
ASSEMBLR	All		Utilities, Storage
SORTMERG	All	EDITOR, SORTLIB	
	SIZE	CENPROCS	Sort
	MESSAGE		Sort
SORTLIB	All	SORTMERG	
CHKPOINT	All	CTRLPROG	
ALGOL	All	CENPROCS, ALGLIB	
COBOL	All	CENPROCS, COBLIB	Utilities, Storage
	SIZE		COBOL F
FORTRAN	All	CENPROCS	Utilities, Storage
	DESIGN	CENPROCS	
	STORAGE	CTRLPROG	
	SIZE (E)		FORTRAN E
	SIZE (H)		FORTRAN H
FORTLIB	All	GRAPHICS	FORTRAN E
	OBJERR		FORTRAN E FORTRAN G FORTRAN H
	ONLNRD		FORTRAN G FORTRAN H
	ONLNPCH		FORTRAN G FORTRAN H
PL1	All	CENPROCS SUPRVSOR PL1LIB	
	SIZE		PL/I
GENERATE	GENTYPE		Operator
	LEPRT		Editor

APPENDIX C: GENERIC UNIT NAMES

Unit names are automatically assigned during system generation to collections of devices for each type of device specified by the UNIT parameter of an IODEVICE macro-instruction. The names and the devices to which they apply follow.

Magnetic Tape Drives

Unit Name	Device Type
2400	2400 9-track Magnetic Tape Drive having only an 800 byte-per-inch (density) capability
2400-1	2400 Magnetic Tape Drive with 7-track Compatibility and without Data conversion
2400-2	2400 Magnetic Tape Drive with 7-track Compatibility and Data conversion
2400-3	2400 or 2415 9-track Magnetic Tape Drive having only a 1600 byte-per-inch (density) capability
2400-4	2400 or 2415 9-track Magnetic Tape Drive having an 800 and 1600 byte-per-inch (density) capability

Direct-Access Devices

Unit Name	Device Type
2301	2301 Drum Storage
2302	Any 2302 Disk Storage Drive
2303	Any 2303 Drum Storage Drive
2311	2311 Disk Storage Drive
2314	2314 Direct-Access Storage Facility
2321	2321 Data Cell

Unit Record Equipment

Unit Name	Device Type
1052	1052 Printer-Keyboard
1403	1403 Printer or 1404 Printer (continuous form only)
1442	1442 Card Read Punch
1443	Any 1443 Printer
2501	Any 2501 Card Reader
2520	2520 Card Read Punch
2540	2540 Card Read Punch (read feed)
2540-2	2540 Card Read Punch (punch feed)
2671	2671 Paper Tape Reader

Graphics Devices

Unit Name	Device Type
1053	1053 Model 4 Printer
2250-1	2250 Model 1 Display Unit
2250-3	2250 Model 3 Display Unit
2260-1	2260 Model 1 Display Station (local attachment)
2260-2	2260 Model 2 Display Station (local attachment)
2280	2280 Film Recorder
2282	2282 Film Recorder/Scanner

APPENDIX D: SUPPORTING ADDITIONAL I/O DEVICES

The limits specified in the IOCONTROL, IODEVICE, and UNITNAME macro-instructions may be increased by following a special procedure. The procedure consists of two steps, namely:

- Redefining SGGBLPAK, a member of the data set SYS1.GENLIB, during the preparation for system generation.
- Inserting additional statements in the input deck to system generation.

The new limits will allow 248 I/O devices, 80 control units and 100 unit names. If these limits are set, system generation can only be performed on a central processing unit that has at least 128K bytes of main storage.

Redefining SGGBLPAK

SGGBLPAK can be redefined by executing the IEHPROGM utility program which is used during the preparation for system generation. The two utility control statements required are:

```
RENAME  DSNAME=SYS1.GENLIB,VOL=2311=DLIB02,MEMBER=SGGBLPAK,          X
        NEWNAME=SGPAK96

RENAME  DSNAME=SYS1.GENLIB,VOL=2311=DLIB02,MEMBER=SGPAK248,          X
        NEWNAME=SGGBLPAK
```

These statements can be included in the input deck for initializing the system data sets during the preparation for system generation, or in a separate execution of the program. The following DD statement must also be present in the step.

```
//DDCD DD DSNAME=SYS1.GENLIB,DISP=OLD
```

To reestablish the standard definitions after system generation, the following statements should be used with IEHPROGM:

```
RENAME  DSNAME=SYS1.GENLIB,VOL=2311=DLIB02,MEMBER=SGGBLPAK,          X
        NEWNAME=SGPAK248

RENAME  DSNAME=SYS1.GENLIB,VOL=2311=DLIB02,MEMBER=SGPAK96,           X
        NEWNAME=SGGBLPAK
```

In all the previous statements, it is assumed that SYS1.GENLIB resides on a 2311 direct-access volume whose serial number is DLIB02. If SYS1.GENLIB resides on a different volume or device, the volume serial number and device type should be modified accordingly. All other parameters must be coded as shown.

Inserting Additional Statements

The following two statements must be inserted in the input deck to the system generation process. They must immediately follow the DD * statement and precede any system generation macro-instructions. The statements are:

```
      COPY SGGBLPAK
&LIMIT(1)  SETB 1
```

IBM provides a starter operating system that can be used for the first system generation. The starter operating system must be initialized and made operational before it can be used as a generating system. The starter operating system and the procedures required to initialize it are described in this appendix.

THE STARTER OPERATING SYSTEM PACKAGE

The starter operating system package consists of an operating system and a set of libraries. The operating system includes:

- Control program
- Data set utilities
- System utilities
- Assembler F
- Linkage Editor E

In addition to the operating system, the starter operating system package includes the libraries listed below. The last four libraries listed (COBOL, FORTRAN, PL/I, and Sort libraries) are optional for customized distribution. Each library is a partitioned data set (PDS).

- SYS1.SAMPLIB (sample library): This library provides 80-character card images. It contains sample programs to test the functioning of components of the generated operating system, the IPL program (IEAIPL00), a dump program (PROCDUMP) and a non-executable example of writing an accounting routine (SAMACTRT). It also contains the following independent utility programs:

  1. IBCDASDI (initialization of direct-access volumes)
  2. IBCDMPRS (dump/restore)
  3. IBCRCVRP (recover/replace)

- SYS1.MODLIB (module library)
- SYS1.GENLIB (generation library) -- This library has a blocking factor of 42.
- SYS1.PROCLIB (procedure library)
- SYS1.MACLIB (macro library) -- This library has a blocking factor of 42.
- SYS1.COBLIB (COBOL library)
- SYS1.FORTLIB (FORTRAN library) -- This library is distributed without members.
- SYS1.PLILIB (PL/I library)
- SYS1.SORTLIB (sort library)

DISTRIBUTION METHODS

The starter operating system package is provided by one of the following distribution methods:

- Complete starter operating system package for the 2311 Disk Storage Drive: This package is distributed on three disk packs or on three tapes for restoring disk packs. The volume serial numbers of the disk packs are DLIB01, DLIB02, and DLIB03.

- Customized starter operating system package for the 2311 Disk Storage Drive: This package is distributed on two disk packs or on two tapes for restoring disk packs. The volume serial numbers of the disk packs are DLIB01 and DLIB02.

- Complete starter operating system package for the 2314 Disk Storage Facility: This package is distributed on one tape for restoring a disk pack. The volume serial number of the disk pack is DLIB01.

The starter operating system package distributed on disk packs is operable (though a backup copy of it should be made). It contains the starter operating system and the libraries as they are shown in Figure 47. (The data sets are not necessarily provided in the order shown.)

The starter operating system package distributed on tape includes all of the contents of the disk packs. However, since the tapes must be restored onto disk packs, two additional independent utility programs -- IBCDASDI and IBCDMPRS -- are provided at the beginning of each tape for use in restoring the disk packs. In Figure 47, the tape layout is shown beside the corresponding disk pack. Only the libraries shown on the disk packs are restored from the tapes. The IBCDASDI and IBCDMPRS programs will always be at the beginning of the tapes, but the other data sets may not be in the order shown. The 2314 distribution is available only on tape, and must be restored onto the disk pack from the tape.

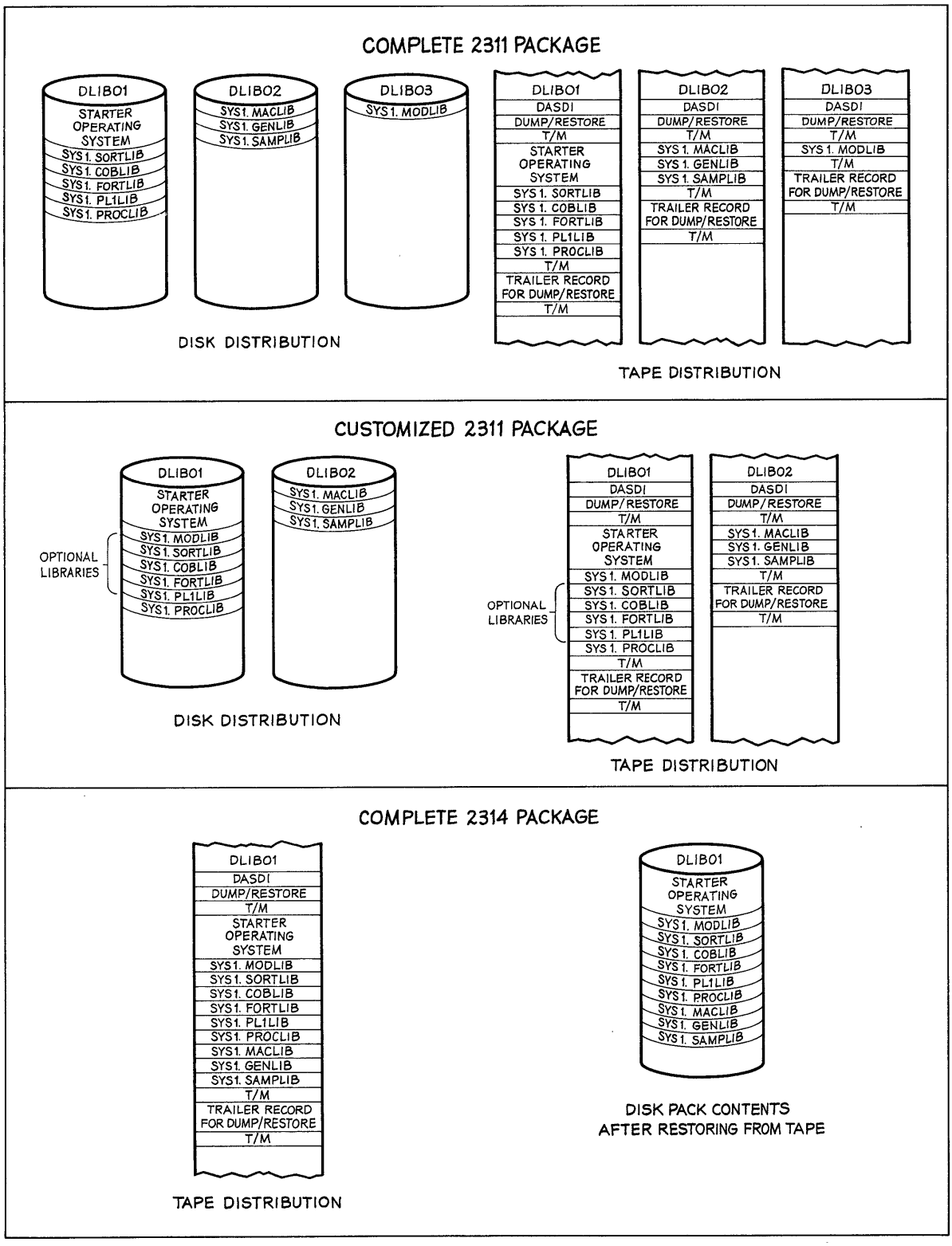


Figure 47. Arrangement of Data Sets

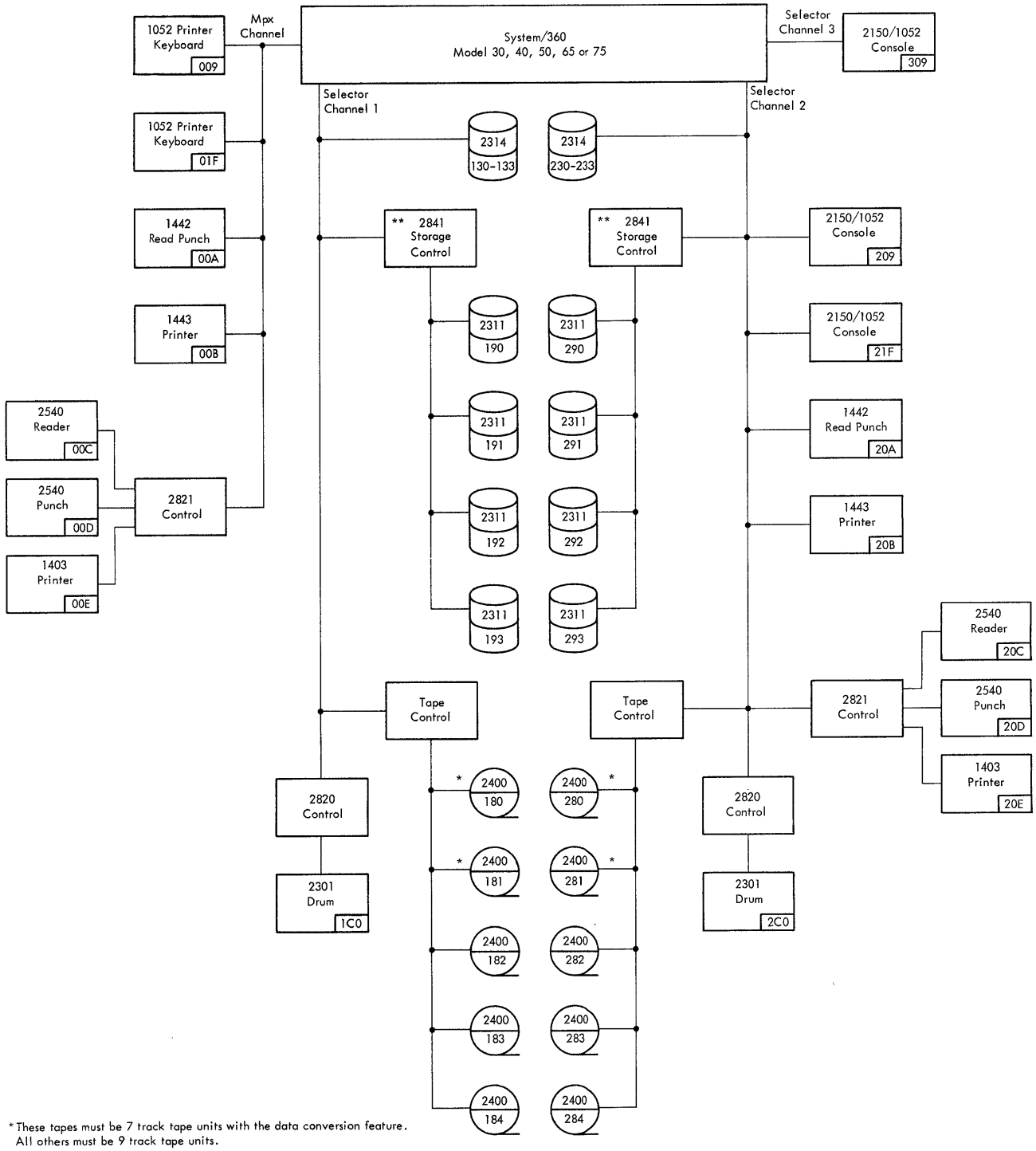
STARTER SYSTEM REQUIREMENTS

The starter operating system requires 64K bytes of main storage for operation. It contains a control program that can support the machine configuration shown in Figure 48 or any subset meeting the minimum requirements listed in Table 8.

The control program permits the use of the generic unit names listed in Table 9; additional unit names provided in the starter operating system are listed in Table 10.

The starter operating system requires that all devices to be used by the system be ready prior to IPL. Any device not ready will automatically be varied off line at IPL time. If any device which was not ready at IPL is required during a job step, the operator should enter a VARY ONLINE command for that device. When using the starter operating system, it is permissible to have devices other than those shown in Figure 48 attached to the system. However, an interruption must not be issued from any of these additional devices while the starter operating system is running. For example, the operator must not make ready any one of these devices.

If system generation is performed with the starter operating system using a 1403 printer with the universal character set feature, the BPS UCS utility program (360P-UT-048) must be executed prior to this in order to load the generator storage. When the UCS program is executed, NO-FOLDING and BLOCK-DATA-CHECK must be specified. This program is described in the publication IBM System/360 Basic Programming Support: Universal Character Set Utility Program Operating Guide, Form C24-3396.



*These tapes must be 7 track tape units with the data conversion feature. All others must be 9 track tape units.

**A 2303 may be substituted for any one of the 2311's attached to the 2841 Storage Controls on Selector Channel 1 and Channel 2. The 2303 addresses will be 197 and 297.

Figure 48. System Configuration

TABLE 8. MINIMUM I/O REQUIREMENTS

Minimum Requirement	Function	Choose from the following				
		Device	Device Address			
			MPX Channel	Selector Channel 1	Selector Channel 2	Selector Channel 3
1	System Console	1052	009,01F		209,21F	309
		2150/1052	009,01F		209,21F	309
2	Storage Units for System Residence and Data Sets	2311		190,191,192,193	290,291,292,293	
1	System Input	2540 Reader	00C		20C	
		1442 Read Punch*	00A		20A	
		2400 (7 Tr-DC)		180,181	280,281	
		2400 (9-Track)		182,183,184	282,283,284	
1	Punch Output	2540 Punch	00D		20D	
		1442 Read Punch*	00A		20A	
		2400 (7-Tr-DC)		180,181	280,281	
		2400 (9-Track)		182,183,184	282,283,284	
1	Print Output	1443	00B		20B	
		1403	00E		20E	
		2400 (7-Tr-DC)		180,181	280,281	
		2400 (9-Track)		182,183,184	282,283,284	
2**	Intermediate (Work) Data Sets	2311		190,191,192,193	290,291,292,293	
		2301		1C0	2C0	
		2400 (9-Track)		182,183,184	282,283,284	
2**	Intermediate (Work) Data Sets	2311		190,191,192,193	290,291,292,293	
		2301		1C0	2C0	
		2303		197	297	
1**	Intermediate (Work) Data Sets	2314		130-133	230-233	

* A single 1442 may serve as either system input or punch output, but not both simultaneously.

** The same direct access storage devices may serve for system residence and sequential and partitioned data sets if sufficient space is available.

Table 9. Generic Unit Names

Magnetic Tape Drives	
Unit Name	Device Type
2400	2400 Series 9-track Magnetic Tape Drive
2400-2	2400 Series Magnetic Tape Drive with Seven Track Compatibility and Data Conversion
Direct-Access Devices	
Unit Name	Device Type
2311	2311 Disk Storage Drive
2301	2301 Drum Storage
2303	2303 Drum Storage
2314	2314 Disk Storage Facility
Unit Record Equipment	
Unit Name	Device Type
1052	1052 Printer Keyboard
1403	1403 Printer 1404 Printer (continuous form only)
1442	1442 Serial Reader Punch
1443	1443 Printer
2540	2540 Reader Punch (read feed)
2540-2	2540 Reader Punch (punch feed)

Table 10. Additional Unit Names Supporting IBM Supplied Cataloged Procedures

Name	Function
SYSSQ	Sequential access on devices at any of the following addresses: 182, 183, 184, 282, 283, 284, 190, 191, 192, 193, 290, 291, 292, 293 (any 9-track tape or 2311 disk storage drive).
SYSDA	Direct access on devices at any of the following addresses: 190, 191, 192, 193, 290, 292, 293, (any 2311); 1C0, 2C0 (any 2301; 197, 297(any 2303): 130, 131, 132, 133, 230, 231, 232, 233 (any 2314).
SYSCP	A 2540 card punch at address 00D or 20D, or 1442 at address 00A or 20A.

PROCESSING THE STARTER PACKAGE

Before the starter operating system package can be used for system generation, it must be initialized and prepared for use. If the package is on tape, preparation consists of:

1. Restoring the system to disk (the tape then becomes a backup copy of the system).

2. Punching the independent utility programs and the sample programs from SYS1.SAMPLIB for later use.

3. Listing the data describing the system.

If the starter operating system package is on disk, preparation consists of:

1. Punching the independent utility programs and the sample programs from SYS1.SAMPLIB for later use.

2. Creating a backup copy of the system on disk or tape.

3. Listing the data describing the system.

Processing of the tape or disk distribution of the starter operating system package is depicted in Figure 49. Detailed processing instructions are contained in the procedure section that follows.

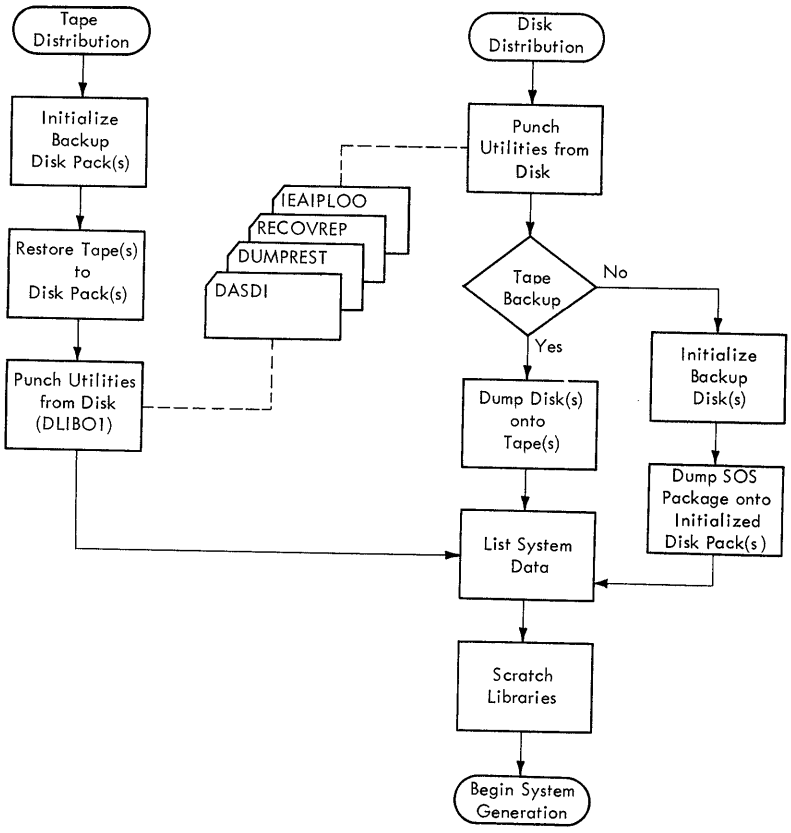


Figure 49. Processing the Starter Package

PROCEDURES FOR PROCESSING THE STARTER PACKAGE

This section contains detailed procedures for processing the various types of starter operating system packages. Procedures are given for:

- Complete or customized 2311 package -- tape distribution.
- Complete or customized 2311 package -- disk pack distribution with tape backup.
- Complete or customized 2311 package -- disk pack distribution with disk pack backup.
- Complete 2314 package -- tape distribution.

In addition, a procedure is given for deleting libraries from the starter operating system.

The procedures include examples of control decks. In these examples, the underlined fields are those that may require modification for different installations. Further explanation of the field requirements is contained in the publication IBM System/360 Operating System: Utilities, Form C28-6586. Users should remember to tailor the control decks for the distribution they receive: a complete 2311 package uses volume serial numbers DLIB01, DLIB02, and DLIB03; a customized 2311 package uses volume serial numbers DLIB01 and DLIB02; and a 2314 package uses only volume serial number DLIB01.

For illustrative purposes, the given procedures assume the set of devices and device addresses listed in Table 11.

Table 11. Sample Configuration

Device Function	Input/Output Device	Address
Printer Keyboard	1052	01F
System Residence and System Data Sets		
DLIB01	2311 (or 2314)	190 (or 130)
DLIB02	2311	191
DLIB03	2311	192
System Input	2540 Reader	00C
Punch Output	2540 Punch	00D
Print Output	1403 Printer	00E
Tape Drives[1]	2400 Tape	180[2]
	2400 Tape	181[2]
	2400 Tape	182[2]

[1]Tape units are required only if the starter package is received on tape or if system backup is created on tape.
[2]Tape units located at addresses 180 and 181 are 7-track tape units with the data conversion feature. The tape unit at 182 is a 9-track tape unit.

2311 Package -- Tape Distribution

The procedure for processing the tape distribution of the 2311 starter operating system package is as follows:

Initialize Disk

1. Mount the disk pack onto which DLIB01 is to be restored.
2. Mount tape #1 (DLIB01) of starter package.
3. Load IBCDASDI program from tape by setting the load selector switches and pressing the console LOAD key. When the program is loaded, the wait state is entered and the hexadecimal value FFFF is displayed in the console lights.
4. Place the following control deck in the input device to initialize the disk.

```
   JOB
   MSG       TODEV=1403,TOADDR=00E
   DADEF     TODEV=2311,TOADDR=190,VOLID=SCRATCH,                X
             FLAGTEST=NO
   VLD       NEWVOLID=111111,OWNERID=DEPT38
   VTOCD     STRTADR=50,EXTENT=10
   END
```

   In the DADEF statement, the FLAGTEST=NO parameter must only be used when the disk is initialized for the first time.

5. Define the control statement input device by pressing the REQUEST key of the printer keyboard. The message DEFINE INPUT DEVICE will be printed. Enter the message INPUT=xxxx cuu where xxxx is the device type, c is the channel address, an uu is the unit address. The device type can be 1442, 2400, or 2540.
6. When the disk initialization is complete, the message END OF JOB is printed on the message output device, and the program enters the wait state.

Restore Tape to DISK

7. Load the IBCDMPRS program from tape by setting the load selector switches and pressing the console LOAD key. When the program is loaded, the wait state is entered and the hexadecimal value FFFF is displayed in the console lights.
8. Place the following deck in the input device to restore the contents of the tape to the disk.

```
   JOB
   MSG       TODEV=1403,TOADDR=00E
   RESTORE   FROMDEV=2400,FROMADDR=180,TODEV=2311,               X
             TOADDR=190,VOLID=111111
   END
```

   when restoring is complete, the serial number of the disk has been changed from 111111 to DLIB01.

(Continued)

9. Define the control statement input device by pressing the REQUEST key of the printer keyboard. The message DEFINE INPUT DEVICE will be printed. Enter the message INPUT=xxxx cuu where xxxx is the device type, c is the channel address, and uu is the unit address. The device type can be 1442, 2400, or 2540.

10. When the restoring is complete, the message END OF JOB is printed on the message output device, and the program enters the wait state. The DLIB01 tape should be removed and stored in the tape library for backup purposes.

Initialize and Restore Second Disk

11. Repeat steps 1 through 10 to initialize and restore the second disk from tape #2 (DLIB02).

Initialize and Restore Third Disk

12. Repeat steps 1 through 10 to initialize and restore the third disk from tape #3 (DLIB03). Users of the customized starter package should skip this step.

Punch Utility and Sample Programs

13. Completion of the preceding steps provides operable disk packs with backup tapes. Proceed to punch the independent utility programs and any sample programs desired. Be sure all necessary volumes are mounted and make ready all devices to be used.

14. The following control deck should be placed in the input device.

```
//JOB1     JOB ACCT123,PROGRAMMER,MSGLEVEL=1
//       EXEC  PGM=IEBPTPCH
//SYSUT1   DD  DSNAME=SYS1.SAMPLIB,                              X
//             DISP=(OLD,KEEP),UNIT=2311,                        X
//             VOLUME=SER=DLIB02
//SYSUT2   DD  UNIT=2540-2
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
            PUNCH TYPORG=PO,MAXNAME=4
            MEMBER NAME=IBCDMPRS
            MEMBER NAME=IBCDASDI
            MEMBER NAME=IEAIPL00
            MEMBER NAME=IBCRCVRP
```

A MEMBER card should be added to the above control deck for each sample program desired for later use, and for SAMACTRT and PROCDUMP, provided the MAXNAME field in the PUNCH control card is adjusted to show the revised number of member cards in the deck. The member name card (first card) should be removed from each member deck punched. (Refer to the "Testing the New System" section of this publication for names of the sample programs.)

(Continued)

15. Set the LOAD UNIT switches on the control panel to the channel, control unit, and device of the system residence volume (DLIB01). Then press the LOAD key.

16. Wait for the READY message and for the WAIT light to be turned on. Then enter a SET command specifying the date, as follows:

a. Press REQUEST key.
b. Wait for READ light to go on.
c. Type SET DATE=yy.ddd to specify date.
d. Hold alternate coding key and press the numeric 5 key (EOB).

17. The starter operating system has built-in START RDR and START WTR commands, which are issued automatically. These commands are:

```
START RDR,00C
START WTR,00E
```

If these are not the addresses of the reader and writer to be used, override these commands by entering new ones for the proper devices.

18. Enter a START command with no parameters. When the job is complete, a READER CLOSED message will be printed, followed by a READY message. Then the system will enter the wait state.

List System Data

19. To list the data describing the system, place the proper control deck shown below in the input device. Then enter a START RDR command followed by a START command.

Control deck for users of the complete 2311 package:

```
//JOB2     JOB ACCT123,PROGRAMMER,MSGLEVEL=1
//STEP1  EXEC  PGM=IEHLIST
//SYSPRINT DD  SYSOUT=A
//DD1      DD  UNIT=2311,DISP=OLD,                              X
//            VOLUME=SER=DLIB01
//DD2      DD  UNIT=(191,,DEFER),DISP=(NEW,KEEP),               X
//            VOLUME=(PRIVATE,RETAIN)
//DD3      DD  UNIT=(192,,DEFER),DISP=(NEW,KEEP),               X
//            VOLUME=(PRIVATE,RETAIN)
//SYSIN    DD  *
           LISTCTLG
           LISTVTOC DUMP
           LISTVTOC DUMP,VOL=2311=DLIB02
           LISTVTOC DUMP,VOL=2311=DLIB03
           LISTPDS DSNAME=(SYS1.PROCLIB,SYS1.COBLIB
                    [,Optional PDSs on System Residence])
           LISTPDS VOL=2311=DLIB02,                             X
               DSNAME=(SYS1.GENLIB,SYS1.SAMPLIB
                    [,Optional PDSs on DLIB02])
           LISTPDS  VOL=2311=DLIB03,                            X
               DSNAME =SYS1.MODLIB
/*
```

(Continued)

Control deck for users of the customized 2311 package:

```
//JOB2     JOB ACCT123,PROGRAMMER,MSGLEVEL=1
//STEP1  EXEC  PGM=IEHLIST
//SYSPRINT DD  SYSOUT=A
//DD1      DD  UNIT=2311,DISP=OLD                                    X
//            VOLUME=SER=DLIB01
//DD2      DD  UNIT=(191,,DEFER),DISP=(NEW,DEEP),                    X
//            VOLUME=(PRIVATE,RETAIN)
//SYSIN    DD  *
            LISTCTLG
            LISTVTOC DUMP
            LISTVTOC DUMP,VOL=2311=DLIB02
            LISTPDS DSNAME=(SYS1.PROCLIB,SYS1MODLIB
                     [,Optional PDSs on System Residence])
            LISTPDS  VOL=2311=DLIB02,                               X
                DSNAME=(SYS1.GENLIB,SYS1.SAMPLIB
                     [,Optional PDSs on DLIB02])
/*
```

No MOUNT statement should be entered between IPL time and JOB2 for the drive on which DLIB02 is mounted. If JOB2 is run immediately after JOB1 (in step 14), eliminate the JOB2 card.

The physical address underlined in the DD2 statement should be the address of the device containing DLIB02. The physical address underlined in the DD3 statement should be the address of the device containing DLIB03.

In the LIST deck, any of the partitioned data sets shown in Figure 47 may be specified in the LISTPDS statement.

2311 Package -- Disk Pack Distribution (Tape Backup)

The procedure for processing the disk pack distribution of the 2311 starter operating system package, with tape backup, is as follows:

Punch Utility and Sample Programs

1. The disk packs are operable as received. Proceed to punch the independent utility programs (needed to create a tape backup copy of the disks) and any sample programs desired. Be sure all necessary volumes are mounted and make ready all devices to be used.

2. The following control deck should be placed in the input device.

```
//JOB1     JOB ACCT123,PROGRAMMER,MSGLEVEL=1
//        EXEC  PGM=IEBPTPCH
//SYSUT1   DD  DSNAME=SYS1.SAMPLIB,                              X
//             DISP=(OLD,KEEP),UNIT=2311,                        X
//             VOLUME=SER=DLIB02
//SYSUT2   DD  UNIT=2540-2
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
             PUNCH TYPORG=PO,MAXNAME=4
             MEMBER NAME=IBCDMPRS
             MEMBER NAME=IBCDASDI
             MEMBER NAME=IEAIPL00
             MEMBER NAME=IBCRCVRP
/*
```

A MEMBER card should be added to the above control deck for each sample program desired for later use, and for SAMACTRT and PROCDUMP, provided the MAXNAME field in the PUNCH control card is adjusted to show the revised number of member cards in the deck. The member name card (first card) should be removed from each member deck punched. (Refer to the "Testing the New System" section of this publication for names of the sample programs.)

3. Set the LOAD UNIT switches on the control panel to the channel, control unit, and device of the system residence volume (DLIB01). Then press the LOAD key.

4. Wait for the READY message and for the WAIT light to be turned on. Then enter a SET command specifying the date, as follows:

   a. Press REQUEST key.
   b. Wait for READ light to go on.
   c. Type SET DATE=yy.ddd to specify date.
   d. Hold alternate coding key and press the numeric 5 key (EOB).

5. The starter operating system has built-in START RDR and START WTR commands, which are issued automatically. These commands are:

```
START RDR, 00C
START WTR, 00E
```

If these are not the addresses of the reader and writer to be used, override these commands by entering new ones for the proper devices.

(Continued)

6.	Enter a START command with no parameters. When the job is complete, a READER CLOSED message will be printed, followed by a READY message. Then the system will enter the wait state.
	Dump Disk to Tape

7.	Place the IBCDMPRS program (punched from SYS1.SAMPLIB in Step 2) in the input device, followed by the control deck shown below.
	
	`JOB`
	`MSG  TODEV=1403,TOADDR=00E`
	`DUMP  FROMDEV=2311,FROMADDR=190,  X`
	`  TODEV=2400,TOADDR=181`
	`END`

8.	Mount the tape that is to contain the backup copy of the first disk pack (DLIB01).
9.	Load the IBCDMPRS program by setting the load selector switches and pressing the console LOAD key. When the program is loaded, the wait state is entered and the hexadecimal value FFFF is displayed in the console lights.
10.	Define the control statement input device by pressing the REQUEST key of the printer keyboard. The message DEFINE INPUT DEVICE will be printed. Enter the message INPUT=xxxx cuu where xxxx is the device type, c is the channel address, and uu is the unit address. The device type can be 1442, 2400, or 2540.
11.	When dumping onto the tape is complete, the message END OF JOB is printed on the message output device, and the program enters the wait state. The tape should be removed and stored in the tape library.
	Dump Second Disk to Tape
12.	Repeat steps 7 through 11 to dump the contents of the second disk pack (DLIB02) onto a second tape.
	Dump Third Disk to Tape
13.	Repeat steps 7 through 11 to dump the contents of the third disk pack (DLIB03) onto a third tape. Users of the customized starter package should skip this step.

(Continued)

List System Data

14. To list the data describing the system, load the operating system, and place the proper control deck shown below in the input device. Then enter a START command with no parameters.

Control deck for users of the complete 2311 package:

```
//JOB2     JOB ACCT123,PROGRAMMER,MSGLEVEL=1
//STEP1  EXEC PGM=IEHLIST
//SYSPRINT DD  SYSOUT=A
//DD1      DD  UNIT=2311,DISP=OLD,                                    X
//           VOLUME=SER=DLIB01
//DD2      DD  UNIT=(191,,DEFER),DISP=(NEW,KEEP),                     X
//           VOLUME=(PRIVATE,RETAIN)
//DD3      DD  UNIT=(192,,DEFER),DISP=(NEW,KEEP),                     X
//           VOLUME=(PRIVATE,RETAIN)
//SYSIN    DD  *
           LISTCTLG
           LISTVTOC DUMP
           LISTVTOC DUMP,VOL=2311=DLIB02
           LISTVTOC DUMP,VOL=2311=DLIB03
           LISTPDS DSNAME=(SYS1.PROCLIB,SYS1.COBLIB
                     [,Optional PDSs on System Residence])
           LISTPDS VOL=2311=DLIB02,                                   X
               DSNAME=(SYS1.GENLIB,SYS1.SAMPLIB
                     [,Optional PDSs on DLIB02])
           LISTPDS VOL=2311=DLIB03,                                   X
               DSNAME=SYS1.MODLIB
/*
```

Control deck for users of the customized 2311 package:

```
//JOB2     JOB ACCT123,PROGRAMMER,MSGLEVEL=1
//STEP1  EXEC  PGM=IEHLIST
//SYSPRINT DD  SYSOUT=A
//DD1      DD  UNIT=2311,DISP=OLD,                                    X
//           VOLUME=SER=DLIB01
//DD2      DD  UNIT=(191,,DEFER),DISP=(NEW,KEEP),                     X
//           VOLUME=(PRIVATE,RETAIN)
//SYSIN    DD  *
           LISTCTLG
           LISTVTOC DUMP
           LISTVTOC DUMP,VOL=2311=DLIB02
           LISTPDS DSNAME=(SYS1.PROCLIB,SYS1MODLIB
                     [,Optional PDSs on System Residence])
           LISTPDS VOL=2311=DLIB02,                                   X
               DSNAME=(SYS1.GENLIB,SYS1.SAMPLIB
                     [,Optional PDSs on DLIB02])
/*
```

The physical address underlined in the DD2 statement should be the address of the device containing DLIB02. The physical address underlined in the DD3 statement should be the address of the device containing DLIB03.

In the LIST deck, any of the partitioned data sets shown in Figure 47 may be specified in the LISTPDS statement.

2311 Package -- Disk Pack Distribution (Disk Backup)

The procedure for processing the disk pack distribution of the 2311 starter operating system package, with disk backup, is as follows:

Punch Utility and Sample Programs

1. The disk packs are operable as received. Proceed to punch the independent utility programs (needed to create a disk backup copy of the disks) and any sample programs desired. Be sure all necessary volumes are mounted and make ready all devices to be used.

2. The following control deck should be placed in the input device.

```
//JOB1     JOB ACCT123,PROGRAMMER,MSGLEVEL=1
//        EXEC  PGM=IEBPTPCH
//SYSUT1   DD  DSNAME=SYS1.SAMPLIB,                              X
//             DISP=(OLD,KEEP),UNIT=2311,                        X
//             VOLUME=SER=DLIB02
//SYSUT2   DD  UNIT=2540-2
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
            PUNCH TYPORG=PO,MAXNAME=4
            MEMBER NAME=DUMPREST
            MEMBER NAME=DASDI
            MEMBER NAME=IEAIPL00
            MEMBER NAME=RECOVREP
/*
```

A MEMBER card should be added to the above control deck for each sample program desired for later use, provided the MAXNAME field in the PUNCH control card is adjusted to show the revised number of member cards in the deck. The member name card (first card) should be removed from each member deck punched. (Refer to the "Testing the New System" section of this publication for names of the sample programs.)

3. Set the LOAD UNIT switches on the control panel to the channel, control unit, and device of the system residence volume (DLIB01). Then press the LOAD key.

4. Wait for the READY message and for the WAIT light to be turned on. Then enter a SET command specifying the date, as follows:

   a. Press REQUEST key.
   b. Wait for READ light to go on.
   c. Type SET DATE=yy,ddd to specify date.
   d. Hold alternate coding key and press the numeric 5 key (EOB).

5. The starter operating system has built-in START RDR and START WTR commands, which are issued automatically. These commands are:

```
START RDR, 00C
START WTR, 00E
```

If these are not the addresses of the reader and writer to be used, override these commands by entering new ones for the proper devices.

(Continued)

6.	Enter a START command with no parameters. When the job is complete, a READER CLOSED message will be printed, followed by a READY message. Then the system will enter the wait state.
	Initialize Backup Disk
7.	Place the IBCDASDI program (punched from SYS1.SAMPLIB in Step 2) in the input device, followed by the control deck shown below. JOB MSG TODEV=1403,TOADDR=00E DADEF TODEV=2311,TOADDR=190,VOLID=SCRATCH, X FLAGTEST=NO VLD NEWVOLID=111111,OWNERID=DEPT38 VTOCD STRTADR=50,EXTENT=10 END In the DADEF statement, the FLAGTEST=NO parameter must be used when the disk is initialized for the first time only.
8.	Mount the disk pack that is to receive the backup copy of the first starter disk pack (DLIB01).
9.	Load the IBCDASDI program by setting the load selector switches and pressing the console LOAD key. When the program is loaded, the wait state is entered and the hexadecimal value FFFF is displayed in the console lights.
10.	Define the control statement input device by pressing the REQUEST key of the printer keyboard. The message DEFINE INPUT DEVICE will be printed. Enter the message INPUT=xxxx cuu where xxx is the device type, c is the channel address, and uu is the unit address. The device type can be 1442,2400, or 2540.
11.	When the disk initialization is complete, the message END OF JOB is printed on the message output device, and the program enters the wait state.
	Restore Backup Disk
12.	Place the IBCDMPRS program (punched from SYS1.SAMPLIB in Step 2) in the input device, followed by the control deck shown below. JOB MSG TODEV=1403,TOADDR=00E DUMP FROMDEV=2311,FROMADDR=190,TODEV=2311, X TOADDR=191,VOLID=111111 END When restoring is complete, the serial number of the disk has been changed from 111111 to DLIB01.
13.	Load the IBCDMPRS program by setting the load selector switches and pressing the console LOAD key. When the program is loaded, the wait state is entered and the hexadecimal value FFFF is displayed in the console lights.

(Continued)

14. Define the control statement input device by pressing the REQUEST key of the printer keyboard. The message DEFINE INPUT DEVICE will be printed. Enter the message INPUT=xxxx cuu where xxxx is the device type, c is the channel address, and uu is the unit address. The device type can be 1442,2400, or 2540.

15. When the dumping is completed, the message END OF JOB is printed on the message output device, and the program enters the wait state.

Initialize and Restore Second Backup Disk

16. Repeat steps 7 through 15 to create a backup copy of the second disk pack (DLIB02).

Initialize and Restore Third Backup Disk

17. Repeat steps 7 through 15 to create a backup copy of the third disk pack (DLIB03). Users of the customized starter package should skip this step.

List System Data

18. To list the data describing the system, load the operating system, and place the proper control deck shown below in the input device. Then enter a START command with no parameters.

Control deck for users of the complete 2311 package:

```
//JOB2     JOB ACCT123,PROGRAMMER,MSGLEVEL=1
//STEP1  EXEC PGM=IEHLIST
//SYSPRINT DD  SYSOUT=A
//DD1      DD  UNIT=2311,DISP=OLD,                                  X
//           VOLUME=SER=DLIB01
//DD2      DD  UNIT=(191,,DEFER),DISP=(NEW,KEEP),                   X
//           VOLUME=(PRIVATE,RETAIN)
//DD3      DD  UNIT=(192,,DEFER),DISP=(NEW,KEEP),                   X
//           VOLUME=(PRIVATE,RETAIN)
//SYSIN    DD  *
           LISTCTLG
           LISTVTOC DUMP
           LISTVTOC DUMP,VOL=2311=DLIB02
           LISTVTOC DUMP,VOL=2311=DLIB03
           LISTPDS DSNAME=(SYS1.PROCLIB,SYS1.COBLIB
                     [,Optional PDSs on System Residence])
           LISTPDS VOL=2311=DLIB02,                                 X
               DSNAME=(SYS1.GENLIB,SYS1.SAMPLIB
                     [,Optional PDSs on DLIB02])
           LISTPDS   VOL=2311=DLIB03,                               X
               DSNAME=SYS1.MODLIB
/*
```

(Continued)

Control deck for users of the customized 2311 package:

```
//JOB2     JOB ACCT123,PROGRAMMER,MSGLEVEL=1
//STEP1  EXEC  PGM=IEHLIST
//SYSPRINT DD  SYSOUT=A
//DD1      DD  UNIT=2311,DISP=OLD,                                  X
//            VOLUME=SER=DLIB01
//DD2      DD  UNIT=(191,,DEFER),DISP=(NEW,KEEP),                   X
//            VOLUME=(PRIVATE,RETAIN)
//SYSIN    DD  *
            LISTCTLG
            LISTVTOC DUMP
            LISTVTOC DUMP,VOL=2311=DLIB02
            LISTPDS DSNAME=(SYS1.PROCLIB,SYS1MODLIB
                      [,Optional PDSs on System Residence])
            LISTPDS   VOL=2311=DLIB02,                             X
                DSNAME=SYS1.GENLIB,SYS1.SAMPLIB
                      [,Optional PDSs on DLIB02])
/*
```

The physical address underlined in the DD2 statement should be the address of the device containing DLIB02. The physical address underlined in the DD3 statement should be the address of the device containing DLIB03.

In the LIST deck, any of the partitioned data sets shown in Figure 47 may be specified in the LISTPDS statement.

2314 Package -- Tape Distribution

The procedure for processing the tape distribution of the 2314 starter operating system package is as follows:

Initialize Disk

1. Mount disk pack onto which DLIB01 is to be restored.

2. Mount the tape distribution (DLIB01) of the starter package.

3. Load the IBCDASDI program from the tape by setting the load selector switches and pressing the console LOAD key. When the program is loaded, the wait state is entered and the hexadecimal value FFFF is displayed in the console lights.

4. Place the following control deck in the input device to initialize the disk.

```
   JOB
   MSG      TODEV=1403,TOADDR=00E
   DADEF    TODEV=2314,TOADDR=130,VOLID=SCRATCH,                X
            FLAGTEST=NO
   VLD      NEWVOLID=111111,OWNERID=DEPT38
   VTOCD    STRTADR=50,EXTENT=10
   END
```

   In the DADEF statement, the FLAGTEST=NO parameter must be used when the disk is initialized for the first time only.

5. Define the control statement input device by pressing the REQUEST key of the printer keyboard. The message DEFINE INPUT DEVICE will be printed. Enter the message INPUT=xxxx cuu where xxxx is the device type, c is the channel address, and uu is the unit address. The device type can be 1442, 2400, or 2540.

6. When the disk initialization is complete, the message END OF JOB is printed on the message output device, and the program enters the wait state.

Restore Tape to Disk

7. Load the IBCDMPRS program from tape by setting the load selector switches and pressing the console LOAD key. When the program is loaded, the wait state is entered and the hexadecimal value FFFF is displayed in the console lights.

8. Place the following control deck in the input device to restore the contents of the tape to the disk.

```
   JOB
   MSG      TODEV=1403,TOADDR=00E
   RESTORE  FROMDEV=2400,FROMADDR=180,TODEV=2314,               X
            TOADDR=130,VOLID=111111
   END
```

   When restoring is complete, the serial number of the disk has been changed from 11111 to DLIB01.

(Continued)

9. Define the control statement input device by pressing the REQUEST key of the printer keyboard. The message DEFINE INPUT DEVICE will be printed. Enter the message INPUT=xxxx cuu where xxxx is the device type, c is the channel address, and uu is the unit address. The device type can be 1442, 2400, or 2540.

10. When the restoring is complete, the message END OF JOB is printed on the message output device, and the program enters the wait state.

    The DLIB01 tape should be removed and stored in the tape library for backup purposes.

Punch Utility and Sample Programs

11. Completion of the preceding steps provides an operable disk pack with a backup tape. Proceed to punch the independent utility programs and any sample programs desired. Be sure the necessary volumes are mounted and make ready all devices to be used.

12. The following control deck should be placed in the input device.

```
//JOB1     JOB ACCT123,PROGRAMMER,MSGLEVEL=1
//       EXEC  PGM=IEBPTPCH
//SYSUT1   DD  DSNAME=SYS1.SAMPLIB,                              X
//            DISP=(OLD,KEEP),UNIT=2314,                         X
//            VOLUME=SER=DLIB01
//SYSUT2   DD  UNIT=2540-2
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
            PUNCH TYPORG=PO,MAXNAME=4
            MEMBER NAME=IBCDMPRS
            MEMBER NAME=IBCDASDI
            MEMBER NAME=IEAIPL00
            MEMBER NAME=IBCRCVRP
/*
```

    A MEMBER card should be added to the above control deck for the sample program desired for later use, and for SAMACTRT and PROCDUMP, provided the MAXNAME field in the PUNCH control card is adjusted to show the revised number of member cards in the deck. The member name card (first card) should be removed from each member deck punched. (Refer to the "Testing the New System" section of this publication for names of the sample programs.)

13. Set the LOAD UNIT switches on the control panel to the channel, control unit, and device of the system residence volume (DLIB01). Then press the LOAD key.

14. Wait for the READY message and for the WAIT light to be turned on. Then enter a SET command specifying the date, as follows:

    a. Press REQUEST key.
    b. Wait for READ light to go on.
    c. Type SET DATE=yy.ddd to specify date.
    d. Hold alternate coding key and press the numeric 5 key (EOB).

(Continued)

15. The starter operating system has built-in START RDR and START WTR commands, which are issued automatically. These commands are:

```
START RDR,00C
START WTR,00E
```

If these are not the addresses of the reader and the writer to be used, override these commands by entering new ones for the proper devices.

16. Enter a START command with no parameters. When the job is complete, a READER CLOSED message will be printed, followed by a READY message. Then the system will enter the wait state.

List System Data

17. To list the data describing the system, place the control deck shown below in the input device. Then enter a START RDR command followed by a START command.

```
//JOB2     JOB ACCT123,PROGRAMMER,MSGLEVEL=1
//STEP1  EXEC  PGM=IEHLIST
//SYSPRINT DD  SYSOUT=A
//DD1      DD  UNIT=2314,DISP=OLD,                                    X
//           VOLUME=SER=DLIB01
//SYSIN    DD  *
           LISTCTLG
           LISTVTOC DUMP
           LISTPDS DSNAME=(SYS1.PROCLIB,                              X
                SYS1.MODLIB,SYS1.GENLIB,                              X
                SYS1.SAMPLIB
                      ([,Optional PDSs on System Residence])
/*
```

If JOB2 IS RUN IMMEDIATELY AFTER JOB1 (in step 12), eliminate the JOB2 card.

In the LIST deck, any of the partitioned data sets shown in Figure 47 may be specified in the LISTPDS statement.

Deleting Libraries

To make additional direct-access storage available for the system generation process, you may elect to delete the SYS1.SAMPLIB library from your operable pack before starting the system generation process. This may be accomplished by loading the operating system and using the control deck shown in the input device.

```
//JOB2      JOB ACCT123,PROGRAMMER,MSGLEVEL=1
//STEP2    EXEC PGM=IEHPROGM
//SYSPRINT DD  SYSOUT=A
//DD1       DD  UNIT=(191,,DEFER),DISP=(NEW,KEEP),                    X
               VOLUME=(PRIVATE,RETAIN)
//SYSIN     DD  *
            SCRATCH DSNAME=SYS1.SAMPLIB.                              X
               VOL=2311=DLIB02,PURGE
/*
```

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

READER'S COMMENT FORM

IBM System/360 Operating System
System Generation

Form C28-6554-3

- Is the material:

	Yes	No
Easy to read?	☐	☐
Well organized?	☐	☐
Complete?	☐	☐
Well illustrated?	☐	☐
Accurate?	☐	☐
Suitable for its intended audience?	☐	☐

- How did you use this publication?
  - ☐ As an introduction to the subject
  - ☐ For additional knowledge

  Other . . . . . .

- Please check the items that describe your position:
  - ☐ Customer personnel
  - ☐ IBM personnel
  - ☐ Manager
  - ☐ Systems Analyst
  - ☐ Operator
  - ☐ Programmer
  - ☐ Customer Engineer
  - ☐ Instructor
  - ☐ Sales Representative
  - ☐ Systems Engineer
  - ☐ Trainee

  Other . . . . . .

- Please check specific criticism(s), give page number(s), and explain below:
  - ☐ Clarification on page(s) . . . . . .
  - ☐ Addition on page(s) . . . . . .
  - ☐ Deletion on page(s) . . . . . .
  - ☐ Error on page(s) . . . . . .

Explanation:

- Thank your for your cooperation. No postage necessary if mailed in the U.S.A.

YOUR COMMENTS PLEASE . . .

This publication is one of a series which serves as reference for systems analysts, programmers and operators of IBM systems. Your answers to the questions on the back of this form, together with your comments, will help us produce better publications for your use. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Please note: Requests for copies of publications and for assistance in utilizing your IBM system should be directed to your IBM representative or to the IBM sales office serving your locality.

Fold Fold

FIRST CLASS
PERMIT NO. 81
POUGHKEEPSIE, N.Y.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

POSTAGE WILL BE PAID BY

IBM Corporation
P.O. Box 390
Poughkeepsie, N.Y. 12602

Attention: Programming Systems Publications
Department D58

Fold Fold

Printed in U.S.A. C28-6554-3

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

IBM Technical Newsletter

File Number S360-31

Re: Form No. C28-6554-3

This Newsletter No. N28-2269

Date September 19, 1967

Previous Newsletter Nos. None

IBM SYSTEM/360 OPERATING SYSTEM
SYSTEM GENERATION

This technical newsletter corresponds to Release 13 and contains amendments to the System Generation publication. Replacement and/or supplemental pages to be inserted in the publication are noted below. Corrections and additions to text and/or illustrations are indicated by a vertical bar to the left of the text or illustrations and a bullet (•) to the left of an illustration caption.

Pages to Be Inserted	Pages to Be Removed
73-74	73-74
95-96	95-96

Summary of Amendments

This technical newsletter describes the generation of the Graphics Subroutine Package (GSP) for FORTRAN IV.

Note: Please file this cover letter at the back of the publication. Cover letters provide a quick reference to amendments and a means of checking receipt of all amendments.

IBM Corporation, Programming Systems Publications, P.O. Box 390, Poughkeepsie, N.Y. 12602

PRINTED IN U.S.A.

OKwdg

IBM Technical Newsletter

File Number S360-31

Re: Form No. C28-6554-3

This Newsletter No. N28-2282

Date November 15, 1967

Previous Newsletter Nos. N28-2269

IBM SYSTEM/360 OPERATING SYSTEM
SYSTEM GENERATION

This technical newsletter corresponds to Release 14 and contains amendments to the System Generation publication. Replacement and/or supplemental pages to be inserted in the publication are noted below. Corrections and additions to text and/or illustrations are indicated by a vertical bar to the left of the text or illustration and a bullet (•) to the left of the illustration caption.

Pages to Be Inserted	Pages to Be Removed
23,24	23,24
43-54.2	43-54
57-60	57-60
91-98	91-98
103-106.1	103-106
111,112	111,112
115-116.2	115,116
123,124	123,124
131,132	131,132
143,144	143,144
155,156	155,156
173,174	173,174

Summary of Amendments

This technical newsletter modifies the IOCONTRL, IODEVICE, UNITNAME, SCHEDULR, SUPRVSOR, FORTRAN, and GENERATE macro instructions. The graphics sample program and the list of generic unit names have been updated. The use of the IEBEDIT utility program to restart Stage II is also described. Figures 26 and 36 have been reprinted to facilitate references to them.

Note: Please file this cover letter at the back of the publication. Cover letters provide a quick reference to changes, and a means of checking receipt of all amendments.

IBM Corporation, Programming Systems Publications, P.O. Box 390, Poughkeepsie, N.Y. 12602

PRINTED IN U.S.A.