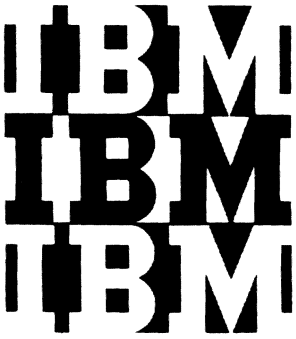
File No. S360-31
GY28-2015-6



Program Logic

Version 3.1

IBM System/360 Time Sharing System System Generation and Maintenance

Provides the information necessary to generate and maintain the IBM System/360 Time Sharing System (TSS/360).

The intended audience for this publication is the system programmer who is responsible for TSS/360 system generation and system maintenance.

The primary system generation services consists of the SYSBLD Prelude, System Build, Startup Prelude, and Startup; they generate a working TSS/360 system.

The first section of the book provides an overview of the system generation process. The next sections of the book describe the SYSBLD, STARTUP, and SYSGEN phases. The fifth section consists of flowcharts for the SYSBLD, STARTUP, and SYSGEN phases. The Appendixes provide information on system tables, global symbols, and data areas referred to by the system generation modules.

PREREQUISITE PUBLICATIONS

The reader must be familiar with the information presented in:

IBM System/360 Time Sharing System: System Generation and Maintenance, GC28-2010
IBM System/360 Time Sharing System: Control Blocks Program Logic Manual, GY28-2011

Seventh Edition (September 1971)

This is a major revision of, and makes obsolete IBM System/360 Time Sharing System: System Generation and Maintenance, GY28-2015-4. A new subroutine was added to the STARTUP phase and changes were made to the SYSBLD and STARTUP phases to support RTAM initialization, dynamic Q-cons, and user modules. Changes in the actual pages are indicated by a vertical bar in the margin at the left of the text.

This edition is current with Version 8, Modification 1, and remains in effect for all subsequent versions or modifications of IBM System/360 Time Sharing System unless otherwise noted. Significant changes or additions to this publication will be provided in new editions or Technical Newsletters. Before using this publication, refer to the latest edition of IBM System/360 Time Sharing System: Addendum, GC28-2043, which may contain information pertinent to the topics covered in this edition. The Addendum also lists the editions of publications that are applicable and current.

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

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

A form is provided at the back of this publication for reader's comments. If the form has been removed, comments may be addressed to IBM Corporation, Time Sharing System/360 Programming Publications, Department 643, Neighborhood Road, Kingston, New York 12401.

This publication, intended for use by the system programmer responsible for the generation and maintenance of the system, is divided into five sections representing an introduction, the three logical divisions of the system generation process, and the flowcharts.

Section 1 is an introduction to the system generation and maintenance process and provides an overall picture of the logic flow from program to program.

Section 2 explains the SYSBLD phase. It presents a brief picture of the SYSBLD Prelude routine and an overall picture of the SYSBLD process, explaining in detail the routines comprising the module.

Section 3 explains the Startup phase. It presents a picture of the Startup Prelude and Startup process, including Quickstart. Included in the Startup description is an introduction to the module, a description of the mainline routine, a description of the Link-loader subroutine a brief explanation of its supporting I/O and Link-loader subroutines, and a description of the Quickstart subroutines.

Section 4 contains a brief explanation of the SYSGEN phase. It presents the relationships of the system generation macro instructions and the APGEN command procedure.

Section 5 contains a set of flowcharts to be used in conjunction with the previous sections.

Also included in this publication is a set of appendixes. These contain descriptions of the system table fields set by the macro instructions and of the global symbols. In addition, all DSECTs that each module refers to are listed.

PREREQUISITE PUBLICATIONS

Familiarity with the material contained in the following publications is essential to the use of this manual.

IBM System/360 Principles of Operation, GA22-6821

IBM System/360 Time Sharing System: Concepts and Facilities, GC28-2003

IBM System/360 Time Sharing System: System Generation and Maintenance, GC28-2010

IBM System/360 Time Sharing System: System Control Blocks Program Logic Manual, GY28-2011

CONTENTS

ILLUSTRATIONS

Figure 1 illustrates the steps involved in generation of System/360 Time Sharing System (TSS/360). These steps are:

1. The user initializes the disks and does a dump/restore from the input tapes which produces an initialized disk configuration.

2. In the SYSBLD phase, the user provides minimum machine configuration parameters to the system, in order that the system may generate the required system tables and control blocks.

3. The user initiates the Startup phase, which takes information created during the previous (SYSBLD) phase and generates a time-sharing system. Because the input information is as yet limited, the output at this point is called a basic time-sharing system.

4. In the SYSGEN phase, operating within this time-sharing environmemt, the user initiates execution of a TSS Assembler, using the SYSGEN macro instructions, with installation-dependent parameters as the input data set, and creates module as the output data set. The user then updates the system tables and control blocks, by using this SYSGEN module as input to the APGEN command procedure.

5. The user initiates another Startup phase, which takes information from the previous (SYSGEN) phase and generates a time-sharing system. The result is now a full-fledged TSS/360 system.

This generation of a TSS/360 operating system is accomplished through the use of three independent modules, (Prelude, System Build (SYSBLD) and Startup), and the execution of two system modules during the System Generation (SYSGEN) phase (TSS Assembler and TSS*****.APGEN command procedure).

PRELUDE MODULE

The Prelude module (CEIAP) is loaded whenever an initial program load (IPL) occurs from the IPL volume. Prelude performs a configuration analysis, and then:

1. Locates the required SYSBLD or Startup module.

2. Reads the required module from disk into main storage.

3. Transfers control to the required module.

Since Prelude's initial function is to load SYSBLD, Prelude in its original state on the IPL tape is referred to as SYSBLD Prelude.

The Prelude program flow for the SYSBLD process is:

1. The SYSBLD Prelude is loaded from the IPL volume.

2. SYSBLD Prelude loads SYSBLD and exits to it.

3. At the completion of SYSBLD, SYSBLD permanently modifies SYSBLD Prelude to locate and read Startup. SYSBLD then writes Prelude back on disk. From this point on Prelude is known as Startup Prelude.

The Prelude program flow for the Startup process is:

1. The Startup Prelude is loaded from the IPL volume.

2. Startup Prelude loads Startup, and exits to it.

The Prelude module is described in detail at the beginning of Section 3, Startup Phase, and is briefly described as it applies to SYSBLD at the beginning of Section 2.

SYSBLD MODULE

The SYSBLD module (CEIFA) is independent of system programs. SYSBLD functions in main storage as a non-reenterable auxiliary module making up a group of stand-alone routines. Running in a nonconversational environment, its main function is to provide information that Startup can use to initialize a basic time-sharing system (of minimum machine configuration).

Initially SYSBLD locates the operator's terminal and interrogates the operator for minimum system parameters. This data is used to update the existing Pathfinding tables, the Symbolic Device Allocation table (CHBSDA), the Available Devices

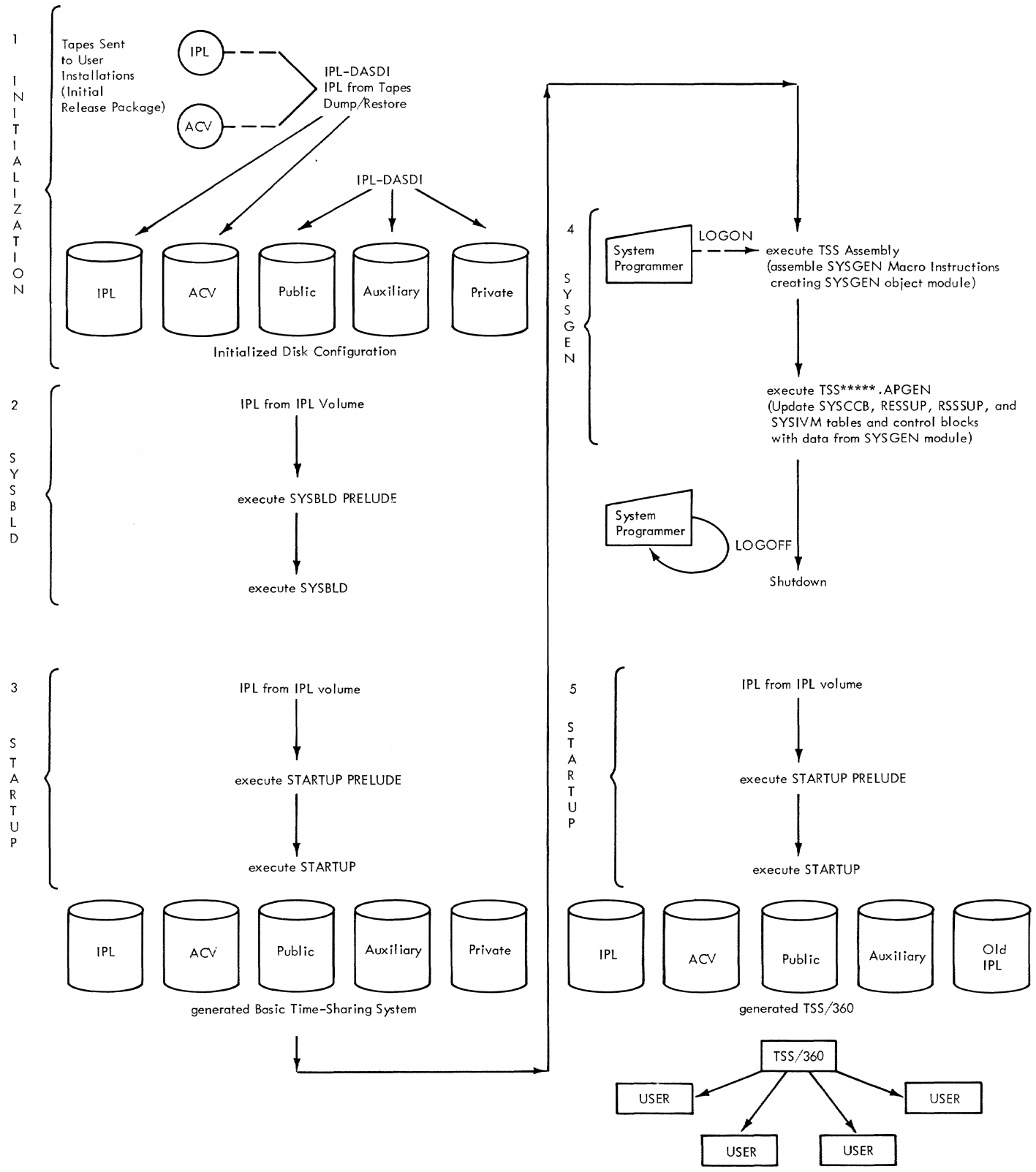


Figure 1. Program Flow of Typical 2311 System Generation Process

tables (CHBAHD, CHBHED, CHBAVE), the System Common table (CHBSCM) and the Configuration Control Block (CHBCCB). Next are created: a system user table (SYSUSE), a system catalog (SYSCAT) and a user library (USERLIB for SYSOPER0, SYSMANGR and TSS*****). Finally the SYSBLD Prelude module is changed to Startup Prelude and written back on disk. SYSBLD finishes with an operator message telling the operator to reload from the IPL volume.

SYSBLD is described in detail in Section 2. See Chart AA, Page 2.

STARTUP MODULE

The Startup module (CEIAA) operates outside the TSS/360 environment. It is executed under the control of a main-line routine. This routine makes use of several independent routines, each with a specific function. Startup generates a basic time-sharing system, a full-fledged system, or is used to restart the system after encountering an error from which recovery cannot be made. Only after Startup has completed its processing can the user work within the structure of a time-sharing system environment.

Startup initializes the system by setting up tables such as the Shared Data Set table (SDST), the Core Block table (CBT), and the Extended Task Status Index (XTSI). It constructs a Task Dictionary (TDY) for the dynamic loader; prints the storage map for Initial Virtual storage (SYSIVM), the Resident Supervisor (RESSUP) and the Resident Support System (RSSSUP); and link-loads the Resident Supervisor and Initial Virtual storage. It creates a set of special device paths, sets protection keys and starts other CPUs in a multi-CPU environment. It also creates the Main Operator Task (MOT).

Once the initialization is complete, the system is given control as Startup exits to the Queue Scanner. At this point an operable time-sharing environment exists.

Startup is described in detail in Section 3.

SYSGEN PHASE

The SYSGEN.MODULE data set is generated under a time-sharing system when system generation macro instructions are assembled under the TSS/360 Assembler. The resulting SYSGEN.MODULE data set is composed of system tables and control blocks, and contains configuration and installation data.

These SYSGEN-produced tables are then incorporated in the existing time-sharing system to initialize a tailored TSS/360. This is accomplished by applying SYSGEN-produced control sections to the existing system tables via the command language and linkage editor control statements contained in the TSS*****.APGEN command procedure. After updating the system tables in place, the TSS/360 must be shut down, so that a Startup sequence from the IPL volume can be initiated in order to use the TSS/360 operating system.

The SYSGEN phase is described in Section 4.

SECTION 2: SYSBLD PHASE

SYSBLD PRELUDE (CEIAP)

Entry to the system build and generation process is accomplished by loading the IPL volume disk, which reads the SYSBLD Prelude routine. Aside from normal functions, SYSBLD Prelude allocates space dynamically for a fifty-eight page buffer area to be utilized by SYSBLD. The SYSBLD Prelude reads the SYSBLD data set off the disk, and branches to the first instruction, thus giving it control. A general register points to a parameter list consisting of the following adcons:

Word 0: address of the data set name to be located.
Word 1: address of the data set name in error message 1.
Word 2: address of the data set name in error message 2.
Word 3: address of the field containing operator's terminal address.
Word 4: address of byte switch to be zeroed.

These fields are changed during SYSBLD execution to produce the Startup Prelude on the same IPL volume. This module is described in greater detail under the heading "Startup Prelude" (Chart AB).

SYSBLD (CEIFA)

Chart AA

SYSBLD, one stand-alone module, consists of several routines that pass control among themselves. The initializing routine sets the base register for the program, and control is passed from one routine to another through the use of adcons. A general register always points to the common save area of SYSBLD, the nineteenth word of which contains the SYSBLD table address. Thus the table is addressable by all the SYSBLD routines. Among these routines are a set of service routines called by the main routines. These do such jobs as I/O, allocating space, and converting external page numbers. Chart AA, Page 2 shows the general flow of control from routine to routine. If any error conditions arise in the service routines, a general register is set with a hexadecimal error code and control is returned to the calling routine. This in turn sends control to the Terminating routine.

SYSBLD operates in main storage, outside the time-sharing environment, as a nonrelocatable, stand-alone module. It is entered through the SYSBLD Prelude.

Input to SYSBLD is made up of the following (see Figure 2):

1. Two disks restored from IBM-supplied tapes.
   a. Initial program load (IPL) VAM-formatted disk with SYSBLD Prelude record.
   b. Auxiliary control volume (ACV) VAM-formatted disk.
2. Information on device type and address as entered by the system operator through his terminal.

The following are tables updated by SYSBLD:

1. Data set TSS*****.RESSUP.G0000V00

   In this data set the addresses of

   five disks (IPL, ACV, public, auxiliary, private)
   operator's terminal
   system programmer terminal
   printer
   card reader

   are placed in the Pathfinding tables

CHBSAC	CHBSCH
CHBDEV	CHBCHL
CHBMCH	

   In addition, the table CHBTDE is updated for device code.

2. Data set TSS*****.RSSSUP.G0000V00

   In this data set the following table is updated:

   CHBECXRB

3. Data set TSS*****.SYSIVM.G0000V00

   In this data set the following tables are updated:

CHBSDA	CHBSCM
CHBAHD	CHBECXVA
CHBAVE	CHBBCT

4. Data set TSS*****.SYSCCB.G0000V00

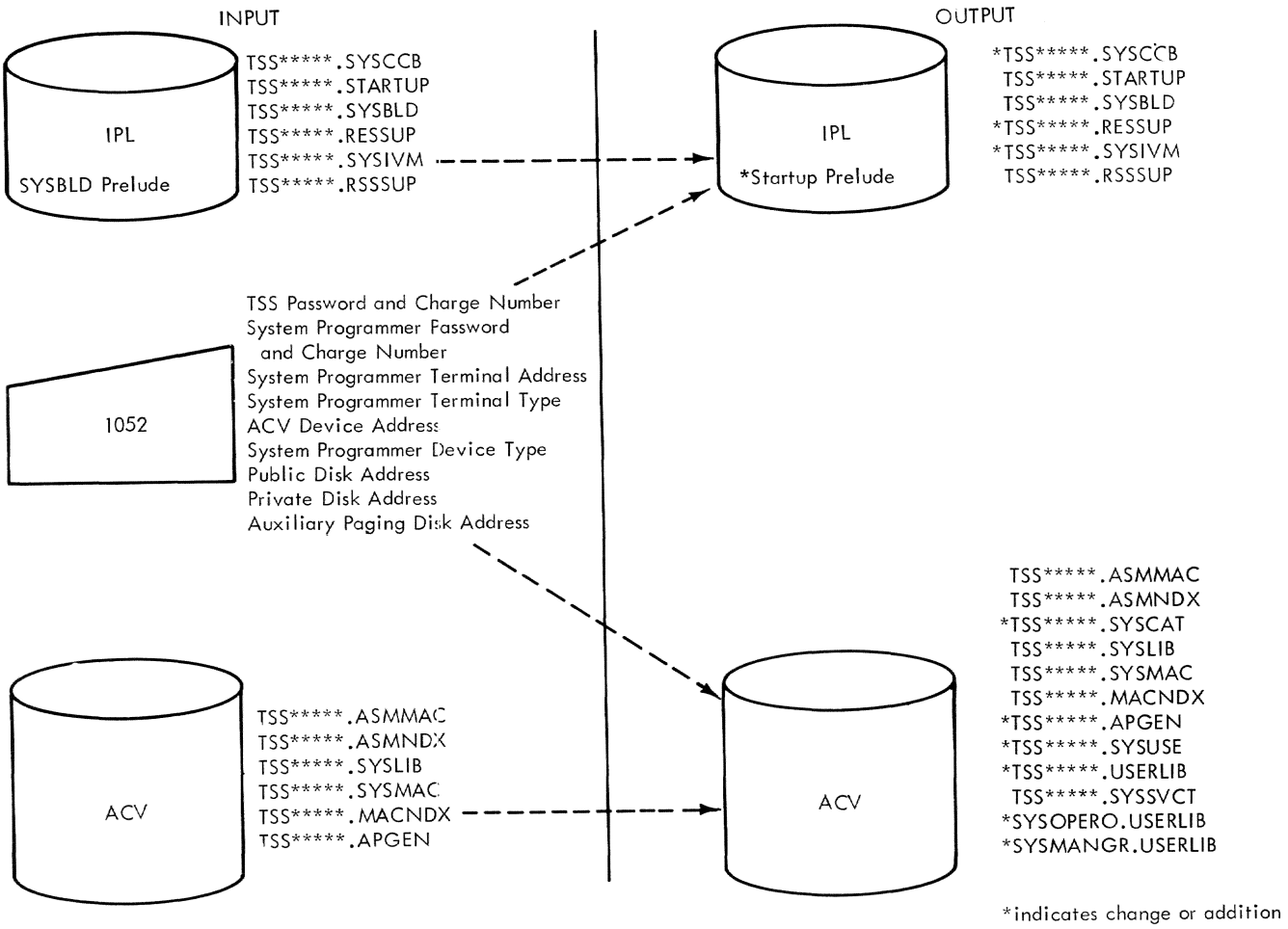


Figure 2. SYSBLD Inputs and Outputs

The Configuration Control Block is filled in as follows:

Drum table with drum path
Transmission Control Path table
with system programmer's terminal
and line type
Printer table with printer path

SYSBLD also creates six new data sets on the ACV volume.

1. A user table with entries for the operator and manager.

2. A user library for user TSS***** containing no entries.

3. A user library for user SYSOPER0 containing no entries.

4. A user library for user SYSMANGR containing no entries.

5. A catalog containing the following data sets for the user TSS*****:

STARTUP	SYSLIB
RESSUP	SYSMAC
SYSIVM	MACNDX
SYSCCB	USERLIB
SYSBLD	APGEN
RSSSUP	ASMMAC
SYSUSE	ASMNDX

and for the user SYSOPER0:

SYSLOG
USERLIB

and for user SYSMANGR:

USERLIB

6. A library for user catalogs (TSS*****.SYSSVCT).

Output from SYSBLD is in the following form (see Figure 2).

Two updated disks:

a. IPL disk with Prelude record changed to read Startup and three data sets modified.

b. ACV disk with six new data sets.

A work area called CHASBD is created by SYSBLD, and is used as the common data area for all steps in the program.

At a successful conclusion of SYSBLD, a message is sent to the operator informing him that the program is complete, and instructing him to load the IPL volume to run Startup. The machine is put in the WAIT state. If an error exit occurs, an appropriate message is produced and the machine is forced into the WAIT state. All the SYSBLD routines are described in detail below.

UPDATE GROUP

Initialization Routine (CEIFA)

Chart AA, Page 2

The purpose of this routine is to create and initialize the SYSBLD table (CHASBD), and make it accessible to all SYSBLD routines. Addressability is set up, base registers assigned, a common save area allocated, and CHASBD addressed in the 19th word of the area. Also at this time, the fifty-eight page buffer (allocated by Prelude for SYSBLD) is zeroed out.

On entry, a general register contains a pointer to the fields in SYSBLD Prelude that must be changed to load Startup. This pointer and the operator's path are stored in the table. The IPL path is extracted from the IPL program status word (PSW - found in location 0 in storage) and saved in the table. The Disk I/O routine (CEIFS) is called to read the IPL volume label and from this, the volume ID and relative page number of the Page Assignment table (PAT) are saved. The disk type and number of pages per volume, determined by examining the volume label, are set in the table.

In the event an error is encountered, a general register is set with an error code, and control is sent to the Terminating routine, (CEIFK). On a normal exit, control is sent to the Convert Paths routine.

Convert Paths Routine (CEIFP)

Chart AA, Page 3

The purpose of this routine is to adjust the operator's terminal path and the IPL path for the checking of operator messages. On entry, the terminal path (stored in the SYSBLD table) is converted from binary to EBCDIC, and is stored back in the table in the form of control unit and device. The IPL path, saved previously by Prelude, is read, converted to EBCDIC in channel, control unit, and device form, and is restored. Control is sent to Interrogate Operator routine.

Interrogate Operator Routine (CEIFB)

Chart AA, Page 3

This routine interrogates the system operator for various system parameters, examines his replies for validity, and rejects any erroneous responses. Upon entry, the user's terminal path is in a parameter register. At this point the system is running in the supervisor state, external interruptions are disabled, and the PSWs are set to handle machine check and program interruptions.

This routine moves a message to the terminal output area, selects the write-to-operator with reply option, and calls the Communicate With Operator routine (CEIFR). Once control is returned, the terminal output area is examined for a valid reply. If acceptable, the reply is stored as an entry in the SYSBLD table. Then the next message is moved in, and the process continues until all parameters are read in. The routine requests the operator's and manager's passwords and charge numbers, the system programmer's terminal address and feature (dedicated versus dial-in), the ACV, private, public, and auxiliary device addresses, the printer path, and the card reader path.

If an erroneous reply is encountered, an appropriate diagnostic message is sent out on the terminal with the write-to-operator option selected. Control is then sent back to re-execute the original request, and the new reply is received or an I/O error occurs. If an error occurs, control is sent to CEIFK. On a normal exit control goes to the Complete SYSBLD Table routine.

Complete SYSBLD Table Routine (CEIFC)

Chart AA, Page 4

This is a housekeeping routine to complete the SYSBLD table, and get the ACV volume ID and relative page number of the PAT. This routine also reads the IPL and ACV PATs into a buffer following the SYSBLD table. The following table entries, having been obtained by CEIFB, are now converted from EBCDIC to hexadecimal notation, and restored in CHASBD:

System Programmer's terminal control unit
Common disk channel

Common disk control unit
ACV device
Private, public, auxiliary paging disk devices
Drum channel
Drum control unit
Drum device
IPL device
Printer and card reader control unit
Printer device
Card reader device

plus the paths created to the:

ACV device
Private, public, auxiliary paging disks
Printer

Next, the ACV path is set in a parameter register. The volume label CCHHR is set in a parameter list. The Disk I/O routine (CEIFS) is called with option '00' to read the volume label. On return, the volume ID and the relative page number of the PAT are set in the table. The PATs for the IPL and ACV volumes are read and moved into buffers following CHASBD.

On return from CEIFS, if general register 15 is nonzero, an error has occurred and control goes to the Terminating routine. On a normal exit, control passes to the Adjust DSCBs routine.

Adjust DSCBs Routine (CEIFQ)

Chart AA, Page 4

The purpose of this routine is to insure that every precataloged generation data group has a DSCB with generation 0, version 0. On entry, CEIFQ gets the address of the IPL PAT buffer, scans the PAT for DSCB pages, and reads each DSCB page into an I/O area. It tests every DSCB on the page and changes the format of the generation data group DSCBs from GxxxxVyy to G0000V00. For each page on which modifications were made, CEIFS (DISK I/O) is called to rewrite that page. Once the IPL volume is complete, the same procedure is followed for the ACV volume. At completion, control is sent to the Update Pathfinder routine.

Update Pathfinder Tables Routine (CEIFD)

Chart AA, Page 5

This routine updates the Pathfinder tables to show the actual device paths to the devices listed. Specifically, the following ten devices are entered in the tables in this order:

Drum
IPL disk
ACV disk
Private, public, auxiliary paging disks
Operator's terminal
System programmer's terminal
Printer
Card reader

It is assumed that each table, or group of tables, will be in a separate CSECT, and no CSECT will exceed a page in length. First the Find DSCB routine (CEIFDS) is called to find and read the format-E DSCB for data set TSS*****.RESSUP.G0000V00. On return, the number of data set and directory pages is stored from the DSCB. The Create List of Pages routine (CEIFV) is called to prepare a list, arranged in virtual page order, of all the external pages containing the data set.

Using the number of directory pages as a count, starting at the beginning of the list, each page in turn is selected and the Page Convert routine (CEIFL) translates it to CCHHR. Then, again using CEIFS, the record is read and moved 256 bytes at a time into the POD/PMD area. The procedure is repeated until the page count is zero. The search routine for the POD, Locate Descriptor in POD (CEIFU), is called to locate the module containing the Pathfinder tables by using the name of one of the tables.

The relative page number field, located in the first page of the member descriptor module (POMFP), is used as an index to the list of external pages in order to select the corresponding external page number. Using CEIFL, the number is converted. CEIFS is called to read the record containing the PMD, and this record is moved to the POD/PMD area.

A field in the header is examined to determine the size of the PMD. If it is longer than one page, successive external page numbers are selected from the list. Again the number is converted, the record read and moved contiguously into the POD/PMD area, until the entire PMD has been read and set up.

Next the Locate DEF routine (CEIFT) is called, and the CSECT containing the Symbolic to Actual Conversion table (CHBSAC) is located. The relative page number of the CSECT (output of CEIFT) is used as an offset, from the start of the member in the list of external pages, to pick up the external page number of the CSECT. The number is converted, and the Disk I/O routine reads the record. CHBSAC is set up for the previously listed devices.

SYSBLD sets the device number (last 4 bits of the path) in the device address field for each entry. The record containing the table is written.

Again the CEIFT routine is called to locate the CSECT containing the device group tables, using CHBD0316, the name of the Drum table. The relative page number of the CSECT is used to locate the external page number containing the table; the number is converted, and the record read. The value of the DEF, a binary number, is used as an offset, from the beginning of the page, to access the correct table. Bits 0-7 of the 13-bit drum path are stored in the first device path byte (DEVPP); bit 0 of the additional path byte (DEVLB) is set to reflect bit 8 of the path. (Bits 9-12 of the path constitute the device number and were stored in CHBSAC. This table also has the pointers to the relevant device group tables set up at SYSGEN time.) The device number (bits 9-12 of the path) is used as an index to select the correct device entry in the table; the symbolic device address, 1 for the drum, is set up, and the available flag set off. Since only one drum type is possible, every device entry is set up during SYSGEN with this device type.

The Locate DEF routine is called to find the location of the Disk Device Group table, CHBD216, and the value, a binary number, is used as an offset from the beginning of the page to access the correct table. The DEVPP field is set with bits 0-7 of the disk path and bit 0 of DEVLB set to reflect bit 8 of the path. Bits 9-12 of the path to each disk will be different, since all five disks are separate devices on the same channel and control unit. For each device in turn, bits 9-12, the device numbers, are used to select the correct device entry. The device type field is set to reflect the disk type -- '01' for 2311 and '04' for 2314 -- and the symbolic device address is inserted. This is X'09' for the IPL, X'0A' for the ACV, X'0B' for the private disk, X'0C' for the public disk, and X'0D' for the auxiliary disk.

Locate DEF is called to find CHB0101, the Operator's Terminal Device Group table. The DEVPP, and bit 0 of the DEVLB, are set. The device number of the operator's terminal is used to locate the correct device entry, the flag set off and the symbolic device address, X'0E' for the operator, is set up. Again, the operator must use a 1052 Model 7 so the device type can be set up for every entry.

The System Programmer's Terminal Device Group table is located (CHB0111), and DEVPP and DEVLB are set. The correct device entry is selected, the flag set off, and the symbolic device, X'0F' for the system programmer's terminal, inserted. The same procedure is followed for the printer and card reader. CHBD0114 is located and the flags set to X'10' and X'11' respectively. The entire page is then rewritten.

The subroutine CEIFT is called to find the CSECT containing the terminal device table, CHBTDE, and read it into main storage. The administrator's terminal entry is updated to contain the device code which was determined from the input to SYSBLD. The updated table is rewritten to disk.

CEIFT is called to find the CSECT containing the Channel table, CHBCHL. The relative page number of the start of the CSECT is again used to select the correct external page number; the number is converted and the record read. The table is constructed by SYSGEN with the channel 0 pointer set up to point to the control units assigned to the multiplexer channel, and dummy pointers set up for the selector channels. The size of the control units assigned to the multiplexer table is set up in the flag area corresponding to channel 0, and the flag set off. The disk channel number, bits 0-4 of the path, is used to select the correct byte pair for the disk channel; the flag is set off, and the selector channel size inserted. Similarly, the byte pair corresponding to the drum channel number is selected, the flag set off, the table size inserted, and the record rewritten.

SYSGEN Minimum Tables creates a CSECT CHBSCH. CHBS2 and CHBS3 are entry points at relative bytes zero and sixty-four respectively. Hence the displacements for the disk and drum channels are known. These displacements are inserted in the channel table for the actual channel address of the disks and drum.

Locate DEF is called to locate the CSECT containing the control units assigned to the Multiplexer Channel table, CHBMCH; the page number is selected and converted, and the record read. The operator's terminal control unit, bits 5-8 of the path, is used as an offset to select the correct entry, and the flag set off. The displacement field is saved. The system programmer's terminal control unit is similarly used to select the correct entry and the flag set off. The displacement in this entry is set to one more than that of the operator's terminal. Similarly, the printer displace-

ment is set to one more than that of the system programmer's terminal. The record is then rewritten.

The Control Unit table, using relevant entries to which these displacements refer, is assembled by SYSGEN, with two contiguous entries for the multiplexer units, the system programmer's terminal control unit being second. CEIFT is called to find the CSECT containing the two control units assigned to selector channel tables, using CHBS3, the Disk table. The page number is selected and converted, and the record read. The value of the DEF, CHBS2, is used to locate the correct table in the page. The value of the disk control unit is used to select the correct entry in the table and the flag set off. This table is assembled with all displacements pointing to the disk control unit in the Control Unit table. CEIFT is called to find the value of CHBS2, the entry point of the control units assigned to the Drum Selector Channel table, and the value used to select the correct table in the page. The value of the drum control unit is used to select the correct entry in the table and the flag, bit 1 of the first byte, set off. Again all displacements are set up by SYSGEN. The record is rewritten, and control passed to the next routine.

If any of the subroutines called detects an error condition, it sets an error code in a register. The CEIFD routine inspects the register on return from every subroutine called. If an error code is found, processing is stopped and control passed to the Terminate routine. Unless the error occurred when writing on the disk, processing can be restarted from the beginning of SYSBLD regardless of where in the routine the error occurred.

At completion, control is passed to the RSS Update Device Tables routine (CEIFRS1), which modifies the Update Pathfinder Tables routine (CEIFD) and branches back to the beginning of CEIFD. Upon completion of this modified routine (CEIFD) control is passed to the VSS Update Device Tables routine (CEIFRS2) which again modifies the Update Pathfinder Tables routine (CEIFD) and again branches back to the beginning of CEIFD (see details of RSS/VSS in the section under Service Routines). Control then passes to the Update Virtual Memory Tables routine.

In addition to the above Update Pathfinder routines, the following steps are executed if the operator's console (1050-7) has the same control unit address as the 2821 control unit.

- A flag is set in the Device Group table, CHBDEV, to allow asynchronous interruptions for the operator's console device (1050-7).
- The "dummy entry" in the Control Unit table, CHBCUT, for the 2821 is partitioned out.
- Entries for the card reader and printer in the Device Group table formerly on the "dummy" 2821 are now located along with the entries for the operator's console on that particular 2821.
- The "displacement" for the printer in the Multiplexer Channel table, CHBMCH, is now set to equal that of the operator's terminal control unit.

Update Virtual Memory Tables Routine (CEIFG)

Chart AA, Page 6

This routine updates information in the following tables: Symbolic Device Allocation table (CHBSDA), Availability table and its header (CHBAVE and CHBAHD), and System Common (CHBSCM).

Upon entry, general register 0 contains the IPL path. The Find DSCB routine (CEIFDS) is called to read the DSCB pertaining to the TSS*****.SYSIVM.G0000V00 data set. This DSCB is moved into the beginning of the I/O area, and the number of directory (POD) pages is saved. A list of external pages for the data set is created, by calling the Create List of Pages routine (CEIFV).

For each table to be processed, the POD page addresses are converted (by CEIFL), read, and moved to the POD/PMD area. The POD is searched for the member to be processed. The first page of PMD is read to determine the PMD length. The remaining PMD pages are then read and moved to the POD/PMD area. The PMD is searched for the definition of the table, and the CSECT is located and read into the common I/O area.

The tables are updated in the order CHBSDA, CHBAHD, CHBAVE, and CHBSCM. After each update, the table is rewritten on the IPL disk and the next table processed.

Control is sent to the Terminating routine in case of an error return. Normally, control is sent to the Update Configuration Control Block routine.

A description of the table updating follows:

CHBSDA: The five disk entries for SDA 9, A, B, C, D are set to the proper device code, 2311 or 2314. The system program-

mer's terminal entry (SDA E) device code is set for the correct device type.

CHBAHD: The device code for the five disks is set to either 2311 or 2314.

CHBAVE: The device code for each of the five disk entries is set to 2311 or 2314.

CHBSCM: The default for disks is set to 2311 or 2314. The count of disks of the type being used is set to five and the count field for the type not being used is set to zero.

Update Configuration Control Block Routine (CEIFI)

Chart AA, Page 7

The purpose of this routine is to insert the actual disk and drum addresses and terminal data in the relevant parts of the Configuration Control Block. On entry, the IPL path is set in general register 0. The IPL volume is searched (via CEIFDS) for the DSCB pertaining to the data set TSS*****. SYSCCB.G0000V00. This DSCB is then moved into the beginning of the I/O area and the number of directory pages is saved. A list of external pages for the data set is created by CEIFV. As in previous routines, the POD page or pages are converted, read, and moved to the POD/PMD area. Next the POD is searched for member CHBCCB. The first page is read to determine the length of the PMD. The remaining pages are read and moved to the POD/PMD area. Now the definition is located, the CSECT found and read into a common I/O area.

The drum path is set in the Drum Path table. The Transmission Control Path table is updated for the administrator terminal. The path is inserted and the type and SAD ORDER set from information stored in the SYSBLD table. The printer path is set in its table. The Channel Control Unit table is set for two selector channels and the page is rewritten.

Normal control is sent to the Create User table routine. Any errors encountered are resolved by the Terminating routine, to which control is given with appropriate codes set.

CREATE NEW DATA SET GROUP

The following four routines are run, one after another, to create five data sets needed by system generation. The User table, three user libraries, the User Catalogs Control Block (SYSSVCT), and a scratch catalog (SYSCAT), containing members for SYSOPER0, TSS***** and SYSMANGR are set up. Addressability is provided for these stored tables, and in each routine a general register contains the ACV path.

Create User Table Routine (CEIFE)

Chart AA, Page 8

This routine inserts the passwords and charge numbers for the system manager and system programmer, with the user identifications, in a stored User table. The routine also writes out the table and constructs a DSCB for the data set. The passwords, stored in the SYSBLD table, are inserted in the User table at their respective entries. The operator's entries are prestored. The Assign External Space routine (CEIFW) is called to allocate 14 pages of external space for the catalog, User table, SYSSVCT (User Catalogs Control Block), and three user libraries. The fifth page number is converted, and the User table is written on the disk. The thirteenth page, SYSSVCT, and the fourteenth page, the SYSSVCT VISAM directory, are converted and written on the disk.

The eighth page number is converted, and a VISAM directory page is written on the disk. The directory page has all zeros, with the exception of the first fullword, which contains a 1.

Next, the physical limits are set up in the stored data set DSCB, the address of a zero DSCB is obtained and the User table DSCB is written in that area. Normal exit is to the Create Catalog routine.

Create User Library (CEIFZA)

Chart AA, Page 8

This routine uses the sixth and seventh pages obtained previously from CEIFW, and creates a DSCB for TSS*****.USERLIB from them. These pages are flagged as assigned, but are not in use. CEIFZA also creates a DSCB for SYSOPER0.USERLIB (using the tenth and eleventh pages) and a DSCB for SYSMANGR.USERLIB (using the ninth and twelfth pages). The DSCBs are then written on the ACV. Control goes to the Create Catalog routine.

Create Catalog Routine (CEIFF)

Chart AA, Page 8

This routine updates the volume ID fields of the ACV and IPL volumes in the previously assembled catalog. The volume ID field for the IPL volume is initialized with the volume ID of the IPL pack. The volume ID field for the ACV is zeroed out.

The address of the IPL or ACV volume is stored in register 0 and CEIFECAT is called

to develop the address of the DSCB for that data set. CEIFECAT stores the address of the DSCB in a field CCCDPT in the Catalog SBLOCK. The DSCB address is in the form SOOOPPPP. S is the slot number, OOO is the relative volume number of the ACV, and PPPP is the relative page number on which the DSCB is located.

Four external pages that were obtained in the Create User table routine are used for the catalog. The POD is written on page one. The catalog entries for SYS-OPER0, TSS*****, and SYSMANGR are written on the second, third, and fourth pages respectively. The DSCB is then written on the ACV.

Normal exit goes to the Update Catalog JFCB routine. As in other routines, abnormal exit goes through the Terminating routine.

Update Catalog JFCB Routine (CEIFTD)

Chart AA, Page 9

This routine places a pointer to the Catalog format-E DSCB in the catalog Job File Control Block (JFCB).

First, the Find DSCB routine is called to read the format-E DSCB for TSS*****. SYSIVM.G0000V00. On return the number of data set and directory pages are stored from the DSCB. The Create List of Pages routine (CEIFV) is called to prepare a list of all the external pages containing the data set (in virtual page order).

Using the number of directory pages as a count, starting at the beginning of the list, each page in turn is selected and the Page Convert routine (CEIFL) translates it to CCHHR. Then, again using CEIFS, the record is read and moved 256 bytes at a time into the POD/PMD area. The procedure is repeated until the page count is zero. The search routine for the POD, Locate Descriptor in POD (CEIFU), is called to locate the module containing the TDT by using the CSECT name CHBTDT.

The relative page number field, located in the first page of the member descriptor module (POMFP), is used as an index to the list of external pages in order to select the corresponding external page number. Using CEIFL, the number is converted. CEIFS is called to read the record containing the PMD, and this record is moved to the POD/PMD area.

A field in the header is examined to determine the size of the PMD. If it is longer than one page, successive external page numbers are selected from the list. Again the number is converted, the record read and moved contiguously into the POD/PMD area, until the entire PMD has been read and set up.

Next the Locate DEF routine (CEIFT) is called, and the CSECT CHBTDT is located. The relative page number of the CSECT (output of CEIFT) is used as an offset, from the start of the member in the list of external pages, to pick up the external page number of the CSECT. The number is converted, and the Disk I/O routine reads the record.

The pointer to the last JFCB is obtained. CEIFTD chains backwards through the JFCBs until the Catalog JFCB is encountered. When the Catalog JFCB is found, the slot number and page number of its format-E DSCB are moved from the SYSBLD table into the TDTDSC field in the JFCB. Chaining continues through the last DSCB.

CEIFS is called to rewrite the TDT page. Control is passed to the Terminating routine if any called subroutine returns an error code in general register 15. Otherwise, control is passed to the Set Up Volumes for Startup routine (CEIFJ).

COMPLETION GROUP

Set Up Volumes for Startup Routine (CEIFJ)

Chart AA, Page 10

The purpose of this routine is to rewrite the PAT, and modify the SYSBLD Prelude into a Startup Prelude. The two PATs, one for the IPL volume and one for the ACV volume, are located in main storage following the SYSBLD table. They are moved to the disk I/O area and CEIFS is called (with the path and parameter area set up) to write them onto the ACV. If the disk routine returns a nonzero register, the Terminating routine is called. Otherwise, the Prelude record is read from the IPL volume. The displacements of the fields to be changed (previously stored in the SYSBLD table) are obtained, and the contents changed. A parameter area is set up and the disk routine called to write the new Prelude over the SYSBLD Prelude on the IPL volume. Control is then passed to the Terminating routine with a normal completion code.

Terminating Routine (CEIFK)

Chart AA, Page 10

This is the routine that handles the termination of SYSBLD. It interprets the error codes sent in general register 15, and sends out a proper message. Following is a list of codes and their meanings:

'00' error free
'F0' machine check interruption
'F4' program interruption
'80' terminal I/O error
'40' disk I/O error
  '41' read
  '48' write
  '43' format-E not found
'20' data processing error
  '21' data set not found
  '22' member not in POD
  '23' list of external pages incomplete
  '24' invalid channel for IPL
  '25' erroneous path (not IPL or AUX)
'10' table is already present
  '11' user table
  '12' catalog
  '14' userlib
'08' restore disks before restarting

On a terminal I/O error, the message is suppressed. Otherwise, a correct message is moved into the message write area, and the Message Write routine (CEIFR) is called to write the message on the terminal. On return from writing the error message, the machine is put in the WAIT state, with the EBCDIC for 'END' in the display register (d-register). If this was code '00', SYSBLD is now complete. If not, SYSBLD must be rerun.

SERVICE GROUP

Locate DEF Entry Routine (CEIFT)

Chart AA, Page 12

This routine is called to locate the value of a DEF, the start of the CSD containing the DEF, and the page number in text of page 0 of the CSECT. All DEFs called for must be simple, relocatable DEFs. When the routine is entered, general register 0 contains a pointer to a parameter list. The PMD is in main storage. Word 0 of the parameter list contains the address of the PMD, and word 1 the address of an eight-byte area containing the name to be located. The machine is in the supervisor state, and running with all interruptions disabled.

In order to locate the first CSD, the routine will add the length of the PMD header field to the PMD start address. Then, to locate the first DEF, it will add the CSD header length to the start address of the CSD. Next, every relocatable DEF name will be compared to the eight-byte field in the parameter list until a match is found. If the available DEF supply is exhausted, the next CSD is located. This is accomplished by adding the length of the first to the start address of the first, in order to address the second and so on until a match is found. When the match is found, the address of the CSD start location, the value of the matching DEF, and the page number in text of page 0 of the CSECT containing this DEF, are returned in the parameter list in words 2, 3, and 4 respectively. General register 15 will be set to zero signifying a normal completion.

If no matching DEF can be found, and the supply of available CSDs is exhausted, general register 15 is set to X'21' before a return is made to the caller.

Page Conversion Routine (CEIFL)

Chart AA, Page 12

This is the routine called to convert an external page number to the actual cylinder, head, and record (CCHHR) address of the record. On entry to the routine, general register 0 contains either a X'01' (for disk conversion 2311) or a X'04' (for disk conversion 2314). Parameter registers contain the page number to be converted and the address of the 64-word save area.

The code type is determined, and control is sent to one of the following two routines:

- For type 2311 disks, the page number is compared with the possible upper and lower bounds to determine its validity. If the page number is valid, the cylinder is obtained by dividing the number by 8. The remainder is then used as an argument in the table look-up to determine track and record number. The full address, including the binary number (which is zero in this case), is constructed in the return registers. The remaining registers are restored, general register 15 is zeroed and control is returned to the caller.

- For type 2314 disks, the page number is also examined for validity. If valid, the page number is divided by 32, to determine the cylinder number. The remainder is divided by 4, to determine the relevant quarter cylinder required. This is then multiplied by 5, to obtain the initial track address of the relevant quarter. The second remainder is then used as the argument to determine track and record value within the quarter cylinder. The value is added to the initial track address to obtain the record address. Returning is accomplished as described above.

If an invalid page number is sent as a parameter, an error code is set in register 15 and control is sent to the calling routine. No further processing takes place.

The error codes are X'A1' for 2311 disk error, X'A2' for 2314 disk error.

Create List of Pages Routine (CEIFV)

Chart AA, Page 11

The purpose of this routine is to create a list of external pages for a data set, ordered by virtual page number.

Prior to entry, general register 0 is set with the path of the device on which the data set resides. Another parameter points to the location where the list is to be constructed. The format-E DSCB will already have been read, and will be in the beginning of the common I/O area.

The count of total pages assigned to the given data set is taken from the format-E DSCB. A field is then primed with the maximum number of possible page entries that can be contained in the DSCB (38 for a format-E DSCB). External page numbers are moved one at a time from the DSCB to the list area. Each time a page is stored, the count of possible entries and the count of total pages assigned are decremented by one. Processing continues until either count becomes zero. If the count of possible entries in that DSCB becomes zero before the count of pages assigned, the chain field of the DSCB is checked and the continuation DSCB is read via CEIFS and moved to the beginning of the I/O area. The maximum count of entries field is primed to 62 for this format-F DSCB and processing continues. The list is complete when the count of total pages assigned becomes zero.

If the continuation DSCB cannot be read, general register 15 is returned from CEIFS with a nonzero value and this code is passed back to the calling routine. Otherwise, a zero code is returned.

Locate Descriptor in POD Routine (CEIFU)

Chart AA, Page 13

The purpose of this routine is to locate the member descriptor in the POD, either by name or by alias.

Prior to entry to this routine, the entire POD will have been read into main storage contiguously. Word 0 of the parameter area contains the address of the POD and word 2 contains the member name address (8 bytes with trailing blanks).

Using the Hash algorithm, as described under Startup (see Section 3), this routine hashes the member name. The hash value is used as an offset to the hash table to pick up the first link in the chain. If either a zero entry or a zero chain is encountered before the member descriptor is located, an error return is made with the code set to X'22'. The chain of names with the same hash value is followed and each name is in turn compared with the given name until a match is found. If the match is made with an alias, the member descriptor pointer is followed to give the member descriptor. The relative page number of page 0 of the member, and the number of bytes in the last page, are returned in word 1 of the parameter area. General register 15 is zeroed, and control is returned to the caller.

Assign External Space Routine (CEIFW)

Chart AA, Page 11

The purpose of this routine is to assign pages of external space from a volume, and mark those pages as assigned in the corresponding PAT. Prior to entry, general register 0 is loaded with the 13-bit device address of the disk from which space is to be assigned. General register 1 contains the address of a word that contains the number of pages requested followed by a word list area in which to return the external page numbers assigned. (Since the number of pages requested can vary, the word list area varies in proportion with the number of pages requested.)

The device address is examined to determine from which device the pages are to be allocated. The corresponding PAT for that device is selected and searched for pages marked available. (The byte in the PAT corresponding to an available page contains X'00'.) For each page found, the external page number is entered in the list area and the byte in the PAT corresponding to that page is changed to X'01', thus indicating that the page is now assigned. When the requested number of pages have been found, register 15 is zeroed and control is returned to the caller. If the PAT is exhausted before the requested number of pages have been found, register 15 is set to X'08' and control is returned to the caller.

Communicate With Operator Routine (CEIFR)

Chart AA, Page 15

This routine serves two purposes. It either sends a message to the operator with a no-reply condition, or sends a message to the operator and waits for a reply. Since there are common message-write and message-read areas, it is the responsibility of the calling routine to set up the message-write area, and clear out any message reply that may already be there. On entry to this routine, the message area must contain the length of the message in the first word,

followed by the message itself. General register 0 contains a 13-bit terminal device address (right adjusted), and general register 1 indicates whether a reply is expected (X'00' = no reply expected, X'04' = reply expected).

The routine relocates the addresses in the CCW list and the address of the PSW to be used when it is first entered. A switch is set to inhibit relocation on subsequent entries. The indicator in general register 1 is used to set the command chaining flag in the first CCW (the write CCW), when a reply is expected. The current new I/O PSW is stored, and the routine sets its own new I/O PSW. The CAW is set with the address of the CCW list, and a Start I/O issued. If the 'busy' condition is present, the routine reissues Start I/O until the condition is cleared. If the 'CSW stored' condition is present, the routine goes to test whether the I/O is complete and successful. The 'not operational' condition causes the routine to set an error return code and return to the calling routine. When the 'available' condition is present, the machine is forced into the WAIT state, with only the multiplexor channel enabled, until an I/O interruption is received. A TEST I/O is initiated and reissued if the 'busy' condition is present. Otherwise, the new I/O PSW is set to its entry condition, and the CSW status bits inspected. The presence of any of the following conditions results in the error check being taken:

program check
protection check
channel data check
channel control check
interface control check
chaining check

When the 'unit exception' occurs during the read, the CAW is set with the address of the second CCW in the list (the read CCW), and the routine loops back to reissue the Start I/O. The presence of a unit check results in a sense command being issued to the terminal. If the check was caused by intervention being required, the alarm bell is rung, and the routine loops back to reissue the original Start I/O. In any other case an error action is taken.

The routine loops and tests I/O until the operation is completed successfully, and the 'device end' condition is present.

Disk I/O Routine (CEIFS)

Chart AA, Page 15

The purpose of this routine is to do disk I/O in one of the three following ways:

to read a record, given its CCHHR

to write a record to its CCHHR

to read a record after a search on key, and provide CCHHR

On entry to the routine, the length of the record in question must be in the first word of common disk I/O area. If the record is to be written out, it must immediately follow the length field. In the case of a read, the record will be read into this area. For the third option, the 44-character key must be moved into this area on entry; then when the record is located, the data portion is read to follow the key, and the record ID is read into a store area and subsequently moved to the input parameter area. General register 0 contains the 13-bit device address, right adjusted, and general register 1 has the address of a 6-byte parameter area (aligned on a word boundary), which contains the 5-byte record ID and a 1-byte code. This byte contains X'00', X'04', or X'08' for options 1, 2, and 3 respectively. If the option is 3, the record ID field contains the cylinder and track from which to begin searching.

The routine now relocates the addresses in the CCW list and the address of the PSWs to be used when the routine is first entered. A switch is set to inhibit relocation on subsequent entries. On option 3, the CAW is set to the address of the CCW that searches for the record ID. Otherwise, the CAW is set to the address of the CCW list which searches for the record ID and reads or writes the located record.

A Start I/O is then issued and the condition code inspected. If the device is busy, the routine loops on Start I/O until the device is free. On condition code 1, the routine goes to test the CSW. On condition code 3, general register 15 is set to X'43' and control goes to the caller. Once successful, the routine goes into the WAIT state until an I/O interruption is received. TEST I/O is issued and the condition code examined. If the condition code is 2, the routine loops on the TEST I/O, otherwise the CSW is inspected. If the 'device end' is not on, the routine branches to TEST I/O until the device is free. If any of the following occur, an error action is taken, and the error code set to X'40':

program check
protection check
channel data check

interface control check
chaining check

The routine attempts the I/O operation 10 times before declaring a hardware error on the following conditions and setting the corresponding error codes:

record not found	X'40'
read error	X'41'
write error	X'48'
nonoperational	X'43'

Otherwise,a normal return is executed.

Find Format-E DSCB Routine (CEIFDS)

Chart AA, Page 16

The purpose of this subroutine is to locate and read a format-E DSCB or to return the location of an available DSCB to the calling routine. If the name of the data set to be located consists of all zeros, the location of an available DSCB is returned.

On entry, general register 0 contains the 13-bit device address of the device to be searched, and general register 1 contains the address of a 12- or (optionally) 13-word parameter list. The parameter list consists of one word in which to return the address of the DSCB that is found, followed by a 44-character data set name. The last word, the optional one, is sent only if the calling routine wishes to have returned the relative page number on which the DSCB was found. (If this parameter is requested, the SBDDS1SW field in the SYSBLD table has been set to X'FF' prior to entry.)

The device to be searched is determined by comparing its address with those of the IPL and ACV volumes as stored in the SYSBLD table, and the corresponding PAT table is selected. The PAT for that device is scanned for pages marked as DSCB pages.

If CEIFDS is searching for an empty DSCB, it tests bits six and seven of the PAT entries. Bits six and seven must be zero, indicating available space or the page is skipped. Once an available page is found, Convert Pages (CEIFL) is called to convert the external page number to a CCHHR. The Disk I/O routine (CEIFS) is called to read the page into the common I/O area. The sixteen DSCB slots per page are examined. If CEIFDS is looking for a non-zero data set name, the data set type field of the DSCB must contain X'01' to indicate format-E DSCB and the data set names must match. If looking for an available DSCB, the data set type must be X'00'. If an available DSCB is found on an already existing DSCB page, a check is made to see if it occupies the twelfth slot. If so, the sixth bit of the corresponding PAT entry is set to indicate that no new format-E DSCB should be allocated from this page. If no empty DSCB can be allocated from the existing DSCB pages, an available page is found and the PAT is modified to indicate that this is a new DSCB page. The page is zeroed and the first slot is selected for the requested DSCB.

When the DSCB is found, the address of the DSCB is set in the parameter list. The SBDDS1SW is checked and the external page number parameter is set if requested.

If any subroutine called returns an error code, CEIFDS passes this code back to its calling routine. If an available page cannot be allocated as a new DSCB page, and error code of X'41' is returned to the calling routine in general register 15.

Checksum (CEIFCKS)

Chart AA, Page 13

This routine develops a check sum for a DSCB. On entry, general register 10 contains the address of the DSCB. General register 0 contains a 0 if this routine is to verify an existing check sum or a 1 if a new check sum is to be developed. For the first option, a return code of zero is returned to the caller if the check sum agrees, or X'04' is returned if it does not agree. Control is returned to the caller.

Update Catalog SBLOCK Routine (CEIFECAT)

Chart AA, Page 14

This routine finds and stores the DSCB address for a data set in the Catalog SBLOCK. After chaining backward through the Catalog SBLOCK to determine the fully qualified name for the data set, CEIFECAT calls CEIFDS fo find the DSCB address. The DSCB address is in the form SOOOPPPP, where S is the slot number, OOO is the relative volume for the ACV, and PPPP is the relative page number on which the DSCB is located. If the DSCB address is found, it is stored in the CCCDPT field in the Catalog SBLOCK and control is returned to the caller. If the DSCB address is not found, control is passed to the Terminating routine.

Update Device Tables for RSS and VSS (CEIFRS1, CEIFRS2)

Chart AA, Page 16

In order to find the CSECT containing the device table CHBECXRB utilized by RSS, the Update Device tables routine modifies the Update Pathfinder tables routine (CEIFD). The Update Pathfinder tables rou-

tine is modified to search the TSS*****. RSSSUP data set for the CSECT containing the table CHBECXRB. (The Update Pathfinder Tables routine initially searches TSS*****. RESSUP for its device tables and initializes them.)

Once the CSECT containing the table has been found, two fields are initialized to reflect the following information for each device type:

- hardware device address
- device defining information

Each entry in CHBECXRB consists of three words that contain the following information:

- symbolic device address — 1 halfword
- physical path (hardware address) — 1 halfword
- alternate physical path — 1 halfword
- flag byte — 1 byte
- reserve byte (unused) — 1 byte
- device defining codes — 1 word

After initializing the table, the page containing the CSECT in which the table resides is written back onto the IPL volume.

Next, the Update Pathfinder Tables routine is modified to search the TSS*****. SYSIVM data set for the CSECT containing the table CHBECXVA. CHBECXVA contains device information utilized by VSS (nonresident portion of TSSS). The body of this table is a duplicate of CHBECXRB and is initialized to reflect the same information. Once initialized, the page containing the CSECT in which this table resides is rewritten onto the IPL volume.

STARTUP PRELUDE (CEIAP)

This program performs certain environmental analyses, stores the results in a communication region, loads the Startup module and transfers control to it. Refer to Section 1 for a preliminary discussion of this module (see also Chart AB).

Attributes: This module is nonrelocatable and resides in real storage.

Entries: Prelude is entered by a hardware IPL or a system restart. In the event of a hardware malfunction, Reconfiguration (CGCMA) is invoked to set up a damage report. When finished processing, Reconfiguration simulates a hardware IPL and calls Prelude.

Exits: Normal exit is made by loading Startup, and sending control there. Errors that cause termination force the machine into the WAIT state and set the instruction address counter to all 1's.

Operation: Prelude is entered through a hardware IPL or a system restart. Base registers are set, and the communication region zeroed out (Figure 3). The extended control registers and the address of the IPL volume are saved. If it is a system restart, the Reconfiguration program will have filled a fixed hardware location called COMAR with the characters "RESTART". Prelude need only compare this cell with its own constant "RESTART" to determine how it has been entered. The address of the operator's terminal is also passed to Prelude by Reconfiguration (it will be moved to the communication region at entry). The volume label of the IPL volume is read, extended mode entered, and the system put into the WAIT state, in anticipation of an I/O interruption. Once the interruption is encountered, the interruption code area contains the 13-bit (extended mode) address of the IPL volume. This is moved into the communication region.

If Prelude has been entered via IPL, it enters the WAIT state and waits for an asynchronous interruption from the operator's terminal. Once the interruption occurs, the address of operator's terminal is in the interruption code area and can be moved into the communication region. Prelude then moves the volume serial number and the address of the IPL volume PAT from the label to the communication region.

Prelude now searches for locations into which it can read the Configuration Control Block (CCB), PAT and DSCB pages. These locations must be determined dynamically for two reasons. First, the prefix storage areas (PSAs) of a multi-CPU system are double addressable, and provision is made that the locations do not overlay Prelude. Second, since it is not necessary for the floating storage addresses (FSAs) of the storage elements (SEs) in the system to be assigned contiguously in main storage, not every address may be valid within any particular installation at any given time. Prelude searches for the lowest available main storage locations for these buffers. Their addresses are placed in the communication region.

The PAT is read in as soon as its buffer location is returned. Prelude then searches the DSCB pages on the IPL volume (using the PAT) for the DSCBs for Startup, the CCB, and Quickstart. If only the Quickstart DSCB if found, a message (QKSTART IN PROGRESS) is issued and processing continues. If the Startup and SYSCCB DSCBs are found, but not the Quickstart DSCB, the STARTUP IN PROGRESS message is issued. If both the Startup and Quickstart DSCBs are found, a message (STARTUP? Y OR N) is issued to the operator. Y requests a normal Startup; N requests a Quickstart. If a normal Startup is wanted, SYSCCB must be present. If the correct DSCBs are not found, a message is issued and Prelude terminates.

In a normal Startup, the entire data sets TSS*****.STARTUP.G0000V00 and TSS*****.SYSCCB.G0000V00 are read. In a Quickstart, the first page of TSS*****.QKSTART.DSxxxxxx is read to get the length of Startup and the CCB. The Startup and SYSCCB data sets are read sequentially from the Quickstart data set, beginning with the third external page entry of the data set.

Simplex is the term used to describe a system consisting of a model 1 CPU; there is no prefixing ability and never more than one CPU in the system. Duplex is the term describing a system of model 2 CPUs. All model 2 CPUs have prefixing ability. A time-sharing system with only one model 2 CPU is referred to as half-duplex. This CPU has prefixing abilities and must not be confused with simplex.

The next task is to determine the identity of the initially-loaded CPU. In a duplex or half-duplex system, the CPU's ID is moved into the first byte of each PSA

F3C	COMAR -- Startup or Restart Indicator	
F44	DAMREP -- Damage Report From Reconfiguration (Restart Only)	
F4C	CREGS -- Contents of Extended Control Registers 8-14	
F68	SESIDS -- Byte Map for Storage Elements (SEs)	
F70	SESTSS -- Number of SEs in TSS	CPUSTSS -- Number of CPUs in TSS
F74	OTHCPU -- IDs of Non-IPL CPUs	SCPID -- ID IPL CPU
F78	CCUTSS -- Byte Map CCUs in TSS	
F7C	PARTND -- Byte Map Partitioned CCUs	
F80	CCBLEN -- CCB Length (in pages)	STPL -- Length of Startup Module
F84	TERMAD -- Address of Operator Terminal	Unused
F88	VOLTPC -- Volume Type Code	VOLAD -- IPL Volume Address
F8C	PRINTER -- Printer Address*	IPLID -- Volume Serial
F90	Volume Serial (Continued)	
F94	DSCBLOC -- DSCB Buffer	
F98	PATLOC -- PAT Buffer	
F9C	LOADPT -- Starting Address for Startup	
FA0	CCBLOC -- CCB Load Point	
FA4	IPLPFX -- Active Prefix of IPL CPU	
	*Initialized by Startup.	

Figure 3. Communication Region -- Prelude to Startup

associated with that CPU. This process is repeated for each CPU within the system.

Addressable location 0 now contains the desired information if and only if one of the PSAs has been activated. Location 0 is checked and, if a valid ID is found, it is moved to the communication region and becomes the ID of the CPU in question. If no valid ID is found, no PSA has been activated, and the error message to activate the prefix and re-IPL is issued at the operator's terminal. Prelude then terminates.

In a simplex system, only one CPU exists. Since there is no prefixing, all SEs are assumed to be in the time-sharing system, and control passes to Build a Table of PSAs.

In a duplex or half-duplex system, it is possible that not all of the devices are attached to the time-sharing system. In this case a configuration analysis is made. The configuration is composed of all SEs, channel control units (CCUs) and CPUs in the system. SEs in the system must be connected to the loaded CPU. A CCU must be connected to the initially loaded CPU and all the SEs in the system. A CCU must also be disconnected from those SEs and CPUs not in the system. Other CPUs must have the same SE/CCU hookup as the initially-loaded CPU. Configuration information is set in the communication region: the number of SEs in TSS, byte maps for these SEs and the CCUs in the system, and a byte map for those CCUs not in the system. Due to the three-fold requirements of the CCUs, an erroneous setting of the configuration console switches could project the false impression that no CCUs exist in the system. In this case a message is issued instructing the operator to correct the switches and Prelude terminates.

Prelude next analyzes the prefix activation and the direct control switch settings on the configuration console. For systems with two or more CPUs, these switches must be activated for all CPUs in the system.

For a half-duplex system, prefixing must be activated, and direct control must be deactivated.

Prelude now checks that there are no duplicate FSAs in the system, since the physical addresses are variable. Should a duplicate exist, a message is issued to the operator to correct the configuration console dials and Prelude terminates. Otherwise, processing continues.

Because PSA areas should not be overlaid, a table of PSAs is built for all CPUs in the system.

Now a storage analysis creates a page map giving the condition of every page in main storage: available to the system, partitioned from the system, or failing. Partitioned pages are those not addressable by the system; that is, no FSA gives that address. "Failing" means a machine check interruption was detected while the page was being used. While the page map is being created, the load point for Startup must be determined. It has to be loaded contiguously into the highest available pages without overlaying any PSA. This load point is now moved into the communication region. If no load point can be found, a message is issued at the terminal and Prelude terminates.

Next the 'Resident Supervisor' flag is set in the byte of the page map that corresponds to the active PSA. In simplex mode, the PSA is in page 0 and the page map is set to reflect this. In duplex or half-duplex mode, the address of the PSA is determined dynamically. The addresses of the primary and alternate PSAs are stored in the first fullword of the primary and alternate PSAs, respectively. Locations 0-3 then contain the address of the PSA that is active. This address is moved into the communication region and the byte in the page map is set to reflect this. If the active PSA had previously been detected as failing, a message is issued and Prelude terminates.

Finally, Prelude reads Startup into the location determined during the storage analysis, and the program exits to that location.

Prelude also includes a few short service routines used to perform the work that is usually done by the system but is not yet available. Routines are set up to: recover from program or machine check interruptions; convert external page numbers to CCHHR form; do I/O initialization; and communicate with the operator.

STARTUP (CEIAA)

Startup (Chart AC) is the module that performs the operations necessary for initialization of the time-sharing system. If a hardware failure occurs, any subsequent execution of Startup constitutes a system restart. The Startup module is non-relocatable, operates in the supervisor state, and resides in real storage. It is a single module containing many subroutines, all running under control of a mainline. In order to run, Startup needs the basic minimum machine requirements (see the System Generation and Maintenance SRL). The program assumes SYSBLD has created the configuration-dependent tables to be used or initialized by Startup. All public volumes must be mounted, and the System Catalog data sets (SYSCAT and SYSSVCT) should be on the ACV volume (public volume 0). Storage need not be contiguous nor start at location 0, but the starting address of the highest storage box must be dialed to not exceed 256,000(n-1), where n is the number of boxes present at the installation.

Startup has two primary functions. The first is the CSECT link-loading of the Initial Virtual storage (SYSIVM), Resident Supervisor (RESSUP)and Resident Support System (RSSSUP) data sets. The second is the creation and initialization of various tables.

RTAM INITIALIZATION

RTAM Initialization provides initial values for the tables required by RTAM to log the main operator onto the system. The LOGON processors require such information from the main operator as his user identification, terminal address, and terminal type; since the operator cannot log himself onto the system, Startup must furnish the information. There are four steps in RTAM Initialization:

1. Calculation of the number of Task Core Table pages (CHBTCT) and the number of system buffer pages (CHBBFP); these depend on the number of terminals connected to the system during system generation.

2. Storage allocation (virtual and real storage) for the Task Core Storage Table pages and system buffer pages.

3. Page table initialization (Shared Page Table and External Shared Page Table).

4. Initialization of certain other tables.

The tables modified, initialized, or created for RTAM are CHBTCT, CHBBFP, CHBMTS, CHBDEV, and CHBTDE.

The storage requirements for CHBTCT, CHBBFP, and CHBMTS are:

1. The virtual storage address of the Task Core table must start at the first page of the first public segment of IVM.

2. The first page of the Task Core table and the first page of system buffer pages must reside in real storage at all times. This is ensured by turning on the "page hold" flag in the XSPT entries reflecting the virtual storage addresses assigned to them.

3. CHBMTS must begin on a 64-byte boundary.

LINK-LOADING

During link-loading, each data set contains a CSECT load list module which contains the names of the CSECTs to be link-loaded. The delta data sets are sorted into three groups, those for IVM, for RESSUP, and for RSSSUP. Within these groups, the order in which the delta data sets were entered is maintained. The delta data sets for IVM are processed first and then SYSIVM is processed. After the load list is read in, the PMDs for CSECTs in the load list are linked into the Task Dictionary table (TDY). If user modules are being loaded, all additional user CSECT names (those not present in the load list) are added at this time. After a PMD is moved into the TDY, the definitions within each of the PMDs CSECTs are resolved. (Preceding the PMDs in the TDY are a header, system hash table, user hash table and storage map.) Private CSECTs are loaded into segment 0 of virtual storage, while the public CSECTs are loaded into segment 1. The Interruption Storage area (ISA) is automatically assigned a virtual storage address of zero. After all PMDs are linked into the TDY, the TDY storage map is created. It is made up of two-word entries for each CSECT in the TDY. The first word is the virtual address for the CSECT. The second is the address of the CSECT's CSD. The entries are in a numerically ascending sequence of virtual storage addresses. The CSECTs are read in a page at a time, and packed into the output buffer. They are then written on the paging drum if they are public CSECTs, or written on the paging disk if they are private CSECTs. Once processing is complete, then both the TDY and the Shared Data Set table (SDST) are written on their respective paging volumes.

The RESSUP and RSSSUP delta data sets and CSECTs are loaded in a similar manner to the SYSIVM CSECTs. However the following location differences exist for the RESSUP CSECTs:

- RESSUP CSECTs except for SERR/Reconfiguration reside in real storage. SERR/Reconfiguration CSECTs reside on all drums in the system.

- The TDY, used only in the loading process, is not written on the paging volume.

- User modules cannot be loaded into real storage.

If a Quickstart is in progress, the link-edited routines are read from the Quickstart data set.

INITIALIZATION OF TABLES

Tables are created, initialized, or modified as follows:

For SYSIVM,

- Task Dictionary table (TDY)
- Interruption Storage Area (ISA)
- Buffer Page (BFP)
- Skeletal Extended Task Status Index (XTSI)
- Terminal Control Table (TCT)
- Symbolic Device Allocation table (SDAT)
- Task Data Definition table (TDT)
- Shared Data Set table (SDST)
- Public Volume table (PVT)

For RESSUP,

- Prefix Storage Area (PSA)
- CPU Status table (CST)
- Special Device Path tables
- Auxiliary Storage Allocation table (ASAT)
- Resident Shared Page Index table (RSPI)
- Core Block table (CBT)
- Shared Page table/External Shared Page table (SPT/XSPT)

- System table
- System Activity and Resources table (SAR)
- Multi-terminal Status Control Block (MTS)
- Device Group table (DEV)
- Terminal Device table (TDE)

For RSSSUP,

- Communication table (RSSCOM)
- Page table/External Page table (PT0-PT3, XPT2-XPT4)
- RSS Symbol table (CHBRST)
- Support System Device Allocation table (SSDAT)

Once this is complete, Startup creates a task (TSI) for the Main Operator's Task (MOT). Startup interrogates the 2702 partitioning lines and enables those lines which are not partitioned out of the configuration. In addition, the malfunction alert fields in control register 6 are enabled for duplex and masked for half-duplex operations.

INPUT TO STARTUP

The four major sources of input to Startup (Figure 4) are:

- The IPL volume
- The Delta volume
- The Operator's 1050 terminal
- The Card Reader

The SYSCCB, SYSIVM, RESSUP (VPAM formatted) and RSSSUP data sets reside on the IPL volume.

The delta volume contains data sets that may modify the SYSIVM, RESSUP, and RSSSUP data sets which reside on the IPL volume. This provides the optional capability for

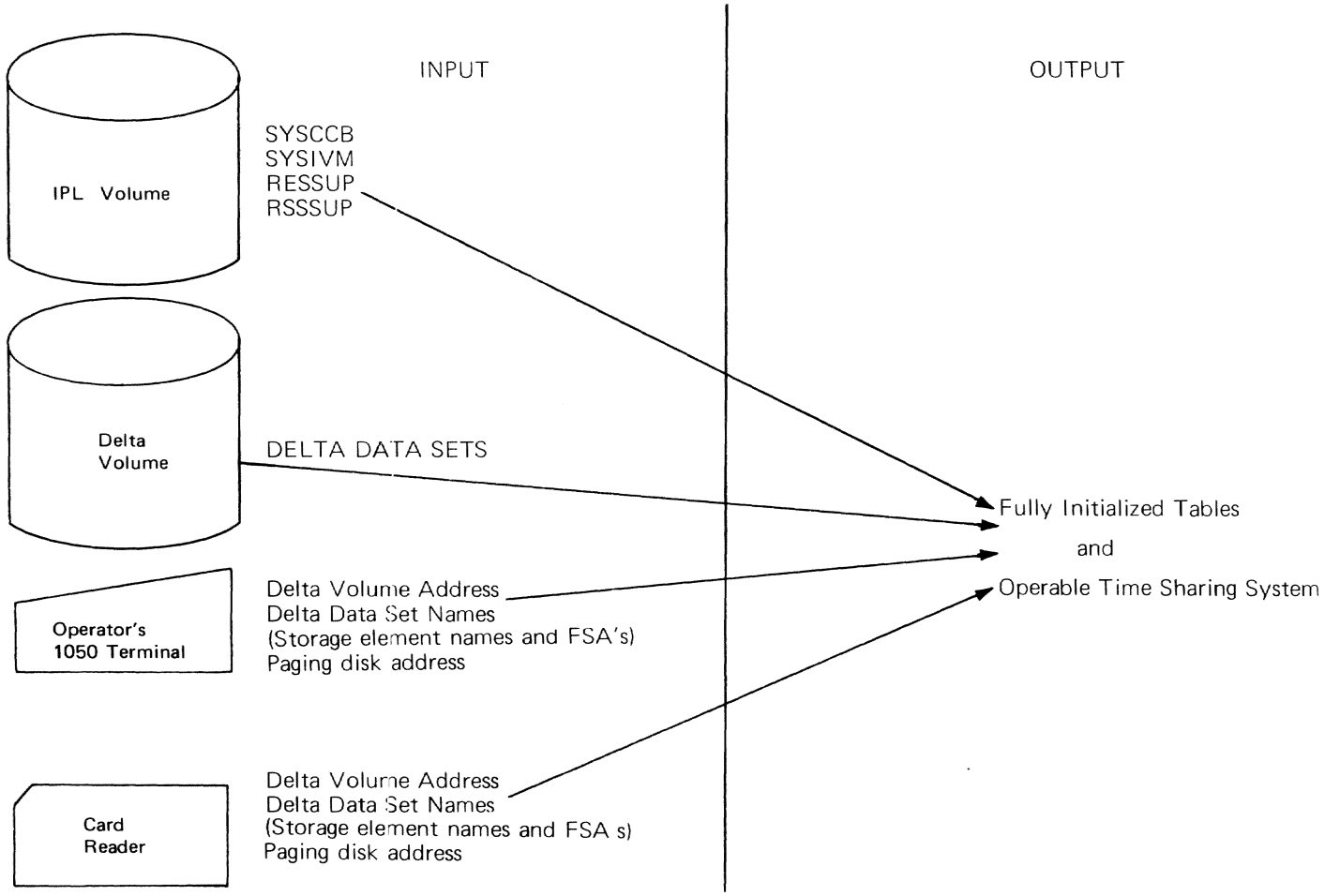


Figure 4. Startup Input and Output

dynamic modification of the time-sharing system at Startup time.

Information from the operator's terminal or card reader includes:

- The address of the delta volume and the names of the delta data sets, if any.
- The names of the storage elements and their corresponding FSAs for simplex CPUs.
- The address of the paging disk.
- The address of the Quickstart volume.
- Load list codes of functions not to be loaded.

OUTPUT FROM STARTUP

The output created by Startup results in a fully initialized and operable time-sharing system. However, there are several error conditions that will cause the termination of the Startup process. Startup terminates if:

- There are too many malfunctioning pages in main storage, thus preventing the allocation of buffer space for Startup.
- The SYSIVM, RESSUP or RSSSUP data sets are not on the IPL volume or have incomplete or missing PODs.
- The space allocated for the Extent table, Read table or storage map is too small.
- The system catalog data set cannot be located.
- Certain critical CSECTs are not present in the Delta or system data sets or their respective load lists.
- There is I/O malfunctioning while the IPL volume or the paging volume is being accessed during the initialization of the SDAT.
- The interval timer has not been enabled or is malfunctioning.
- The operator's terminal is not represented in the system pathfinding table.
- The initial virtual storage requires more than 16 segments.
- The operator cancels Startup by replying 'N' when asked CONTINUE Y or N after Startup discovers errors with delta data sets.

In any of these cases the operator is notified of the cause of termination at his terminal. If there is I/O malfunctioning during an attempt to use the operator's terminal, a message is issued on the printer.

STARTUP INTERFACE

Using normal linkage conventions, Startup communicates with the following external routines which are called to complete Startup processing:

- Inter-CPU Communication (CEIACC) issues external starts to the non-IPL CPUs.
- Set Path (CEAA5S) partitions paths to the I/O devices.
- Pathfinding (CEAA5P) locates paths to the I/O devices.
- Reverse Pathfinding (CEAA5R) returns a path to the system and determines its symbolic device address.
- Supervisor Core Allocation (CEAL1A) allocates main storage.
- Task Initiation (CEAMT1) creates and activates a TSI for MOT.
- Queue GQE on TSI (CEAAFQ) queues a GQE (containing a simulated asynchronous interruption) on the TSI for MOT.

At the conclusion of Startup, the initially-loaded CPU exits to the Queue Scanner and the other CPUs exit to the Dispatcher.

INTERNAL TABLES

Startup creates and uses internal tables, buffers and work areas (Figure 5), as follows:

AUXILIARY STORAGE DEVICE LIST buffer - set up by the SDAT initializer. This contains the symbolic device address, the device type code, and the extents for all unavailable pages on auxiliary devices. This list is used by the Auxiliary Storage Allocation routine to create directories for disk and drum in the Auxiliary Storage Allocation table.

BUFFERS AND TABLES CREATED BY STARTUP	ADDRESS	LENGTH
Auxiliary Storage Device List Buffer	ASDLST	Variable
CCB Buffer	CCBAD	Variable
Extent Table	EXTAB	150 Words
Input Buffer	INPTAD	1 Page
Load List Buffer	LDTBL, LDTBLR	Variable
Storage Byte Map	MEMAD	512 Bytes
Old POD Buffer	OPODAD	Variable
Output Buffer	OUPTAD	1 Page
Drum List Table	PGSRSDA	4 Words
PSA Buffer	PSABUF	1 Page
RESSUP/RSSSUP Symbol Table	SYMAD	Variable
SDAT Buffer	SDATB, SDATBH	Variable
SSDAT Buffer	SSDATB, SSDATBH	Variable
Shared Page Table Buffer	SPTAD	1 Page
TDY Page Table	TDYTAB	62 Words
Work Area	DSCBF	140 Bytes
Work Buffer	WORKBF	1 Page
XTSI Buffer	XTSIAD	1 Page

Figure 5. Startup-Created Buffers and Tables

CONFIGURATION CONTROL BLOCK buffer - a data set read in by Prelude. This is the table set up by SYSGEN (and modified by SYSGEN). It contains configuration-dependent information utilized by Startup. It supplies: the path of the drum to be used as the paging drum; the address of an alternate printer should the current one malfunction; and the number of Shared Data Set table pages. It also contains information used to update the CPU Status table (CST) and the Pathfinding tables.

EXTENT table - a multipurpose area. This table contains the extents for the SYSIVM, RESSUP and RSSSUP data sets. During initialization of the Pathfinding tables, it contains a list of partitionable control units. The routine to initialize the SDAT uses the Extent table as a drop area to contain the symbolic device addresses, and virtual storage pointers to the SDAT entries for all volumes containing the System Catalog. It is also used when the JFCB for the System Catalog is updated.

INPUT buffer - information from the IPL volume is read into this buffer, as are the three data sets used during link-loading. While the SDAT is being initialized, the first 144 bytes of the buffer are used as a read-in area for the format-4 DSCB contained on the disks. The area following these bytes is used to contain the Auxiliary Storage Device List.

LOAD LIST buffer - contains the load list for the SYSIVM, RESSUP, or RSSSUP data sets.

STORAGE BYTE MAP - used to determine the availability of pages in main storage. It supplies information for initialization of the Core Block table and setting of the storage protection keys.

OLD POD buffer - used by the link loader to hold the POD for either the SYSIVM, RESSUP or RSSSUP data set, whichever is being processed.

OUTPUT buffer - the buffer from which data is written on the primary paging volume.

DRUM LIST table - used during the link loading phase to contain the hardware addresses of the drums in the system, the first drum being the paging drum. The list is used to locate the paging drum and then provide addresses for all the drums, so the SERR/Reconfiguration modules may be written on them. The hardware addresses are converted to symbolic device addresses, and this list is used by the Auxiliary Storage Allocation routine to insert the drum addresses into their respective drum directories in the ASAT.

PSA buffer - used as a temporary storage area to hold the PSA CSECTs. The PSA CSECTs are moved to PSABUF after they are read in during RESSUP processing. After the PSA is initialized, and just before exiting from Startup, the PSA is moved to the actual PSA area.

RESSUP/RSSSUP SYMBOL table - contains the symbol and address of every entry point in the Resident Supervisor and the Resident Support System.

SDAT buffer - used during initialization of the SSDAT.

SHARED PAGE TABLE buffer - used during initialization of the Shared Page table.

TDY PAGE table - contains main storage addresses of the TDY for SYSIVM. This is a table of one or more three-word groups or extents. The first two words contain the lowest and highest real storage addresses relocatable by a common factor, namely, the third word. The end of the table is a double word of ones.

WORK AREA - used as a DSCB and volume label read-in area. All DSCBs for SYSIVM, RESSUP and RSSSUP data sets are read in here.

WORK buffer - used to create the Free Space table and during SDAT processing.

XTSI buffer - used during initialization of the XTSI.

The following is a description of the logic flow of each of the Startup routines and subroutines.

STARTUP MAINLINE (CEIAA)

Startup (Chart AC) consists of a mainline routine which calls a number of independent routines with specific functions, and a set of common subroutines. Upon entry, the mainline executes several housekeeping steps, and sets up the base registers. It also sets up a communication area within Startup, and moves into it the drop area set up and created by Prelude. A check is then made to see whether the interval timer has been activated. If it has not, a message 'Activate TIMER and Re-IPL' is issued and Startup terminates. Otherwise, the Enable routine is called.

The Enable subroutine enables the 2702 and 2703 lines so that a terminal user can dial into the time-sharing system. On entry, it does housekeeping, relocates CCWs and adcons according to Startup's load point, and obtains the addresses of the lines from the Transmission Control Path table (CCBTPT) of the CCB. The line's type and address are obtained from each entry in the table. The Enable subroutine determines the SAD order specified, and the device class (dedicated versus dial-in). An appropriate CCW list is constructed. The address of the list is set in the CAW and a SIO instruction issued. The complete CCW chain is not executed until someone dials into the line, ending the Enable command that is command-chained to the rest of the list. A message is sent out on any line dialed-in.

On return, the addressing capability of the system (24- versus 32-bit addressing) is determined. In order to do this, the system mask in the extended PSW is set to X'08' indicating that the 32-bit addressing feature is operating. If the 32-bit optional hardware feature has not been installed, a program interruption occurs when the next sequential instruction is reached. A switch (MCHSZE) is set according to the presence or absence of the 32-bit addressing feature. This switch is tested at various subsequent points to enable the system to operate in the proper mode.

In a multi-CPU environment, control next passes to a mainline routine, P400X. In simplex mode this routine is bypassed. The purpose is to create a PSA list containing the addresses of the active PSA of each CPU in the system, excluding the loaded CPU. P400X does this by finding the address of the primary PSA of the CPU in the CCB. The routine then inspects the Page Map for a X'3C' to determine if that page is available as a user page. If so, the Page Map indicator is reset to X'94' indicating that this page is partitioned. The address of the primary PSA is moved into the active PSA list. Should this primary page be unavailable, the address of the alternate PSA of the CPU is found. If this alternate page is available, the partitioned flag is set in the page map and the address is moved into the active PSA list. If this alternate page is unavailable, the non-loaded CPU cannot be started up, since it

has no active PSA. In this case, the address placed in the active PSA list is 0. This list, set up by P400X, is used later by Startup to set up the CPU Status table (CST), and the Storage Byte Map is used to allocate available pages, and set up the Core Block table (CBT).

The operator is asked for the address of the card reader in the event that operator response messages are to be bypassed. If a Quickstart is taking place, this question is bypassed and all queries are made to the operator's console.

A test is made to determine whether this is a simplex system or not. If it is, the SIMFSA routine is called. If not, control passes to P100X. SIMFSA is the routine to request information regarding the FSAs on the SEs attached to a simplex CPU. The operator may partition out a storage element (block of 256K bytes) by not mentioning that storage element's identification. However, the operator may not partition out the FSA that contains the PSA, or the FSA that contains Startup (just loaded), or Startup's buffers. If operator response messages are not bypassed, OPER is called with the message code asking the operator to enter the FSA code for each SE in TSS in this format:

```
XN,XN,XN,.....
```

where

```
X = SE ID (A,B,C,.....)
N = FSA code (0,1,2,.....for 0-256K,
    256K-512K, 512K-768K.....)
```

If operator response messages are bypassed, the READCARD routine is called to read a card containing this information. The operands are then accessed via GETFLD. All are checked for validity. An incorrect response condition is recognized if any of these conditions exist:

- Operands are longer than two characters in length.
- The number of responses exceeds the number of SEs at the installation.
- An SE ID is not within the allowable range, which is the count of SEs at the installation. (If four SEs exist, the allowable range is A to D.)
- An FSA code is not within the allowable range which is the numeric count of SEs at the installation, beginning with 0. (If four SEs exist, the allowable range is 0 to 3.)
- Two operands in the reply mention the same SE.
- Any FSA code not mentioned that contains the PSA, Startup, or Startup's buffers.

On any incorrect response condition, the operator is informed and asked to reenter the information. Should the same FSA code be assigned to two SEs, the operator is informed. If this is due to an incorrect response, he can then reenter the information correctly. If two or more SEs are actually set to the same FSA, the condition must be corrected and the system reinitiated by an IPL.

Once a completely valid response is received, the information is rearranged in extended control register 10 format and placed in the Startup communication region. This is later moved to the SE Status table in the PSA. Control comes back to the mainline routine P100X. The purpose of this routine is to move the Configuration Control Block, the Storage Byte Map, the IPL PAT page and an IPL DSCB page from their locations assigned by the Prelude routine to the buffers allocated them in Startup. (Information as to their origin, length, and destination is available to P100X in Startup's communication region.) The information is then moved in blocks of 256 bytes until the entire block has been transported. The origin of the Page Map is at a fixed location, 190_{16}. The destination was allocated by GETMEM. The length of the map is proportional to the number of SEs in the particular installation. There are 64 bytes per SE. Once the move is complete, control continues on to the DMLST routine.

Create List of Available Drums (DMLST) is another subroutine of the mainline. Its purpose is to identify all available drums of the system, and ask the operator to supply the address and device type code of an auxiliary device to be used as the paging device. First, the routine searches each drum entry in the Drum Path table of the CCB for one available path to each drum. All entries are checked one drum at a time. If a good path is found, it is stored in a list. If no path is found, the next entry is checked until all are investigated. The device attached to the first path in the list is used as the paging drum. A count of available drums is made, and the results are saved for the ASATRT routine. If operator response messages are not bypassed, a message is issued (via OPER) requesting the operator to supply the address of a paging disk. If operator response messages are bypassed, READCARD is called to read a card which contains the address of the paging disk. A check is made to see that the IPL volume address is not specified as the paging disk. If it is, the operator is prompted for another

address. A check is also made to see whether the paging pack is VAM2-formatted. If not, the operator is prompted for another address. If so, and if a paging volume is mounted (byte 17 of the volume label = X'40'), processing continues. If a paging drive is not mounted, the operator is prompted for another address. The paging disk is used as an additional paging device. If there are no drums available, all Startup paging is done on this paging disk.

Subroutine RTMPGS is invoked next. It calculates the size of CHBTCT and CHBBFP (RTAM tables). The sizes of these CSECTs depend upon the number of terminals connected to the system during system generation. The results are saved in the communication region for use after link-loading.

The Link loader, Startup's main processing routine, is next invoked to load SYSIVM. (A complete description of the Link loader is provided later in this section.)

After SYSIVM has been loaded, a switch (PGWRT) is set and the link loader is recalled. The switch setting indicates that RESSUP/RSSSUP is to be loaded. If a Quickstart is taking place, the link-loader is bypassed and QKREAD is called to read in the link-edited version of IVM, RESSUP, and RSSSUP.

If Delta data sets were specified, a message is now issued at the operator's terminal. This message notifies the operator that he may remove or replace either the IPL volume or the delta volume, or both. If no Delta data sets were specified, the operator is notified that he may remove the IPL volume. If the Delta volume is not removed and it is marked as public, it is used as a public volume. If the IPL volume is not removed, the 2314 unit that it is on is marked as system-dedicated. If a Quickstart data set is to be created on either the IPL or Delta volume, the demount message for that volume is not issued.

If this was a normal Startup, a test is made to see if a Quickstart data set is to be created. If so, CEIAB is invoked to create that data set. (See Chart AC, p.8.)

If any Q-cons were loaded, the pseudo-register value is displayed on the printer. A subroutine (QRDR) is then invoked to update the dynamic loader and task common. The Q-con hash table and CXD values are inserted in the dynamic loader PSECT, and the CXD value is placed in task common. The current value of the CXD is displayed on the printer.

Now the mainline invokes a series of subroutines to do table initialization. Startup mainline may now communicate with the RESSUP that is now in main storage. The first subroutine, Initialize Pathfinding Tables (SETPTH), partitions channels and control units (via RESSUP Pathfinding) that are outside the time-sharing system environment. The subroutine's functions depend on the machine configuration at the installation. The routine disables I/O interruptions and determines whether or not the system is simplex.

In a simplex system, control bypasses the above-mentioned partitioning and goes to the Special routine. In a duplex or half-duplex system, the subroutine uses the switch settings on the configuration console, the channel controller and the Correspondence List tables of the CCB to find and partition those channels and control units not attached to the system. On entry, the following fields are filled in with the corresponding parameter information:

SIMDUP	simplex/duplex indicator
CCBAD	pointer to CCB
SPATH	pointer to CEAA5R
SPATH2	pointer to CEAA5S
SSYS	pointer to CHBSYS
SSTA	pointer to CHBSAC
PAGAD	address of primary paging device
TERMAD	address of operator's terminal
VOLAD	address of IPL volume
SDABF1	buffer pointer
PGSRSDA	list of drum addresses
PARTND	prelude map of partitioned CPUs
CREG8-CREG15	extended control registers 8-15

SETPTH locates the CCU Map in the communication region which has been filled in by Prelude following a configuration analysis. It searches the table for a CCU found to be partitioned. The list of channels connected to the partitioned channel controller is passed on to the Set Path routine (CEAA5S) of the Pathfinding module in RESSUP, for partitioning. Error checks are made and should a channel be found to be nonexistent in the Pathfinding tables, an error message is sent to the operator. CEAA5S does not complete the list if an error is encountered, so any remaining channels to be partitioned are presented to Set Path again. Every CCU is processed in this manner. In order to process CUs, SETPTH inspects each bit in extended control registers 12 and 13. Each bit corresponds to a CU switch setting. If the particular CU is attached to a CCU, the bit setting is on. If partitioned, the setting is off. There is a Correspondence List in

the CCB which shows the paths in variable number of halfword entries of each switch represented in extended control registers 12 and 13. SETPTH obtains the address of this list. It determines which paths are to be set as partitioned in the Pathfinding tables and builds a path list as obtained from the Correspondence List. CEAA5S is then invoked to partition the list and the same error procedures are followed as for CCUs. Once all the CUs have been processed, control is sent to the Special routine.

The Special routine creates the following special device path tables:

SERR/Reconfiguration Path table
Operator's Device Path table
IPL Volume Path table

Pointers to these tables are set in the appropriate fields of the System table. Special begins by copying extended control registers 8-15 into the System table, and calls on the RESSUP Reverse Pathfinding routine (CEAA5R) to obtain the SDA of the paging volumes. The operator's terminal address is stored in the System table, and its SDA is likewise obtained. Should CEAA5R return an indication that an asynchronous interruption entry does not exist in Pathfinding for the terminal, or that the terminal cannot accept asynchronous interruptions, a diagnostic is issued (via OPER) and Startup terminates. Otherwise, SETPTH finds the entry in the CHBSAC for the SDA. This provides a pointer to the Device Group table.

The Operator's Device Path table is created from information provided by the above-mentioned tables, in the form of fullword entries. DIRSIZ is invoked to obtain available storage for the table. The Operator's Device Path table is moved and the System table is initialized with the number of entries and the location of the Operator's Device Path table. Should an error occur in obtaining or attempting to find the SDA, the creation of the Operator's Device Path table is bypassed but processing continues. The same process is followed if the IPL Volume Path table cannot be found.

The SERR/Reconfiguration Path table is created only if drums are present in the system. A list giving a path to each drum is provided as input to Special. The SDA for each entry is obtained again via CEAA5R, and is stored back in the list. The SDA is used as above to locate all paths to each drum. Each first halfword of each fullword entry in the path table is set with the SDA+1 of each drum. Any errors are handled as described above. On exit this output is generated:

PPDSDA SDA of paging drum
TERSDA SDA of operator's terminal
VOLSDA SDA of IPL volume
PGSRSDA list of SDAs for all drums
Certain fields of the CHBSYS
Three Path tables

Control now returns to the mainline routine.

Mainline now invokes PGXTSI, a subroutine used to complete the initialization of the XTSI, XPT, and XSPT. If the system is duplex or half-duplex, the routine also gathers the information which is to be loaded into extended control registers 4 and 6 later on in Startup. The CCT table in the CCB is used to determine the channel controllers and channels in the system. One word, CREG4, is initialized to represent the channels in the system. (This word is also initialized for a simplex system.) A second word, CREG6, is initialized to represent the channel controllers in the system and, in a duplex system, indicate that the malfunction alert fields are enabled.

Initialization of the XPT/XSPT is completed by invoking LOCXPT to find the origin of the XPT/XSPT and then storing the SDA of the paging device in the first two bytes of each XPT/XSPT entry.

XPT2 and XPT3 are now located in the RSS Communication table. The symbolic device address of the paging disk is initialized in each entry.

Startup calls the Reserve Pages routine (RESRVP) next. The Reserve Pages routine reserves as many pages for RSS on the paging disk as there are RSSSUP and Symbol table pages. These pages will contain images of those read-only pages of the Resident Supervisor which will be overlaid when RSS is loaded.

The external address of each reserved page is stored in an XPT4 entry in the RSS Communication table.

The next routine invoked is BFRPGET. Two pages of supervisor main storage are obtained (via GETMEM), the first to be used as the first page of CHBTCT, and the second to be used as the first page of CHBBFP. The real storage addresses of these two pages are converted into page table format (halfword block addresses) and are stored in the Shared Page Table entry reflecting the virtual storage addresses assigned to the TCT and BFP.

INTDE is invoked next to initialize all RTAM tables: CHBTDE, CHBTCT, CHBDEV, CHBBFP, and CHBMTS. See the section 'RTAM

Initialization' for more information on these tables.

The next routine invoked is SDATRT, the SDAT Processing routine. It is used to initialize the SDAT and to provide various device-related information for use by Startup. On entry, the following input is located in the communication region:

SSDATL	address of XSPT entry for SDAT pages
SSDAT	VMA of SDAT header
SDALST	VMA of first entry in SDAT
SSDATP	number of external SDAT pages
PAGAD	physical address of paging drum
PPDSDA	SDA of paging drum
SPDSDA	SDA of paging disk
VOLSDA	SDA of IPL volume
VSSDAT	VMA of VSS SDAT
RSSDATL	location of first RSS XPT
RSSDATP	number of RSS SSDAT pages
SSSDAT	location of RSS SSDAT header
SPVTP	number of PVT pages
SPVTL	address of XSPT entry for PVT pages

The SDAT is a shared virtual storage table consisting of an 8-byte header and a variable number of 64-byte entries. The header and each entry begin on doubleword boundaries. Entries are arranged in ascending order of SDA fields. The SDAT resides on more than one page, but does not necessarily begin or end on a page boundary.

Concurrent with processing of the SDAT is processing of the Support System Device Allocation table (SSDAT) and the Public Volume table (PVT). SSDAT combines information currently in SDAT and Pathfinding tables. SSDAT consists of a header and a body portion. The header includes pointers and entries for the operator's console and three RSS residence devices. The body consists of entries for all symbolic devices defined in the system.

Both the header and body are initialized by the SDATRT routine. Entries for the operator's console, primary paging device, and secondary paging device are made in the SSDAT header. The VAM-formatted flag is set for the paging disk. The header to be placed in RESSUP (CHBECXRA), has adcons pointing to the first and last SSDAT entries in the RSSSUP SSDAT (CHBECXRB). The header to be placed in SYSIVM has adcons pointing to the first and last SSDAT entries in the SYSIVM SSDAT (CHBECXVA). Startup resolves these adcons, which were initialized by SYSGEN.

The maximum number of possible volumes allowed in the system is determined during system generation. This number is set in the PVT header field PVTMCT and space is allocated by SYSGEN for as many entries as indicated by this field. Startup initializes the volume serial number, symbolic device address, device code, and pages relocated flag for each legal public volume that it encounters.

SDA500, a routine within the SDATRT, is called to process individual 64-byte SDAT entries. RESSUP Pathfinding is invoked to determine whether a path is available to the device whose SDA is being processed. If it is not available, the available flag is set off and the partitioned flag is set on. No further processing is required, and control is returned to the SDATRT mainline. If there is an available path, RESSUP Reverse Pathfinding is invoked to free the path. Control is returned to the mainline, unless the device is direct access, in which case the device type returned by the Pathfinding subroutine is checked.

If the device is a 2301 drum and appears in the list of accessible drums (PGSRSDA) established previously in Startup by DMLST and Special, a variable byte entry for the drum is entered into the Auxiliary Storage Device List (ASDLST) created by SDATRT. A count is kept of auxiliary devices in the field DEVTOT. If the drum is not in PGSRSDA, the partitioned flag is set on, and the available flag is set off in the SDAT entry. Control returns to SDATRT mainline.

At this point, it is known that the device is either a 2311 or a 2314. The external storage, available, not reserved, and IOREQ allowed flags are set on in the SDAT entry. The SDA of the device is then compared with the SDA given earlier for the paging disk; if the SDAs are the same, the auxiliary storage, reserved, IOERQ not allowed, private, and not available flags are set on in the SDAT entry. If the SDAs are not the same, processing continues. The volume identification is then read from the disk and checked to see if the disk is VAM2-formatted. If it is not, it is assumed that the volume is SAM-formatted. In this case, the format-4 DSCB is read and the device constants and gross available space data are set in the SDAT entry from the VTOC DSCB. The appropriate flag in the VTOC DSCB is examined to determine whether the device is private or public and the corresponding flag bit set in the SDAT entry. If the volume is public, it is placed in the public volume chain and a check is made for the ACV volume (public volume 0). If the volume is the ACV, it is checked to see if the SYSCAT and SYSSVCT data sets are on the volume. If they are, their pointers are placed in the TDT; if they are not, Startup terminates.

If the disk is VAM2-formatted, it is checked to see if it is the IPL volume. If it is, and if the volume is still mounted, the external storage, unavailable, and reserved flags are set on in the SDAT. If the device is not the IPL volume, or if the volume has been dismounted, processing continues. The current device is again checked to see if it is the secondary paging volume; if it is, Initial Virtual storage extents are moved to the auxiliary device list and the PAT page is read in. If it is not, the secondary paging volume, a check is made to see if a paging pack has been mounted. If it has (X'40' in byte 17 of the volume label), the auxiliary storage, private, unavailable, reserved, and IOERQ not allowed flags are set on in the SDAT.

If no paging pack is mounted, the public/private flag in the volume table is checked. If the public flag is not set, the volume is marked private in the SDAT. If the public flag is set on, the volume serial number, the current symbolic device address and the device code are inserted into the appropriate PVT entry. Startup determines which entry in the PVT will be initialized for this volume by examining the relative volume number field in the volume label. If the relative volume number exceeds the maximum allowable limit, Startup issues a message to the operator indicating the device address and the relative volume number of the volume in question. Startup then asks the operator to remove or replace the volume, and after doing so, reprocesses all pertinent tables. If the PVT entry corresponding to the volume being processed has already been initialized, Startup issues a message to the operator indicating the symbolic device addresses of the two duplicates and their relative volume numbers, and asks the operator to remove one of the volumes. Startup then reprocesses all pertinent tables.

If the volume qualifies on both of the above-mentioned checks, its PVT entry is initialized as described above and it is entered in the public chain. The PAT for this volume is also checked to determine whether this volume contains pages which have been relocated by the RESTORE program. If so, the pages relocated flag is set in the PVT entry for this volume.

Startup reads in the PAT table for all public and auxiliary volumes. It examines each PAT entry and maintains a count for every entry that represents an unused page (X'00'). It also examines each data page entry (X'01') to determine if the Allocated flag is set. If an entry is found (X'41'), it is cleared and the available count incremented. The accumulated available page count is stored in the PVT entry for the volume. The PAT table is then rewritten to external storage to reflect the change. This is done when a page assignment is made from the volume which requires the rewriting of the PAT table. If the volume being processed is auxiliary storage, the PAT information is used to set up the auxiliary storage allocation bit directory. If the volume is public storage, the PAT information is used to set the Available Space field for the volume in the Public Volume table (PVT) and to recover lost space from the volume.

The PAT for each public volume is mapped into shared virtual storage by initializing the next available entry or entries with the external address of the PAT. (The virtual address of the PAT is determined by the relative XSPT entry.) The virtual address and relative page number of the PAT are then placed in the SDAT entry for that volume.

A flag is set in the XSPT entry to indicate that this virtual page resides on external rather than auxiliary storage.

If, when trying to gain access to a device, during any part of this processing, Startup finds the device to be malfunctioning, the phased out flag is set on, the 'available' flag is set off in SDAT, and the device is set as malfunctioning in the pathfinding table. If a direct access device is located and has an available path, but is not mounted, further processing is bypassed for that entry.

After all SDAT entries have been processed, a check is made for missing public volumes. Startup has kept track of the highest relative public volume it has encountered. This information is used to initialize the PVTECT field (actual volume count) in the PVT header. All PVT entries up to the one with the highest volume number encountered are then examined. If any of these entries have not been initialized, Startup issues a message to the operator indicating which relative volume numbers are missing. Startup then asks the operator to mount the missing volumes on any available drive, after which all tables and their entries are re-initialized.

WRXTSI writes the XTSI on the paging drum, and the head and slot number of the XTSI page on the drum, and its SDA, are saved in the System table. LOCXPT is invoked, with the SPT pointer as input, to determine the address of XSPT. The SDA of the paging drum is set in the first two bytes of each 12-byte XSPT entry corresponding to a virtual storage address allocated by Startup. Control returns to the mainline.

ASATRT is invoked next. This routine initializes the Auxiliary Storage Allocation table (ASAT), which is used to control allocation of pages on auxiliary storage devices. It is first set up for the paging volumes. After the execution of Startup, pages in use on the paging drum are those occupied by shared IVM, SERR/Reconfiguration path table, the skeletal XTSI and the Shared Data Set table. If the paging device is a disk, the unavailable pages are PAT pages, IVM pages and error pages.

The extents of the unavailable pages on each auxiliary device have been previously set up by the SDATRT. SDATRT uses information from the PAT for each auxiliary disk plus information gathered by the Linkloader routines to make this determination. It is important for Auxiliary Storage Allocation to know exactly which pages it may utilize so that Supervisor time is not consumed trying to use malfunctioning or occupied pages. Additionally, ASATRT initializes count fields for each cylinder on an auxiliary disk. These fields indicate the number of pages available on each cylinder. This makes it possible for the Supervisor to minimize seek time by minimizing the number of cylinders on which it has to perform write operations.

The highest external page number on the paging drum (the XTSI page) is supplied by the PGXTSI routine. Pages occupied by SERR/Reconfiguration are provided by the SERR100 and SERREND routines for each drum in the system.

On the paging drum, pages 882-890 are reserved for standard error retry procedures. Page 885 is unavailable because of error problems. Checks are not made for bad pages on either drum or disk. Bit directories for 2311 or 2314 disks include only 199 cylinders, since the last four are reserved for standard error retry procedures.

The number of auxiliary storage devices as determined by the SDAT routine is checked to determine whether there are any additional devices for which to set up tables. If there are, the device type in the auxiliary storage device list is checked and the appropriate table is created. These bit directories reside in real storage immediately following the Resident Supervisor.

The directories for devices of the same type are chained together as each table is built. The following information is updated in the ASAT header: number of drums, number of drum pages available, number of auxiliary disk pages available, address of the first disk directory, and address of the first drum directory. For each bit directory, a subheader is also filled in with the following information: number of pages available on drum or disk, SDA, address of next drum or disk, directory in chain (a circular chain where last points to first). In disk directories, a subheader is filled in with the number of bytes in the directory. After all directories are built, the low drum availability threshold is calculated and entered in the ASAT header.

Startup reads in the System Common (CHASCM) and uses the value in SCMAUX to calculate a new value for the augmentation of auxiliary storage which is stored (overlays) in SCMAUX and the System Common is rewritten to auxiliary storage.

If a new Quickstart data set has just been created on a public volume, the relative public volume number, the relative page number, and the slot number for the E-DSCB are set in System Common for use by CZACB. The third level data set name qualifier (DSxxxxxx, where xxxxxx is the volume identification on the pack) is also put into System Common at this time.

Once the tables have been set up, the reserved list (five pages) is set up for use by Supervisor Core Allocation, and a list containing pointers to these pages is built. Control returns to the mainline.

The next routine invoked is ANZSDA, which analyzes all the direct access devices attached to the system. The SDAT table (CHBSDA) and the PAT pages for each device are examined for the following information for each device:

- The symbolic device address
- The physical device address
- The device type (2311, 2314, or 2301)
- The status of the device (partitioned, not ready, etc.)
- The relative volume number of the pack on the device (if public)
- The number of available pages on the device
- The VOLID of the pack on the device.

The PAT pages are checked for validity to insure that for each device:

- Relative page zero has a PAT entry of X'C0'
- The PAT self-descriptor entries are X'7F'

- The byte following the last PAT entry contains X'FF'

- No other entry except the last byte following the last PAT entry contains X'FF'.

The result of the analysis is displayed on the printer.

If any of the above checks fail, the SDAT entry (SDAINV) corresponding to the device being processed is marked X'80', the number of available pages in the Public Volume table is set to zero (if the device is public), and, after all devices have been processed the entire SDAT and PVT tables are re-written on the primary paging device. A message is sent to the operator informing him of an invalid PAT on the device being processed. Control then returns to the mainline.

The next routine called is the Allocate SERR Operating Pages routine (SOAPGS). This routine allocates real storage to be used as the operating area for System Error Recording and Retry. SOAPGS uses GETMEM to search for the storage area. An attempt is made to allocate the two-page operating area in the same storage area as the active PSA. Since it is possible to have more than one PSA (in duplex mode), there may be more than one two-page SERR operating area. The address of each area allocated is saved in its corresponding PSA, in the field PSA-SOA. If no area is available, the field is set to zero, and control is returned to the mainline.

The JPSA routine is invoked next. It uses information in the communication region pertaining to the number of CCUs and SEs in the system, together with the CPU Status table of the CCB, to initialize the CPU Status table in the PSAs for each CPU in the system, excluding the initially-loaded CPU. It does this by locating the SE section of the CPU Status table in the CCB, extracting appropriate information, and storing it in the PSA CSTs. Information in the communication region pertinent to JPSA is as follows:

CCBAD	pointer to CCB
SESIDS	SE Map of IDs
CPUSTSS	number of CPUs in TSS
PSAS	list of active PSAs for the CPUs
RESTART	RESTART indicator
OTHCPU	map of IDs of non-IPL CPUs (if applicable)

The floating storage address for each SE in the installation is initialized in the SE entries, as taken from the extended control registers for a duplex or half-duplex system, or from the operator input for a simplex system. The number of SEs is stored in a field called SETOT. The partitioned flag is set on for all those SEs that are partitioned from the system. The number of CPUs in TSS is set in the CST header. If the count is greater than one, a new machine check PSW is placed in the PSA buffer for the loaded CPU. This buffer will be moved to the real PSA for this CPU at the end of Startup. All other PSAs are initialized now. In the event of a restart, if a CPU is failing, its entry in the CPU Status table is set with the malfunctioning bit on. Partitioned CPUs are set as such in their PSAs.

The inactive prefix indicator for primary or alternate prefixes is set in the CPU Status table entry for each CPU in the system. Now CPU entries are rearranged in the CPU Status table so that the entry for a CPU will appear as the first entry in its copy of the table. The IDs of CPUs are stored in a byte (WRDCT) to be used by the Intercom routine for issuing an external start. The MTS fields MTSMAX and MTSTLM are set to the value found in CCBCON and the MTSCUR fields is set to 1.

The SAR is read in and the SARMCN and SARCNL fields are set to the volue found at CCBCON. The SARMMA and SARMAL fields are set to the value from CCBMTT, the SARMBT and SARBTL fields with the CCBBAT value, and the SARMRM and SARRML fields are set to the CCBBAK value. Control is now returned to the mainline. At this point RELMEM is called and all the pages that have been reserved for Startup are released in the Page Map.

SCBTL is invoked for the initialization of the Core Block table. The Page Map, created by the main storage analysis done in Prelude, and updated by routines in Startup, provides input for this routine. Now each byte within the map reflects the condition of its corresponding page of real storage. The byte settings and their meanings are:

3C = user page

88 = failing page (machine check detected during storage analysis)

94 = partitioned page (addressing interruption during storage analysis)

DC = RESSUP page

FC = reserved for RESSUP

SCBTL moves the bytes from Page Map to their corresponding entries in the CBT, and initiates the user page chain. All entries flagged as user pages are chained with both forward and reverse links. Two entries in the CBT header are filled in: a pointer to the user chain and the number of user pages. Control passes to P4800X.

This routine sets the storage protection keys for each halfpage block of main storage. It bases these settings on the flags in each byte of the Page Map. The keys and their meanings are:

1 = user page
5 = RESSUP page
4 = reserved for RESSUP

If the page is partitioned or failing it is not assigned a key. Control now passes to the CRRSPI (Create RSPI) routine. CRRSPI calls Supervisor Core Allocation to assign main storage for the SPTs and XSPTs. The SPT/XSPT is then moved to the area reserved by Supervisor Core Allocation. The SPT/XSPT origin is stored in the RSPI.

PARTMP is then called to analyze the Core Block table and the partitioning registers, and to display on the printer a map of all storage on the system during this Startup. The information printed describes:

- Each storage element (A through H)
- The CPU (none, 1, 2, or both) the SE is connected to
- The real storage location of any pages in error (failing storage locations are indicated by a Core Block table entry of X'88')
- All partitioned storage.

ENDABLE is called at this point to reinitialize each terminal line. Now any user who dials in on a dedicated terminal will interrupt the system and request to log-on. In order to do this a disable followed by an enable is issued to every line. Each line is then inspected to determine if the line is valid, and if the correct data set line adapter is specified. For each dedicated line, a Prepare command is sent out. If during Endable an unrecoverable I/O error occurs, Startup calls Reverse Pathfinding (CEAA5R) and pathfinding (CEAA5P) respectively to determine if the control unit was partitioned at SETPTH time. If the unit was partitioned, Startup ignores that bank of lines; if the control unit is available, and Startup is able to perform successful I/O to other lines on that particular group, a malfunction message is sent to the printer; if no successful I/O is performed on a group of terminal lines, pathfinding (CEAA5S) is called to partition the control unit, and a message is written to the printer. At the end of the ENDABLE routine, if any message was written on the printer, an appropriate diagnostic message is sent to the operator's console.

SETTSK, the subroutine to create the Main Operator's task, creates a TSI with a task ID of 1, creates a GQE pointing to this TSI, and queues the GQE on the TSI. Task Initiation (CEAMT1) sets up a TSI and returns a pointer to it. The pointer is stored in the asynchronous interruption entry for the operator's terminal in the Device Group tables (Pathfinding). Certain fields in TSI are set; the task ID is set to 1, the system operator priority flag is set on, the interruption pending flag for an asynchronous task is set on, and the intertask acceptance flag is set off. A 64-byte GQE is provided by Supervisor Core Allocation (CEAL1A). Stored in this GQE is a pointer to the TSI, an interruption code (a path to the operator's terminal), and the symbolic device address of the operator's terminal. This GQE is now queued on the TSI's asynchronous interruption queue, via the Queue GQE on TSI routine (CEAAFQ).

The QSCNPSA routine is invoked and Startup exits directly to the Queue Scanner. The SPSABUF routine is called to move the PSA of the initially-loaded CPU from the PSA buffer to its preassigned location in main storage.

At this point, if there are any other CPUs to be started up in the system, Intercom is invoked to issue external starts to them.

When the other CPUs are ready, the initially-loaded CPU sets a flag and exits to the Queue Scanner. This flag acts as a signal to the other CPUs that they may now exit to the Dispatcher.

Startup is complete, and an operable TSS/360 has been created.

LINK-LOADER MAINLINE

Link-Loader (EIAA5)

Chart AC, Page 2

During Startup, the Initial Virtual storage routines in TSS*****.SYSIVM and IVM Delta data sets are link-loaded onto system paging devices. The Resident Supervisor routines in TSS*****.RESSUP and RESSUP

Delta data sets together with the Resident Support System routines in TSS*****.RSSSUP and RSS Delta data sets are link-loaded and written into main storage or the paging disk respectively. Startup uses a CSECT, called a load list, within each data set to determine which CSECTs must be loaded for Initial Virtual storage and the Resident Supervisor.

The operator or card reader input provides for loading:

- Alternate load lists.
- Alternate and/or additional CSECTs.

Prior to searching the system data set SYSIVM (RESSUP), Startup looks for the load list in a hierarchy of data sets specified by the operator or by card reader input. When the load list is found, Startup enables the operator (from the operator's terminal or the card reader) to exclude the loading of certain csects. The message

```
ENTER CODE FOR FUNCTIONS NOT TO BE
LOADED.  ALL FUNCTIONS WANTED=EOB
```

prompts the operator for the codes of the messages to be excluded. STARTUP does not load those modules that have the selected function code in the last byte of the load list entry. When the codes have been entered, Startup sequentially retrieves and loads each CSECT named in the load list but not excluded; again, the hierarchy of data sets specified by the operator or by card reader input is searched before the system data set. In the case of user modules, either the module name or a CSECT name can appear in the load list. In either case, Startup loads all CSECTS in the module. The System Programmer's Delta data sets remain in the system from Startup to shutdown. CSECTs in these data sets will override versions of the same CSECTs which reside in the system data sets. Programs of each user who logs on during this session will run with all of these modifications to Initial Virtual storage, the Resident Supervisor and the Resident Support System.

If Delta data sets are requested, a list of Delta data set names and their hierarchy numbers is printed out.

The Link-loader interrogates a load switch set by the Startup mainline to determine whether to load Initial Virtual storage or the Resident Supervisor/Resident Support System. The mainline initially sets the switch to one so that Initial Virtual storage is loaded first. The switch is zero when the Resident Supervisor/ Resident Support System is to be loaded. The name of the data set currently being processed is stored in location DSNAM. DSNAM is tested at various points in the code to determine if Initial Virtual storage, Resident Supervisor, or Resident Support System is being processed. Most of the associated tables are initiated in this routine.

Input to the Link-loader consists of the RESSUP, RSSSUP and SYSIVM data sets (which reside on the IPL volume), as modified by those Delta data sets that have been specified by the operator or by card reader input. (No Delta data sets need be specified, in which case no modification will take place.) The Link-loader can then be divided into the following three logical sections:

SECTION ONE is used for processing Initial Virtual storage, Resident Supervisor and the Resident Support System.

Initial housekeeping is performed upon entry. Base registers are initialized, and CCW addresses are relocated during Initial Virtual storage processing.

Throughout IVM processing, whenever a reference is made to a PMD group, a PMD, or a CSD, a test is made to determine if paging has been done. If so, PAGTDY is called to insure that the page is in main storage before processing continues. Whenever a real storage address is specified, it could be a virtual real storage address, in which case paging could be done.

If Delta data sets are to be loaded, the address of the Delta volume and names of the Delta data sets must be supplied by the operator or by input from the card reader. If card input is used, cards are read until an end-of-file condition occurs or a card with END in columns 1-3 is encountered. Each name is checked for the required format. If a name is valid, it is entered into one of three lists. When the lists are complete, they specify the hierarchy of search for data sets in Initial Virtual storage, the Resident Supervisor and the Resident Support System, respectively.

The volume label and PAT pages are then read from the Delta data set volume. A check is made to see if the specified Delta data sets exist on the volume. For each of the data sets, the following procedure is carried out.

The DSCB is read into DSCBF, and the EXTENT routine is called to pick up the external page numbers on which the data set resides. An internal table, EXTAB, is then created, consisting of 6-byte entries containing extent information. The number of Partitioned Organization Directory (POD) pages is found in the format-E DSCB for

each data set. Buffer space is allocated for these pages, and the starting address of the POD is stored in the CCW list for reading the POD. Then, for every page of POD, EXTAB is searched to determine the location of the page. The first POD page is virtual page 0 of the data set. (These POD pages temporarily occupy a portion of the area later used for the TDY.) The first page is read in, and the CCW is updated to point to the next higher page. If the POD is missing or incomplete for a given data set, a diagnostic message is issued. If no POD can be found (System or Delta) comprising Initial Virtual storage, the Resident Supervisor or the Resident Support System, Startup is terminated; otherwise Startup continues.

LOADL is called to perform the following function for processing Initial Virtual storage, the Resident Supervisor and the Resident Support System. Beginning with the POD for the first data set in the hierarchy, a search is made for the load list member descriptor. The search continues sequentially through the hierarchy until a load list is found. Now the address of the input buffer is located from INPTAD, and moved to the CCW list used to read the load list CSECT text pages. LOADL computes the size of the buffer area for the load list text and saves the starting address in LDTBL. The list contains only the names of the CSECTs to be processed by the Link-loader. A hierarchical search through each data set is conducted for each CSECT name in the load list. The PODs are searched in hierarchical order until all CSECT names are located. Then the hierarchy number of the data set in which a CSECT is located is entered in the load list entry for that CSECT. The first virtual page number of the module in which the CSECT is located is taken from the member descriptor in the POD, and also placed in the corresponding load list entry for that CSECT. For quick access to any name in the load list, a hashing table is created by the routine LLLNK.

Note: For LOADL processing of user modules, the module name, or any CSECT name in the POD, is sufficient to cause all CSECTs in the module to be loaded.

Next, the routine BGNTDY is called to initialize the TDY heading area. More pages are allocated for Initial Virtual storage heading than for the Resident Supervisor/Resident Support System heading, due to Initial Virtual storage's greater size. The pages provided are contiguous, with area set aside for the paging of TDY, and the starting location of the Task Dictionary table (TDY) is stored in location TDYAD. Since the TDY for IVM may exceed the main storage limits, Startup pages all the TDY in excess. The number of resident TDY pages varies with the configuration as follows:

2 storage boxes	64 pages
3 storage boxes	128 pages
4 storage boxes	288 pages

All TDY pages in excess are written on the drum and paged as necessary during system operation. The page table entries describing each of these pageable TDY pages consist of 16 bytes:

Bytes 0-3 -- virtual page address

Bytes 4-7 -- number of pages in PMD group

Bytes 8-11 -- external address on drum

Bytes 12-15 -- virtual address of beginning of PMD group

Then the BLDTDY routine is called to build the TDY. This routine performs another scan of the load list entries. Program module dictionaries (PMDs) for all entries in the load list are linked into the TDY. (Any individual PMD may be applicable for more than one CSECT.) For user modules, any CSECT in the PMD that is not in the load list has a load list entry generated for it at this time. For each load list entry, the following information is incorporated:

- A pointer to the Control Section Dictionary (CSD).
- An updated virtual page number (reflecting the displacement of the CSECT within the module).

At this point, DSNAM is checked to see if RESSUP is being processed. If so, then DSNAM is modified to indicate that RSSSUP is to be processed concurrently with RESSUP. The same process of building a Delta data set name list, locating the load list, and locating each CSECT in the load list is followed for RSSSUP. The PMDs for RSSSUP are linked into the same TDY as those for RESSUP. Therefore, when FIXPMD is invoked, all cross-references between RESSUP and RSSSUP modules are resolved.

In the FIXPMD routine, standard entry points and complex definitions are resolved for each PMD.

Next the MAPGEN routine is called to create the Memory Map table in the TDY heading area. These entries are created in ascending sequence of CSECT base addresses.

SECTION TWO is used for processing only the IVM data set(s).

Two previously created buffers are used to process the initial virtual storage page tables: XTSI buffer and Shared Page table buffer. First the segment and page tables are created. The Segment table (ST) contains: a full word entry per segment, the displacement (relative to the XTSI origin) of the segment 0 Page table (PT), and a PT availability indicator, a 0 in bit 31. Segment 0 is used to store private CSECTs (containing information pertinent to a specific user or task). The TSI and the XTSI are used to keep track of specific tasks. Segment 1's entry contains the number of pages in the segment and the PT availability indicator, a 1 in bit 31. This is a shared segment, and its page tables reside in real storage. If there is public or private segment overflow, additional Page tables are created and their addresses are placed in the corresponding Segment table entries. CSECTs in the shared segments are public CSECTs, so their page tables are applicable for all users.

Next, the Auxiliary Segment table (AST) is created with a doubleword entry per segment. Segment 0's entry has a bit set in the flag byte, indicating that the segment is assigned. The entry for segment 1 has bits set to indicate the segment is assigned, shared, and that the page tables are in main storage. The Shared Page table number is stored in the fifth byte. The entries for the remaining segments (2-15) are zeroed. Every time a page is assigned a segment 0 virtual address, a Page table and an External Page table (XPT) entry is made in the XTSI. For every virtual address assigned to segment 1, an entry is made in the SPT and XSPT. The LOCXPT routine is invoked to create dummy page table entries, in the event there is an odd number of pages in the segment. This also returns the origin of the XPT or XSPT, as well as the address of the location following the XPT or XSPT. FORMPT is then called to create PT and XPT entries for segment 0, or SPT and XSPT entries for segment 1. The same procedure is followed for additional public or private segments. Then Linkloader sets the 'update in place' flag in the XSPT entries of all the SDAT pages. Construction of the PT/XPT is accomplished via the XTSIRT routine. Construction of the SPT/XSPT is accomplished via the SHPTRT routines. These two routines are designed to facilitate the construction of Page tables for overflow segments.

NAMLOC is called to locate the virtual storage addresses for a list of selected IVM entry names. These are found in NAMTAB2, and their addresses are stored in the communication region. GETEXT is used to read the text for each CSECT in the TDY storage map. It then modifies the text in the buffer, packs it into the the output buffer, and writes it on the paging drum (in the case of the public segment), or the paging disk (in the case of the private segment). If a print map is requested, SYMGEN and SORDID are invoked to print a storage map of Initial Virtual storage. RELTDY is called to convert all the real addresses in the TDY to virtual addresses. The private segments of Initial Virtual storage, and the associated tables, are now written onto the paging disk. Next, WRTDY is called to write out the TDY pages on the paging disk, with the exception of those allocated pages not being used. (The XTSI is written on the paging drum.) As each page is written, its external page number is entered in the appropriate XPT entry by the SETPT routine. Zeros are stored in the first word of the XPT entry for any unused allocated pages. The following fixed areas in the XTSI are now initialized:

A PSW
The estimated time
The number of bytes available in the first page
The XTSI page count
The current timer value
The user timer value

The public segments are now written onto the paging drum (for minimal access time). Any of the allocated pages that are not used are not written. The SETPT routine is invoked to enter the external page numbers in the appropriate XSPT entries. Zeros are stored in the first word of the XSPT entry for unused pages. The 'update-in-place' flag is set for XSPT entries in the SDST.

The real storage block addresses for the first pages of CHBTCT and CHBBFP are placed in their respective Shared Page Table entries. The 'page hold' flag is then turned on in the XSPT entries for the first pages of CHBTCT and CHBBFP. These are IVM CSECTS (tables) used by RTAM, but the first page of each must reside permanently in real storage (that is, they must not be paged out). See the section 'RTAM Initialization'.

SECTION THREE processes the RESSUP and RSSSUP data sets.

The NAMLOC routine is invoked to find the real storage addresses for the list of RESSUP entry names (NAMTAB). These addresses are stored in the Startup communication region.

SYMGEN is now called to create the combined RESSUP/RSSSUP Symbol table. If a RESSUP map has been requested it will be printed at this time. RCOMTB is called to create the RSS Communication table. ADDPGS is called to add the RESSUP/RSSSUP Symbol table pages to shared IVM. WRSYMTB is

called to write the Symbol table onto the external pages reserved for it. These pages are located on the paging drum. RELMEM is called to release the Symbol table buffers and GETEXT is called to read in the text for each CSECT in the TDY storage map. It modifies the text, packs it into the output buffer, and then moves it to a preassigned location in main storage or the paging drum. The PSA modules, however, remain in the buffer PSA-BUF, and if the PPV is a drum, all drums have the SERR/Reconfiguration modules written on them.

The address of the RSS Communication table is inserted into the System table field SYSRCT. BDSDST is called to initialize the Shared Data Set table and create a member entry for the RESSUP/RSSSUP Symbol table. The SDST is then written on the paging drum as a part of shared IVM. Buffers used by the Link-loader are now released and control returns to Startup mainline.

The following describes the Link-loader subroutines.

LINK-LOADER SUBROUTINES

Create Extent Table (EXTENT)

This subroutine is called by Link-loader subroutines to create a table of extents from format-E and format-F DSCBs of a data set. On entry, the following fullword parameters are passed:

- Next available location in EXTAB.
- Total number of pages in the data set.
- Location of first external page entry in format-E DSCB.

For each extent field, an entry of three halfwords is created in EXTAB:

- First external page number.
- First virtual page number.
- Last virtual page number.

After all extent fields have been processed, an entry of all ones is created to mark the end of the EXTAB. Control is then returned to the caller.

Load and Process Load List (LOADL)

Chart AC, Page 4

This subroutine scans sequentially through the PODs that comprise IVM/RESSUP/RSSSUP to find the load list member descriptor. If a load list cannot be located, a diagnostic message is issued and Startup is terminated. The module name is CHBVM during the IVM processing, CHBRC during RESSUP processing, and CHBR5 for RSSSUP processing. The routine allocates storage for the load list and stores the buffer origin in LDTBL. Next it reads the text into an input buffer, and moves the text to LDTBL.

A load list entry is five words in length, and contains the following information:

- CSECT name (8 bytes).
- Flag byte 1 (reserved).
- Flag byte 2 (=X'80' for page boundary control, =X'40' for read-only flag, =X'10' for IVM pages SETXP allowed, =X'20' for user modules).
- Virtual page number of PMD (after the PMD is loaded, these two bytes contain the addresses of the first virtual page of text).
- Pointer to CSD for the CSECT.
- Hash chain pointer.
- Function byte (for selective loading).

Two words of all ones mark the end of the load list.

Begin Load List (BGNLL)

Chart AC, Page 4

This subroutine performs the following functions:

- Reads in one POD at a time (according to hierarchy).
- Links to LLSCAN. After returning from LLSCAN, each entry for which a corresponding name was found in the POD will have its virtual page number and hierarchy fields initialized.
- Hashes and links all load list entries except those for which no matching name was found (LLLNK subroutine).

Scan Load List (LLSCAN)

Chart AC, Page 4

For each POD in the hierarchy that is read, this subroutine performs one scan of the load list, and searches the POD for a CSECT name corresponding to each CSECT name in the load list. If a CSECT name was found on a previous scan, the entry is

bypassed. When an alias is found for that CSECT in the POD, the related member descriptor is located, and the first virtual page number of the member is moved into the LLVRPN field of the load list entry. The hierarchy number of the POD in which the CSECT was found is moved into a field of the load list entry for that CSECT name. For user modules, the module name or any CSECT in the load list is sufficient.

On the last scan performed by this subroutine, the following procedure is performed for each CSECT name not found:

- A diagnostic is issued.
- The CSECT name is blanked out in the load list.

Begin Task Dictionary Table (BGNTDY)

This subroutine performs the following functions:

- Allocates storage for TDY heading area, and stores the starting address of the TDY in TDYAD.
- Initializes the fields in the TDY heading area.

Build Task Dictionary Table (BLDTDY)

Chart AC, Page 5

This subroutine performs one scan of the load list when processing a RSSSUP or RESSUP CSECT. When processing an IVM CSECT, three scans are made of the load list, one for CHBTCT, one for CHBBFP, and one for the remainder of the IVM control sections in the load list. TCT and BFP must be processed first, since they reside in virtual storage, starting at the first page of the first public segment in IVM. If a CSECT's PMD has not been loaded, and the CSECT is not from a user module, the subroutine: links to LDPMD in order to load the PMD; links to UPDLL to update the virtual page number field (which will now reflect the displacement of the CSECT within the module); initializes the pointer to the CSD; and links to ALLOC to allocate storage for each CSECT in the load list. After the above scans are complete, three scans are made for user modules, if any exist. The modules are handled as above.

After the load list scan is complete, control returns to Link-loader mainline.

Load PMD Into TDY (LDPMD)

This subroutine performs the following functions:

- Links to READIN, to read in the PMD page(s).
- Stores the hierarchy number in the first byte of the PMD. (If RSS deltas are present, the hierarchy number is first incremented by the number of RESSUP deltas present.)
- Links to ADDPMD, to allocate storage for a PMD in the TDY, and begin initialization of the PMD preface.
- Performs further initialization of fields in the PMD preface.
- Moves the PMD into the space allocated for it in the TDY.

Update Load List (UPDLL)

This subroutine searches the load list for names corresponding to each CSECT name in the PMD. If a matching load list name cannot be found, a test is made to see if this is a user module. If it is not, the CSECT is not processed and a diagnostic message is issued indicating that that CSECT is missing from the load list.

If the CSECT is in a user module, a load list entry for the CSECT name is added to the end of the load list, and the new load list entry is processed.

If a name is found that has already been processed (because it appeared in a previous PMD), the CSECT is bypassed. If a CSECT name is found but the hierarchy number does not match that of the current PMD, the CSECT is also bypassed.

If a valid matching name is found, the pointer to the CSD is initialized in the load list entry, and the virtual page number field is updated to contain the CSECT's first virtual page number.

Storage Allocation for IVM and RESSUP (ALLOC)

ALLOC is called by the BLDTDY subroutine for each CSECT name in the load list for which the corresponding PMD has not been loaded. It allocates real storage for CSECTs of the RESSUP data set, and virtual storage for CSECTs of the IVM data sets. ALLOC also resolves relocatable and absolute definitions, via the LINK subroutine.

IVM and RESSUP CSECTs are handled differently by ALLOC, as follows:

IVM: Control sections are assigned virtual storage addresses solely on the basis of their attributes, with the exception of the ISA module, which is allocated a virtual address of zero. Private CSECTs are

assigned virtual addresses starting with segment 0. Public CSECTs are assigned virtual addresses starting with segment 1.

If a CSECT has its page boundary flag set, it is automatically aligned on a page boundary. If the page boundary flag is not set, the CSECT is assigned a starting address at the next doubleword boundary within a page, if it does not overflow that page. Otherwise, it is aligned on the next page boundary. (The only exception to this is for the CHBMTS CSECT (Multiterminal Status Control Block), which is assigned a starting address at the next 64-byte boundary.) Paging operations are minimized by this set of rules, since no CSECT will overflow a page boundary unless its length is greater than one page.

The following rules are followed in assigning protection keys for each load list entry: If a CSECT is privileged, it is assigned a protection key of X'0F'. Nonprivileged CSECTs that are read-only are assigned a protection key of X'0A'. CSECTs that are neither privileged nor read-only are assigned a protection key of X'05'.

Note: If a user module is privileged or has the system attribute, a message is issued to the printer by ALLOC and the privileged attribute is ignored.

RESSUP: The control sections in the PSA page (PSA, CST, Recovery Nucleus, Intercom, SERR Bootstrap) are assigned addresses in page 0. The SERR/Reconfiguration modules are assigned values, simply for sorting and identification purposes. In all other cases addresses are assigned in lower main storage starting at C(LOWAD). Provision is made for allocating RESSUP text around unavailable main storage by calling the GETMEM routine each time a CSECT requires main storage outside the last assigned page. If the limits of available main storage are exceeded, a diagnostic message is sent to the operator. The attributes are not inspected during RESSUP processing.

When processing a RESSUP load list entry, a check is made for the presence of a read-only flag. The presence of this flag indicates that this CSECT and all subsequent RESSUP CSECTs are read-only and may be overlaid by RSS. The address of the CSECT that contains the read-only flag is saved for the RCOMTB routine. The address that is saved is actually the address of the first page boundary following the beginning of the CSECT. CSECTs in RSSSUP are assigned virtual addresses starting with the beginning of segment 2 of virtual storage.

In either case, RESSUP or IVM, ALLOC sets the PMD link in the CSD headings and links all relocatable and absolute definitions by calling the LINK routine. These are linked into one of the TDY hash tables. Once all the CSDs have been processed in a PMD, ALLOC returns control to the LDPMD routine.

Process Complex Definitions in PMD (FIXPMD)

This subroutine processes standard entry point definitions and complex definitions associated with a PMD. The module name (SEP) is posted in a hashing chain by the Hash subroutine. The CSD link of the module name DEF is set to all ones. In this way, a SEP will never be linked to an undefined complex DEF. A search is made for the CSD associated with the SEP of the module. Its attributes are examined. The first CSD having the PROTOTYPE (PSECT) attribute becomes the SEP CSD. If there is no PSECT, the first CSD becomes the SEP CSD. Its address is saved for later processing. The LINK subroutine is called to link the complex DEFs in each CSD into the hash chains. The CSD links of complex DEFs are set to all 1s. MODFY processes the complex DEF modifier for each PMD page. The CSD links of complex DEFs are updated to point to the defining CSD, and complex DEF processing is complete. FIXPMD now picks up the address of the SEP CSD. If there are no modifiers for the SEP, the V-value and R-value of the first DEF in the SEP CSD are stored in corresponding entries of the module name DEF. The V-value of the module name DEF is initialized by the MODFY routine as it processes the SEP modifiers. The CSD link is now set to point to the SEP CSD. This is the CSD of the CSECT whose base defines the R-value of the DEF. If a CXD-ref is present, its value is set to the current CXD value. Control is now returned to the caller.

Modify PMD and Text Pages (MODFY)

This is a subroutine to process relocation modifiers and perform modifications on PMD or text pages. It modifies PMDs when processing standard entry points or complex definitions. It modifies text when processing external or internal references. MODFY picks up, analyzes and acts upon the relocation modifiers in the PMD. A relocation modifier contains the following information:

- Number of bytes to modify.
- Number of reference whose V- or R-value is used to modify the PMD or text.
- What operation to perform.
- Starting byte in page to be modified.

MODFY receives information from the caller. Each parameter is passed in a general register.

- Number of modifiers for page to be modified.
- Address of first modifier for the page.
- Address of CSD (or PMD).
- Address of PMD preface.
- Pointer to the page.
- Address of a reference table.

If a reference is encountered and has not been defined and it is not a Q-ref or a CXD-ref, DEFINE is invoked to initialize the V- and R-values before the reference can be used to modify PMD or text. Using the relocation modifier, MODFY now picks up the bytes of PMD or text, and adds or subtracts the V-value, or substitutes the R-value of the reference. The modified field is then returned to its original location. Only one page can be processed at a time so at the end of each page, MODFY returns to caller.

Compute and Link Defs into Hash Chains (LINK)

This subroutine computes the value of a DEF or a Q-ref and calls HASH to post the DEF or Q-ref on the appropriate TDY hash chains. On entry, LINK expects:

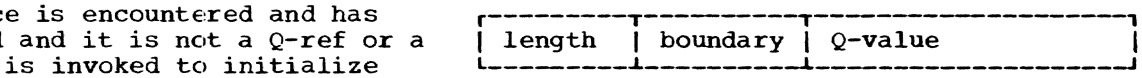
- the CSECT address,
- the corresponding CSD pointer,
- the first DEF or Q-ref pointer,
- the number of DEFs or Q-refs, and
- an indication of the type of DEF (relocated, absolute, complex, or Q-ref).

LINK also sets up and maintains a hole table associated with the Q-ref value it calculates.

On entry, the post indicator for HASH is set on. If the DEF is a user module, the double posting flag is set on for HASH. If this is not a Q-ref call, the V- and R-constants are increased by the CSECT base address, if necessary, the CSD link is set (for complex defs, the CSD link is set to 1s at this time), and HASH is called to post the DEF on the appropriate chains.

If this is a Q-ref call, HASH is called to search to see if a Q-ref by this name has already been posted. If one has, HASH processes the Q-ref (see below) and the next Q-ref is examined. If no Q-ref with this name has already been posted, the attributes of the current Q-ref are examined to see if the Q-ref will fit in any of the holes, if there are any, in the Q-ref value table. If a fit is possible, the Q-value is placed in the Q-ref, the hole is marked filled, and HASH is called to post the Q-ref.

If a new entry is needed, the next available location is adjusted to the proper boundary. If a hole is created, the value of the hole is placed in the hole table. A hole table entry is one word long and has the format:



The first byte contains the length of the hole; the second byte is the boundary value (X'01' = byte boundary, X'02' = halfword boundary, X'04' = fullword boundary, X'08' = doubleword boundary); the last two bytes contains the Q-value of the start of the hole.

The CXD value is updated by the attributes of the current Q-ref and HASH is called to post the Q-ref value on the Q-ref chain.

Hash Routine (HASH)

This is a subroutine used to post or to search for a definition in the TDY. On entry, the address of a REF or DEF, the address of the PMD preface, and the function code are passed as parameters. A REF address and function code of 0 are sent if the routine is to search, and a DEF address and function code of 1 are sent if the routine is to post. If double posting is required, the high-order bit is set on. A ref address and function code of 2 is sent if it is a Q-ref call, with the high-order bit on for a queue search and off for a queue post. If the first half of a REF or DEF name is 0, the module sequence number, obtained from the PMD preface, replaces it. The name is hashed, and the hash value is multiplied by four. If the function is to post, the hash value is used to search for a 0 search link in the hash chain based on the System Hash table. Once found, the address of the DEF is stored in the search link. If the chain is empty, the address of the DEF is stored in the hash table and the search link of the new DEF is zeroed. If double posting is required (that is, posting of the DEF on both the system non-privileged and user hash chains), the user hash chain is then searched; if the chain is empty, the current DEF value is stored in the chain. If another DEF has been

posted with the same name as the DEF to be posted, an error message is issued.

If the function is to search, the hash value is used to search for a definition in a hash chain based in the System Hash table. If the DEF is found, its address is returned in a parameter register; if not, a zero code is returned in the register.

If the call is for a Q-ref, the hashed name is searched for on the Q-ref chain. If the name is not found, the Q-ref is posted either directly into or at the end of the Q-ref hash table chain. If the hashed name is found, the new Q-ref is posted at the end of the same hash chain. A test is made to see if the attributes of the two Q-refs agree. If they do not, a diagnostic message is issued and the attributes of the first Q-ref are assumed. The value of the original Q-ref is given to the new Q-ref and control returns.

If paging is needed while chaining a search or post, the first time through a minor paging buffer is allocated for the size of the major buffer. Each time paging is indicated, PAGTDY is called with a minor buffer indicator, ensuring that the CSD is in main storage but does not overlay the major buffer.

Initialize Reference Entries in CSD (DEFINE)

This subroutine is used to find the definition that resolves a reference, and to initialize the V-value, R-value, and CSD link of the reference. On entry, the address of the CSD (or PMD) of the reference, the address of the PMD preface and a pointer to the reference are passed as input. DEFINE invokes HASH to supply the address of the definition that resolves a reference. Once found, the V- and R-values and the CSD link of the definition are moved to the corresponding words of the reference. Should a noncomplex definition be undefined, a zero code is returned and a diagnostic message is issued. If a complex definition's CSD contains all ones, it is considered undefined and a diagnostic is issued for it. The V- and R-values of undefined references are set to all 1s, and the CSD link is set to the address of the undefined reference's CSD. If a reference paired with a complex definition is defined by a complex definition, a warning is issued. The user count in the definition's CSD heading is incremented by one and control is returned to the caller.

Locate Name in TDY (NAMLOC)

This subroutine searches the TDY for a given CSECT or entry point name and returns its address to the caller. The routine hashes the name supplied as an entry parameter, searches for it in the corresponding hashing chain in the TDY, and returns its address. If the name cannot be found, a diagnostic message is issued and control is returned to the caller.

Create TDY Storage Map (MAPGEN)

This subroutine creates the Storage Map table in the TDY with entries in order of ascending CSECT base addresses. Every Control Section Dictionary (CSD) in the TDY is located and the corresponding CSECT address is stored as an entry in the Storage Map table. These addresses are stored at the point where the CSECT address is greater than that of the preceding entry and less than that of the following entry. The CSECT address is stored in the first word of a doubleword entry, the CSD address in the second word. If a map is to be printed, main storage is obtained for building a temporary map. The start of the map is saved and the first word of each page points to the next page. The pointer on the last page is all ones. A temporary map entry contains the following information for each CSD in the TDY:

- Pointer to the CSECT
- Pointer to the CSD
- Version ID (8 bytes)
- Hierarchy number (1 byte)
- Unused (3 bytes)

A count is kept in the TDY heading of the number of Storage Map entries. Should the count exceed the maximum, an error message is issued and Startup terminates. Otherwise, control is returned to the caller.

Locate XPT or XSPT Origin (LOCXPT)

This routine computes the origin of the External Page table or the Extended Shared Page table, which is aligned on a fullword boundary following the Page table or Shared Page table for the source segment. The calling program passes: an indicator as to whether or not the routine may need to create dummy entries, the number of pages in the segment, the type of page (regular or shared), and the Page table origin. The end of the PT or SPT is calculated from the number of pages in the segment. If the number is even, the XPT or XSPT begins in the next location. If it is odd, a dummy entry is created. If the indicator specifies only the locate option, no dummy entries are created. The end of the XPT/XSPT is calculated from its number of pages, and again, if odd, a dummy may be created, depending upon the caller's option. LOCXPT

then returns to the caller with the starting address of the XPT or XSPT, and the address of the location following it.

Form Page Table (FORMPT)

This subroutine is called to create a Page table and External Page table entry for all pages of IVM in either a private or shared segment. For the private segment, the PT and XPT form a part of the XTSI and are written onto the paging drum. The Shared Page tables (SPT and XSPT) reside in real storage. Their addresses are stored in the Resident Shared Page Index table (RSPI).

On entry, an indicator is passed to FORMPT telling: whether the segment is private or shared, the number of pages in the segment, and the PT origin or XPT origin. For each page in the given segment, the PT or XPT entry is set to unavailable, with a main storage address of 0. An XPT entry for the private segment, or an XSPT for the shared segment, is now created for each page. The entries contain certain bit settings:

preferred paging device = drum

type = program

availability = assigned

In a shared segment, the shared and auxiliary flags are set on and the GQE pointer is set to 0 in each XSPT entry. Control is returned to the caller.

Load and Modify Text (GETEXT)

GETEXT is invoked three times by the Link-loader: once when processing IVM, once for RESSUP and once for RSSSUP. The routine reads the text pages from the IPL volume into an input buffer, modifies text according to information in the CSD for each module, packs it into the output buffer, and then moves it either into a previously assigned storage location (RESSUP) or to the paging volumes (IVM or RSSSUP).

The TDY has previously been set up. The Storage Map in the TDY, containing the load address for each CSECT (or PSECT) and a pointer to the associated CSD, determines the loading order. Text is processed according to ascending CSECT base addresses in the storage map. The member descriptors in the POD are used to locate text pages on the IPL or Delta volume. If the member descriptor cannot be found, a diagnostic message is issued and the entry is bypassed.

After the text is read and, if necessary, modified in the input buffer, it is moved into the output buffer on doubleword or page boundaries according to the address in the storage map. Once the output buffer is ready, it is written on the paging drum for shared IVM and RSSSUP, on the paging disk for private IVM, or moved to a real storage location if RESSUP.

When processing IVM pages, the external page number is stored in the XPT, if private, or in the XSPT, if shared. The ISA is the first page of IVM text to be processed.

When processing RSSSUP pages, the external page number is stored in XPT2 of the RSS Communication table.

Various tables are initialized before they are written on the paging volume. When CSECTs in the name lists (CSECTs referred to by later portions of Startup) are processed, the number of external pages used by them on the paging volume, and the addresses of their entries in the XSPT and XPT, are saved in the Startup communication region.

As GETEXT processes RESSUP, the PSA modules are taken first. PSA modules are saved in a Startup buffer, and are not loaded for the initially-loaded CPU until later on in Startup.

The SERR/Reconfiguration modules are written on all drums in the system, and are not loaded into main storage. They are the last modules to be processed. A provision is made for loading RESSUP text around unavailable main storage pages. Control is returned to the caller.

Relocate TDY Entries (RELTDY)

This subroutine is invoked to convert the Task Dictionary table entries from real storage addresses (RSAs) to virtual storage addresses (VSAs). It is the first routine of two (the other being RELTAB/RELTBX) called to complete the conversion.

RELTDY must first complete the Relocation table (TDYTAB). On entry the first two words of a three-word group (first RSA, last RSA) are complete. RELTDY must provide the relocation factor to convert RSA to VSA, word 3. After the entire table is complete, two full words of ones are set following the last group.

Next the input parameters are determined for each call to RELTAB/RELTBX. The routine RELTAB or RELTBX performs the actual relocation on the following entries:

1. Pointers in the system privileged, system nonprivileged, and user hash tables (RELTAB).

2. Control Section Dictionary pointers in the TDY storage map (RELTAB).

3. PMD group headers (RELTAB).

4. Each Standard Entry Point (SEP) CSD link (RELTBX).

5. Each SEP search link (RELTBX).

6. PMD preface link to next PMD preface (RELTBX).

7. All SEP reference CSD links (RELTAB).

8. Each PMD link in CSD heading (RELTBX).

9. All Definition CSD links (RELTBX).

10. All Reference CSD links (RELTBX).

11. TDY heading link to PMD group (RELTBX).

12. TDY heading pointers to System Hash Table, User Hash Table, and storage map (RELTAB).

13. Pointers in the dynamic loader Q-ref hash table (RELTAB).

If, during address conversion, RELTDY finds that conversion will cause private segment overflow, the three-word entry in TDYTAB containing pages in both segments is split into two three-word entries. The first three-word entry is for the end pages in the original segment. The second entry is for the first pages in the overflow segment. Each entry in the TDYTAB corresponds to a section of the TDY which is located on contiguous pages in main storage.

If paging has been done, the last entry will be contiguous virtual real storage addresses with the high-order bit on.

Once control is returned from the RELTAB/RELTBX routine and the final addresses have been converted, control is returned to the caller.

Relocation Table Processing (RELTAB, RELTBX)

This subroutine is called by RELTDY to complete the conversion of RSAs to VSAs for the TDY. There are two entry points, RELTAB and RELTBX. Entry at RELTAB will relocate a series of TDY addresses, the number of which is furnished as a parameter. Also passed is a pointer to the first address to be processed, and a length value, indicating the increment to the first address to reach the second, and so on. Only one TDY address is processed by RELTBX and the address of that entry is passed as a parameter.

The TDYTAB is a table of one or more three-word groups. If the TDY pages are loaded contiguously in main storage, there is only one group in TDYTAB. The first word of each group has the real address of the first page of the group, and the second word contains the real address of the last page of the group. The third word contains the logical relocation factor to be applied to each page within the group. There will be a group for every set of noncontiguous pages in the TDY. The end delimiter of TDYTAB is two full words of all 1s.

Processing of RELTBX and RELTAB is identical, except that RELTBX relocates only one page, while RELTAB relocates the number of pages it receives.

The following steps are taken:

1. The TDY RSA parameter is inspected. If 0, control goes to step 3; no relocation is necessary. If not 0, relocation is necessary and the first three-word group of TDYTAB is inspected to see if a match can be found within the group which corresponds to the RSA. There will always be at least one group in TDYTAB. If the address is within the group, control passes to Step 2. If not, the next TDY entry is inspected until the right group is found.

2. Once the proper TDYTAB group is located, the routine relocates the RSA by the value in the third word of the group, and the new value, now a VSA, is stored back into the TDY.

3. If there are any more RSAs requiring relocation, control is sent back to step 1 and the processing repeated for all given RSAs. When all RSAs have been relocated, control returns to the RELTDY routine.

Set External Page Number in XPT/XSPT (SETPT)

This subroutine is called to set the corresponding external page number or head and slot number in the XPT or XSPT entry for a VMA. It is called whenever a page of text, previously assigned a virtual storage address, is written on an external page of the paging volume. The protection key for each page is set according to the value in the load list entry for the CSECT to which this page belongs. On entry, two Startup cells contain certain input information:

STARTAD = virtual storage address.
XPGNO = external page number.

First, the segment number is determined from the virtual storage address. The

length of the segment and the starting address of the PT or SPT are used as parameters to invoke the subroutine LOCXPT, which locates the starting address of the XPT or XSPT. When control is returned to SETPT, the entry of the page containing the virtual address is located by searching the XPT/XSPT entries arranged in ascending virtual address sequence. The external page number (disk), or head and slot number (drum), and protection key are stored in the entry, and control is returned to the caller.

If RSSSUP is being processed, entries are made in XPT2 of the RSS Communication table. The protection key is not set for RSS pages.

Read Page From IPL Volume (READIN)

This is the subroutine called to read a page of a data set from the IPL or Delta volume. It searches EXTAB for the extent containing the virtual page number which was supplied as a parameter. It calculates the corresponding external page number. If the virtual page number cannot be found in EXTAB, a diagnostic message is issued and READIN sets an error code and returns to the caller. Otherwise, a subroutine is called to convert the external page number to a 2311 or 2314 CCHHR. The CCHHR is set in a CCW area to read a page into the Startup input buffer. READIN sends the address of the IPL or Delta volume to EIAA2, which builds and executes the channel program, checks for I/O errors and returns. The caller's virtual page number is incremented by one, and control is returned to him.

Delta Data Set Routine (DELDS, DELTBL, DELBTB)

Chart AC, Pages 3 and 4

This subroutine asks the operator (via OPER) if a Quickstart data set is to be built. The operator responds with the device address of the volume on which the Quickstart data set is to be built. The response is checked to make sure that the volume specified is VAM2-formatted. If so, parameters are initialized to indicate to the Link-loader (EIAA5) that a Quickstart data set is to be created after link-loading IVM and RESSUP.

DELDS then asks the operator or reads a card to determine if Delta data sets are to be loaded. If the system is not to be modified, the operator or card reply is 'N' and Startup proceeds to load Initial Virtual storage, the Resident Supervisor and the Resident Support System from the system data sets on the IPL volume.

If the reply is 'Y', DELDS obtains buffers for reading in a Delta data set name list and for building a Delta data set name list. The address of the Delta volume and the Delta data set specification are supplied by the operator or by card input. If card input is used, cards are read until an end-of-file condition occurs or a card with END in columns 1-3 is encountered. Then DELDS checks to see that the device address is the proper length, and that the characters in the data set specifications are valid. (The address of the Delta data set volume must be different from that of the IPL volume and the paging disk.)

Next, DELDS obtains the volume label and PAT from the Delta data set volume. The DELTBL routine is invoked to build a table of Delta data set names.

If the operator has specified 'ALL' as the data set specification, all appropriately qualified Delta data sets on the volume will be searched prior to the system data sets on the IPL volume. The list of names of data sets to be searched will be generated in the order in which they appear on the DSCB pages.

If individual data set names were supplied (as the specification), each is checked for the required format. If correct, each is entered into the IVM, RSSSUP or RESSUP list in the order specified by the operator or by card input.

If all 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 by the operator. If a specified Delta data set name does not appear among the DSCBs on the DSCB pages on the Delta data set volume, a diagnostic message is issued and the name is not entered in the list. If no DSCBs can be found (system or Delta), comprising Initial Virtual storage, the Resident Supervisor, or the Resident Support System, Startup is terminated; otherwise Startup continues. If one or more Delta data set names are (explicitly or implicitly) specified more than once, a diagnostic message is issued, and only the first specification is entered into the list.

The system data sets TSS*****.SYSIVM, TSS*****.RESSUP, and TSS*****.RSSSUP are always placed last in the list. The DELTBL routine is invoked to make entries into the data set table (DSTBL). As entries are made into the DSTBL, the external pages of that data set are entered into the extent table (EXTAB) via the EXTENT subroutine.

Initialize the XTSI and Page Table Pages (XTSIRT)

The XTSIRT routine is called to create an External Task Status Index (XTSI) for the Main Operator's Task. First, the fixed format portion of the XTSI is initialized. This entails setting up the following fields:

- the cold start PSW
- estimated time
- user time value
- XTSI page count

Next, the Segment table is initialized. This table begins on the first eight-word boundary following the fixed portion of the XTSI. The Segment table entry for segment 0 is set to the relative origin of the Page table (PT) within the XTSI page. The Segment table entry for the shared segments is initialized by the Shared Page Table routine.

Now, the Auxiliary Segment table (AST) is initialized. The 'assigned' flags are set in the AST entries for segment 0 and segment 1, indicating that these segments have already been allocated.

Next, the Page table (PT) and External Page table (XPT) for segment 0 (private) are constructed within the XTSI page. An even number of pages must exist within segment 0. Dummy entries are created within the PT if an odd number of pages exist within segment 0. LOCXPT is called to locate the origin of the XPT within the XTSI page.

FORMPT is then called to create PT and XPT entries for segment 0. If there is an overflow private segment, it is determined whether or not its PT and XPT will fit in the original XTSI page. If they will fit, the PT and XPT are created in the same manner as for segment 0 and the 'assigned' flag is set in the AST entry for that segment. If the PT and XPT will not fit on the original XTSI page, a new Page table page is allocated and a pointer to this page is set in the XTSI. A flag is also set in the AST entry to indicate that the Page table for this segment is located on an overflow XTSI page. When the XTSI is paged out (the PGXTSI routine), the external location of the particular XTSI page containing the Page table for this segment is entered in the appropriate AST entry. When the first additional Page table page is allocated, the XTSI overflow flag is set on.

The Page table page header is initialized next. For each additional Page table page that is allocated, forward and backward pointers are set and the count of the Page table pages in the XTSI is incremented by 1.

Following the Page table page header, the PT and XPT is constructed. Each PT and XPT is preceded by a Page table header. The size of the Page table varies as a function of the number of pages within the segment which have been allocated. Therefore, the block size is set in the Page table header. The 'segment availability' flag is set to active.

If there is more than one private overflow segment, it is determined whether an additional PT and XPT can fit on the existing Page table page. If it can, the count of the number of Page tables in the Page table page is incremented by 1. This field, which is located in the Page table page header, is initially set to 1 whenever a new Page table page is allocated. If an additional PT and XPT cannot fit on the existing Page table page, a new Page table page is allocated. The PT and XPT is then constructed for this segment. A pointer to this Page table page is set in the Segment table entry for this segment and the 'assigned' flag is set in the AST. The segment number is stored in the header, for cross-referencing purposes. Pointers are established to keep track of available space on the Page table page.

Control returns to Link-loader mainline.

Initialize SPT and XSPT for Public Segments (SHPTRT)

This routine locates and forms the Shared Page table (SPT) and External Shared Page table (XSPT) for each public segment. This is accomplished in the following manner.

First, LOCXPT is invoked. The LOCXPT subroutine computes the origin of the XSPT and creates dummy Page table entries in the event that there are an odd number of pages in the segment. FORMPT is invoked next. The FORMPT subroutine creates External Shared Page table entries. All the Shared Page tables reside in main storage.

For each additional overflow public segment (if any), an additional page of main storage is allocated for the SPT and XSPT. The Segment table entry and AST entry are initialized for each overflow segment to reflect the length of the segment and the SPT number. In addition, the 'PT in main storage', 'shared segment', and segment 'assigned' flags are set in the AST for each overflow public segment.

The SPT and XSPT have now been created in buffers. These tables will be relocated later by the CRRSPI routine.

Control now returns to Link-loader mainline.

Write Task Dictionary Table (WRTDY)

This routine writes the Task Dictionary table (TDY) out onto the paging disk. First, the TDY header and map are written. The unused pages of the TDY header are not written. Then, the TDY itself is written out.

As each page is written, its external page number is entered in the appropriate XPT entry by the SETPT routine.

Control returns to Link-loader mainline.

Build RSS Communication Table (RCOMTB)

The RCOMTB routine creates the RSS Communication table (RSSCOM).

A virtual storage area is defined by its page tables. The RSS Communication table defines the virtual storage used by RSS. The Page tables define which external pages are mapped into each segment of virtual storage. RSSCOM resides in real storage following the last Resident Supervisor CSECT. RCOMTB decides whether or not RSSCOM will fit on the last Resident Supervisor page. If not, an additional page is allocated.

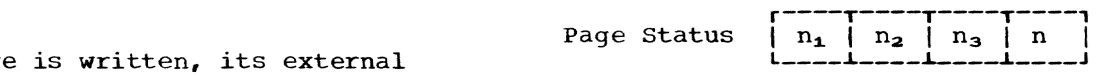
RCOMTB creates Page tables PT2 and XPT2 which are initialized with real storage and external storage addresses of the nonresident portion of the RSSSUP data set. This Page table format is identical to that of the virtual storage Page table format of segment 2. RSS uses this format because the Dynamic Address Translation hardware is active during RSS execution.

Additional Page tables are created in RSSCOM. PT3 and XPT3 are initialized with the address of the RESSUP/RSSSUP Symbol table. (This Symbol table resides on the paging drum as a portion of IVM.)

XPT4 contains as many entries as the sum of entries in PT2 and PT3. The entries are the external addresses of the read-only portion of the Resident Supervisor which must be reloaded after the execution of RSS.

When the nonresident portion of RSS is to be executed, it is loaded into the portion of real storage occupied by the read-only portion of the Resident Supervisor. RCOMTB determines whether there are enough pages in the read-only portion of the Resident Supervisor for the nonresident portion of RSS. If there are not enough pages, RSS is responsible for allocating this room for the remainder of itself.

To facilitate this during system startup, Startup fills as many PT2 and PT3 addresses as it can. It passes this "Page Status" information to a resident RSS control module. "Page Status" information is described as follows:



where

n_1 = total number of Segment Two Page Table (PT2) entries required to page in the transient portion of RSS on a demand basis;

n_2 = number of Segment Two Page addresses resolved by Startup (page addresses of the pageable portions of the Resident Supervisor);

n_3 = total number of Segment Three Page Table (PT3) entries required to page in the Symbol Dictionary on a demand basis;

n = number of Segment Three Page addresses resolved by Startup (remaining page addresses of the pageable portions of the Resident Supervisor.

RSS then computes the number of real storage pages it needs from "Page Status" and pages the remainder of RSS into real storage.

PT0 contains a map of all real storage addresses from virtual page X'00' to X'FF'. PT1 contains a map of all real storage addresses from virtual page X'100' to X'1FF'.

A Segment table is initialized with entries for PT0, PT1, PT2, PT3, and PT4. Each entry contains the length and address of the respective page table. Control returns to link-loader mainline.

Write RESSUP/RSSSUP Symbol Table (WRSYMTB)

The Write Symbol table routine (WRSYMTB) writes the RESSUP/RSSSUP Symbol table on the paging drum. It records the external address in the XSPT and in XPT3 of the RSS Communication table. This table had previously been built by SYSGEN.

Add Pages to Shared IVM (ADDPGS)

ADDPGS adds new pages to shared IVM by expanding the last assigned SPT/XSPT. If not enough room exists in the last assigned SPT/XSPT, ADDPGS creates a new SPT/XSPT. The Segment table and Auxiliary Segment table entries for the public segment involved are updated.

ADDPGS is called in order to add the RESSUP/RSSSUP Symbol table pages to shared IVM.

Build Shared Data Set Table (BDSDST)

The Build Shared Data Set table routine initializes the Shared Data Set table and sets up a member entry for the RSS Symbol table. This member entry is generated so that VSS may calculate the virtual storage address of the RSS Symbol table. Since the RSS Symbol table is placed into virtual storage following the link-loading of Initial Virtual storage, it is not possible to resolve this address in the normal manner. (See FIXPMD routine for normal address resolution.)

The address of the RSS Communication table is computed by VSS from the Shared Page table number and the byte address relative to the beginning of the Shared Page table.

The member entry for the RSS Symbol table (CHBRST) is located by referring to the hashing chain pointer which Startup initializes in the SDST header. A pointer to the SDST is placed in the ISA and the SDST is written on the paging drum. SETPT is called to initialize page table entries for the SDST.

QUICKSTART SUBROUTINES

Read in Quickstart Data Set (QKREAD)

QKREAD reserves pages for the buffers it needs, and then calls RDSCB to locate the Quickstart format-E DSCB on the IPL volume. The recording buffer pages are located within the DSCB, and all three are read in.

For each entry in the recording buffer, up to the first entry of X'FF', the next DSCB location is found (QKRD90), and the page is read from the Quickstart IPL volume (QKRD80R) and written to the primary or secondary paging device, as necessary (QKRD80W).

After IVM is brought back in, if the system is duplex or half duplex, QKREAD checks that the PSAs are in the same location, and the PSA buffer is read back in. The remainder of the pages in the recording buffer, up to the first X'FF' entry, are read into the location specified in the recording buffer (QKRD80R) from the next location in the DSCB (QKRD90), and the RESSUP flag is set in the byte map. If there are any drums, the SERR/Reconfiguration pages are also read in from the Quickstart volume and written on each drum in the system.

The XTSI and page table page buffers are read back from their DSCB location (QKRD90) into the buffer location indicated in the Communication Region.

The buffers are released, and control returns to offer the demounting of the IPL pack.

QKRD80R -- reads from the Quickstart IPL volume to a buffer.

QKRD80W -- writes a buffer. If the device type code in the recording buffer is X'00' (indicating 2301), it will write to the primary paging device. Otherwise, it will write to the secondary paging device.

QKRD90 -- locates the next available relative page number entry for the Quickstart data set.

At any point, if an unrecoverable I/O error occurs, the message ERROR READING QUICK START DATA SET -- QUICK START TERMINATED is issued to the operator and Quickstart terminates. Quickstart also terminates if reading in the Quickstart data set overlays the Page Allocation table on the secondary paging volume.

Quickstart Data Set Creator (CEIAB)

Chart AC, Page 8

The Quickstart routine consists of a mainline routine that calls a number of independent subroutines with specific functions, and a set of common subroutines. Upon entry, the mainline (Chart AC, Pages 8-10) sets up standard linkage and executes initializing steps. Quickstart then links to a routine (DSCBE) to create the format-E DSCB for the Quickstart data set (or to locate the DSCB if the data set already exists). Quickstart then calls subroutine DSC30 to locate and update the DSCB entry to reserve three pages for the recording buffers. DSC30 is called again to locate and reserve pages for Startup and the CCB, and DSC60 is called to write out the CCB pages. The number of Startup pages and the number of CCB pages is recorded in the recording buffer.

Link-edited IVM is then written on the Quickstart volume, beginning immediately after the CCB. For each assigned segment, the external page table (XPT), or the external shared page table (XSPT) is located in main storage; the PAGRT subroutine is called to write all pages in that segment onto the Quickstart data set. RSS page tables are then written as a part of IVM. Upon completion of IVM a word of X'FF' is put in the recording buffer.

The PSA buffer (or the page 0 buffer in a simplex configuration) is next written on the Quickstart volume, and the address of the PSA is placed in the recording buffer. In a duplex system, the address of the other CPU's PSA is also placed in the recording buffer.

By use of the byte map table (MEMAD) used by Startup, all pages that are assigned to RESSUP and RSSSUP (that is, all pages with X'DC' in this byte) are written on the Quickstart volume, and the addresses of these pages are placed in the recording buffer. Once RESSUP has been written, a one-word entry of X'FF' is made in the recording buffer.

The XTSI and all assigned page table buffers are then written on the Quickstart volume and their addresses are placed in the Communication Region of Startup.

The recording buffer itself is then written in the Quickstart data set in the pages reserved for it. The last DSCB is closed out. If there are excess pages in the Quickstart data set from a previous Quickstart data set creation, these pages are then freed. The DSCB buffer and the PAT for the Quickstart volume are written in the Quickstart data set.

Startup is then located and written on the Quickstart volume on the pages previously reserved for it.

Write IVM Page (PAGRT)

Chart AC, Page 11

This subroutine writes pages from an external page table or from an external shared page table onto the Quickstart volume. For each nonzero entry in the XPT or the XSPT for a given segment, PAGRT reads that relative page number off the primary or secondary paging device. The relative page number is obtained from the XPT or XSPT entry. For a drum, the slot/page format is first converted to a relative page number format, and Startup subroutine EIAA2 is called to perform the READ into the Read/Write buffer. This buffer is then written on the next available page of the Quickstart data set. The device-type code of the direct access device for the current segment, and the relative page number of that device, are recorded in the recording buffer entry (public segments are written to a drum if one exists).

The PAGRT subroutine is called once for each assigned segment in IVM.

Create Format-E DSCB (DSCBE)

Chart AC, Page 12

This subroutine creates a format-E DSCB for the Quickstart data set, or locates a previous one on the Quickstart volume.

DSCB first calls GETPAT, a Startup subroutine, to read the DSCB pages on the Quickstart volume. Each DSCB is searched for the identifier TSS*****.QKSTART; if this is found, the relative page number and the slot number are saved and return is made to the mainline. If no format-E DSCB is located by the PATLOC subroutine, a new DSCB page is located (by calling PATLOC) and its relative page number is saved. Slot 0 is marked as the slot number for the format-E DSCB. Fixed format-E information (the identifier TSS*****.QKSTART concatenated with .DSxxxxxx, where xxxxxx is the volume identification of the Quickstart volume) is then moved in. VAM-sequential and fixed-length flags are set, and the record length is set to 4096 bytes, as is the field containing the number of bytes on the last page. The format-E indicator is set and control is returned to the mainline.

Locate DSCB Word/Free Page (DSCBF/DSCBA)

Chart AC, Page 13

If entry is at DSCBF, this subroutine locates the next available word in the DSCB. If entry is at DSCBA, the subroutine frees all remaining pages in the Quickstart data set.

If there is more room in the current DSCB, the displacement is updated, the location is calculated by displacement into the DSCB buffer and, if pages are not being freed, the number of pages in the data set is updated by one and control is returned to the mainline.

If pages are being freed (entry was at DSCBA) the next relative page number, if it exists, has its entry freed in the PAT table, the entry is freed in the DSCB, the number of pages freed is updated, and transfer is made to see if the next displacement is in the current DSCB. When there are no more pages, the DSCB type is set in the E-format DSCB, the Checksum value is calculated, and control returns to the mainline.

If the current DSCB is full, a check is made to determine if the Quickstart data set previously existed, or if entry was to free pages. If neither is true and it is not a format-E DSCB, the format-F indicator is set, and a check is made to see if another DSCB can fit on the page. The new slot number is saved internally, as well as in the forward pointer of the previous DSCB, along with the relative page number for the DSCB page. The Checksum value is calculated, and, if a new page has not been started, the displacement is set to zero and the location of this DSCB within the buffer is calculated in preparation for an exit. If entry was to free a page, on the first time through, the type is set in the DSCB and control branches to calculate the Checksum value.

If this is not the first time through, the type and Checksum are changed to zeros, and control branches to check if the new DSCB page is the same as the old.

If there are no more DSCB pages on the chain, and if a page is not being freed, the exist flag is turned off in the Quickstart byte, the new-data-set-larger flag is turned on, and control branches to get a new DSCB page. If a page is being freed, all other pages have been freed and the subroutine exits to the mainline.

Locate Available Page (PATLOC)

This subroutine locates an available page on the Quickstart volume by searching the PAT for an indicator of X'00'. If the end of the PAT is reached, the message INSUFFICIENT SPACE ON QUICKSTART VOLUME - QUICK START CREATOR ABORTED is sent to the operator and control returns to Startup. Otherwise, the relative page number of the page just located and a pointer to its PAT entry are returned.

Update Buffer Location (RECPG)

This routine updates the address of the next available location in the recording buffer. If a buffer page is full, a check is made to see if it was the last recording page. If it was not, the current buffer is written out again, the current buffer address is updated, the next available location pointer is reset and control returns to the mainline. If no new page is needed, just the pointer is updated. If the last page was just filled up, the message

```
RECORDING BUFFER FULL
QUICKSTART CREATOR ABORTED
```

is sent to the operator and control returns to the Startup module.

Buffer Cleanup (DSC20)

This subroutine clears out the contents of a buffer.

XSPT Entry Convertor (DSC25)

This subroutine converts the XSPT entry of a drum from slot/record format to relative page number format.

Read from Quickstart Volume (DSC50)

This subroutine reads from the Quickstart volume into a buffer. If an unrecoverable I/O error occurs, a message is sent to the operator:

```
I/O ERROR CREATING QUICKSTART DATA SET
QUICKSTART CREATOR ABORTED
```

and control returns to the mainline.

Write to Quickstart Volume (DSC60)

This subroutine writes the contents of a buffer onto the Quickstart volume. If an unrecoverable I/O error occurs, a message is sent to the operator:

```
I/O ERROR CREATING QUICKSTART DATA SET
QUICKSTART CREATOR ABORTED
```

and control returns to mainline.

Calculate Checksum (DSC75)

This subroutine calculates the Checksum of a DSCB and stores it in the previous DSCB.

Error Exit (ERREXA)

This is the routine given control when an error occurs. A message is issued to the operator indicating the type of error encountered. If the Quickstart data set already existed or if the PAT had already been written, the pages are freed and only the format-E DSCB indicator remains. Otherwise, no indication is made and control returns to Startup. On a recursive call to the error routine, a message is issued to the operator:

```
CATASTROPHIC ERROR ENCOUNTERED - QUICK-
START CREATOR TERMINATED
```

and CEIAB terminates.

COMMON STARTUP SUBROUTINES

Read/Write Operator Routine (OPER)

This routine is called to issue a desired message and accept a reply, if requested. On entry, a general register

contains the number of the message to be issued. On the first pass, OPER relocates all address constants in the message address table. Then the parameter message number is used to index into the table, to obtain the length and address of the appropriate message. Another parameter register indicates one of the options (write-only, read-only, or read/write), and if applicable, the length of the read. STERM is invoked to do the actual I/O on the operator's terminal. Control returns to the caller.

Read Cards Routine (READCARD)

This routine is called when operator response messages are to be entered as card input and not through the operator's terminal. If a card is read successfully, it is printed at the operator's terminal (via OPER) and control is returned to the caller with register 1 pointing to the address of the card buffer. If an end-of-file occurs, zeros are returned in register 1.

Print Message Routine (PRINTER)

This routine is called to print a message on the printer, and, if the message corresponds to a major error, to print the message on the operator's terminal and terminate Startup. Parameter registers specify the message number and the severity of the error. PRINTER calls EIAA2 to send the message to the printer. Normally, control is now returned to the caller. However, if a major error occurs, PRINTER invokes OPER to issue the same message to the operator's terminal, and then enters the wait state.

Get Field Routine (GETFLD)

This routine is called to locate a desired field within a given area. (A field is defined as a group of characters within two delimiters. The delimiters include the beginning and end of the area, a comma, a tab character, (x'15'), and a string of one or more blanks.) GETFLD scans through the area until the given field is located and then returns the length and address of the field. If the field is not found, the length is set to 0. The address is set to -1 if the end of the given area was reached before the desired field, or to 0 if the field contained no information.

Operator's Terminal I/O Subroutine (STERM)

This is a subroutine called by Startup routines when a message must be printed on the operator's terminal. Its purpose is to initiate and control any communications between Startup and the system operator. On entry, parameter registers contain either the address of a CCW list, or data to be inserted into a prestructured CCW. This is determined by checking the high order byte of the first parameter register. If nonzero, the prestructured CCW is used; otherwise the address of the CCW list is set in the CAW. I/O processes are initiated via an SIO, and comprehensive error checks are undertaken to insure the successful initiation and completion of all the I/O steps. If the communication fails after ten retries, a message is issued on the printer. Should there be any malfunction with printer I/O, this process is tried again 10 times. In any event, this subroutine terminates Startup whenever the system operator's terminal cannot be reached. On a normal exit, at successful completion, registers are restored and control returned to the caller.

Move Text (MVTEXT)

This subroutine moves bytes of text from one location to another. In general it moves information from an input buffer to the output buffer. Entry parameters consist of the number of bytes to be moved (specified in INAD), and the address of their destination (specified in OUTAD). Bytes are moved in blocks of 256 and are moved in ascending sequence by location, beginning at INAD. Should the last move require less than 256 bytes, only the actual number of bytes remaining in the block are moved. Control is returned to the caller at the completion of the move.

Write Page on Paging Volume (OUTPG)

This subroutine writes a page from the output buffer onto either the paging disk or the paging drum. Two cells are set up in main storage before entry to OUTPG takes place, XPGNDK for the disk (the primary paging volume) and XPGNDR for the drum (the secondary paging volume). XPGNO contains the external page number on the paging volume where the output buffer page is written. PAGAD contains the device and hardware address type of the primary paging volume and SPGAD contains the device type and the hardware address type of the secondary paging volume. After determining the device type, OUTPG branches to ATRAN for a 2311 disk, to CTRAN for a 2314 disk, or to BTRAN for a 2301 drum. These routines convert the external page number to its corresponding CCHHR. Then, having set entry parameters for it, OUTPG calls EIAA2 to execute the CCW list for writing the output buffer on this CCHHR of the paging volume. If an I/O error occurs, OUTPG increments the external page number by one and retries the whole process. Once the attempt is successful, the output buffer is zeroed and control returns to the caller.

Create Symbol Table (SYMGEN) and Print Storage Map (SORDID)

This subroutine is called to create a table containing names and values of CSECTs, and their entry points in Initial Virtual storage or RESSUP/RSSSUP. The user is given the option of requesting a printed map of either Initial Virtual storage and/or the Resident Supervisor. If a map is to be printed, storage temporarily allocated for the map in MAPGEN is released and SORDID is called. SORDID sorts the CSECT names both alphabetically and numerically. The CSECT names, their respective real or virtual addresses, the version IDs and hierarchy numbers are arranged in ascending sequence within three columns. These names and values are printed in the form of a storage map, a page at a time, on the printer. Whether or not printing of the storage map is requested, linkage is provided to the RESSUP/RSSSUP Symbol table for RSS/VSS.

The symbols and their addresses are located for both data sets in the temporary map created by MAPGEN. The number of entries is found in the TDY heading and the origin of the table is in a field called JSMPBG. Each entry points to a CSD. A 24-byte entry is created for each definition in the CSD; an 8-byte left-adjusted name, a 4-byte value field, an 8-byte version ID, a 3-byte hierarchy number and an unused byte. The version ID is either an 8-byte name specified by the creator of the CSECT or a binary coded time stamp. The high order bit of the version ID is tested, and if it is off, the version ID is converted to the form: mm/dd/yy hh:mm:ss. Entries are made alphabetically. If the number of potential entries exceeds the buffer space allowed, a diagnostic message is issued but the subroutine is not terminated. A heading is added to the symbol table identifying the storage map. The symbol table is printed in hexadecimal print characters.

The buffer space for RESSUP's storage map is a fixed size of one and a half pages. IVM's buffer size is determined dynamically. The following are considerations for the RESSUP/RSSSUP Symbol table only:

- A combined symbol table of RESSUP/RSSSUP is created for the use of RSS/VSS. This symbol table is later written on the primary paging device by the Write Symbol Table (WRSYMTB) routine.
- SERR/Reconfiguration modules are bypassed. At the conclusion of SYMGEN, control is returned to the caller.

Reserve Space for PMDs in TDY (ADDPMD)

This subroutine is called to allocate storage for PMDs in the TDY, and initialize the PMD preface. The size of the PMD being processed is obtained from the user data following the module's member descriptor entry in the POD. For Initial Virtual storage, the first PMD is loaded eight pages above the TDY origin; for RESSUP, two pages. If enough space remains in a page, a second PMD is also assigned to it. If not, the PMD is assigned the next higher available page. If paging is needed, the initial major buffer and the buffer for BLDTBL are allocated at this time. All addresses allocated from this point on will be virtual real storage, indicated by the high-order bit being on. Space for PMDs can be allocated around malfunctioning pages. The table TDYTAB is used by IVM for relocating the TDY around these pages. The routine initializes only the first two words in any three-word group in TDYTAB. These words correspond to the lowest and highest real storage addresses within each contiguous group of TDY pages. The third word of a TDYTAB group is the common relocation factor.

After the first PMD in a group is allocated, the group header is initialized. Each new group is linked to the TDY heading and to its preceding group. Once paging begins, if a new group is needed, BLDTBL is called to enter this table into the page table and to page it out. At this time, if the old buffer was not large enough for the new PMD, ADDPMD will get a larger buffer and a new minor buffer if it has previously been needed. The first group's pointer to the preceding group is set to 0. The TDY heading is updated to point to each new group, and within each group header the fields pointing to 'last PMD in group' and 'end of group' are updated. After each PMD is allocated, the PMD preface link to the next PMD preface is initialized, and the remainder of the preface is zeroed out. As control is returned to the caller, the address of the PMD preface is sent as an exit parameter.

Generalized Input/Output Subroutine (EIAA2)

This is a subroutine used to initiate and control the I/O activity between the CPU and the channels. It is used to gain access to the direct access devices and the printer. On entry, the address of the I/O device is set in parameter register 1. A field called IOBYT is used to inform EIAA2 of the options desired by the calling routine. These options indicate:

- Whether the calling routine has supplied a CCW list

- Whether a read or write operation is desired

- Whether relocation is desired for disks if the first try is unsuccessful and the PAT indicates that there is a relocation entry for this external page

If EIAA2 is to build the CCW list, the calling routine must specify pertinent information in the following fields:

- IOLEN The number of data bytes to transfer

- IOAD The beginning location of the transfer area

The CAW is then set up, and an SIO instruction is issued to the I/O device. Continuous comprehensive error checks are made to insure successful completion of the I/O activity. If a malfunction occurs, successive retries are attempted in order to complete the operation. Unrecoverable hardware errors result in a diagnostic message being issued to the system operator. The message informs him of the malfunctioning device, and either CSW status bytes or sense bytes are displayed, depending on the nature of the failure. EIAA2 must also determine if it is necessary to terminate Startup. It terminates if the IPL volume is in use when the failure occurs, or when the paging volume malfunctions during the initialization of SDAT.

If a track condition check occurs on an I/O operation to a disk, IOBYT is checked to see whether the relocation flag is set. If it is, the relocation control entry in the volume's PAT is checked to see whether there are any relocation entries. If there are, and if one of these entries is for the page in question, EIAA2 retries the operation on the relocated page. These relocation entries are set by the RESTORE program when it encounters a track condition check.

If a command reject and a seek check occur when using the paging disk, it is concluded that this disk is not VAM2-formatted. An appropriate message is sent to the operator and Startup terminates.

Startup continues in all other cases. If there should be a recoverable error (form checks, dropping ready status, etc.), this routine issues a message to the operator telling him of the problem and asking him to reply when the situation has been rectified. Startup waits for the reply. If a nonrecoverable error occurs on the printer, maintenance is required. This routine automatically switches to another printer if one is available. If not, Startup continues with printing suppressed. Once the I/O routine has fulfilled its function, control is returned to the caller via the return register. Depending on the calling routine, the activity requested, and the results of EIAA2, the appropriate return code is set in IOBYT and control returns to the calling routine.

Write SERR/Reconfiguration Modules on Drums (SERR100, SERREND)

This subroutine is invoked to write all SERR and Reconfiguration modules on all the drums in the system. On entry, the following fields of information are available to the routine:

XPGNO	First available page on paging drum.
PGSRSDA	Table of drum hardware addresses.
NODMSYS	Number of available drums.
OUPTAD	Output buffer address.

The GETEXT routine processes the SERR/Reconfiguration modules last. It calls SERR100 to write each module on all the drums in the system. Each call results in one module being written on every drum. The module is in the output buffer when the SERR100 routine is entered. It writes the first SERR module on the paging drum, and stores the BBCCHHR of that module in the System table. This same address is then used as the starting location for the SERR/Reconfiguration modules on each drum in the system. The external page number of the module just written is stored in the first half of a fullword entry for each drum in the list. This list is used later by the ASATRT routine. When a Reconfiguration module is written on the paging drum, its BBCCHHR is saved in the System table. Since it is possible that an I/O failure could occur when trying to write on the drum, it could happen that, though the first external page numbers of the modules are the same, the address of the last module could vary from drum to drum. SERR100 will skip any malfunctioning pages and write in the next available one. SERREND is invoked by GETEXT to store the external page number of the last module written on each drum. This is saved in the second half of the entry for each drum in ASATRT's list. Control returns to the caller.

Get a Block of Main Storage (GETMEM)

This subroutine is called to allocate a block of available storage. At entry, these parameter fields have been set:

GTDRN	Direction of search for next page, if requested page(s) are not available.
GTFLG	Flag to be placed in page map.
GTNUM	Number of requested contiguous pages.
GTNCA	Test byte for termination.
RTSTRT	Starting address of assigned block.

GETMEM is the routine called upon by the Startup routines whenever they wish for main storage allocation or wish to inspect the availability of a block of main storage. Each byte within the page map reflects the condition of its corresponding page of main storage. The byte settings and their meanings can be found in the SCBTL routine description. The search commences at the beginning of the map if GTDRN equals GTDUP, or at the end of the map if GTDRN equals GTDDOWN. The search continues in the specified direction until the requested block of contiguous pages is located. The value in GTFLG is then inserted in the bytes corresponding to the available pages, and the starting address of the block is returned to the caller in RTSTRT.

If the requested block is not available, GTNCA is inspected. If the field is set to GTNCAR, control is returned to the caller with RTSTRT set to 0, indicating that the block of storage was unavailable and not allocated. If GTNCA is set to GTNCAT and the block was unavailable, GETMEM issues a diagnostic message to the operator indicating that Startup must be terminated. The system then goes into the wait state.

If GTNCA is set to GTNCARNF, and GTDRN equals GTDUP, only the first block following the last assigned block is searched for availability. If this block is available, the requested flags are set and the starting address is returned to the caller in RTSTRT. If the first block is not available, control returns immediately to the caller with RTSTRT set to zero. This feature enables real storage modules to be packed into contiguous pages.

Create the Resident Shared Page Index Table (CRRSPI)

This routine creates the Resident Shared Page Index table (RSPI) after RESSUP text has been loaded in main storage. It also obtains main storage for each SPT and XSPT and moves the tables from their buffers into lower main storage.

The RSPI contains information regarding the origin and length of each SPT. Space for the RSPI is allocated on the last page of the Resident Supervisor if there is sufficient room. Otherwise, a new Resident Supervisor page is allocated for the RSPI.

Three items are initialized in the System table (CHBSYS).

- The pointer to the RSPI
- The RSPI count
- The next available SPT number

Supervisor Core Allocation is called to reserve main storage for each SPT and XSPT. The SPT number, length and origin are then stored in the RSPI entry corresponding to the segment number in which they are to be located. Then the SPT and XSPT is moved to the storage location that has been reserved for it. The 'reserved for RESSUP' flag is set in the Core Block table (CHBCBT) for each SPT page.

Control returns to Startup mainline.

Read Data Set Control Block (RDSCB)

RDSCB is a generalized routine used to find a DSCB on a VAM2-formatted pack. Two options are provided. The first option supplies the DSCB when given the external page number of the DSCB and the slot number of the DSCB within the page as input. The second option supplies the DSCB when given the data set name.

For the first option, the field DSNAD contains the address of the slot number and external page number as input parameters. For the second option DSNAD contains the address of the data set name.

On entry, general register 0 contains a 1 if option one is desired or the length of the data set name if option two is desired. General register 1 contains the device address on which the data set resides.

If option one is specified, RDSCB reads in the specified DSCB page if it is not already in main storage. The requested DSCB is then moved into DSCBF.

On a normal return, the field DSCBF will contain the requested DSCB. General register 0 will contain a 1 in its high order bit if the DSCB was not found.

If option two is specified, a check is made to see whether a DSCB page for this volume is presently in main storage. If it is, each DSCB on this page is checked for the requested name until a match is found. If no match is found or if there is no page

in main storage, the PAT is checked for all DSCB pages on this volume. These pages are searched sequentially until either the DSCB is located or there are no more DSCB pages.

Read Page Assignment Table (GETPAT)

GETPAT searches the PAT table for a volume and returns the external page number of a given relative DSCB page. The calling routine must supply the relative DSCB page number. (The first relative DSCB page is zero.) General register 0 contains the relative DSCB page number and general register 1 contains the address of the device on which the DSCB page resides.

On a normal return, GETPAT supplies the requested external page number in page 0. If the given relative page does not exist, the high order bit of general register 0 is set to one.

NIPLPAT contains the address of the PAT pages for this volume. DSCBF contains the volume label if this was the first call for this volume.

The PAT for the IPL volume was read in by PRELUDE in order to locate the Startup DSCB and is therefore resident in main storage. If the requested volume is not the IPL volume or the volume specified on the previous call, the PAT is read in. The address of this volume and the length of its PAT is saved for the next call. (The PAT is one page long in the case of a 2311 volume and two pages in length for a 2314 volume.)

The PAT is then searched for the requested relative DSCB page. When the DSCB page is located, its external page number is computed from the displacement of the qualifying byte within the PAT.

Page Task Dictionary Table (PAGTDY)

This routine is called throughout the link loader, whenever a virtual real storage address is encountered. At entry, general register 0 must contain a 0 or 4 to indicate which of two buffers is to be used. REQAD1 or REQAD2 contains the virtual real main storage address requested. BLDTBL is called to write out the group currently in the buffer. The PMD group containing the virtual real storage address is then read into the buffer. RETAD1 (RETAD2) is set with the real storage address which points to the virtual real storage address requested.

Build Task Dictionary Table (BLDTBL)

This subroutine is called by ADDPMD to write PMD groups that must be paged onto external storage. Space is temporarily allocated by BLDTBL on the primary paging volume, starting with the highest external page number and decending to the lowest. This allocation is done on the assumption that space permanently assigned by Startup on that device will begin with the lowest available page number and work up. BLDTBL also builds a table containing information to permit retrieval of these PMD groups.

For each page, there is an entry in the table, consisting of four words:

1. displacement of page relative to beginning of TDY
2. number of pages in group
3. external page number
4. displacement of this group relative to beginning of TDY

BLDTBL may also be called by PAGTDY to update the external images of PMD groups previously written out.

Entry to the SYSGEN phase begins after a time sharing system has been generated, and a system programmer has logged on and assembled the system generation macro instructions. These macro instructions are an input data set for the TSS assembler. The operands of the macro instructions take the form of configuration-dependent parameters and installation option indicators. There are thirteen macro instructions in SYSGEN. Twelve of them may be assembled in any order. The thirteenth, GENSCB, must be last. They must be processed by the TSS/360 assembler, and their output creates the SYSGEN.MODULE data set. The first twelve macro instructions create no code. They use input supplied by the systems programmer to equate global symbols to configuration options. These global symbols are then used by GENSCB to fill in the system tables. Refer to Appendixes A and B for chart descriptions of the fields affected, and the global settings.

Figure 6 gives an overall view of the following macros:

CPU saves the identity of the CPUs, their model numbers, features and the primary and alternate prefixes assigned to each.

STEM saves the number of storage elements in the installation and their model number.

CCU saves the number of channel controllers in the installation.

CHANNEL determines whether selector or multiplexor channels are present, and saves the hexadecimal address of those in existence. It also supplies a total count of both.

DCU saves the addresses and IDs of all the control units in the installation. It also saves their model and unit numbers.

DEVGRP saves the symbolic and actual addresses, the paths, units and model numbers of all the devices in the system. It also supplies various I/O parameters.

OPCNSL saves the addresses and paths to the operator consoles.

CLOP saves the command language default options.

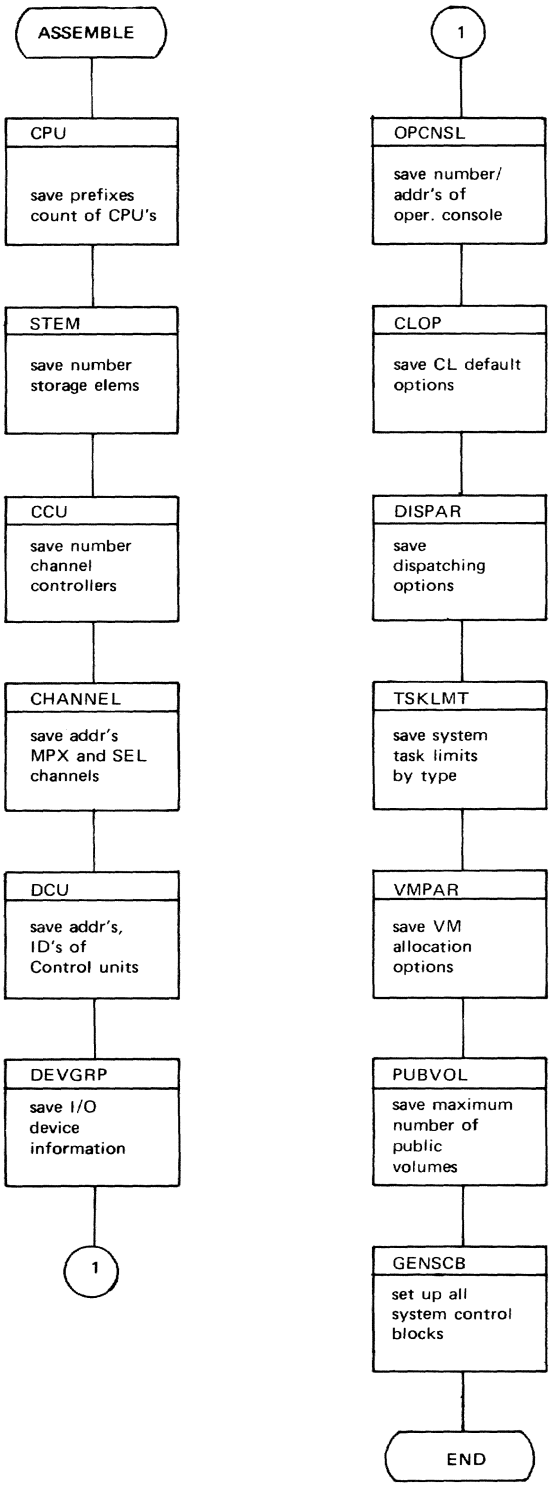


Figure 6. SYSGEN Macro Logic Flow

DISPAR saves the dispatching or scheduling options of the system.

TSKLMT saves the system limits on size and number of tasks (specified by type) allowed to operate at any one time.

VMPAR saves the virtual storage options that specify the parameters for handling and allocating virtual storage.

PUBVOL saves the maximum number of public volumes.

In all the above macro instructions, checks are made that valid information has been supplied. In conversational mode, a diagnostic message is issued on the terminal. The Assembler waits for corrected information. In nonconversational mode, a diagnostic message is issued on the output device, but processing continues through GENSCB with possible erroneous results. If any errors are encountered in GENSCB, the entire macro processing must be done over again.

GENSCB uses information supplied in the form of global symbols, and fills in the system tables. This macro generates code in the form of CSECTs. Chart AE presents the flow of GENSCB. First, all the global and local symbols are defined. The System table (CHBSYS) and the System Common table (CHBSCM) are generated using data from DISPAR, VMPAR, TSKLMT, and CLOP. Then the Scan table (CHBSCN) and Scan Master Control table (CHBSMC) are generated. Looping through a channel matrix, the Channel tables (CHBCHL, CHBMCH, and CHBSCH) are generated. Then, looping on channel control unit addresses, the Control Unit table (CHBCUT) and its flags are set. Using a control unit and channel matrix, the Device Group tables (CHBDEV) are generated along with a Symbolic to Actual Conversion table (CHBSAC). The Symbolic Device Allocation table (CHBSDA) can then be generated. Then, picking up the global symbols and inserting their values into the CSECT, the following tables are generated:

System Operator ID table (CHBSOT)

Available Device table (CHBHED)

Configuration Control Block (CHBCCB) containing the following tables:

- CPU Status table (CCBCST)
- Drum Path table (CCBDPT)
- Transmission Control Path table (CCBTPT)
- Channel Control Unit table (CCBCCT)
- Correspondence List (CCBCLT)
- Printer Path table (CCBPRT)
- Paging Statistical Data table (CHBPSD)
- Statistical Data table (CHBSDT)
- Core Block Table Header (CHBCBH)
- Core Block table (CHBCBT)
- Support System Device Allocation header (CHBECXRA)
- Support System Device Allocation table (CHBECXRB)
- Virtual Memory Support System Allocation table (CHBECXVA)
- Public Volume table (CHBPVT)
- Terminal Device Table (CHBTDE)
- Remote Job Entry Table (CHBRJE)
- Bulk Communication Table (CHBBCT)

The second half of this SYSGEN phase occurs when the system programmer executes the TSS*****.APGEN command procedure to use this SYSGEN.MODULE data set to update the system tables and control blocks on the IPL Volume. Following is a list of the commands and linkage editor statements within. TSS*****.APGEN. (Refer to: IBM System/360 Time Sharing System: System Generation and Maintenance, GC28-2010, "Appendix G: Procedure for Making TSS/360 Operational. 8. Contents of the APGEN procedure," for the procedure to both modify and execute this command procedure.)

Contents of the APGEN Command procedure:

```
  LOGON TSS
SECURE (DA=1,2314)
DDEF CEIDALIB,VP,DSNAME=SYSGEN.MODULE,
       DISP=OLD
DDEF GENCCB,VP,DSNAME=SYSCCB(0),DISP=OLD
RUN LNK
CCBLNK,N,,GENCCB,VID,N,Y
RENAME CHBECXVA,CHBECXRA,CHBECXRB
RENAME SYSGEX
RENAME CHBSYS,CHBSAC,CHBDEV,CHBCHL,CHBMCH
RENAME CHBSCH,CHBCUT,CHBSCN,CHBSMC,CHBCBH
RENAME CHBCBT,CHBPSD,CHBSCM,CHBSDA,CHBPVT
RENAME CHBSOT,CHBHED,CHBAHD,CHBAVE,CHBSDT
RENAME CHBTDE,CHBRJE,CHBSST,CHBBCT
INCLUDE CEIDALIB(SYSGEN)
END
RELEASE GENCCB
ERASE SOURCE.CCBLNK
DDEF GENSUP,VP,DSNAME=RESSUP(0),DISP=OLD
RUN LNK
SUPLNK,N,,GENSUP,VID,N,Y
RENAME CHBECXVA,CHBECXRB
RENAME SYSGEX
RENAME CHBCCB,CHBSCM,CHBSDA,CHBSOT,CHBHED
RENAME CHBAHD,CHBAVE,CHBSDT,CHBPVT,CHBBCT
INCLUDE CEIDALIB(SYSGEN)
END
RELEASE GENSUP
ERASE SOURCE.SUPLNK
DDEF GENIVM,VP,DSNAME=SYSIVM(0),DISP=OLD
RUN LNK
IVMLNK,N,,GENIVM,VID,N,Y
RENAME CHBECXRA,CHBECXRB
RENAME SYSGEX
RENAME CHBCCB,CHBSYS,CHBSAC,CHBDEV,CHBMCH
```

```
RENAME CHBSCH,CHBCUT,CHBCHL,CHBSMC,CHBSCN
RENAME CHBCBH,CHBCBT,CHBPSD,CHBTDE,CHBRJE,
RENAME CHBSST,CHBBCT
INCLUDE CEIDALIB(SYSGEN)
END
RELEASE GENIVM
ERASE SOURCE.IVMLNK
DDEF GENRSS,VP,DSNAME=RSSSUP(0),DISP=OLD
RUN LNK
RSSLNK,N,,GENRSS,VID,N,Y
RENAME SYSGEX
RENAME CHBSYS,CHBSAC,CHBDEV,CHBCHL,CHBMCH
RENAME CHBSCH,CHBCUT,CHBSCN,CHBSMC,CHBCBH
RENAME CHBSOT,CHBHED,CHBAHD,CHBAVE,CHBSDT
RENAME CHBCBT,CHBPSD,CHBSCM,CHBSDA,CHBPVT
RENAME CHBCCB,CHBTDE,CHBRJE,CHBSST,CHBBCT
RENAME CHBECXVA,CHBECXRA
INCLUDE CEIDALIB(SYSGEN)
END
ERASE SOURCE.RSSLNK
RELEASE GENRSS
RELEASE CEIDALIB
  LOGOFF
```

The flowcharts in this manual have been produced by an IBM program, using ANSI symbols. The symbols are defined in the left column below, and examples of their use are shown at the right.

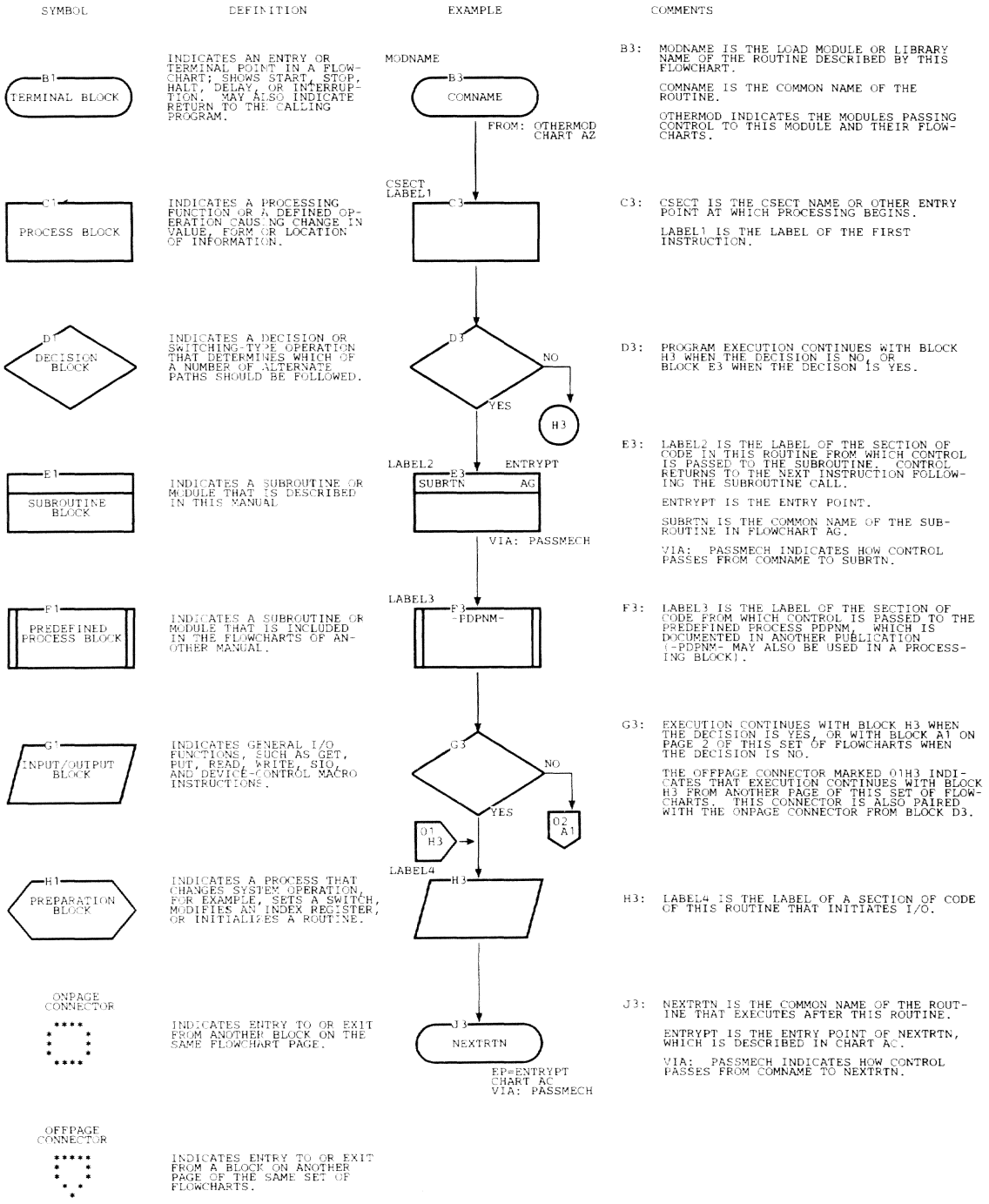


Chart AA. SYSBLD (CEIFA) Overview (Page 1 of 16)

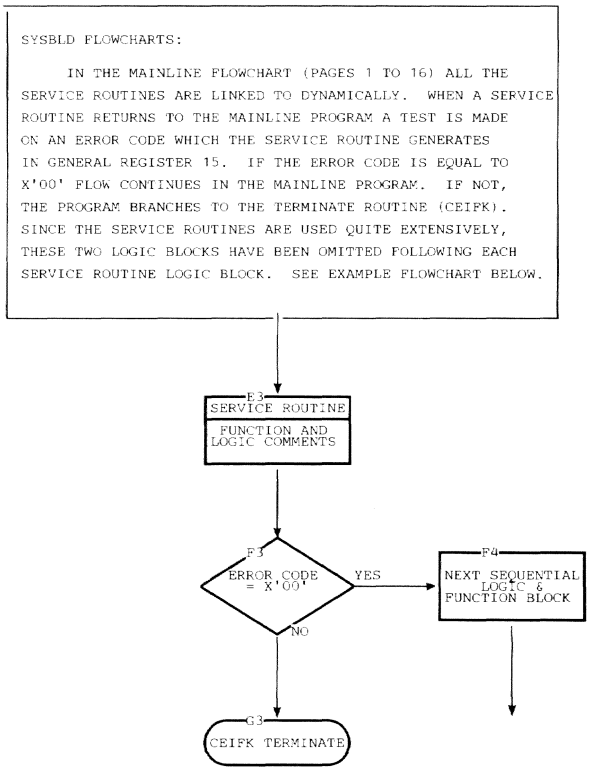


Chart AA. SYSBLD (CEIFA) (Page 2 of 16)

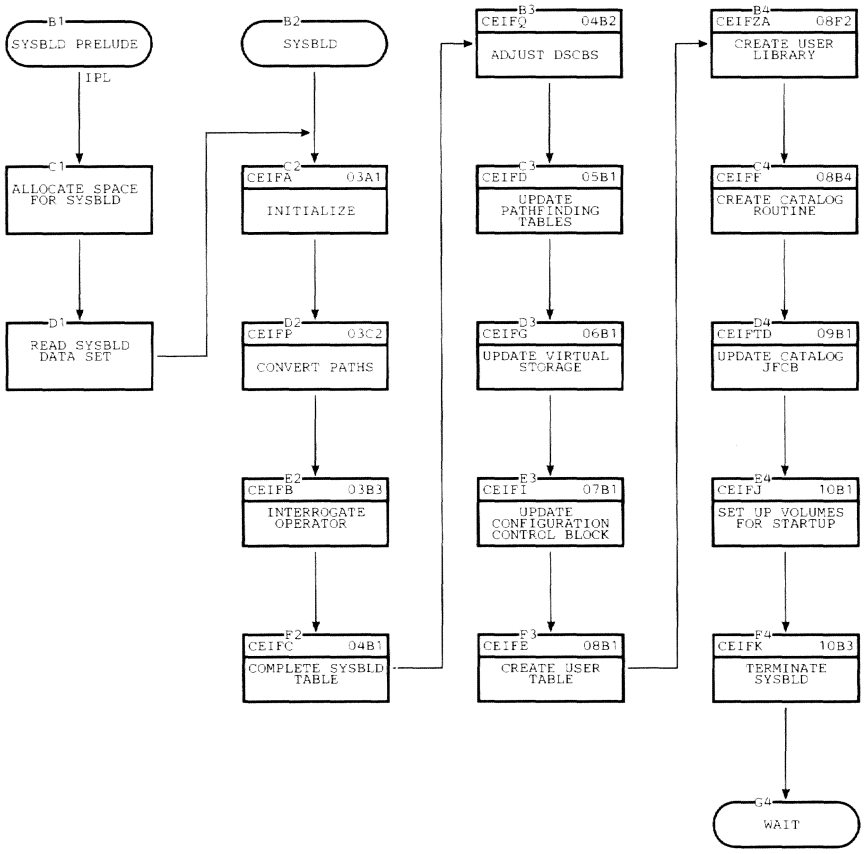


Chart AA. SYSBLD (CEIFA) (Page 3 of 16)

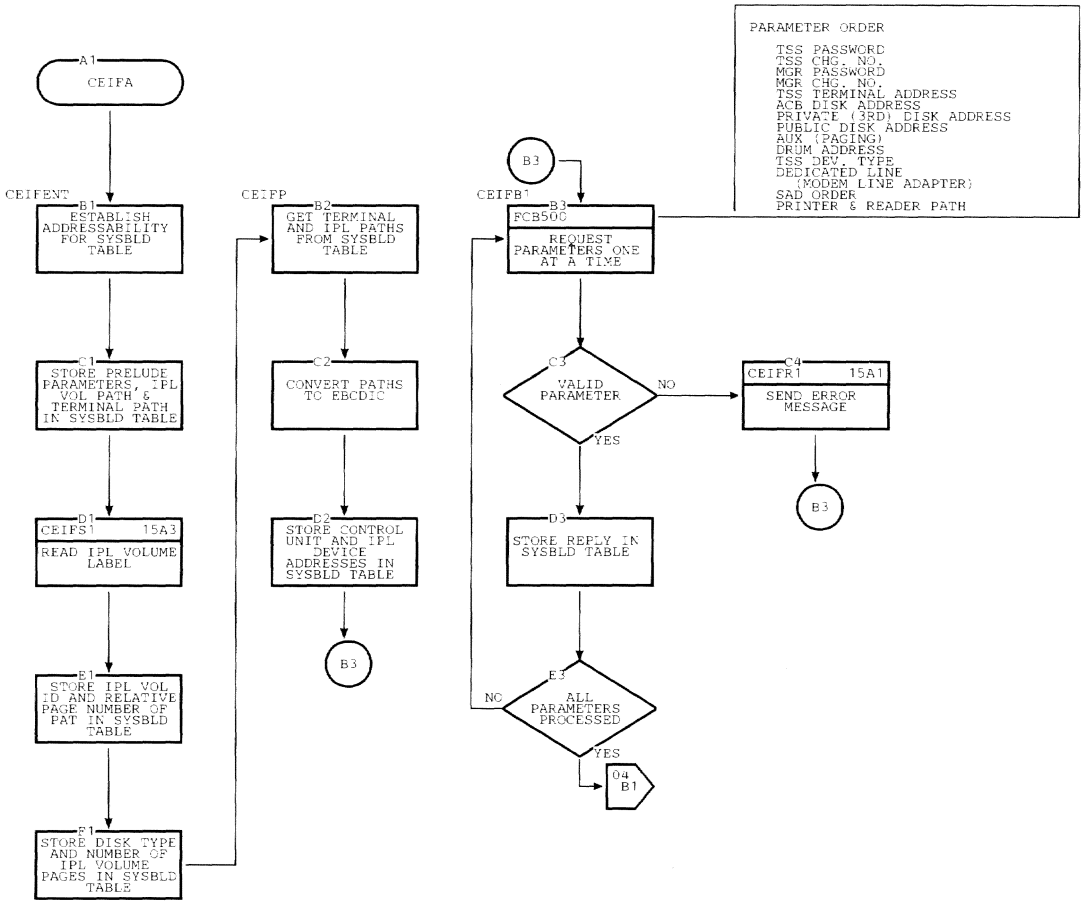


Chart AA. SYSBLD (CEIFA) (Page 4 of 16)

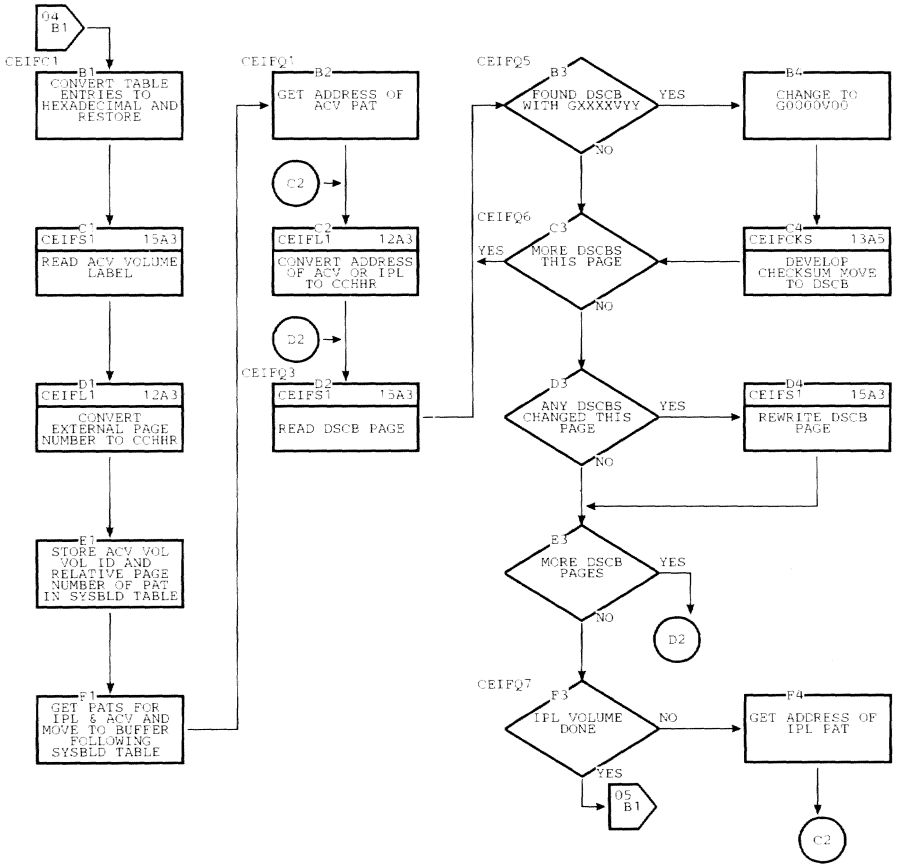


Chart AA. SYSBLD (CEIFA) (Page 5 of 16)

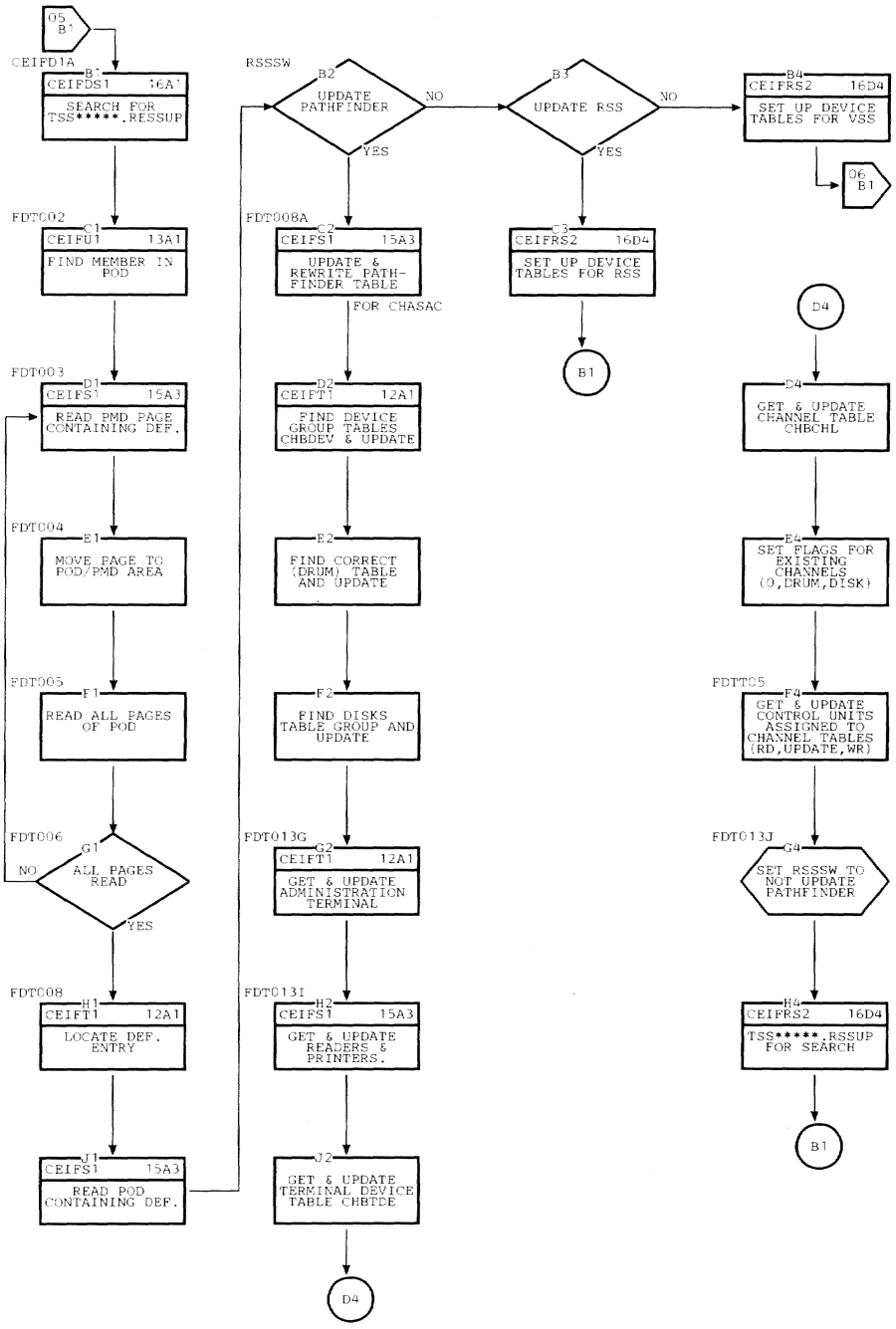


Chart AA. SYSBLD (CEIFA) (Page 6 of 16)

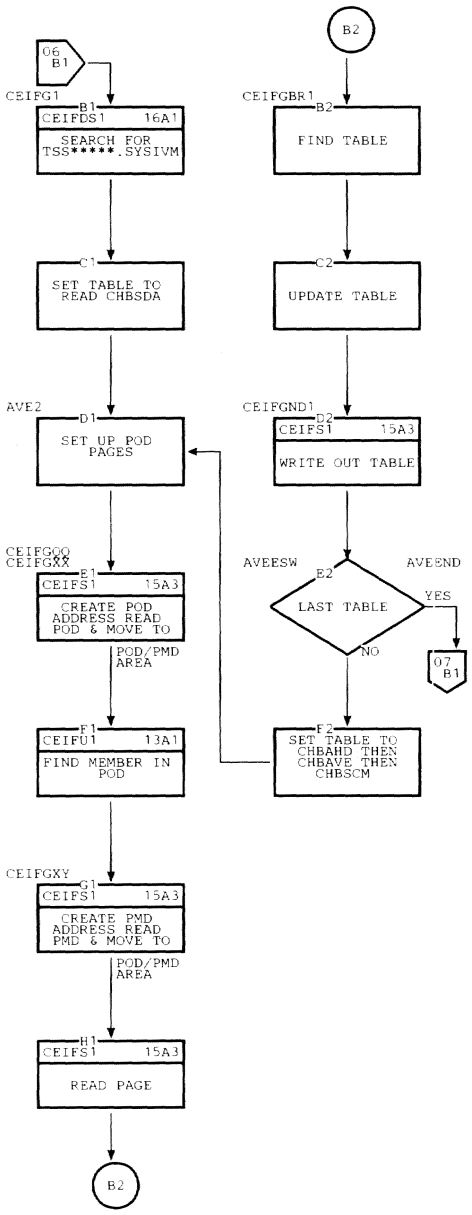


Chart AA. SYSBLD (CEIFA) (Page 7 of 16)

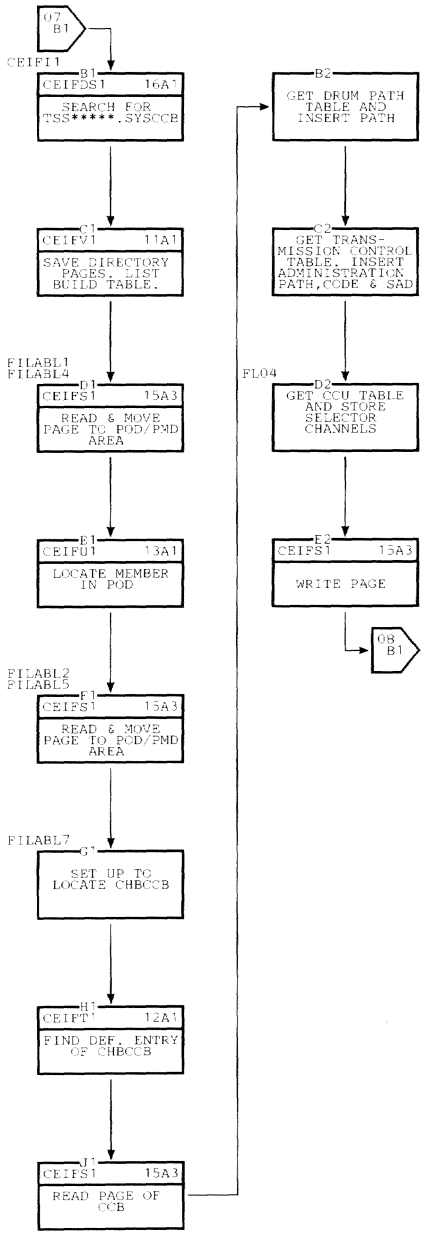


Chart AA. SYSBLD (CEIFA) (Page 8 of 16)

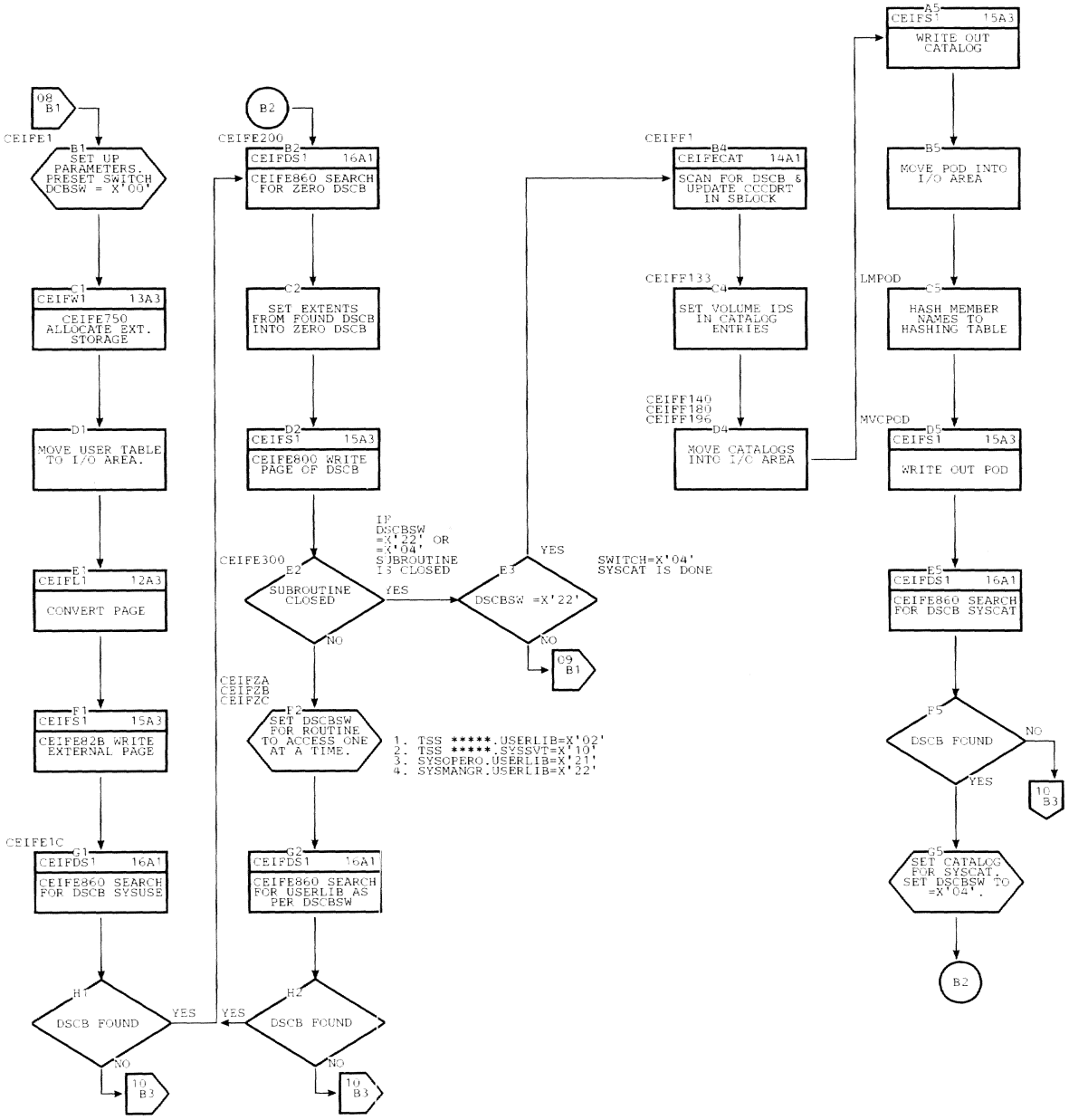


Chart AA. SYSBLD (CEIFA) (Page 9 of 16)

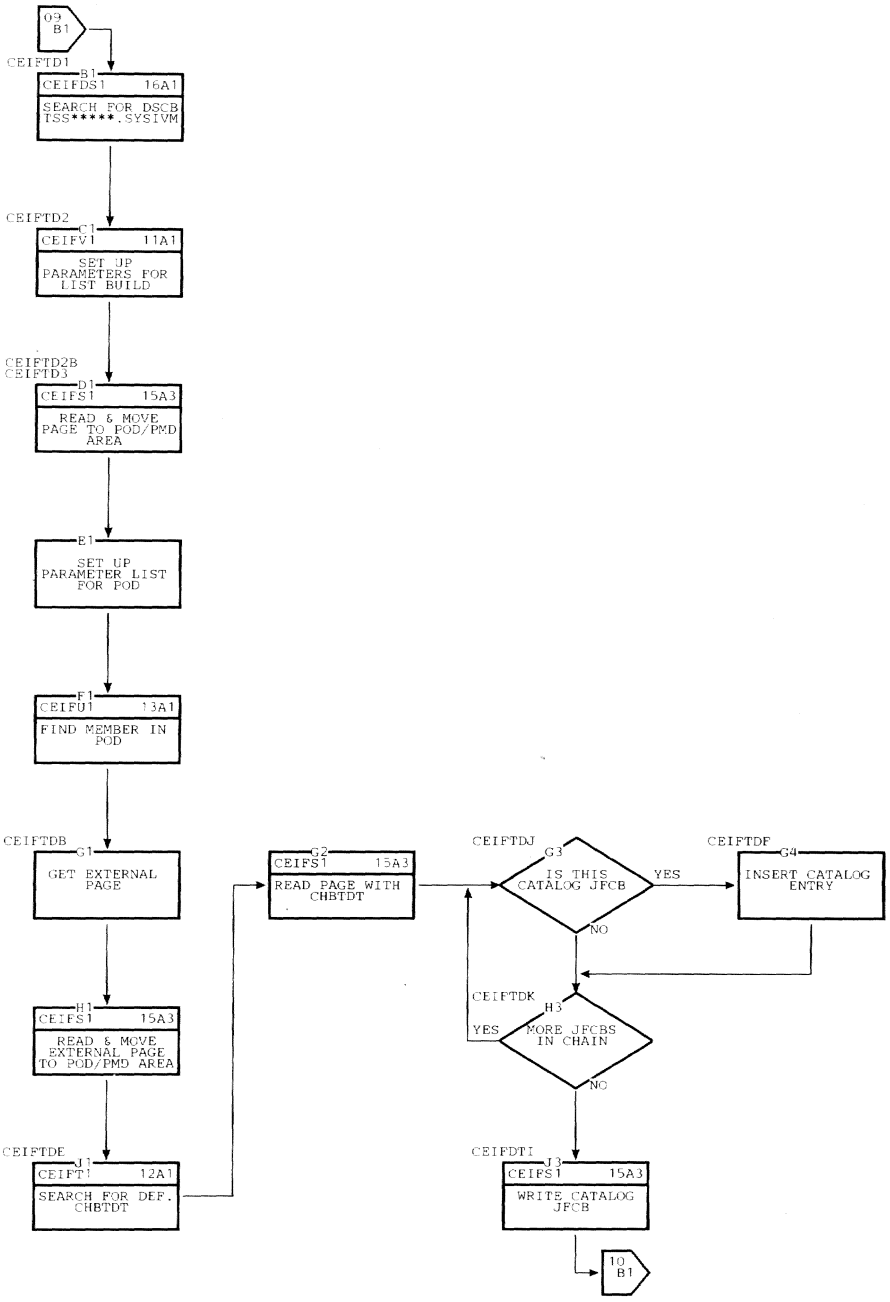


Chart AA. SYSBLD (CEIFA) (Page 10 of 16)

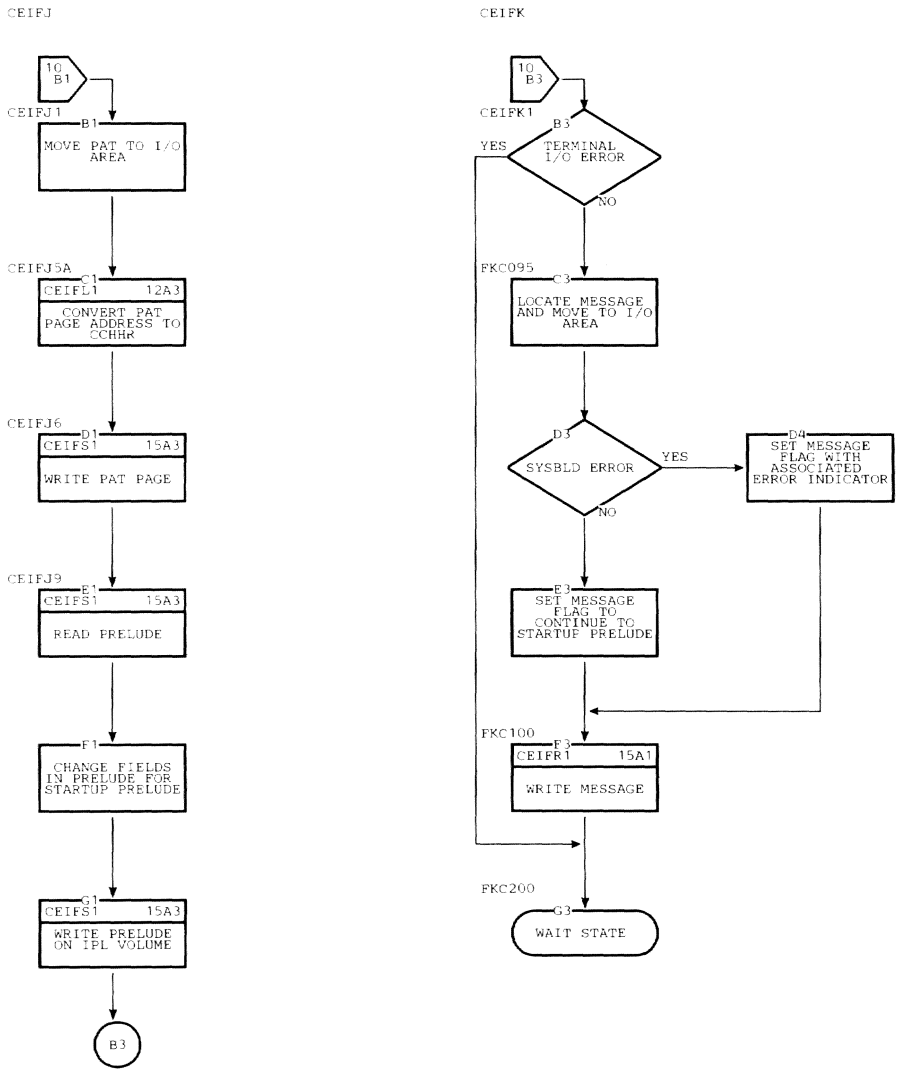


Chart AA. SYSBLD (CEIFA) (Page 11 of 16)

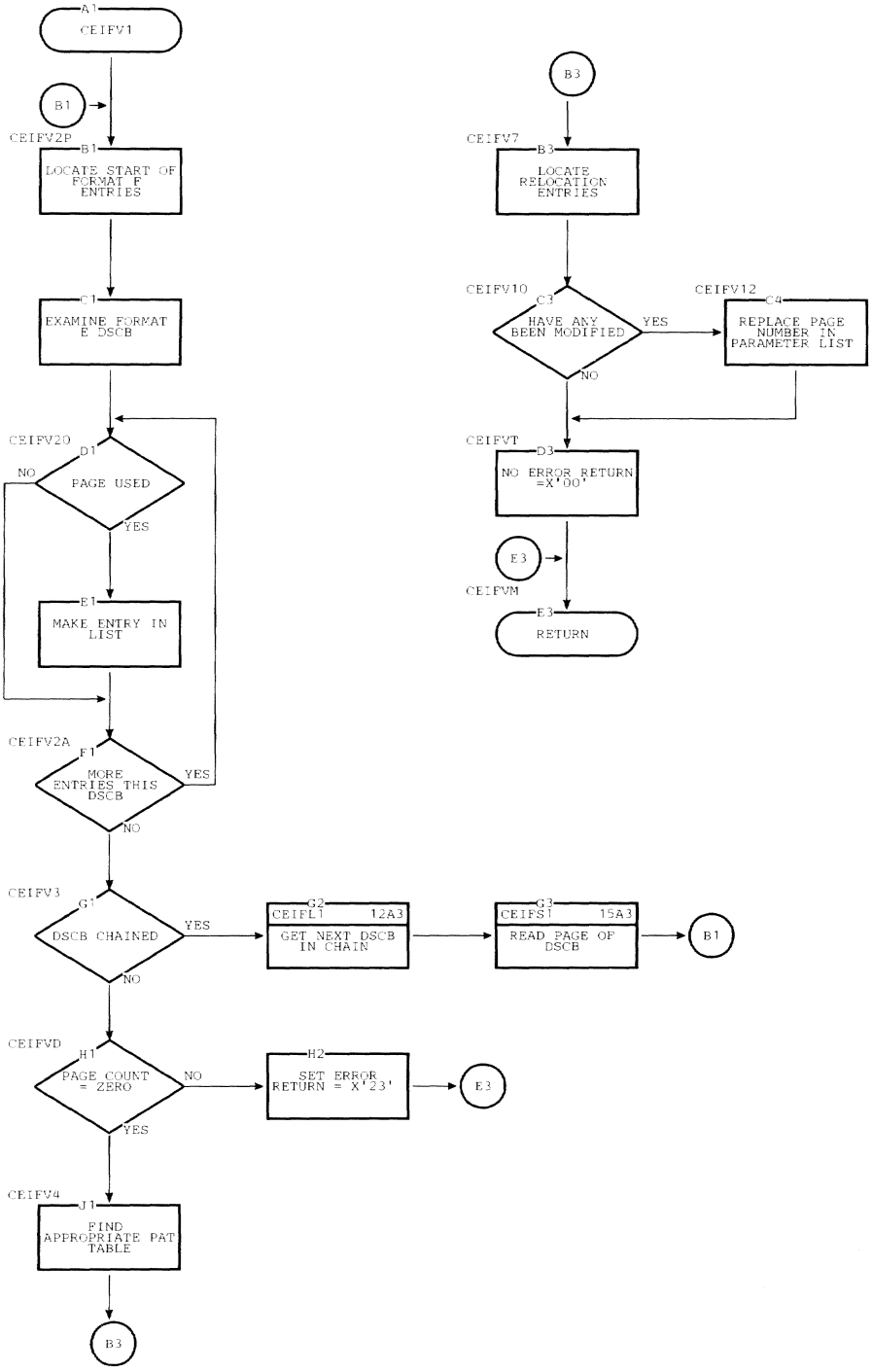


Chart AA. SYSBLD (CEIFA) (Page 12 of 16)

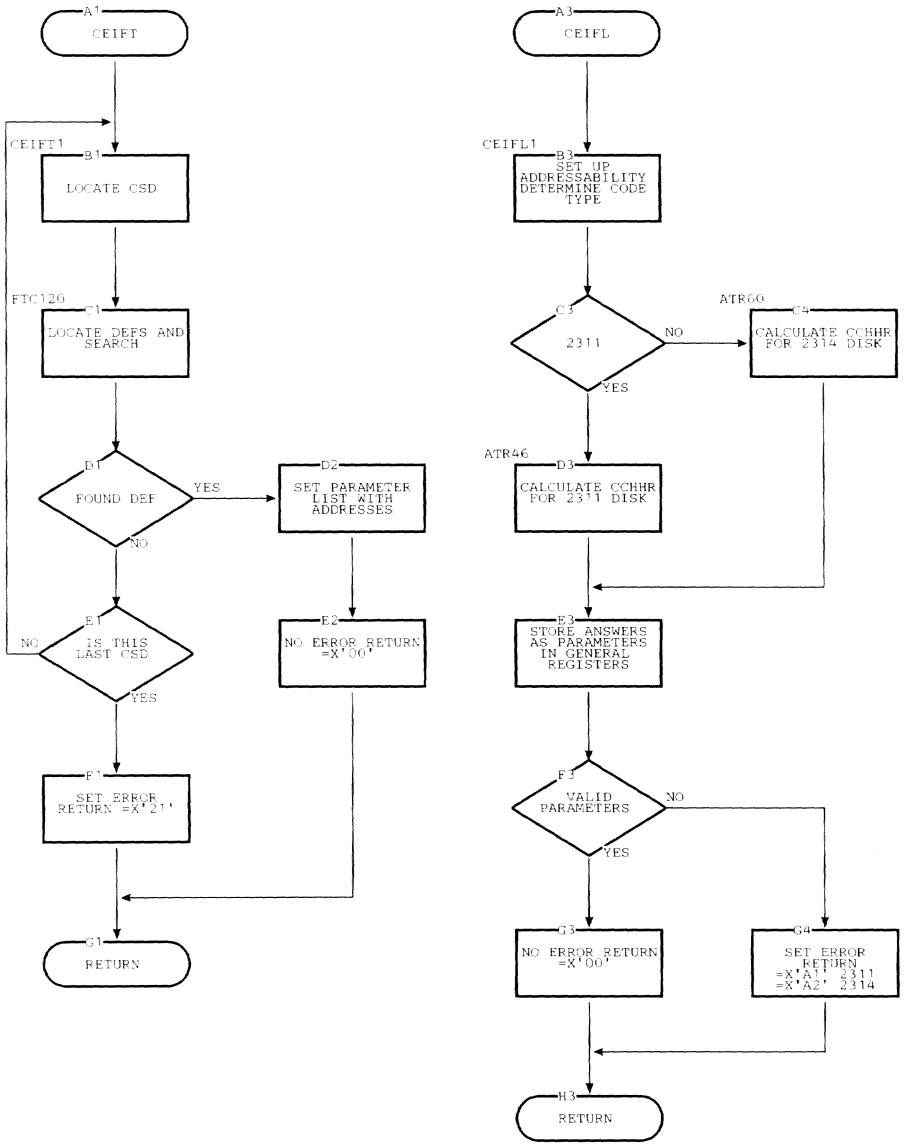


Chart AA. SYSBLD (CEIFA) (Page 13 of 16)

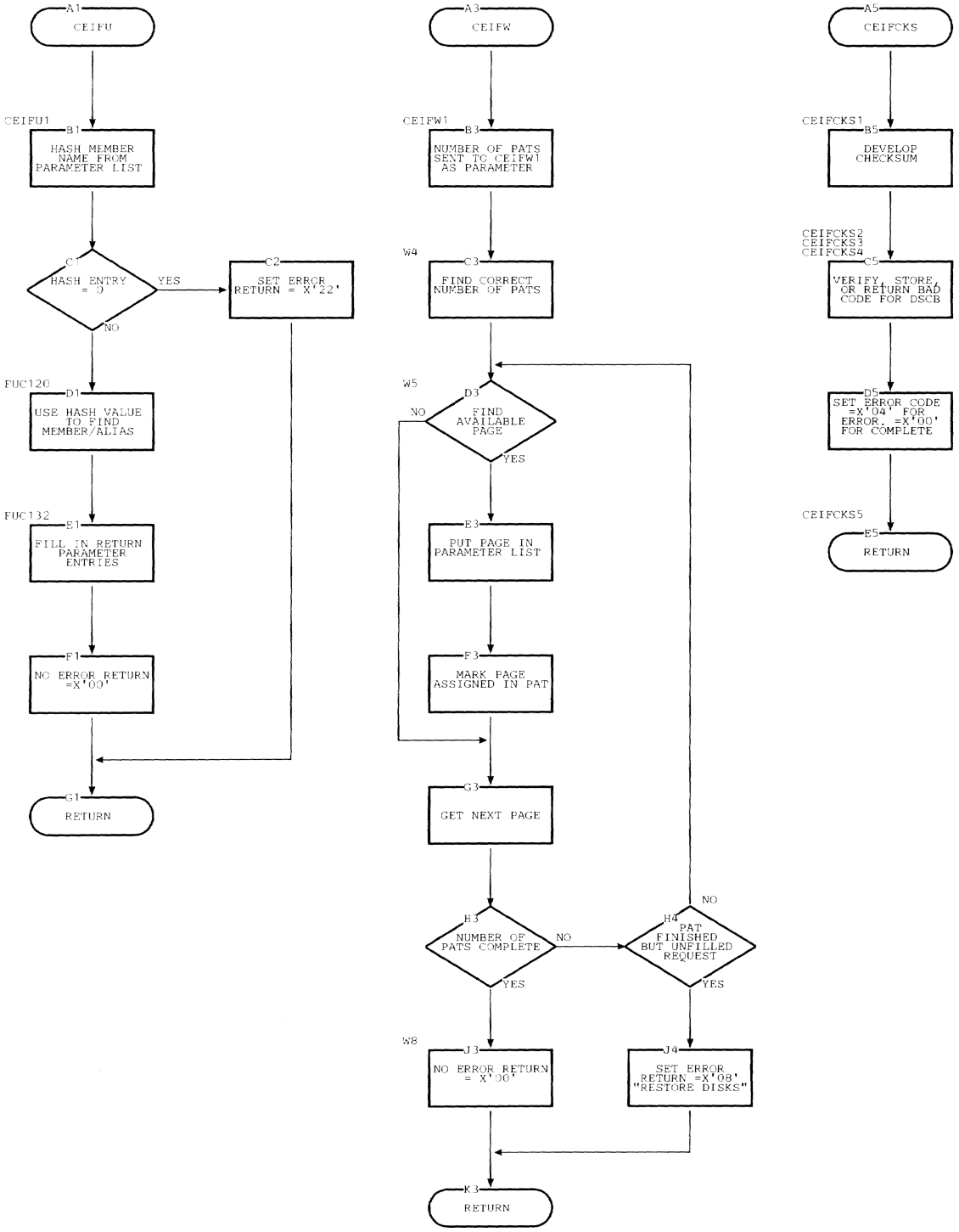


Chart AA. SYSBLD (CEIFA) (Page 14 of 16)

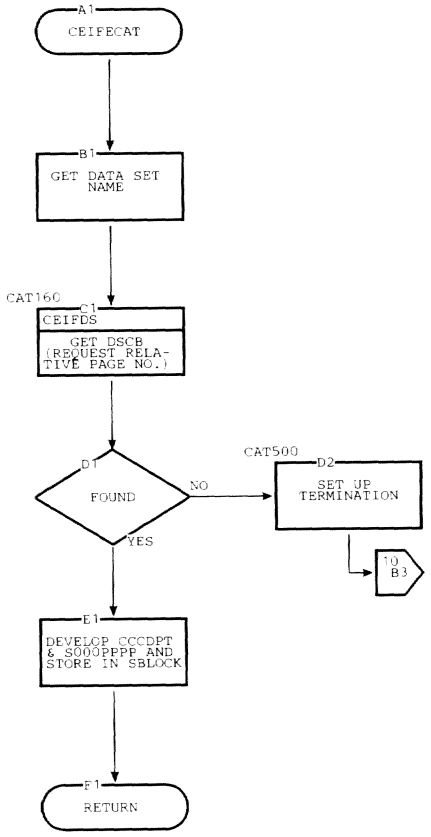


Chart AA. SYSBLD (CEIFA) (Page 15 of 16)

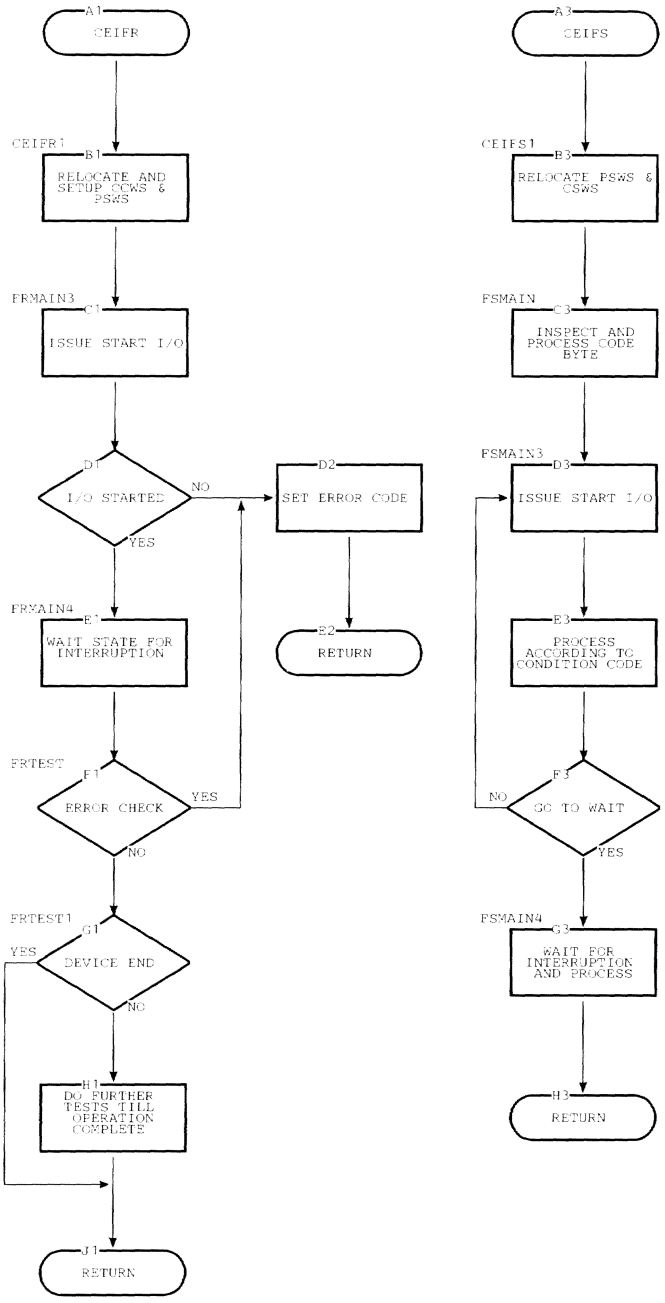


Chart AA. SYSBLD (CEIFA) (Page 16 of 16)

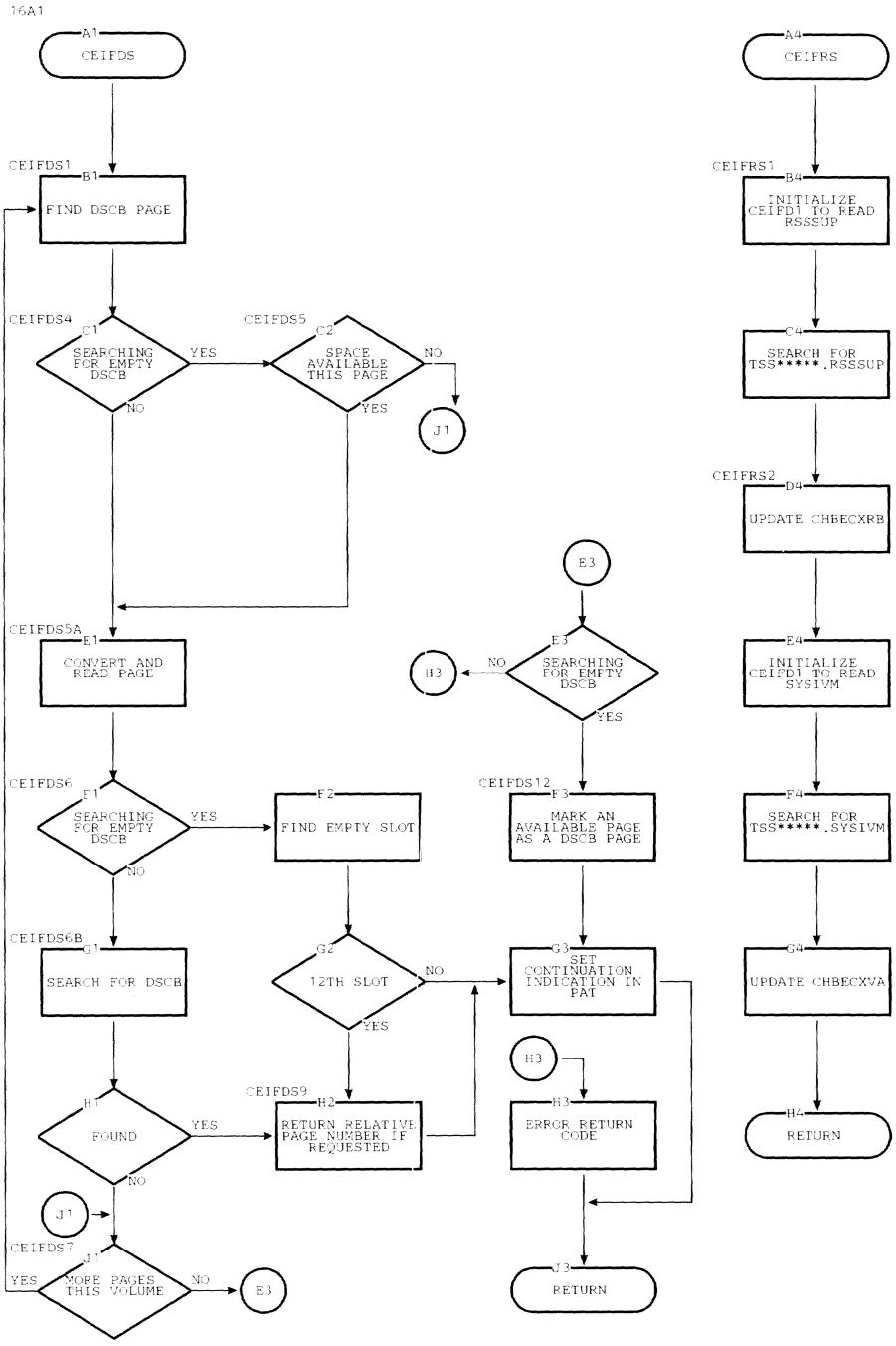


Chart AB. SYSBLD/STARTUP Prelude (CEIAP)

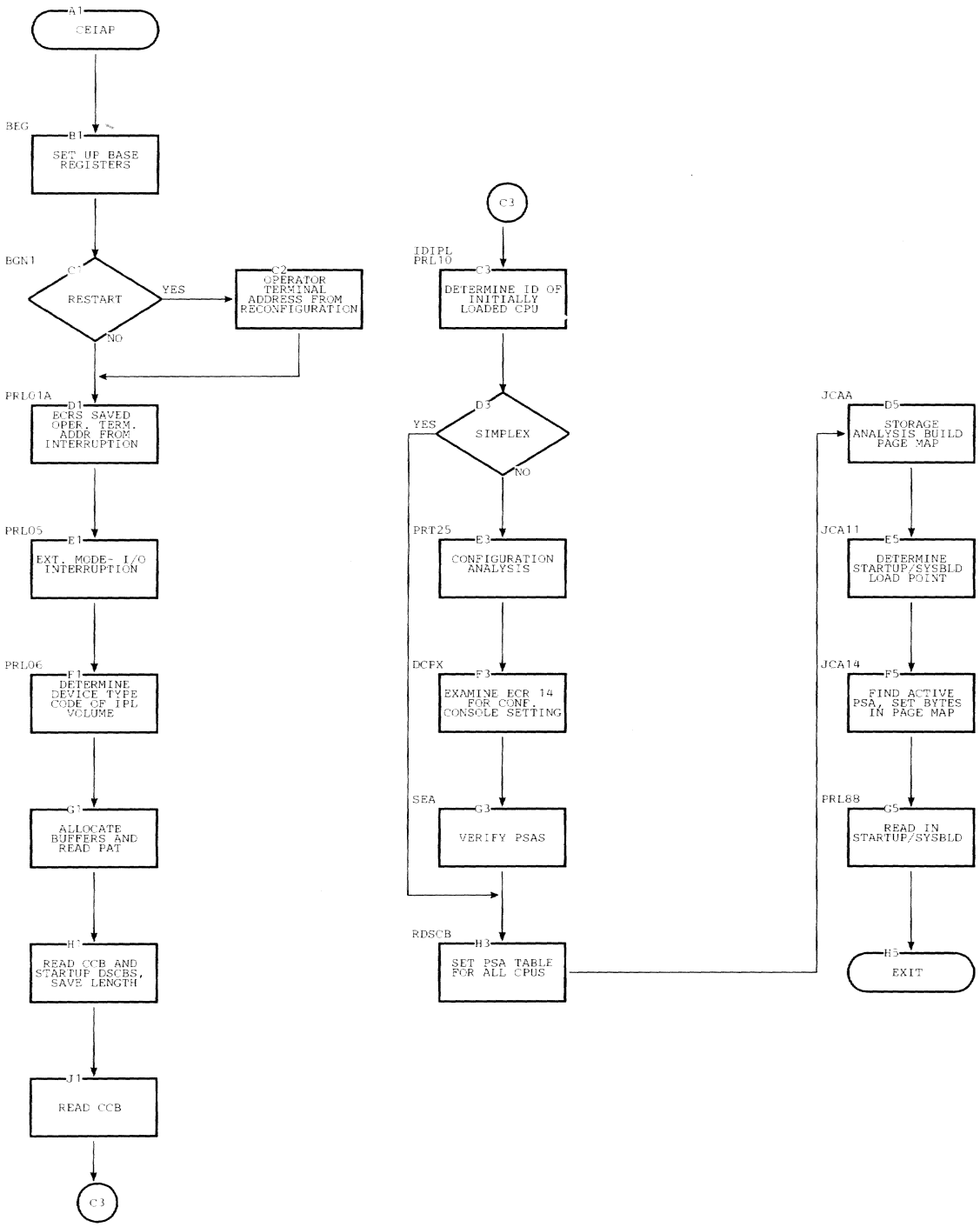


Chart AC. Startup (CEIAA) (Page 1 of 15)

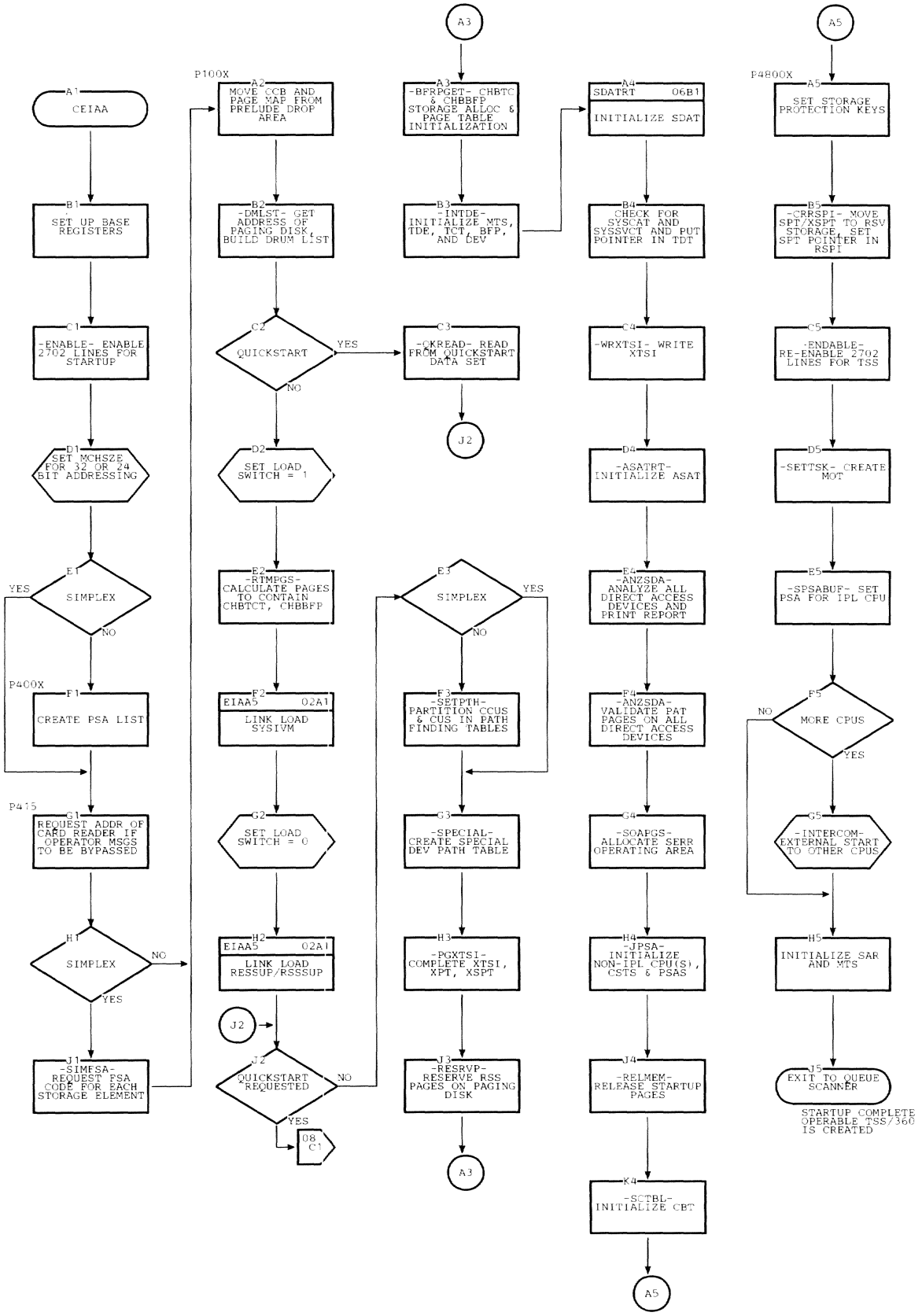


Chart AC. Startup (CEIAA) (Page 2 of 15)

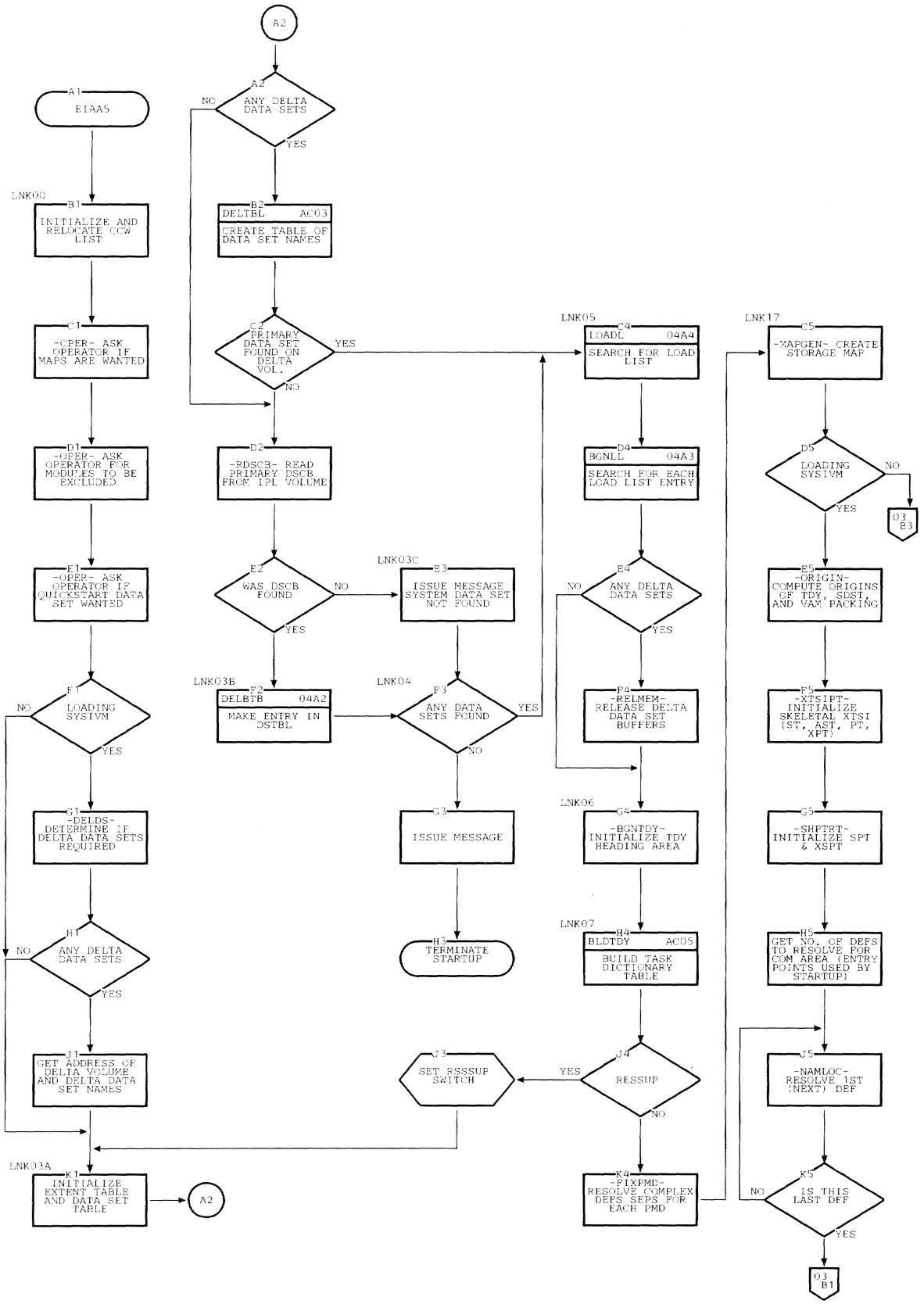


Chart AC. Startup (CEIAA) (Page 3 of 15)

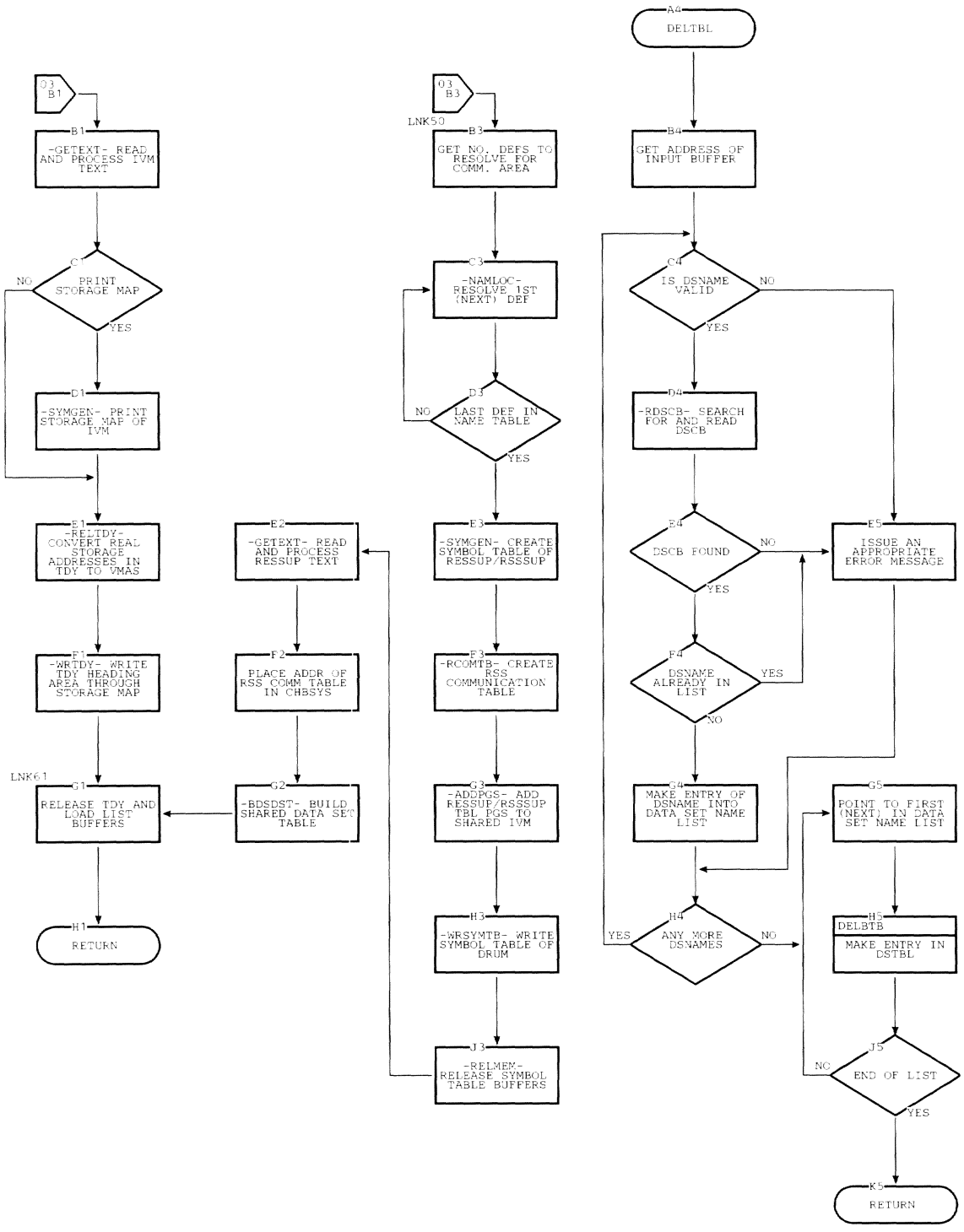


Chart AC. Startup (CEIAA) (Page 4 of 15)

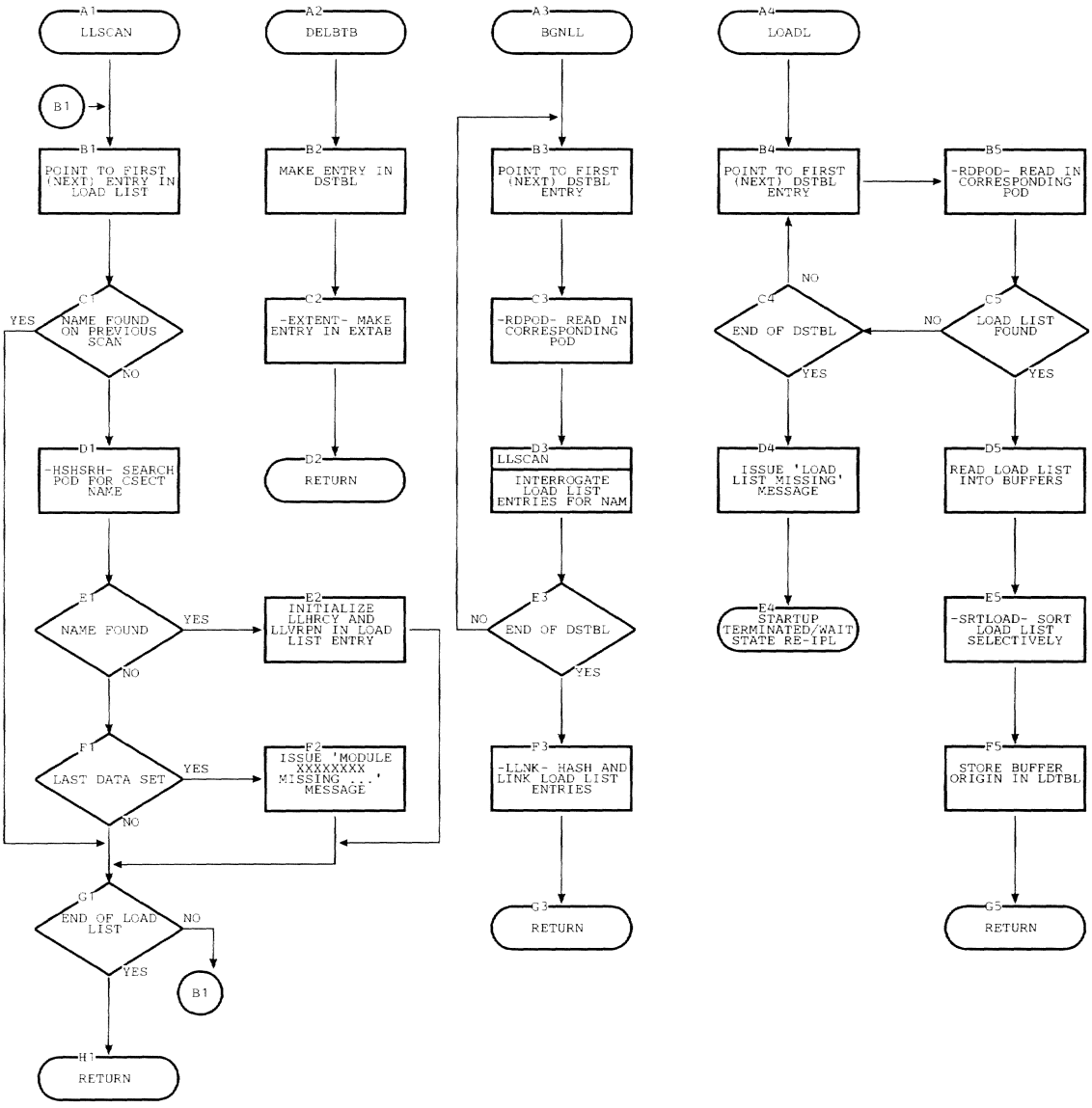


Chart AC. Startup (CEIAA) (Page 5 of 15)

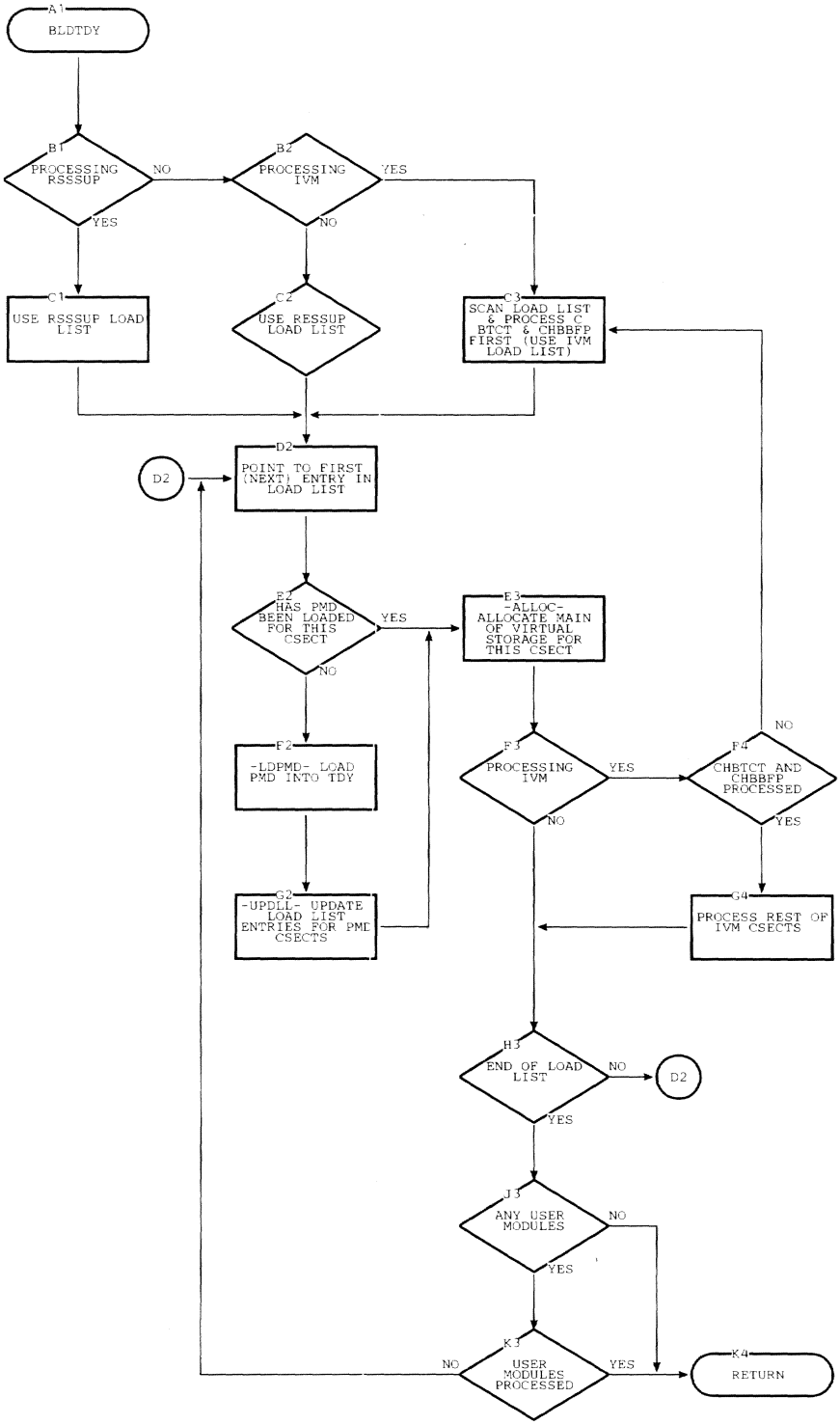


Chart AC. Startup (CEIAA)(Page 6 of 15): SDAT Mainline - SDATRT

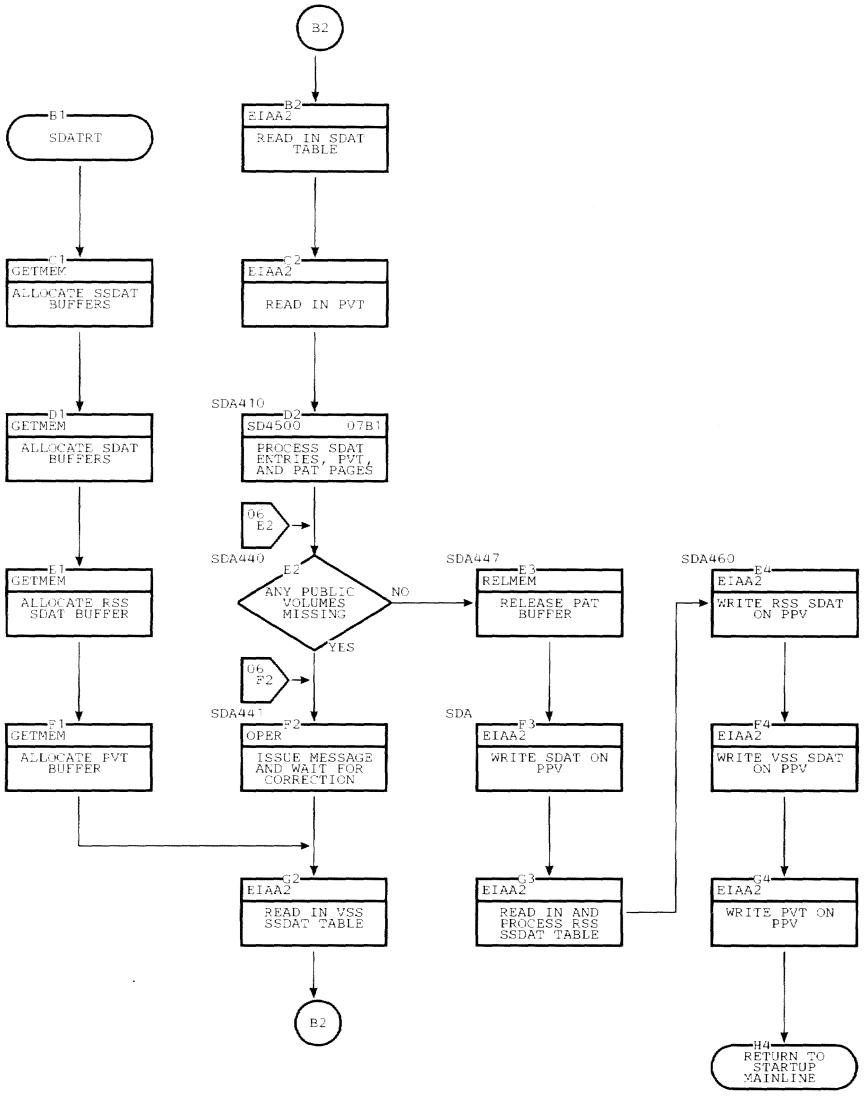


Chart AC. Startup (CEIAA)(Page 7 of 15): SDAT Entry Processor - SDA500

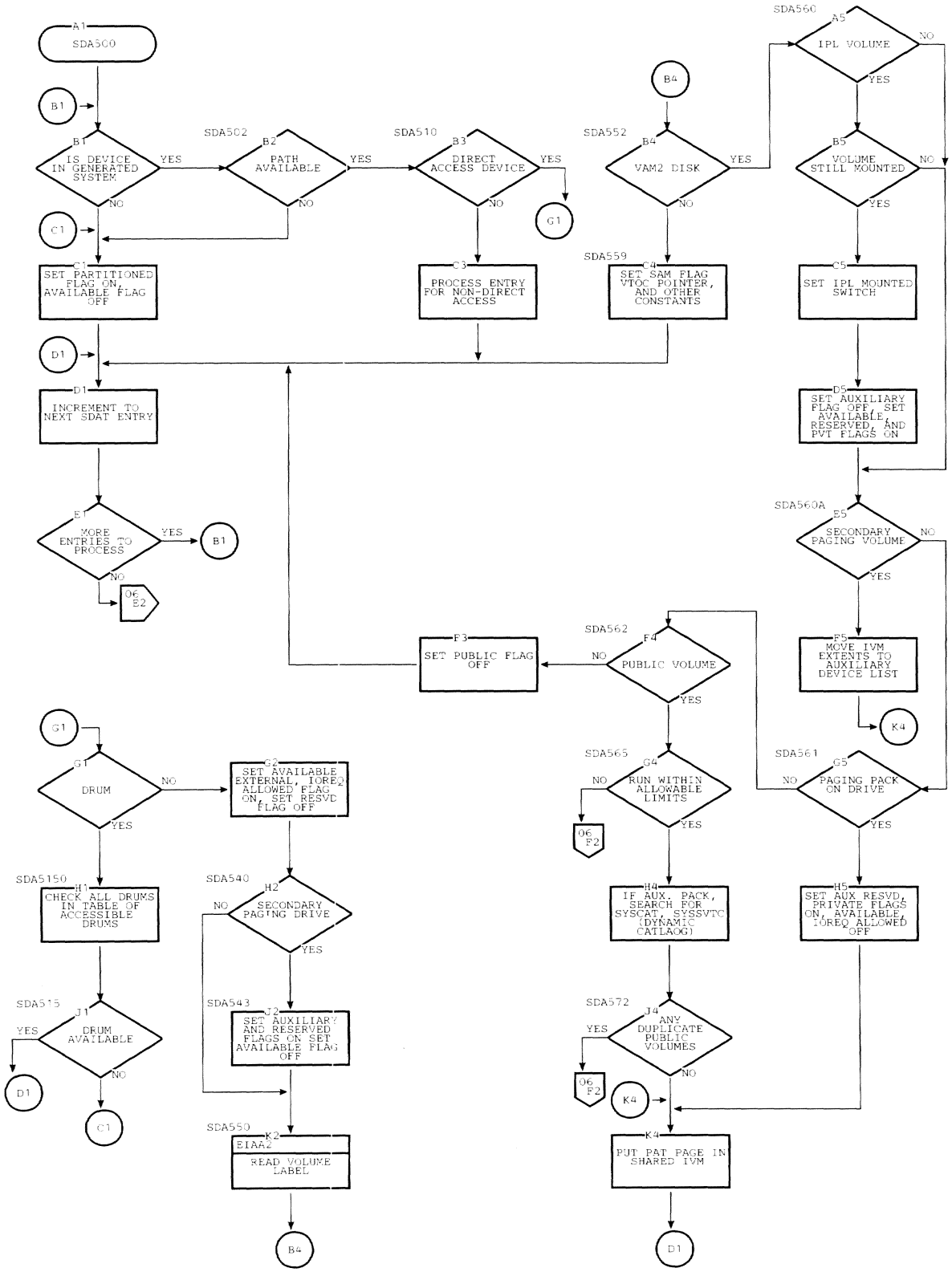


Chart AC. Quickstart (CEIAB) (Page 8 of 15): Quickstart Mainline - Initialization, Write IVM

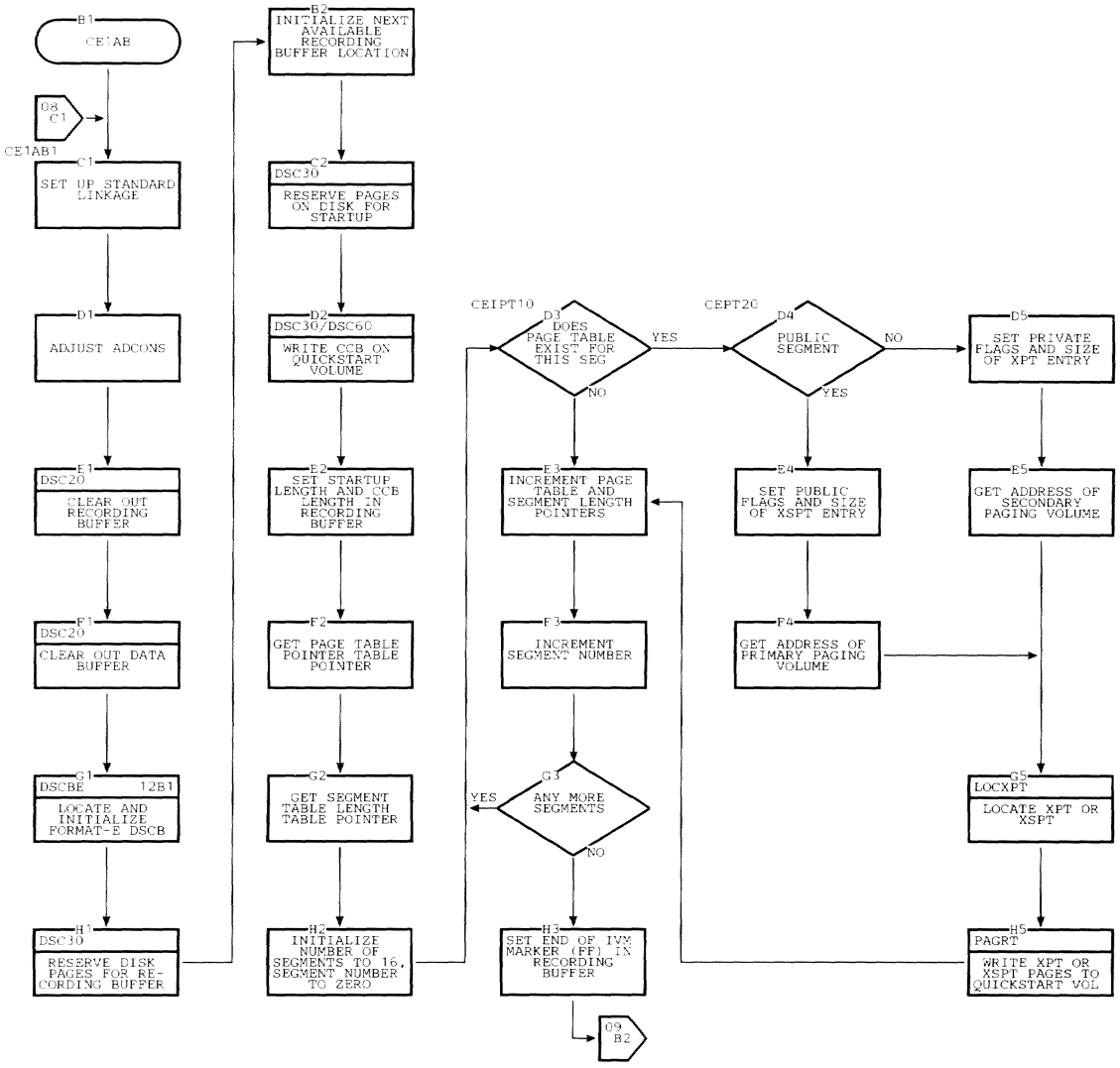


Chart AC. Quickstart (CEIAB) (Page 9 of 15): Quickstart Mainline - Write RESSUP, XTSI, Page Table Pages

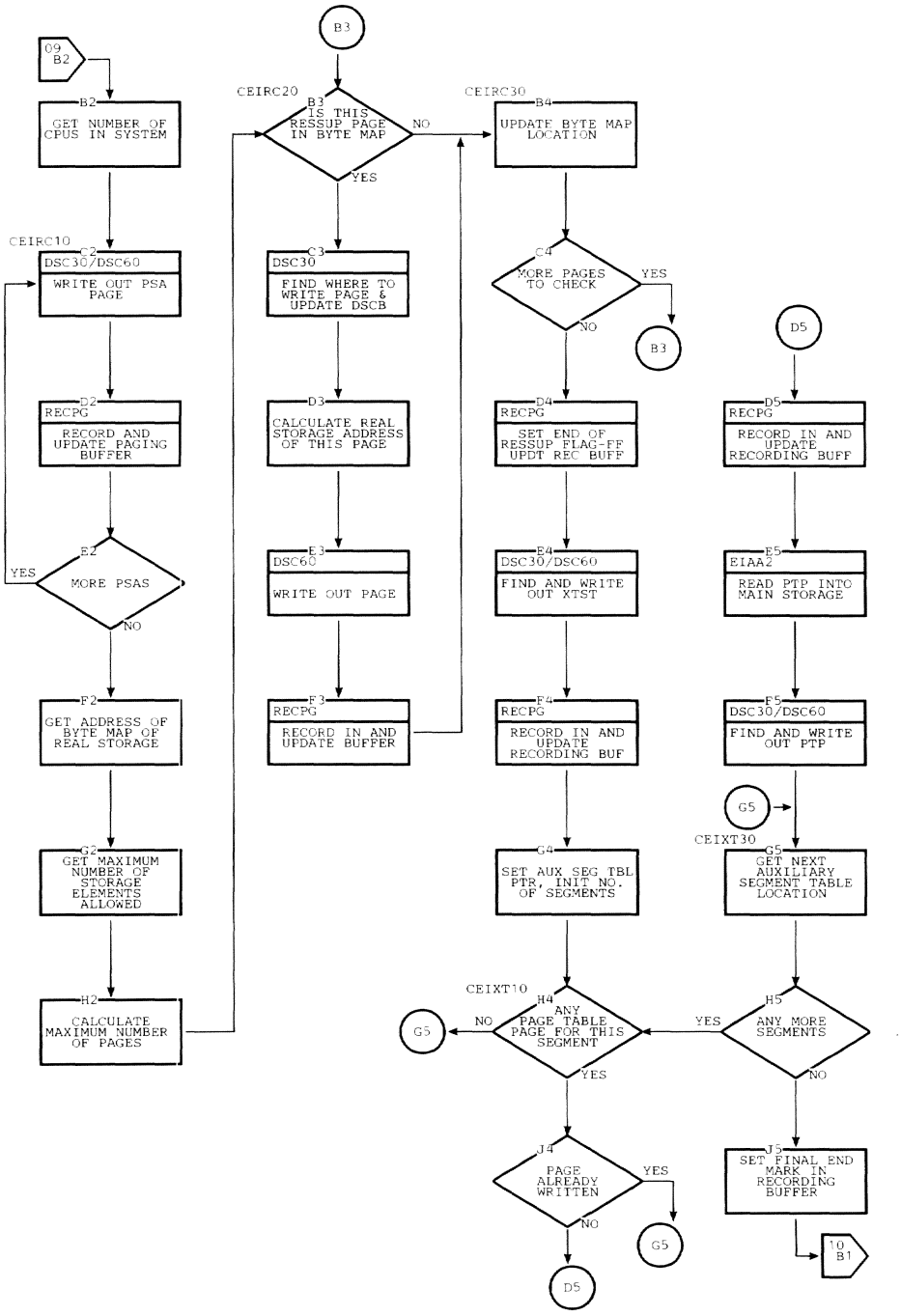


Chart AC. Quickstart (CEIAB)(Page 10 of 15): Quickstart Mainline Continued

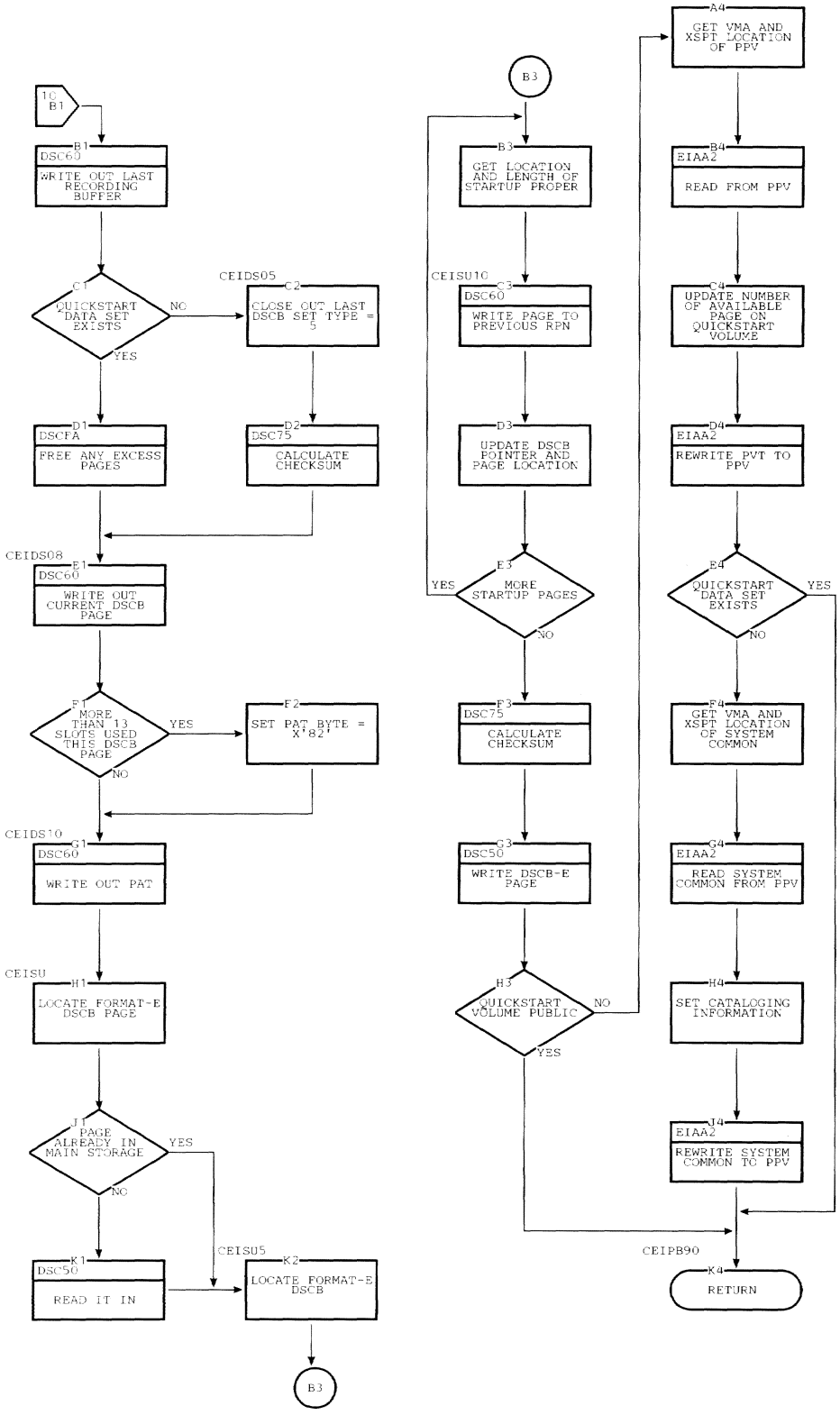


Chart AC. Quickstart (CEIAB)(Page 11 of 15): PAGRT

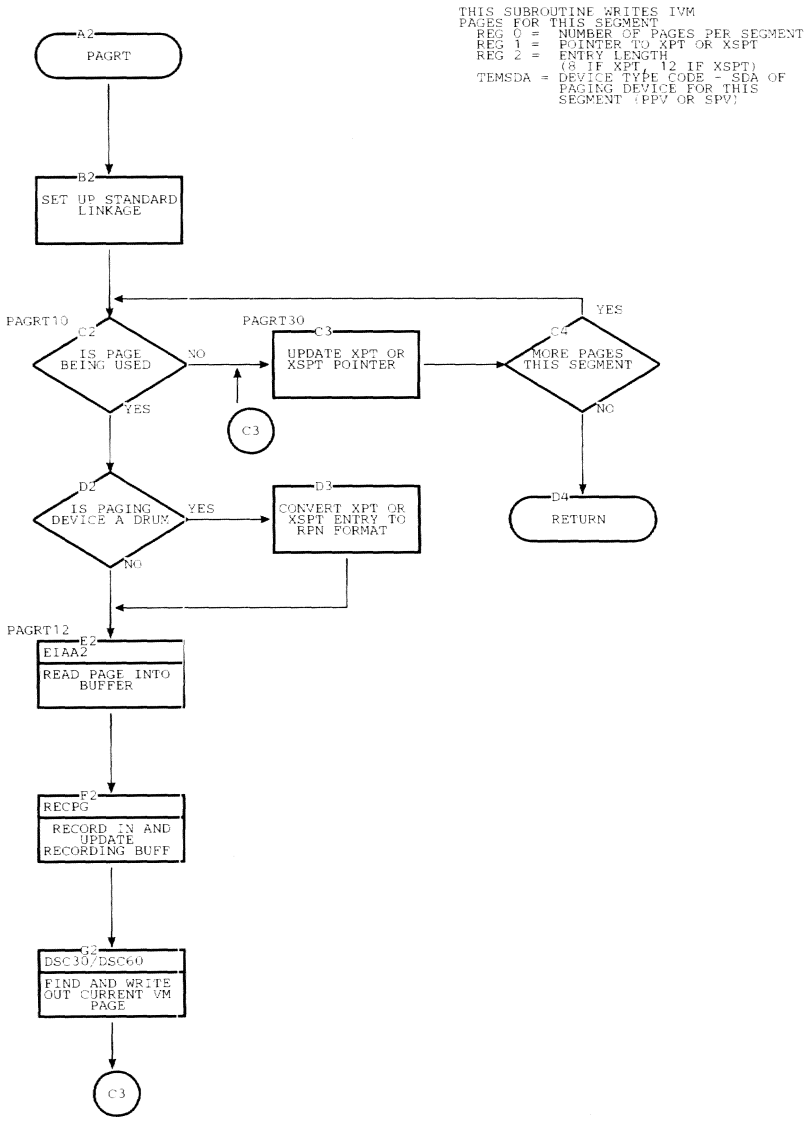


Chart AC. Quickstart (CEIAB)(Page 12 of 15): DSCBE Locate, Create Format-E DSCB

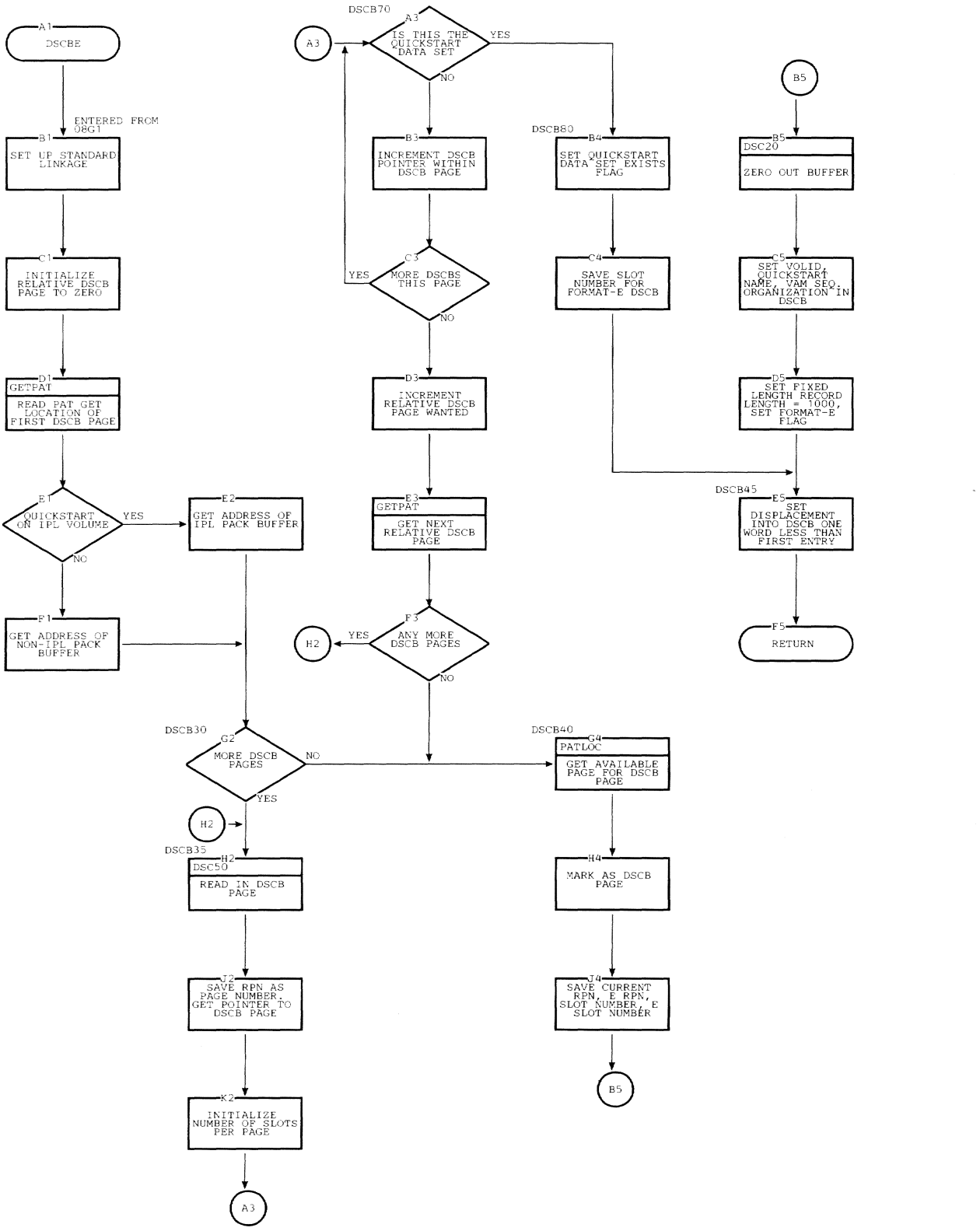


Chart AC. Quickstart (CEIAB) (Page 13 of 15): DSCF/DSCFA Page 1

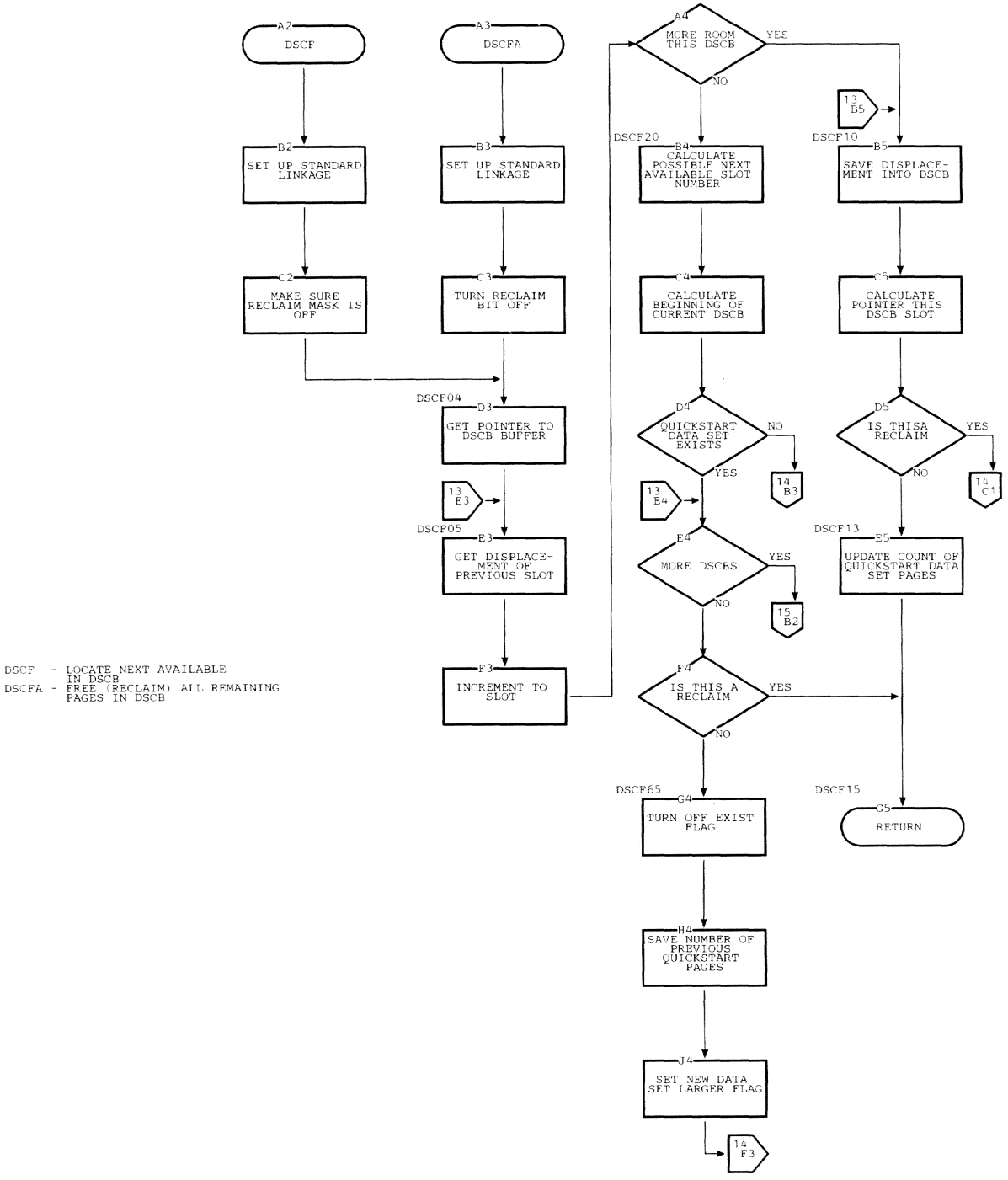


Chart AC. Quickstart (CEIAB)(Page 14 of 15): DSCF/DSCFA Page 2

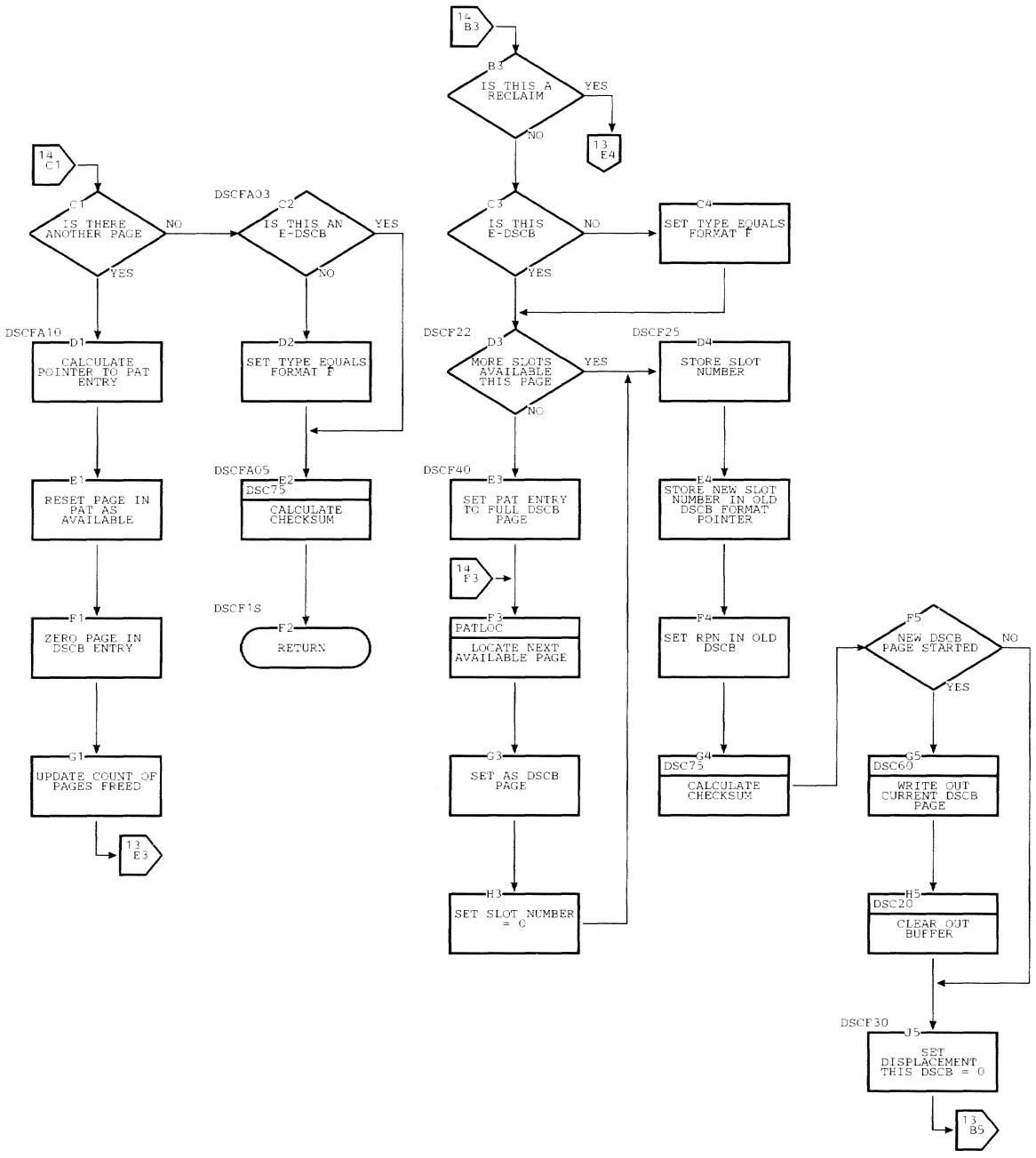


Chart AC. Quickstart (CEIAB) (Page 15 of 15): DSCF/DSCFA Page 3

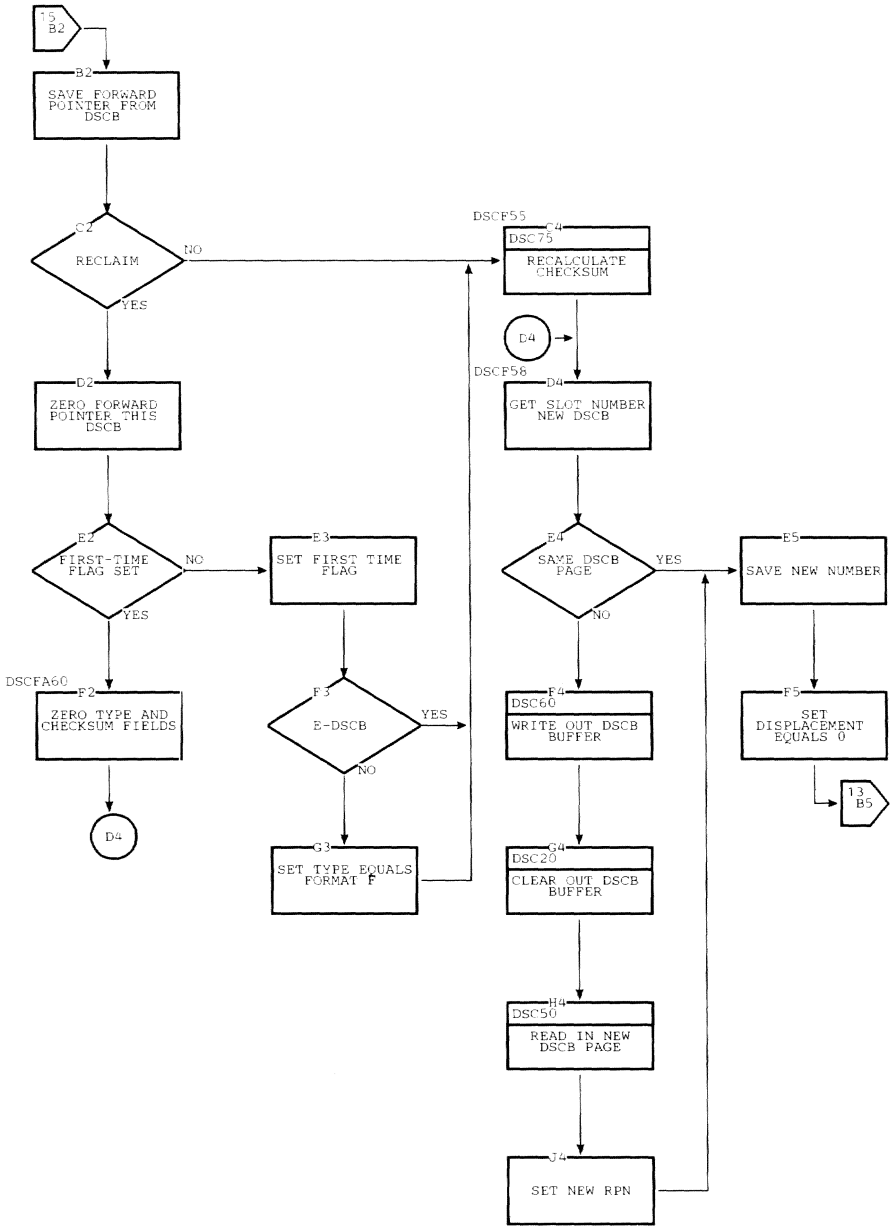


Chart AE. GENSCB Macro (CEIDA) (Page 1 of 2)

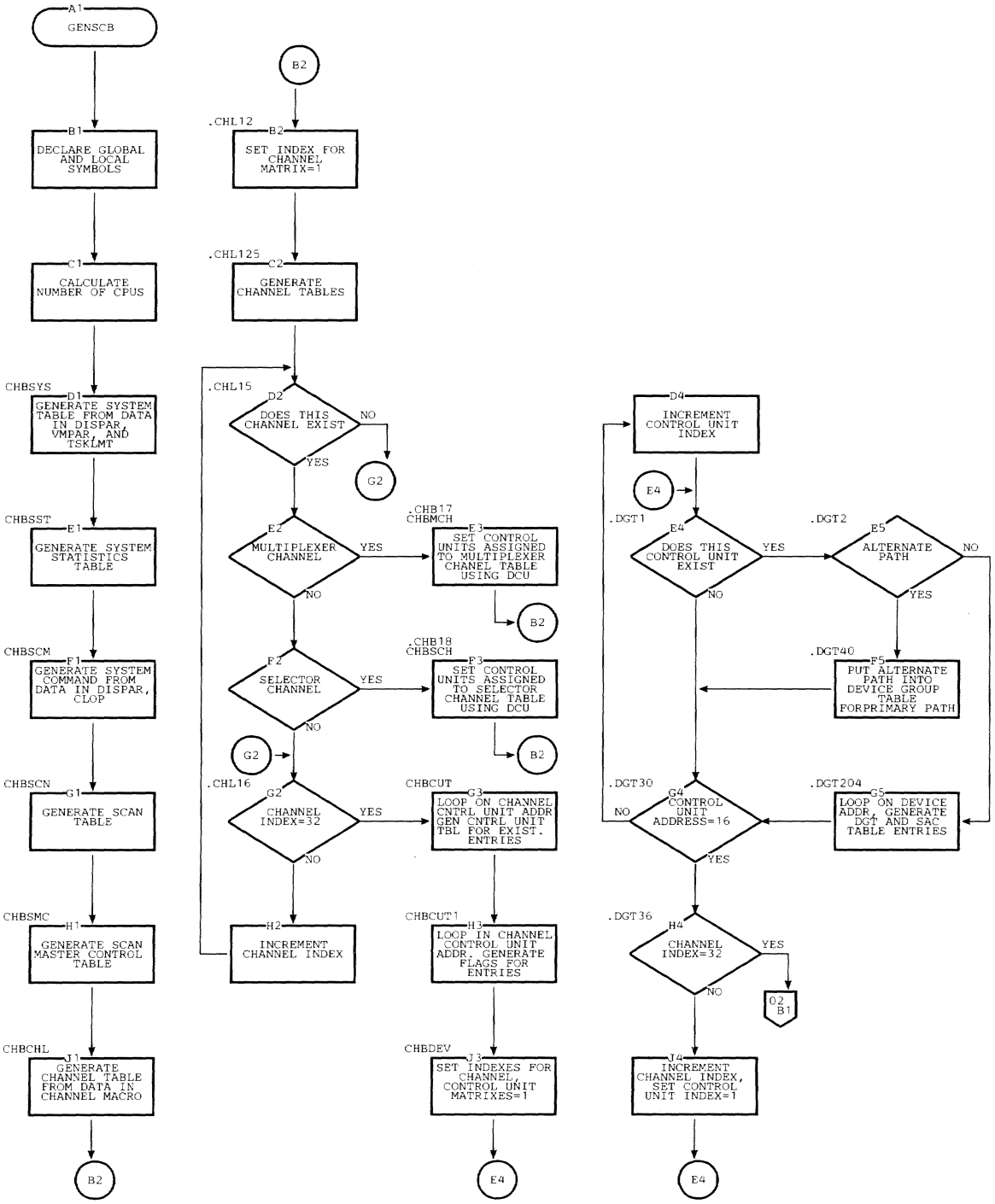
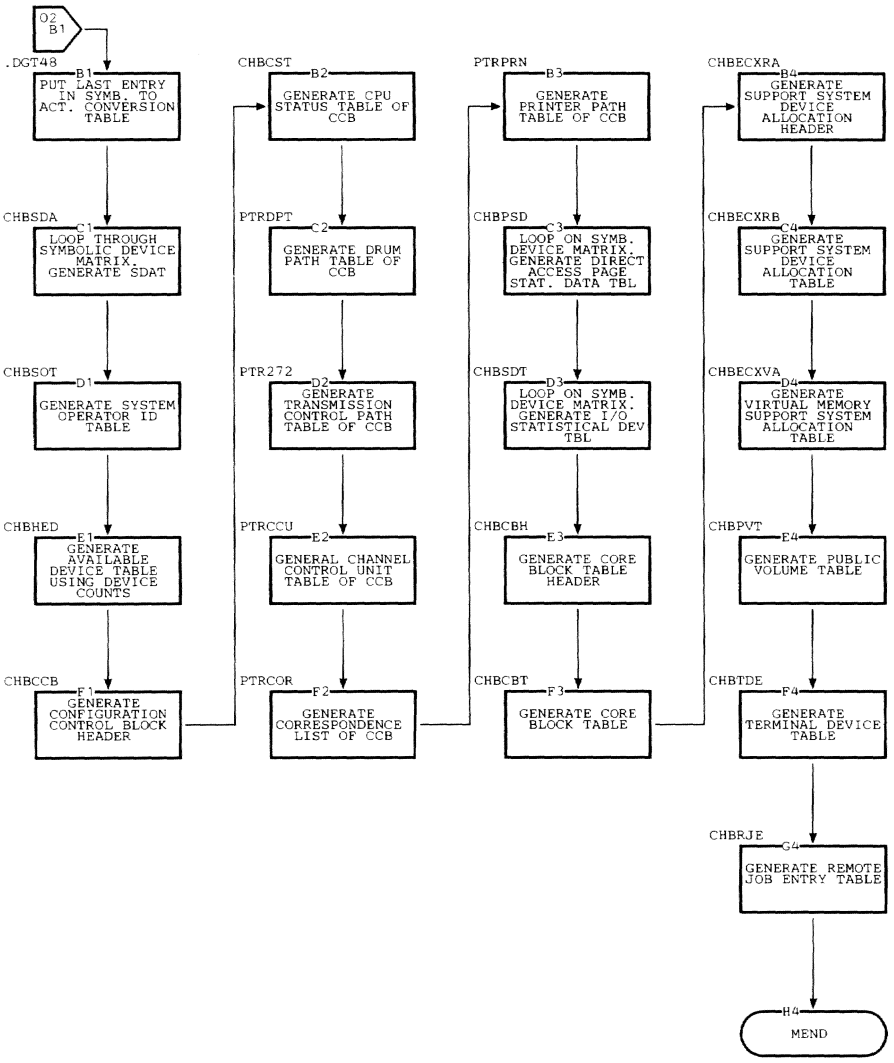


Chart AE. GENSCB Macro (CEIDA) (Page 2 of 2)



This appendix is constructed as follows: the fields are listed in the order in which they are set by the GENSCB macro; the name of each field is followed by a description of the entry, then the name of the macro in which the global symbol is set, the operand of that macro, and the default option are listed.

SYSTEM TABLE (CHBSYS)

FIELD	DESCRIPTION	MACRO	OPERAND	DEFAULT
SYSLOW	Low Core Threshold (LOW)	DISPAR	LCT(1)	1
SYSHI	Low Core Threshold (HIGH)	DISPAR	LCT(2)	10
SYSFLI	Flags	VMPAR	OPTIONS	00
SYSECB	Estimated Core Blocks Available			90
SYSXPG	Number of XTSI pages to trigger XTSI paging	TSKLMT	XTSIPGS	0
SYSMGPTP	Number of page table pages for migration			8
SYSTLM	System TSI Limit	TSKLMT	TASKS	50
SYSTID	Last Task ID Number Assigned			16
SYSRSC	Count of Pages in Supervisor Core's Reserve List			2
SYSMND	Lower Shared Page Drum Threshold			250
SYSMXD	Upper Shared Page Drum Threshold			300
SYSDATA	Number of entries to expand page table	TSKLMT	VLPTE	3
SYSMXS	Maximum Number of Shared Pages			30
SYSMNS	Minimum Number of Shared Pages			15
SYSSCH	SVC Charge Value			77
SYSMWX	Maximum Shared Pages To Purge	DISPAR	PURGSH	22
SYSMWT	Scan Shared Pages Threshold	DISPAR	THRESH	10
SYSBUF	Buffer Size On Drum	DISPAR	BUFSIZ	64
SYSXTS	XTSI size limit	TSKLMT	PGTBL	16
SYSPSL	Maximum number of public segments allowed	TSKLMT	PSLMT	6
SYSAST	Auxiliary Stop Threshold	DISPAR	AUXSP(1)	SYSBUF+10_{10}
SYSAPT	Auxiliary Primary Threshold	DISPAR	AUXSP(2)	100
SYSRS1	PSW To Enter RSS Via Program Interruption			00040F00 V(CEHAPA)
SYSRS2	PSW To Enter RSS Via SVC Interruption			00040F00 V(CEHASA)
SYSRS3	PSW To Enter RSS Via Interruption Key			00040F00 V(CEHAEA)
SYSRS4	PSW To Enter RSS Via I/O Interruption			00040F00 V(CEHADA)
SYSRS5	PSW To Enter RSS Via Channel Interruption Processor			00040F00 V(CEHACA)
SYSRS6	PSW To Enter RSS Via GQE on TSI			00040F00 V(CEHAQA)
SYSDLY	TSEND Delay Time			524288
SYSDTRL	Delta Length (Timer Count Per Master Tick)			255
SYSTSEM	TSE Maximum Count			255
SYSTCR	Task Core Requirement			55
SYSIDL	Idle Timer Setting			9615
SYSCCAIV	Contiguous Storage Interruption Timer			3
SYSPT1	Pointer to fixed area of CHASST			V(SSTLHT)
SYSPT2	Pointer to drum area of CHASST			V(SSTLH2)
SYSPT3	Pointer to disc area of CHASST			V(SSTLH3)
SYSBLK	Maximum pages blocked to drum	DISPAR	BLKSZE	12
SYSBLK2	Combined maximum pages blocked to drum and disk	DISPAR	BLKSZE	20

SYSTEM COMMON (CHBSCM)

SCMLPR	Installation Legitimate Privilege Classes	CLOP	PRVLG	78000000_{16}
SCMPDC	Number of Public Devices			1
SCMCFM	Installation Default for Card Forms	CLOP	CFM	5081
SCMPFM	Installation Default for Printer Forms	CLOP	PFM	419K
SCMNCP	Nonconversational order priority			
SCMOCF	Operator Confirmation Default	CLOP	OPCONF	Y
SCMBPR	Batch priority code default			
SCMOMS	Operator Message Option Default	CLOP	OPMSG	M
SCMTAP	Installation Tape Default	CLOP	TATYPE	00
SCMTA1	Number of 7-track Tapes	DEVGRP		0
SCMTA2	Number of 7-track Data Conversion Tapes	DEVGRP		0
SCMTA3	Number of 9-track Tapes	DEVGRP		2
SCMDA	Installation Direct Access Default	CLOP	DATYPE	01
SCMDA1	Number of 2311 Direct Access Devices	DEVGRP		3
SCMDA3	Number of 2314 Direct Access Devices	DEVGRP		0
SCMPTN	Number of Paper Tapes	DEVGRP		0
SCMPUN	Number of Punches	DEVGRP		1
SCMRDN	Number of Readers	DEVGRP		1
SCMPRN	Number of Printers	DEVGRP		1
SCMDET	Total Number of Devices	DEVGRP		A
SCMTDN	Installation Tape Density Default	CLOP	DEN	43
SCMORG	Installation DS Organization Default	CLOP	DSORG	05
SCMLAB	Installation Label Type Default	CLOP	LABTYP	02
SCMPRV	Installation Default Privilege Class	CLOP	PRVLG(1)	D
SCMPSP	Installation Default Primary Page Space Allocation	CLOP	DAPAGES(1)	16
SCMSSP	Installation Default Secondary Page Space Allocation	CLOP	DAPAGES(2)	08
SCMPSC	Installation Default Primary Cylinder Space Allocation	CLOP	DACYLS(1)	2
SCMSSC	Installation Default Secondary Cylinder Space Allocation	CLOP	DACYLS(2)	1
SCMSST	Installation Default Secondary Track Space Allocation	CLOP	DATRKS	8
SCMUL1	User Library Primary Page Space Allocation	CLOP	LIBPGS(1)	16
SCMUL2	User Library Secondary Page Space Allocation	CLOP	LIBPGS(2)	08
SCMMAV	Default Maximum Auxiliary Storage	CLOP	MAV	256
SCMAUX	Auxilliary space overcommitment			
SCMIDP	Installation default external priority	CLOP		8
SCMQST	Minimum pages on ACV permitting allocation	GENSCB		
SCMIPL	Installation Default Prompt Limit	CLOP	PRMTLMT	5
SCMATH	Installation Default Authorization	CLOP	AUTH	U
SCMFIR	Installation Default FORTRAN Interruption Recovery	CLOP	FIR	Y
SCMTIM	Task time interval	VMPAR	TIME	01000000
SCMMVD	Multi-volume flag	GENSCB		80
SCMATV	ACV threshold valve	GENSCB		12C

CHANNEL TABLE (CHBCHL)

CHLCUT	Control Unit Table Pointer			CHBCUT
CHLPTR	Multiplexer/Selector Channel Table Pointers	CHANNEL	SEL,MPX	0
CHLCS	Control Units Assigned Table Size	DCU	ADDRESS	00
CHLFLG	Flags for Each Channel	CHANNEL	SEL,MPX	02
CHLDIGn	Device Interaction Groups (Note)	DEVGRP	NONE	N/A

Note: No entries are made for non-existent channels for the fields CHLDIG1 and CHLDIG2. The setting of the fields is determined by analysis by SYSGEN, the analysis being done on the scan table (CHBSCN).

MULTIPLEXER CHANNEL TABLE (CHBMCH)

MCHF	Not used			0
MCHFLG	Flag Field	DCU	ADDRESS	4000
MCHCTD	Control Unit Table Displacement			0

SELECTOR CHANNEL TABLE (CHBSCH)

SCHFLG	Flag Field	DCU	ADDRESS	4000
SCHCTD	Control Unit Table Displacement			0

Note: MCHFLG and SCHFLG are set to X'0000' if the control unit exists, and to X'4000' if the control unit does not exist. MCHCTD and SCHCTD are calculated for existing control units by SYSGEN.

CONTROL UNIT TABLE (CHBCUT)

CUTMAX	Number of Control Unit Entries in Table	DCU	ADDRESS	0
CUTFP	Control Unit Table Flag Area			CHBCUT1
CUTDGP	Device Group Table Pointer for Control Unit N	DCU	ADDRESS	
CUTFLG	Flags for Control Unit N	DCU	ADDRESS	
CUTS	Switch Flag in CUTFLG			00
CUTC	Entry Type Flag in CUTFLG			00
CUTDIGn	Device Interaction Group	DEVGRP	NONE	N/A

Note: No entries are made for nonexistent control units, for the fields CUTDGP and CUTFLG. The settings of the flags CUTS and CUTC are determined by analysis by SYSGEN, the analysis being done on the ADDRESS operand of the DCU Macro.

No entries are made for nonexistent control units for the fields CUTDIG1 through CUTDIG8. The setting of the fields is determined by analysis by SYSGEN, the analysis being done on the scan table (CHBSCN).

DEVICE GROUP TABLE (CHBDEV)

DEVLOCK	Lock Byte			
DEVMAX	Maximum Device Address in Table	DEVGRP	ADDRESS	0
DEVF	Table Flags	DEVGRP	PATH	0
DEVAEP	Asynchronous Interruption List Pointer	DEVGRP	PATH	0
DEVPP	Actual Paths to Device	DEVGRP	PATH	SYSGEN VCON
DEVFLG	Device Flags			0
DEVTP	Device Type	DEVGRP	UNIT	01
DEVSDA	System Symbolic Device Address	DEVGRP	ADDRESS	0
DEVASD	System Symbolic Device Address	DEVGRP	ADDRESS	0
DEVI	Asynchronous Interruption Flags			0

SYMBOLIC TO ACTUAL ADDRESS CONVERSION TABLE (CHBSAC)

SACDA	Actual Device Address	DEVGRP	ADDRESS	0
SACDP	Device Group Table Pointer			SEE NOTE

Note: The Device Group Table Pointer is the displacement from CHBDEV to the corresponding Device Group Table.

SYMBOLIC DEVICE ALLOCATION TABLE (CHBSDA)

SDAHPS	First Public Device			0
SDAHAL	Address of Last Entry			SDALST
SDALOC	System Lock Byte			0
SDAFLA	First Flag Byte	DEVGRP	TYPE, IOREQ	82
SDASDA	Symbolic Device Address	DEVGRP	ADDRESS	0
SDADEV	Device Code	DEVGRP	UNIT, FEATURE	00000000
SDATID	Task ID			0
SDAMRB	Maximum Number of IORCBs	DEVGRP	MAXIO	2
SDAUSC	User Count			0
SDAFLB	Second Flag Byte	DEVGRP	TYPE	80
SDADM4	Third Flag Byte			0
SDASPC	Total Space Capacity of Volume			0
SDATAP	Tape Position Code			0
SDAVID	Volume ID			0
SDANLC	Number of Logical Cylinders/Volume			0
SDALCS	Number of Tracks/Logical Cylinder			0
SDATRL	Number of Available Bytes/Track			0
SDAOHI	Overhead for Keyed Record			0
SDAOHL	Overhead for Last Keyed Record on Track			0
SDAOHK	Overhead Bytes to be Subtracted if no Key			0
SDATOL	Tolerance/512 Gives Effective			0

SYSTEM OPERATOR ID TABLE HEADER (CHBSOT)

SOTLNG	Number of Table Entries - Binary			4
SOTBCK	Symbolic Device Address of Backup Terminal	OPCHSL	ADDRESS	0
SOTUID	User ID - EBCDIC			
SOTDES	Destination Flag			
SOTLOG	Logon Flag			
SOTTID	Task ID - Binary			
SOTSIN	Console Symbolic Device Address			

SYSTEM OPERATOR ID TABLE (CHBSID)

SIDUID	User ID - EBCDIC			SYSOP000, 1, 2, 3
SIDDES	Destination Flag			00, 01, 02, 04
SIDLOG	Logon Flag			01, 00, 00, 00
SIDTID	Task ID - Binary			0
SIDSIN	Console Symbolic Device Address	OPCNSL	ADDRESS	0

AVAILABLE DEVICE TABLE (CHBHED)

HEDLCK	Lock Byte			0
HEDCNT	Count of Subqueue Headers	DEVGRP	UNIT, ADDRESS	0
HEDSPR	Spare Bytes			0

SUBQUEUE HEADERS (CHBAHD)

AHDDTC	Device Type Code (Same as SDSDEV in SDAT)	DEVGRP	UNIT	SAME AS SDAT
AHDADR	Pointer to First Subqueue Entry			SET BY SYSGEN
AHDLCK	Header Lock Byte			00
AHDCNT	Number of Entries in the Subqueue	DEVGRP	UNIT, ADDRESS	NO HEADER

SUBQUEUE ENTRIES (CHBAVE)

AVEDEV	Full Device Code (Same as SDASEV in SDAT)	DEVGRP	UNIT	NO ENTRY
AVEPNT	Pointer to SDAT Entry			SET BY SYSGEN

CONFIGURATION CONTROL BLOCK HEADER (CHBCCB)

CCBNDM	Number of Drums of Installation	DEVGRP	UNIT, ADDRESS	1
CCBDPP	Relative Pointer to Drum Path Table			SEE NOTE
CCBTPP	Relative Pointer to Transmission Control Path Table			SEE NOTE
CCBNCC	Number of Channel Controllers at Installation	CHANNEL	MPX,SEL	
CCBCPT	Relative Pointer to Channel Controller Table			SEE NOTE
CCBPCL	Relative Pointer to Correspondence List			SEE NOTE
CCBNPR	Number of Printers at Installation	DEVGRP	UNIT, ADDRESS	
CCBPPT	Relative Pointer to Printer Path Table			SEE NOTE
CCBLSD	Length of Shared Data Set Table	VMPAR	SDST	10
CCBLDA	Low Drum Availability Constant	DISPAR	LDMTR	2%
CCBCON	Maximum number of conversational tasks	TSKLMT	CONV	
CCBMTT	Maximum number of MTT administrator's tasks	TSKLMT	MTTADM	
CCBVMB	Storage Assigned to Variable-length Control Sections	VMPAR	VCSLNG	20
CCBTER	Number of user and operator's terminals	GENSCB		
CCBBUF	Buffer size for input data	VMPAR	TBUFS	200
CCBBAT	Maximum number of patch tasks	TSKLMT	BATCH	
CCBBAK	Maximum number of background tasks	TSKLMT	BACK	

Note: Relative Pointers above are calculated by SYSGEN, and each is equal to the number of bytes from CHBCCB to the beginning of the specific table.

CCB CPU STATUS TABLE (CCBCST)

CSTID0	Identity Byte			0
CSTMDL	Model Number of CPUs at Installation	CPU	MODEL	2
CSTNOP	Number of CPUs at Installation	CPU	NUM	1
CSTNAP	Number of Storage Elements in TSS Domain	STEM	NUM	2
CSTSET	Relative Pointer to SE Status Table			# Bytes from CCBCST
CSTID1	ID Field for External Interruption			SEE NOTE
CSTID2	ID Field for External Start			BELOW
CSTID3	Interruption Code on Malfunction Alert			
CSTCST	CPU Status (00=Available)	CPU	NUM	80
CSTPF1	Primary Prefix	CPU	PRFX	0
CSTPF2	Alternate Prefix	CPU	ALTPRFX	0
CSTSST	SE Status (00=Available)	STEM	NUM	80
CSTFSA	Floating Storage Address			0

Note: The following values are filled in by SYSGEN for existing CPUs:

CPU#	ID1	ID2	ID3
1	80	08	20
2	40	04	10
3	20	02	08
4	10	01	04

CCB DRUM PATH TABLE (CCBDPT)

CCBNPD	Number of Paths to This Drum	DEVGRP	PATH	0
CCBPTD	Path to Drum			NO ENTRY

CCB TRANSMISSION CONTROL PATH TABLE (CCBTPT)

CCBADD	Path to Transmission Control Line		DEVGRP	UNIT	NO ENTRY
CCBDTC	Device Type Code	Same as SDAT	DEVGRP	UNIT	NO ENTRY
CCBDCL	Device Class	SDADEV Field	DEVGRP	UNIT	NO ENTRY
CCBUNT	Unit Type		DEVGRP	UNIT	NO ENTRY
CCBOPF	Optional Features		DEVGRP	FEATURE	NO ENTRY

CCB CHANNEL CONTROL UNIT TABLE (CCBCCT)

CCBNCH	Number of Channel Connected to Channel Cont.	CHANNEL	MPX,SEL	0
CCBCAD	Channel Address	CHANNEL	MPX,SEL	NO ENTRY

CCB CORRESPONDENCE LIST (CCBCLT)

CCBPCM	Path to Control Unit	DEVGRP	PATH	NO ENTRY

CCB PRINTER PATH TABLE (CCBPRT)

CCBNPP	Number of Paths to This Printer	DEVGRP	PATH	0
CCBPTP	Path to Printer	DEVGRP	PATH	NO ENTRY

DIRECT ACCESS PAGING STATISTICAL DATA RECORD HEADER (CHBPSD)

PSDLSD	Length of SDR Entry (80 Bytes)	80
PSDLWA	Last Word Address	LASTWORD

DAPSDR ENTRY

PSDSDA	Symbolic Device Address	DEVGRP	ADDRESS	NO ENTRY
PSDFB	Flag Byte			0
PSDLSA	Last Seek Address			0
PSDLP	Path Last Used			0
PSDEIC	Total Error - Incident Count			0
PSDRET	Total Retry Count			0
PSDRTH	Retry Thresholds	DEVGRP	UNIT	SEE NOTE
PSDTS	Error Time Stamp			0
PSDSDR	SDR Buckets (64 half-bytes)			0

Note: Retry thresholds are as follows:

For 2301 Drum - X'0101010A0505000000000000'
For 2311 or 2314 Disk - X'01020A020A0A0A0A0A000000'
For 2302 - X'020202010A0A0A0A0A010000'
For other Devices - No entry in DAPSDR

SCAN MASTER CONTROL TABLE (CHBSMC)

SMCMLB	Master Lock Byte	0
SMCMCT	Master Count of GQEs	0
SMCDCT	Count of Digs	calculated by SYSGEN

SMCCMF	Master Count of Matched Facilities			0
SMEDLB	Dig Lock Byte			0
SMEBFG	Flag Byte			0
SMECMF	Count of Matched Facilities			0
SMEFEA	First Entry Address			SYSGEN adcon
SMECEA	Current Entry Address			0
SMELEA	Last Entry Address			SYSGEN adcon

SCAN TABLE (CHBSCN)

SCNFB1	Flag Byte Number 1			0
SCNDID	DIG code			calculated by SYSGEN
SCNLOK	Lock Byte			0
SCNPRO	Processor Pointer			SYSGEN VCON
SCNFQE	First Queue Entry			0
SCNLQE	Last Queue Entry			0

I/O STATISTICAL DATA TABLE (CHBSDT)

SDTLSD	Length of an SDR Entry (72 Bytes)			72
SDTLBA	Last Byte Address of SDT			SYSGEN ADCON
SDTSDA	Symbolic Device Address	DEVGRP	ADDRESS	0 ENTRY
SDTFB	Flag Bytes			0
SDTLP	Path Last Used (Actual I/O Address)			0
SDTEIC	Total Error Incident Count			0
SDTRET	Total Retry Count			0
SDTRTH	Retry Thresholds	DEVGRP	UNIT	SEE NOTE
SDTTS	Time Stamp at Error Incident N			0
SDTSDB	SDR Buckets (64 half-bytes)			0

Note: Retry Thresholds are as follows:

For 2400 Tapes - X'0F0503052805000000000000'
For 2311 or 2314 - X'0A0A0A0AFF020A0A000000'
For 2301 Drum - X'0100010A0505000000000000'
For 2540 Reader - X'0502050500000000000000000'
For 1050 - X'0303030303030303030A00'
For 2780 - X'0303120003030303000000'
For any other Device - X'000000000000000000000000'
For 2540 Punch - X'020202020000000000000000'
For 1403 Printer - X'050303030200000000000000'

CORE BLOCK TABLE HEADER (CHBCBH)

CBHUNA	Pointer to Unassigned Chain			0
CBHPNX	Pointer to Pending Non-XTSI-PSW Chain			0
CBHPXP	Pointer to Pending XTSI-PSW Chain			0
CBHAVC	Count of Available Blocks			0
CBHLOCK	Lock Byte for CBT			0
CBHBSE	Base Address for Start of Memory			0
CBHSZE	Number of Core Blocks in Memory	STEM	NUM	0

CORE BLOCK TABLE (CHBCBT)

CBTFLK	Entry - Forward Link			0
CBTTPT	TSI Pointer			0
CBTVMA	Virtual Memory Address			0

CBTFLG	Flags			0
CBTRLK	Reverse Link			0

PUBLIC VOLUME TABLE (CHBPVT)

PVTMCT	Maximum Volume Entry Count	PUBVOL	MAXVOL	10

SUPPORT SYSTEM DEVICE ALLOCATION TABLE (CHBECX)

ECXBFDE	PTR to 1st Non-resident Device Entry			SYSGEN VCON
ECXBLDE	PTR to Last Non-resident Device Entry			SYSGEN VCON
ECXBSDA	Symbolic Device Address	DEVGRP	ADDRESS	
ECXBPHP	Physical Path	DEVGRP	ADDRESS, PATH	
ECXBPHP2	Alternate Physical Path	DEVGRP	ADDRESS, PATH	SEE NOTE
ECXBDEV	Device Defining Information	DEVGRP		

Note: If no alternate physical path exists, the half-word field is set to X'FFFF'.

TERMINAL DEVICE TABLE (CHBTDE)

TDEFTD	First Terminal Device Pointer			SYSGEN ADCON
TDELTD	Last Terminal Device Pointer			SYSGEN ADCON
TDESDA	Symbolic Device Address	DEVGRP	ADDRESS	0
TDEDEV	Device Code	DEVGRP	UNIT, FEATURE	00000000

REMOTE JOB ENTRY TABLE (CHBRJE)

Note: Retry Threshold for 2780:
X'030312000303030303000000'

SYSTEM STATISTICS TABLE (CHBSST)

SSTLHT	SST total length - bytes	calculated by SYSGEN
SSTLH2	SST area 2 - bytes	calculated by SYSGEN
SSTLH3	SST area 3 - bytes	calculated by SYSGEN

BULK COMMON TABLE (CHBBCT)

BCTDEF	Default base time	3000
BCTSOI	Operator Intervention	50
BCTRUS	Real time measurement unit	1000
BCTEUS	Master service unit of time	100
BCTALM	ABEND limit	50
BCTBSN	Packed decimal batch sequence number	255C
BCTFL3	Interruption-driven flag (initially on)	
BCTALL	Length of S-entries	

This appendix is structured as follows:

- The system generation macro instructions are arranged in alphabetical order;
- The global symbols set by each macro instruction are listed; and
- For each global symbol there appears:

  the macro operand that sets the symbol (lower case entries indicate positional operand), the name of the System Control Block that is affected by the symbol, and the value assigned to the global symbol, if the macro operand is defaulted.

Whenever a hardware address is found in a global symbol, its decimal value plus one is used.

GLOBAL SYMBOL	DESCRIPTION	MACRO OPERAND	SYSTEM CONTROL BLOCK AFFECTED	DEFAULT
CCU	Number of Channel Control Units			
JCCUNO	Number of channel control units	numccu	CHBCCB	0
CHANNEL	Describe Multiplexer and Selector Channels			
CHL(I)={1|2}*	If the MPX operand is present, each address has its corresponding subscripted variable of CHL set to 1; if the SEL operand is present, each address has its corresponding subscripted variable of CHL set to 2.	MPX SEL	CHBCHL, CHBMCH, CHBSCH, CHBCCB, CHBDEV, CHBCCT	0
NOCHN	The sum of the number of addresses in the MPX and SEL operands.	MPX SEL	--	0
ONECH	Indicates the CHANNEL macro has been processed.	--	--	0
SELSOB	*I is the representation of a channel hardware address.	SELSUB	CHBMCH	(C,D,E,F)
CLOP	Specify Command Language Options			
JSCMTAP={A0|E0|00}	Tape drive type default	TATYPE={7|7DC|9}	CHBSCM	00
JSCMDA={1|8}	Direct access device type default	DATYPE={2311|2314}	CHBSCM	1
JSCMMWT	Maximum wait time for operator response	MOWT	CHBSCM	6

GLOBAL SYMBOL	DESCRIPTION	MACRO OPERAND	SYSTEM CONTROL BLOCK AFFECTED	DEFAULT
JSCMOCF={Y / N}	Confirmation or no confirmation of commands wanted	OPCONF{Y / N}	CHBSCM	Y
JSCMOMS={M / C}	Format of system messages to the operator	OPMSG{M / C}	CHBSCM	M
JSCMIPL	Installation's prompting limit	PRMTLMT	CHBSCM	5
JSCMPSP	$Integer_1$-Number of pages used for default for VAM primary storage allocation	DAPAGES	CHBSCM	16
JSCMSSP	$Integer_2$-Number of pages used for default for VAM secondary storage allocation	DAPAGES	CHBSCM	08
JSCMPSC	$Integer_1$ - Number of cylinders used for default for SAM primary storage allocation	DACYLS	CHBSCM	2
JSCMSSC	$Integer_2$-Number of cylinders used for default for SAM secondary storage allocation	DACYLS	CHBSCM	1
JSCMSST	Number of tracks used for default for SAM secondary storage allocation	DATRKS	CHBSCM	8
JSCMTDN={3 / 43 / 83}	Tape recording density default	DEN={0 / 1 / 2}	CHBSCM	43
JSCMORG={1 / 4 / 5 / 6}	Data set organization default	DSORG={PS / VI / VS / VP}	CHBSCM	5
JSCMLAB={01 / 02 / 04}	Tape label default	LABTYP={NL / SL / SUL}	CHBSCM	2
JSCMATH={U / P}	Default authorization assigned to a user when joined to the system	AUTH={U / P}	CHBSCM	U
JSCMPRV, JSCMLP0, JSCMLP1, JSCMLP2, JSCMLP3	Default privilege class assigned to a user when joined to the system, and other privilege classes	PRVLG	CHBSCM	D,120,0, 0,0
JSCMCFM	Installation default identification for card forms	CFM	CHBSCM	CARDS
JSCMPFM	Installation default identification for printer forms	PFM	CHBSCM	PAPER
JSCMUL1	$Integer_1$-Primary space allocation for a userlib	LIBPGS	CHBSCM	16
JSCMUL2	$Integer_2$-Secondary space allocation for a userlib	LIBPGS	CHBSCM	8

GLOBAL SYMBOL	DESCRIPTION	MACRO OPERAND	SYSTEM CONTROL BLOCK AFFECTED	DEFAULT
JSCMFIR={Y|N}	FORTRAN interruption recovery on boundary violations desired	FIR={Y|N}	CHBSCM	Y
JSCMMAV	Default maximum auxiliary storage	MAV	CHBSCM	256
JSCMIDP	Default external priority	PRIO	CHBSCM	8
CPU*	Describe a Central Processing Unit			
JCSTNOP(I)={1,2,3,4}	Each subscripted variable is assigned the identity of the CPU (except for a CPU1, MODEL 1).	cpuno={1,2,3,4}	CHBCST	
JCSTMDL(I)={1,2}	Each subscripted variable is assigned the model number of the CPU	MODEL={1,2}	CHBCST	2
JF1(I)=FLTADD	Each subscripted variable is set to FLTADD if floating storage addressing is specified for the CPU	FEATURE	--	null
JF2(I)=PTSNS	Each subscripted variable is set to PTSNS if partitioning sensing is specified for the CPU.	FEATURE	--	null
JF3(I)=XCNTRL	Each subscripted variable is set to XCNTRL if extended direct control is specified for the CPU.	FEATURE	--	null
JCSTPF1(I)	Primary prefix storage area value	PRFX	CHBCST	null
JCSTPF2(I)	Secondary prefix storage area value *In all CPU global symbols, I represents the identity of the CPU.	ALTPRFX	CHBCST	null
DCU	Describe a Device Control Unit			
HCU(I)	Each subscripted variable is assigned the highest hardware address (excluding channel portion) from among the device control units attached to the channel represented by I.	ADDRESS	CHBSCH	0
DCULAB(LABEL)	Each subscripted variable is set to the character value of the channel and device control unit of the first entry of the ADDRESS operand.	ADDRESS, LABEL	--	null
HLAB	The highest value specified in the LABEL operand	LABEL	--	0
CORLST(LABEL)	For each device control unit with a label less than 33, an entry in the list consists of a halfword containing a count of the paths to the control unit through one controller, followed by the actual paths (one per halfword); a nonexisting entry consists of a halfword of 'FFFF'.	LABEL	CHBCLT	null

GLOBAL SYMBOL	DESCRIPTION	MACRO OPERAND	SYSTEM CONTROL BLOCK AFFECTED	DEFAULT
*DCUN(M)	Effectively, this is an NxM matrix where N represents a channel and M a device control unit. For the first entry of the ADDRESS operand, the DCUN(M) entry is set as follows (N and M are the channel and control unit of the first entry): 1) If DCUN(M) is null, it is set to the concatenation of the unit code and the model of the device and a left parenthesis. 2) If DCUN(M) is not null and its first four characters are asterisks, the first five characters are replaced by the unit code and model number. For subsequent ADDRESS entries, the DCUN(M) entry is set as follows (N and M are the channel and control unit of the second entry): 1) If the channel of the subsequent entry is the same as the channel of the first entry, and if DCUN(M) is null, DCUN(M) is set to the channel and device control unit of the first entry concatenated to a zero and right parenthesis. However, if DCUN(M) is not null and the first four characters are asterisks, the first five characters are replaced by the unit code and model number. 2) If the channel of the subsequent entry is not the same as the channel of the first entry and, if DCUN(M) is null, DCUN(M) is set to the concatenated character value of the channel and control unit of the first entry, zero, and the number of entries in the ADDRESS operand. The DCU entry for the first ADDRESS entry has an asterisk set to replace the left parenthesis originally set. However, if DCUN(M) is not null and the first four characters are asterisks, the first five characters are replaced by the unit code and model number.	UNIT, ADDRESS, MODEL={1, 2, 3, 5, 7}	CHBCUT, CHBSCH, CHBDEV, CHBDPT	null

GLOBAL SYMBOL	DESCRIPTION	MACRO OPERAND	SYSTEM CONTROL BLOCK AFFECTED	DEFAULT
N270X	If UNIT = 2701, or 2702, or 2703 and if the channel specified in a subsequent entry is different from the channel specified in the first ADDRESS entry, N270X is incremented by one so as to point to the next available subscripted variable in T270X.	UNIT, ADDRESS	CHBTPT	0
T270X(N270X)	Each subscripted variable is assigned the channel and control unit portion from a 2701, 2702 or 2703 ADDRESS entry that has a channel different from the channel of the first entry of the ADDRESS operand. *This global symbol is also set by the DEVGRP macro instruction.	UNIT, ADDRESS	CHBTPT	null
DEVGRP	Describe a Device Group			
PNO(NM)	Effectively, this is an NxM matrix where N is the channel and M is the device control unit. An element of the matrix provides the number of paths specified in the PATH operand for the particular channel and device control unit.	PATH	CHBDEV	0
MD(NM)	Effectively, this is an NxM matrix where N is the channel and M is the device control unit. Each element of the matrix is set to the maximum number of devices specified for a particular channel and control unit.	ADDRESS, PATH	CHBDEV	0
DEVNM(D)	Effectively, this is a NxMxD matrix where NMD represents the hardware address of a device. Each element of the matrix is set to the unit and type of device whose hardware address is NMD.	ADDRESS, PATH, UNIT, TYPE	CHBDEV, CHB272, CHBPSD, CHBSDT	null
DEVSNM(D)	Effectively, this is a NxMxD matrix where NMD represents the hardware address of a device. Each element of the matrix is set to the symbolic address that corresponds to the hardware address NMD.	ADDRESS, PATH	CHBDEV, CHB272	0
SYM(S)	Each subscripted variable is assigned the hardware device address and type and IOREQ bit corresponding to the symbolic device address represented by S.	ADDRESS, PATH, TYPE, IOREQ={YES / NO}	CHBSDA, CHBDPT, CHBPSD, CHBSCN, CHBSDT	null
SYMA(S)	Each subscripted variable is assigned the MAXIO value for the device whose symbolic address is represented by S.	ADDRESS, MAXIO	CHBSDA	null

GLOBAL SYMBOL	DESCRIPTION	MACRO OPERAND	SYSTEM CONTROL BLOCK AFFECTED	DEFAULT
SYMF(S)	Each subscripted variable represents the features on a device whose symbolic address is represented by S.	ADDRESS, FEATURE	CHBSDA, CHB272	null
HYSM	The highest symbolic address specified	ADDRESS	CHBSDT, CHBSDA, CHBPSD, CHBSCN	0
CTU	Effectively, CTU is a table with an entry for each unit represented by U. An entry is incremented by one each time a device for this particular unit is processed.	UNIT, ADDRESS	CHBSCM, CHBHED, CHBAHD, CHBAVE, CHBCCB	0
SU(CTU)	Effectively, this is a UxCTU matrix where U represents a unit and CTU represents the n th (n=1,...) occurrence of a device with this unit number. Each element of the matrix is set to the symbolic address of the device.	ADDRESS, UNIT	CHBAVE	0
APRNT	A pointer to the next available entry in the CPRNT table.	UNIT, PATH, ADDRESS	--	0
CPRNT(APRNT)	Each time a device in the group of 1403 printers is processed, a new block of symbolic variables in CPRNT is started. The first entry in a block is the number of paths specified in the PATH operand. If n is the number of paths, then the next n entries represent all the hardware addresses of a specific printer.	UNIT, PATH, ADDRESS	CHBPRN	null
GAPRNT	A pointer to the next available block in CPRNT.	UNIT, PATH	CHBCCB	0
JSCMTA1	Number of 7-track tape drives	FEATURE, UNIT	CHBSCM	0
JSCMTA2	Number of 7-track tape drives with the data conversion feature	FEATURE, UNIT	CHBSCM	0
JSCMTA3	Number of 9-track tape drives	FEATURE, UNIT	CHBSCM	2
PGDRUM	Number of paging drums	TYPE, UNIT	CHBSYS	0

GLOBAL SYMBOL	DESCRIPTION	MACRO OPERAND	SYSTEM CONTROL BLOCK AFFECTED	DEFAULT
ECXMMN(S)	Effectively, this is a NxS matrix, where N represents the path number and S represents the symbolic device address. The MM portion of the matrix is the index for the SDA value; i.e., ECX01N(S) is the matrix for SDA 1-255, ECX02N(S) is the matrix for SDA 256-510, etc. The value of the matrix is set to the actual hardware address of the device.	ADDRESS, PATH	CHBECXRA, CHBECXRB, CHBECXVA	
*DCUN(M)	Effectively, this is a NxM matrix where N represents a channel and M a device control unit. If there is more than one entry in the PATH operand and DCUN(M) is null (N and M represent the channel and control unit of the first PATH entry) DCUN(M) is set to ****0* where the last asterisk denotes more than one path to a device control unit and the first four asterisks are used for the device control unit type until it is defined in the DCU macro. If DCUN(M) is not null, an asterisk is placed after the five characters representing the unit and model. For subsequent PATH entries, the appropriate DCU entry is set to the character value of the channel and device control unit of the first PATH entry followed by a number indicating the position of the entry in the PATH operand. *This global symbol is also set by the DCU macro instruction.	PATH	CHBCUT, CHBSCH, CHBDEV, CHBDPT	null
HISYMX(Y)	Effectively, this is a XxY matrix where X represents the channel and Y a device control unit. The value is set equal to the high SDA on the control unit.	ADDRESS	CHBSCN	null
LOSYMX(Y)	Effectively, this is a XxY matrix where X represents the channel and Y a device control unit. The value is set equal to the low SDA on the control unit.	ADDRESS	CHBSCN	null
DISPAR	Specify the Dispatching Algorithm			
JSYSLOW	$Integer_1$-Lower limit on available storage before starting a time-slice for a new task	LCT	CHBSYS	1

GLOBAL SYMBOL	DESCRIPTION	MACRO OPERAND	SYSTEM CONTROL BLOCK AFFECTED	DEFAULT
JSYSHI	$Integer_2$-Upper limit on available storage before starting a time-slice for a new task	LCT	CHBSYS	10
JCCBLDA	Percentage of paging drum(s) to be allocated to auxiliary storage before overflow to paging disk(s)	LDMTR	CHBCCB	2%
JSYSMWX	Maximum Shared Pages To Purge	PURGSH	CHBSYS	22
JSYSMWT	Scan Shared Pages Threshold	THRESH	CHBSYS	10
JSYSBUF	Buffer Size On Drum	BUFSIZ	CHBSYS	64
JSYSAST	Auxiliary Stop Threshold	AUXSP(1)	CHBSYS	JSYSBUF + 10_{10}
JSYSAPT	Auxiliary Primary Threshold	AUXSP(2)	CHBSYS	100
JSYSBKD	Maximum pages blocked to drum	BLKSZE(1)	CHBSYS	12
JSYSBDD	Combined maximum pages blocked to disk and drum	BLKSZE(2)	CHBSYS	20
JSYSTCR	Task Core Requirement	TCR	CHBSYS	55
OPCNSL JOPCNSL(I)	Describe Operator Consoles Each subscripted variable is assigned the concatenation of the hardware and symbolic device addresses specified for an operator console.	ADDRESS	CHBSOT	null
PUBVOL PUBNUM	Describe Public Volume Configuration Maximum number of public volumes in system.	MAXVOL	CHBPVT	10
STEM	Describe Processor Storage Units			
PSUNO= {1 2 3 4}	Number of processor storage units	number= {1 2 3 4}	CHBCBT, CHBCBH, CHBCST	null
PSUMDL= {2 12}	Model number of the processor storage units	MODEL= {2 12}	--	null

GLOBAL SYMBOL	DESCRIPTION	MACRO OPERAND	SYSTEM CONTROL BLOCK AFFECTED	DEFAULT
TSKLMT	Specify Task Limitations			
JCCBCON	Maximum number of conversational tasks that may exist concurrently	CONV	CHBCCB	
JCCBMTT	Maximum number of MTT administrator tasks that may exist concurrently	MTTADM	CHBCCB	
JCCBNCV	Maximum number of batch tasks that may exist concurrently	BATCH	CHBCCB	
JSYSTLM	Maximum number of conversational and batch tasks that can exist concurrently		CHBSYS	
JCCBBAK	Maximum number of background tasks that may exist concurrently	BACK	CHBCCB	
JPAL1	Maximum size of page tables for one task	PGTBL	CHBSYS	16
JSYSDAT	Number of entries to expand page table	VLPTE	CHBSYS	3
JSYSPSL	Maximum number of public segments allowed	PSLMT	CHBSYS	6
JSCMAUX	Present Aux storage space available	OVRAUX	CHBSCM	.00
JSYSXPG	Maximum number of pages that will trigger XTSI paging	XTSIPGS	CHBSYS	0
VMPAR	Specify Virtual Storage Parameters			
JSYSPS= PUBSEG	Public segment has sharable virtual storage routines	OPTIONS	CHBSYS	null
JSYSPC= PACKSEG	Private modules page packed into segments	OPTIONS	CHBSYS	null
JSYSWA= WCDRUM	Validity check paging drum writes	OPTIONS	CHBSYS	null
JSYSWE= WCDISK	Validity check auxiliary paging disk write	OPTIONS	CHBSYS	null
JPAL4= SEGMENT	Variable-length control sections are assigned a segment	VCSLNG	CHBCCB	SEGMENT
JPAL5= integer value	Variable-length control sections are assigned a segment (256 pages) or a specific number of pages	VCSLNG	CHBCCB	20
CCBSDST	Number of pages assigned to the shared data set table	SDST	CHBCCB	10
JSCMTIM	Task time interval	TIME	CHBSCM	01000000
JCCBBUF	Number of buffers for input data	TBUFS	CHBCCB	200

APPENDIX C: DATA REFERENCES BY SYSTEM GENERATION MODULES

This appendix presents a list of all DSECTs that are referred to by the System Generation and Maintenance modules. The modules are listed by ID and the DSECTs by their symbolic names. Refer to the listings for further information.

DSECT ID	Modules				DSECT NAME
	CEIAA (STARTUP)	CEIAP (PRELUDE)	CEIFA (SYSBLD)	CEIDA (SYSGEN)	
CHAAHD			X	X	Available Devices Table Subqueue Header
CHAASA	X				Auxiliary Storage Allocation
CHAASB	X				ASAT Bit Directory 2311
CHAASC	X				ASAT Bit Directory 2314
CHAAST	X				Auxiliary Segment Table
CHAAVE			X	X	Available Devices Table Subqueue Entries
CHACBH	X	X		X	Core Block Table Header
CHACBT	X	X		X	Core Block Table
CHACCB	X	X		X	Configuration Control Block
CHACCC			X		Catalog SBLOCK
CHACHL			X	X	Channel Table
CHACST	X	X			CPU Status Table
CHACUT				X	Control Unit Table
CHADAV			X		DSCB Format C
CHADEV			X	X	Device Group Table
CHADSE	X	X	X		DSCB Format E
CHADSF	X	X	X		DSCB Format F
CHAECX	X		X	X	Support System Device Allocation Table
CHAGQE	X				General Queue Entry
CHAHED			X	X	Available Device Table Header
CHAISA	X				Interruption Storage Area
CHAMAP	X				Memory Map Table
CHAMCH			X		Multiplexer Channel Table
CHAPGH	X				PMD Group Header
CHAPGT	X				Page Table
CHAPOD	X		X		Partitioned Organization Directory

DSECT ID	Modules				DSECT NAME
	CEIAA (STARTUP)	CEIAP (PRELUDE)	CEIFA (SYSBLD)	CEIDA (SYSGEN)	
CHAPOE	X		X		Directory Alias Descriptor
CHAPOM	X		X		Directory Member Descriptor
CHAPSA	X	X			Prefixed Storage Area
CHAPSD				X	Direct Access Paging Statistical Data Record
CHAPVT	X			X	Public Volume Table
CHARJE				X	Remote Job Entry Table
CHARSP	X				Resident Shared-Page Index
CHASAC			X	X	Symbolic to Actual Address Conversion Table
*CHASBD			X		SYSBLD Table
CHASCH			X		Selector Channel Table
CHASCM			X	X	System Common Table
CHASCN				X	Scan Table
CHASDA	X		X	X	Symbolic Device Allocation Table
CHASDS	X				Shared Data Set Table
CHASDT				X	I/O Statistical Data Table
CHASGT	X				Segment Table
CHASID				X	CHASOT Entry
CHASMC				X	Scan Master Control Table
CHASOT				X	System Operator ID Table
CHASST				X	System Statistics Table
CHASYS	X			X	System Table
CHATDE			X	X	Terminal Device Table
CHATDH	X				TDY Heading
CHATDT	X				Task Data Definition Table
CHATDY	X		X		Task Dictionary Table
CHATSI	X				Task Status Index
CHAUSE			X		User Table
CHAVTC	X		X		DSCB Format 4
CHAXPT	X				External Page Table
CHAXSP	X				Shared External Page Table
CHAXTS	X				Extended Task Status Index

*Internal to SYSBLD only

IBM TSS S/360 Sysgen Printed in U.S.A. GY28-2015-6

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

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

IBM Technical Newsletter

File Number S360-31

Base Publication No. GY28-2015-6

This Newsletter No. GN28-3218

Date February 1, 1972

Previous Newsletters None

IBM System/360 Time Sharing System:
System Generation and Maintenance

This Technical Newsletter provides replacement pages for the subject publication. Pages to be inserted and/or removed are:

17-18

A change to the text is indicated by a vertical line to the left of the change.

Summary of Amendments

A modification has been made to CEIAP's error exit description.

IBM Corporation, Dept. 643, Neighborhood Road, Kingston, N. Y. 12401

PRINTED IN U S A