File No. S360-31
Order No. GC28-2010-7

IBM Systems Reference Library

IBM Time Sharing System

System Generation and Maintenance

This publication explains how an installation specifies, creates, maintains, and modifies an installation-adapted IBM Time Sharing System. The intended audience is the system programmer charged with system generation and maintenance.

Part I is an overall description of system generation and maintenance; Part II describes the creation of a basic time-sharing system; Part III describes the macro instructions used to adapt the basic system to the installation's needs; Part IV explains the system maintenance process, and Part V shows how to modify the system for the duration of a startup-to-shutdown session. The appendixes contain examples of system generation, maintenance, and modification, reference material on device addressing and system libraries, a sample machine configuration, and sample system generation macro instructions.

Prerequisite Publications

The reader must be familiar with the information presented in: IBM Time Sharing System: Concepts and Facilities, GC28-2003

Eighth Edition (December 1976)

This is a major revision of, and makes obsolete, GC28-2010-6.

This edition is current with Release 3.0 of IBM Time Sharing System/370 (TSS/370), and remains in effect for all subsequent versions or modifications of TSS unless otherwise noted. Significant changes or additions to this publication will be provided in new editions or Technical Newsletters.

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 this form has been removed, comments may be addressed to IBM Corporation, Time Sharing System, Dept. 80M, 1133 Westchester Avenue, White Plains, New York 10604.

PREFACE

This publication explains how to specify, create, maintain, and modify an installation-adapted IBM Time Sharing System. This book is intended for system programmers who perform system generation and maintenance.

Part I, the Introduction, presents the procedures to be followed for generation and maintenance of IBM Time Sharing System (TSS). The reader is guided to later sections by indications of what is accomplished at each step in the procedures.

Part II describes the use of the initial release of TSS to create a basic time-sharing system under which an individual installation's system can be generated.

Part III describes the system generation method of assembling macro instructions that specify an installation's machine configuration and system parameters, together with the methods of applying them to the system.

Part IV explains how the system-maintenance process modifies the installation's TSS with IBM-supplied changes.

Part V presents a method of dynamically modifying TSS during startup, to test the effects of system modifications before they are permanently inserted into the system.

The appendixes contain examples of system generation, maintenance, and dynamic modification based on a sample time-sharing machine configuration. Also, the appendixes include reference material on device addressing, the system libraries, and examples of system-generation macro instructions.

PREREQUISITE READING

This publication assumes a general knowledge of TSS from IBM Time Sharing System: Concepts and Facilities, GC28-2003.

SUGGESTED READING

The command system for TSS users is described in IBM Time Sharing System: Command System User's Guide, GC28-2001.

The command system facilities for managers and administrators are in IBM Time Sharing System: Manager's and Administrator's Guide, GC28-2024.

The programs for direct access storage device initialization and dump/restore are discussed in IBM Time Sharing System: Independent Utilities, GC28-2038.

For operators, information on the startup procedure and the command system facilities is in IBM Time Sharing System: Operator's Guide, GC28-2033.

A list of all messages issued during the system generation and maintenance processes, and directions for actions to be taken, are in IBM Time Sharing System: System Messages, GC28-2037.

Definitions of all TSS macro instructions, except those for system generation, are in either IBM Time Sharing System: Assembler User Macro Instructions, GC28-2004 or in IBM Time Sharing System: System Programmer's Guide, GC28-2008.

The TSS assembler language is in IBM Time Sharing System: Assembler Language, GC28-2000.

The TSS linkage editor is described in IBM Time Sharing System: Linkage Editor, GC28-2005.

CONTENTS

The IBM Time Sharing System (TSS) is adapted specifically to a user's installation by the system generation process that defines the installation's hardware configuration, its task-management parameters, and its command-system options. TSS may be modified by the system maintenance process, which applies IBM-supplied maintenance packages to the system data sets, or TSS may be modified by the dynamic modification process for the duration of one startup-to-shutdown session.

BASIC SYSTEM

An operable version of TSS is needed to run the system generation process. To construct a basic system, IBM supplies each installation with magnetic tapes that contain such a version.

Using disk initialization and tape-to-disk dump/restore utility programs (See Figure 1), the IBM-supplied restore tape for the initial public volume is written onto a disk. Then, one additional system disk and any drum that will be used for paging are initialized.

The basic system contains three pre-joined users; the system operator, the system manager, and the system programmer. At the completion of disk and drum initialization, and the public volume restore, the operator can start the Time Sharing System, as described in the Operator's Guide.

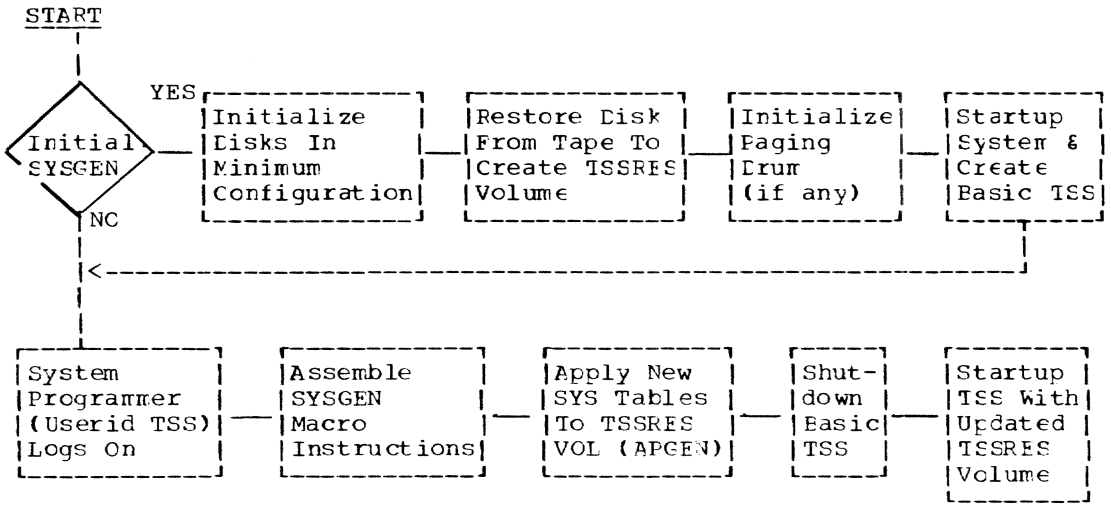


Figure 1. Time Sharing System Generation Procedures

SYSTEM GENERATION

An installation-adapted TSS is created by using the system generation procedure: the specification (via SYSGEN macro instructions) of data for system tables that contain configuration-dependent information and some installation parameters; the assembly of these macro instructions into control sections containing system tables; and the application of these system control sections to the operational TSS .

Using the basic system, the prejoined system programmer (who has access to all system data sets) logs on at a terminal (other than the operator's) and assembles the system generation macro instructions (described in Part III). When the assembly is terminated, he ascertains that he has an error-free assembly that accurately reflects the installation's parameters and configuration-dependent tables.

Using the object module produced by his assembly, the system programmer executes the APGEN procedure (described in Part III). This procedure modifies the system by replacing certain system modules with the content of the SYSGEN assembly object module. This procedure completes the modification of the system. Subsequent system startups will now reflect the installation specified SYSGEN data.

After an installation has been operating under TSS for some time, some of the installation's parameters may need to be changed. Another system generation will be required to make the changes. The procedure, shown in the bottom side of Figure 1, is a subset of the original SYSGEN procedure. Since an operable TSS exists, the system programmer (userid TSS) can log on immediately, assemble the system generation macro instructions, and execute the APGEN procedure. After this process has been completed, TSS must be shut down; then, using the newly updated TSSRES volume, the system operator must restart TSS.

SYSTEM MAINTENANCE

The TSS linkage editor and command system are used to modify system data sets according to the IBM-supplied maintenance packages; installations that have not made their own changes and are using a standard system can maintain their systems by using the IBM-supplied maintenance packages (change tape and change script).

However, installations that have made their own changes to the standard system can modify the change script. This type of system alteration should be attempted only by a system programmer who is thoroughly familiar with the interfaces among the system components.

The change scripts (command procedures contained in data sets) are then run to apply the maintenance changes. When the system maintenance run has been completed, the modified version of the system is made available by shutting down TSS and restarting the new version of the system from the IPL volume. This procedure is shown in Figure 2.

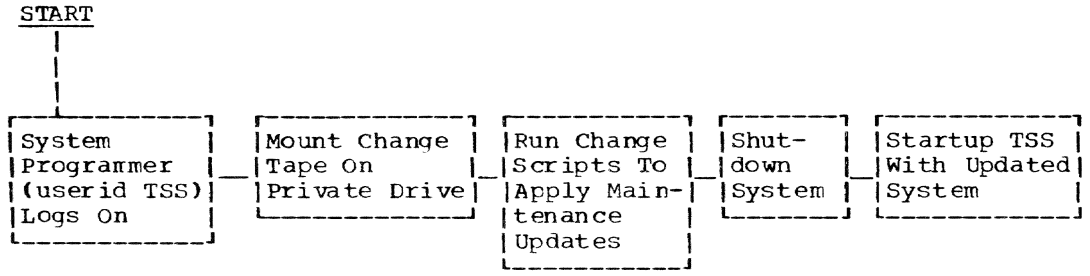


Figure 2. Time Sharing System Maintenance Procedures

DYNAMIC MODIFICATION

TSS is designed so that the contents of initial virtual storage and resident supervisor can be dynamically modified during startup. Changes to the initial virtual storage, the resident supervisor, and the resident support system routines can be tested without permanently updating the system. All changes must be placed in delta data set form on one VAM volume. Changes made by dynamically modifying the system remain in effect until a SHUTDOWN command is issued.

The procedures to be followed and the specifications to be made during startup are detailed in Part V.

PART II: BUILDING THE BASIC SYSTEM

SECTION 1: INITIAL RELEASE PACKAGE

The initial release package contains the material necessary to build a basic TSS under which the system can be generated (SYSGEN). Two nine-track tapes contain the utility programs and data sets that constitute the Time Sharing System. Two 3330 Model 1 volumes (or larger) are needed to start up the basic TSS.

The TSSRES (public volume zero) restore tape contains the data sets needed for system residence. A detailed description of all system data sets is given in Appendix B. Four of these data sets (RESSUP, SYSIVM, RSSSUP, and SYSCCB) contain tables that describe the minimum machine configuration. These tables were generated by the SYSGEN described in Appendix D.

Volume Preparation

The steps required for the initial system generation are shown in Figure 3. Before building the system, two 3330 Model 1 (or larger) direct access volumes and a 2305 Drum (if any) must be VAM formatted by the Direct Access Storage Device Initialization Program (DASDI). This stand-alone utility program is described in Independent Utilities. One of the disk volumes is initialized as a public VAM volume and the other disk as a paging volume. When the disks have been initialized, the Direct Access Storage Device Dump/Restore (DASDDR) Program (see Independent Utilities) can be loaded from the utility tape. Using suitable control cards the TSSRES restore tape can be written onto the newly initialized public volume. This disk becomes the first public volume (relative public volume zero) of the basic system.

When the volumes have been readied, the operator sets the selector switches for the TSSRES volume and presses the LOAD key. This procedure, called IPL (initial program loading), reads in Prelude (a loader for the Independent Access Method, IAM. The operator requests that a TSS long start be performed. IAM then locates, reads, and passes control to the STARTUP module.

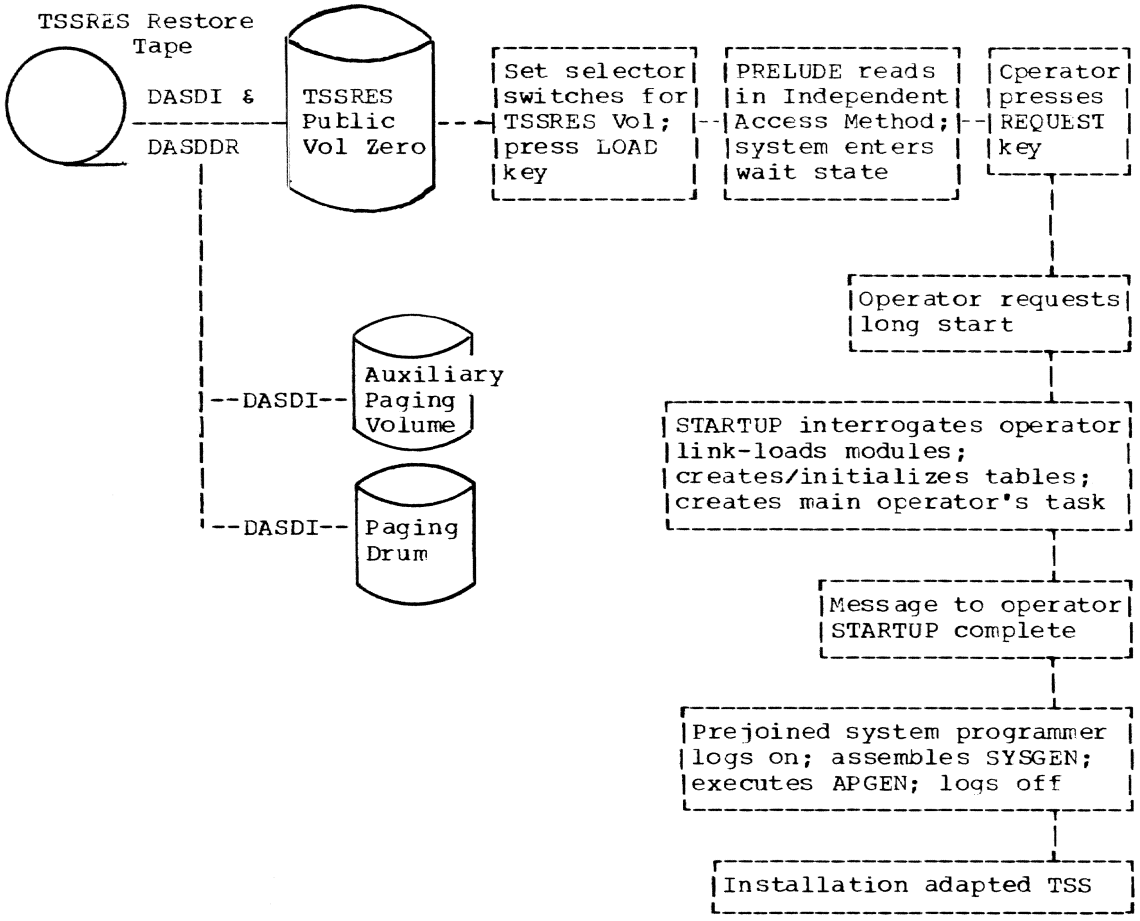


Figure 3. Initial System Generation

SECTION 2: STARTUP

Startup, the module that performs the operations necessary for initializing TSS, has three primary functions:

1. Link-load the CSECTs in the initial virtual storage (from the SYS-IVM data set), in the resident supervisor (from the RESSUP data set), and in the resident support system (from the RSSSUP data set);

2. Create or initialize various tables;

3. Create the main-operator task and pass control to it.

During the Startup procedure, all public volumes must be mounted; the system catalog must be on one of these volumes.

Initially, Startup asks the address of the card reader. If the address is entered and there are cards in it, Startup reads the cards in answer to the questions below. If the address is defaulted, the following is a summary of a conversation between the operator and the system.

Startup interrogates the operator for the address of the paging disk. During link-loading, Startup creates storage maps for initial virtual

storage, resident supervisor, and resident support system. (The resident supervisor and resident support system are link-loaded together.) The message

"PRINT MAPS?"

asks the operator's direction for the disposition of these maps.

The message

ENTER CODE FOR FUNCTIONS NOT TO BE LOADED

prompts the operator to exclude certain modules or functions from the load list. The last byte of the last word in each entry in the load list contains the code, which must have been preassigned (see the description of the LLIST macro instruction in System Programmer's Guide). Modules containing Q-refs may be loaded into initial virtual storage. If two Q-refs of the same name are encountered and their attributes do not agree, Startup issues a message on the printer:

ATTRIBUTES OF Q-REF xxxxxxxx IN MODULE yyyyyyyy DO NOT AGREE WITH PREVIOUS Q-REF OF THE SAME NAME -- FIRST REFERENCE USED.

If a CSECT in a user module is privileged, the privileged attribute is ignored and a message is issued on the printer:

THE PRIVILEGED ATTRIBUTE OF CSECT xxxxxxxx WILL BE IGNORED.

If a CSECT in a user module has the system attribute, the attribute is ignored and a message is issued on the printer:

WARNING USER CSECT xxxxxxxx HAS SYSTEM ATTRIBUTE.

If duplicate CSECT names are encountered when loading a user module, the duplicate CSECT is not loaded and a message is issued:

CSECT xxxxxxxx IN MODULE yyyyyyyy HAS DUPLICATE NAME OF PREVIOUSLY LOADED CSECT -- FIRST CSECT USED.

User modules in initial virtual storage must have all relocatable constants resolved (that is, there must be no calls to modules outside initial virtual storage); if an unresolved reference is processed by Startup, a warning message is issued on the printer:

UNRESOLVED REF xxxxxxxx IN MODULE yyyyyyyy.

Startup checks for the validity of the PMD of a module to be loaded into real or virtual storage; if an invalid PMD is found, a message is issued on the printer:

PMD CSECT/MODULE xxxxxxxx INVALID -- CSECT/MODULE IGNORED.

The operator then receives:

QUICK START REQUESTED? IF Y ENTER ADDR OF PACK FOR QUICK START DATA SET---EOB=N.

The operator may reply with a three- or four-character physical device address when the Quickstart data set is to be created, or with EOB to signify that creation of a Quickstart data set is not wanted. The address must specify a device on which is mounted a public or private VAM volume (perhaps the TSSRES volume) that is on-line during the startup operation and that has been initialized (DASDI) with a copy of the Prelude on cylinder 0, track 1, record 1. If a paging pack is specified, Startup issues the message

QUICK START VOLUME ON PAGING DRIVE QUICK START CREATION ABORTED

and regular processing continues. Only one Quickstart data set may reside on a pack: subsequent Quickstart data sets created on the same pack will overlay the existing data set. If sufficient space is not available for the data set, the message

INSUFFICIENT SPACE FOR QUICK START DATA SET QUICK START CREATION ABORTED

is issued and regular processing continues.

The request for

"DELTA DATA SETS?"

permits the operator to initiate the dynamic modification procedure (described in Part V). Upon completion of link-loading, the following message is then issued on the printer:

THE PSEUDO-REGISTER VALUE AT THE COMPLETION OF STARTUP IS xxxxxxxx BYTES, if there are Q-refs.

Then the operator verifies/enters the date and time. The Resident Support System (RSS) is entered by Startup to allow patches to be made before any TSS system operation. The requests for BULKIO and system hardware configuration validation complete the Startup process. A message stating that the startup process has been completed will be sent to the operator. The prejoined system programmer can logon and perform the system generation procedure as described in Part III.

There are several program switches in STARTUP which can be set by the system programmer to affect the operation of STARTUP. These switches are located and set using the independent VAM2 utility (VAM2UT):

Entry Point Name	Original Contents	Set To	Causes
REMIPL	47F0	4700	Question is issued to the operator asking if private IPL volume is to be removed.
RSSCALL	8200	4700	Eliminates call of RSS for patch cards.
PRINTALL	58D0	4700	STARTUP cards are printed on operator console.
JMPSWT	00	FF	Bypasses printing of STARTUP cards.
PRIEQPUB	4710	47F0	Makes all on-line volumes appear public.
		4780	Reverses meaning of public/private bit on volume (see note below).

Note: The PRIEQPUB switch can be used to operate a one-volume system, leaving the volumes from the normal user system on-line. The one-volume system can be used for test purposes, maintenance on the public storage of the user system, or for copying all the user public volumes as part of a vital records retention program. Public volumes will be recognized as private volumes without operator intervention.

SECTION 3: QUICKSTART

Provided that a Quickstart data set was created during the initial Startup (see above), the operator may initiate a Quickstart, a fast initialization procedure. After a Quickstart data set has been created (during the initial startup procedure), any subsequent IPL can be made from the Quickstart volume. If that volume should be the TSSRES volume, IAM inquires if Quickstart or Startup is requested:

```
System:    ?
Operator:  QS
```

If the Quickstart volume is not the TSSRES volume, Startup informs the operator

```
QUICK START IN PROGRESS.
```

If an unrecoverable I/O error occurs during the Quickstart process, the operation terminates with the message

```
ERROR READING QUICK START DATA SET
QUICK START TERMINATED
```

The hardware configuration may be changed for a Quickstart except that the physical addresses occupied by the resident supervisor routines during the creation of the Quickstart data set must be available during Quickstart.

Note: A Quickstart data set has VAM sequential organization, thereby allowing system access to the data set. The data set name is TSS*****.QKSTART.DSxxxxxx, where xxxxxx is the volume identification of the pack. The data set is cataloged if it is created on a public volume. The Quickstart data set may be created on any VAM-formatted volume (other than the paging volume) that has been initialized (DASDI) with a copy of Prelude on cylinder 0, track 1, record 1. A new Quickstart data set should be created whenever a new or revised module is added to the system.

The Independent Access Method (IAM) data set has a VPAM organization and is created and cataloged, if created on a public volume, whenever a Quickstart dataset is created. The IAM data set name is TSS*****.SYSIAM.DSxxxxxx, where xxxxxx is the volume identification of the pack.

SECTION 1: SYSGEN

The system generation procedure adapts a basic TSS to a specific installation's machine configuration, and to its task-management and command-system system options. Information about the installation is supplied in the SYSGEN macro instructions by the system programmer. The assembly of these macro instructions results in a module containing control sections. The APGEN procedure applies these control sections to update the tables in initial virtual storage, resident support system, resident supervisor and configuration control block.

The source coding for the SYSGEN module is a set of SYSGEN macro instructions. GENSCB must be the last macro instruction. At least one of each macro is required.

The module name supplied by the prejoined system programmer during the assembly process must be SYSGEN. The system generation macro instructions are defined in the GENMAC and GENNDX data sets. Before assembling the macro instructions, the ASMMAC, ASMNDX, GENMAC, and GENNDX data sets must be defined and the ddnames must be supplied to the assembler as supplemental libraries and indexes. (Assembly parameter explanations are in Command System User's Guide.) If a user, other than the prejoined system programmer (userid TSS), assembles the system generation macro instructions, he must have issued SHARE commands for ASMMAC, ASMNDX, GENMAC, and GENNDX.

The SYSGEN macro instructions may be assembled conversationally from a terminal, or in a nonconversational task (see Figure 4). If the system programmer is assembling conversationally, the macro instructions may be entered from the terminal or they may be defined, by using command system facilities, in a data set that was prestored on a public volume.

From the terminal, the assembler creates a source data set for the macro instructions as it processes them. From a prestored data set, successive lines are fetched and processed by the assembler. In both cases, the SYSGEN module must be assembled into a JOBLIB with SYSGEN.MODULE as its dsname.

In both cases, if any errors are detected prior to processing GENSCB, the macro instructions may be corrected and reentered; after processing GENSCB, the system programmer must rerun the complete assembly. However, if the system programmer is assembling in nonconversational mode and a macro instruction error is found, the assembler will process subsequent macro instructions -- but the complete assembly must then be rerun (see Appendix G). The TSS assembly procedures are described in the Command System User's Guide.

The linkage editor is used in the APGEN procedure to separate the SYSGEN object module into initial virtual storage, resident support system, resident supervisor, and configuration control block control sections that are then used to update, in place, the appropriate system data sets. These control sections contain tables for use by the system.

Section 2, "System Generation Macro Instructions," details each macro instruction and its parameters. Appendix A gives the format of each macro instruction with a short definition of the symbolic mnemonics used; and Appendix E, a sample list for the multi-processor machine configuration (pictured in Appendix D). The conventions used to code the

system generation macro instructions are consistent with the requirements of the assembler.

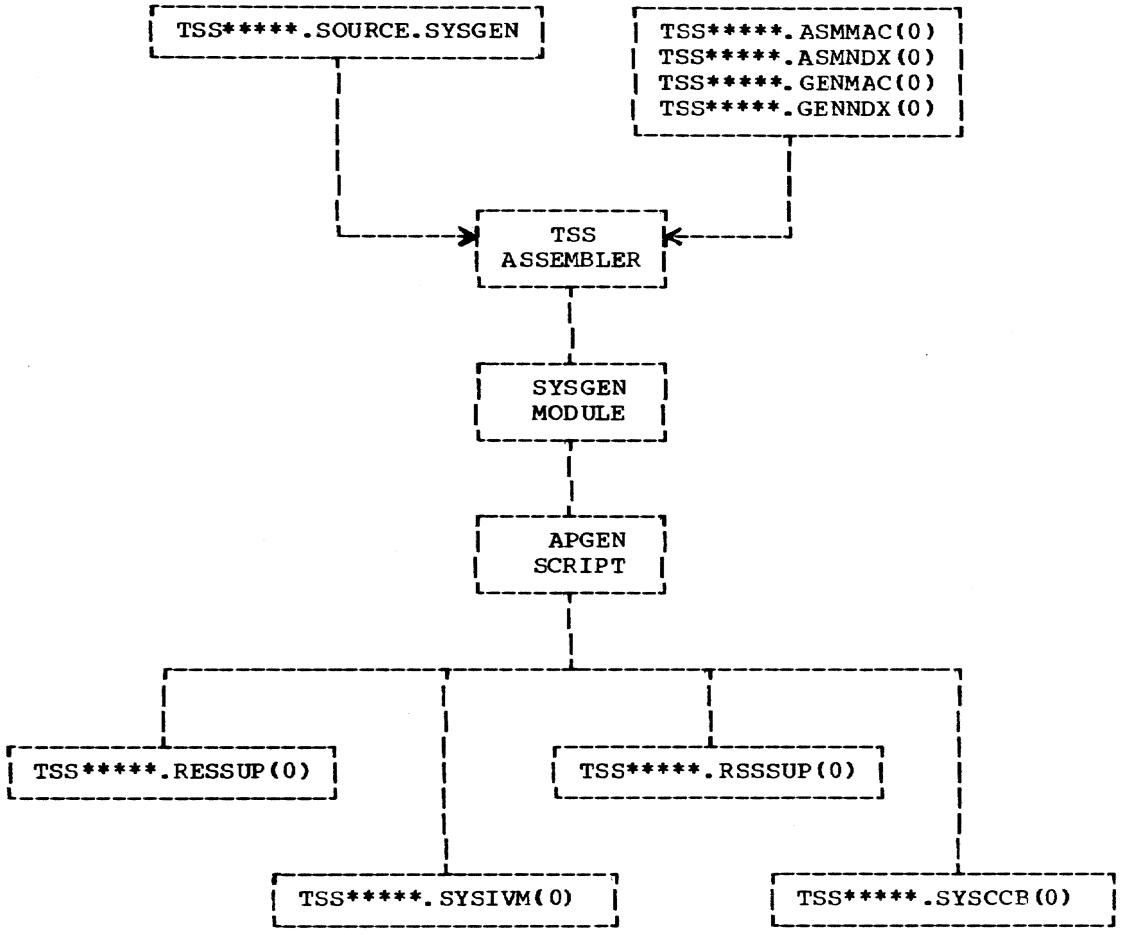


Figure 4. System Generation Phase

SECTION 2: SYSTEM GENERATION MACRO INSTRUCTIONS

Detailed descriptions of all system generation macro instructions are presented in this section, categorized by function: configuration, task-management and command-system, and system-table generation. Within each subsection, the descriptions are in alphabetical order, and each description has this arrangement:

1. Macro instruction name.
2. Brief statement of the macro instruction's functions.
3. Illustration of the macro instruction's format.
4. Detailed description of each operand.
5. System default value; when none specified, there is no system default.
6. Programming notes, where applicable, to highlight proper use, restrictions, and limitations.

CONFIGURATION MACRO INSTRUCTIONS

The configuration macro instructions must be specified in the following order: CPU, STEM, CHANNEL, DCU and DEVGRP. Unless the macro instructions accurately reflect the installation's configuration and all its device features, TSS will not work efficiently; system errors may occur. For example, such an error would occur if the asynchronous feature was omitted from a description of a terminal device.

Other features, such as control units and switches, are required equipment for TSS . No mention need be made of such features or equipment in the macro instruction specifications, unless specifically required by the macro instruction; their existence is implied when the associated equipment is specified.

If, subsequent to the initial system generation run, new devices are to be added to the system, another system generation run is necessary. All the information supplied by the initial system generation macro instructions, plus the additional device information, must be used for the second system generation run.

CHANNEL -- Describe Channels on CPU

The CHANNEL macro instruction specifies the addresses of the byte and block multiplexer and selector channels in the system configuration.

Name	Operation	Operand
[symbol]	CHANNEL	CPU={0|1},MPX=(hexinteger,...) [,SEL=(hexinteger,...)] [,BMX=(hexinteger,...)]

CPU

specifies the address of the CPU associated with the channels.

Specified as: one hexadecimal digit, 0 or 1

MPX

specifies the addresses of byte multiplexer channels.

Specified as: one hexadecimal digit, 0 to D

SEL
specifies the addresses of selector channels.

Specified as: one hexadecimal digit from 0 to D

BMX
specifies the addresses of block multiplexer channels

Specified as: one hexadecimal digit from 0 to D

Programming Notes: One CHANNEL macro is required for each CPU with channels attached. The channel addresses are mutually exclusive on each CPU.

CPU -- Describe Central Processing Units

The CPU macro instruction describes the number of central processing units.

Name	Operation	Operand
[symbol]	CPU	nocpu-{1|2}

nocpu
specifies the number of CPUs, including attached processors, in the system configuration.

Specified as: 1 or 2

DCU -- Describe a Device Control Unit

The DCU macro instruction identifies an individual device control unit in the system configuration. Input/output devices that include their own control units also need a DCU macro instruction.

Name	Operation	Operand
cuname	DCU	UNIT=code,PATH=(hexinteger-hexinteger,...) [,MODEL={1|2|3|4|5|6|7|8}] [,SHARED={Y|N}]

cuname
specifies the symbolic name by which this control unit will be known. This control unit name may be used as an operand to the VARY command. Each control unit must have a unique cuname.

Specified as: one to eight character alphameric name

UNIT
specifies one of the control unit types. The 1052 unit must be specified only for the operator consoles, which are hard-wired terminals. NONSTD indicates a control unit that TSS software does not support but which can be handled via IOREQ (described in Assembler User Macro Instructions).

Specified as: 1052, 2314, 2403, 2501, 2701, 2702, 2703, 2803, 2821, 2835, 2840, 2841, 3066, 3213, 3215, 3272, 3505, 3704, 3705, 3800, 3803, 3811, 3830, NONSTD

PATH
specifies the base control unit address and the range of addresses to which the control unit will answer. The address is a four digit hexadecimal number. The first digit of the control unit address is the CPU address, the second digit is the channel address, the last two digits are the base address of the control unit. The range is a dash followed by a two digit hexadecimal number that specifies the largest possible address which is answered by the control unit.

Specified as: Four hexadecimal digits followed by a dash followed by two hexadecimal digits.

Programming Notes: The range must include all addresses that the control unit will respond to, even if the devices are not specified in a DEVGRP macro or even if the physical devices are not installed on the control unit.

MODEL
specifies the control unit model number. This operand must be provided if the model number exists.

Specified as: one hexadecimal digit from 1 to 8

SHARED
specifies whether or not a NONSTD control unit has a single sense register shared among attached devices.

Specified as: Y or N

System Default: N

DEVGRP -- Describe a Device Group

The DEVGRP macro instruction describes a group of devices of the same type that are connected to the same control units.

Name	Operation	Operand
gpname	DEVGRP	UNIT=code,PATH=((cuname,hexinteger),...)
		,ADDRESS=((hexinteger,hexinteger),...)
		[,MODEL=code]
		[,FEATURE=(code,...)]
		[,MAXIO=integer]
		[,TYPE={RESERVED|ASYNCH}]
		[,IOREQ={YES|NO}]
		[,CONTRL={2701|2702|2703}]
		[,HOLD={Y|N}] [,DDSDA={Y|N}]
		[,ABUF=integer] [,DCM=integer]

gpname
specifies the symbolic name by which this device group will be known. This device group name may be used as an operand to the

VARY command. Several DEVGRP macro instructions may be used to logically describe one device group by repeating the gpname. Examples of string switched devices are given in Appendix J.

Specified as: one to eight character alphameric name

UNIT

specifies the device type. The 1052 printer-keyboard, 1053 printer and 1056 card reader are included under the 1050 unit designation. The punch and reader functions of the IBM 2540 card read-punch are referenced by 2540P and 2540R, which are separately addressed and must be listed in separate DEVGRP macro instructions. The 3330 Model 11 is specified as 333B. NONSTD indicates a device that TSS software does not support but which can be handled via ICREQ (described in Assembler User Macro Instructions).

Specified as: 1050, 1403, 2305, 2311, 2314, 2401, 2402, 2403, 2501, 2540P, 2540R, 2741, 2780, 3066, 3211, 3213, 3215, 3270, 3270C, 3330, 333B, 3350, 3420, 3505, 3525, 3704, 3705, 3800, TTY33, TTY35, NONSTD

Programming Note: When sysgening the 148 or 158 operator's console as a 3270, the unit value in the DEVGRP macro must be given as:

UNIT=3270C

This denotes a 3270 without a PAI key. RTAM will thus assume that an exclamation (!) character entered by itself is an attention request.

PATH

specifies the list of control units which can access the device group. Each path entry specifies the control unit name (specified in the DCU macro instruction) and that portion of the hardware address contributed by the device group interface to the control unit. The cuname consists of one to eight alphameric characters. The device group interface address modifier consists of one or two hexadecimal integers.

Specified as: (alphameric name, hexadecimal digits)

Programming Note: The device group interface address modifier is usually used to specify string switched direct access storage configurations. For most device groups, this parameter is specified as zero.

ADDRESS

specifies the relative hardware address and the symbolic device address. The first operand, the hardware address, is the address of the device relative to the base control unit address and consists of one or two hexadecimal digits. The second operand, the symbolic device address (SDA), consists of one to three hexadecimal digits; it is used as the external identification of the device.

Specified as: (hexadecimal digits, hexadecimal digits)

Programming Note: The complete physical address of a device is formed by adding the base control unit address (specified in the DCU macro instruction) to the device group interface address modifier (specified in the PATH operand of the DEVGRP macro instruction) to the hardware address (specified in the ADDRESS operand of the DEVGRP macro instruction).

MODEL
specifies the device model number; must be provided if the device has a model number.

Specified as: any digit, or N1

System default: null

FEATURE
specifies applicable optional features that are on the devices and/or associated control units.

Specified as:

Code	Feature
TROFL	Record-overflow feature on 2311 or 2314
SCAN	File-scan feature on 2841
DATACONV	Data-conversion feature on tape control
BTS	Burster-Trimmer-Stacker Feature on the 3800 Printer
WCGS	Writable Character Generation Storage Positions Feature on the 3800 Printer
7-TRACK	7-track head on any tape drive
9-TRACK	9-track head on any tape drive
AUTOCALL	Automatic dialer
DIAL	Dial-up line
DEDICATED	Dedicated (hard-wired) line
SADZERO SADONE SADTWO SADTHREE	Communications lines on 2702 Transmission Control Unit initialized by corresponding set adapter (SAD) value
UCS	Universal character set printer feature on the 2821 control unit for the 1403 printer
BSYN	Bisynchronous adapter
SINGDENS	Single density 3420
DUALDENS	Dual density 3420
COLBNRY	Column binary feature on 2501 or 2540 Reader or 2540 Punch.

System default: For 2702 or 2703, if neither DIAL nor DEDICATED is specified (only one may be specified), DIAL will be assumed. For 3420 SINGDENS will be assumed.

Note: The SAD (set adapter) value associates the 2702 Transmission Control Unit and line oscillator (terminal adapter n, where n is 0, 1, 2, 3) with the addressed communication line. The SAD value is ignored on

the 2703 and 3705 Transmission Control Units and is invalid on the 2701 Data Adapter Unit.

The features of nonstandard units (UNIT=NONSTD) must be specified by setting up an eight-digit hexadecimal field that corresponds to the device-code field (SDADEV) in the TSS symbolic device allocation table. This table describes the characteristics and status of each separately allocatable input-output device in the system (see System Control Blocks).

The form used to specify features of nonstandard units is

FEATURE=(X'hhhhhhhh')

This procedure also may be used for standard units.

Note: The subfields are not in the same order as are those in the IBM Operating System. In addition, some codes differ from those used in the Operating System. The user should check all code values against the meanings defined for CHASDA in System Control Blocks; otherwise unpredictable results may occur.

MAXIO
specifies, for any one task, the maximum number of input/output requests that may exist at one time for a device in this group. For a device that is to be serviced by MSAM (multiple sequential access method), the installation specification for MAXIO determines the number of page buffers used and should be at least three for printers supporting the FORMAT option.

Specified as: decimal number

System default: 2

TYPE
specifies the type of device (if required).

Specified as:

Code	Meaning
RESERVED	Device reserved for system use, such as tape drive needed by system error routines, SIPL, etc.
ASYNCH	An asynchronous interruption from this device causes a task to be created, if none exists. Caution: If ASYNCH is specified, it is the user's responsibility to insure that the UNIT operand specifies a legitimate terminal or RJE device of the system.

System default: null

IOREQ
specifies if IOREQ is to be permitted to access the devices described by this macro instruction; YES specifies that IOREQ is permitted; NO, that it is not.

Specified as: YES or NO

System default: NO

CONTRL
specifies the transmission control unit to which all terminals in the group are attached.

Specified as: 2701,2702,2703

System default: 2701 if UNIT=2780, 2702 otherwise.

Note: with the IBM 2780, if CONTRL=2701 is not specified the system default is CONTRL=2701, and the assembly completes with a warning message.

HOLD
specifies whether or not devices in this device group will be marked held at startup time.

Specified as: Y or N

System default: N

DDSDA

specifies whether or not "U" class users may specify devices in this device group with the SDA in the UNIT operand of the DDEF command (described in Command System User's Guide).

Specified as: Y or N

System Default: N

ABUF
specifies the number of 64-byte blocks that the system may allocate as buffers for a terminal device.

Specified as: decimal number 1 - 255

DCM
specifies to the system the specific device control module that is to be called on an initial asynchronous interrupt from a terminal device.

Specified as: decimal number 1 - 255

Programming Note: The recommended values for ABUF, DCM and HOLD are:

device type	ABUF	DCM	HOLD
2741	32 or 64	2	Y
TTY33, TTY35	32 or 64	3	Y
1052-7	64	4	N
3213	64	4	N
3215	64	4	N
3270	64	7	N
3270C	64	7	N
3066	64	8	N

OPCNSL -- Define Operator Consoles

The OPCNSL macro instruction is used to define to the system the default system operator consoles and the message codes to be routed to them.

Name	Operation	Operand
[symbol]	OPCNSL	(SDA[,ALARM,RSS,RMS,SYSMSG,HRDCPY,SYSERR]),...

SDA
specifies the symbolic device address of the operator console.

Specified as: a hexadecimal number denoting the console to receive the given message types. The following message types are supported:

ALARM - this SDA contains an alarm which is to be rung when a SYS-ERROR occurs and the system is stopping.

HRDCPY - denotes a hard copy console on which all system messages are to be printed.

RSS - denotes the console to be used as the RSS Master System Programmer console. RSS when activated will use the specified console as SYSIN/SYSOUT.

RMS - denotes the console to receive any messages issued by the Recovery Management System (RMS).

SYSMSG - denotes the consoles to receive any supervisor messages issued through the SYSERROR code 7 facility.

SYSERR - denotes the consoles to receive any SYSERR displays. The SYSERR display will be written to the denoted console but the console marked as 'RSS' is the one used by RSS for command input and output.

Programming Note: The device type of the specified SDAs must be one of the following: 1052-7, 3213, 3215, 3270, 3270C, 3066.

The message types may be specified for more than one SDA. The system will broadcast the following message types to all the consoles marked to receive the same: RMS, SYSMSG and SYSERR. A copy of the message will also be sent to all consoles marked as HRDCPY types.

For the ALARM and RSS console indicators, if more than one console is so marked, the system will use the first console found online and ignore the rest.

Any console marked with the ALARM attribute must support the 'ALARM' CCW operation code (X'0B'); otherwise, the alarm will not be sounded.

The terminal used to IPL the system is marked in the console table as the IPL console. This is the default console for all messages and for the RSS SYSIN/SYSOUT if one was not given. If the console was not already in the table, STARTUP adds the console to the table using one of the extra entries built by SYSGEN.

SYSGEN builds in 10 extra entries to be used for dynamically adding consoles to the table.

STEM -- Describe Processor Storage Elements

The STEM macro instruction is used to specify the number of pages (4096 byte blocks) of real storage in the system.

Name	Operation	Operand
[symbol]	STEM	number-integer,sup-integer

number
specifies the number of pages of real storage in the system

Specified as: decimal number from 256 to 4096

System default: 256

sup
specifies the number of pages of real storage to be assigned for use by the resident supervisor. These pages begin at real storage location zero and continue contiguously for the number of pages specified. Pages of real storage within the "sup" range cannot be varied by the VARY command.

Specified as: a decimal number from 128 to 4096

Default: 256

Programming note: On very large system configurations, the "sup" value should be no less than 256 pages. A smaller number of pages may cause system performance degradation, or the system may abort with insufficient storage for supervisor requests. The "sup" pages include the resident supervisor object code, tables, and transient real storage required by the resident supervisor.

TASK MANAGEMENT AND COMMAND SYSTEM MACRO INSTRUCTIONS

The task-management, system generation macro instructions that specify various parameters for dispatching and controlling tasks within TSS are DISPAR, TSKLMT, and VMPAR.

The command-system system generation macro instruction CLOP provides default parameters that are to be supplied by the system for various commands (and their macro instruction forms).

Because these macro instructions determine, to some degree, the efficiency of system operations, their parameters must reflect the work being processed at an installation. The choice of parametric values depends upon an installation's configuration, workload, and system objective. To adjust parameters, another system generation run is required.

CLOP -- Specify Command System Options

The CLOP macro instruction specifies certain default parameters for command system and macro instruction processing.

Name	Operation	Operand
[symbol]	CLOP	PRMTLMT=integer
		,PRVLG=(code,...)
		,AUTH={U|P}[,DSORG=code]
		,DATYPE={2311|2314|3330|333B|3350}
		,TATYPE={7|7DC|9Dn}
		,DEN={0|1|2|3|4}[,LABTYP={NL|SUL|SL}]
		,DAPAGES=(integer[1],integer[2])
		,DACYLS=(integer[1],integer[2])
		,DATRKS=integer
		,LIBPGS=(integer[1],integer[2])
		[,CFM=alphnum][,PFM=alphnum]
		[,MAV=integer]
		[,PRIO=integer]

PRMTLMT
specifies the installation's prompting limit; that is, the maximum number of times the system will repeat a message when signaling the operator and when prompting the user. The highest value that can be assigned is 255.

Specified as: decimal number

PRVLG
specifies the privilege classes that may be assigned at join-time. Each privilege class is designated by an alphabetic character other than A (reserved for system operator) and F (reserved for system manager). The first privilege class specified becomes the default class. Privilege classes are described in Manager's and Administrator's Guide.

Specified as: alphabetic character (other than A and F)

AUTH
specifies the default authority code assigned to a user when he is joined to the system; P indicates system programmer authorization, U indicates no system programmer authorization.

Specified as: P or U

DSORG
specifies the default for the positional dsorg (data set organization) parameter of the DDEF command and macro instruction.

Specified as:

Code	Organization
PS	physical sequential
VI	virtual index sequential
VS	virtual sequential
VP	virtual partitioned

System default: VS

DATYPE
specifies the type of direct access storage device to be defaulted in commands and macro instructions.

Specified as: 2311, 2314, 3330, 333B, or 3350

TATYPE
specifies the default value for the tape drive type if it is omitted in a DDEF command or macro instruction.

Specified as:

Value	Meaning
7	7-track tape
7DC	7-track tape with data conversion
9D2	9-track tape, density 800 bpi
9D3	9-track tape, density 1600 bpi
9D4	9-track tape, density 6250 bpi

DEN
specifies the default value for the DEN (tape recording density) parameter of the DDEF command or macro instruction.

Specified as:

DEN value	Tape recording density (bits/inch)	
	7-track	9-track
0	200	---
1	556	---
2	800	800
3	---	1600
4	---	6250

Note: No checking is performed to ensure that the DEN value is compatible with the value of TATYPE. DEN values of 0 and 1 should not be entered for 9-track tapes. DEN values of 3 and 4 should not be entered for 7-track tapes.

LABTYP
specifies, for magnetic tape data sets, the default for the parameter of the DDEF command or macro instruction.

Specified as:

Code	Meaning
NL	No labels
SL	Standard labels
SUL	Standard and user labels

System default: SL

Note: The next four operands specify the primary and secondary data set storage allocation that is to be assumed if this information is defaulted in the SPACE parameter of the DDEF command or macro instruction. Primary storage is the initial space allocated to a data set; secondary storage is the space to be allocated each time the space previously allocated to the data set has been exhausted and more data is to be written. At most, a three-digit number is permitted for each suboperand. The values assigned for primary and secondary storage are determined independently.

DAPAGES
specifies the number of pages used for the default primary- and secondary-storage allocation of VSAM, VISAM, and VPAM data sets.

Specified as: (decimal number, decimal number)

DACYLS
specifies the number of cylinders used for the default primary- and secondary-storage allocation of BSAM or QSAM data sets.

Specified as: (decimal number, decimal number)

DATRKS
specifies the number of tracks used for the installation default secondary-space allocation of BSAM or QSAM data sets.

Specified as: (decimal number)

LIBPGS
specifies the number of pages to be allocated for primary and secondary storage of a USERLIB.

Specified as: (decimal number, decimal number)

CFM
designates the installation default identification for punched card forms. It provides the name by which the system will identify the installation's standard card form.

Specified as: a string of not more than six characters; if a longer string is specified, it will be truncated to six characters from the right.

System default: CARDS

PFM
designates the installation default identification for printer forms. It provides the name by which the system identifies the standard combination of paper form, print chain, and carriage control tape used at the installation (see the description of SYSUCS and SYSURS under the discussion of the SETUR macro instruction in System Programmer's Guide). In the system received from IBM, the default symbol provides the values shown in Appendix G of System Programmer's Guide.

Specified as: a string of not more than six characters; if a longer string is specified, it will be truncated from the right.

System default: PAPER

MAV
specifies the default number of auxiliary storage pages assigned to a task during its execution.

Specified as: decimal number

System default: 256

PRIO

specifies the default priority that will be assigned a user being joined to the system without a specified priority.

Specified as: a decimal number, 0 to 9.

System default: 8

DISPAR -- Specify the Dispatching Algorithm

The DISPAR macro instruction specifies paging and timing parameters for the dispatching algorithm, which is the TSS mechanism for scheduling available processing time among the tasks within the system.

Name	Operation	Operand
[symbol]	DISPAR	LCT=(integer$_1$,integer$_2$)
		[,LDMTR=integer][,PURGSH=integer]
		[,THRESH=integer][,BUFSIZ=integer]
		[,AUXSP=(integer$_1$,integer$_2$)]
		[,INITLVL=integer][,TCR=integer]
		[,BIKSZE=(integer$_1$,integer$_2$)]

LCT

specifies the number of pages that must be available (or about to be written) in storage before a time-slice for another task can be started. A lower and an upper threshold must be specified for the available pages.

Specified as: (decimal number, decimal number)

LDMTR

specifies the percentage of the paging drums that remains to be allocated to auxiliary storage before overflow to paging disks occurs.

Specified as: integer

System default: 2

PURGSH

specifies the maximum number of shared pages to purge

Specified as: decimal number

System default: 22

THRESH

specifies the scan shared pages threshold

Specified as: decimal number

System default: 10

BUFSIZ
specifies the buffer size on the drum

Specified as: decimal number

System default: 64

AUXSP
specifies the auxiliary stop and primary threshold values

Specified as: (decimal number, decimal number)

Note: The second number must be greater than the first.

System default: BUFSIZ+10,100

INITLVL
specifies the initial Schedule Table level to be used for task creation.

Specified as: integer, 0 to 255

System default: 20

TCR
specifies the task core requirement, the value by which Startup divides the number of available main storage blocks to calculate the maximum number of tasks that can exist concurrently in the dispatchable list.

Specified as: decimal number

System default: 55

BLKSZE
specifies for each task the maximum number of pages to be written in a blocked format to drum and the combined maximum number of pages to be written in a blocked format to drum and disk.

Specified as: (decimal number, decimal number)

Note: The second number must not be less than the first.

System default: (12, 20)

PUBVOL -- Generate Public Volume Table

The PUBVOL macro instruction specifies the parameter needed to generate the public volume table.

Name	Operation	Operand
[symbol]	PUBVOL	MAXVOL=integer

MAXVOL
specifies the maximum number of public volumes that are in the configuration.

Specified as: decimal number

Note: One and only one PUBVOL macro instruction is required for system generation. The number of public volumes specified should be sufficiently large to allow for future expansion.

TSKLMT -- Specify Task Limitations

The TSKLMT macro instruction specifies the system limits on the size and number of tasks, by type, that will be allowed to operate concurrently.

Name	Operation	Operand
[symbol]	TSKLMT	CONV=integer,MTTADM=integer,BATCH=integer ,BACK=integer,PGTEL=integer [,PSLMT=integer] [,OVRAUX=integer][,RTAMBUF=integer]

CONV
specifies the maximum number of conversational tasks that the operator may allow with the FLOW command. This value also represents the initial conversational task limit; an attempt to create a conversational task beyond this limit produces a message at the user terminal, and the keyboard is set to receive the next input from the terminal.

Specified as: decimal number, greater than or equal to 1.

MTTADM
specifies the maximum number of MTT administrator tasks that the operator may allow with the FLOW command. This value also represents the initial MTT administrator task limit; an attempt to create an MTT task beyond this limit produces a message at the user terminal, followed by an underscore.

Specified as: decimal number

BATCH
specifies the maximum number of tasks, initiated by a command procedure read in by BULKIO or by an EXECUTE command, that the operator may allow with the FLOW command. The system will initiate new tasks only if the number of batch tasks currently executing is within this linit.

Specified as: decimal number

Note: This limit must be set less than the task limit allowed any individual user (see System Programmer's Guide, Appendix H).

BACK
specifies the maximum number of background tasks (conversational jobs made nonconversational by issuance of BACK commands) that the operator may allow with the FLOW command. This value also represents the initial background task limit and an attempt to create a task beyond this limit will produce a diagnostic message at the user terminal, followed by an underscore.

Specified as: decimal number

PGTBL
specifies the maximum number of main-storage pages that may be occupied by the page tables (and associated information) for one task. This limitation constricts the actual size of virtual storage that may be used by a task. If this limit is exceeded, the associated task is terminated abnormally.

Specified as: decimal number

PSLMT
specifies the maximum number of public segments to be allocated to any operational task.

Specified as: decimal number

System default: 100

OVRAUX
specifies the percentage of auxiliary storage space that an installation can overcommit.

Specified as: decimal number, 0 to 99

System default: 0

RTAMBUF
specifies the number of work pages to be allocated in virtual storage for the terminal access method at task logon time.

Specified as: decimal number 1 to 16

System default: 8

VMPAR -- Specify Virtual Storage Parameters

The VMPAR macro instruction specifies parameters used in the allocation and manipulation of virtual storage.

Name	Operation	Operand
[symbol]	VMPAR	[VCSLNG={integer|SEGMENT}],SDST=integer
		[,OPTIONS=(code,...)]
		[,TIME=integer]

VCSLNG
specifies the increment (in pages) to be added to a request for a variable amount of virtual storage. SEGMENT specifies a 255-page increment.

Specified as: integer (in range 1-255), or SEGMENT

System default: 20 (also assumed if an invalid integer is specified).

SDST
specifies the number of pages to be assigned to the shared data set table.

Specified as: integer (maximum 256)

OPTIONS
specifies the virtual storage request or supervisor options.

Specified as:

PUBSEG — shared VM allocation requests are to be packed into any available shared segment group. These segment groups are known as public segments. If PUBSEG is not specified, each shared request will be allocated a new shared segment group.

Note: Packing refers to assignment of successive pages of a segment, not to condensing information within pages.

PACKSEG — private VM allocation requests are allocated on page boundaries. If PACKSEG is not specified, each private request will be allocated on a 16 page boundary.

WCDISK — Page-writes on disk used as an auxiliary paging device and writes of VSAM, VISAM, and VPAM data sets (which must be on direct access devices) are to be validity-checked. For BSAM data sets, the corresponding option is specified individually by the user in either the DDEF command or macro instruction or the DCB macro instruction.

TIME
specifies the initial task time interval to be assigned for each task. Upon elapse of the specified interval, a task is abnormally terminated, in the background mode; or the user is notified, in the conversational mode. The TIME command can be used to change the task timer settings for each task.

Specified as: eight unsigned, unpacked decimal digits HHMMSSth, where the maximum value for each field is:

HH (hours) ≤07
MM (minutes) ≤59
SS (seconds) ≤59
t (1/10 sec) ≤9
h (1/100 sec) ≤9

The composite range is from 1 second to 7-1/2 hours.

System default: one hour

SYSTEM TABLE GENERATION MACRO INSTRUCTION

GENSCB -- Generate System Control Blocks

The GENSCB macro instruction causes the generation of the object module (SYSGEN) that contains the system control blocks. These blocks reflect the parameters that have been specified in other system generation macro instructions.

Name	Operation	Operand
[symbol]	GENSCB	

Programming Note: GENSCB must be written as the last macro instruction.

SECTION 1: PURPOSE

System Maintenance is the procedure by which the system programmer (userid TSS) applies IBM-supplied maintenance packages to an operational system. The TSS command system modifies system data sets according to the script and data sets on the change tape, as described below. (See Figure 5 for a diagram of the system maintenance procedure.)

For each maintenance release, IBM distributes a VAM change tape to be used during an operational session on the latest released TSS system and, if applicable, a tape that contains a copy of the latest independent utilities. The change tape contains a copy of the modified source modules as file one, an update script (of command language statements) that controls the update as file two, and several files that, when copied to direct access devices by the update script, become VPAM, VSAM and VISAM data sets that contain the replacement modules and changes to the system data sets to complete the maintenance release update in place.

The change tape may be used in its distributed form to update a system that is to be modified by IBM-supplied maintenance packages exclusively. Or, before applying the change volume to a system containing user modifications, the specific requirements of the installation may be accommodated by using the existing system to modify the change data set after it is read from the tape to external storage. The replacement modules, on external storage, are in delta data set form for SYSIVM, RESSUP, and RSSSUP. Consequently, if the update script is suitably modified to direct these data sets to a delta volume, the dynamic modification procedure can be used to ascertain the effects of these changes before the system update procedure permanently modifies the system (see PART V). The system effects of SYSLIB changes can be ascertained (prior to system update) by the JOBLIB facility of the dynamic loader (except SYSPRO and SYSMLF changes). (However, if STARTUP, PRELUDE, SYSIAM, or SYSUTL are included, they must be moved to the IPL volume to be tested.)

The modified source modules (along with the original source modules) are always distributed as one blocked tape file (SYSIMO format). Multiple VISAM source data sets are created from this file by the use of the SYSIMO program. A complete description of this program may be found in Appendix C.

SECTION 2: THE CHANGE TAPE

The data sets that may reside on the change tape are listed below. Of these, only those needed for the modifications in a specific maintenance release will be on the change tape for that release. The modified source modules and TAPE.SCRIPT are the first two files on all maintenance release tapes. The remaining files include only changes for those data sets impacted during any specific maintenance release.

The change tape consists only of data sets owned by userid TSS*****; when all data sets are impacted, the tape contains:

File	Tape File Name	Public/Private Name	Purpose
1.	None	TSS source modules	A copy of the source modules (in SYSIMO format) modified for this release.
2.	TAPE.UPDATE	SCRIPT.UPDATE	An executable script of necessary commands to cause the update to be applied.
3.	TAPE.ASMMAC	ASMMAC(+1)	A replacement for ASMMAC macro instructions.
4.	TAPE.ASMNDX	ASMNDX(+1)	A replacement for the ASMMAC macro index.
5.	TAPE.SYSMAC	SYSMAC(+1)	A replacement for the SYSMAC macro instructions.
6.	TAPE.MACNDX	MACNDX(+1)	A replacement for the SYSMAC macro index.
7.	TAPE.GENMAC	GENMAC(+1)	A replacement for GENMAC macro instructions.
8.	TAPE.GENNDX	GENNDX(+1)	A replacement for the GENMAC macro index.
9.	TAPE.UTLMAC	UTLMAC(+1)	A replacement for UTLMAC macro instructions.
10.	TAPE.UTLNDX	UTLNDX(+1)	A replacement for UTLMAC macro index.
11.	TAPE.SYSIAM	UPDATE.SYSIAM	A VPAM data set containing the replacement module for the independent access method.
12.	TAPE.STARTUP	UPDATE.STARTUP	A VPAM data set containing the replacement module for STARTUP
13.	TAPE.SYSUTL	UPDATE.SYSUTL	A VPAM data set containing the replacement modules for the independent utilities (including UTLMAC when applicable).
14.	TAPE.PRELUDE	UPDATE.PRELUDE	A VPAM data set containing the replacement module for PRELUDE. The control section for PRELUDE may be moved to the IPL volume by RSS.
15.	TAPE.SYSIVM	IVM99.CHANGES	A VPAM data set containing the replacement modules for SYSIVM
16.	TAPE.RESSUP	SUP99.CHANGES	A VPAM data set containing the replacement modules for RESSUP
17.	TAPE.RSSSUP	RSS99.CHANGES	A VPAM data set containing the replacement modules for RSSSUP
18.	TAPE.SYSLIB	UPDATE.SYSLIB	A VPAM data set containing the replacement modules for SYSLIB

19.	TAPE.SYSERP	UPDATE.SYSERP	A VPAM data set containing the replacement modules for SYSERP (including ERPMAC when applicable).
20.	TAPE.UPDATE.ASMMAC	UPDATE.ASMMAC	A copy of the replacement regions for ASMMAC macro instructions.
21.	TAPE.UPDATE.SYSMAC	UPDATE.SYSMAC	A copy of the replacement regions for SYSMAC macro instructions.
22.	TAPE.UPDATE.GENMAC	UPDATE.GENMAC	A copy of the replacement regions for GENMAC macro instructions.
23.	TAPE.UPDATE.UTLMAC	UPDATE.UTLMAC	A copy of the replacement regions for UTLMAC macro instructions.
24.	TAPE.VSSLIB	UPDATE.VSSLIB	A VPAM data set containing the un-linkedited VSS replacement modules.
25.	TAPE.FORLIB	UPDATE.FORLIB	A VPAM data set containing the un-linkedited FORTRAN replacement modules.
26.	TAPE.APGEN	APGEN	A replacement for the APGEN data set.
27.	TAPE.VSSLINK	SCRIPT.VSSLINK	An executable script of commands for the linkage edit of VSS.
28.	TAPE.FORLINK	SCRIPT.FORTRAN	An executable script of commands for the linkage edit of FORTRAN.
29.	TAPE.PPLI.xxxxxxxx	SCRIPT.PPLI.xxxxxxxx	An executable script of commands for the installation of program products for the program product language interface.
30.	TAPE.UPDATES	UPDATE.MODULE	A region data set containing modifications that were made to the source modules for this release.

When there are data sets that have no change and thus are not included on the maintenance release tape, each file will move ahead to keep the file numbers sequential. (For example, if TAPE.SYSMAC, TAPE.MACNDX TAPE.GENMAC, TAPE.UTLMAC, TAPE.UTLNDX, and TAPE.GENNDX were omitted, then TAPE.SYSIAM would be file number 5.)

The change tape script assumes that the change tape is 1600 bpi and that the IPL volume is TSSRES. If your tape is not 1600 bpi or if your IPL volume has a different volume ID, the script must be modified before it is executed.

SECTION 3: UPDATE STRUCTURE

The maintenance release script (file 2) is divided into four logical parts: part 1, which performs a tape-to-VAM copy of all modified macro libraries, module libraries, and executable scripts; part 2, which reviews all VAM data sets created by the tape-to-VAMs; part 3, which catalogs macro libraries as generation+1 and copies the changed modules to the appropriate system data sets; and part 4, which modifies the SYSPRO, SYSMLF, SYSPRD, and SYSPRX members of SYSLIB when applicable.

Parts 1 and 2 perform the setup of the changes and do not impact a running system, except for the external storage required and any new macro instructions introduced in part 1. Parts 3 and 4 perform the actual update to bring the system up to the release level. This logical structure gives you the flexibility of running the update as one script (see Appendix H, Figure 9), or of using the Text Editor to divide the script into two or more sections (see Appendix H, Figure 10) or running a SYSGEN assembly, and introducing user macro instructions, or inspecting changes on a section to section basis.

SECTION 4: MAINTENANCE PROCEDURES

The procedure to be followed by the installation when updating the system is described below (see also Figure 5). Appendix H, "Sample Maintenance Procedures," contains typical command sequences to be entered from a terminal, as well as sample update scripts. The detailed procedure and descriptive information associated with a specific maintenance release are provided by the SCRIPT.UPDATE data set that is read in from the change tape. The procedure is as follows:

1. IPL the new DASDI utility from the independent utility tape and create an IPL volume with a volume identification of QSSIPL. Note: This is required only if CEIAP (Prelude) has changed in this maintenance release.

2. Start up your latest TSS system. Recommendation: Prior to execution of a maintenance update, any direct access storage containing system data sets that are affected by the update should be saved on tape by the DASDDR utility. It is advisable for the user to have a reasonably recent back-up tape for his currently running system.

3. The system programmer (userid TSS), as the owner of all data sets on the change tape as well as of the system data sets, must bring in and execute the update script. (See Appendix H, Figure 9.)

4. When the update process is complete, the operator should shut down the system and then save all changed volumes using the DASDDR utility.

5. If prelude has been altered in this release, it will be necessary to do a disk-to-disk save of your new IPL volume to the QSSIPL volume created in step 1 of this procedure, using the DASDDR utility. Note: Steps 1 and 5 should be skipped if there has been no change to Prelude.

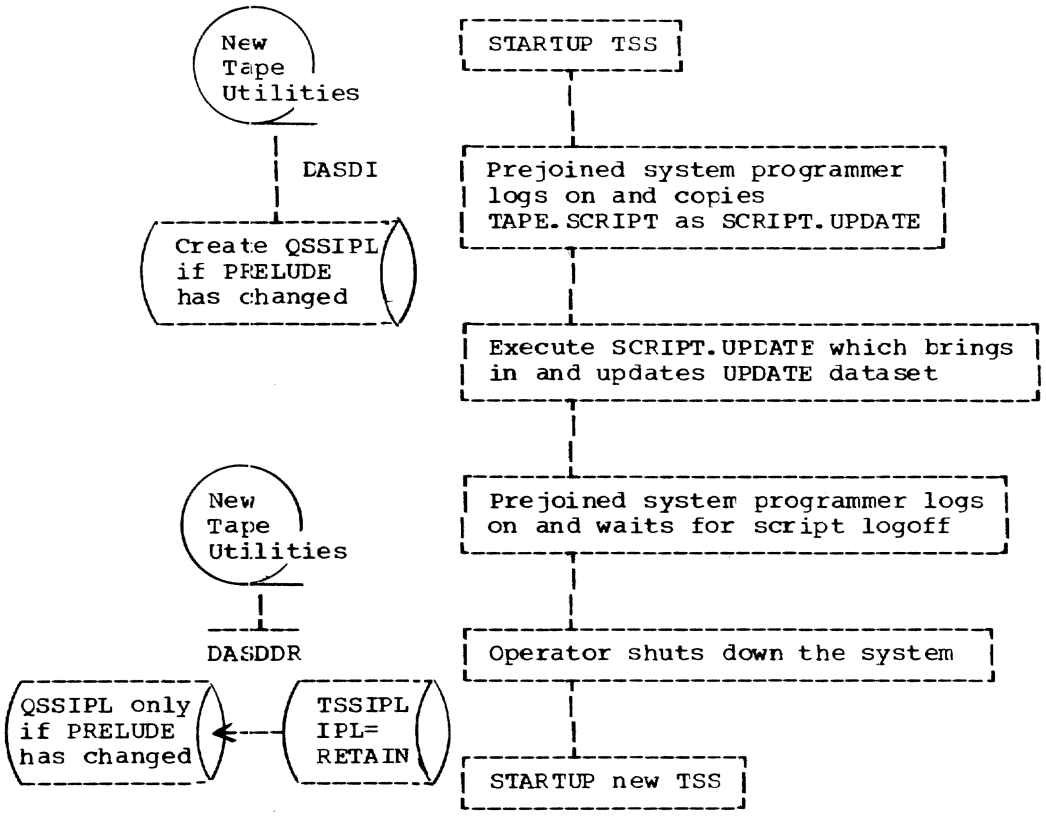


Figure 5. System Maintenance Release

PART V: DYNAMIC MODIFICATION OF THE SYSTEM

SECTION 1: OVERVIEW

Dynamic modification is the process of testing the effects of changes to the system without permanently updating the system. The changes may be either from IBM-supplied maintenance packages or user-initiated modifications. The changes, which remain in effect from startup until shutdown, are contained in delta data sets and must be on one private or public volume (delta data set volume), described in Section 2.

During the startup procedure, delta data sets are searched for initial virtual storage, resident supervisor, and resident support system control sections; then, system data sets on the IPL volume are searched. After the system's query "DELTA DATA SETS?", the order of search for these data sets is specified by the operator.

There are three load lists: one each for the resident supervisor, initial virtual storage, and resident support system. Only the control sections that are identified by name in a loadlist are loaded. Since these load lists are control sections and are located by STARTUP in the same manner as any other control section, an alternate load list, from that contained in the system data sets, can be included in a delta data set. Such a load list might be used to add control sections to initial virtual storage, the resident supervisor, or the resident support system (see the description of the LLIST macro instruction in System Programmer's Guide). Tasks executed during the current session will run with all such modifications to initial virtual storage, the resident supervisor, and the resident support system. To permanently modify the system, the same delta data sets that changed the system during the startup procedure may be incorporated by the system maintenance process (described in Part IV).

SECTION 2: DELTA DATA SETS

Delta data sets are used at startup time to dynamically modify the contents of initial virtual storage, the resident supervisor, and the resident support system. A delta data set:

1. Has VPAM organization.
2. Requires the use of a private or public direct access device.
3. Is on a volume that has a known serial number (one volume must contain all the delta data sets used in a single startup session).
4. Has a dsname with a second-order qualification of IVMnn, SUPnn, or RSSnn (nn = integer from 00 to 99), depending upon whether the data set contains SYSIVM, RESSUP, or RSSSUP modules. The first-order qualification of the data set name is the eight-character user identification that the system automatically prefixes to data set names. At join-time, fewer than eight characters may be specified for a user identification; asterisks are then appended to extend the identification in the data set name to eight characters. The complete data set name, including the user identification, is restricted to 44 characters.

To construct such a delta data set, use the DDEF command to define the data set as a JOBLIB and describe its characteristics to the system. Then using the TSS Assembler, create object modules for inclusion in the data set so that each object module becomes a member of the partitioned delta data set. All future assembled modules will be stored in this data set until another DDEF command with OPTION=JOBLIB is issued or until the user logs off.

Example: Build a data set named USERID1*.IVM01.ZZ on a private volume that has serial number 222222. This terminal session might occur:

```
SYSTEM:  (keyboard unlocks)

USER:    logon userid1,password

SYSTEM:  TASKID=00E4 LOGON AT 13:56 ON 09/15/76

USER:    ddef ivm,vp,dsname=ivm01.zz,unit=(da,3330),
         volume=(,222222),option=joblib,disp=new

         .
         .     assemble object modules for this data set
         .

         logoff
```

The changes to the SYSIVM, RESSUP, and RSSSUP data sets that are supplied in an IBM maintenance package may be made into delta data sets by properly naming and placing the object module libraries. The changes may then be tested prior to a permanent system update.

SECTION 3: DYNAMIC MODIFICATION PROCEDURE

STARTUP first asks the operator for the codes of those functions not to be loaded. Any function or module that can be omitted from the load has a code in the load list entry; codes 01 through BF are reserved for the system, codes C0 through FF are reserved for the user (see System Programmers Guide). If modules are to be excluded from the load at STARTUP, the operator may enter the codes at the terminal or via the card reader. If no codes are entered (no modules are to be excluded), an EOB is entered at the terminal or a blank card or an END card is entered at the card reader.

STARTUP next asks the operator whether dynamic modification of SYSIVM, RESSUP, or RSSSUP is wanted: DELTA DATA SETS? Y OR N. If the system is not to be modified, the operator types N; STARTUP proceeds to load SYSIVM, RESSUP, and RSSSUP from the system data sets on the IPL volume (see Figure 6). When he types Y, he receives a message requesting the device address and data set specification for the delta data set volume. He answers this request in this format:

```
xxxx, dsname,...
```

where

1. xxxx is the three- or four-character hardware address of the device on which the delta data set volume has been mounted prior to running STARTUP;

2. dsname specifies the data set name (either fully or partially qualified) of a delta data set that is to be searched prior to searching the system data sets.

3. An EOB is entered at the terminal or an END card is entered at the card reader to terminate the list.

If a specific number of data sets is to be searched, the order in which the data set names are specified by the system operator is the order in which STARTUP will search. Each data set name must be specified with its complete user identification (including any asterisks). If all the data sets identified by a partially qualified data set name are to be searched in the order in which they are found on the volume, only the partially qualified data set name need be entered at the console by the system operator.

When STARTUP has all the delta data set information, it loads the initial virtual storage, the resident supervisor, and the resident support system as specified. If minor errors occur, a message is sent to the the operator and processing continues. See Appendix I for an example of a terminal session of dynamic modification.

The operator may cancel any invalid information he has entered and then reenter the correct information. To avoid reentering the series when one data set name in a series is incorrectly specified, the operator may continue the list with the correct data set name; the incorrect entry receives a diagnostic message and is bypassed. If a specific dataset name cannot be found on the delta volume, the data set is ignored and the operator is notified.

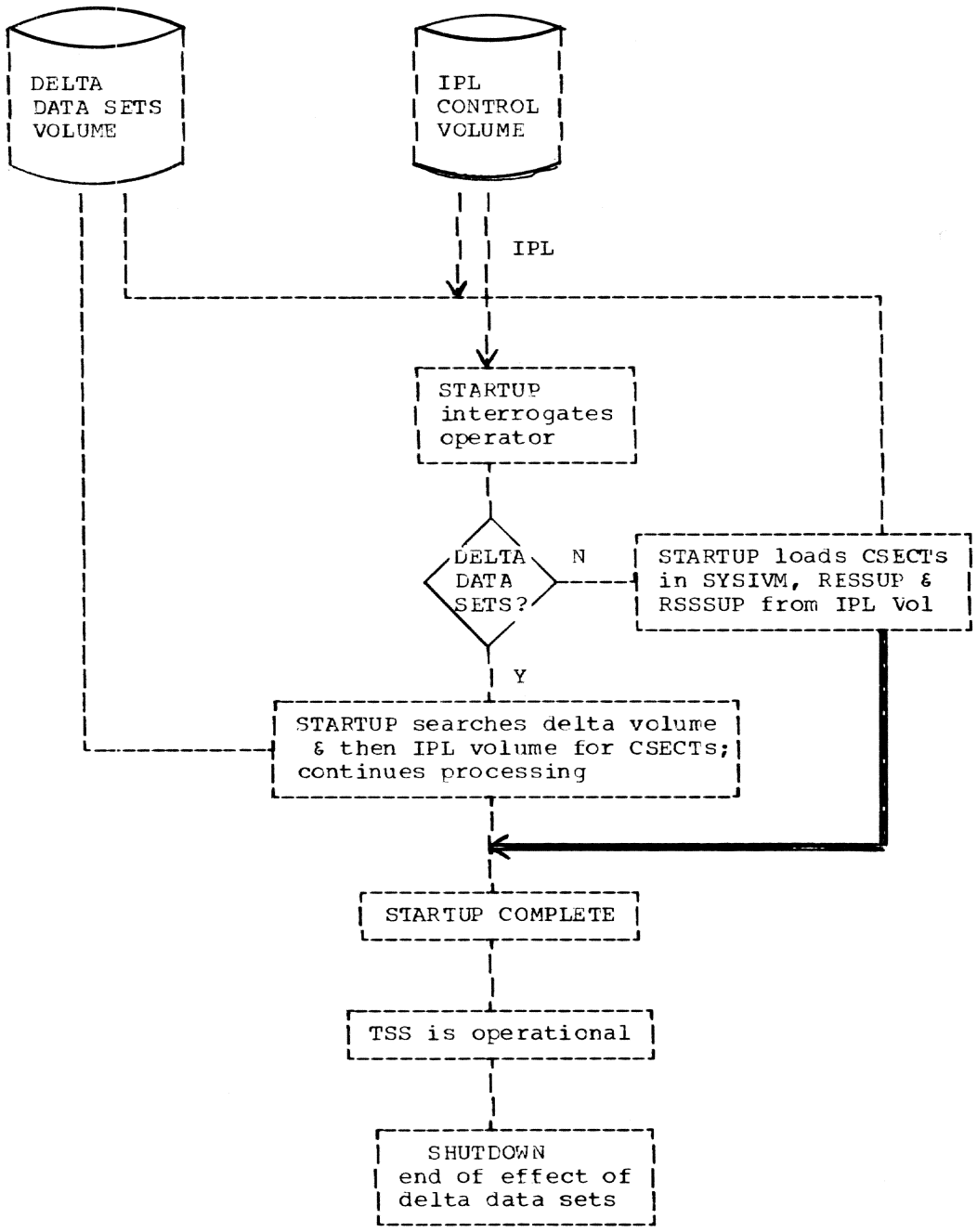


Figure 6. Dynamic Modification Procedure

APPENDIX A: MACRO INSTRUCTION FORMATS

1. Symbolic Mnemonics used in Macro Instructions

symbol - string of one-to-eight alphameric characters; the first must be alphabetic. (In TSS the alphabetic characters are A/Z, $, #, and @; the alphameric are the alphabetic plus 0-9.)

operand name - single word, usually mnemonic; tells what the operand represents.

coded value - string of characters written exactly as shown (numbers or uppercase letters).

value mnemonic - single mnemonic that summarizes a variety of values that can be specified for the operands. From the set of values identified, the user chooses one and writes that value. The value mnemonics used are:

alphnum - string of alphameric characters; the first need not be alphabetic. The limit on the number of characters is given under the macro instruction.

code - set of specific values, one of which is written; shown explicitly in format illustration if there are three or fewer; or "code" appears in format and the actual values in the text.

hexinteger - hexadecimal number; the limit of the number of digits is given under the macro instruction.

integer - decimal number; the limit of the number of digits is given under the macro instruction.

2. Configuration Macro Instructions

Name	Operation	Operand
[symbol]	CHANNEL	CPU={0|1},MPX=(hexinteger,...) [,SEL=(hexinteger,...)] [,BMX=(hexinteger),...]

Name	Operation	Operand
[symbol]	CPU	nocpu-{1|2}

Name	Operation	Operand
cuname	DCU	UNIT=code,PATH=(hexinteger-hexinteger,...) [,MODEL={1|2|3|4|5|6|7|8}] [,SHARED={Y|N}]

Name	Operation	Operand
gpname	DEVGRP	UNIT=code,PATH=((cuname,hexinteger),...) ,ADDRESS=((hexinteger,hexinteger),...) [,MODEL=code] [,FEATURE=(code,...)] [,MAXIO=integer] [,TYPE={RESERVED|ASYNCH}] [,IOREQ={YES|NO}] [,CONTRL={2701|2702|2703}] [,HOLD={Y|N}] [,DDSDA={Y|N}] [,ABUF=integer] [,DCM=integer]

Name	Operation	Operand
[symbol]	OPCNSL	(SDA[,ALARM,RSS,RMS,SYSMSG,HRDCPY,SYSERR]),...

Name	Operation	Operand
[symbol]	STEM	number-integer,sup-integer

3. Task Management and Command System Macro Instructions

Name	Operation	Operand
[symbol]	CLOP	PRMTLMT=integer ,PRVLG=(code,...),AUTH={U|P} [,DSORG=code],DATYPE={2311|2314|3330|333B|3350} ,TATYPE={7|7DC|9Dn},DEN={0|1|2|3|4} [,LABTYP={NL|SUL|SL}] ,DAPAGES=(integer[1],integer[2]) ,DACYLS=(integer[1],integer[2]) ,DATRKS=integer ,LIBPGS=(integer[1],integer[2]) [,CFM=alphnum][,PFM=alphnum] [,MAV=integer][,PRIO=integer]

Name	Operation	Operand
[symbol]	DISPAR	LCT=(integer[1],integer[2]) [,LDMTR=integer][,PURGSH=integer] [,THRESH=integer][,BUFSIZ=integer] [,AUXSP=(integer[1],integer[2])] [,INITLVL=integer][,TCR=integer] [,BLKSZE=(integer[1],integer[2])]

Name	Operation	Operand
[symbol]	PUBVOL	MAXVOL=integer

Name	Operation	Operand
[symbol]	TSKLMT	CONV=integer,MTTADM=integer,BATCH=integer ,BACK=integer,PGTBL=integer [,PSLMT=integer] [,OVRAUX=integer] [,RTAMBUF=integer]

Name	Operation	Operand
[symbol]	VMPAR	[VCSLNG={integer|SEGMENT}],SDST=integer [,OPTIONS=(code,...)][,TIME=integer]

4. System Table Generation Macro Instruction

Name	Operation	Operand
[symbol]	GENSCB	

APPENDIX B: STRUCTURE OF TSSRES

The system data sets required for starting up and running TSS are stored initially on one volume, TSSRES (TSS resident volume). All these datasets, except SYSLOG (the system operator log), SYSBWQ (the system operator's work queue), and USERLIBs for SYSMANGR and SYSOPERO, are cataloged under the userid TSS*****, with a password of IBM.

The major system data sets, shown in Figure 7, that reside on TSSRES are:

TSS*****.SYSCCB (system configuration control block) is a generation data group in which each VPAM data set contains a description of the system configuration. SYSCCB, created at system generation time, is used by STARTUP to acquire such information as: (1) paths to printers; (2) number of central processing units; and (3) number of pages in real storage.

TSS*****.STARTUP (system STARTUP module) is a generation data group in which each VPAM data set contains one member -- the STARTUP module (CEIAA). STARTUP initiates the Time Sharing System. STARTUP is called by performing an initial program load (IPL) of the TSSRES volume. The IPL process reads in PRELUDE which, after initial processing, reads in the independent access method and STARTUP. Control is then passed by the independent access method to STARTUP to continue the Time Sharing System.

TSS*****.SYSIAM.DSTSSRES is a VPAM data set which contains one member -- the independent access method module (CEIAM). IAM is the input/output interface that is used by STARTUP and the independent utilities. If a quick start data set is built at startup time, a copy of CEIAM is moved to the quick start volume and given the name TSS*****.SYSIAM.DSxxxxxx, where xxxxxx is the volume ID of the quick start volume.

TSS*****.RESSUP (system resident supervisor) is a generation data group in which each VPAM data set contains the modules of TSS routines that permanently reside in main storage. Control sections from these modules are link-loaded, by STARTUP, into available main storage. The presence of these control sections in this data set is a necessary but not sufficient condition for such link-loading; STARTUP will load only those control sections whose names are included in a special list found in RESSUP.

TSS*****.SYSIVM (system initial virtual storage) is a generation data group in which each VPAM data set contains the system modules that are automatically provided for each user at task-initiation time. The presence of these control sections in this data set is a necessary but not sufficient condition for such link-loading; STARTUP will load only those control sections whose names are included in a special list found in SYSIVM.

TSS*****.RSSSUP (resident support system) is a generation data group in which each VPAM data set contains the modules of Time Sharing Support System (TSSS) routines that reside in main storage during TSSS execution. Control sections from these modules are link-loaded, by STARTUP, into a virtual storage which will be mapped into main storage during TSSS execution. The presence of these control sections in this data set is a necessary but not sufficient condition for such link-loading; Startup will load only those control sections whose names are included in a special list found in RSSSUP.

TSS*****.SYSUTL (independent utility library) is a generation data group in which each VPAM data set contains the modules of the independent utilities. Control sections from these modules are read into main storage by the independent access method (IAM), which in turn was read in by PRELUDE. The presence of control sections in this data set is a necessary but not sufficient condition for loading by IAM; IAM will load only those control sections that are specifically required for the STARTUP process or the independent utilities.

TSS*****.SYSLIB (system library) is a generation data group in which each VPAM data set is a collection of modules that are automatically eligible for loading into each user's virtual storage by the dynamic loader. SYSLIB contains virtual storage system modules not found in SYSIVM. Also contained in SYSLIB are: SYSPRO (system procedure library), SYSPRX (system prototype profile), SYSPRD (system procedure dictionary), and SYSMLF (system message file).

TSS*****.SYSSVCT (user catalog index) is a VISAM data set that has a user identification code for each user joined to the system. This data set also contains pointers to each user catalog (USERCAT) in the system. New user identification codes, along with their corresponding USERCATs, are added to SYSSVCT by the JOIN command and removed by the QUIT command. SYSSVCT initially has three user identification codes: SYSOPER0, SYSMANGR, and TSS*****.

TSS*****.SYSCAT (system catalog) is a VPAM data set that contains as members copies of USERCATs for all users that have logged on during one TSS session. The system catalog is purged of USERCATs at shutdown time. SYSCAT is initially a null data set.

TSS*****.USERCAT (user catalog) is a VSAM data set that contains pointers for each cataloged data set owned by TSS*****. This data set also contains information on ownership and sharing privileges for shared data sets. USERCATs are created by the JOIN command and deleted by the QUIT command. Entries in a user's catalog are maintained by the CATALOG, DELETE, ERASE, PERMIT, SHARE, DDEF, RC, and RT commands.

TSS*****.SYSMAC (system macro library) is a generation data group in which each VISAM region data set contains definitions of all macro instructions necessary to support normal nonprivileged user assemblies.

TSS*****.MACNDX (system macro library index) is a generation data group in which each VSAM data set is an index to the system macro library.

TSS*****.ASMMAC (system assembler macro library) is a generation data group in which each VISAM region data set contains definitions of TSS macro instructions, used in: system generation, on-line test system (OLTS), assembling both FORTRAN and the supervisor (those not in SYSMAC), and the system DSECTs.

TSS*****.ASMNDX (system assembler macro library index) is a generation data group in which each VSAM data set is an index to the system assembler macro library.

TSS*****.GENMAC (system generation macro library) is a generation data group in which each VISAM region data set contains definitions of TSS macro instructions used specifically for system generation.

TSS*****.GENNDX (system generation macro library index) is a generation data group in which each VSAM data set is an index to the system generation macro library.

TSS*****.UTLMAC (utility macro library) is a generation data group in which each VISAM region data set contains definitions of TSS macro instructions used specifically for independent programs (CEI) and utility programs (CEB).

TSS*****.UTLNDX (utility macro library index) is a generation data group in which each VSAM data set is an index to the utiltiy macro library.

TSS*****.SYSERP (error recording edit and print library) is a generation data group in which each VPAM data set contains the modules necessary to run VMEREP; this data set also contains ERPMAC, the macro library needed to assemble the EREP modules.

TSS*****.SYSUSE (system user table) is a VISAM data set that has one entry for each currently authorized user of the system. Each user's entry contains his identity, password, charge number, job priority, command-set privilege, and authorization attributes. The entries are in user-identity sequence and are variable in length to a maximum of 256 bytes. SYSUSE initially contains entries for the system operator, the system manager, and the system programmer (userid TSS). The data set is maintained by the system administrator. Entries are added by the JOIN command and removed by the QUIT command.

TSS*****.USERLIB (user library) is a VPAM data set that is opened and placed on the JOBLIB chain when TSS logs on. This data set is initially empty.

TSS*****.SOURCE.SYSGEN (sample system generation source) is a VISAM data set that contains a sample multi-processor SYSGEN.

TSS*****.SYSGEN.MODULE (sample system generation module) is a VPAM data set that contains the object module created by assembling SOURCE.SYSGEN.

TSS*****.APGEN (apply the system generation module) is a VISAM data set containing the command system and linkage editor control statements that apply the system generation object module to TSS.

TSS*****.FORLIB (FORTRAN library) is a VPAM data set that contains the un-linkedited object modules for the TSS FORTRAN compiler and tne TSS FORTRAN run time library.

TSS*****.SCRIPT.FORTRAN (FORTRAN linkedit script) is a VISAM data set that contains the executable script to perform the linkage edit for the TSS FORTRAN compiler and run time library.

TSS*****.VSSLIB (virtual support system library) is a VPAM data set that contains the un-linkedited object modules for the virtual support system (VSS).

TSS*****.SCRIPT.VSSLINK (VSS linkedit script) is a VISAM data set that contains the executable script to perform the linkage edit for the virtual support system (VSS).

TSS*****.SCRIPT.PPLI.xxxxxxxx (program product installation scripts) is a set of VISAM data sets that contain the executable scripts necessary for the installation of certain program products. The xxxxxxxx is a name describing the particular program product to which the script applies.

SYSMANGR.USERCAT (user catalog) is a VSAM data set that contains pointers, sharing privileges, and ownership information for each data set cataloged for SYSMANGR.

SYSMANGR.USERLIB (user library) is a VPAM data set that is opened and placed on the JOBLIB chain when SYSMANGR logs on. This data set is initially empty.

SYSOPER0.USERCAT (user catalog) is a VSAM data set that contains pointers, sharing privileges, and ownership information for each data set cataloged for SYSOPER0.

SYSOPER0.USERLIB (user library) is a VPAM data set that is opened and placed on the JOBLIB chain when SYSOPER0 logs on. This data set is initially empty.

SYSOPER0.SYSLOG (system log) is a generation data group in which each VISAM data set contains a record of all communication between the system and the operator. A print is issued for the current SYSLOG data set when TSS is shut down. A new SYSLOG data set is created when TSS is started up.

SYSOPER0.SYSBWQ (batch work queue) is a VISAM data set that contains a queue of work for the batch monitor and/or the BULKIO task. Entries are placed on the batch work queue by the PRINT and EXECUTE commands. Entries are removed from the batch work queue by the batch monitor.

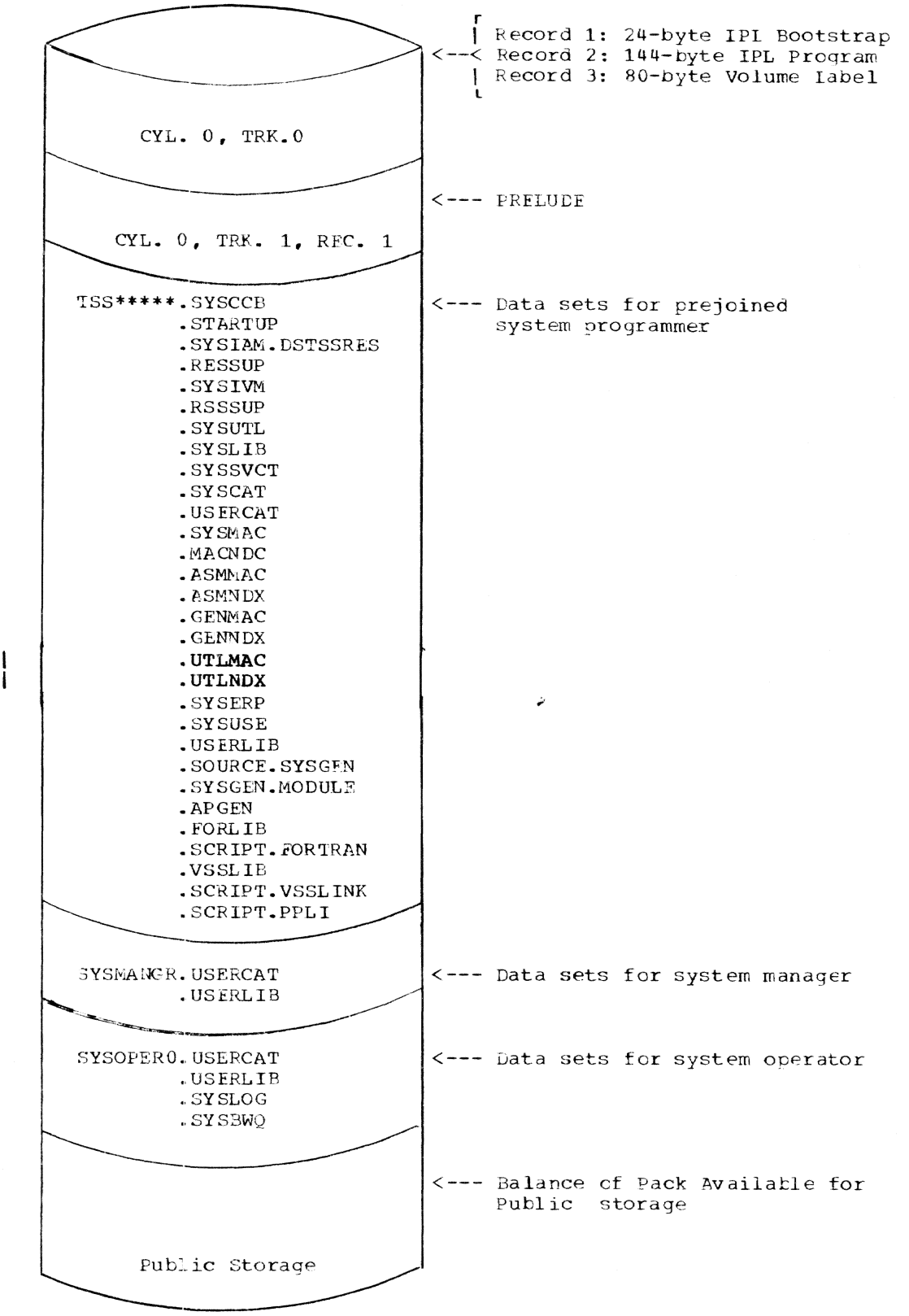


Figure 7. TSS Resident Volume (TSSRES)

APPENDIX C: THE SYSIMO PROGRAM

The purpose of the SYSIMO program is to read a specially formatted and blocked BSAM tape file and create multiple VISAM source data sets.

The program is invoked by the command:

```
CALL SYSIMO
```

The user is then prompted for information by three messages:

message:	SPECIFY INPUT TAPE -- VOLUME=( )/DDNAME=
response:	VOLUME=(,volserno) where volserno is the volume serial number of a nine track, non-labeled, default density tape.
response:	DDNAME=ddname where ddname is the data definition name previously assigned to the tape.
message:	SPECIFY DATA SET SELECTION --
response:	ALL all data sets on tape are to be copied to disk.
response:	SCAN all data set names on tape are to be listed, no copy is done.
response:	dsname,dsname,... a selected set of data sets are to be copied to disk. The set of data set names may be continued to multiple-lines by typing a dash "-" immediately following the last dsname on a line.
response:	dsname,ETC all data sets from dsname on are to be copied to disk.
response:	dsname,ETC,SCAN all data set names from dsname on are to be listed. No copy is done.
response:	dsname,dsname,INC all data sets between the two dsnames, inclusively, are to be copied to disk. The first dsname should precede the second on the tape.
response:	dsname,dsname,INC,SCAN all data set names between the two dsnames, inclusively, are to be listed. No copy is done.
message:	SPECIFY OUTPUT DISK -- VOLUME=( ),TYPE= /PUB
response:	VOLUME=(,volserno),TYPE=direct access type where volserno is the volume serial number of the disk that is to contain the source data sets, and direct access type is either 2311, 2314, 3330, 333B, or 3350. If VOLUME is specified and TYPE is omitted, 3330 is assumed.
response:	PUB the source data sets are copied to public storage.

The TSS starter system machine configuration is shown in Figure 8. The macro instructions needed to generate this configuration are in Appendix E. A description of the symbolic device address assignment is in Appendix F.

A minimum machine configuration may be formed from a subset of the starter system hardware addresses. Once TSS is operational on such a minimum machine configuration, a SYSGEN that describes your particular hardware configuration may be performed.

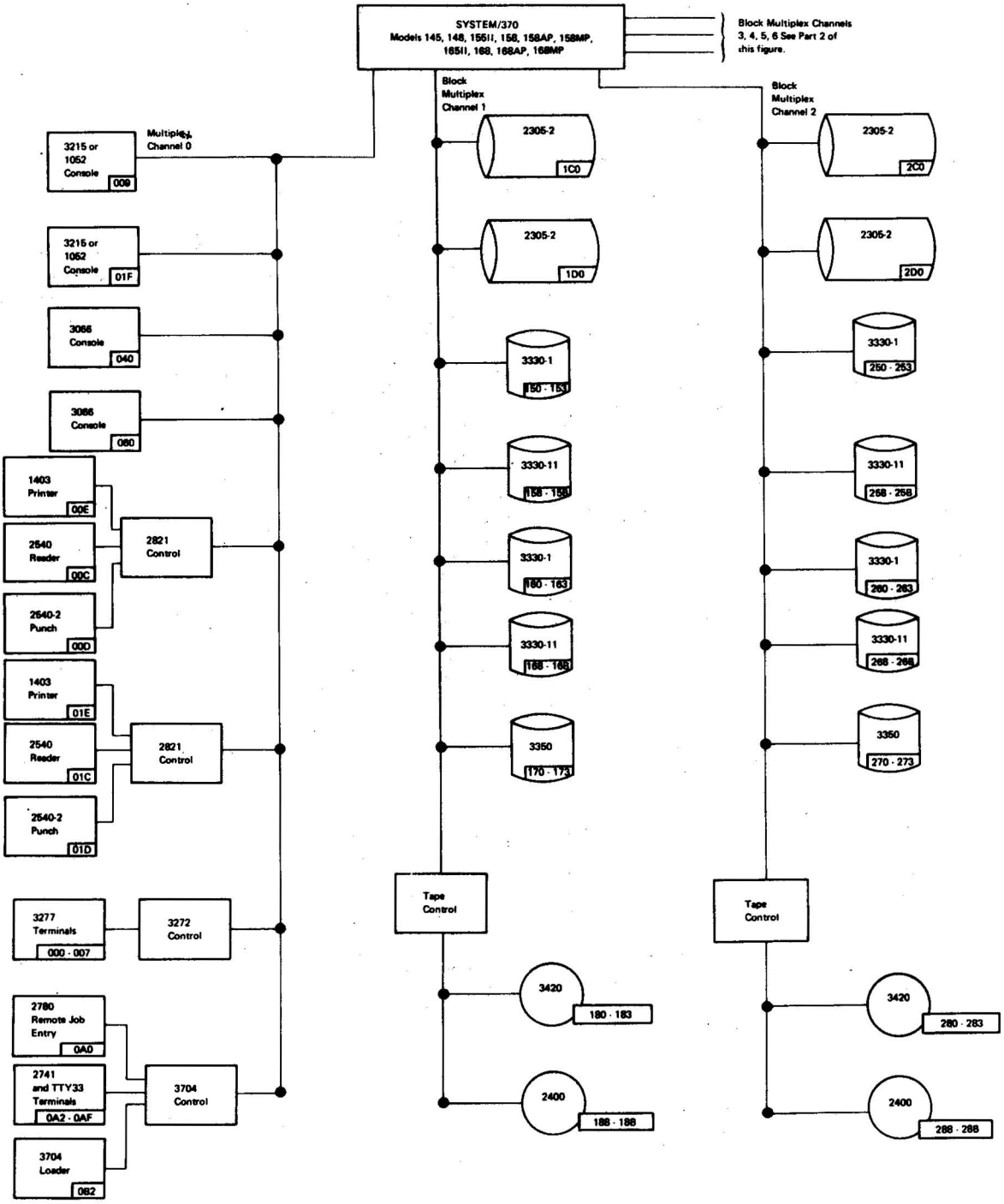


Figure 8. (Part 1 of 2) TSS Starter System Machine Configuration

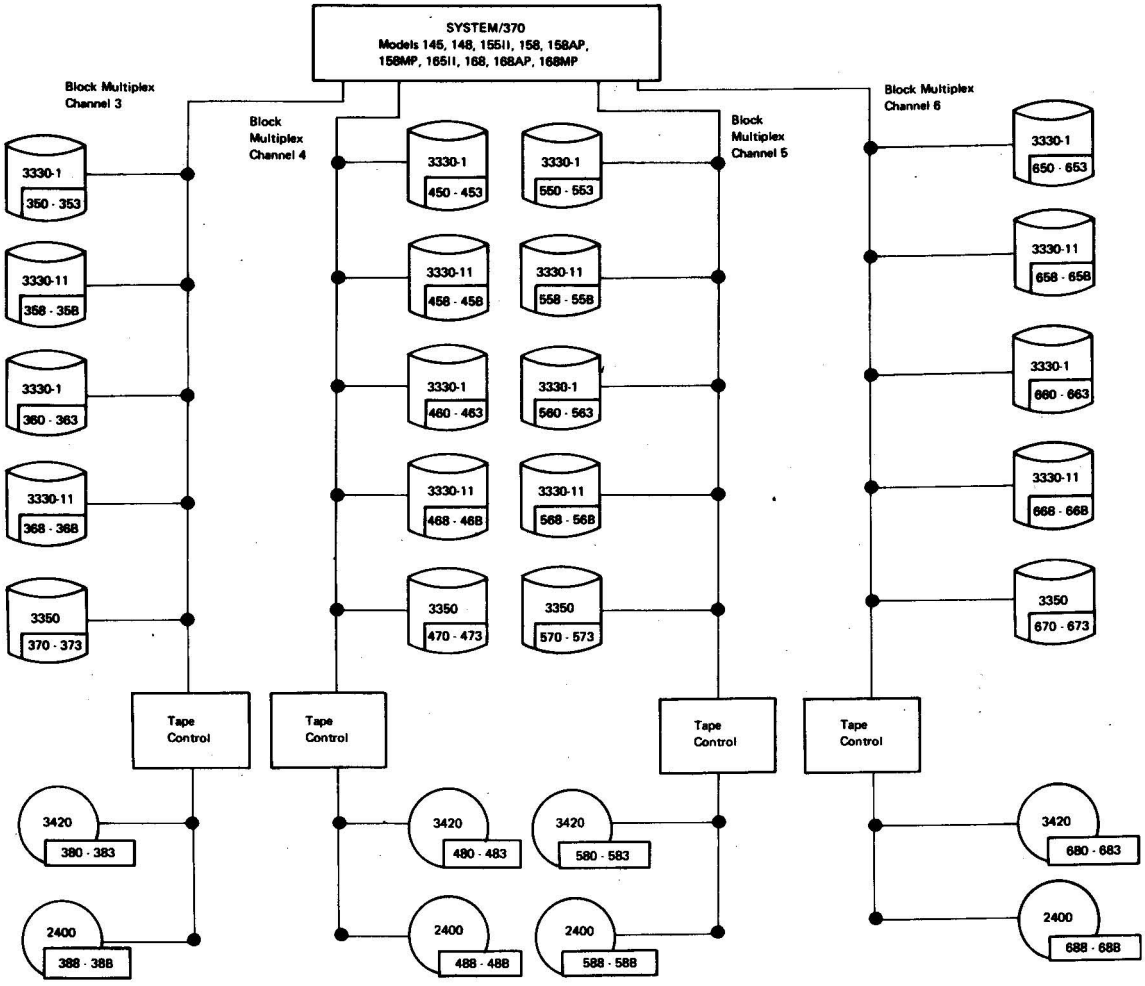


Figure 8. (Part 2 of 2) TSS Starter System Machine Configuration

APPENDIX E: STARTER SYSTEM SYSGEN MACRO INSTRUCTIONS

These system generation macro instructions create the starter TSS machine configuration shown in Figure 8 (Appendix D). A copy of these instructions is contained on the TSSRES volume under the name TSS*****.SOURCE.SYSGEN.

```
          TITLE 'TSS/370 STARTER SYSTEM GENERATION'
*
SYSGEX    START
*
          CPU   2
*
          STEM  2048,512
*
          CHANNEL CPU=0,MPX=(0),BMX=(1,2,3,4,5,6)
          CHANNEL CPU=1,MPX=(0),BMX=(1,2,3,4,5,6)
*
*                   OPERATOR DEVICE CONTROL UNITS
*
DSP00     DCU   UNIT=3066,PATH=(0040-40,1040-40)
DSP01     DCU   UNIT=3066,PATH=(1060-60,0060-60)
*
OPR00     DCU   UNIT=1052,MODEL=7,PATH=(0009-09,1009-09)
OPR01     DCU   UNIT=1052,MODEL=7,PATH=(101F-1F,001F-1F)
*
*                   DRUM CONTROL UNITS
*
DRM00     DCU   UNIT=2835,MODEL=2,PATH=(01C0-C7,11C0-C7)
DRM01     DCU   UNIT=2835,MODEL=2,PATH=(01D0-D7,11D0-D7)
*
DRM02     DCU   UNIT=2835,MODEL=2,PATH=(12C0-C7,02C0-C7)
DRM03     DCU   UNIT=2835,MODEL=2,PATH=(12D0-D7,02D0-D7)
*
*                   DISK CONTROL UNITS
*
DSK00     DCU   UNIT=3830,MODEL=1,PATH=(0150-5F,1150-5F)
DSK01     DCU   UNIT=3830,MODEL=1,PATH=(0160-6F,1160-6F)
DSK02     DCU   UNIT=3830,MODEL=1,PATH=(0170-77,1170-77)
*
DSK03     DCU   UNIT=3830,MODEL=1,PATH=(1250-5F,0250-5F)
DSK04     DCU   UNIT=3830,MODEL=1,PATH=(1260-6F,0260-6F)
DSK05     DCU   UNIT=3830,MODEL=1,PATH=(1270-77,0270-77)
*
DSK06     DCU   UNIT=3830,MODEL=1,PATH=(0350-5F,1350-5F)
DSK07     DCU   UNIT=3830,MODEL=1,PATH=(0360-6F,1360-6F)
DSK08     DCU   UNIT=3830,MODEL=1,PATH=(0370-77,1370-77)
*
DSK09     DCU   UNIT=3830,MODEL=1,PATH=(1450-5F,0450-5F)
DSK10     DCU   UNIT=3830,MODEL=1,PATH=(1460-6F,0460-6F)
DSK11     DCU   UNIT=3830,MODEL=1,PATH=(1470-77,0470-77)
*
DSK12     DCU   UNIT=3830,MODEL=1,PATH=(0550-5F,1550-5F)
DSK13     DCU   UNIT=3830,MODEL=1,PATH=(0560-6F,1560-6F)
DSK14     DCU   UNIT=3830,MODEL=1,PATH=(0570-77,1570-77)
*
DSK15     DCU   UNIT=3830,MODEL=1,PATH=(1650-5F,0650-5F)
DSK16     DCU   UNIT=3830,MODEL=1,PATH=(1660-6F,0660-6F)
DSK17     DCU   UNIT=3830,MODEL=1,PATH=(1670-77,0670-77)
```

```
*
*                  TAPE CONTROL UNITS
*
TAP00     DCU    UNIT=2803,MODEL=1,PATH=(0180-8F,1180-8F)
TAP01     DCU    UNIT=2803,MODEL=1,PATH=(1280-8F,0280-8F)
TAP02     DCU    UNIT=2803,MODEL=1,PATH=(0380-8F,1380-8F)
*
TAP03     DCU    UNIT=2803,MODEL=1,PATH=(1480-8F,0480-8F)
TAP04     DCU    UNIT=2803,MODEL=1,PATH=(0580-8F,1580-8F)
TAP05     DCU    UNIT=2803,MODEL=1,PATH=(1680-8F,0680-8F)
*
*                  UNIT RECORD CONTROL UNITS
*
UR00      DCU    UNIT=2821,MODEL=1,PATH=(000C-0E,100C-0E)
UR01      DCU    UNIT=2821,MODEL=1,PATH=(101C-1E,001C-1E)
*
*                   TERMINAL CONTROL UNITS
*
TU00      DCU    UNIT=3272,MODEL=1,PATH=(0000-07,1000-07)
TM00      DCU    UNIT=3704,MODEL=1,PATH=(00A0-AF,10A0-AF)
TRMIPL    DCU    UNIT=3704,MODEL=1,PATH=(00B2-B2,10B2-B2)
*
*                  OPERATOR DEVICE GROUPS
*
DISPLAY0 DEVGRP UNIT=3066,TYPE=ASYNCH,IOREQ=YES,PATH=(DSP00,0),        -
               ADDRESS=(0,240),ABUF=64,DCM=8
DISPLAY1 DEVGRP UNIT=3066,TYPE=ASYNCH,IOREQ=YES,PATH=(DSP01,0),        -
               ADDRESS=(0,260),ABUF=64,DCM=8
*
OPER0     DEVGRP UNIT=1050,MODEL=7,ABUF=64,DCM=4,PATH=(OPR00,0),       -
               ADDRESS=(0,209)
OPER1     DEVGRP UNIT=1050,MODEL=7,ABUF=64,DCM=4,PATH=(OPR01,0),       -
               ADDRESS=(0,21F)
*
*                  DRUM DEVICE GROUPS
*
DRUM00    DEVGRP UNIT=2305,MODEL=2,TYPE=PAGING,PATH=(DRM00,0),         -
               ADDRESS=(0,1)
DRUM01    DEVGRP UNIT=2305,MODEL=2,TYPE=PAGING,PATH=(DRM01,0),         -
               ADDRESS=(0,9)
DRUM02    DEVGRP UNIT=2305,MODEL=2,TYPE=PAGING,PATH=(DRM02,0),         -
               ADDRESS=(0,11)
DRUM03    DEVGRP UNIT=2305,MODEL=2,TYPE=PAGING,PATH=(DRM03,0),         -
               ADDRESS=(0,19)
*
*                  DISK DEVICE GROUPS
*
DISK00    DEVGRP UNIT=3330,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK00,0),   -
               ADDRESS=((0,40),(1,41),(2,42),(3,43))
DISK01    DEVGRP UNIT=333B,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK00,8),   -
               ADDRESS=((0,48),(1,49),(2,4A),(3,4B))
DISK02    DEVGRP UNIT=3330,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK01,0),   -
               ADDRESS=((0,50),(1,51),(2,52),(3,53))
DISK03    DEVGRP UNIT=333B,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK01,8),   -
               ADDRESS=((0,58),(1,59),(2,5A),(3,5B))
DISK04    DEVGRP UNIT=3350,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK02,0),   -
               ADDRESS=((0,60),(1,61),(2,62),(3,63))
*
DISK05    DEVGRP UNIT=3330,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK03,0),   -
               ADDRESS=((0,80),(1,81),(2,82),(3,83))
DISK06    DEVGRP UNIT=333B,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK03,8),   -
               ADDRESS=((0,88),(1,89),(2,8A),(3,8B))
DISK07    DEVGRP UNIT=3330,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK04,0),   -
               ADDRESS=((0,90),(1,91),(2,92),(3,93))
DISK08    DEVGRP UNIT=333B,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK04,8),   -
```

```
               ADDRESS=((0,98),(1,99),(2,9A),(3,9B))
DISK09    DEVGRP UNIT=3350,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK05,0),     -
               ADDRESS=((0,A0),(1,A1),(2,A2),(3,A3))
*
DISK10    DEVGRP UNIT=3330,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK06,0),     -
               ADDRESS=((0,C0),(1,C1),(2,C2),(3,C3))
DISK11    DEVGRP UNIT=333B,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK06,8),     -
               ADDRESS=((0,C8),(1,C9),(2,CA),(3,CB))
DISK12    DEVGRP UNIT=3330,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK07,0),     -
               ADDRESS=((0,D0),(1,D1),(2,D2),(3,D3))
DISK13    DEVGRP UNIT=333B,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK07,8),     -
               ADDRESS=((0,D8),(1,D9),(2,DA),(3,DB))
DISK14    DEVGRP UNIT=3350,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK08,0),     -
               ADDRESS=((0,E0),(1,E1),(2,E2),(3,E3))
*
DISK15    DEVGRP UNIT=3330,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK09,0),     -
               ADDRESS=((0,100),(1,101),(2,102),(3,103))
DISK16    DEVGRP UNIT=333B,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK09,8),     -
               ADDRESS=((0,108),(1,109),(2,10A),(3,10B))
DISK17    DEVGRP UNIT=3330,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK10,0),     -
               ADDRESS=((0,110),(1,111),(2,112),(3,113))
DISK18    DEVGRP UNIT=333B,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK10,8),     -
               ADDRESS=((0,118),(1,119),(2,11A),(3,11B))
DISK19    DEVGRP UNIT=3350,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK11,0),     -
               ADDRESS=((0,120),(1,121),(2,122),(3,123))
*
DISK20    DEVGRP UNIT=3330,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK12,0),     -
               ADDRESS=((0,140),(1,141),(2,142),(3,143))
DISK21    DEVGRP UNIT=333B,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK12,8),     -
               ADDRESS=((0,148),(1,149),(2,14A),(3,14B))
DISK22    DEVGRP UNIT=3330,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK13,0),     -
               ADDRESS=((0,150),(1,151),(2,152),(3,153))
DISK23    DEVGRP UNIT=333B,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK13,8),     -
               ADDRESS=((0,158),(1,159),(2,15A),(3,15B))
DISK24    DEVGRP UNIT=3350,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK14,0),     -
               ADDRESS=((0,160),(1,161),(2,162),(3,163))
*
DISK25    DEVGRP UNIT=3330,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK15,0),     -
               ADDRESS=((0,180),(1,181),(2,182),(3,183))
DISK26    DEVGRP UNIT=333B,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK15,8),     -
               ADDRESS=((0,188),(1,189),(2,18A),(3,18B))
DISK27    DEVGRP UNIT=3330,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK16,0),     -
               ADDRESS=((0,190),(1,191),(2,192),(3,193))
DISK28    DEVGRP UNIT=333B,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK16,8),     -
               ADDRESS=((0,198),(1,199),(2,19A),(3,19B))
DISK29    DEVGRP UNIT=3350,IOREQ=YES,FEATURE=(TROFL),PATH=(DSK17,0),     -
               ADDRESS=((0,1A0),(1,1A1),(2,1A2),(3,1A3))
*
*                 TAPE DEVICE GROUPS
*
TAPE00    DEVGRP UNIT=3420,MODEL=8,FEATURE=DUALDENS,PATH=(TAP00,0),      -
               ADDRESS=((0,70),(1,71),(2,72),(3,73))
TAPE01    DEVGRP UNIT=2402,FEATURE=(9-TRACK),PATH=(TAP00,0),             -
               ADDRESS=((8,78),(9,79),(A,7A),(B,7B))
*
TAPE02    DEVGRP UNIT=3420,MODEL=8,FEATURE=DUALDENS,PATH=(TAP01,0),      -
               ADDRESS=((0,B0),(1,B1),(2,B2),(3,B3))
TAPE03    DEVGRP UNIT=2402,FEATURE=(9-TRACK),PATH=(TAP01,0),             -
               ADDRESS=((8,B8),(9,B9),(A,BA),(B,BB))
*
TAPE04    DEVGRP UNIT=3420,MODEL=8,FEATURE=DUALDENS,PATH=(TAP02,0),      -
               ADDRESS=((0,F0),(1,F1),(2,F2),(3,F3))
TAPE05    DEVGRP UNIT=2402,FEATURE=(9-TRACK),PATH=(TAP02,0),             -
               ADDRESS=((8,F8),(9,F9),(A,FA),(B,FB))
*
```

```
TAPE06   DEVGRP UNIT=3420,MODEL=8,FEATURE=DUALDENS,PATH=(TAP03,0),      -
               ADDRESS=((0,130),(1,131),(2,132),(3,133))
TAPE07   DEVGRP UNIT=2402,FEATURE=(9-TRACK),PATH=(TAP03,0),             -
               ADDRESS=((8,138),(9,139),(A,13A),(B,13B))
*
TAPE08   DEVGRP UNIT=3420,MODEL=8,FEATURE=DUALDENS,PATH=(TAP04,0),      -
               ADDRESS=((0,170),(1,171),(2,172),(3,173))
TAPE09   DEVGRP UNIT=2402,FEATURE=(9-TRACK),PATH=(TAP04,0),             -
               ADDRESS=((8,178),(9,179),(A,17A),(B,17B))
*
TAPE10   DEVGRP UNIT=3420,MODEL=8,FEATURE=DUALDENS,PATH=(TAP05,0),      -
               ADDRESS=((0,1B0),(1,1B1),(2,1B2),(3,1B3))
TAPE11   DEVGRP UNIT=2402,FEATURE=(9-TRACK),PATH=(TAP05,0),             -
               ADDRESS=((8,1B8),(9,1B9),(A,1BA),(B,1BB))
*
*                 UNIT RECORD DEVICE GROUPS
*
READER00 DEVGRP UNIT=2540R,IOREQ=YES,FEATURE=(COLBNRY),PATH=(UR00,0),   -
               ADDRESS=(0,20C)
PUNCH00  DEVGRP UNIT=2540P,IOREQ=YES,FEATURE=(COLBNRY),PATH=(UR00,0),   -
               ADDRESS=(1,20D)
PRINTER0 DEVGRP UNIT=1403,MODEL=N1,MAXIO=4,IOREQ=YES,FEATURE=(UCS),     -
               PATH=(UR00,0),ADDRESS=(2,20E)
*
READER01 DEVGRP UNIT=2540R,IOREQ=YES,FEATURE=(COLBNRY),PATH=(UR01,0),   -
               ADDRESS=(0,21C)
PUNCH01  DEVGRP UNIT=2540P,IOREQ=YES,FEATURE=(COLBNRY),PATH=(UR01,0),   -
               ADDRESS=(1,21D)
PRINTER1 DEVGRP UNIT=1403,MODEL=N1,MAXIO=4,IOREQ=YES,FEATURE=(UCS),     -
               PATH=(UR01,0),ADDRESS=(2,21E)
*
*                 TERMINAL DEVICE GROUPS
*
TUBE01   DEVGRP UNIT=3270,ABUF=64,DCM=7,TYPE=ASYNCH,                   -
               PATH=(TU00,0),                                          -
               ADDRESS=((0,270),(1,271),(2,272),(3,273),               -
               (4,274),(5,275),(6,276),(7,277))
*
RJE00    DEVGRP UNIT=2780,MAXIO=3,FEATURE=(DIAL,BSYN),                  -
               PATH=(TM00,0),ADDRESS=(0,2A0)
*
TERMDIAL DEVGRP UNIT=2741,ABUF=32,DCM=2,HOLD=Y,TYPE=ASYNCH,             -
               FEATURE=(DSLNADP,DIAL,AUTOCALL),PATH=(TM00,0),          -
               ADDRESS=((2,2A2),(3,2A3),(4,2A4),(5,2A5),               -
               (6,2A6),(7,2A7),(8,2A8),(9,2A9),                        -
               (A,2AA),(B,2AB),(C,2AC),(F,2AF))
*
TERMTTY  DEVGRP UNIT=TTY33,ABUF=32,DCM=3,HOLD=Y,TYPE=ASYNCH,            -
               FEATURE=(DSLNADP,DIAL),PATH=(TM00,0),                   -
               ADDRESS=(D,2AD)
*
TERMDED  DEVGRP UNIT=2741,ABUF=32,DCM=2,HOLD=Y,TYPE=ASYNCH,             -
               FEATURE=(DSLNADP,DEDICATED),PATH=(TM00,0),              -
               ADDRESS=(E,2AE)
*
LDR3704  DEVGRP UNIT=3704,MAXIO=3,IOREQ=YES,PATH=(TRMIPL,0),            -
               ADDRESS=(0,2B2)
*
         OPCNSL (240,ALARM),(260,ALARM),                                -
               (209,ALARM,HRDCOPY),(21F,ALARM,HRDCOPY)
*
*                 TASK MANAGEMENT AND COMMAND SYSTEM
*
         TSKLMT CONV=200,BATCH=100,BACK=50,MTTADM=10,RTAMBUF=8,         -
               PGTBL=32,PSLMT=256
```

```
*
         DISPAR LCT=(1,10),LDMTR=2,AUXSP=(250,500),BLKSZE=(12,12)
*
         VMPAR  VCSLNG=20,SDST=25,OPTIONS=(PUBSEG,PACKSEG),              -
               TIME=07000000
*
         CLOP   TATYPE=9D3,DATYPE=333B,DEN=3,DAPAGES=(38,62),FIR=Y,      -
               DACYLS=(3,5),DATRKS=20,LIBPGS=(20,10),CFM=CARDS,          -
               PRIO=8,PFM=PAPER,DSORG=VS,MAV=512,AUTH=U,                 -
               PRVLG=(D,B,E,C,S,T,U),PRMTLMT=4,LABTYP=NL
*
         PUBVOL  MAXVOL=20
*
         GENSCB
         END
```

APPENDIX F: RULES FOR ASSIGNING SYMBOLIC DEVICE ADDRESS NOTATIONS

A symbolic device address, designated by as many as three hexadecimal digits, uniquely identifies a device although the device may have several hardware addresses. Symbolic device addresses should be arranged as if in table format.

1. Addresses of paging devices should be at the beginning of the table; drum storage units must precede disk storage units. Thus, the first paging drum would have an address of 1.

2. Each 2305 drum has eight addresses reserved for it. Only the first is assigned by the system programmer; the others are reserved by the system. Therefore, if an address on N is assigned to a 2305 drum, the system also reserves the addresses N+1 to N+7 for it.

3. After addresses have been assigned to all drum storage units, the next five addresses are reserved by the system for its own use. Thus, if the address N were assigned by an installation to the last 2305 drum, an address of N+D is the next available address.

4. Addresses are next assigned to disk storage units for paging and to system residence devices. Subsequent addresses are assigned to other devices; those devices that are apt to be used most frequently should have the lowest symbolic addresses.

5. Unused groups of addresses increase the size of several resident supervisor tables. While these gaps are useful for assigning addresses to new devices, they should be used with care.

The symbolic device addresses (SDAs) assigned to the devices in the machine configuration of Appendix D are shown in Table 1; corresponding physical device addresses (PDAs) are also shown. The first digit of the PDA is the cpu address, the second digit is the channel address, and the third and fourth digits are the control unit-device address.

Table 1. Symbolic Device Address Assignment in a Starter System

SDA	PDA	DEVICE
1	01C0 ,11C0	2305-2
9	01D0 ,11D0	2305-2
11	02C0 ,12C0	2305-2
19	02D0 ,12D0	2305-2
40-43	0150-53,1150-53	3330-1
48-4B	0158-5B,1158-5B	3330-11
50-53	0160-63,1160-63	3330-1
58-5B	0168-6B,1168-6B	3330-11
60-63	0170-73,1170-73	3350
70-73	0180-83,1180-83	3420
78-7B	0188-8B,1188-8B	2402
80-83	0250-53,1250-53	3330-1
88-8B	0258-5B,1258-5B	3330-11
90-93	0260-63,1260-63	3330-1
98-9B	0268-6B,1268-6B	3330-11

Table 1 -- continued

A0-A3	0270-73,1270-73	3350
B0-B3	0280-83,1280-83	3420
B8-BB	0288-8B,1288-8B	2402
C0-C3	0350-53,1350-53	3330-1
C8-CB	0358-5B,1358-5B	3330-11
D0-D3	0360-63,1360-63	3330-1
D8-DB	0368-6B,1368-6B	3330-11
E0-E3	0370-73,1370-73	3350
F0-F3	0380-83,1380-83	3420
F8-FB	0388-8B,1380-8B	2402
100-103	0450-53,1450-53	3330-1
108-10B	0458-5B,1458-5B	3330-11
110-113	0460-63,1460-63	3330-1
118-11B	0468-6B,1468-6B	3330-11
120-123	0470-73,1470-73	3350
130-133	0480-83,1480-83	3420
138-13B	0488-8B,1488-8B	2402
140-143	0550-53,1550-53	3330-1
148-14B	0558-5B,1558-5B	3330-11
150-153	0560-63,1560-63	3330-1
158-15B	0568-6B,1568-6B	3330-11
160-163	0570-73,1570-73	3350
170-173	0580-83,1580-83	3420
178-17B	0588-8B,1588-8B	2402
180-183	0650-53,1650-53	3330-1
188-18B	0658-5B,1658-5B	3330-11
190-193	0660-63,1660-63	3330-1
198-19B	0668-6B,1668-6B	3330-11
1A0-1A3	0670-73,1670-73	3350
1B0-1B3	0680-83,1680-83	3420
1B8-1BB	0688-8B,1688-8B	2402
209	0009 ,1009	1050-7
20C	000C ,100C	2540R
20D	000D ,100D	2540P
20E	000E ,100E	1403
21C	001C ,101C	2540R
21D	001D ,101D	2540P
21E	001E ,101E	1403
21F	001F ,101F	1050-7
240	0040 ,1040	3066
260	0060 ,1060	3066
270-277	0000-07,1000-07	3270
2A0	00A0 ,10A0	2780
2A2-2AC	00A2-AC,10A2-AC	2741
2AD	00AD ,10AD	TTY33
2AE	00AE ,10AE	2741
2B2	00B2 ,10B2	3704

APPENDIX G: PROCEDURE FOR MAKING TSS OPERATIONAL

The TSS starter system can run with the minimum TSS hardware configuration, and is delivered as two file-protected,unlabeled 9-track, 1600 bpi magnetic tape reels. The first tape contains the stand-alone independent utilities. The second tape is a disk restore for TSSRES (the first public volume).

1. Tape Contents

   a. The utility programs (except core dump) may be loaded directly from the independent utility tape. Table 2 shows the order of the four utilities on the tape.

      Table 2. Independent utility order

Program	Module ID
DASDI	CEBDI
DUMP/RESTORE	CEBDR
VAM2UT	CEBVP
Core Dump	CEBCP

   b. The contents of the disk volume TSSRES after being restored is described in Appendix B.

2. Initialize disks from independent utility tape

   a. Mount the independent utility tape, or the tape reel copied from the original tape, on any available 9-track, 1600 bpi tape drive and set the address of this drive in the LCAD unit switches on the operator's console. Mount two disk packs, 3330 or larger, on available disk drives.

   b. Be certain that the load-unit switches contain the three-digit address of the utility tape mounted in accordance with 2a, above. Press the STOP key and then the LOAD key. The LOAD light will come on, and TSS DASDI will be loaded from the tape. When the load procedure is complete, the WAIT light will come on.

   c. Press the REQUEST key on the console keyboard; this message will be typed out:

      CEB105A DEFINE INPUT DEVICE. DASDI X.X

      (x.x indicates the release level of the TSS associated with this version of DASDI).

      The operator enters the response in either upper- or lower-case characters:

      input=xxxx cuu EOB

      where xxxx is the type of the device in which the DASDI control cards have been placed (2540), and cuu is the physical hexade-

cimal address of that device -- c is the channel and uu is the unit (00E, 061). EOB is the end-of-block character.

The DASDI control cards are specified in Independent Utilities. These control cards enable the installation to initialize or "DASDI" the two disk packs necessary for a minimum configuration. DASDI is also used to initialize other public volumes, define paging space on new public volumes (other than TSSRES), and to initialize private disk volumes. The characteristics of the two packs needed for the minimum configuration are shown in Table 3.

Table 3. Disk pack characteristics

TODEV	FORMTYPE	VOLTYPE	NEWVOLID
3330 or larger	VAM2	PUBLIC	TSSRES
3330 or larger	VAM2	PAGING	PAGING

d. When DASDI is complete, the message END OF JOB is printed on the message device (1052 or 1403). If control cards for all devices to be initialized have been loaded, DASDI will continue with the next device in a stacked-job environment with no further intervention.

e. In addition to the minimum number of disk packs to be initialized, the 2301 and 2305 drums in the installation's total system configuration must also be initialized via DASDI. (See the Independent Utilities manual.)

3. Restoring TSSRES tape to disk

a. When all disk packs and drums have been initialized via DASDI, place control cards for dump/restore in the card reader (2540) and IPL the independent utility tape once again without rewinding. The system returns the message:

CEB105A DEFINE INPUT DEVICE TSS DUMP/RESTORE x.x

when the request key is pressed. This second IPL will bring in the dump/restore program. The keyword parameters TODEV=xxxx and TOADDR=cuu on the control cards (see below) should reflect the device and physical address of the disk pack that will be the TSSRES public volume (See the Independent Utilities manual).

4. Startup basic system

The following is the operating procedure for starting up a basic TSS system. See Section 2 for notes concerning the minimum configuration.

a. Load PRELUDE from TSSRES volume by setting load selector switches and pressing console LOAD key; PRELUDE locates and reads the Independent Access Method module; the wait state is entered.

b. Define the main operator's terminal by pressing the REQUEST key. The following is an example of system prompting messages and operator responses that take place during a TSS Startup:

```
SYSTEM:     ?
OPERATOR:   ls   (This indicates that a TSS long start is desired)

SYSTEM:     ENTER PHY ADDRESS OF CARD RDR OR DEFAULT = ECB.
OPERATOR:   (presses ECB)
                                        (If the address of a card reader
                                        had been entered, the following
                                        operator responses would have
                                        been read from cards)

SYSTEM:     ENTER ADDR. OF PAGING DISK
OPERATOR:   204

SYSTEM:     PRINT MAPS? 0 1 2 3 ... NONE IVM RESSUP BOTH
OPERATOR:   3

SYSTEM:     ENTER CODE FOR FUNCTIONS NOT TO BE LOADED
            ALL FUNCTIONS WANTED = EOB
OPERATOR:   99  (The system performance monitor is not to be loaded)

SYSTEM:     QUICK START REQUESTED?  IF Y ENTER ADDR OF PACK
            FOR QUICK START DATA SET .  N=EOB
OPERATOR:   201                         (Assuming TSSRES  pack on 201
                                        and Quickstart data set is
                                        to be created.)

SYSTEM:     DELTA DATA SETS? Y OR N
OPERATOR:   n

SYSTEM:     CURRENT CLOCK VALUE IS 05/14/36 07.23.11 IS CLOCK CORRECT?
OPERATOR:   n

SYSTEM:     ENTER DATE AND TIME AS
            MM/DD/YY HH.MM.SS
OPERATOR:   11/17/76 10.04.00

SYSTEM:     DEPRESS TOD SWITCH
OPERATOR:   (Presses the time of day clock enable switch)

SYSTEM:     CURRENT CLOCK VALUE IS 11/17/76 10.04.01 IS CLOCK CORRECT?
OPERATOR:   y

SYSTEM:     $    (Enters the Resident Support System)
OPERATOR:   run    (RSS exits to TSS)

SYSTEM:     BULKIO REQUIRED?
OPERATOR:   y

SYSTEM:        Y ACCEPTED
            10:06:49          SYSOPER0 CZAFM LOGON AT 10:06 TASKID=0001
                               SDA=0078
            10:06:49          TSS00001 BATCH MONITOR HAS BEEN INITIALIZED
            10:06:49 R=0001   SYSOPER0 VALIDATE THE SYSTEM HARDWARE
                               CONFIGURATION.  REPLY OK WHEN DONE
OPERATOR:   (Presses REQUEST)
            reply 1,ok

SYSTEM:     STARTUP COMPLETE, USERS MAY LOG ON
            10:07:12 SYSOPER0 CZAFM LOGON AT 10:07 TASKID=0002
            10:07:31 SYSOPER0 *CZAWS* ASNBD'S DONE: A010-01 A011-02
```

5. Assemble system generation macro instructions

   When STARTUP is complete, a system programmer may log on (userid TSS,password IBM). The system generation source statements for

the first system generation are typed from a terminal or read from a card reader. This section shows the script necessary to generate a system. The dsname of the data set (defined as a JOBLIB) that will contain the system generation module is SYSGEN.MODULE.

```
SYSTEM
PROGRAMMER:     (presses  REQUEST key)
                logon tss,ibm

SYSTEM:         TASKID=00E4 LOGON AT 10:08 ON 11/15/76
SYS. PROGR.:    ddef dd1,vp,dsname=sysgen.module,option=joblib
                ddef mac1,vi,dsname=asmmac(0)
                ddef ndx1,vs,dsname=asmndx(0)
                ddef mac2,vi,dsname=genmac(0)
                ddef ndx2,vs,dsname=genndx(0)
                asm sysgen,n,(mac1,ndx1,mac2,ndx2),,n,n,y,y,n,n,y

SYSTEM:         100
                    .
                    .
                    .
                System programmer enters system generation
                macro instructions; refer to Appendix E
                for example
                    .
                    .
                    .
```

If the assembly is error-free, the system prompts with an underscore. Otherwise, the system asks the system programmer to enter modifications.

```
SYS. PROGR.:    print list.sysgen(0),,,edit,erase

SYSTEM:         PRINT BSN0264
SYS. PROGR.:    logoff
SYSTEM:         LOGOFF AT 10:32 ON 11/17/76
                TERMINAL LOGICALLY DISCONNECTED, RECONNECT OR
                HANG UP.
```

If this procedure is to be repeated without modifications, before attempting to reassemble the SYSGEN macro instructions the system programmer must explicitly erase the SOURCE.SYSGEN data set and the SYSGEN.MODULE data set.

6. Execute the APGEN procedure

When the system programmer has determined the accuracy and completeness of assembly, he executes the APGEN script to generate a TSS that is adapted to his installation.

Note: APGEN takes the output of the SYSGEN assembly and updates system modules and tables residing on the TSSRES volume. (Assume system programmer has logged on.)

```
SYS. PROGR.:    execute apgen

SYSTEM:         BSN=0265
SYS. PROGR.:    logoff
SYSTEM:         LOGOFF AT 10:59 ON 11/17/76
                TERMINAL LOGICALLY DISCONNECTED, RECONNECT OR
                HANG UP.
```

At this time, the SYSGEN.MODULE data set can be erased if the procedure to assemble the SYSGEN macro instructions is to be repeated.

The following warnings should be kept in mind.

a. The APGEN procedure updates the TSSRES volume in place; whenever APGEN is run, the TSSRES volume used will contain the new system. Be sure to make save tapes of any systems which should be kept. Only the TSSRES volume is affected during APGEN.

b. The data set names used during the first log-on must be the same as those in APGEN. This can be assured by following the script exactly, except for user password and charge number.

c. During the first STARTUP, there must be no public VAM2 disk (other than the TSSRES) on the system. If there is such a public VAM2 disk, one of the system queues will appear on it and that disk will always be required as part of the system. Furthermore, no housekeeping routines may be run against that disk since the erasure of that queue will make the system inoperable.

d. System generation data sets should be referred to by relative notation. When versions of TSS are released, the absolute generation numbers may be changed dynamically during the updating process.

e. Messages, like the following, received during APGEN may be ignored:

```
W***THE STANDARD ENTRY POINT OF INPUT MODULE SYSGEN CAN-
NOT BE SAVED AS AN AUXILIARY ENTRY POINT.

W***THE FOLLOWING EXTERNAL REFERENCES ARE UNRESOLVED:
CEHAPA, CEHASA, CEHAEA, CEHADA, CEHACA, CEHAQA, CEAKTS,
CEAB81, CEAB91, CEAA40, CEAIAA, CEANBA, CEAA31, CEAHQP,
CEANAA, CHBECXRB, CHBECXRC
```

f. Messages, like the following, printed on the Startup memory map may be ignored:

```
CSECT CIPIO1 MISSING OR POD FORMAT ERROR.
```

g. The assembler work space must be expanded if exceptionally large assemblies are to be run. The following patches will expand the work spaces:

```
SET CEVW1.(,4)=X'00000180'
SET CEVW2.(,4)=X'00000180'
SET CEVW3.(,4)=X'0000005C'
```

These patches may be entered through PCS by properly authorized users and should not be permanently applied to your system, since they will cause unnecessary paging for small assemblies.

h. If the system crashes, all current accounting information for the active users will be lost. This information will not be written out after the crash. The user may run UPDTUSER to recover permanent storage accounting.

After executing the APGEN script and logging off, the system programmer has initialized a software system tailored to the installation's hardware system. Recommended: shut down the system and make a save tape of the new TSSRES volume. When the operator IPLs from the new TSSRES volume, PRELUDE reads in STARTUP. After STARTUP is completed, the system manager can log on to join other users to the system; TSS is ready for use.

7. After the APGEN script logs off, the software system is tailored to the installation's hardware system. APGEN has updated the TSSRES in place and completed the system generation process. The system should now be shut down and, using the DUMP/RESTORE program, a dump should be made to tape of the new TSSRES.

   CAUTION: Saving the TSSRES volume should facilitate recovery from system or hardware problems that may occur in the next step.

8. After the startup is complete, TSS is ready for use. After all users are joined, it is advisable to shut down the system and make a dump of the TSSRES volume to tape.

9. With the starter system complete, the public volumes may now be added. If there are old public volumes as the result of a previous TSS release, the volumes can now be introduced, the system started up again, and the catalog reconstructed using the CVV command, or the public volumes reconstructed and cataloged using the RPS command. The TSSRES volume and all public volumes should then be saved.

10. If new public volumes were added, or old public volumes introduced, that contain paging bands, the dedicated paging pack may be removed.

APPENDIX H: SAMPLE MAINTENANCE PROCEDURES

This sample maintenance procedure is for a system that has changes for several system data sets. In the first example, the maintenance release tape contains no change for any of the system generation macro instructions (that requires a SYSGEN assembly), or any utility macro instructions, or for prelude, and contains the following files:

File Number	Contents
1	Replacement source modules
2	TAPE.SCRIPT
3	TAPE.ASMMAC
4	TAPE.ASMNDX
5	TAPE.SYSMAC
6	TAPE.MACNDX
7	TAPE.GENMAC
8	TAPE.GENNDX
9	TAPE.SYSIAM
10	TAPE.STARTUP
11	TAPE.SYSUTL
12	TAPE.SYSIVM
13	TAPE.RESSUP
14	TAPE.RSSSUP
15	TAPE.SYSLIB
16	TAPE.SYSERP
17	TAPE.UPDATE.ASMMAC
18	TAPE.UPDATE.SYSMAC
19	TAPE.UPDATE.GENMAC
20	TAPE.APGEN
21	TAPE.UPDATES

After starting up your current operational system, it is advisable to restrict the system's use to userid TSS during the update session. After the session is complete, you should shut down and save all your system volumes. (See Figure 9 for the conversational session.)

```
LOGON TSS,password          *Logon system programmer*
DDEF DDTV,PS,TAPE.SCRIPT,(DEN=3),(TA,9D3),,(,VAMTAP),(2,NL),OLD

                            *Define and have the 1600 bpi
                             change tape mounted*

TV TAPE.SCRIPT,SCRIPT.UPDATE

                            *Tape to VAM copy of the script data set*

RELEASE DDTV                *Release VAMTAP for the execute*

EXECUTE SCRIPT.UPDATE       *Start the update task*

LOGOFF                      *And LOGOFF the conversational task*

Note:  Allow the execute task to run to completion and examine SYS-
       OUT before shutting down and saving your new system.
```

Figure 9. Conversational Update Task

For the purposes of the second example, the maintenance release tape contains changes to the SYSGEN macro instructions that make a new SYSGEN assembly necessary. (See Figure 10 for the conversational session.)

```
LOGON TSS password                *Logon system programmer*
DDEF DDTV,PS,TAPE.SCRIPT,(DEN=3),(TA,9D3),,(,VAMTAP),(2,NL),CLD

                                  *Define and have the 1600 bpi
                                   change tape mounted*

TV TAPE.SCRIPT,SCRIPT.UPDATE      *Tape to VAM copy of the script to
                                   public storage*
PRINT SCRIPT.UPDATE               *Print a copy for edit purposes*

EDIT SCRIPT.UPDATE2               *Start a second script
0000100 LOGON TSS                  for parts three and four*

0000200_EXCERPT SCRIPT.UPDATE,N1=15300,N2=LAST

                                  *Excerpt parts three and four of the
                                   second script*
       _END                       *The second script is complete*

EDIT SCRIPT.UPDATE                *Remove parts three and four
0019500_EXCISE 15300,LAST          from SCRIPT.UPDATE*
_INSERT 15200,100
0015300 LOGOFF                    *SCRIPT.UPDATE is now parts one and
                                   two*
_END

PRINT SCRIPT.UPDATE               *Revised

PRINT SCRIPT.UPDAT2               *Excerpted

EXECUTE SCRIPT.UPDATE             *This will complete parts one and two
                                   and setup for the update*

Note:  Wait for this nonconversational task to complete and examine
       SYSOUT before continuing.
DDEF SRC1,VI,NEW.SYSMAC,DISP=OLD
DDEF NDX1,VS,NEW.MACNDX,DISP=OLD   *Now your DDEFs
DDEF SRC2,VI,NEW.ASMMAC,DISP=OLD    will set up the new
DDEF NDX2,VS,NEW.ASMNDX,DISP=OLD    SYSGEN macros*
DDEF SRC3,VI,NEW.GENMAC,DISP=OLD
DDEF NDX3,VS,NEW.GENNDX,DISP=OLD
DDEF LIB,VP,SYSGEN.MODULE,OPTION=JOBLIB,DISP=NEW
ASM SYSGEN,Y,(SRC1,NDX1,SRC2,NDX2,SRC3,NDX3)
RELEASE LIB
EXECUTE NEW.APGEN                 *This new APGEN was included in part
                                   one*
Note.  Wait for APGEN to complete and log off

RELEASE SRC1                      *Release macro libraries for
RELEASE NDX1                       execute*
RELEASE SRC2
RELEASE NDX2
RELEASE SRC3
RELEASE NDX3

EXECUTE SCRIPT.UPDATE2            *Now you can complete the update*

LOGOFF                            *Conversational task complete*

Note:  When the last task (SCRIPT.UPDATE2) logs off, and you have
       examined SYSOUT your new system is complete and you can shut-
       down and save it.
```

Figure 10. Update and Assemble SYSGEN Macro Instructions

APPENDIX I: SAMPLE DYNAMIC MODIFICATION PROCEDURES

1. Load PRELUDE from the IPL volume. When the independent access method module has been read in, the system enters the wait state.

2. Press the REQUEST key to define the main operator's terminal.

3. The STARTUP process proceeds as shown in Appendix G, under 4, until the request for dynamic modifications is to be answered.

```
SYSTEM:     DELTA DATA SETS? Y OR N.
OPERATOR:   y
```

4. To dynamically modify the system with a private volume, having hardware address 0207 and containing data sets in the following order, the operator may specify one of various searches.

```
Order on volume:
   USERID1*.IVM01.XX
   USERID2*.IVM01.YY
   USERID3*.IVM02.XX
   USERID1*.SUP02.Z
   USERID1*.SUP02.Y
   USERID1*.IVM03.AA
   USERID1*.IVM04
   USERID2*.SUP03.BB.C
   USERID3*.RSS01.W
```

These are the searches that result, for the various responses to:

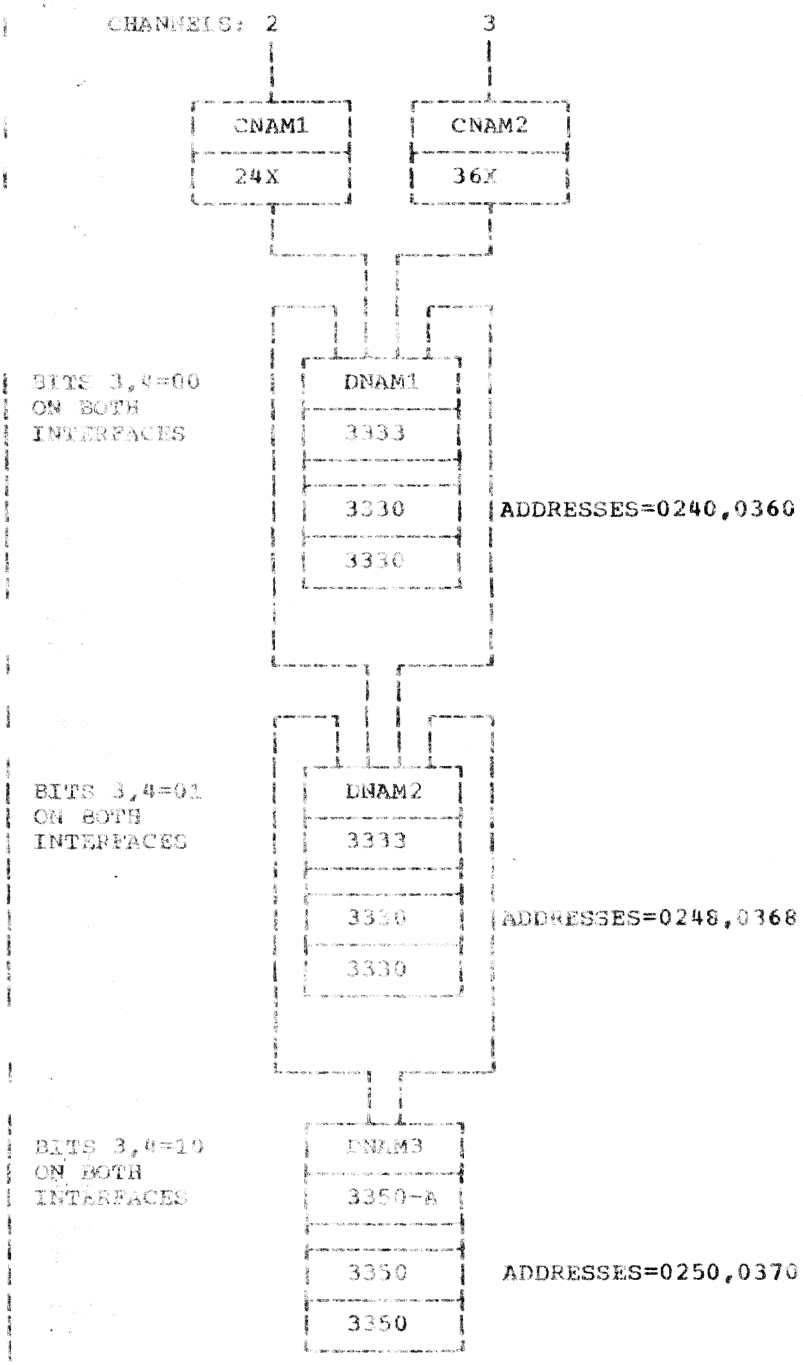
```
SYSTEM:     ENTER DEVICE ADDRESS AND DATA SET SPECIFICATIONS.

OPERATOR:   0207,userid1*

     Resulting search:
       USERID1*.IVM01.XX    TSS*****.RSSSUP     USERID1*.SUP02.Z
       USERID1*.IVM03.AA                        USERID1*.SUP02.Y
       USERID1*.IVM04                           TSS*****.RESSUP
       TSS*****.SYSIVM

OPERATOR:   0207,userid2*.ivm01,userid3*

     Resulting search:
       USERID2*.IVM01.YY    USERID3*.RSS01.W    TSS*****.RESSUP
       USERID3*.IVM02.XX    TSS*****.RSSSUP
       TSS*****.SYSIVM

OPERATOR:   0207,userid1*.sup02.y,userid1*.sup02.z

     Resulting search:
       TSS*****.SYSIVM      TSS*****.RSSSUP     USERID1*.SUP02.Y
                                                USERID1*.SUP02.Z
                                                TSS*****.RESSUP
```

5. The STARTUP process continues as in Appendix G, under 4.

APPENDIX J: STRING SWITCH SYSGEN EXAMPLES

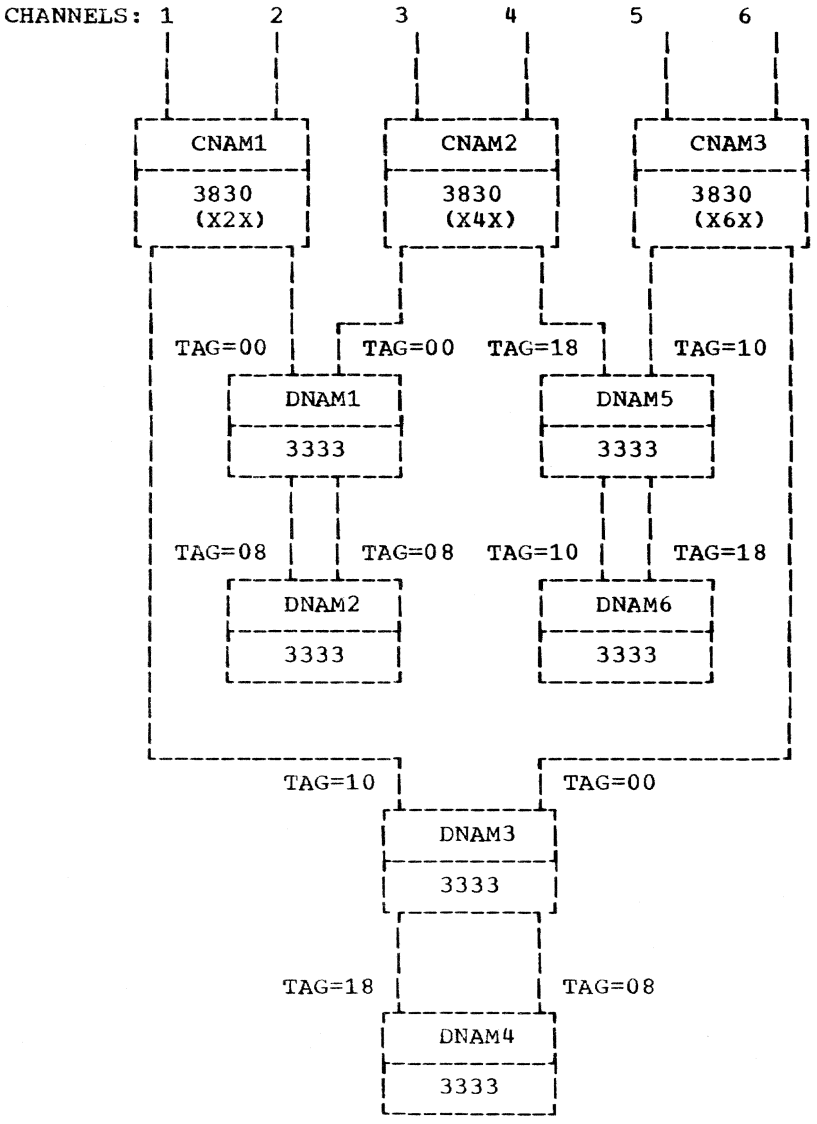
To clarify the SYSGEN macros necessary to generate the pathfinding tables for string switched devices, two examples are given in this appendix.

Example 1: symmetric string switched devices -- two channels, two control units, and three device groups -- is as follows:



```
CNAM1 DCU     UNIT=3830,MODEL=2,PATH=(0240-5F)
CNAM2 DCU     UNIT=3830,MODEL=2,PATH=(0360-7F)

DNAM1 DEVGRP UNIT=3330,PATH=((CNAM1,0),(CNAM2,0)),
             ADDRESS=((0,16),(1,17))
DNAM2 DEVGRP UNIT=3330,PATH=((CNAM1,8),(CNAM2,8)),
             ADDRESS=((0,18),(1,19))
DNAM3 DEVGRP UNIT=3350,PATH=((CNAM1,10),(CNAM2,10)),
             ADDRESS=((0,1A),(1,1B),(2,1C))
```

Example 2: complex (32 drive addressing) string switched devices -- six channels, three control units, and six device groups -- is as follows:



Device groups have the following base addresses:

```
      DNAM1 0120,0220,0340,0440
      DNAM2 0128,0228,0348,0448
      DNAM3 0130,0230,0560,0660
```

```
|       DNAM4 0138,0238,0568,0668
|       DNAM5 0358,0458,0570,0670
|       DNAM6 0350,0450,0578,0678
```

| Note that the string switch tags are not symmetrical and contribute bits 3 and 4 of the device address. Three DCU statements and six DEVGRP statements will be required. The 3330 devices have been omitted for clarity.

```
| CNAM1 DCU     UNIT=3830,MODEL=2,PATH=(0120-3F,0220-3F)
| CNAM2 DCU     UNIT=3830,MODEL=2,PATH=(0340-5F,0440-5F)
| CNAM3 DCU     UNIT=3830,MODEL=2,PATH=(0560-7F,0660-7F)

| DNAM1 DEVGRP  UNIT=3330,PATH=((CNAM1,0),(CNAM2,0)),
|               ADDRESS=((0,10),(1,11),...)
| DNAM2 DEVGRP  UNIT=3330,PATH=((CNAM1,8),(CNAM2,8)),
|               ADDRESS=((0,18),(1,19),...)
| DNAM3 DEVGRP  UNIT=3330,PATH=((CNAM1,10),(CNAM3,0)),
|               ADDRESS=((0,20),(1,21),...)
| DNAM4 DEVGRP  UNIT=3330,PATH=((CNAM1,18),(CNAM3,8)),
|               ADDRESS=((0,28),(1,29),...)
| DNAM5 DEVGRP  UNIT=3330,PATH=((CNAM2,18),(CNAM3,10)),
|               ADDRESS=((0,30),(1,31),...)
| DNAM6 DEVGRP  UNIT=3330,PATH=((CNAM2,10),(CNAM3,18)),
|               ADDRESS=((0,38),(1,39),...)
```

INDEX

Where more than one page reference is given, the first page number indicates the major reference. All references are within plus or minus one of the indicated page number(s).

IBM Technical Newsletter

This Newsletter No.	GN20-3809
Date	February 11, 1978
Base Publication No.	GC28-2010-7
File No.	S370-31
Previous Newsletters	None

IBM Time Sharing System
System Generation and Maintenance

This Technical Newsletter, a part of Release 3.6 of the IBM Time Sharing System, provides replacement pages for the above-named publication. Pages to be replaced (or added) are:

Cover Page - Edition Notice	43, 44
Preface - Table of Contents	47, 48
11, 12	49, 50
13, 14	51, 52
15, 16	57, 58
17, 18	59, 60
18.1 - 18.2 (add)	62.1, 62.2 (add)
27, 28	62.3, 62.4 (add)
29, 30	63, 64
35, 36	65, 66
37, 38	
39, 40	*62.4 is blank
41, 42	

Changed areas in text and figures are indicated by a vertical bar in the left and/or right margins.

Please file this cover letter at the back of the manual to maintain a complete record of the changes.

IBM Corporation, Time Sharing System, Dept. 80M, 1133 Westchester Ave., White Plains, N.Y. 10604

Printed in U.S.A.

| File No. S370-31
Order No. GC28-2010-7

Systems Reference Library

IBM Time Sharing System

System Generation and Maintenance

This publication explains how an installation specifies, creates, maintains, and modifies an installation-adapted IBM Time Sharing System. The intended audience is the system programmer charged with system generation and maintenance.

Part I is an overall description of system generation and maintenance; Part II describes the creation of a basic time-sharing system; Part III describes the macro instructions used to adapt the basic system to the installation's needs; Part IV explains the system maintenance process, and Part V shows how to modify the system for the duration of a startup-to-shutdown session. The appendixes contain examples of system generation, maintenance, and modification, reference material on device addressing and system libraries, a sample machine configuration, and sample system generation macro instructions.

Prerequisite Publications

The reader must be familiar with the information presented in: IBM Time Sharing System: Concepts and Facilities, GC28-2003

Eighth Edition (December 1976)

This is a major revision of, and makes obsolete, GC28-2010-6.

This edition is current with Release 3.0 of IBM Time Sharing System/370 (TSS/370), and remains in effect for all subsequent versions or modifications of TSS unless otherwise noted. Significant changes or additions to this publication will be provided in new editions or Technical Newsletters.

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 this form has been removed, comments may be addressed to IBM Corporation, Time Sharing System, Dept. 80M, 1133 Westchester Avenue, White Plains, New York 10604.

GC28--2010--7

IBM TSS System Generation & Maintenance Printed in U.S.A. GC28--2010--7

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
360 Hamilton Avenue, White Plains, New York 10601
(International)