SY33-8553-4
File No. S370/S4300-36

Systems

DOS/VSE Logical Transients and Dump Phases Logic

Program Numbers 5745-SC-CKR
5745-SC-AIT
5745-SC-DOC

IBM

Summary of Amendments

Edition SY33-8553-4 documents:

- New processor support
- New device support
  - 3278 Model 2A Display Console
  - 8809 Magnetic Tape Unit
  - PRT1 printers (3289 Model 4, and 3203-5)
  - 3284/3286/3287 console printer
- Dumps in SVA and extended storage dumps
- Deletion of obsolete supervisor options
- DASD volume recognition by means of new VOLUME command
- Mode setting for the 8809 Magnetic Tape Unit by means of new SETMOD command

The logic of the Checkpoint/Restart routines is now included in this publication. In the library of DOS/VS, the logic of these routines is documented in the LIOCS Volume 1 Logic manual.

Fifth Edition (February, 1979)

This is a major revision of, and obsoletes, SY33-8553-3 and Technical Newsletter SN33-9236. This edition applies to the IBM Disk Operating System/Virtual Storage Extended, DOS/VSE, and to all subsequent releases until otherwise indicated.

Changes and additions to the text or illustrations are indicated by a vertical line to the left of the change. Changes are continually made to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest IBM System/370 Bibliography, GC20-0001, for the editions that are applicable and current.

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

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Laboratory, Publications Department, Schoenaicher Strasse 220, D-7030 Boeblingen, Germany. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

PREFACE

This Program Logic Manual (PLM) is a detailed guide to the IBM DOS/VSE logical transient programs. It supplements the program listings by providing descriptive text and flowcharts.

The lists that follow give the titles of companion system control PLMs and prerequisite publications.

For overall system control logic description, this PLM is to be used with the following PLMs:

- DOS/VSE Supervisor Logic, SY33-8551.
- DOS/VSE Error Recovery and Recording Transients Logic, SY33-8552.
- DOS/VSE Serviceability Aids Logic, SY33-8554.
- DOS/VSE Initial Program Load and Job Control Logic, SY33-8555.
- DOS/VSE Linkage Editor Logic, SY33-8556.
- DOS/VSE Librarian Logic, SY33-8557.

Prerequisite to the effective use of the PLMs are the following publications:

- IBM System/370 Principles of Operation, GA22-7000.
- Introduction to DOS/VSE, GC33-5370.
- DOS/VSE System Management Guide, GC33-5371.
- DOS/VSE Data Management Concepts GC24-5138.
- Guide to the DOS/VSE Assembler, GC33-4024.

Titles and abstracts of other related publications are listed in the IBM System/370 Bibliography, GC20-0001.

In this publication, system and component names as listed below should be read as indicated:

System/component name	To be read as
DOS/VS	DOS/VSE (see Note below)
ECPS:DOS/VS	ECPS:VSE
DOS/VS POWER/VS-E	VSE/POWER
DOS/VS VSAM-E	VSE/VSAM

Note: Unless that name explicitly refers to DOS/VS Release 34 or an earlier DOS/VS release.

FIGURES

CHARTS

General Charts

Detailed Charts

This manual describes (a) the logical transient programs and (b) the dump phases that are executed in the SVA.

LOGICAL TRANSIENTS

Logical transient programs, also referred to as B-transients, are not resident in real storage. These transients are loaded or fetched from the core image library into a transient area in the supervisor. The majority of the B-transients are loaded into the Logical Transient Area (LTA), when required. The CRT transients, $$BOCRTn, are an exception. They are required for display operator console support and are loaded into the CRT Transient Area (CRTTRNS) when required by the CRT system task.

NON-CRT TRANSIENTS

An SVC 2 instruction loads a B-transient phase into the LTA (1792 bytes) and executes it. A prefix of $$B to the name of a phase identifies it as a B-transient. The normal return of control to the resident supervisor is an SVC 11, but some of the transient programs exit by fetching another B-transient with an SVC 2. In the latter case, the calling B-transient is overlaid by the transient being fetched.

Register 1 is loaded with the address of the transient phase name before the SVC 2 is issued. The fetch or load routine, then, has access to the name for searching the directories for the desired transient. Refer to section 'Logical Transient Supervisor Calls' for a more detailed explanation.

CRT TRANSIENTS

CRT transient routines provide support for the display operator consoles. They reside in the core image library and are fetched into the CRT Transient Area by an SVC 48. This happens each time the supervisor encounters a request for a display console. Since these requests may be for different services, the support is provided by 30 different transient phases. Each CRT phase name begins with the characters $$BOCRT. Each phase fits in the CRT Transient Area which is 1156 bytes long.

GROUPING OF B-TRANSIENTS

A number of B-transients are not described in this manual, but in the manuals for the system components to which they are related, such as LIOCS and Job Control. The B-transient programs, which are covered in this manual, can be grouped by the various functions performed:

- Attention routines
- Terminator routines (see also Dump Phases, below)
- Various special service programs
- Display operator console support
- Checkpoint/Restart routines

DUMP PHASES

The terminator routines consist of two groups: B-transients, which are executed in the LTA, and the dump phases ($IJBDUMP), which are executed in the SVA. The LTA is not used or locked during execution of a system dump or a PDUMP macro; it is locked, however, during execution of the DUMP command. When abnormal termination occurs in the task owning the LTA, the area is not freed until the system dump is complete. The system dump routines will then dump the LTA containing the error or that may have provoked the error in the failing task.

SVC is detected by microprogramming, which loads the SVC new PSW from real storage location 96. Certain SVCs are involved in processing B-transient operations. This section describes those supervisor calls which are directly used in logical transients. The Supervisor Logic PLM listed in the Preface contains a list of all SVCs used in the system.

SVC 0: Execute the channel program (EXCP). The address of the user's command control block (CCB/IORB) must be supplied in general register 1 before this SVC is issued. If POWER/VS is active and the I/O operation for the device needs to be emulated, control is given to the SVC 0 appendage in the POWER/VS nucleus code.

SVC 2: Fetches a B-transient. Loads a B-transient program (phase name prefix equals $$B) from the system core image library or a private core image library to the B-transient area and control is passed to the B-transient at its load address plus 8 bytes. The directory entry for the phase may be found in storage, in the system core image library directory, or in the private core image library directory (if a PCIL is assigned).

The storage address of the B-transient phase name, or the address of a parameter list must be supplied in general register 1 before this SVC is issued.

An address in general register 0 is ignored. The B-transient is loaded at the beginning address of the B-transient area. General register 15 is loaded with this address and may be used by B-transients as a base register. Return may be either to the interrupted program or to the highest priority program ready to run.

Only one program can use the B-transient area at a time. If the B-transient program is SVC 7 bound, another program is selected. This program becomes SVC 2 bound (waiting for the B-transient area) if it issues an SVC 2. Another program is then selected.

Note: SVC 2 also traps fetches for the former dump routines $$BDUMP, $$BJDUMP, and $$BPDUMP. Whenever an SVC 2 is issued for one of these routines, the terminator in the supervisor is entered and the SVA-resident dump routine $IJBDUMP is initialized.

SVC 3: Provides an interface between the supervisor and $$BEOJ4. SVC 3 waits for termination of I/O requests that belong to the partition or task which is being canceled or has reached end-of-job.

SVC 4: Loads a phase from the system core image library or a private core image library (PCIL) and returns to task selection. The directory entry for the phase may be found in storage, in the system core image library directory, or in the private core image library directory (if a PCIL is supported and assigned). A directory entry found in the SDL is not loaded if it is active and its phase itself is SVA-resident.

The storage address of the phase name or the address of a parameter list must be supplied in general register 1 before this SVC is issued. The user may override the link-edited load address by supplying a load address in general register 0. Upon return to the user, general register 1 contains the phase entry address adjusted for any changes in the phase's load address, and general register 0 points to the active incore directory entry if one was supplied, or found by a local or system directory list search.

SVC 6: Cancels a program (task) or partition. This is usually achieved by the requesting program, task, or subtask issuing a CANCEL or CANCEL ALL macro.

If a subtask issues CANCEL, only that subtask is terminated. If a maintask issues CANCEL, or a subtask issues CANCEL ALL, then the entire partition is canceled, the maintask being always the last to be terminated.

CANCEL macro issued by maintask without subtasks: the issuing task is terminated normally.

- Cancel code X'23' is posted to the issuer's PIB.
- Message '(issuer) CANCELED DUE TO PROGRAM REQUEST'.

CANCEL macro issued by subtask: the issuing subtask is terminated normally.

- Cancel code X'23' is posted to the issuer's PIB.
- Message '(issuer) CANCELED DUE TO PROGRAM REQUEST'.

CANCEL macro issued by maintask with subtasks attached: the maintask is terminated normally; attached subtasks are terminated abnormally.

- Cancel code X'1D' is posted to each subtask PIB.
- Cancel code X'17' is posted to the maintask PIB.
- Message '(subtask) CANCELED DUE TO MAINTASK TERMINATION'.
- Message '(issuer) CANCELED DUE TO PROGRAM REQUEST'.
- A dump is generated at the start of the termination of the maintask if the DUMP option is active (DUMP=YES).

CANCEL ALL macro issued by a subtask: the issuing subtask is terminated normally; other subtasks and the maintask are terminated abnormally.

- Cancel code X'23' is posted to the issuing subtask PIB.
- Cancel code X'1C' is posted to each of the other subtasks PIBs and to the maintask PIB.
- Message '(issuer) CANCELED DUE TO PROGRAM REQUEST'.
- Message '(main or subtask) CANCELED DUE TO CANCEL ALL MACRO'.
- A dump is generated at termination of the subtask if the DUMP option is active.

If linkages to the user's AB routines have been established through the STXIT (AB) macro, these routines are entered for all tasks that are terminated abnormally by the CANCEL or CANCEL ALL macro, so the task that issues an SVC 6 never enters its AB routine. (Exception: the issuer was a B-transient.)

An AB routine normally terminates through a DETACH, EOJ, or CANCEL macro, but an abnormal condition encountered in an AB routine also terminates that AB routine.

SVC 7: Waits for the completion of an I/O operation or for a timer interruption to occur. It supplies the supervisor support for the WAIT macro.

If the traffic bit (CCB/IORB) or event bit (TECB) has been posted, SVC 7 branches directly to task selection and task is dispatched immediately after the SVC7 call.

If the traffic bit or event bit has not been posted, the following action is taken:

- If a system task was interrupted, the system task is deactivated.
- The PIB flag of the interrupted program is set to I/O-bound (not ready to run) and its PSW is set up to reissue SVC 7.

SVC 8: Supplies the supervisory support to temporarily return from a logical transient to the problem program. This SVC may be issued only from the logical transient area (LTA) and does not free this area. The entry address to the problem program must be specified in general register 14. The task selection exit loads the problem program registers. General registers 0 and 1 are passed to the problem program.

To return to the logical transient, the problem program issues an SVC 9.

SVC 9: Supplies the supervisory support to return to the logical transient after an SVC 8 has been issued. An SVC 9 may be issued only by the problem program. The task selection exit loads the logical transient registers. General registers 0 and 1 are passed to the logical transient programs.

SVC 11: Returns from a B-transient releasing the B-transient area. SVC 11 is invalid if issued by other than a B-transient. The logical transient area is released for use by other programs or tasks. Return is to the highest priority program ready to run.

SVC 11 is also used to return from the SVA-resident terminator routine ($IJBDUMP) to the supervisor. The terminator routine in the supervisor will release the SVA-resident terminator routine.

SVC 14: This is the normal end of job (EOJ). Cancel code X'10' is posted to the PIB for the program issuing the SVC 14. The next time the canceled program is selected on general exit, a branch is made to the terminator routines.

SVC 22: Seizes the system and provides a release from such a seizure in a multiprogramming system. This SVC may be issued only by job control and logical transient programs. The PSW protection field must be zero, otherwise the issuing program is canceled.

The first SVC 22 issued seizes the system. Until the next SVC 22 is issued, the task selection mechanism is disabled so that the issuing task is the only user task that can be selected. The next SVC re-enables the task selection mechanism.

If the low-order byte of general register 0 is zero, the system mask is set to disable all interruptions; if non-zero, the system mask is set to enable all interruptions.

If general register 0 is negative, the user protection key is set in the user's PSW.

SVC 23: Retrieves the load address for a specified phase from the directory entry for the phase. The program issuing an SVC 23 is canceled if the PSW protection key does not equal 0. (Only job control and B-transient programs can issue an SVC 23.)

The user must specify the storage address of the phase name in general register 1 and the address where the load address is to be stored in general register 0. The main fetch subroutine scans the System Directory List, the system core image library and the private core image library (if supported and assigned) for a directory entry for the phase. The load address is retrieved and stored at the address specified by general register 0. The high-order byte of the storage area is not changed.

If the phase is relocatable the load address returned is the relocated load address.

SVC 26: Validate address limits. The program issuing an SVC 26 is canceled if the PSW protection key does not equal 0. (Only job control and B-transient programs can issue an SVC 26.)

The upper address must be specified in general register 2, and the lower address must be specified in general register 1.

If an SVC 26 was issued by a CRT-transient routine, control is always returned to the CRT routine. In case of an error, a flag byte is set before control is returned. For any other routine, if either address is outside the requestor's partition, the task is canceled (ERR 25).

SVC 36: Frees a track or block that is held by the task issuing the FREE. An attempt to free a track not owned by the requestor results in cancelation of that task.

Exits on a successful FREE are to task selection, or to the DETACH routine if the FREE was issued by that routine.

SVC 39: Performs normal termination of a subtask. DETACH may be issued by either the subtask being terminated or by the main task. If DETACH is issued by a problem program, the cancel code X'10' (normal end-of-job) is set in the subtask PIB and the terminator is entered. At the end of the termination process, DETACH is issued by an EOJ transient routine, setting the subtask's PIB inactive (byte 0 = X'80'), and posting its ECB for termination.

SVC 48: Fetches a CRT-transient phase and makes the CRT task dispatchable. The program issuing an SVC 48 is canceled if the PSW protection key does not equal zero.

The first SVC 48 is issued at IPL time to activate the CRT hooks in the supervisor and to load the CRT root phase $$BOCRTA into the CRT transient area (CRTTRNS). Each subsequent SVC 48 loads the specified phase via Fetch into the CRT transient area and sets the PSW address to the start address of the CRT phase; exit is taken to the dispatcher.

SVC 54: This supervisor call is only valid in 370-mode. In ECPS:VSE-mode the same function is provided by PFREE. SVC 54 provides supervisor support for the FREEREAL macro to release page frames to the page pool. These page frames may be released from a real partition or the SDAID area.

When the request is issued by the terminator (the page frames belonging to a real partition are freed), the lower and upper addresses (lower address being any address in the first page of the area, upper being any address in the last page of the area) of the real partition are passed to the SVC 54 routine in the registers 2 and 3.

A zero value in register 2 indicates that the request is issued by SDAID. In this case the lower and upper limit of the area to be released are obtained from the internal page manager address fields. Control is passed immediately to task selection if no SDAID area exists.

The page frames are freed, one after the other, by updating the corresponding Page Frame Table entries (PFTE). The PFIX counter is reset and the partition PFIX counter in the Storage Management Control Block (SMCB) is decreased by one.

In the PFTE the page frame is indicated as unused. The address bits of the PFTE are cleared, the storage key of the page frame is stored in bits 8-11, the page data set bit is reset, and the high address bit is set. The released page frames are enqueued on top of the unused page frames selection queue. The counter for the number of page frames in the selection queue is increased by one.

If SVC 54 is issued by the terminator, the number of active virtual partitions (entry in SYSCOM) is increased by one, and the translation mode bit in PIBDATFL is set.

If SVC 54 is issued by SDAID, the counter in the SMCB (which shows the number of page frames that are fixed in the page pool) is reset to zero.

The SVC 54 posts any task and the CCW translation routine ready to run, if they are in the wait state, and if more than the minimum number of page frames is available in the page selection queue (PSQ).

SVC 59: In 370-mode, SVC 59 initializes the page table and page frame table entries belonging to specific pages. In ECPS:VSE-mode, SVC 59 invalidates the pages of the area specified by the input parameters. The task issuing an SVC 59 is canceled if the storage protection key in the PSW is not 0.

The following parameters are passed to this routine:

R3: Address located in the first page of the area to be invalidated

R4: Address located in the last page of the area to be invalidated.

370-mode only: If the area to be invalidated belongs to the active part of the partition (virtual partition: the TRAM bit is set; real partiton: the TRAM bit is reset), the corresponding page table entries (PTE) are set to X'00STK4', where STK equals the PIK and is stored in bits 8-11. If the area to be invalidated belongs to the non-active part of the partition, the corresponding PTEs are set to X'80STK0', where STK has the same meaning as above. Each PTE within the area defined by R3 and R4 is initialized in that way. If the page referred to by an entry is in real storage, the page frame table entry of the corresponding page frame is initialized as follows:

- The page frame is marked as unused (bit 4 in S370FLG is set) and the PFIX counter is set to zero.
- The page frame is removed from the page selection queue and enqueued at the top of the invalid page frame queue.
- If a page is found to be PFIXed, the system enters the hard wait state (debug mode only).

The key of the storage block in this page frame is set to zero.

ECPS:VSE-mode only: Each page within the area defined by R3 and R4 is invalidated as follows:

- If the page is disconnected, the reference, change, and PDS bit is reset.
- If the page is connected, the same action is taken as for disconnected; in addition, the hold bit of the connected page frame is reset.
- If the page is addressable, the corresponding page frame table entry is removed from the page selection queue, and the page is disconnected by resetting the reference, change, and PDS bit.

ATTENTION ROUTINES

This group of B-transients consists of phases whose names start with $$BATT.

The attention task is activated when the operator presses the request key on the console printer-keyboard or on the display console once or twice in quick succession, or when he presses the interrupt key on the system control panel. The physical attention transient $$ABERRZ is fetched into the Physical Transient Area.

On a display console, the operator may enter attention commands without pressing the request key first (see below).

If the operator pressed the request key twice in quick succession because he wants an emergency cancel, $$ABERRZ processes the interrupt and the logical attention transients are not activated.

In the other cases, $$BATTNA is fetched into the LTA. If $$BATTNA is entered because the operator pressed the interrupt key, $$BATTNB establishes linkage to the Background Operator Communication routine. Control is returned to the supervisor, unless attention is pending.

To service the attention request, $$BATTNA reads input statement information and loads $$BATTNH, which selects the appropriate command processor.

On a display console, the operator may enter attention commands without pressing the request key first. $$BATTNA moves the command from the attention buffer in the CRT save area CRTSAV to the I/O buffer in $$BATTNA. It then loads $$BATTNH which selects the correct command processor.

$$BATTNA is always loaded together with the code of $$BATTNB, which is contained in a separate control section. That code is overlaid by other attention phases as the need for execution of those phases arises. The code of $$BATTNA, which is contained in the first control section, normally remains in the Logical Transient Area as a root phase as long as the attention task is active. However, if $$BATTNA is overlaid by another attention phase, its code is always reloaded when the current phase completes execution. Figure 1 shows a layout of the Logical Transient Area containing the attention routines.

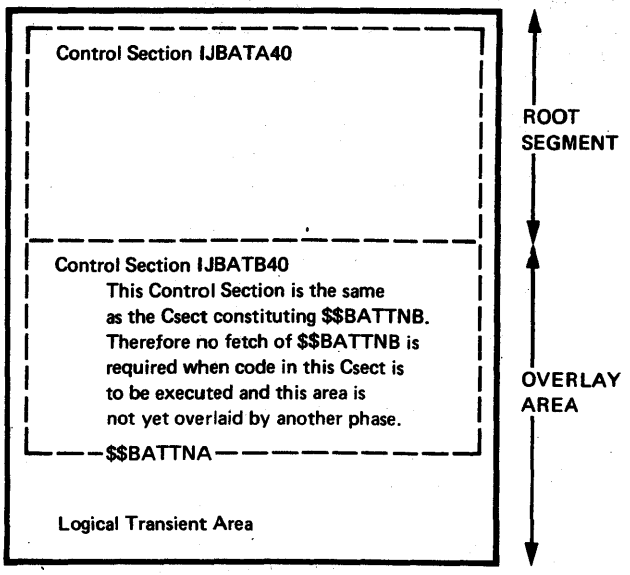


Figure 1. Layout of the Logical Transient Area containing the Attention Routines

Commands accepted by the attention routines are:

- ALLOC: Permits the operator to allocate the <u>virtual</u> address area among foreground and background programs ($$BATTNE).
- ALLOCR: Permits the operator to allocate the <u>real</u> address area among foreground and background programs ($$BATTNE).
- ALTER: Permits the operator to alter one to sixteen bytes of virtual storage ($$BATTNT).
- BATCH: Initiates job processing in a foreground partition, or restarts job processing in a stopped partition ($$BATTNG).
- CANCEL: Modifies the PIB of the specified partition to cancel the current task(s) in the specified partition ($$BATTNC).
- DSPLY: Permits the operator to display sixteen bytes of virtual storage ($$BATTNU).
- DUMP: Permits the operator to dump part (for example a partition) or all of real or virtual storage depending on whether the partition runs in real or virtual mode ($$BATTNV).

- ENDSD: Terminates SDAID routines ($$BATTN3).
- IGNORE: Used to ignore control statements that contain errors. Causes control to be returned immediately to $$BATTNA to read next command ($$BATTNC).
- LFCB: Permits the operator to change the contents of the FCB (forms control buffer) of a printer ($$BATTN8, F1, F4, F5).
- LOG: Permits logging of job control statements for all partitions on SYSLOG. The appropriate switch is set in each partition communications region ($$BATTNC).
- LUCB: Permits the operator to change the contents of the UCB (universal character set buffer) of a printer ($$BATTN9, U1, U2).
- MAP: Provides a map of storage utilization ($$BATTND, $$BATTNI). An example is given in DOS/VSE System Control Statements.
- MODE: Provides control over the recording mode for recoverable machine checks and allows setting various RMSR options for recording of device errors ($$BATTNQ, R, S, Y, Z).
- MSG: Causes control to be given to a foreground program operator communications routine previously activated by a STXIT macro ($$BATTNB).
- NEWVOL: Reactivates job control in the specified partition, which was waiting for a volume to be mounted ($$BATTNC).
- NOLOG: Suppresses logging of job control statements on SYSLOG. The appropriate switch is reset in each partition communications region ($$BATTNC).
- PAUSE: Indicates job control pauses for operator communication at the end of the current job step in the specified partition, or, optionally, at end of job of the current program in the specified partition ($$BATTNC).
- PRTY: Allows the operator to display or change the priorities of the partitions in the system, which were set when the system was generated ($$BATTN2).
- SETDF: Allows the operator to set and/or reset default values for the 3800 printer ($$BATTN7, $$BATTS1, $$BATTS2).
- SETMOD: Allows the operator to change the mode of a 8809 tape unit from streaming to start-stop or vice versa ($$BATTNK).
- SIZE: Allows the operator to subdivide a partition permanently into an area for program execution and a GETVIS area ($$BATTNF).
- START: This command has the same function as the BATCH command ($$BATTNG).
- TPBAL: Allows the operator to display or change TP Balancing ($$BATTN2).
- VOLUME: Provides the operator with the volume identifiers of the volumes mounted on a DASD device, together with an indication of whether or not the volumes are in use ($$BATTNO).

Chart 01. Overview of Supervisor Entry into Attention Transients

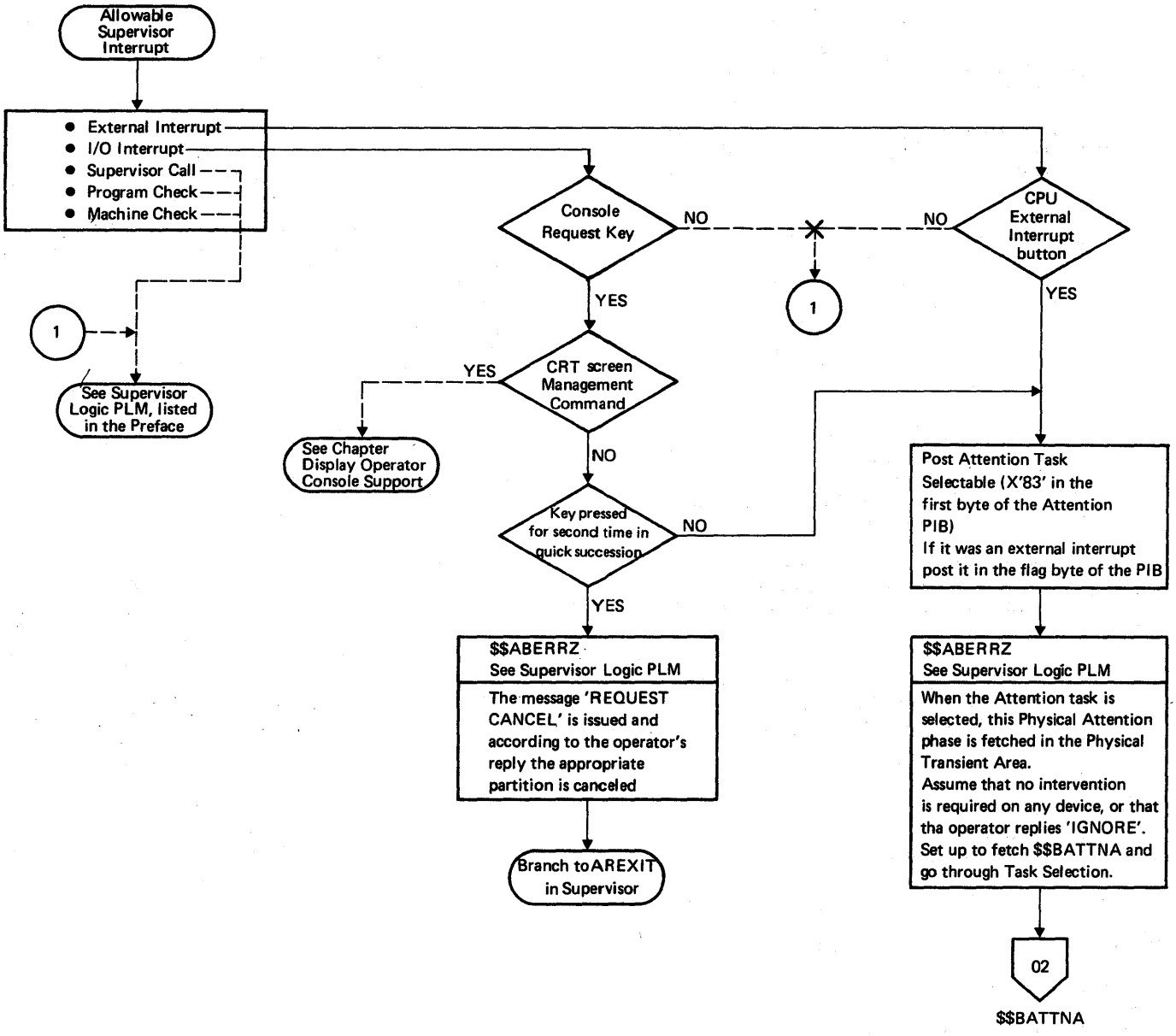
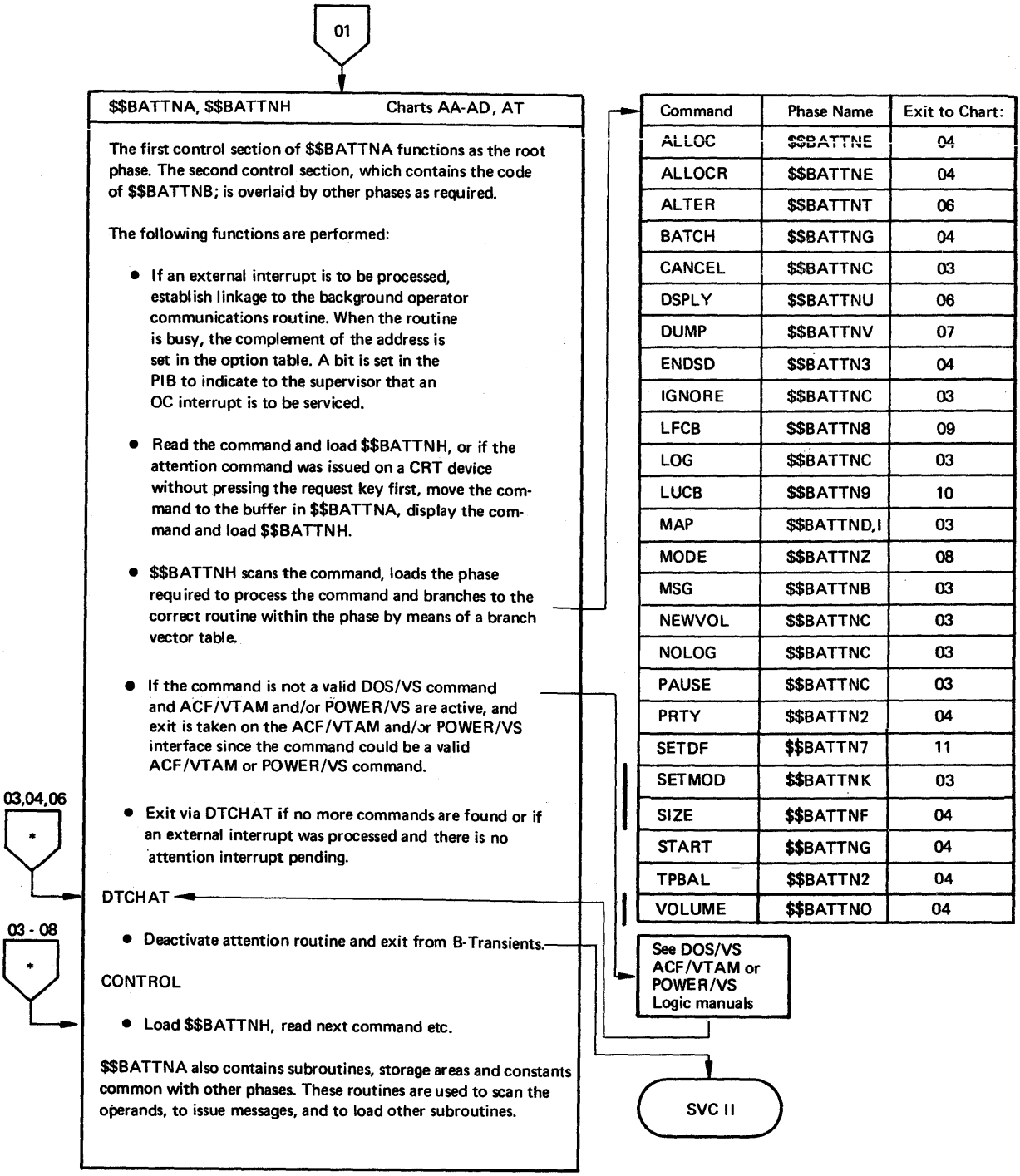


Chart 02. Logical Transient Attention Routines (Part 1 of 10)



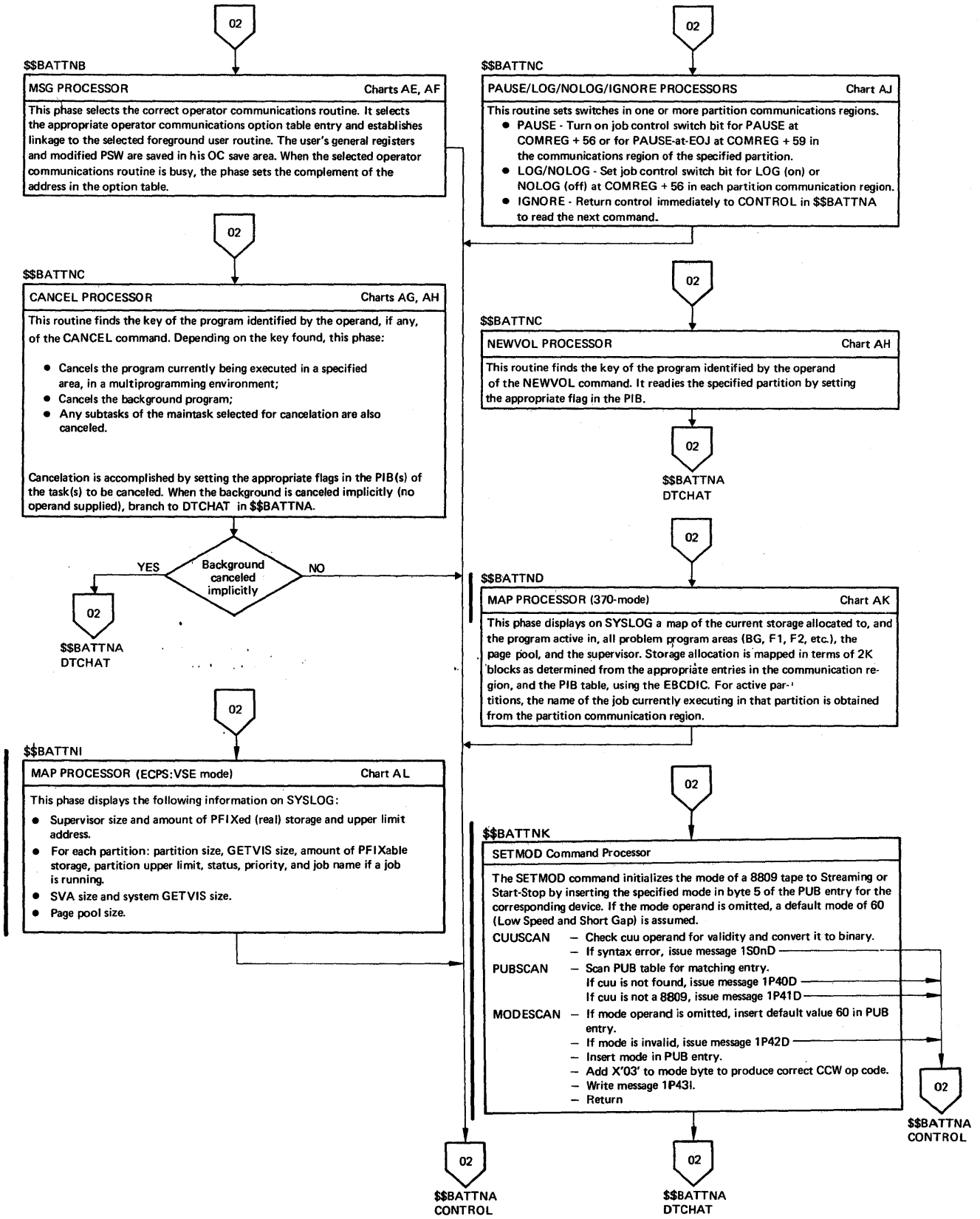
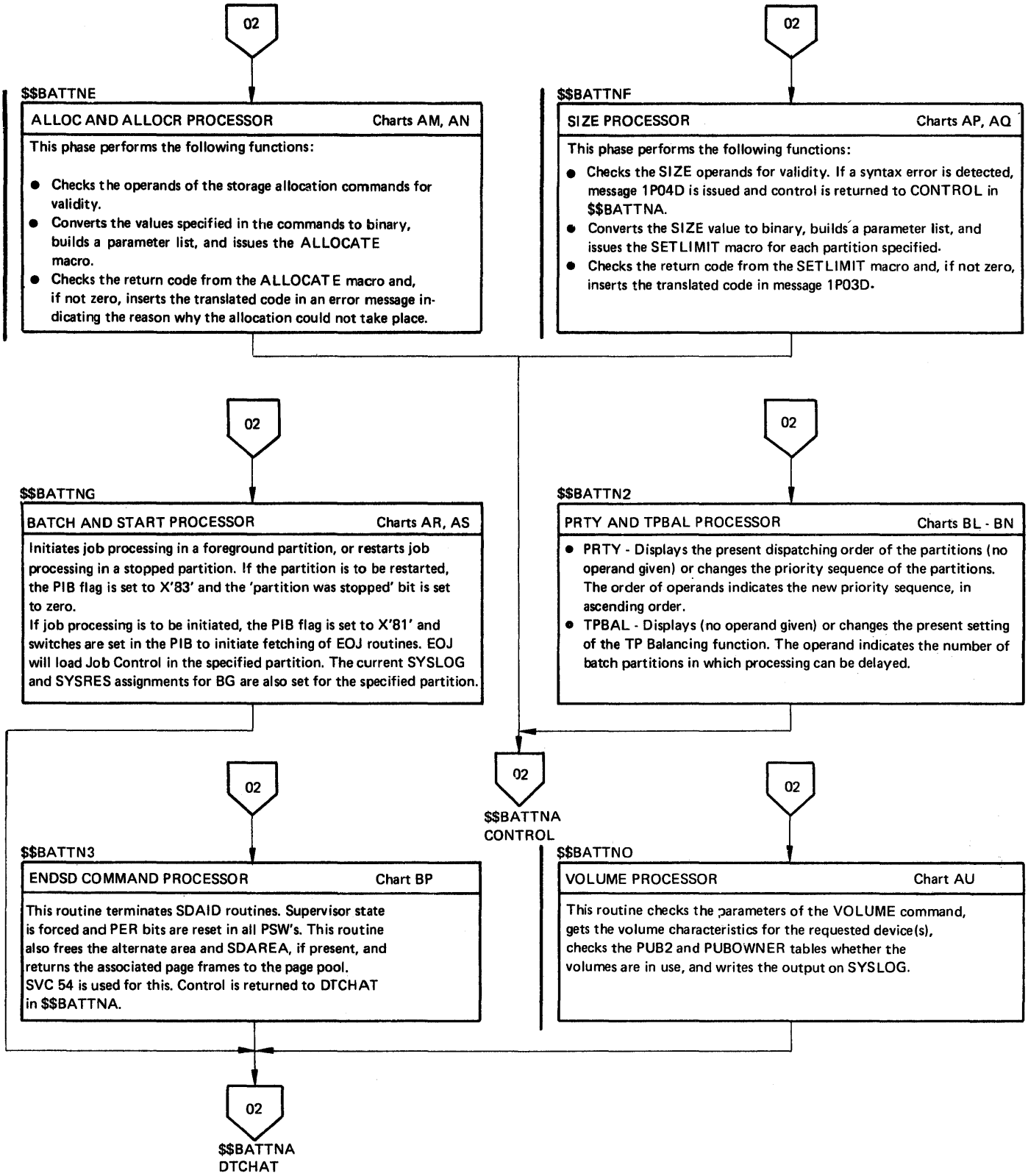

02
$$BATTNB
MSG PROCESSOR
Charts AE, AF
This phase selects the correct operator communications routine. It selects the appropriate operator communications option table entry and establishes linkage to the selected foreground user routine. The user's general registers and modified PSW are saved in his OC save area. When the selected operator communications routine is busy, the phase sets the complement of the address in the option table.
02
$$BATTNC
CANCEL PROCESSOR
Charts AG, AH
This routine finds the key of the program identified by the operand, if any, of the CANCEL command. Depending on the key found, this phase:
Cancels the program currently being executed in a specified area, in a multiprogramming environment;
Cancels the background program;
Any subtasks of the maintask selected for cancelation are also canceled.
Cancelation is accomplished by setting the appropriate flags in the PIB(s) of the task(s) to be canceled. When the background is canceled implicitly (no operand supplied), branch to DTCHAT in $$BATTNA.
Background canceled implicitly
YES
NO
02
$$BATTNA
DTCHAT
02
$$BATTNI
MAP PROCESSOR (ECPS:VSE mode)
Chart AL
This phase displays the following information on SYSLOG:
Supervisor size and amount of PFIXed (real) storage and upper limit address.
For each partition: partition size, GETVIS size, amount of PFIXable storage, partition upper limit, status, priority, and job name if a job is running.
SVA size and system GETVIS size.
Page pool size.
02
$$BATTNA
CONTROL
02
$$BATTNC
PAUSE/LOG/NOLOG/IGNORE PROCESSORS
Chart AJ
This routine sets switches in one or more partition communications regions.
PAUSE - Turn on job control switch bit for PAUSE at COMREG + 56 or for PAUSE-at-EOJ at COMREG + 59 in the communications region of the specified partition.
LOG/NOLOG - Set job control switch bit for LOG (on) or NOLOG (off) at COMREG + 56 in each partition communication region.
IGNORE - Return control immediately to CONTROL in $$BATTNA to read the next command.
02
$$BATTNC
NEWVOL PROCESSOR
Chart AH
This routine finds the key of the program identified by the operand of the NEWVOL command. It readies the specified partition by setting the appropriate flag in the PIB.
02
$$BATTNA
DTCHAT
02
$$BATTND
MAP PROCESSOR (370-mode)
Chart AK
This phase displays on SYSLOG a map of the current storage allocated to, and the program active in, all problem program areas (BG, F1, F2, etc.), the page pool, and the supervisor. Storage allocation is mapped in terms of 2K blocks as determined from the appropriate entries in the communication region, and the PIB table, using the EBCDIC. For active partitions, the name of the job currently executing in that partition is obtained from the partition communication region.
02
$$BATTNK
SETMOD Command Processor
The SETMOD command initializes the mode of a 8809 tape to Streaming or Start-Stop by inserting the specified mode in byte 5 of the PUB entry for the corresponding device. If the mode operand is omitted, a default mode of 60 (Low Speed and Short Gap) is assumed.
CUUSCAN – Check cuu operand for validity and convert it to binary.
– If syntax error, issue message 1S0nD
PUBSCAN – Scan PUB table for matching entry.
If cuu is not found, issue message 1P40D
If cuu is not a 8809, issue message 1P41D
MODESCAN – If mode operand is omitted, insert default value 60 in PUB entry.
– If mode is invalid, issue message 1P42D
– Insert mode in PUB entry.
– Add X'03' to mode byte to produce correct CCW op code.
– Write message 1P43I.
– Return
02
$$BATTNA
CONTROL
02
$$BATTNA
DTCHAT

Chart 04. Logical Transient Attention Routines (Part 3 of 10)



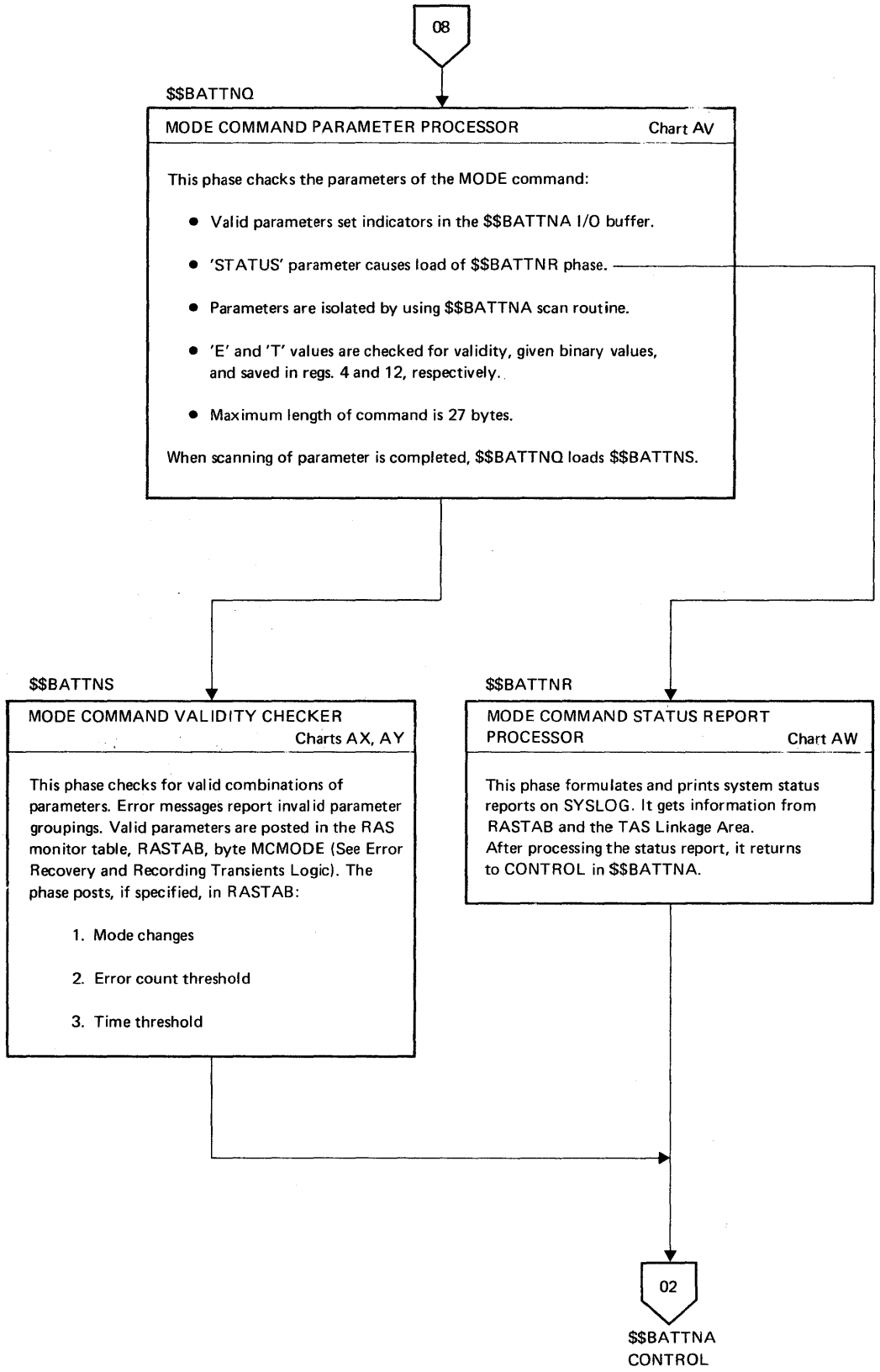

08
$$BATTNQ
MODE COMMAND PARAMETER PROCESSOR
Chart AV
This phase chacks the parameters of the MODE command:
• Valid parameters set indicators in the $$BATTNA I/O buffer.
• 'STATUS' parameter causes load of $$BATTNR phase.
• Parameters are isolated by using $$BATTNA scan routine.
• 'E' and 'T' values are checked for validity, given binary values, and saved in regs. 4 and 12, respectively.
• Maximum length of command is 27 bytes.
When scanning of parameter is completed, $$BATTNQ loads $$BATTNS.
$$BATTNS
MODE COMMAND VALIDITY CHECKER
Charts AX, AY
This phase checks for valid combinations of parameters. Error messages report invalid parameter groupings. Valid parameters are posted in the RAS monitor table, RASTAB, byte MCMODE (See Error Recovery and Recording Transients Logic). The phase posts, if specified, in RASTAB:
1. Mode changes
2. Error count threshold
3. Time threshold
$$BATTNR
MODE COMMAND STATUS REPORT PROCESSOR
Chart AW
This phase formulates and prints system status reports on SYSLOG. It gets information from RASTAB and the TAS Linkage Area.
After processing the status report, it returns to CONTROL in $$BATTNA.
02
$$BATTNA
CONTROL

Chart 06. Logical Transient Attention Routines (Part 5 of 10)

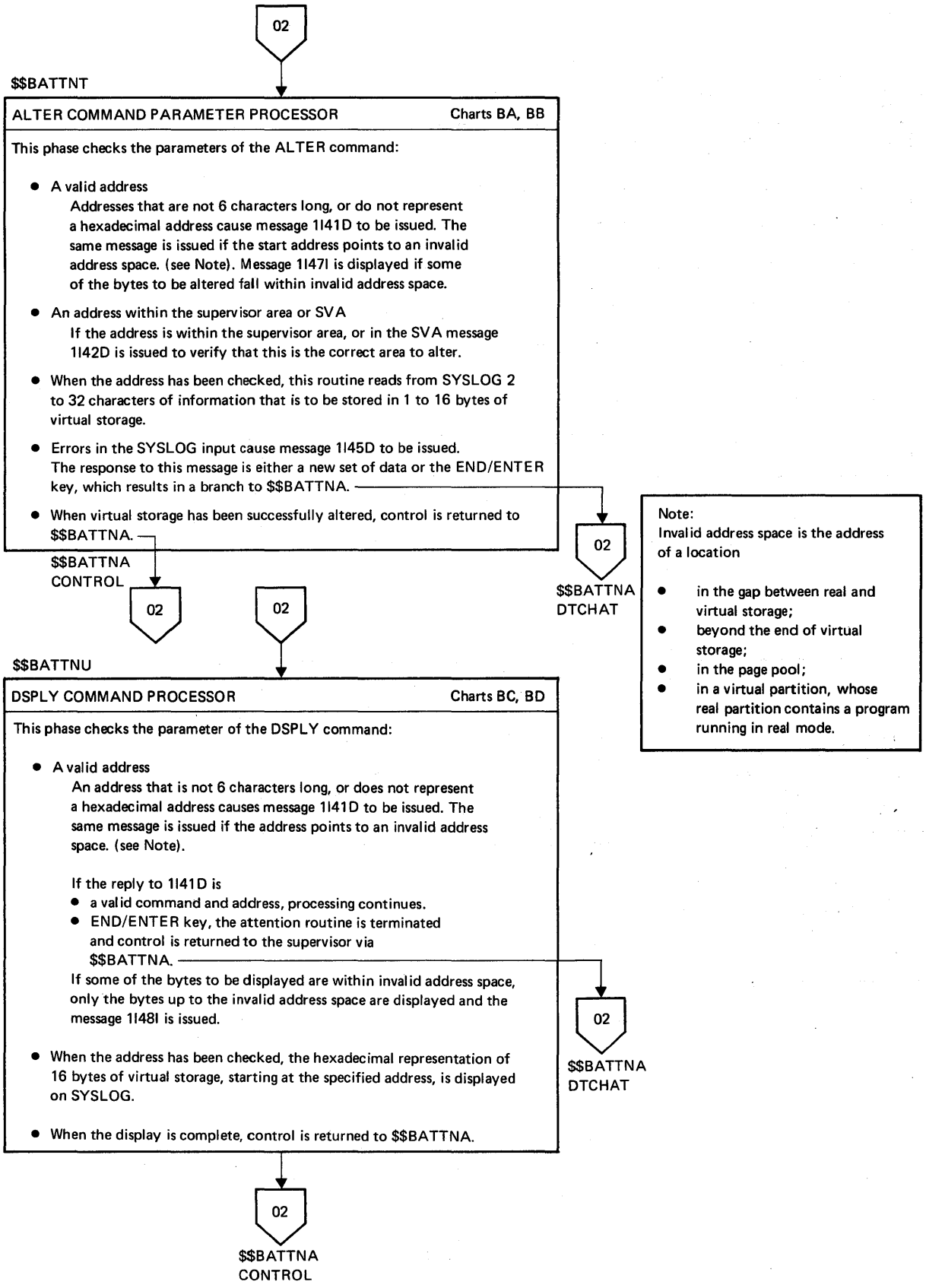


Chart 07. Logical Transient Attention Routines (Part 6 of 10)

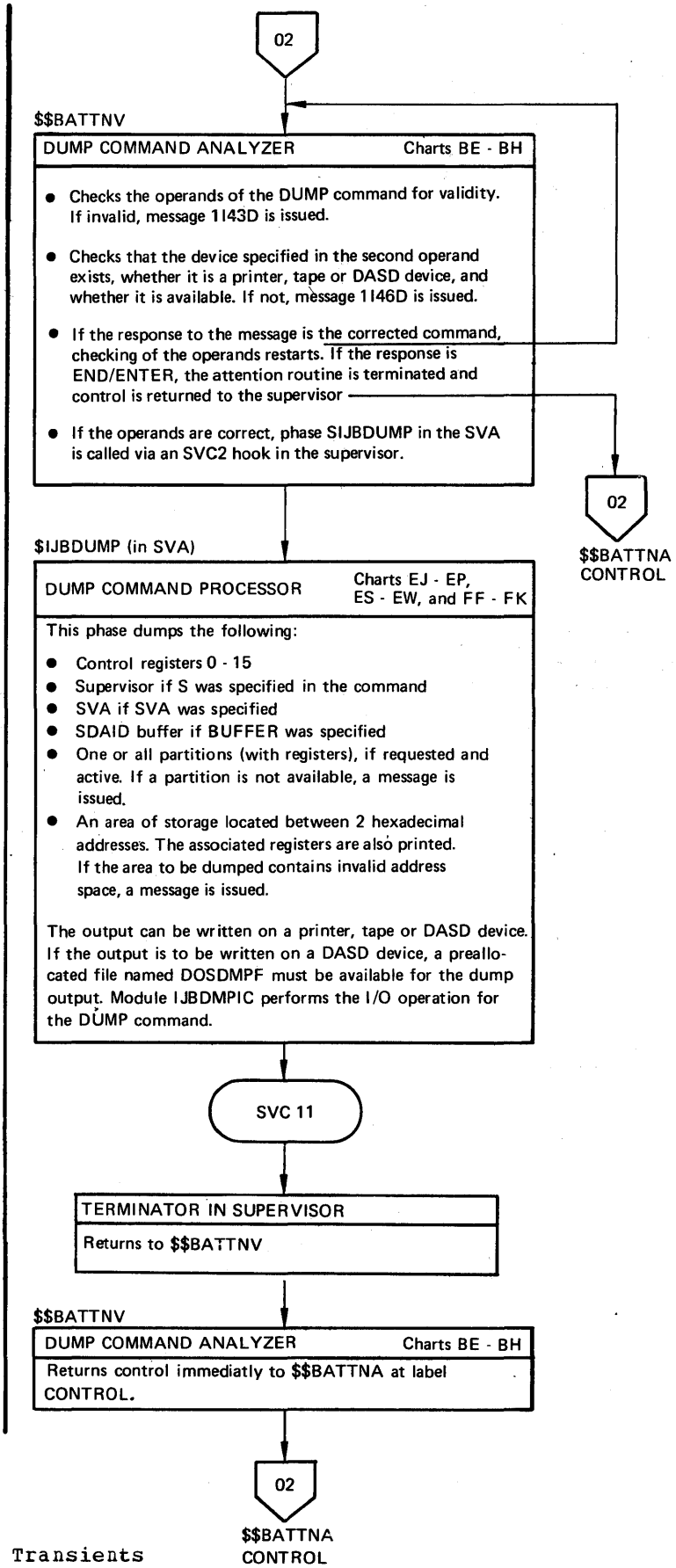
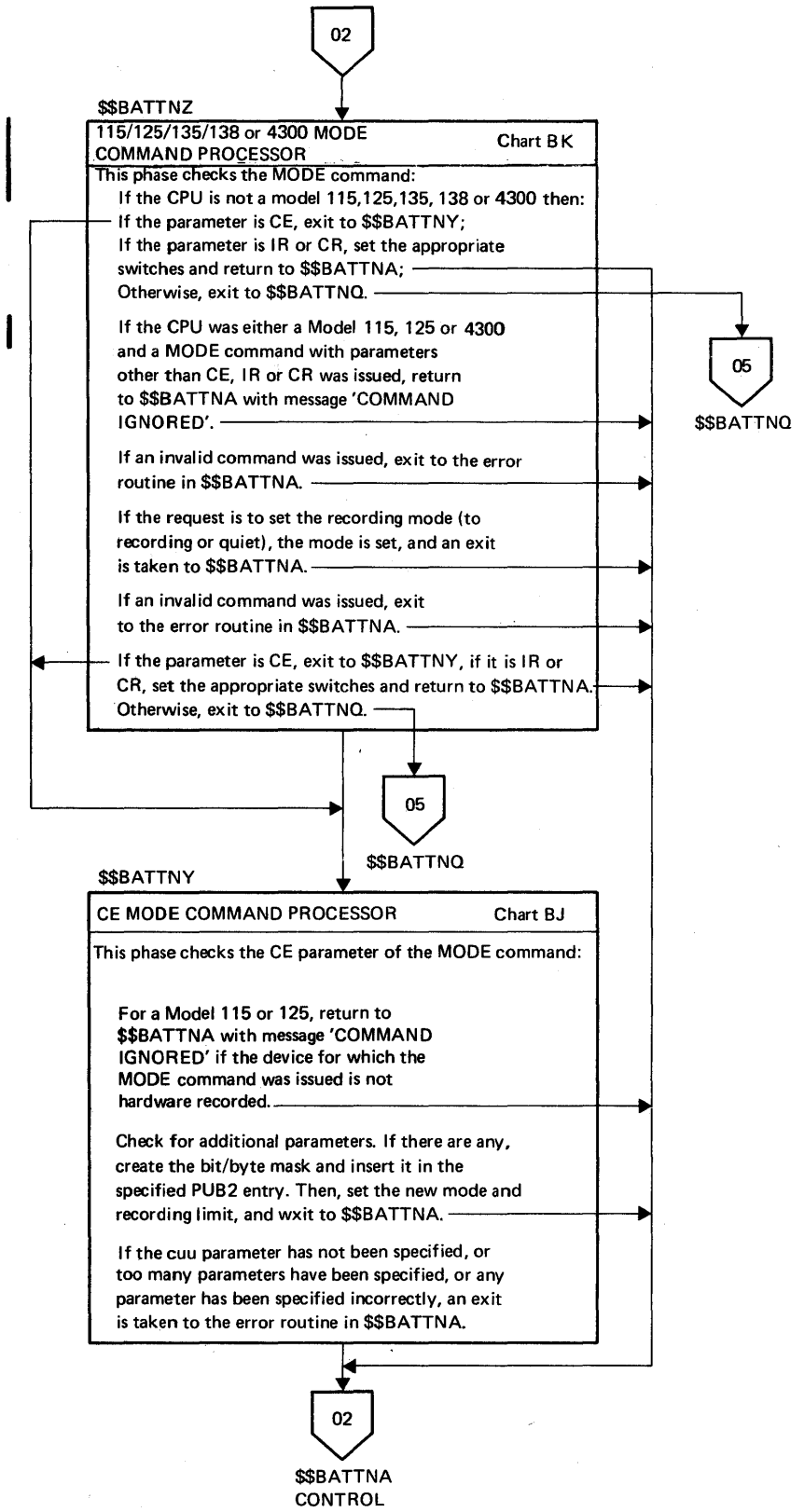


Chart 08. Logical Transient Attention Routines (Part 7 of 10)



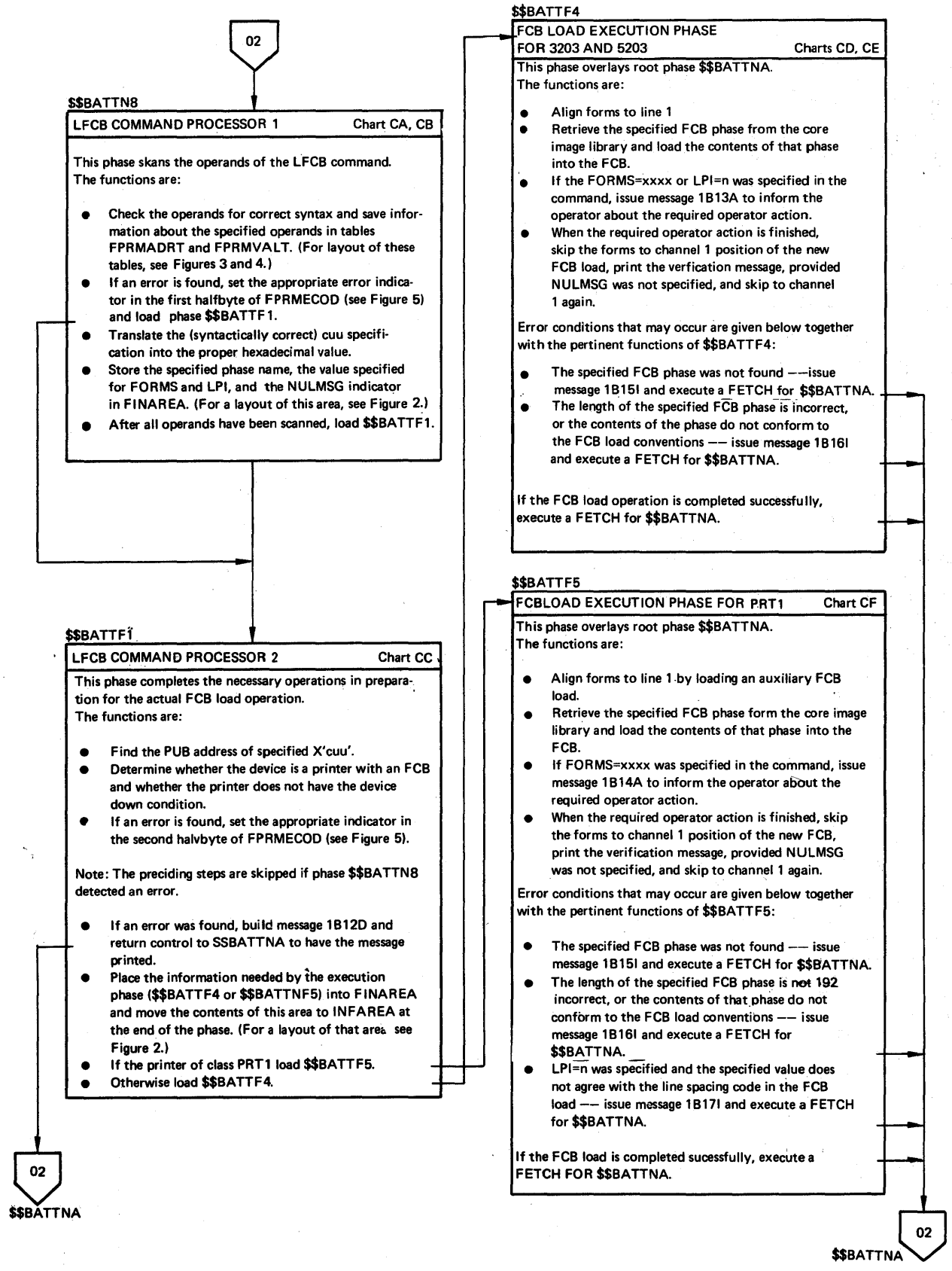

02
$$BATTN8
LFCB COMMAND PROCESSOR 1 Chart CA, CB
This phase skans the operands of the LFCB command.
The functions are:
• Check the operands for correct syntax and save information about the specified operands in tables FPRMADRT and FPRMVALT. (For layout of these tables, see Figures 3 and 4.)
• If an error is found, set the appropriate error indicator in the first halfbyte of FPRMECOD (see Figure 5) and load phase $$BATTF1.
• Translate the (syntactically correct) cuu specification into the proper hexadecimal value.
• Store the specified phase name, the value specified for FORMS and LPI, and the NULMSG indicator in FINAREA. (For a layout of this area, see Figure 2.)
• After all operands have been scanned, load $$BATTF1.
$$BATTF1
LFCB COMMAND PROCESSOR 2 Chart CC
This phase completes the necessary operations in preparation for the actual FCB load operation.
The functions are:
• Find the PUB address of specified X'cuu'.
• Determine whether the device is a printer with an FCB and whether the printer does not have the device down condition.
• If an error is found, set the appropriate indicator in the second halvbyte of FPRMECOD (see Figure 5).
Note: The preciding steps are skipped if phase $$BATTN8 detected an error.
• If an error was found, build message 1B12D and return control to SSBATTNA to have the message printed.
• Place the information needed by the execution phase ($$BATTF4 or $$BATTNF5) into FINAREA and move the contents of this area to INFAREA at the end of the phase. (For a layout of that area see Figure 2.)
• If the printer of class PRT1 load $$BATTF5.
• Otherwise load $$BATTF4.
02
$$BATTNA
$$BATTF4
FCB LOAD EXECUTION PHASE
FOR 3203 AND 5203 Charts CD, CE
This phase overlays root phase $$BATTNA.
The functions are:
• Align forms to line 1
• Retrieve the specified FCB phase from the core image library and load the contents of that phase into the FCB.
• If the FORMS=xxxx or LPI=n was specified in the command, issue message 1B13A to inform the operator about the required operator action.
• When the required operator action is finished, skip the forms to channel 1 position of the new FCB load, print the verfication message, provided NULMSG was not specified, and skip to channel 1 again.
Error conditions that may occur are given below together with the pertinent functions of $$BATTF4:
• The specified FCB phase was not found —issue message 1B15I and execute a FETCH for $$BATTNA.
• The length of the specified FCB phase is incorrect, or the contents of the phase do not conform to the FCB load conventions — issue message 1B16I and execute a FETCH for $$BATTNA.
If the FCB load operation is completed successfully, execute a FETCH for $$BATTNA.
$$BATTF5
FCBLOAD EXECUTION PHASE FOR PRT1 Chart CF
This phase overlays root phase $$BATTNA.
The functions are:
• Align forms to line 1 by loading an auxiliary FCB load.
• Retrieve the specified FCB phase form the core image library and load the contents of that phase into the FCB.
• If FORMS=xxxx was specified in the command, issue message 1B14A to inform the operator about the required operator action.
• When the required operator action is finished, skip the forms to channel 1 position of the new FCB, print the verification message, provided NULMSG was not specified, and skip to channel 1 again.
Error conditions that may occur are given below together with the pertinent functions of $$BATTF5:
• The specified FCB phase was not found — issue message 1B15I and execute a FETCH for $$BATTNA.
• The length of the specified FCB phase is not 192 incorrect, or the contents of that phase do not conform to the FCB load conventions — issue message 1B16I and execute a FETCH for $$BATTNA.
• LPI=n was specified and the specified value does not agree with the line spacing code in the FCB load — issue message 1B17I and execute a FETCH for $$BATTNA.
If the FCB load is completed sucessfully, execute a FETCH FOR $$BATTNA.
02
$$BATTNA

Chart 10. Logical Transient Attention Routines (Part 9 of 10)

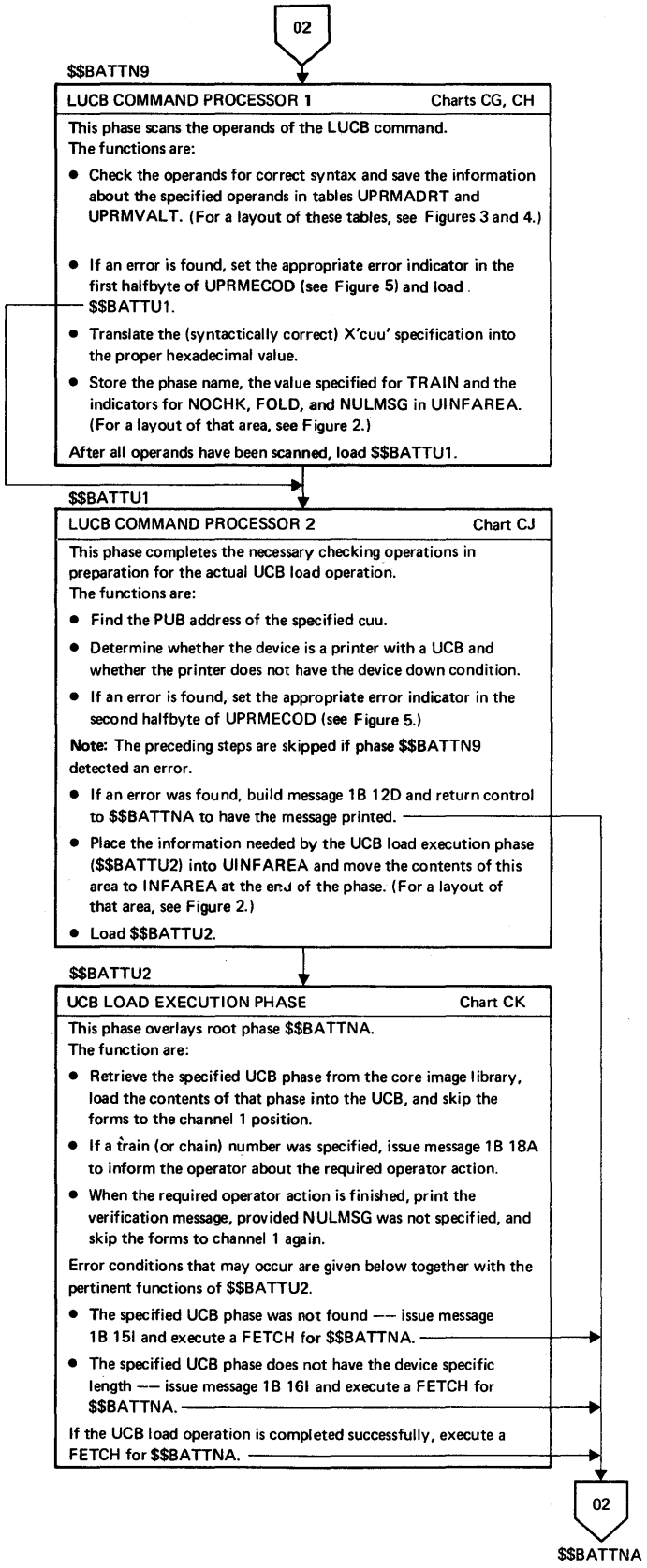
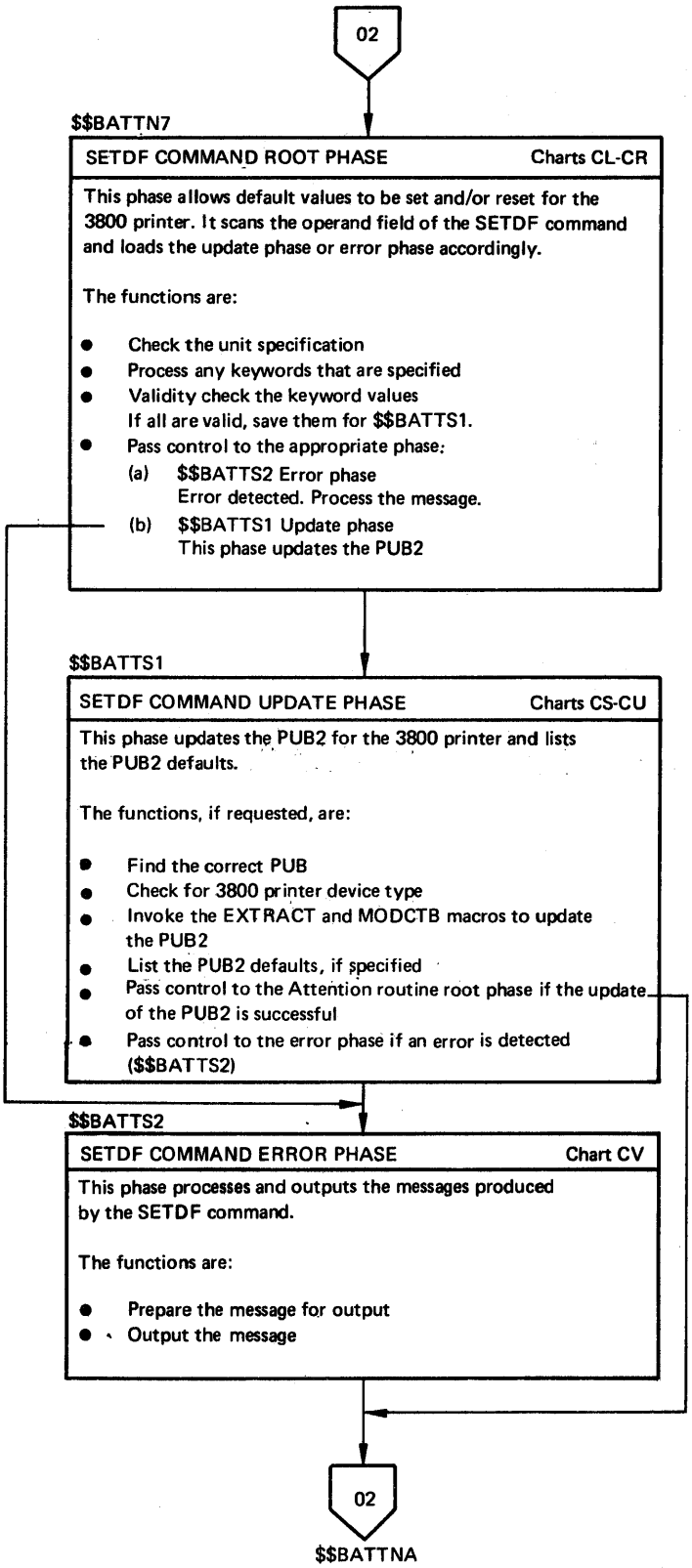


Chart 11. Logical Transient Attention Routines (Part 10 of 10)



FINFAREA	
Bytes	Contents
0- 7	specified phase name
8-11	form number, if specified, or blank
12	not used
13	number of lines per inch, if specified, or blank
14-15	hex value of specified unit address
16-19	address of PUB
20-22	specified cuu as printable characters
23	not used
24	Information byte bit 0 - not used bit 1 - FOLD was specified bit 2 - NOCHK was specified bit 3 - NULMSG was specified bit 4 - FORMS=xxxx was specified bit 5 - LPI=n was specified bits 6 and 7 - not used
25	X'00' (used to clear table FPRMADRT)

UINFAREA	
Bytes	Contents
0- 7	specified phase name
8-13	train number, if specified, or blank
14-15	hex value of specified unit address
16-19	address of PUB
20-22	specified cuu as printable characters
23	not used
24	Information byte bit 0 - not used bit 1 - FOLD was specified bit 2 - NOCHK was specified bit 3 - NULMSG was specified bit 4 - TRAIN=xxxxxx was specified bit 5 - LPI=n was specified bits 6 and 7 - not used
25	X'00' (used to clear table UPRMADRT)

Note: This area (FINFAREA in $$BATTN8 and UINFAREA in $$BATTN9) is used to build an information record. This record is built by phases $$BATTN8 and $$BATTF1 for an LFCB command and by phases $$BATTN9 and $$BATTU1 for an LUCB command. Phases $$BATTF1 and $$BATTU1, respectively, move this record to INFAREA before they call the appropriate load execution phase ($$BATTF4 or $$BATTF5 for an LFCB command, $$BATTU2 for an LUCB command).

Figure 2. Layout of Communication Areas in $$BATTN8 and $$BATTN9

Bytes	Contents
0- 3	address of keyword
4- 5	length of keyword
6- 7	sequence number of operand
8	not used
9	delimiter
10-13	address of keyword value
14-15	length of keyword value
16-17	sequence number of operand
18	not used
19	delimiter

Note: This table is called FPRMADRT in $$BATTN8 and UPRMADRT in $$BATTN9.

Figure 3. Layout of Parameter Address Tables in $$BATTN8 and $$BATTN9

Bytes	Contents
0- 9	specified keyword
10-19	specified keyword value

Note: This table is called FPRMVALT in $$BATTN8 and UPRMVALT in $$BATTN9.

Figure 4. Layout of Parameter Value Tables in $$BATTN8 and BATTN9

Code (in hex)	Meaning
F0	Operand is invalid.
70	Length of positional operand or of keyword if incorrect or the delimiter or the operand or the keyword is erroneous.
30	Keyword value has wrong length or the value's delimiter is erroneous.
10	Keyword is specified more than once.
07	No PUB entry was found for the specified channel and unit address.
03	Specified device has no FCB (for an LFCB command or macro) or no UCB (for an LUCB command).
01	Specified device is down.

Note: This error code byte is called FPRMECOD in $$BATTN8 and UPRMECOD in $$BATTN9.

Figure 5. Error Code Bytes in $$BATTN8 and $$BATTN9

A program is terminated under its own control by issuing an EOJ, DUMP, or CANCEL macro or through operator action or a program error or certain I/O failures. When a program is terminated, the following actions are taken:

1. All I/O operations that the program has requested are allowed to quiesce.

2. DASD extents used by this program for DASD file protection are dequeued. This feature is a system generation option.

3. If the program is terminated normally, $JOBCTLA is called.

4. If the program is terminated abnormally, an error message is printed and a dump is issued on SYSLST if the dump option was specified.

5. If the terminating task is a subtask, it is detached from the system's task selection mechanism.

   For a list of cancel codes see Figure 6.

Chart 12. Terminator Routines (Part 1 of 3)

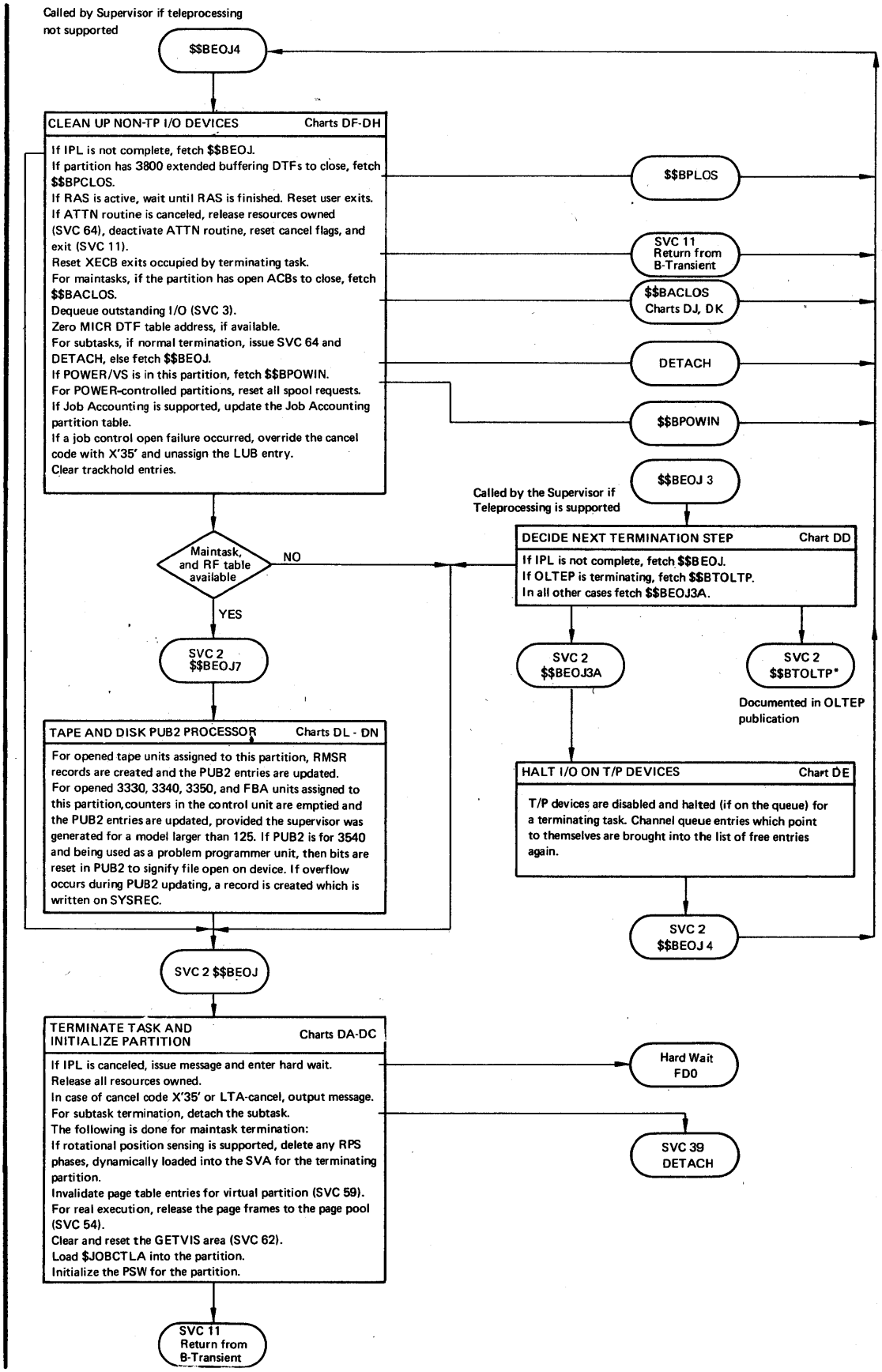
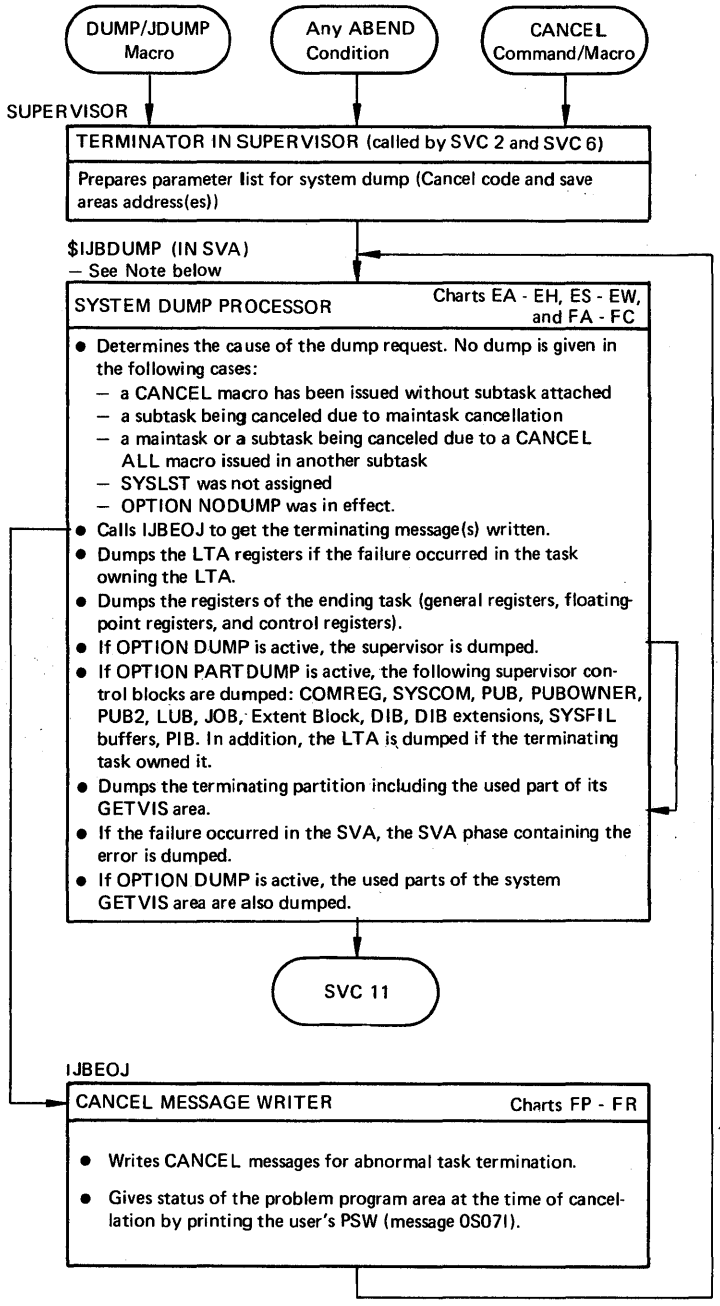
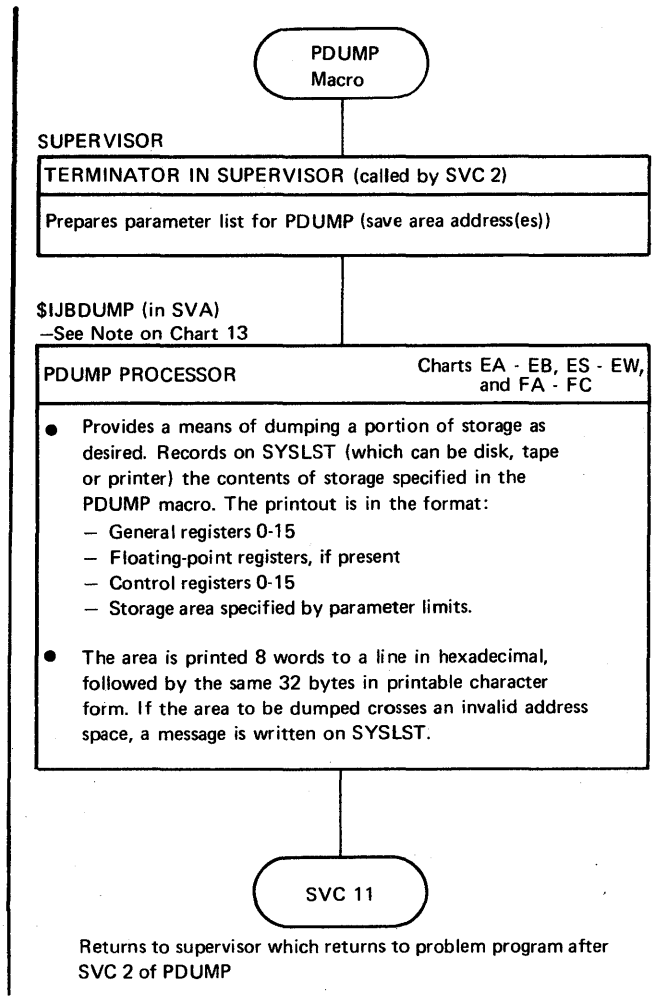


Chart 13. Terminator Routines (Part 2 of 3)



Note:
$IJBDUMP contains the following routines:

IJBSDUMP (Charts EA - EP):	– System dump monitor – PDUMP monitor – DUMP command monitor
IJBDMPPA (Charts EQ - ER):	Dumps supervisor control blocks
IJBDMPAR (Charts ES - EW):	Prepares an area of storage for output and validates addresses
IJBDMPIO (Charts FA - FC):	Performs I/O for system dump and PDUMP macro
IJBDMPIC (Charts FF - FK):	Performs I/O for DUMP command
IJBEOJ (Charts FP - FR):	Writes termination messages on SYSLOG and SYSLST



Returns to supervisor which returns to problem program after SVC 2 of PDUMP

Cancel Code (hex)	Message Code	Descriptive Part of Message or Condition
0F	0P80I	Invalid read from or write to system file on FBA
10	.....	Normal EOJ
11	0V07I	No channel program translation for unsupported device
12	0V06I	Insufficient buffer space for channel program translation
13	0V05I	CCW with count greater than 32K
14	0V04I	Page pool too small
15	0V02I	Page fault in disabled program
16	0V01I	Page fault in MICR stacker select or P.F. appendage routine
17	0S02I	Program request (same as 23 but causes dump because subtasks were attached when maintask issued CANCEL macro)
18	.....	Eliminates cancel message when maintask issues DUMP macro with subtasks attached
19	0P74I	I/O operator option
1A	0P73I	I/O error
1B	0P82I	Channel failure
1C	0S14I	CANCEL ALL macro
1D	0S12I	Main task termination
1E	0S13I	Unknown ENQ requestor
1F	0P81I	CPU failure
20	0S03I	Program check
21	0S04I	Illegal SVC
22	0S05I	Phase not found
23	0S02I	Program request
24	0S01I	Operator intervention
25	0P77I	Invalid address
26*	0P71I	SYSxxx not assigned(unassigned LUB code)
27	0P70I	Undefined logical unit

Figure 6. Cancel Codes and Messages (Part 1 of 2)

Cancel Code (hex)	Message Code	Descriptive part of Message or Condition
2A	0V10I	I/O error on page data set
2B	0P84I	I/O error during fetch from private core image library
2C	0V09I	Illegal parameter passed by P.F. appendage routine
2D	0P88I	Program cannot be executed/restarted due to failing storage block
2E	0S16I	Invalid resource request (possible deadlock)
2F	0V03I	More than 255 PFIX requests for 1 page
30	0P72I	Reading past /& statement (on SYSRDR or SYSIPT)
31	0P75I	I/O error queue overflow (error queue overflow)
32	0P76I	Invalid DASD address
33	0P79I	Invalid first CCW
34		Reserved
35	0P85I	Job control open failure
36	0V08I	Page fault in I/O appendage routine
37		Reserved
38	0V11I	Wrong privately translated CCW
39		Reserved
40	5J95I	Invalid termination of VTAM
41	5J96I	Invalid VTAM condition code
42	0P86I	Violated DASD file protection
	0P83A**	Supervisor catalog failure
	0P87A**	IPL failure
all others	0P78I	Unrecogized cancel code xx

* If the CCB is not available, the logical unit is SYSxxx.

** The cancel code is not significant in case of a supervisor catalog or IPL failure, because the system is placed in the wait state without any further processing by the Terminator.

Note: In addition to recognizing the cancel codes above, the Terminator also recognizes the same codes with the X'80' bit on (cancel occurred in LTA).

Figure 6. Cancel Codes and Messages (Part 2 of 2)

Loading Print Control Buffers

The print control buffers of a printer can be loaded as follows:

- Automatically during IPL. $$BUFLDR together with $$BUFLD1 and $$BUFLD2 are provided to load the FCB and UCB of a pertinent printer with the standard control information. (See DOS/VSE Initial Program Load and Job Control Logic.)
- Dynamically by issuing the LFCB or LUCB attention command. (See the section Attention Routines).
- As a separate job step by executing the SYSBUFLD program. (See DOS/VSE Initial Program Load and Job Control Logic.)
- Dynamically by issuing the LFCB macro in a problem program. (This macro can only be used to load the FCB of a printer).

$$BATTF0 is executed when an LFCB macro is issued in a problem program (see Chart 15). This phase performs the necessary checking functions for the requested FCB load operation and calls $$BATTF2 (for a PRT1 printer) or $$BATTF3 (for a 3203 or 5203 printer). The phases $$BATTF2 and $$BATTF3 return control to the problem program that issued the LFCB macro. The phases indicate the result of the load operation to the problem program by means of a return code in register 15. These return codes are:

Return code	Meaning
00	The FCB load operation has been completed successfully.
04	The assigned printer is of class PRT1 and the LPI operand specified in the macro does not agree with the FCB image.
08	No LUB is available for the specified logical unit.
0C	The specified logical unit has not been assigned or is currently unassigned.
10	The specified logical unit has been assigned to a device without an FCB.
14	The printer assigned to the specified logical unit is down.
18	The specified FCB image has not been found.
1C	The specified FCB image is invalid for the printer assigned to the specified logical unit.

Bytes	Contents
0-7	Specified phase name
8-10	xxx of SYSxxx specified in the macro
11	FORMS indicator: X'F1' = FORMS=xxx was specified
12-15	Form number specified in the macro
16	LPI value specified in the macro or X'F0'
17	NULMSG indicator: X'F1' = NULMSG was specified

Figure 7. Layout of the LFCB Macro Information Area

Bytes	Contents
0-7	Specified phase name
8-11	Form number, if specified, or blank
12	Not used
13	Number of lines per inch, if specified, or blank
14-15	Logical unit class and number
16-19	Address of PUB
20-23	Address of user register 15 save area
24	Information byte Bits 0-2: Not used Bit 3 : NULMSG was specified Bit 4 : FORMS=xxxx was specified Bit 5 : LPI=n was specified Bits 6-7: Not used
25	Not used

Figure 8. Layout of LFCB Macro Communication Area

$$BSYSWR

$$BSYSWR is used by MAINT, IPL, $LNKEDT, CORGZ and $LIBSTAT to turn on a bit in the communications region, which allows programs to write on DASD devices that are DASD file protected. In addition, the transient moves the address of the label information area to the communication region.

$$BCCHHR

$$BCCHHR is used by IPL to scan the core image directory for all modules whose names begin with $$RAST. It builds an in-core directory (load list) for the RAS phases. This directory is located in the RAS table in the supervisor. The directory facilitates the fetching function of the RAS modules into the RAS transient area (RTA).

$$BPCLOS - Automatic Close for 3800 Printer Files

The objective of this routine is to perform close processing for any 3800 printer files opened in extended buffering mode and left unclosed by the user program; this ensures that all buffered data is printed.

Entry: From $$BEOJ4 when a main task terminates; from $$BCLOSE and $$BCLOS2 when close processing initiated by $$BPCLOS is complete.

Exits: To $$BCLOSE to close a 3800 printer file left unclosed by a user program; to $$BEOJ4 when all 3800 printer files have been closed.

Method: This module uses the chain of DTFXWAs (DTF extension work areas) to identify 3800 printer files opened in extended buffering mode and left unclosed by the user program.

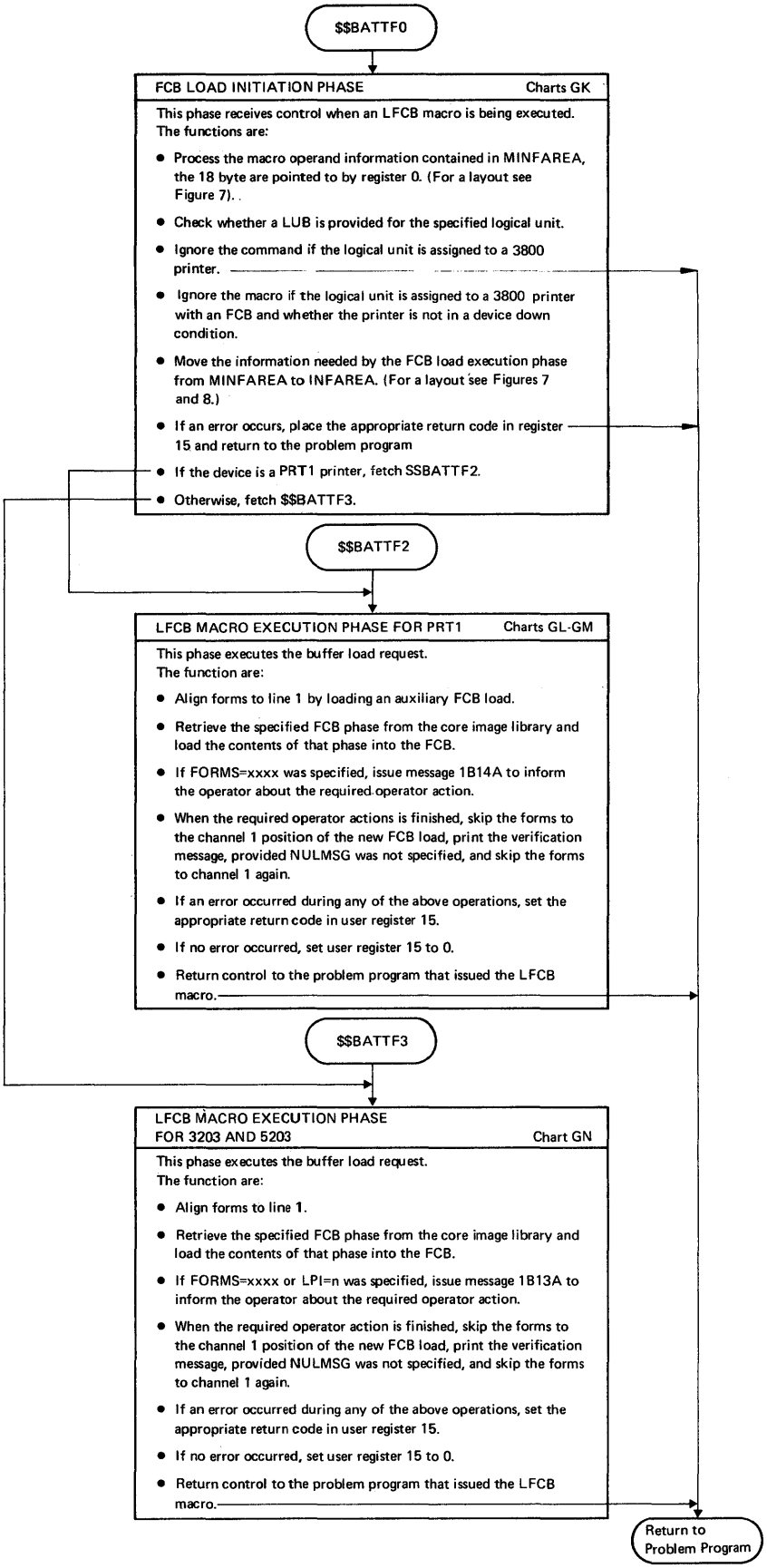
The DTFXWA chain is anchored in the anchor table extension (ATX). The ATX is created when the 3800 open routine IJDPR3 issues CDLOAD for the non-executable module IJDANCHX. When the IJDANCHX module is loaded, it becomes the ATX itself. Thus, the ATX is pointed to by the entry in the anchor table for module IJDANCHX.

$$BPLCOS selects the first DTFXWA on the chain, builds a close parameter list using the DTF pointer in the DTFXWA, and initiates close processing for the file by transferring control to $$BCLOSE. Before invoking $$BCLOSE, $$BPCLOS marks the DTFXWA as having been already selected for close processing. Thus, if close fails and the DTFXWA is left on the chain, it will have been marked as already processed.

After completing their processing, the close routines will fetch $$BPCLOS which will initiate close processing for the next unprocessed file on the DTFXWA chain. This will continue until all files have been processed.

If all files have been processed, but there are still DTFXWAs on the chain, this indicates that one or more of the files did not close successfully. In this case, a message is issued to inform the user of the situation. After all of the 3800 printer files have been processed, $$BPCLOS returns control to $$BEOJ4.

Chart 15. LFCB Macro Processor



DISPLAY OPERATOR CONSOLE SUPPORT

CRT transient routines provide support for the display operator consoles 125D (DOC) and 3277, and 3278 Model 2A. CRT transients reside in the core image library and are fetched into the CRT transient area (CRTTRNS) each time the supervisor encounters a request for a 125D or 3277/3278. Since these requests may require different services, CRT support is divided into 30 transient phases. Each phase fits into the CRT transient area (CRTTRNS) which is 1156 bytes long.

CRT Transient Logical Interrelationship

Each time the CRT support is activated, control is first transferred to phase $$BOCRTA (see Chart 16). Since this phase is also the last phase executed before the CRT support is deactivated, $$BOCRTA is always in the CRT transient area (CRTTRNS), even when the CRT support is not active.

The CRT transient phases perform the following functions:

- Channel Program Processing. The channel programs for a printer-keyboard console I/O request (referred to as SYSLOG channel programs, although the device may be assigned to any programmer logical unit) are interpreted and translated (see Charts 17 and 18).

- Screen Management.
  Screen management performs two logically different functions:

  a. K-command processing and/or automatic screen management during any I/O. This is handled by phases $$BOCRTK through $$BOCRTQ, $$BOCRTV through $$BOCRTY (see Charts 19 and 20).

  b. Redisplay of messages recorded on the hard-copy disk file. This is handled by phases $$BOCRT1 through $$BOCRT6 (see Chart 21).

- Error Condition Handling.
  Errors may arise from wrong parameters of K- and D-(redisplay) commands. These errors are detected and their repair is included throughout the screen management phases.

  The hard-copy file phase $$BOCRTH, the redisplay phases $$BOCRT3 and $$BOCRT5, and phase $$BOCRTU must write to (or read from) the disk on which the hard-copy file resides. Disk errors that cannot be handled by the supervisor-resident ERP routine are processed by $$BOCRTB (see Chart 22).

  Errors that appear during screen I/O operations are handled by phase $$BOCRTE, which in turn may fetch $$BOCRTF (see Chart 22).

A summary of the CRT transient phases is given in Charts 16 through 22. It is followed by a detailed description of each phase in alphabetical order.

Differences Between CRT Support and 1052 Support

CRT support and 1052 support are compatible with the following exceptions:

- CRT does not support program-controlled interrupts (PCI). The PCI bit in any user-supplied CCW chain is ignored, that is, the CCW is handled as if the PCI bit had been generated off.

- An SVC 0 given for a display operator console always causes the associated data to be displayed on a new line, even if the preceding SVC 0 has ended with a 'Write without carriage return' command. (This applies only to read and write operations.)

- When a Read command is encountered whose Skip-Data bit is generated on, the user is allowed to enter all his data. However, the data is <u>not</u> displayed in the message area.

- If an incorrect length is detected for a read operation, the Incorrect-Length bit in the CSW is set on only if the byte count is not exhausted and the SLI (Suppress Length Indicator) bit in the associated CCW is zero.

Chart 16. CRT Root Phases

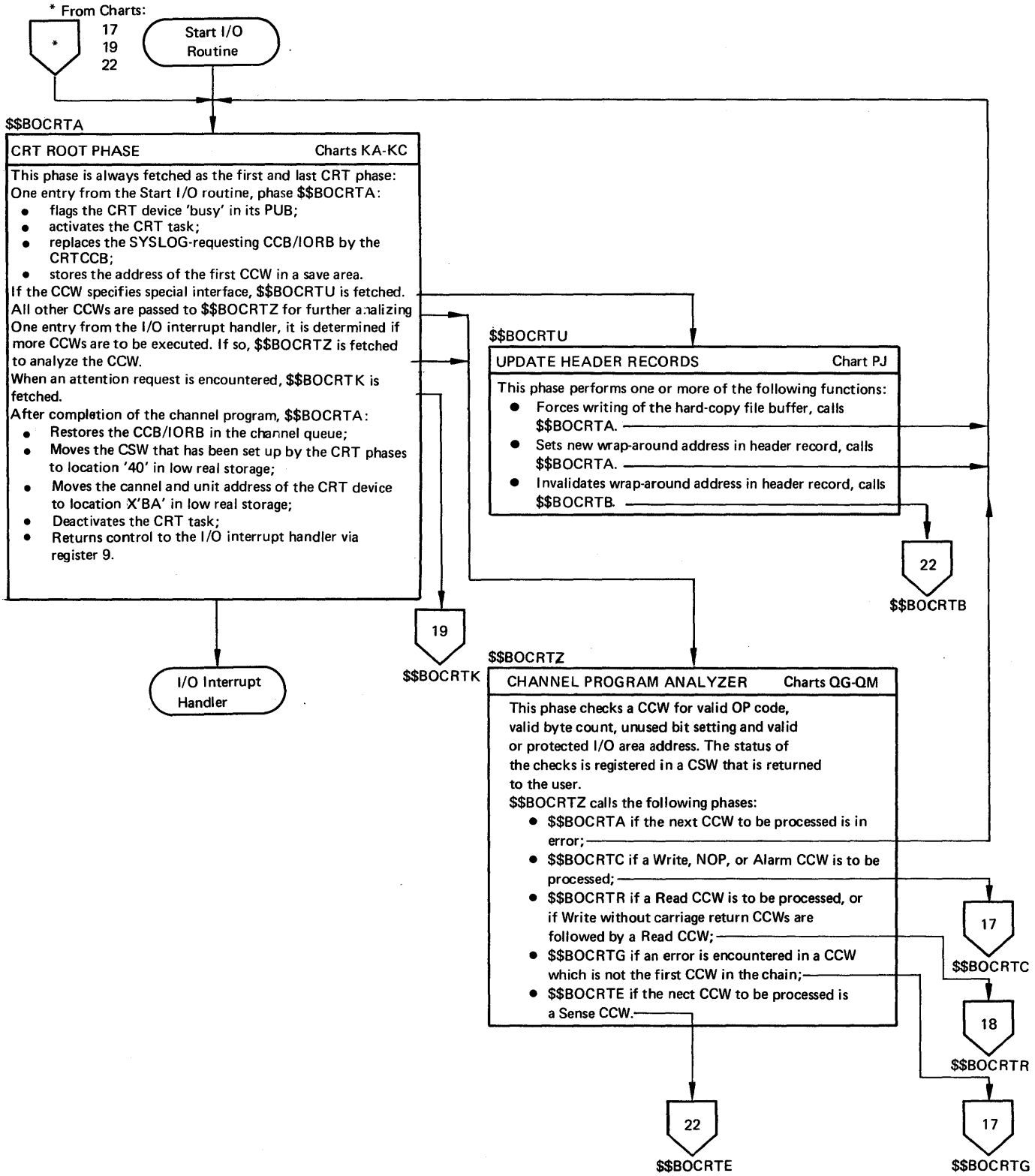


Chart 17. CRT Write Processor Phases

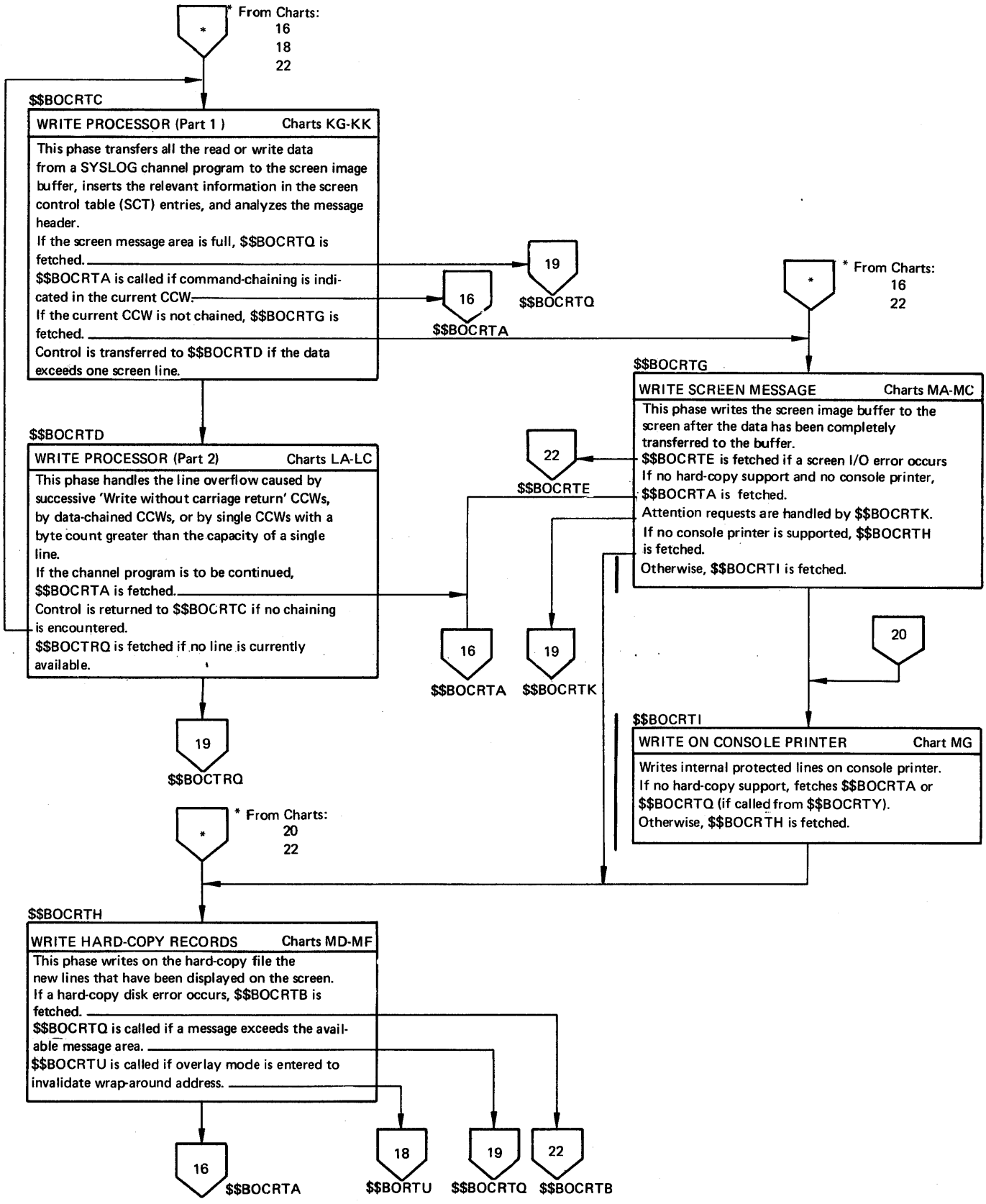


Chart 18. CRT Read Processor Phases

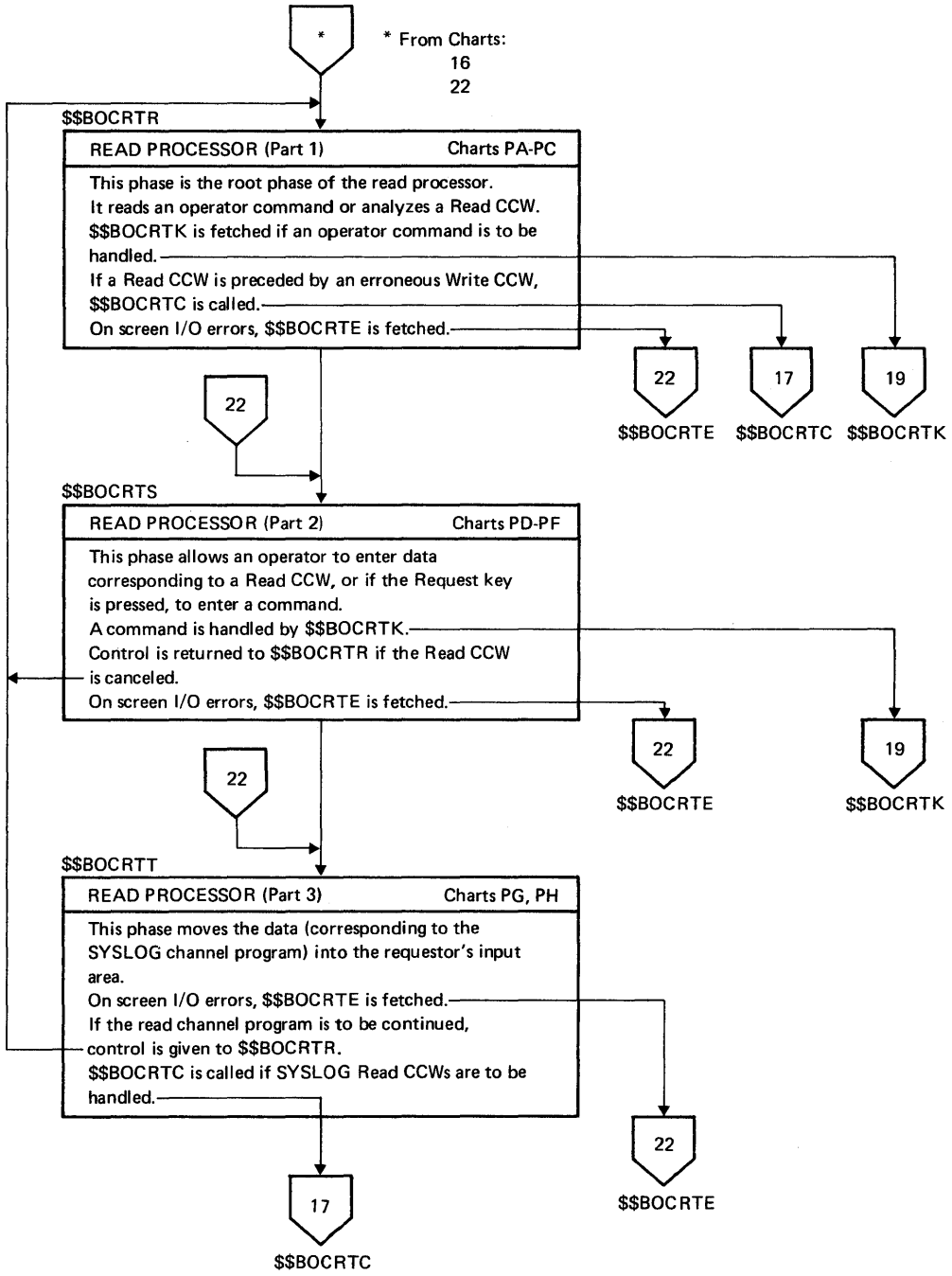


Chart 19. CRT Screen Management Phases (Part 1 of 2)

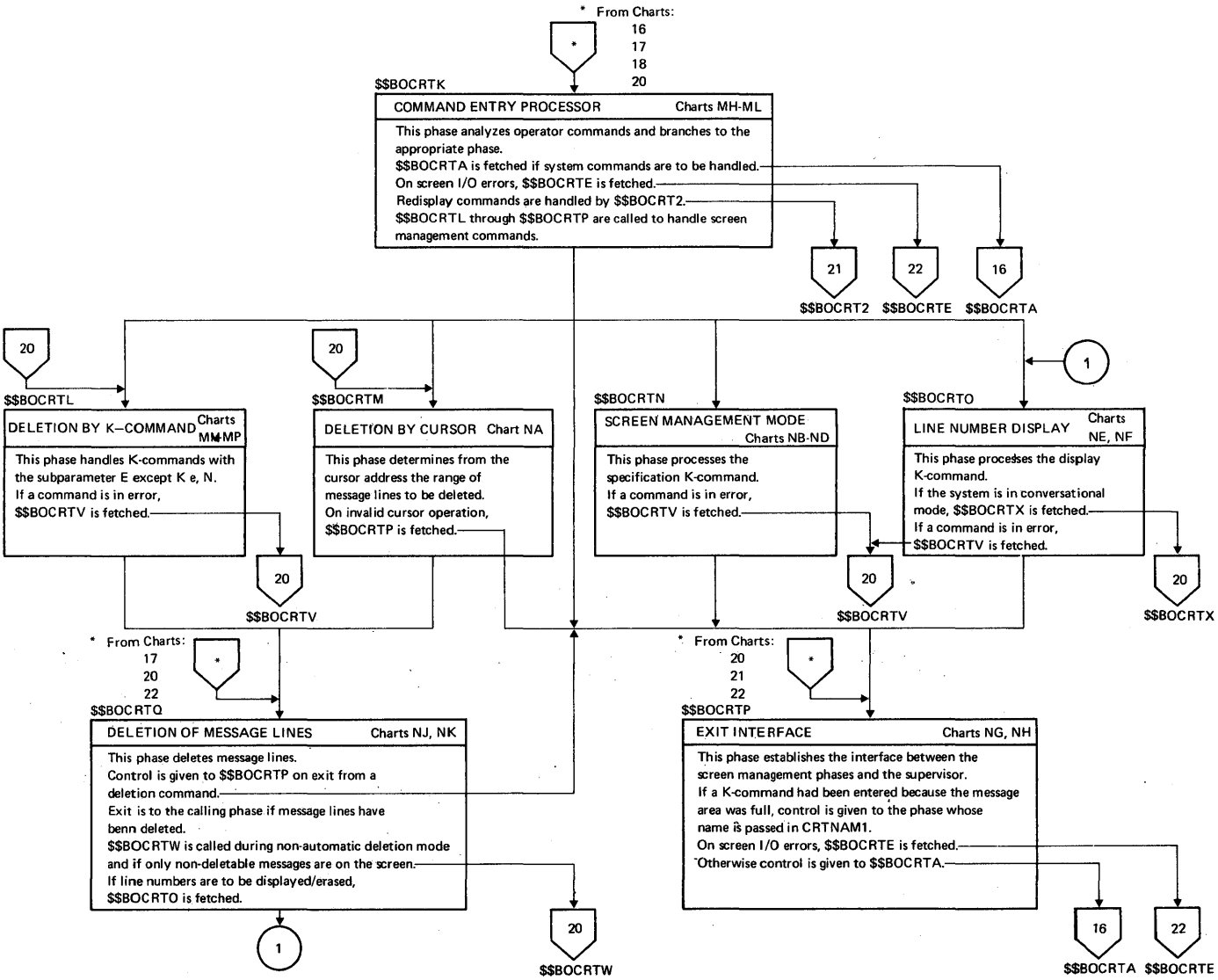


Chart 20. CRT Screen Management Phases (Part 2 of 2)

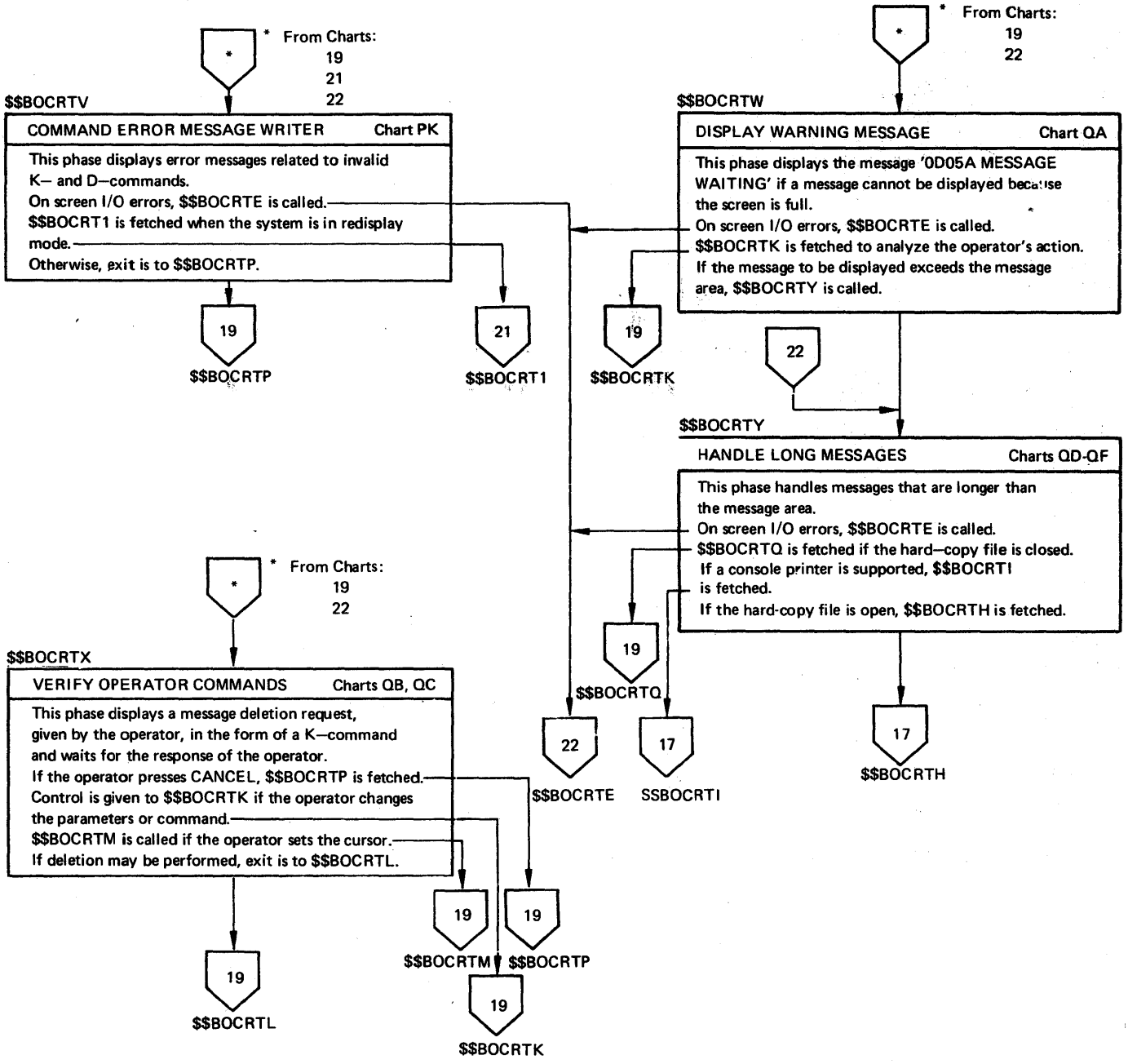


Chart 21. CRT Redisplay Phases

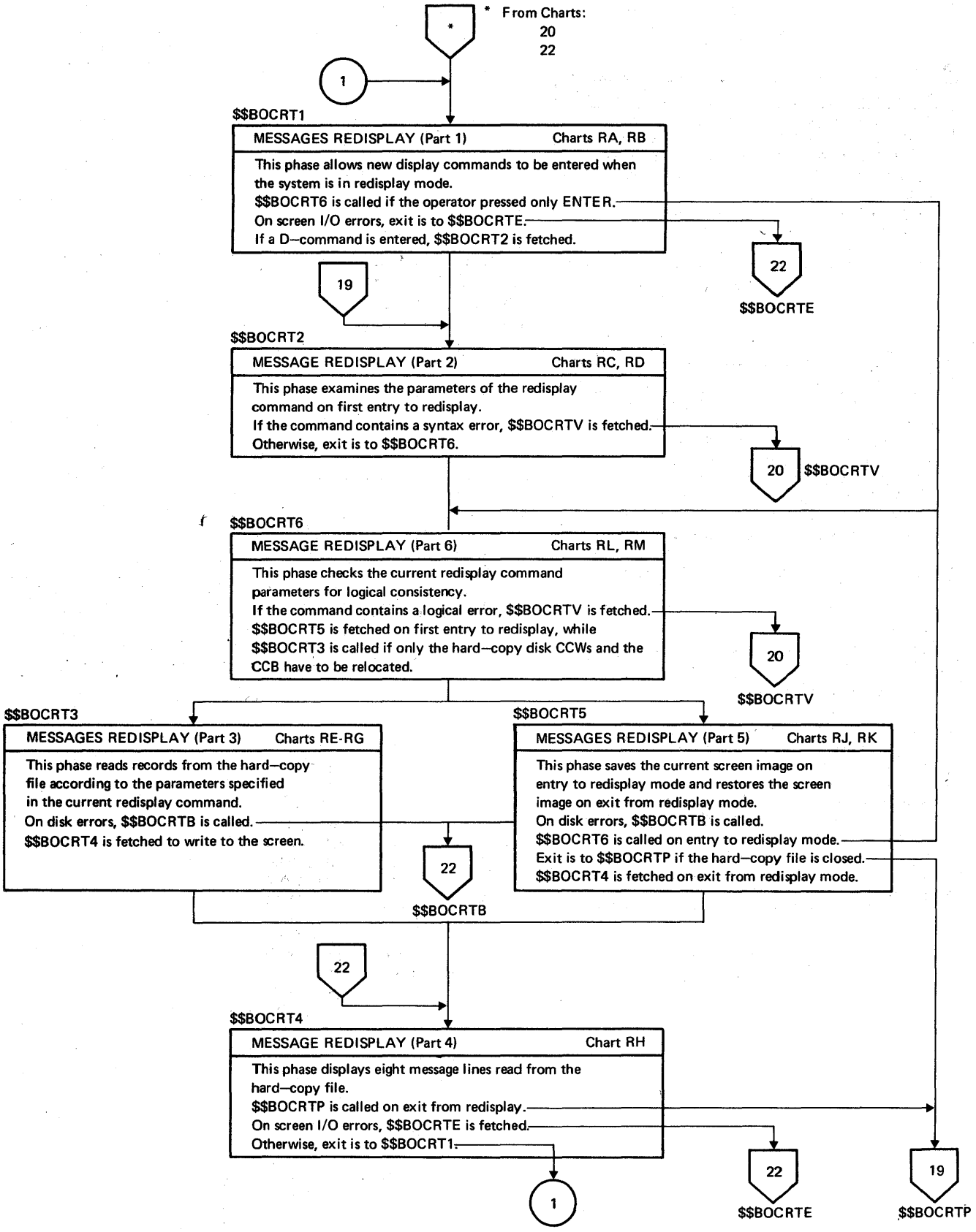
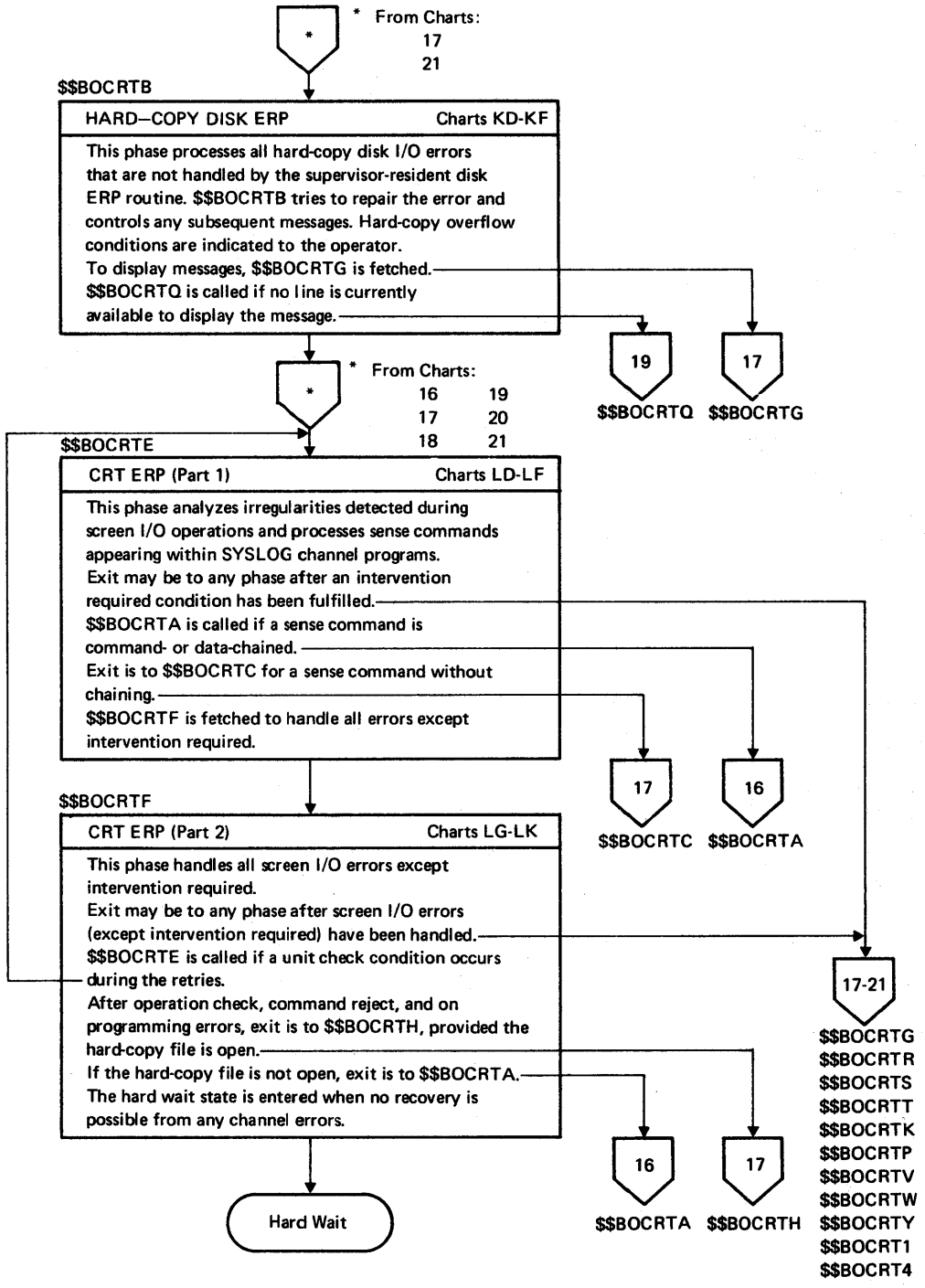


Chart 22. CRT ERP Phases



$$BOCRTA - CRT Root Phase

Any request for a display operator console (125D or 3277/3278) is passed directly from the Start-I/O routine in the supervisor to this phase to initiate the CRT transient routines. The CRT routines handle all requests on the display operator consoles.

Entry: From the Start-I/O routine in the supervisor when a 125D or 3277/3278 request is encountered; from phases $$BOCRTG, $$BOCRTE, $$BOCRTH, and $$BOCRTZ when execution of the channel program has been completed; from phases $$BOCRTC, $$BOCRTD, and $$BOCRTE when the channel program contains command chaining; from phase $$BOCRTK after an attention interrupt has been handled; from phase $$BOCRTU after the PRINTLOG utility has been completed; from phase $$BOCRTP after a screen management command has been completed; from phase $$BOCRTF when an irrecoverable screen I/O error has occurred and the hard-copy file is not open.

Exit: To phase $$BOCRTZ to analyze the channel program; to phase $$BOCRTU to update the wrap-around pointer of the hard-copy file; to phase $$BOCRTK to handle an attention request.

Method: On entry from the Start-I/O routine, the CRT device is flagged 'busy' in its PUB. The CRT task is activated and the SYSLOG-requesting CCB/IORB in the he channel queue is replaced by the CRTCCB. The address of the first (or only) CCW of the scheduled SYSLOG channel program is directly taken from the CAW and stored in a save area that can be referenced by other phases. If the CCW specifies wrap-around updating, phase $$BOCRTU is fetched. All other CCWs are passed to phase $$BOCRTZ for further analyzing. User CCWs are checked whether or not they start on a doubleword boundary before being passed to $$BOCRTZ.

On entry from the I/O interrupt handler, $$BOCRTA determines whether more commands are to be executed for the current channel program. If so, $$BOCRTZ is fetched after the new CCW address has been saved in CRTSAV.

If the channel program has been completed or if it is discontinued, the channel queue entry is restored to reflect the user status. In addition, the CSW set up by the processing phases is moved to storage location X'40', and the channel and unit address of the CRT device is moved to storage location X'BA'.

After the CRT task has been deactivated, control is returned to the I/O interrupt handler.

$$BOCRTB - Hard-Copy Disk ERP

Hard-copy disk I/O errors, which are not handled by the supervisor-resident disk ERP routine, are processed by this phase. $$BOCRTB tries to repair the error and controls any subsequent messages. In addition, hard-copy overflow conditions are indicated to the operator.

Entry: From the CRT transients that access the hard-copy disk file ($$BOCRTH, $$BOCRT3, and $$BOCRT5) and from $$BOCRTQ when the message area was full.

Exit: To phase $$BOCRTG to display messages; to phase $$BOCRTQ if no line is currently available.

Method: If less than two tracks are available for recording on the hard-copy file, phase $$BOCRTH sets a warning flag (see Figure 11) and, if subsequently the beginning of the file is overlaid, also the overflow flag. Since both indicators give rise to a warning message, they are set off together by $$BOCRTB in order to avoid repeated display of the warning message. Phase $$BOCRTH does not call $$BOCRTB but $$BOCRTA if the warning flag has already been set previously by phase $$BOCRTH.

If disk errors occur during redisplay, the messages already redisplayed on the screen are removed and the redisplay mode is set off. If the disk error is disastrous, CSW and sense information passed via the disk CCB/IORB and the field HCSNS in the save area CRTSAV are displayed in a second message line. Further recording on the hard-copy file is then prevented by setting off the open indicator (see Figure 11).

$$BOCRTB may issue the following messages:

```
0D20E HARD COPY FILE SHOULD BE PRINTED
0D25E HARD COPY FILE IN OVERLAY MODE
0D26E IRRECOVERABLE I/O ERROR.  HC DISCONTINUED
      CSW=xxxxxxx     SNS=x
0D29E INCORRECT LENGTH DURING I/O FOR HARD
      COPY
```

$$BOCRTC - Write Processor (Part 1 of 2)

This phase transfers all the read or write data from a SYSLOG channel program to the screen image buffer, analyzes the message type, and inserts the relevant information in the screen control table (SCT) entries. The format of an SCT entry is shown in Figure 10.

Entry: From phase $$BOCRTZ for Write, NOP, and Alarm CCWs; from phase $$BOCRTT for

Read CCWs; from $$BOCRTR if phase $$BOCRTA has indicated that a CCW chain is to be discontinued; from phase $$BOCRTQ when the message area was full; from phase $$BOCRTE and $$BOCRTD if the last CCW in a chain has been processed and the message type must be evaluated.

Exit: To phase $$BOCRTA when the current CCW indicates command chaining; to phase $$BOCRTG when the current CCW is not chained; to $$BOCRTD when the data in a single CCW or in data-chained CCWs exceeds the space available in one screen line; to $$BOCRTQ when the screen message area is full.

Method: Data of Read CCWs that are contained in the channel program passed to $$BOCRTC has already been entered in the screen entry area. However, since the data is part of the message it must be moved to the screen message area buffer like the write data. If a Read command had been cut off, that is, the operator pressed ENTER before the byte count had been exhausted, such CCWs are indicated by the read processor phases, which store the relevant information in the save area CRTSAV. The address of the last Read CCW with data is stored in CONTCCW and the byte count is stored in CONTRDSV. Residual count and CCW address are stored into the requestor's CCB/IORB. If a Read CCW has the SKIP flag set on, no data is moved into the message area.

A message header analysis is performed when the whole message text has been inserted into the screen image buffers. Each SYSLOG message that is issued after IPL by a program via an SVC 0 interrupt and that begins with a CCW other than a Sense CCW is preceded by the appropriate partition identifier (BG, F1, etc). This identifier is followed by a message header (see Figure 9 for the message format).

If the message begins with a Sense CCW, the header analysis resolves the message type as 'undefined'. This is also the case if the message is less than five characters long, since the type will then always be indicated by a blank, which is resolved as undefined.

Each message has a corresponding SCT entry. The format of an SCT entry is shown in Figure 10. The line status information in byte 0 is required for screen management.

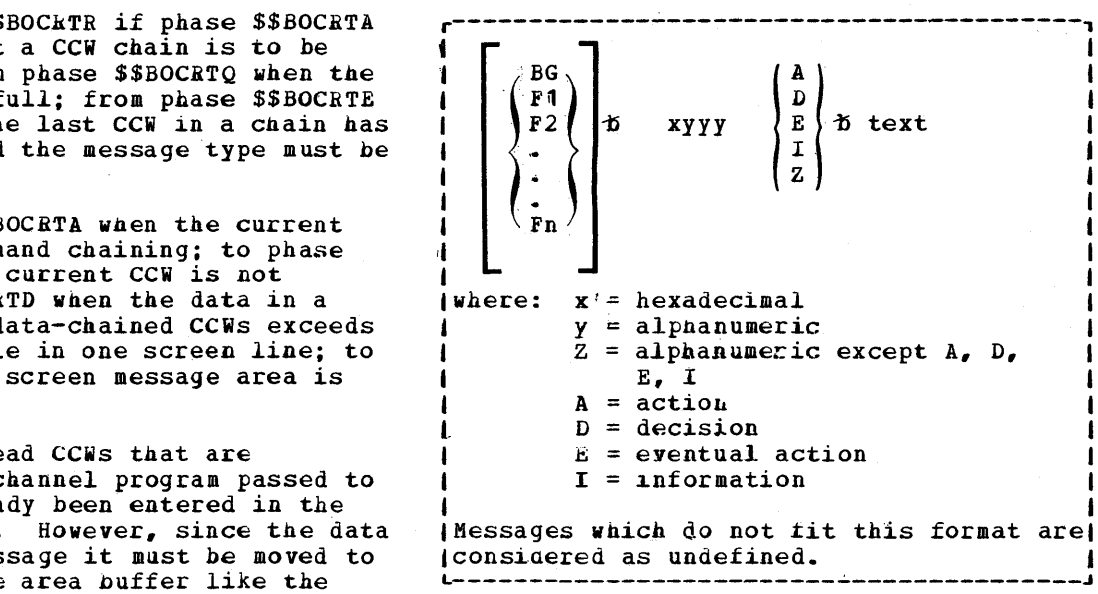


Figure 9. CRT Message Format

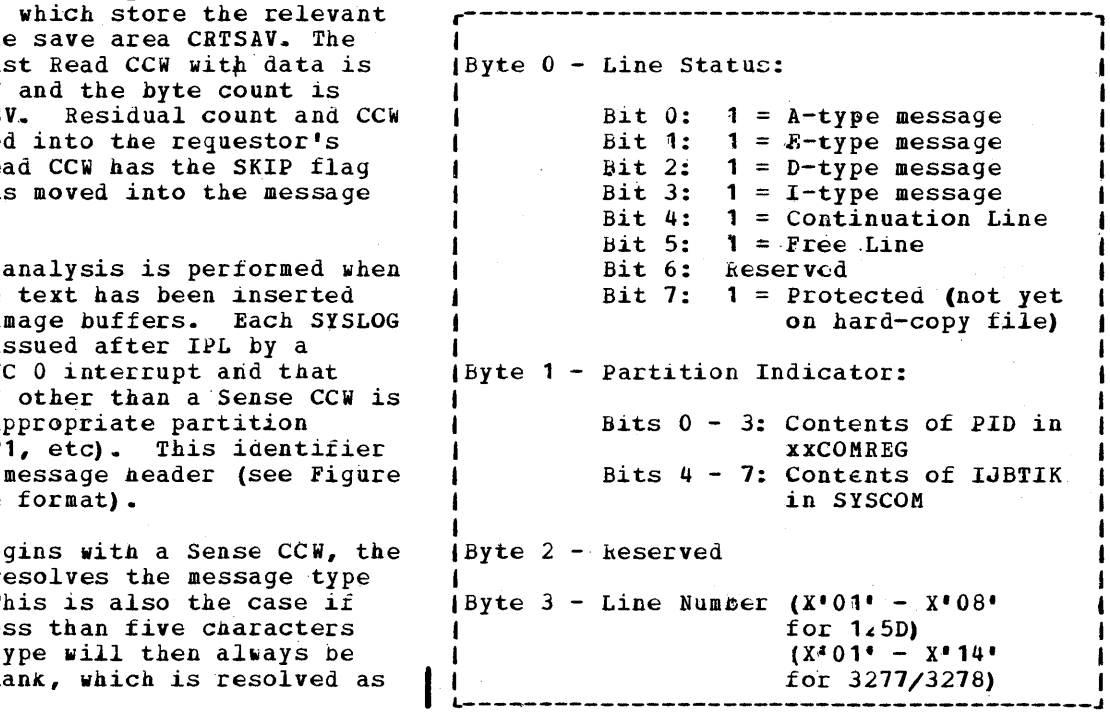


Figure 10. Screen Control Table (SCT) Entry

$$BOCRTD - Write Processor (Part 2 of 2)

This phase handles the line overflow caused by successive Write without carriage return CCWs, by data-chained CCWs, or by single CCWs with a byte count greater than the capacity of a single line.

Entry: From phase $$BOCRTC, which has detected the line overflow; from phase $$BOCRTQ when the message area was full.

Exit: To phase $$BOCRTA when the channel program is continued; to phase $$BOCRTC if no chaining has been encountered: to phase $$BOCRTQ if no line is currently available.

Method: For the 125D, each line of the message area offers 53 bytes for the message text; for the 3277/3278, each line offers 76 bytes for the message text. Since this is much less than the line capacity of a printer-keyboard console, messages must often be split up and distributed over several lines.

If a message does not fit into one line, a backward scan by 22 character positions is performed on such a message for a blank, comma, or full stop. The message is split up at the first of these separation characters, if any, and continued in position four (for 125D) or five (for 3277/3278) of the next line. Apart from this separation method, $$BOCRTD follows very closely the method of $$BOCRTC.

$$BOCRTE - CRT ERP (Part 1 of 2)

This phase analyzes irregularities detected during screen I/O operations and processes Sense commands appearing within SYSLOG channel programs.

Entry: From $$BOCRTZ for processing of regular Sense CCWs; from any other phase in which a screen I/O error occurred.

Exit: To phase $$BOCRTA on a Sense command with command or data chaining; to phase $$BOCRTC on a Sense command without chaining; to phase $$BOCRTF for handling of any error except 'intervention required'; to any other phase when an intervention required condition has been fullfilled.

Method: Sense commands from SYSLOG channel programs are executed as they occur within the channel program. CCW chaining, however, is broken off. The I/O routine works (disabled) with SIO and TIO instructions. Only one retry is performed on any error during the Sense operation. The I/O routine is also used to analyze screen I/O errors.

If intervention is required for the SYSLOG printer, message 0D33A is displayed in the instruction line (for the 125D). For the 3277/3278, the message 0D38A is moved to low core. The Post flag and the Disaster flag in the CRTCCB are set off and the phase waits for the 'ready' interrupt. If these interrupt conditions are error-free, the calling phase is fetched again. The name of this phase is taken from CRTNAM2 in CRTTAB. Before the phase is fetched, its registers are reloaded from the save area located at the end of $$BOCRTE, where they were stored on entry to phase $$BOCRTE.

$$BOCRTE may issue the following message:

```
0D33A INTERVENTION REQUIRED FOR SYSLOG
      PRINTER
```

$$BOCRTF - CRT ERP (Part 2 of 2)

This phase handles screen I/O errors except 'intervention required'.

Entry: From phase $$BOCRTE.

Exit: To phase $$BOCRTE if a unit check condition occurs during the retries; to $$BOCRTH after operation check, command reject, and on programming errors (program check or protection check), provided the hard-copy file is open (otherwise exit to $$BOCRTA).

The hard wait state is entered when no recovery is possible from channel chaining checks or channel interface checks. In that case, the code X'00000CCC' is entered in general register 11 and in bytes 0 through 3 in low real storage.

Method: The method is similar to the one used by phase $$BOCRTE. Three retries are carried out in the following cases: on channel errors, on equipment check, and on undetermined errors (that is, the unused sense bits are on). The retry counter is passed between phases $$BOCRTE and $$BOCRTF in field DOCEIND in the save area CRTSAV.

An equipment check may occur only for the SYSLOG printer. The hard-copy open flag is reset if recovery fails. Each message is displayed in the instruction line, because it is possible that $$BOCRTG cannot clear the entire message area, for instance, if an extremely long message is currently displayed.

If there is no recovery from operation check and undetermined sense indications, command reject, protection check, or program check, all the interphase conditions (indicated in DOCxIND in CRTSAV) are reset to the initial status. Thus, for example, a redisplay may be terminated.

$$BOCRTF may issue the following messages:

0D34E EQUIPMENT CHECK ON SYSLOG PRINTER
0D35E IRRECOVERABLE SCREEN I/O ERROR
0D36E UNDETERMINED SCREEN I/O ERROR. SNS=x

$$BOCRTG - Write Screen Message

This phase writes the screen image buffer to the screen after the data of the SYSLOG channel program has been completely transferred to the buffer.

Entry: From phases $$BOCRTB, $$BOCRTC, $$BOCRTE, and $$BOCRTF. The phase may also be fetched when an attention request is encountered and after an attention request condition has been fulfilled.

Exit: To phase $$BOCRTH for writing the hard-copy file; to phase $$BOCRTI for writing on the console printer; to phase $$BOCRTE on screen I/O errors; to phase $$BOCRTK when an attention request is encountered; to $$BOCRTA when the hard-copy file is not open or does not exist.

Method: Partition and subtask indicators are inserted into the relevant SCT entry (see Figure 10). The sound indicator DOCC02 in DOCxIND (in CRTSAV) and the message-type indicator are examined in order to determine whether or not the audible alarm should be generated. The screen message area buffer is scanned for messages belonging to the same partition and subtask as the current message. If there are any, the relevant lines are indicated as deletable by screen management. When a Read CCW is in the current channel program, the relevant message will also be indicated as deletable but the original message type will nevertheless be recorded on the hard-copy file. The latter does not apply for E-type messages.

The new message to be displayed is scanned for unprintable characters whose representation is smaller than X'40'. Such characters are replaced by blanks to avoid that screen control characters appear within the data. Before the screen image buffer is displayed, it is checked whether phase $$BOCRTB has indicated that intervention is required for the hard-copy disk device. If so, message 0D26A is displayed in the instruction line. Subsequently, a WAIT is issued on the disk CCB/IORB, which is passed in the CRTSAV CSECT.

The CRTIC macro is used for the screen write operation. Before the exit is taken, the line print control flags (for SYSLOG printer) in the Attribute Bytes in CRTSAV are set off. If the hard-copy file is not open, the line protection is reset (see Figure 10). Otherwise, the address of the first line to be printed on the hard-copy file, the number of lines to be printed, and the relevant SCT entry address are passed to phase $$BOCRTH in general registers 6, 2, and 13, respectively.

$$BOCRTG may issue the following message:
0D28A INTERVENTION REQD FOR HARD COPY DISK DEVICE

$$BOCRTH - Write Hard-Copy Records

This phase writes on the hard-copy disk file the new lines that have been displayed on the screen.

Entry: From phase $$BOCRTG; from phase $$BOCRTI if a console printer is supported; from phase $$BOCRTF after irrecoverable screen I/O errors; from phase $$BOCRTY when messages that exceed the message area capacity are to be handled.

Exit: To phase $$BOCRTA; to phase $$BOCRTB on hard-copy disk errors; for a message that exceeds the message area capacity to phase $$BOCRTQ when the maximum number of lines per message area has just been written onto the hard-copy file; to phase $$BOCRTU when overlay mode is entered in order to invalidate the wrap-around address in the header record.

Method: Starting with the message line in the screen image buffer pointed to by register 6 (and for as many lines as indicated in register 2) a hard-copy record is built in the hard-copy file buffer (for record format see Figure 12). Whenever the buffer is full, it is written to the actual disk address (ACTDSKA) of the hard-copy file by an SVC15-SVC7 sequence, then the actual disk address is updated. Incorrect length or irrecoverable I/O errors indicated in the CCB/IORB are passed to phase $$BOCRTB for recovery (see Figure 11, bits 6 and 7).

A test for remaining free space in the hard-copy file is then performed.

If there is only space left for two records, an indicator (see Figure 11, bit 2) is set which causes a warning message to be issued by phase $$BOCRTB.

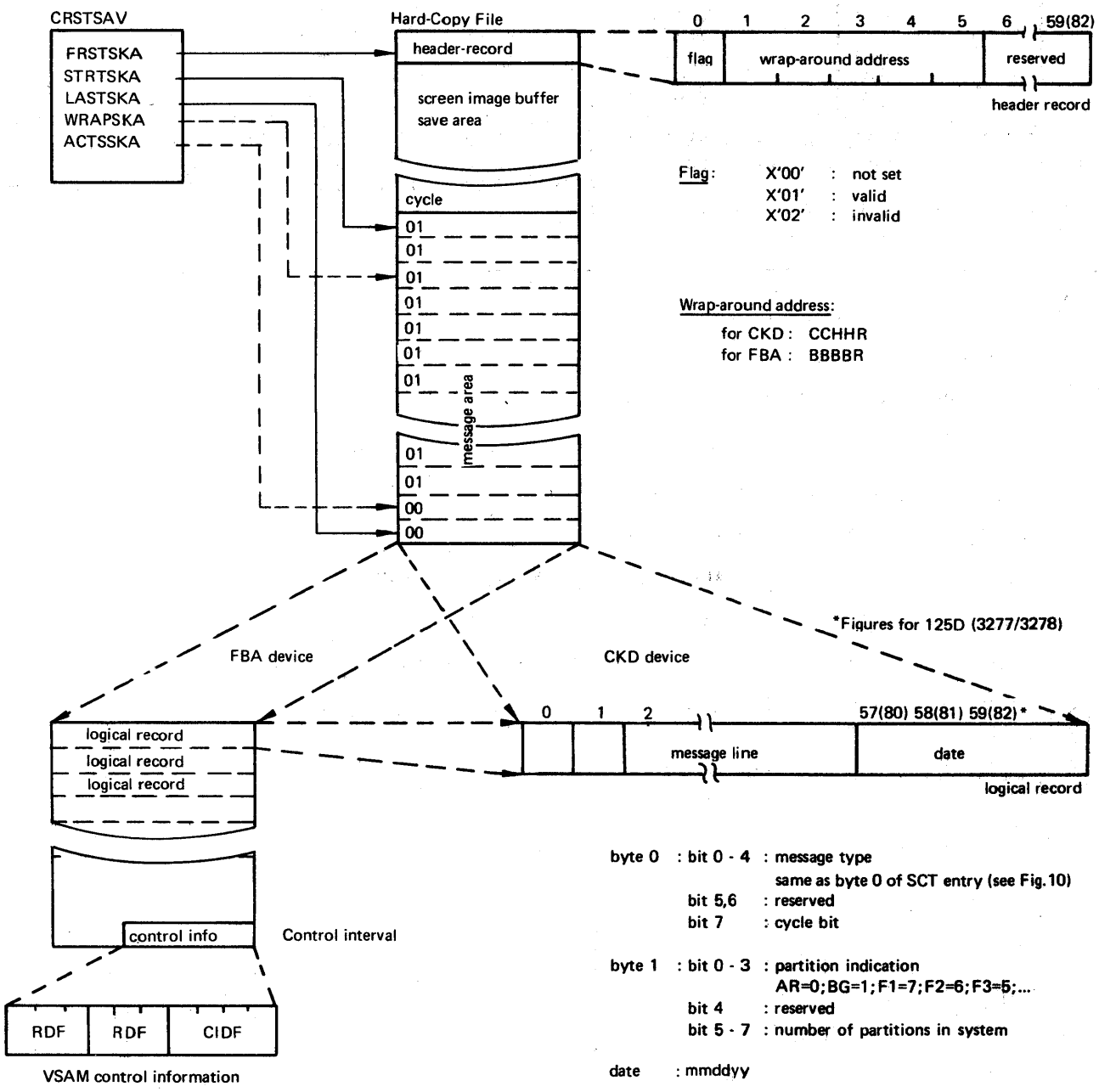
If there is no more space left, i.e. the wrap-around point is reached (ACTDSKA = WRAPSKA), overlay mode is entered (see Figure 11, bit 1) and $$BOCRTU is called to invalidate the current wrap-around address in the header record.

If overlay mode is active (see Figure 11, bit 1), the wrap-around point (WRAPSKA) is made equal to the actual disk address.

If the physical end of the hard-copy file extent is reached (ACTDSKA = LASTSKA), then the actual disk address is reset to the start of the message area (ACTDSKA = STRTSKA) and the cycle bit is inverted (see Figure 12).

```
CRTFLGHC  Hard Copy Control Flags:

          CRTHOPN  - Bit 0:  1 =Hard-copy file is open

          CRTHCOVR - Bit 1:  1 =Hard-copy file is in overlay mode. This bit is reset
                               together with bit 2 by $$BOCRTB after the
                               corresponding messages have been issued.

          CRTHCWRN - Bit 2:  1 =Warning message should be issued

          CRTIPL   - Bit 3:  1 =IPL switch

          HFTOPEN  - Bit 4:  1 =Hard-copy file must be created

          HFEQUNO  - Bit 5:  1 =Hard-copy file not in use

          HCERR    - Bit 6:  1 =Hard-copy file has irrecoverable error

          HCINCL   - Bit 7:  1 =Incorrect length during recording
```

Figure 11. Hard-Copy Control Flags



Address	Description	CKD format	FBA format
FRSTSKA	Start of hard-copy file	CCHH	BBBB
LASTSKA	End of hard-copy file	CCHHR	BBBBR
STRTSKA	Start of message area	CCHHR	BBBBR
WRAPSKA	Wrap-around address	CCHHR	BBBBR
ACTDSKA	Actual address	CCHHR	BBBBR

Figure 12. Hard-Copy File Format (Part 1 of 2)

Notes:

Messages displayed on the screen of a display operator console are collected in the hard-copy file, which occupies one extent on the DASD device allocated to SYSREC.

Logical Format

The hard-copy file contains a header record, the screen image buffer save area, and the message area.

The header record is the first record in the file. It contains the wrap-around address, which is the disk address of the first record in the message area which was not printed by the PRINTLOG utility.
The header record can be:

'not set', then wrap-around address = STRTSKA
'valid', then the wrap-around address is valid
'invalid', then the wrap-around point was overlayed and wrap-around address = ACTDSKA

The screen image buffer save area starts at the second logical record. Its length depends on the screen size: for 125D it consists of 8 records, for 3277/3278 it consists of 20 records. The current contents of the screen image buffer are saved in it during redisplay processing.

The message area starts at STRTSKA and ends at LASTSKA. Records containing the message are written to it in wrap-around mode. Each logical record contains a cycle bit which is inverted every time a new cycle is started, i.e. whenever writing starts again at STRTSKA. The point where the cycle bit changes from one record to the next defines the actual address ACTDSKA. New records are added to this point.

Physical Format

The hard-copy file starts at FRSTSKA and ends at LASTSKA. The logical records have a fixed length which depends on the device type of SYSLOG: it is 60 bytes for 125D and 83 bytes for 3277/3278.

For hard-copy files residing on CKD devices the physical record equals a logical record. The last record is an EOF-record with a zero data length. Minimum size is 10 tracks.

For hard-copy files residing on FBA devices a number of logical records is blocked in a VSAM-format control interval. The CI length equals the FBA block size. The last CI of the file contains a SEOF (software end-of-file). Minimum size is 50 blocks.

Figure 12. Hard-Copy File Format (Part 2 of 2)

$$BOCRTI - Write on 3284/3286/3287/3288 SYSLOG Printer

This phase writes all lines from the message area to the console printer if the appropriate support is generated.

Entry: From $$BOCRTG or $$BOCRTY after the message area is written to the screen.

Exit: To $$BOCRTA if no hard-copy support and call from $$BOCRTG; to $$BOCRTQ if no hard-copy support and call from $$BOCRTY; to $$BOCRTH if hard-copy support exists.

Method: The message area is scanned for internal protected lines. All internal protected lines are moved into the output area of the console printer. The contents are scanned for trailing blanks and the byte count for the print operation is reduced accordingly. When the print operation (SVC15, physical addressing) is not successful, a message is displayed in the instruction line, which must be answered by the operator in one of the following ways: ignore printing of the current line, or retry printing of the current line, or cancel the printer support.

$$BOCRTK - Command Entry Processing

This phase analyzes operator commands.

Entry: From phase $$BOCRTA and $$BOCRTG to handle an attention request; from phase $$BOCRTW during non-automatic line delection; from phases $$BOCRTR and $$BOCRTS when operator commands have been encountered in the input area.

Exit: To phases $$BOCRTL through $$BOCRTP and $$BOCRT2 on screen management commands; to phase $$BOCRTA on system commands; to phase $$BOCRTE on screen I/O errors; to any phase which was interrupted previously by an operator request.

Method: The input area is read and the data checked. The exit taken depends on the result of the test.

Pressing the CANCEL key has no effect at all as the exit interface routine $$BOCRTP is fetched. The cursor in the message area indicates a deletion request. System commands (and the Request key) are passed to the Attention routine.

Valid redisplay commands must begin with 'D' followed by one or more blanks only or by several blanks and an 'L'. K-commands are unique in DOS/VSE. A check is made for 'K' only, 'K' followed by blank(s), and for 'K' accompanied by subparameters: 'E', 'S', and 'D'. The full command is converted to uppercase characters.

$$BOCRTL - Deletion by K-Command

This phase handles K-commands with the subparameter E except K E,N.

Entry: From phase $$BOCRTK; from phase $$BOCRTX during message deletion in conversational mode.

Exit: To phase $$BOCRTQ; to phase $$BOCRTV on erroneous commands.

Method: The syntax of the command is checked and an error is indicated when the command is incomplete or has invalid parameters.

The following commands are correct:

```
K
K  E
K  E,SEG
K  E,n   where  1≤n≤8 (for 125D), or
                1≤n≤20 (for 3277/3278)
K  E,m,n where  1≤m,n≤8, and m≤n
                (for 125D), or
                1≤m,n≤20, and m≤n
                (for 3277/3278).
```

All these commands must be followed by at least one blank. The SEG parameter corresponds to the value specified on a specification K-command, that is K E,SEG is equivalent to K E,1,n. K E,n is equivalent to K E,m,n.

If the line numbers specified in the command do not point exactly to the beginning or end of the message, the specified deletion range is reduced in order to include only the beginning of a message to be deleted.

The final value of the line range to be deleted is passed to phase $$BOCRTQ in field SEGVAL2, which is contained in the save area CRTSAV.

$$BOCRTM - Deletion by Cursor

This phase determines from the cursor address the corresponding range of message lines that must be deleted.

Entry: From phase $$BOCRTK.

Exit: To phase $$BOCRTP on invalid cursor operation; to phase $$BOCRTQ on valid cursor operation.

Method: The address of the cursor is compared with the addresses of the different screen lines. If the cursor address lies beyond the message area, the error message 0D02I is set up, which is later displayed by $$BOCRTP. The message is not set up if the cursor is set to the first position of the input area.

If the cursor address can be assigned a message line address, it is checked whether the line contains the end of a message. If not, the line range must be redefined.

The range of lines to be deleted is passed to phase $$BOCRTQ in field SEGVAL2, which is contained in the save area CRTSAV.

This phase may set up the following message:

0D02A ILLEGAL CURSOR ORATION

$$BOCRTN - Screen Management Mode Specification

This phase processes the specification K-command.

Entry: From phase $$BOCRTK.

Exit: To phase $$BOCRTP; to phase $$BOCRTV in case of error.

Method: The commands processed must be of the form

K S,REF
K S,parm where parm is at least one of the parameters:

DEL={Y|N},CON={Y|N},ALM={Y|N} SEG=n (where 1≤n≤8 for 125D, 1≤n≤20 for 3277/3278)

PRT={Y|N}

The command K S,PRT,REF results in the current DEL,CON,ALM,SEG,PRT parameter values displayed in the instruction line (by phase $$BOCRTP).

The command K S,parm updates the screen management parameters. These parameters are represented in CRTSAV by the fields

DOCNIND Bit 0: 0 = DEL=N
1 = DEL=Y

Bit 1: 0 = CON=N
1 = CON=Y

Bit 2: 0 = ALM=N
1 = ALM=Y

Bit 4: 0 = PRT=N
1 = PRT=Y

SEGVAL1+1 = SEG value

$$BOCRTQ - Line Number Display

This phase processes the display K-command.

Entry: From phase $$BOCRTK, from phase $$BOCRTQ for display/deletion of line numbers.

Exit: To phase $$BOCRTX when the system operates in conversational mode (CON=Y); to phase $$BOCRTP when the system is not in conversational mode; to phase $$BOCRTV in case of error.

Method: In general, this phase is fetched for processing of the commands

K D

K D,N

K D,N,HOLD.

In all cases, the line numbers are inserted into the appropriate buffer locations, but K D,N,HOLD causes a flag to be set (DOC008) to prevent the numbers from disappearing when the next command is entered.

Whether or not the phase that handles the K E,N command is fetched, is indicated by flag DOC001. This is the special case, not included in phase $$BOCRTL, where the subparameter 'L' is operated on. DOC004 indicates that the phase is fetched during processing of a deletion command in conversational mode.

DOC002 indicates that the line numbers must be deleted. This takes place during message deletion in conversational mode.

$$BOCRTP - Exit Interface

This phase establishes the interface between the screen management phases and the supervisor.

Entry: From phases $$BOCRTO, $$BOCRTM, $$BOCRTN, $$BOCRTV, $$BOCRT4, $$BOCRT5; from phases $$BOCRTE and $$BOCRTF after recovery from screen I/O errors; from phase $$BOCRTK when a command is canceled; from phase $$BOCRTQ after line deletion by command; from $$BOCRTX when a deletion request has been canceled during conversational mode.

Exit: To phase $$BOCRTA when the next Attention interrupt should be handled by the supervisor; to any interrupted phase whose name is passed in CRTNAM1 as indicated in CRTQPD (in the table CRTTAB); to phase $$BOCRTE on screen I/O errors.

Method: The line numbers are deleted, if the HOLD parameter is not active. The instruction line and entry area are blanked. When entry was from phase $$BOCRTV, only the entry area is blanked, while the message area is displayed, that is, the error message from phase $$BOCRTV remains on the screen.

$$BOCRTQ - Deletion of Message Lines

This phase deletes message lines.

Entry: From phases $$BOCRTB through $$BOCRTD during channel program interpretation; from phase $$BOCRTX during deletion in conversational mode; from $$BOCRTL for deletion of messages by K-command; from $$BOCRTM for deletion of messages according to cursor positioning; from $$BOCRTY during processing of extremely long messages.

Exit: To the calling phase; to phase $$BOCRTP on exit from a deletion command; to phase $$BOCRTW during non-automatic deletion mode (DEL=N) and during handling of messages that exceed the screen capacity; to phase $$BOCRTO for displaying/erasing line numbers during message deletion in conversational mode.

Method: The number of lines representing the range of deletion is passed in field SEGVAL2, which is contained in the save area CRTSAV. The lines following the specified range are checked to determine whether they belong to the message(s) to be deleted.

A-, D-, and E-type messages within the specified range are not deleted. Messages of this type are deleted only when directly pointed to by a command or by the cursor. If the cursor is set to line n, not only the message pointed to is deleted (irrespective of its type), but also all deletable messages contained in lines 0 through n. The screen image buffer may also contain lines protected against deletion, that is, if they have neither been displayed on the screen nor recorded on the hard-copy file.

$$BOCRTR - Read Processor (Part 1 of 3)

This phase is the root phase of the read processor and analyzes the read commands or reads an operator command.

Entry: From phase $$BOCRTA; from phase $$BOCRTS when an operator command is to be read; from phase $$BOCRTT when processing of the Read CCW is to be continued; from phase $$BOCRTE and $$BOCRTF after successful recovery of a screen I/O error.

Exit: To phase $$BOCRTS; to phase $$BOCRTK for handling of operator commands; to phase $$BOCRTC if a Read CCW is preceded by an erroneous Write CCW; to phase $$BOCRTE on screen I/O errors.

Method: On first entry into the phase, the input area is read to determine whether an operator command is dealt with or a SYSLOG channel program is to be interpreted. If the phase is fetched again by phase $$BOCRTT, a Read channel program is processed and the data may not fit in the entry area.

If a Read CCW is preceded by the CCW sequence 'Write without carriage return - TIC', the data of the Write CCW up to 80 characters is moved into the entry area before the operator is allowed to enter data.

If the operator wants to enter a command while a SYSLOG Read CCW is handled, he can do so by pressing the REQUEST key, but the message 'READ IS WAITING' appears in the warning line. The operator can then press the Cancel key to cancel the command just entered and allow the SYSLOG channel program to be continued.

$$BOCRTR may issue the following messages:

0D09D READ IS WAITING. CANCEL OR CONTINUE

$$BOCRTS - Read Processor (Part 2 or 3)

This phase allows the operator to enter data corresponding to a Read CCW.

Entry: From phase $$BOCRTR; from phases $$BOCRTE and $$BOCRTF after successful recovery of a screen I/O error.

Exit: To phase $$BOCRTT; to phase $$BOCRTK when a command has been entered; to phase $$BOCRTE on screen I/O errors; to phase $$BOCRTR if the Read CCW is canceled.

Method: The message 'ENTER RESPONSE' or 'ENTER COMMAND' is displayed in the instruction line and a read of the input area is started. The latter allows as much data to be entered as specified in a corresponding SYSLOG channel program. However, if the Request key is pressed, a command may be entered instead of data.

Pressing the Cancel key, whether accompanied by entered data or not, allows the data to be reentered.

$$BOCRTS may issue the following messages:

0D08A ENTER COMMAND

0D07D ENTER RESPONSE

$$BOCRTT - Read Processor (Part 3 of 3)

Moves the data (corresponding to the SYSLOG channel program) into the requestor's input area.

Entry: From phase $$BOCRTS; from phases $$BOCRTE and $$BOCRTF after successful recovery of a screen I/O error.

Exit: To phase $$BOCRTR for continuation of the read channel program; to phase $$BOCRTC when the SYSLOG Read CCWs are handled; to phase $$BOCRTE if an error has occurred during screen I/O.

Method: The read byte count is reduced by the residual count of the previous read operation in phase $$BOCRTS. If the number of bytes is less than the length of the entry area and the current CCW is not data-chained, the data just read is moved to the requestor's input area, the screen entry area is cleared, and control is given to $$BOCRTC.

If the byte count exceeds the entry area length, the read operation may have to be repeated. The data is moved to the requestor's input area and phase $$BOCRTR is fetched again. The data is not moved if the corresponding CCW has the skip flag set on.

The fact that a byte count exceeds the amount of data entered and that the entry area is not yet exhausted indicates that the read channel program has been finished.

$$BOCRTU - Update Hard-Copy File Wrap-Around Address

This phase updates the wrap-around address in the header record, or invalidates the wrap-around address in the header record, or forces writing of the current hard-copy buffer.

Entry: The Printlog utility or IJBJC7 issues an SVC0 instruction accompanied by a special CCW. This activates $$BOCRTA, which fetches $$BOCRTU.

$$BOCRTH fetches $$BOCRTU if overlay mode is entered.

Exit: If overlay mode is active to $$BOCRTB, else to $$BOCRTA.

Method: For the update function, the wrap-around address WRAPSKA is written to the header record (see Figure 12). For the invalidate function, the header record is invalidated (see Figure 12). For the force I/O function, the current hard-copy buffer is written to the hard-copy file.

$$BOCRTV - Command Error Message Writer

This phase displays error messages related to invalid K- and D-commands.

Entry: From phase $$BOCRTL, $$BOCRTN, $$BOCRTO, $$BOCRT2, and $$BOCRT6; from phases $$BOCRTE and $$BOCRTF after recovery from screen I/O errors.

Exit: To phase $$BOCRTP; to phase $$BOCRT1 when the system is in redisplay mode; to phase $$BOCRTE in case of screen I/O errors.

Method: The cursor is set to the position passed in field CRTPOS1 in CRTSAV by the different phases. This indicates to the operator the invalid character or parameter. Lines 1 through the instruction line of the screen are displayed.

$$BOCRTV may issue the following messages:

0D01A CONTROL COMMAND ERROR
0D04A DELETION REQUEST INCONSISTENT.
INVALID RANGE
0D30A INVALID DISPLAY COMMAND

$$BOCRTW - Warning During Non-Automatic Deletion Mode

This phase displays message 0D05A:

1. When the system operates in the non-automatic deletion mode (DEL=N) and a message is issued but cannot be displayed because the screen is full.

2. When a message is issued but cannot be displayed because the screen is full with non-deletable messages.

Entry: From phase $$BOCRTQ; from phases $$BOCRTE and $$BOCRTF after recovery from screen errors.

Exit: To phase $$BOCRTK to analyze the operator's action; to phase $$BOCRTY when a message should be displayed that exceeds the message area capacity; to phase $$BOCRTE on screen I/O errors.

Method: When the message area buffer contains lines that belong to the same message and these lines have not yet been displayed on the screen or recorded on the hard-copy file, phase $$BOCRTY is fetched. Otherwise, 'MESSAGE WAITING' is displayed in the warning line and control is passed to phase $$BOCRTK upon the response of the operator. The message area is displayed to show the operator that it is completely full.

$$BOCRTW issues the following message:

0D05A MESSAGE WAITING

$$BOCRTX - Verification of Operator Commands during Message Deletion

This phase displays a message deletion request given by the operator in the form of a K-command, and waits for the response of the operator.

Entry: From phase $$BOCRTO (display numbers preceding the lines); from phases $$BOCRTE and $$BOCRTF after recovery from screen I/O errors.

Exit: To phase $$BOCRTP if the operator cancels the deletion; to phase $$BOCRTL if the operator does not change the deletion parameters; to phase $$BOCRTM if the operator sets the cursor; to phase $$BOCRTK if the operator changes the deletion parameters or enters a command other than a deletion command.

Method: The range of deletion passed in field SEGVAL2 in CRTSAV is displayed in a corresponding deletion command in the entry area together with the message 'DELETION REQUESTED', which is displayed in the instruction line. The system waits for the operator response. The answer is read and analyzed. If only the Request Key is pressed, the answer is considered invalid. If the answer modifies the deletion range or the type of command, then control is passed to $$BOCRTK.

$$BOCRTX issues the following message:

0D03A DELETION REQUESTED

$$BOCRTY - Handling of Messages Longer than Eight Lines

This phase handles messages that are longer than eight lines. The support is for both DEL=N and DEL=Y.

Entry: From phase $$BOCRTW; from phases $$BOCRTE and $$BOCRTF on successful recovery from screen I/O errors.

Exit: To phase $$BOCRTH when the hard-copy file is open; to phase $$BOCRTI if a console printer is supported; to phase $$BOCRTQ when the hard-copy file is not open; to phase $$BOCRTE on any screen error.

Method: The message buffer is displayed on the screen and, at the same time, the message lines are printed on the SYSLOG printer, if available. The latter is controlled by the print-line control flag in the Attribute Byte in CRTSAV. After the lines have been displayed and printed, the print flag in the corresponding Attribute Bytes is reset. If the system is in non-automatic deletion mode (DEL=N), the message 'MESSAGE TOO LONG CONTINUE BY ENTER' is displayed and the system waits for the operator response. Any response other than pressing ENTER is ignored. Depending on whether or not the hard-copy file is open, phase $$BOCRTH is fetched (file is open) or the internal line-protection flag (see Figure 10) is reset and phase $$BOCRTQ is given control. If a console printer is supported, $$BOCRTI is fetched.

When DEL=Y is active, the message 'CONTINUE BY ENTER' is displayed in the instruction line, the message 'MESSAGE TOO LONG' is displayed in the warning line, and the operator's response is waited for. Then the procedure follows the method for DEL=N above, but the number of lines to be printed is taken from the value of the SEG parameter.

$$BOCRTY may issue the following message:

0D06A MESSAGE TOO LONG
CONTINUE BY ENTER (in warning line)

$$BOCRTZ - Channel Program Analyzer

This phase diagnoses and checks any user-supplied CCW or CCW chain for validity. The status of the diagnostic run is reflected in a CSW that is returned to the user. In addition, provisions are made to discontinue a CCW chain at the point where an error is encountered.

Entry: From phase $$BOCRTA to analyze the channel program.

Exit: To phase $$BOCRTA if the next CCW to be processed is in error, to phase $$BOCRTC if the next CCW to be processed is a Write, NOP, or Alarm CCW; to phase $$BOCRTE if the next CCW to be processed is a Sense CCW; to phase $$BOCRTG if an error is encountered in a CCW which is not the first CCW in a chain of CCWs; to phase $$BOCRTR if the next CCW to be processed is a Read CCW or if one or more 'Write without carriage return' CCWs are followed by a Read CCW. Note that TIC commands contained within a chain of CCWs have no effect on the exit sequence described above.

Method: On entry from phase $$BOCRTA, the address of the next CCW to be processed is loaded and the CCW itself, as well as any CCW that is data-chained to the preceding CCW, is checked for validity. The following checks are performed:

1. Check for valid OP-code (omitted in case of data-chained CCWs).
2. Check for valid byte count (omitted in case of a TIC command).
3. Check for unused bit setting (omitted in case of a TIC command).
4. Check for valid or protected I/O area address (omitted if the CCW is not a user-supplied CCW, if it is a NOP or Audible Alarm CCW, or if the Skip bit is set to 1).

The current status of the checks performed is registered in a CSW that is returned to the user. This CSW will contain the correct residual count, the correct status information, and the correct CCW address + 8. However, before the CSW is returned to the user, it may be modified by the processing phases if these phases detect an error or if an interrupt condition occurs during processing.

After the various checks have been completed, phase $$BOCRTZ fetches the proper processing phase to handle all CCWs that are considered valid by $$BOCRTZ. If phase $$BOCRTZ finds that a CCW chain is to be discontinued, it also passes the address of the first invalid CCW in the chain to the processing phases.

$$BOCRT1 - Message Redisplay (Part 1 of 6)

This phase allows new display commands to be entered when the system is in redisplay mode.

Entry: From phase $$BOCRT4; from phases $$BOCRTE and $$BOCRTF after recovery from screen I/O errors; from phase $$BOCRTV after a redisplay command error.

Exit: To phase $$BOCRT6 if only the ENTER key is pressed; to phase $$BOCRT2 if a D-command is entered; to phase $$BOCRTE on any screen I/O error.

Method: Any pending request indicated in CRTFLG2 in CRTTAB or a second entry in the channel queue causes 'MESSAGE WAITING' to be displayed in the warning line. The phase waits for an Attention interrupt provided such an interrupt is not already pending. The entry area is read and the input checked. A CANCEL encountered in the input area causes the process just described to be repeated.

Pressing the Request key is an invalid operation during redisplay and causes the message 'REQUEST IGNORED' to be issued. The message 'INVALID DISPLAY COMMAND' appears if any data that does not begin with a 'D' has been entered.

$$BOCRT1 may issue the following messages:

0D30A INVALID DISPLAY COMMAND

0D97A REQUEST IGNORED

0D05A MESSAGE WAITING

$$BOCRT2 - Message Redisplay (Part 2 of 6)

This phase examines the parameters of the redisplay command on first entry to redisplay.

Entry: From phase $$BOCRTK on entry to redisplay; from phase $$BOCRT1 during redisplay.

Exit: To phase $$BOCRTV when the command contains a syntax error; otherwise, to phase $$BOCRT6.

Method: The whole command is transformed into uppercase characters. The command is scanned and an indicator is set for each parameter which is essential for redisplay (for instance, partition parameter or forward or backward parameter).

If the syntax of the command is violated, exit is to phase $$BOCRTV. This happens in the following cases:

- First character is no 'D'.
- Second character, if any, is not blank.
- Second character is blank but the next non-blank character is neither 'L' nor 'E'.
- An 'E' is immediately followed by a non-blank character.
- An 'L' is not immediately followed by a blank or a comma.
- One of the parameters 'ALL', 'BG', 'AR', 'F1', ..., 'B', 'F' is not followed by a comma or blank.
- A parameter that follows one of the preceding parameters is neither 'R' nor a decimal digit, or is a decimal number with more than three digits.

$$BOCRT3 - Message Redisplay (Part 3 of 6)

This phase reads from the hard-copy file according to the parameters specified in the current redisplay command.

Entry: From phase $$BOCRT6.

Exit: To phase $$BOCRT4 to write to the screen; to phase $$BOCRTB on a disk error.

Method: Using the current redisplay command parameters, which are passed in flags and counters, the disk channel program is updated to read the hard-copy records to be displayed next.

$$BOCRT4 - Message Redisplay (Part 4 of 6)

This phase displays eight message lines read from the hard-copy file.

Entry: From phase $$BOCRT3 to display the screen during redisplay; from phases $$BOCRTE and $$BOCRTF after recovery from screen I/O errors; from phase $$BOCRT5 on exit from redisplay.

Exit: To phase $$BOCRTP on exit from redisplay; to phase $$BOCRT1; to phase $$BOCRTE on any screen I/O error.

Method: When redisplay should be finished according to the current command, the screen is restored as it was before the redisplay mode was entered.

'*** END OF FILE ***' or '*** BEGIN OF FILE ***' is indicated on the screen when the disk address ACTDSKA (where the next record is to be written by phase $$BOCRTH) has been reached by redisplaying forward or backward. 'NO MORE MESSAGES ON HC FILE' appears if attention routine messages or all messages are redisplayed and no more messages are available.

In any case, the instruction line is written to indicate to the operator:

- That the system is in redisplay mode.
- The direction of redisplay (forward or backward.)
- The number of lines between the current image and the (currently) last record on the file.

If the partition is specified in the redisplay command, it is also indicated.

$$BOCRT5 - Message Redisplay (Part 5 of 6)

This phase saves the current screen image on entry to redisplay mode and restores the screen image on exit from redisplay mode.

Entry: From phase $$BOCRT6.

Exit: To phase $$BOCRT6 on entry to redisplay mode; to phase $$BOCRTP if the hard-copy file is not open; to phase $$BOCRT4 on exit from redisplay mode; to phase $$BOCRTB on hard-copy disk errors.

Method: The hard-copy disk channel program and CCB/IOAB are relocated. The current screen image area is saved in (or restored from) records 1 through 8 (for the 125D) or 1 through 20 (for the 3277/3278) of the screen image buffer in the hard-copy file (see Figure 12). If an irrecoverable disk I/O error occurs, phase $$BOCRTB finishes redisplay and the current screen image may be lost.

$$BOCRT5 may issue the following message:

0D37I HC FILE NOT OPENED, REDISPLAY NOT POSSIBLE

$$BOCRT6 - Message Redisplay (Part 6 of 6)

This phase checks the current redisplay command parameters for logical consistency.

Entry: From phase $$BOCRT2 on entry to redisplay; from phase $$BOCRT5 after saving the current screen image; from phase $$BOCRT1 if only ENTER has been pressed during redisplay.

Exit: To phase $$BOCRTV if any logical error in the command is detected; to phase $$BOCRT3 if phase $$BOCRT6 is fetched only for relocating the hard-copy disk CCWs and CCB/IORB; to phase $$BOCRT5 on the first entry to redisplay.

Method: The partition parameter is the only essential subparameter allowed on entry to redisplay. If the partition parameter is specified in the first redisplay command, the line counter used during redisplay is set to zero and phase $$BOCRT5 is fetched. The same is true if 'ALL' is specified explicitly or applied by default.

Checkpoint is a record of the status of a problem program taken at desired intervals. Restart is the resumption in the execution of the program from one of the checked points, rather than from the beginning, if processing terminates for any reason before the normal end-of-job.

When a problem program is expected to run for an extended period of time, provision should be made for taking checkpoint information periodically during the run. This information describes the status of the job and the system (storage, I/O status, registers) at the time the records are written. Thus, restarting at the last checkpoint position is possible if processing is terminated for any reason before the normal end-of-job. For example, processing terminates if a malfunction, such as a power failure, occurs.

CHECKPOINT

A CHKPT macro must be issued to write a checkpoint. The CHKPT macro expansion fetches the checkpoint transient program $$BCHKPT to write a checkpoint on a specified tape unit, or $$BCHKPD to write the checkpoint on a disk. Any number of checkpoints (up to decimal 9999) can be taken on the same unit. A number in the checkpoint header label identifies a particular checkpoint.

If the checkpoint is taken on a separate magnetic tape file with standard labels (defined by the user with a DTFPH macro), the labels must either be checked by an OPEN macro or bypassed by an MTC command before the first checkpoint is taken. If the checkpoint is taken on a disk file, the user must define (by a DTFPH macro with MOUNTED=SINGLE, TYPEFLE=OUTPUT) and open the file before a CHKPT macro is issued. A CISIZE parameter defined with the DTFPH is ignored by the checkpoint/restart routines.

Checkpoint does not save the following:

- The SVA and the system GETVIS area. If a checkpointed program uses phases in the SVA or occupies space in the system GETVIS area, the user must make sure that the same locations are occupied at restart time.
- The floating-point registers.
- Any linkages to the checkpointed program set up by the STXIT or SETPFA macros.
- Any XECBs defined by the checkpointed program.
- Any timer values set by the SETIME macro.
- The program mask in the checkpointed program's PSW.

If required, this information must be defined at restart time by a user restart routine.

Note: A checkpoint must not be taken if any gating mechanism is in use by the checkpointed program (for example, USE/RELEASE).

CHKPT Macro

The CHKPT macro instruction has the following format:

```
Name     Operation   Operands

[name]   CHKPT       SYSnnn,{rest-addr|(r1)}
                     [,end-address|,(r2)]
                     [,tpointer|,(r3)]
                     [,dpointer|,(r4)]
                     [,filename|,(r5)]
```

For an explanation of the operands, see DOS/VSE Macro Reference, GC24-5140.

The checkpoint macro generates a 28- or 32-byte field with the format shown in Figure 13 when the fifth and/or sixth operands are specified. If these operands are omitted, bytes 14-19 do not appear, and a 22-byte field is generated.

The macro then generates a calling sequence that places the address of this field in register 0 and the address of the tape or disk checkpoint logical transient name in register 1, and issues an SVC 2 to fetch the transient phase.

Checkpoint Routines (see Chart 23)

The CHKPT macro determines the checkpoint device type and fetches the appropriate routine. If the checkpoint device is a tape, phases $$BCHKPT, $$BCHKP2, and $$BCHKP3 are fetched. If the checkpoint device is a disk, phases $$BCHKPD, $$BCHKPE, $$BCHKPF, and $$BCHKPG are fetched. The phases check the conditions necessary to take a checkpoint and write the checkpoint records on the specified device.

To write an error (or the 'successful') message, the common phase $$BRMSG1 is called. If the checkpoint was successful and a tape reposition table is specified, $$BCHK3G is fetched to perform an 'erase gap on open tape output files'. Control is then returned to the problem program.

Bytes	Contents
0-3 (0-3)	A four-byte address or X'FF' followed by a three-byte register number to specify a restart address.
4-7 (4-7)	A four-byte address or X'FF' followed by a three-byte register number to specify a high byte address. This field is zeros if the end address is not specified.
8-11 (8-B)	A four-byte address or X'FF' followed by a three-byte register number to specify tape reposition status information. This field is zeros if 'tpointer' is not specified.
12 (C)	X'01'
13 (D)	The system unit number from SYSxxx.
14-21 (E-15)	C'$$BCHKPT' if CHKPT device is a tape and 'dpointer' is omitted.
14-15 (E-F)	X'0000'
16-19 (10-13)	A four-byte address or X'FF' followed by a three-byte register number to specify DASD verification information. This field is zeros if 'dpointer' is not specified.
20-27 (14-1B)	C'$$BCHKPT' if the filename parameter is not specified.
20-23 (14-17)	A four-byte address of the user-specified DTFPH for the disk checkpoint file, or X'FF' followed by a three-byte register number (where the register must contain the four-byte address of the DTFPH table).
24-31 (18-1F	C'$$BCHKPD' if the filename parameter is specified.

Numbers in parentheses are displacements in hexadecimal notation.

Figure 13. Information Field Generated by CHKPT Macro

RESTART

Control statements necessary to restart a job from a checkpoint are:

1. JOB statement specifying the same job name as the checkpointed job.

2. ASSGN statements for assigning I/O devices to symbolic unit names.

3. DLBL and EXTENT statements of the checkpoint file if the file is located on disk and if user labels are used.

4. RSTRT statement specifying the unit that contains the checkpoint, and the checkpoint ID number taken from the message that was printed when the checkpoint was taken.

The format of the RSTRT statement follows:

// RSTRT SYSxxx,nnnn,[filename]

SYSxxx Symbolic unit name of the device on which the checkpoint records are stored. This unit must have been previously assigned.

nnnn Identification of the checkpoint record to be used for restarting. This serial number is four characters. It corresponds to the checkpoint identification used when the checkpoint was taken.

filename Symbolic name of the disk checkpoint file to be used for restarting. It must be identical to the filename of the DTFPH to describe the disk checkpoint file and the fifth parameter of the CHKPT macro instruction. This operand applies only when a disk is specified as the checkpoint device.

Restart Routines (see Chart 23)

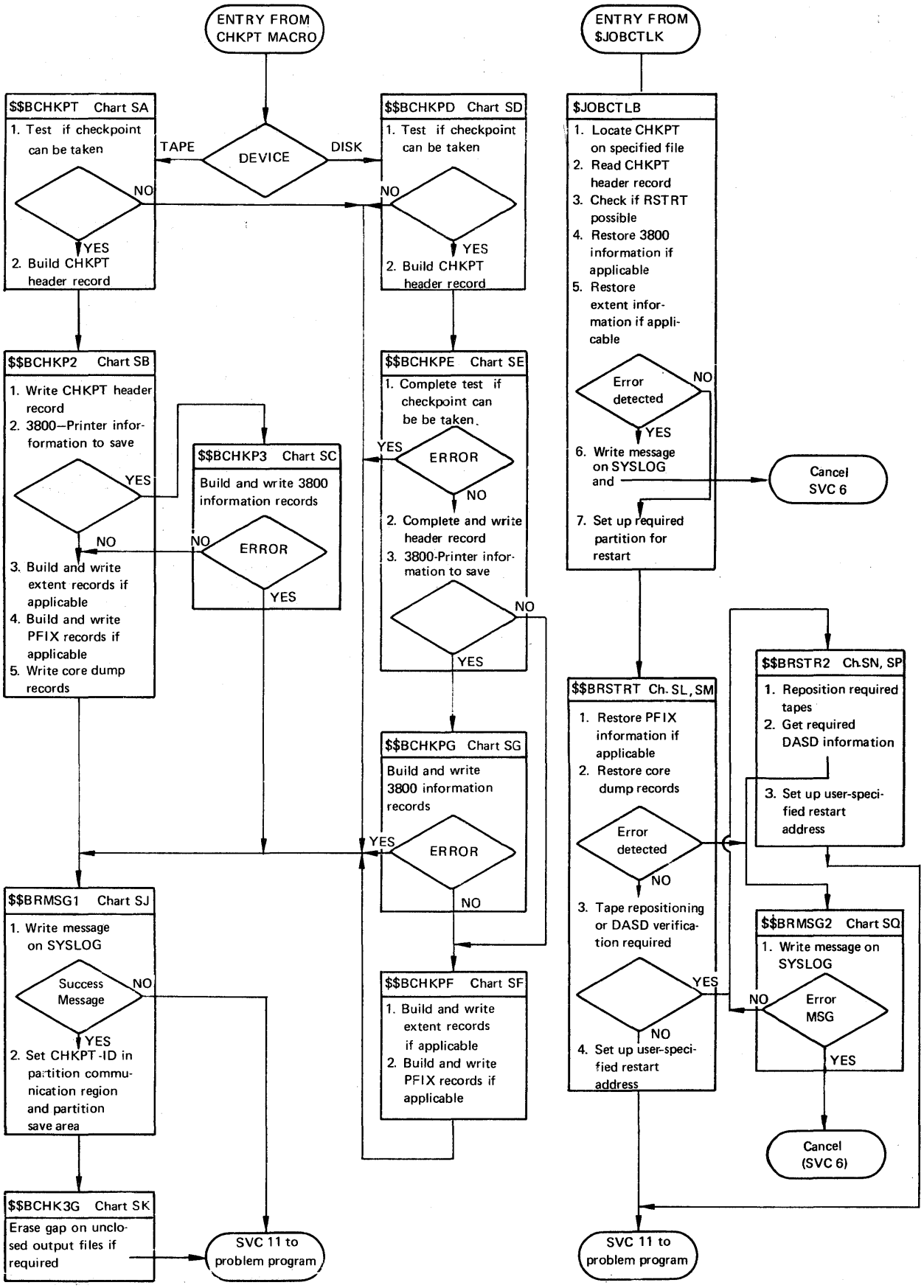
The job control phase $JOBCTLK checks the RSTRT statement for syntactical correctness and builds a parameter list. Control is then passed to the job control phase $JOBCTLB which determines the restart device, locates the checkpoint, reads the header, save, 3800-printer, and extent records, checks the RSTRT partition whether the required functions are supported and whether the partition allocations match those at checkpoint time, restores required parts of the partition communication region, and clears the partition.

Job control then fetches phase $$BRSTRT which restores PFIX information and the problem program area.

If DASD verification and/or tape repositioning is required, $$BRSTR2 is fetched.

If no error condition occurs, control is passed to the restored program. $$BRMSG2 is fetched to write an error message and cancel the program or to write an information message and continue fetching $$BRSTR2 or cancel the program, depending on the operator's reponse.

Chart 23. Checkpoint/Restart Routines



$$BCHKPT - Tape Checkpoint Phase 1

This phase determines if the checkpoint can be taken and starts building the checkpoint header and save record.

Entry: From a CHKPT macro expansion when the unit on which the checkpoint is taken is a magnetic tape device.

Exits: To $$BCHKP2 to complete the checkpoint, or to $$BRMSG1 to print an error message indicating the error condition.

Method: The CHKPT macro provides phase $$BCHKPT with the address (in register 0) of the checkpoint parameter list (see Figure 13).

This phase determines if

- the checkpoint is issued by a maintask
- no subtasks are attached to this maintask
- no I/O is pending on a TP-device
- no tracks/blocks are held
- the checkpoint device is a tape
- the user-specified end address is valid.

If the end address is not supplied by the user, the partition logical end address will be used (PPEND from the partition communication region).

If the GETVIS flag in the partition communication region is on, the user-specified or logical end address is ignored and the whole partition will be saved.

The phase then calculates

- the number of 16K partition dump records.
- the number of PFIX information records (if PFIXing has taken place).
- the number of extent information records (if DASD file protection is supported and an entry in the JIB table or system extent area exists).
- the number of 3800-printer information records (if the 3800 printer is supported and assigned).

The calculated values are stored in the checkpoint header and save record area in the common workarea of the tape checkpoint phases.

$$BCHKPT also retrieves and stores (in the header and save record area) any necessary partition communication region information, the updated checkpoint-ID, and the partition allocations.

The layout of the header and save record is shown in Figure 14.

TAPE Bytes	DISK Bytes	CONTENTS
0-11 (0-B)	0-11 (0-B)	/// CHKPT // (Checkpoint header record-id)
12-13 (C-D)	12-13 (C-D)	Number of program dump records (in binary format)
14-15 (E-F)	14-15 (E-F)	Disk: Not used Tape: Number of checkpoint records following the header record (in binary format), including the number of 3800-printer, extent, and PFIX information records, the program dump records, the save and the trailer record. This field is used by MTMOD to skip a CHKPT file located in a user data file and by Restart to skip a CHKPT file while scanning a tape for a requested CHKPT.
16-19 (10-13)	16-17 (10-11)	Tape: Checkpoint identification number in unpacked decimal format. Disk: Checkpoint identification number in binary format.
	18-19 (12-13)	Only for checkpoints on FBA-devices: Byte 0: Number of blocks needed for information records Byte 1: Number of blocks needed for program dump records
For tape checkpoints the above is the header and trailer record layout. The following information is stored in the save record. For disk checkpoints this is one record (header record).		
0-3 (0-3)	20-23 (14-17)	Highest storage address checkpointed
4-67 (4-43)	24-87 (18-57)	Values of general registers from the partition save area (9-15, 0-8)
68-87 (44-57)	88-107 (58-6B)	Partition communication region bytes 12-31
88-97 (58-61)	108-117 (6C-75)	Partition communication region bytes 36-45
98-99 (62-63)	118-119 (76-77)	Partition communication region bytes 56-57
100-101 (64-65)	120-121 (78-79)	Partition communication region bytes 92-93
102 (66)	122 (7A)	Partition communication region byte 78
103 (67)	123 (7B)	Partition communication region byte 98 bit 0 and byte 134 bit 7
104 (68)	124 (7C)	Byte 12 of the partition PIB

Figure 14. Checkpoint Header and Save Record Format (Part 1 of 2)

105 (69)	125 (7D)	Checkpoint configuration flag:	
		X'01' Real (runmode of CHKPT part.)	X'10' CHKPT device FBA
		X'02' GETVIS flag on	X'20' CHKPT device tape
		X'04' PFIX records saved	X'40' ECPS:VSE-Mode
		X'08' DASD file protect option	X'80' 370-Mode
106-107 (6A-6B)	126-127 (7E-7F)	Length of job label area saved from the partition save area	
108-127 (6C-7F)	128-147 (80-93)	Temporary partition boundaries	
128-147 (80-93)	148-167 (94-A7)	Permanent partition boundaries	
148-149 (94-95)	168-169 (A8-A9)	Number of PFIX information records	
150-151 (96-97)	170-171 (AA-AB)	Number of extent information records	
152-153 (98-99)	172-173 (AC-AD)	Number of 3800-printer information records	
154-159 (9A-9F)	174-179 (AE-B3)	Volume serial number of SYSCAT	

Figure 14. Checkpoint Header and Save Record Format (Part 2 of 2)

$$BCHKP2 - Tape Checkpoint Phase 2

This phase completes and builds the checkpoint records and writes them on the specified tape.

Entry: From $$BCHKPT to start writing the checkpoint and from $$BCHKP3 to continue the checkpoint after 3800-printer information records have been written.

Exit: To $$BCHKP3 to build and write 3800 printer information records, to $$BRMSG1 to write an error (or the 'successful') message.

Method: The information needed to continue the checkpoint is passed to $$BCHKP2 in the common workarea located in the second part of the logical transient area and pointed to by a register loaded in $$BCHKPT.

To complete the header and save record, the number of checkpoint records following the header is calculated and stored, and (if required) the volume serial number of the device assigned to SYSCAT is retrieved and stored.

The phase then

- writes the header and save record on the checkpoint tape (for format see Figure 14).
- calls $$BCHKP3 if 3800-printer information records are to be written (for format see Figure 17).
- builds and writes extent information records if DASD file protection is supported. For disk devices assigned to programmer units, extent information is saved from the JIB and the system extent area, and for SYSCAT extent information is saved from the system extent area. (For the format of extent information records, see Figure 15).
- builds and writes PFIX information records. The PFIX information is retrieved by using the PFIXCHPT macro which returns the PFIX information in a supplied workarea (for format see Figure 16).
- writes program dump records.
- writes the trailer record (same as the header record).

Bytes	Contents
0-2 (0-2)	'XTN' (extent record-id)
3 (3)	Flag Byte: X'00' extents from JIB table X'01' extents from system extent area X'02' SYSCAT extents saved
4-5 (4-5)	Number of extent entries in this record (max. 20); each entry is 12 bytes long and is formatted as follows:
6-7 (6-7)	cuu of logical unit
8 (8)	PUB device type code
9 (9)	Extent type-id: a) if extent is from JIB table: X'20' 2311, 2314 or 2319 X'10' 3330, 3340 or 3350 b) if extent is from system extent area: X'40' Short form used X'00' Long form used
10-11 (A-B)	Lower and upper extent for 2311, 2314 or 2319; Lower extent for 3330, 3340 or 3350
12-13 (C-D)	Not used for 2311, 2314 or 2319; Upper extent for 3330, 3340 or 3350
10-13 (A-D)	Upper extent from system extent area
14-17 (E-11)	Not used if extent is from JIB table; Lower extent from system extent area

Figure 15. Format of Extent Information Record

Records are written without repositioning the user-specified tape. The following record lengths (in bytes) are used:

Header record	20
Save record	160
3800-printer, extent, and PFIX information records	each 256
Program dump records	16K
Last program dump record (multiple of 2K)	2 to 16K
Trailer record	20

If end-of-reel is encountered while checkpoint records are being written, the tape is backspaced to where it was initially positioned, and the header record is scratched. Alternate tapes assigned for the checkpoint file are not checked.

Then $$BRMSG1 is called to write an error message and to return control to the user program. Header and trailer records are written in user mode, while the save record, extent records, and program storage dump records are written in data conversion mode. The density remains the same for all records.

When a checkpoint has successfully been taken, $$BRMSG1 is called to write the 'successful' message and to return to the user program.

Bytes	Contents
0-3 (0-3)	'PFIX' (PFIX record-id)
4-256 (4-100)	Information as retrieved via PFIXCHPT macro

Figure 16. Format of PFIX Information Record

Bytes	Contents
0-3 (0-3)	'PRT ' (3800-printer record-id)
4-5 (4-5)	Logical unit number
6-7 (6-7)	Reserved
8-76 (8-4A)	Information as retrieved via QSETPRT macro

Figure 17. Format of 3800-Printer Information Record

$$BCHKP3 - Tape Checkpoint Phase 3

This phase builds 3800-printer information records and writes them on the specified tape checkpoint file.

Entry: From $$BCHKP2.

Exit: To $$BCHKP2 or, in case of errors, to $$BRMSG1.

Method: The information required to build and write the 3800-printer information records is passed to $$BCHKP3 in the common workarea of the checkpoint phases pointed to by a register loaded in $$BCHKPT.

This phase detects the logical units to which a 3800 printer is assigned, sets up the workareas and I/O areas, and retrieves the 3800-printer information using the QSETPRT macro.

If no error is returned by the macro, the 3800-printer information record is written on the checkpoint tape, one for each printer assigned. (For the format of the 3800-printer information record see Figure 17). If the whole partition LUB table is scanned, phase $$BCHKP2 is fetched to continue the checkpoint.

$$BCHKPD - Disk Checkpoint Phase 1

This phase determines if the checkpoint can be taken and starts building the checkpoint header record.

Entry: From the CHKPT macro expansion if a disk device is used as the checkpoint unit.

Exits: To $$BCHKPE if the checkpoint is to continue; to $$BRMSG1 to write a diagnostic message on SYSLOG and to return control to the problem program if a checkpoint cannot be taken.

Method: The CHKPT macro provides phase $$BCHKPD with the address (in register 0) of the checkpoint parameter list (see Figure 13).

This phase determines if

- the checkpoint is issued from a maintask
- no subtasks are attached to this maintask
- no tracks/blocks are held
- the DTFPH is open, mounted single, and an output file
- the checkpoint device is valid
- the user-specified end address is valid.

If the end address is omitted, the partition logical end address will be used (PPEND from the partition communication region).

If the GETVIS flag in the partition communication region is on, the user-specified or logical end address is ignored and the whole partition will be saved.

The phase then calculates

- the number of 2K program dump records for 2314, 3330, 3350, and FBA devices, or of 1K program dump records for 2311 or 3340 devices.
- the number of blocks needed for information records and program dump records for FBA devices.
- the number of extent information records (if DASD file protection is supported and an entry in the JIB table or system extent area exists).
- the number of 3800-printer information records (if the 3800 printer is supported and assigned).

The calculated values are stored in the checkpoint header record area (see Figure 14) in the common workarea and I/O area of the disk checkpoint phases.

$$BCHKPD also saves the registers from the problem program save area, the temporary partition boundaries, and the volume serial number of SYSCAT, which is necessary if extent information is to be saved (provided VSAM is supported, SYSCAT is assigned, and extents are found in the system extent area).

$$BCHKPE - Disk Checkpoint Phase 2

This phase continues checking the required checkpoint conditions and building the checkpoint header record, and writes the checkpoint header record on the checkpoint file.

Entry: From $$BCHKPD.

Exit: To $$BCHKPF if the checkpoint is to continue; to $$BCHKPG if the checkpoint is to continue and 3800-printer information is to be saved; to $$BRMSG1 if a checkpoint cannot be taken. The message writer is called to write a diagnostic message on SYSLOG, and to return control to the problem program.

Method: The information needed to continue the checkpoint is passed to $$BCHKPE in the common workarea located in the second part of the logical transient area and pointed to by a register loaded in $$BCHKPD.

This phase

- calculates the number of PFIX information records needed (if PFIXing is supported and has taken place).
- determines if enough space has been allocated for this checkpoint. If checkpoints are already in the specified file and the remaining space does not suffice to contain the current checkpoint, the file is used from the beginning and the existing checkpoint(s) is (are) owerwritten by the current record.
- checks that no I/O request is pending on a teleprocessing device belonging to the checkpoint partition.
- completes the checkpoint header record by moving the required partition communication region information, setting the checkpoint-id, and saving the permanent partition allocations.
- initializes the channel programs for the specified checkpoint device.
- writes the header record on the checkpoint file.

$$BCHKPF - Disk Checkpoint Phase 3

This phase builds and writes checkpoint records on the checkpoint disk file.

Entry: From $$BCHKPE or from $$BCHKPG if 3800-printer information records are written.

Exit: To $$BRMSG1, the message writer, which writes a message on SYSLOG indicating that a successful checkpoint was taken.

Method: The information needed to build and write the checkpoint records is passed to $$BCHKPF in the common workarea of the checkpoint phases located in the second part of the logical transient area and pointed to by a register loaded in $$BCHKPE.

The phase

- builds and writes extent information records if DASD file protection is supported. For disk devices assigned to programmer units, extent information is saved from the JIB table, and for disk devices assigned to SYSCAT, extent information is saved from the system extent area. (For the format of extent information records, see Figure 15.)
- builds and writes PFIX information records. The PFIX information is retrieved by using the PFIXCHPT macro which returns the PFIX information in a supplied workarea (for format, see Figure 16).
- writes program dump records on the checkpoint file.

The records are written with the following lengths (in bytes):

Header record		180
3800-printer, extent, and PFIX information records	each	256
Program dump records on 2314, 2319, 3330, 3350, and FBA devices		2K
Program dump records on 2311 and 3340 devices		1K

The block number needed to write an information or program dump record on an FBA device is calculated by $$BCHKPD and depends on the block size of the device.

$$BCHKPG - Disk Checkpoint Phase 4

This phase builds 3800-printer information records and writes them on the disk checkpoint file.

Entry: From $$BCHKPE.

Exit: To $$BCHKPF to continue the checkpoint, or to $$BRMSG1 to write an error message on SYSLOG.

Method: The information necessary to build and write the 3800-printer information records is passed to $$BCHKPG in the common workarea of the checkpoint phases pointed to by a register loaded in $$BCHKPE.

This phase detects the logical units to which a 3800 printer is assigned, sets up the workareas and I/O areas, and retrieves the 3800-printer information using the QSETPRT macro.

If no error is returned from the macro, the 3800-printer information record is written on the checkpoint file, one for each

printer assigned. (For the format of the 3800-printer information record, see Figure 17). If the whole partition LUB table is scanned, phase $$BCHKPF is fetched to continue the checkpoint.

$$BCHK3G - Tape and Disk Checkpoint, Last Phase

Objective: To perform an erase-gap command for output files that are not closed from the tape repositioning table for logical files.

Entry: From $$BRMSG1 if a successful checkpoint was taken.

Exit: To the user via an SVC 11; to $$BRMSG1 to write a warning message if errors in the user-supplied reposition table are detected.

Method: The addresses of the checkpoint parameter list and the partition save area are supplied in registers by $$BRMSG1. If a tape reposition table for logical files is specified, that table is scanned, and for each specified output tape an erase gap command is performed. Any entry which is not of the expected type (DTFMT) or not for a tape device is ignored and message 0C18I ERRORS DETECTED IN REPOSITIONING TABLE is issued. When the end of the table is reached and no error was found, control is given to the user issuing an SVC 11. If errors are detected, $$BRMSG1 is called to write a message.

$$BRMSG1 - Checkpoint Message Writer

This phase builds and writes checkpoint messages and returns control to the user, or fetches the erase-gap routine $$BCHK3G.

Entry: From phase $$BCHKPT, $$BCHKP2, $$BCHKP3, $$BCHKPD, $$BCHKPE, $$BCHKPF, $$BCHKPG or $$BCHK3G, with registers 2 to 6 loaded with information necessary to get and complete the message text.

Exit: Depends on the kind of message written.

- For an internal error message a dump is issued and the job is canceled.
- For an error message, register zero is cleared and given to the user via the partition save area. The user is given control via SVC 11.
- For the 'successful' message, the checkpoint number is updated in the partition communication region and stored in register zero within the partition save area (in unpacked decimal format). If a logical reposition table is specified for the current checkpoint, control is given to $$BCHK3G; otherwise control is given to the user via SVC 11.

Method: The specified message number is used to get the address of the corresponding message text, and to calculate the values necessary to convert and complete the message text. The message is written on SYSLOG using physical IOCS.

$$BRMSG1 may issue the following messages:

```
0C00I  CHKPT NO. xxxx WAS TAKEN ON
       SYSxxx=cuu
0C02I  CHKPT LOGICAL UNIT NOT TAPE
       SYSxxx=cuu        CHECKPOINT IGNORED
0C03I  I/O REQUEST PENDING ON TP DEVICE
       CHECKPOINT IGNORED
0C04I  INVALID END ADDRESS SPECIFIED
       CHECKPOINT IGNORED
0C05I  CHKPT DTFPH IS NOT OPEN
       FILE=filename     CHECKPOINT IGNORED
0C06I  CHKPT DTFPH MOUNTED=ALL
       FILE=filename     CHECKPOINT IGNORED
0C07I  CHKPT DTFPH NOT OUTPUT
       FILE=filename     CHECKPOINT IGNORED
0C08I  CHKPT UNIT NOT A VALID DISK
       SYSxxx=cuu        CHECKPOINT IGNORED
0C09I  INSUFF. SPACE ALLOCATION
       FILE=filename     CHECKPOINT IGNORED
0C10I  SUBTASK ISSUED CHKPT
       CHECKPOINT IGNORED
0C11I  SUBTASKS ATTACHED
       CHECKPOINT IGNORED
0C12I  TRACKS HELD
       CHECKPOINT IGNORED
0C13I  INSUFF. SPACE FOR CHKPT ON
       SYSxxx=cuu        CHECKPOINT IGNORED
0C14I  CHKPT DEVICE NOT ASSIGNED SYSxxx
       CHECKPOINT IGNORED
0C15I  CHKPT LOGICAL UNIT INVALID SYSxxx
       CHECKPOINT IGNORED
0C16I  QSETPRT FAILED RC=X'ooonxxrr'
       SYSxxx=cuu        CHECKPOINT IGNORED
0C17I  INTERNAL CHKPT ERROR IN $$BCHKxx
       macroname FAILED RC=X'nn'
       CHECKPOINT IGNORED
0C18I  ERRORS DETECTED IN REPOSITIONING
       TABLE
0C19I  CHKPT DEVICE NOT READY SYSxxx=cuu
       CHECKPOINT IGNORED
```

$$BRSTRT - Restore Checkpointed Partition

This phase restores PFIX information saved at checkpoint time and the problem program area.

Entry: From $JOBCTLB.

Exit: To $$BRSTR2 if a tape repositioning and/or a DASD verification table is specified in the checkpoint parameter list; to the problem program via SVC 11; to $$BRMSG2 if an error is detected.

Method: The values necessary to continue with restart are supplied in the first 200 bytes of the cleared partition by $JOBCTLB.

$$BRSTRT

- saves the information passed from $JOBCTLB.
- corrects the partition logical end address if the checkpoint partition was running in real mode and GETVIS is indicated.
- reads and (if applicable) restores PFIX information using the PFIXREST macro.
- reads the program dump records into the partition.
- positions the checkpoint tape after the trailer label (if tape checkpoint).
- updates the checkpoint DTFPH (if disk checkpoint).

The checkpoint parameter list is then retrieved from the restored problem program to check if a tape reposition or DASD verfication table is specified. If specified, phase $$BRSTR2 is called. If not, the restart address is retrieved from the checkpoint parameter list, stored into the PSW, and control is given to the problem program issuing an SVC 11.

$$BRSTR2 - Tape and DASD Verification

This phase repositions tapes as specified in the user-supplied tape reposition table and/or verifies DASD devices as specified in the user-supplied DASD verification table, both pointed to by an address in the checkpoint parameter list.

Entry: From $$BRSTRT or $$BRMSG2 if an information message was written and the operator response was retry or ignore.

Exit: To $$BRMSG2 to write an error or information message, or to the problem program via SVC 11.

Method: Pointers to the information needed to handle the tape reposition and DASD verfication table are supplied in registers loaded by $$BRSTRT.

Tape repositioning:

- Logical repositioning, if specified:
  The entries in the logical reposition table are checked and a message is issued if the DTF type is incorrect (not DTFMT), the device is not assigned, and the assigned device is not a tape. If the DTF is not specified for output or is not open, the entry is ignored.

  The number of records specified by the DTF are read. If a VOL1 label is found, the volume serial number and the volume sequence number are printed on SYSLOG so that the operator can determine if the correct reel is mounted, or mount a new tape, or cancel the job, or continue. Standard Labels and tape marks are bypassed and do not decrement the record count.

  If a backward direction is indicated in the DTF, it must be a non-standard or unlabeled tape; if it is not, the entry is ignored. If the file was open at the time the checkpoint was taken, the user must position the tape to a point past the tapemark following the last record, but before any non-standard trailer labels. This phase then moves the tape backwards over the tapemark (and any noise record), and positions the tape the prescribed number of records from the end of the file.

- Physical repositioning:
  After the logical reposition table is processed, $$BRSTR2 checks for physical reposition entries. If any, the defined logical unit is checked whether it is assigned to a tape unit. If not, a message is issued; otherwise the tape is spaced forward by the specified number of files (tapemarks) and then by the specified number of records.

- Logical and physical repositioning:
  If any checkpoint records are detected, they are skipped without incrementing the record count. If an unexpected tapemark is read an error message is issued.

DASD verification:

After tape reposition is processed or is omitted, the DASD verification table is handled. The logical unit specified in the table is checked whether it is assigned to a DASD device. If it is, the VOL1 label is read and the volume serial number is written on SYSLOG for verification. The operator can cancel the job, mount a new pack and retry, or ignore the message and continue. After the verification table is processed, the restart address is stored in

the PSW and control is given to the problem program issuing an SVC 11.

$$BRMSG2 - Restart Message Writer

This phase builds and writes messages on SYSLOG. For information messages the operator response is analyzed to continue the program.

Entry: From phase $$BRSTRT and $$BRSTR2 with a register containing the message number and pointers to information necessary to complete the message text.

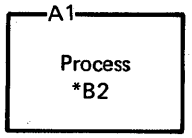
Exit: For error messages the program is canceled. For internal error messages a dump is issued and the program is canceled. For information messages the operator response is checked for validity and the program is canceled, or control is given back to $$BRSTR2.

Method: The given message number is used to find the corresponding message text and to calculate the values necessary to convert and complete the message text. The message is written on SYSLOG using physical IOCS. If a response is expected the answer is compared with a table entry for that message. If the response is not valid, the operator is notified to repeat his answer. If it is valid, the program continues.

$$BRMSG2 may issue the following messages:

```
0R09I  INTERNAL RSTRT ERROR IN $$BRSTRx
       SETLIMIT FAILED RC=X'xx' SIZE=yyyyK
0R13I  INTERNAL RSTRT ERROR IN $$BRSTRx
       EXPECTED RECORD NOT FOUND
0R15I  INTERNAL RSTRT ERROR IN $$BRSTRx
       macroname FAILED RC=X'nn'
0R20A  RIC TAPE REPO: SER xxxxxx SEQ xxxx
       SYSxxx=cuu
0R21A  IC TAPE REPO: TAPE MARK IN DATA
       SYSxxx=cuu
0R22A  IC TAPE REPO: DEVICE NOT A TAPE
       SYSxxx=cuu
0R23A  IC TAPE REPO: DTFTYPE X'nn' INVALID
       filename
0R24A  IC TAPE/DASD: UNIT NOT ASSIGNED
       SYSxxx
0R25A  RIC DASD VERI: SER xxxxxx ASSIGNED
       SYSxxx=cuu
0R26A  RIC DASD VERI: VOL.SER.NO. INVALID
       SYSxxx=cuu
0R27A  IC DASD VERI: DEVICE NOT A DISK
       SYSxxx=cuu
0R28A  RIC DASD VERI: DEVICE NOT READY
       SYSxxx=cuu
0R29A  IC DASD VERI: LOG.UNIT INVALID
       SYSxxx
0I39A  INVALID RESPONSE
```

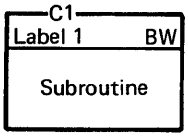
EXPLANATION OF FLOWCHART SYMBOLS



DESCRIPTION

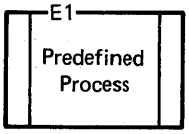
A group of program instructions that perform a processing function of the program. The label, if any, is shown above the block.

*B2
If any additional explanation is required, its location on the chart is identified by an asterisk and the block ID.

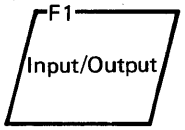


Description of a subroutine. The starting label of the routine appears above the stripe. If the subroutine is documented in detail on another flowchart, the ID of this flowchart is also shown.

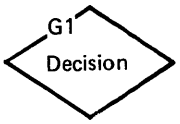
An instruction, or group of instructions, that changes portions of a routine or initializes a routine for given conditions.



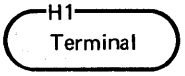
A group of operations not detailed in the flowcharts in this manual, such as user's routines.



Any function of an input/output device or program, usually branching to an I/O routine to perform the function stated in the block.



Points where the program branches to alternate processing, based upon variable conditions such as program switch settings and test results.



The beginning, end or point of interruption in a program.

On-page connector. An entry from or an exit to another function on the same flowchart. The number in the connector identifies the corresponding entry or exit on the chart.

Filinp

Off-page connector, an entry from, or an exit to, a given point on another flowchart. The characters in the connector identify the chart and block. The corresponding label, if any, is placed outside the connector. For multiple entries and exits, an asterisk appears in the connector and the characters are listed nearby.

EXAMPLE

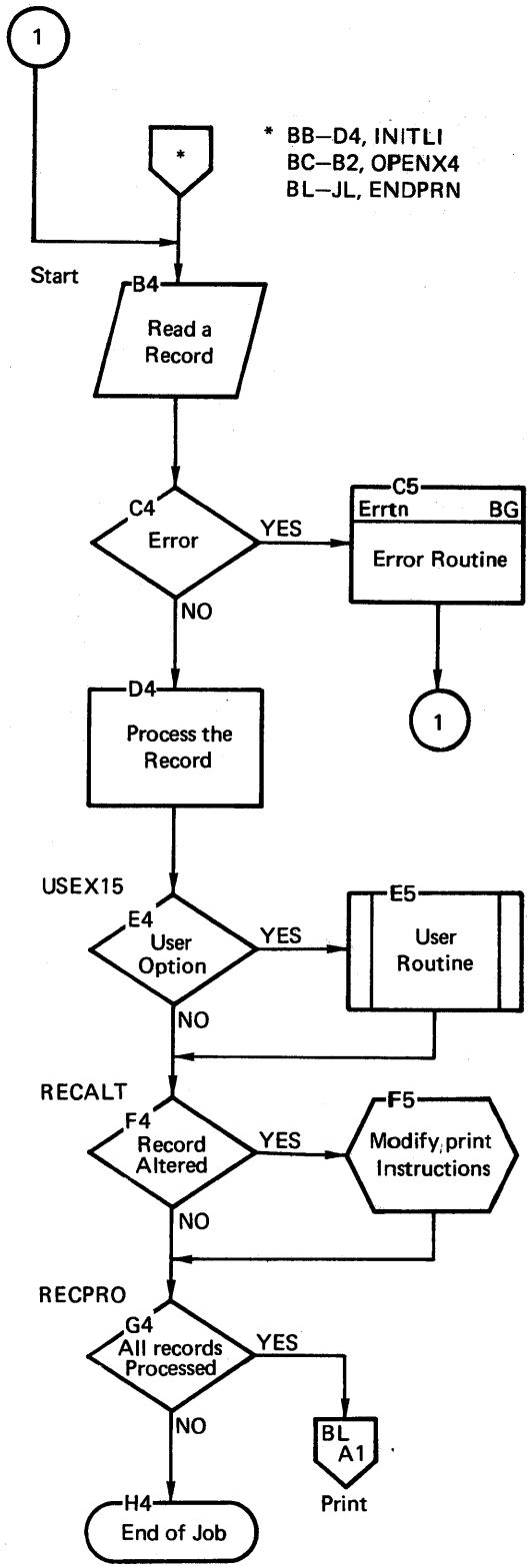


Chart AA. $$BATTNA - Attention Root Phase
(Refer to Chart 02)

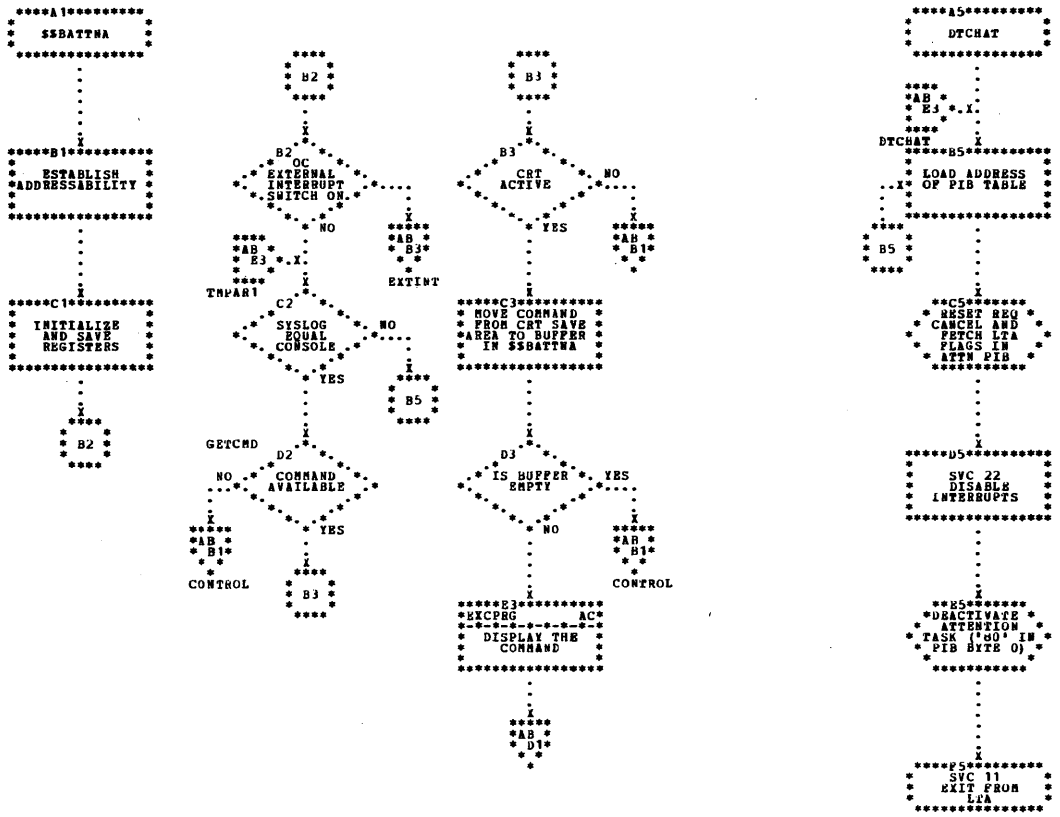


Chart AB. $$BATTNA - Control Routine
(Refer to Chart 02)

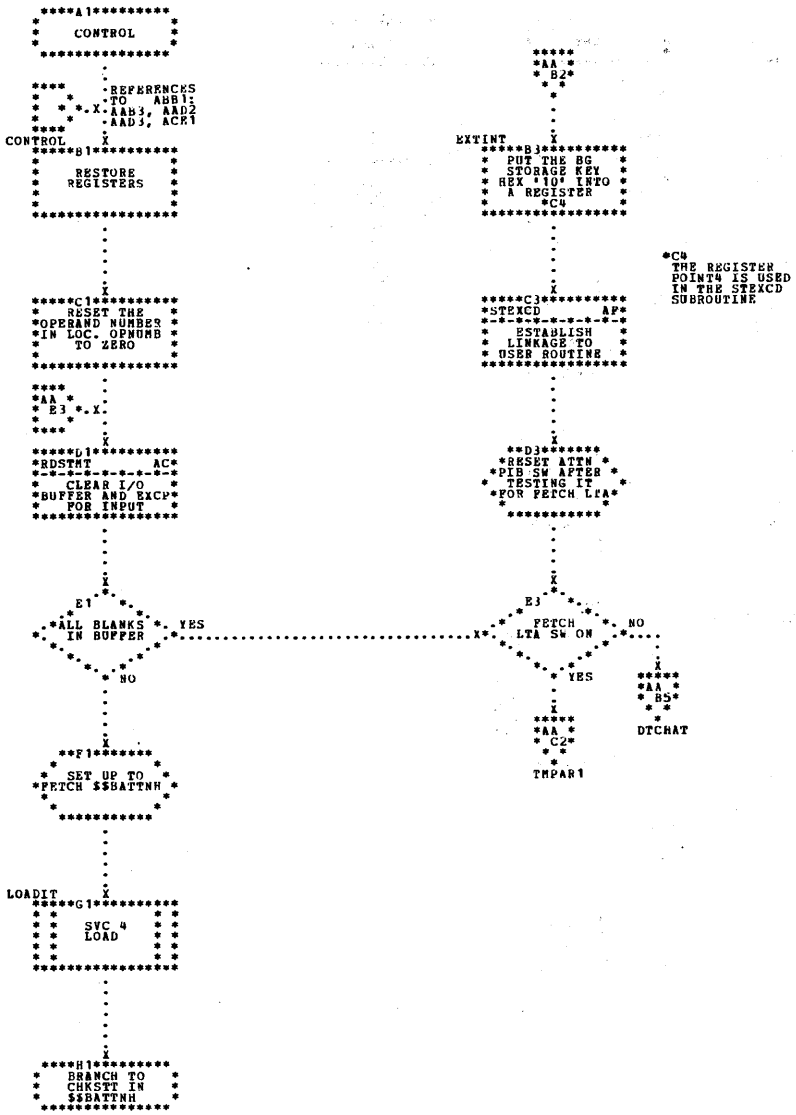


Chart AC. $$BATTNA - Error Message Routines
(Refer to Chart 02)

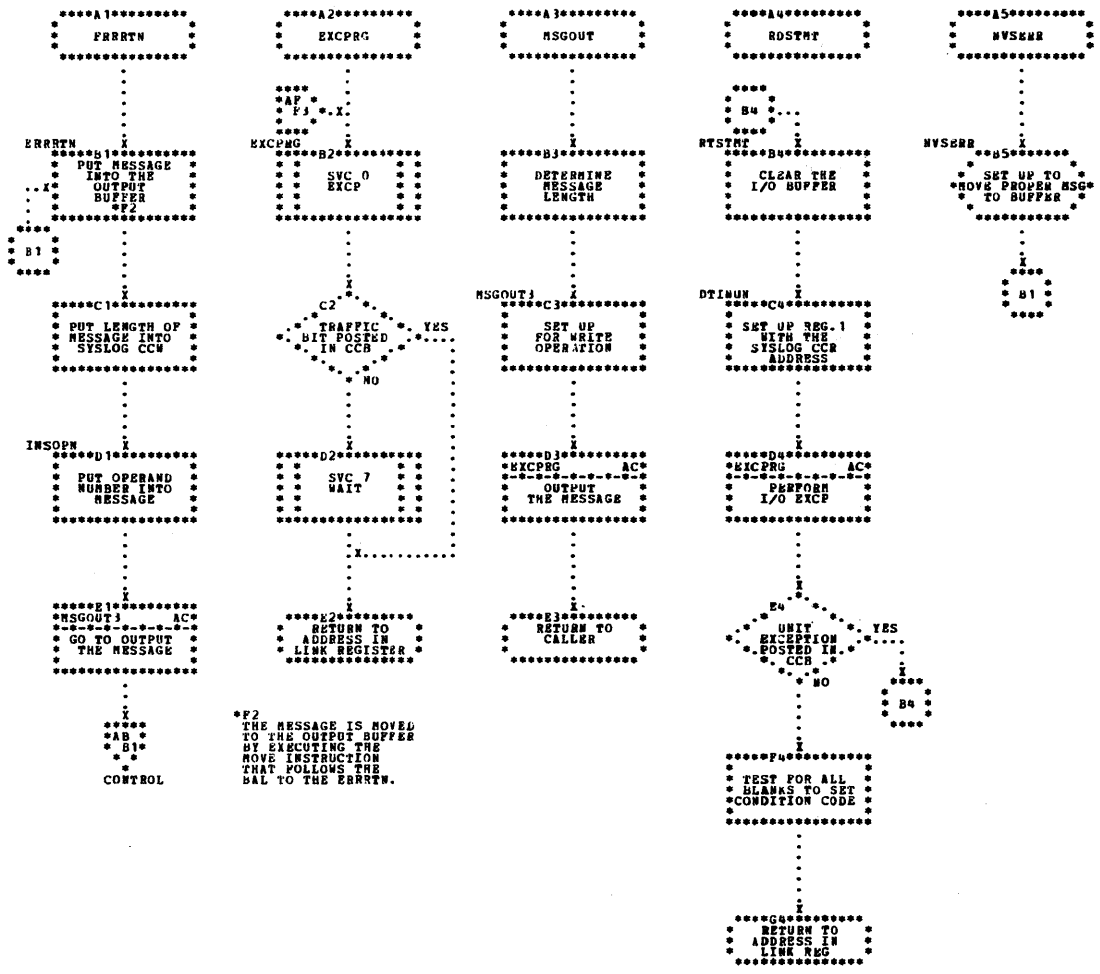


Chart AD. $$BATTNA - General Scan Routines
(Refer to Chart 02)

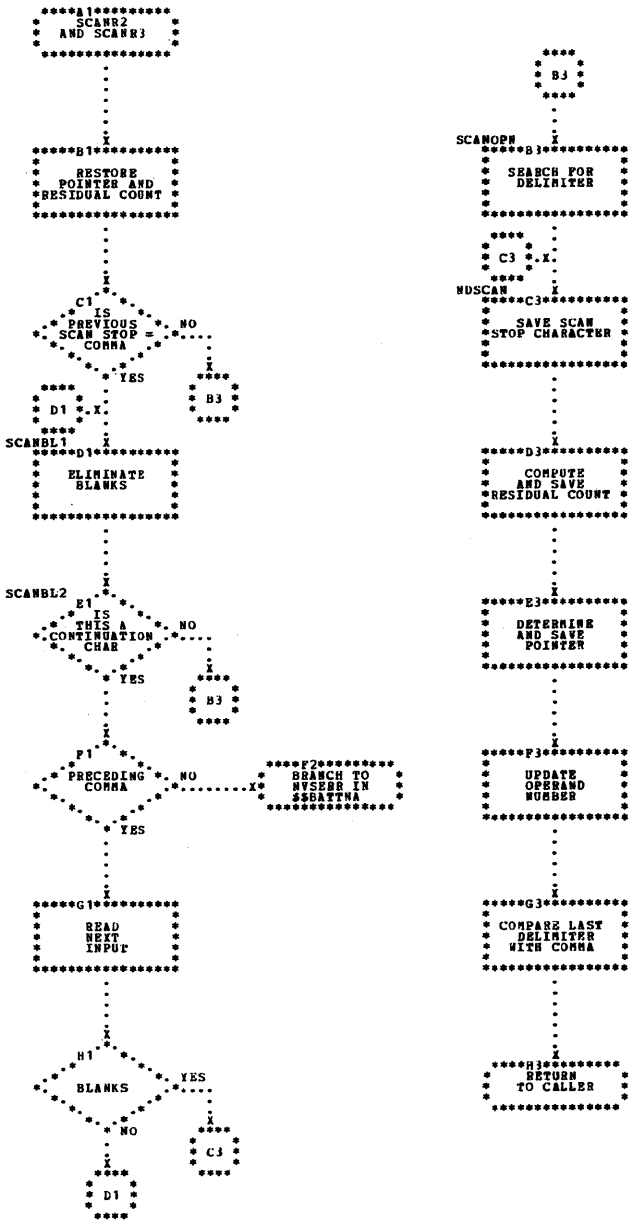


Chart AE. $$BATTNB - MSG Command Processor
(Refer to Chart 03)

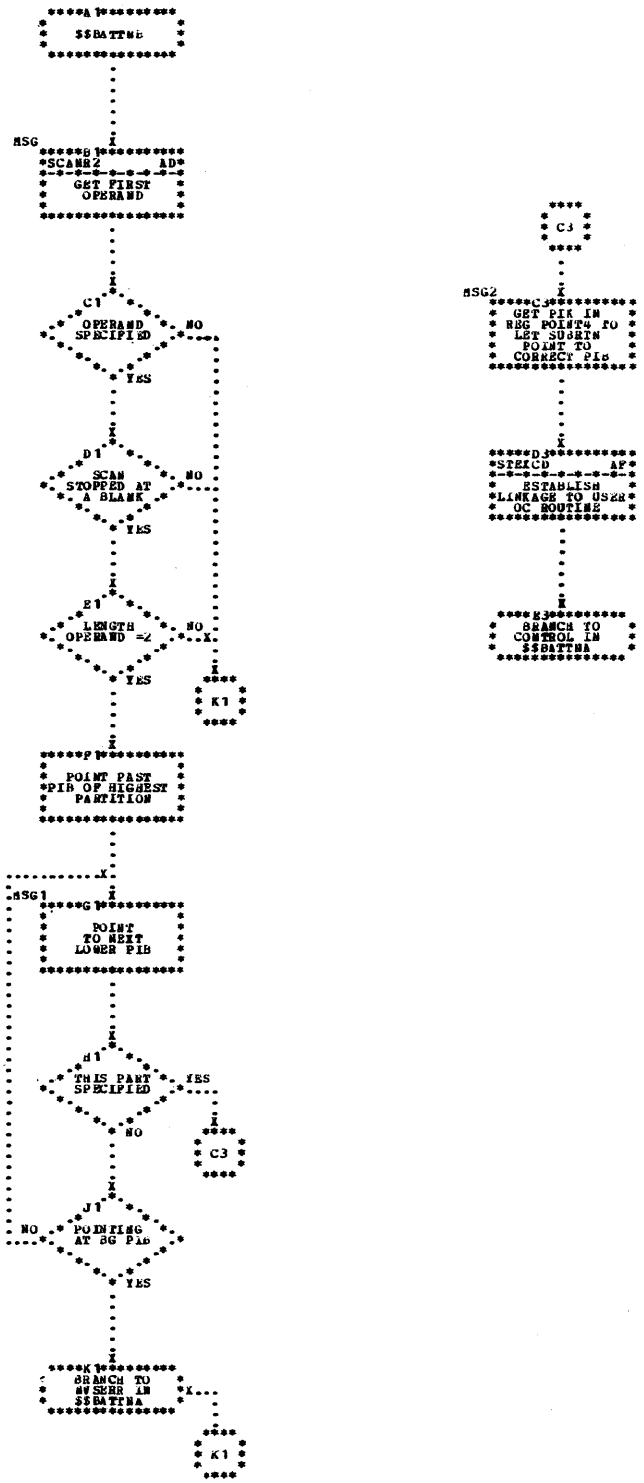


Chart AF. $$BATTNB - Set Operator Communications and Exit Linkage
(Refer to Chart 03)

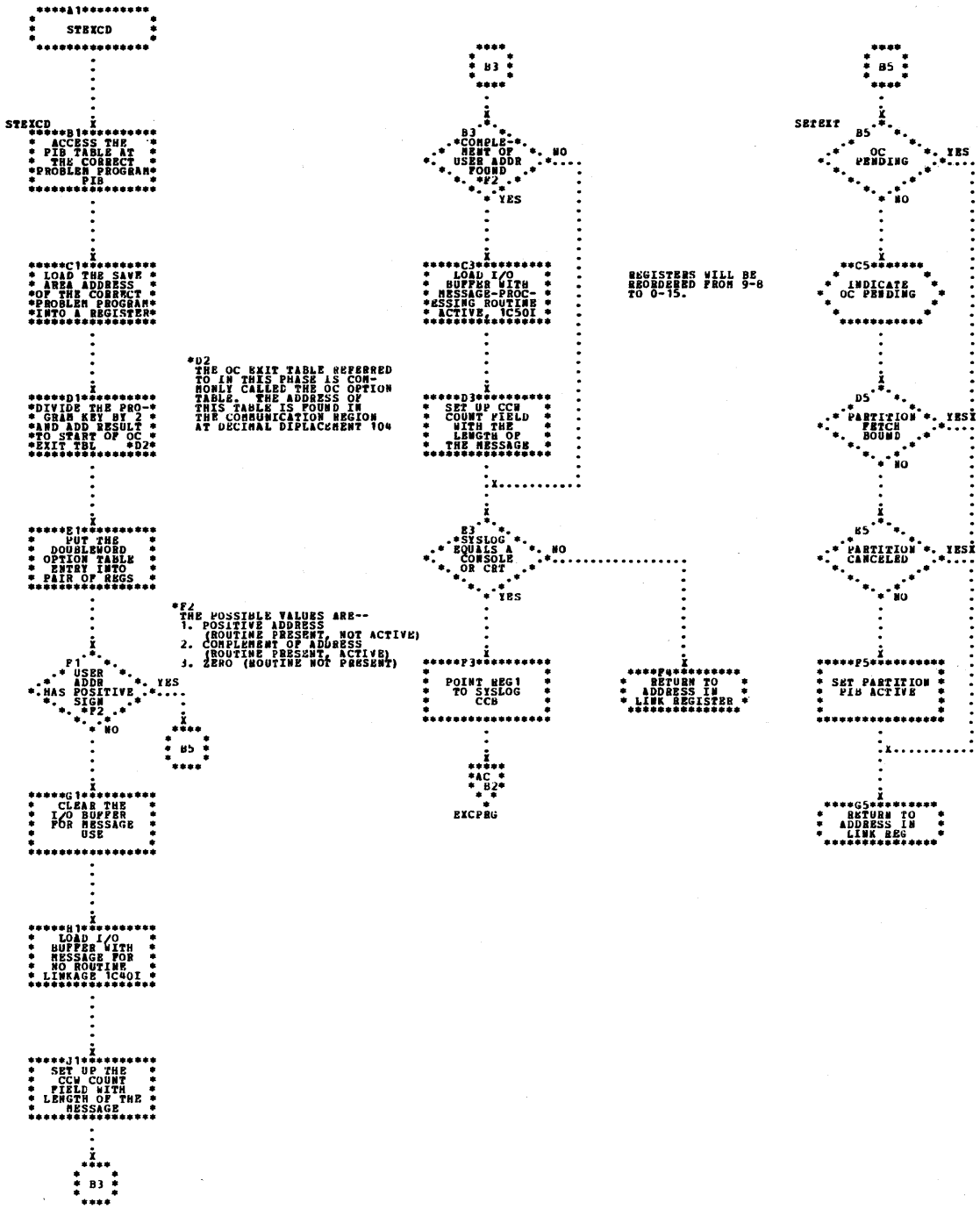


Chart AG. $$BATTNC - CANCEL Command Processor (Part 1 of 2)
(Refer to Chart 03)

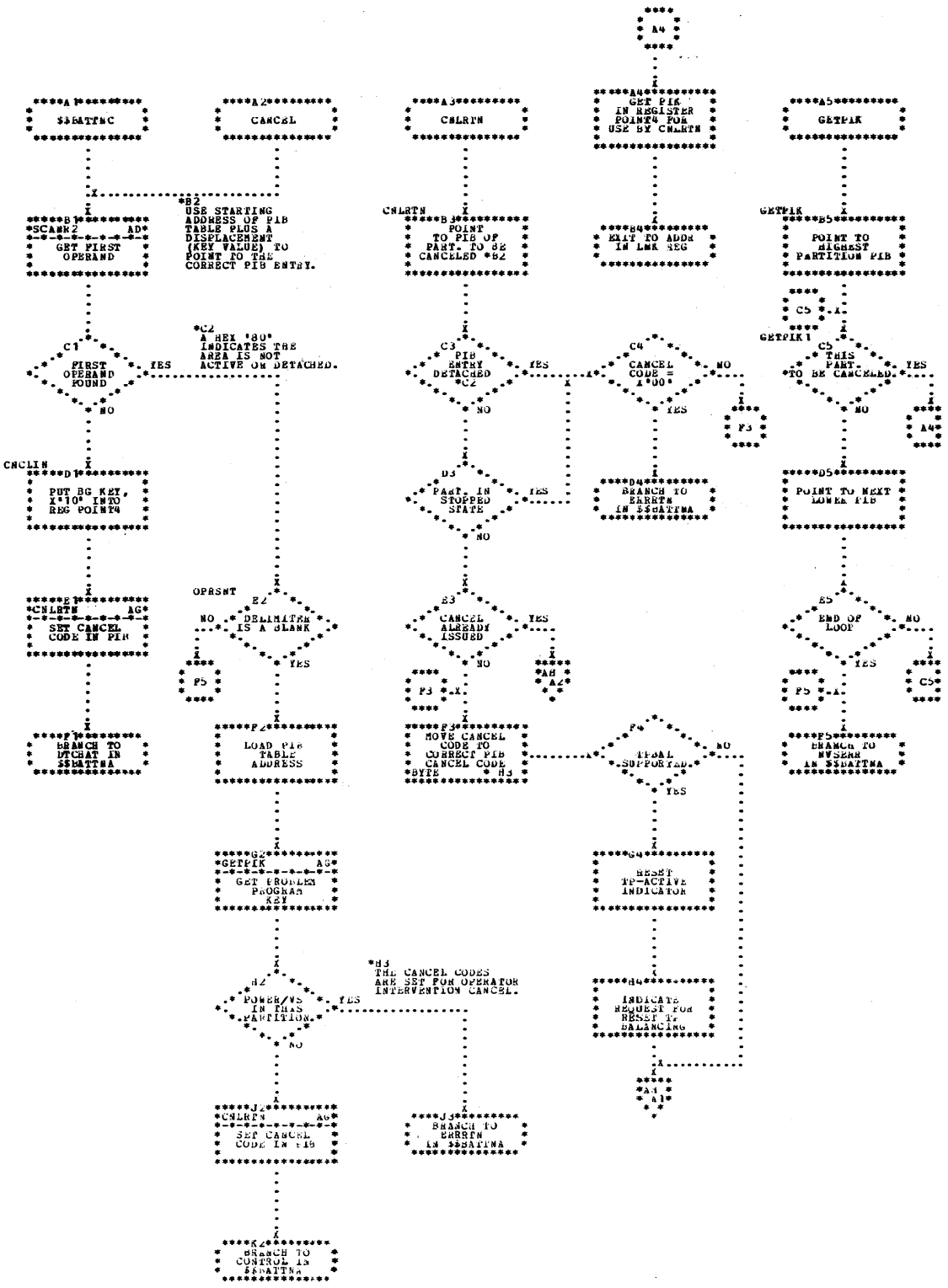


Chart AH. $$BATTNC - CANCEL Command Processor (Part 2 of 2) and NEWVOL Command Processor (Refer to Chart 03)

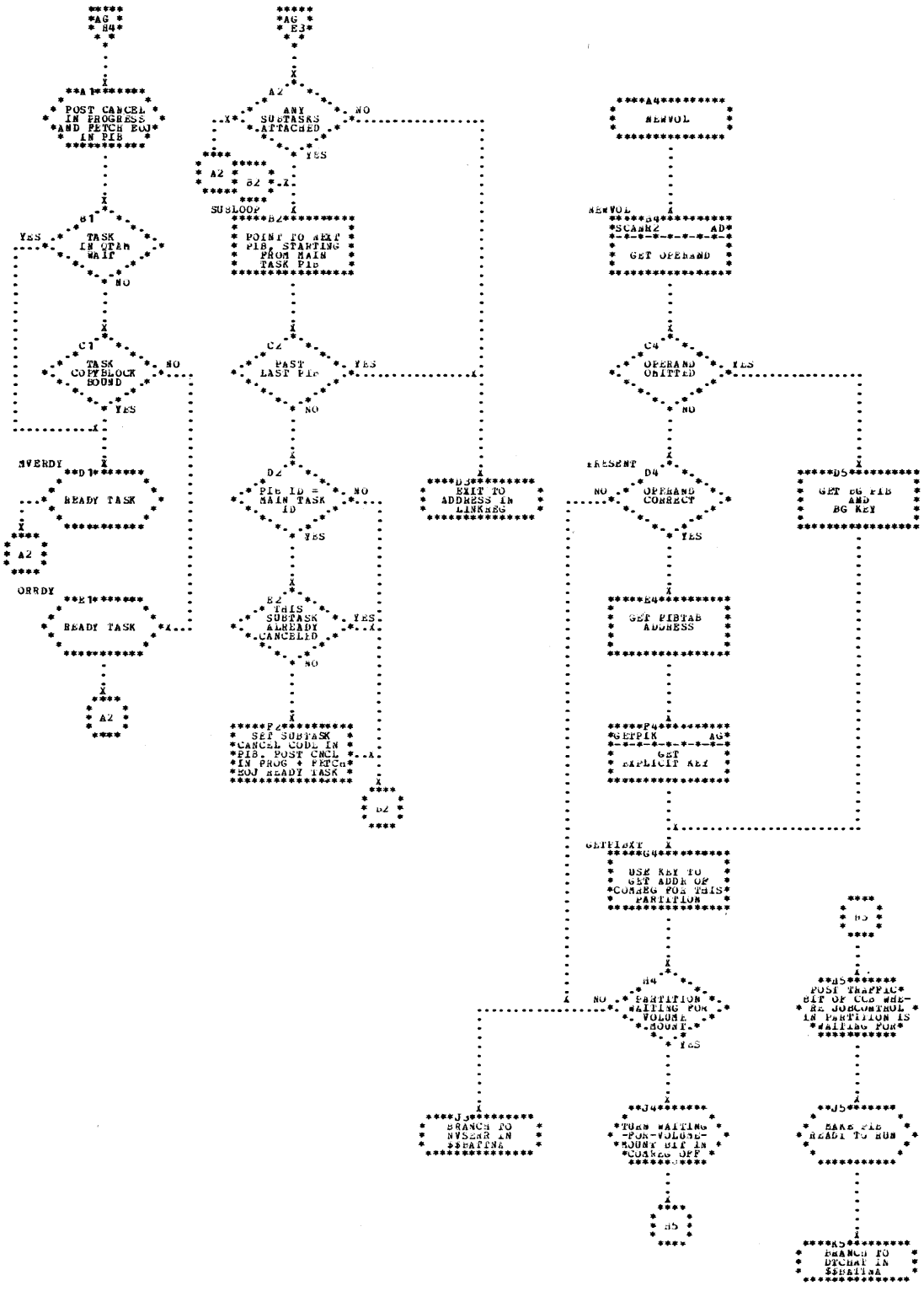


Chart AJ. $$BATTNC - PAUSE, LOG, NOLOG, and IGNORE Command Processor
(Refer to Chart 03)

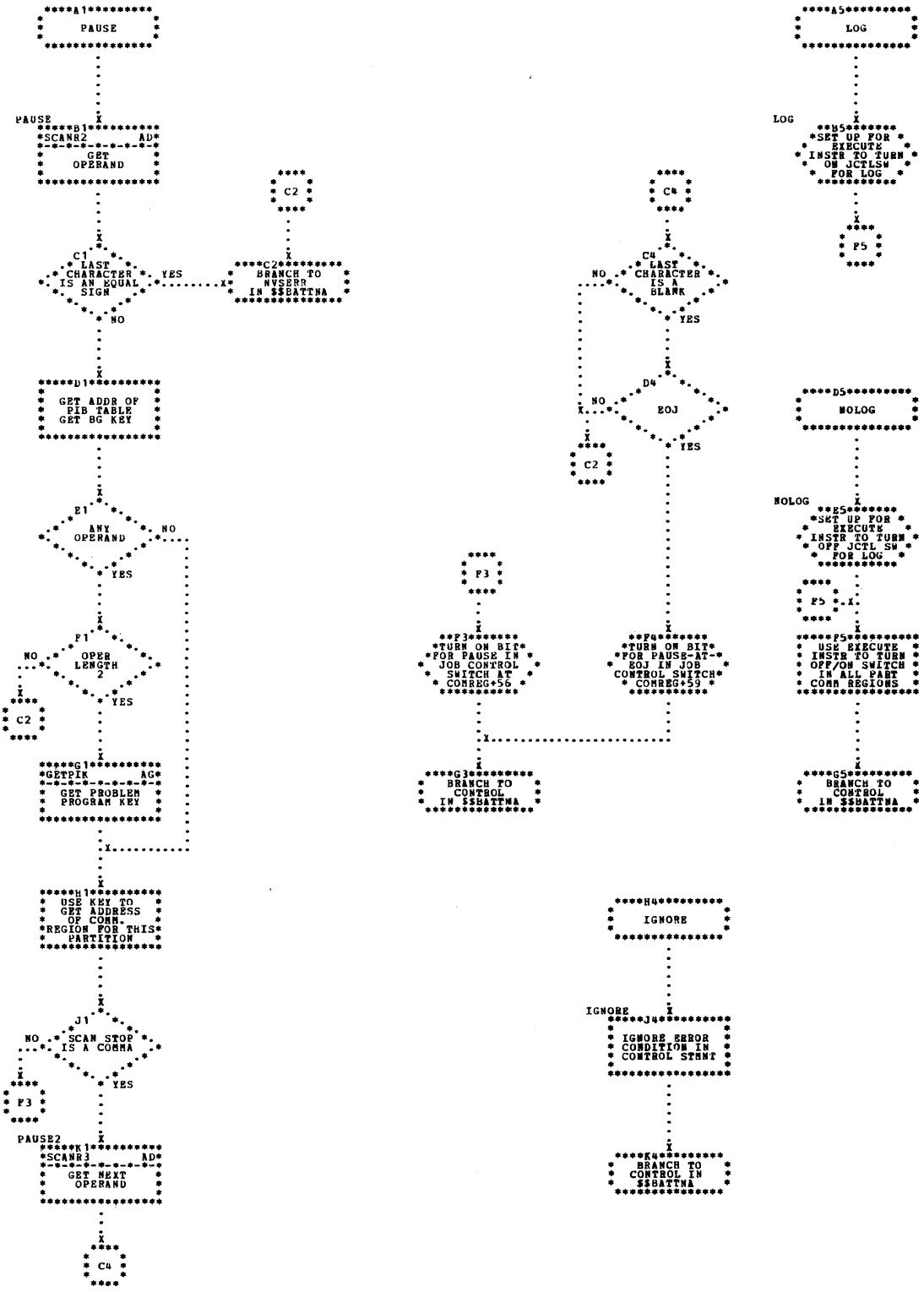


Chart AK. $$BATTND - MAP Command Processor (/370-Mode)
(Refer to Chart 03)

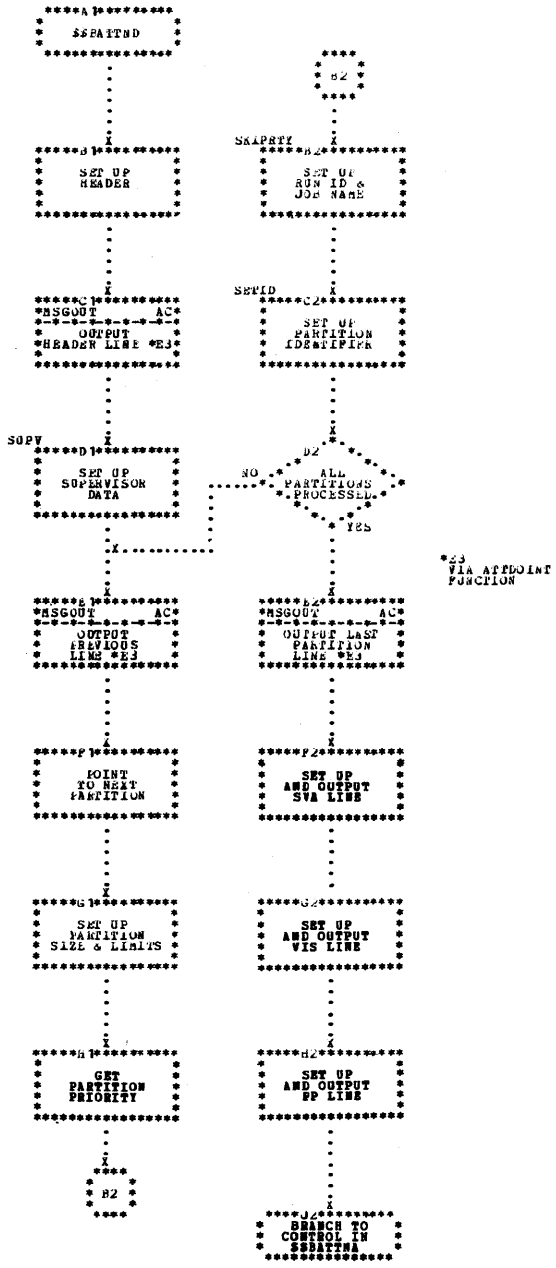


Chart AL. $$BATTNI - MAP Command Processor (ECPS:VSE-Mode)
(Refer to Chart 03)

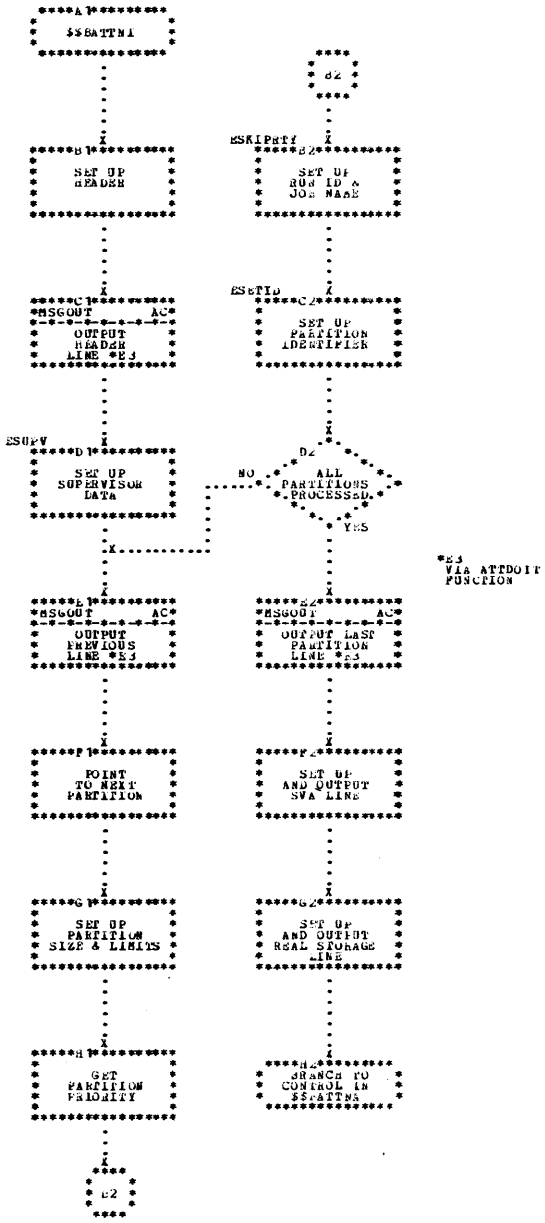


Chart AM. $$BATTNE - ALLOC/ALLOCR Command Processor (Part 1 of 2)
(Refer to Chart 04)

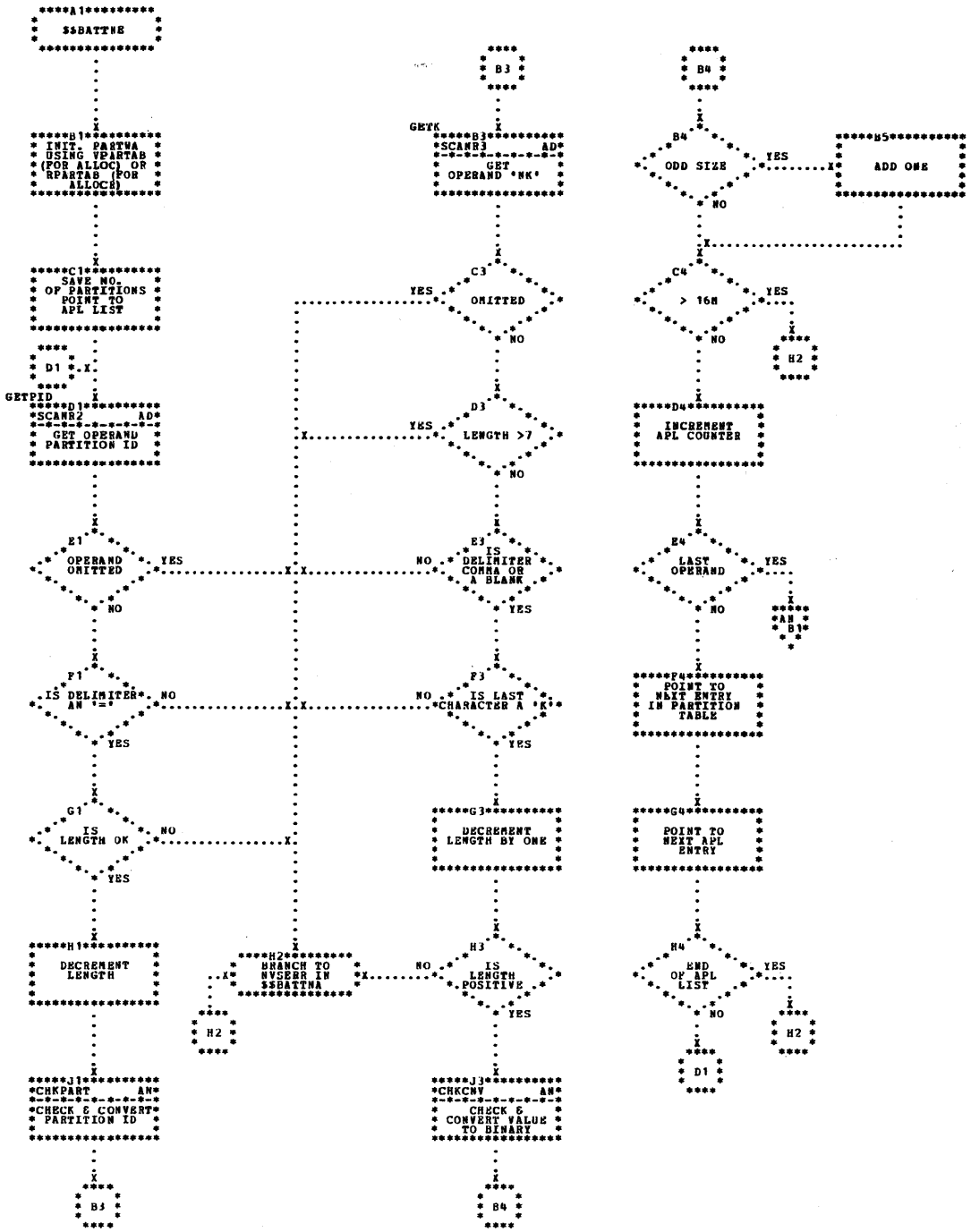


Chart AN. $$BATTNE - ALLOC/ALLOCR Command Processor (Part 2 of 2)
(Refer to Chart 04)

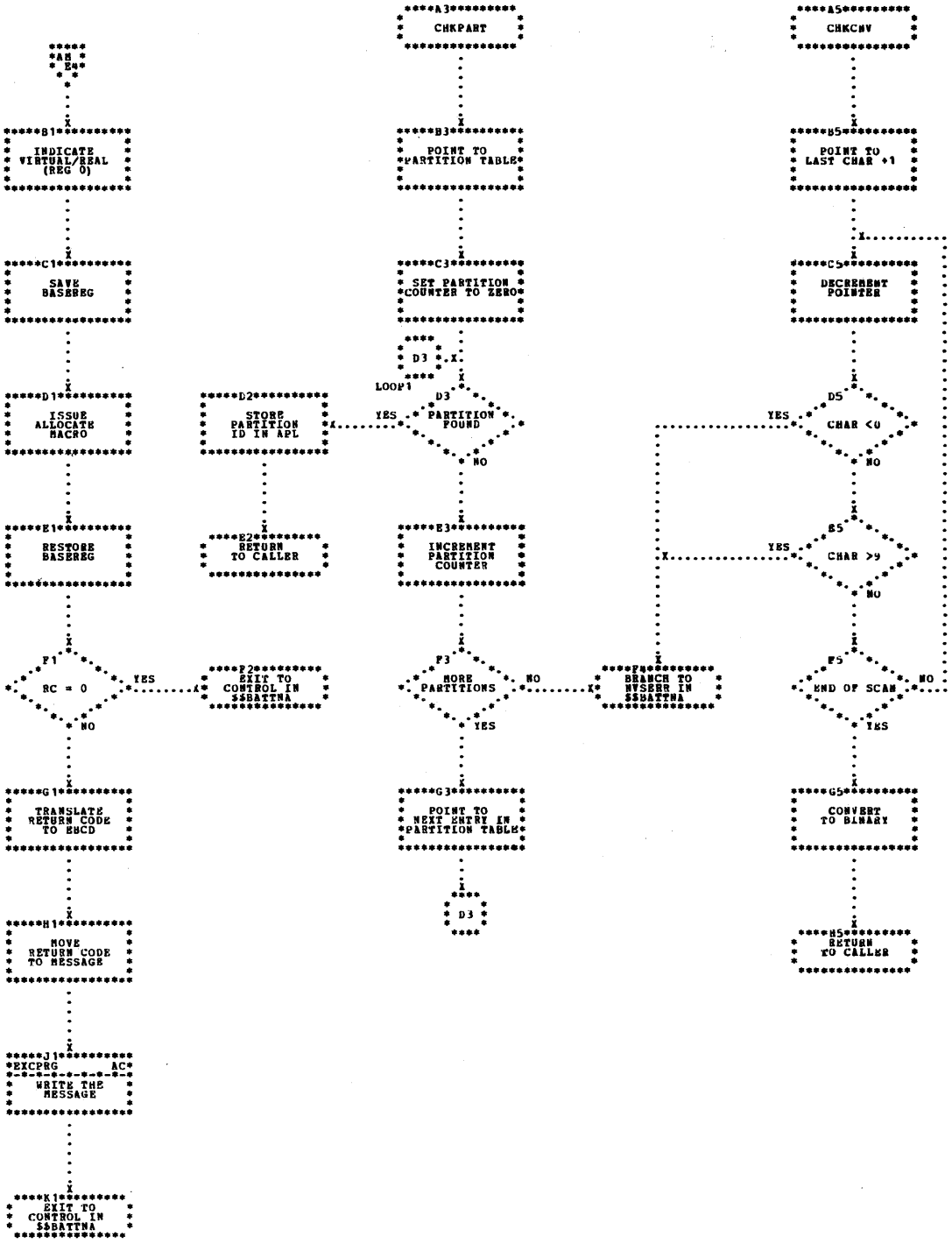


Chart AP. $$BATTNF - SIZE Command Processor (Part 1 of 2)
(Refer to Chart 04)

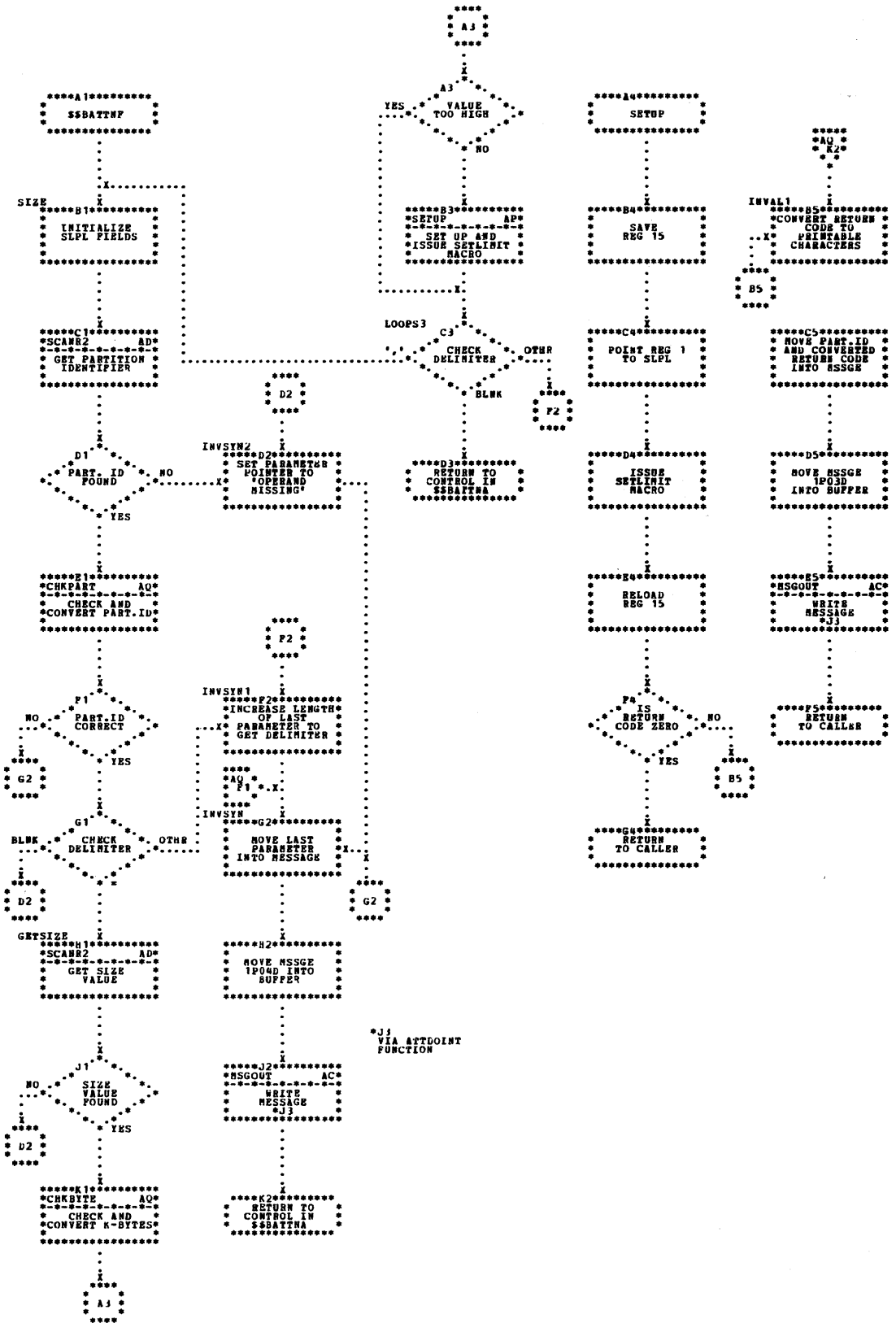


Chart AQ. $$BATTNF - SIZE Command Processor (Part 2 of 2)
(Refer to Chart 04)

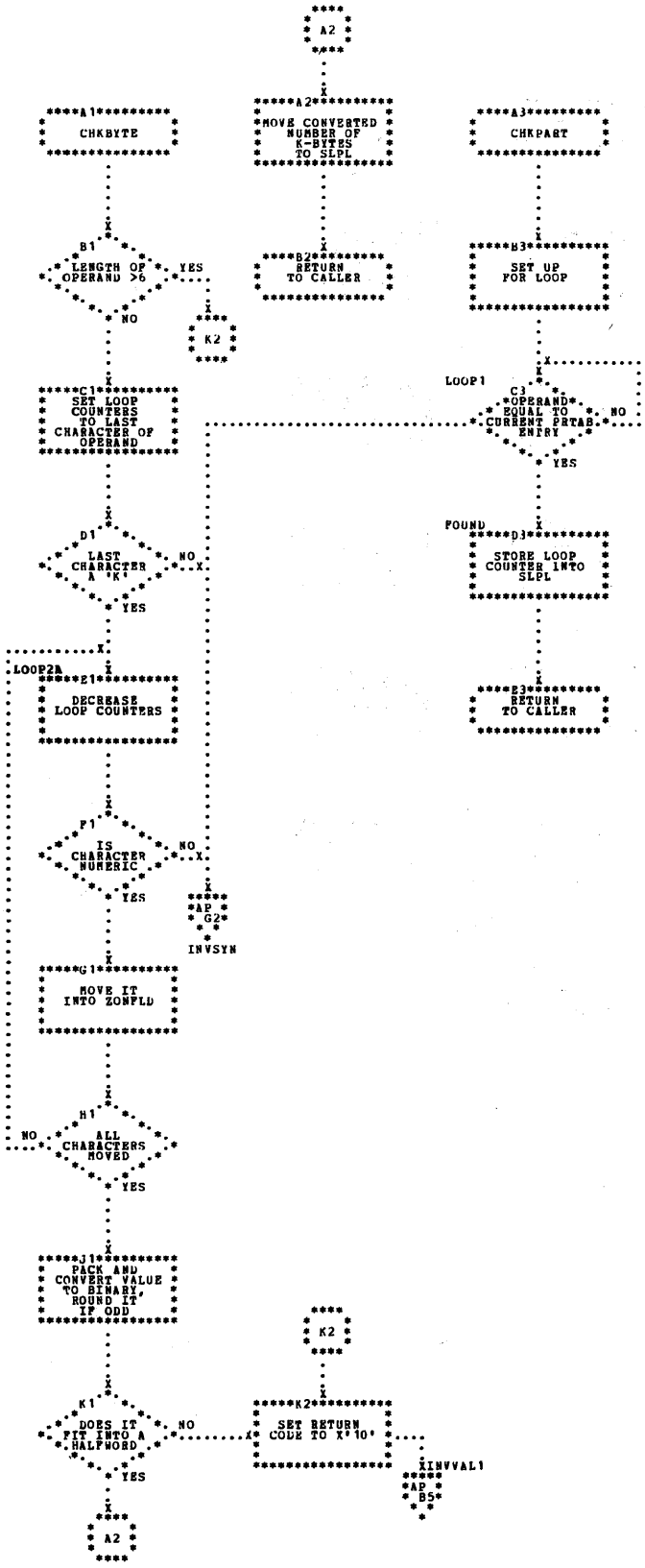


Chart AR. $$BATTNG - BATCH and START Command Processor (Part 1 of 2)
(Refer to Chart 04)

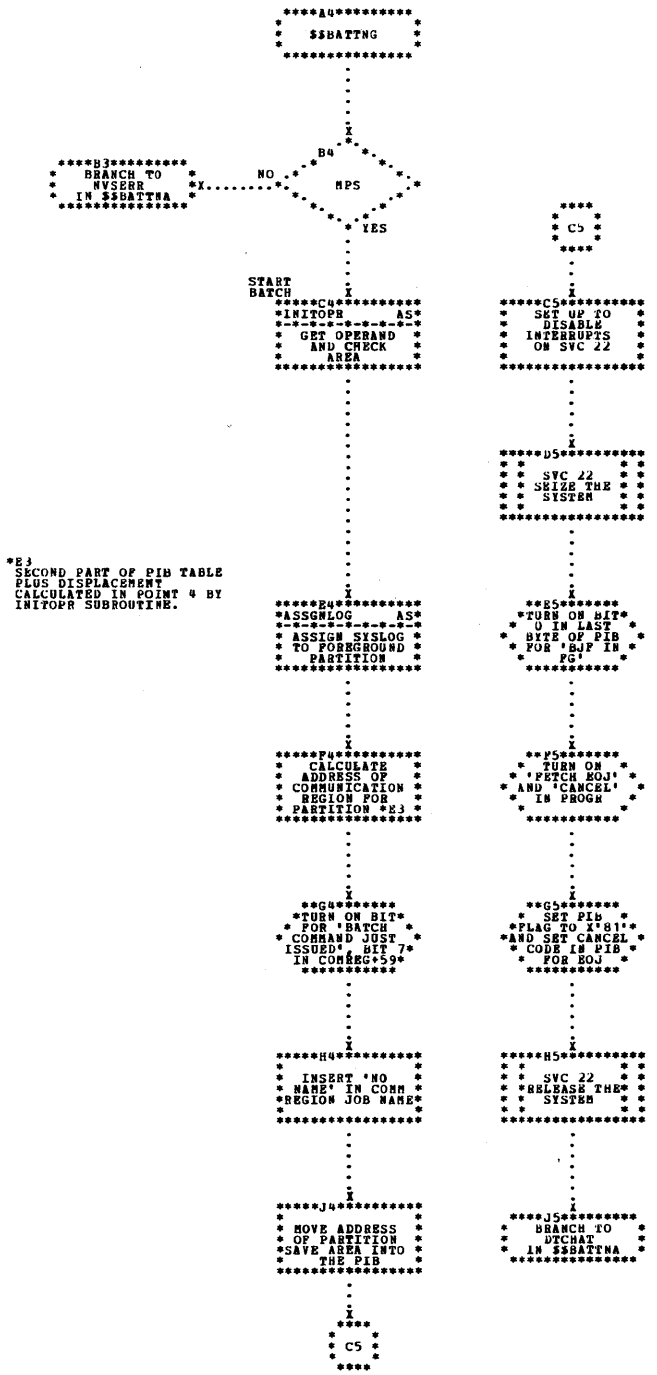


Chart AS. $$BATTNG - BATCH and START Command Processor (Part 2 of 2)
(Refer to Chart 04)

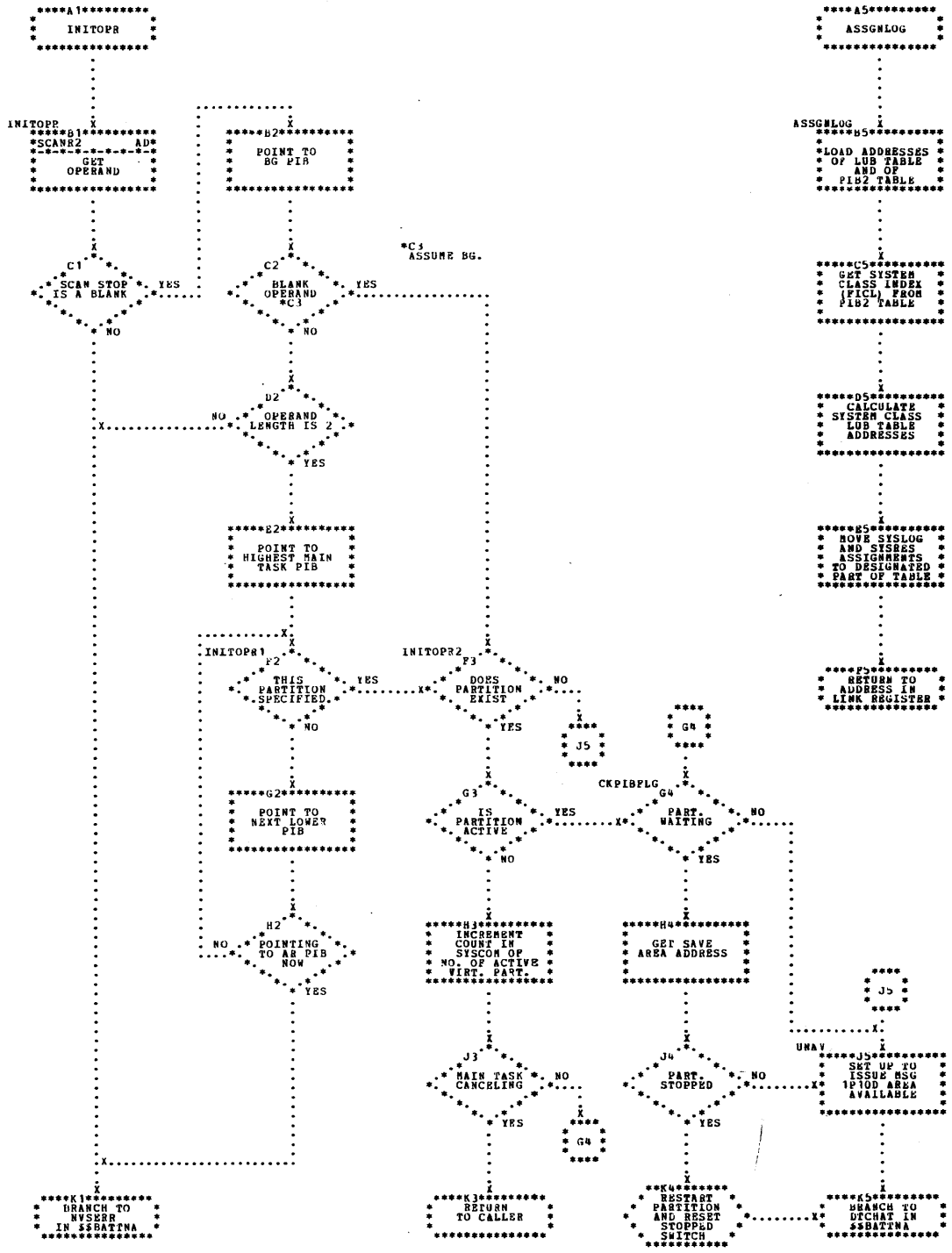


Chart AT. $$BATTNH - Command Scanner
(Refer to Chart 02)

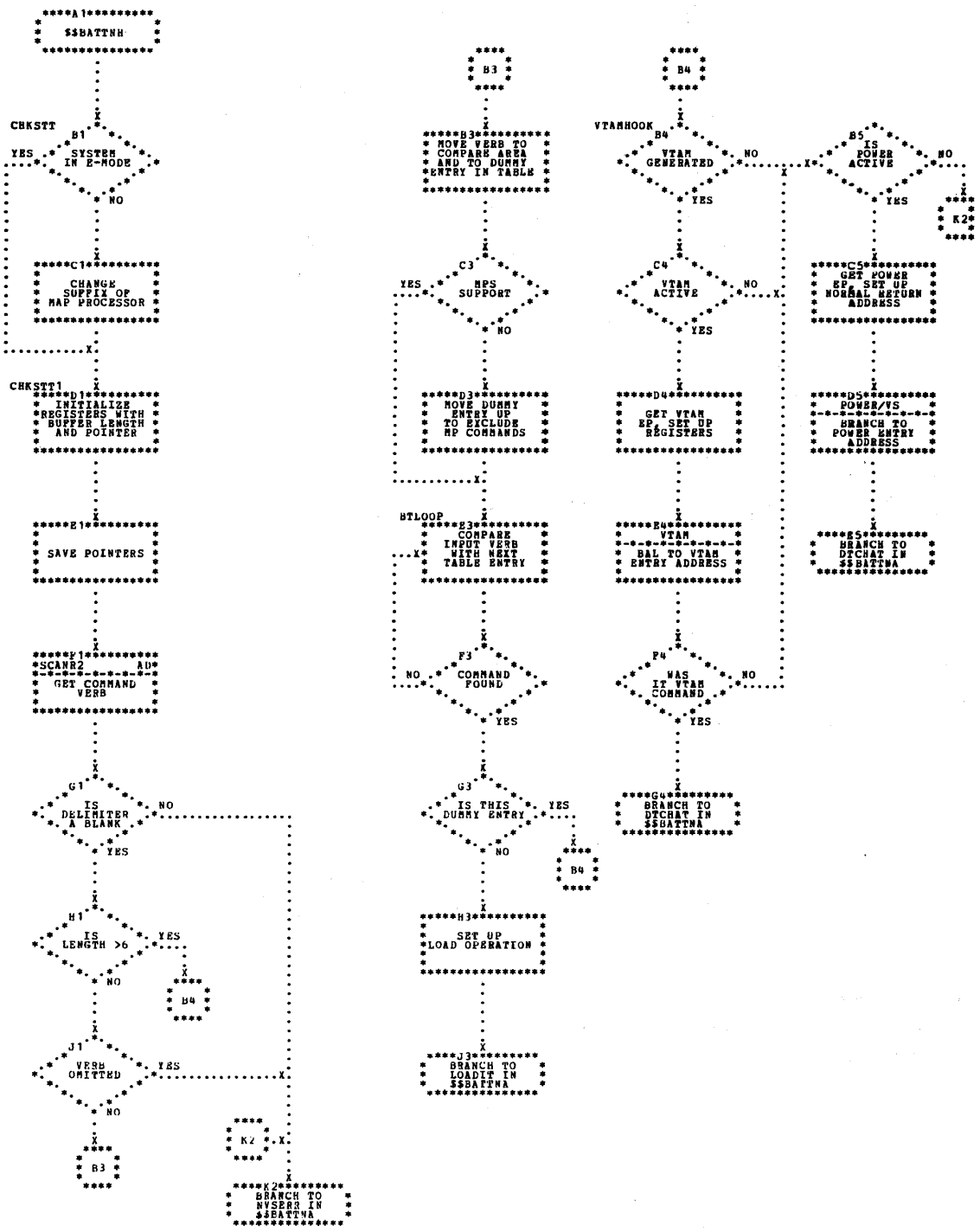


Chart AU. $$BATTNO - VOLUME Command Processor
(Refer to Chart 04)

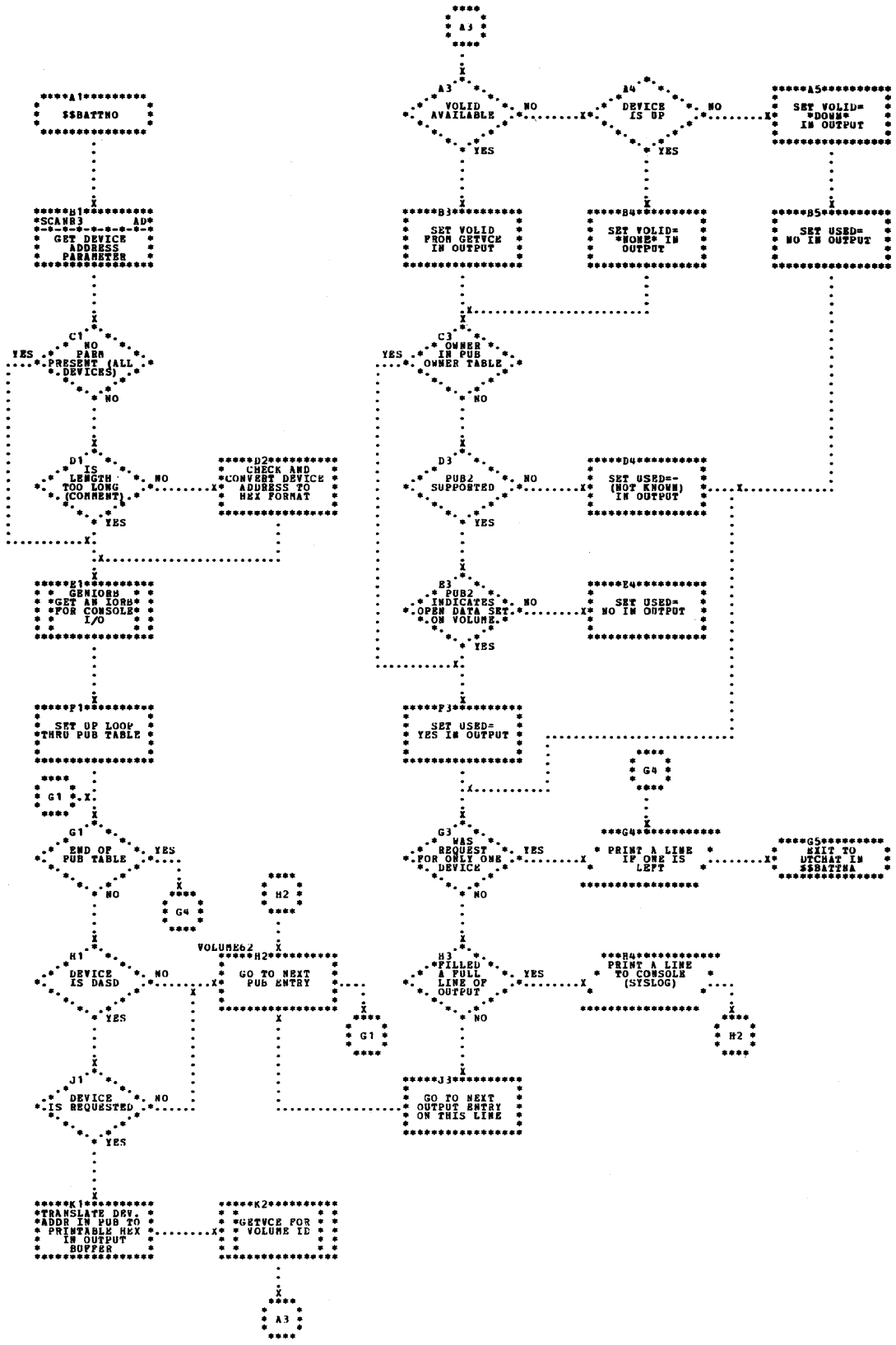


Chart AV. $$BATTNQ - MODE Command Parameter Processor
(Refer to Chart 05)

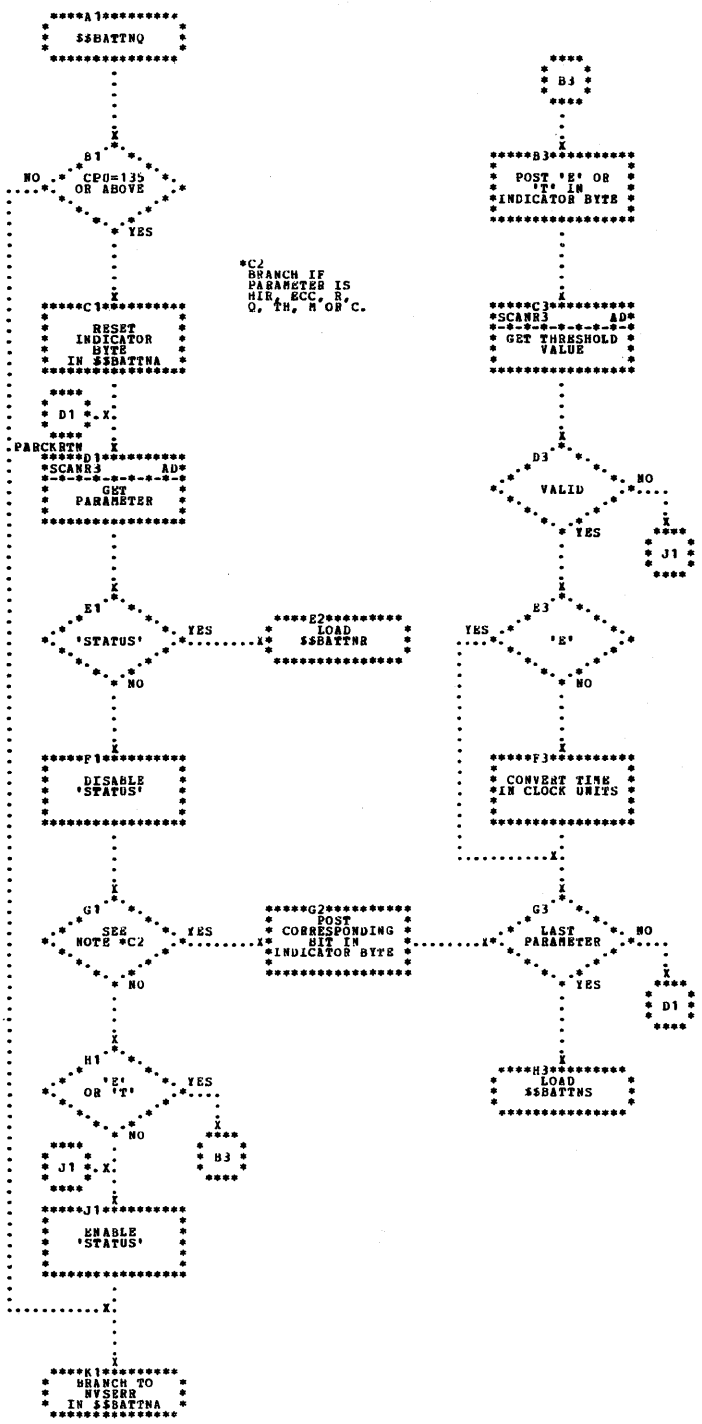


Chart AW. $$BATTNR - MODE Command Status Report Processor
(Refer to Chart 05)

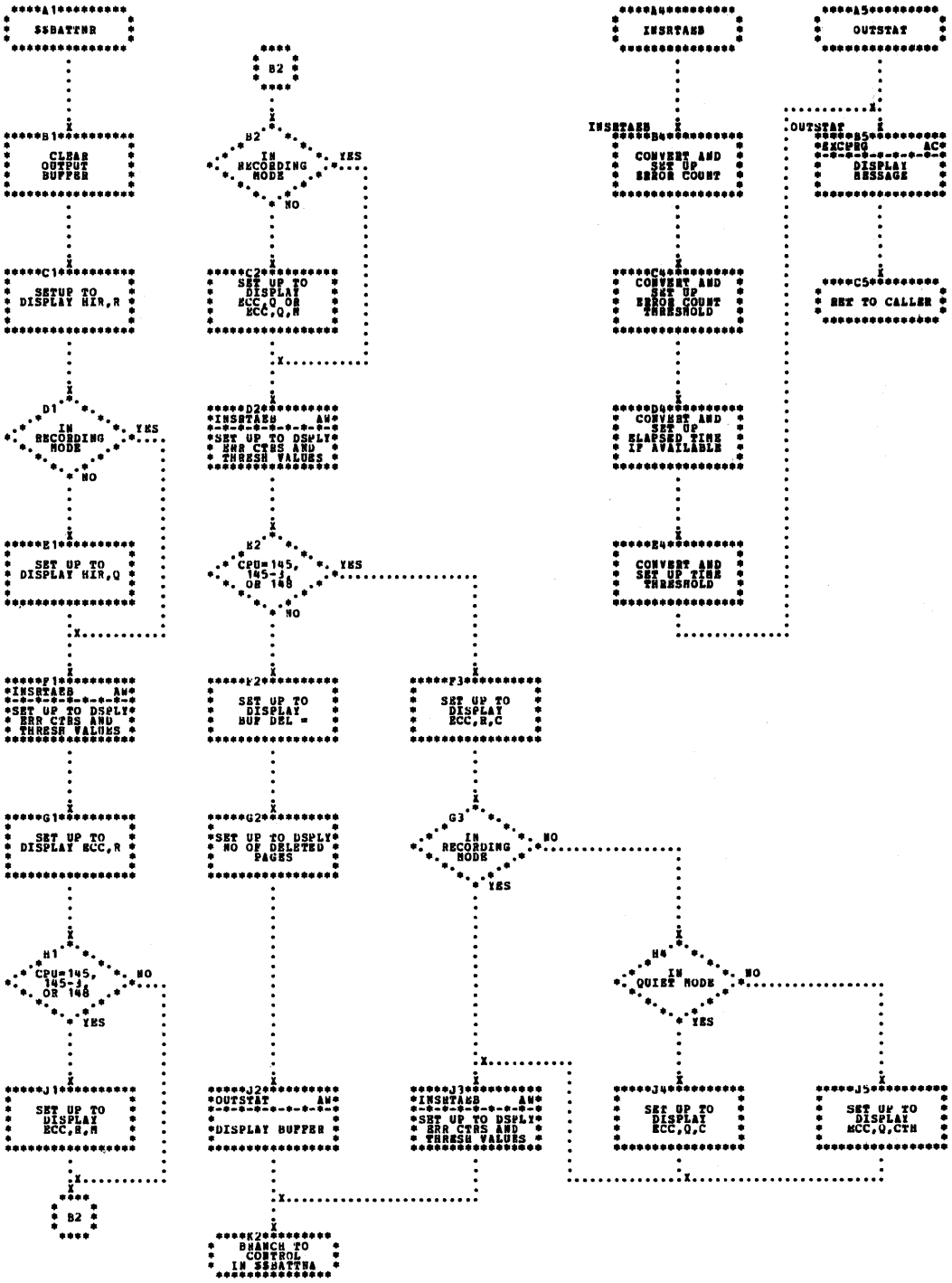


Chart AX. $$BATTNS - MODE Command Validity Checker (Part 1 of 2)
(Refer to Chart 05)

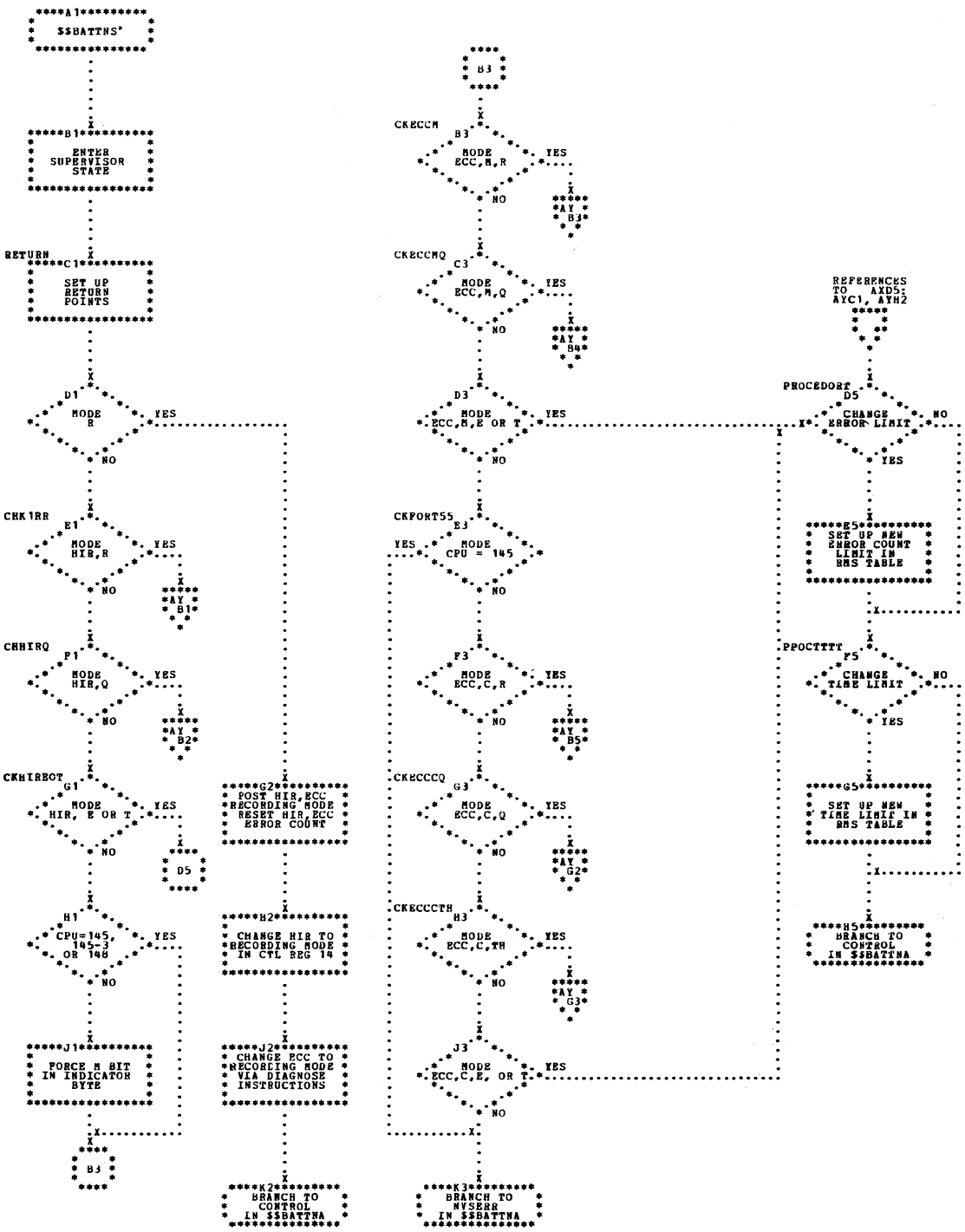


Chart AY $$BATTNS - MODE Command Validity Checker (Part 2 of 2)
(Refer to Chart 05)

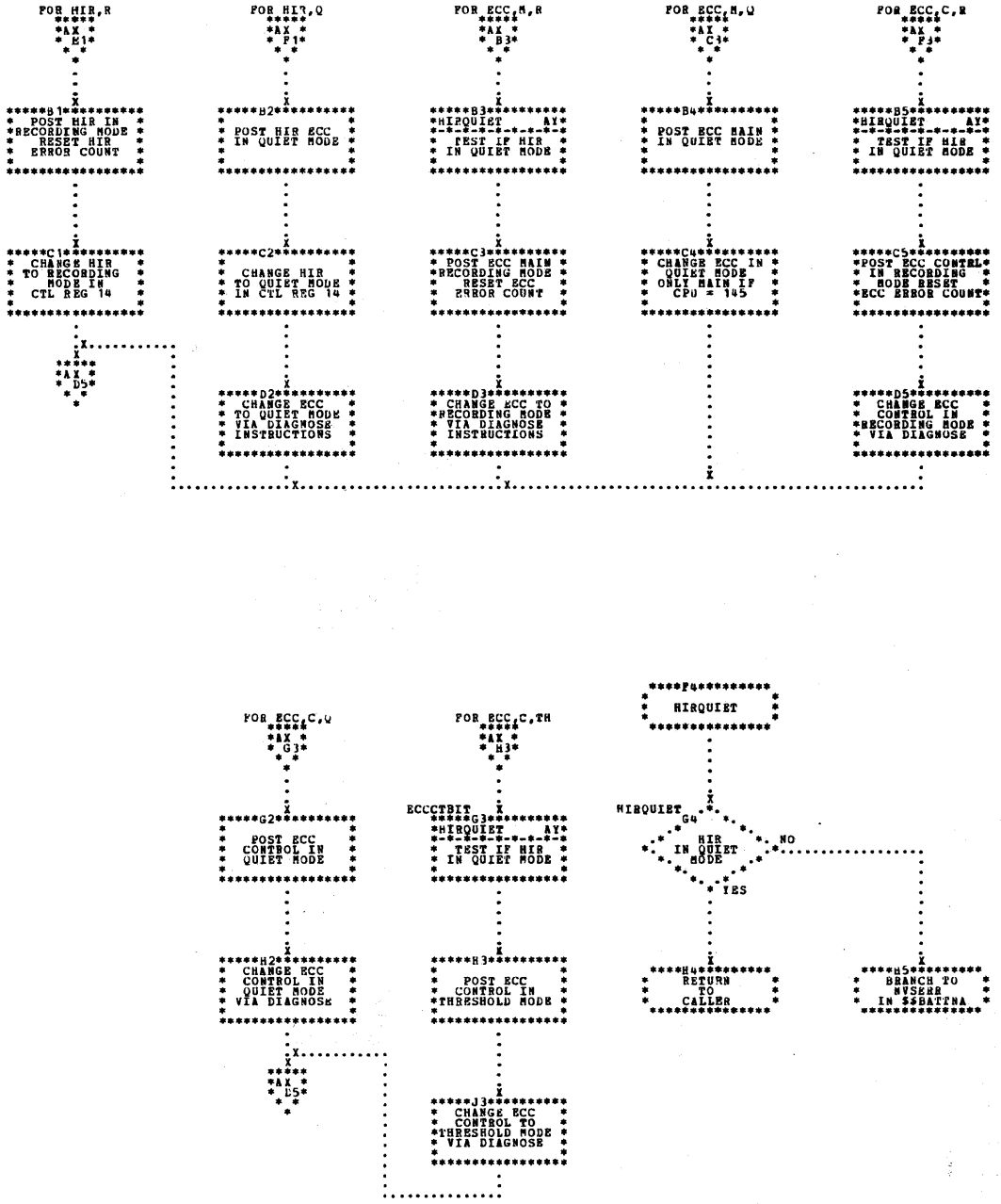


Chart BA. $$BATTNT - ALTER Command Processor (Part 1 of 2)
(Refer to Chart 06)

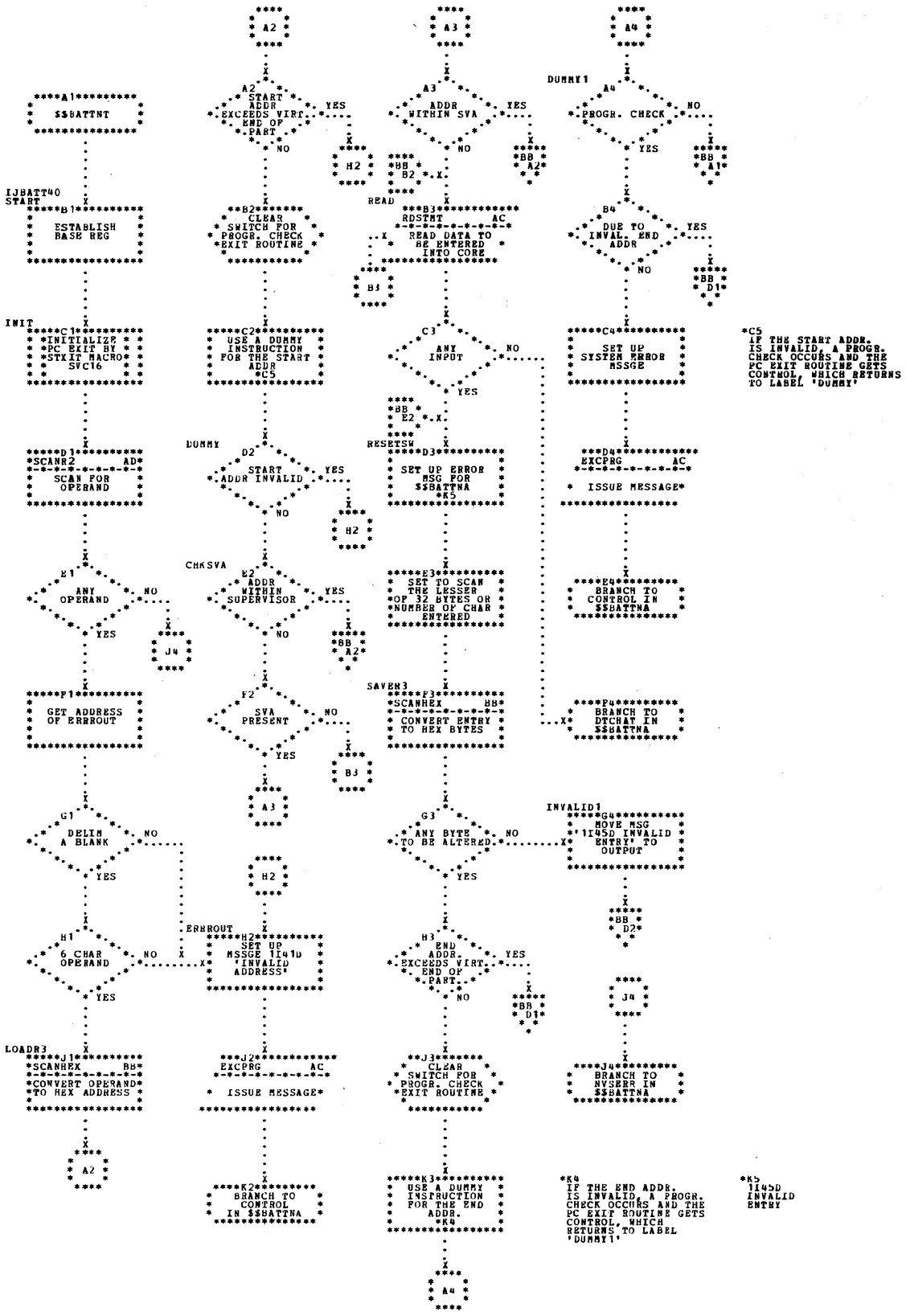


Chart BB. $$BATTNT - ALTER Command Processor (Part 2 of 2)
(Refer to Chart 06)

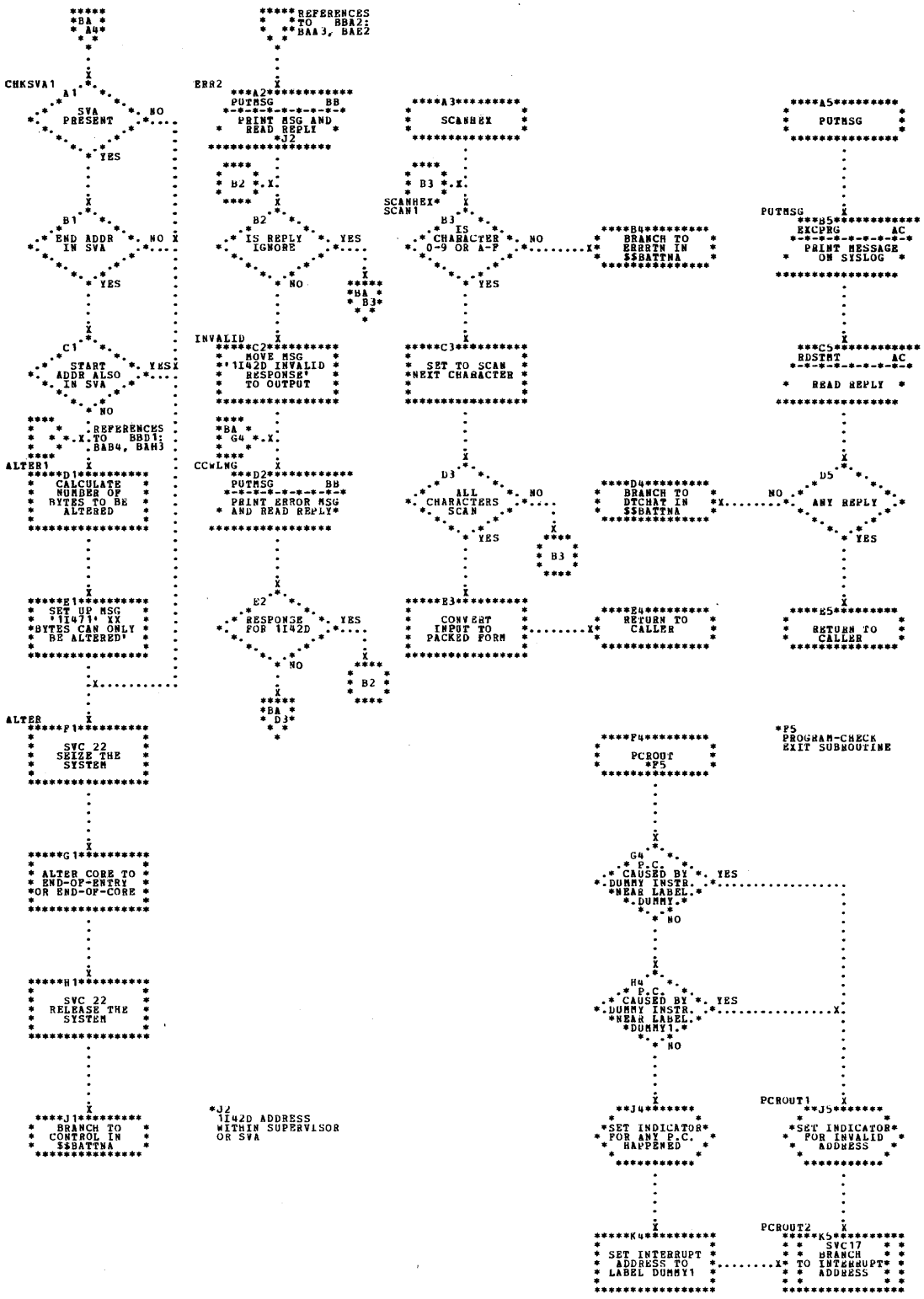


Chart BC. $$BATTNU - DSPLY Command Processor (Part 1 of 2)
(Refer to Chart 06)

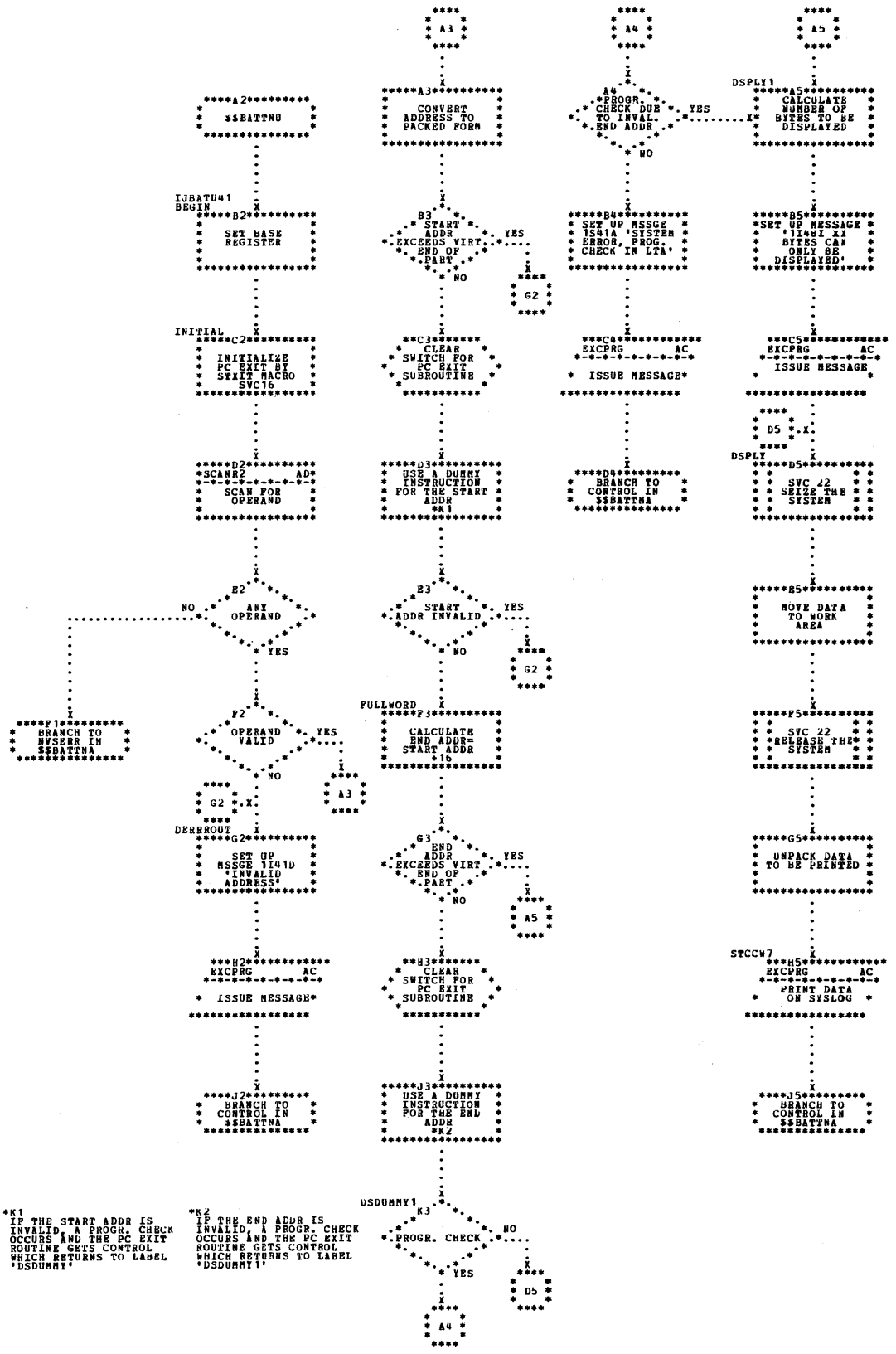


Chart BD. $$BATTNU - DSPLY Command Processor (Part 2 of 2)
(Refer to Chart 06)

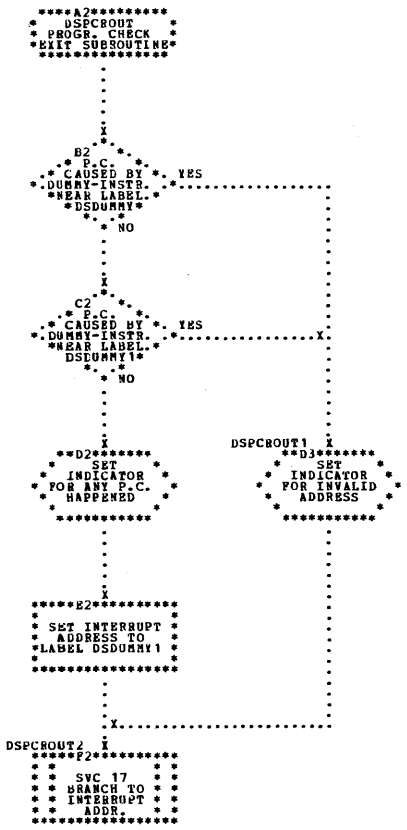


Chart BE. $$BATTNV - DUMP Command Scan Routine (Part 1 of 4)
(Refer to Chart 07)

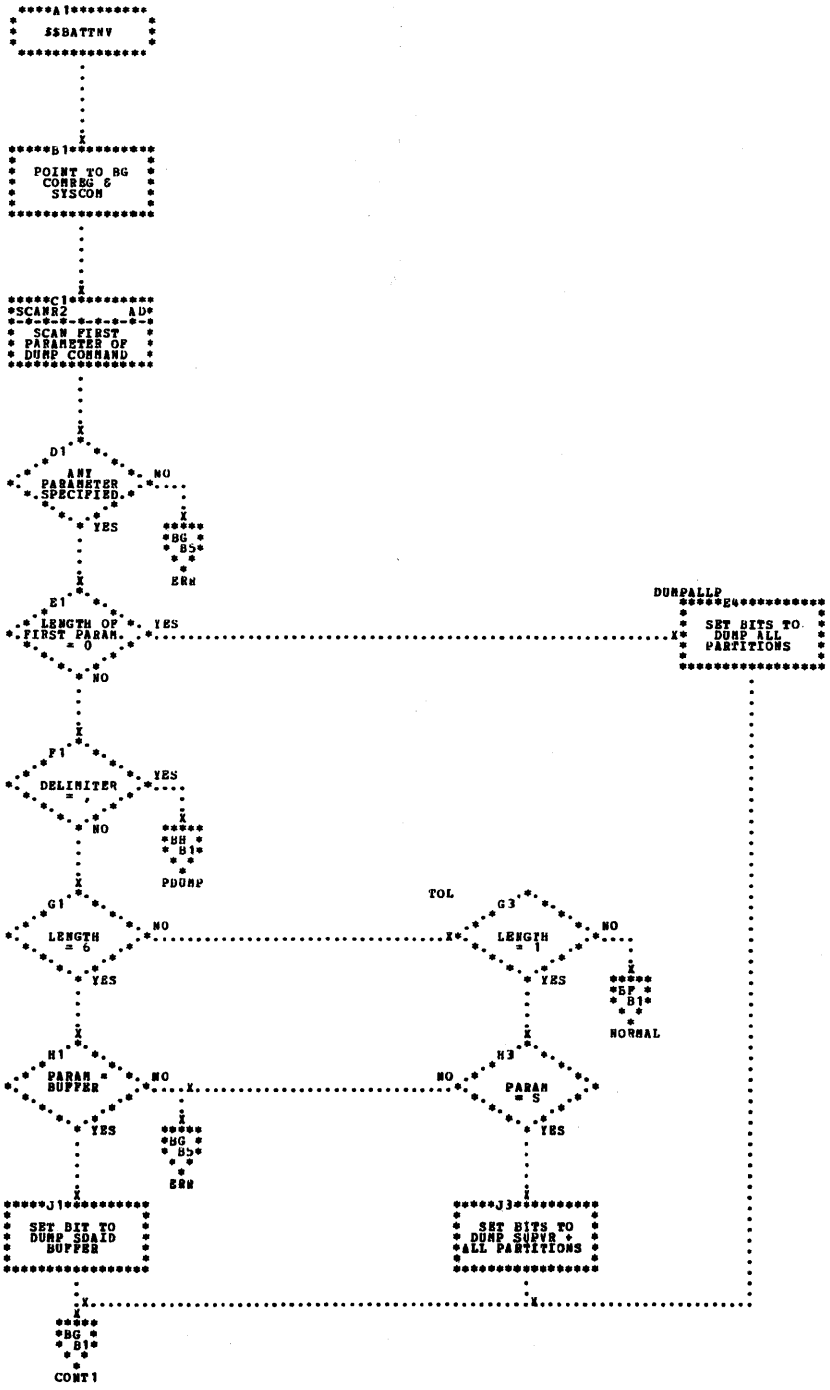


Chart BF. $$BATTNV - DUMP Command Scan Routine (Part 2 of 4)
(Refer to Chart 07)

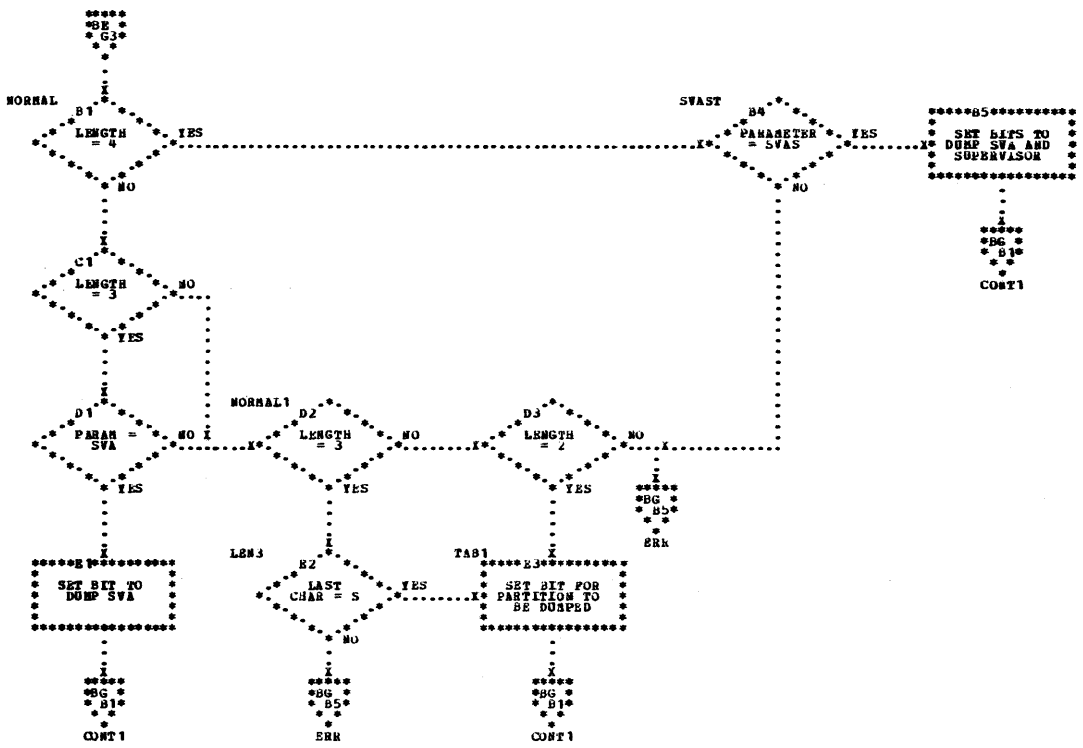


Chart BG. $$BATTNV - DUMP Command Scan Routine (Part 3 of 4)
(Refer to Chart 07)

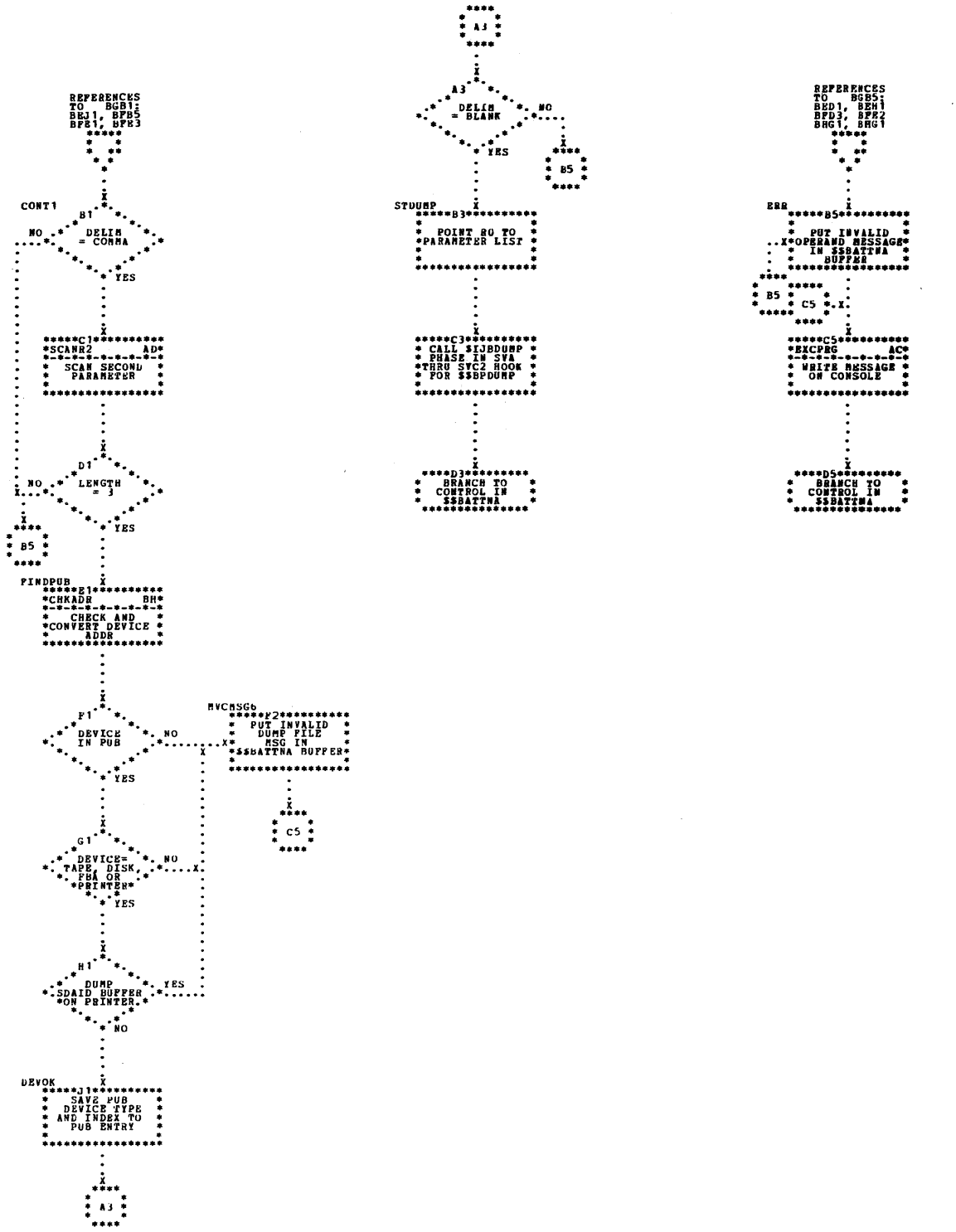


Chart BH. $$BATTNV - DUMP Command Scan Routine (Part 4 of 4)
(Refer to Chart 07)

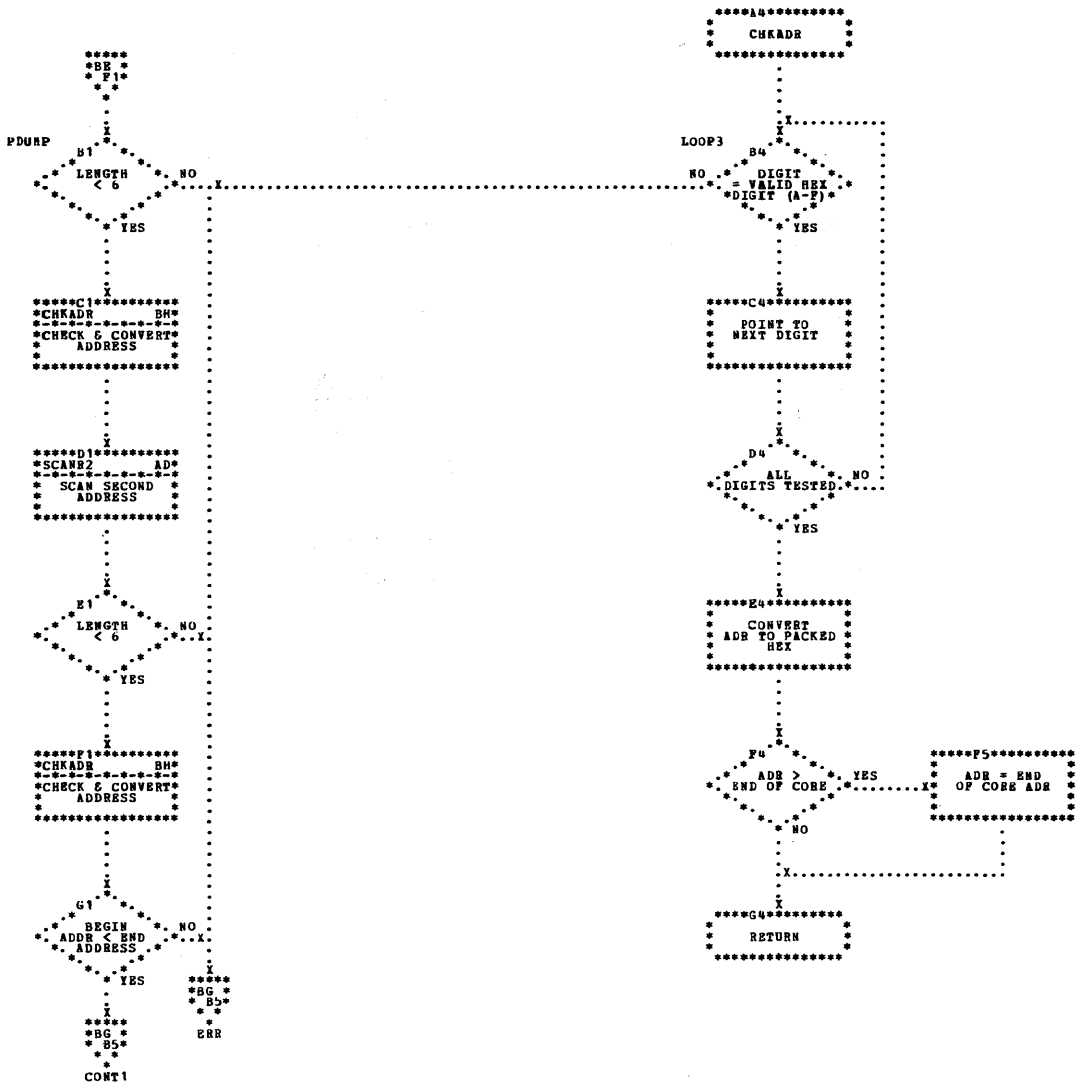


Chart BJ. $$BATTNY - CE MODE Command Processor
(Refer to Chart 08)

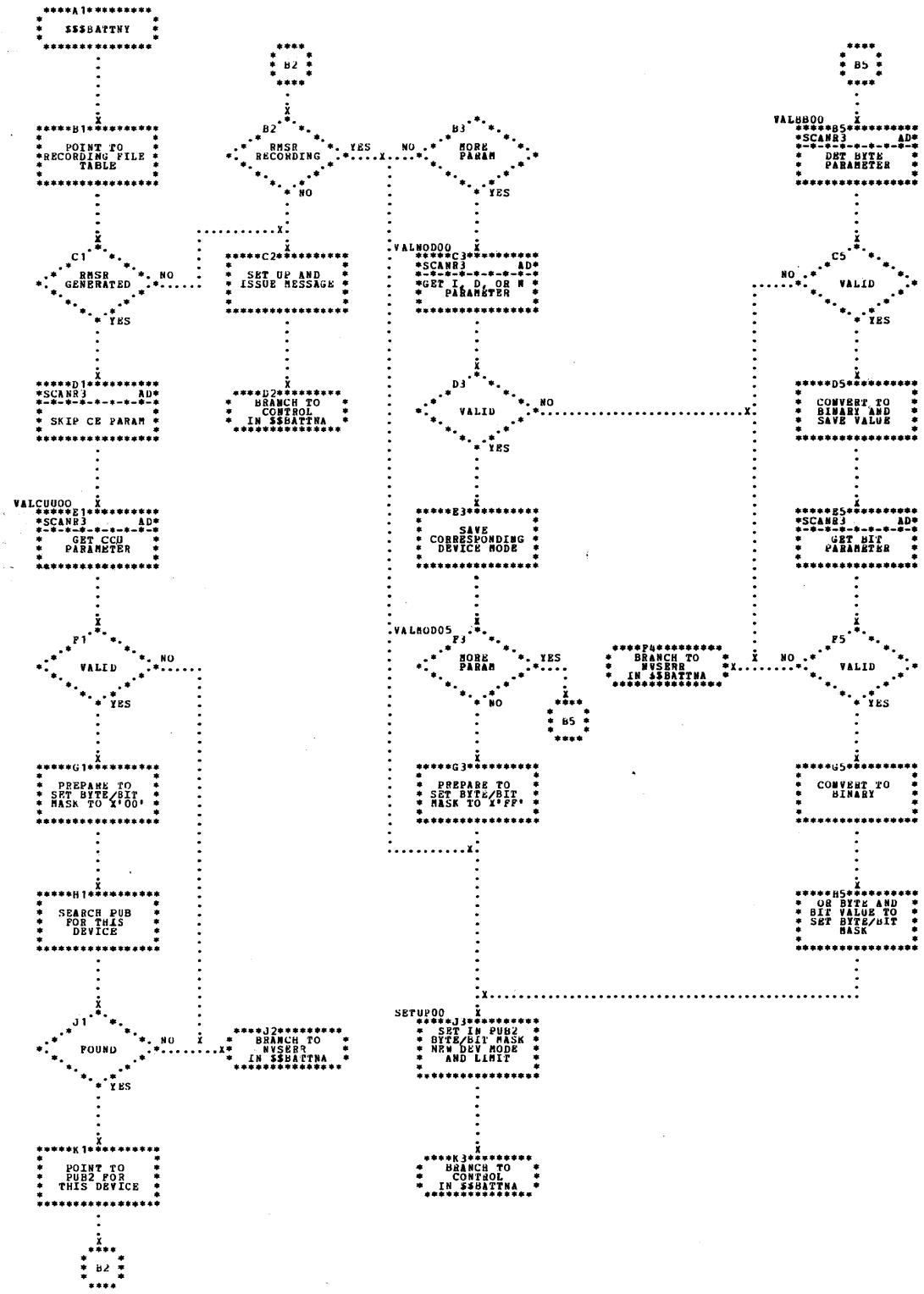


Chart BK. $$BATTNZ - 115/125/135/138 or 4300 MODE Command Processor
(Refer to Chart 08)

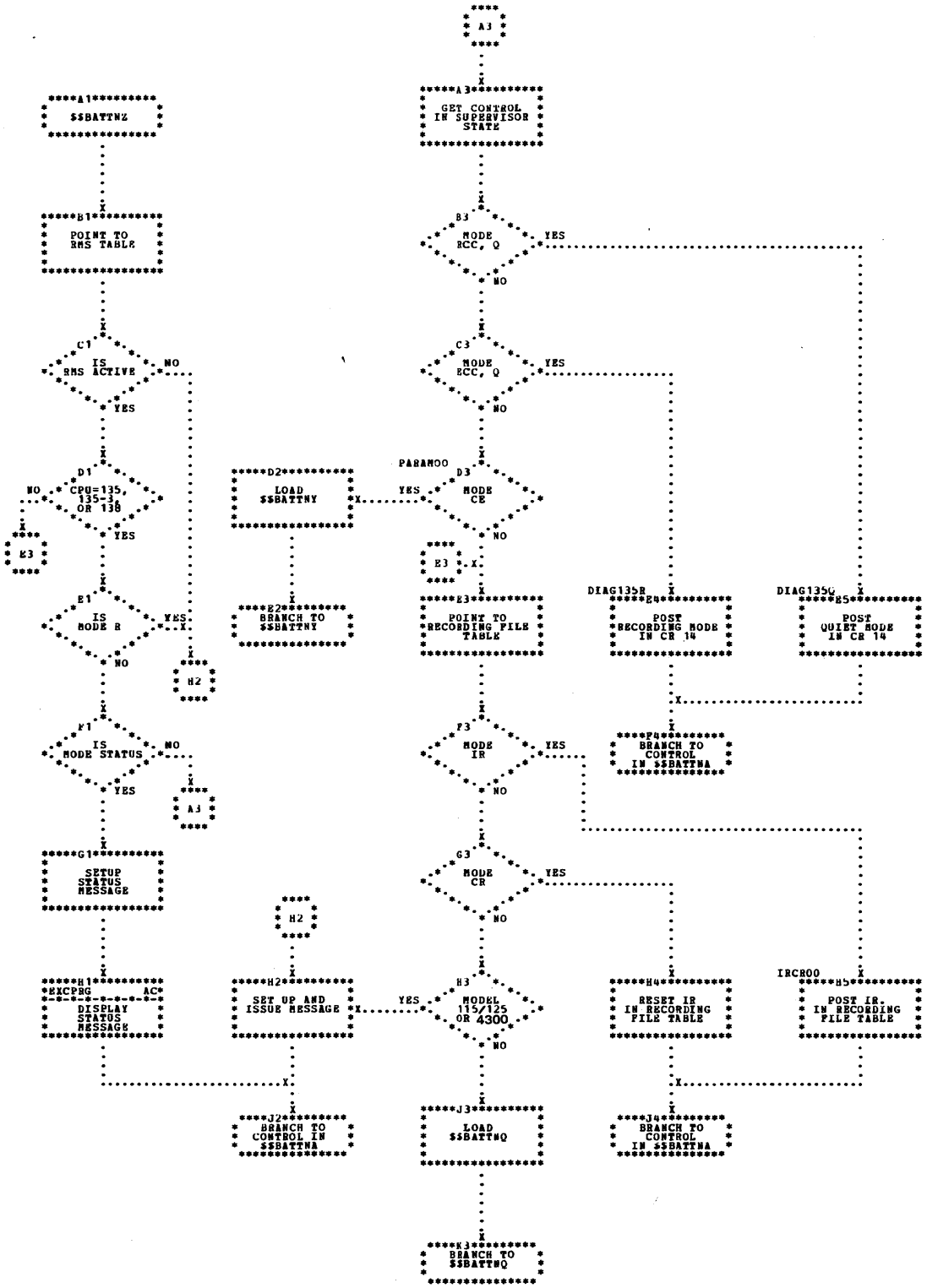


Chart BL. $$BATTN2 - PRTY and TPBAL Command Processor (Part 1 of 3)
(Refer to Chart 04)

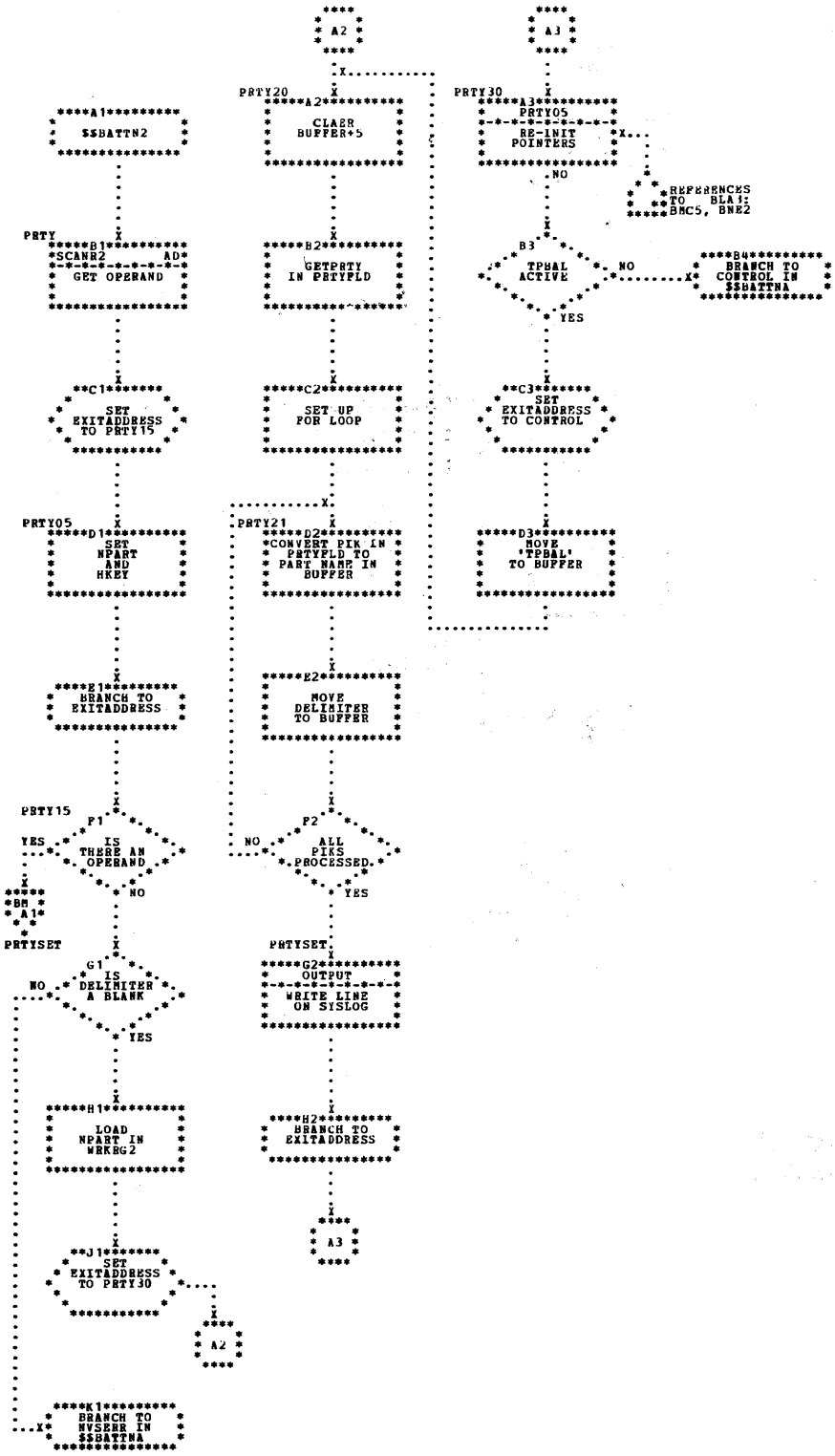


Chart BM. $$BATTN2 - PRTY and TPBAL Command Processor (Part 2 of 3)
(Refer to Chart 04)

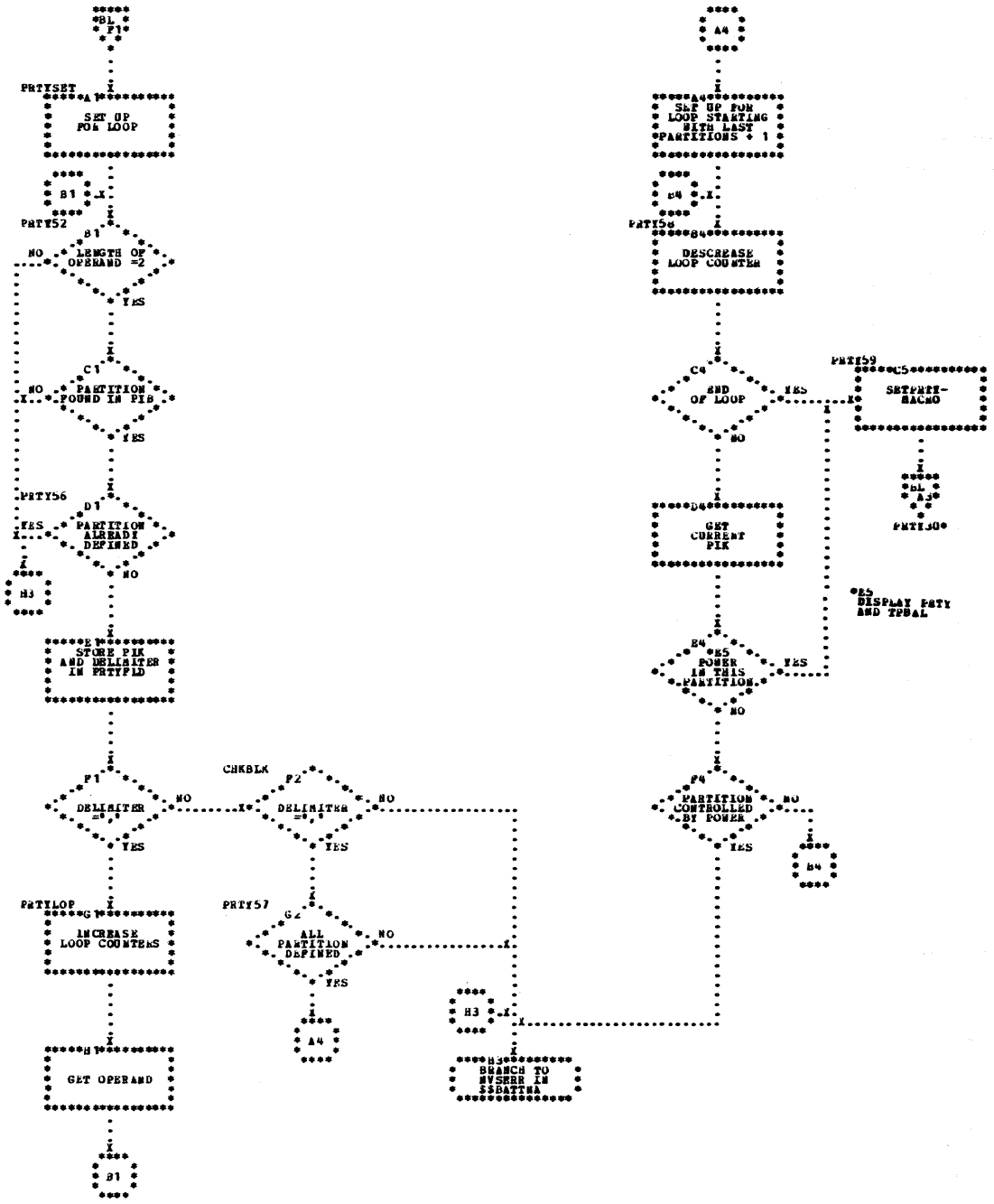


Chart BN. $$BATTN2 - PRTY and TPBAL Command Processor (Part 3 of 3)
(Refer to Chart 04)

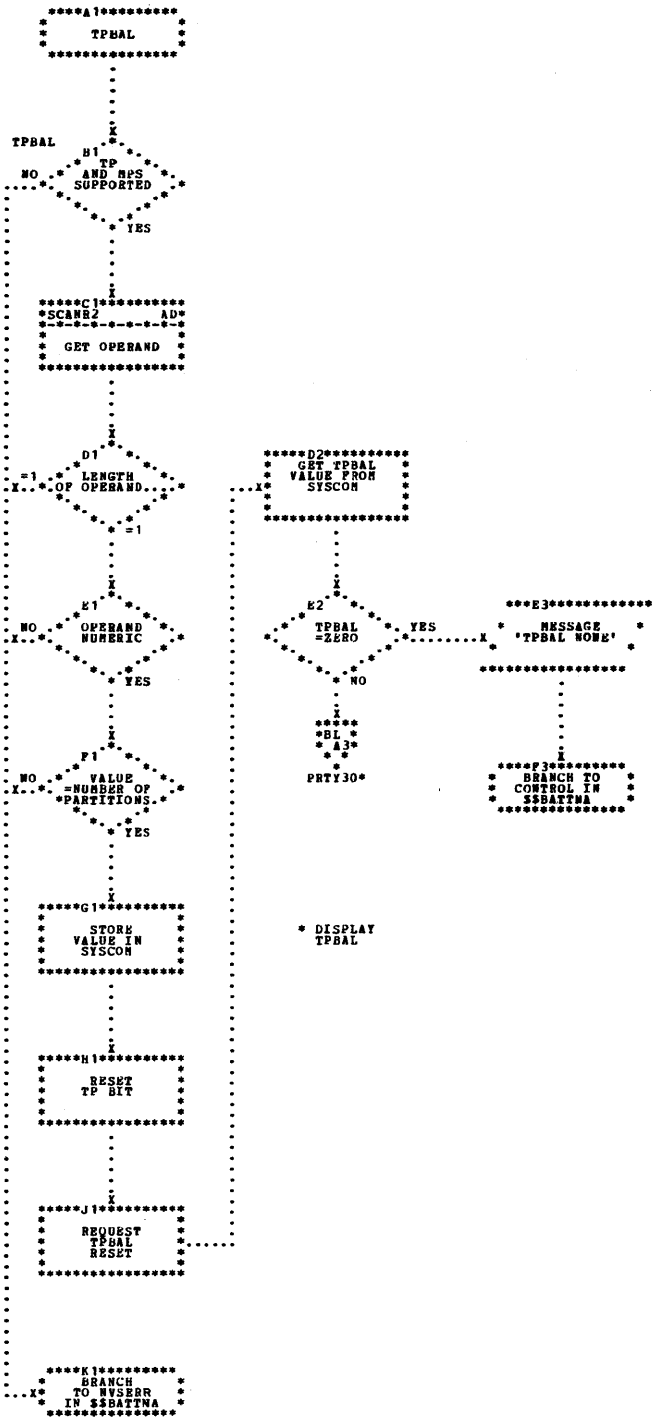


Chart BP. $$BATTN3 - ENDSD Command Processor
(Refer to Chart 04)

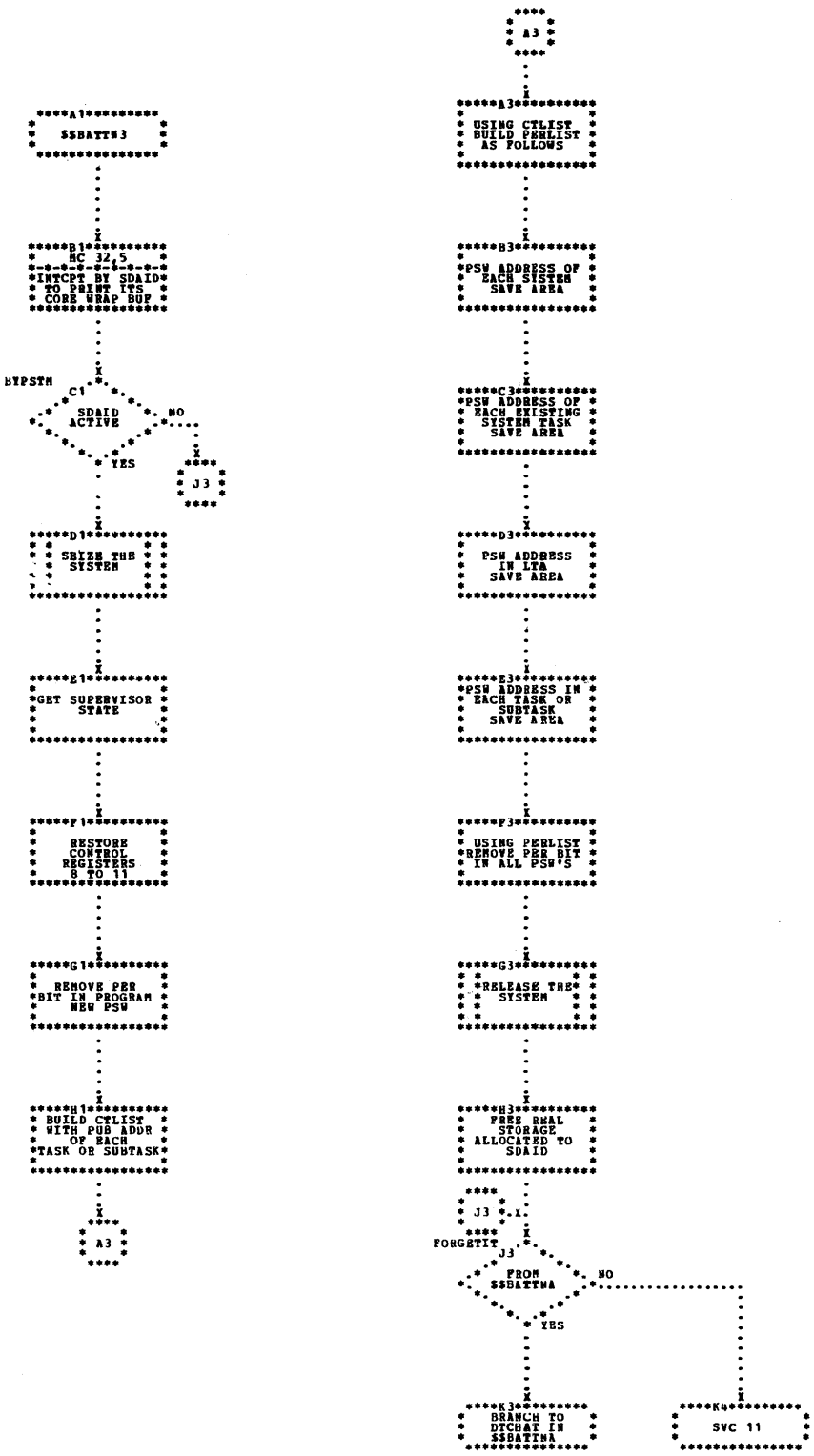


Chart CA. $$BATTN8 - LFCB Command Processor 1 (Part 1 of 2)
(Refer to Chart 09)

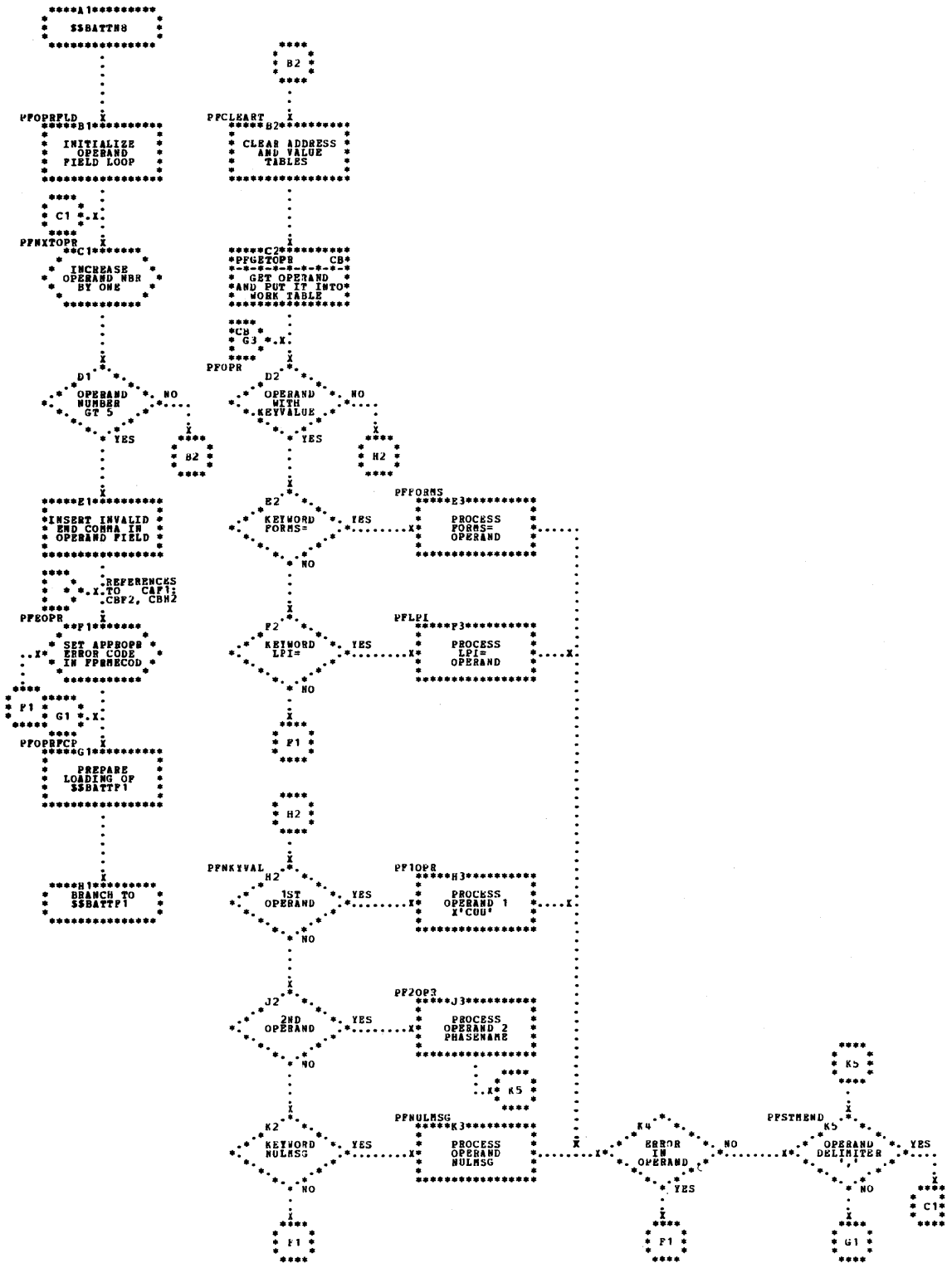


Chart CB. $$BATTN8 - LFCB Command Processor 1 (Part 2 of 2)
(Refer to Chart 09)

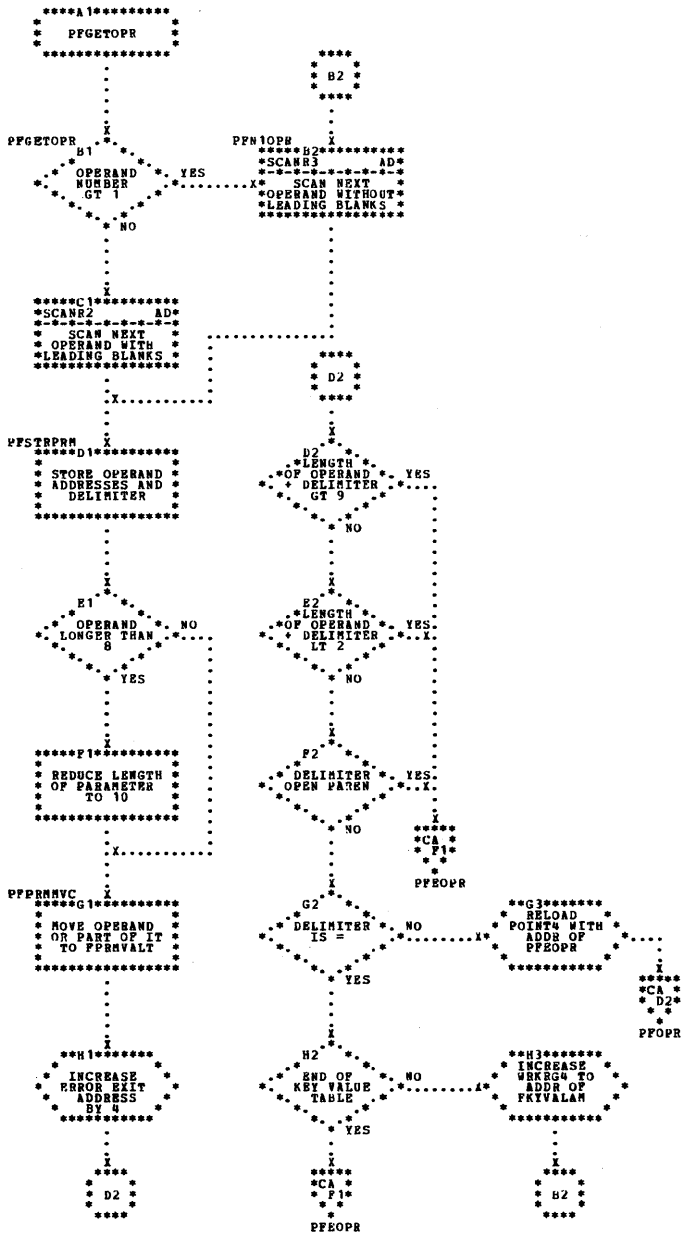


Chart CC. $$BATTF1 - LFCB Command Processor 2
(Refer to Chart 09)

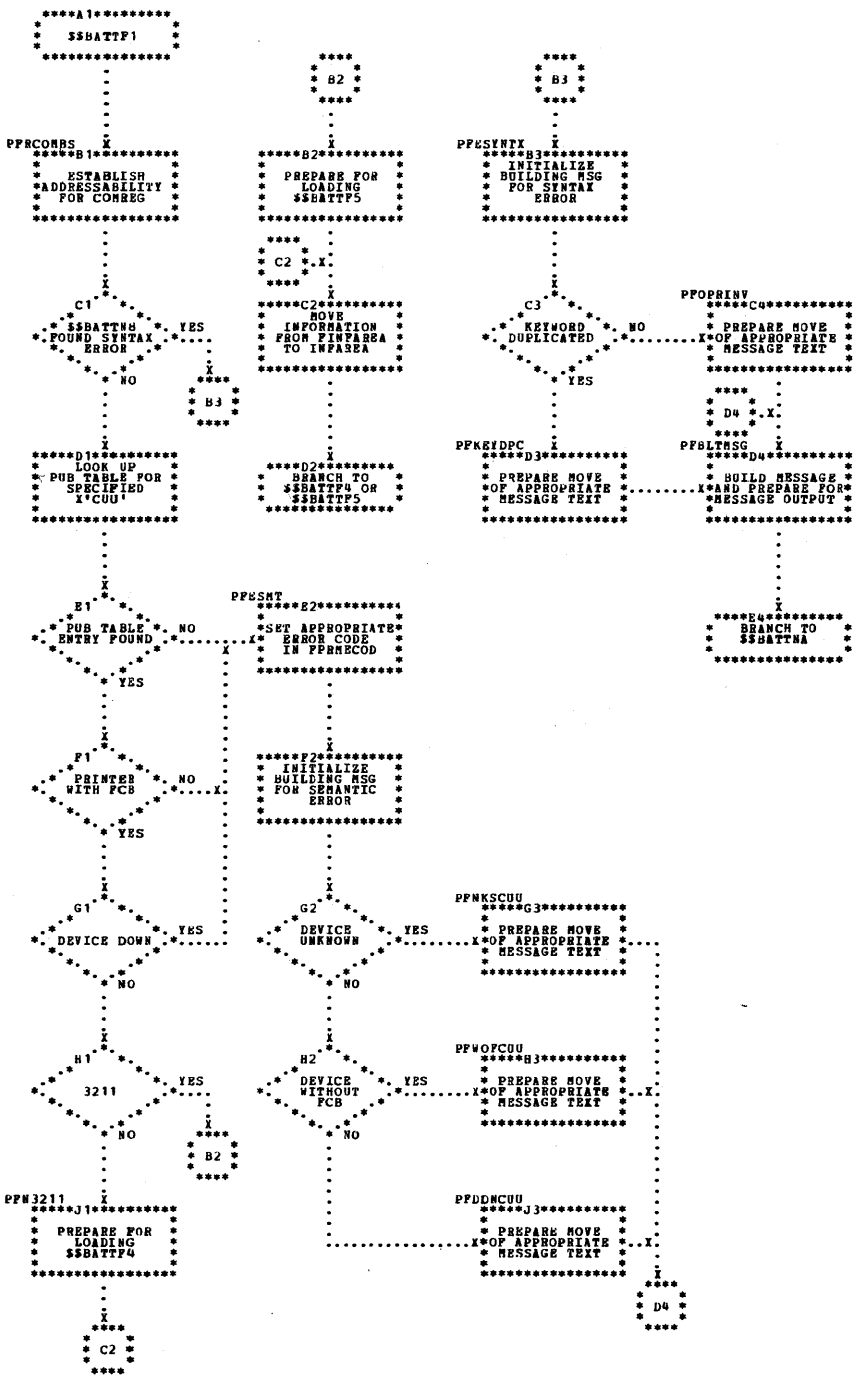


Chart CD. $$BATTF4 - FCB Load Execution for 3203 and 5203 (Part 1 of 2) (Refer to Chart 09)

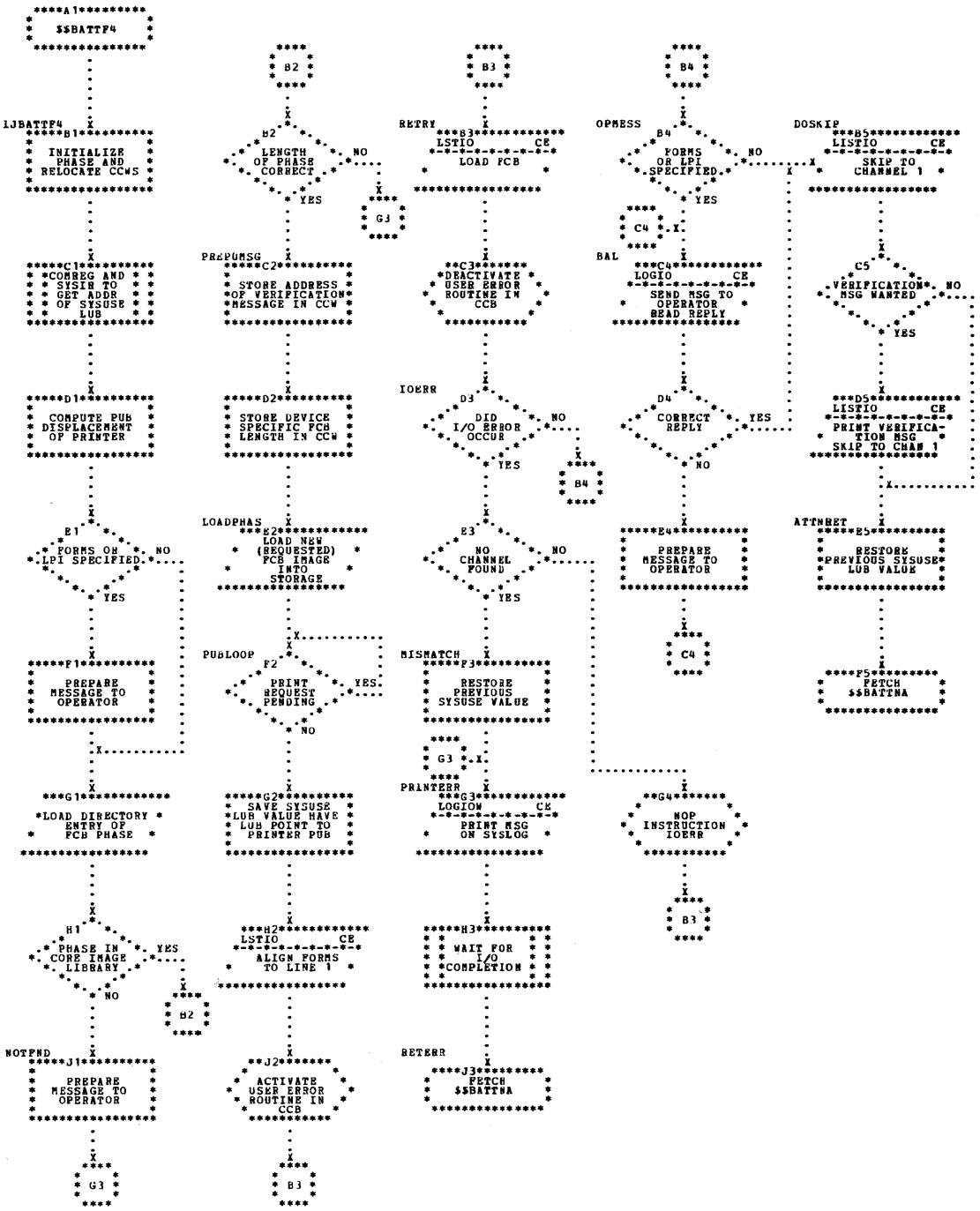


Chart CE. $$BATTF4 - FCB Load Execution for 3203 and 5203 (Part 2 of 2)
(Refer to Chart 09)

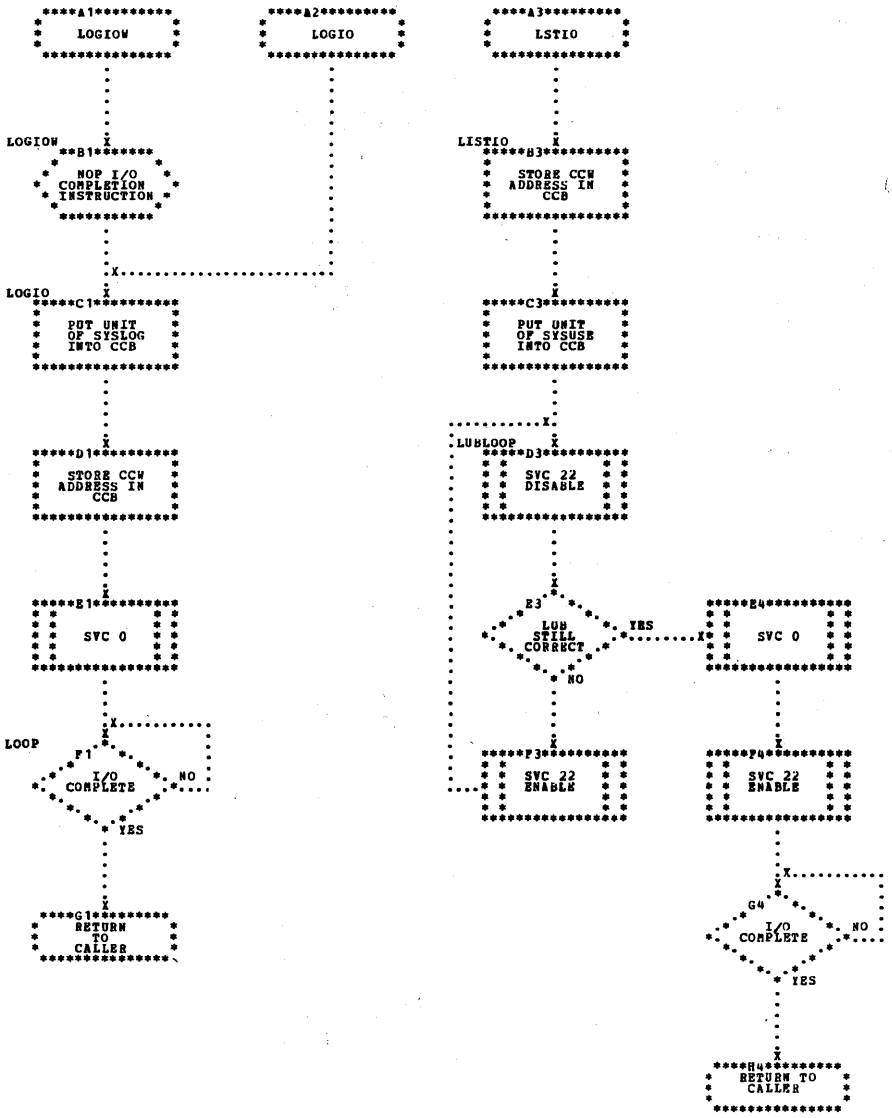


Chart CF. $$BATTF5 - FCB Load Execution for PRT1
(Refer to Chart 09)

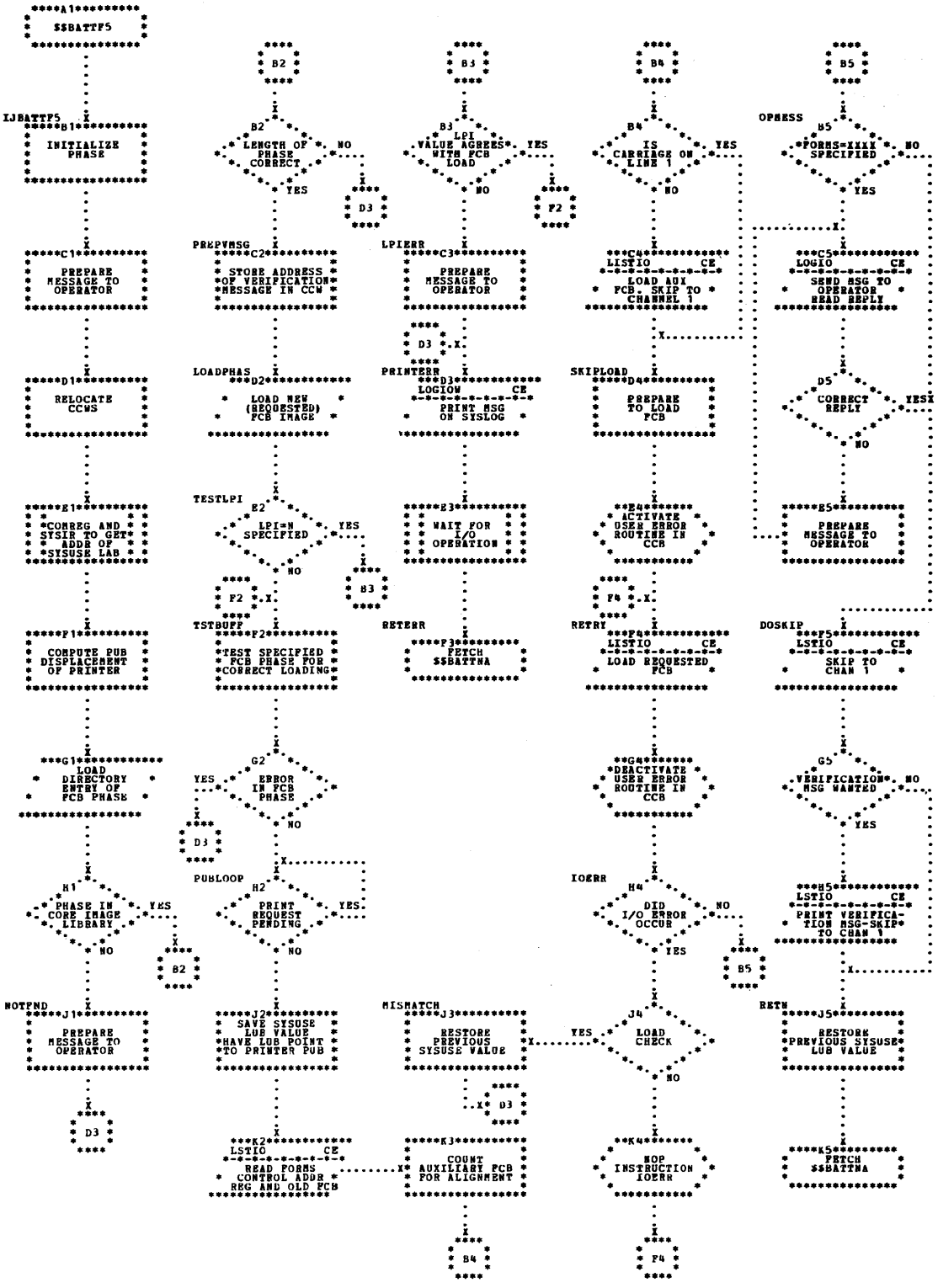


Chart CG. $$BATTN9 - LUCB Command Processor 1 (Part 1 of 2)
(Refer to Chart 10)

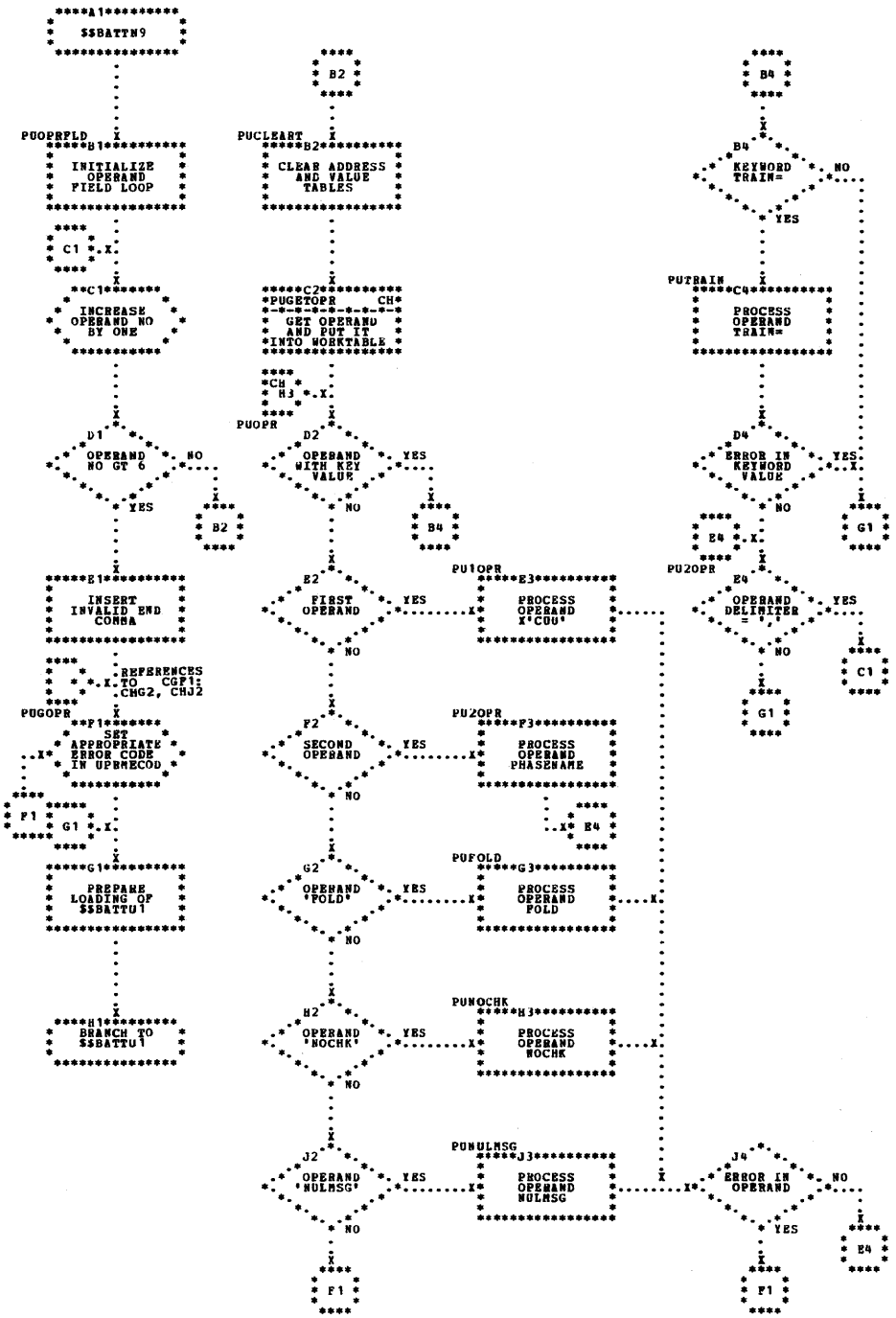


Chart CH. $$BATTN9 - LUCB Command Processor 1 (Part 2 of 2)
(Refer to Chart 10)

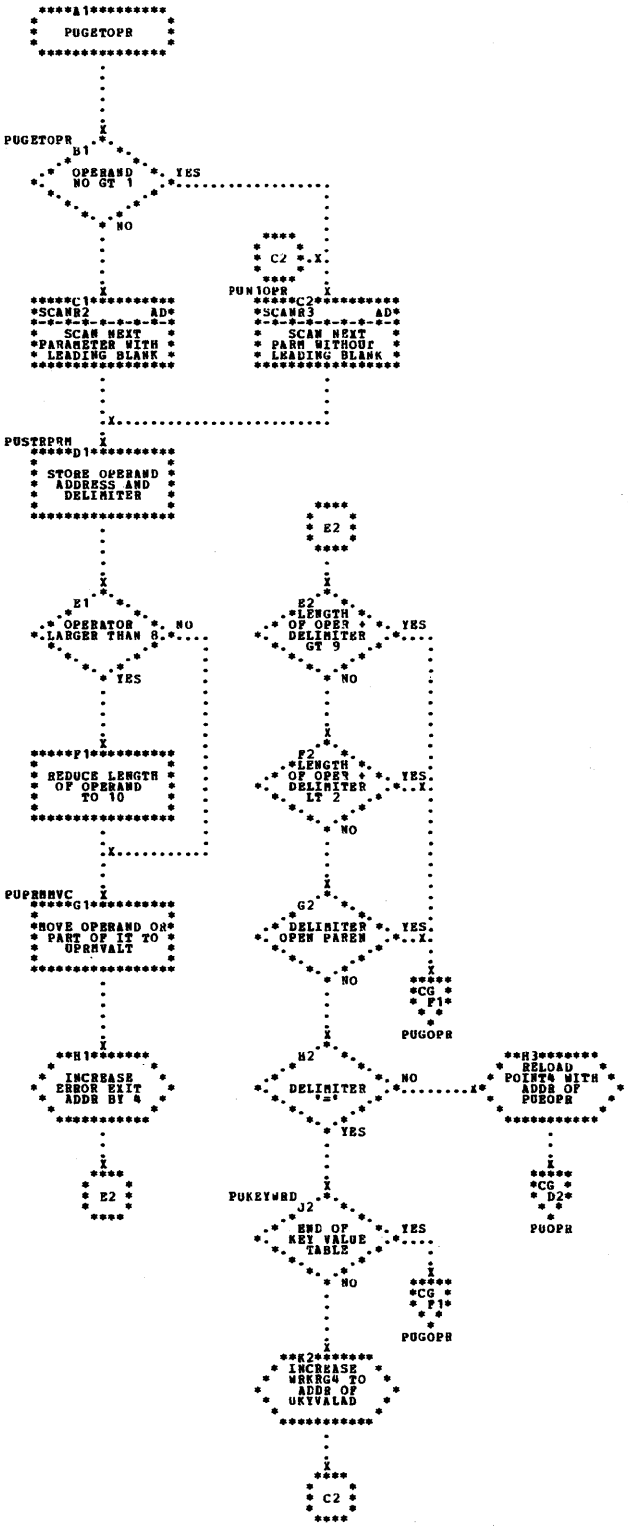


Chart CJ. $$BATTU1 - LUCB Command Processor 2
(Refer to Chart 10)

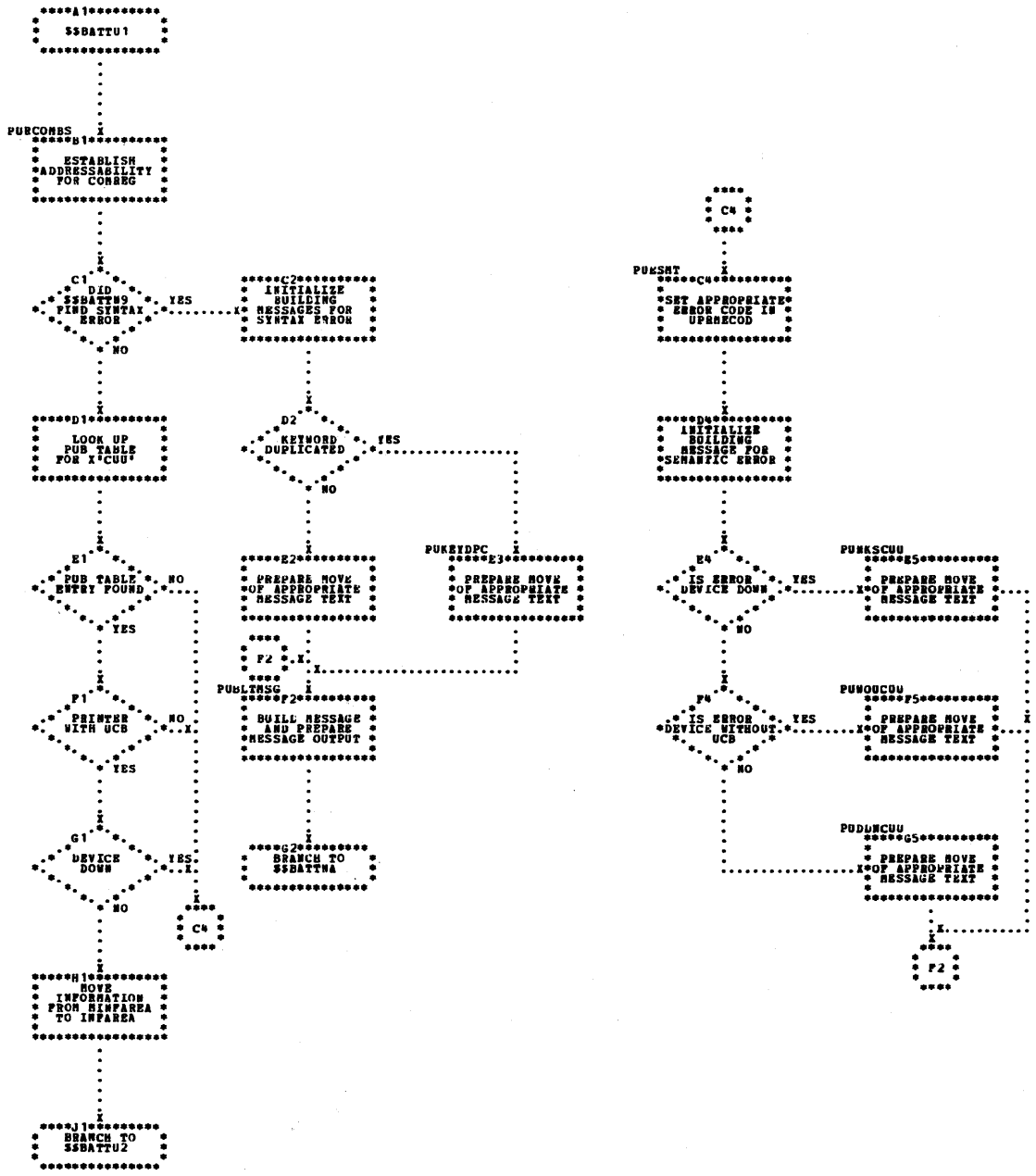


Chart CK. $$BATTU2 - UCB Load Function
(Refer to Chart 10)

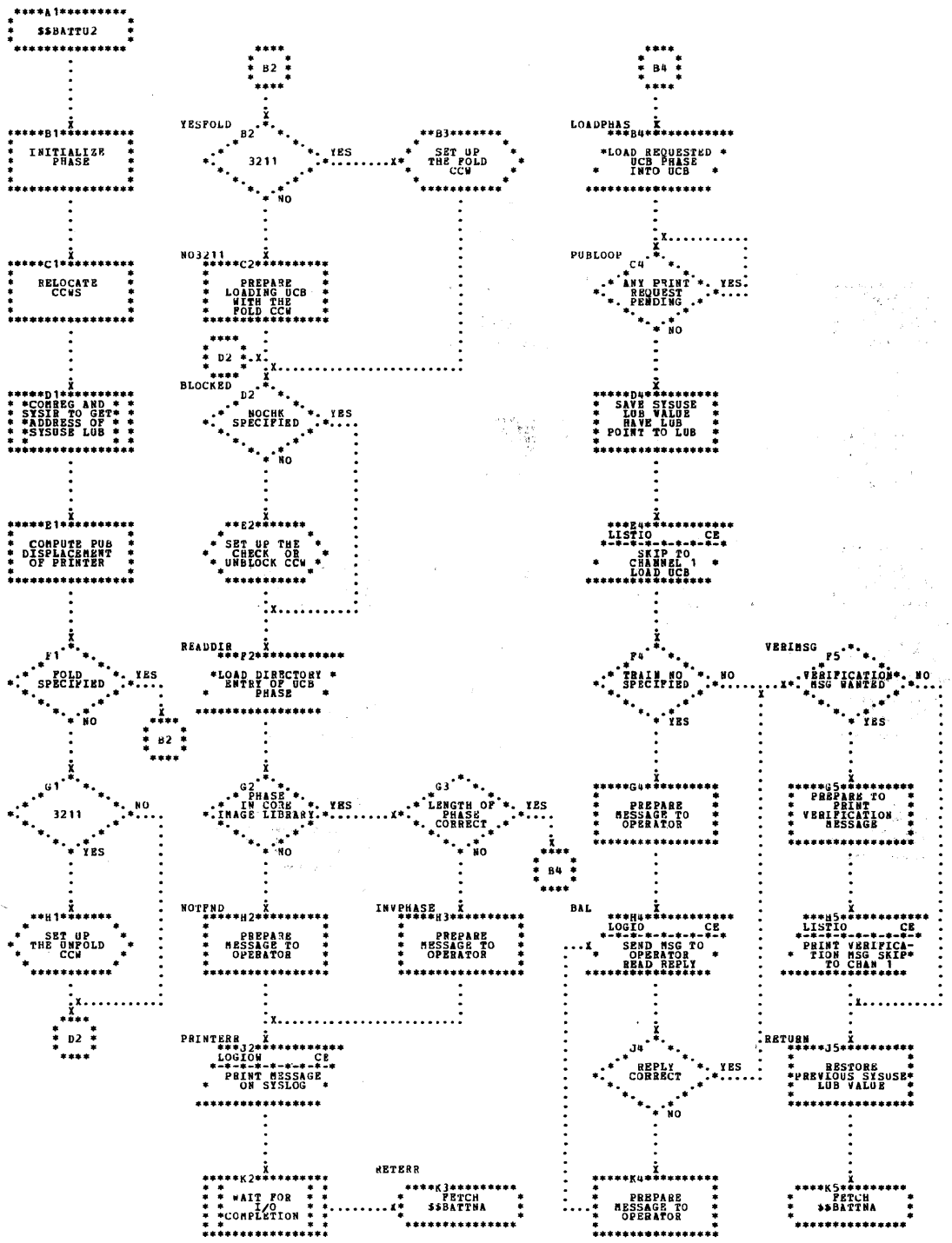


Chart CL. $$BATTN7 - SETDF Root Phase (Part 1 of 6)
(Refer to Chart 11)

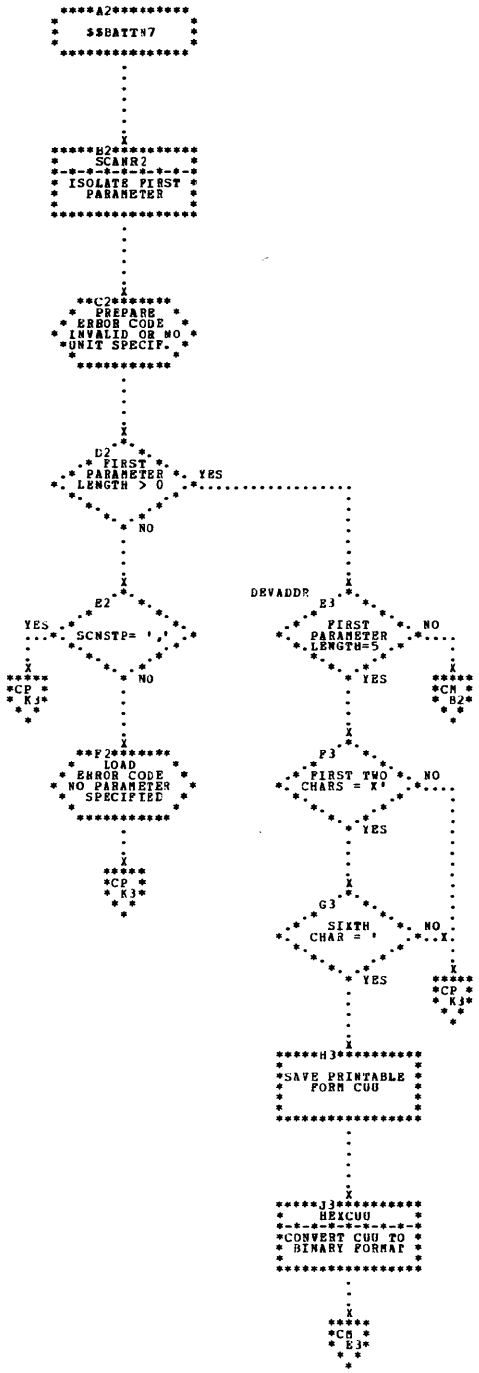


Chart CM. $$BATTN7 - SETDF Root Phase (Part 2 of 6)
(Refer to Chart 11)

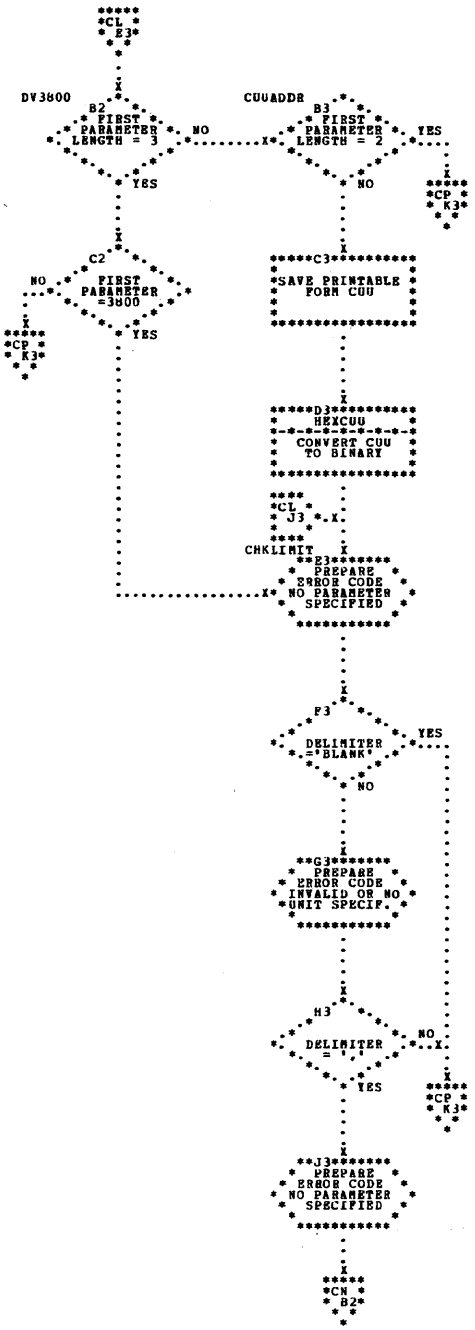


Chart CN. $$BATTN7 - SETDF Root Phase (Part 3 of 6)
(Refer to Chart 11)

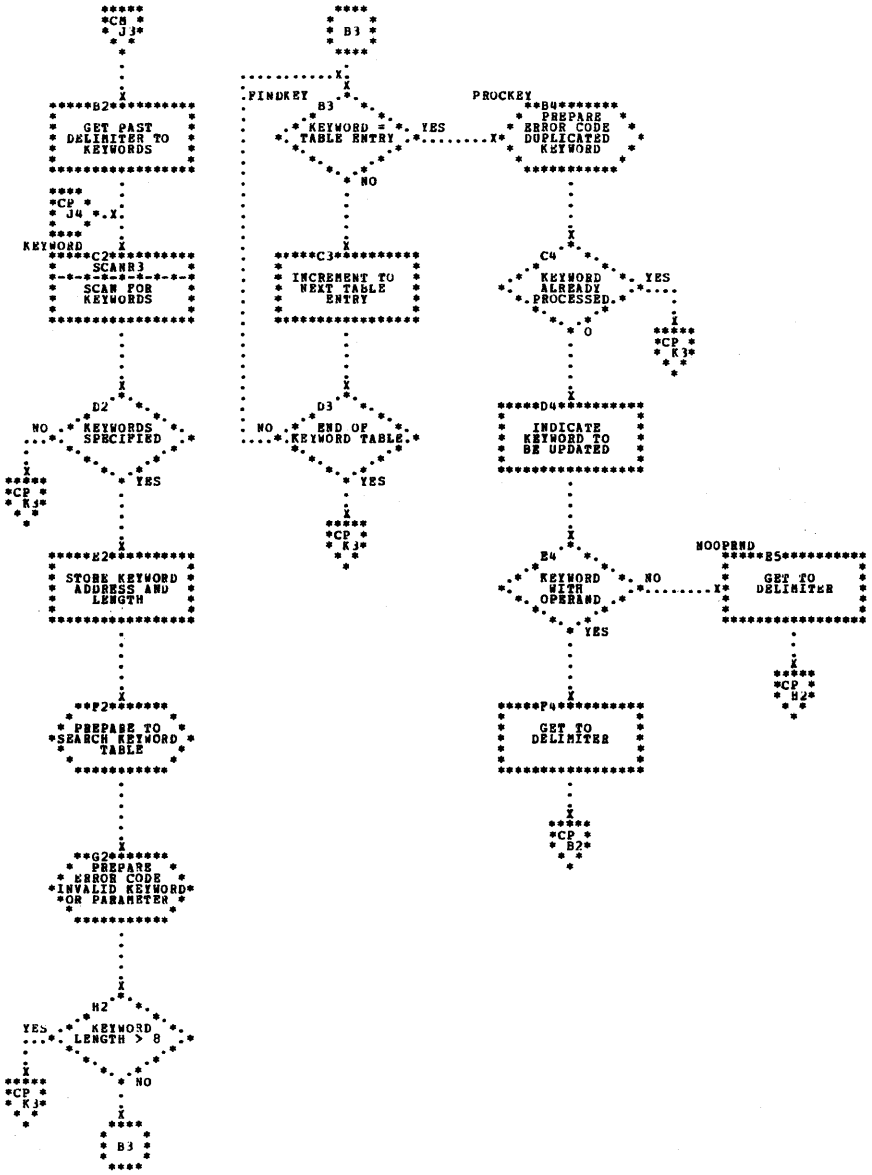


Chart CP. $$BATTN7 - SETDF Root Phase (Part 4 of 6)
(Refer to Chart 11)

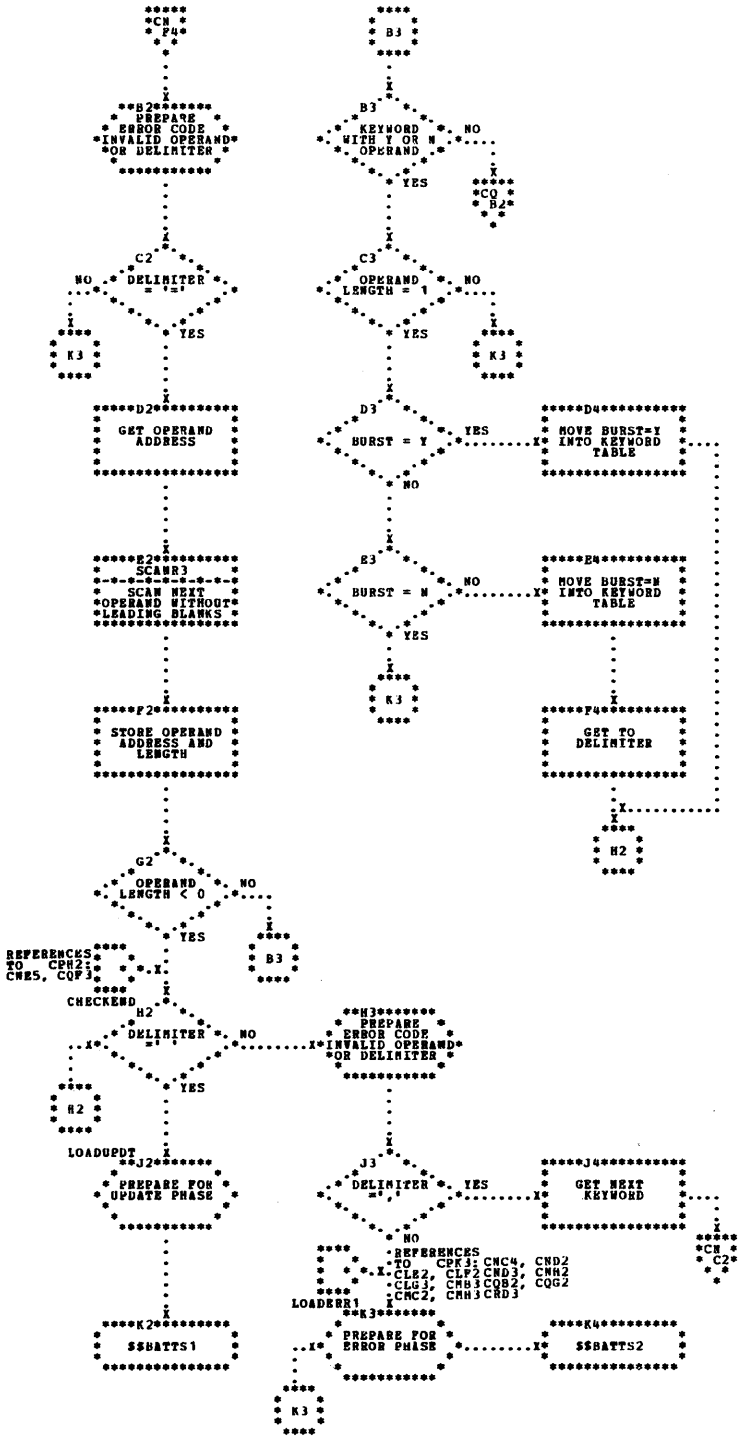


Chart CQ. $$BATTN7 - SETDF Root Phase (Part 5 of 6)
(Refer to Chart 11)

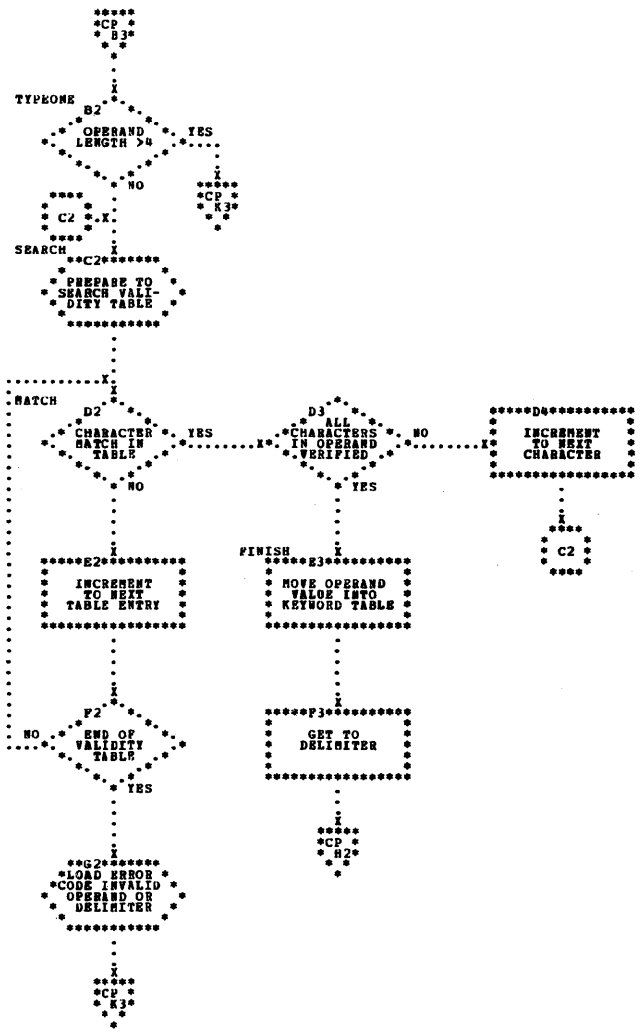


Chart CR. $$BATTN7 - SETDF Root Phase (Part 6 of 6)
(Refer to Chart 11)

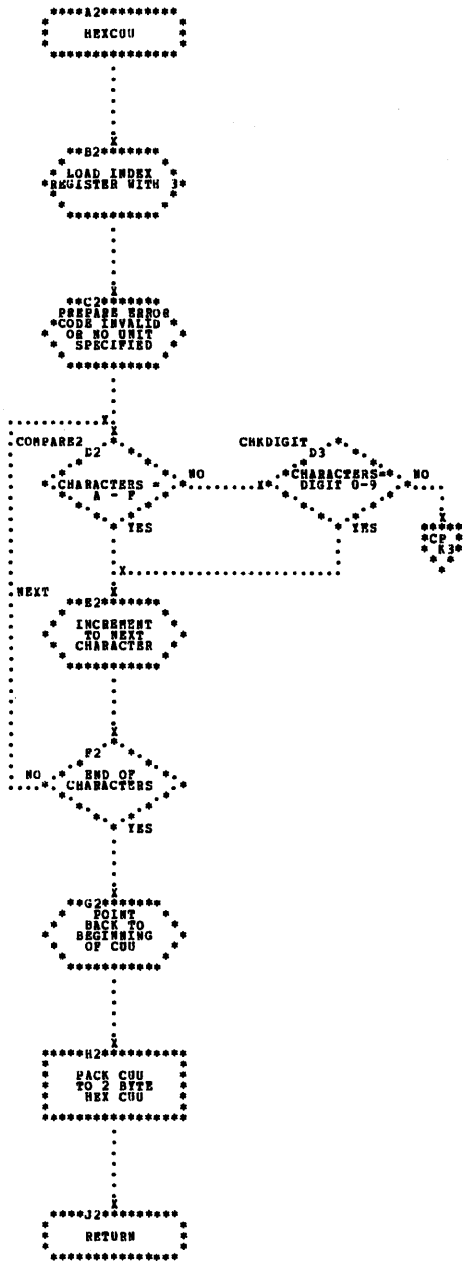


Chart CS. $$BATTS1 - SETDF Update Phase (Part 1 of 3)
(Refer to Chart 11)

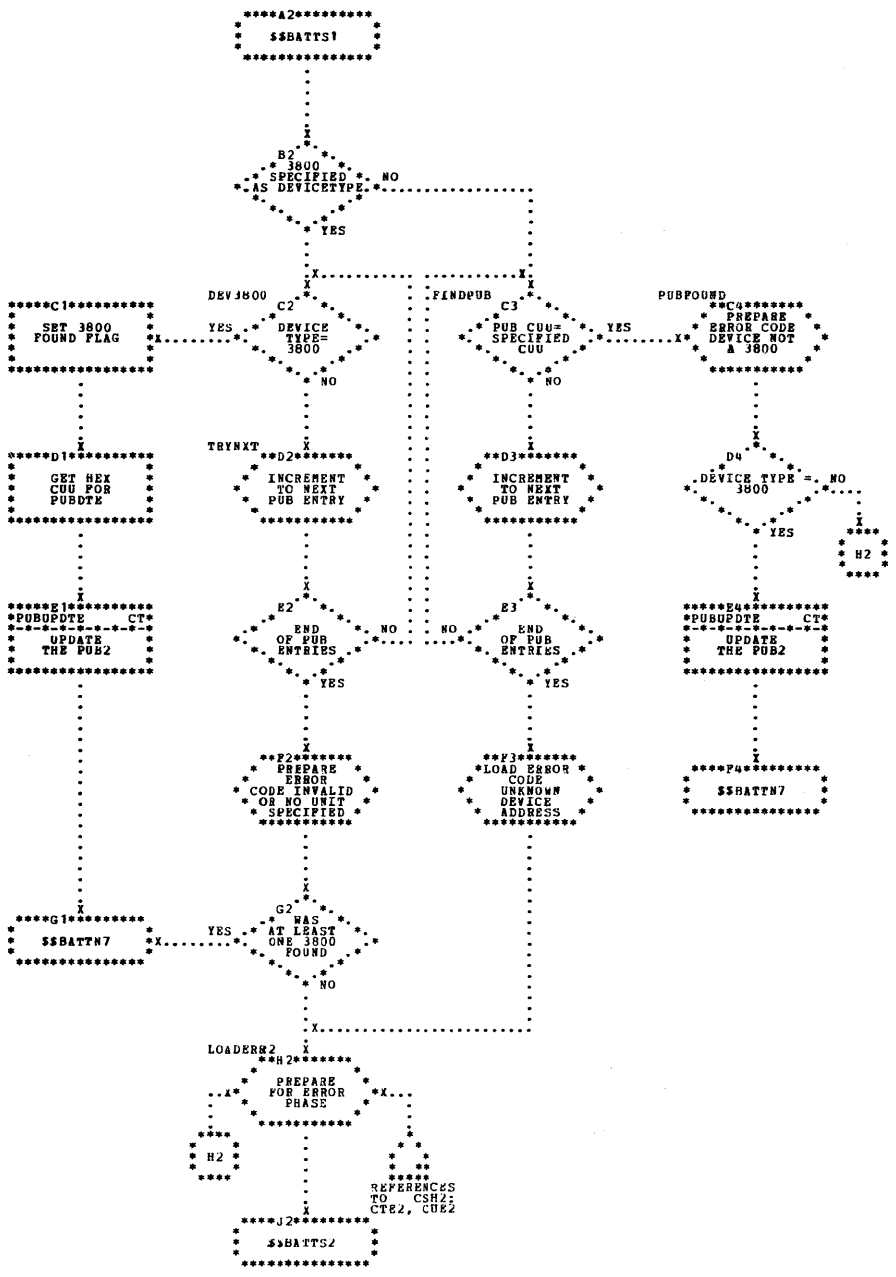


Chart CT. $$BATTS1 - SETDF Update Phase (Part 2 of 3)
(Refer to Chart 11)

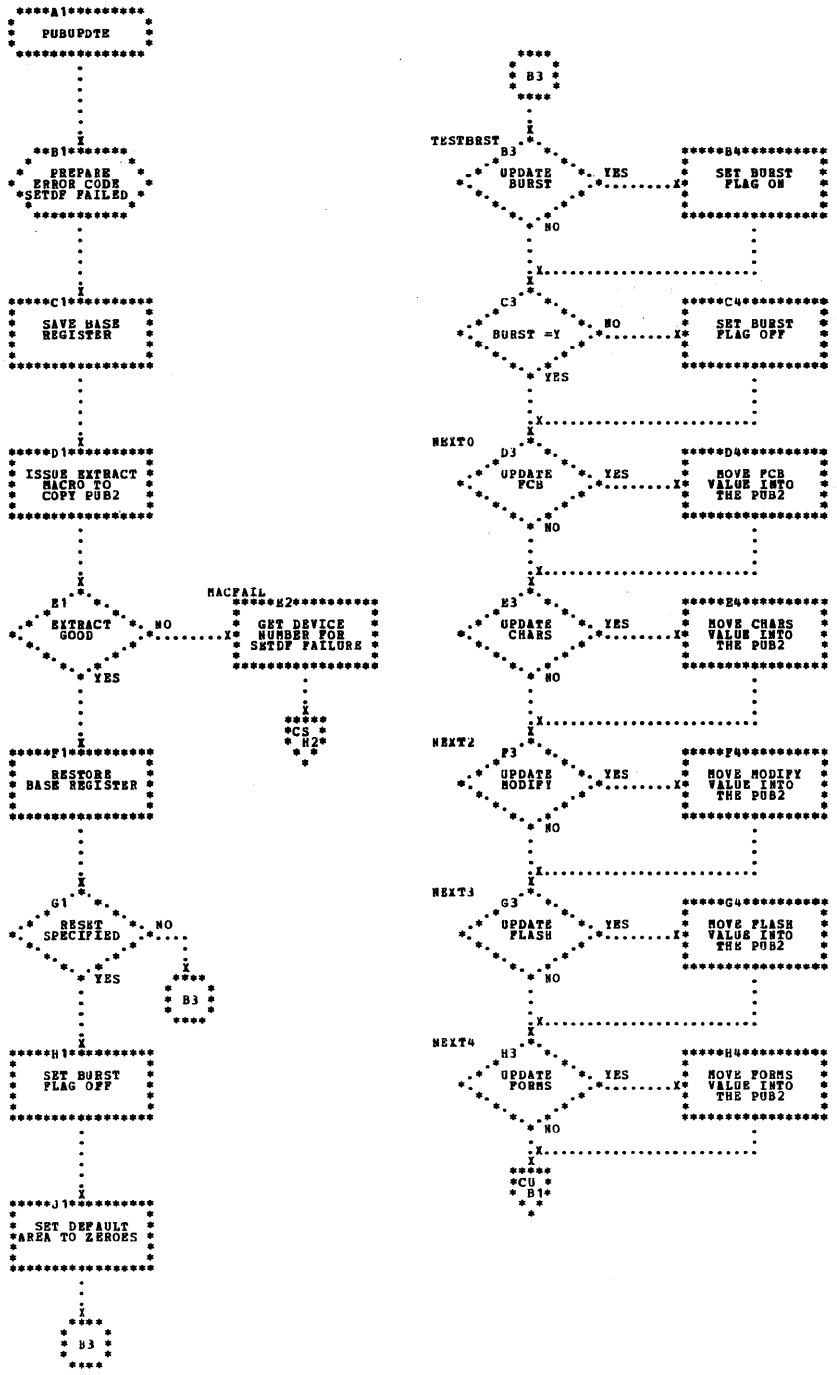


Chart CU. $$BATTS1 - SETDF Update Phase (Part 3 of 3)
(Refer to Chart 11)

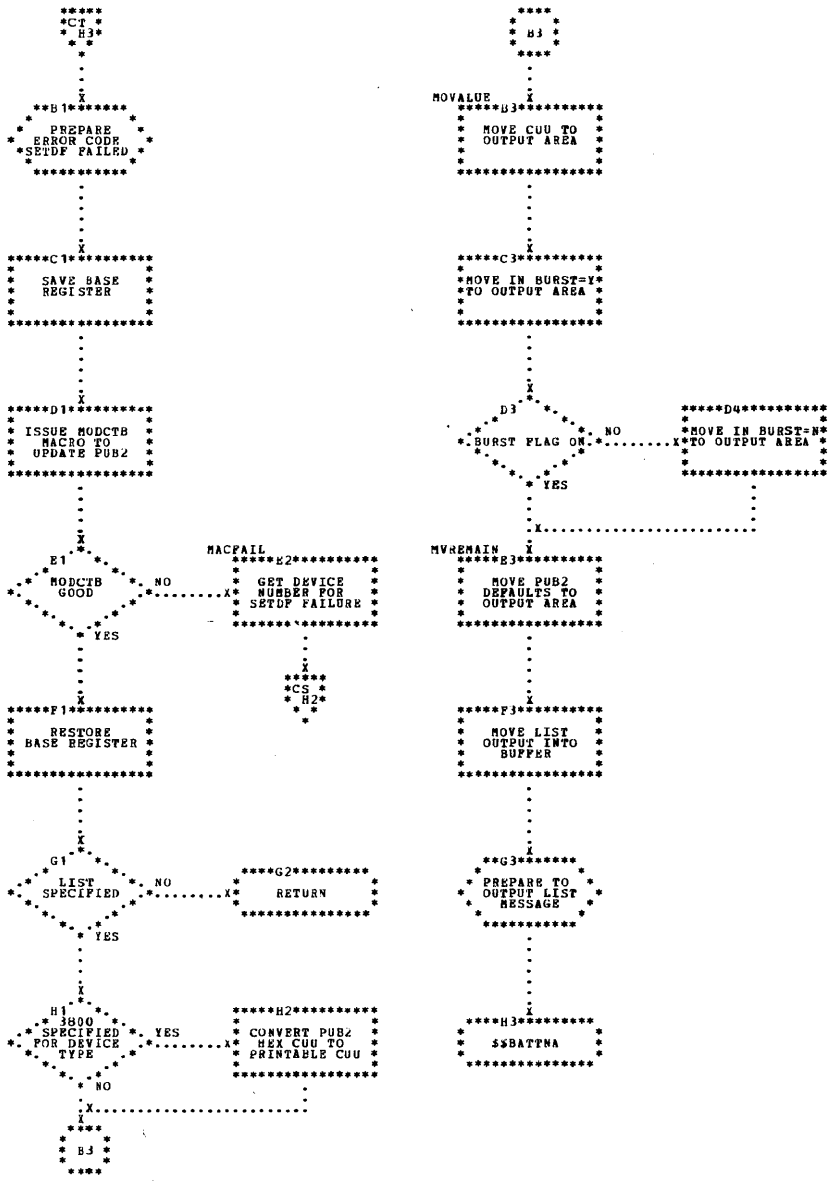


Chart CV. $$BATTS2 - SETDF Error Phase
(Refer to Chart 11

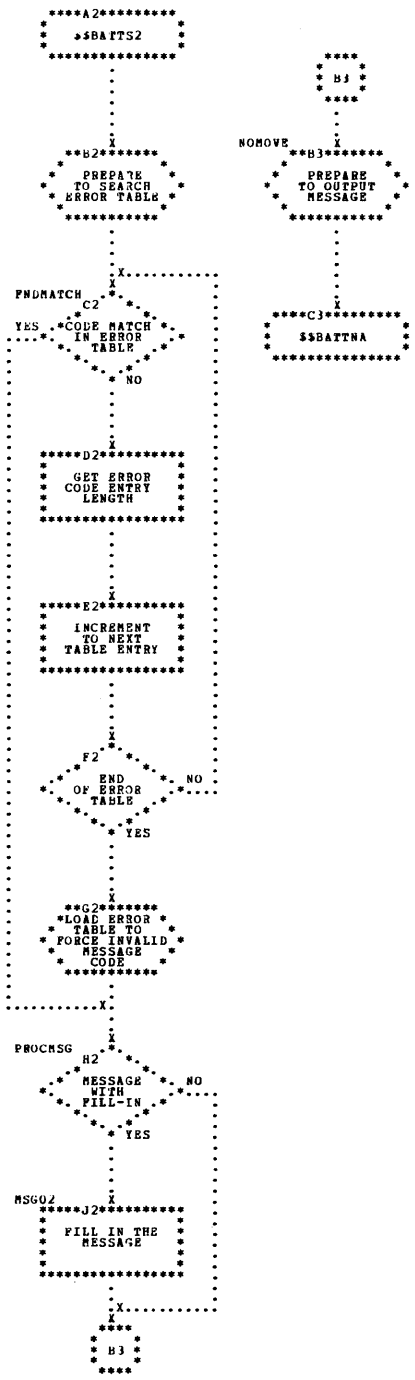


Chart DA. $$BEOJ - Terminate Task and Initialize Partition (Part 1 of 3)
(Refer to Chart 12)

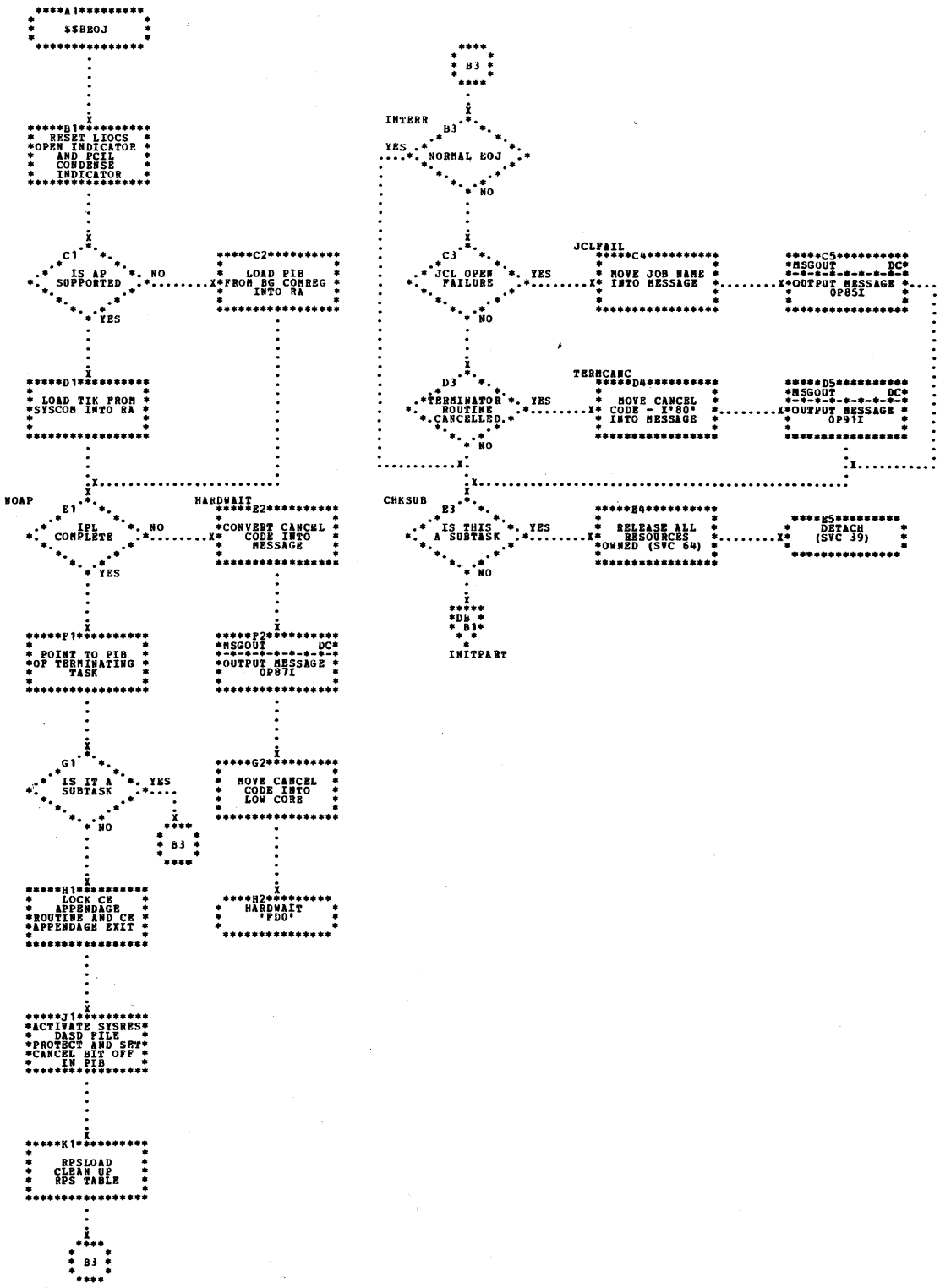


Chart DB. $$BEOJ - Terminate Task and Initialize Partition (Part 2 of 3)
(Refer to Chart 12)

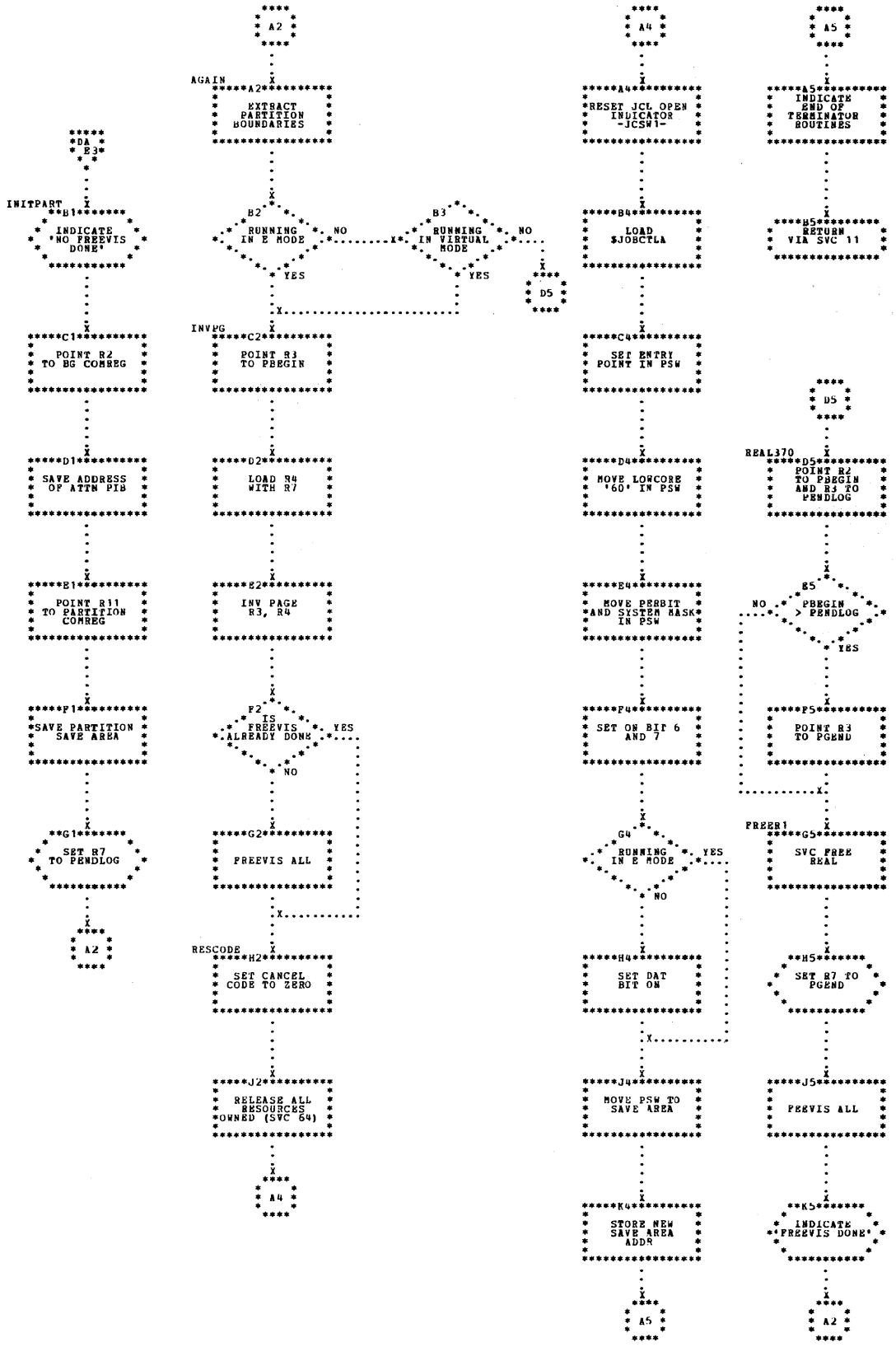


Chart DC. $$BEOJ - Terminate Task and Initialize Partition (Part 3 of 3) (Refer to Chart 12)

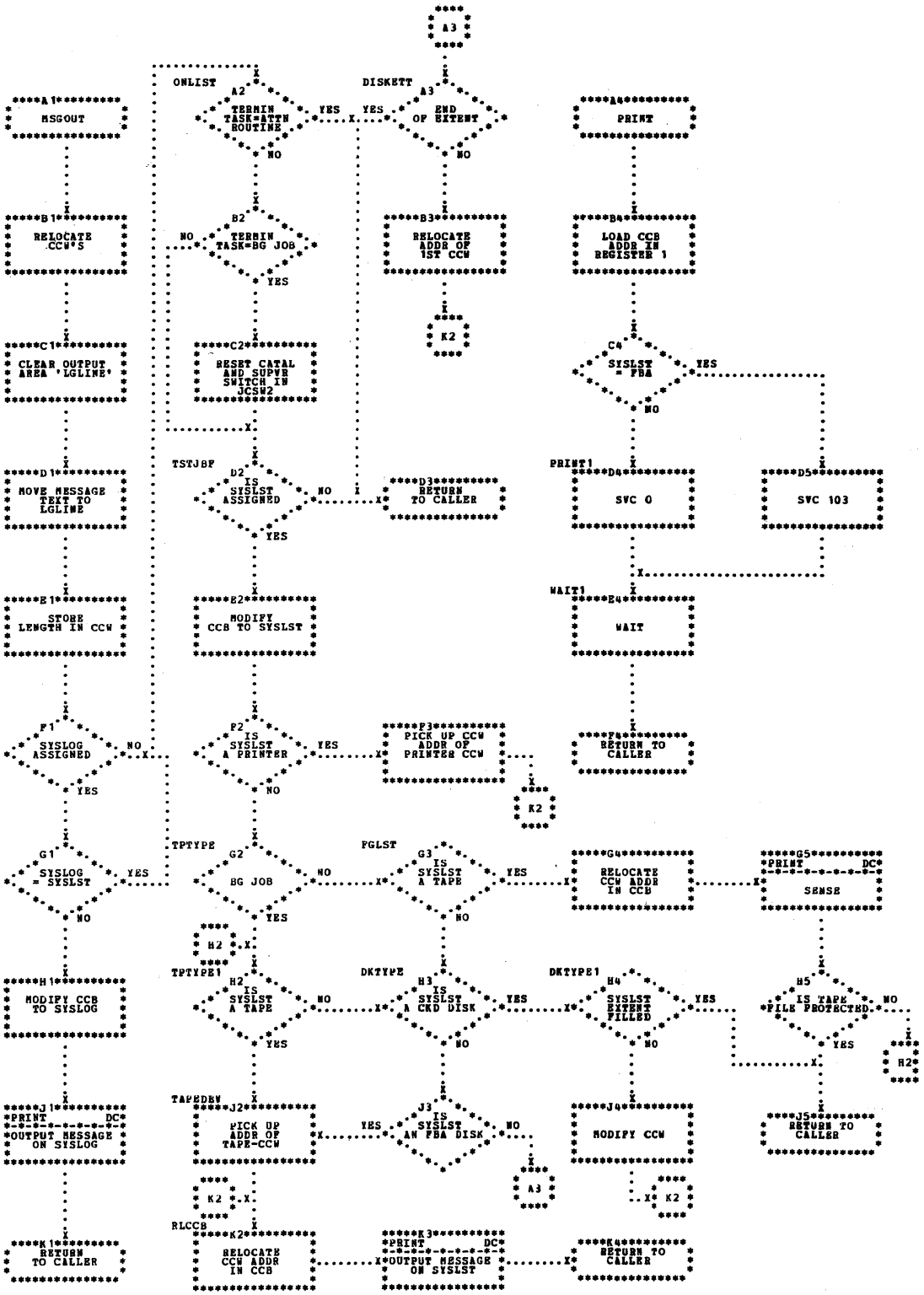


Chart DD. $$BEOJ3 - Decide Next Step in Termination Handling
(Refer to Chart 12)

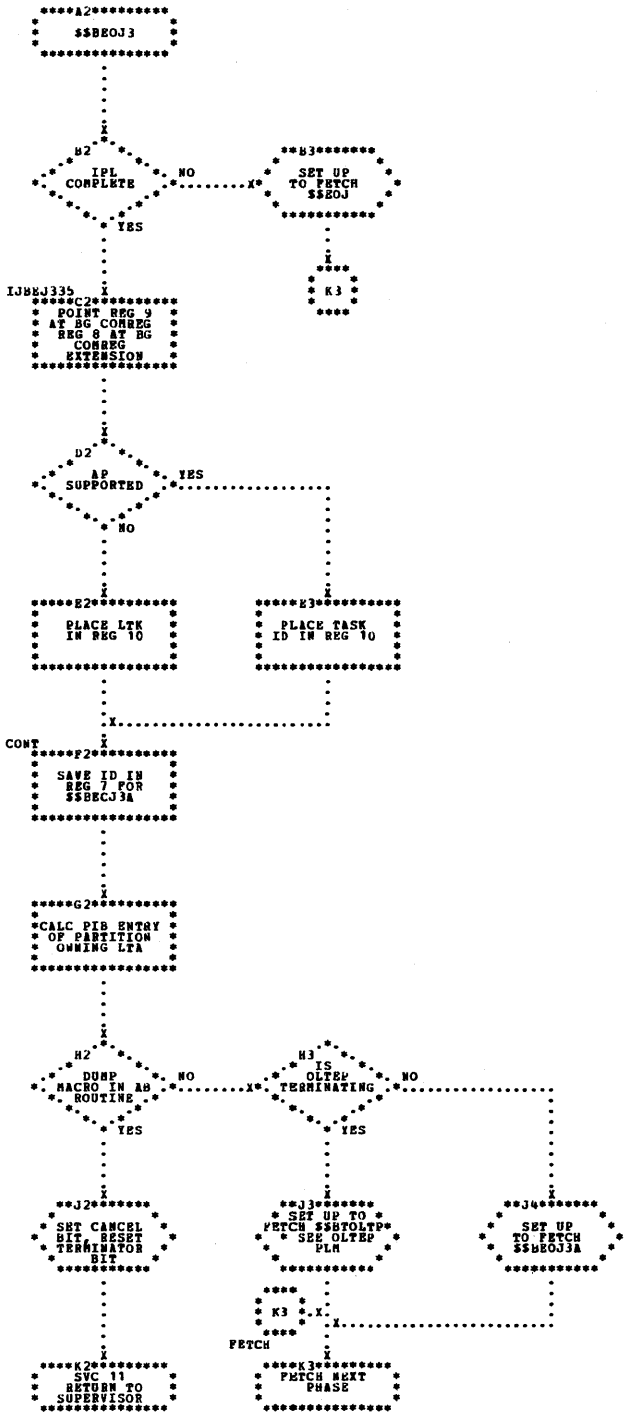


Chart DE. $$BEOJ3A - Quiesce I/O for T/P Devices
(Refer to Chart 12)

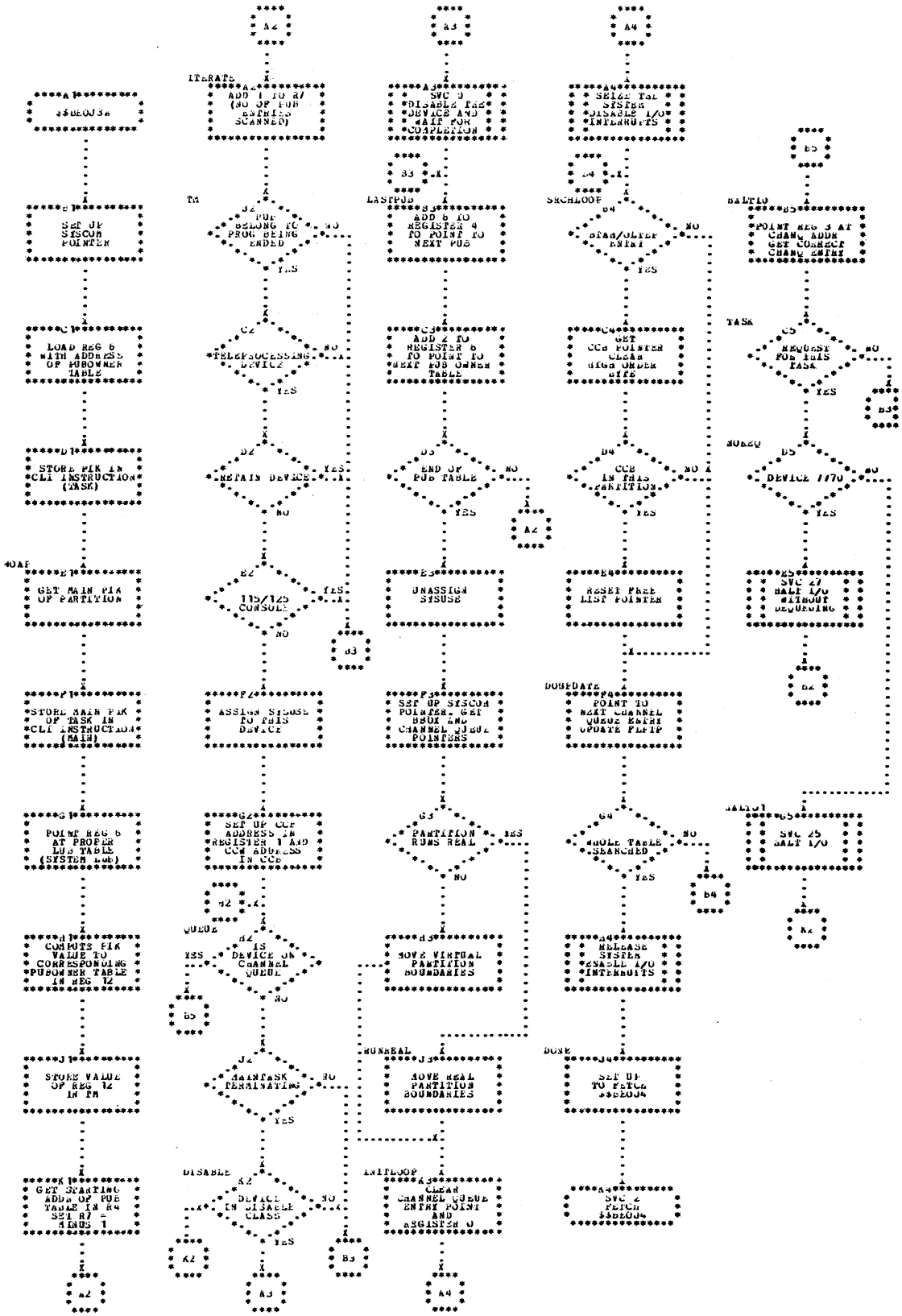


Chart DF. $$BEOJ4 - Clean up Non-TP I/O Devices (Part 1 of 3)
(Refer to Chart 12)

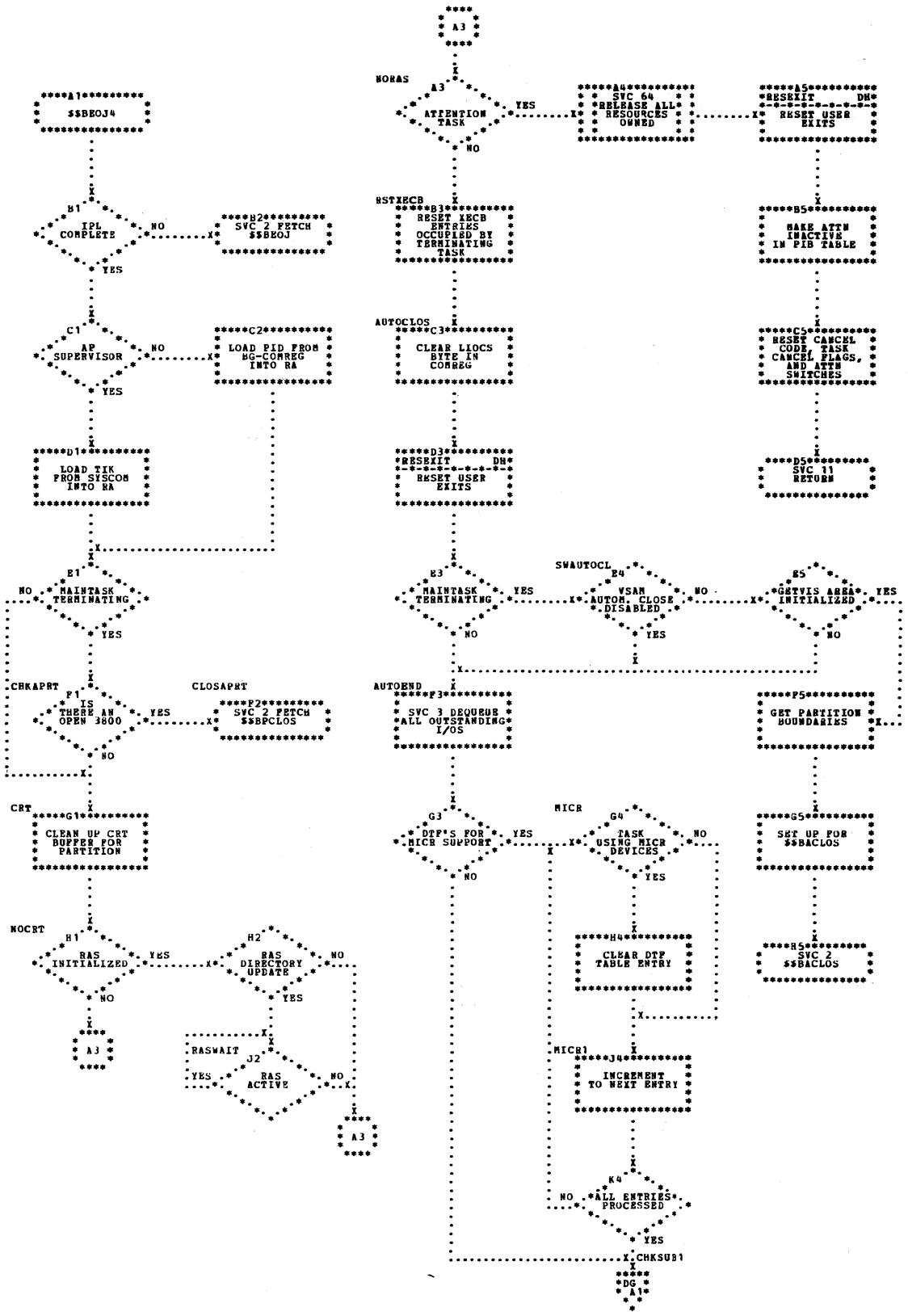


Chart DG. $$BEOJ4 - Clean up Non-TP I/O Devices (Part 2 of 3)
(Refer to Chart 12)

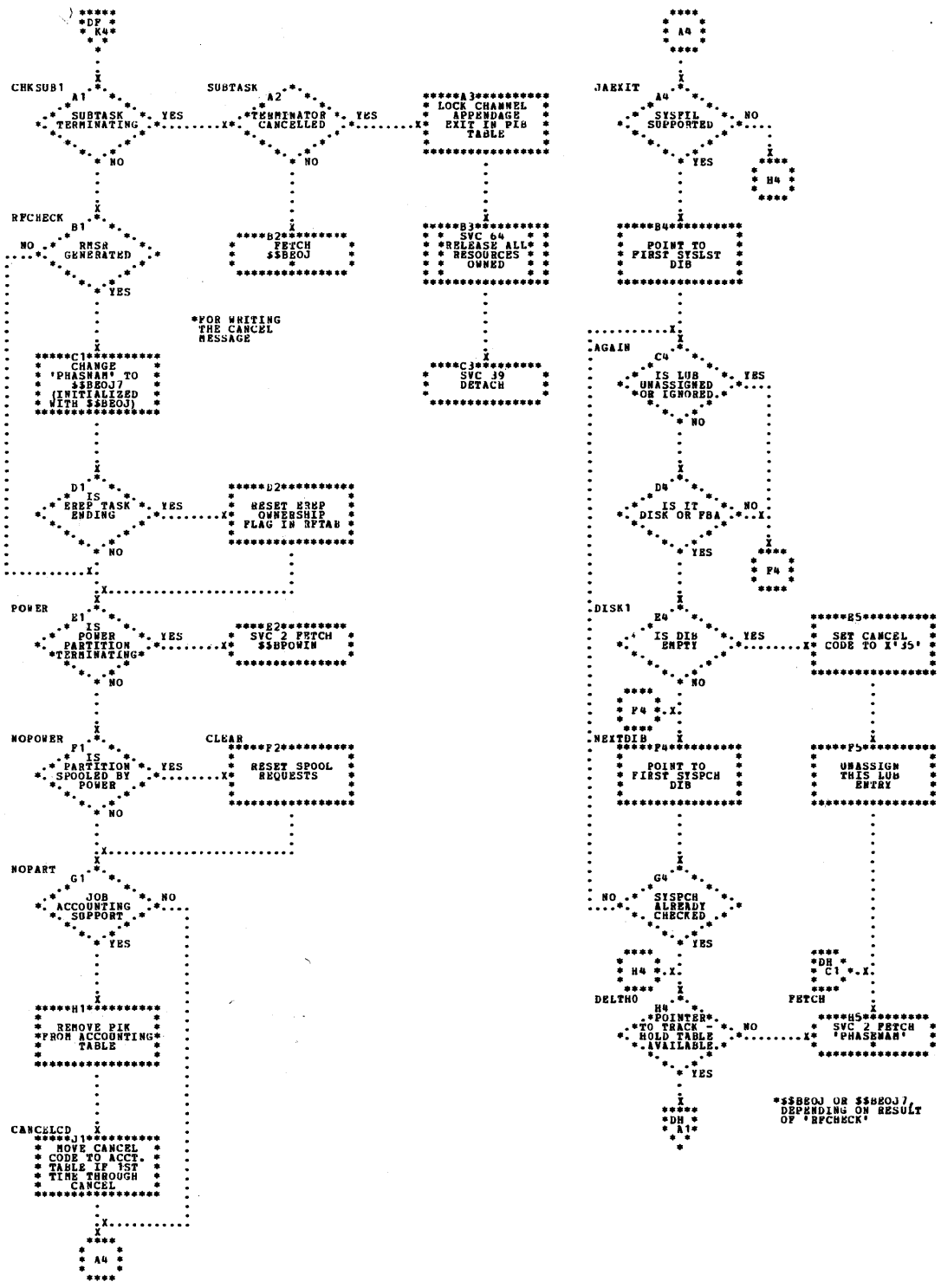


Chart DH. $$BEOJ4 - Clean up Non-TP I/O Devices (Part 3 of 3)
(Refer to Chart 12)

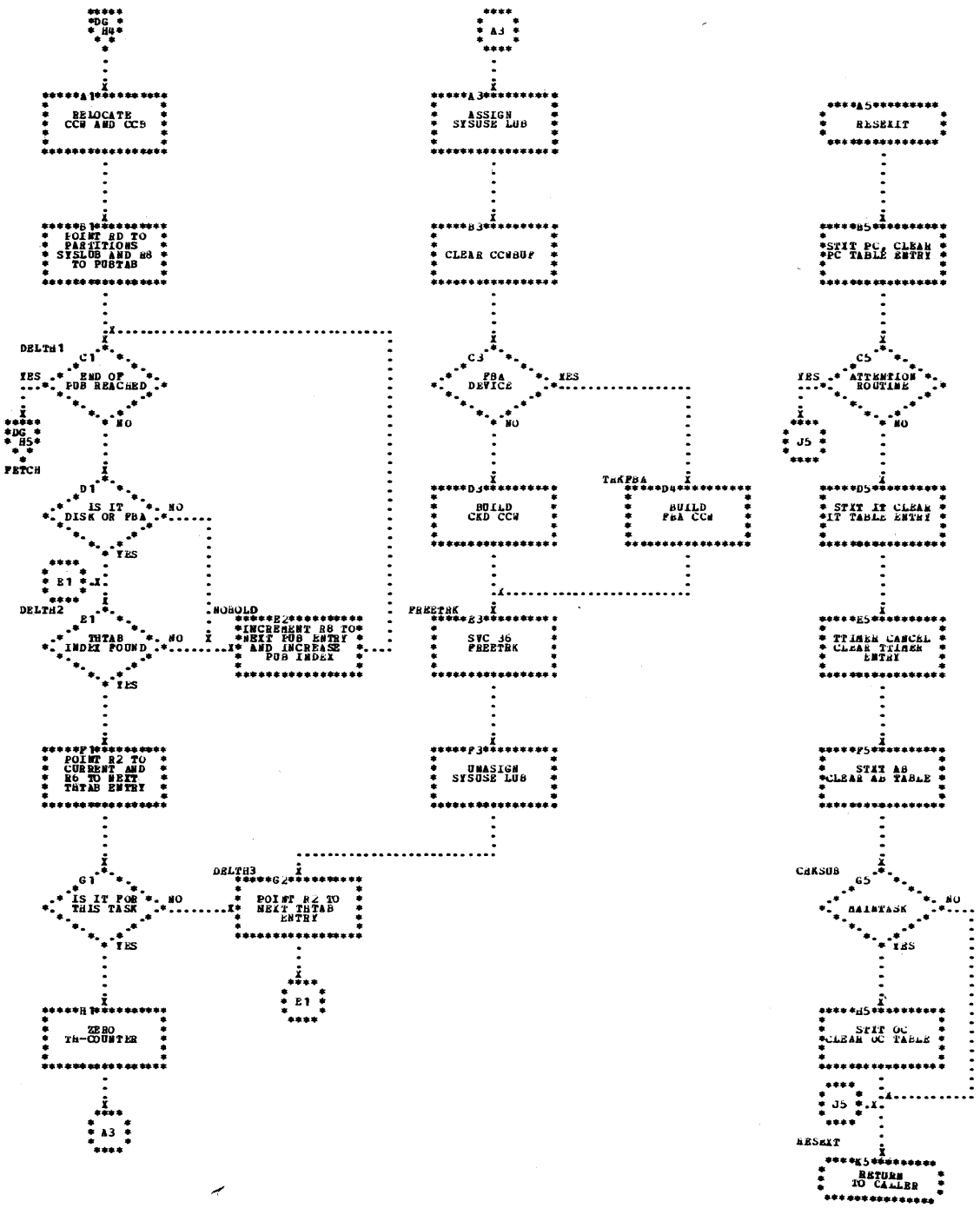


Chart DJ. $$BACLOS - VSAM Automatic Close (Part 1 of 2)
(Refer to Chart 12)

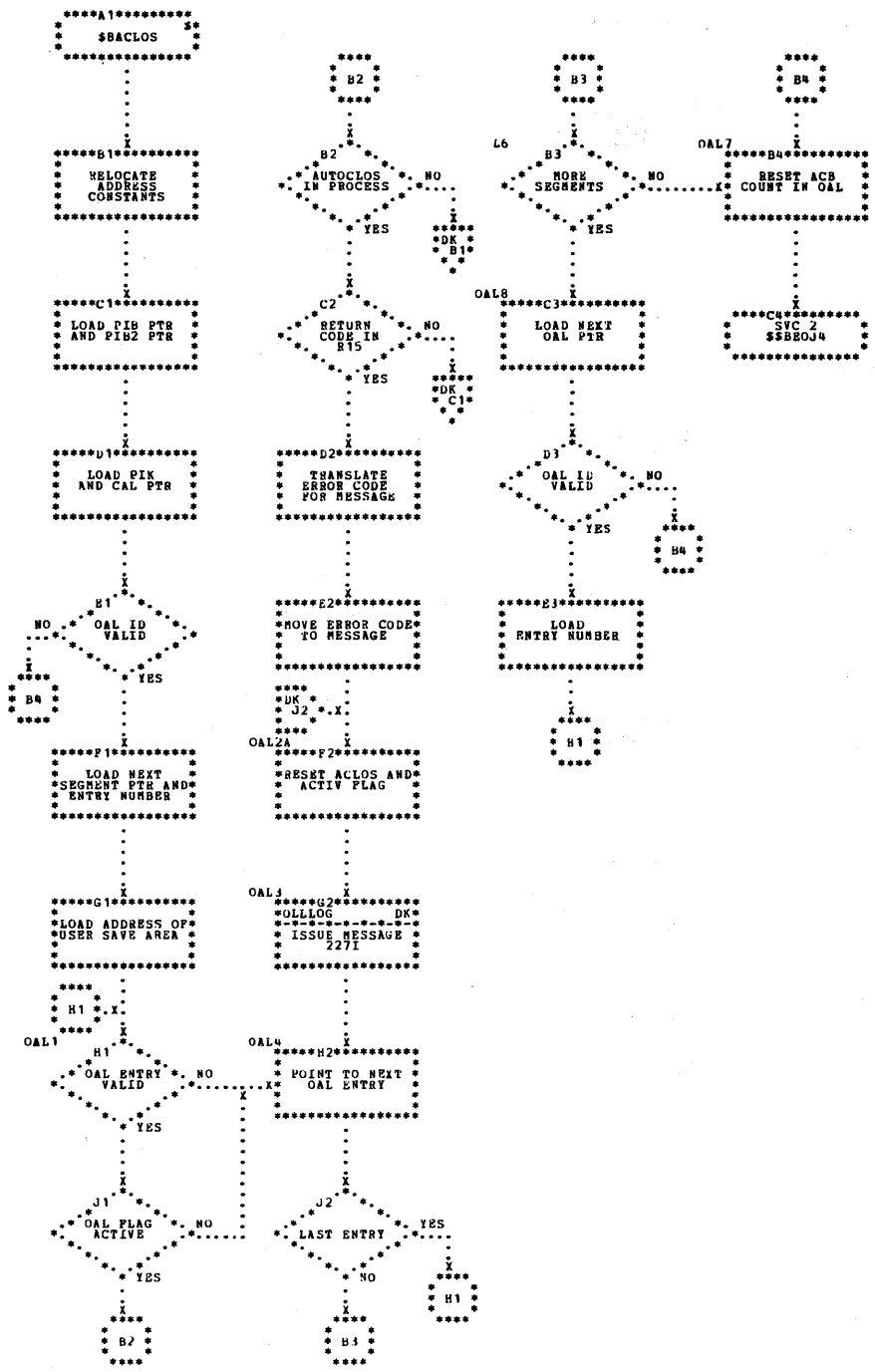


Chart DK. $$BACLOS - VSAM Automatic Close (Part 2 of 2)
(Refer to Chart 12)

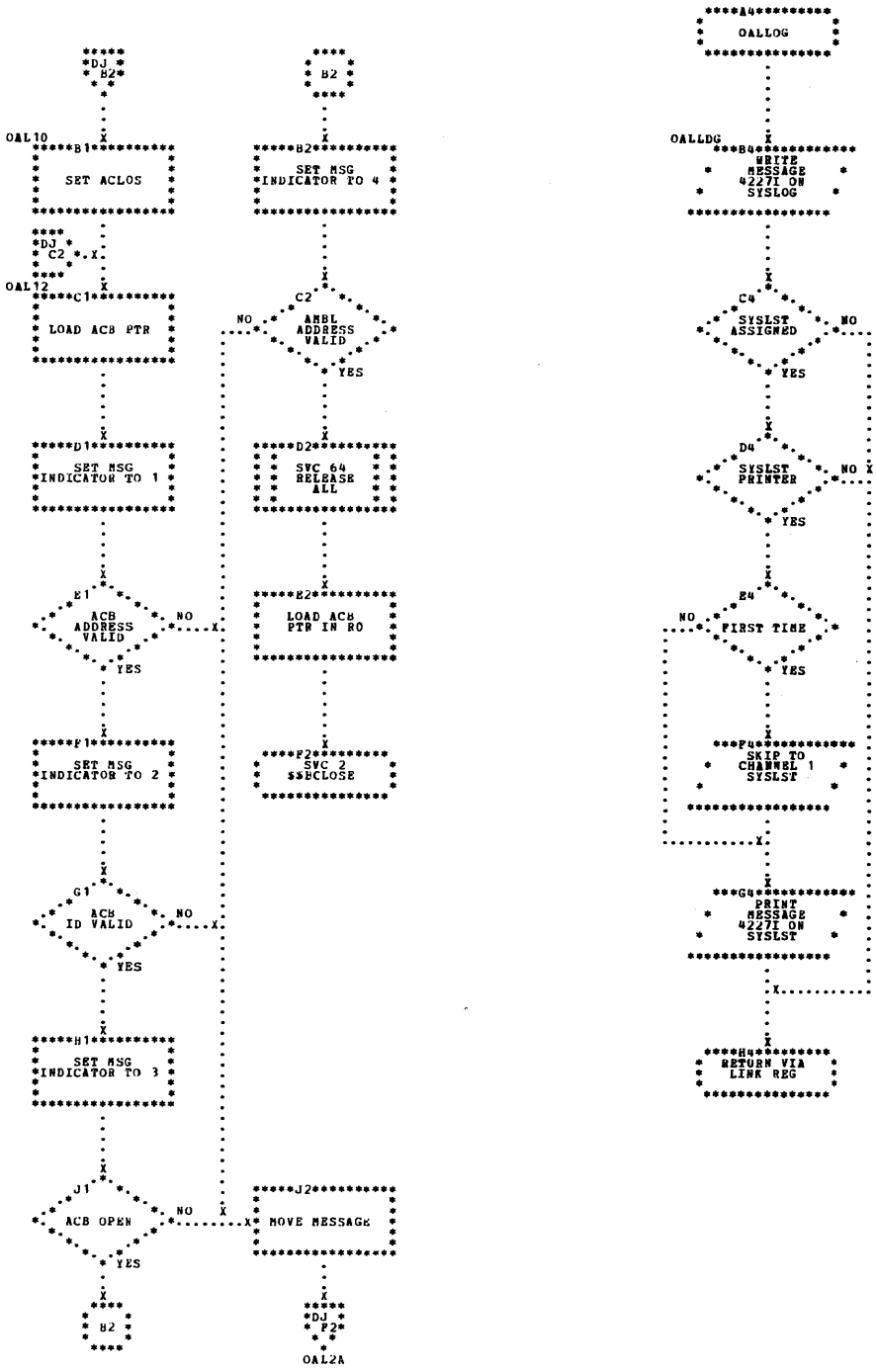


Chart DL. $$BEOJ7 - Tape and Disk PUB2 Processor (Part 1 of 3)
(Refer to Chart 12)

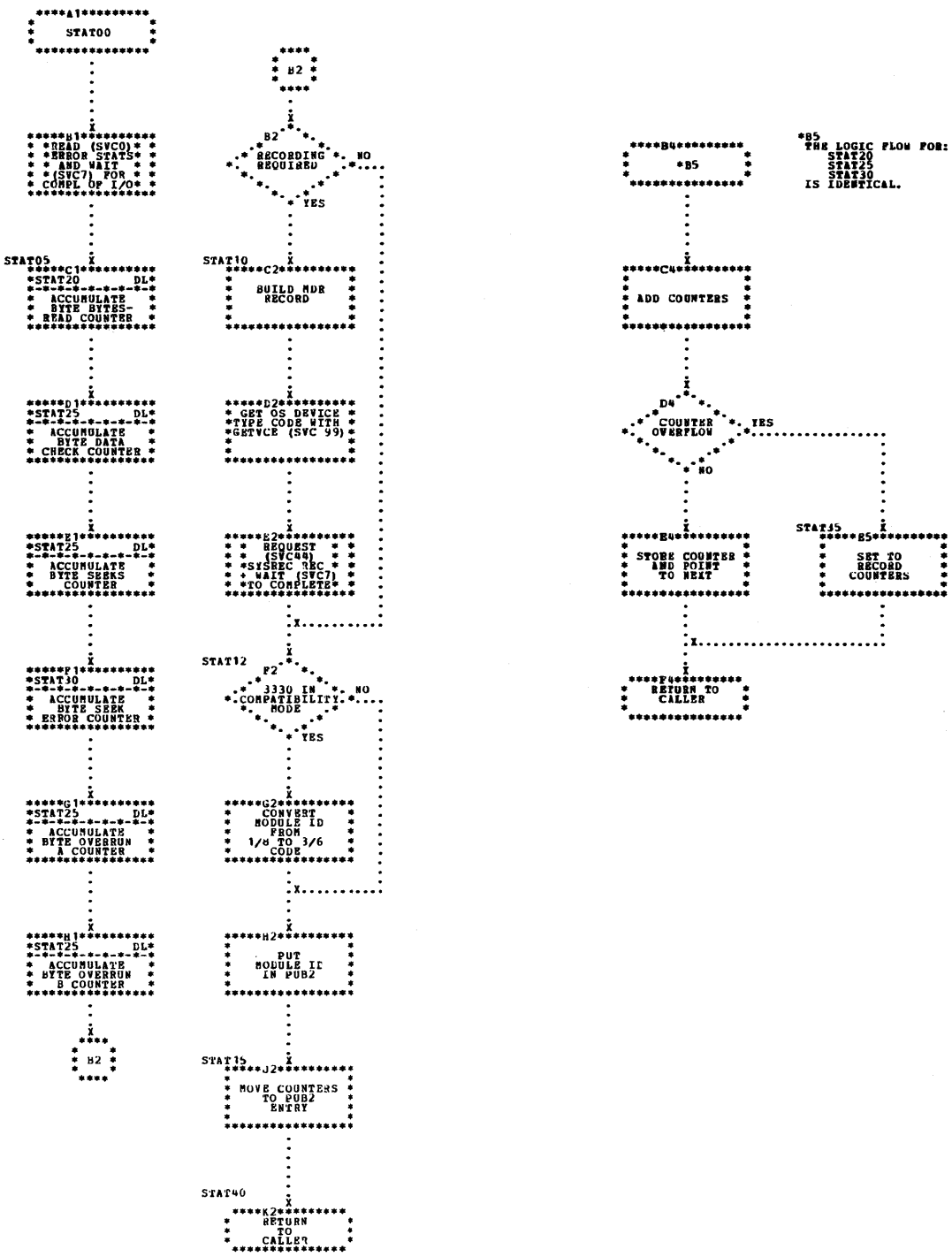


Chart DM. $$BEOJ7 - Tape and Disk PUB2 Processor (Part 2 of 3)
(Refer to Chart 12)

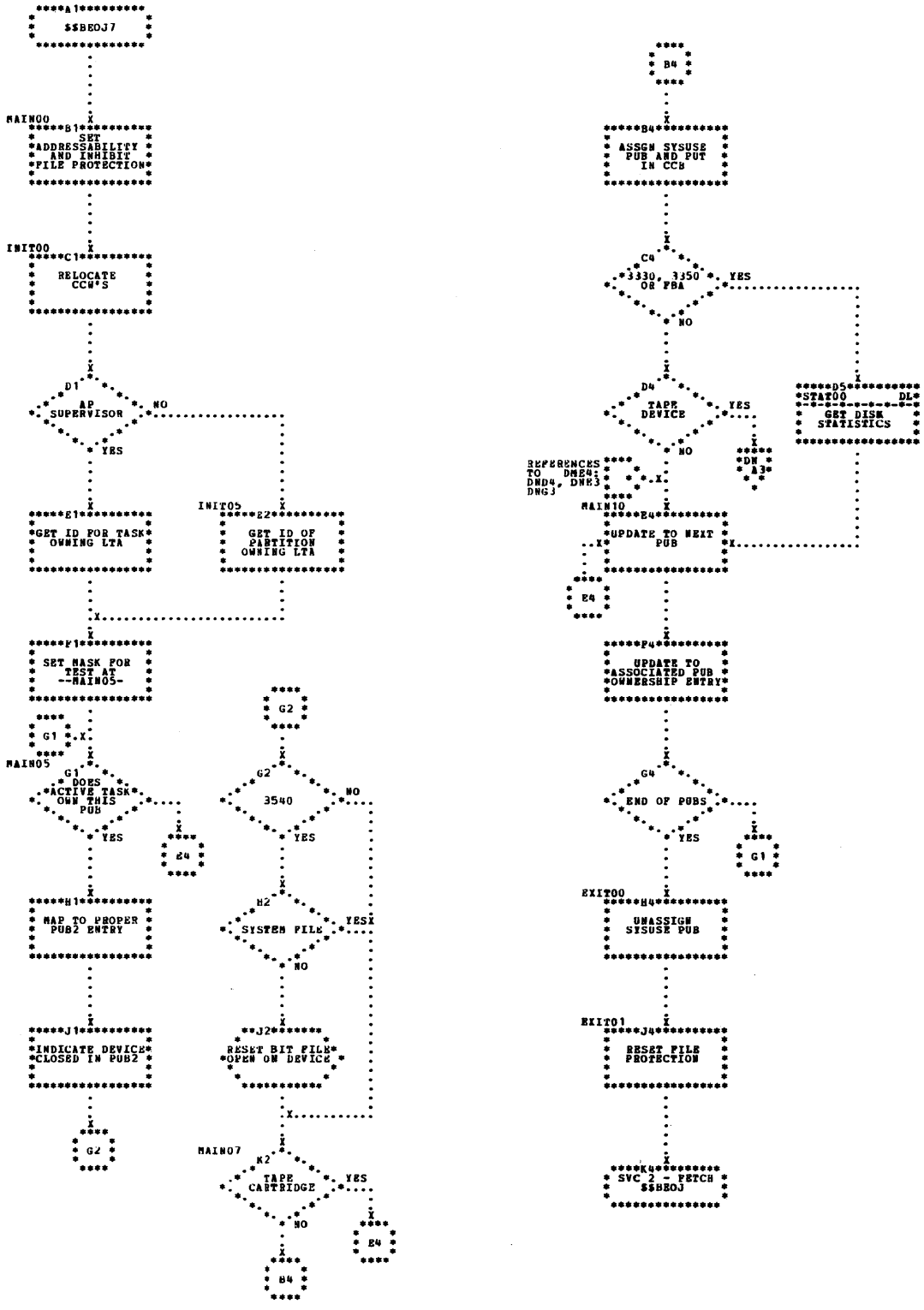


Chart DN. $$BEOJ7 - Tape and Disk PUB2 Processor (Part 3 of 3)
(Refer to Chart 12)

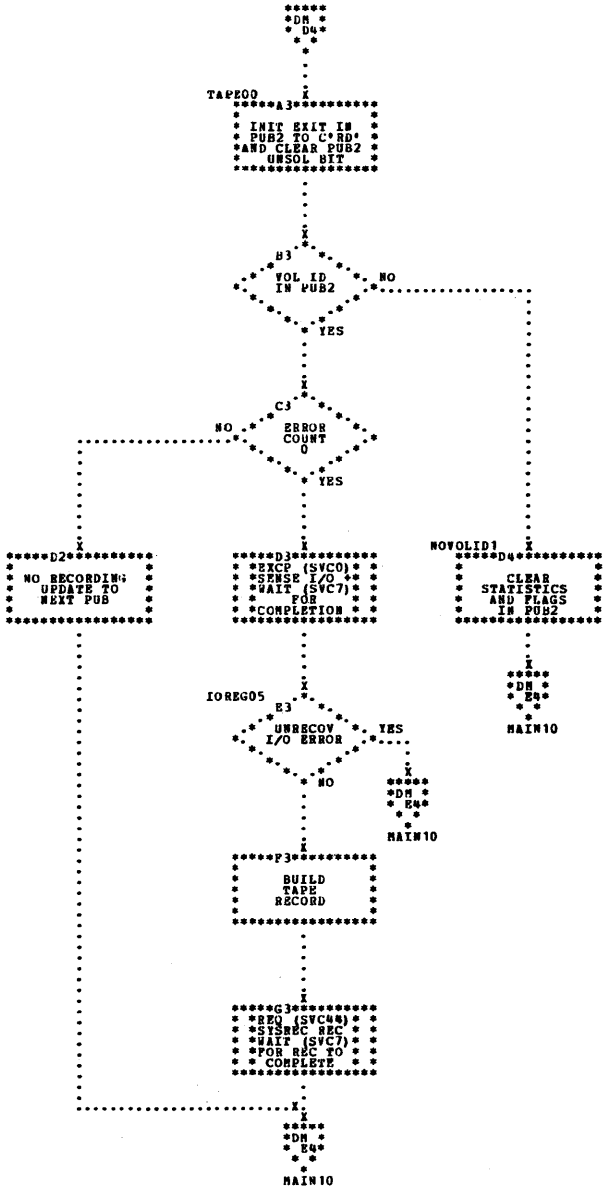


Chart EA. IJBSDUMP - Dump Monitor (Part 1 of 13)
(Refer to Chart 13)

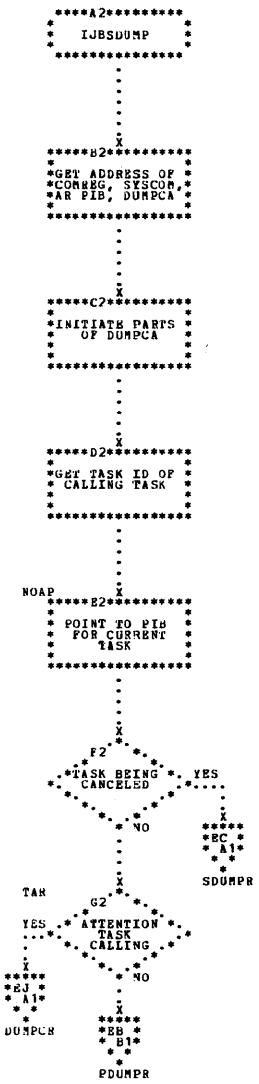


Chart EB. IJBSDUMP - Dump Monitor (Part 2 of 13)
(Refer to Chart 13)

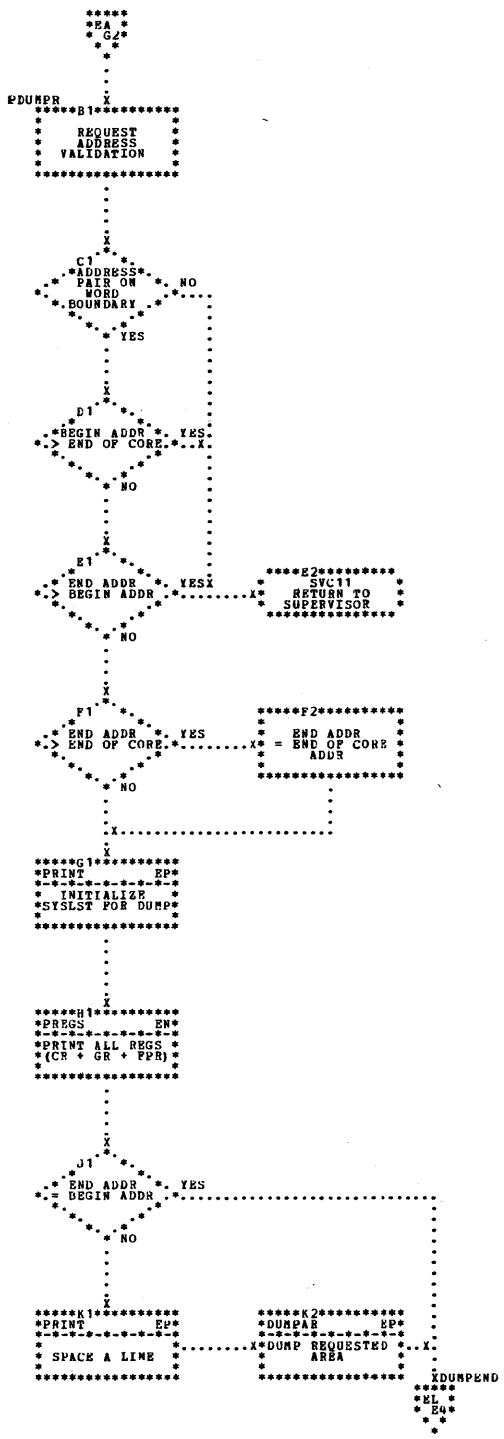


Chart EC. IJBSDUMP - Dump Monitor (Part 3 of 13)
(Refer to Chart 13)

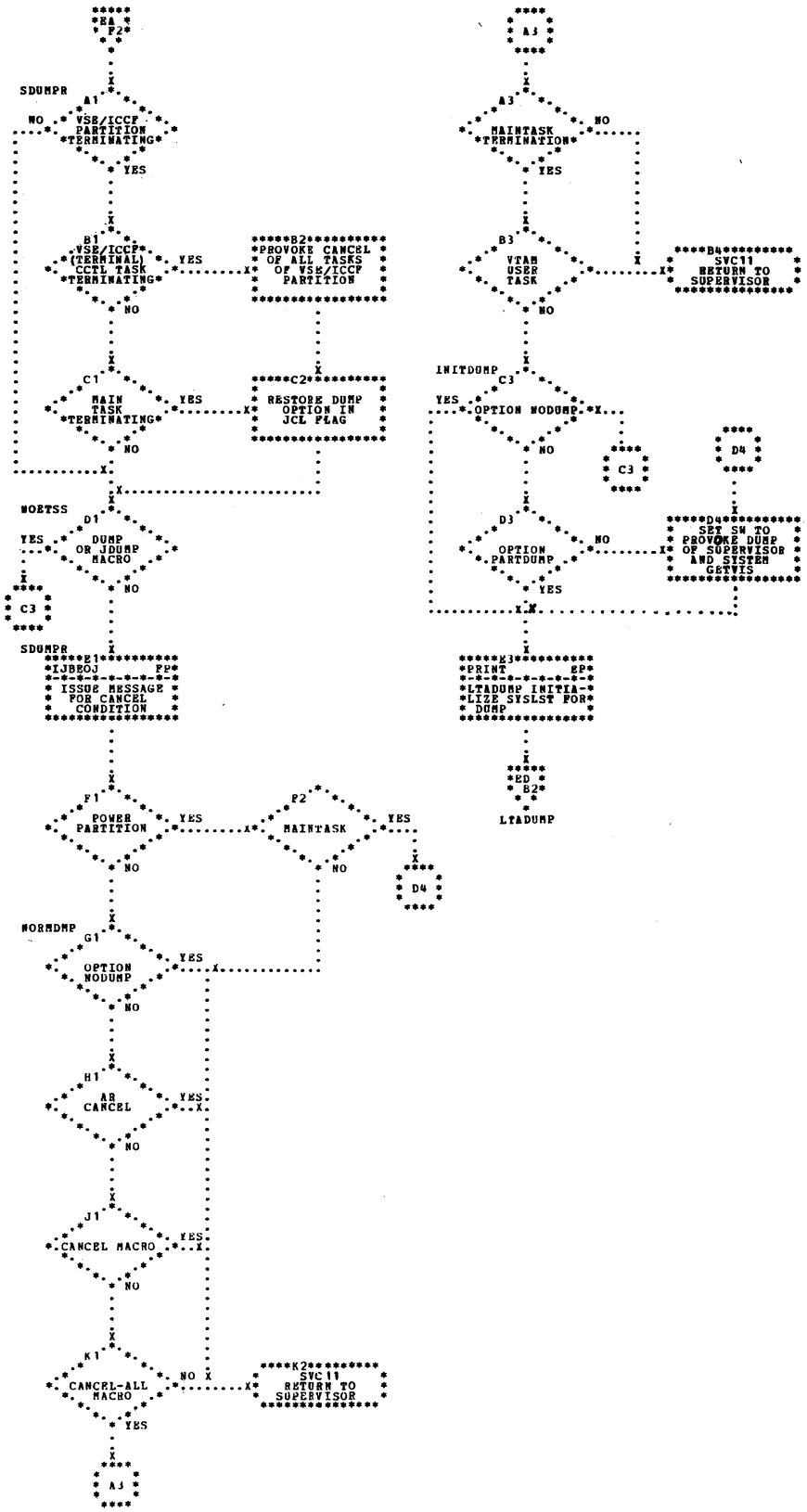


Chart ED. IJBSDUMP - Dump Monitor (Part 4 of 13)
(Refer to Chart 13)

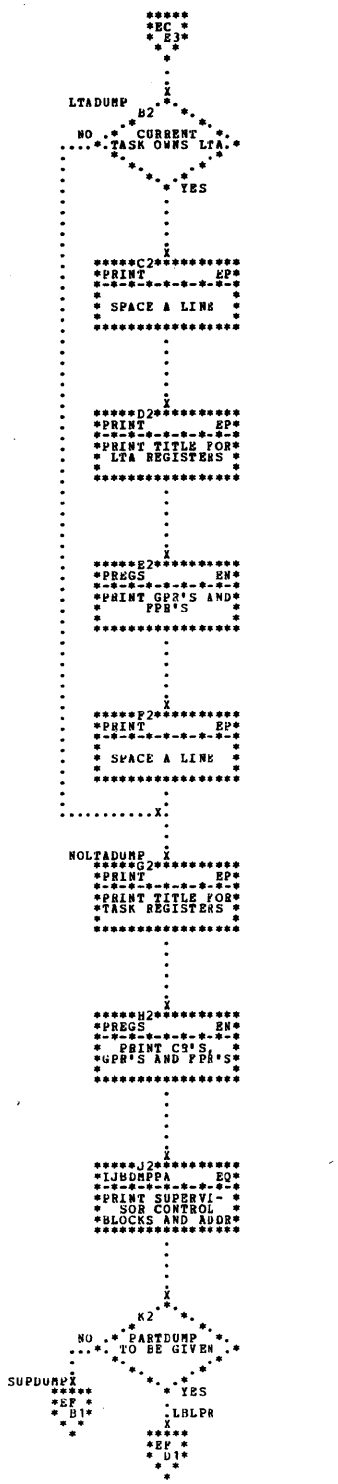


Chart EF. IJBSDUMP - Dump Monitor (Part 5 of 13)
(Refer to Chart 13)

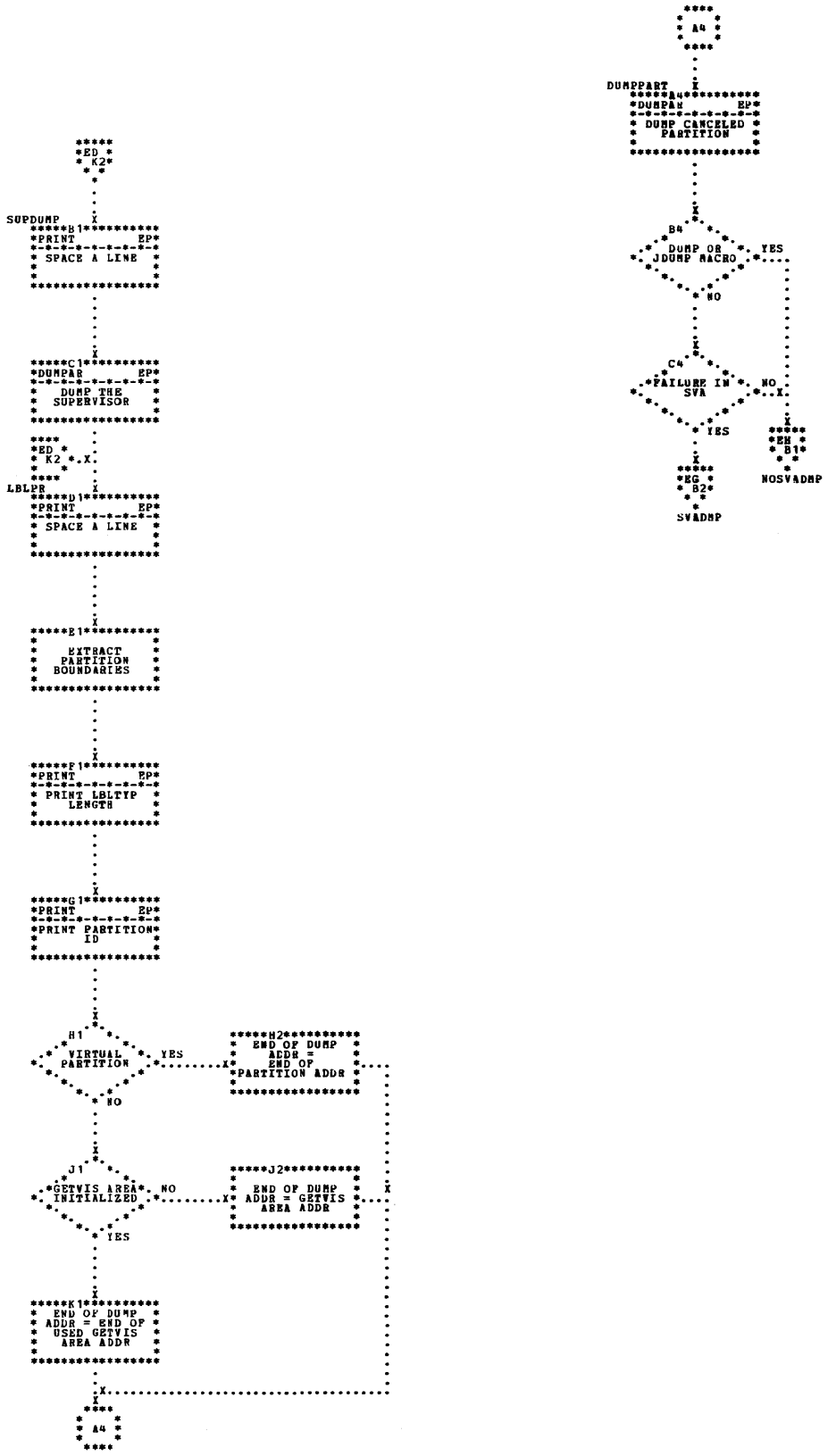


Chart EG. IJBSDUMP - Dump Monitor (Part 6 of 13)
(Refer to Chart 13)

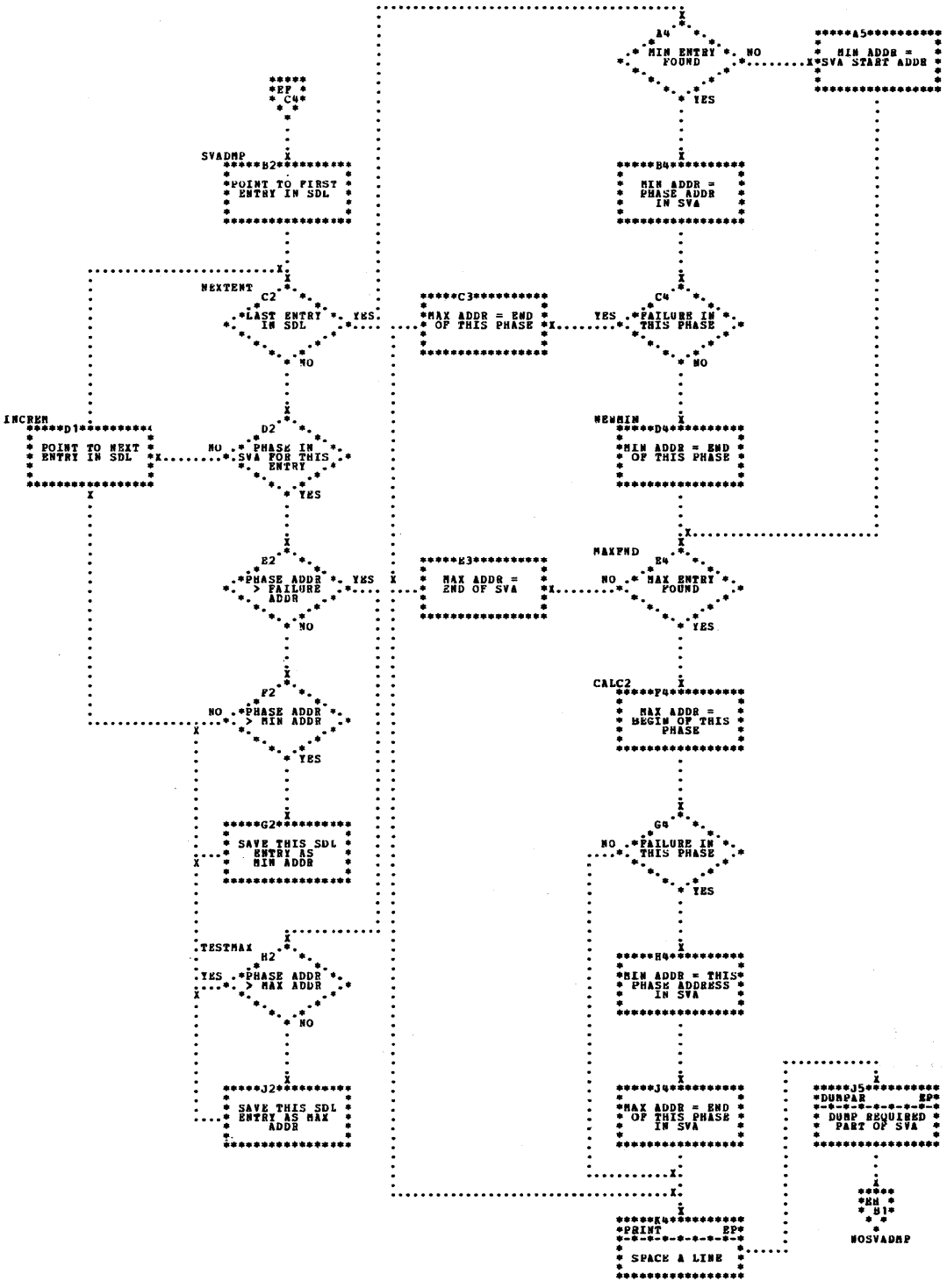


Chart EH. IJBSDUMP - Dump Monitor (Part 7 of 13)
(Refer to Chart 13)

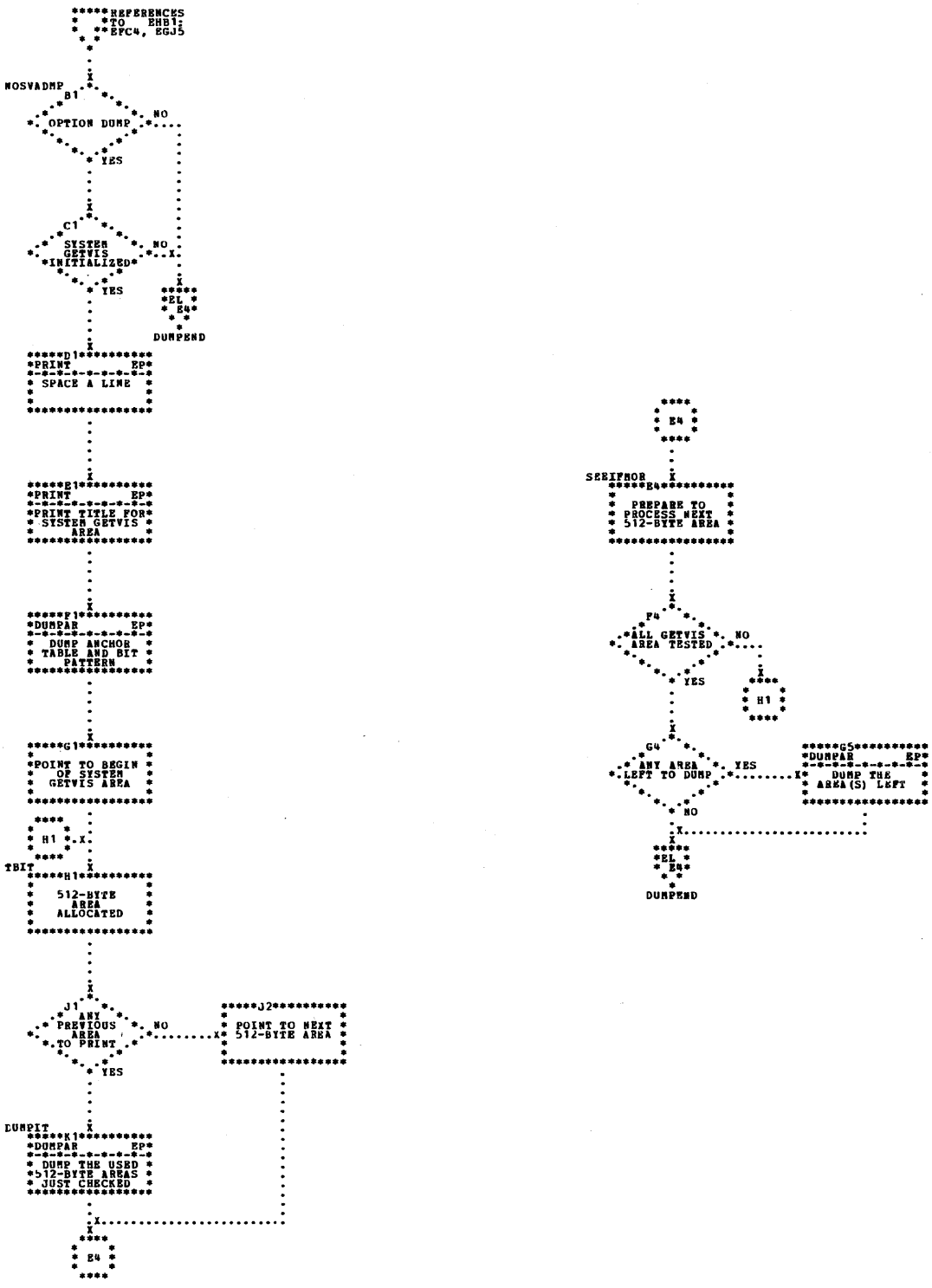


Chart EJ. IJBSDUMP - Dump Monitor (Part 8 of 13)
(Refer to Chart 13)

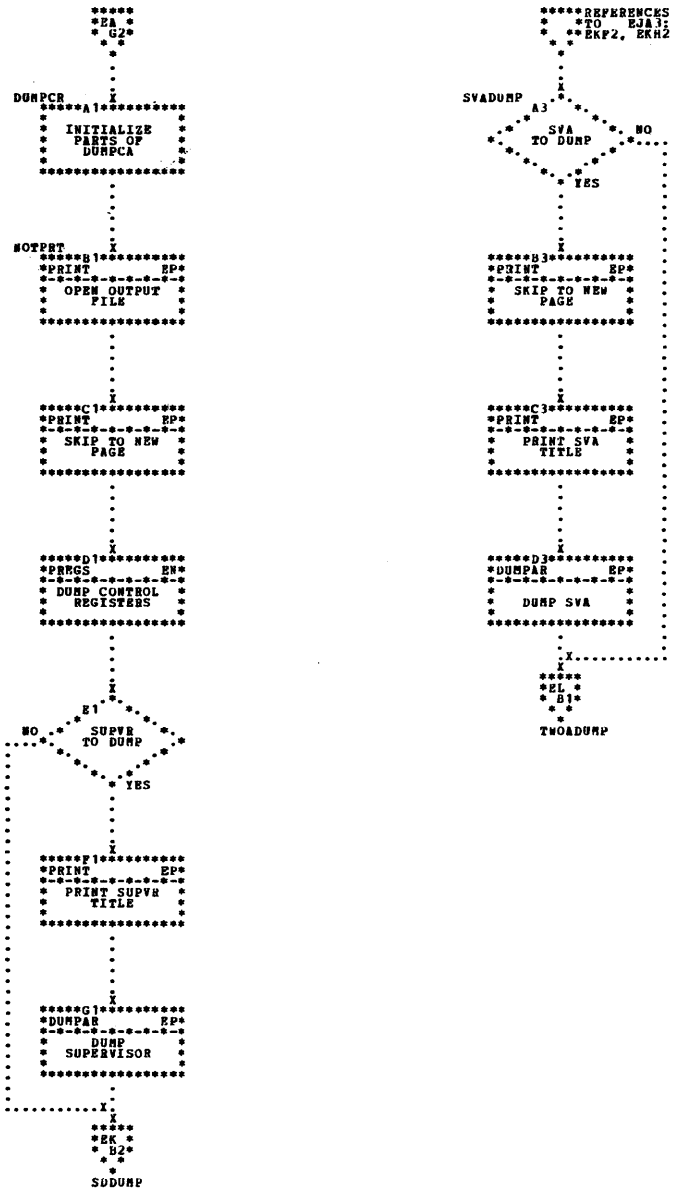


Chart EK. IJBSDUMP - Dump Monitor (Part 9 of 13)
(Refer to Chart 13)

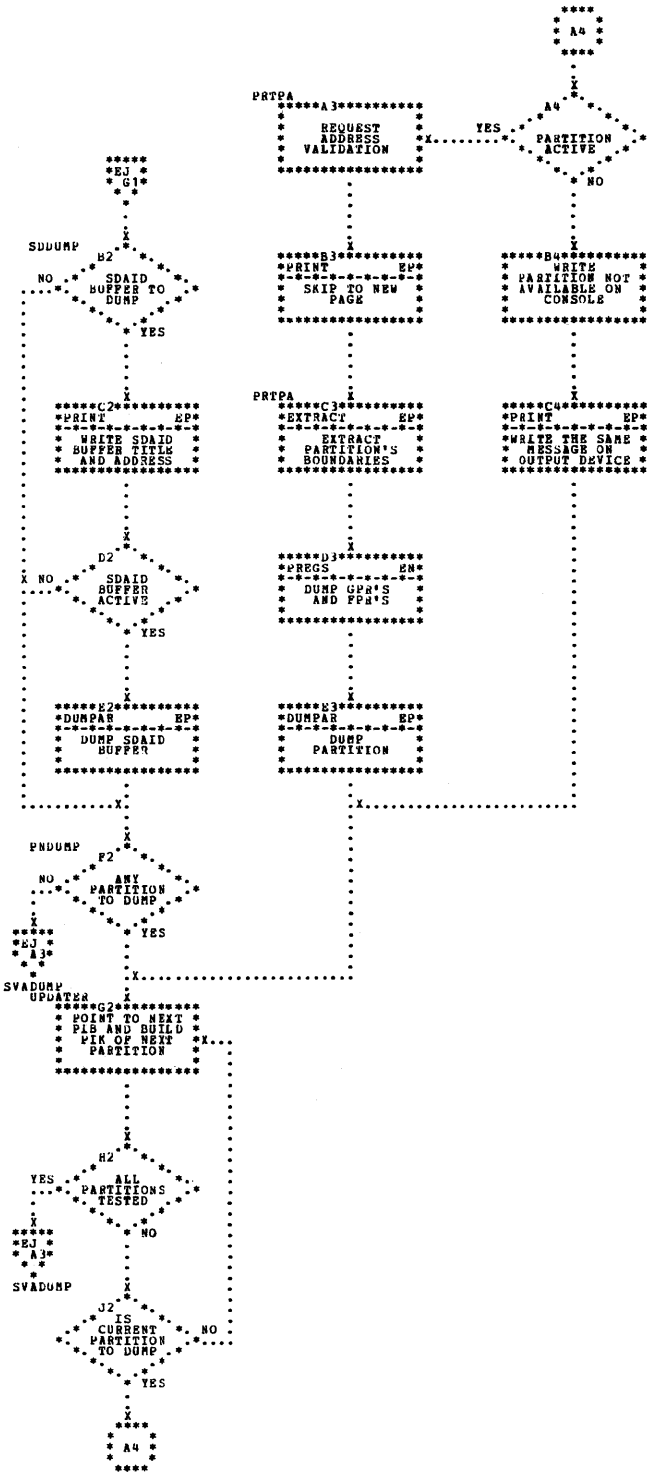


Chart EL. IJBSDUMP - Dump Monitor (Part 10 of 13)
(Refer to Chart 13)

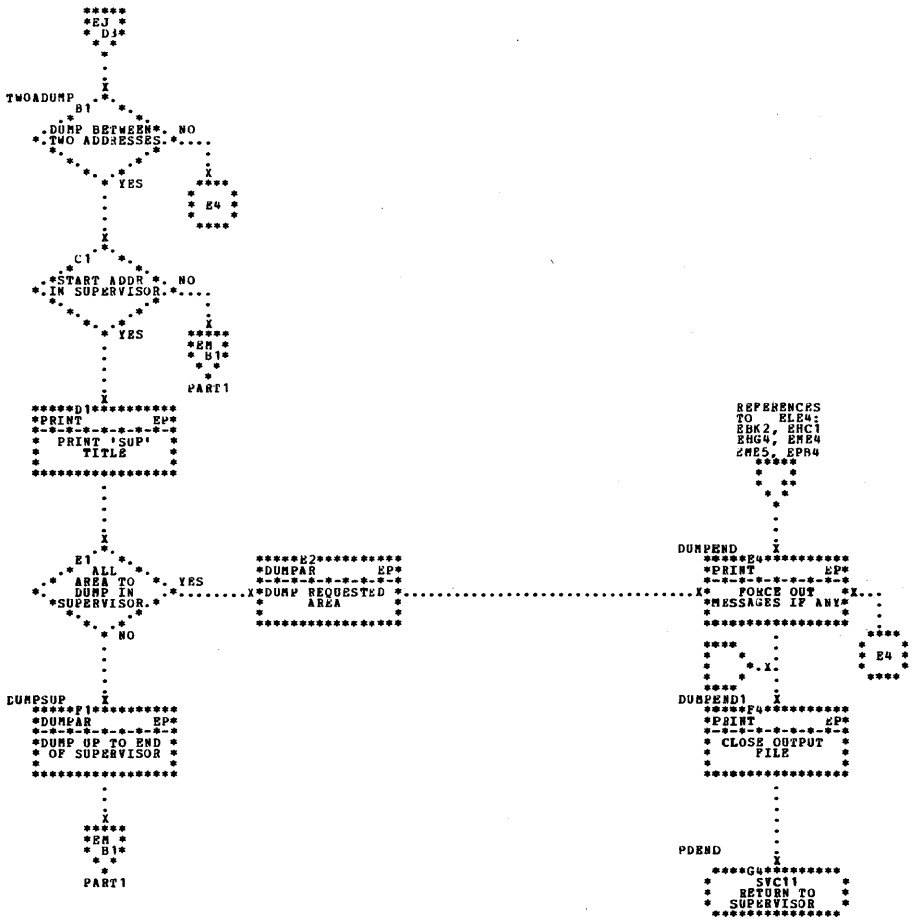


Chart EM. IJBSDUMP - Dump Monitor (Part 11 of 13)
(Refer to Chart 13)

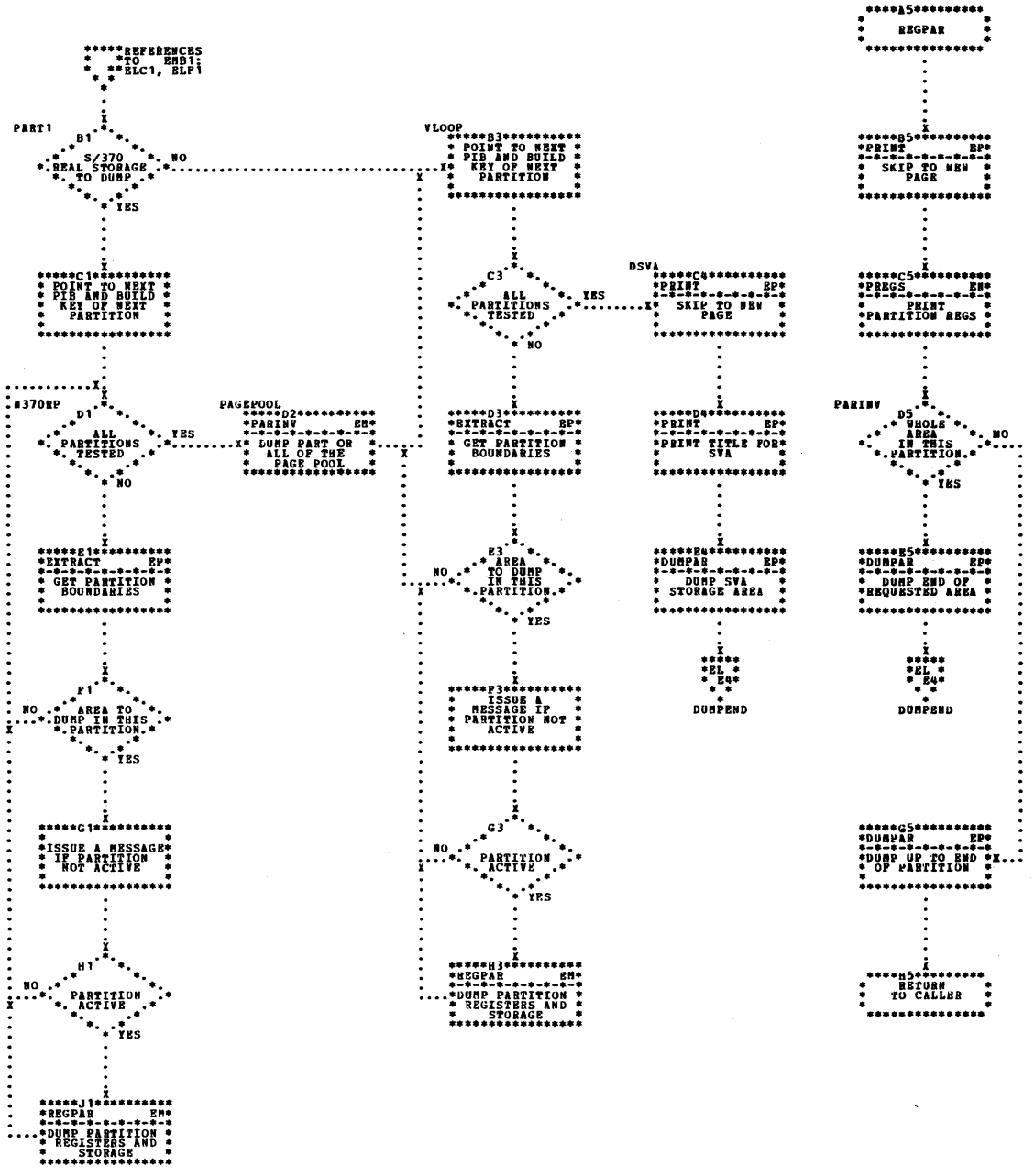


Chart EN. IJBSDUMP - Dump Monitor (Part 12 of 13)
(Refer to Chart 13)

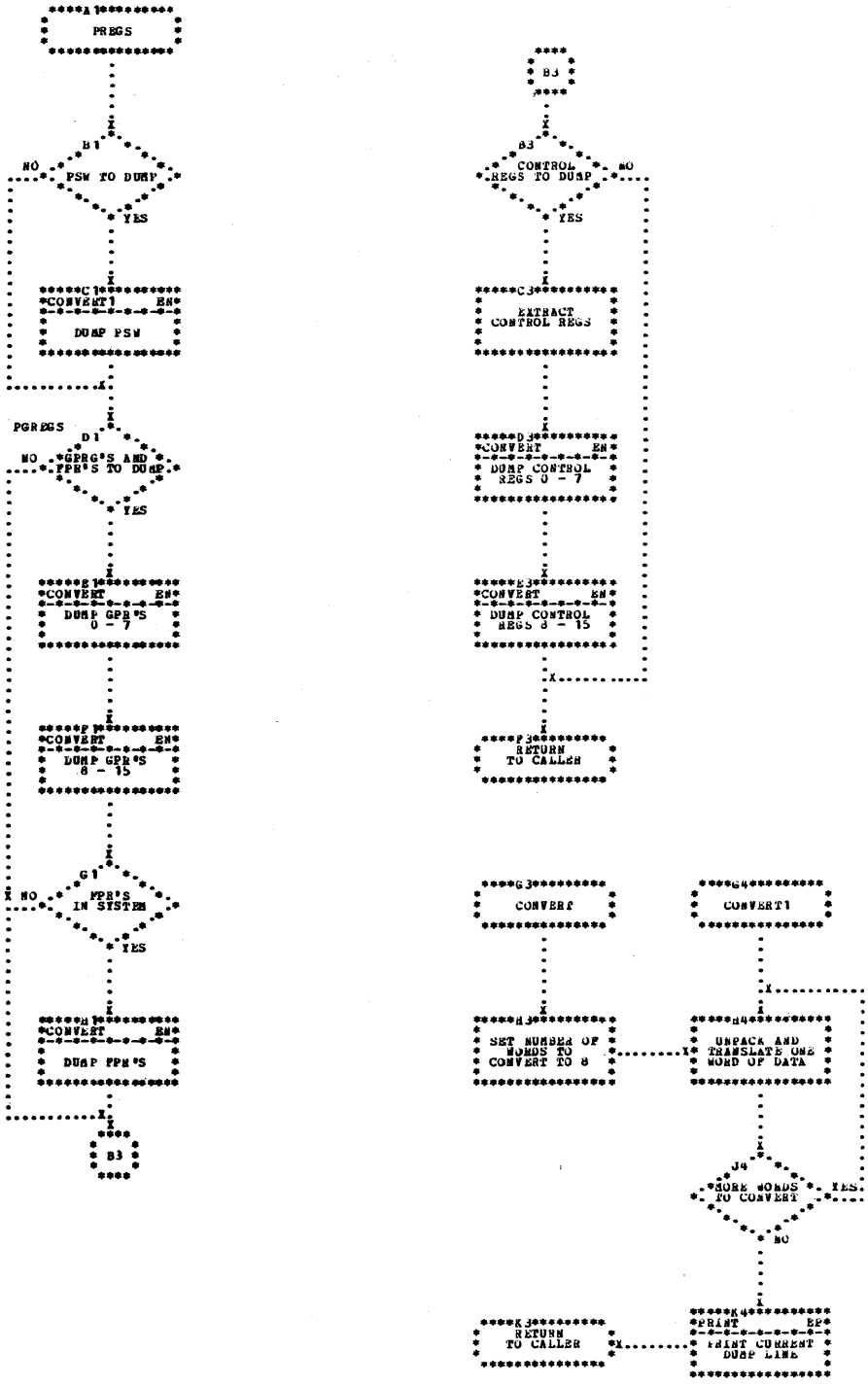


Chart EP. IJBSDUMP - Dump Monitor (Part 13 of 13)
(Refer to Chart 13)

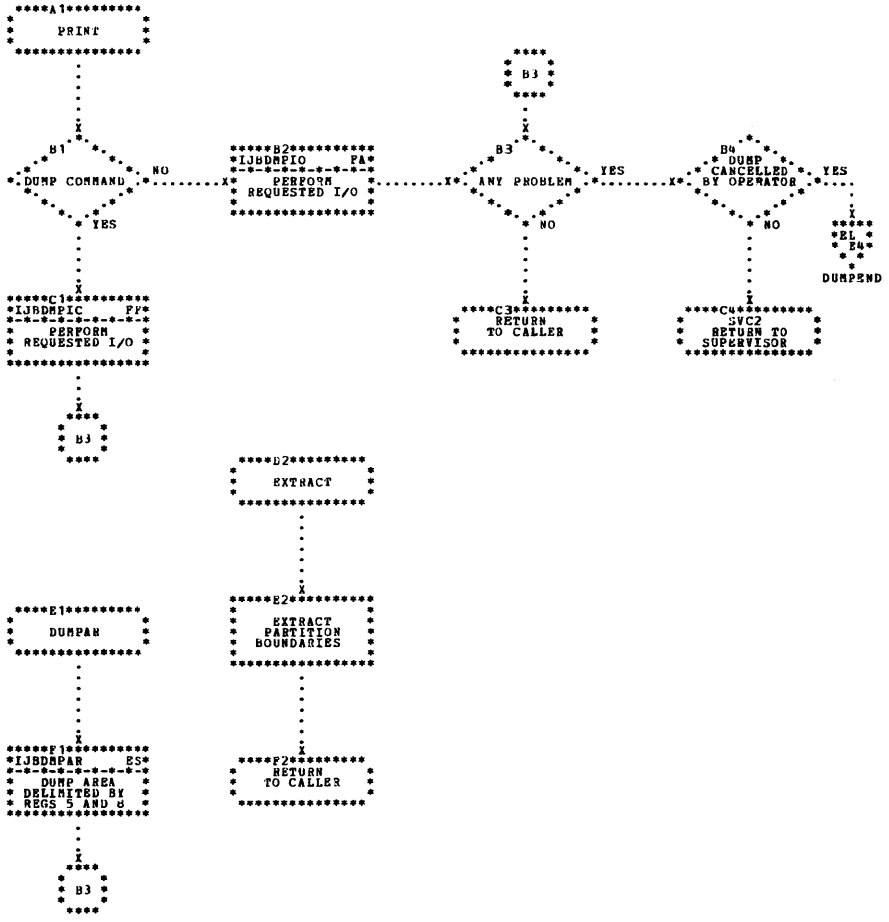


Chart EQ. IJBDMPPA - Dump Supervisor Control Blocks (Part 1 of 2)

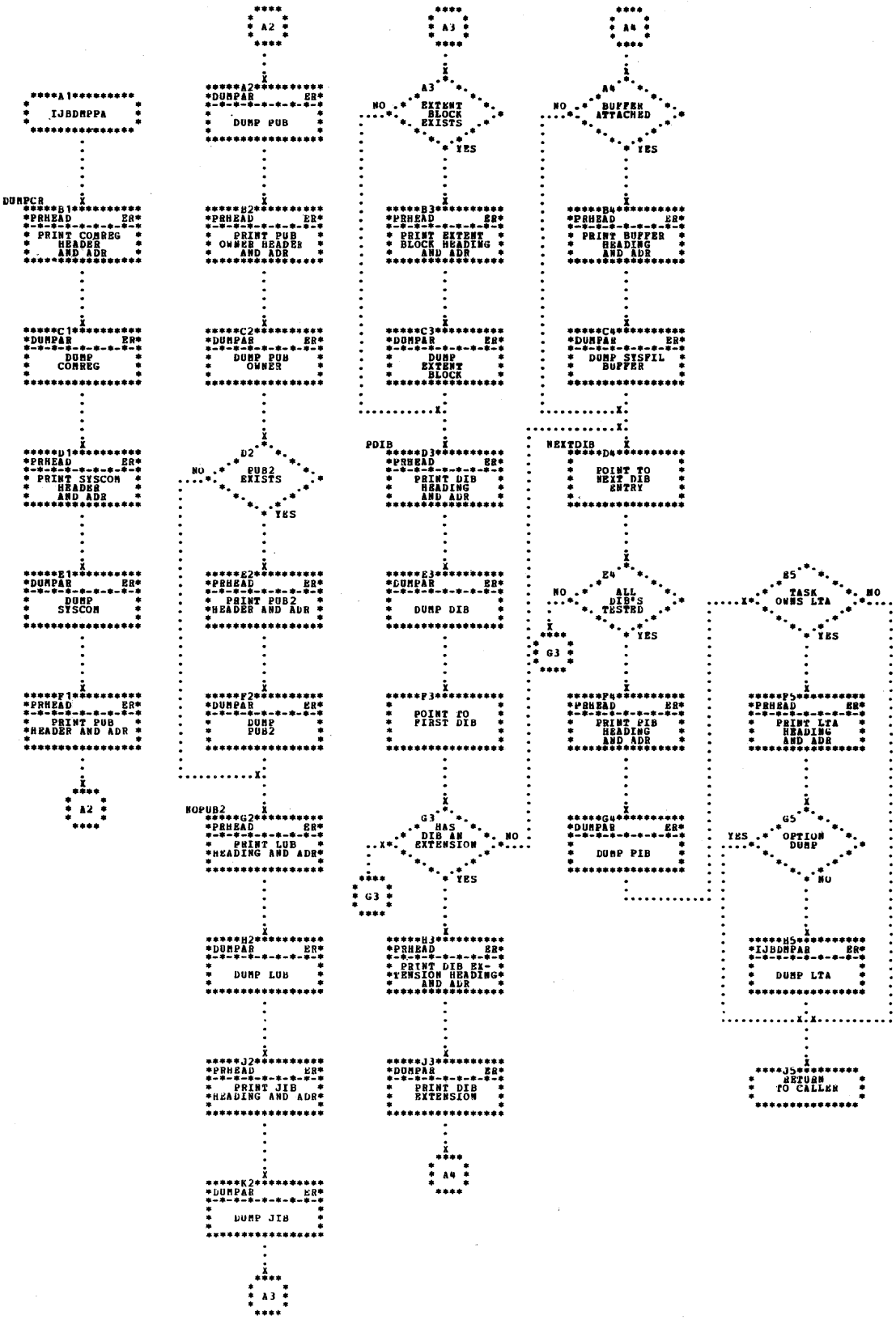


Chart ER. IJBDMPPA - Dump Supervisor Control Blocks (Part 2 of 2)

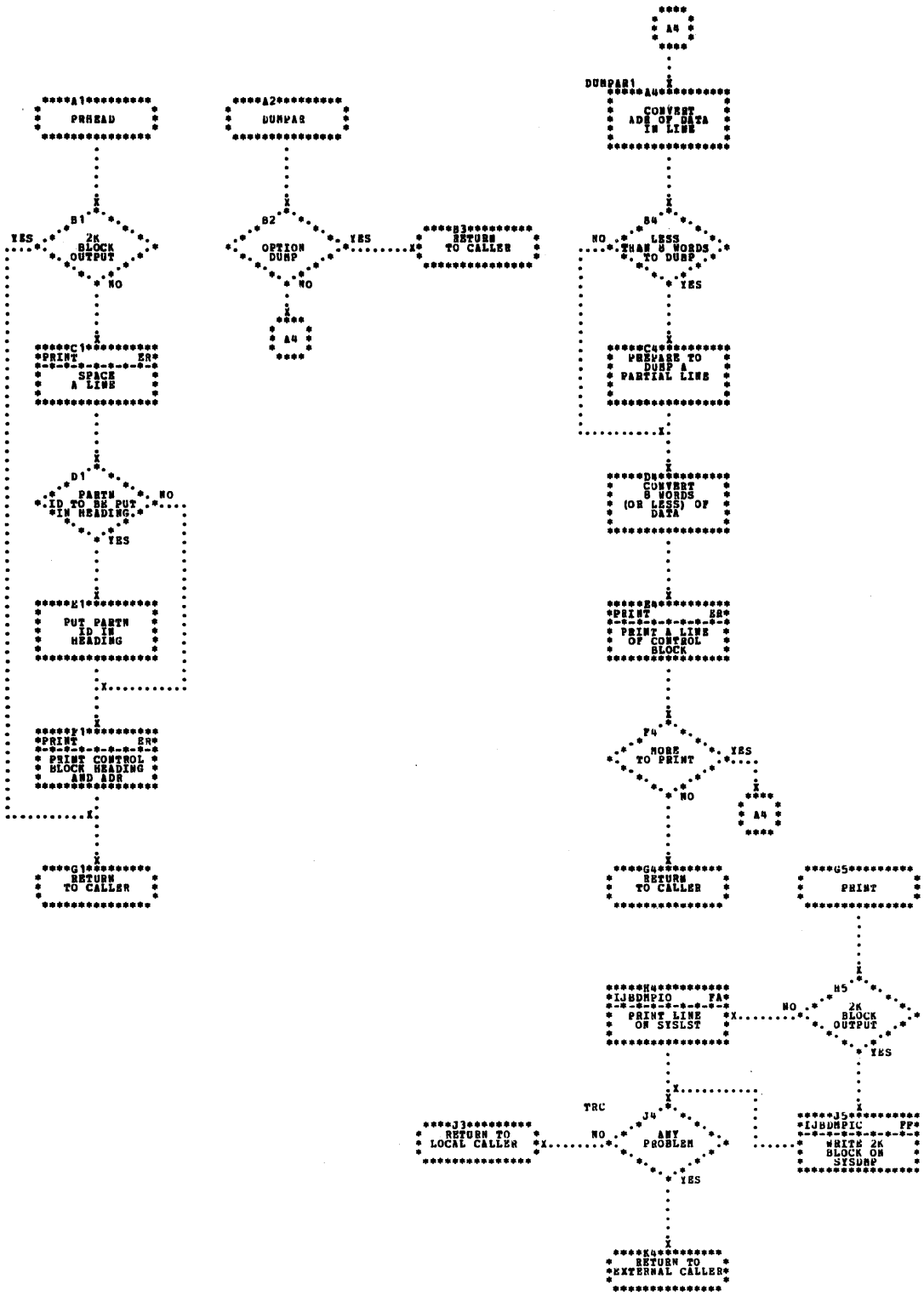


Chart ES. IJBDMPAR - Storage Dump Routine (Part 1 of 5)
(Refer to Chart 13)

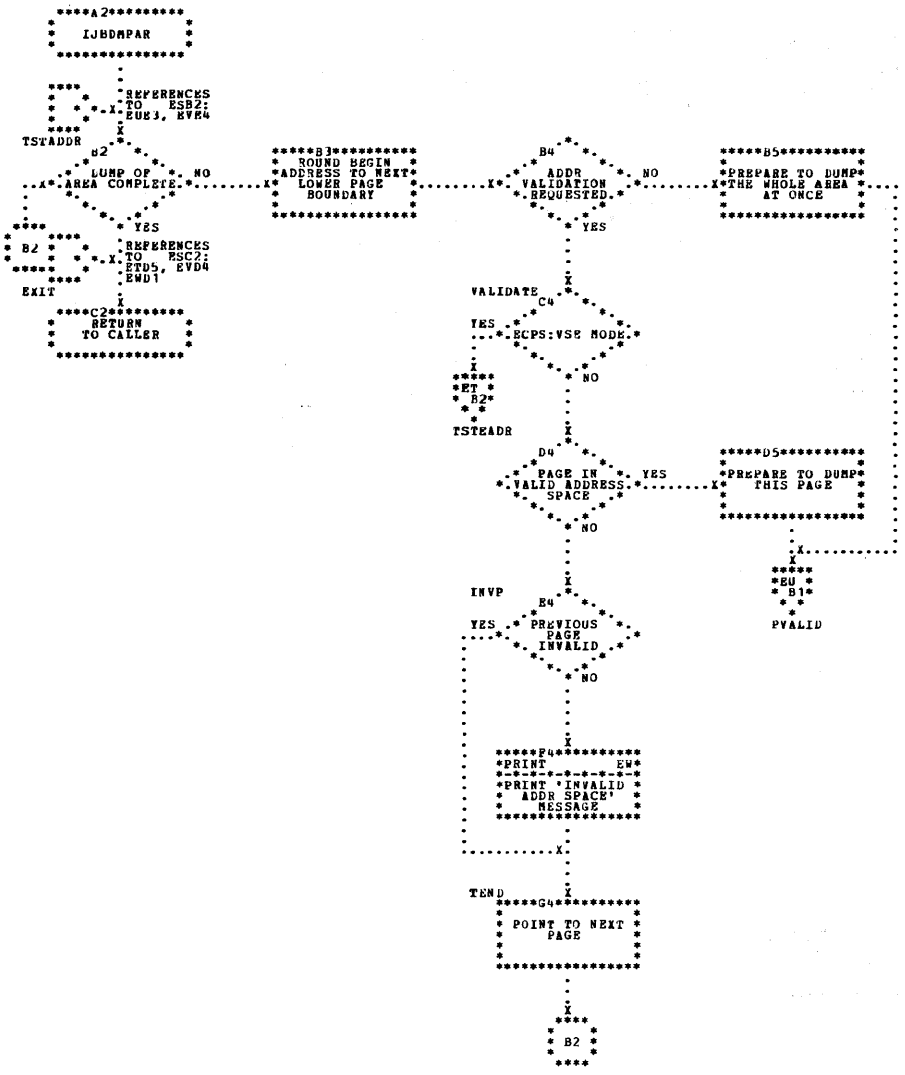


Chart ET. IJBDMPAR - Storage Dump Routine (Part 2 of 5)
(Refer to Chart 13)

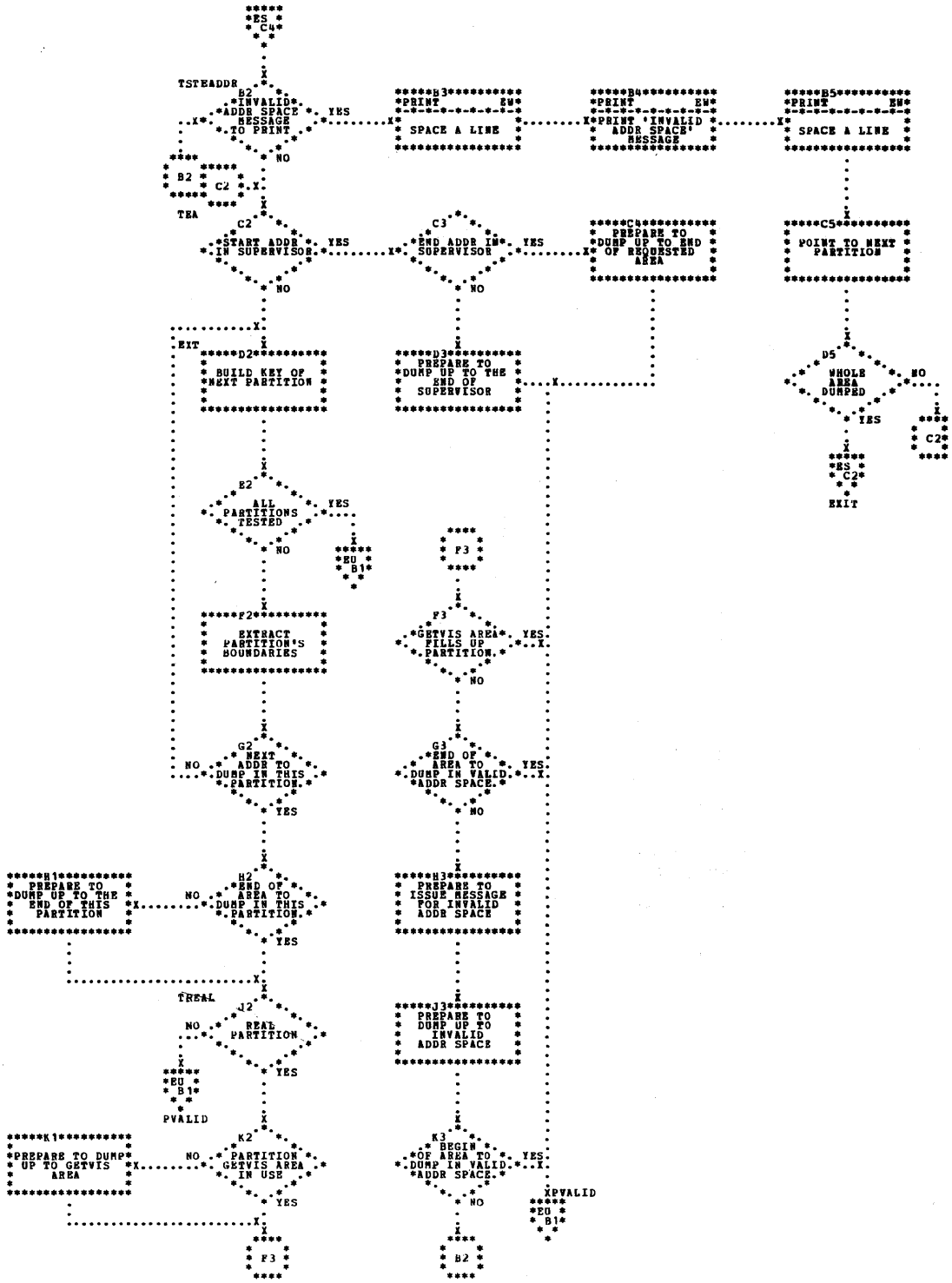


Chart EU. IJBDMPAR - Storage Dump Routine (Part 3 of 5)
(Refer to Chart 13)

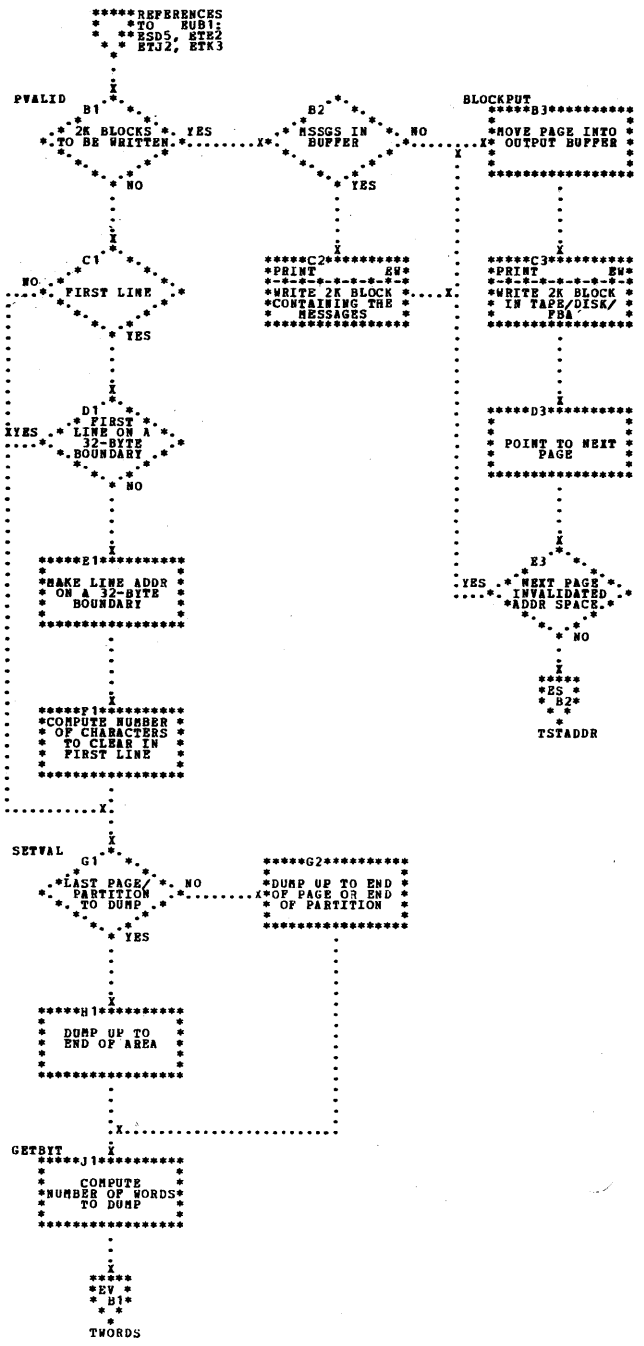


Chart EV. IJBDMPAR - Storage Dump Routine (Part 4 of 5)
(Refer to Chart 13)

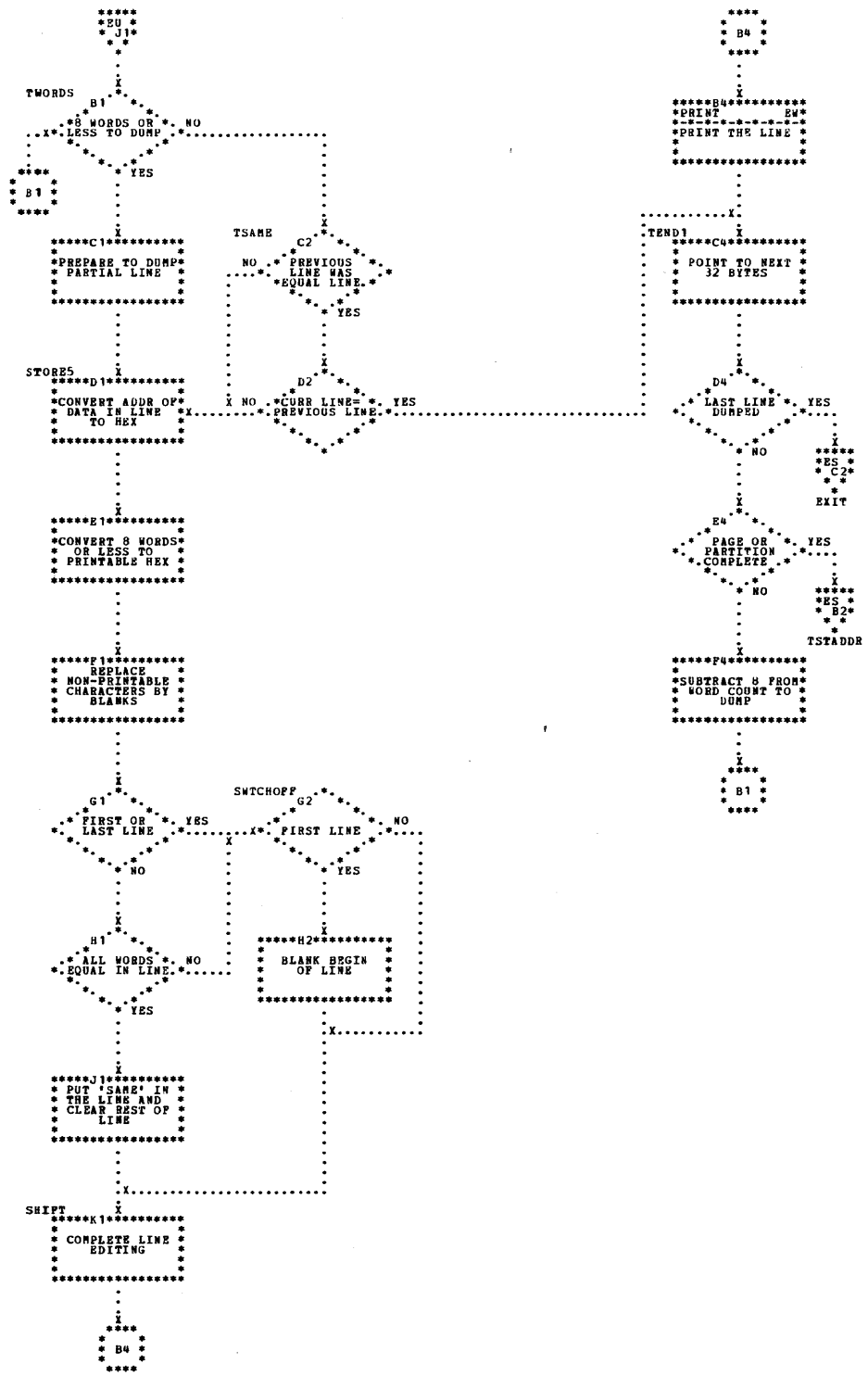


Chart EW. IJBDMPAR - Storage Dump Routine (Part 5 of 5)
(Refer to Chart 13)

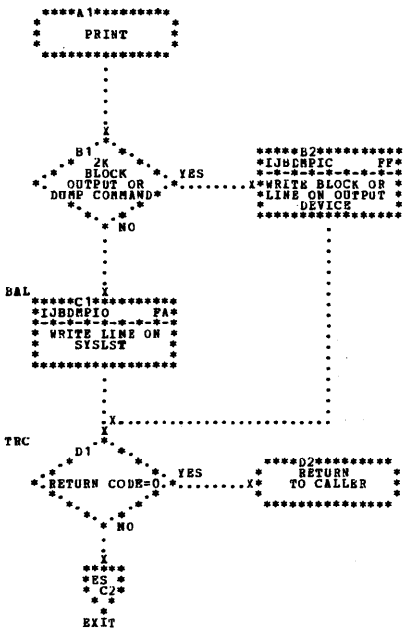


Chart FA. IJBDMPIO - System Dump I/O Routine (Part 1 of 3)
(Refer to Chart 13)

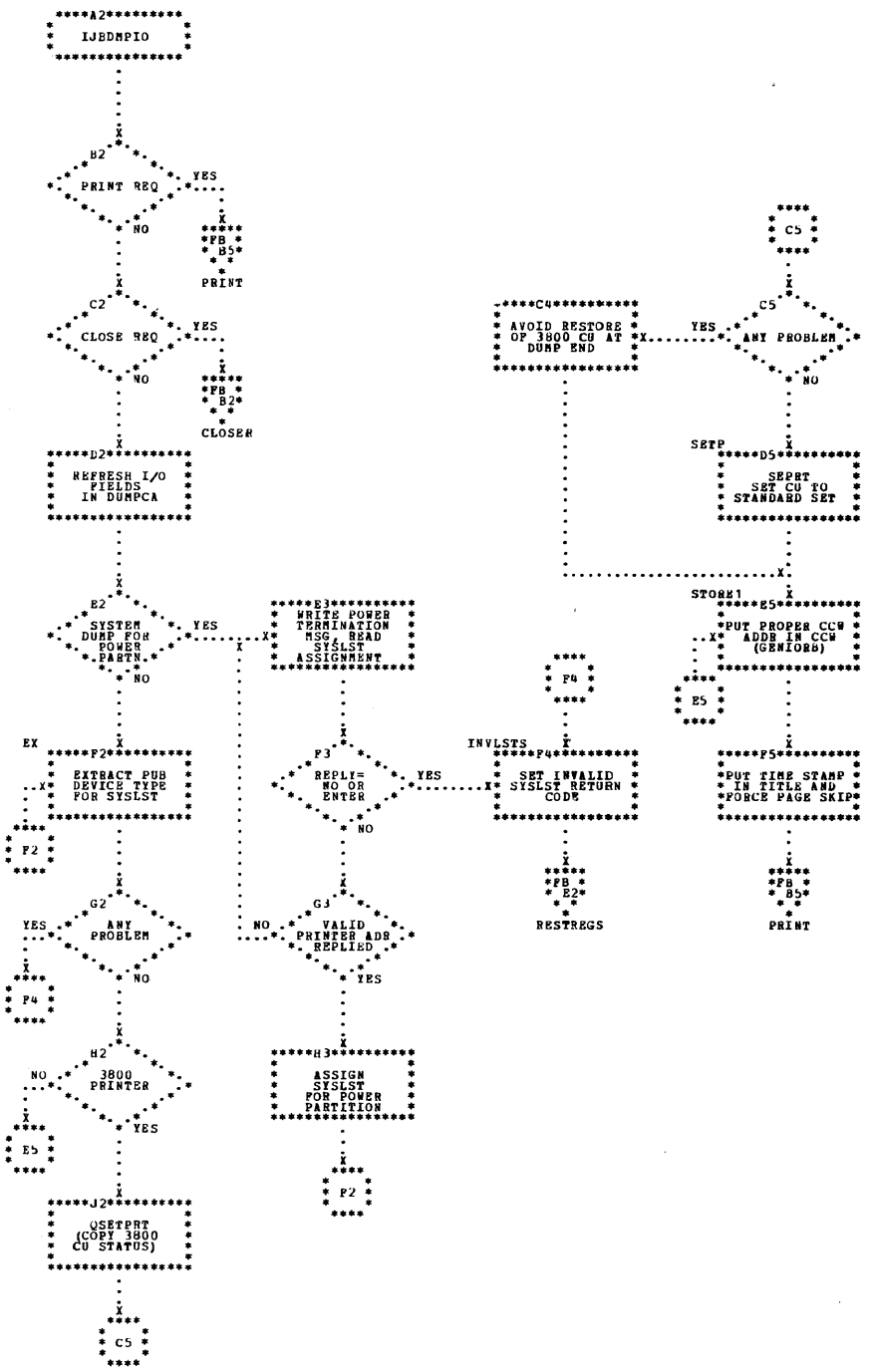


Chart FB. IJBDMPIO - System Dump I/O Routine (Part 2 of 3)
(Refer to Chart 13)

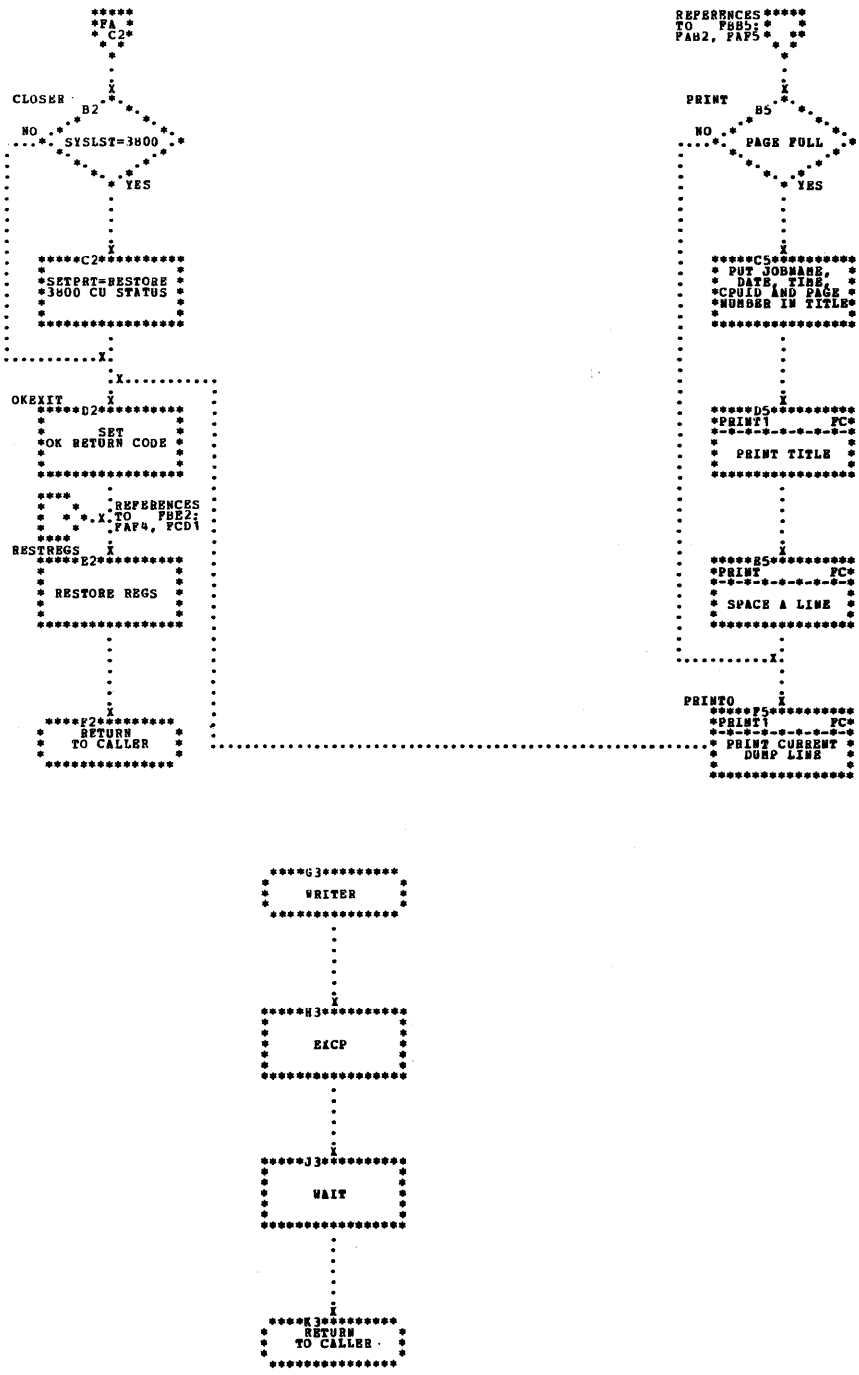


Chart FC. IJBDMPIO - System Dump I/O Routine (Part 3 of 3)
(Refer to Chart 13)

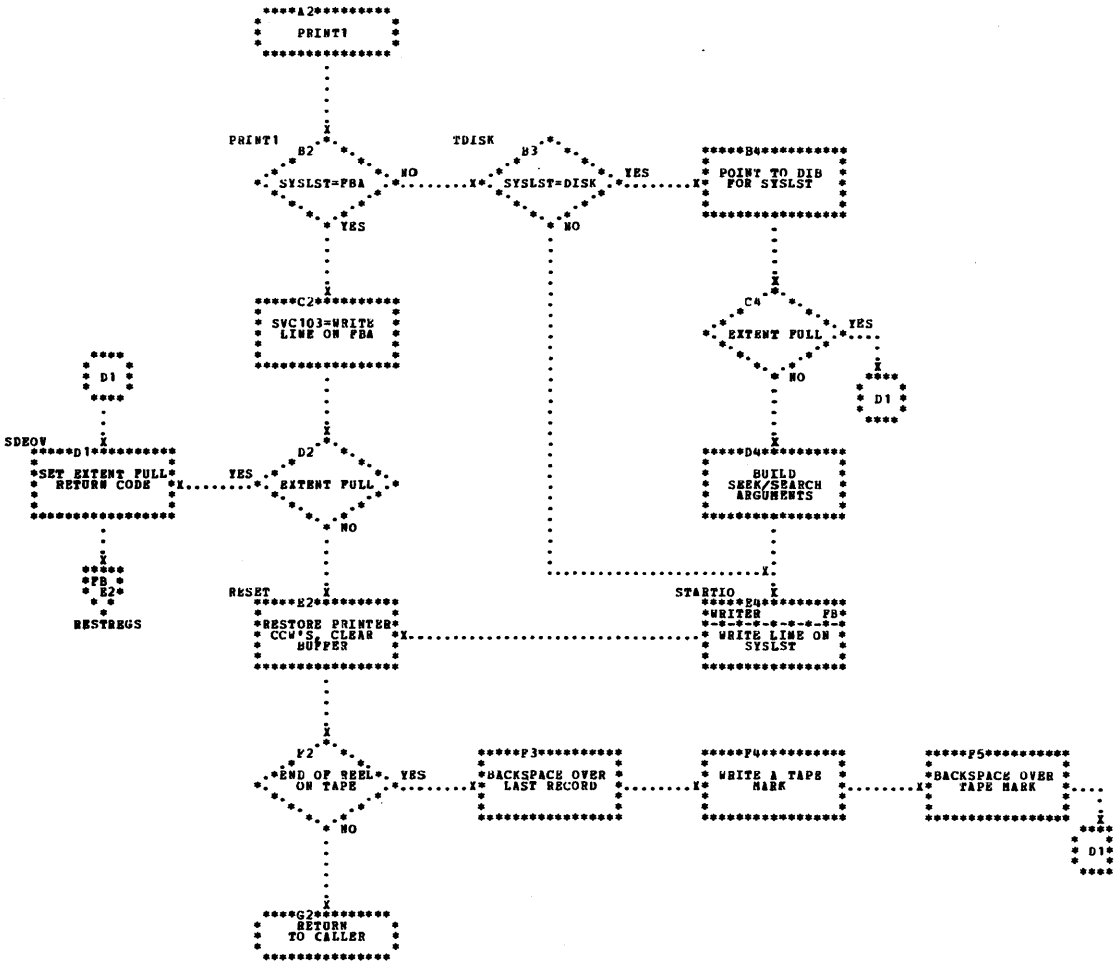


Chart FF. IJBDMPIC - DUMP Command I/O Routine (Part 1 of 5)
(Refer to Chart 13)

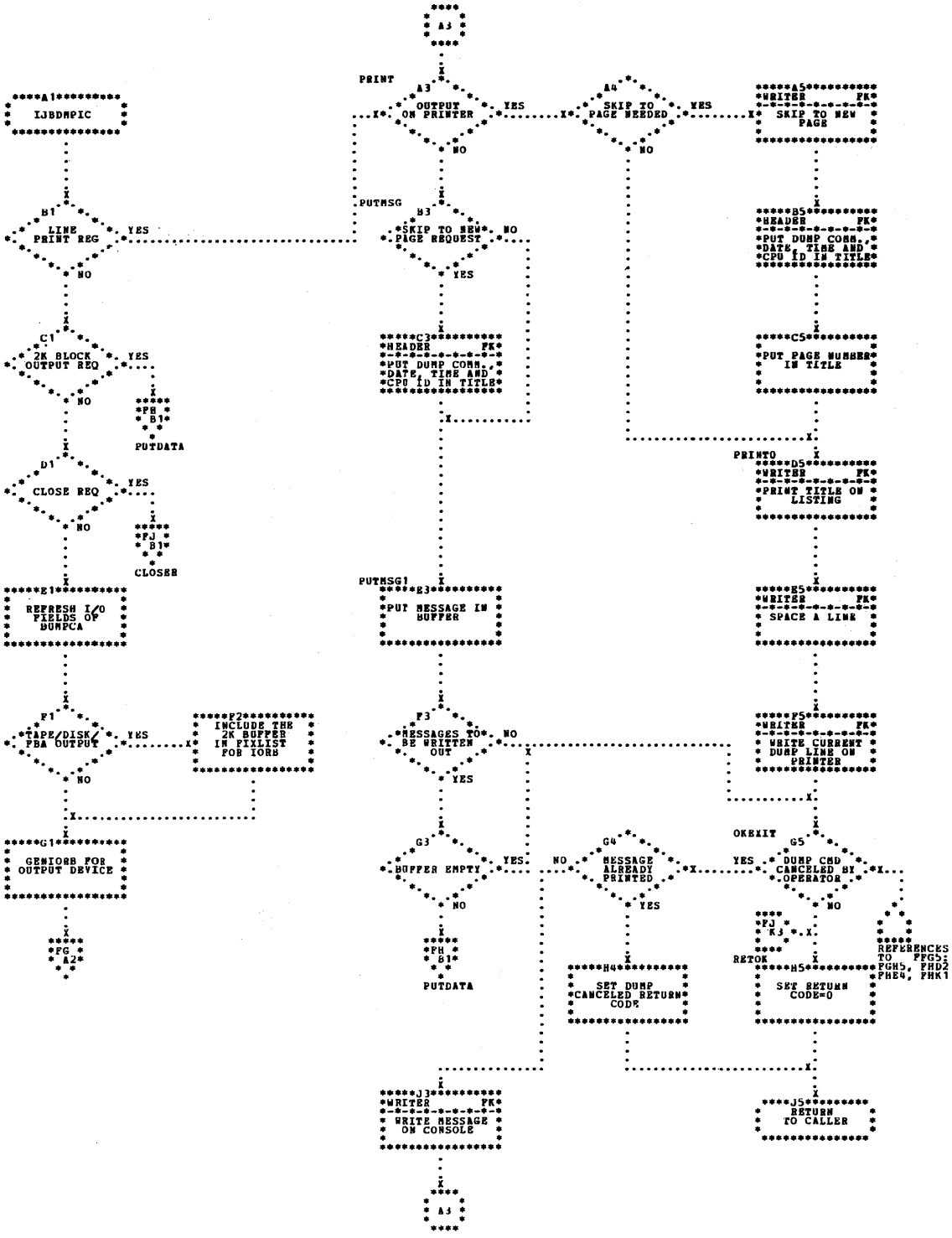


Chart FG. IJBDMPIC - DUMP Command I/O Routine (Part 2 of 5)
(Refer to Chart 13)

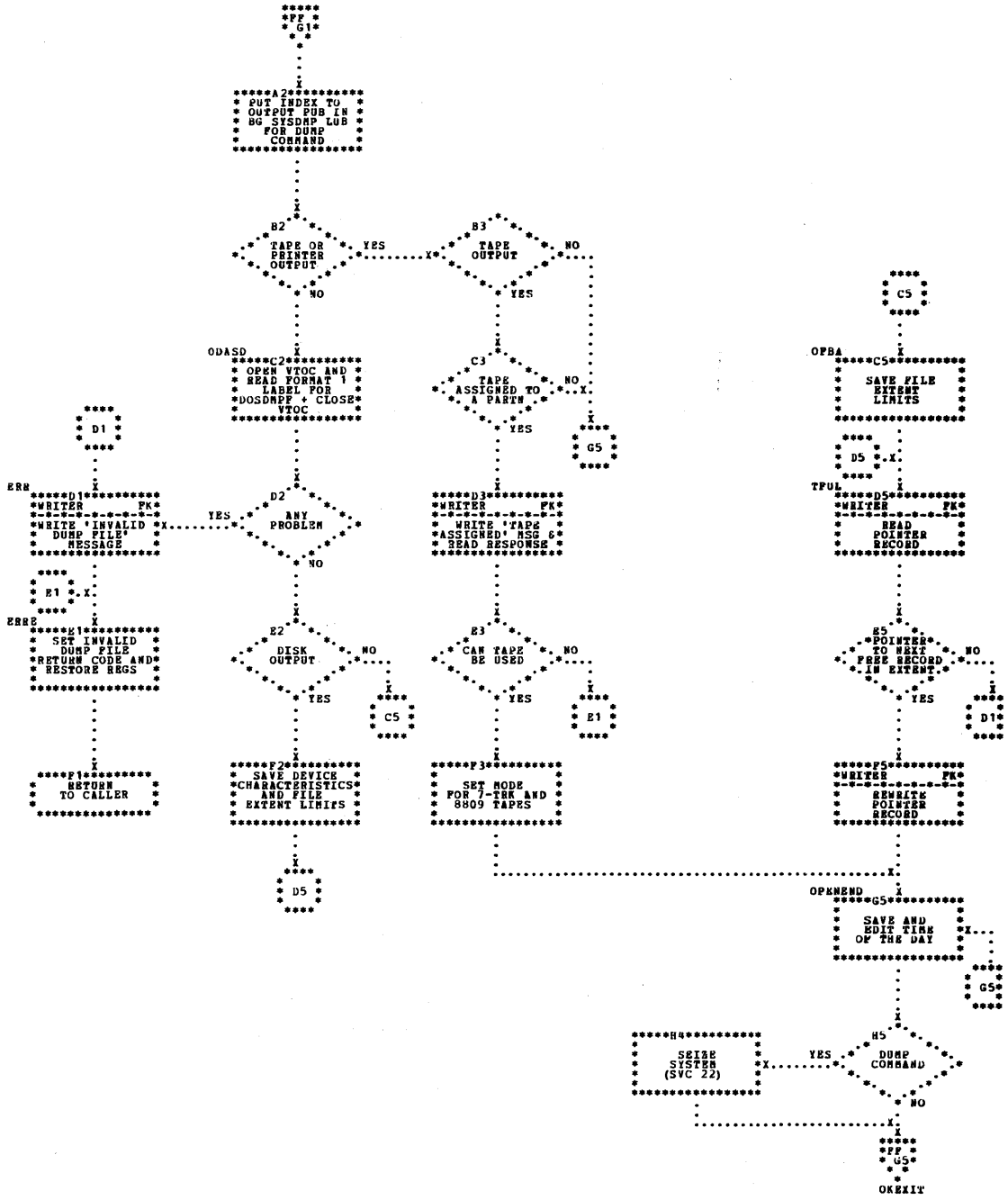


Chart FH. IJBDMPIC - DUMP Command I/O Routine (Part 3 of 5)
(Refer to Chart 13)

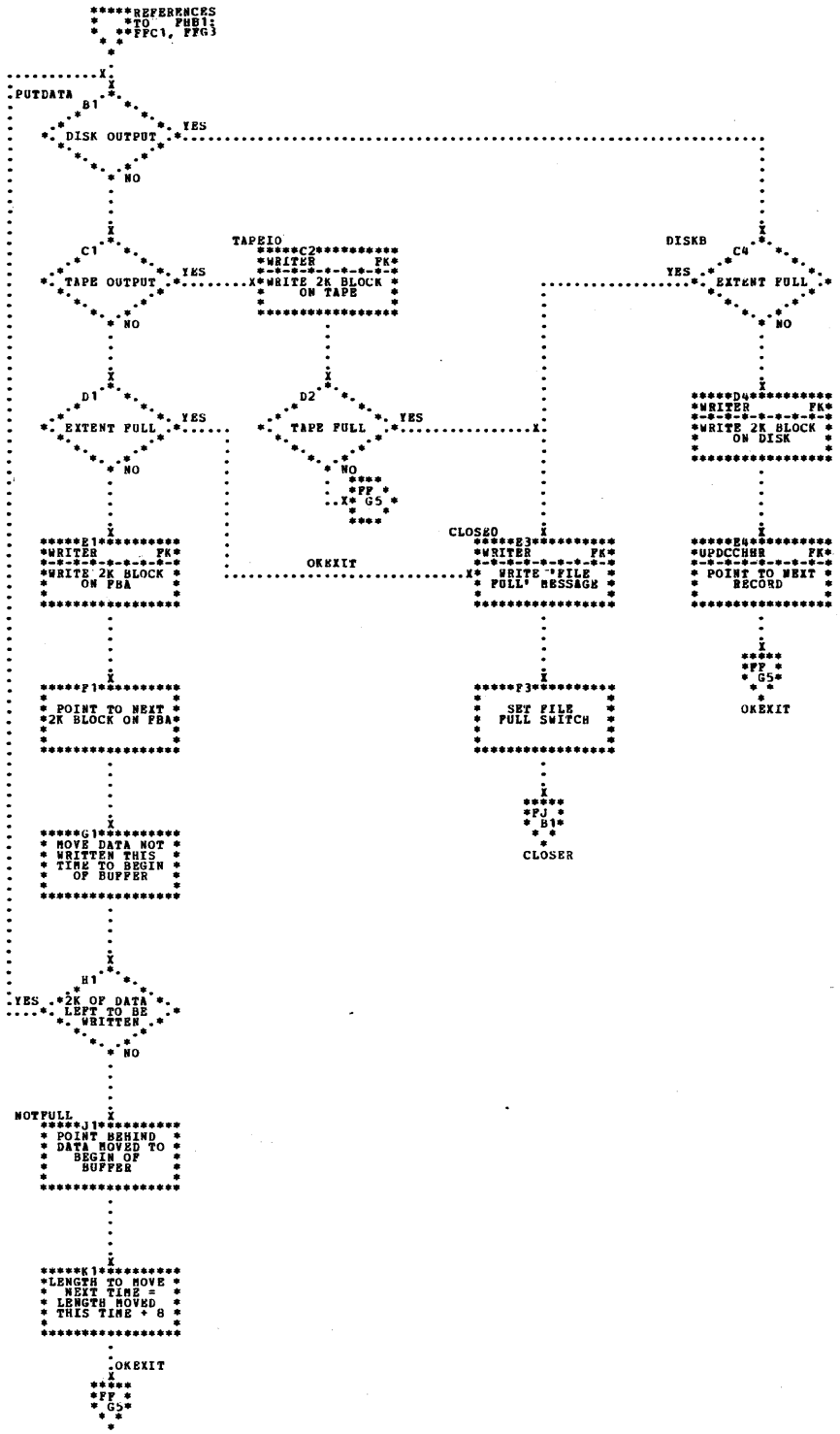


Chart FJ. IJBDMPIC - DUMP Command I/O Routine (Part 4 of 5)
(Refer to Chart 13)

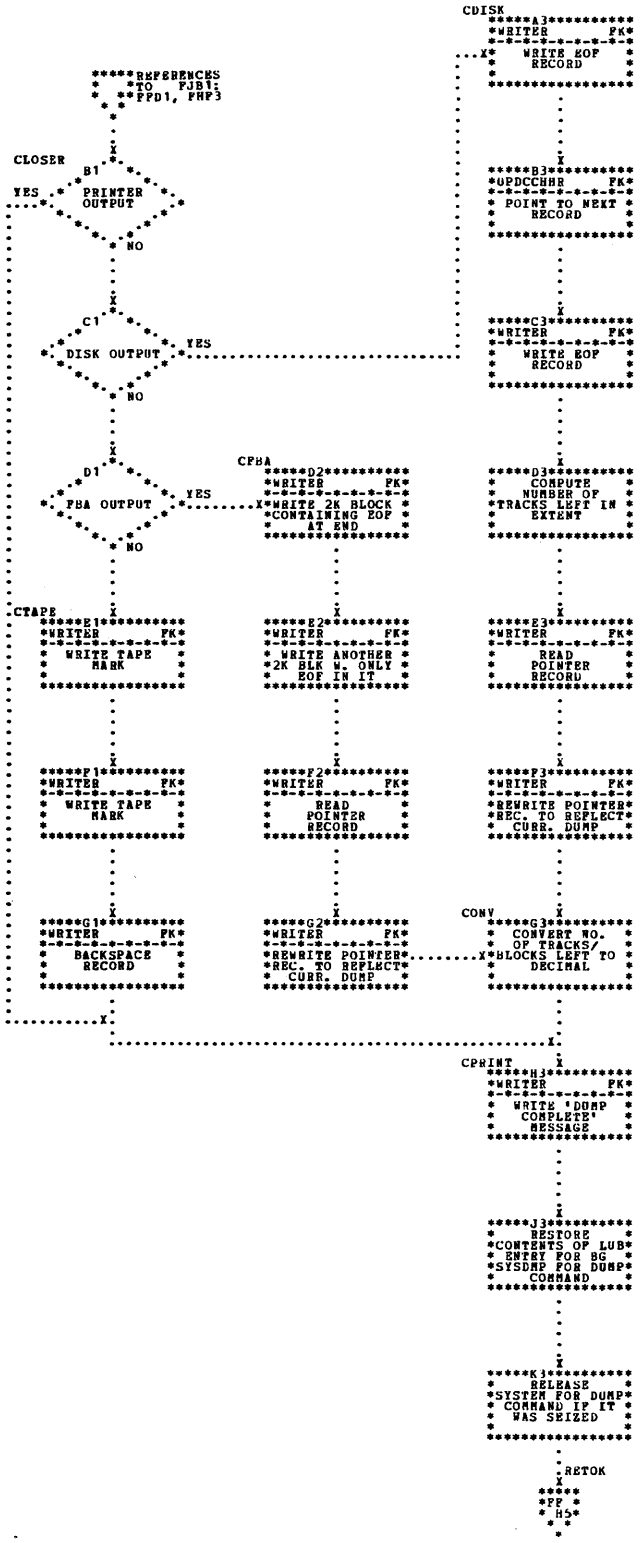


Chart FK. IJBDMPIC - DUMP Command I/O Routine (Part 5 of 5)
(Refer to Chart 13)

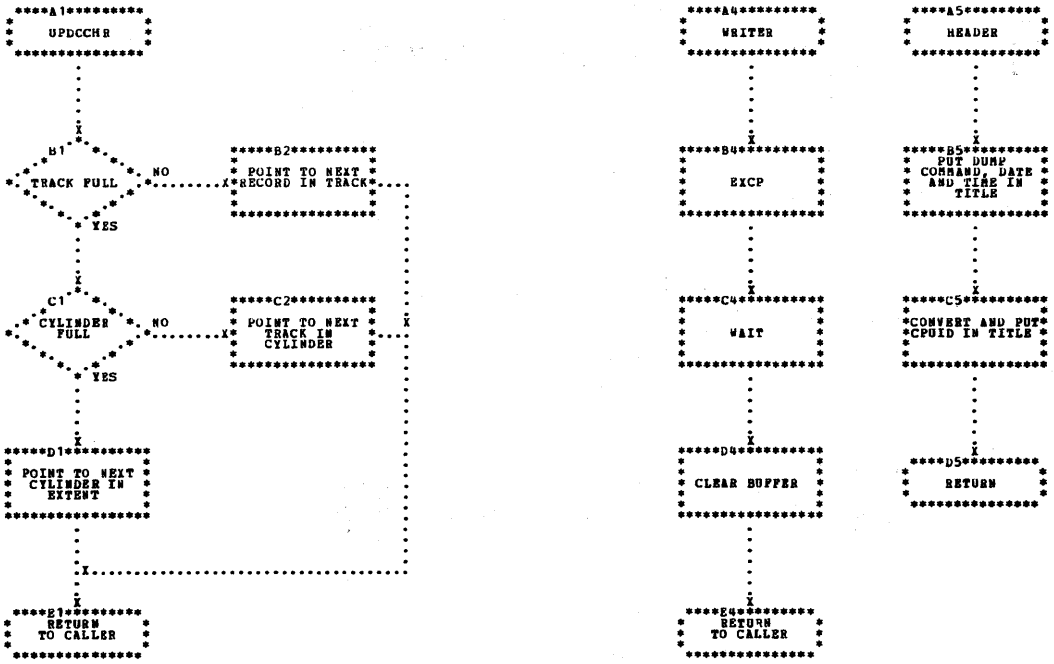


Chart FP. IJBEOJ - Cancel Message Writer (Part 1 of 3)
(Refer to Chart 13)

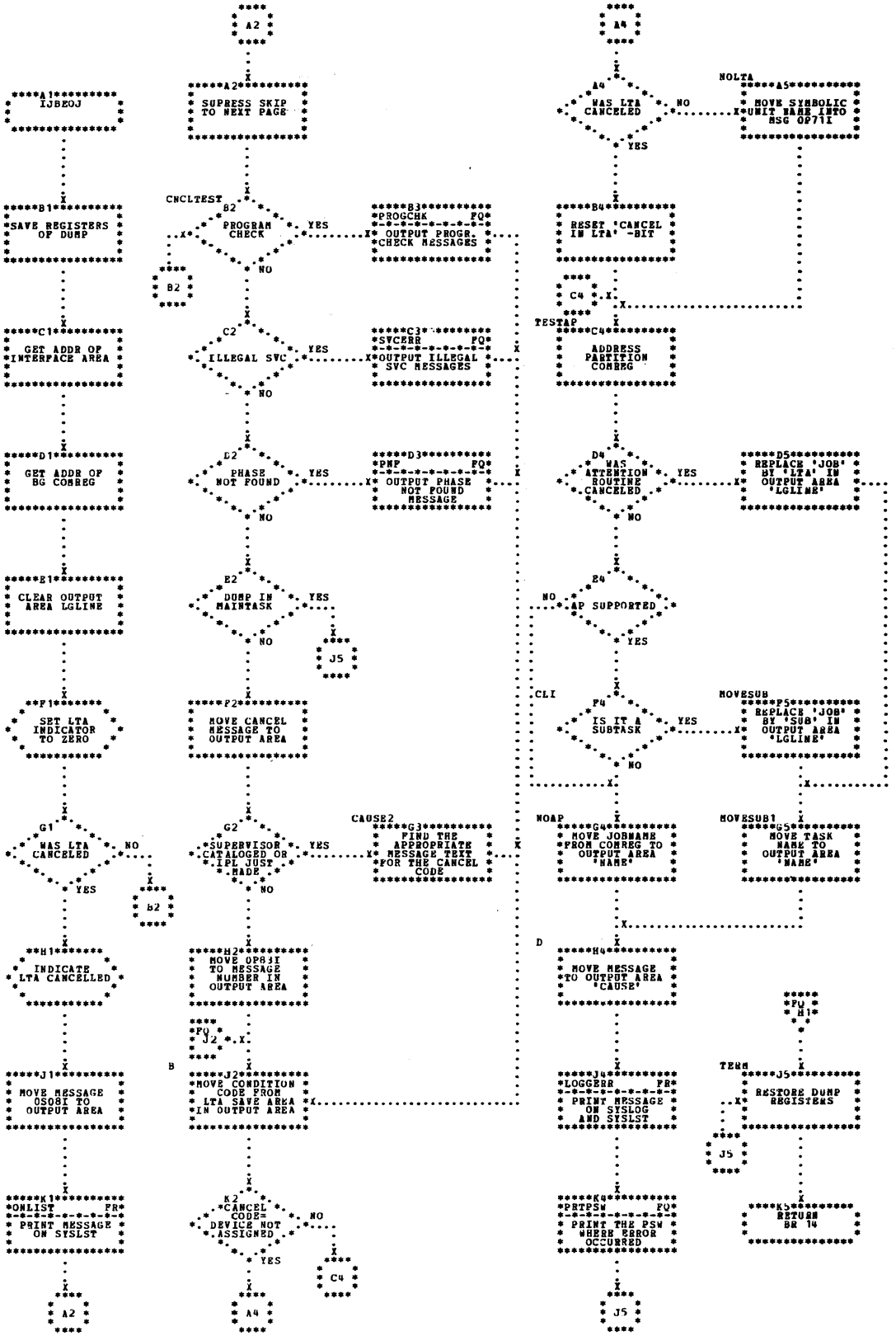


Chart FQ. IJBEOJ - Cancel Message Writer (Part 2 of 3)
(Refer to Chart 13)

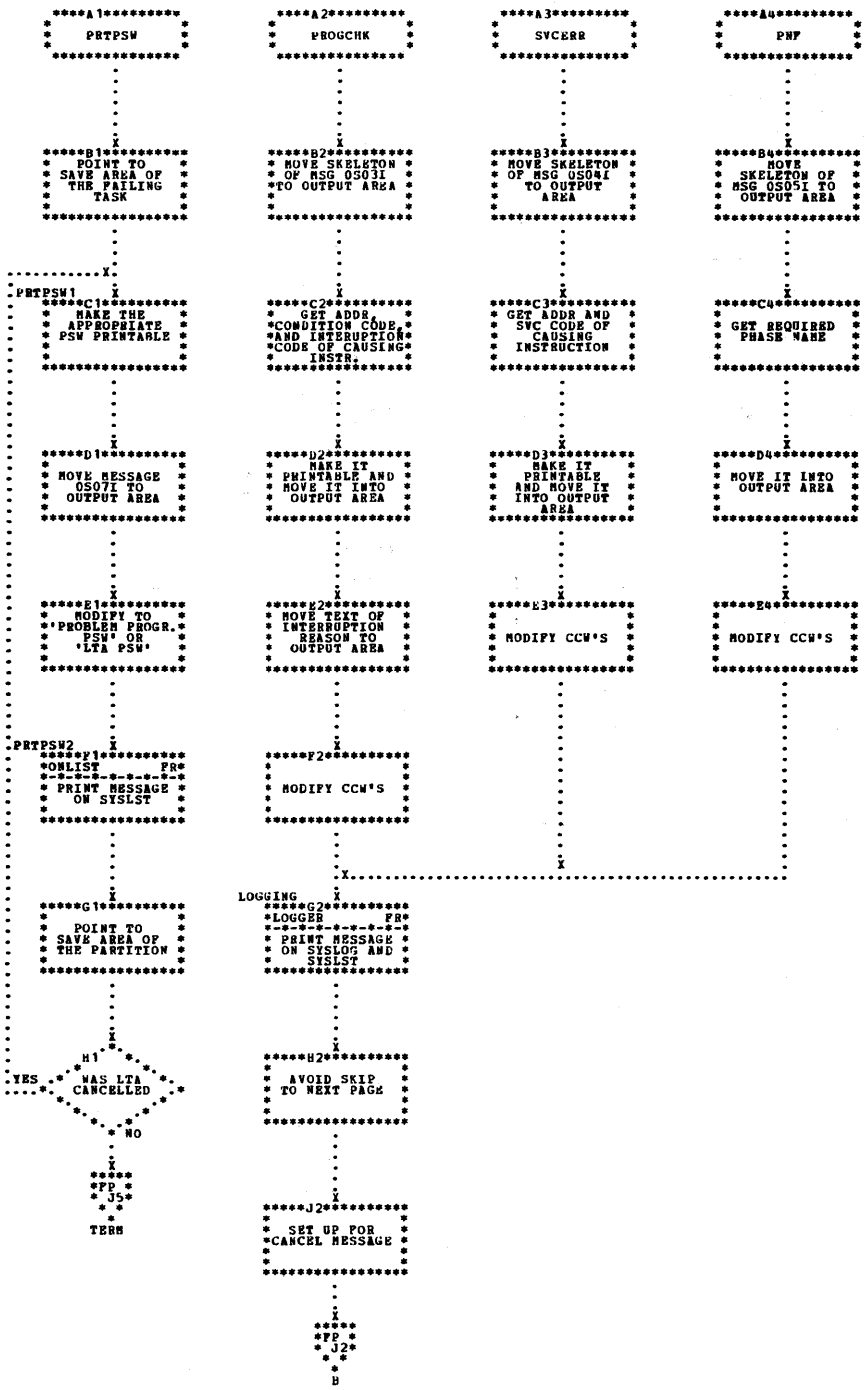


Chart PR. IJBEOJ - Cancel Message Writer (Part 3 of 3)
(Refer to Chart 13)

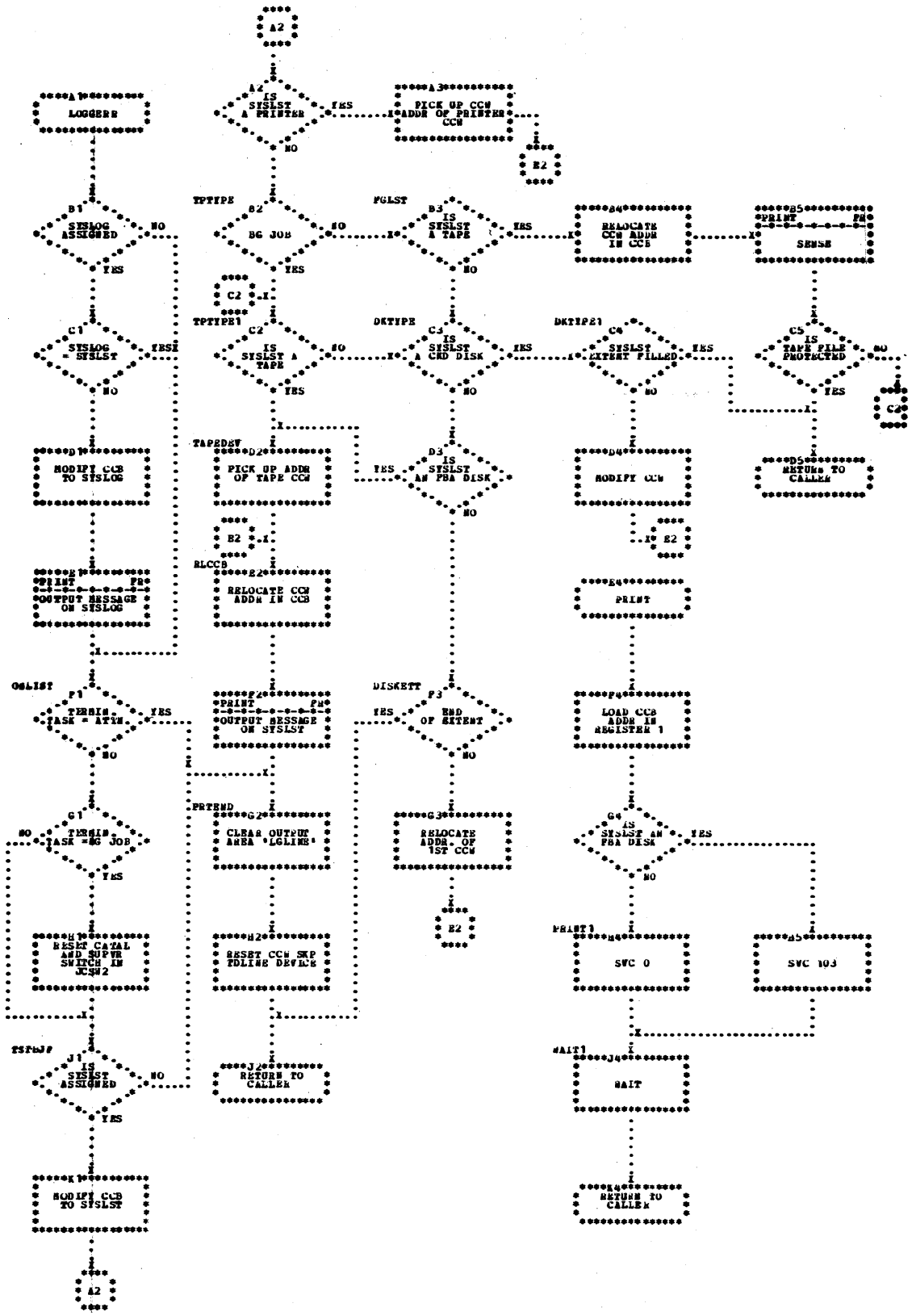


Chart GG. $$BSYSWR - Set up a Write on SYSRES Operation; Move Label Cylinder Address to COMREG

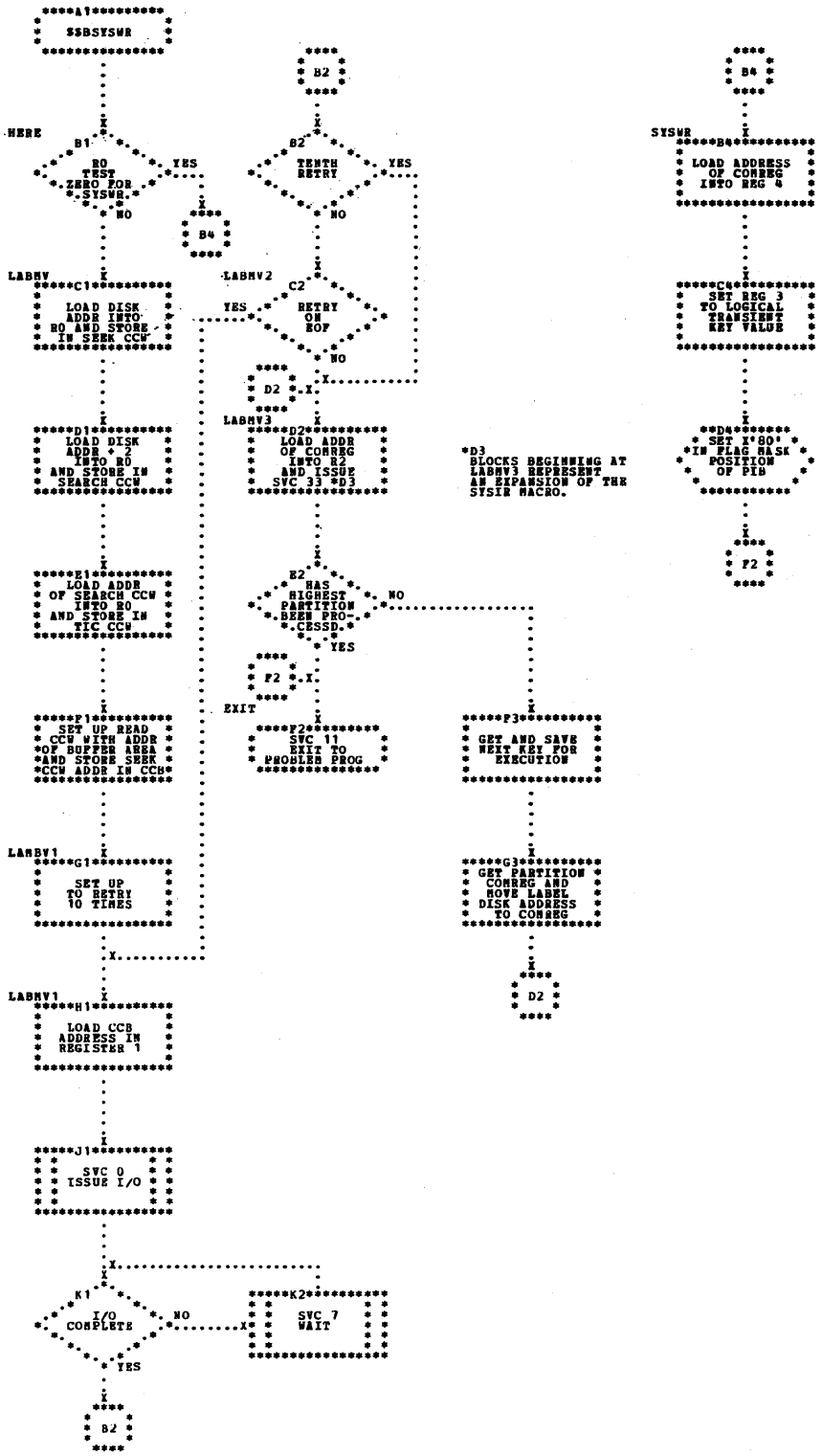


Chart GJ. $$BCCHHR - Core Image Directory Scan

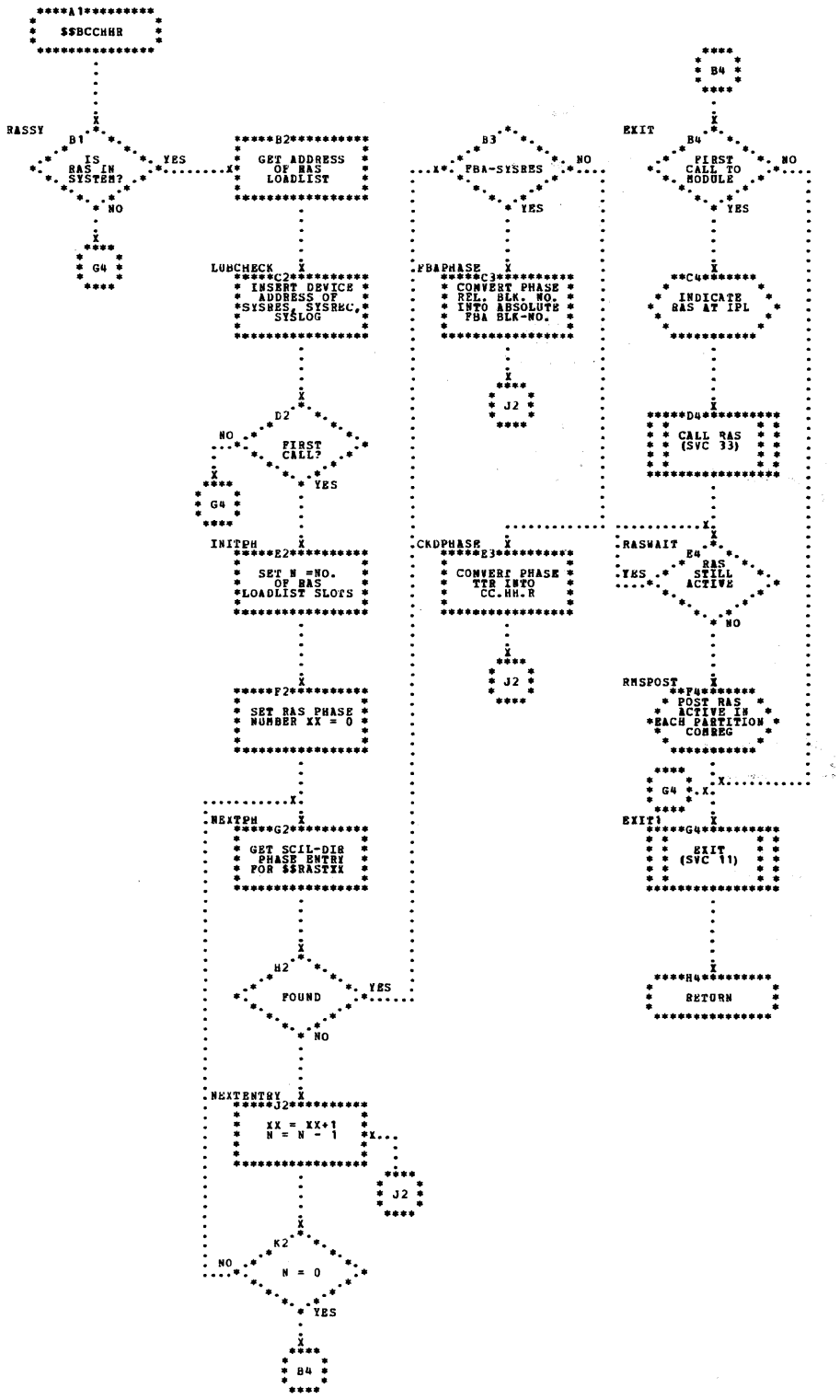


Chart GK. $$BATTF0 - FCB Load Initiation
(Refer to Chart 15)

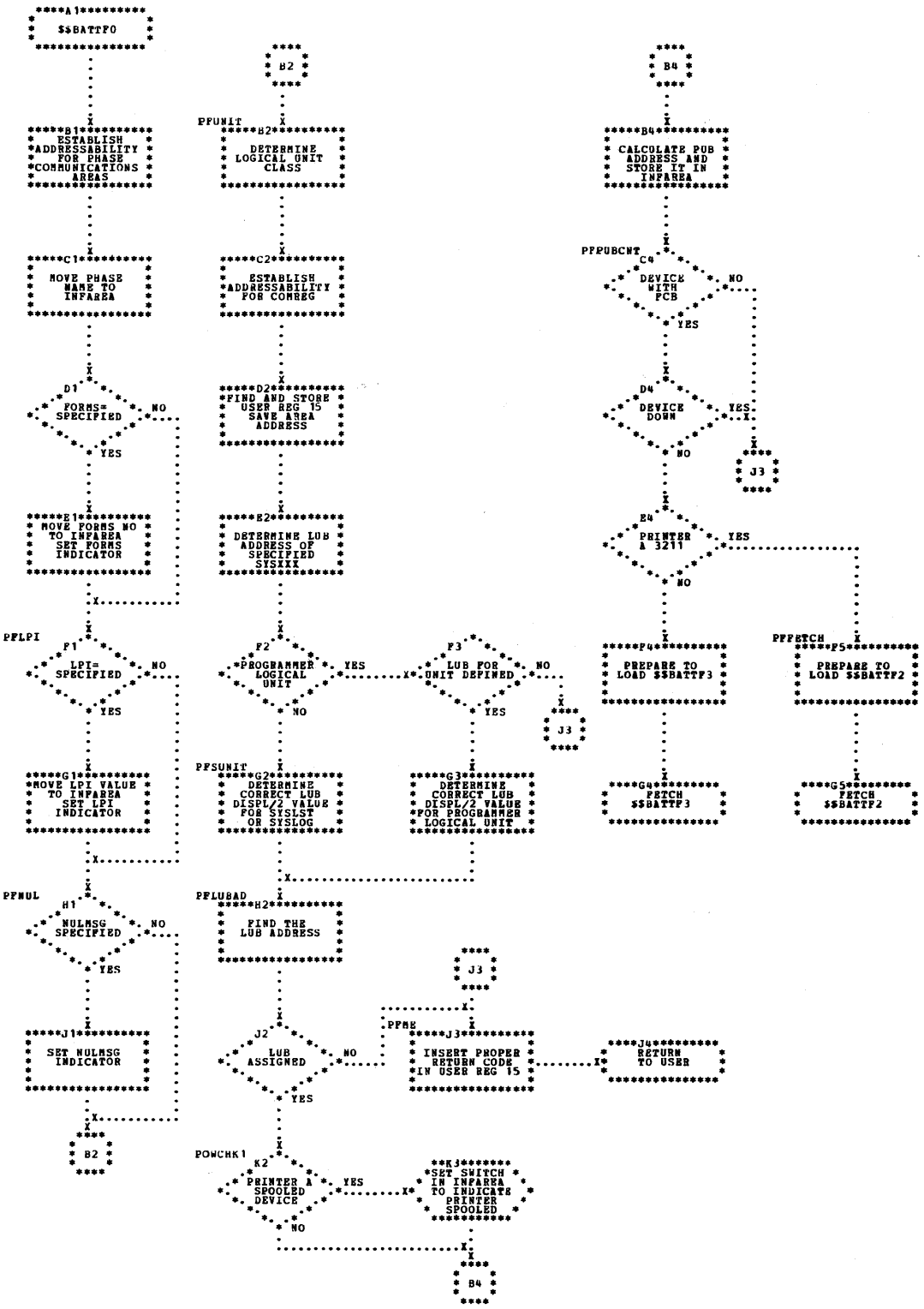


Chart GL. $$BATTF2 - LFCB Macro Execution for PRT1
(Refer to Chart 15)

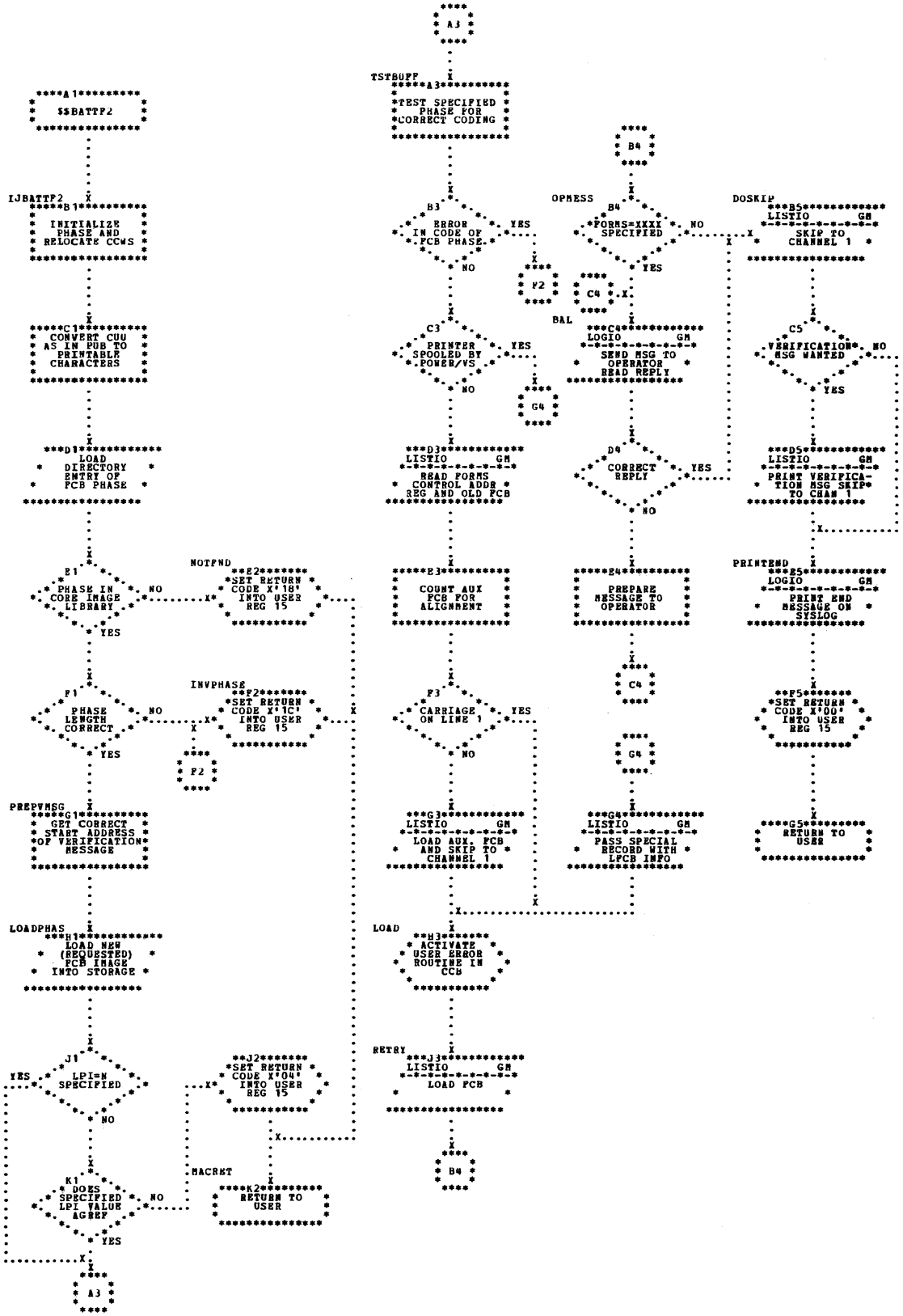


Chart GM. $$BATTF2 - LFCB Macro Execution Subroutines
(Refer to Chart 15)

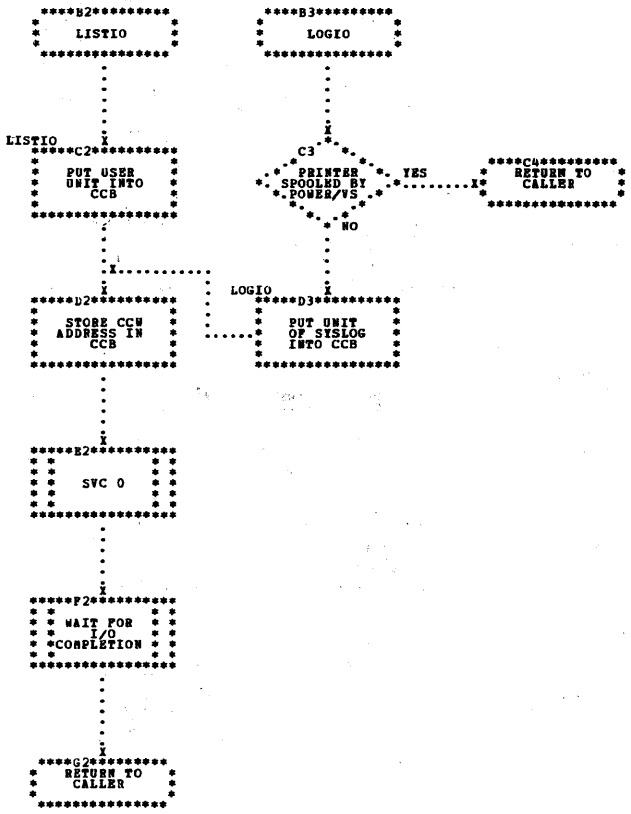


Chart GN. $$BATTF3 - LFCB Macro Execution for 3203 and 5203
(Refer to Chart 15)

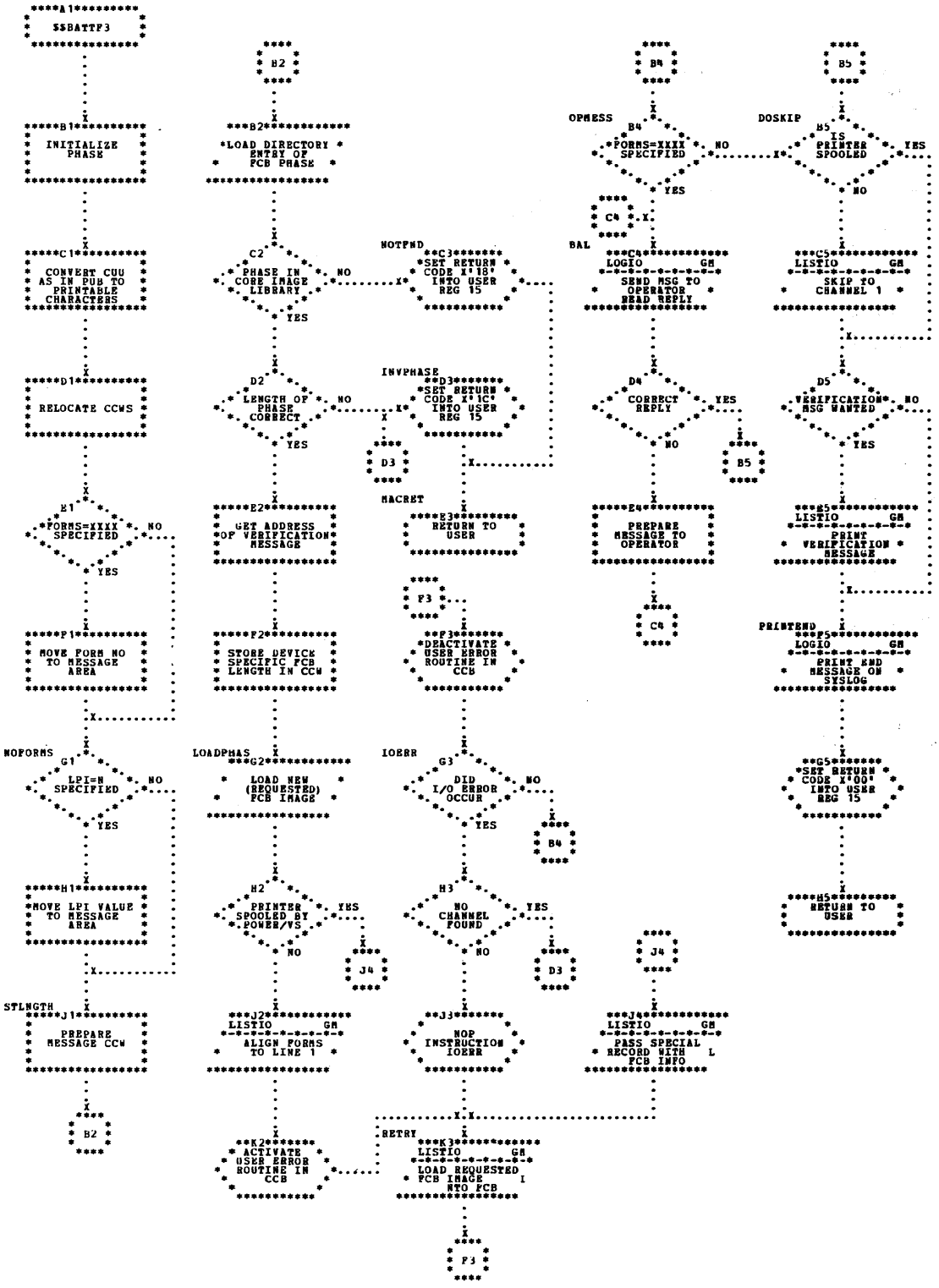


Chart KA. $$BOCRTA - CRT Root Phase (Part 1 of 3)
(Refer to Chart 16)

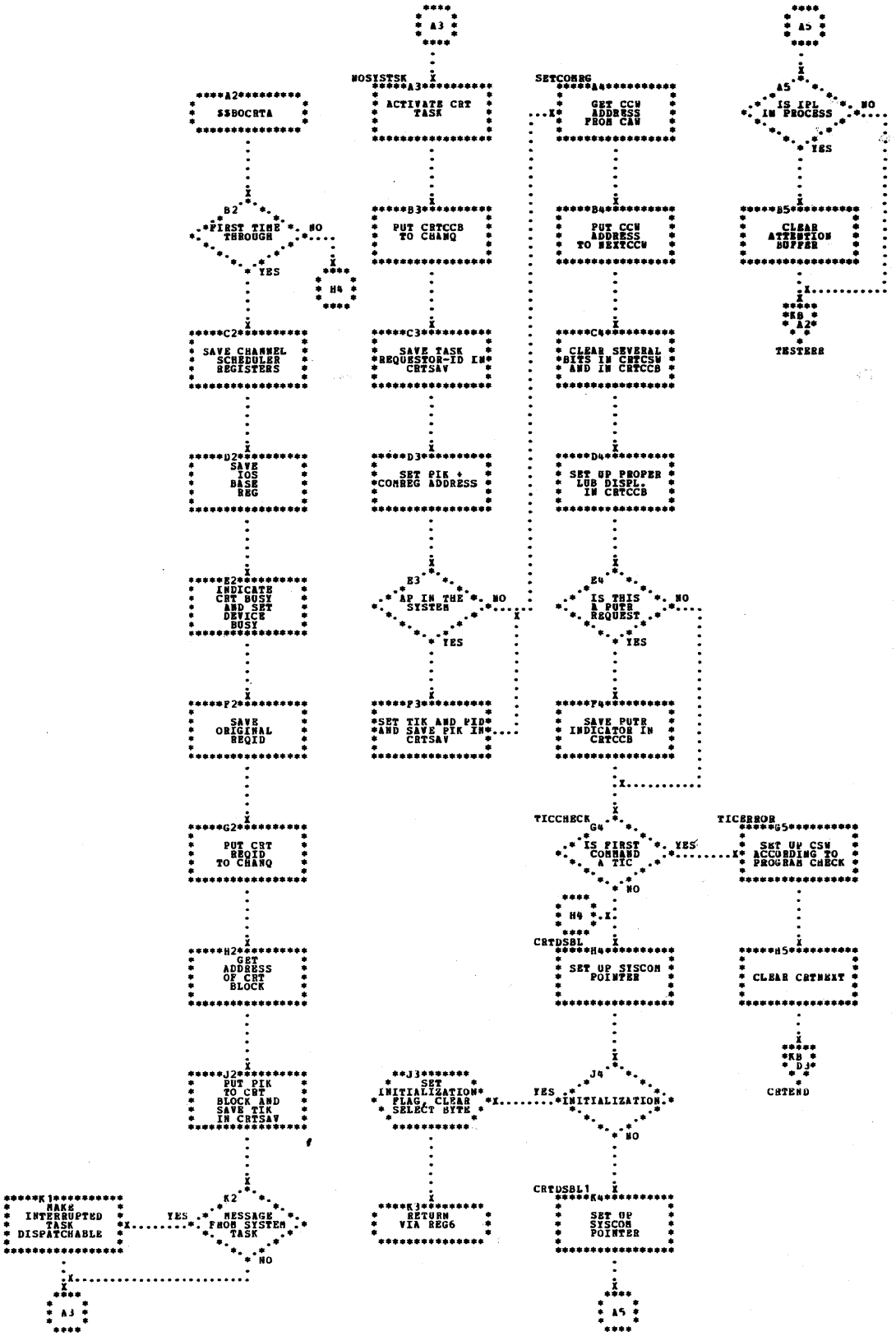


Chart KB. $$BOCRTA - CRT Root Phase (Part 2 of 3)
(Refer to Chart 16)

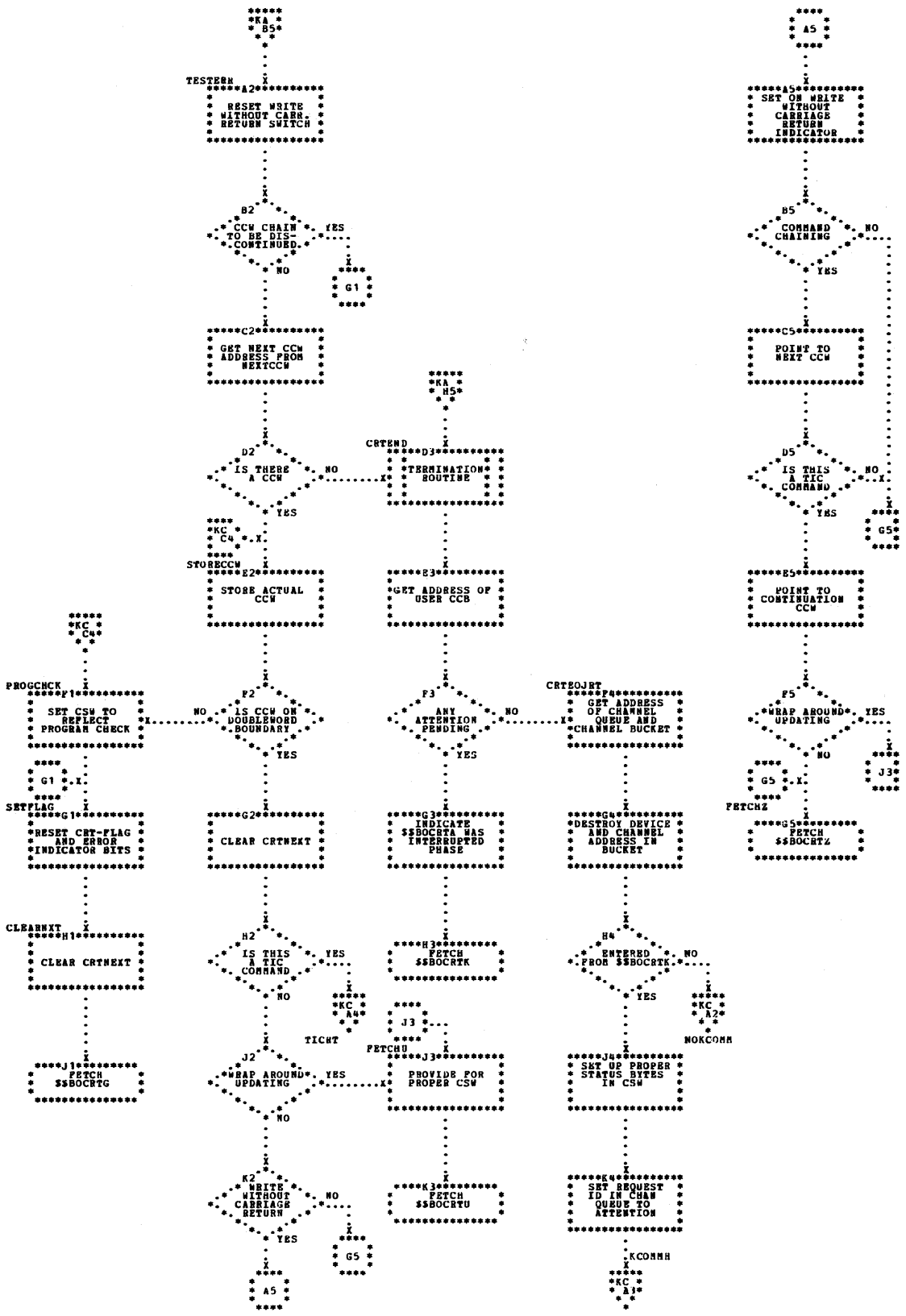


Chart KC. $$BOCRTA - CRT Root Phase (Part 3 of 3)
(Refer to Chart 16)

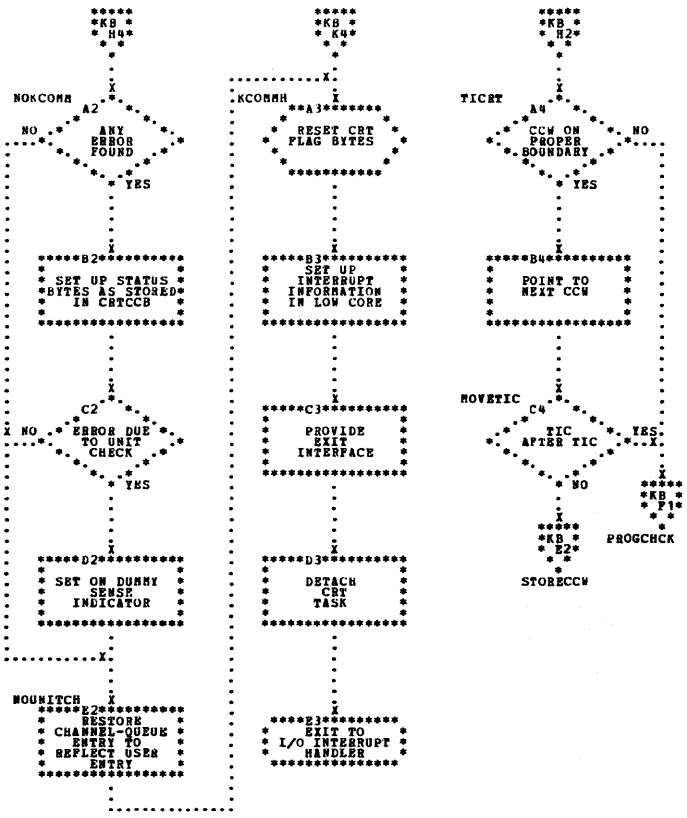


Chart KD. $$BOCRTB - Hard-Copy Disk ERP (Part 1 of 3)
(Refer to Chart 22)

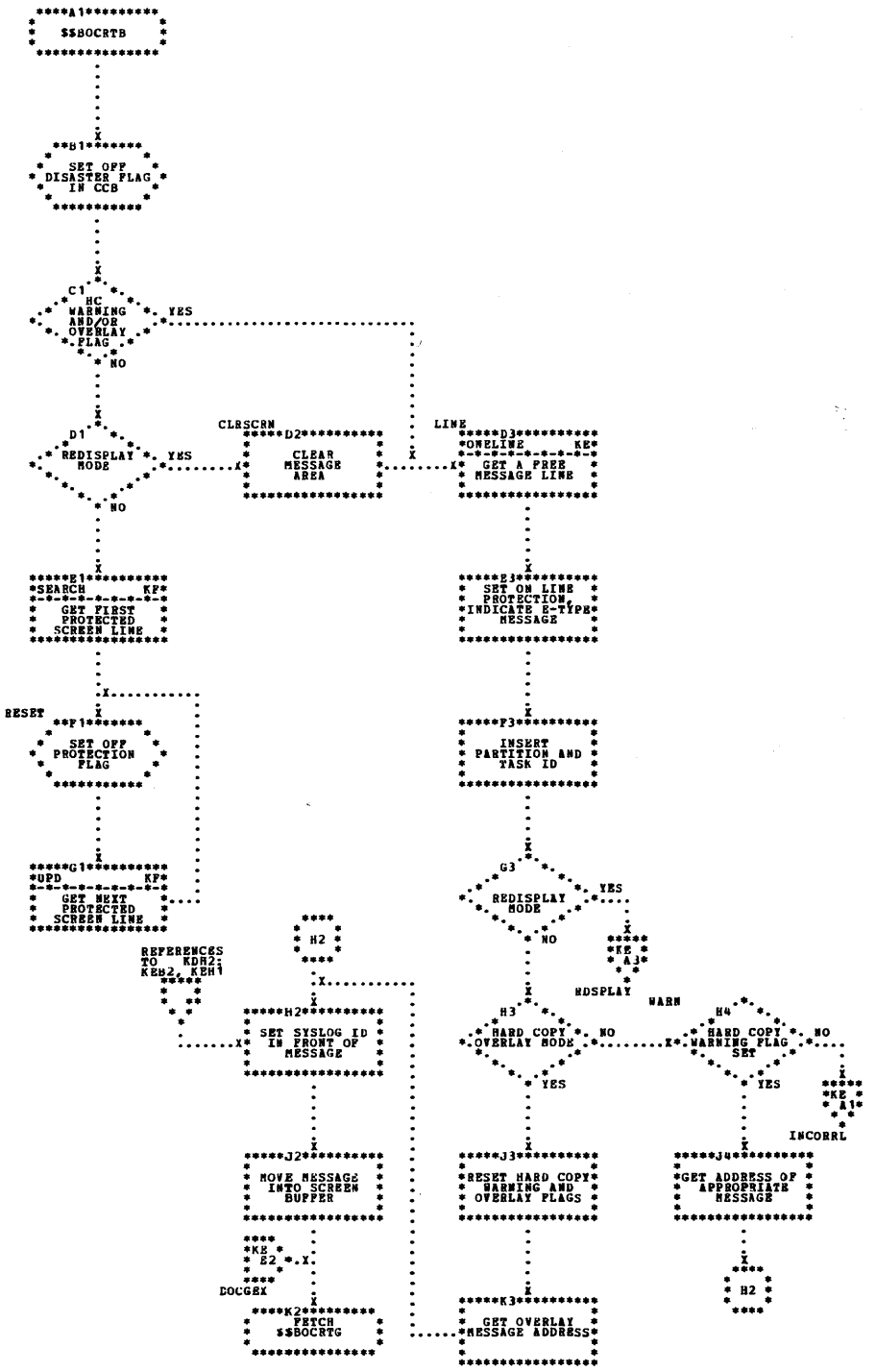


Chart KE. $$BOCRTB - Hard-Copy Disk ERP (Part 2 of 3)
(Refer to Chart 22)

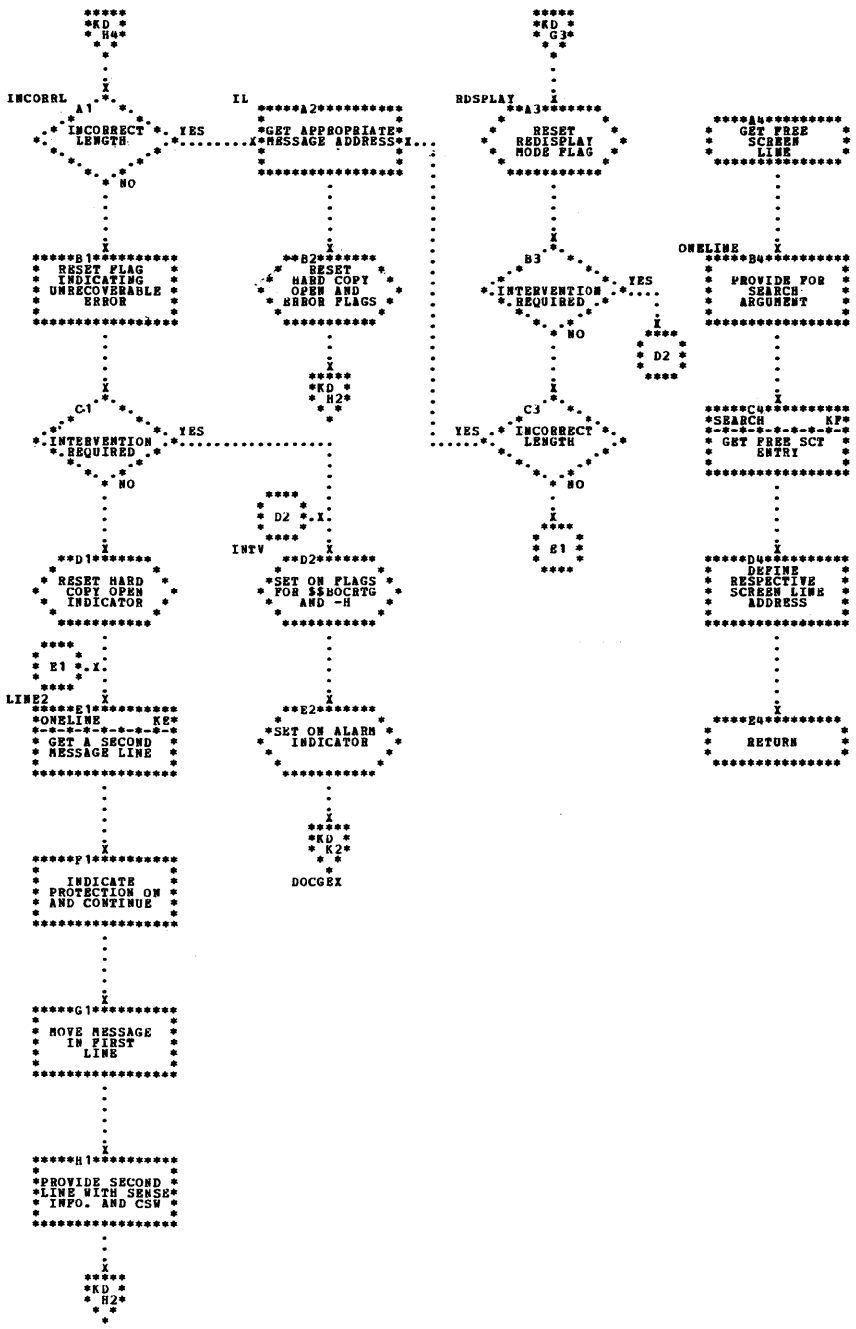


Chart KF. $$BOCRTB - Hard-Copy Disk ERP (Part 3 of 3)
(Refer to Chart 22)

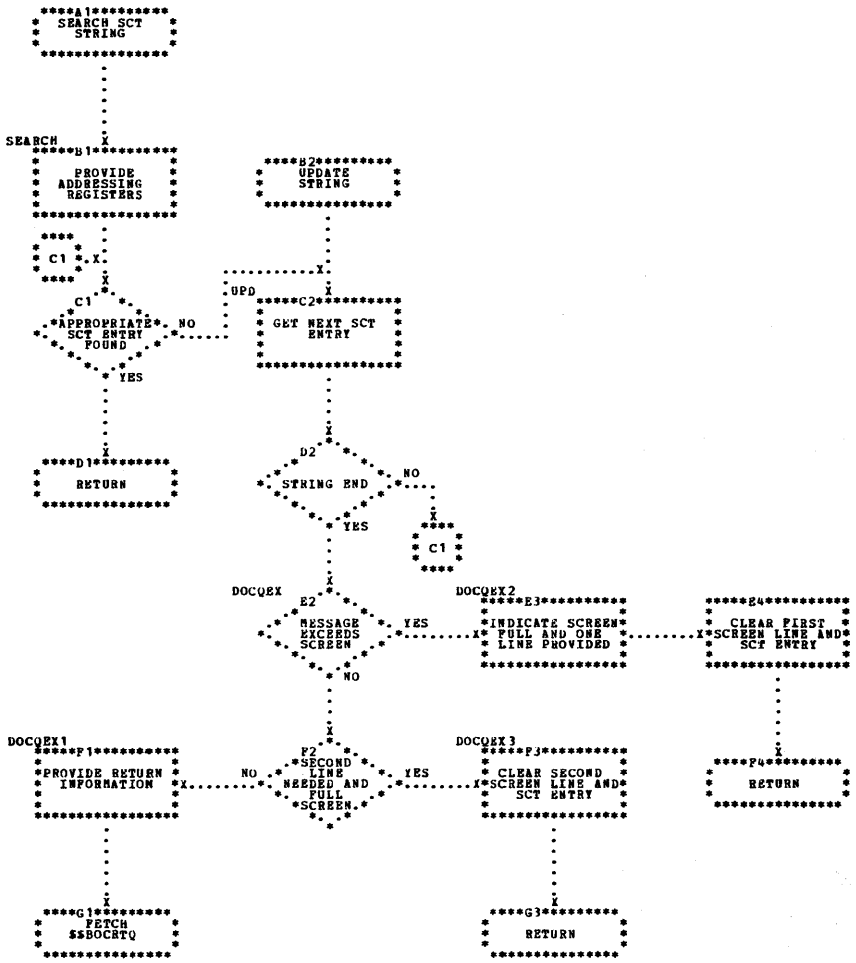


Chart KG. $$BOCRTC - Write Processor Part 1 (Part 1 of 4)
(Refer to Chart 17)

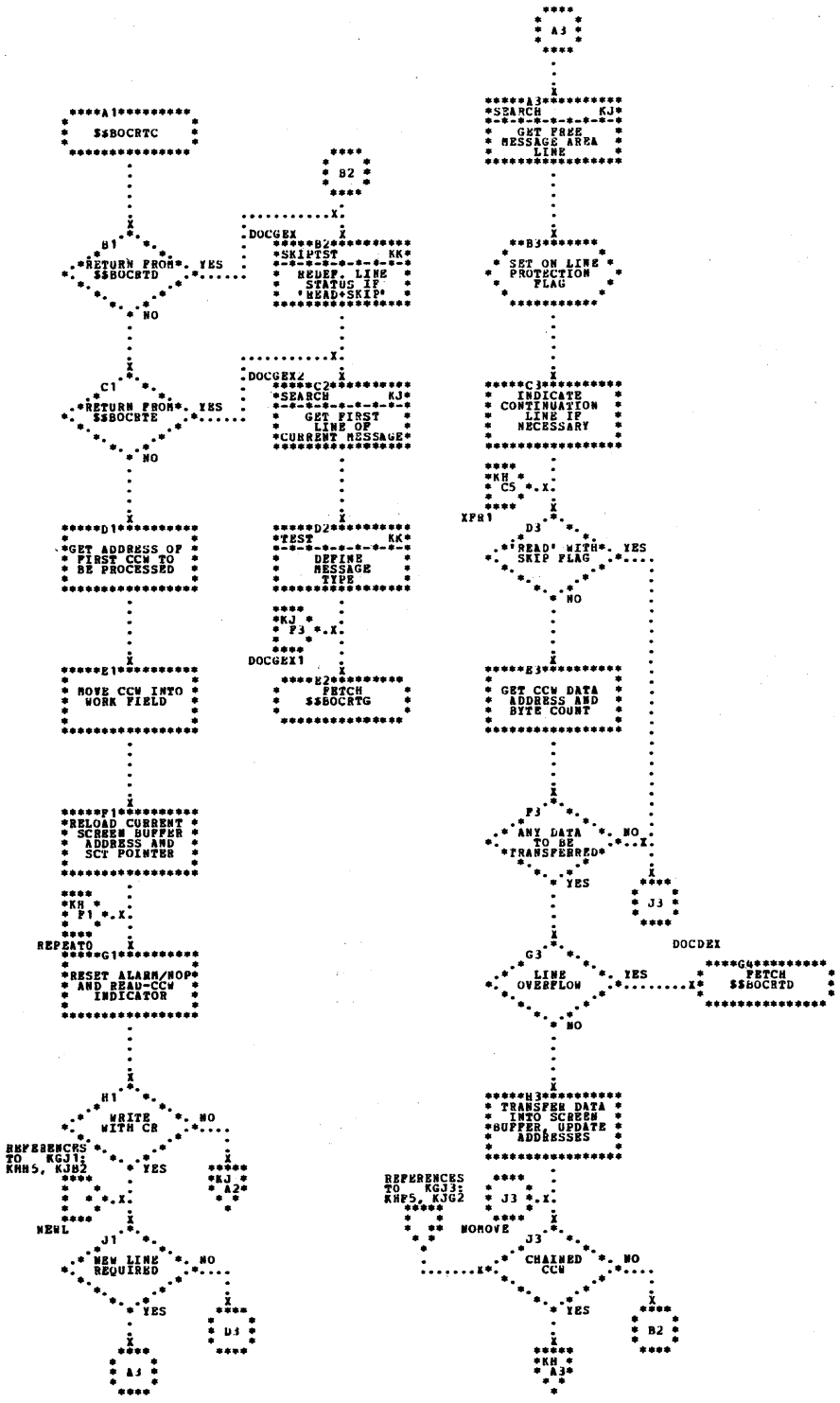


Chart KH. $$BOCRTC - Write Processor Part 1 (Part 2 of 4)
(Refer to Chart 17)

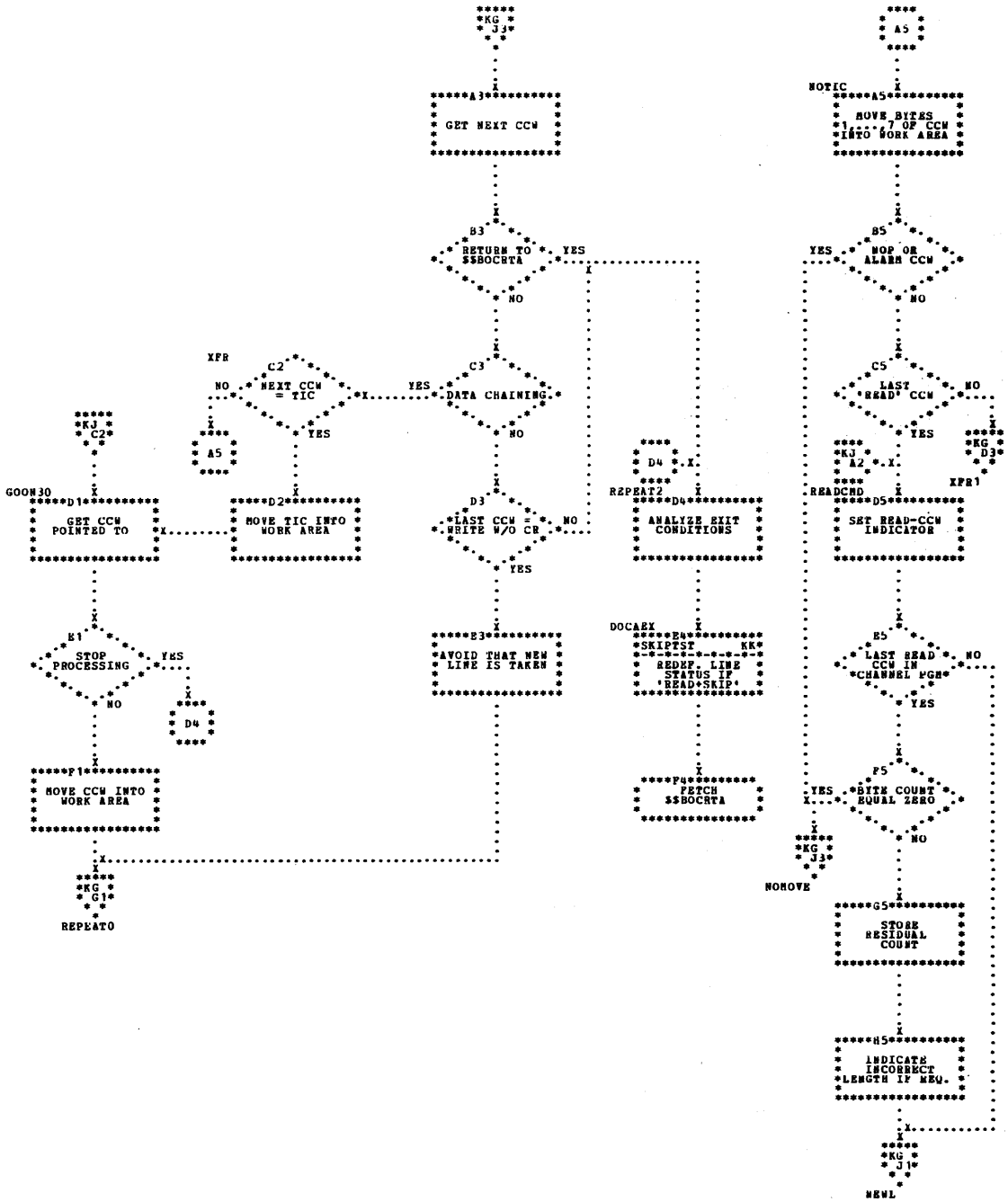


Chart KJ. $$BOCRTC - Write Processor Part 1 (Part 3 of 4)
(Refer to Chart 17)

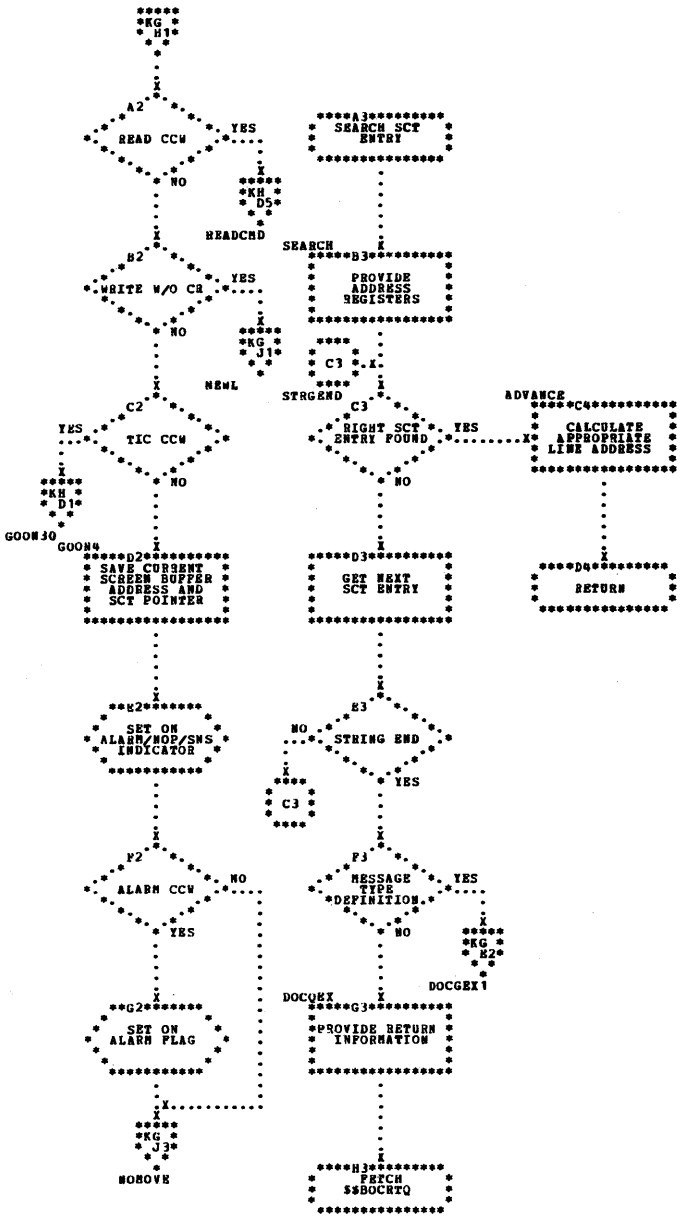


Chart KK. $$BOCRTC - Write Processor Part 1 (Part 4 of 4)
(Refer to Chart 17)

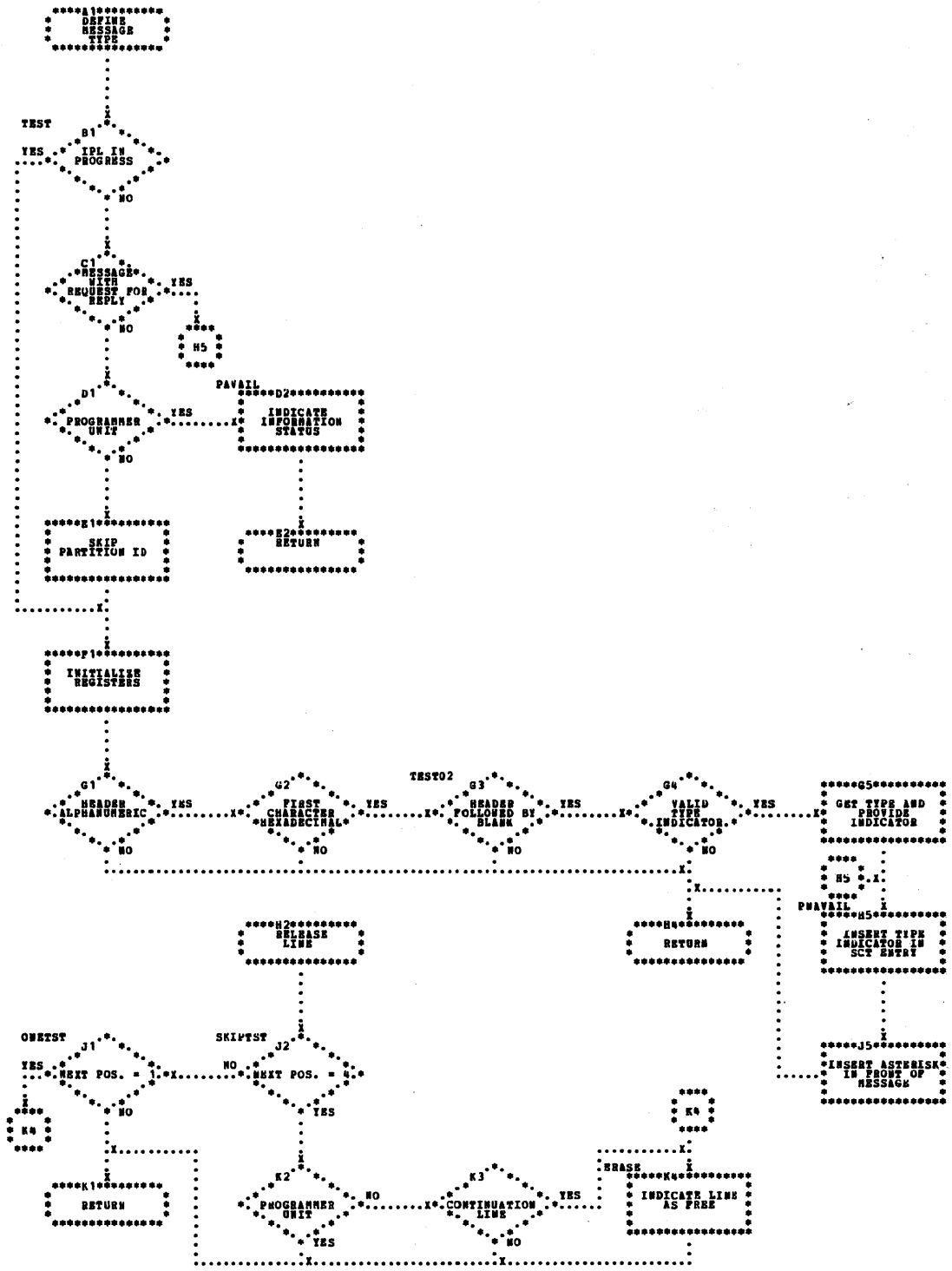


Chart LA. $$BOCRTD - Write Processor Part 2 (Part 1 of 3)
(Refer to Chart 17)

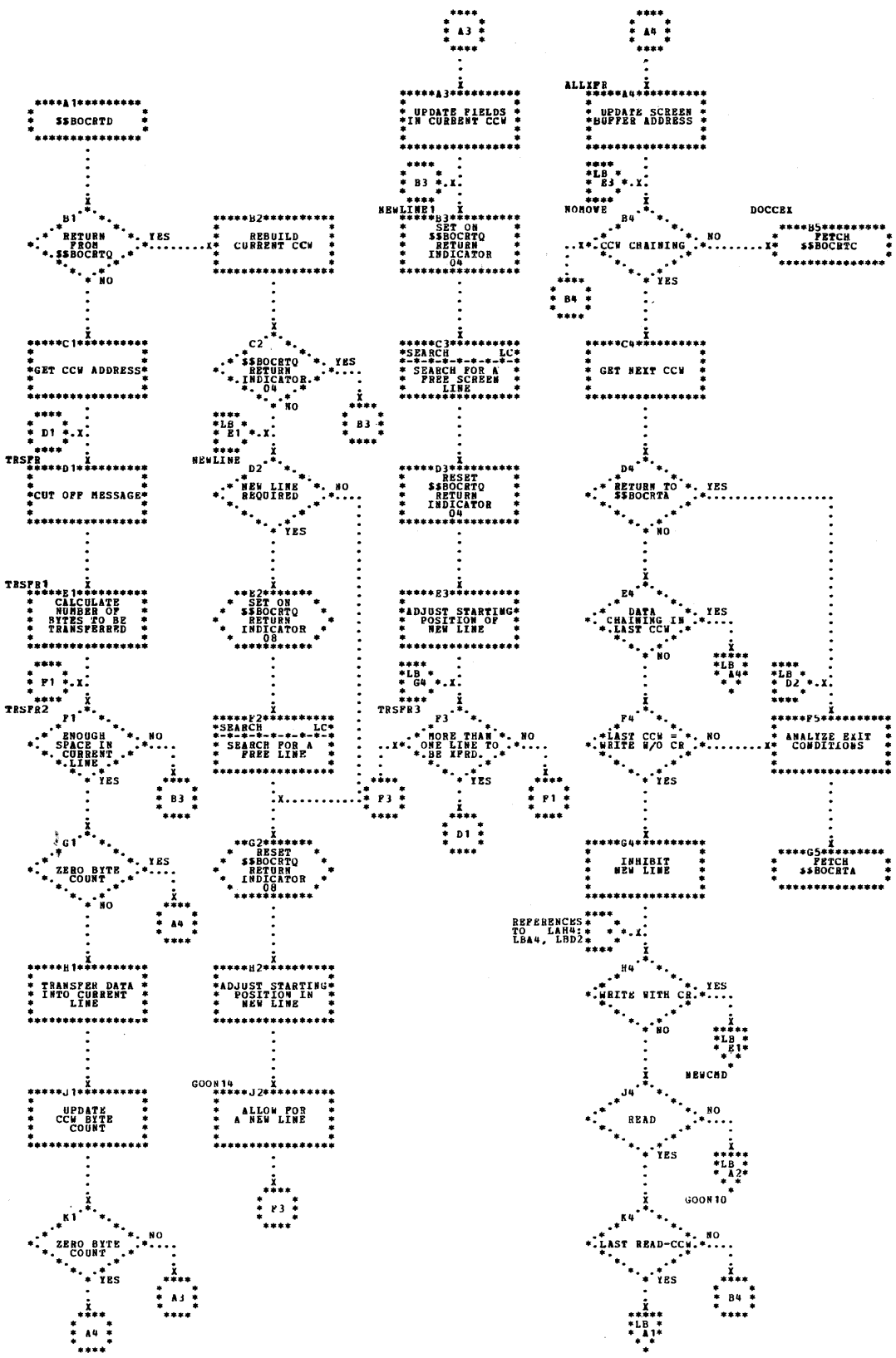


Chart LB. $$BOCRTD - Write Processor Part 2 (Part 2 of 3)
(Refer to Chart 17)

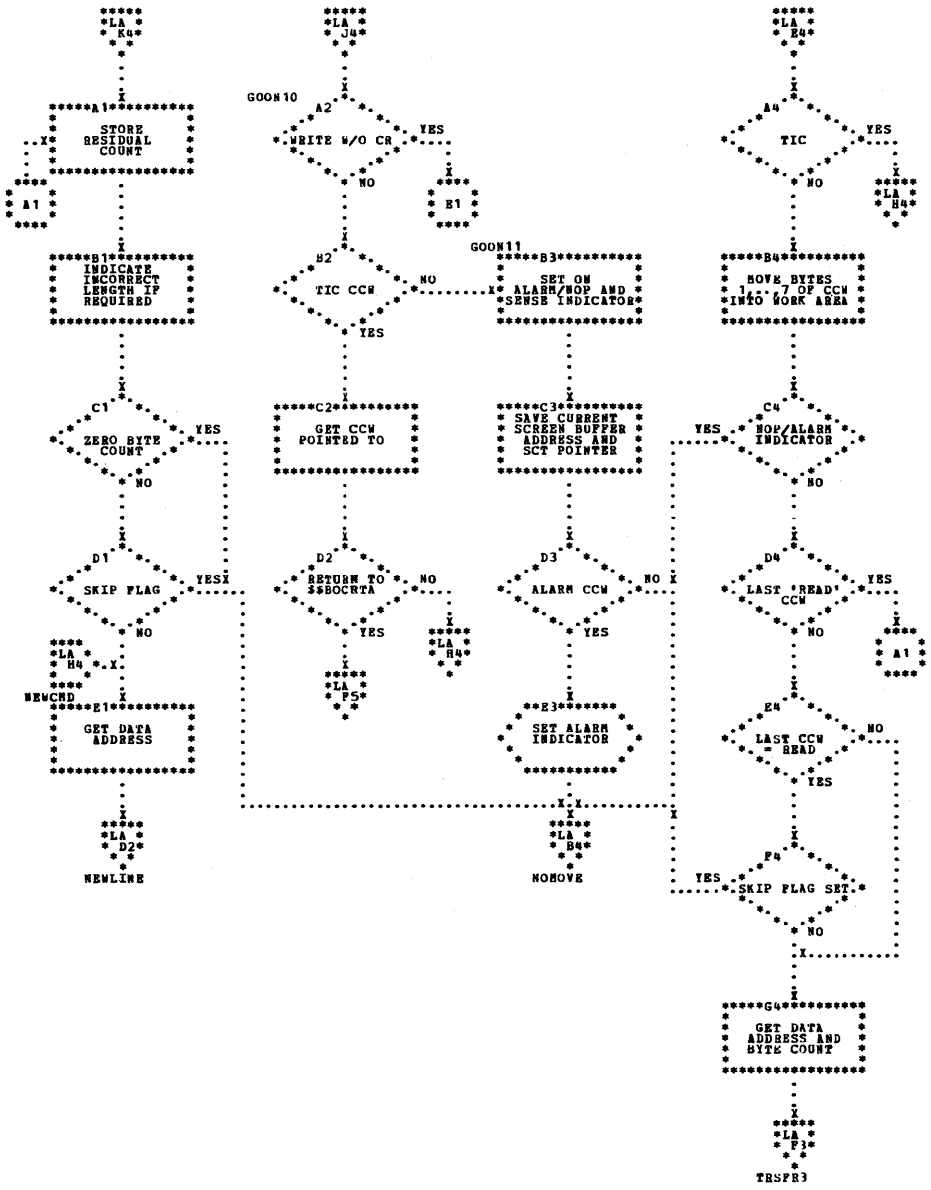


Chart LC. $$BOCRTD - Write Processor Part 2 (Part 3 of 3)
(Refer to Chart 17)

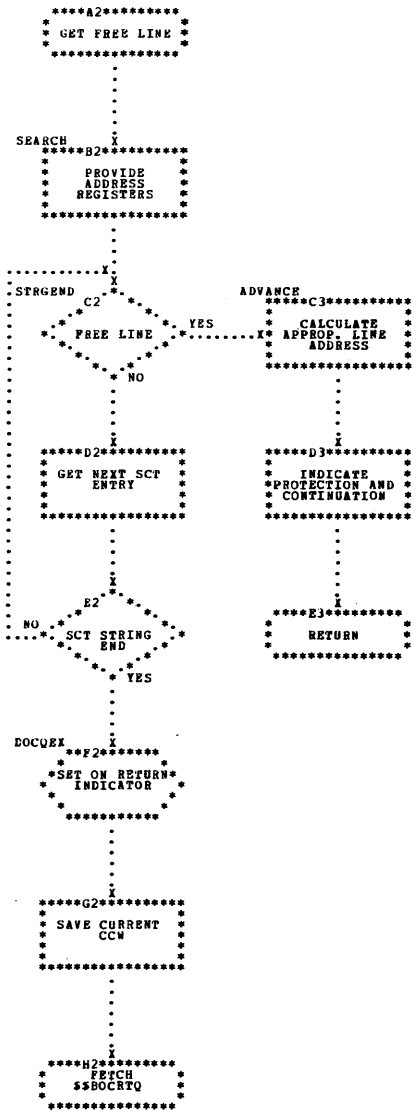


Chart LD. $$BOCRTE - CRT ERP Part 1 (Part 1 of 3)
(Refer to Chart 22)

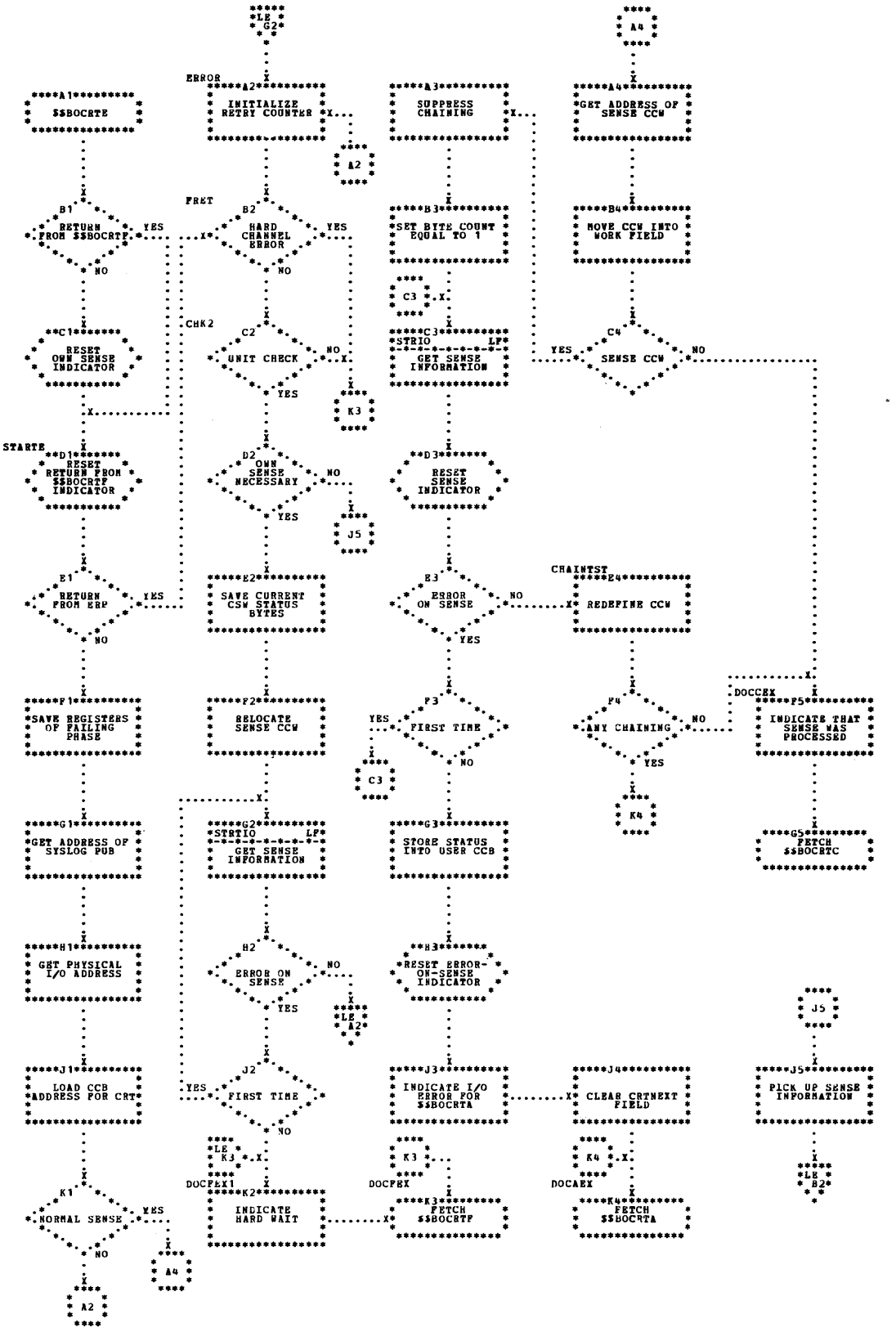


Chart LE. $$BOCRTE - CRT ERP Part 1 (Part 2 of 3)
(Refer to Chart 22)

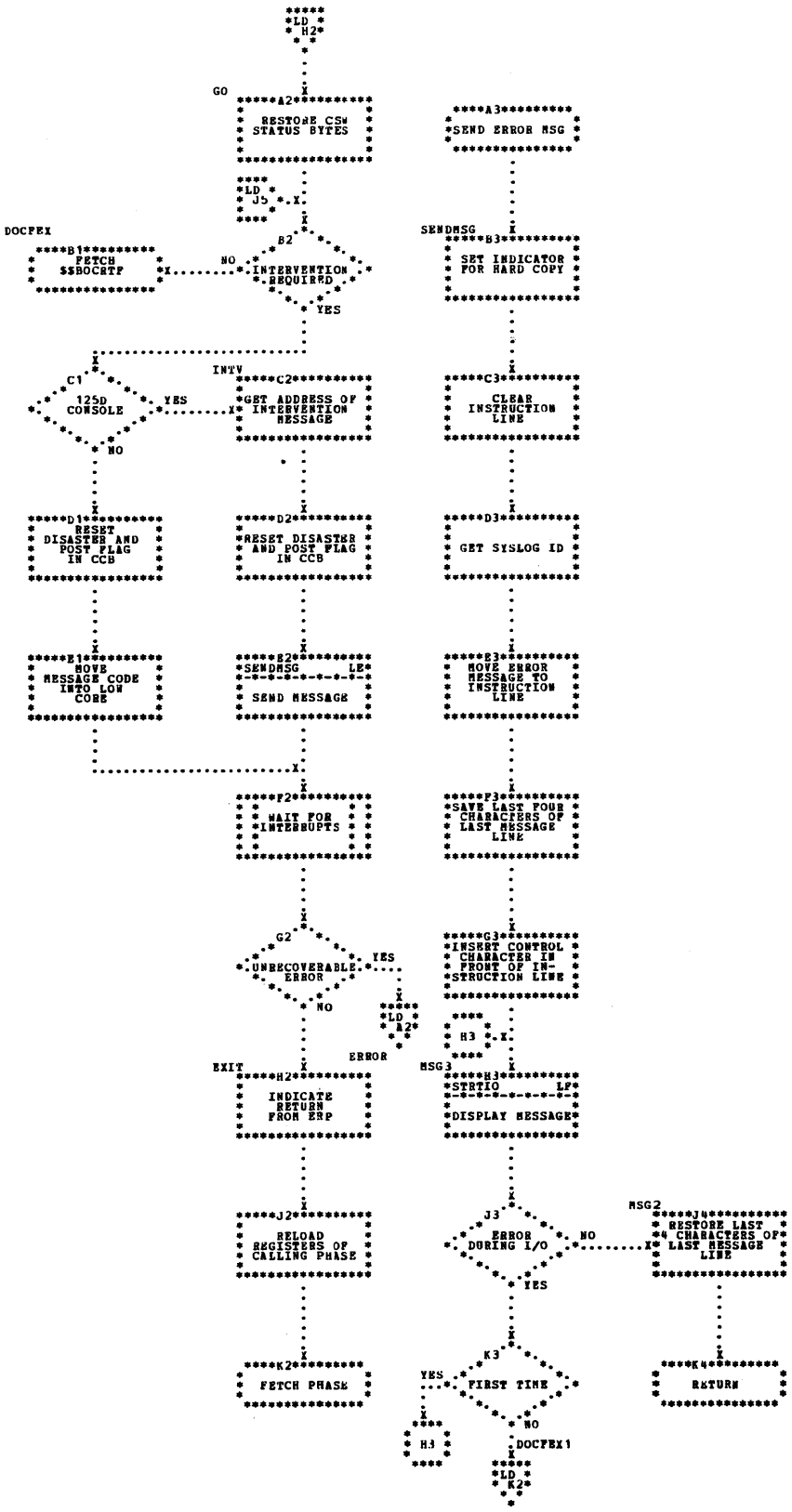


Chart LF. $$BOCRTE - CRT ERP Part 1 (Part 3 of 3)
(Refer to Chart 22)

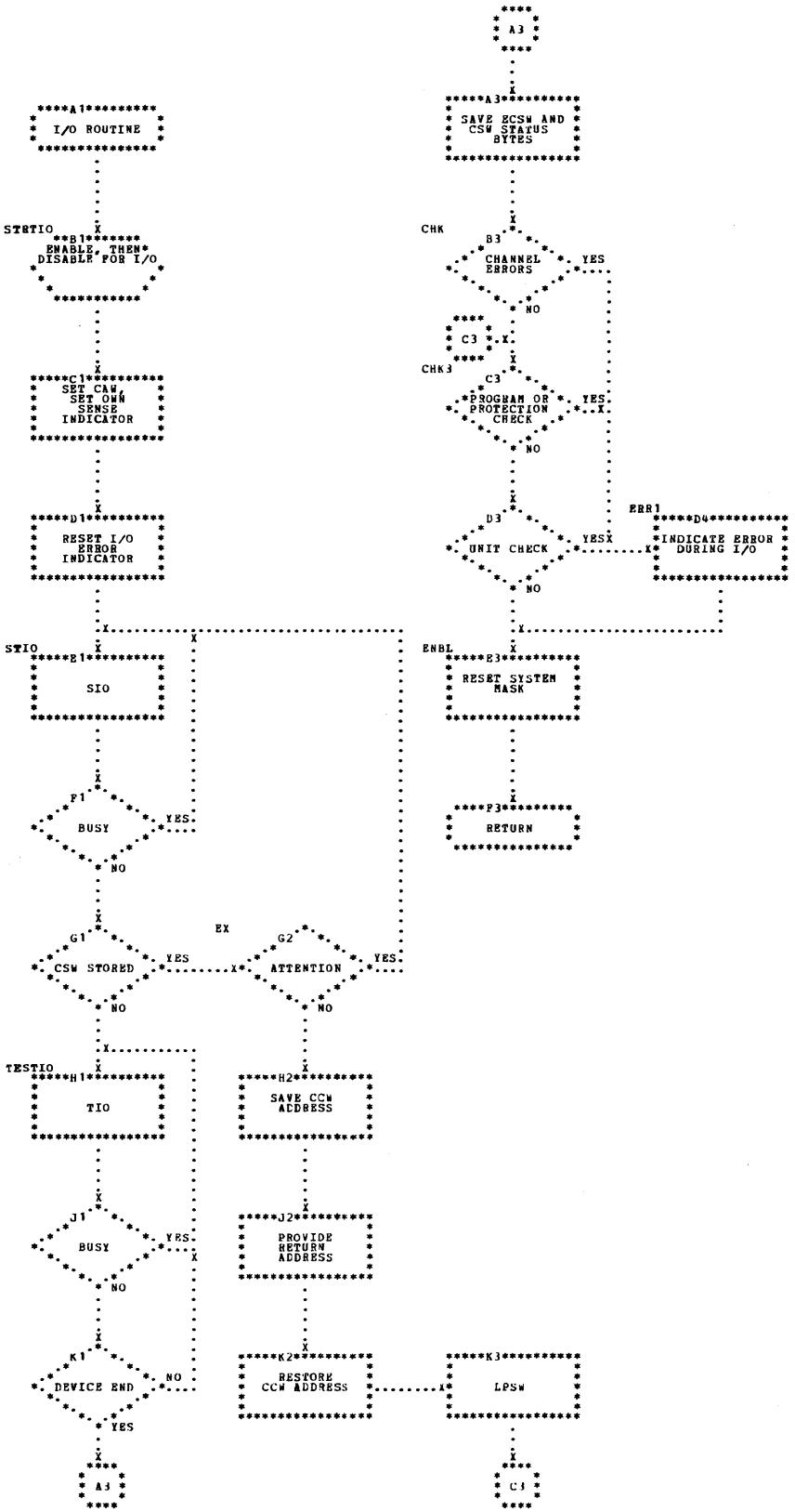


Chart LG. $$BOCRTF - CRT ERP Part 2 (Part 1 of 4)
(Refer to Chart 22)

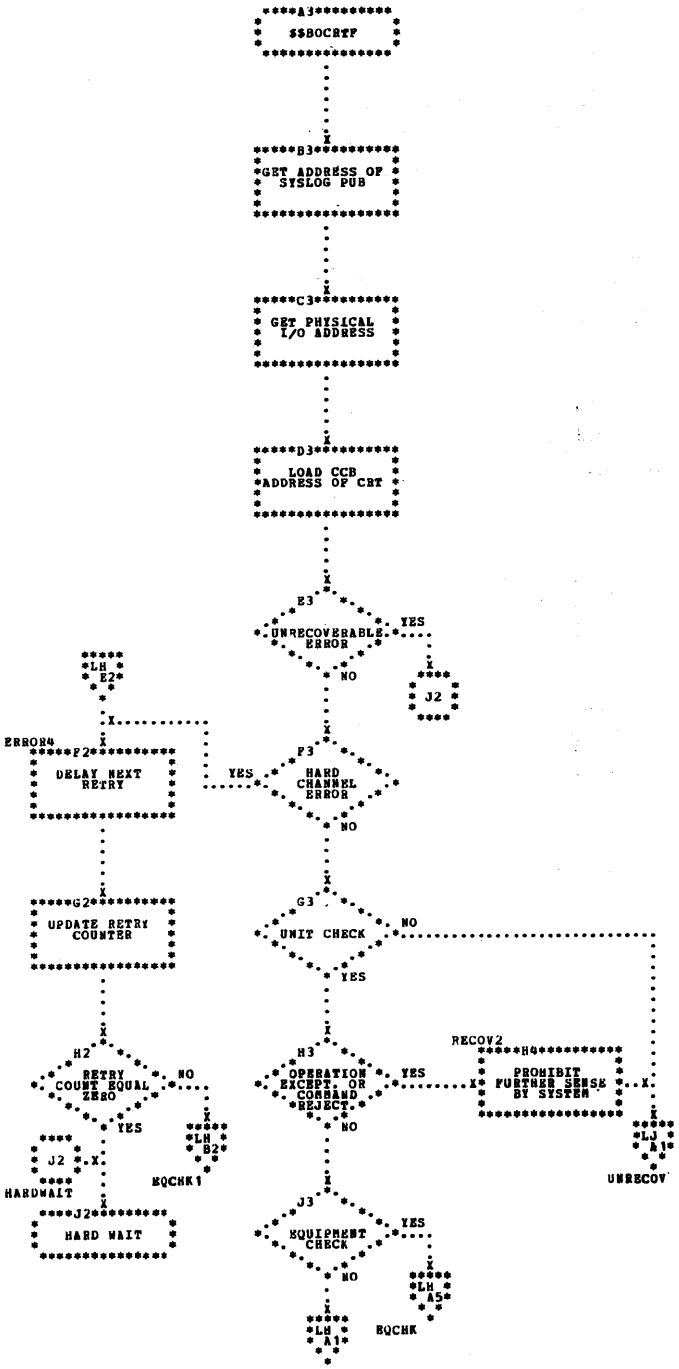


Chart LH. $$BOCRTF - CRT ERP Part 2 (Part 2 of 4)
(Refer to Chart 22)

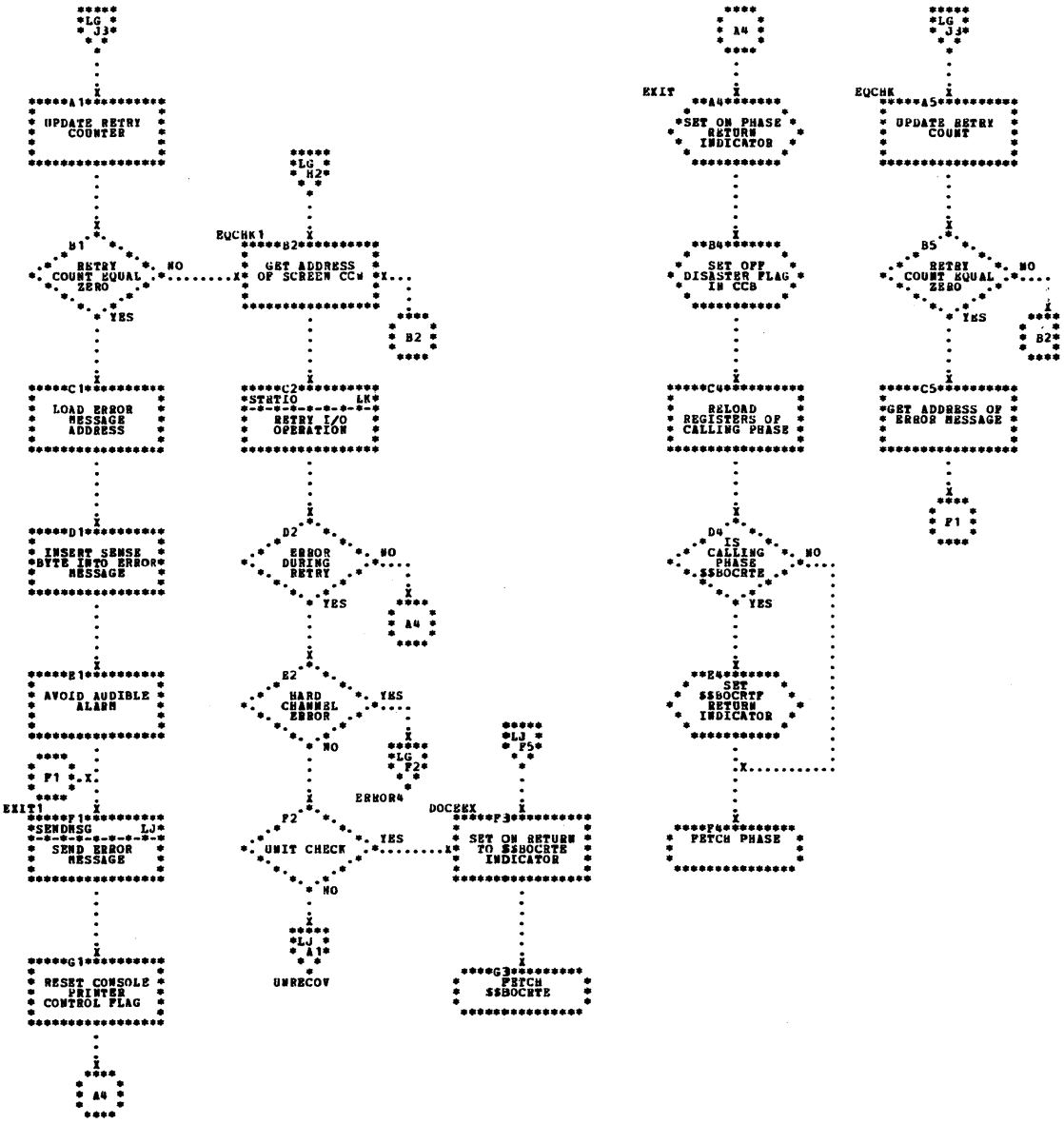


Chart LJ. $$BOCRTF - CRT ERP Part 2 (Part 3 of 4)
(Refer to Chart 22)

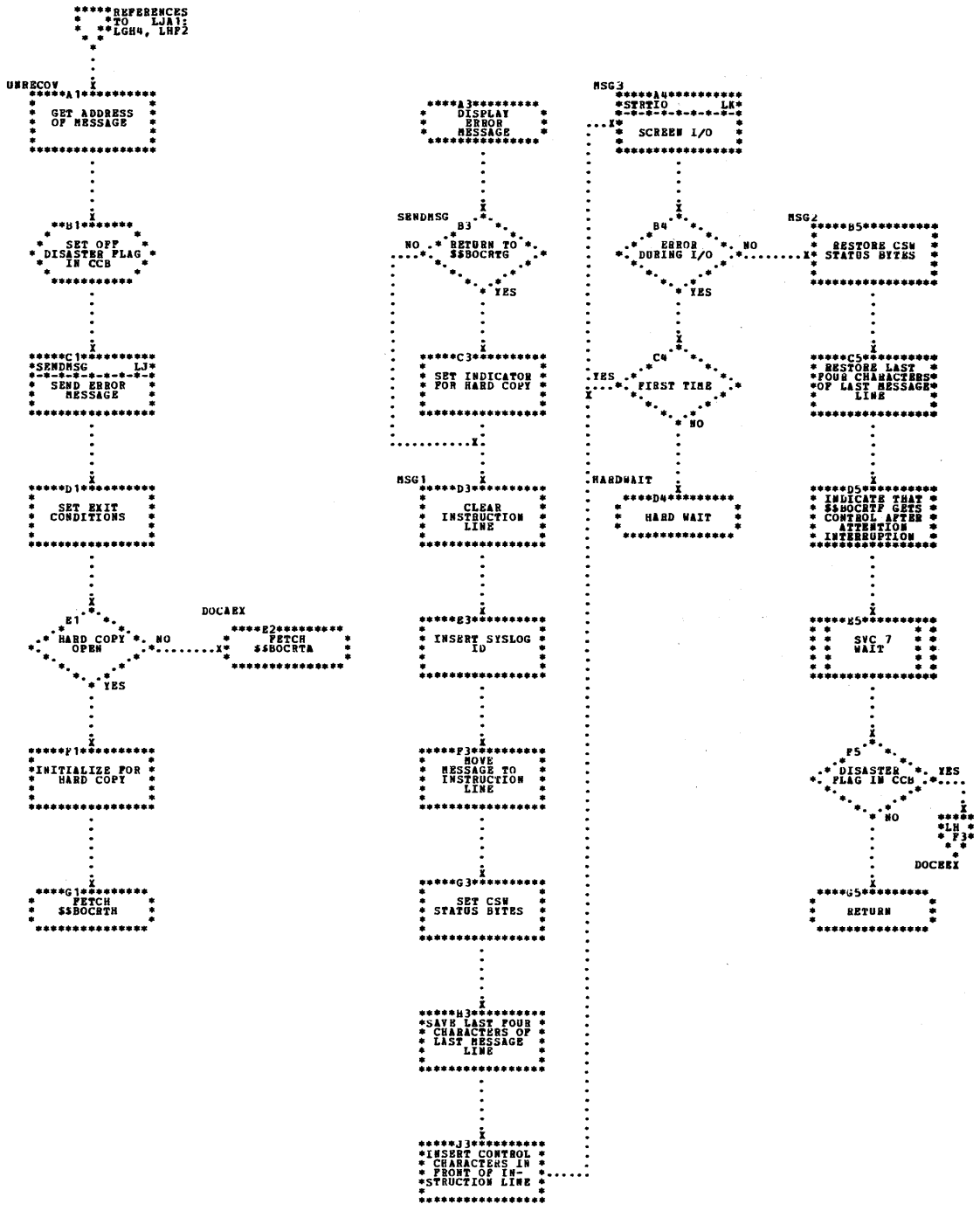


Chart LK. $$BOCRTF - CRT ERP Part 2 (Part 4 of 4)
(Refer to Chart 22)

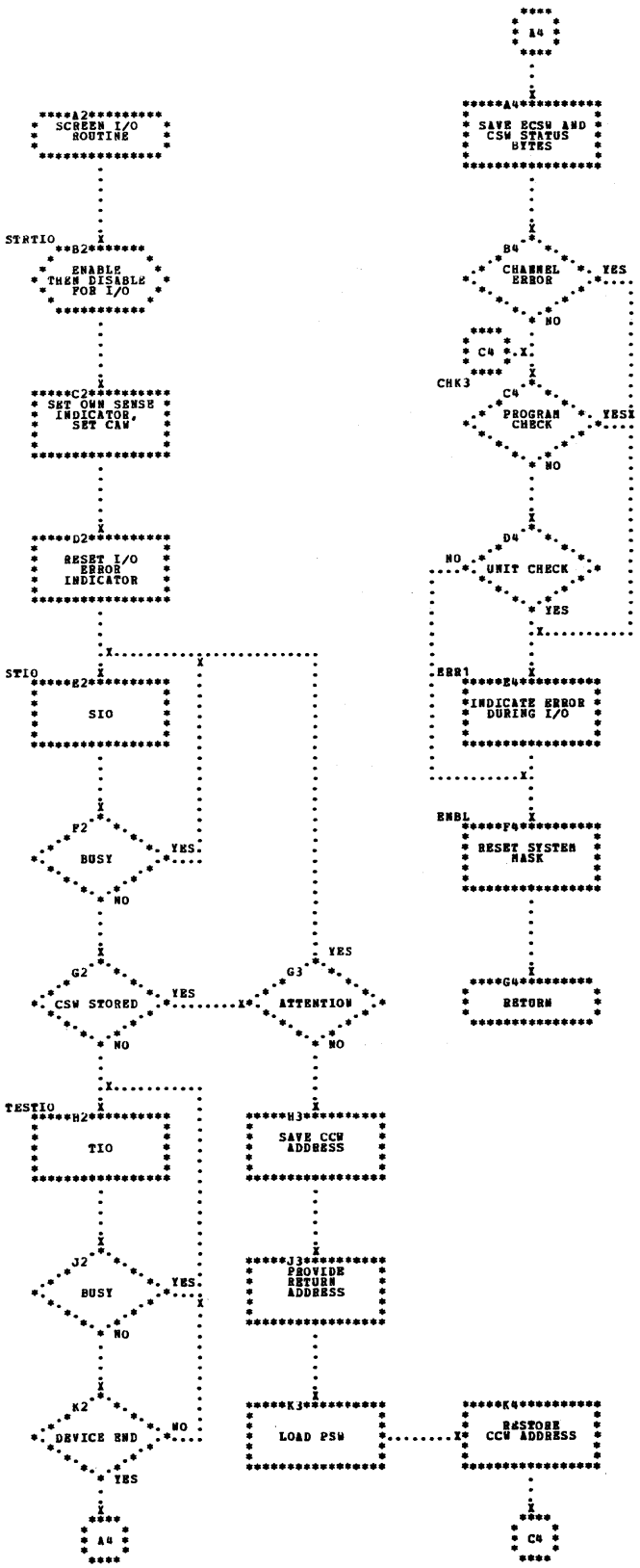


Chart MA. $$BOCRTG - Write Screen Message (Part 1 of 3)
(Refer to Chart 17)

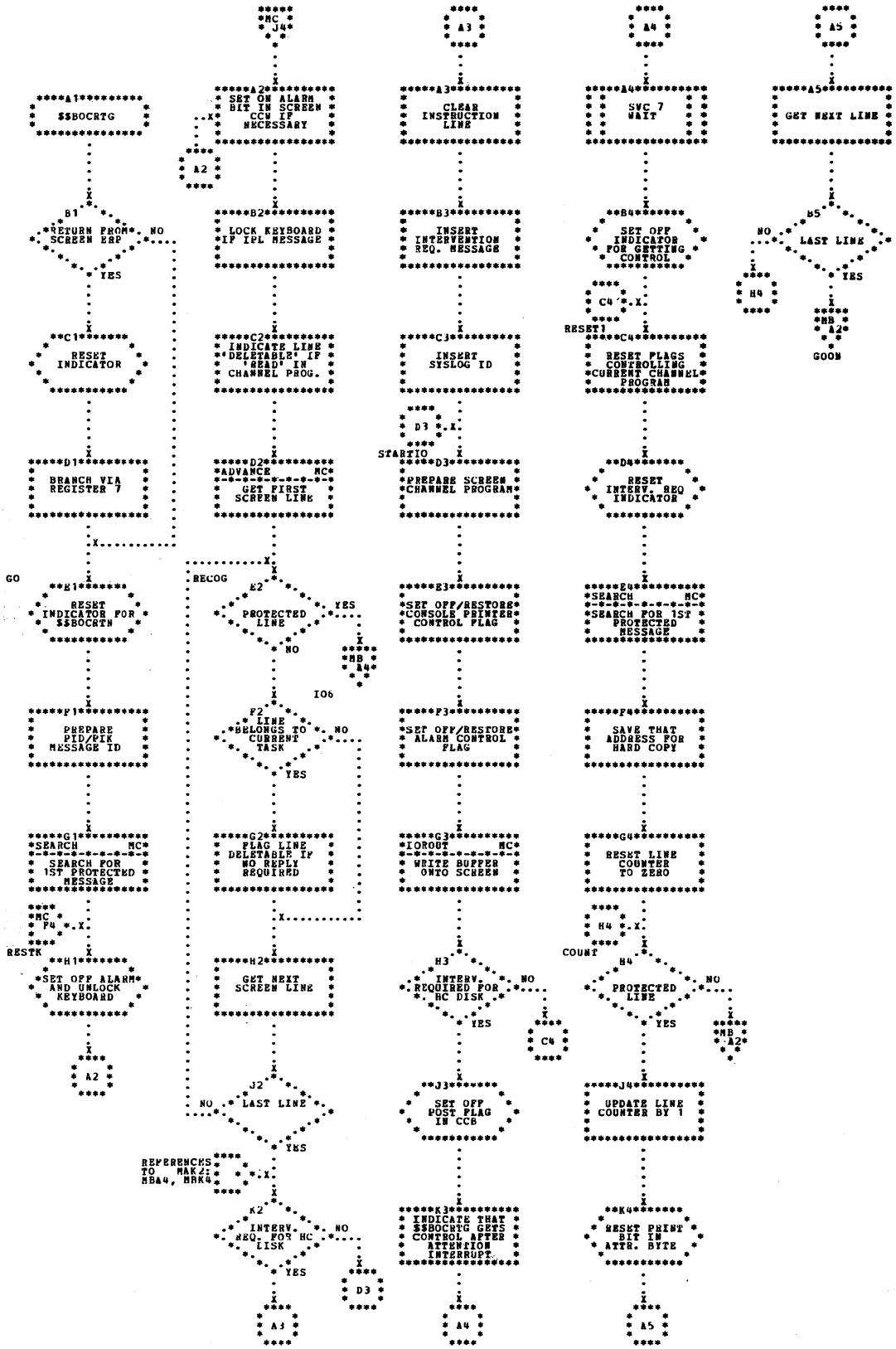


Chart MB. $$BOCRTG - Write Screen Message (Part 2 of 3)
(Refer to Chart 17)

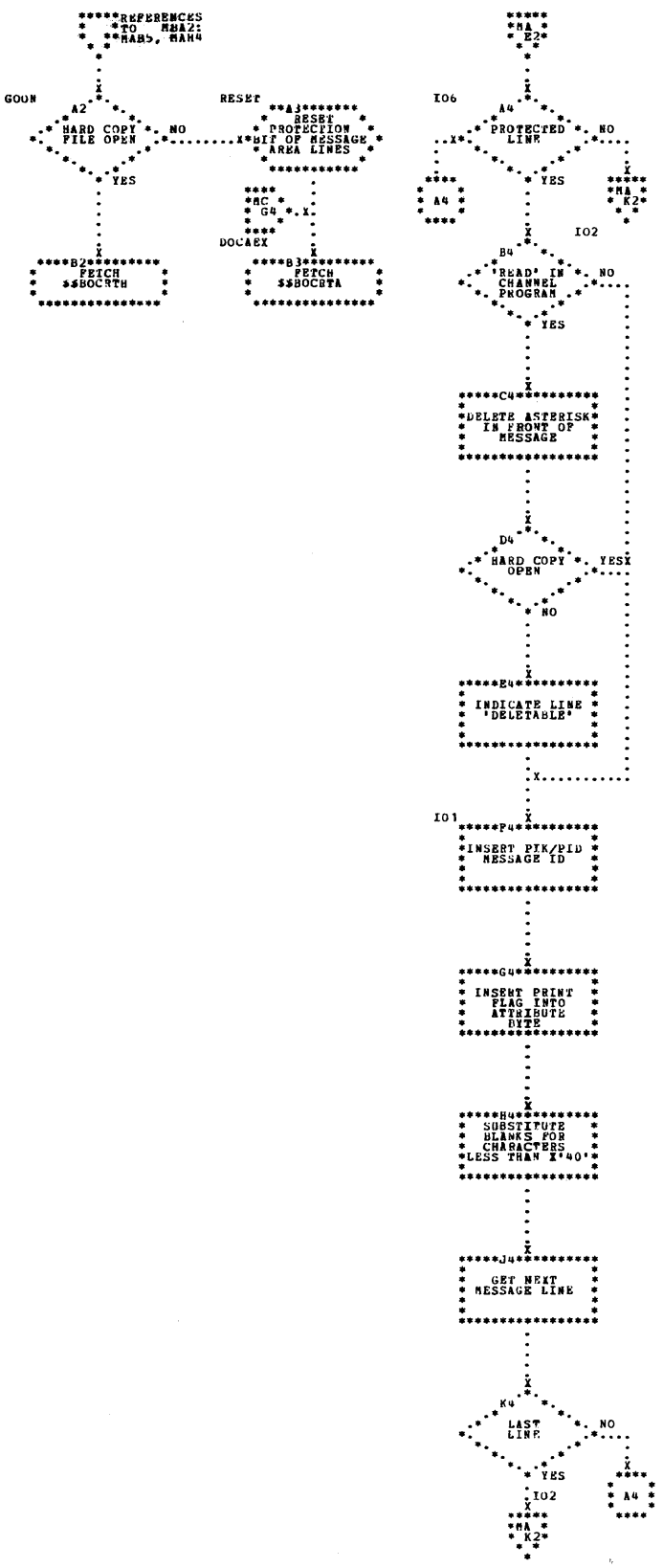


Chart MC. $$BOCRTG - Write Screen Message (Part 3 of 3)
(Refer to Chart 17)

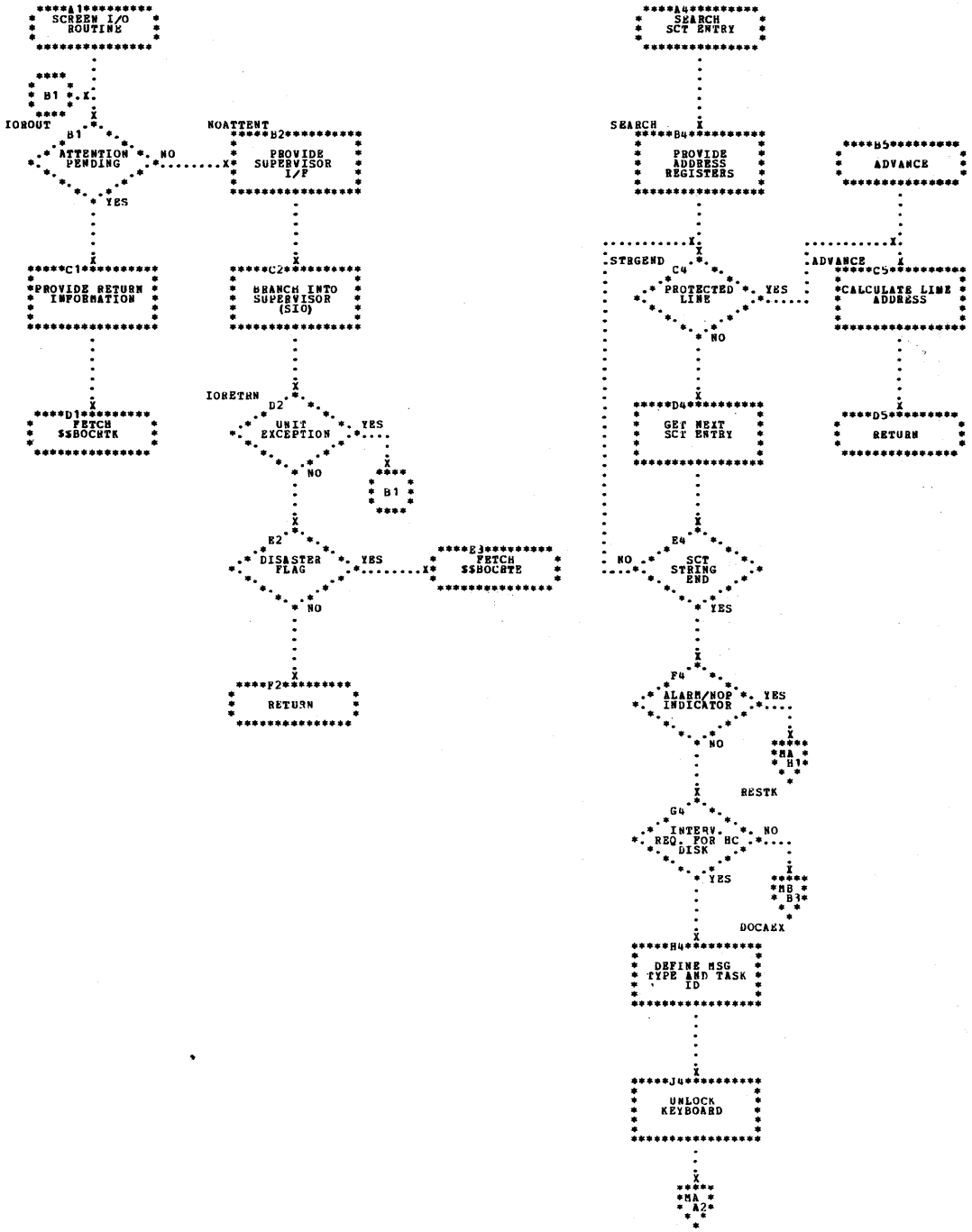
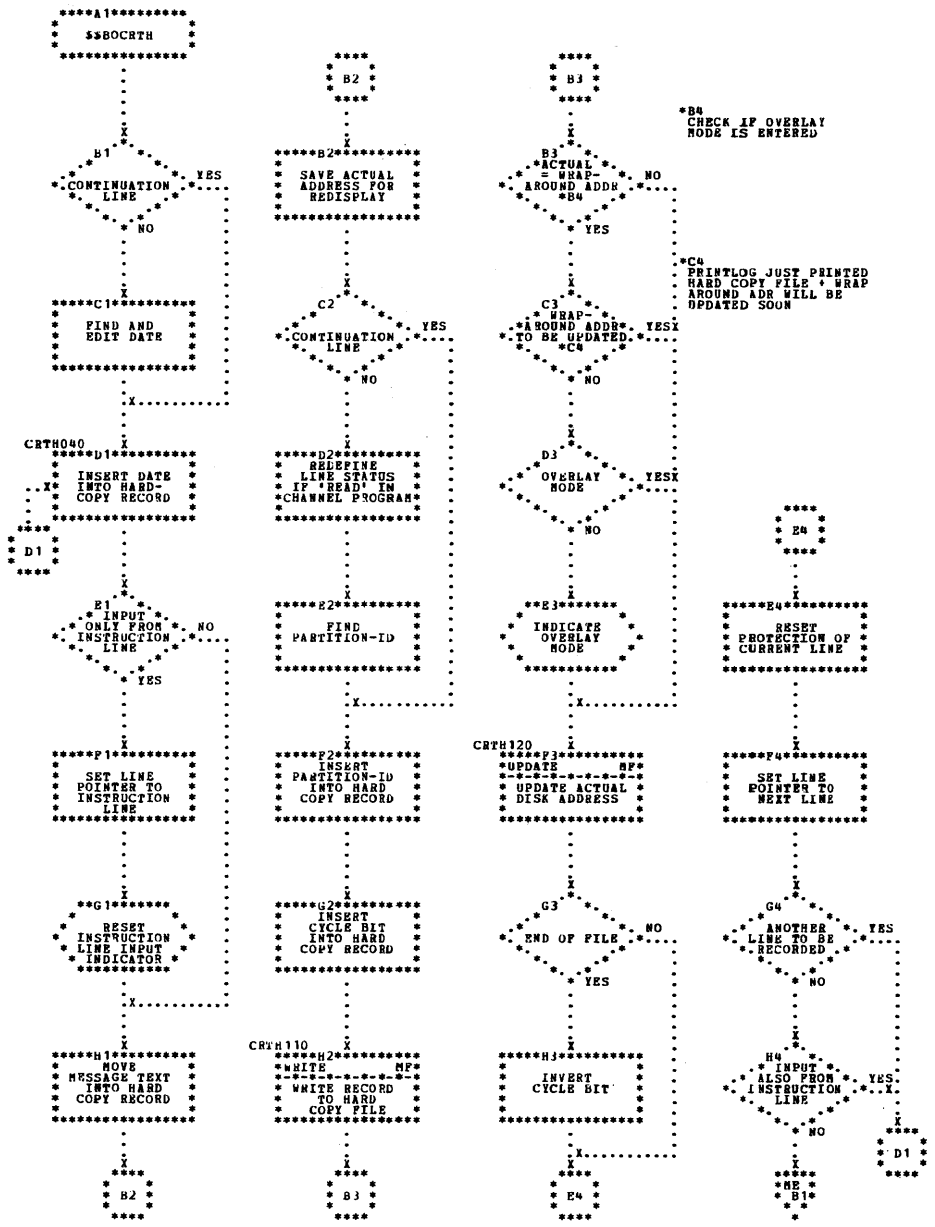


Chart MD. $$BOCRTH - Write Hard-Copy Records (Part 1 of 3)
(Refer to Chart 17)



Definitions:

actual address	= disk address of the next free record position in hard-copy file
wrap-around address	= disk address up to which hard-copy file was already printed by system utility PRINTLOG
overlay mode	= parts of the hard-copy file which were not printed by system utility PRINTLOG are overwritten

Chart ME. $$BOCRTH - Write Hard-Copy Records (Part 2 of 3)
(Refer to Chart 17)

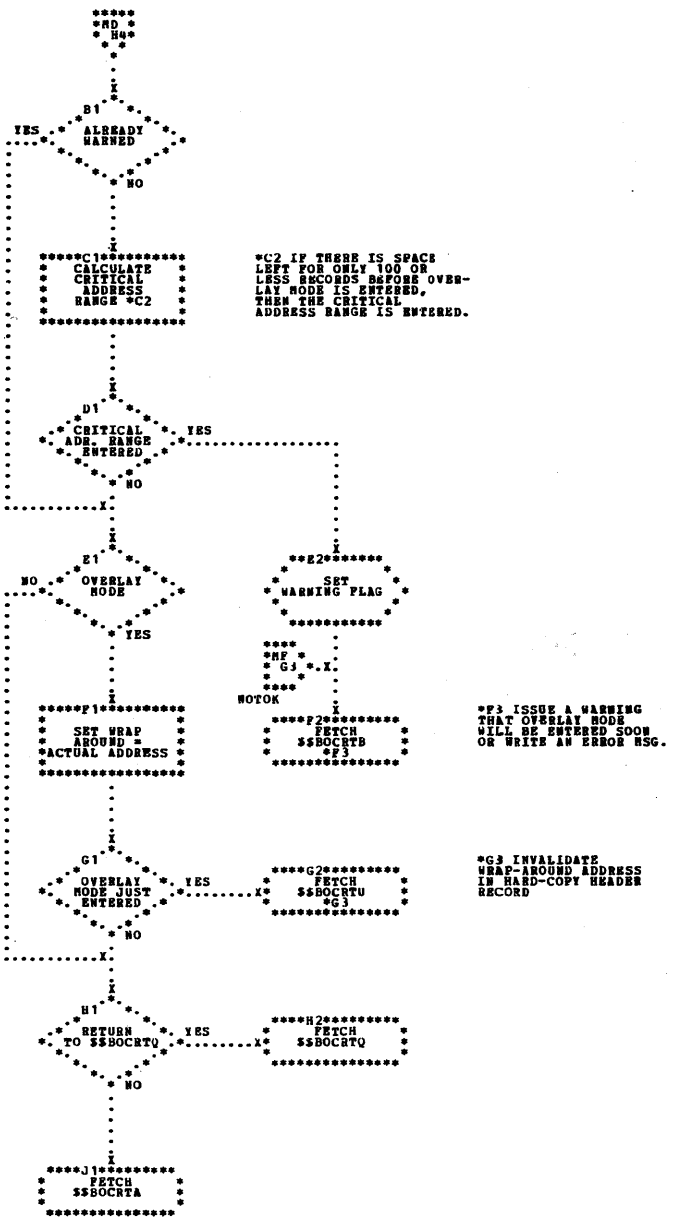


Chart MF. $$BOCRTH - Write Hard-Copy Records (Part 3 of 3)
(Refer to Chart 17)

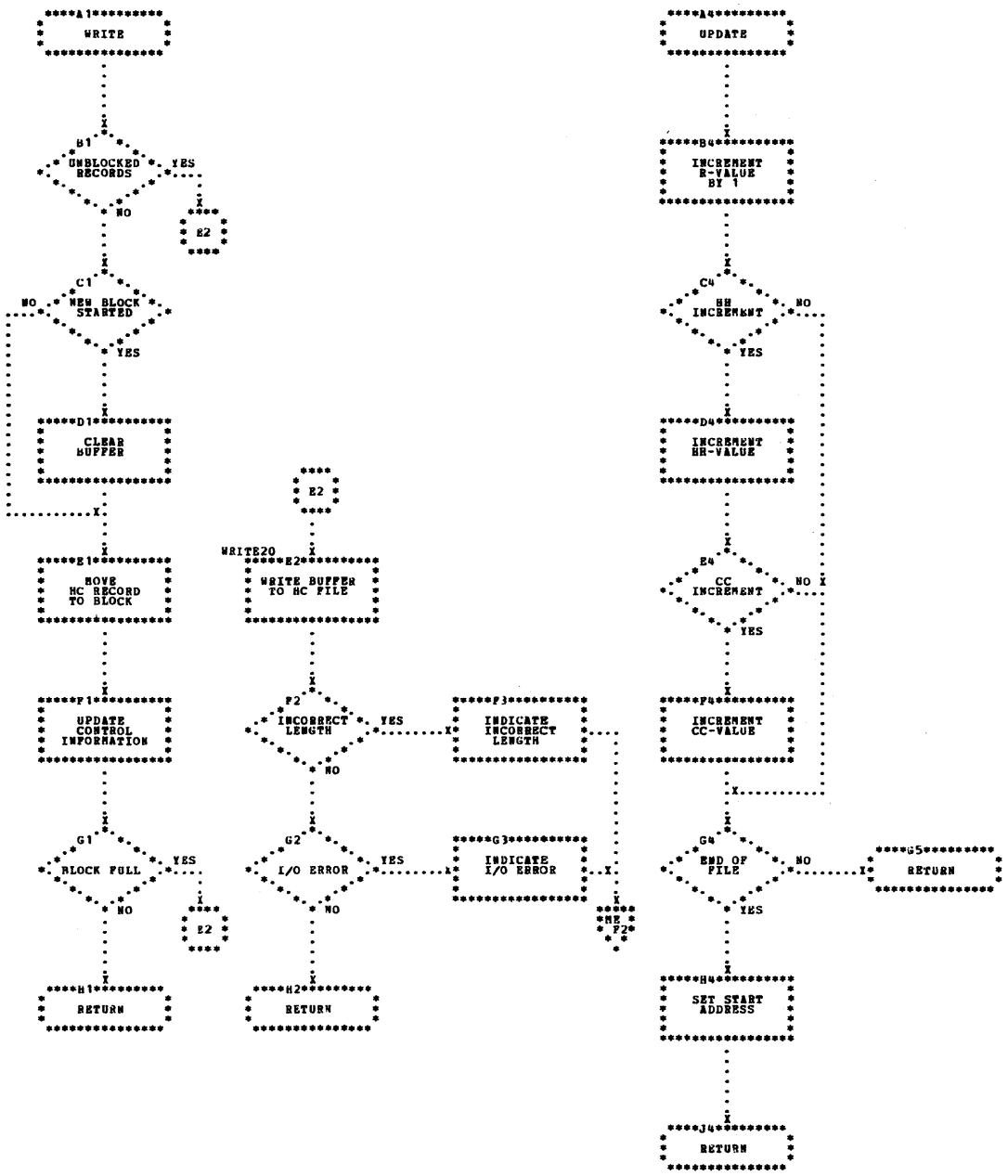


Chart MG. $$BOCRTI - Write on 3284/3286/3287/3288 Console Printer
(Refer to Chart 17)

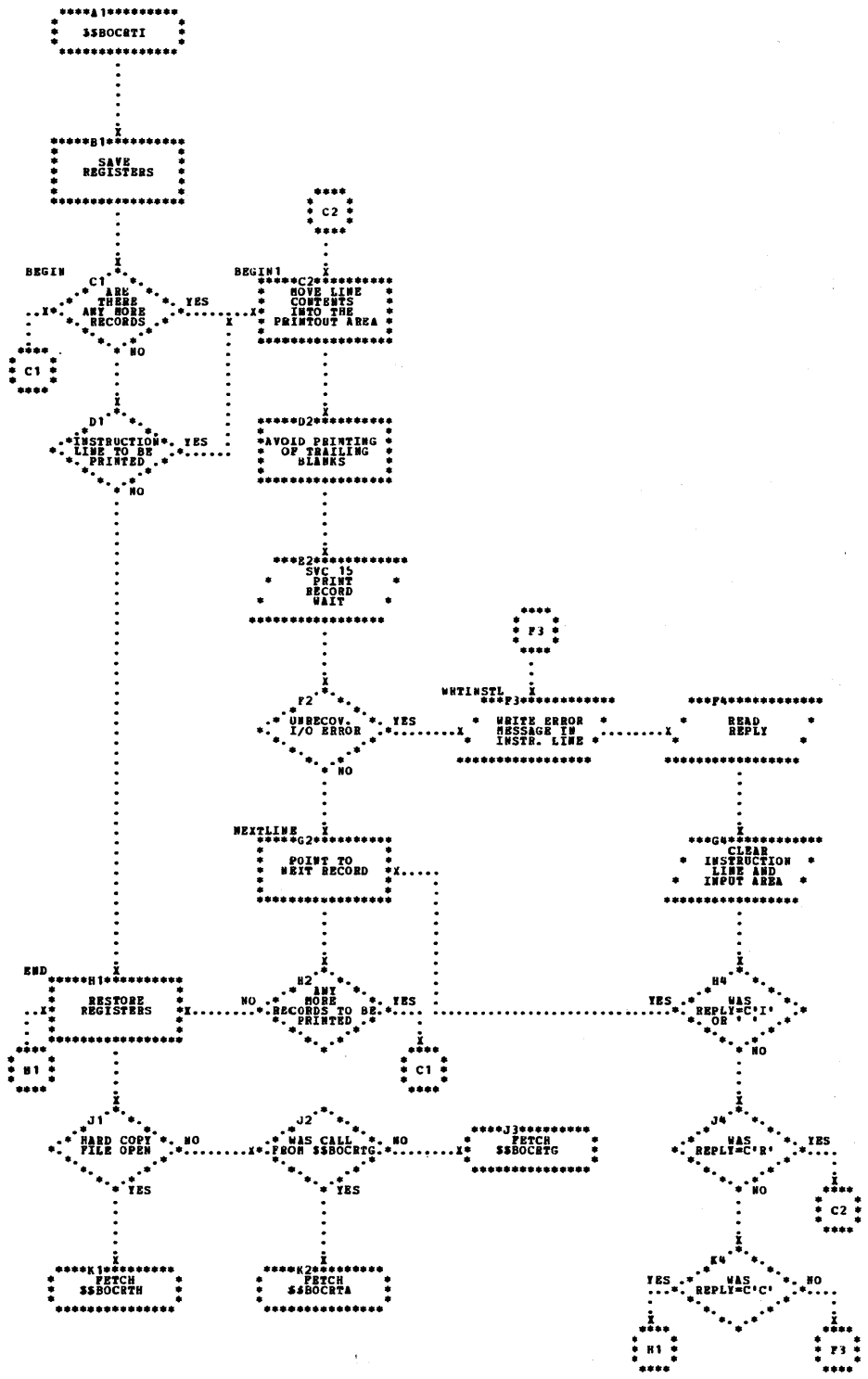


Chart MH. $$BOCRTK - Command Entry Processor (Part 1 of 4)
(Refer to Chart 19)

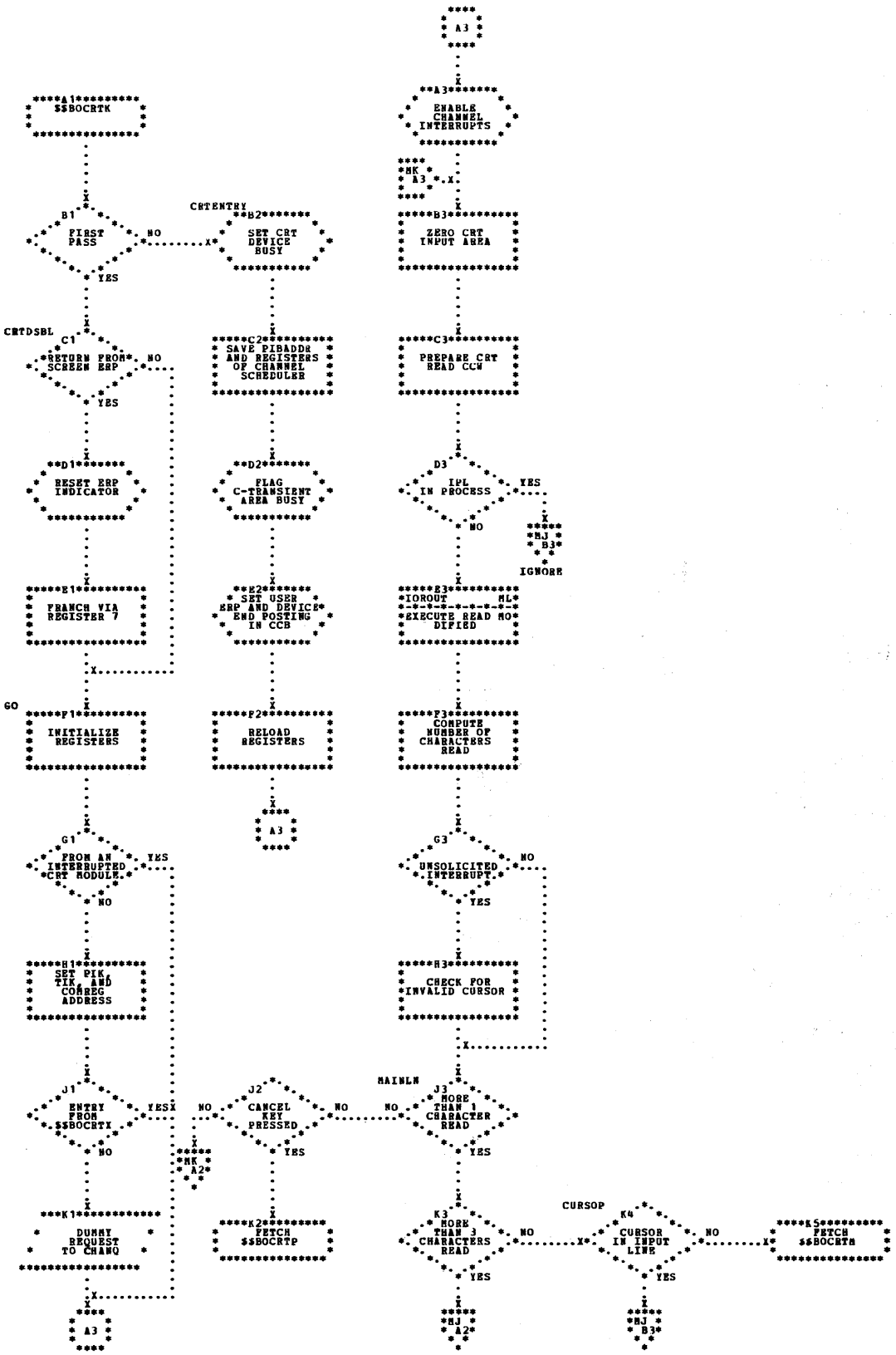


Chart MJ. $$BOCRTK - Command Entry Processor (Part 2 of 4)
(Refer to Chart 19)

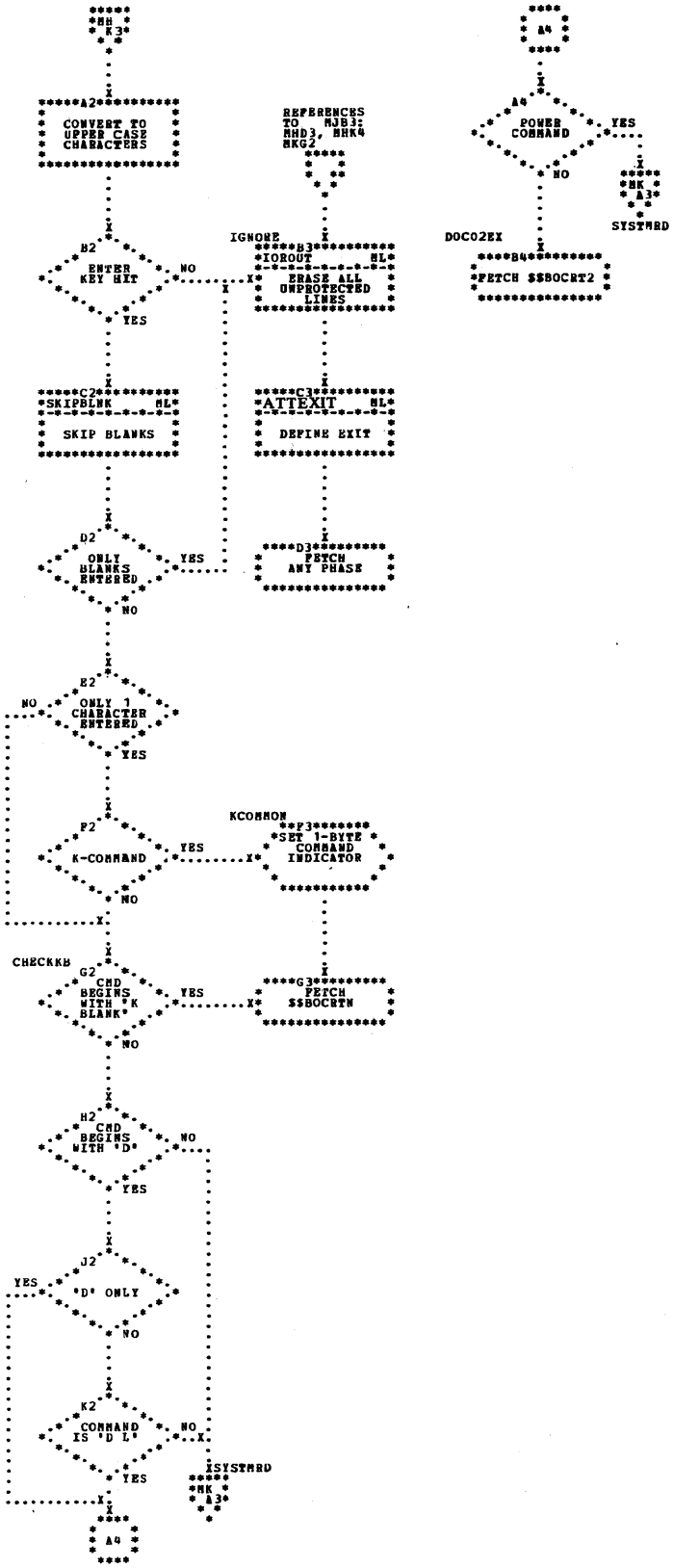


Chart MK. $$BOCRTK - Command Entry Processor (Part 3 of 4)
(Refer to Chart 19)

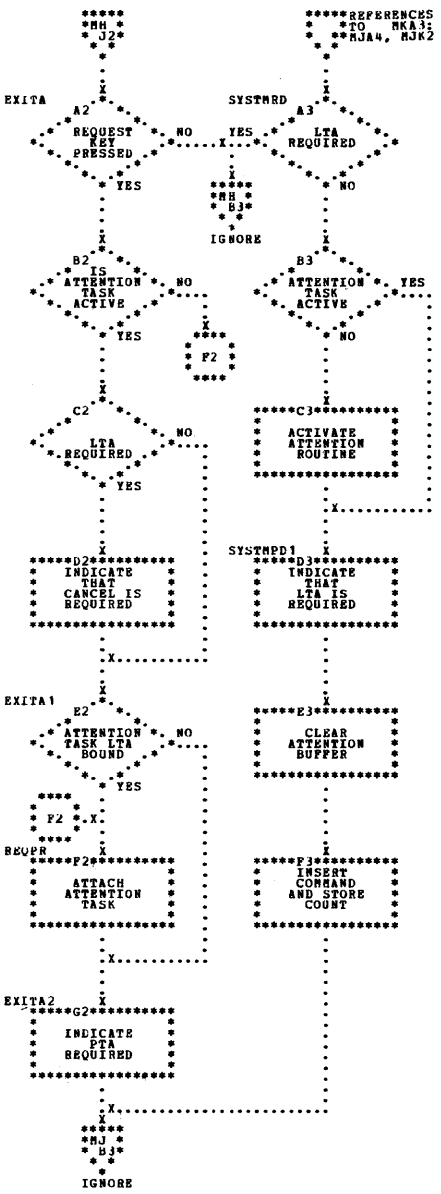


Chart ML. $$BOCRTK - Command Entry Processor (Part 4 of 4)
(Refer to Chart 19)

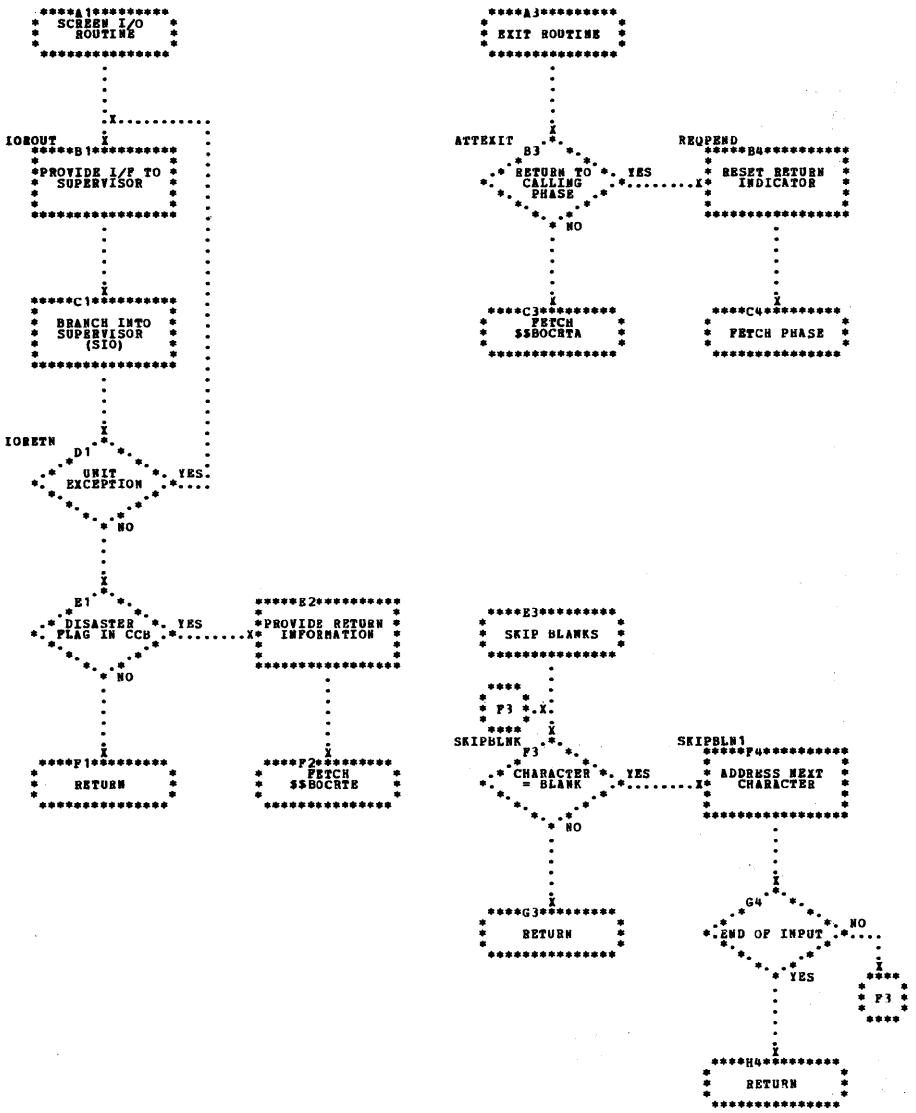


Chart MM. $$BOCRTL - Deletion by K-Command (Part 1 of 3)
(Refer to Chart 19)

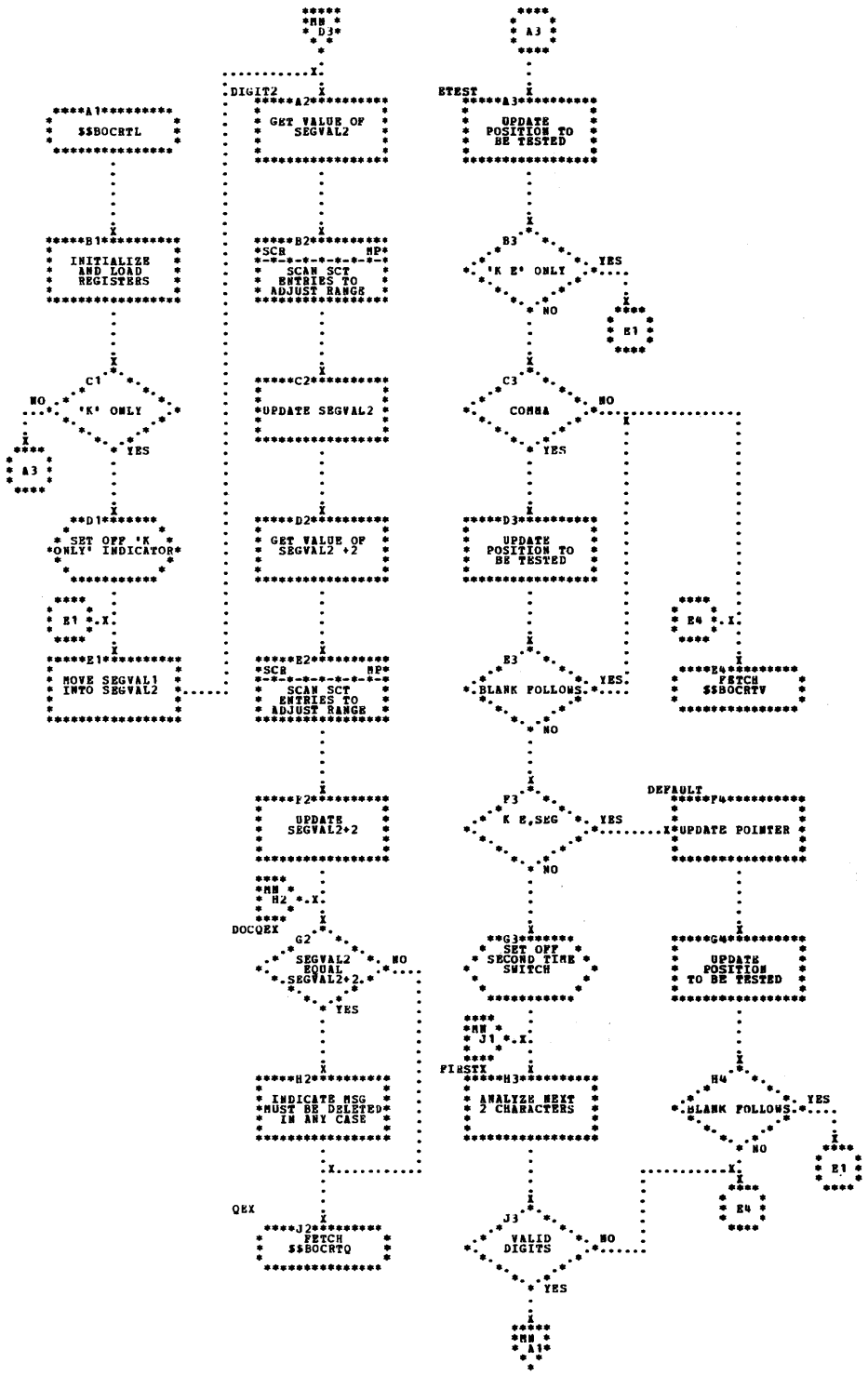


Chart MN. $$BOCRTL - Deletion by K-Command (Part 2 of 3)
(Refer to Chart 19)

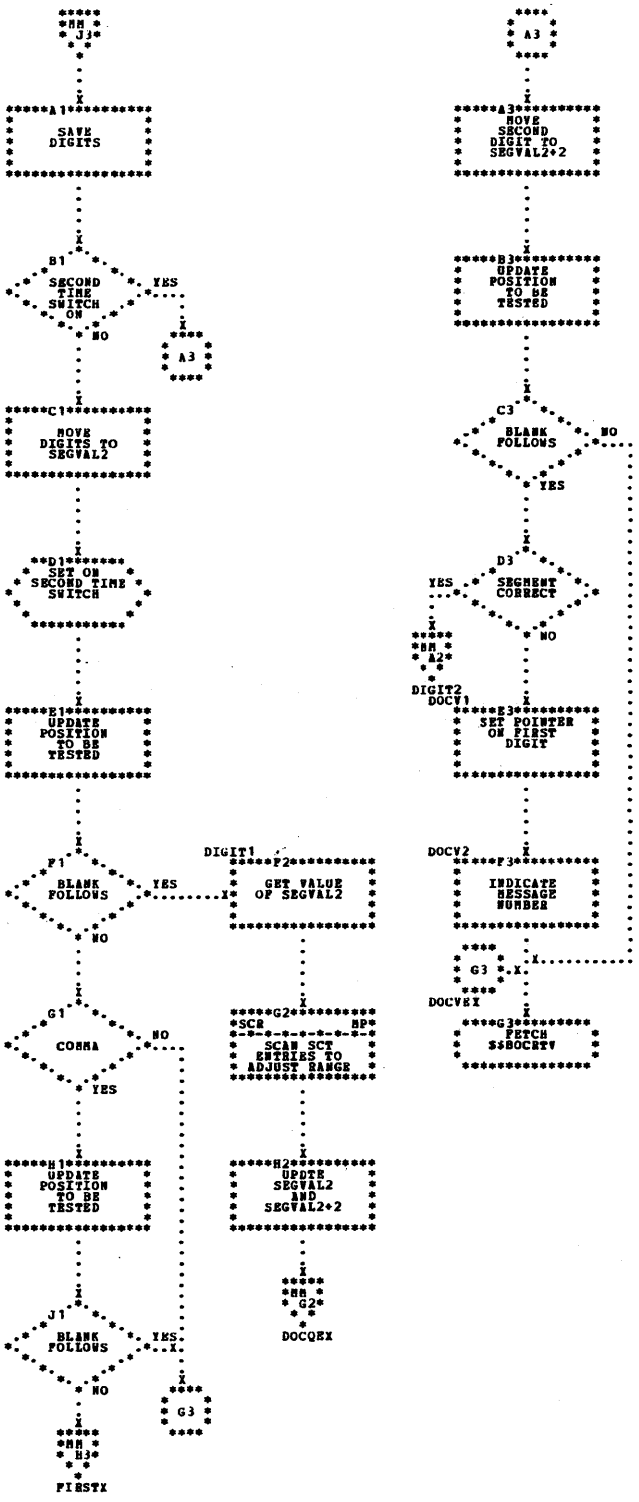


Chart MP. $$BOCRTL - Deletion by K-Command (Part 3 of 3)
(Refer to Chart 19)

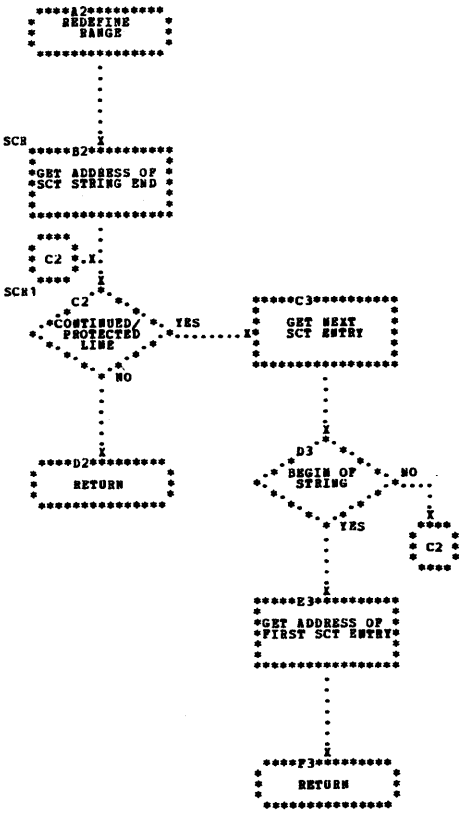


Chart NA. $$BOCRTM - Deletion by Cursor
(Refer to Chart 19)

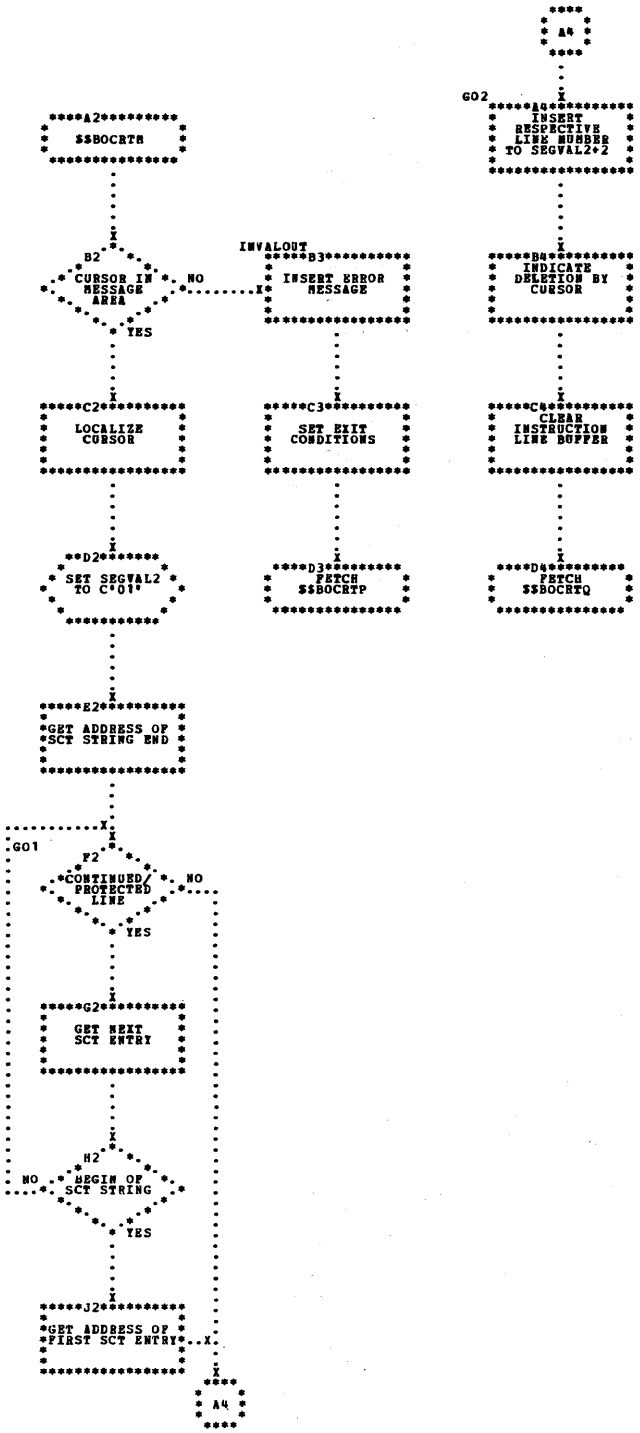


Chart NB. $$BOCRTN - Screen Management Mode (Part 1 of 3)
(Refer to Chart 19)

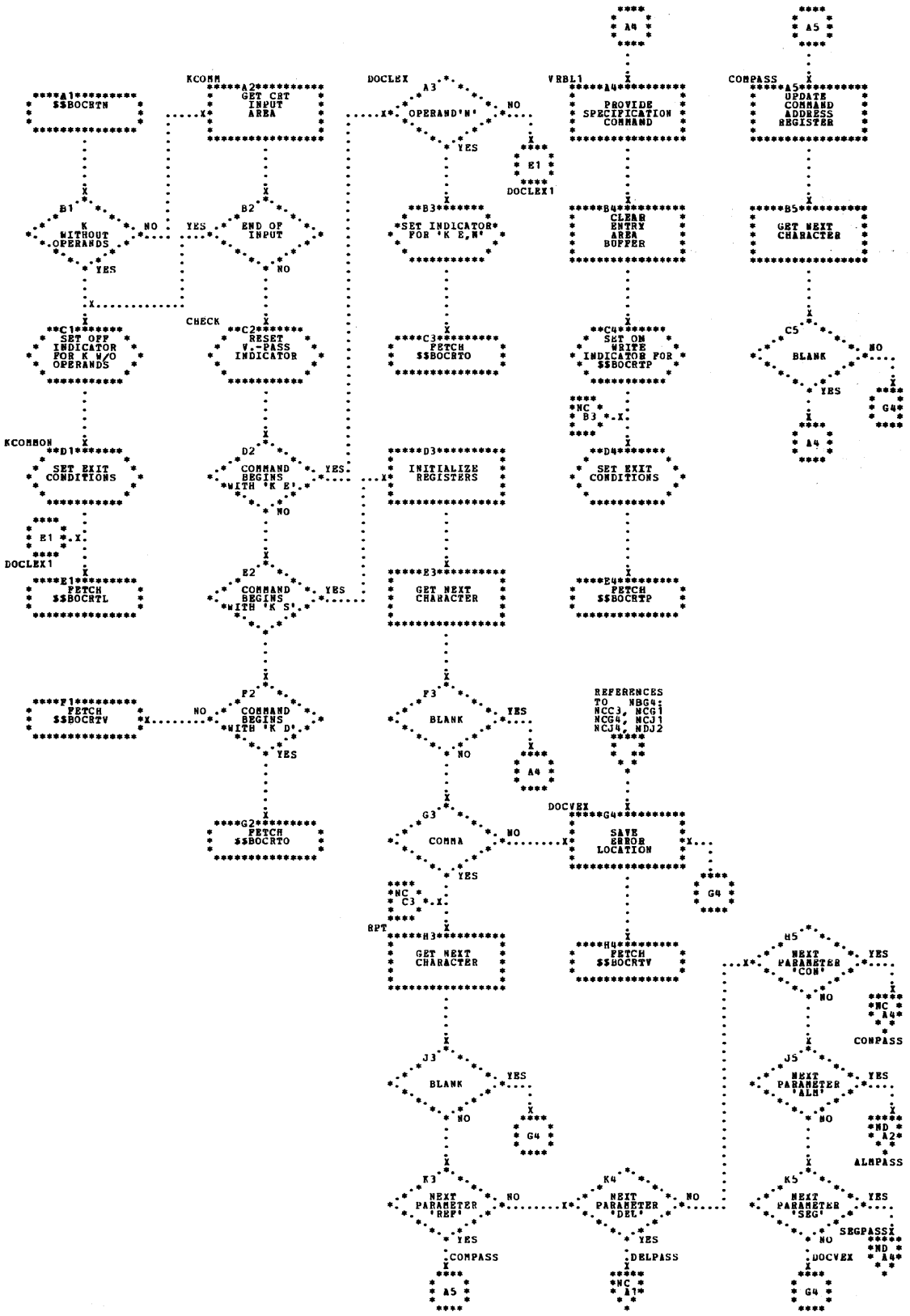


Chart NC. $$BOCRTN - Screen Management Mode (Part 2 of 3)
(Refer to Chart 19)

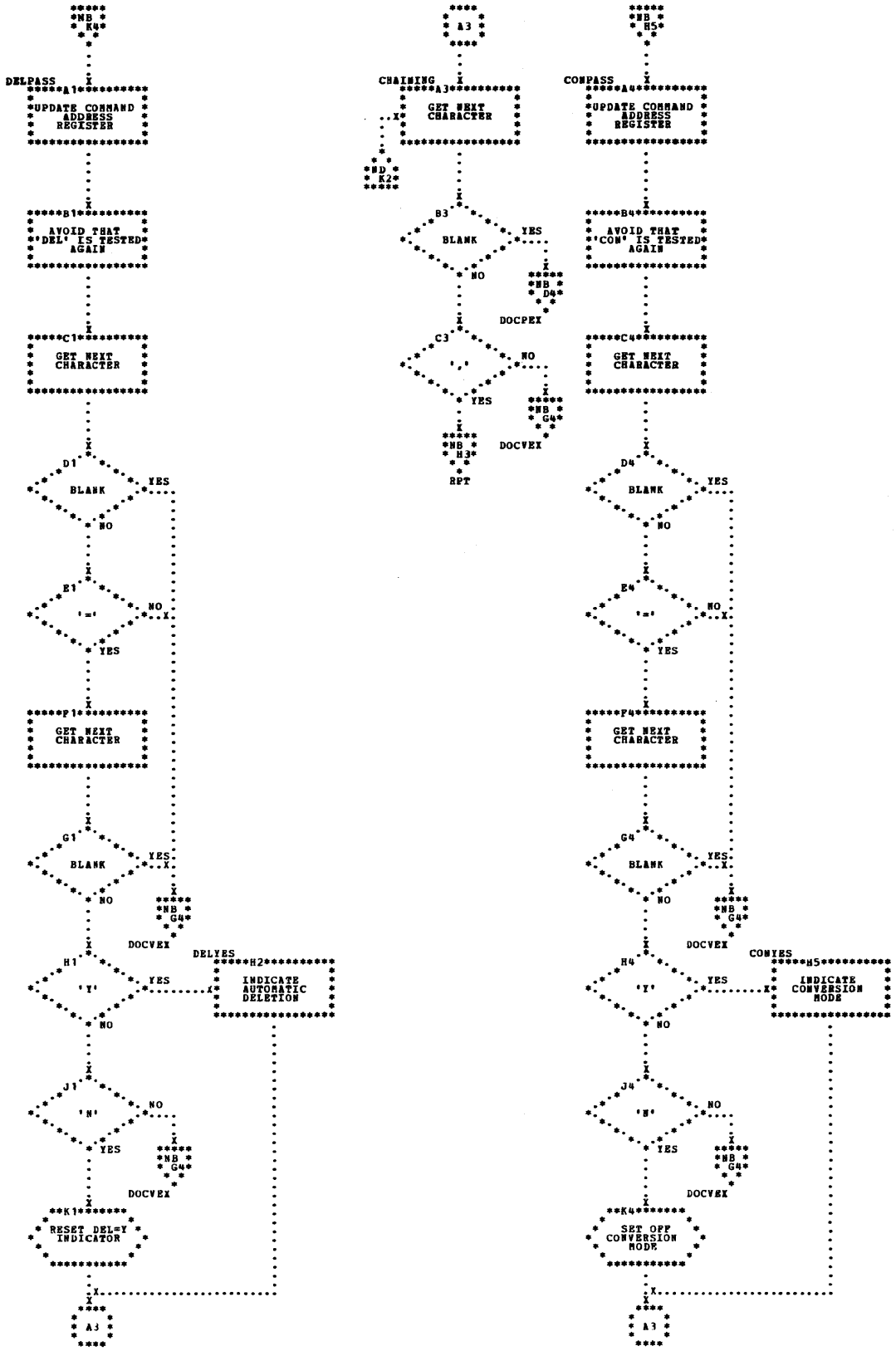


Chart ND. $$BOCRTN - Screen Management Mode (Part 3 of 3)
(Refer to Chart 19)

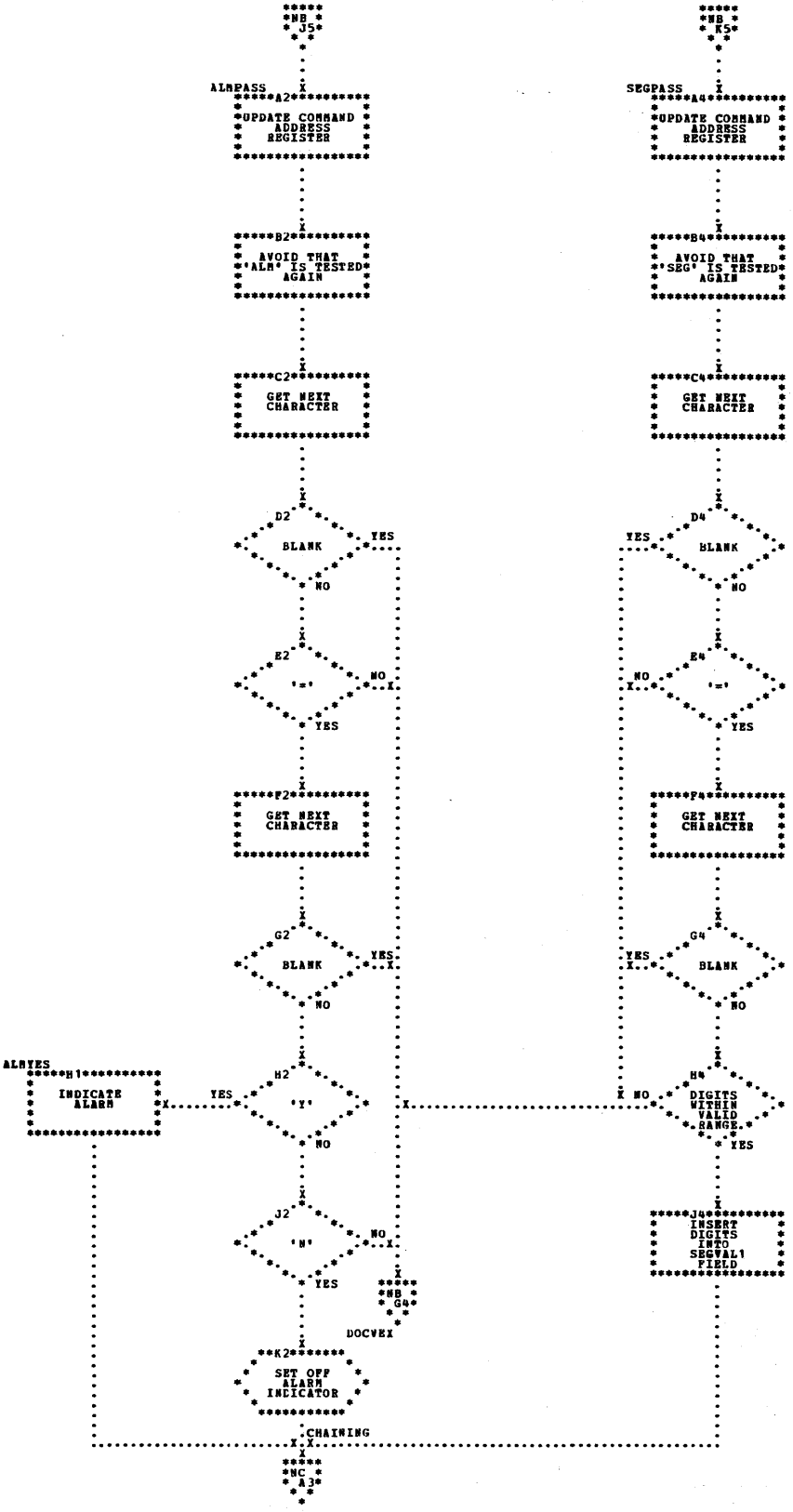


Chart NE. $$BOCRTO - Line Number Display (Part 1 of 2)
(Refer to Chart 19)

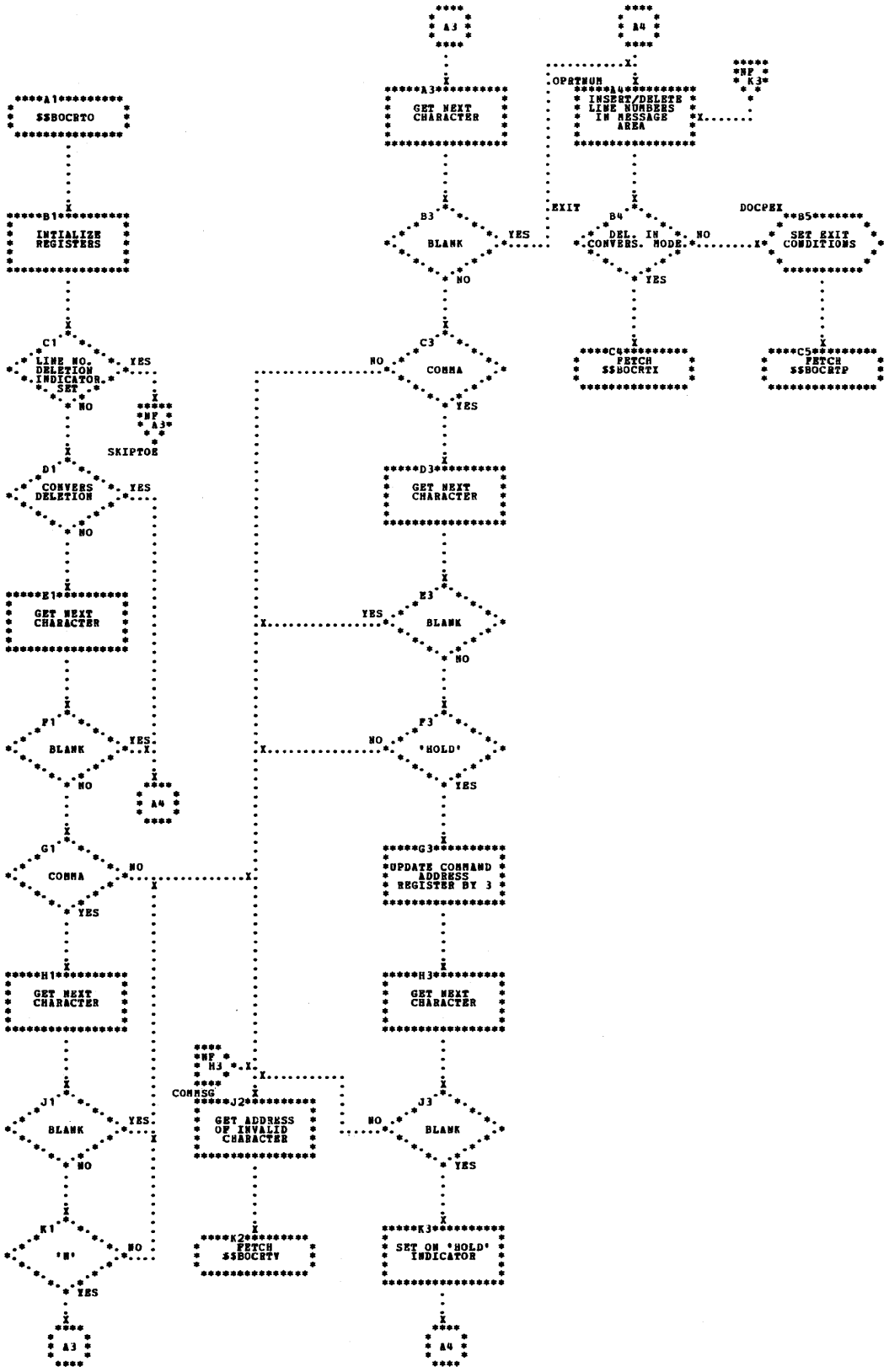


Chart NF. $$BOCRTO - Line Number Display (Part 2 of 2)
(Refer to Chart 19)

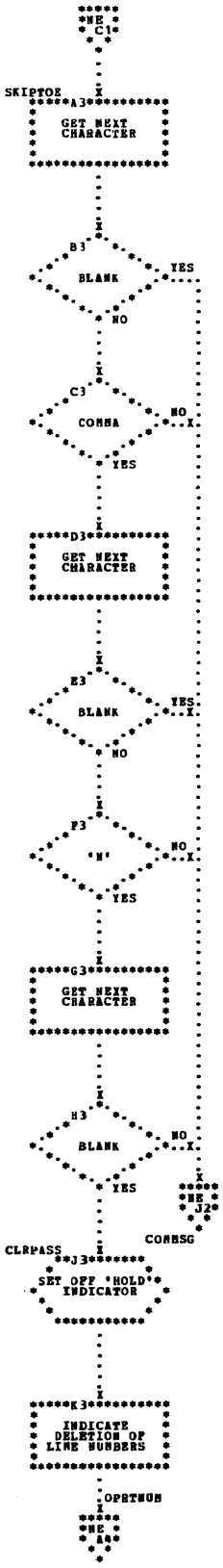


Chart NG. $$BOCRTP - Exit Interphase (Part 1 of 2)
(Refer to Chart 19)

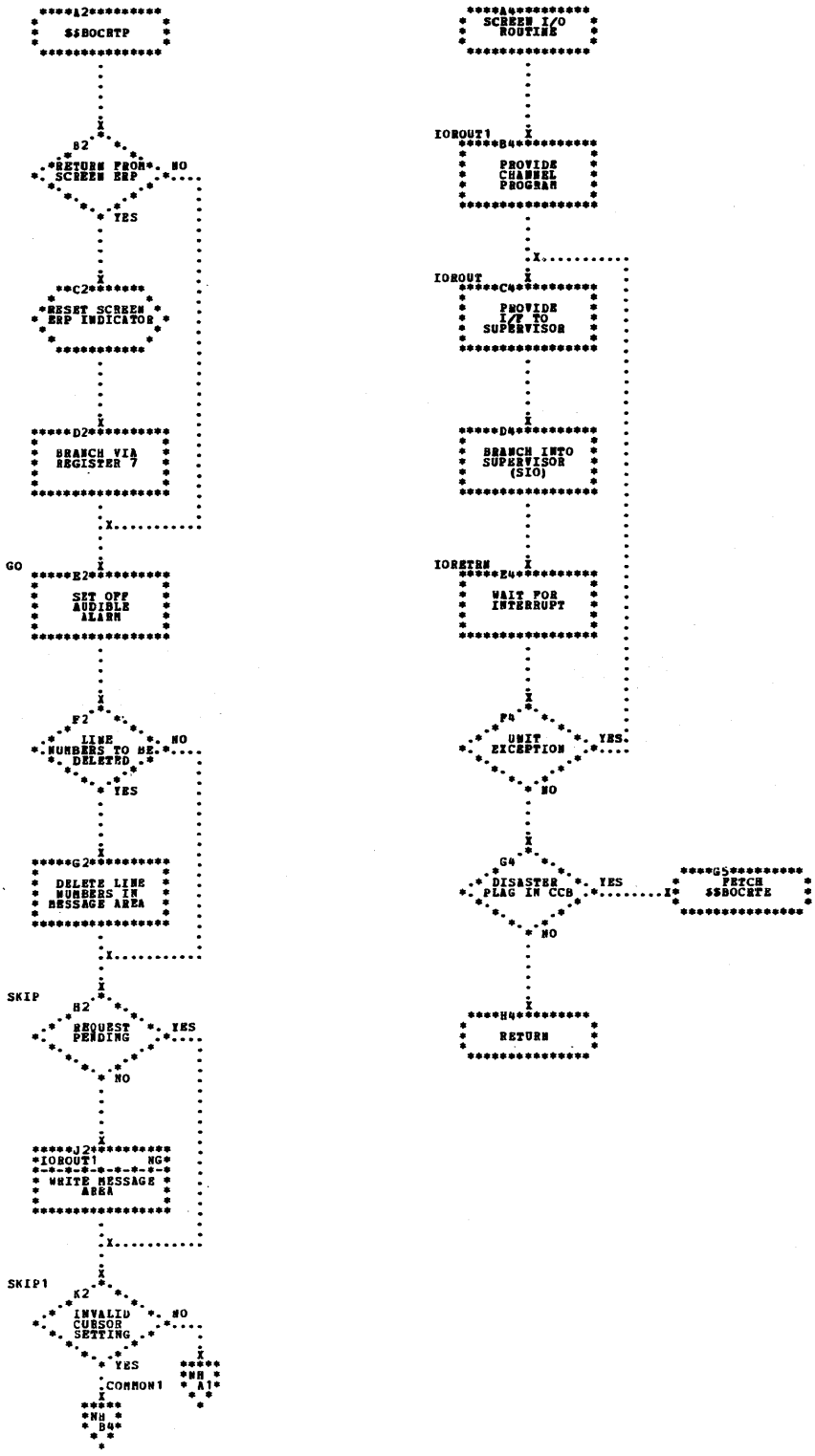


Chart NH. $$BOCRTP - Exit Interphase (Part 2 of 2)
(Refer to Chart 19)

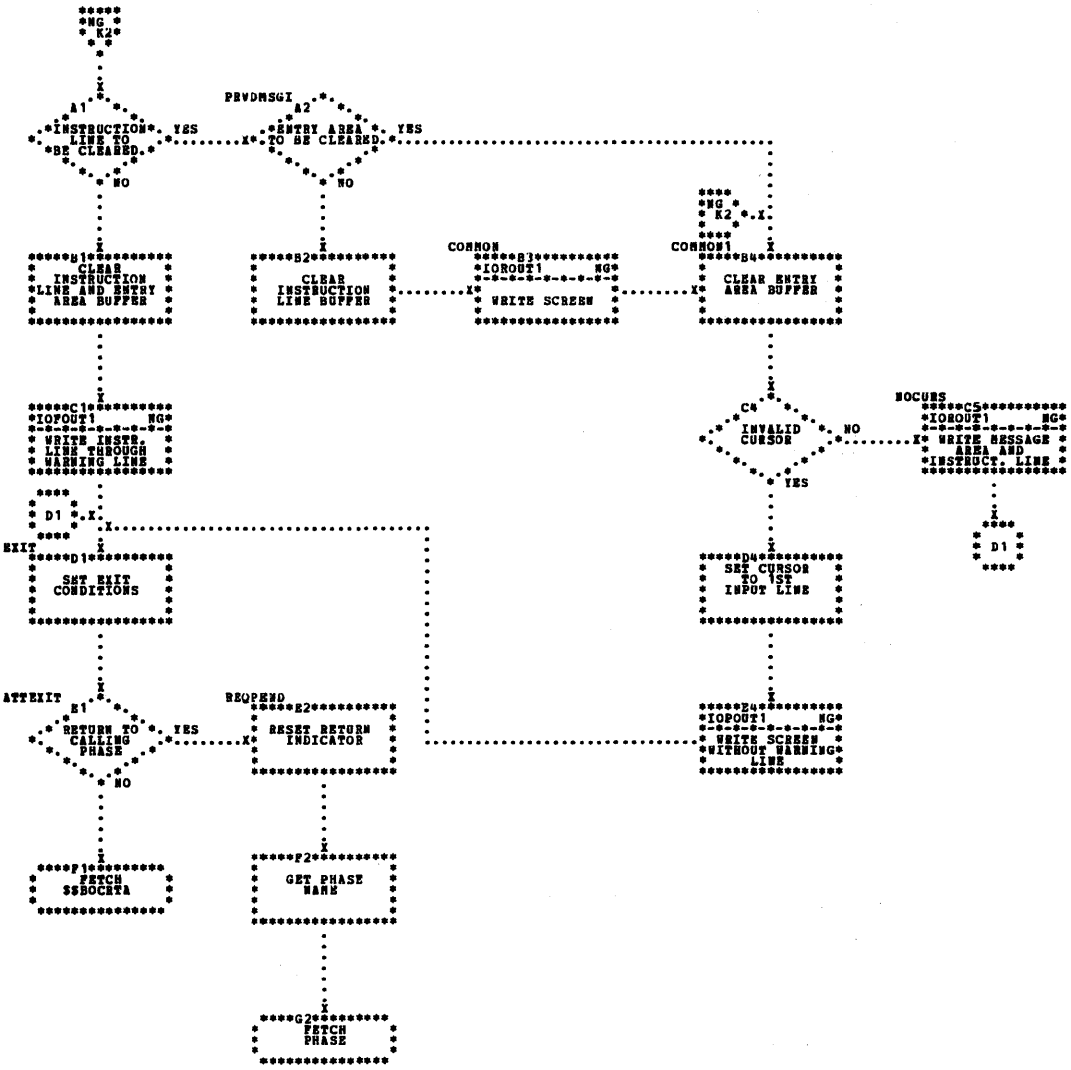


Chart NJ. $$BOCRTQ - Deletion of Message Lines (Part 1 of 2)
(Refer to Chart 19)

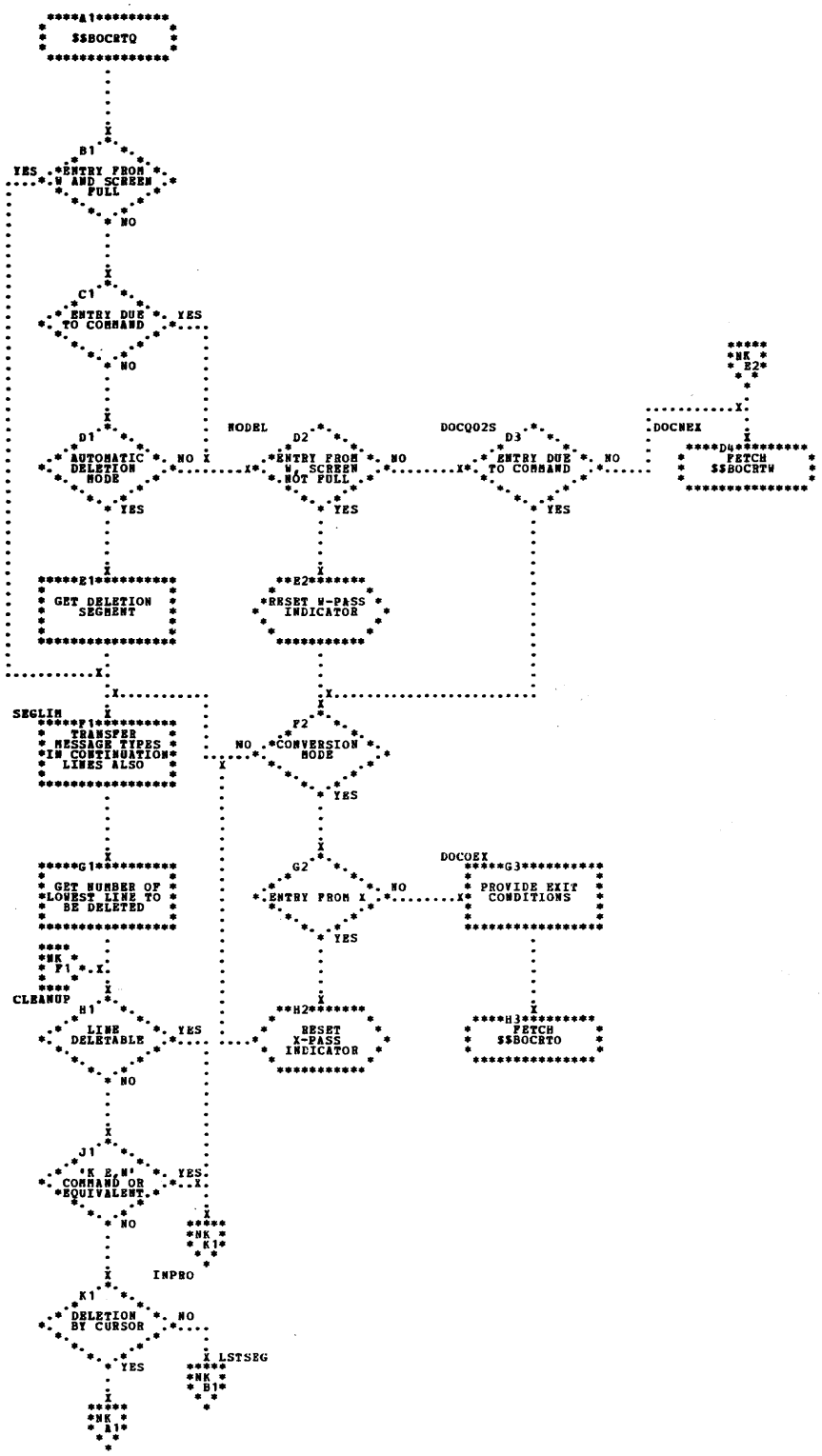


Chart NK. $$BOCRTQ - Deletion of Message Lines (Part 2 of 2)
(Refer to Chart 19)

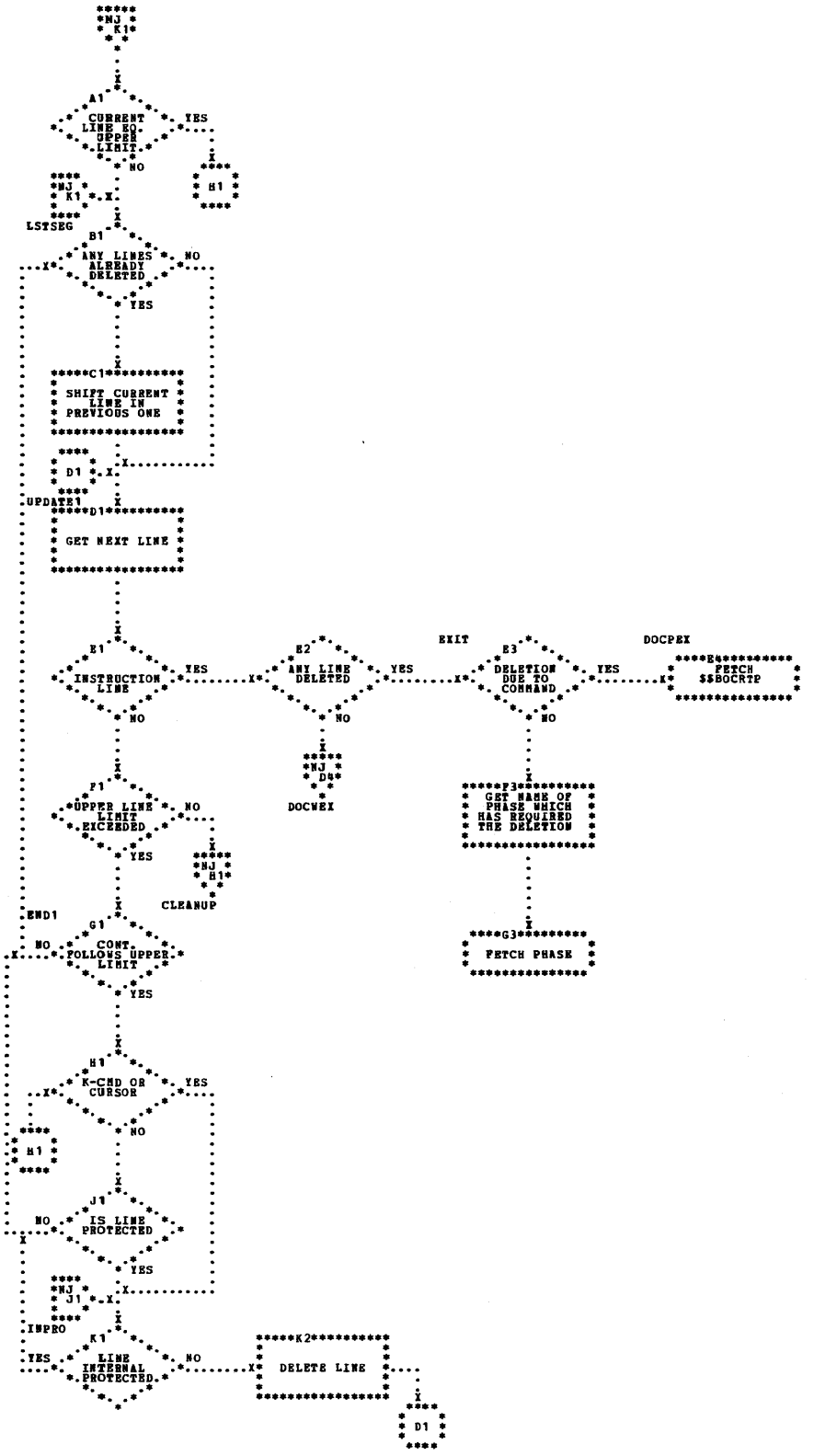


Chart PA. $$BOCRTR - Read Processor Part 1 (Part 1 of 3)
(Refer to Chart 18)

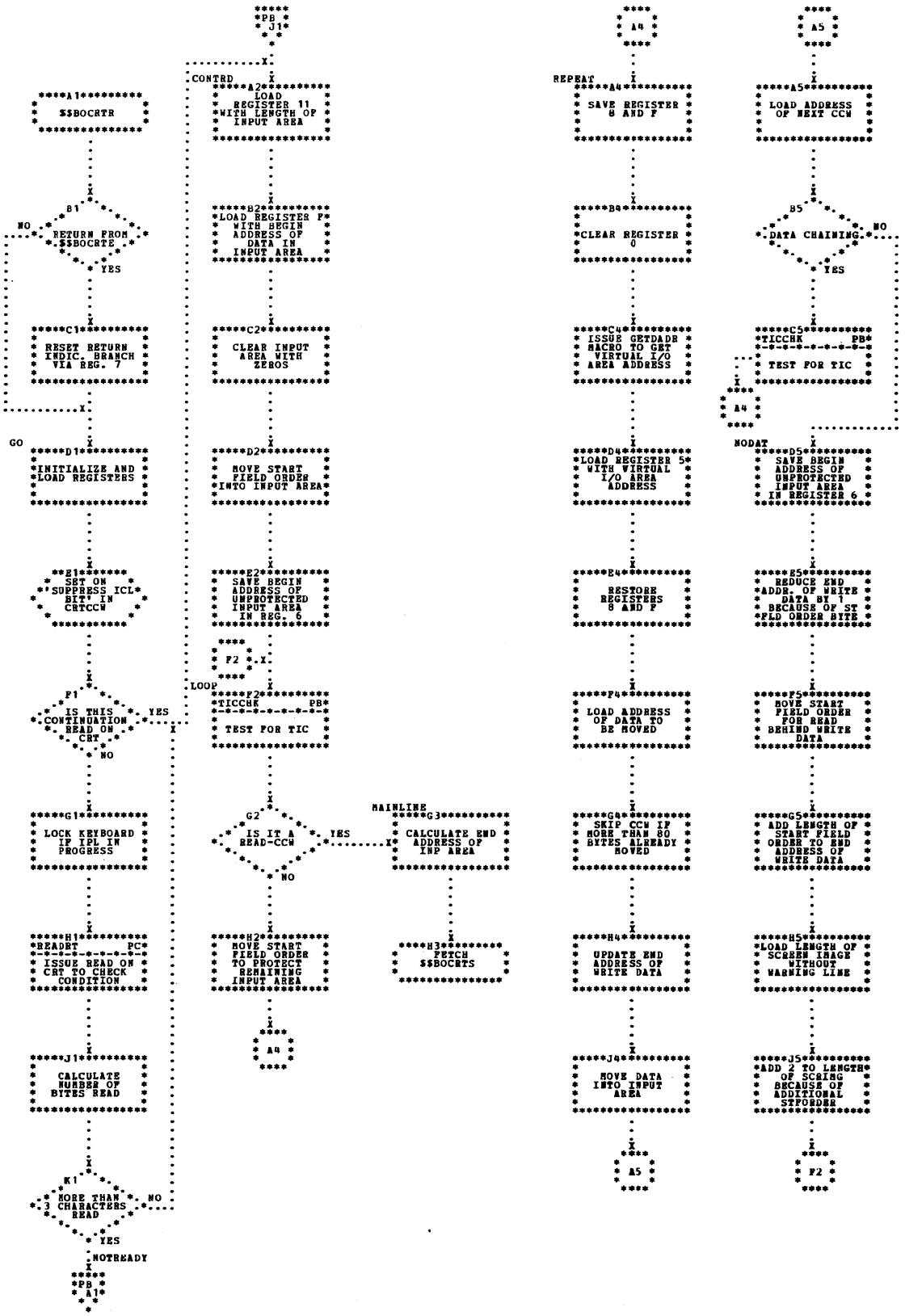


Chart PB. $$BOCRTR - Read Processor Part 1 (Part 2 of 3)
(Refer to Chart 18)

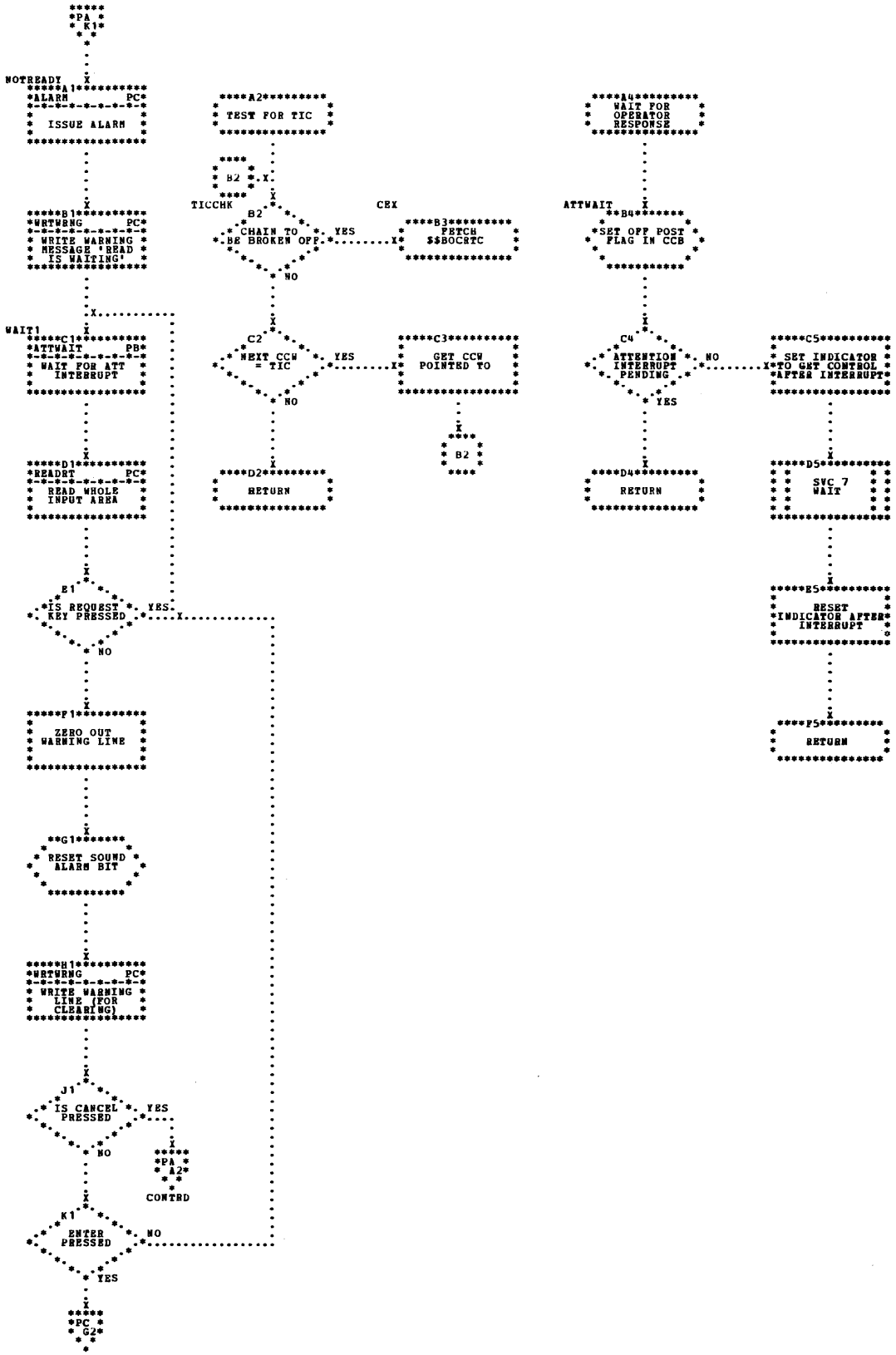


Chart PC. $$BOCRTR - Read Processor Part 1 (Part 3 of 3)
(Refer to Chart 18)

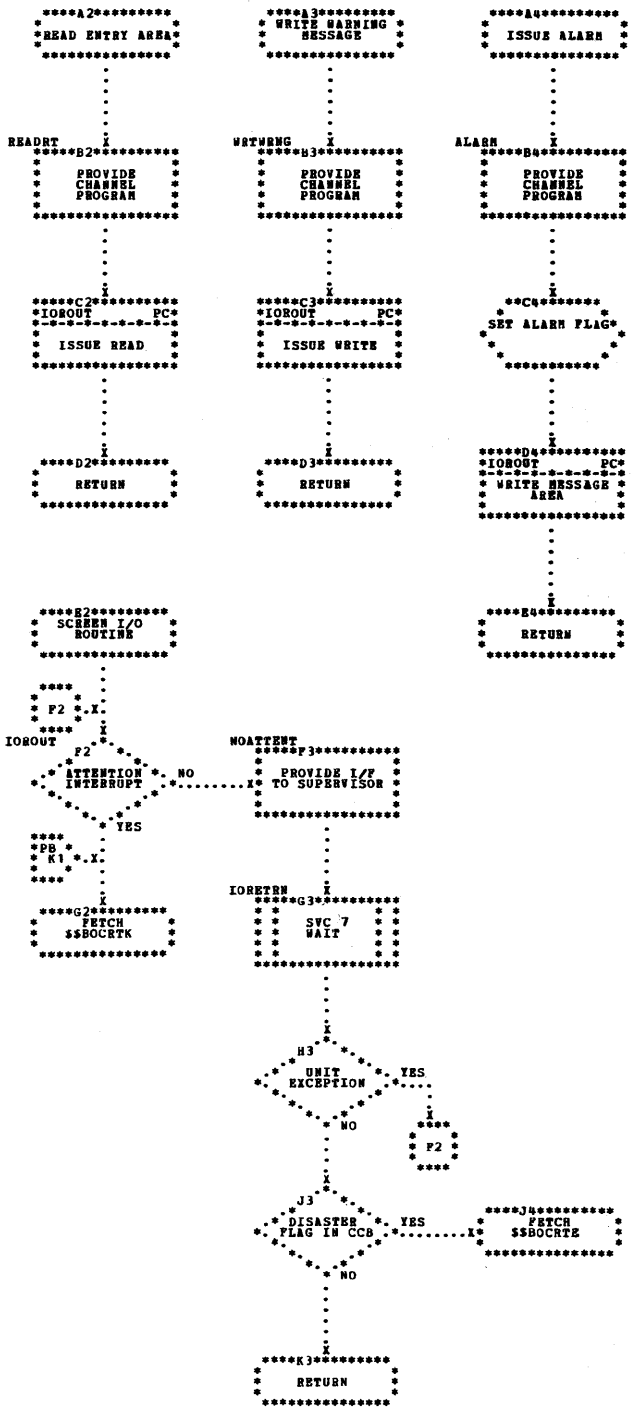


Chart PD. $$BOCRTS - Read Processor Part 2 (Part 1 of 3)
(Refer to Chart 18)

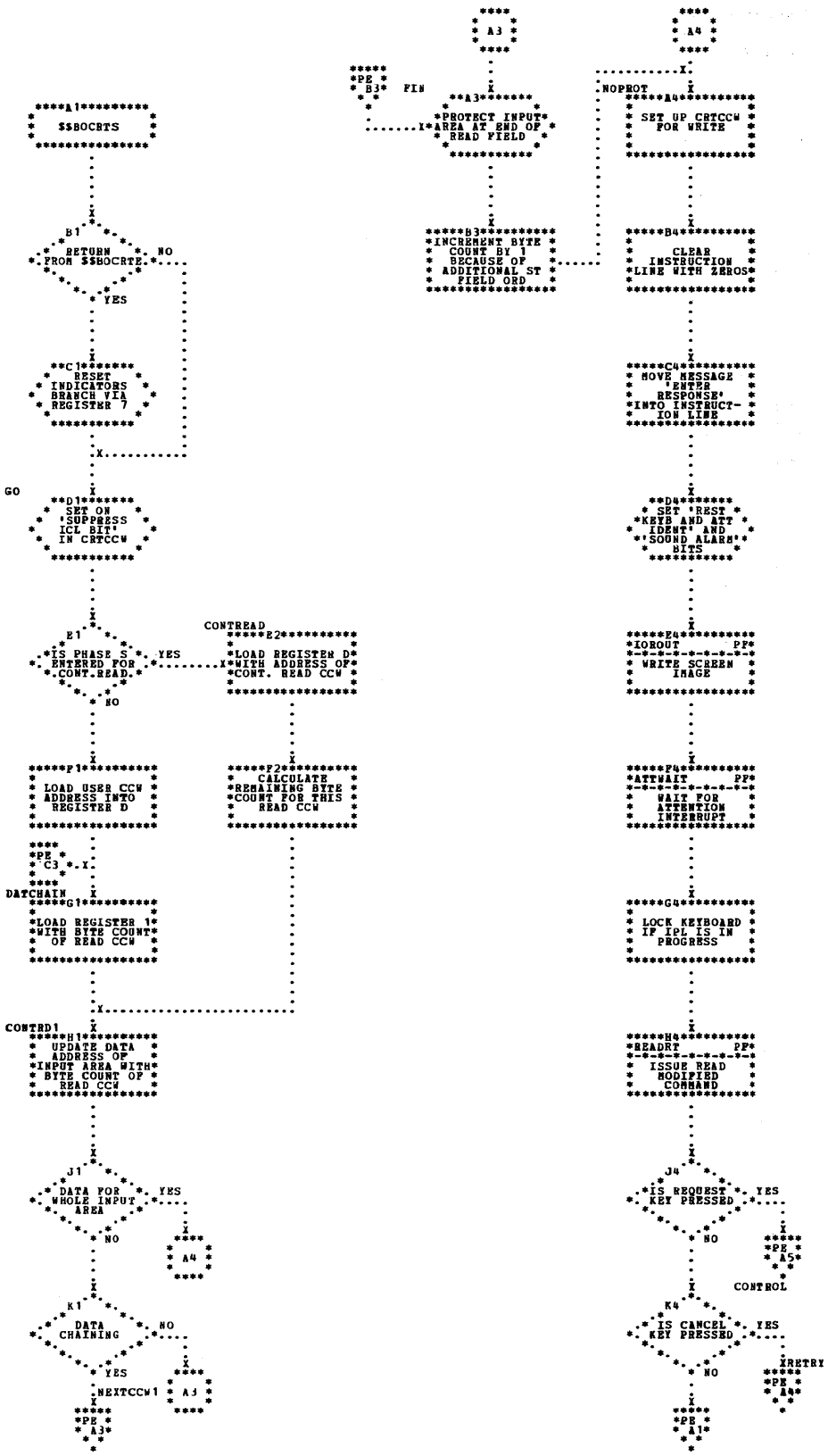


Chart PE. $$BOCRTS - Read Processor Part 2 (Part 2 of 3)
(Refer to Chart 18)

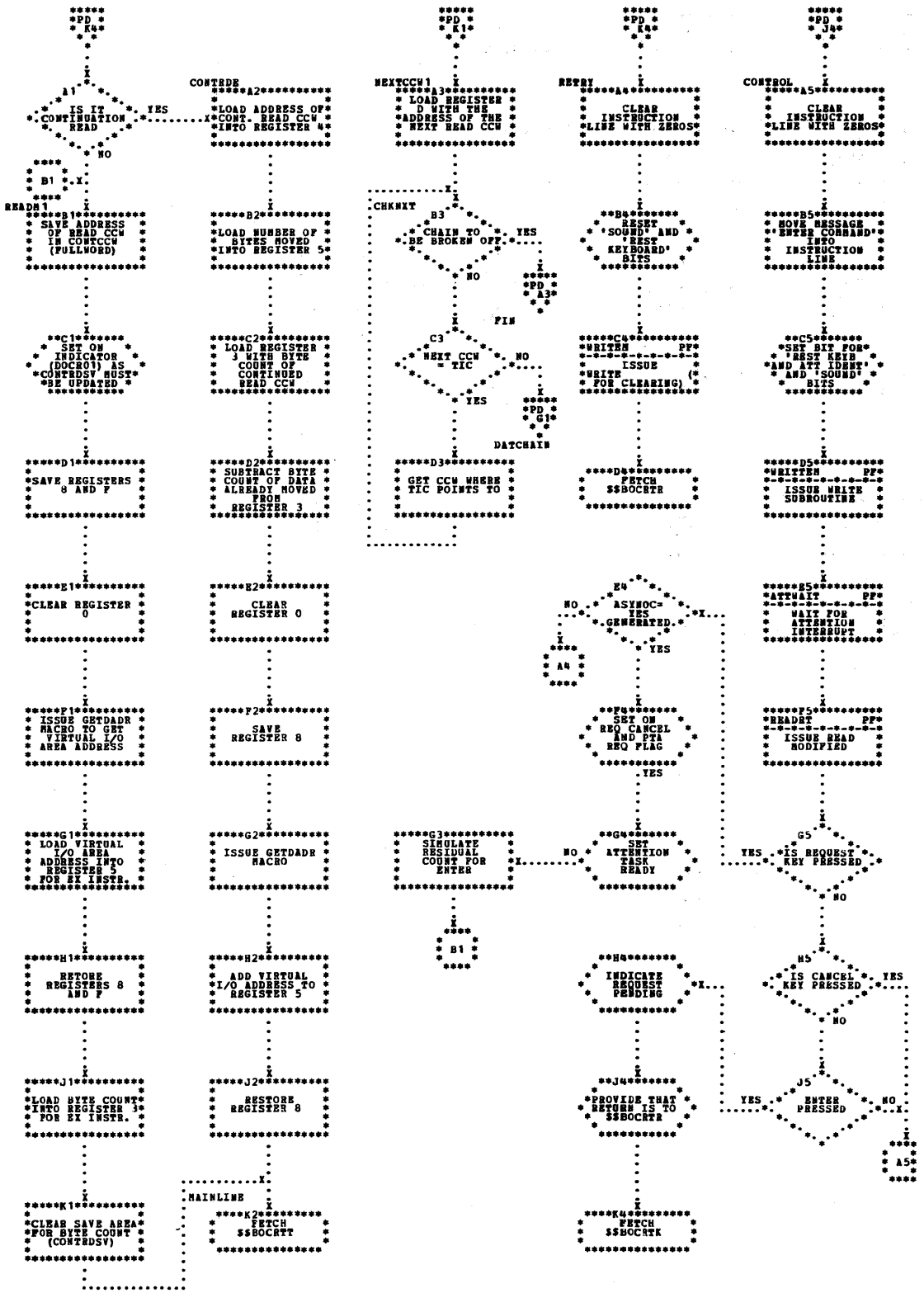


Chart PF. $$BOCRTS - Read Processor Part 2 (Part 3 of 3)
(Refer to Chart 18)

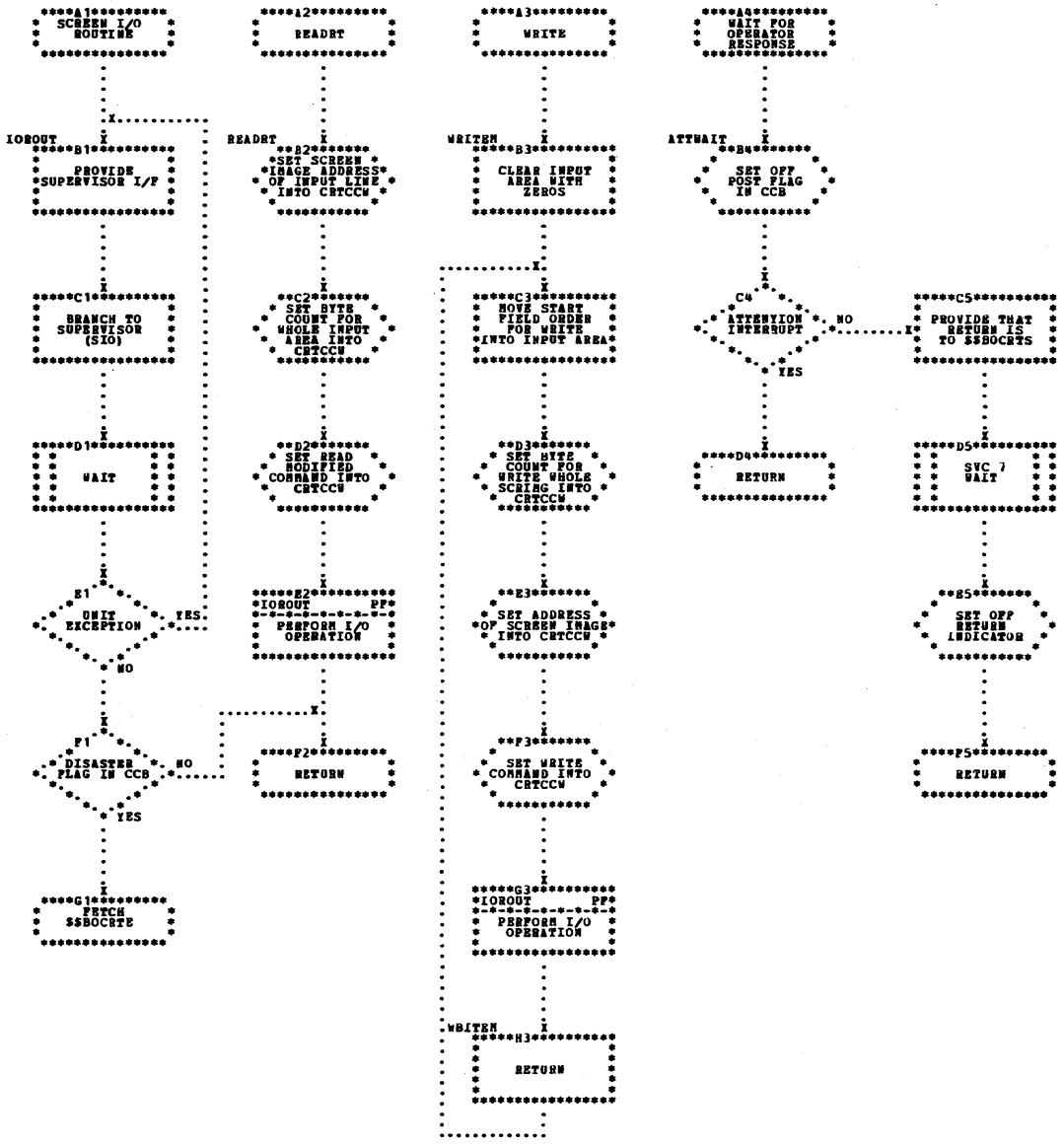


Chart PG. $$BOCRTT - Read Processor Part 3 (Part 1 of 2)
(Refer to Chart 18)

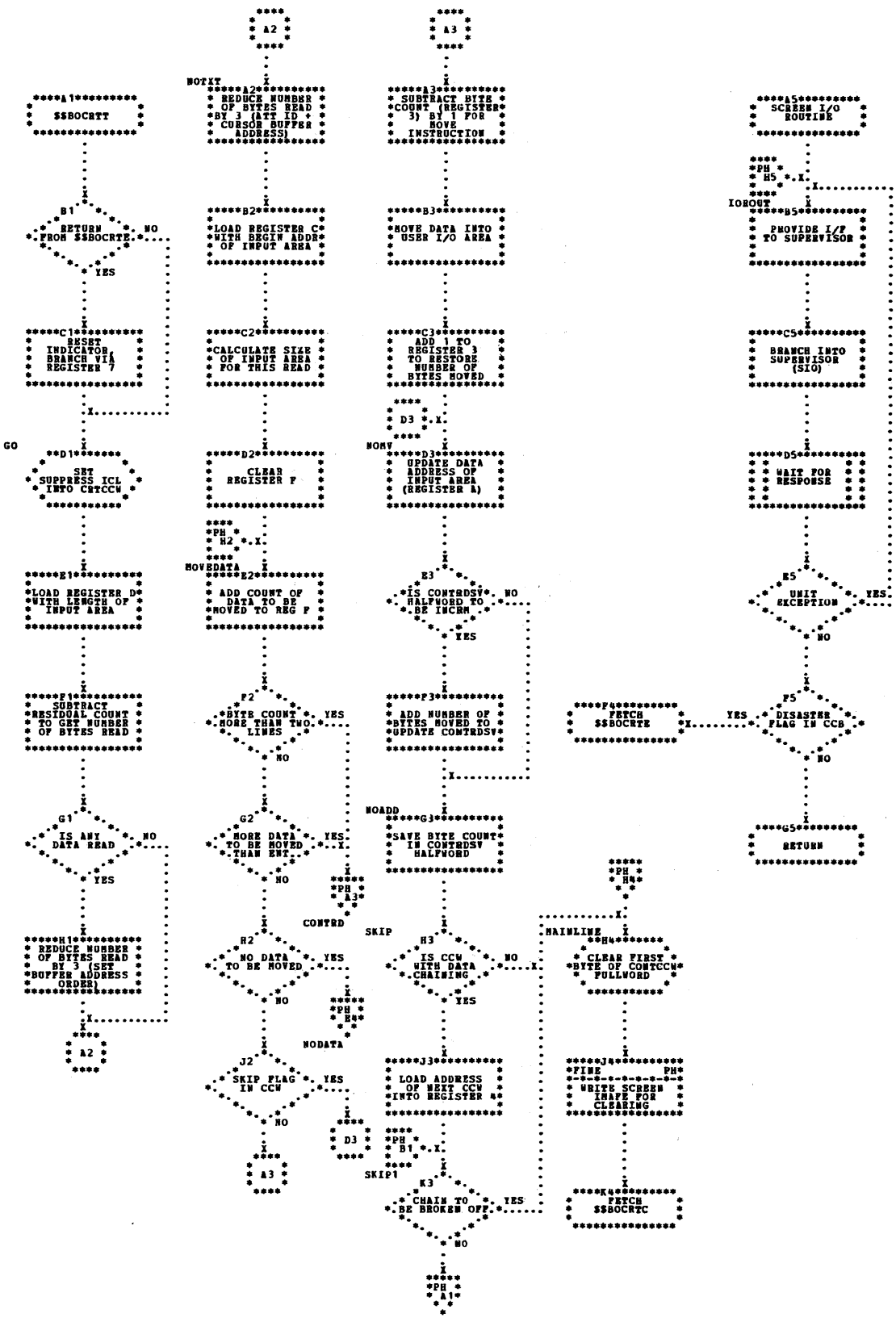


Chart PH. $$BOCRTT - Read Processor Part 3 (Part 2 of 2)
(Refer to Chart 18)

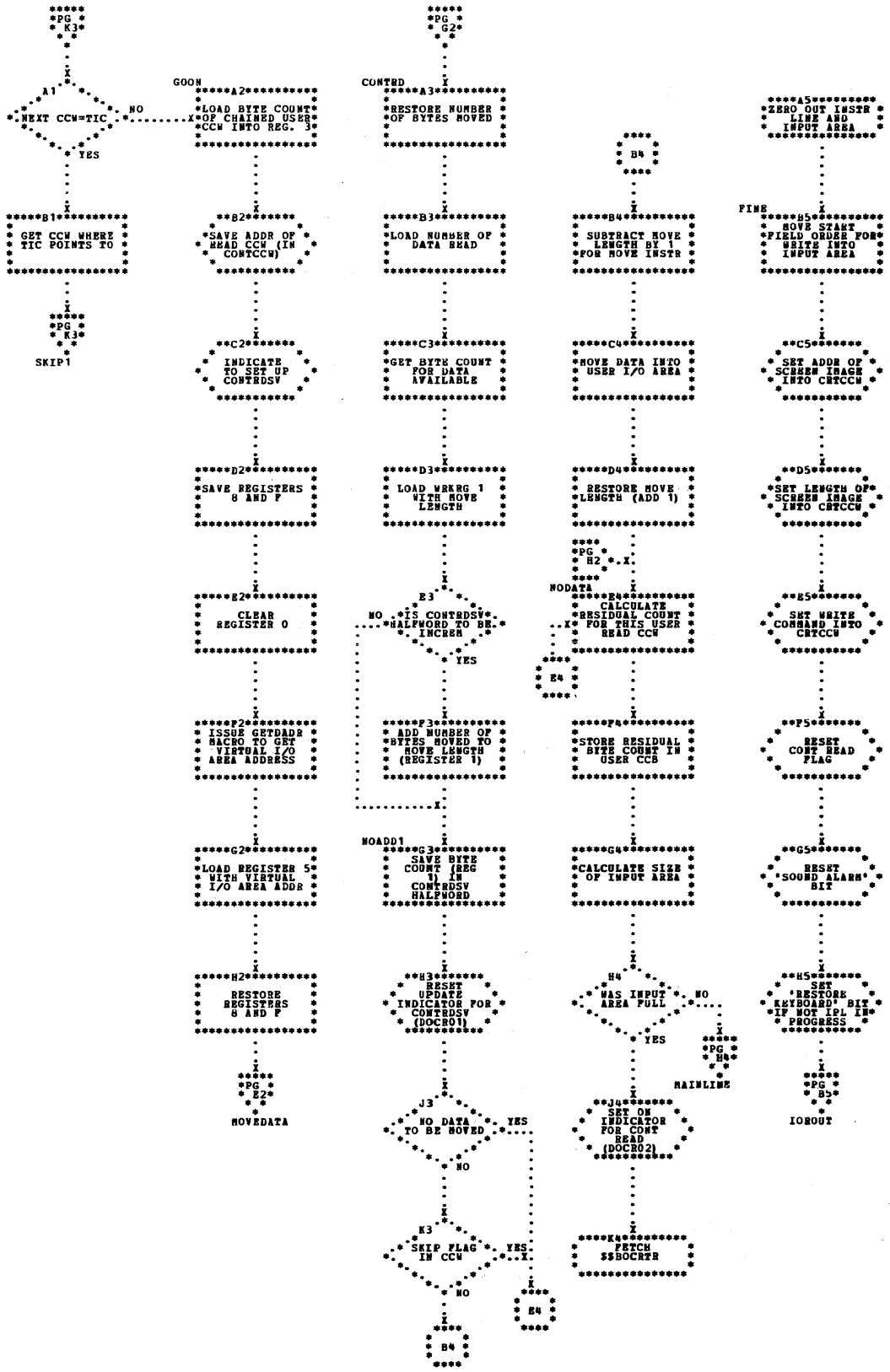
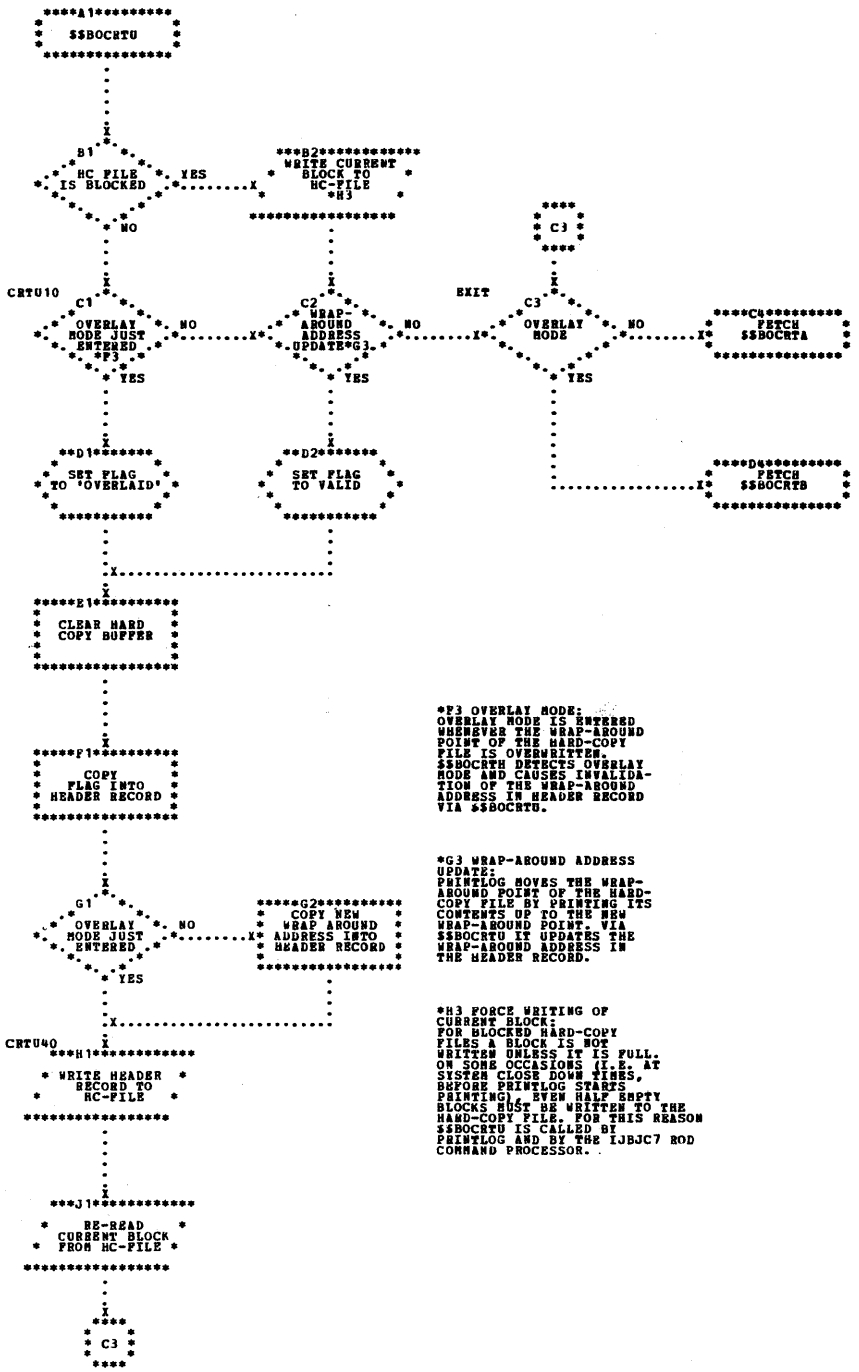


Chart PJ. $$BOCRTU - Update Wrap-Around Address
(Refer to Chart 16)



*F3 OVERLAY MODE:
OVERLAY MODE IS ENTERED WHENEVER THE WRAP-AROUND POINT OF THE HARD-COPY FILE IS OVERWRITTEN. $$BOCRTH DETECTS OVERLAY MODE AND CAUSES INVALIDATION OF THE WRAP-AROUND ADDRESS IN HEADER RECORD VIA $$BOCRTU.

*G3 WRAP-AROUND ADDRESS UPDATE:
PRINTLOG MOVES THE WRAP-AROUND POINT OF THE HARD-COPY FILE BY PRINTING ITS CONTENTS UP TO THE NEW WRAP-AROUND POINT. VIA $$BOCRTU IT UPDATES THE WRAP-AROUND ADDRESS IN THE HEADER RECORD.

*H3 FORCE WRITING OF CURRENT BLOCK:
FOR BLOCKED HARD-COPY FILES A BLOCK IS NOT WRITTEN UNLESS IT IS FULL. ON SOME OCCASIONS (I.E. AT SYSTEM CLOSE DOWN TIMES, BEFORE PRINTLOG STARTS PRINTING) EVEN HALF EMPTY BLOCKS MUST BE WRITTEN TO THE HARD-COPY FILE. FOR THIS REASON $$BOCRTU IS CALLED BY PRINTLOG AND BY THE IJBJC7 ROD COMMAND PROCESSOR.

Chart PK. $$BOCRTV - COMMAND Error Message Writer
(Refer to Chart 20)

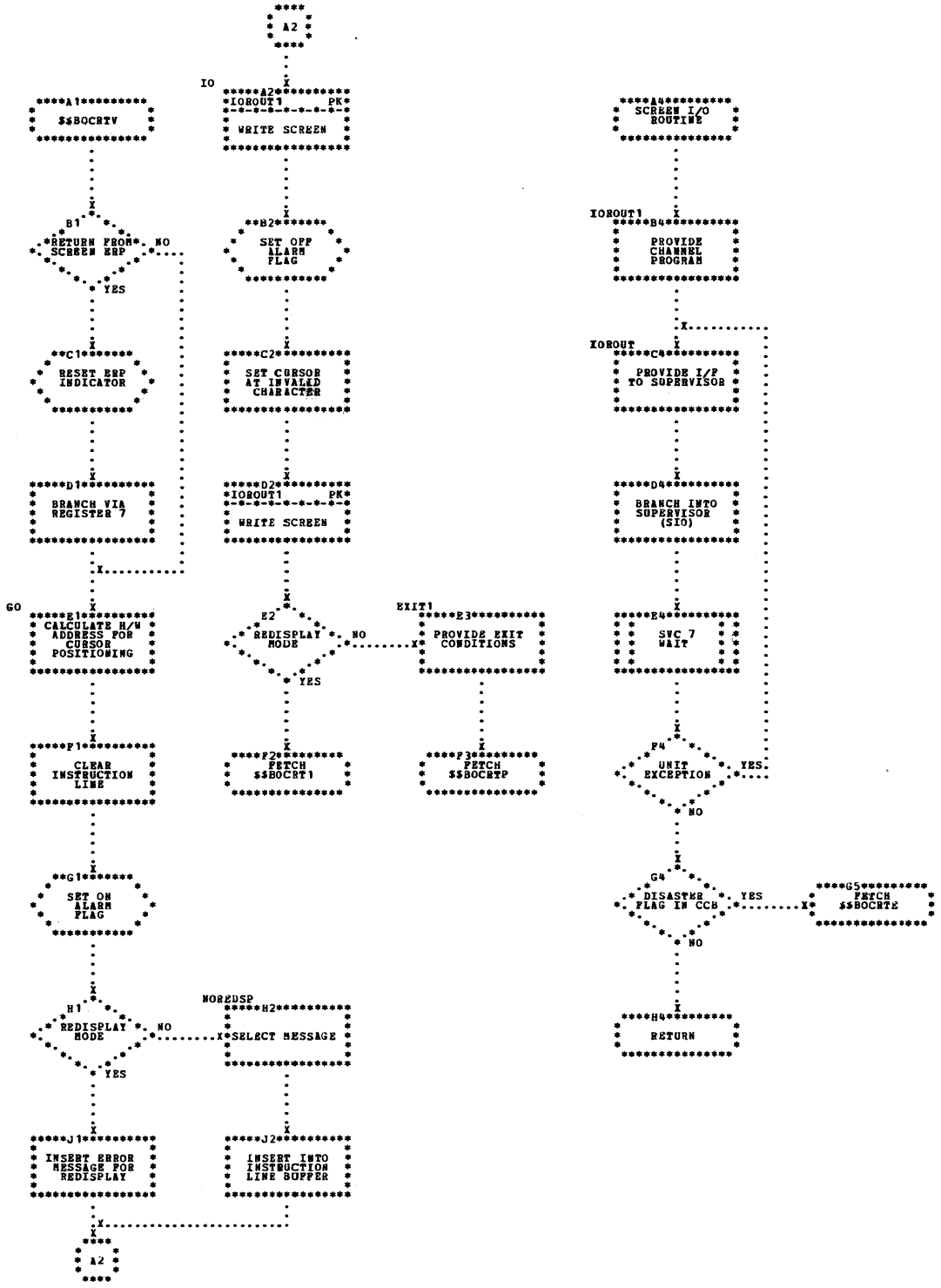


Chart QA. $$BOCRTW - Display Warning Message
(Refer to Chart 20)

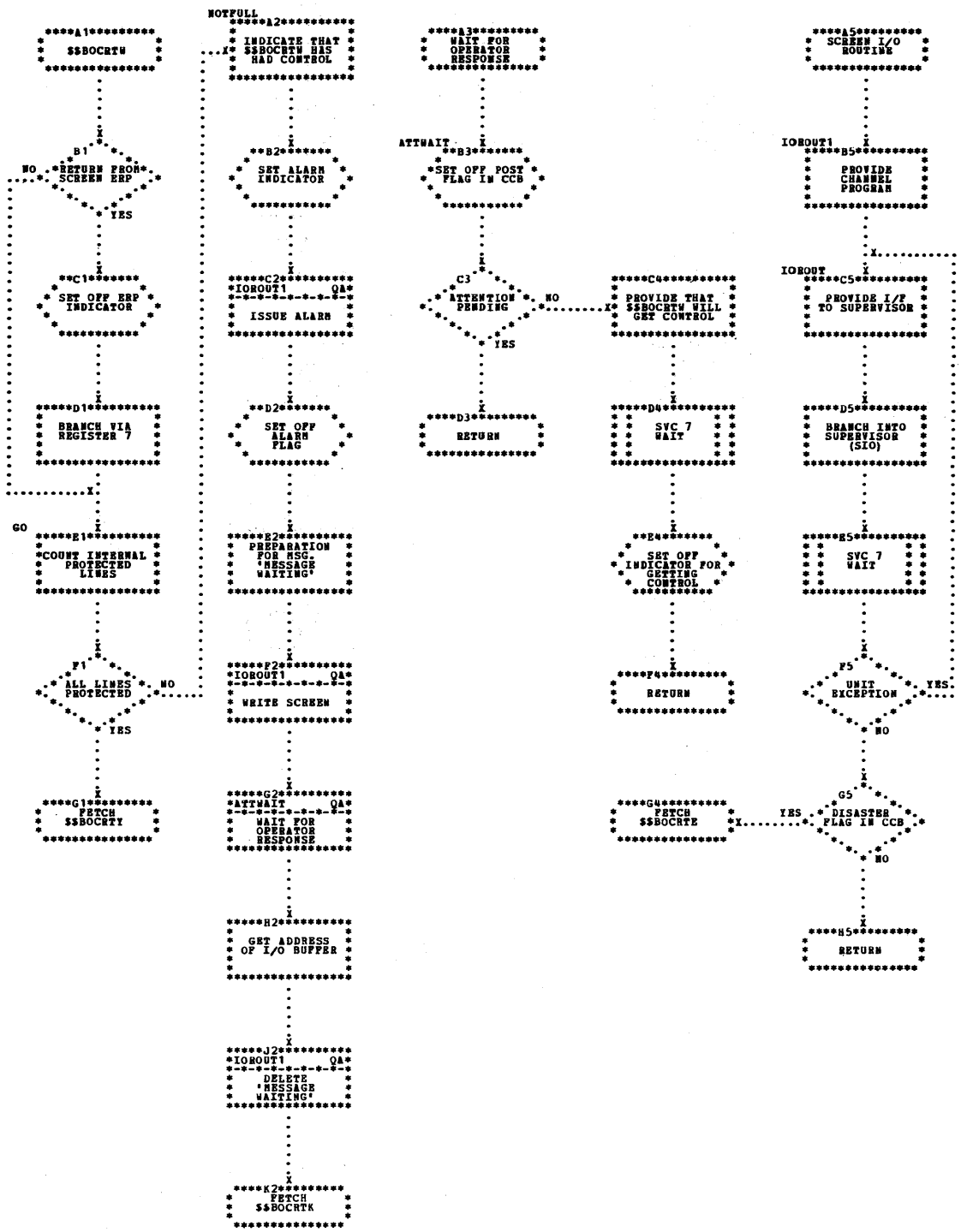


Chart QB. $$BOCRTX - Verify Operator Commands (Part 1 of 2)
(Refer to Chart 20)

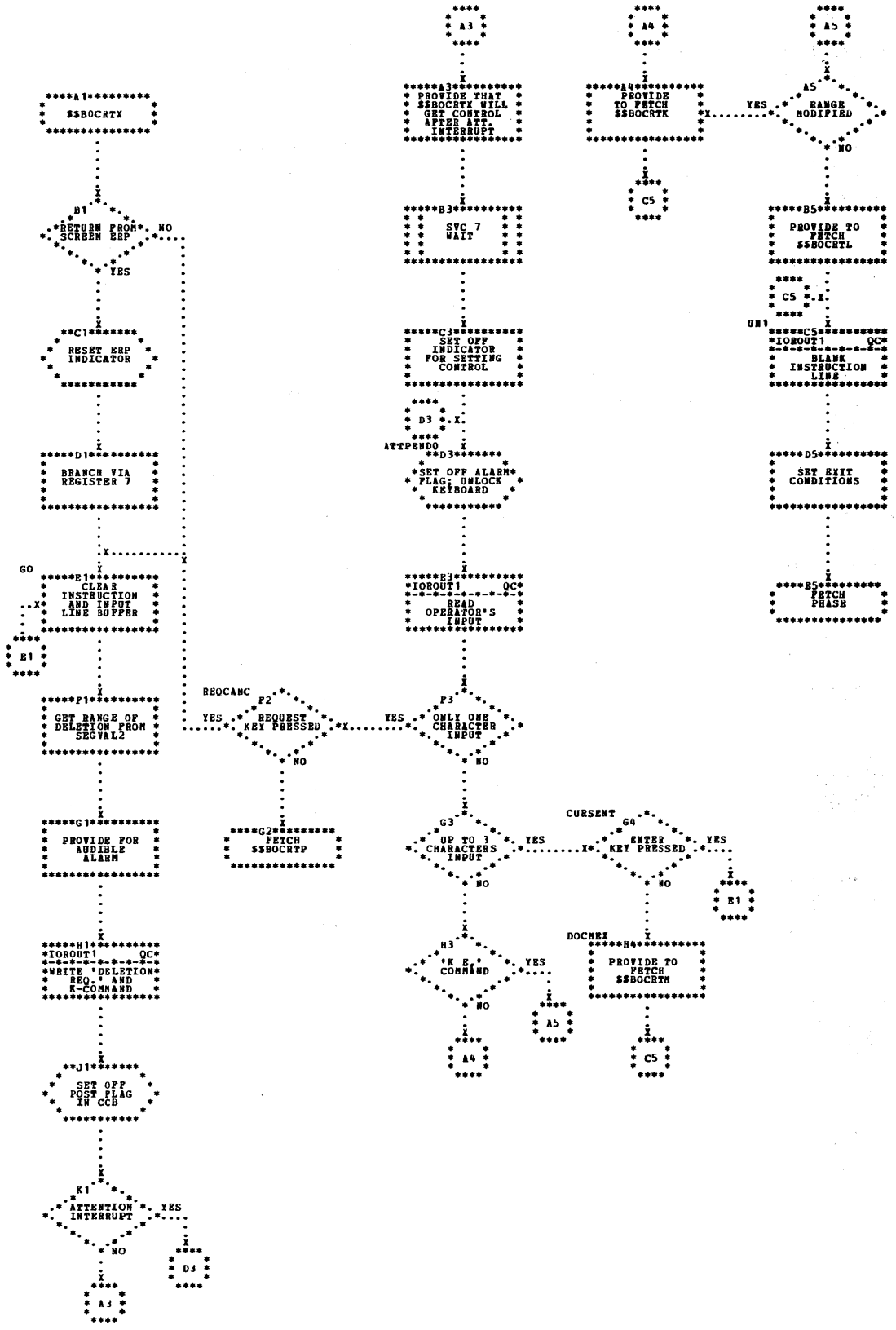


Chart QC. $$BOCRTX - Verify Operator Commands (Part 2 of 2)
(Refer to Chart 20)

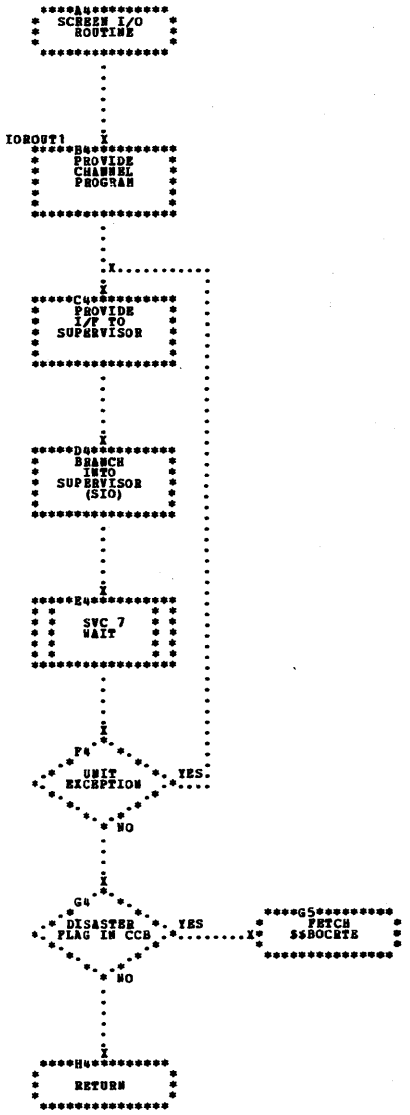


Chart QD. $$BOCRTY - Handle Long Messages (Part 1 of 3)
(Refer to Chart 20)

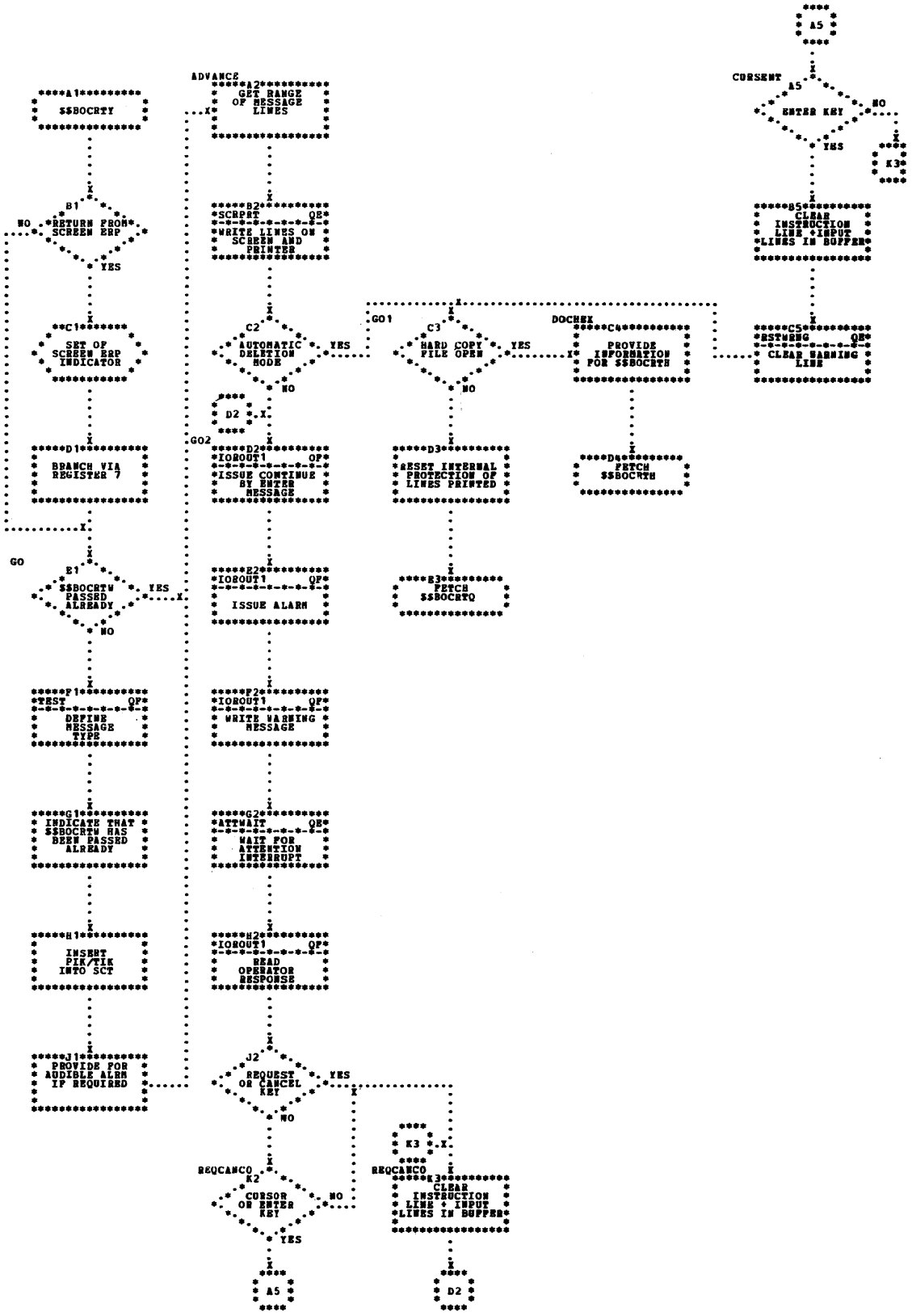


Chart QE. $$BOCRTY - Handle Long Messages (Part 2 of 3)
(Refer to Chart 20)

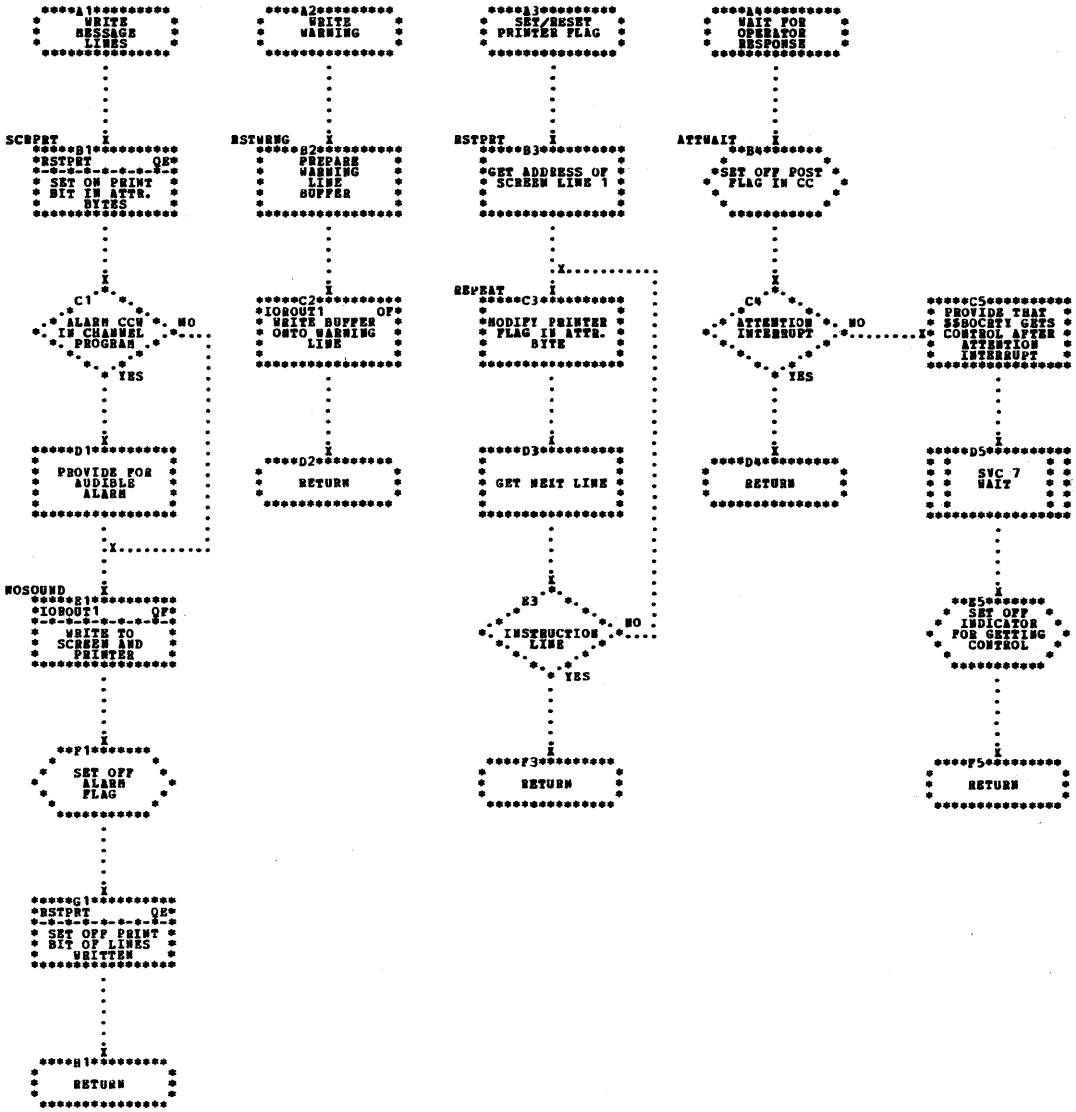


Chart QF. $$BOCRTY - Handle Long Messages (Part 3 of 3)
(Refer to Chart 20)

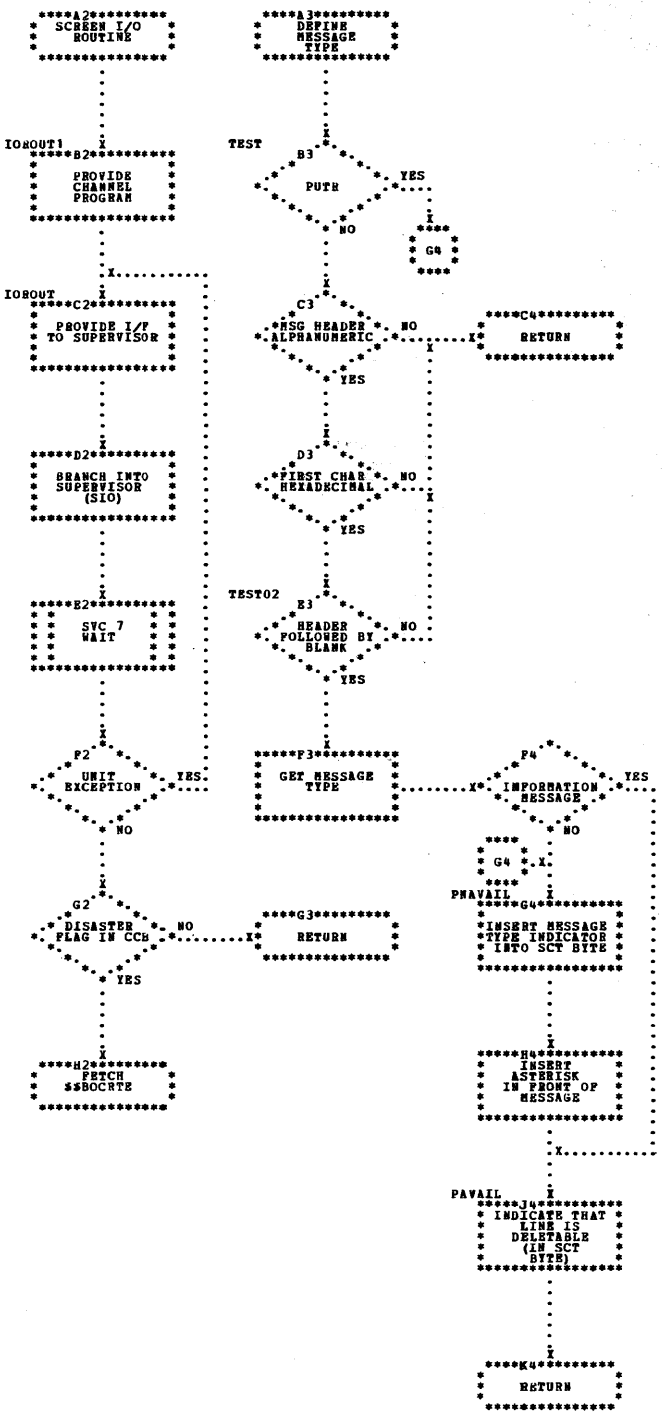


Chart QG. $$BOCRTZ - Channel Program Analyzer (Part 1 of 6)
(Refer to Chart 16)

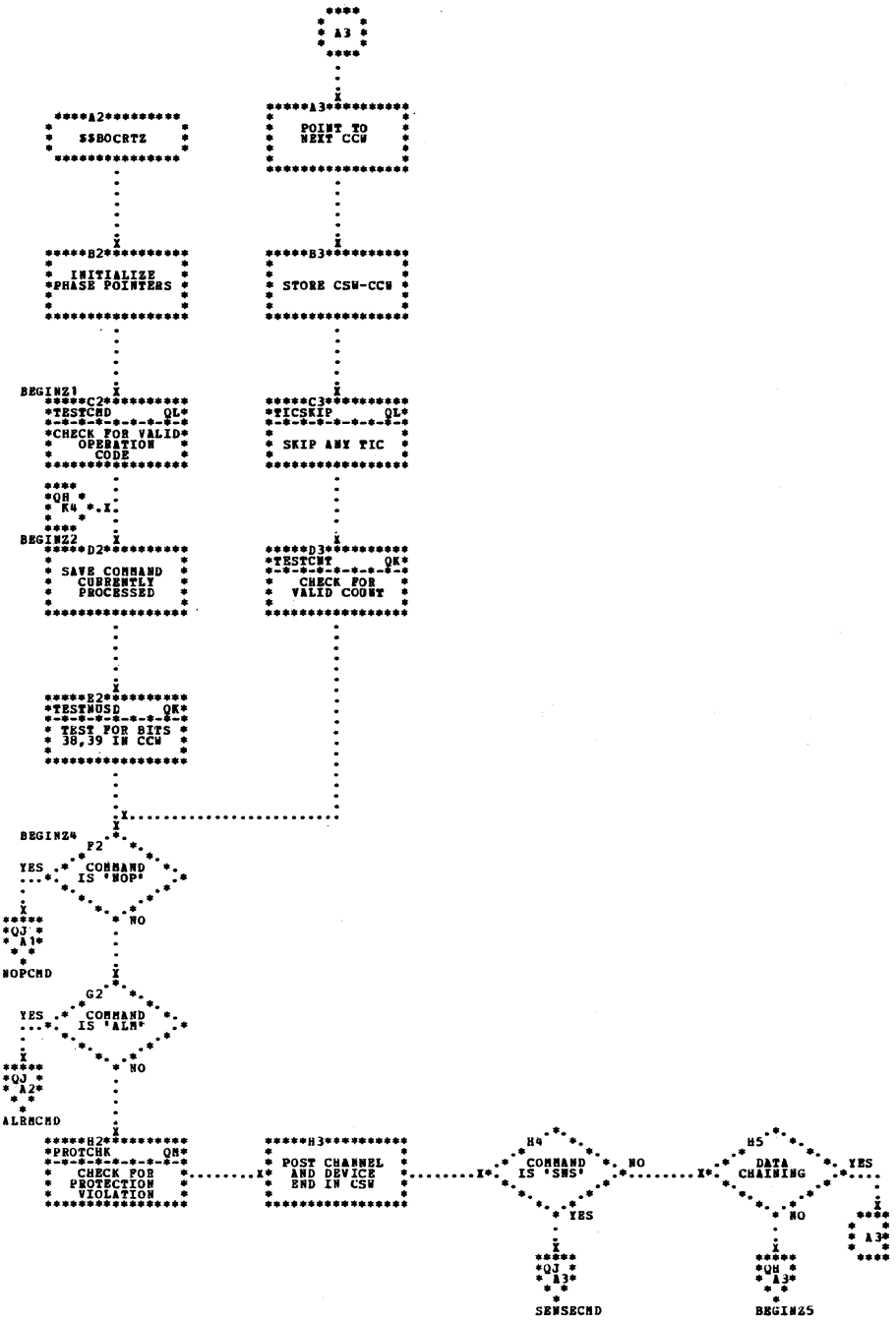


Chart QH. $$BOCRTZ - Channel Program Analyzer (Part 2 of 6)
(Refer to Chart 16)

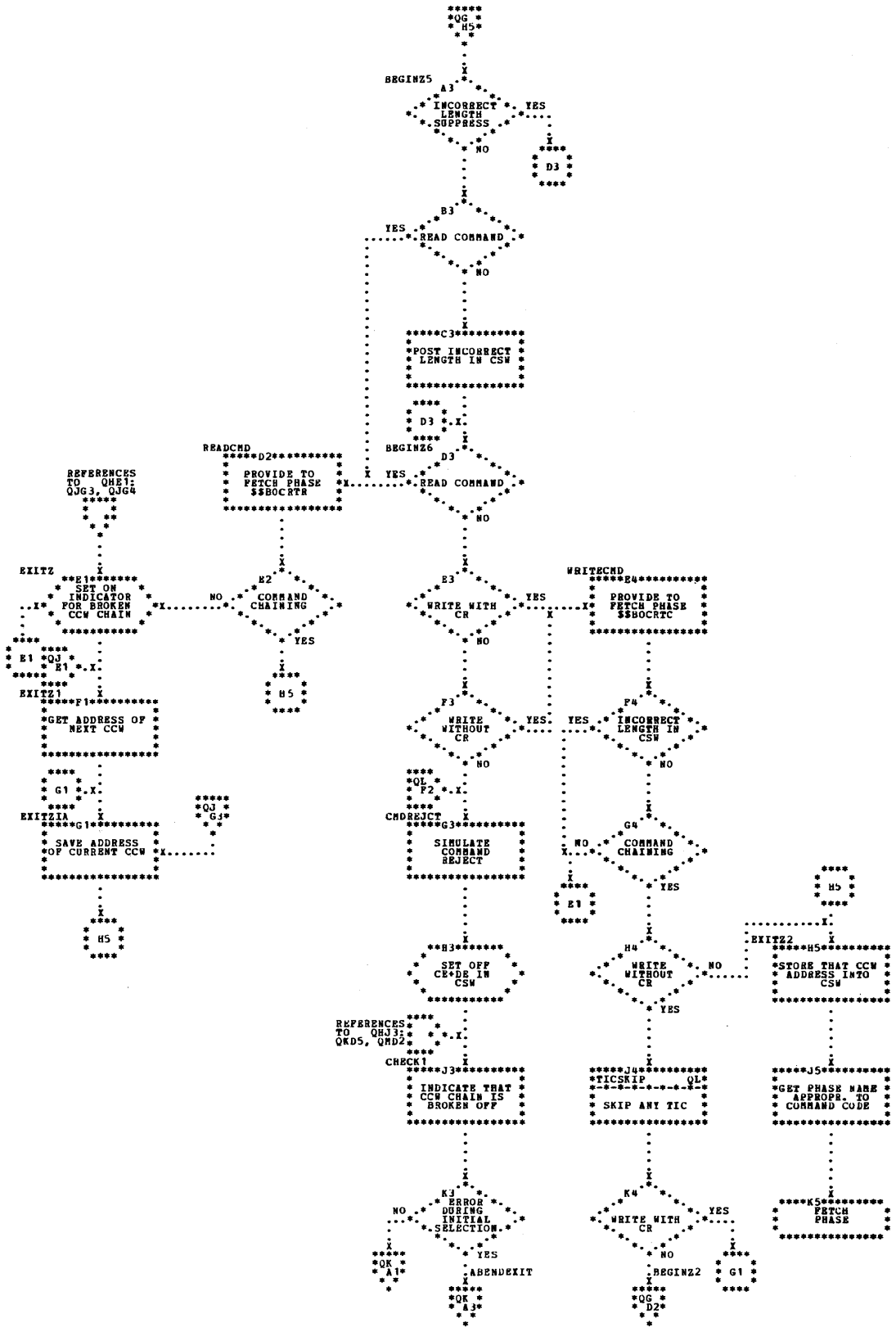


Chart QJ. $$BOCRTZ - Channel Program Analyzer (Part 3 of 6)
(Refer to Chart 16)

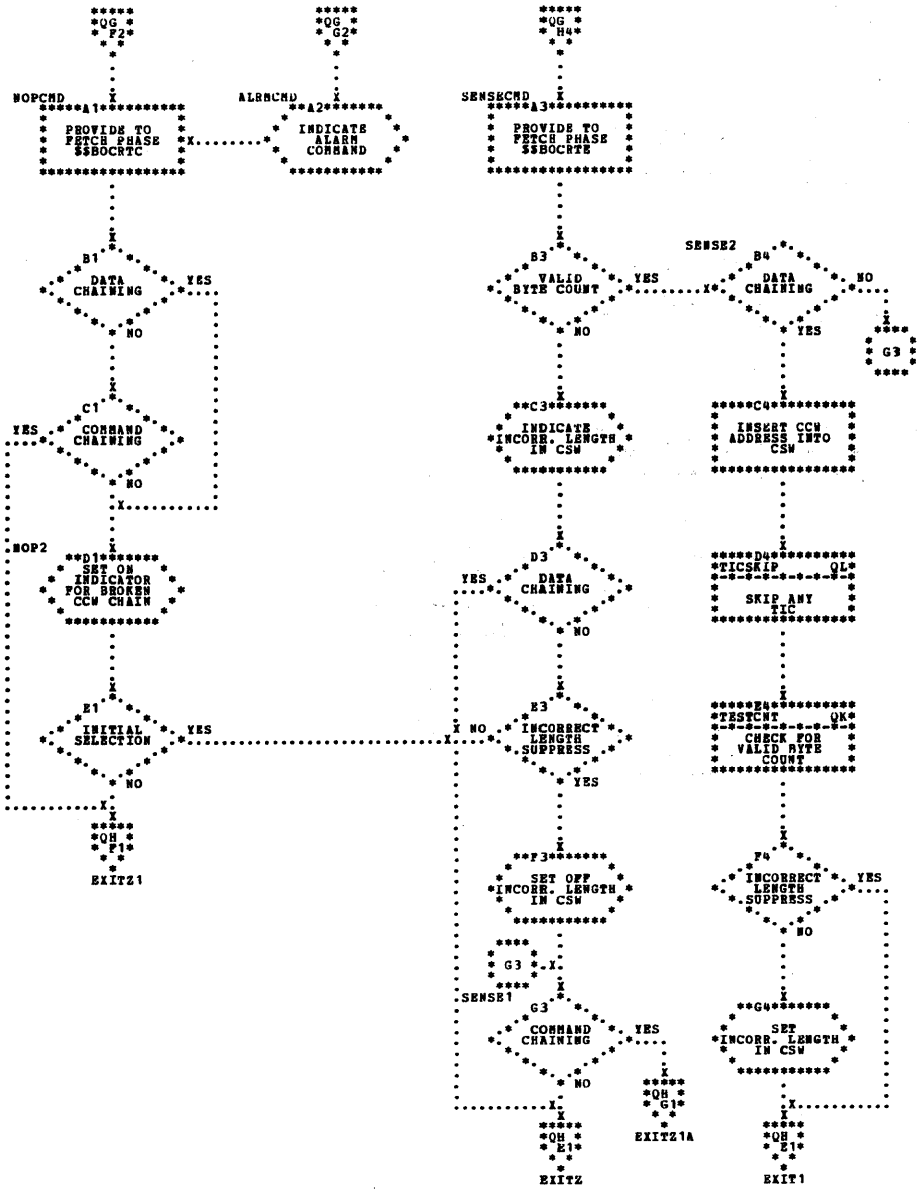


Chart QK. $$BOCRTZ - Channel Program Analyzer (Part 4 of 6)
(Refer to Chart 16)

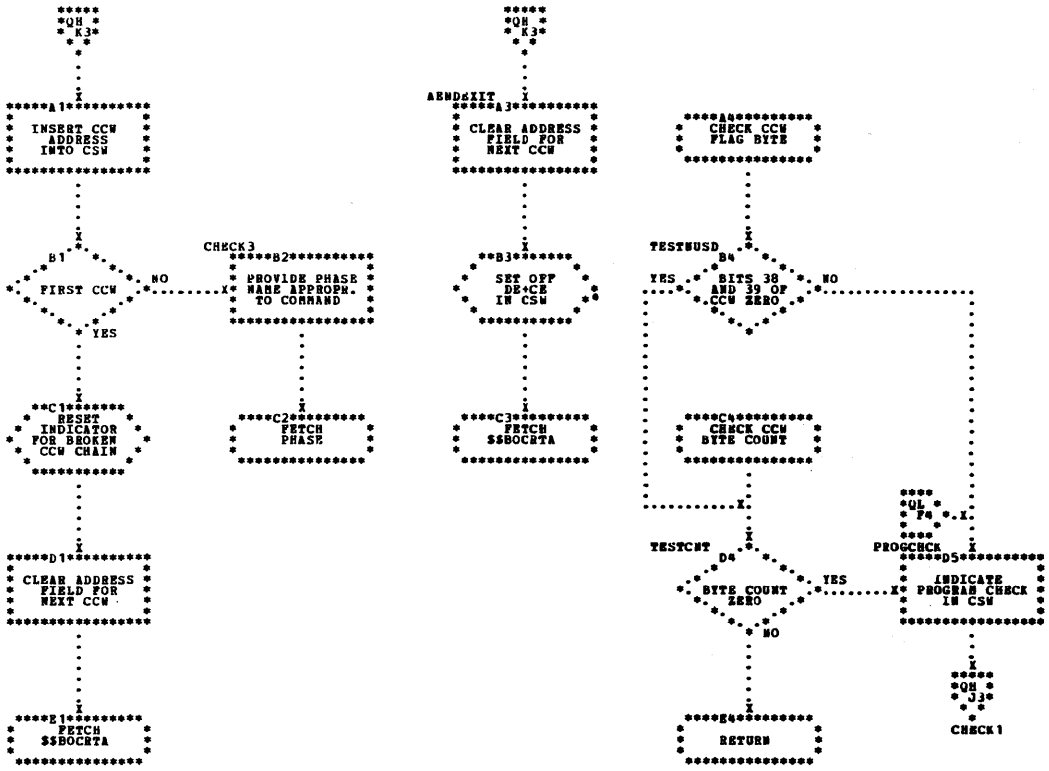


Chart QL. $$BOCRTZ - Channel Program Analyzer (Part 5 of 6)
(Refer to Chart 16)

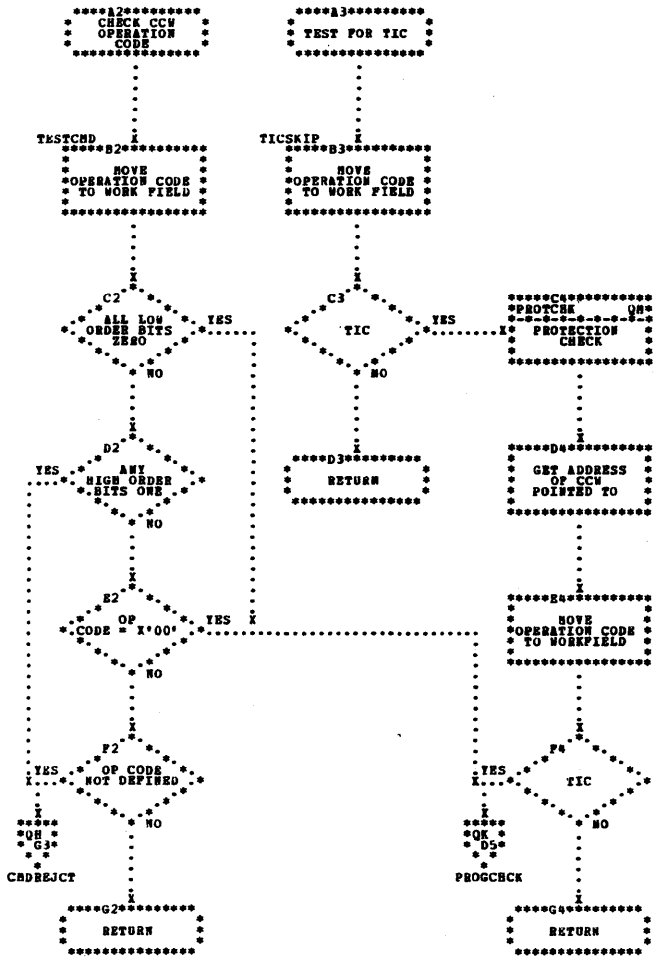


Chart QM. $$BOCRTZ - Channel Program Analyzer (Part 6 of 6)
(Refer to Chart 16)

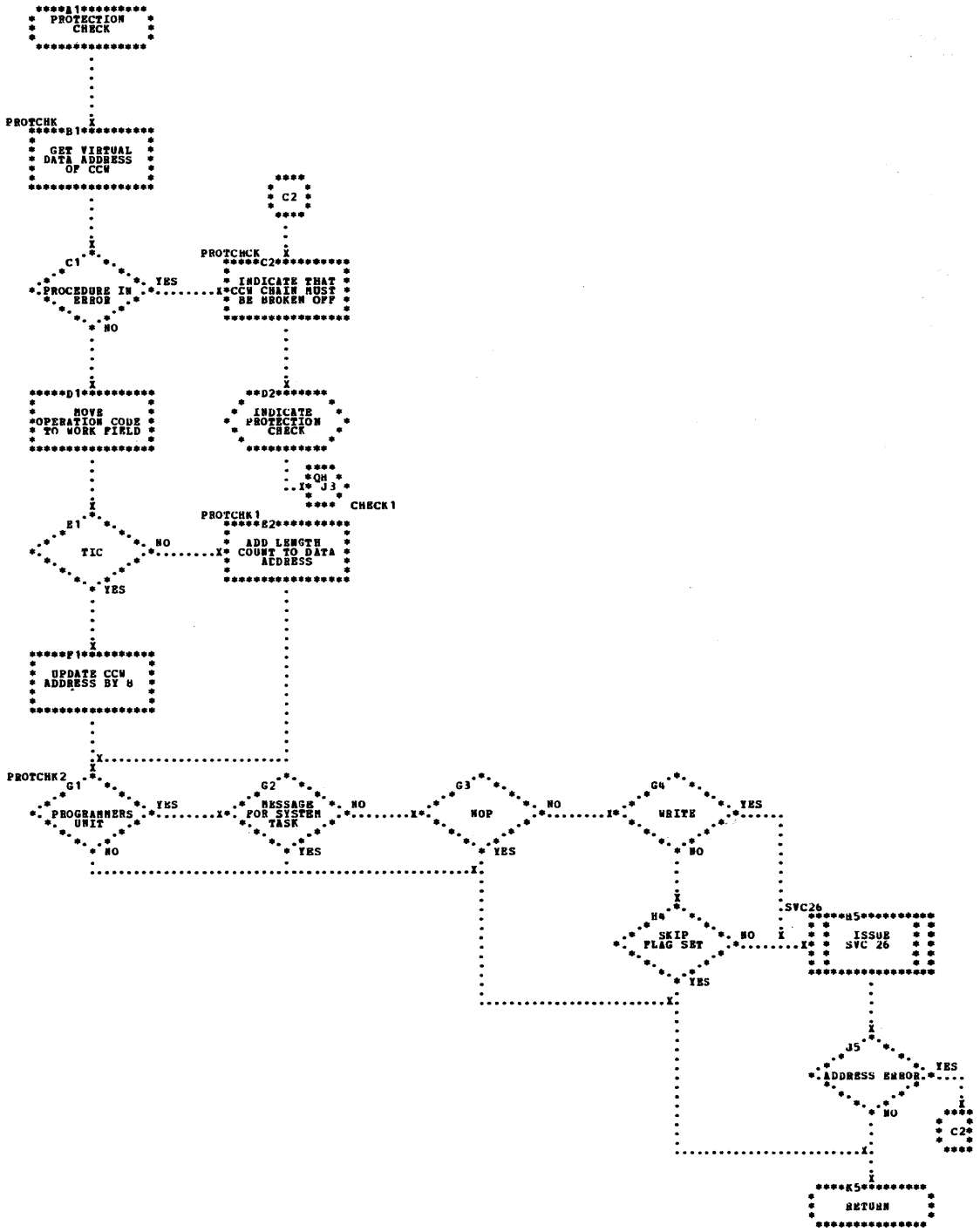


Chart RA. $$BOCRT1 - Message Redisplay Part 1 (Part 1 of 2)
(Refer to Chart 21)

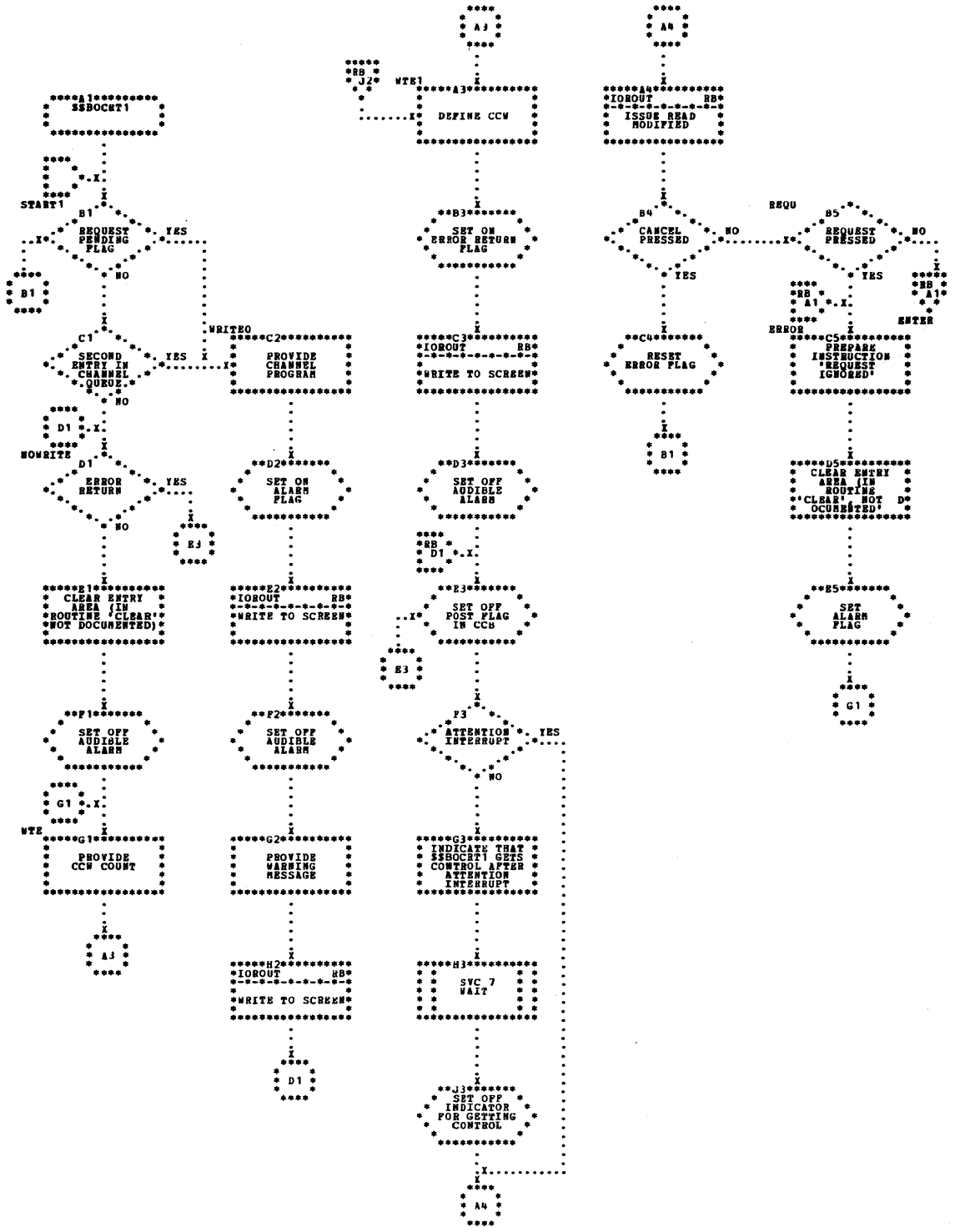


Chart RB. $$BOCRT1 - Message Redisplay Part 1 (Part 2 of 2)
(Refer to Chart 21)

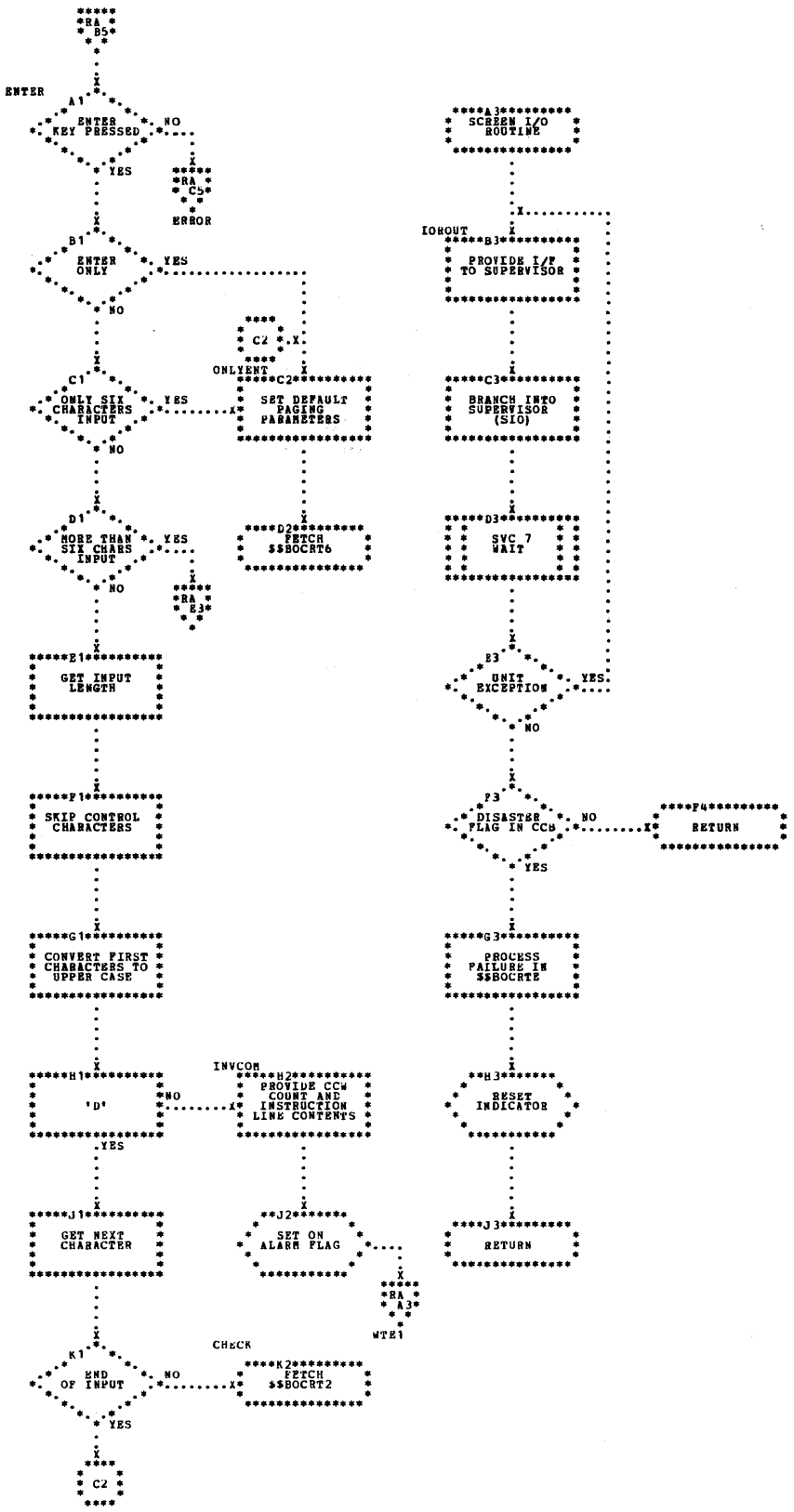


Chart RC. $$BOCRT2 - Message Redisplay Part 2 (Part 1 of 2)
(Refer to Chart 21)

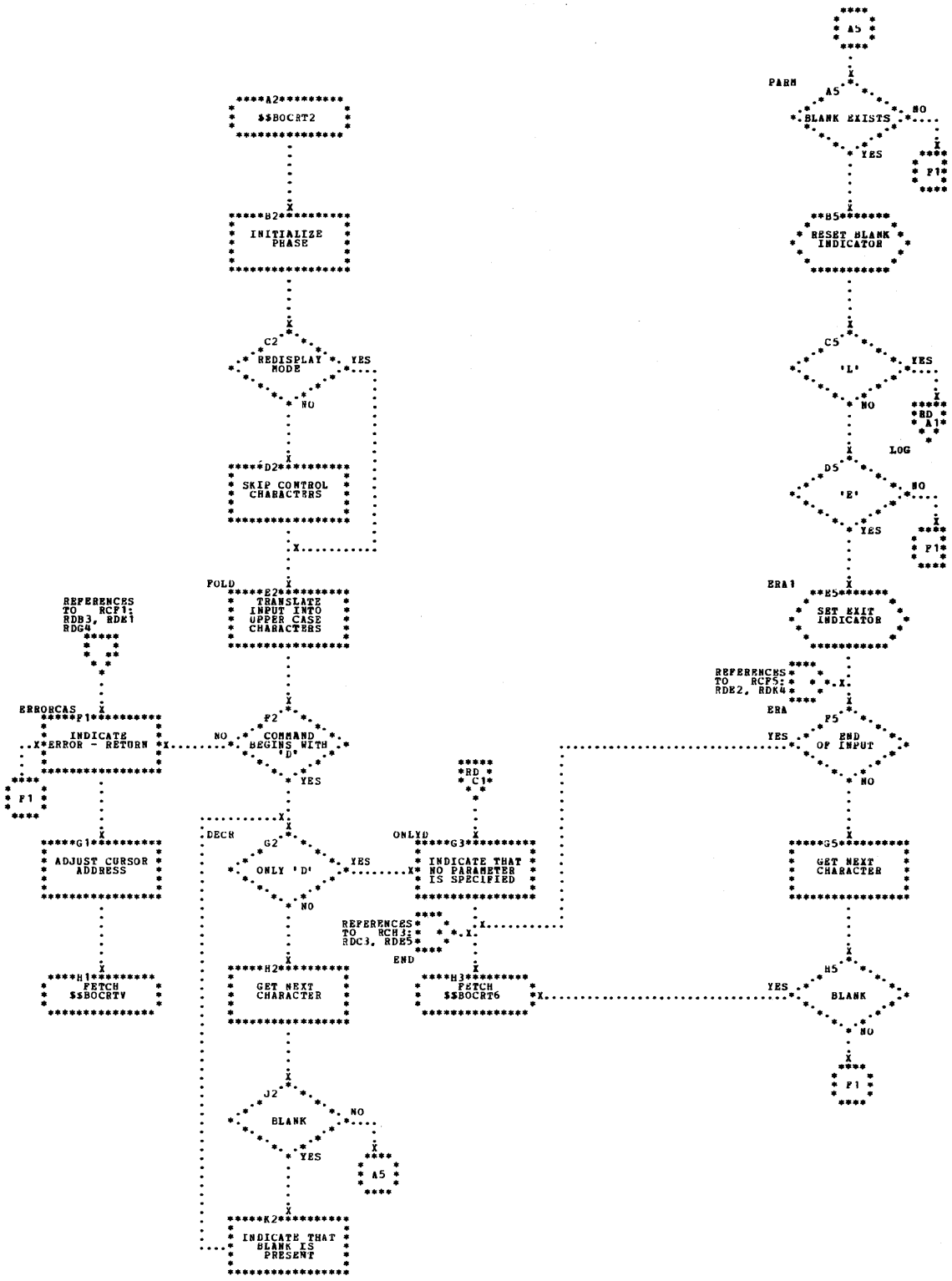


Chart RD. $$BOCRT2 - Message Redisplay Part 2 (Part 2 of 2)
(Refer to Chart 21)

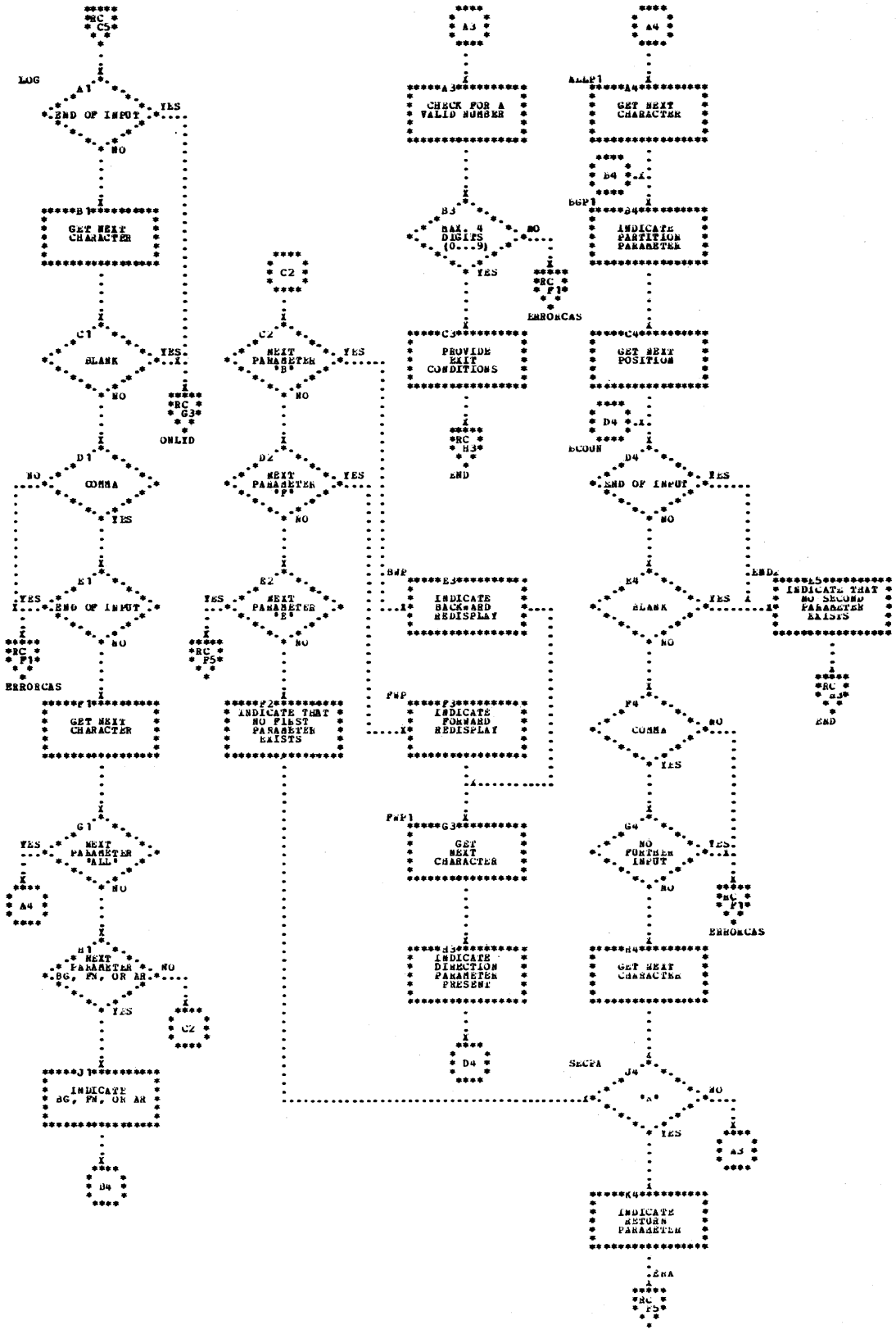


Chart RE. $$BOCRT3 - Message Redisplay Part 3 (Part 1 of 3)
(Refer to Chart 21)

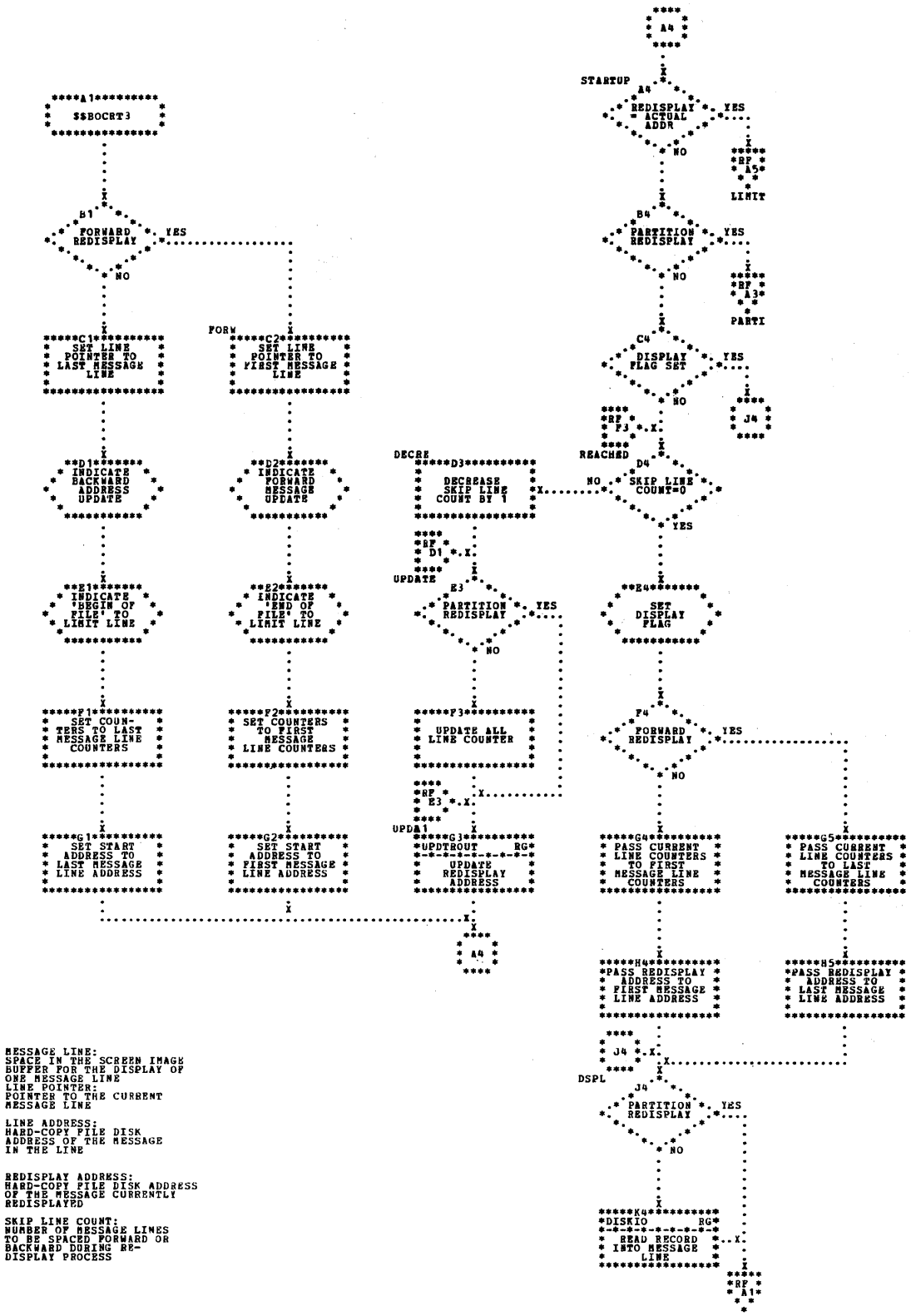


Chart RF. $$BOCRT3 - Message Redisplay Part 3 (Part 2 of 3)
(Refer to Chart 21)

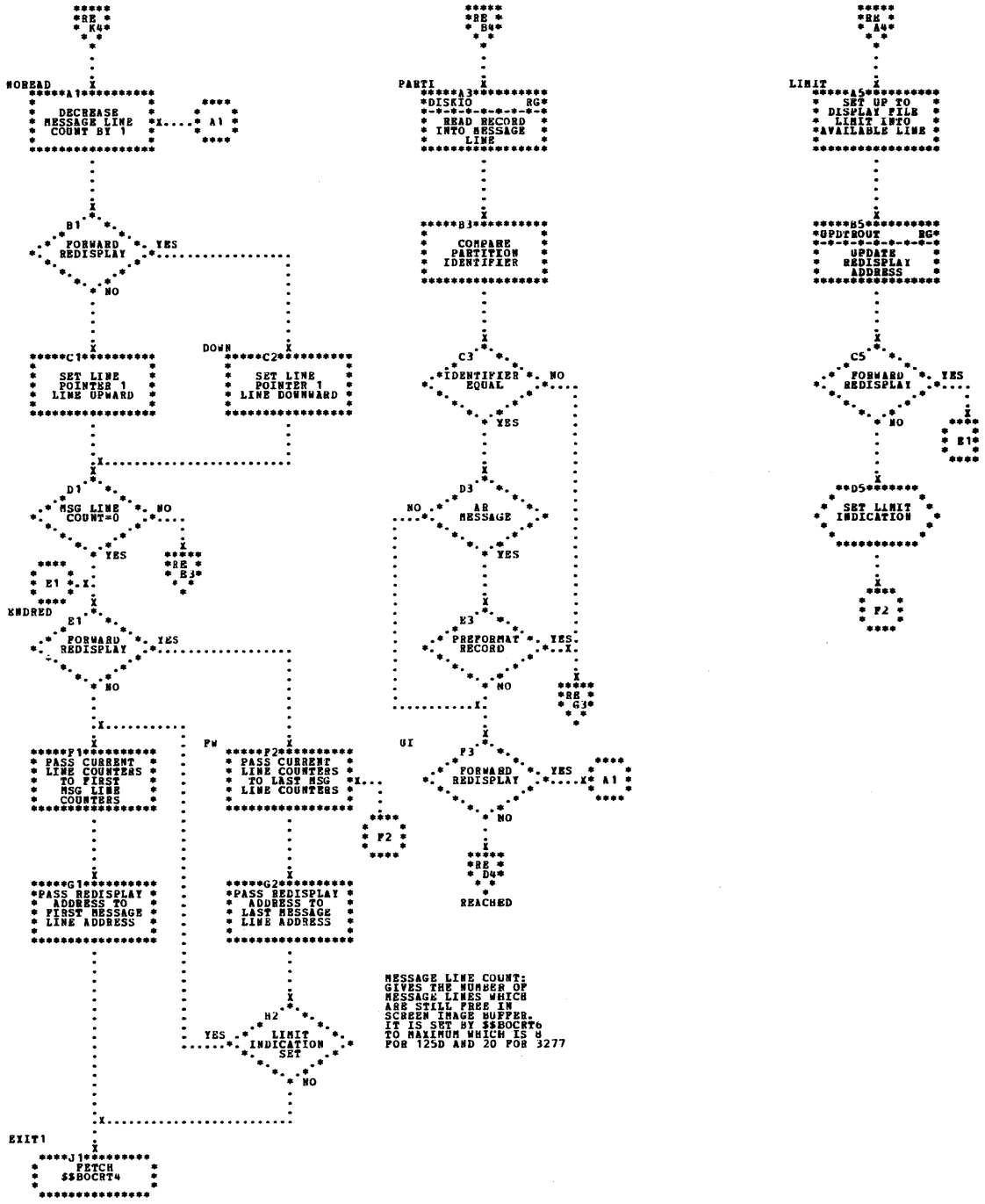


Chart RG. $$BOCRT3 - Message Redisplay Part 3 (Part 3 of 3)
(Refer to Chart 21)

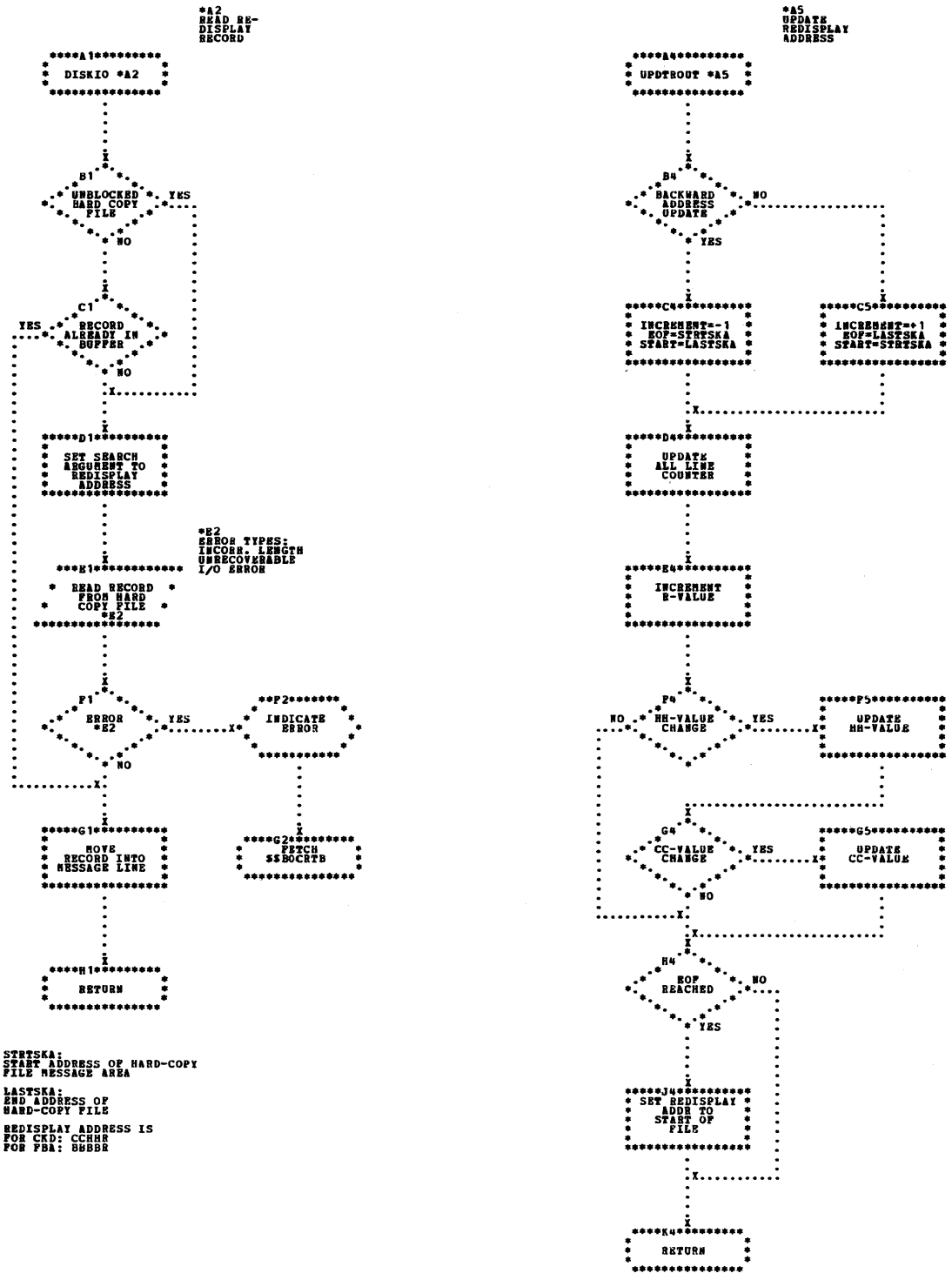


Chart RH. $$BOCRT4 - Message Redisplay Part 4
(Refer to Chart 21)

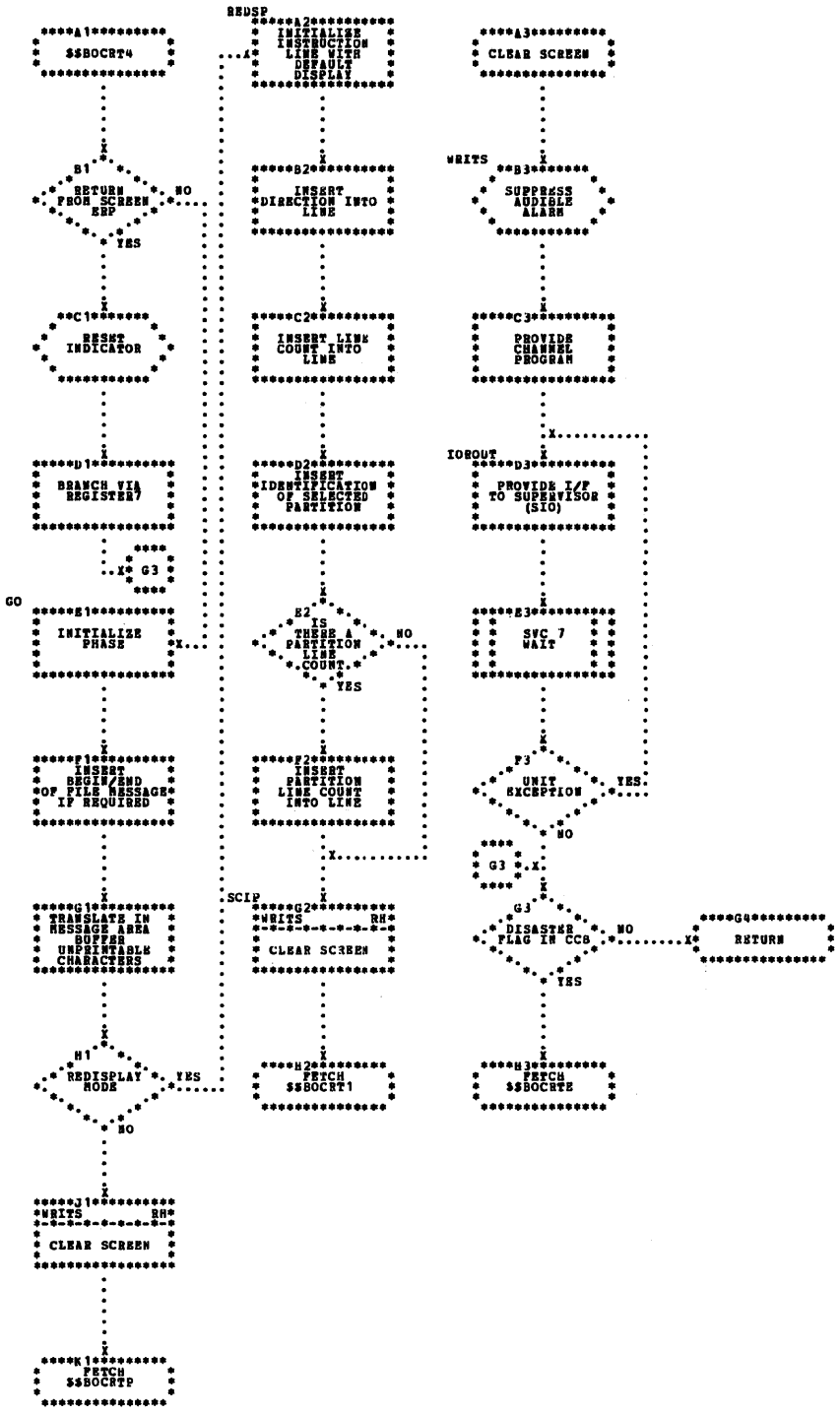
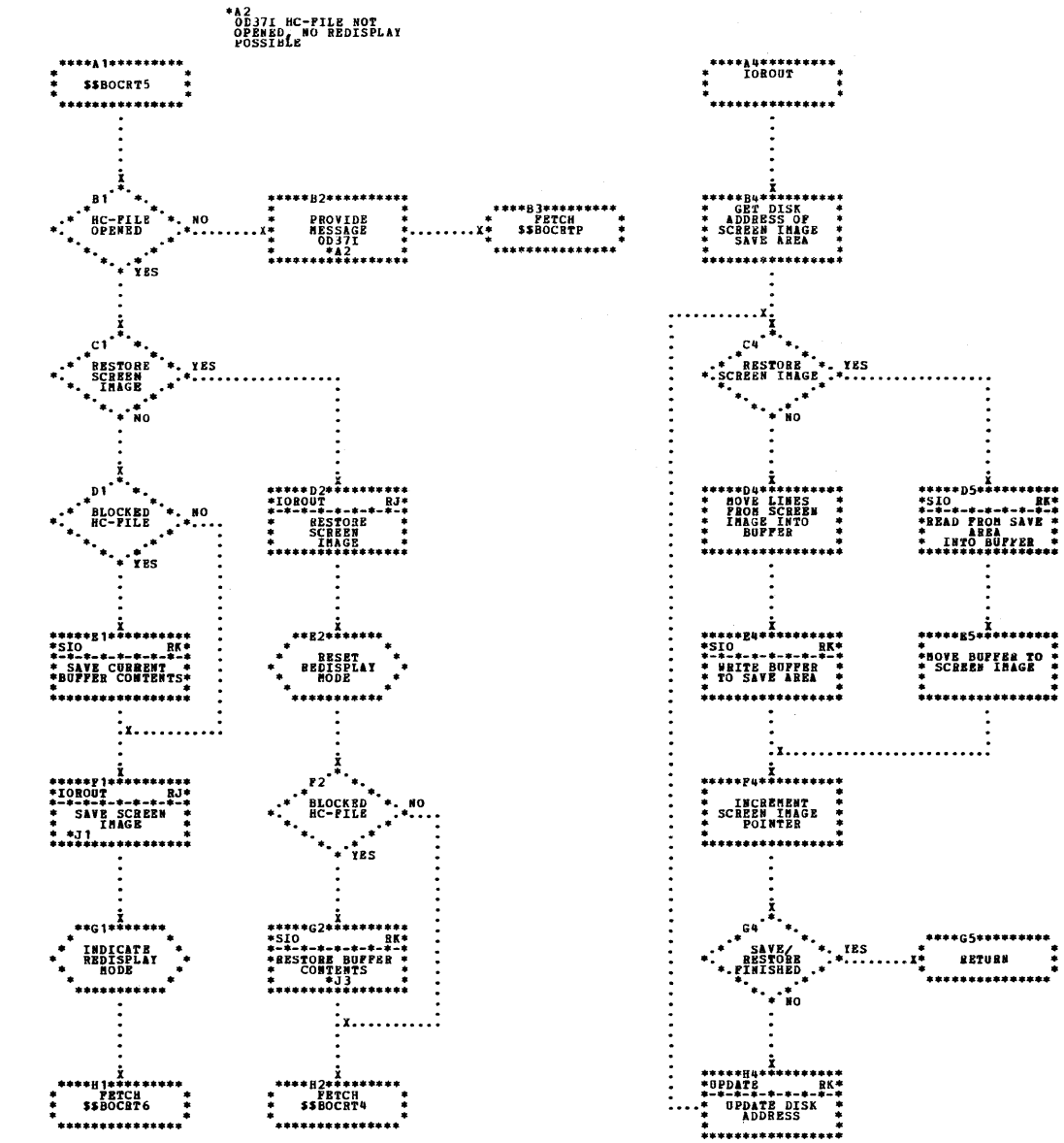


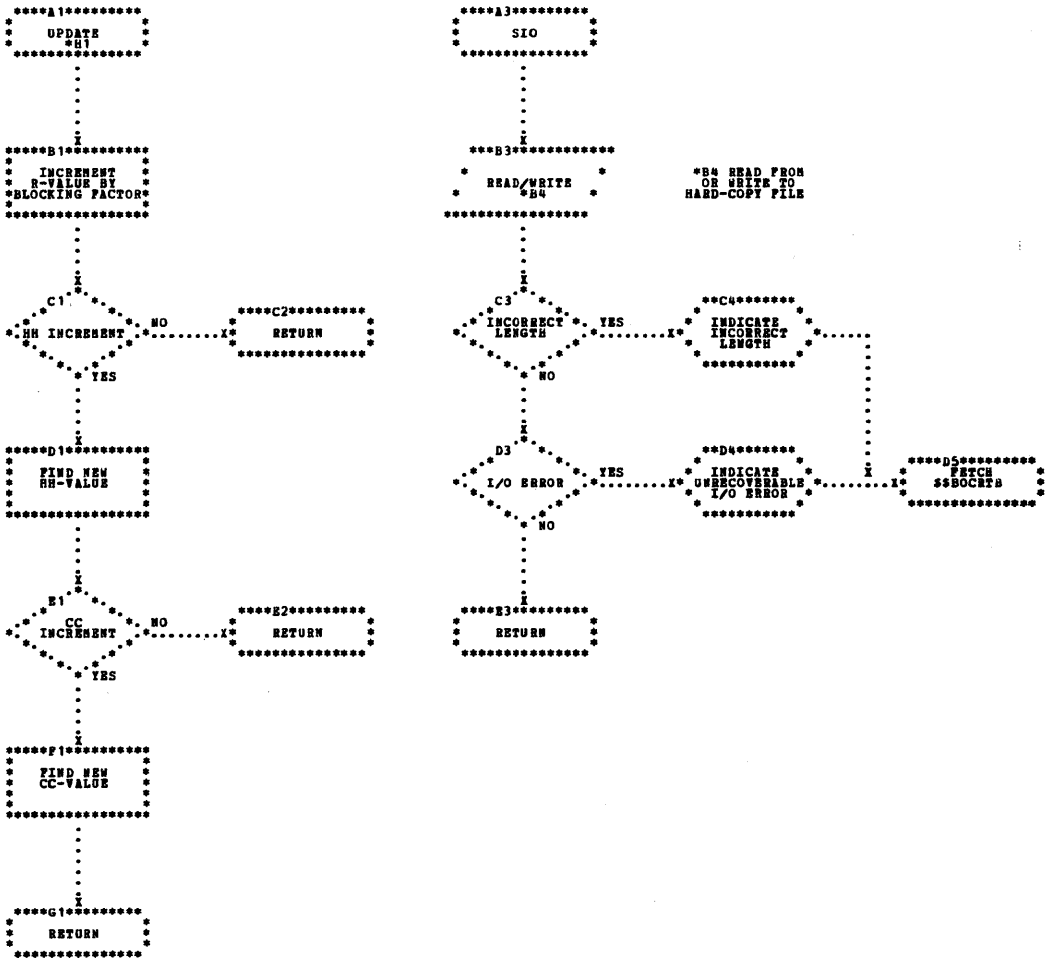
Chart RJ. $$BOCRT5 - Message Redisplay Part 5 (Part 1 of 2)
(Refer to Chart 21)



*J1
SAVE SCREEN IMAGE:
THE CONTENTS OF SCREEN IMAGE BUFFER ARE SAVED IN THE SCREEN IMAGE SAVE AREA. IT IS PART OF THE HARD-COPY FILE. IT STARTS AT RECORD NUMBER 2. ITS LENGTH DEPENDS ON THE SIZE OF THE SCREEN (8 RECORDS FOR 125D; 20 RECORDS FOR 3277).

*J3
RESTORE SCREEN IMAGE:
THE ORIGINAL CONTENTS OF SCREEN IMAGE BUFFER ARE RESTORED FROM SCREEN IMAGE SAVE AREA IN HARD-COPY FILE.

CHart RK. $$BOCRT5 - Message Redisplay Part 5 (Part 2 of 2)
(Refer to Chart 21)



*H1 FOR CKD DEVICES:
THE DISK ADDRESS CCHHR IS
INCREMENTED BY 1 (BLOCKING
FACTOR = 1)

FOR FBA DEVICES:
THE RELATIVE BLOCKNUMBER
EXTENDED BY A ONE BYTE
RECORD VALUE (BBBBR) IS
TREATED AS A CCHHR ADDRESS

Chart RL. $$BOCRT6 - Message Redisplay Part 6 (Part 1 of 2)
(Refer to Chart 21)

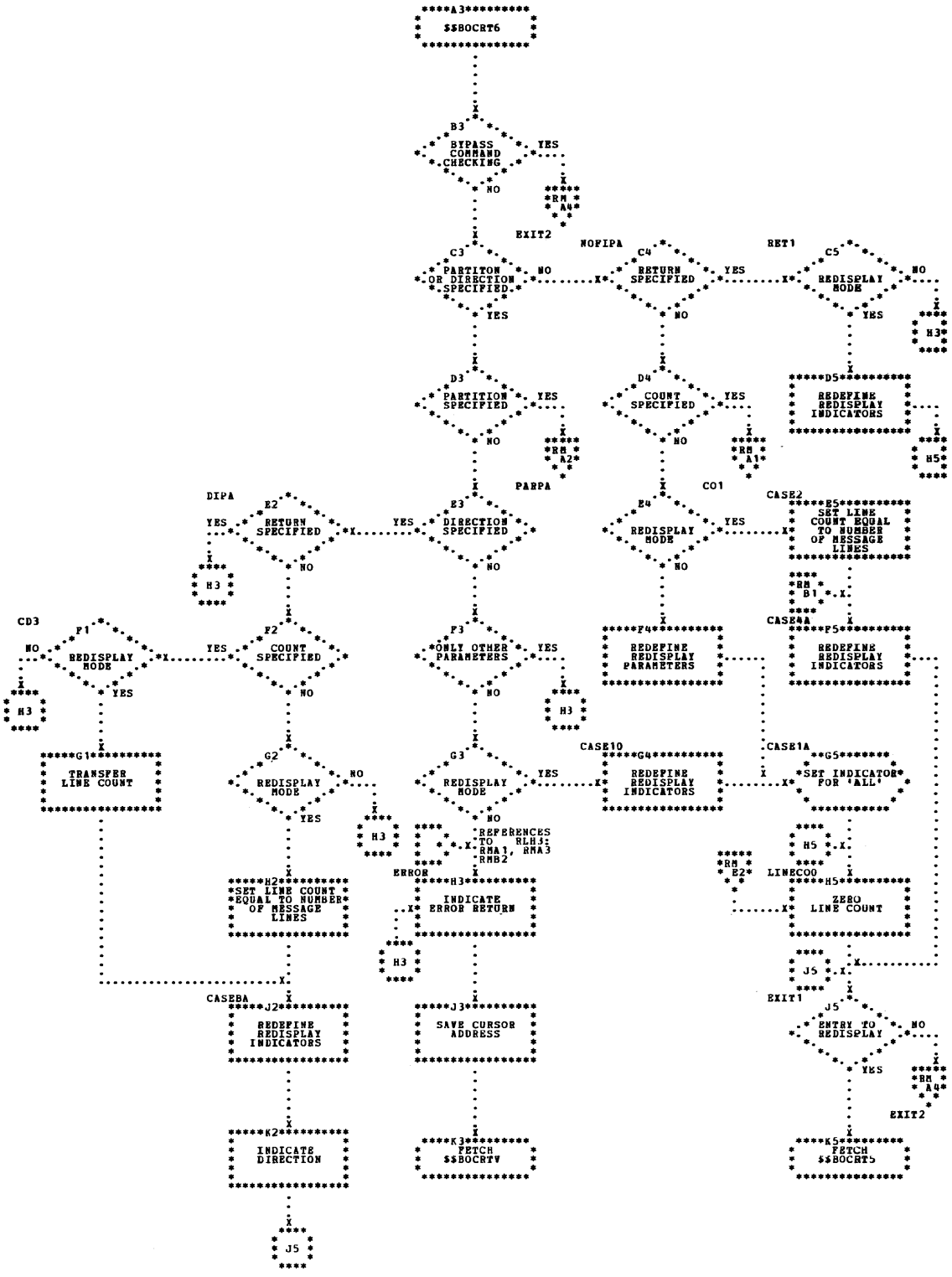


Chart RM. $$BOCRT6 - Message Redisplay Part 6 (Part 2 of 2)
(Refer to Chart 21)

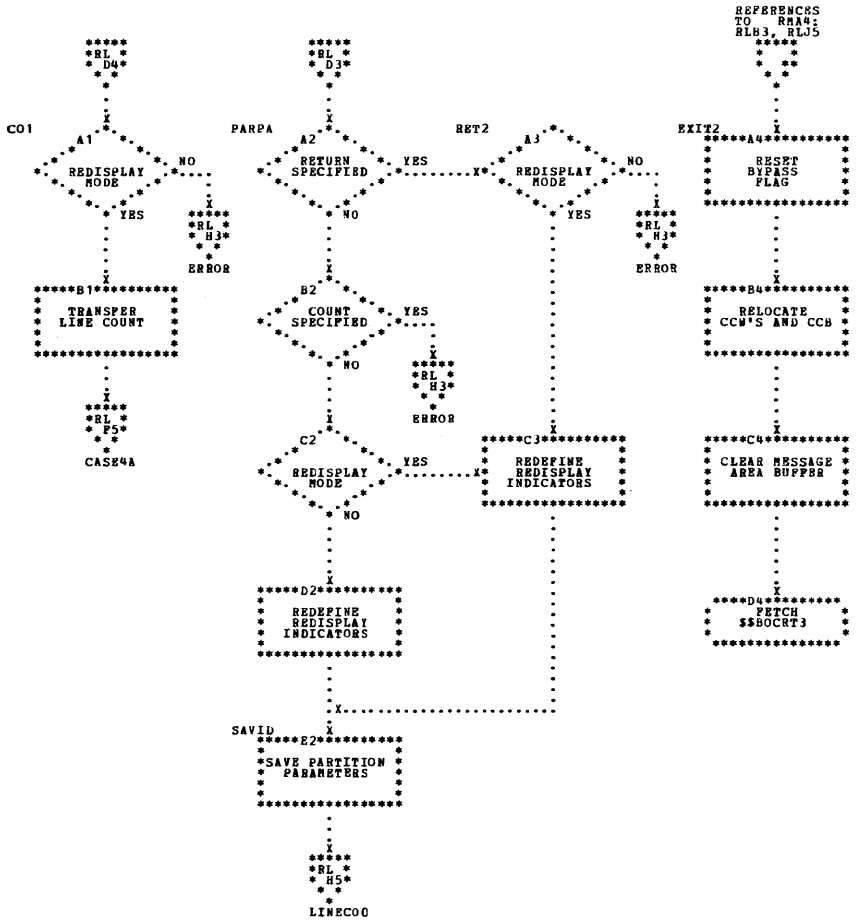


Chart SA. $$BCHKPT - Test Checkpoint Conditions
(Refer to Chart 23)

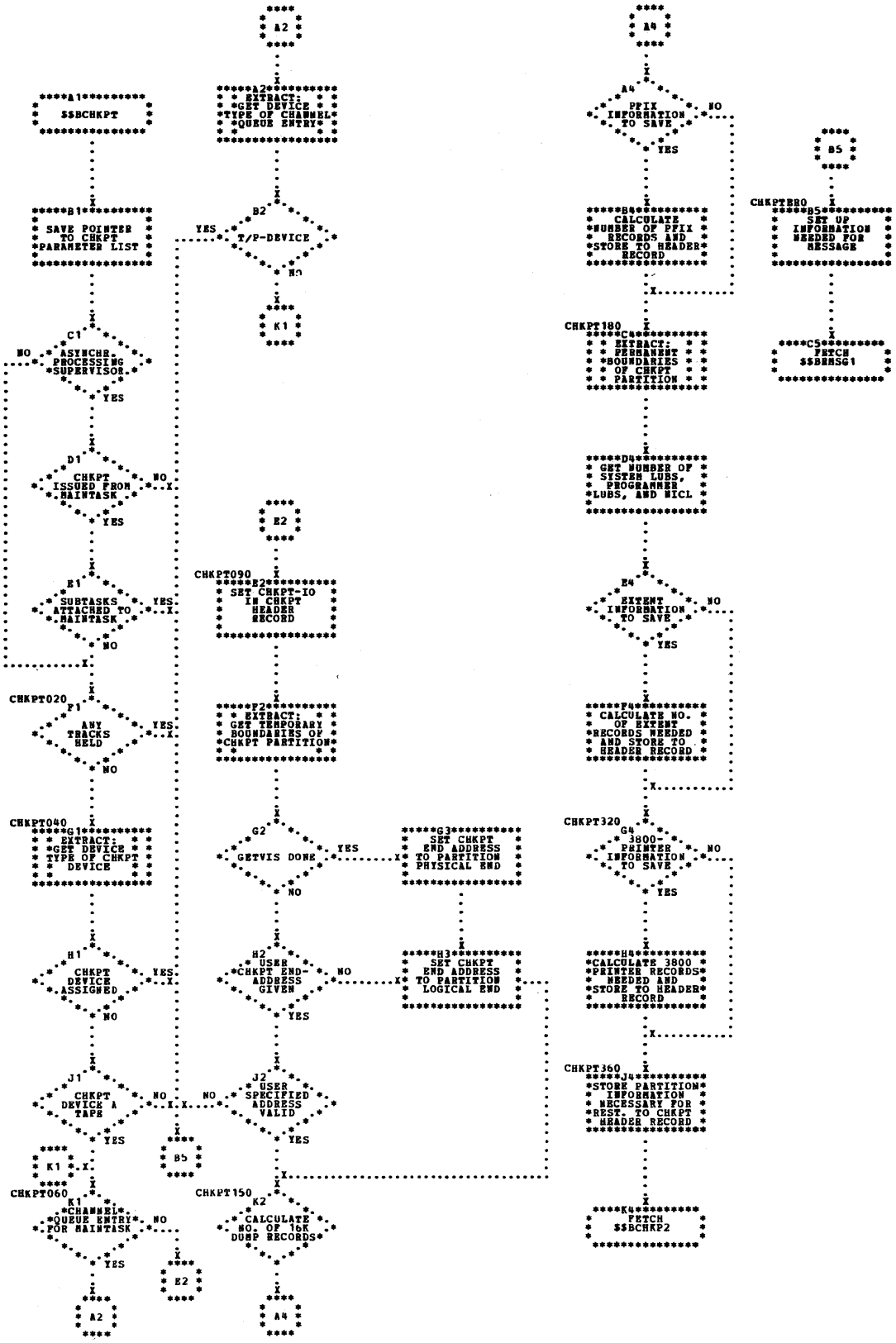


Chart SB. $$BCHKP2 - Build and Write Checkpoint Records
(Refer to Chart 23)

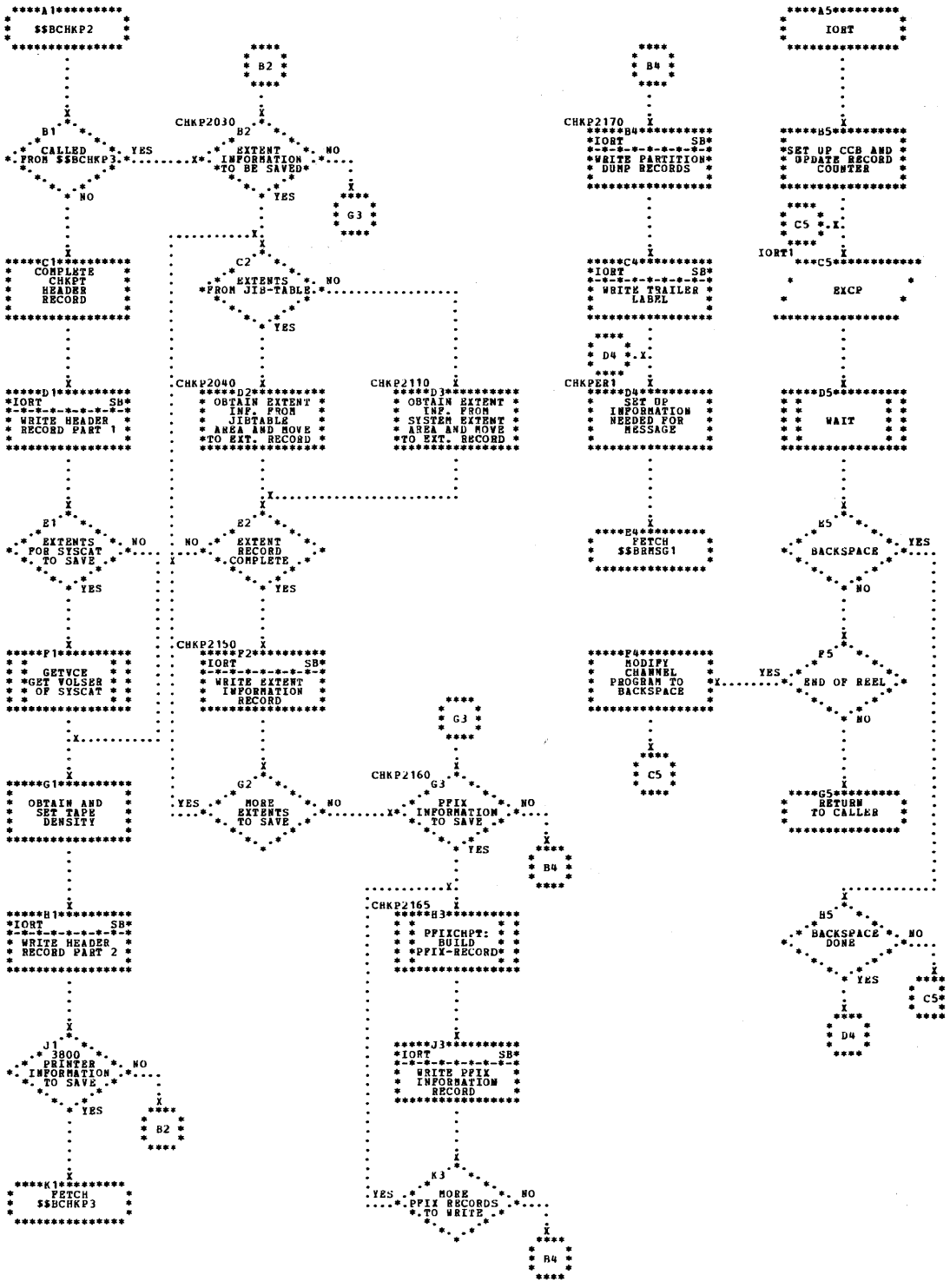


Chart SC. $$BCHKP3 - Build and Write 3800 Checkpoint Records
(Refer to Chart 23)

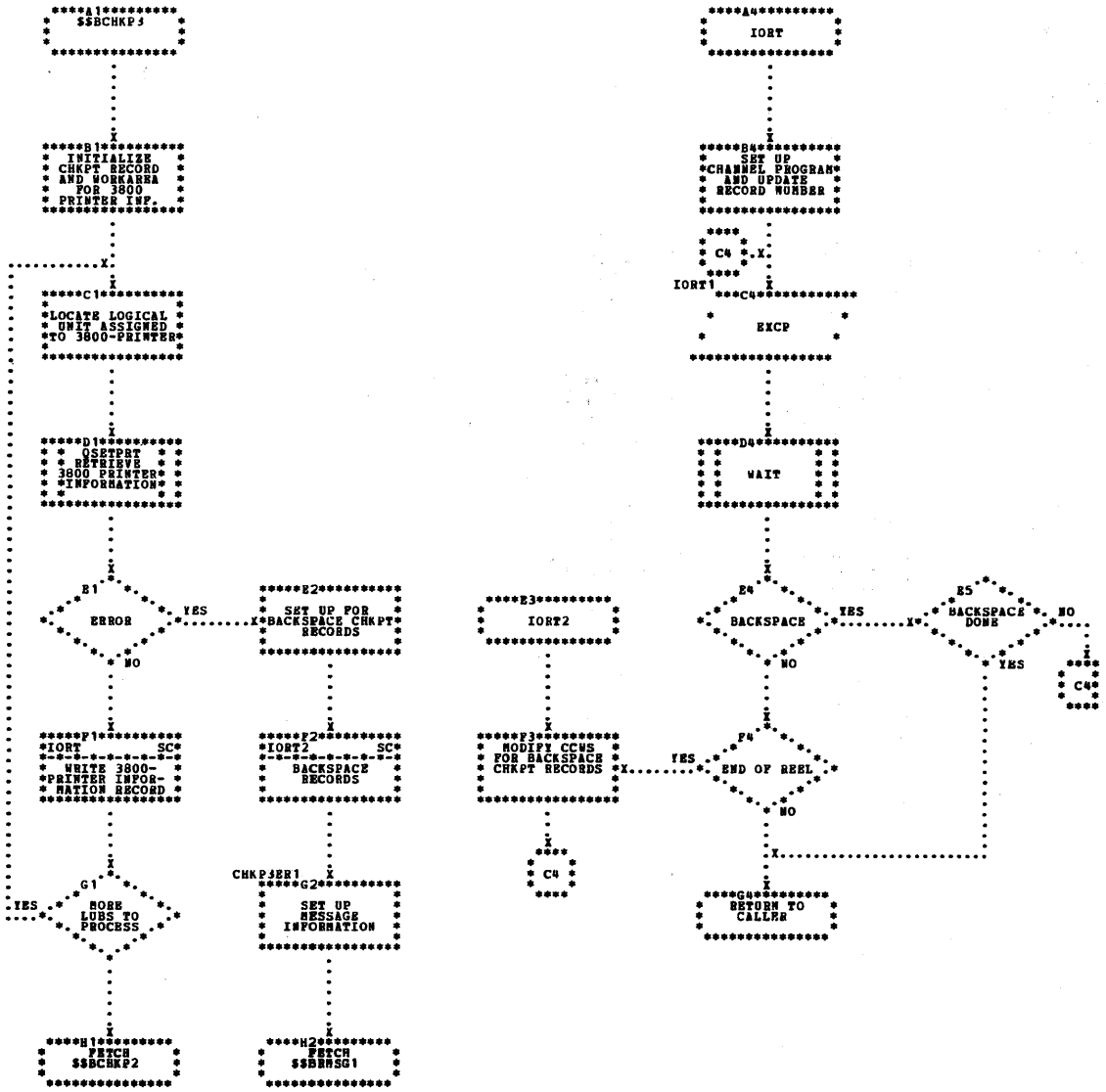


Chart SD. $$BCHKPD - Test Checkpoint Conditions
(Refer to Chart 23)

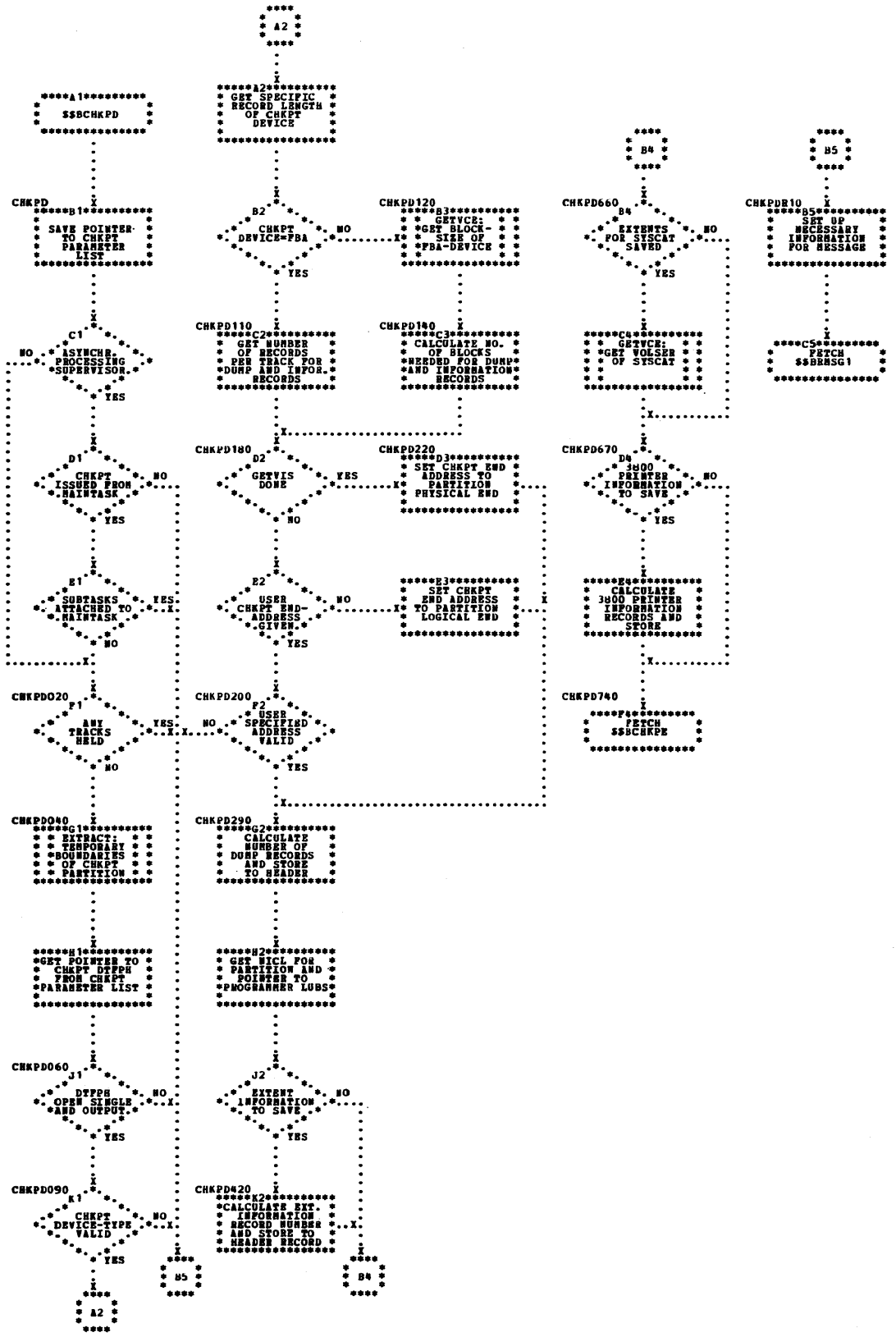


Chart SE. $$BCHKPE - Complete Test and Write Checkpoint Header
(Refer to Chart 23)

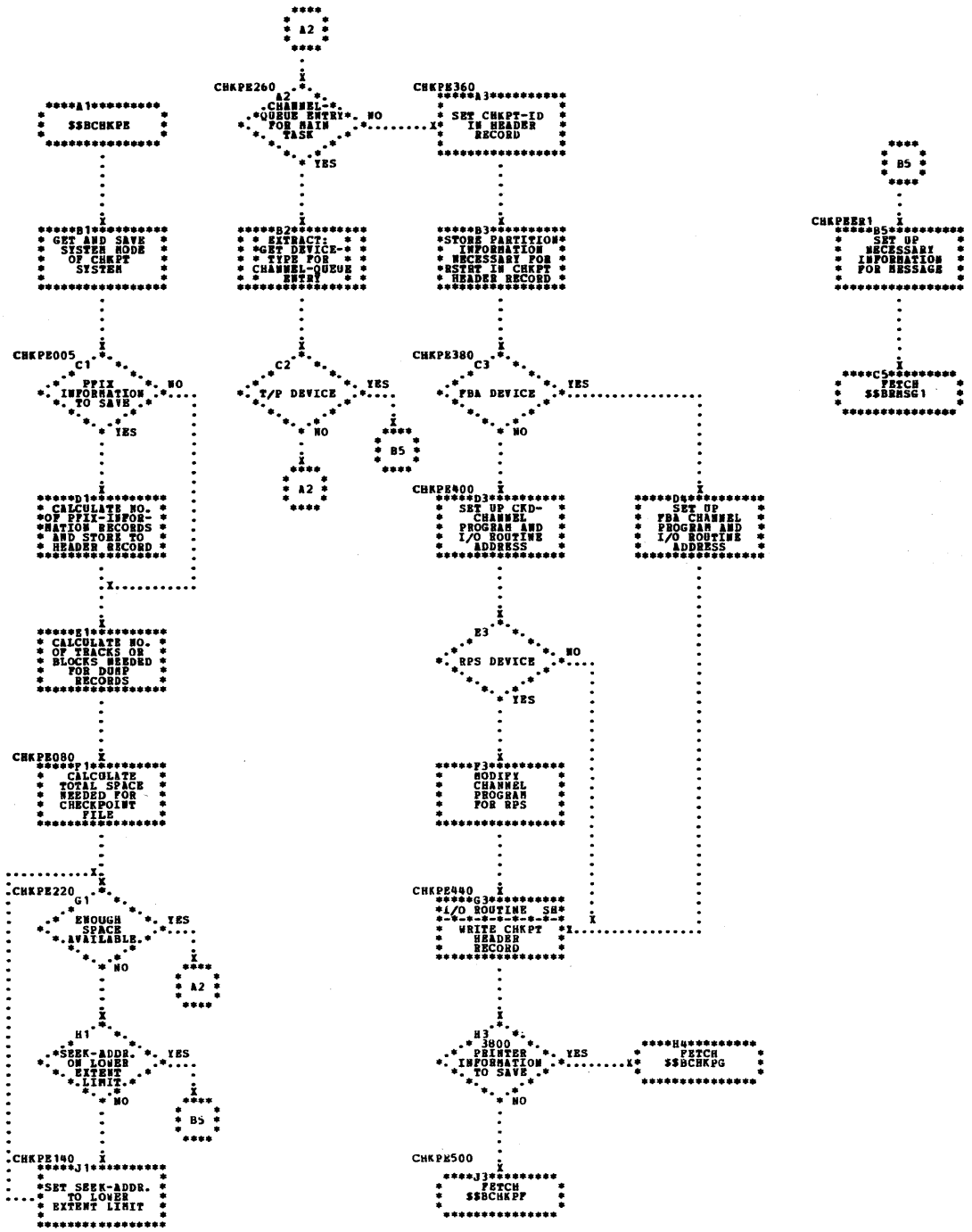


Chart SF. $$BCHKPF - Write Checkpoint Records
(Refer to Chart 23)

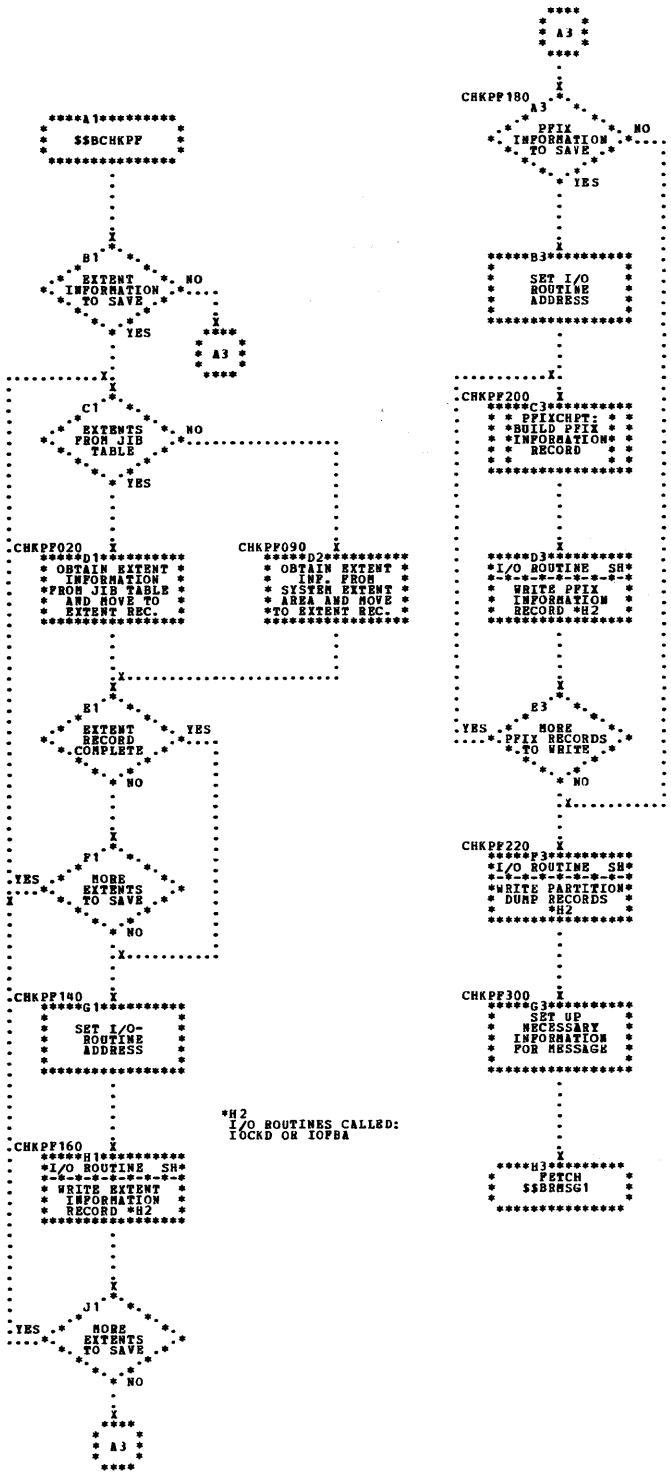


Chart SG. $$BCHKPG - Write 3800 Printer Information Records
(Refer to Chart 23)

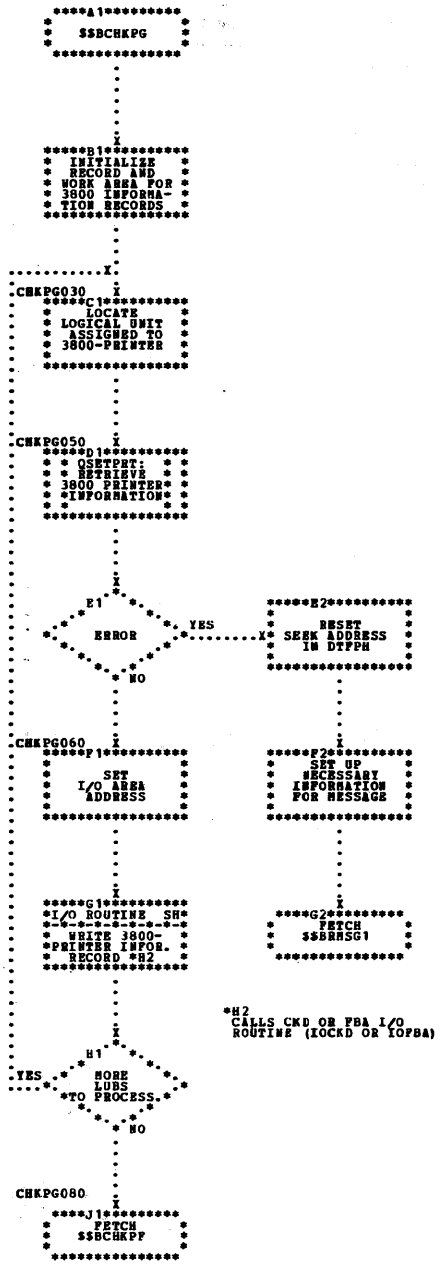


Chart SH. $$BCHKPG - I/O Subroutines for Writing Checkpoint Records on Disk (Refer to Chart 23)

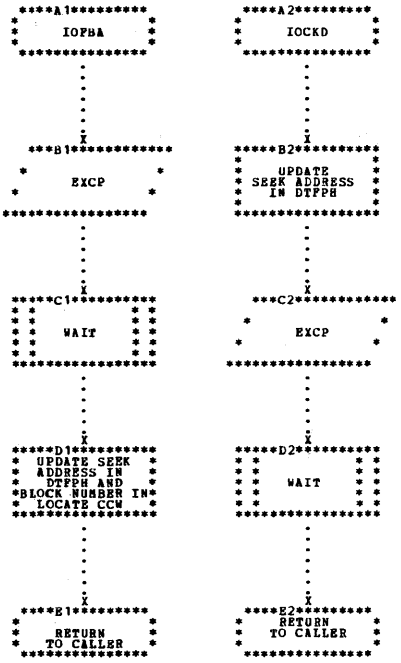


Chart SJ. $$BRMSG1 - Checkpoint Message Routine
(Refer to Chart 23)

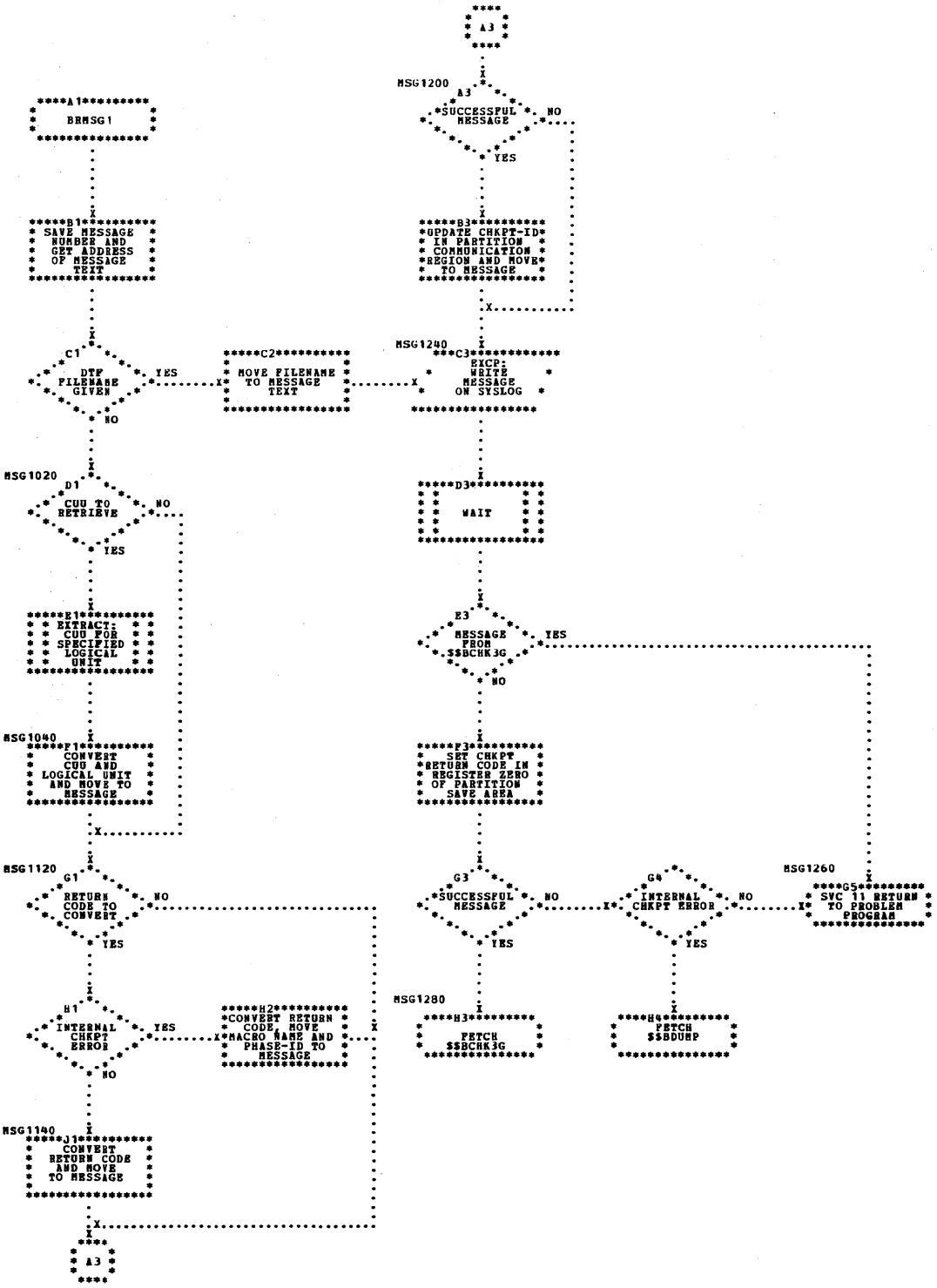


Chart SK. $$BCHK3G - Checkpoint Erase-Gap Routine
(Refer to Chart 23)

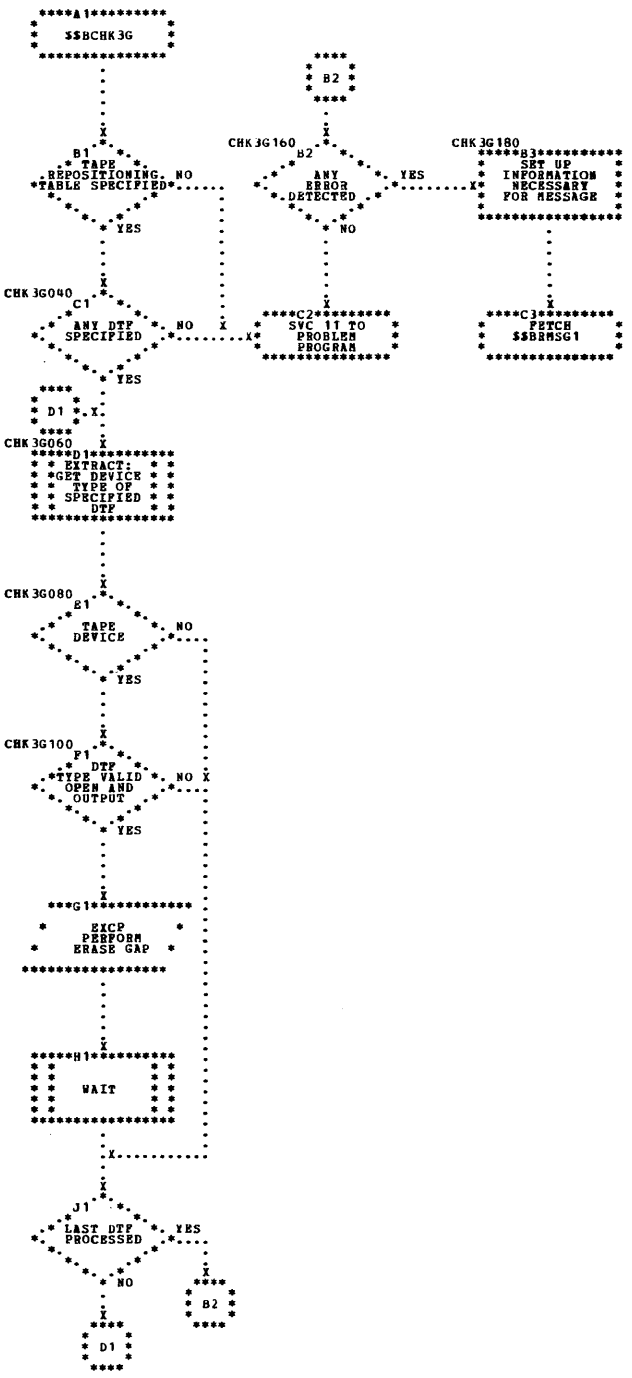


Chart SL. $$BRSTRT - Restore Problem Program (Part 1 of 2)
(Refer to Chart 23)

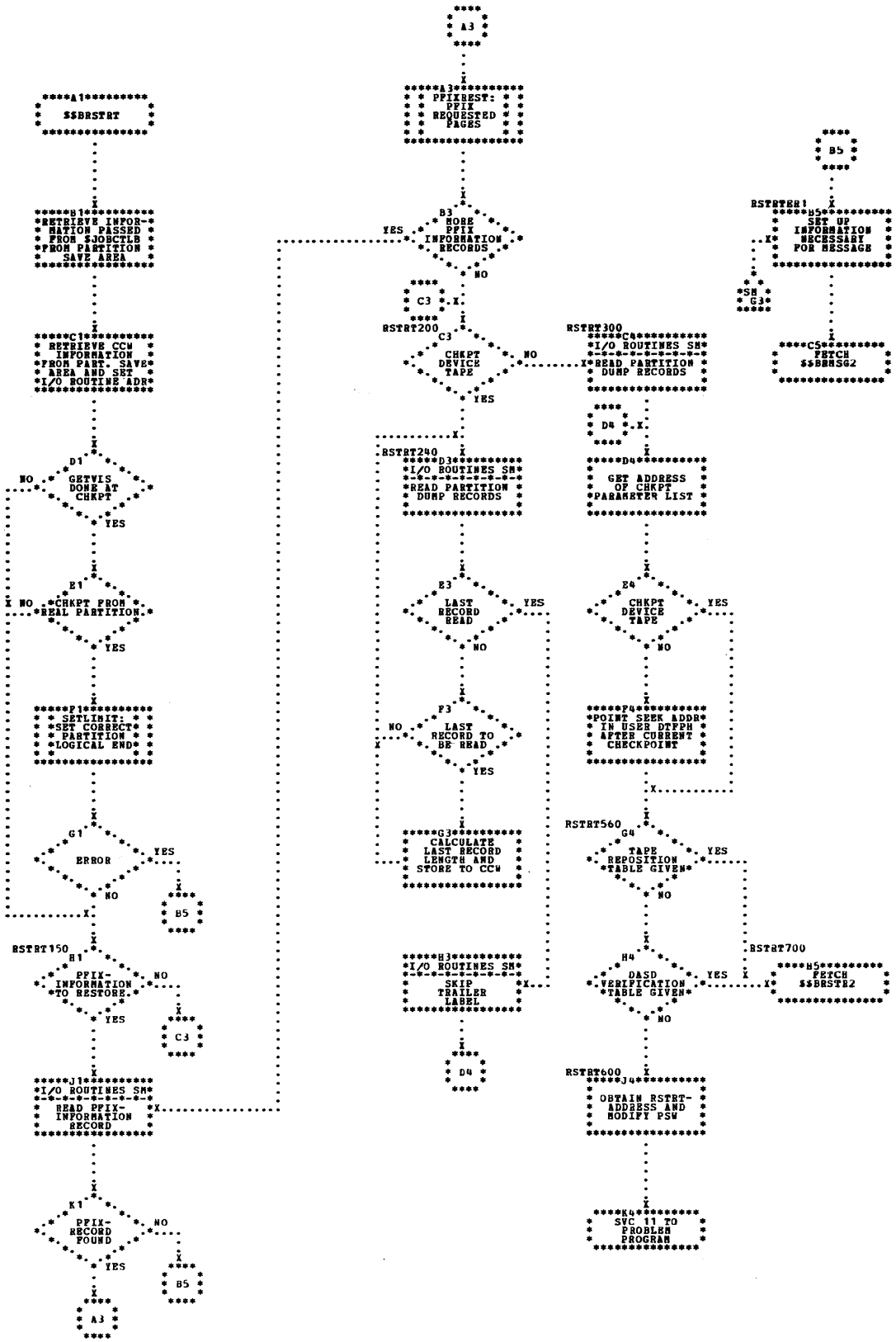


Chart SM. $$BRSTRT - Restore Problem Program (Part 2 of 2)
(Refer to Chart 23)

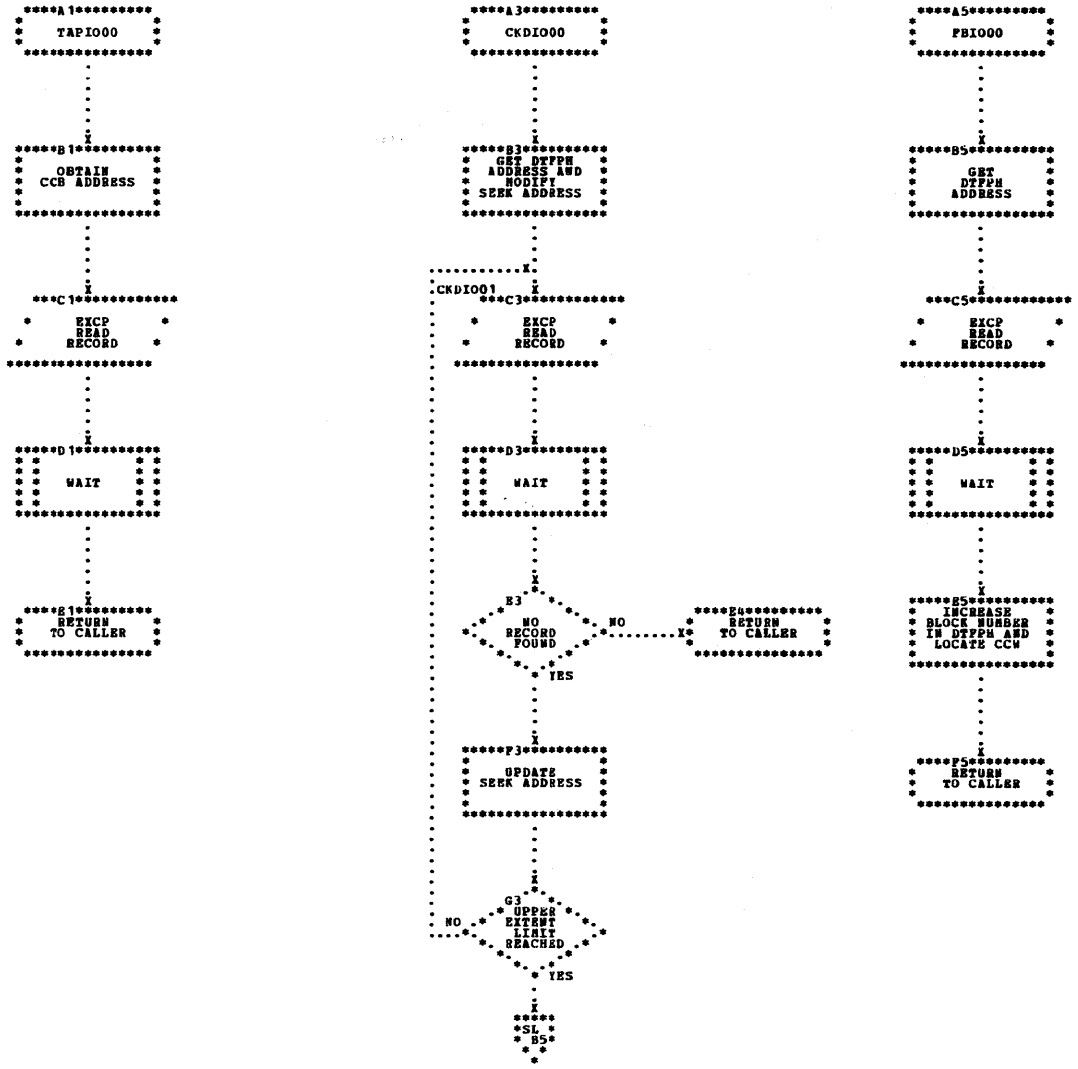


Chart SN. $$BRSTR2 - Tape Repositioning and DASD Verification Routine (Part 1 of 2)
(Refer to Chart 23)

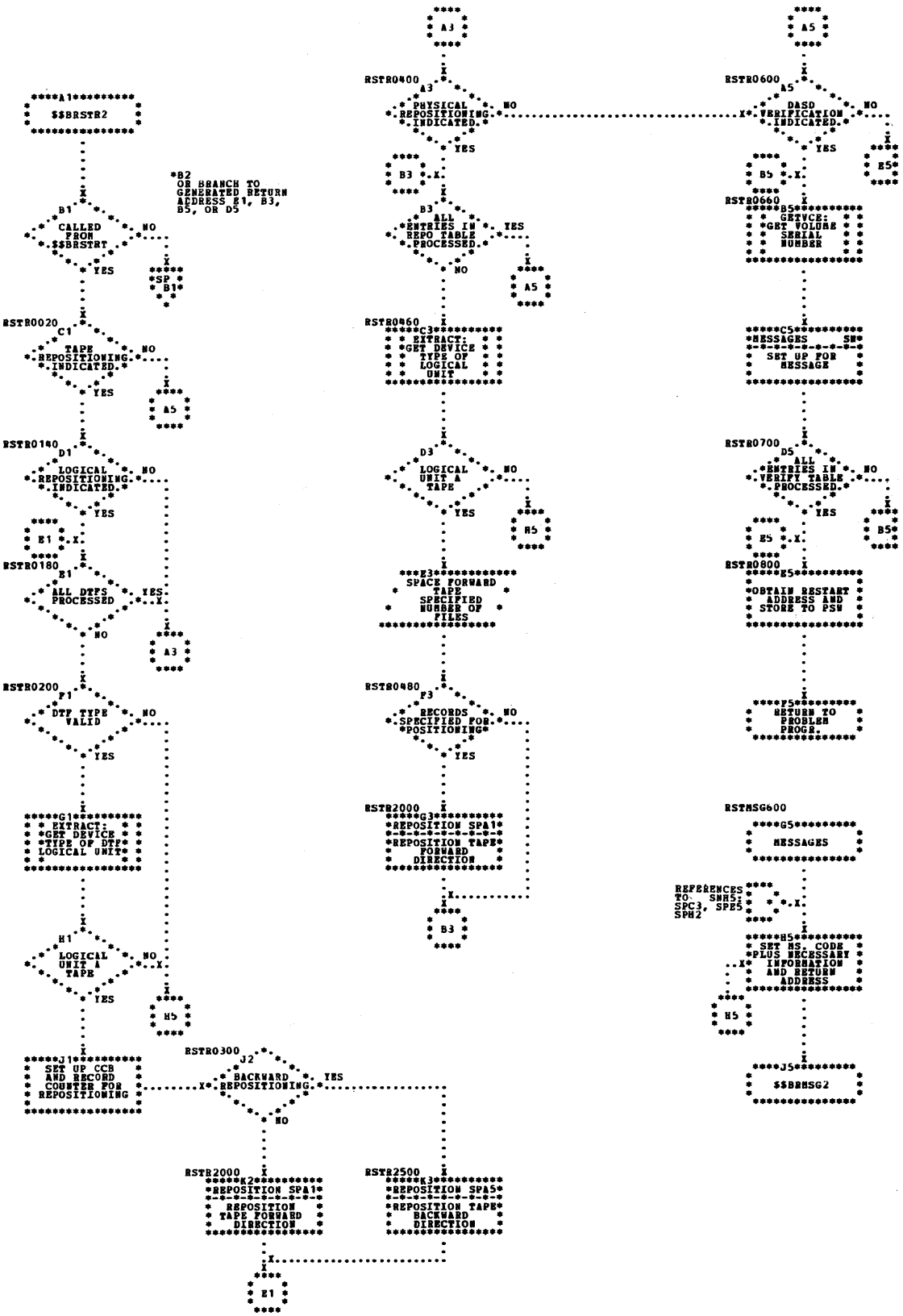


Chart SP. $$BRSTR2 - Tape Repositioning and DASD Verification Routine (Part 2 of 2)
(Refer to Chart 23)

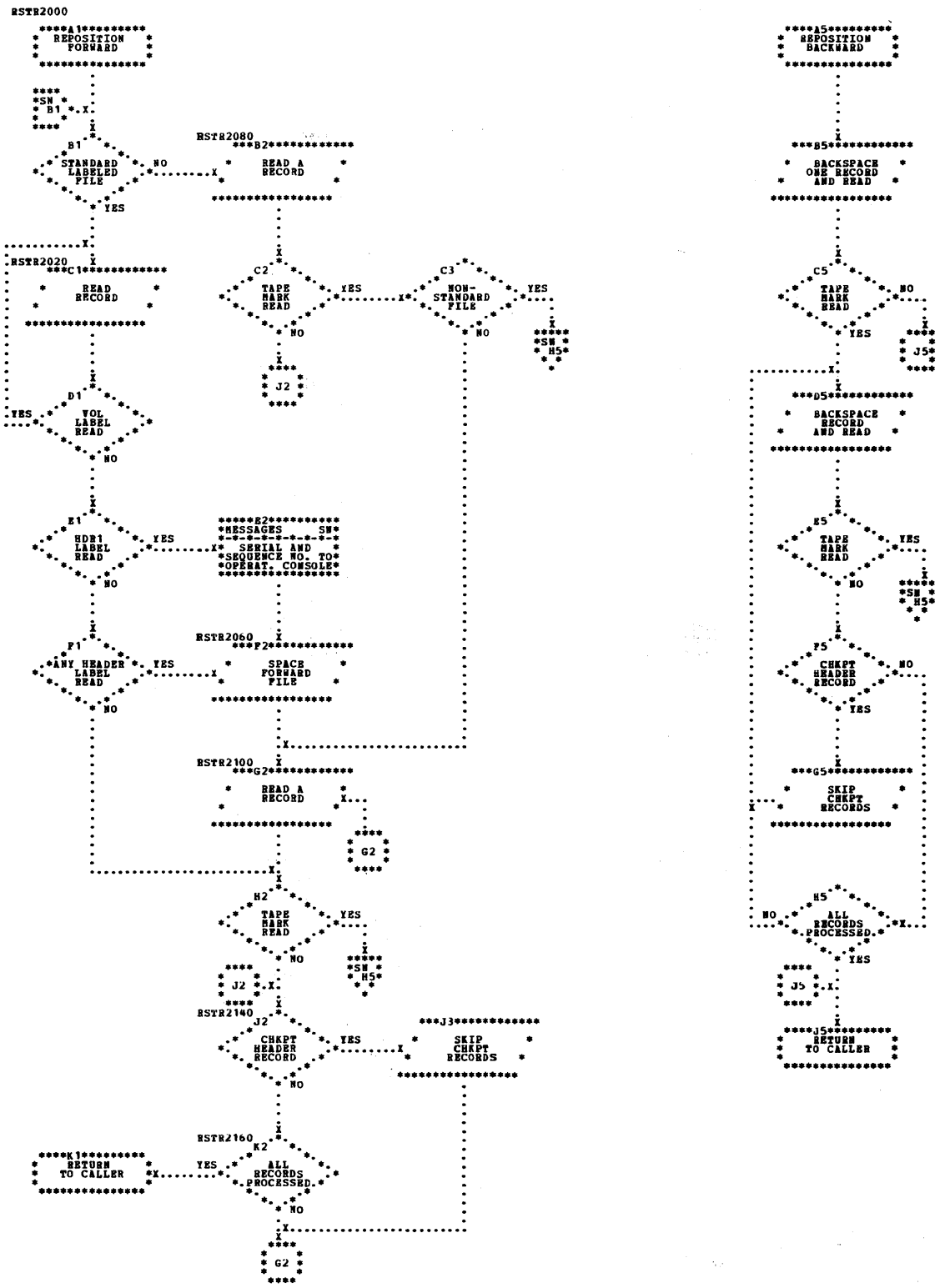
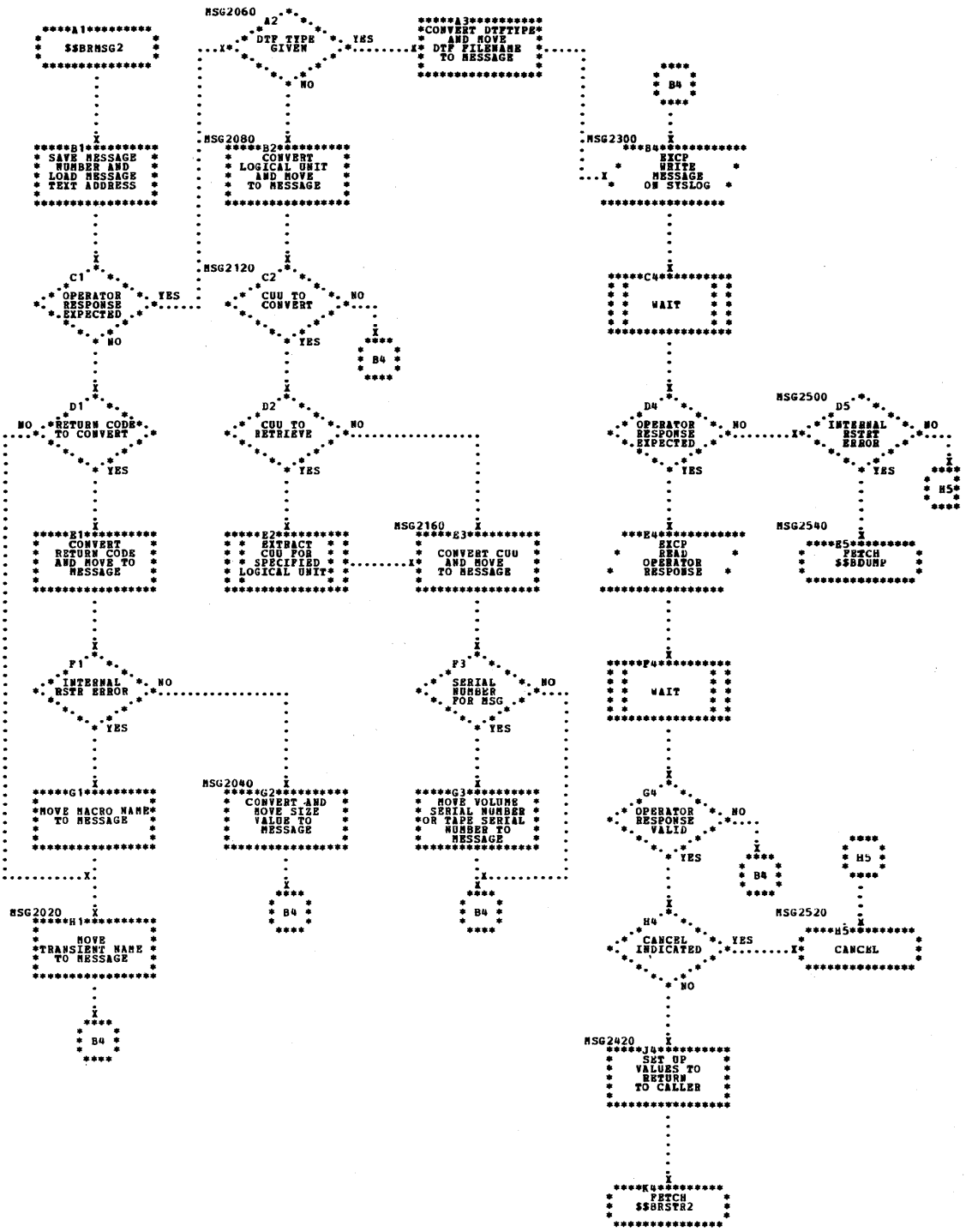


Chart SQ. $$BRMSG2 - Restart Message Routine



Message	Phase	Chart
0C00I	$$BCHKP2	SB
	$$BCHKPF	SF
0C02I	$$BCHKPT	SA
0C03I	$$BCHKPT	SA
	$$BCHKPE	SE
0C04I	$$BCHKPT	SA
	$$BCHKPD	SD
0C05I	$$BCHKPD	SD
0C06I	$$BCHKPD	SD
0C07I	$$BCHKPD	SD
0C08I	$$BCHKPD	SD
0C09I	$$BCHKPE	SE
0C10I	$$BCHKPT	SA
	$$BCHKPD	SD
0C11I	$$BCHKPT	SA
	$$BCHKPD	SD
0C12I	$$BCHKPT	SA
	$$BCHKPD	SD
0C13I	$$BCHKP2	SB
	$$BCHKP3	SC
0C14I	$$BCHKPT	SA
	$$BCHKPD	SD
0C15I	$$BCHKPT	SA
0C16I	$$BCHKPG	SG
	$$BCHKP3	SC
0C17I	$$BCHKPT	SA
	$$BCHKPD	SD
	$$BCHKPE	SE
0C18I	$$BCHK3G	SK
0C19I	$$BCHKPD	SD
0D01A	$$BOCRTV	PK
0D02I	$$BOCRTM	NA
0D03A	$$BOCRTX	QB
0D04D	$$BOCRTV	PK
0D05A	$$BOCRTW	QA
	$$BOCRT1	RA
0D06A	$$BOCRTY	QD
0D07D	$$BOCRTS	PD
0D08A	$$BOCRTS	PD
0D09D	$$BOCRTR	PA
0D20E	$$BOCRTB	KD
0D25E	$$BOCRTB	KD
0D26E	$$BOCRTB	KD
0D28A	$$BOCRTG	MA
0D29E	$$BOCRTB	KD
0D30A	$$BOCRTV	PK
	$$BOCRT1	RA
0D33A	$$BOCRTE	LD
0D34E	$$BOCRTF	LG
0D35E	$$BOCRTF	LG
0D36I	$$BOCRTF	LG
0D37I	$$BOCRT5	RJ
0D38A	$$BOCRTE	LD
0D39D	$$BOCRTI	MG
0D40A	$$BOCRTI	MG
0D41D	$$BOCRTI	MG
0D97A	$$BOCRT1	RA
0P70I	IJBEOJ	FP
0P71I	IJBEOJ	FP
0P72I	IJBEOJ	FP
0P73I	IJBEOJ	FP
0P74I	IJBEOJ	FP
0P75I	IJBEOJ	FP
0P76I	IJBEOJ	FP
0P77I	IJBEOJ	FP
0P78I	IJBEOJ	FP
0P79I	IJBEOJ	FP
0P80I	IJBEOJ	FP

0P81I	IJBEOJ	FP
0P82I	IJBEOJ	FP
0P83A	IJBEOJ	FP
0P84I	IJBEOJ	FP
0P85I	$$BEOJ	DA
0P86I	IJBEOJ	FP
0P87A	$$BEOJ	DA
0P88I	IJBEOJ	FP
0P91I	$$BEOJ	DA
0R09I	$$BRSTRT	SL
0R13I	$$BRSTRT	SL
0R15I	$$BRSTR2	SN
0R20A	$$BRSTR2	SN
0R21A	$$BRSTR2	SN
0R22A	$$BRSTR2	SN
0R23A	$$BRSTR2	SN
0R24A	$$BRSTR2	SN
0R25A	$$BRSTR2	SN
0R26A	$$BRSTR2	SN
0R27A	$$BRSTR2	SN
0R28A	$$BRSTR2	SN
0R29A	$$BRSTR2	SN
0S00I	IJBEOJ	FP
0S01I	IJBEOJ	FP
0S02I	IJBEOJ	FP
0S03I	IJBEOJ	FP
0S04I	IJBEOJ	FP
0S05I	IJBEOJ	FP
0S07I	IJBEOJ	FP
0S08I	IJBEOJ	FP
0S12I	IJBEOJ	FP
0S13I	IJBEOJ	FP
0S14I	IJBEOJ	FP
0S16I	IJBEOJ	FP
0S17I	IJBEOJ	FP
0V02I	IJBEOJ	FP
0V03I	IJBEOJ	FP
0V04I	IJBEOJ	FP
0V05I	IJBEOJ	FP
0V06I	IJBEOJ	FP
0V07I	IJBEOJ	FP
0V08I	IJBEOJ	FP
0V09I	IJBEOJ	FP
0V10I	IJBEOJ	FP
0V11I	IJBEOJ	FP
1B12D	$$BATTF1	CC
	$$BATTU1	CJ
1B13A	$$BATTF3	GN
	$$BATTF4	CD
1B14A	$$BATTF2	GL
	$$BATTF5	CF
1B15I	$$BATTF4	CD
	$$BATTF5	CF
	$$BATTU2	CK
1B16I	$$BATTF4	CD
	$$BATTF5	CF
	$$BATTU2	CK
1B17I	$$BATTF5	CF
1B18A	$$BATTU2	CK
1B19I	$$BATTF2	GL
	$$BATTF3	GN
1B20A	$$BATTF2	GL
	$$BATTF3	GN
	$$BATTF4	CD
	$$BATTF5	CF
	$$BATTU2	CK
1C40I	$$BATTNA	AA
	$$BATTNB	AE
1C50I	$$BATTNA	AA
	$$BATTNB	AE
1I30D	$$BATTNC	AG
1I41A	$$BATTNT	BF
	$$BATTNU	BH
1I42D	$$BATTNT	BF
1I43D	$$BATTNV	BE
1I44I	$$BATTNX	BN

1I45D	$$BATTNT	BF
1I46D	$$BATTNV	BE
	$IJBDUMP	EA
1I47I	$$BATTNT	BF
1I48I	$$BATTNU	BH
1I49I	$IJBDUMP	EA
1I51I	$IJBDUMP	EA
1I52I	$IJBDUMP	EA
1P01D	$$BATTNE	AM
1P02I	$$BATTNE	AM
1P03D	$$BATTNF	AP
1P04D	$$BATTNF	AP
1P1nD	$$BATTNG	AT
1Q62I	$$BATTNC	AG
1S0nD	$$BATTNB	AE
	$$BATTNC	AG
	$$BATTNE	AM
	$$BATTNF	AQ
	$$BATTNG	AT
	$$BATTNH	AV
	$$BATTNN	BA
	$$BATTNQ	BB
	$$BATTNS	BD
	$$BATTNT	BF
	$$BATTNU	BH
	$$BATTNY	BT
	$$BATTNZ	BW
	$$BATTN2	BV
5J95I	IJBEOJ	FP
5J96I	IJBEOJ	FP
P100I	$$BATTS1	CS
P101I	$$BATTS2	CV
P102I	$$BATTN7	CL
P103I	$$BATTN7	CL
P104I	$$BATTN7	CL
P105I	$$BATTS1	CS
P106I	$$BATTS1	CS
P107I	$$BATTN7	CL
	$$BATTS1	CS
P108I	$$BATTS1	CS
P109I	$$BATTN7	CL
P201I	$$BPCLOS	

APPENDIX B: TOTAL MESSAGE CROSS-REFERENCE LIST

All messages issued by the DOS/VS system control programs, with a reference to the PLM and issuing phase, are listed in DOS/VS Supervisor Logic, SY33-8551. For cause and action of each message, see DOS/VS Messages, GC33-5379.

Label	Phase	Location
ADVANCE	$$BOCRTC	KJC4
ADVANCE	$$BOCRTD	LCC3
ADVANCE	$$BOCRTG	MCC5
ADVANCE	$$BOCRTY	QDA2
AENDEXIT	$$BOCRTZ	QKA3
AGAIN	$$BEOH4	DGC4
AGAIN	$$BEOJ	DBA2
ALARM	$$BOCRTR	PCB4
ALLP1	$$BOCRT2	RDA4
ALLXFR	$$BOCRTD	LAA4
ALMPASS	$$BOCRTN	NDA2
ALMYES	$$BOCRTN	NDH1
ALRMCMD	$$BOCRTZ	QJA2
ALTER	$$BATTNT	BBF1
ALTER1	$$BATTNT	BBD1
ASSGNLOG	$$BATTNG	ASB5
ATTEXIT	$$BOCRTK	MLB3
ATTEXIT	$$BOCRTP	NHE1
ATTNRET	$$BATTF4	CDE5
ATTPENDO	$$BOCRTX	QBD3
ATTWAIT	$$BOCRTR	PBB4
ATTWAIT	$$BOCRTS	PFB4
ATTWAIT	$$BOCRTW	QAB3
ATTWAIT	$$BOCRTY	QEB4
AUTOCLOS	$$BEOJ4	DFC3
AUTOEND	$$BEOJ4	DFF3
B	IJBEOJ	FPJ2
BAL	$$BATTF2	GLC4
BAL	$$BATTF3	GNC4
BAL	$$BATTF4	CDC4
BAL	$$BATTU2	CKH4
BAL	IJBDMPAR	EWC1
BATCH	$$BATTNG	ARC4
BCOUN	$$BOCRT2	RDD4
BEGIN	$$BATTNU	BCB2
BEGIN	$$BOCRTI	MGC1
BEGINZ1	$$BOCRTZ	QGC2
BEGINZ2	$$BOCRTZ	QGD2
BEGINZ4	$$BOCRTZ	QGF2
BEGINZ5	$$BOCRTZ	QHA3
BEGINZ6	$$BOCRTZ	QHD3
BEGIN1	$$BOCRTI	MGC2
BGP1	$$BOCRT2	RDB4
BLOCKED	$$BATTU2	CKD2
BLOCKPUT	IJBDMPAR	EUB3
BTLOOP	$$BATTNH	ATE3
BWP	$$BOCRT2	RDE3
BYPASS	$$BSDRUP	GHD4
BYPSTM	$$BATTN3	BPC1
CALC2	IJBSDUMP	EGF4
CANCELCD	$$BEOH4	DGJ1
CASEBA	$$BOCRT6	RLJ2
CASE1A	$$BOCRT6	RLG5
CASE10	$$BOCRT6	RLG4
CASE2	$$BOCRT6	RLE5
CASE4A	$$BOCRT6	RLF5
CAUSE2	IJBEOJ	FPG3
CCWLNG	$$BATTNT	BBD2
CDISK	IJBDMPIC	FJA3
CD3	$$BOCRT6	RLF1
CEX	$$BOCRTR	PBB3
CFBA	IJBDMPIC	FJD2
CHAINING	$$BOCRTN	NCA3
CHAINTST	$$BOCRTE	LDE4
CHECK	$$BOCRTN	NBC2
CHECK	$$BOCRT1	RBK2
CHECKEND	$$BATTN7	CPH2
CHECKKB	$$BOCRTK	MJG2
CHECK1	$$BOCRTZ	QHJ3
CHECK3	$$BOCRTZ	QKB2
CHHIRQ	$$BATTNS	AXF1
CHK	$$BOCRTE	LFB3
CHKAPRT	$$BEOJ4	DFF1
CHKBLK	$$BATTN2	BMF2
CHKDIGIT	$$BATTN7	CRD3
CHKLIMIT	$$BATTN7	CME3
CHKNXT	$$BOCRTS	PEB3
CHKPD	$$BCHKPD	SDB1
CHKPDR10	$$BCHKPD	SDB5
CHKPD020	$$BCHKPD	SDF1
CHKPD040	$$BCHKPD	SDG1
CHKPD060	$$BCHKPD	SDJ1
CHKPD090	$$BCHKPD	SDK1
CHKPD110	$$BCHKPD	SDC2
CHKPD120	$$BCHKPD	SDB3
CHKPD140	$$BCHKPD	SDC3
CHKPD180	$$BCHKPD	SDD2
CHKPD200	$$BCHKPD	SDF2
CHKPD220	$$BCHKPD	SDD3
CHKPD290	$$BCHKPD	SDG2
CHKPD420	$$BCHKPD	SDK2
CHKPD660	$$BCHKPD	SDB4
CHKPD670	$$BCHKPD	SDD4
CHKPD740	$$BCHKPD	SDF4
CHKPEER1	$$BCHKPE	SEB5
CHKPER1	$$BCHKP2	SBD4
CHKPE005	$$BCHKPE	SEC1
CHKPE080	$$BCHKPE	SEF1
CHKPE140	$$BCHKPE	SEJ1
CHKPE220	$$BCHKPE	SEG1
CHKPE260	$$BCHKPE	SEA2
CHKPE360	$$BCHKPE	SEA3
CHKPE380	$$BCHKPE	SEC3
CHKPE400	$$BCHKPE	SED3
CHKPE440	$$BCHKPE	SEG3
CHKPE500	$$BCHKPE	SEJ3
CHKPF020	$$BCHKPF	SFD1
CHKPF090	$$BCHKPF	SFD2
CHKPF140	$$BCHKPF	SFG1
CHKPF160	$$BCHKPF	SFH1
CHKPF180	$$BCHKPF	SFA3
CHKPF200	$$BCHKPF	SFC3
CHKPF220	$$BCHKPF	SFF3
CHKPF300	$$BCHKPF	SFG3
CHKPG030	$$BCHKPG	SGC1
CHKPG050	$$BCHKPG	SGD1
CHKPG060	$$BCHKPG	SGF1
CHKPG080	$$BCHKPG	SGJ1
CHKPTER0	$$BCHKPT	SAB5

Label	Phase	Location
CHKPT020	$$BCHKPT	SAF1
CHKPT040	$$BCHKPT	SAG1
CHKPT060	$$BCHKPT	SAK1
CHKPT090	$$BCHKPT	SAE2
CHKPT150	$$BCHKPT	SAK2
CHKPT180	$$BCHKPT	SAC4
CHKPT320	$$BCHKPT	SAG4
CHKPT360	$$BCHKPT	SAJ4
CHKP2030	$$BCHKP2	SBB2
CHKP2040	$$BCHKP2	SBD2
CHKP2110	$$BCHKP2	SBD3
CHKP2150	$$BCHKP2	SBF2
CHKP2160	$$BCHKP2	SBG3
CHKP2165	$$BCHKP2	SBH3
CHKP2170	$$BCHKP2	SBB4
CHKP3ER1	$$BCHKP3	SCG2
CHKSTT	$$BATTNH	ATB1
CHKSTT1	$$BATTNH	ATD1
CHKSUB	$$BEOJ	DAE3
CHKSUB	$$BEOJ4	DHG5
CHKSUB1	$$BEOH4	DGA1
CHKSVA	$$BATTNT	BAE2
CHKSVA1	$$BATTNT	BBA1
CHK1RR	$$BATTNS	AXE1
CHK2	$$BOCRTE	LDC2
CHK3	$$BOCRTE	LFC3
CHK3	$$BOCRTF	LKC4
CHK3G040	$$BCHK3G	SKC1
CHK3G060	$$BCHK3G	SKD1
CHK3G080	$$BCHK3G	SKE1
CHK3G100	$$BCHK3G	SKF1
CHK3G160	$$BCHK3G	SKB2
CHK3G180	$$BCHK3G	SKB3
CKDI001	$$BRSTRT	SMC3
CKDPHASE	$$BCCHHR	GJE3
CKECCCQ	$$BATTNS	AXG3
CKECCCTH	$$BATTNS	AXH3
CKECCM	$$BATTNS	AXB3
CKECCMQ	$$BATTNS	AXC3
CKFORT55	$$BATTNS	AXE3
CKHIREOT	$$BATTNS	AXG1
CKPIBFLG	$$BATTNG	ASG4
CLEANUP	$$BOCRTQ	NJH1
CLEAR	$$BEOH4	DGF2
CLEARNXT	$$BOCRTA	KBH1
CLI	IJBEOJ	FPF4
CLOSAPRT	$$BEOJ4	DFF2
CLOSER	IJBDMPIC	FJB1
CLOSER	IJBDMPIO	FBB2
CLOSEO	IJBDMPIC	FHE3
CLRPASS	$$BOCRTO	NFJ3
CLRSCRN	$$BOCRTB	KDD2
CMDREJCT	$$BOCRTZ	QHG3
CNCLIN	$$BATTNC	AGD1
CNCLTEST	IJBEOJ	FPB2
CNLRTN	$$BATTNC	AGB3
COMMON	$$BOCRTP	NHB3
COMMON1	$$BOCRTP	NHB4
COMMSG	$$BOCRTO	NEJ2
COMPARE2	$$BATTN7	CRD2
COMPASS	$$BOCRTN	NBA5
CONPASS	$$BOCRTN	NCA4
CONT	$$BEOJ3	DDF2
CONTRD	$$BOCRTR	PAA2
CONTRD	$$BOCRTT	PHA3
CONTRDE	$$BOCRTS	PEA2
CONTRD1	$$BOCRTS	PDH1
CONTREAD	$$BOCRTS	PDE2
CONTROL	$$BATTNA	ABB1
CONTROL	$$BOCRTS	PEA5
CONT1	$$BATTNV	BGB1
CONV	IJBDMPIC	FJG3
CONYES	$$BOCRTN	NCH5
COUNT	$$BOCRTG	MAH4
CO1	$$BOCRT6	RMA1
CPRINT	IJBDMPIC	FJH3
CRT	$$BEOJ4	DFG1
CRTDSBL	$$BOCRTA	KAH4
CRTDSBL	$$BOCRTK	MHC1
CRTDSBL1	$$BOCRTA	KAK4
CRTEND	$$BOCRTA	KBD3
CRTENTRY	$$BOCRTK	MHB2
CRTEOJRT	$$BOCRTA	KBF4
CRTH040	$$BOCRTH	MDD1
CRTH110	$$BOCRTH	MDH2
CRTH120	$$BOCRTH	MDF3
CRTU10	$$BOCRTU	PJC1
CRTU40	$$BOCRTU	PJH1
CTAPE	IJBDMPIC	FJE1
CURSENT	$$BOCRTX	QBG4
CURSENT	$$BOCRTY	QDA5
CURSOP	$$BOCRTK	MHK4
CUUADDR	$$BATTN7	CMB3
D	IJBEOJ	FPH4
DATCHAIN	$$BOCRTS	PDG1
DECR	$$BOCRT2	RCG2
DECRE	$$BOCRT3	RED3
DEFAULT	$$BOCRTL	MMF4
DELPASS	$$BOCRTN	NCA1
DELTH0	$$BEOH4	DGH4
DELTH1	$$BEOJ4	DHC1
DELTH2	$$BEOJ4	DHE1
DELTH3	$$BEOJ4	DHG2
DELYES	$$BOCRTN	NCH2
DERROUT	$$BATTNU	BCG2
DEVOK	$$BATTNV	BGJ1
DEV3800	$$BATTS1	CSC2
DIAG135Q	$$BATTNZ	BKE5
DIAG135R	$$BATTNZ	BKE4
DIGIT1	$$BOCRTL	MNF2
DIGIT2	$$BOCRTL	MMA2
DIPA	$$BOCRT6	RLE2
DISABLE	$$BEOJ3A	DEK2
DISKB	IJBDMPIC	FHC4
DISKETT	$$BEOJ	DCA3
DISKETT	IJBEOJ	FRF3
DISK1	$$BEOH4	DGE4
DKTYPE	$$BEOJ	DCH3
DKTYPE	IJBEOJ	FRC3
DKTYPE1	$$BEOJ	DCH4
DKTYPE1	IJBEOJ	FRC4
DOCAEX	$$BOCRTC	KHE4
DOCAEX	$$BOCRTE	LDK4
DOCAEX	$$BOCRTF	LJE2
DOCAEX	$$BOCRTG	MBB3
DOCCEX	$$BOCRTD	LAB5
DOCCEX	$$BOCRTE	LDF5
DOCDEX	$$BOCRTC	KGG4
DOCEEX	$$BOCRTF	LHF3
DOCFEX	$$BOCRTE	LDK3
DOCFEX	$$BOCRTE	LEB1

Label	Phase	Location
DOCFEX1	$$BOCRTE	LDK2
DOCGEX	$$BOCRTB	KDK2
DOCGEX	$$BOCRTC	KGB2
DOCGEX1	$$BOCRTC	KGE2
DOCGEX2	$$BOCRTC	KGC2
DOCHEX	$$BOCRTY	QDC4
DOCLEX	$$BOCRTN	NBA3
DOCLEX1	$$BOCRTN	NBE1
DOCMEX	$$BOCRTX	QBH4
DOCNEX	$$BOCRTQ	NJD4
DOCOEX	$$BOCRTQ	NJG3
DOCO2EX	$$BOCRTK	MJB4
DOCPEX	$$BOCRTO	NEB5
DOCPEX	$$BOCRTQ	NKE4
DOCQEX	$$BOCRTB	KFE2
DOCQEX	$$BOCRTC	KJG3
DOCQEX	$$BOCRTD	LCF2
DOCQEX	$$BOCRTL	MMG2
DOCQEX1	$$BOCRTB	KFF1
DOCQEX2	$$BOCRTB	KFE3
DOCQEX3	$$BOCRTB	KFF3
DOCQO2S	$$BOCRTQ	NJD3
DOCVEX	$$BOCRTL	MNG3
DOCVEX	$$BOCRTN	NBG4
DOCV1	$$BOCRTL	MNE3
DOCV2	$$BOCRTL	MNF3
DONE	$$BEOJ3A	DEJ4
DOSKIP	$$BATTF2	GLB5
DOSKIP	$$BATTF3	GNB5
DOSKIP	$$BATTF4	CDB5
DOSKIP	$$BATTF5	CFF5
DOUPDATE	$$BEOJ3A	DEF4
DOWN	$$BOCRT3	RFC2
DSDUMMY1	$$BATTNU	BCK3
DSPCROUT1	$$BATTNU	BDD3
DSPCROUT2	$$BATTNU	BDF2
DSPL	$$BOCRT3	REJ4
DSPLY	$$BATTNU	BCD5
DSPLY1	$$BATTNU	BCA5
DSVA	IJBSDUMP	EMC4
DTCHAT	$$BATTNA	AAB5
DTINUN	$$BATTNA	ACC4
DUMMY	$$BATTNT	BAD2
DUMMY1	$$BATTNT	BAA4
DUMPALLP	$$BATTNV	BEE4
DUMPAR1	IJBDMPPA	ERA4
DUMPCR	IJBDMPPA	EQB1
DUMPCR	IJBSDUMP	EJA1
DUMPEND	IJBSDUMP	ELE4
DUMPEND1	IJBSDUMP	ELF4
DUMPIT	IJBSDUMP	EHK1
DUMPPART	IJBSDUMP	EFA4
DUMPSUP	IJBSDUMP	ELF1
DV3800	$$BATTN7	CMB2
ECCCTBIT	$$BATTNS	AYG3
ENBL	$$BOCRTE	LFE3
ENBL	$$BOCRTF	LKF4
END	$$BOCRTI	MGH1
END	$$BOCRT2	RCH3
ENDRED	$$BOCRT3	RFE1
END1	$$BOCRTQ	NKG1
END2	$$BOCRT2	RDE5
ENTER	$$BOCRT1	RBA1
EQCHK	$$BOCRTF	LHA5
EQCHK1	$$BOCRTF	LHB2

Label	Phase	Location
ERA	$$BOCRT2	RCF5
ERASE	$$BOCRTC	KKK4
ERA1	$$BOCRT2	RCE5
EREPIN	$$BSDRUP	GHB4
EREPOUT	$$BSDRUP	GHG3
ERR	$$BATTNV	BGB5
ERR	IJBDMPIC	FGD1
ERRE	IJBDMPIC	FGE1
ERROR	$$BOCRTE	LDA2
ERROR	$$BOCRT1	RAC5
ERROR	$$BOCRT6	RLH3
ERRORCAS	$$BOCRT2	RCF1
ERROR4	$$BOCRTF	LGF2
ERRROUT	$$BATTNT	BAH2
ERRRTN	$$BATTNA	ACB1
ERR1	$$BOCRTE	LFD4
ERR1	$$BOCRTF	LKE4
ERR2	$$BATTNT	BBA2
ESETID	$$BATTNI	ALC2
ESKIPRTY	$$BATTNI	ALB2
ESUPV	$$BATTNI	ALD1
ETEST	$$BOCRTL	MMA3
EX	$$BOCRTE	LFG2
EX	IJBDMPIO	FAF2
EXCPRG	$$BATTNA	ACB2
EXIT	$$BCCHHR	GJB4
EXIT	$$BOCRTE	LEH2
EXIT	$$BOCRTF	LHA4
EXIT	$$BOCRTO	NEB4
EXIT	$$BOCRTP	NHD1
EXIT	$$BOCRTQ	NKE3
EXIT	$$BOCRTU	PJC3
EXIT	$$BSYSWR	GGF2
EXIT	IJBDMPAR	ESC2
EXITA	$$BOCRTK	MKA2
EXITA1	$$BOCRTK	MKE2
EXITA2	$$BOCRTK	MKG2
EXITZ	$$BOCRTZ	QHE1
EXITZIA	$$BOCRTZ	QHG1
EXITZ1	$$BOCRTZ	QHF1
EXITZ2	$$BOCRTZ	QHH5
EXIT00	$$BEOJ7	DMH4
EXIT01	$$BEOJ7	DMJ4
EXIT1	$$BCCHHR	GJG4
EXIT1	$$BOCRTF	LHF1
EXIT1	$$BOCRTV	PKE3
EXIT1	$$BOCRT3	RFJ1
EXIT1	$$BOCRT6	RLJ5
EXIT2	$$BOCRT6	RMA4
EXT	IJBDMPAR	ETD2
EXTINT	$$BATTNA	ABB3
FBAPHASE	$$BCCHHR	GJC3
FETCH	$$BEOH4	DGH5
FETCH	$$BEOJ3	DDK3
FETCHU	$$BOCRTA	KBJ3
FETCHZ	$$BOCRTA	KBG5
FGLST	$$BEOJ	DCG3
FGLST	IJBEOJ	FRB3
FIN	$$BOCRTS	PDA3
FINDKEY	$$BATTN7	CNB3
FINDPUB	$$BATTNV	BGE1
FINDPUB	$$BATTS1	CSC3
FINE	$$BOCRTT	PHB5
FINISH	$$BATTN7	CQE3
FIRSTX	$$BOCRTL	MMH3

Label	Phase	Location
FNDMATCH	$$BATTS2	CVC2
FOLD	$$BOCRT2	RCE2
FORGETIT	$$BATTN3	BPJ3
FORW	$$BOCRT3	REC2
FOUND	$$BATTNF	AQD3
FREER1	$$BEOJ	DBG5
FREETRK	$$BEOJ4	DHE3
FRET	$$BOCRTE	LDB2
FULLWORD	$$BATTNU	BCF3
FW	$$BOCRT3	RFF2
FWP	$$BOCRT2	RDF3
FWP1	$$BOCRT2	RDG3
GETBYT	IJBDMPAR	EUJ1
GETCMD	$$BATTNA	AAD2
GETK	$$BATTND	AMB3
GETPIBXT	$$BATTNC	AHG4
GETPID	$$BATTND	AMD1
GETPIK	$$BATTNC	AGB5
GETPIK1	$$BATTNC	AGC5
GETSIZE	$$BATTNF	APH1
GO	$$BOCRTE	LEA2
GO	$$BOCRTG	MAE1
GO	$$BOCRTK	MHF1
GO	$$BOCRTP	NGE2
GO	$$BOCRTR	PAD1
GO	$$BOCRTS	PDD1
GO	$$BOCRTT	PGD1
GO	$$BOCRTV	PKE1
GO	$$BOCRTW	QAE1
GO	$$BOCRTX	QBE1
GO	$$BOCRTY	QDE1
GO	$$BOCRT4	RHE1
GOON	$$BOCRTG	MBA2
GOON	$$BOCRTT	PHA2
GOON10	$$BOCRTD	LBA2
GOON11	$$BOCRTD	LBB3
GOON14	$$BOCRTD	LAJ2
GOON30	$$BOCRTC	KHD1
GOON4	$$BOCRTC	KJD2
GO1	$$BOCRTM	NAF2
GO1	$$BOCRTY	QDC3
GO2	$$BOCRTM	NAA4
GO2	$$BOCRTY	QDD2
HALTIO	$$BEOJ3A	DEB5
HALTO1	$$BEOJ3A	DEG5
HARDWAIT	$$BEOJ	DAE2
HARDWAIT	$$BOCRTF	LGJ2
HARDWAIT	$$BOCRTF	LJD4
HERE	$$BSYSWR	GGB1
HIRQUIET	$$BATTNS	AYG4
IGNORE	$$BATTNC	AJJ4
IGNORE	$$BOCRTK	MJB3
IJBATTF2	$$BATTF2	GLB1
IJBATTF4	$$BATTF4	CDB1
IJBATTF5	$$BATTF5	CFB1
IJBEJ335	$$BEOJ3	DDC2
IL	$$BOCRTB	KEA2
INCORRL	$$BOCRTB	KEA1
INCREM	IJBSDUMP	EGD1
INIT	$$BATTNT	BAC1
INITDUMP	IJBSDUMP	ECC3
INITIAL	$$BATTNU	BCC2
INITLOOP	$$BEOJ3A	DEK3
INITOPR	$$BATTNG	ASB1
INITOPR1	$$BATTNG	ASF2
INITOPR2	$$BATTNG	ASF3
INITPART	$$BEOJ	DBB1
INITPH	$$BCCHHR	GJE2
INIT00	$$BEOJ7	DMC1
INIT05	$$BEOJ7	DME2
INPRO	$$BOCRTQ	NKK1
INSOPN	$$BATTNA	ACD1
INSRTAEB	$$BATTNR	AWB4
INTERR	$$BEOJ	DAB3
INTV	$$BOCRTB	KED2
INTV	$$BOCRTE	LEC2
INVALID	$$BATTNT	BBC2
INVALID1	$$BATTNT	BAG4
INVALOUT	$$BOCRTM	NAB3
INVAL1	$$BATTNF	APB5
INVCOM	$$BOCRT1	RBH2
INVLSTS	IJBDMPIO	FAF4
INVP	IJBDMPAR	ESE4
INVPG	$$BEOJ	DBC2
INVPHASE	$$BATTF2	GLF2
INVPHASE	$$BATTF3	GND3
INVPHASE	$$BATTU2	CKH3
INVSYN	$$BATTNF	APG2
INVSYN1	$$BATTNF	APF2
INVSYN2	$$BATTNF	APD2
IO	$$BOCRTV	PKA2
IOERR	$$BATTF3	GNG3
IOERR	$$BATTF4	CDD3
IOERR	$$BATTF5	CFH4
IOREG05	$$BEOJ7	DNE3
IORETN	$$BOCRTK	MLD1
IORETRN	$$BOCRTG	MCD2
IORETRN	$$BOCRTP	NGE4
IORETRN	$$BOCRTR	PCG3
IOROUT	$$BOCRTG	MCB1
IOROUT	$$BOCRTK	MLB1
IOROUT	$$BOCRTP	NGC4
IOROUT	$$BOCRTR	PCF2
IOROUT	$$BOCRTS	PFB1
IOROUT	$$BOCRTT	PGB5
IOROUT	$$BOCRTV	PKC4
IOROUT	$$BOCRTW	QAC5
IOROUT	$$BOCRTY	QFC2
IOROUT	$$BOCRT1	RBB3
IOROUT	$$BOCRT4	RHD3
IOROUT1	$$BOCRTP	NGB4
IOROUT1	$$BOCRTV	PKB4
IOROUT1	$$BOCRTW	QAB5
IOROUT1	$$BOCRTX	QCB4
IOROUT1	$$BOCRTY	QFB2
IORT1	$$BCHKP2	SBC5
IORT1	$$BCHKP3	SCC4
IO1	$$BOCRTG	MBF4
IO6	$$BOCRTG	MBA4
IRCR00	$$BATTNZ	BKH5
ITERATE	$$BEOJ3A	DEA2
JAEXIT	$$BEOH4	DGA4
JCLFAIL	$$BEOJ	DAC4
KCOMM	$$BOCRTN	NBA2
KCOMMH	$$BOCRTA	KCA3
KCOMMON	$$BOCRTK	MJF3
KCOMMON	$$BOCRTN	NBD1

Label	Phase	Location
KEYWORD	$$BATTN7	CNC2
LABMV	$$BSYSWR	GGC1
LABMV1	$$BSYSWR	GGH1
LABMV2	$$BSYSWR	GGC2
LABMV3	$$BSYSWR	GGD2
LAMBV1	$$BSYSWR	GGG1
LASTPUB	$$BEOJ3A	DEB3
LBLPR	IJBSDUMP	EFD1
LEN3	$$BATTNV	BFE2
LIMIT	$$BOCRT3	RFA5
LINE	$$BOCRTB	KDD3
LINECOO	$$BOCRT6	RLH5
LINE2	$$BOCRTB	KEE1
LISTIO	$$BATTF2	GMC2
LISTIO	$$BATTF4	CEB3
LOAD	$$BATTF2	GLH3
LOADERR1	$$BATTN7	CPK3
LOADERR2	$$BATTS1	CSH2
LOADIT	$$BATTNA	ABG1
LOADPHAS	$$BATTF2	GLH1
LOADPHAS	$$BATTF3	GNG2
LOADPHAS	$$BATTF4	CDE2
LOADPHAS	$$BATTF5	CFD2
LOADPHAS	$$BATTU2	CKB4
LOADR3	$$BATTNT	BAJ1
LOADUPDT	$$BATTN7	CPJ2
LOG	$$BATTNC	AJB5
LOG	$$BOCRT2	RDA1
LOGGING	IJBEOJ	FQG2
LOGIO	$$BATTF2	GMD3
LOGIO	$$BATTF4	CEC1
LOGIOW	$$BATTF4	CEB1
LOOP	$$BATTF4	CEF1
LOOP	$$BOCRTR	PAF2
LOOPS3	$$BATTNF	APC3
LOOP1	$$BATTNE	AND3
LOOP1	$$BATTNF	AQC3
LOOP2A	$$BATTNF	AQE1
LOOP3	$$BATTNV	BHB4
LPIERR	$$BATTF5	CFC3
LSTSEG	$$BOCRTQ	NKB1
LTADUMP	IJBSDUMP	EDB2
LUBCHECK	$$BCCHHR	GJC2
LUBLOOP	$$BATTF4	CED3
MACFAIL	$$BATTS1	CTE2
MACFAIL	$$BATTS1	CUE2
MACRET	$$BATTF2	GLK2
MACRET	$$BATTF3	GNE3
MAINLINE	$$BOCRTR	PAG3
MAINLINE	$$BOCRTS	PEK2
MAINLINE	$$BOCRTT	PGH4
MAINLN	$$BOCRTK	MHJ3
MAIN00	$$BEOJ7	DMB1
MAIN05	$$BEOJ7	DMG1
MAIN07	$$BEOJ7	DMK2
MAIN10	$$BEOJ7	DME4
MATCH	$$BATTN7	CQD2
MAXFND	IJBSDUMP	EGE4
MICR	$$BEOJ4	DFG4
MICR1	$$BEOJ4	DFJ4
MISMATCH	$$BATTF4	CDF3
MISMATCH	$$BATTF5	CFJ3
MOVALUE	$$BATTS1	CUB3
MOVEDATA	$$BOCRTT	PGE2
MOVESUB	IJBEOJ	FPF5
MOVESUB1	IJBEOJ	FPG5
MOVETIC	$$BOCRTA	KCC4
MSG	$$BATTNB	AEB1
MSGOUT3	$$BATTNA	ACC3
MSG02	$$BATTS2	CVJ2
MSG1	$$BATTNB	AEG1
MSG1	$$BOCRTF	LJD3
MSG1020	$$BRMSG1	SJD1
MSG1040	$$BRMSG1	SJF1
MSG1120	$$BRMSG1	SJG1
MSG1140	$$BRMSG1	SJJ1
MSG1200	$$BRMSG1	SJA3
MSG1240	$$BRMSG1	SJC3
MSG1260	$$BRMSG1	SJG5
MSG1280	$$BRMSG1	SJH3
MSG2	$$BATTNB	AEC3
MSG2	$$BOCRTE	LEJ4
MSG2	$$BOCRTF	LJB5
MSG2020	$$BRMSG2	SQH1
MSG2040	$$BRMSG2	SQG2
MSG2060	$$BRMSG2	SQA2
MSG2080	$$BRMSG2	SQB2
MSG2120	$$BRMSG2	SQC2
MSG2160	$$BRMSG2	SQE3
MSG2300	$$BRMSG2	SQB4
MSG2420	$$BRMSG2	SQJ4
MSG2500	$$BRMSG2	SQD5
MSG2520	$$BRMSG2	SQH5
MSG2540	$$BRMSG2	SQE5
MSG3	$$BOCRTE	LEH3
MSG3	$$BOCRTF	LJA4
MVCMSG6	$$BATTNV	BGF2
MVERDY	$$BATTNC	AHD1
MVREMAIN	$$BATTS1	CUE3
NDSCAN	$$BATTNA	ADC3
NEWCMD	$$BOCRTD	LBE1
NEWL	$$BOCRTC	KGJ1
NEWLINE	$$BOCRTD	LAD2
NEWLINE1	$$BOCRTD	LAB3
NEWMIN	IJBSDUMP	EGD4
NEWVOL	$$BATTNC	AHB4
NEXT	$$BATTN7	CRE2
NEXTCCW1	$$BOCRTS	PEA3
NEXTDIB	$$BEOH4	DGF4
NEXTDIB	IJBDMPPA	EQD4
NEXTENT	IJBSDUMP	EGC2
NEXTENTRY	$$BCCHHR	GJJ2
NEXTLINE	$$BOCRTI	MGG2
NEXTPH	$$BCCHHR	GJG2
NEXT0	$$BATTS1	CTD3
NEXT2	$$BATTS1	CTF3
NEXT3	$$BATTS1	CTG3
NEXT4	$$BATTS1	CTH3
NOADD	$$BOCRTT	PGG3
NOADD1	$$BOCRTT	PHG3
NOAP	$$BEOJ	DAE1
NOAP	$$BEOJ3A	DEE1
NOAP	IJBEOJ	FPG4
NOAP	IJBSDUMP	EAE2
NOATTENT	$$BOCRTG	MCB2
NOATTENT	$$BOCRTR	PCF3
NOCRT	$$BEOJ4	DFH1
NOCURS	$$BOCRTP	NHC5
NODAT	$$BOCRTR	PAD5

Label	Phase	Location
NODATA	$$BOCRTT	PHE4
NODEL	$$BOCRTQ	NJD2
NOETSS	IJBSDUMP	ECD1
NOFIPA	$$BOCRT6	RLC4
NOFORMS	$$BATTF3	GNG1
NOHOLD	$$BEOJ4	DHE2
NOKCOMM	$$BOCRTA	KCA2
NOLOG	$$BATTNC	AJE5
NOLTA	IJBEOJ	FPA5
NOLTADUMP	IJBSDUMP	EDG2
NOMOVE	$$BATTS2	CVB3
NOMOVE	$$BOCRTC	KGJ3
NOMOVE	$$BOCRTD	LAB4
NOMV	$$BOCRTT	PGD3
NOOPRND	$$BATTN7	CNE5
NOPART	$$BEOH4	DGG1
NOPCMD	$$BOCRTZ	QJA1
NOPOWER	$$BEOH4	DGF1
NOPROT	$$BOCRTS	PDA4
NOPUB2	IJBDMPPA	EQG2
NOP2	$$BOCRTZ	QJD1
NORAS	$$BEOJ4	DFA3
NOREAD	$$BOCRT3	RFA1
NOREDSP	$$BOCRTV	PKH2
NOREQ	$$BEOJ3A	DED5
NORMAL	$$BATTNV	BFB1
NORMAL1	$$BATTNV	BFD2
NORMDMP	IJBSDUMP	ECG1
NOSOUND	$$BOCRTY	QEE1
NOSVADMP	IJBSDUMP	EHB1
NOSYSTSK	$$BOCRTA	KAA3
NOTFND	$$BATTF2	GLE2
NOTFND	$$BATTF3	GNC3
NOTFND	$$BATTF4	CDJ1
NOTFND	$$BATTF5	CFJ1
NOTFND	$$BATTU2	CKH2
NOTFULL	$$BOCRTW	QAA2
NOTFULL	IJBDMPIC	FHJ1
NOTIC	$$BOCRTC	KHA5
NOTOK	$$BOCRTH	MEF2
NOTPRT	IJBSDUMP	EJB1
NOTREADY	$$BOCRTR	PBA1
NOTXT	$$BOCRTT	PGA2
NOUNITCH	$$BOCRTA	KCE2
NOVOLID1	$$BEOJ7	DND4
NOWRITE	$$BOCRT1	RAD1
NO3211	$$BATTU2	CKC2
NVSERR	$$BATTNA	ACB5
N370RP	IJBSDUMP	EMD1
OALLDG	$$BACLOS	DKB4
OAL1	$$BACLOS	DJH1
OAL10	$$BACLOS	DKB1
OAL12	$$BACLOS	DKC1
OAL2A	$$BACLOS	DJF2
OAL3	$$BACLOS	DJG2
OAL4	$$BACLOS	DJH2
OAL6	$$BACLOS	DJB3
OAL7	$$BACLOS	DJB4
OAL8	$$BACLOS	DJC3
ODASD	IJBDMPIC	FGC2
OFBA	IJBDMPIC	FGC5
OKEXIT	IJBDMPIC	FFG5
OKEXIT	IJBDMPIO	FBD2
ONELINE	$$BOCRTB	KEB4
ONETST	$$BOCRTC	KKJ1
ONLIST	$$BEOJ	DCA2
ONLIST	IJBEOJ	FRF1
ONLYD	$$BOCRT2	RCG3
ONLYENT	$$BOCRT1	RBC2
OPENEND	IJBDMPIC	FGG5
OPMESS	$$BATTF2	GLB4
OPMESS	$$BATTF3	GNB4
OPMESS	$$BATTF4	CDB4
OPMESS	$$BATTF5	CFB5
OPRSNT	$$BATTNC	AGE2
OPRTNUM	$$BOCRTO	NEA4
ORRDY	$$BATTNC	AHE1
OUTCALL	$$BSDRUP	GHJ2
OUTSTAT	$$BATTNR	AWB5
PAGEPOOL	IJBSDUMP	EMD2
PARAMOO	$$BATTNZ	BKD3
PARCKRTN	$$BATTNQ	AVD1
PARINV	IJBSDUMP	EMD5
PARM	$$BOCRT2	RCA5
PARPA	$$BOCRT6	RMA2
PARTI	$$BOCRT3	RFA3
PART1	IJBSDUMP	EMB1
PAUSE	$$BATTNC	AJB1
PAUSE2	$$BATTNC	AJK1
PAVAIL	$$BOCRTC	KKD2
PAVAIL	$$BOCRTY	QFJ4
PCROUT1	$$BATTNT	BBJ5
PCROUT2	$$BATTNT	BBK5
PDEND	IJBSDUMP	ELG4
PDIB	IJBDMPPA	EQD3
PDUMP	$$BATTNV	BHB1
PDUMPR	IJBSDUMP	EBB1
PFBLTMSG	$$BATTF1	CCD4
PFCLEART	$$BATTN8	CAB2
PFDDNCUU	$$BATTF1	CCJ3
PFEOPR	$$BATTN8	CAF1
PFESMT	$$BATTF1	CCE2
PFESYNTX	$$BATTF1	CCB3
PFFETCH	$$BATTF0	GKF5
PFFORMS	$$BATTN8	CAE3
PFGETOPR	$$BATTN8	CBB1
PFKEYDPC	$$BATTF1	CCD3
PFLPI	$$BATTF0	GKF1
PFLPI	$$BATTN8	CAF3
PFLUBAD	$$BATTF0	GKH2
PFME	$$BATTF0	GKJ3
PFNKSCUU	$$BATTF1	CCG3
PFNKYVAL	$$BATTN8	CAH2
PFNUL	$$BATTF0	GKH1
PFNULMSG	$$BATTN8	CAK3
PFNXTOPR	$$BATTN8	CAC1
PFN1OPR	$$BATTN8	CBB2
PFN3211	$$BATTF1	CCJ1
PFOPR	$$BATTN8	CAD2
PFOPRFCP	$$BATTN8	CAG1
PFOPRFLD	$$BATTN8	CAB1
PFOPRINV	$$BATTF1	CCC4
PFPRMMVC	$$BATTN8	CBG1
PFPUBCNT	$$BATTF0	GKC4
PFRCOMBS	$$BATTF1	CCB1
PFSTMEND	$$BATTN8	CAK5
PFSTRPRM	$$BATTN8	CBD1
PFSUNIT	$$BATTF0	GKG2
PFUNIT	$$BATTF0	GKB2
PFWOFCUU	$$BATTF1	CCH3

Label	Phase	Location
PF1OPR	$$BATTN8	CAH3
PF2OPR	$$BATTN8	CAJ3
PGREGS	IJBSDUMP	END1
PNAVAIL	$$BOCRTC	KKH5
PNAVAIL	$$BOCRTY	QFG4
PNDUMP	IJBSDUMP	EKF2
POWCHK1	$$BATTF0	GKK2
POWER	$$BEOH4	DGE1
PPOCTTTT	$$BATTNS	AXF5
PREPUMSG	$$BATTF4	CDC2
PREPVMSG	$$BATTF2	GLG1
PREPVMSG	$$BATTF5	CFC2
PRESENT	$$BATTNC	AHD4
PRINT	IJBDMPIC	FFA3
PRINT	IJBDMPIO	FBB5
PRINTEND	$$BATTF2	GLE5
PRINTEND	$$BATTF3	GNF5
PRINTERR	$$BATTF4	CDG3
PRINTERR	$$BATTF5	CFD3
PRINTERR	$$BATTU2	CKJ2
PRINTO	IJBDMPIC	FFD5
PRINTO	IJBDMPIO	FBF5
PRINT1	$$BEOJ	DCD4
PRINT1	IJBDMPIO	FCB2
PRINT1	IJBEOJ	FRH4
PROCEDORT	$$BATTNS	AXD5
PROCKEY	$$BATTN7	CNB4
PROCMSG	$$BATTS2	CVH2
PROGCHCK	$$BOCRTA	KBF1
PROGCHCK	$$BOCRTZ	QKD5
PROTCHCK	$$BOCRTZ	QMC2
PROTCHK	$$BOCRTZ	QMB1
PROTCHK1	$$BOCRTZ	QME2
PROTCHK2	$$BOCRTZ	QMG1
PRTEND	IJBEOJ	FRG2
PRTPA	IJBSDUMP	EKA3
PRTPA	IJBSDUMP	EKC3
PRTPSW1	IJBEOJ	FQC1
PRTPSW2	IJBEOJ	FQF1
PRTY	$$BATTN2	BLB1
PRTYLOP	$$BATTN2	BMG1
PRTYSET	$$BATTN2	BMA1
PRTY05	$$BATTN2	BLD1
PRTY15	$$BATTN2	BLF1
PRTY20	$$BATTN2	BLA2
PRTY21	$$BATTN2	BLD2
PRTY30	$$BATTN2	BLA3
PRTY52	$$BATTN2	BMB1
PRTY56	$$BATTN2	BMD1
PRTY57	$$BATTN2	BMG2
PRTY58	$$BATTN2	BMB4
PRTY59	$$BATTN2	BMC5
PRVDMSGI	$$BOCRTP	NHA2
PUBFOUND	$$BATTS1	CSC4
PUBLOOP	$$BATTF4	CDF2
PUBLOOP	$$BATTF5	CFH2
PUBLOOP	$$BATTU2	CKC4
PUBLTMSG	$$BATTU1	CJF2
PUCLEART	$$BATTN9	CGB2
PUDDNCUU	$$BATTU1	CJG5
PUESMT	$$BATTU1	CJC4
PUFOLD	$$BATTN9	CGG3
PUGETOPR	$$BATTN9	CHB1
PUGOPR	$$BATTN9	CGF1
PUKEYDPC	$$BATTU1	CJE3
PUKEYWRD	$$BATTN9	CHJ2
PUNKSCUU	$$BATTU1	CJE5
PUNOCHK	$$BATTN9	CGH3
PUNULMSG	$$BATTN9	CGJ3
PUN1OPR	$$BATTN9	CHC2
PUOPR	$$BATTN9	CGD2
PUOPRFLD	$$BATTN9	CGB1
PUPRMMVC	$$BATTN9	CHG1
PURCOMBS	$$BATTU1	CJB1
PUSTRPRM	$$BATTN9	CHD1
PUTDATA	IJBDMPIC	FHB1
PUTMSG	$$BATTNT	BBB5
PUTMSG	IJBDMPIC	FFB3
PUTMSG1	IJBDMPIC	FFE3
PUTRAIN	$$BATTN9	CGC4
PUWOUCUU	$$BATTU1	CJF5
PU1OPR	$$BATTN9	CGE3
PU2OPR	$$BATTN9	CGE4
PU2OPR	$$BATTN9	CGF3
PVALID	IJBDMPAR	EUB1
QEX	$$BOCRTL	MMJ2
QUEUE	$$BEOJ3A	DEH2
RASSY	$$BCCHHR	GJB1
RASWAIT	$$BCCHHR	GJE4
RASWAIT	$$BEOJ4	DFJ2
RDSPLAY	$$BOCRTB	KEA3
REACHED	$$BOCRT3	RED4
READ	$$BATTNT	BAB3
READCMD	$$BOCRTC	KHD5
READCMD	$$BOCRTZ	QHD2
READDIR	$$BATTU2	CKF2
READM1	$$BOCRTS	PEB1
READRT	$$BOCRTR	PCB2
READRT	$$BOCRTS	PFB2
REAL370	$$BEOJ	DBD5
RECOG	$$BOCRTG	MAE2
RECOV2	$$BOCRTF	LGH4
REDSP	$$BOCRT4	RHA2
REPEAT	$$BOCRTR	PAA4
REPEAT	$$BOCRTY	QEC3
REPEAT0	$$BOCRTC	KGG1
REPEAT2	$$BOCRTC	KHD4
REQCANC	$$BOCRTX	QBF2
REQCANC0	$$BOCRTY	QDK3
REQPEND	$$BOCRTK	MLB4
REQPEND	$$BOCRTP	NHE2
REQPR	$$BOCRTK	MKF2
REQU	$$BOCRT1	RAB5
RESCODE	$$BEOJ	DBH2
RESET	$$BOCRTB	KDF1
RESET	$$BOCRTG	MBA3
RESET	IJBDMPIO	FCE2
RESETSW	$$BATTNT	BAD3
RESET1	$$BOCRTG	MAC4
RESEXT	$$BEOJ4	DHJ5
RESTK	$$BOCRTG	MAH1
RESTREGS	IJBDMPIO	FBE2
RETERR	$$BATTF4	CDJ3
RETERR	$$BATTF5	CFF3
RETERR	$$BATTU2	CKK3
RETN	$$BATTF5	CFJ5
RETOK	IJBDMPIC	FFH5
RETRY	$$BATTF2	GLJ3
RETRY	$$BATTF3	GNK3
RETRY	$$BATTF4	CDB3

Label	Phase	Location
RETRY	$$BATTF5	CFF4
RETRY	$$BOCRTS	PEA4
RETURN	$$BATTNS	AXC1
RETURN	$$BATTU2	CKJ5
RET1	$$BOCRT6	RLC5
RET2	$$BOCRT6	RMA3
RFCHECK	$$BEOH4	DGB1
RLCCB	$$BEOJ	DCK2
RLCCB	IJBEOJ	FRE2
RMSPOST	$$BCCHHR	GJF4
RPT	$$BOCRTN	NBH3
RSTMSG600	$$BRSTR2	SNG5
RSTPRT	$$BOCRTY	QEB3
RSTRTER1	$$BRSTRT	SLB5
RSTRT150	$$BRSTRT	SLH1
RSTRT200	$$BRSTRT	SLC3
RSTRT240	$$BRSTRT	SLD3
RSTRT300	$$BRSTRT	SLC4
RSTRT560	$$BRSTRT	SLG4
RSTRT600	$$BRSTRT	SLJ4
RSTRT700	$$BRSTRT	SLH5
RSTR0020	$$BRSTR2	SNC1
RSTR0140	$$BRSTR2	SND1
RSTR0180	$$BRSTR2	SNE1
RSTR0200	$$BRSTR2	SNF1
RSTR0300	$$BRSTR2	SNJ2
RSTR0400	$$BRSTR2	SNA3
RSTR0460	$$BRSTR2	SNC3
RSTR0480	$$BRSTR2	SNF3
RSTR0600	$$BRSTR2	SNA5
RSTR0660	$$BRSTR2	SNB5
RSTR0700	$$BRSTR2	SND5
RSTR0800	$$BRSTR2	SNE5
RSTR2000	$$BRSTR2	SNG3
RSTR2000	$$BRSTR2	SNK2
RSTR2000	$$BRSTR2	SPA1
RSTR2020	$$BRSTR2	SPC1
RSTR2060	$$BRSTR2	SPF2
RSTR2080	$$BRSTR2	SPB2
RSTR2100	$$BRSTR2	SPG2
RSTR2140	$$BRSTR2	SPJ2
RSTR2160	$$BRSTR2	SPK2
RSTR2500	$$BRSTR2	SNK3
RSTWRNG	$$BOCRTY	QEB2
RSTXECB	$$BEOJ4	DFB3
RTSTMT	$$BATTNA	ACB4
RUNREAL	$$BEOJ3A	DEJ3
SAVER3	$$BATTNT	BAF3
SAVID	$$BOCRT6	RME2
SCANBL1	$$BATTNA	ADD1
SCANBL2	$$BATTNA	ADE1
SCANOPN	$$BATTNA	ADB3
SCAN1	$$BATTNT	BBB3
SCIP	$$BOCRT4	RHG2
SCR	$$BOCRTL	MPB2
SCRPRT	$$BOCRTY	QEB1
SCR1	$$BOCRTL	MPC2
SDDUMP	IJBSDUMP	EKB2
SDEOV	IJBDMPIO	FCD1
SDUMPR	IJBSDUMP	ECA1
SDUMPR	IJBSDUMP	ECE1
SEARCH	$$BATTN7	CQC2
SEARCH	$$BOCRTB	KFB1
SEARCH	$$BOCRTC	KJB3
SEARCH	$$BOCRTD	LCB2
SEARCH	$$BOCRTG	MCB4
SECPA	$$BOCRT2	RDJ4
SEEIFMOR	IJBSDUMP	EHE4
SEGLIM	$$BOCRTQ	NJF1
SEGPASS	$$BOCRTN	NDA4
SENDMSG	$$BOCRTE	LEB3
SENDMSG	$$BOCRTF	LJB3
SENSECMD	$$BOCRTZ	QJA3
SENSE1	$$BOCRTZ	QJG3
SENSE2	$$BOCRTZ	QJB4
SETCOMRG	$$BOCRTA	KAA4
SETEXT	$$BATTNB	AFB5
SETFLAG	$$BOCRTA	KBG1
SETID	$$BATTND	AKC2
SETP	IJBDMPIO	FAD5
SETUP00	$$BATTNY	BJJ3
SETVAL	IJBDMPAR	EUG1
SHIFT	IJBDMPAR	EVK1
SIZE	$$BATTNF	APB1
SKIP	$$BOCRTP	NGH2
SKIP	$$BOCRTT	PGH3
SKIPBLNK	$$BOCRTK	MLF3
SKIPBLN1	$$BOCRTK	MLF4
SKIPLOAD	$$BATTF5	CFD4
SKIPRTY	$$BATTND	AKB2
SKIPTOE	$$BOCRTO	NFA3
SKIPTST	$$BOCRTC	KKJ2
SKIP1	$$BOCRTP	NGK2
SKIP1	$$BOCRTT	PGK3
SRCHLOOP	$$BEOJ3A	DEB4
START	$$BATTNT	BAB1
START	$$BSDRUP	GHB2
STARTE	$$BOCRTE	LDD1
STARTIO	$$BOCRTG	MAD3
STARTIO	IJBDMPIO	FCE4
STARTUP	$$BOCRT3	REA4
START1	$$BOCRT1	RAB1
STAT05	$$BEOJ7	DLC1
STAT10	$$BEOJ7	DLC2
STAT12	$$BEOJ7	DLF2
STAT15	$$BEOJ7	DLJ2
STAT35	$$BEOJ7	DLE5
STAT40	$$BEOJ7	DLK2
STCCW7	$$BATTNU	BCH5
STDUMP	$$BATTNV	BGB3
STEXCD	$$BATTNB	AFB1
STIO	$$BOCRTE	LFE1
STIO	$$BOCRTF	LKE2
STLNGTH	$$BATTF3	GNJ1
STORECCW	$$BOCRTA	KBE2
STORE1	IJBDMPIO	FAE5
STORE5	IJBDMPAR	EVD1
STRGEND	$$BOCRTC	KJC3
STRGEND	$$BOCRTD	LCC2
STRGEND	$$BOCRTG	MCC4
STRTIO	$$BOCRTE	LFB1
STRTIO	$$BOCRTF	LKB2
SUBLOOP	$$BATTNC	AHB2
SUBTASK	$$BEOH4	DGA2
SUPDUMP	IJBSDUMP	EFB1
SUPV	$$BATTND	AKD1
SVADMP	IJBSDUMP	EGB2
SVADUMP	IJBSDUMP	EJA3
SVAST	$$BATTNV	BFB4
SVC26	$$BOCRTZ	QMH5
SWAUTOCL	$$BEOJ4	DFE4

Label	Phase	Location
SWTCHOFF	IJBDMPAR	EVG2
SYSTMPD1	$$BOCRTK	MKD3
SYSTMRD	$$BOCRTK	MKA3
SYSWR	$$BSYSWR	GGB4
TAB1	$$BATTNV	BFE3
TAPEDEV	$$BEOJ	DCJ2
TAPEDEV	IJBEOJ	FRD2
TAPEIO	IJBDMPIC	FHC2
TAPE00	$$BEOJ7	DNA3
TAR	IJBSDUMP	EAG2
TASK	$$BEOJ3A	DEC5
TBIT	IJBSDUMP	EHH1
TDISK	IJBDMPIO	PCB3
TEA	IJBDMPAR	ETC2
TEND	IJBDMPAR	ESG4
TEND1	IJBDMPAR	EVC4
TERM	IJBEOJ	FPJ5
TERMCANC	$$BEOJ	DAD4
TEST	$$BOCRTC	KKB1
TEST	$$BOCRTY	QFB3
TESTAP	IJBEOJ	FPC4
TESTBRST	$$BATTS1	CTB3
TESTCMD	$$BOCRTZ	QLB2
TESTCNT	$$BOCRTZ	QKD4
TESTERR	$$BOCRTA	KBA2
TESTIO	$$BOCRTE	LFH1
TESTIO	$$BOCRTF	LKH2
TESTLPI	$$BATTF5	CFE2
TESTMAX	IJBSDUMP	EGH2
TESTNUSD	$$BOCRTZ	QKB4
TEST02	$$BOCRTC	KKG3
TEST02	$$BOCRTY	QFE3
TFUL	IJBDMPIC	FGD5
TICCHECK	$$BOCRTA	KAG4
TICCHK	$$BOCRTR	PBB2
TICERROR	$$BOCRTA	KAG5
TICRT	$$BOCRTA	KCA4
TICSKIP	$$BOCRTZ	QLB3
TM	$$BEOJ3A	DEB2
TMPAR1	$$BATTNA	AAC2
TOL	$$BATTNV	BEG3
TPBAL	$$BATTN2	BNB1
TPTYPE	$$BEOJ	DCG2
TPTYPE	IJBEOJ	FRB2
TPTYPE1	$$BEOJ	DCH2
TPTYPE1	IJBEOJ	FRC2
TRC	IJBDMPAR	EWD1
TRC	IJBDMPPA	ERJ4
TREAL	IJBDMPAR	ETJ2
TRKFBA	$$BEOJ4	DHD4
TRSFR	$$BOCRTD	LAD1
TRSFR1	$$BOCRTD	LAE1
TRSFR2	$$BOCRTD	LAF1
TRSFR3	$$BOCRTD	LAF3
TRYNXT	$$BATTS1	CSD2
TSAME	IJBDMPAR	EVC2
TSTADDR	IJBDMPAR	ESB2
TSTBJF	IJBEOJ	FRJ1
TSTBUFF	$$BATTF2	GLA3
TSTBUFF	$$BATTF5	CFF2
TSTEADDR	IJBDMPAR	ETB2
TSTJBF	$$BEOJ	DCD2
TWOADUMP	IJBSDUMP	ELB1
TWORDS	IJBDMPAR	EVB1
TYPEONE	$$BATTN7	CQB2
UI	$$BOCRT3	RFF3
UNAV	$$BATTNG	ASJ5
UNRECOV	$$BOCRTF	LJA1
UN1	$$BOCRTX	QBC5
UPD	$$BOCRTB	KFC2
UPDATE	$$BOCRT3	REE3
UPDATER	IJBSDUMP	EKG2
UPDATE1	$$BOCRTQ	NKD1
UPDA1	$$BOCRT3	REG3
VALBB00	$$BATTNY	BJB5
VALCUU00	$$BATTNY	BJE1
VALIDATE	IJBDMPAR	ESC4
VALMOD05	$$BATTNY	BJF3
VALNOD00	$$BATTNY	BJC3
VERIMSG	$$BATTU2	CKF5
VLOOP	IJBSDUMP	EMB3
VOLUME62	$$BATTNO	AUH2
VRBL1	$$BOCRTN	NBA4
VTAMHOOK	$$BATTNH	ATB4
WAIT1	$$BEOJ	DCE4
WAIT1	$$BOCRTR	PBC1
WAIT1	IJBEOJ	FRJ4
WARN	$$BOCRTB	KDH4
WBITEM	$$BOCRTS	PFH3
WRITECMD	$$BOCRTZ	QHE4
WRITEM	$$BOCRTS	PFB3
WRITE0	$$BOCRT1	RAC2
WRITE20	$$BOCRTH	MFE2
WRITS	$$BOCRT4	RHB3
WRTINSTL	$$BOCRTI	MGF3
WRTWRNG	$$BOCRTR	PCB3
WTE	$$BOCRT1	RAG1
WTE1	$$BOCRT1	RAA3
XFR	$$BOCRTC	KHC2
XFR1	$$BOCRTC	KGD3
YESFOLD	$$BATTU2	CKB2

DOS/VSE Logical Transients and Dump Phases Logic Printed in U.S.A. SY33-8553-4

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601

READER'S COMMENT FORM

This sheet is for comments and suggestions about this manual. We would appreciate *your* views, favorable or unfavorable, in order to aid us in improving *this* publication. This form will be sent directly to the author's department. Please include your name and address if you wish a reply. Contact your IBM branch office for answers to technical questions about the system or when requesting additional publications. Thank you.

Name

Address

What is your occupation?

How did you use this manual?

As a reference source

As a classroom text

As a self-study text

Your comments* and suggestions:

*** We would especially appreciate your comments on any of the following topics:**

Clarity of the text	Accuracy	Index	Illustrations	Appearance	Paper
Organization of the text	Cross-references	Tables	Examples	Printing	Binding

YOUR COMMENTS, PLEASE . . .

This manual is part of a library that serves as a reference source for system analysts, programmers and operators of IBM systems. Your answers to the questions on the back of this form, together with your comments, will help us produce better publications for your use. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Please note: Requests for copies of publications and for assistance in utilizing your IBM system should be directed to your IBM representative or to the IBM sales office serving your locality.

Fold

Fold

FIRST CLASS
PERMIT NO. 1359
WHITE PLAINS, N. Y.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . .

IBM Corporation
1133 Westchester Avenue
White Plains, N.Y. 10604

Attention: Department 813 BP

Fold

Fold

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue. White Plains, N.Y., U.S.A. 10601

CUT ALONG THIS LINE

DOS/VSE Logical Transients and Dump Phases Logic Printed in U.S.A. SY33-8553-4