File Number S360-36 (DOS)
Order No. GY24-5152-1

Program Logic

DOS Logical Transients

Program Number 360N-CL-453

This reference publication describes the internal logic of the IBM System/360 Disk Operating System, Logical Transient Programs. It is for persons involved in program maintenance and for system programmers who are altering the program design. Program logic information is not needed for normal operation of these programs. This publication is a supplement to the program listing.

Effective use of this manual requires an understanding of IBM System/360 operation and of IBM System/360 Disk Operating System control and service programs, macro instructions, and operating procedures. Reference publications for this information are listed in the Preface.

DOS Release 25

Second Edition (April 1971)

This publication was formerly titled IBM System/360 Disk Operating System, Logical Transient Programs. Although titles of some DOS publications (including this one) have been simplified, the change does not affect the contents of the publications. This edition, GY24-5152-1, is a major revision of, and obsoletes, GY24-5152-0.

This edition applies to Release 25 of IBM System/360 Disk Operating System and to all subsequent releases until otherwise indicated in new editions or Technical Newsletters. Changes are continually made to the specifications herein; before using this publication in connection with the operation of IBM systems, consult the latest System/360 and System/370 SRL Newsletter, GN20-0360, for the editions that are applicable and current.

Summary of Amendments

This edition documents the addition of the following information: Private Core Image Library, System/370 MODE Command and Recovery Management Support, Job Accounting Interface, OLTEP (On-Line Test Executive Program), Data Set Security, and small maintenance enhancements.

Changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

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 reader's comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Programming Publications, Department G60, P. O. Box 6, Endicott, New York 13760. Comments become the property of IBM.

This Program Logic Manual (PLM) is a detailed guide to the IBM System/360 Disk Operating System 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, prerequisite, and related publications.

Note: Although titles of some DOS publications have been simplified, the change does not affect the contents of the publications.

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

- Introduction to DOS Logic, GY24-5017.
- DOS Librarian, GY24-5079.
- DOS Linkage Editor, GY24-5080.
- DOS IPL and Job Control Programs, GY24-5086.
- DOS Supervisor and Related Transients, GY24-5151.
- DOS System Service Programs, GY24-5153.

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

- IBM System/360 Principles of Operation, GA22-6821.
- DOS System Control and Service, GC24-5036.
- IBM System/360 Disk and Tape Operating Systems, Assembler Language, GC24-3414.

Publications related in subject matter to the seven system control PLMs are:

- DOS Supervisor and I/O Macros, GC24-5037.
- DOS System Generation, GC24-5033.
- DOS Operating Guide, GC24-5022.
- DOS Messages, GC24-5074.
- DOS Data Management Concepts, GC24-3427.

Titles and abstracts of other related publications are listed in the IBM System/360 and System/370 Bibliography, GA22-6822.

This manual consists of four major sections. The first section discusses the logical transient functions covered in this manual. The second section describes the supervisor calls frequently used by logical transients. The third section details the internal logic flow. The last section is comprised of appendixes with label lists, error messages, supplemental figures, microfiche listings, and other references for program analysis.

The flowchart symbols used in this manual conform with the flowcharting standards of the American National Standards Institute, Inc. Numerals, such as 00, identify the program or general level flowcharts. The detailed flowcharts are identified by letters AA through ZZ. Please refer to Appendix F for an explanation of these flowchart symbols.

LOGICAL TRANSIENT PROGRAMS 11
B-Transient Grouping 11
Transient Attention Routines (Charts 01-04) 11
Program Initiator (Charts 02-07) . . . 13
Terminator (Charts 08-15) 13

LOGICAL TRANSIENT SUPERVISOR CALLS . . . 21
Supervisor Calls 21

CHARTS 25

APPENDIX A: LABEL LIST 186

APPENDIX B: ERROR MESSAGE CROSS REFERENCE 200

APPENDIX C: SUPERVISOR REFERENCE FIGURES 202

APPENDIX D: LABEL INFORMATION FORMAT ON SYSRES 232

APPENDIX E: PROGRAM KEY DEFINITIONS . .234
PID (Partition Identifier) 234
PIK (Program Interrupt Key) 234
LID (Logical Transient Identification) 234
LTK (Logical Transient Key) 234
RID (Requestor Identification) . . .234
RIK (Requestor I/O Key) 235
FIK (Fetch I/O Key) 235

APPENDIX F: EXPLANATION OF FLOWCHART SYMBOLS 236

APPENDIX G: MICROFICHE CROSS-REFERENCE INDEX 237

GLOSSARY 238

INDEX 241

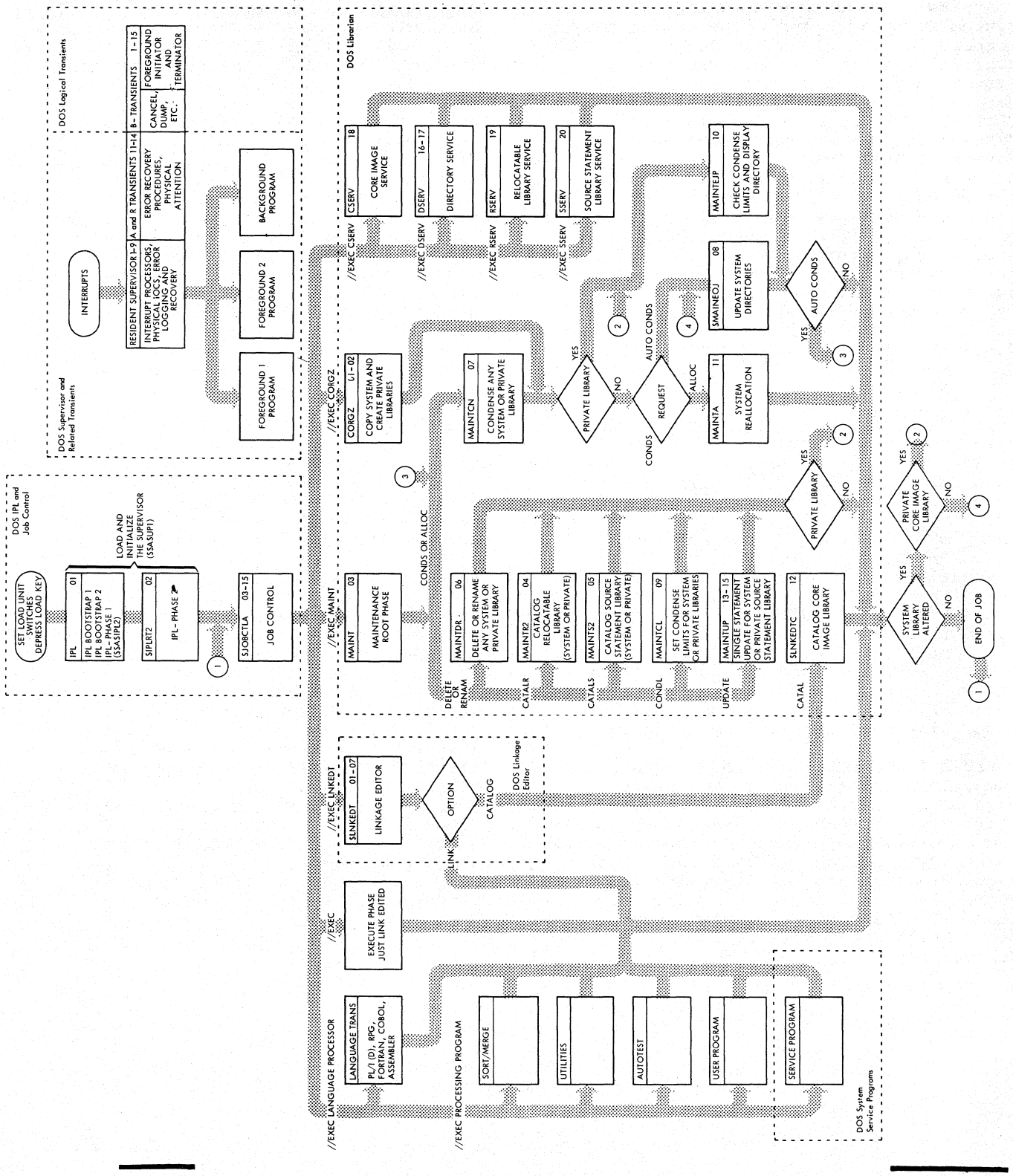
Figure 1. Supervisor Storage Allocation 12
Figure 2. Cancel Codes and Messages . 15
Figure 3. MAP Output 16
Figure 4. Interphase Communication Area (For A-Transients $$ANERRZ, Y, and 0) 16
Figure 5. List I/O Examples 17
Figure 6. Terminator Phases (Part 1 of 3) 18
Figure 7. DOS Supervisor Calls (Part 1 of 2) 23
Figure 8. I/O Table Interrelationship 202
Figure 9. Supervisor Communications Region (Part 1 of 5) 203
Figure 10. Communications Region Extensions 208
Figure 11. Tables for MICR DTF Addresses and Pointers 209
Figure 12. Disk Information Block (DIB) Table 210
Figure 13. Error Recovery Block (ERBLOC) and Error Queue Entry 211
Figure 14. Option Tables 212
Figure 15. Second Part of Program Information Block (PIB) Table 213
Figure 16. First Part of Program Information Block (PIB) Table 214
Figure 17. PIB Flag Expansions 215
Figure 18. CHANQ, LUBID, REQID, LUBDSP, and TKREQID Tables 216
Figure 19. Physical Unit Block (PUB) Table 217
Figure 20. Job Information Block (JIB) Table 218
Figure 21. Tape Error Block (TEB) Table 219
Figure 22. TEBV Table Showing Status Block and Error Blocks (Part 1 of 2) .220
Figure 23. Console Buffering (CBTAB) Table and Work Areas 222
Figure 24. Logical Unit Block (LUB) Table 223
Figure 25. Track Hold (TKHDTAB) Table 224
Figure 26. Device Type Codes 225
Figure 27. Density Data 226
Figure 28. Example of the CHANQ Table Operation 227
Figure 29. Task Selection Procedure .227
Figure 30. Job Accounting Interface Common Table (ACCTCOMN) 228
Figure 31. Job Accounting Interface Partition Table (ACCTxx*) 229
Figure 32. RMS Monitor Table (RASTAB) (Part 1 of 2) 230
Figure 33. Format of SYSRES Tape Label Information 232
Figure 34. Format of SYSRES DASD Label Information 233

Chart 00. Disk Operating System Program Flow 10
Chart 01. Overview of Supervisor Entry into B-Transients 25
Chart 02. Logical Transient Root Phase 26
Chart 03. Logical Transient Attention Routines (Part 1 of 2) 27
Chart 04. Logical Transient Attention Routines (Part 2 of 2) 28
Chart 05. Logical Transient Initiator (Part 1 of 2) 29
Chart 06. Logical Transient Initiator (Part 2 of 2) 30
Chart 07. MODE Command Processor . . . 31
Chart 08. Logical Transient Terminator (Part 1 of 8) 32
Chart 09. Logical Transient Terminator (Part 2 of 8) 33
Chart 10. Logical Transient Terminator (Part 3 of 8) 34
Chart 11. Logical Transient Terminator (Part 4 of 8) 35
Chart 12. Logical Transient Terminator (Part 5 of 8) 36
Chart 13. Logical Transient Terminator (Part 6 of 8) 37
Chart 14. Logical Transient Terminator (Part 7 of 8) 38
Chart 15. Logical Transient Terminator (Part 8 of 8) 39
Chart AA. $$BATTNA - Nonresident Attention/Initiator Root Phase 40
Chart AB. $$BATTNA - Control Routine . 41
Chart AC. $$BATTNA - Root Phase Subroutines 42
Chart AD. $$BATTNA - General Scan Routines 43
Chart AE. $$BATTNB - MSG Statement Processor 44
Chart AF. $$BATTNB - Set Operator Communications and Exit Table Linkage . 45
Chart AG. $$BATTNC - CANCEL Statement Processor (Part 1 of 2) 46
Chart AH. $$BATTNC - CANCEL Statement Processor (Part 2 of 2) 47
Chart AJ. $$BATTNC - PAUSE, LOG, NOLOG, and IGNORE Statement Processors . 48
Chart AK. $$BATTND - MAP Statement Processor 49
Chart AL. $$BATTND - Output MAP Subroutines (Part 1 of 2) 50
Chart AM. $$BATTND - Output MAP Subroutines (Part 2 of 2) 51
Chart BA. $$BATTNE - ALLOC Statement Processor (Part 1 of 4) 52
Chart BB. $$BATTNE - ALLOC Statement Processor (Part 2 of 4) 53
Chart BC. $$BATTNF - ALLOC Statement Processor (Part 3 of 4) 54
Chart BD. $$BATTNF - ALLOC Statement Processor (Part 4 of 4) 55
Chart BE. $$BATTNG - START and BATCH Statement Processors 56
Chart BF. $$BATTNG - START and BATCH Subroutines 57
Chart BG. $$BATTNH - START Statement Processor Channel Program 58
Chart BH. $$BATTNH - Subroutines . . . 59
Chart CA. $$BATTNI - ASSGN Statement Processor (Part 1 of 2) 60
Chart CB. $$BATTNI - ASSGN Statement Processor (Part 2 of 2) 61
Chart CC. $$BATTNI - Validate SYSXXX Subroutine 62
Chart CD. $$BATTNI - Validity Check Channel and Unit 63
Chart CE. $$BATTNI - Unassign Subroutine 64
Chart CF. $$BATTNI - Scan LUBs and JIBs Subroutines 65
Chart CG. $$BATTNI - Conversion Subroutines 66
Chart CH. $$BATTNI - UNA Statement Processor 67
Chart CJ. $$BATTNI - Miscellaneous Subroutines 68
Chart CK. $$BATTNJ - LISTIO Statement Processor 69
Chart CL. $$BATTNJ - Subroutines . . . 70
Chart CM. $$BATTNJ - Locate Assignment Routine 71
Chart CN. $$BATTNJ - Output List (Part 1 of 2) 72
Chart CP. $$BATTNJ - Output List (Part 2 of 2) 73
Chart DA. $$BATTNK - VOL Statement Processor 74
Chart DB. $$BATTNK - TPLAB Statement Processor 75
Chart DC. $$BATTNK - TLBL Statement Processor 76
Chart DD. $$BATTNK - Check, Convert, and Concatenate Subroutines 77
Chart DE. $$BATTNK - Process File ID and Date Operands 78
Chart DF. $$BATTNK - DLBL Statement Processor (Part 1 of 2) 79
Chart DG. $$BATTNK - DLBL Statement Processor (Part 2 of 2) 80
Chart DH. $$BATTNK - Output Label Data Subroutines 81
Chart DJ. $$BATTNL - DLAB Statement Processor 82
Chart DK. $$BATTNL - Extract Operand Routine 83
Chart DL. $$BATTNL - XTENT Statement Processor (Part 1 of 3) 84
Chart DM. $$BATTNL - XTENT Statement Processor (Part 2 of 3) 85
Chart DN. $$BATTNL - XTENT Statement Processor (Part 3 of 3) 86
Chart DP. $$BATTNL - XTENT Processor Subroutines 87

Chart EA. $$BATTNM - EXEC Statement Processor 88
Chart EB. $$BATTNM - Output Last Block of Label Data 89
Chart EC. $$BATTNM - Move Last Block Routine 90
Chart ED. $$BATTNM - Move Subroutine and Initialize for FG Program Load Routine 91
Chart EE. $$BATTNM - UCS Statement Processor 92
Chart EF. $$BATTNM - UCS Subroutines . 93
Chart EG. $$BATTNN - TIMER Statement Processor 94
Chart EH. $$BATTNO - EXTENT Statement Processor (Part 1 of 3) 95
Chart EJ. $$BATTNO - EXTENT Statement Processor (Part 2 of 3) 96
Chart EK. $$BATTNO - EXTENT Statement Processor (Part 3 of 3) 97
Chart EL. $$BATTNO - EXTENT Processor Subroutines 98
Chart EM. $$BATTNO - Process Track Operands 99
Chart EN. $$BATTNO - LBLTYP Statement Processor 100
Chart EP. $$BATTNP - READ Statement Processor 101
Chart EQ. $$BATTNP - HOLD or RELSE Statement Processor 102
Chart FA. $$BEOJ - Terminated Program I/O Handling and EOJ Processing (Part 1 of 3) 103
Chart FB. $$BEOJ - Terminated Program I/O Handling and EOJ Processing (Part 2 of 3) 104
Chart FC. $$BEOJ - Terminated Program I/O Handling and EOJ Processing (Part 3 of 3) 105
Chart FD. $$BEOJ - Message Output Subroutine and Zero Option Table Subroutine 106
Chart FE. $$BEOJ1 - Prepare Cancel Cause Message 107
Chart FF. $$BEOJ1 - Output Cancel Message on SYSLST 108
Chart FG. $$BEOJ2 - Select Cancel Message and Program/Task Identification 109
Chart FH. $$BEOJ2 - Select I/O Device and Output the Cancel Message 110
Chart FJ. $$BEOJ2A - Select Cancel Message and Program/Task Identification 111
Chart FK. $$BEOJ2A - Select I/O Device and Output the Cancel Message . .112
Chart FL. $$BEOJ3 - Quiesce I/O for TP Devices 113
Chart FM. $$BEOJ4 - Quiesce I/O for Non-TP Devices 114
Chart FN. $$BEOJ5 - Release Tracks Held by Task/Partition 115
Chart GA. $$BESTVA - Phase 1 of Tape Volume Error Statistics 116
Chart GB. $$BESTVB - Phase 2 of Tape Volume Error Statistics 117
Chart GC. $$BESTVC - Phase 3 of Tape Volume Error Statistics 118
Chart GD. $$BESTVD - Phase 4 of Tape Volume Error Statistics (Part 1 of 2) .119
Chart GE. $$BESTVD - Phase 4 of Tape Volume Error Statistics (Part 2 of 2) .120
Chart GF. $$BESTVE - Phase 5 of Tape Volume Error Statistics 121
Chart GG. $$BESTVF - Phase 6 of Tape Volume Error Statistics 122
Chart HA. $$BTERM - Reset Foreground PUB Ownership and Detach Attention Routine 123
Chart HB. $$BTERM - Reset JIBs for I/O Device of Terminated Program 124
Chart HC. $$BTERM - Get TEB Statistics and Reset TEBs 125
Chart HD. $$BTERM - Print Message and TEB Statistics Subroutine 126
Chart JA. $$BILSVC - Prepare Information about Cancel Cause 127
Chart JB. $$BILSVC - Select I/O Device and Prepare to Output a Message .128
Chart JC. $$BILSVC - Message Initialization and Output Subroutines .129
Chart JD. $$BPSW - Prepare Canceled Program's PSW for Output Message and PIOCS Subroutine 130
Chart JE. $$BPSW - Select I/O Device, and Prepare to Output a Message 131
Chart JF. $$BPCHK - Prepare Information for Message about PC Cancel and Select I/O Device 132
Chart JG. $$BPCHK - Set Up for I/O and Output the Message 133
Chart KA. $$BDUMP - Translating System Dump, Monitor Background/Foreground Program Dump (Part 1 of 3) 134
Chart KB. $$BDUMP - Translating System Dump, Monitor Background/Foreground Program Dump (Part 2 of 3) 135
Chart KC. $$BDUMP - Translating System Dump, Monitor Background/Foreground Program Dump (Part 3 of 3) 136
Chart KD. $$BDUMPB - Translating System Dump, Background/Foreground Dump on Printer or Tape (Part 1 of 2) .137
Chart KE. $$BDUMPB - Translating System Dump, Background/Foreground Dump on Printer or Tape (Part 2 of 2) .138
Chart KF. $$BDUMPB - Translating System Dump, Reset Storage Print Routine and Edit a Line Subroutines . .139
Chart KG. $$BDUMPB - Translating System Dump, Subroutine to Edit and Print a Line 140
Chart KH. $$BDUMPB - Translating System Dump, Edit and Print a Line and Prepare Page Headings Subroutines . . .141
Chart KJ. $$BDUMPD - Translating System Dump, Background/Foreground Dump on Disk (Part 1 of 2) 142
Chart KK. $$BDUMPD - Translating System Dump, Background/Foreground Dump on Disk (Part 2 of 2) 143
Chart KL. $$BDUMPD - Translating System Dump, Reset Storage Print Routine and Edit a Line Subroutines . .144

Chart KM. $$BDUMPD - Translating System Dump, Subroutines to Edit and Print a Line 145
Chart KN. $$BDUMPD - Translating System Dump, Edit and Write a Line and Prepare Page Headings Subroutines . . .146
Chart LA. $$BPDUMP - Translating System Dump, Background/Foreground Parameter Dump (Part 1 of 3) 147
Chart LB. $$BPDUMP - Translating System Dump, Background/Foreground Parameter Dump (Part 2 of 3) 148
Chart LC. $$BPDUMP - Translating System Dump, Background/Foreground Parameter Dump (Part 3 of 3) 149
Chart LD. $$BPDUMP - Translating System Dump, Subroutine to Edit and Print a Line 150
Chart LE. $$BPDUMP - Translating System Dump, Edit and Print a Line and Prepare Page Headings Subroutines . . .151
Chart LF. $$BDUMP - Standard System Dump, Monitor Background Program Dump .152
Chart LG. $$BDUMP - Standard System Dump, Monitor Foreground Program Dump .153
Chart MA. $$BDUMPF - Standard System Dump, Foreground Program Dump (Part 1 of 2) 154
Chart MB. $$BDUMPF - Standard System Dump, Foreground Program Dump (Part 2 of 2) 155
Chart MC. $$BDUMPF - Standard System Dump, Prepare Page Headings and PIOCS Subroutines 156
Chart MD. $$BDUMPF - Standard System Dump, Prepare and Edit a Line Subroutine 157
Chart ME. $$BDUMPF - Standard System Dump, Line Test Subroutines (Part 1 of 2) 158
Chart MF. $$BDUMPF - Standard System Dump, Line Test Subroutines (Part 2 of 2) 159
Chart MG. $$BDUMPB - Standard System Dump, Initialization for BG Storage Dump on Printer or Tape 160
Chart MH. $$BDUMPB - Standard System Dump, BG Dump on Printer or Tape 161
Chart MJ. $$BDUMPB - Standard System Dump, Prepare Page Headings and PIOCS Subroutines 162
Chart MK. $$BDUMPB - Standard System Dump, Prepare and Edit a Line Subroutine 163
Chart ML. $$BDUMPB - Standard System Dump, Line Test Subroutines 164
Chart MM. $$BDUMPD - Standard System Dump, Dump on Disk Device (Part 1 of 2) 165
Chart MN. $$BDUMPD - Standard System Dump, Dump on Disk Device (Part 2 of 2) 166
Chart MP. $$BDUMPD - Standard System Dump, Prepare Page Headings and PIOCS Subroutines 167
Chart MQ. $$BDUMPD - Standard System Dump, Prepare and Edit a Line Subroutine 168
Chart MR. $$BDUMPD - Standard System Dump, Line Test Subroutines 169
Chart NA. $$BPDUMP - Standard System Dump, Parameter Storage Dump Monitor . .170
Chart NB. $$BPDUM1 - Standard System Dump, Initialize Parameter Dump on Printer or Tape 171
Chart NC. $$BPDUM1 - Standard System Dump, Parameter Storage Dump on Printer or Tape 172
Chart ND. $$BPDUM1 - Standard System Dump, Line Test Subroutines 173
Chart NE. $$BPDUM1 - Standard System Dump, Prepare and Edit a Line Subroutine 174
Chart NF. $$BSYSWR - Set up a Write on SYSRES Operation; Move Label Cylinder Address to COMREG 175
Chart NG. $$BSDRUP - EREP Processing, Suppress File-Ready-for-Recording Switch 176
Chart NH. $$BCCHHR - Core Image Directory Scan (Part 1 of 3) 177
Chart NJ. $$BCCHHR - Core Image Directory Scan (Part 2 of 3) 178
Chart NK. $$BCCHHR - Core Image Directory Scan (Part 3 of 3) 179
Chart PA. $$BATTNQ - MODE Command Parameter Processor (Part 1 of 2) . . .180
Chart PB. $$BATTNQ - MODE Command Parameter Processor (Part 2 of 2) . . .181
Chart PC. $$BATTNR - MODE Command Status Report Processor (Part 1 of 2) .182
Chart PD. $$BATTNR - MODE Command Status Report Processor (Part 2 of 2) .183
Chart PE. $$BATTNS - MODE Command Validity Checker (Part 1 of 2) 184
Chart PF. $$BATTNS - MODE Command Validity Checker (Part 2 of 2) 185

Chart 00. Disk Operating System Program Flow



Logical transient programs, also referred to as B-transients, are not resident in main storage. These transients are loaded or fetched from the core image library into the 1200-byte area of the supervisor called the Logical Transient Area (LTA).

The physical organization of the supervisor, including the location of the LTA, depends on the supervisor options specified at system generation. Figure 1, which shows the physical layout of the supervisor in core, is a main storage map of the assembled supervisor.

An SVC 2 instruction loads and executes a B-transient phase. A prefix of $$B to the name of a phase identifies it as a B-transient. The normal return to supervisor nucleus control 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 name before the SVC 2 is issued. The fetch or load routine, then, has access to the name for searching the disk directories or tape records for the desired transient. Refer to Logical Transient Supervisor Calls section for a more detailed explanation.

B-TRANSIENT GROUPING

The supervisor B-transient programs can be grouped by the various functions performed. These functions are: transient attention routine, program initiator, and program terminator.

TRANSIENT ATTENTION ROUTINES (CHARTS 01-04)

This group of B-Transients consists of $$BATTNA-$$BATTNH and $$BATTNN.

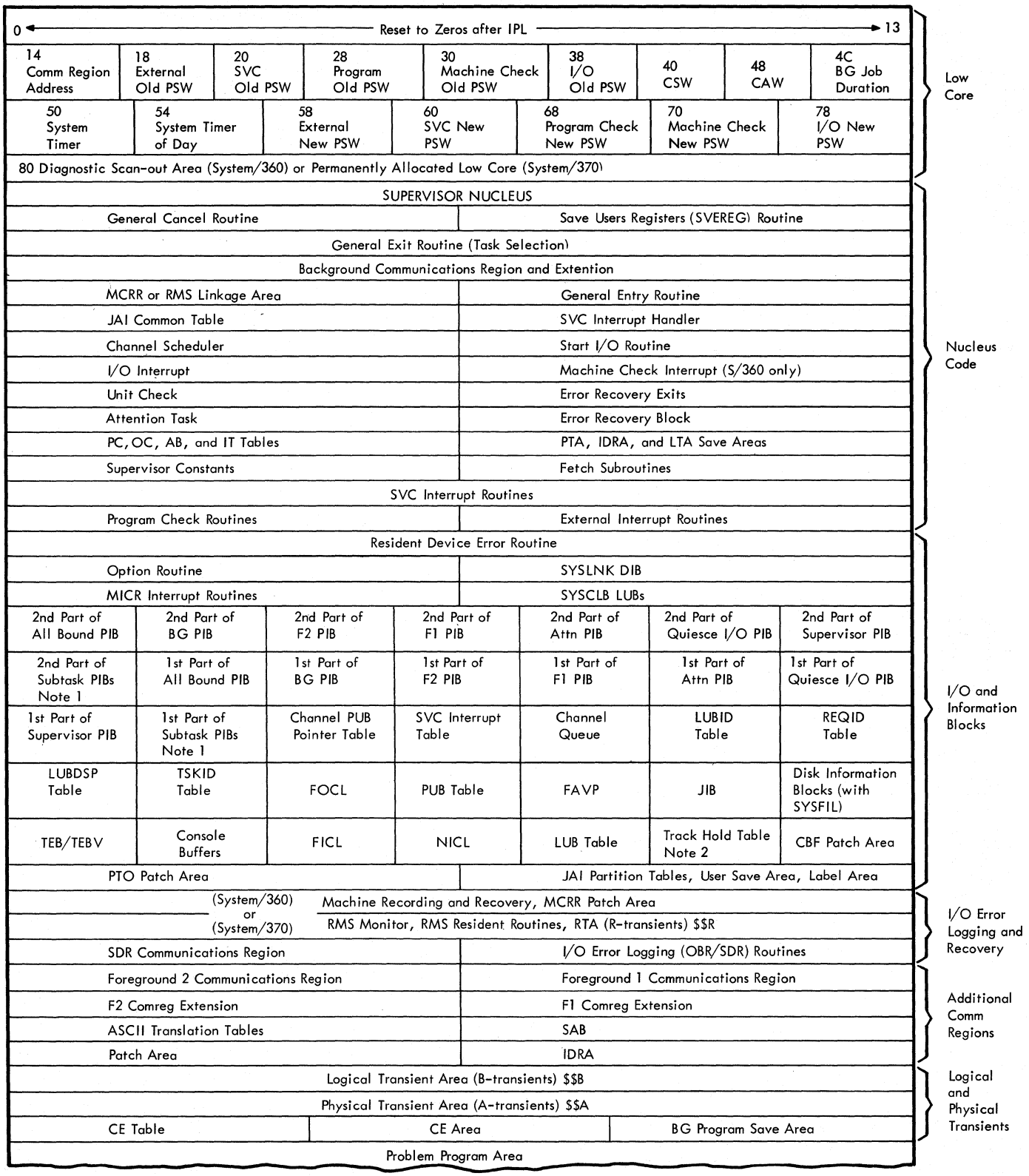
Attention commands are submitted when the operator presses the request key on the 1052 keyboard. The system's attention transient routine ($$BATTNA) is loaded, and this calls $$BATTNB which issues the message READY FOR COMMUNICATIONS. It then reads input statement information and selects the appropriate statement processor.

Commands accepted by the nonresident attention routines are:

- 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.
- CANCEL: Indicates one of the programs in the system is to be canceled. See Figure 2 for cancel code information.
- MAP: Provides a map of main-storage utilization. See Figure 3.
- ALLOC: Permits the operator to allocate storage among foreground and background programs.
- MSG: Causes control to be given to a foreground program operator communications routine previously activated by a STXIT command.
- TIMER: Causes interval timer support to be given to the program specified.
- START: Indicates the foreground initiation function has begun.
- BATCH: Initiates a dormant background or batched foreground area.
- LOG, NOLOG: Permits or suppresses logging of job control and single program statements on SYSLOG.
- IGNORE: Permits input from SYSRDR after a READ is issued.
- MODE: Controls error threshold values and requests status information in the MCAR/CCH function of System/370 RMS (Recovery Management Support).

Note: If the operator has pressed the request key to satisfy an operator intervention condition or to cancel the job, the physical attention transients ($$ANERRZ, Y, and 0) process the attention interrupt.

When the physical attention routines are processing the interrupt, they perform parameter passing by using a common area called the interphase communications area. Figure 4 shows this area and its relationship to the entire A-transient area.



Note 1: Total of 9 subtasks PIBs generated.
Note 2: Maximum of 225 entries generated.

Figure 1. Supervisor Storage Allocation

PROGRAM INITIATOR (CHARTS 02-07)

This group of B-Transients consists of $$BATTNA, $$BATTNC, $$BATTNI-$$BATTNM, and $$BATTNO-$$BATTNS.

Single foreground programs are initiated by the operator through the 1052 assigned to SYSLOG. The operator may initiate a single program whenever an allocated foreground area does not contain a program.

The operator initiates a single program by pressing the 1052 request key. The attention interrupt causes control to be given to the system's Attention routine.

> Note: If the transient area is in use by a routine other than the Attention routine, the attention interrupt is posted and serviced when the transient area becomes available.

The Attention routine reads a command from the operator. The command START (F1 or F2) indicates a single program is to be initiated. The Attention routine determines if the area specified is allocated and does not contain a program. If so, it transfers control to the single program initiator; otherwise, the operator is notified that an invalid command has been given.

The single program initiator reads subsequent commands required to initiate the program. These commands are used primarily to specify I/O assignments and label information. When an I/O assignment is attempted, the following verification is made:

1. The symbolic unit is a valid logical unit.
2. The symbolic unit is contained within the number specified for the area at system generation.
3. If the symbolic unit is to be assigned to a non-DASD, the device must not be in use by the other foreground program nor can it be assigned to a background job either as a standard, temporary, or alternate unit.

Figure 5 illustrates a LISTIO example.

The label information for each file in the job is written on SYSRES as a label information block for later retrieval and processing by the data management routines. A main storage area for label information is required under the same conditions as for background jobs, and is calculated and reserved by the initiator for self-relocating foreground programs. For nonrelocatable foreground programs, the label information area is determined by the LBLTYP statement.

When the EXEC statement is encountered, the initiator directs the supervisor to provide loading information for the program to be invoked. If the program has not been cataloged, the operator is notified by the initiator. He may correct the command (for example, if the name was misspelled) or cancel the initiation.

After the loading information is received, the initiator checks the load address to determine if it is zero, which indicates that a self-relocating program is to be loaded. The initiator sets up the load address so that the program will be loaded following the label information area. It also calculates the entry point to the program by adding the address at which it will be loaded to the previously-calculated entry point (derived when the program was linkage edited and cataloged onto the system). If a nonrelocatable program is loaded, the information used is that derived when the program was cataloged.

Diagnostics, such as the program being outside the limits of the foreground area, are not performed by the initiator, but are performed by the supervisor when the program is loaded. The supervisor then causes the program to be terminated.

When initial control is given to the foreground program, register 2 contains the address of the uppermost byte of storage available to this program. This may be used to calculate the total storage available to the program. A foreground program can dynamically determine the storage available to it by storing the contents of this register for later reference.

Note that a program capable of either foreground or background operation (with proper linkage editing) can utilize the same programming to determine its storage allocation independently of its actual area assignment.

TERMINATOR (CHARTS 08-15)

A single 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 single program is terminated, the following actions are taken:

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

2. Tape error statistics for all tape drives assigned to the program being terminated, and on which an error has occurred, are logged out on SYSLOG. Tape error statistics for all tape volumes assigned to the program being terminated can be logged on SYSLOG or stored on a disk file, depending on the user option chosen at system generation time. The statistic counters are reset. These features are system generation options.

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

4. All I/O assignments made for the program are canceled so that these devices will be available to subsequent programs. The assignments are not canceled if they are to be held across jobs by the HOLD command.

5. The operator is notified that the program is completed. The storage used by the program remains allocated for the foreground area.

6. The program is detached from the system's task selection mechanism.

See Figure 6 for an overall view of the terminator phases.

After a foreground program is completed, the operator may initiate another program for the area by pressing the SYSLOG request key and continuing with the procedure described in Program Initiator.

Cancel Code (Hex)	Message Code	Descriptive Part of Message (or Condition)	Label
10	----	Normal EOJ	ERR10
17	0S02I	(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	ERRGO
1C	0S14I	CANCEL ALL Macro	ERR1C
1D	0S12I	Maintask Termination	ERR1D
1E	0S13I	Unknown ENQ Requestor	ERR1E
1F	0P81I	CPU Failure	ERRGO
20	0S03I or 0S11I	Program Check	ERR20
21	0S04I or 0S09I	Illegal SVC	ERR21
22	0S05I or 0S06I	Phase Not Found	ERR22
23	0S02I	Program Request	ERR23
24	0S01I	Operator Intervention	ERR24
25	0P77I	Invalid address or insufficient core allocation to a partition.	ERR25
26**	0P71I	SYSXXX Not Assigned (unassigned LUB code)	ERR26
27	0P70I	Undefined Logical Unit (invalid LUB code in CCB)	ERR27
28	----	(QTAM cancel in progress)	EXT02
30	0P72I	Reading Past /& Statement (on SYSRDR or SYSIPT)	ERR30
31	0P75I	I/O Error Queue Overflow (error queue overflow or no CHANQ entry available for ERP)	ERR31
32	0P76I	Invalid DASD Address (disk) Irrecoverable I/O Error (tape)	ERR32
33	0P79I	No Long Seek (disk)	ERR33
34	0P84I	I/O Error during fetch (unrecoverable I/O error during fetch of non$ phase)	ERRGO
35	0P851	Job Control Open Failure	-----
40	----	(load $$BEOJ)	EXT02
80	----	(cancel occurred in LTA)	EXT02
FF	0P78I	Unrecognized Cancel Code	-----
FF*	0P83x	Supervisor Catalog Failure	-----

All cancel-codes except in connection with DUMP-macro (code=X'00' is not a true cancel-condition) initially have a value X'40' higher than indicated above, but the X'40' bit is stripped by the SUPVR before fetching the Terminator. In addition to recognizing the cancel-codes above, the Terminator also recognizes the same codes with the X'80' bit on. The X'80' bit is tested for by $$BEOJ and subsequently reset.

*This cancel code is not significant in case of a supervisor catalog failure, because the system is placed in a wait state without any further processing by the Terminator. Thus, there is no conflict between this cancel code and the preceding X'FF' cancel code.

**If the CCB is unavailable, the logical unit is SYSxxx.

Figure 2. Cancel Codes and Messages

SP			upper limit	
BG		size	upper limit	NAME
F2		size	upper limit	NAME
F1	T	size	upper limit	NAME
field 1		field 2	field 3	field 4

Field 1 - area identification

SP - supervisor
BG - background area
F2 - foreground area 2
F1 - foreground area 1
T - indicates which program has interval timer support.

Field 2 - length of area

The number of bytes allocated to the corresponding area of storage, where 1K equals 1024 bytes of storage.

Field 3 - area upper storage limit

The highest storage address allocated to the corresponding area in decimal.

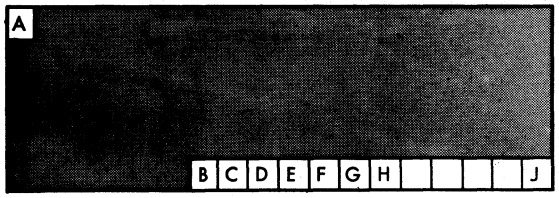
Field 4 - user name

BG - background job name
F2 - foreground 2 program name
F1 - foreground 1 program name

Absence of a name indicates there is no active program in the area.

Figure 3. MAP Output

552-Byte Physical Transient Area



The labels which are associated with these bytes are as designated below. Byte A is the first byte of the Physical Transient Area, Byte J is the last. Bytes B through H constitute the interphase communications area. When phases Z, Y and 0 are fetched or refetched, these bytes (B through H) are not overlaid and remain with information for the other phases.

Byte	Label	Phase
A	IJBPAR1	Z
	IJBPAR2	Y
	IJBPAR3	0
B	PARLTK	Z
	PARCOMM-1	
C	IJBPAR1+504	
	PARCOMM	Z
	IJBPAR2+504	
	PARCOMM2	Y
	PARCOMMC	Z
	PARCOMMD	Y
F	PARCOMMI	Z
	PARCOMMJ	Y
	PARCOMMC+3	
	PARCOMMD+3	
D, E, G, H	Addressed by incrementing or decrementing one of these labels.	

Note: Bytes C, D and E are used to indicate the program(s) F1, F2 or BG, to be canceled.

Bytes F, G and H indicate the programs which use devices which require operator intervention.

Byte B indicates if a canceled program has fetched a logical transient.

4

Figure 4. Interphase Communication Area (For A-Transients $$ANERRZ, Y, and 0)

1

```
// LISTIO SYS

                *** BACKGROUND ***

I/O UNIT   CMNT    CHNL    UNIT    MODE

 SYSRDR              0       0C
 SYSIPT              0       0C
 SYSPCH              0       0D
 SYSLST              1       0A
 SYSLOG              0       1F
 SYSLNK              1       91
 SYSRES              1       92
 SYSSLB             ** UA **
 SYSRLB             ** UA **
 SYSREC              1       91
```

2

```
// LISTIO PROG

                *** BACKGROUND ***

I/O UNIT   CMNT     CHNL    UNIT    MODE

 SYS000              0       91
 SYS001              0       91
 SYS002              0       91
 SYS003              0       91
 SYS004             ** UA **
 SYS005             ** UA **
```

3

```
// LISTIO F2

                *** FOREGROUND 2 ***

I/O UNIT   CMNT    CHNL    UNIT    MODE

 SYSRDR             ** UA **
 SYSIPT             ** UA **
 SYSPCH             ** UA **
 SYSLST             ** UA **
 SYSLOG             ** UA **
 SYSLNK             ** UA **
 SYSRES              1       92
 SYSSLB             ** UA **
 SYSRLB             ** UA **
 SYSREC              1       91

                *** FOREGROUND 2 ***

I/O UNIT   CMNT    CHNL    UNIT    MODE

 SYS000             ** UA **
 SYS001             ** UA **
 SYS002             ** UA **
 SYS003             ** UA **
 SYS004             ** UA **
 SYS005             ** UA **
 SYS006             ** UA **
 SYS007             ** UA **
 SYS008             ** UA **
 SYS009             ** UA **
 SYS010             ** UA **
 SYS011             ** UA **
 SYS012             ** UA **
 SYS013             ** UA **
 SYS014             ** UA **
 SYS015             ** UA **
```

Note: The 1st line of each sample shows the control statement as it was logged by job control.

1. List all system units.
2. List all background programmer units.
3. List all foreground 2 units.
4. List all units.
5. List a specific unit (SYSXXX).
6. List the logical units assigned to all physical devices.
7. List all unassigned units.
8. List all down units.
9. List all logical units assigned to a specified physical unit.

4

```
// LISTIO ALL

                *** BACKGROUND ***

I/O UNIT   CMNT    CHNL    UNIT    MODE

 SYSRDR              0       0C
 SYSIPT              0       0C
 SYSPCH              0       0D
 SYSLST              1       0A
 SYSLOG              0       1F
 SYSLNK              1       91
 SYSRES              1       92
 SYSSLB             ** UA **
 SYSRLB             ** UA **
 SYSREC              1       91

                *** BACKGROUND ***

I/O UNIT   CMNT    CHNL    UNIT    MODE

 SYS000             ** UA **
 SYS001              1       91
 SYS002              1       91
 SYS003              1       91
 SYS004             ** UA **
 SYS005             ** UA **
 SYS006             ** UA **
 SYS007             ** UA **
 SYS008             ** UA **
 SYS009             ** UA **
 SYS010             ** UA **
 SYS011             ** UA **
 SYS012             ** UA **
 SYS013             ** UA **
 SYS014             ** UA **
 SYS015             ** UA **

                *** FOREGROUND 2 ***

I/O UNIT   CMNT    CHNL    UNIT    MODE

 SYSRDR             ** UA **
 SYSIPT             ** UA **
 SYSPCH             ** UA **
 SYSLST             ** UA **
 SYSLOG             ** UA **
 SYSLNK             ** UA **
 SYSRES              1       92
 SYSSLB             ** UA **
 SYSRLB             ** UA **
 SYSREC              1       91

                *** FOREGROUND 2 ***

I/O UNIT   CMNT    CHNL    UNIT    MODE

 SYS000             ** UA **
 SYS001             ** UA **
 SYS002             ** UA **
 SYS003             ** UA **
 SYS004             ** UA **
 SYS005             ** UA **
 SYS006             ** UA **
 SYS007             ** UA **
 SYS008             ** UA **
 SYS009             ** UA **
 SYS010             ** UA **
 SYS011             ** UA **
 SYS012             ** UA **
 SYS013             ** UA **
 SYS014             ** UA **
 SYS015             ** UA **

                *** FOREGROUND 1 ***

I/O UNIT   CMNT    CHNL    UNIT    MODE

 SYSRDR             ** UA **
 SYSIPT             ** UA **
 SYSPCH             ** UA **
 SYSLST             ** UA **
 SYSLOG             ** UA **
 SYSLNK             ** UA **
 SYSRES              1       92
 SYSSLB             ** UA **
 SYSRLB             ** UA **
 SYSREC              1       91
```

```
                *** FOREGROUND 1 ***

I/O UNIT   CMNT    CHNL    UNIT    MODE

 SYS000             ** UA **
 SYS001             ** UA **
 SYS002             ** UA **
 SYS003             ** UA **
 SYS004             ** UA **
 SYS005             ** UA **
 SYS006             ** UA **
 SYS007             ** UA **
 SYS008             ** UA **
 SYS009             ** UA **
 SYS010             ** UA **
 SYS011             ** UA **
 SYS012             ** UA **
 SYS013             ** UA **
 SYS014             ** UA **
 SYS015             ** UA **
```

5

```
// LISTIO SYSRDR

                *** BACKGROUND ***

I/O UNIT   CMNT     CHNL    UNIT   MODE

 SYSRDR              0       0C
```

6

```
// LISTIO UNITS

CHNL    UNIT     OWNER    I/O UNIT    CMNT    MODE

  0      0C       BG       SYSRDR
  0      0C       BG       SYSIPT
  0      0D       BG       SYSPCH
  0      0E                * UA *
  0      1F       BG       SYSLOG
  0      1F       BG       SYSIN
  1      0A       BG       SYSLST
  1      90                * UA *
  1      91       BG       SYSLNK
  1      91       BG       SYSREC
  1      91       BG       SYS001
  1      91       BG       SYS002
  1      91       BG       SYS003
  1      91       F2       SYSREC
  1      91       F1       SYSREC
  1      92       BG       SYSRES
  1      92       F2       SYSRES
  1      92       F1       SYSRES
  1      80                * UA *
  1      81                * UA *
  1      82                * UA *
  1      83                * UA *
  1      84                * UA *
```

7

```
// LISTIO UA

       *** UNASSIGNED ***

           CHNL    UNIT

             1      80
             1      81
             1      82
             1      83
```

8

```
// LISTIO DOWN

           *** DOWN ***

           CHNL    UNIT

            ** NONE **
```

9

```
// LISTIO X@01F@

CHNL    UNIT     OWNER    I/O UNIT    CMNT    MODE

  0      1F       BG       SYSLOG
```

Figure 5. List I/O Examples

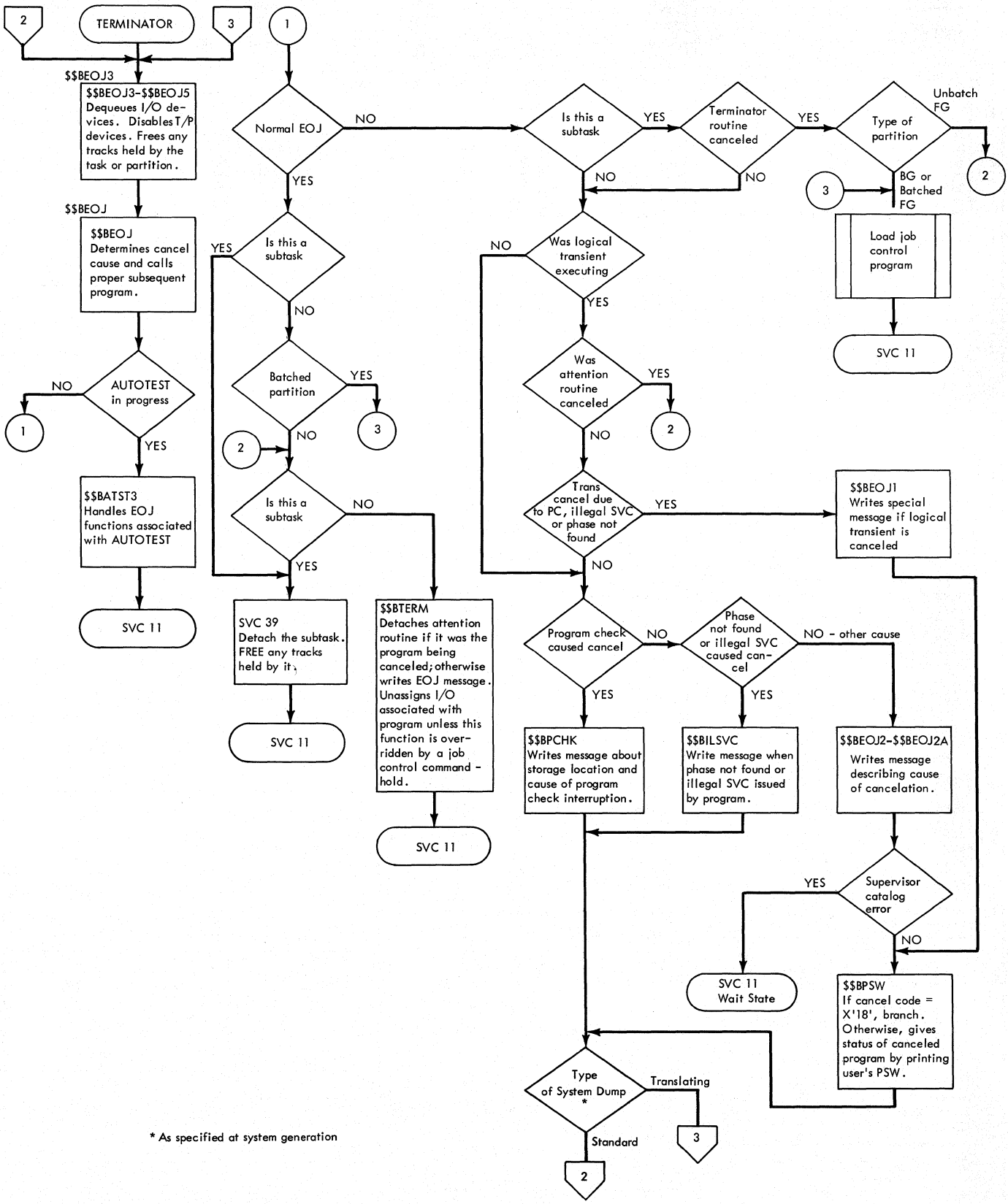


Figure 6. Terminator Phases (Part 1 of 3)

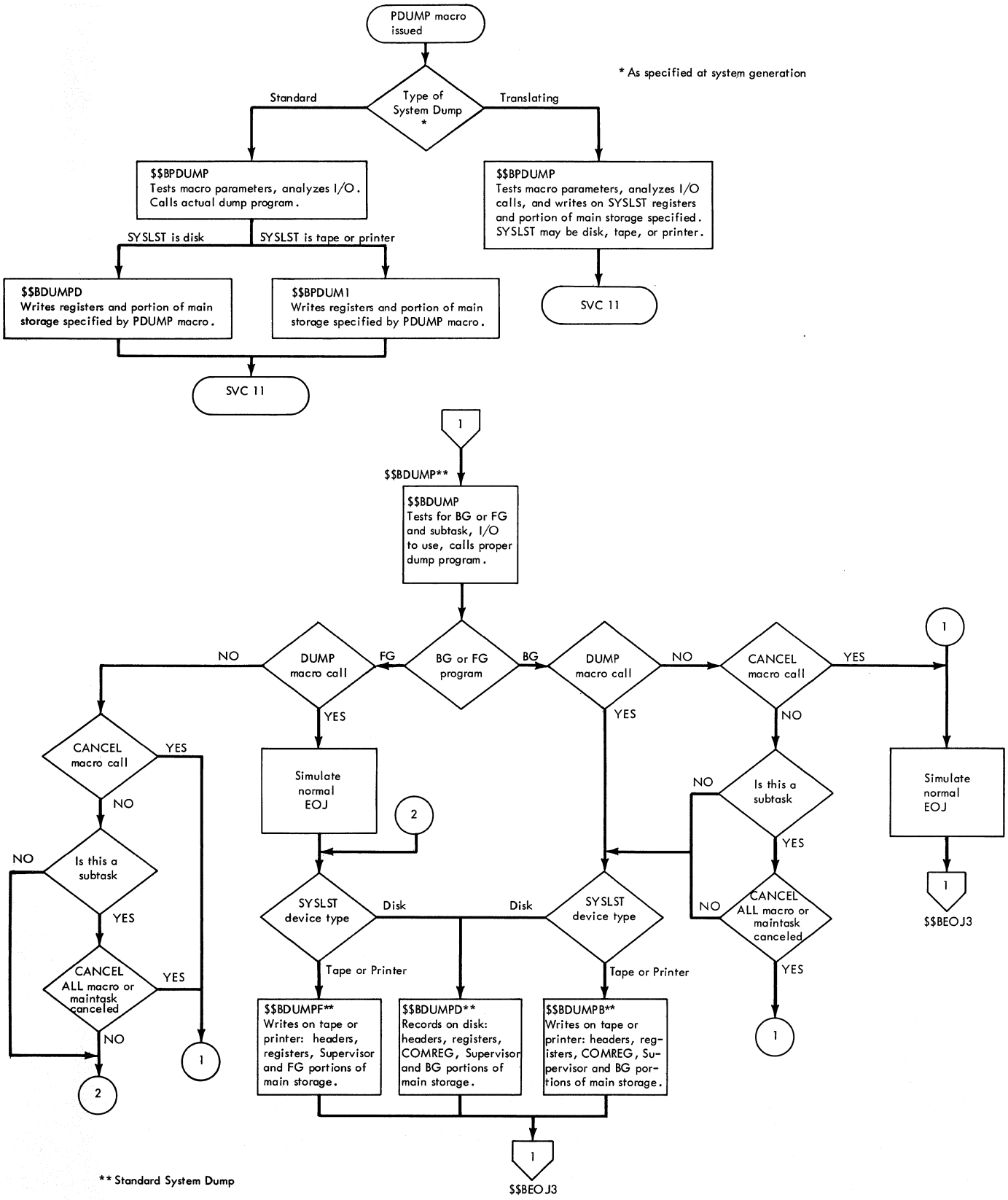


Figure 6. Terminator Phases (Part 2 of 3)

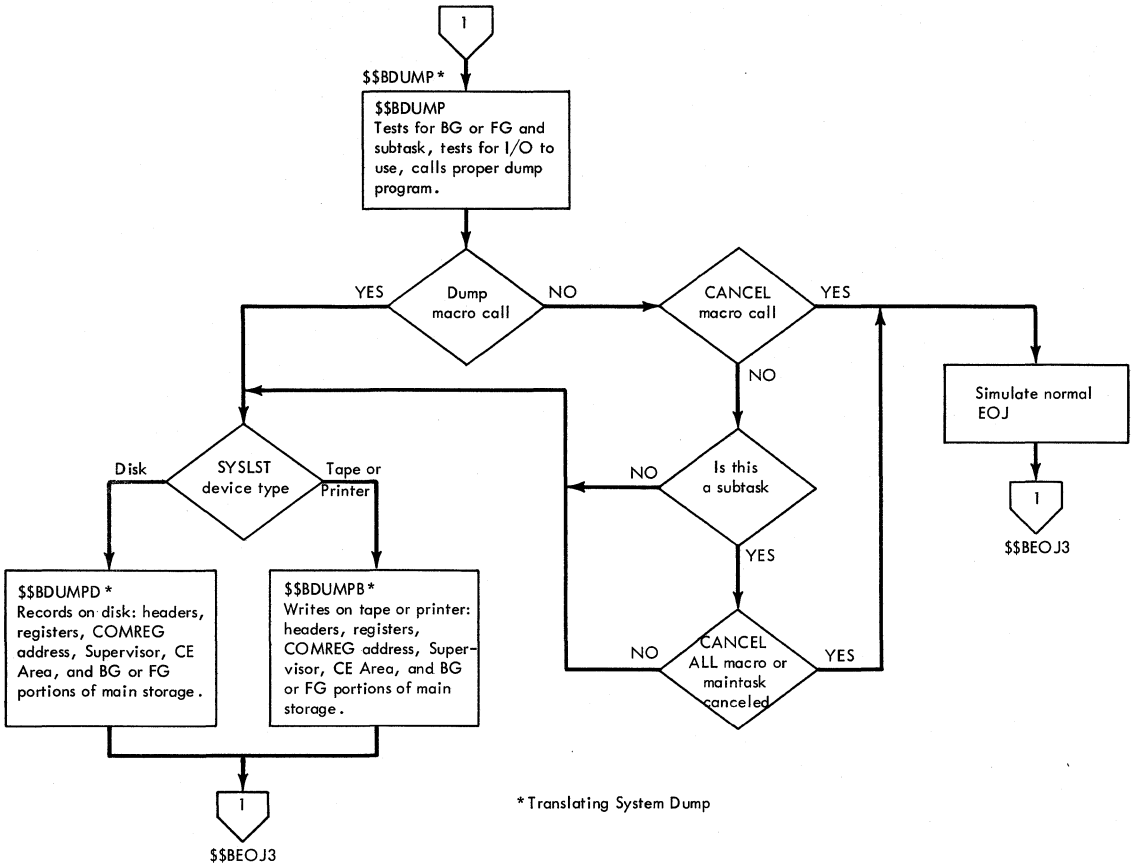


Figure 6. Terminator Phases (Part 3 of 3)

SUPERVISOR CALLS

SVC is detected by microprogramming, which loads the SVC new PSW from core 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.

For reference purposes, Figure 7 contains a list of all SVCs used in the system. Refer to the Supervisor and Related Transients PLM listed in the Preface for an explanation of the SVCs outside the scope of this manual.

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 (see Notes, item 1) to the B-transient area (refer to Figure 1), and enters the B-transient at its load address plus 8 bytes. The storage address of the B-transient phase name must be supplied in general register 1.

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.

Notes:

1. If PCIL (Private Core Image Library) is assigned, it is searched first for the proper phase. If the phase is not found in the PCIL, the system core image library is then searched. When the first character of the phase name is $, the system core image library is searched first and the private core image library last.

2. Supervisor may branch directly to the SVC 2 routine when fetching a B-transient. If the transient is not in the library when referenced by the supervisor, the system enters the wait state.

SVC 6: Cancels a program (task) or partition. If either a subtask or maintask (or only program in the partition) issues a CANCEL, cancel code X'23' is posted to the PIB for the task issuing the cancel. If a CANCEL ALL is issued, cancel code X'1C' is posted.

A simple cancel issued by a subtask performs the same function as DETACH (see SVC 39), but also posts the ECB's byte 2, bits 0 and 1, and issues a subtask cancellation message. When CANCEL is issued by a maintask, the partition is canceled.

A CANCEL ALL macro issued by a subtask cancels the entire partition. In this case, the AB exits for all tasks that have them are taken, except for the subtask issuing the CANCEL ALL. (Refer to Figures 15-17 for the format of the PIB table, and to Figure 2 for cancel codes.) The next time the canceled program is selected on general exit, the supervisor branches to the SVC 2 routine to fetch the cancel B-transient program $$BEOJ3 if teleprocessing is supported, or $$BEOJ4 if teleprocessing is not supported.

SVC 8: Supplies the supervisory support to temporarily return from a B-transient program to the problem program. The B-transient area is not released. The task selection exit loads the problem program registers. An SVC 9 is used to return to the B-transient program.

SVC 9: Supplies the supervisory support for returning to the B-transient after an SVC 8 is issued. The task selection exit loads the B-transient registers.

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 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. Refer to Figures 15-17 for the format of the PIB tables. The next time the canceled program is selected on general exit, a branch is made to the SVC 2 routine to fetch the cancel B-transient program

$$BEOJ3 if teleprocessing is supported, or $$BEOJ4 if teleprocessing is not supported. Job control is loaded by $$BEOJ to perform the end-of-job-step.

SVC 22: Seizes the system and provides a release from such a seizure. The SVC 22 is ignored if supervisor was generated without the MPS option. The program issuing an SVC 22 is canceled if the PSW protection key field does not equal 0. (Only job control and B-transient programs can issue an SVC 22.)

The first SVC 22 issued seizes the system and the next one issued releases the system. The last byte of register 0 replaces the system mask. If register 0 is negative, the protection key is replaced by the protection key of the PIK.

The task selection mechanism is altered by the first SVC 22 so that only supervisor or quiesce I/O tasks and the program that issued the SVC 22 can be selected. The next SVC 22 issued restores the task selection mechanism. The contents of the last byte of general register 0 are again used as the system mask.

Return from each SVC 22 is directly to the interrupted program.

Note:

- There is no way to cancel a program that has seized the system.
- The program must have no pending I/O operations.
- The program cannot issue supervisor calls while the system is seized.

SVC 23: Loads phase header. Retrieves the load address for a specified phase from the system core image directory or a private core image library (PCIL). The program issuing an SVC 23 is canceled if supervisor was generated without the MPS option or 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 address of the core image 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 core image directory and retrieves the load address. If the phase is found in the directory, the load address (3 bytes) is stored at the address specified by general register 0. If the phase is not found, the supervisor returns control to the interrupted program.

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. The upper address must be within main storage, and the lower address must be higher than the end of supervisor address, or the program is canceled (ERR25). Return is to the interrupted program. No task selection is performed.

With MPS option, the PIK of the program issuing the SVC 26 must equal the storage protection key for both addresses or the program is canceled (ERR25).

With batch operation, SVC 26 is ignored unless storage protection has been specified.

Macro Supported	SVC	Function
EXCP	0	Execute channel programs.
FETCH	1 2 3	Fetch any phase. Fetch a logical transient (B-transient). Fetch or return from a physical transient (A-transient).
LOAD	4	Load any phase.
MVCOM	5	Modify supervisor communications region.
CANCEL	6	Cancel a problem program or task.
WAIT	7	Wait for a CCB or TECB.
	8	Transfer control to the problem program from a logical transient (B-transient).
LBRET	9	Return to a logical transient (B-transient) from the problem program after a SVC 8.
SETIME	10*	Set timer interval.
	11 12 13	Return from a logical transient (B-transient). Logical AND (Reset) to second job control byte (displacement 57 in communications region). Logical OR (Set) to second job control byte (displacement 57 in communications region).
EOJ	14	Cancel job and go to job control for end of job step.
	15	Same as SVC 0 except ignored if CHANQ table is full. (Primarily used by ERP).
STXIT (PC)	16*	Provide supervisor with linkage to user's PC routine for program check interrupts.
EXIT (PC)	17*	Return from user's PC routine.
STXIT (IT)	18*	Provide supervisor with linkage to user's IT routine for interval timer interrupts.
EXIT (IT)	19*	Return from user's IT routine.
STXIT (OC)	20*	Provide supervisor with linkage to user's OC routine for external or attention interrupts (operator communications).
EXIT (OC)	21*	Return from user's OC routine.
	22* 23*	The first SVC 22 seizes the system for the issuing program by disabling multiprogram operation. The second SVC 22 releases the system (enables multiprogram operation). Load phase header. Phase load address is stored at user's address.
SETIME	24*	Provide supervisor with linkage to user's TECB and set timer interval.
	25* 26* 27*	Issue HALT I/O on a teleprocessing device. Validate address limits. Special HIO on teleprocessing devices.
EXIT (MR)	28*	Return from user's stacker select routine (MICR type devices only).
	29*	Provide return from multiple wait macros WAITF and WAITM (except MICR type devices).
QWAIT	30*	Wait for a QTAM element.
QPOST	31* 32 33 34	Post a QTAM element. Reserved. Reserved for internal macro COMRG. Reserved for internal macro GETIME.

* = optional

Figure 7. DOS Supervisor Calls (Part 1 of 2)

Macro Supported	SVC	Function
HOLD	35*	Hold a track for use by the requesting task only.
FREE	36*	Free a track held by the task issuing the FREE.
STXIT (AB)	37*	Provide supervisor with linkage to user's AB routine for abnormal termination of a task.
ATTACH	38*	Initialize a subtask and establish its priority.
DETACH	39*	Perform normal termination of a subtask. It includes calling the FREE routine to free any tracks held by the subtask
POST	40*	Inform the system of the termination of an event and ready any waiting tasks.
DEQ	41*	Inform the system that a previously enqueued resource is now available.
ENQ	42*	Prevent tasks from simultaneous manipulation of a shared data area (resource).
	43*	Provide supervisor support for external creation and updating of SDR records.
	44*	Provide supervisor support for external creation of OBR records.
	45*	Provide emulator interface.
	46*	Provide OLTEP with the facility to operate in supervisory state.
	47*	Provide return from wait multiple WAITF for MICR type devices.
	48	Reserved.
	49	Reserved.
	50	Reserved for LIOCS error recovery.

* = optional

Figure 7. DOS Supervisor Calls (Part 2 of 2)

Chart 01. Overview of Supervisor Entry into B-Transients

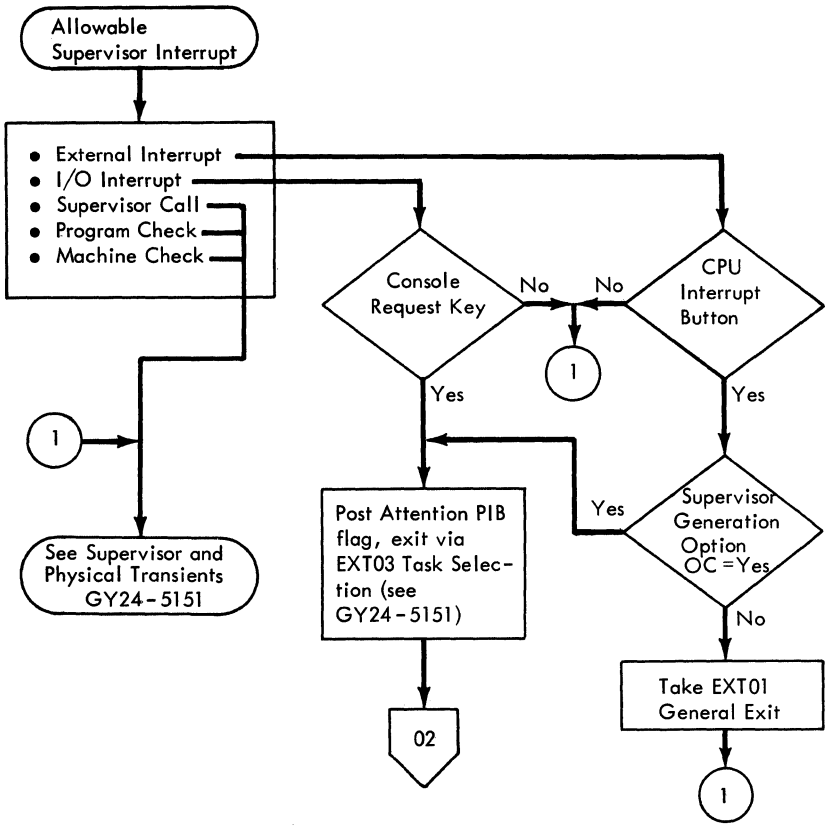
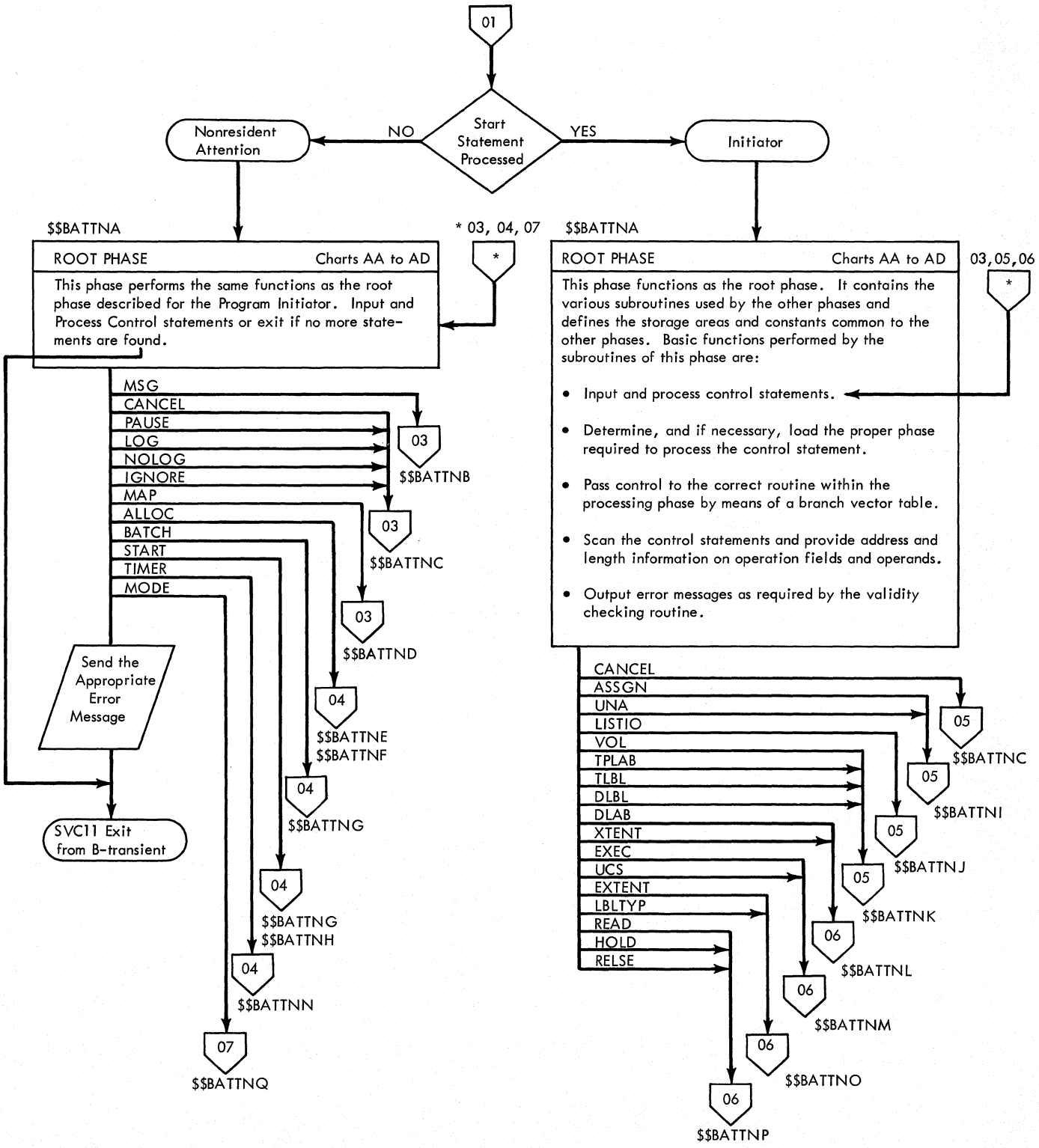


Chart 02. Logical Transient Root Phase



NOTE: If foreground initiation is in process, the root phase remains resident in the logical transient area and the initiating routines are loaded and executed from the foreground area of the program being initiated.

Chart 03. Logical Transient Attention Routines (Part 1 of 2)

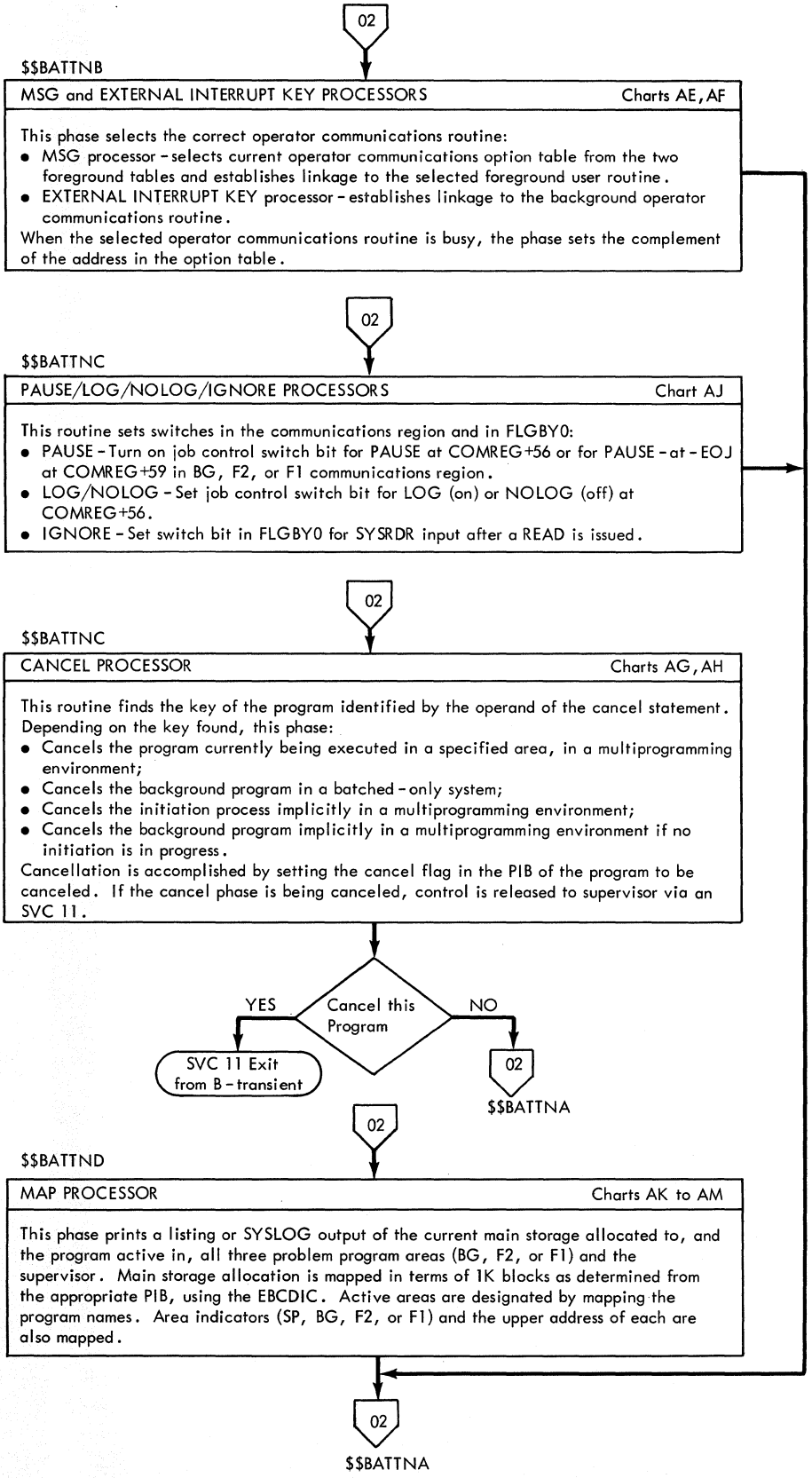


Chart 04. Logical Transient Attention Routines (Part 2 of 2)

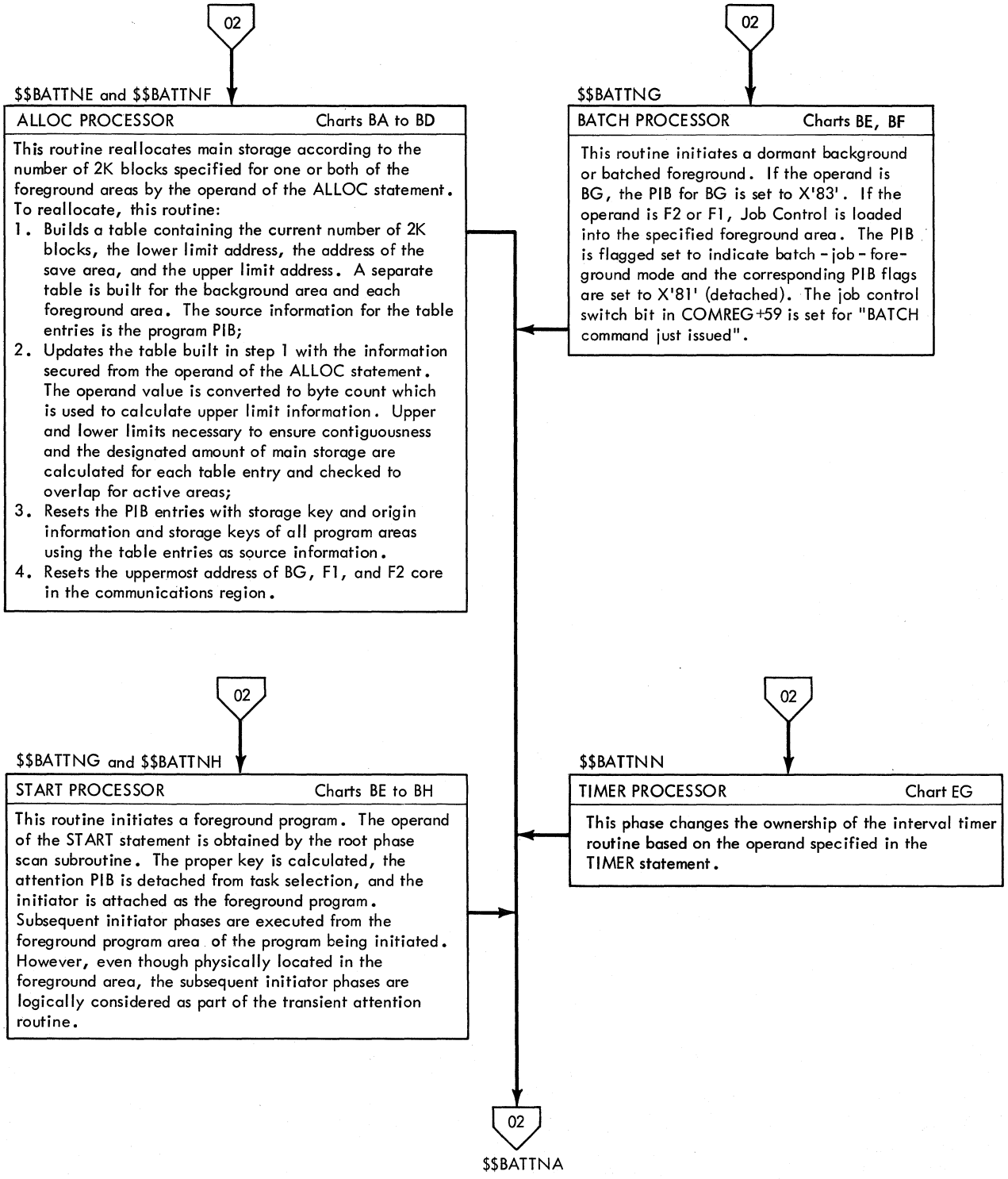


Chart 05. Logical Transient Initiator (Part 1 of 2)

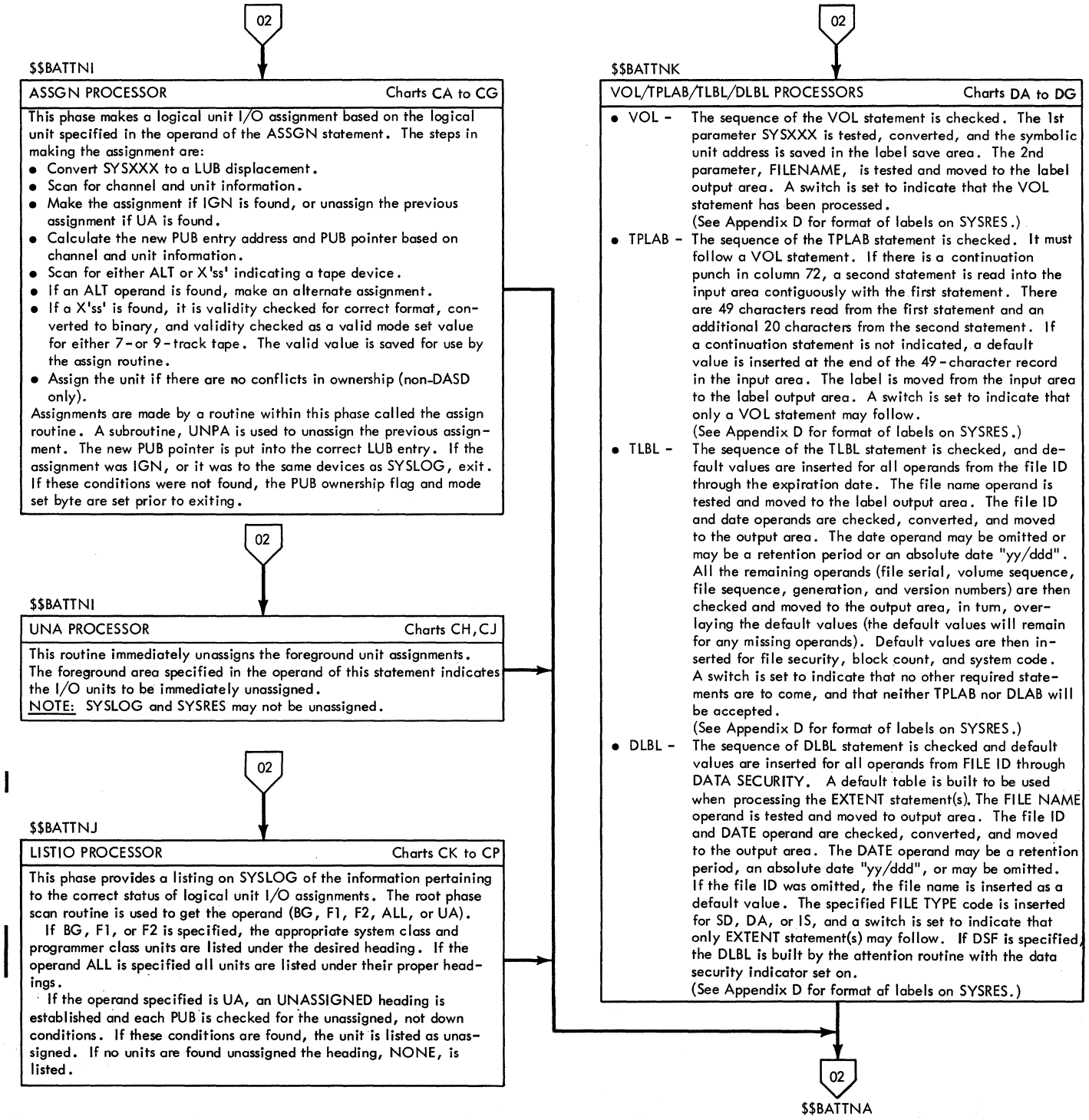


Chart 06. Logical Transient Initiator (Part 2 of 2)

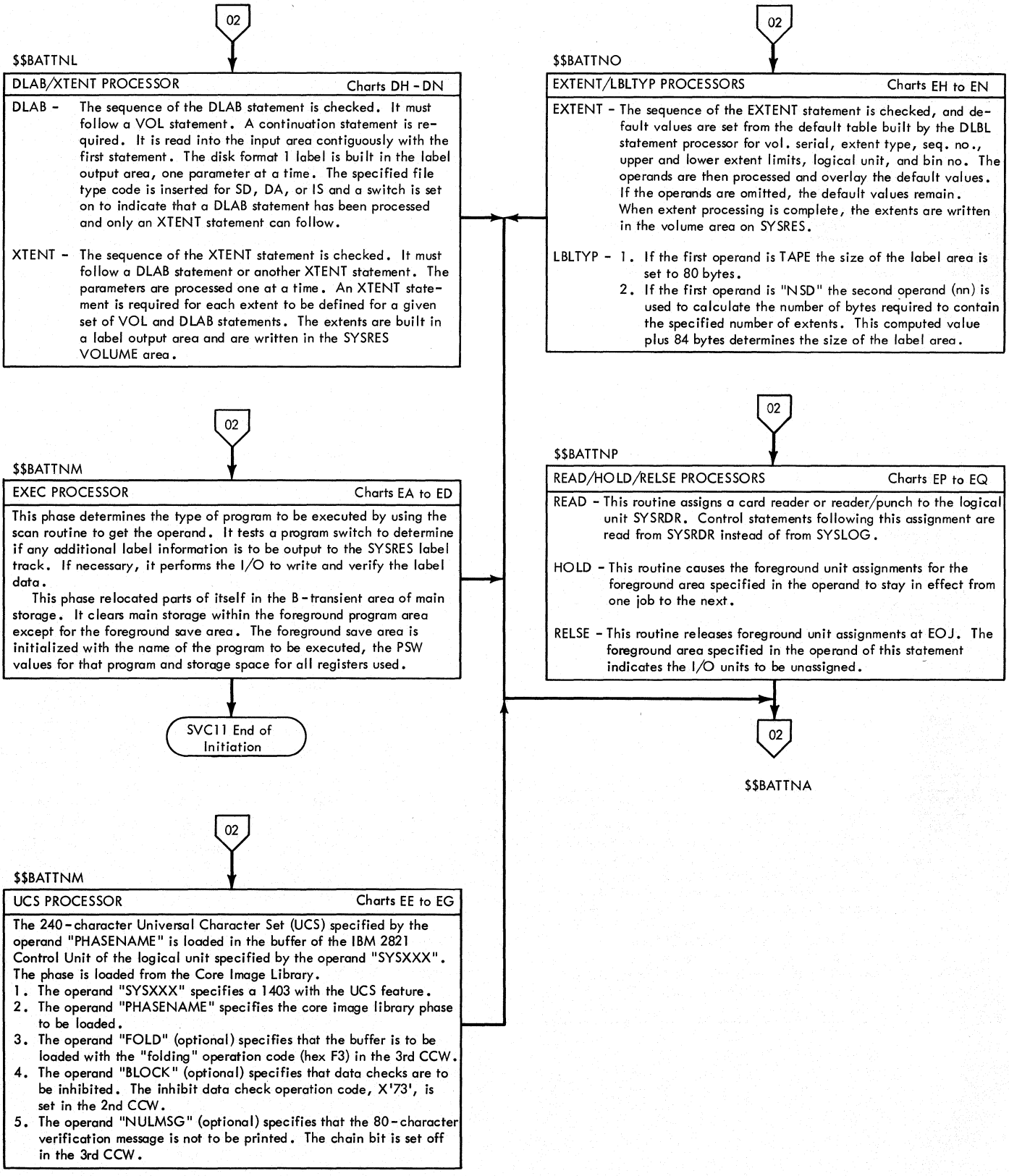


Chart 07. MODE Command Processor

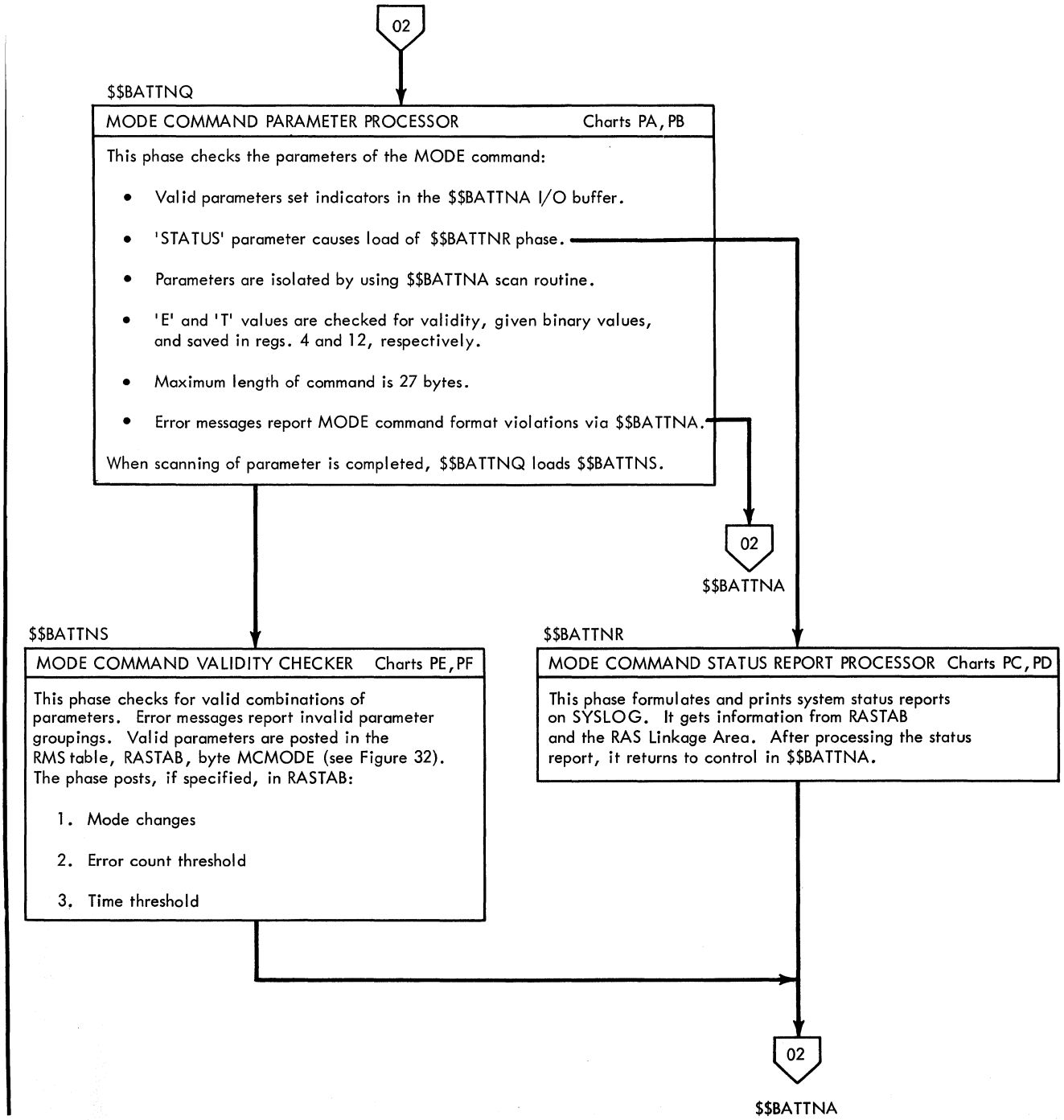


Chart 08. Logical Transient Terminator (Part 1 of 8)

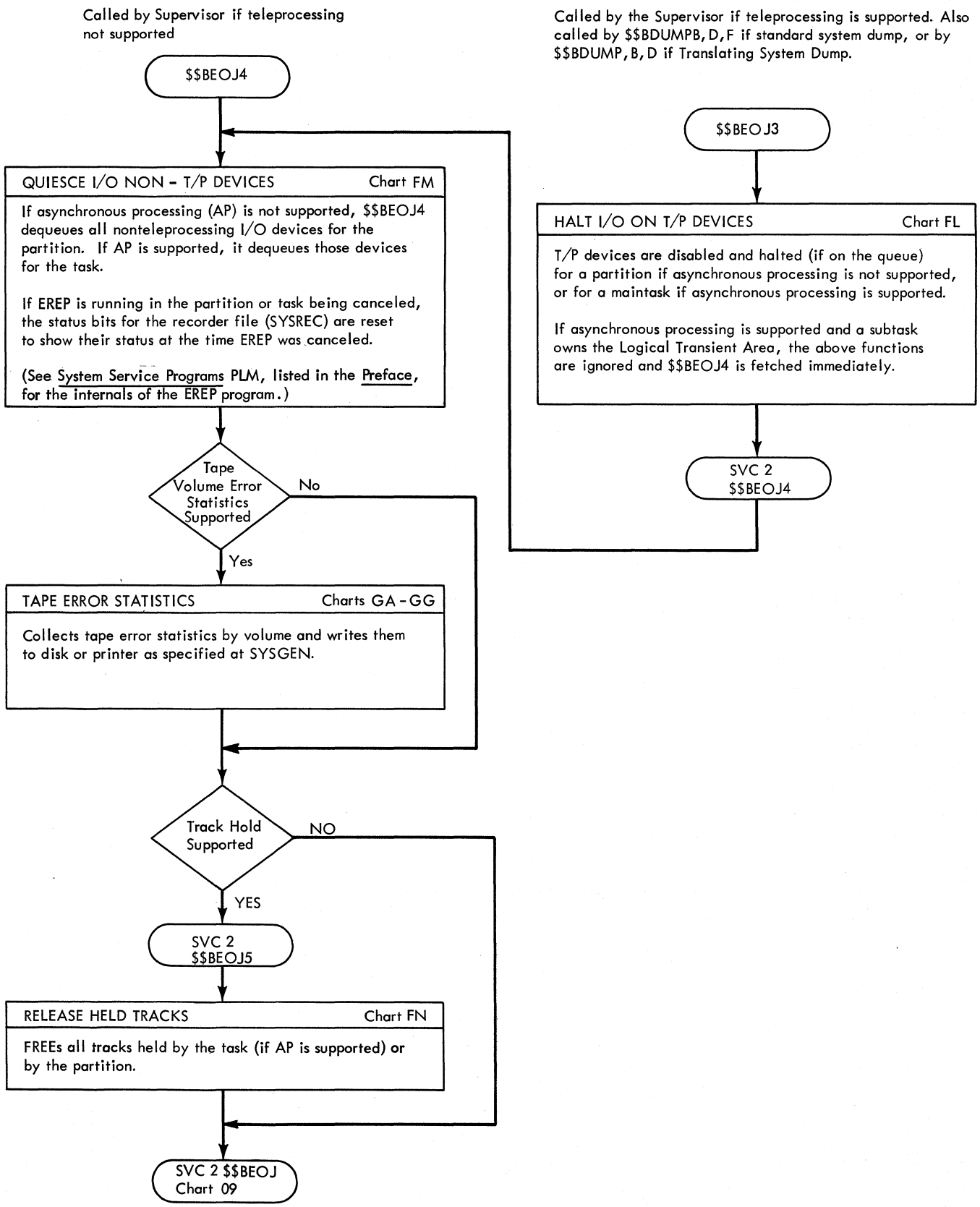


Chart 09. Logical Transient Terminator (Part 2 of 8)

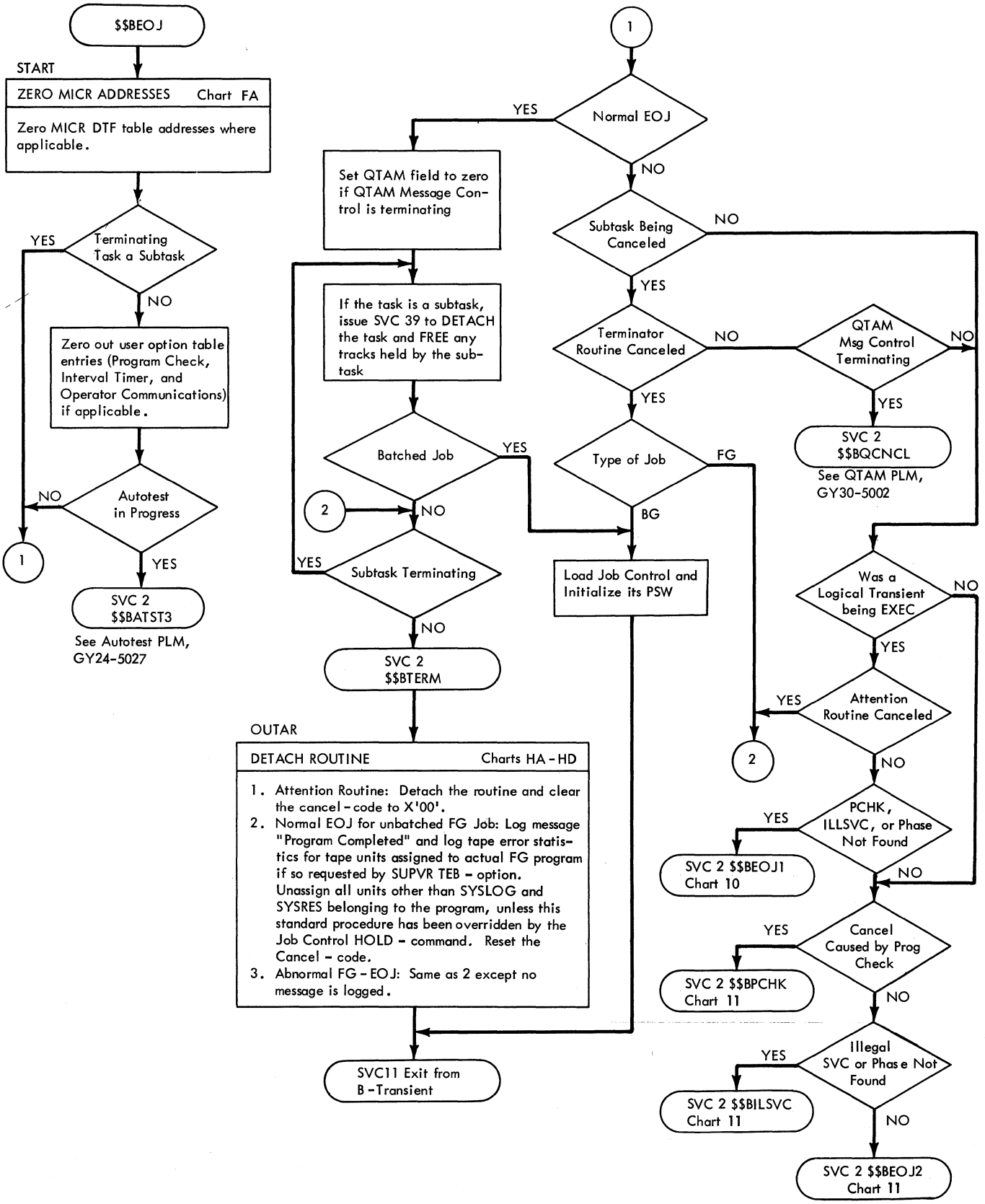


Chart 10. Logical Transient Terminator (Part 3 of 8)

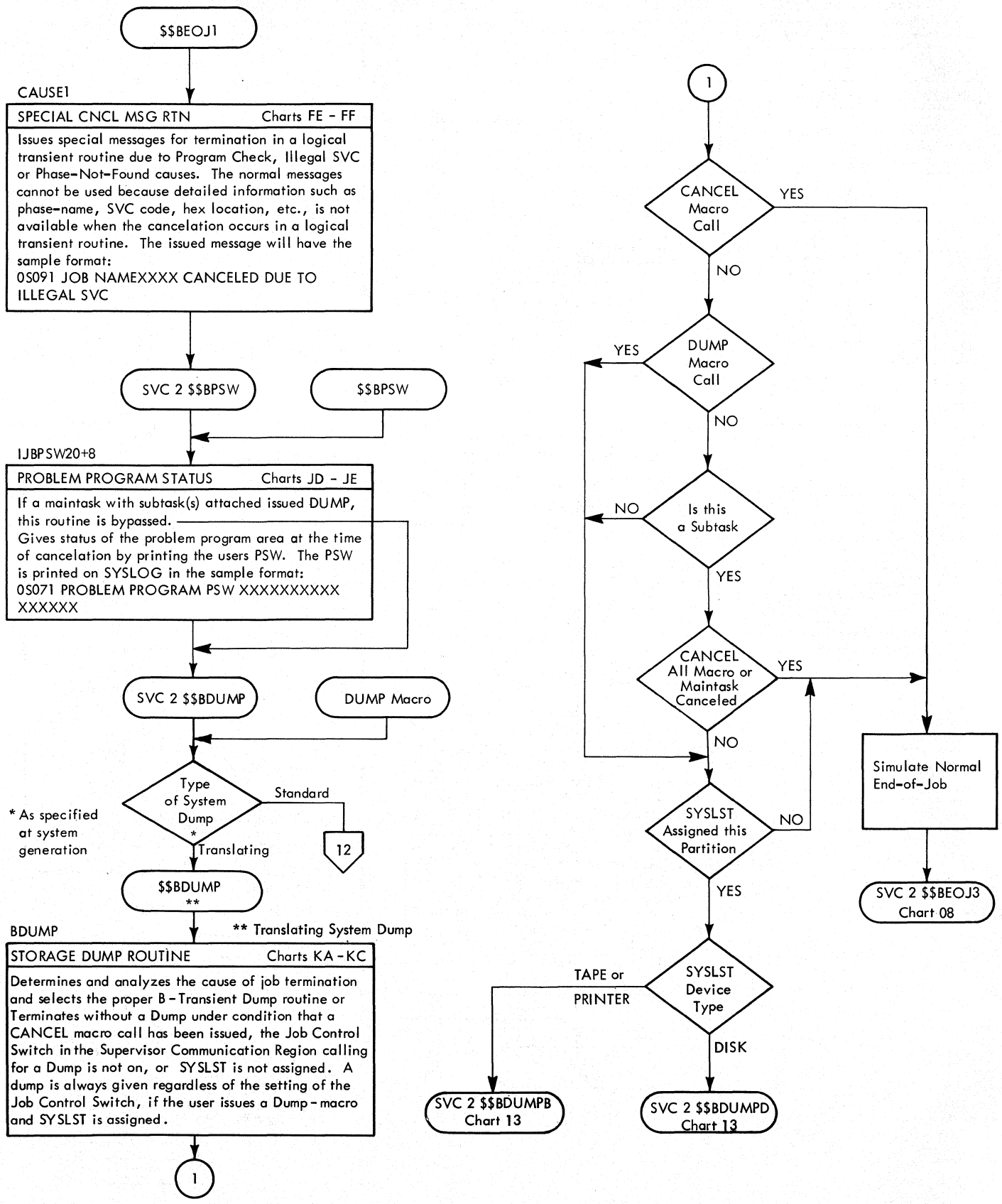


Chart 11. Logical Transient Terminator (Part 4 of 8)

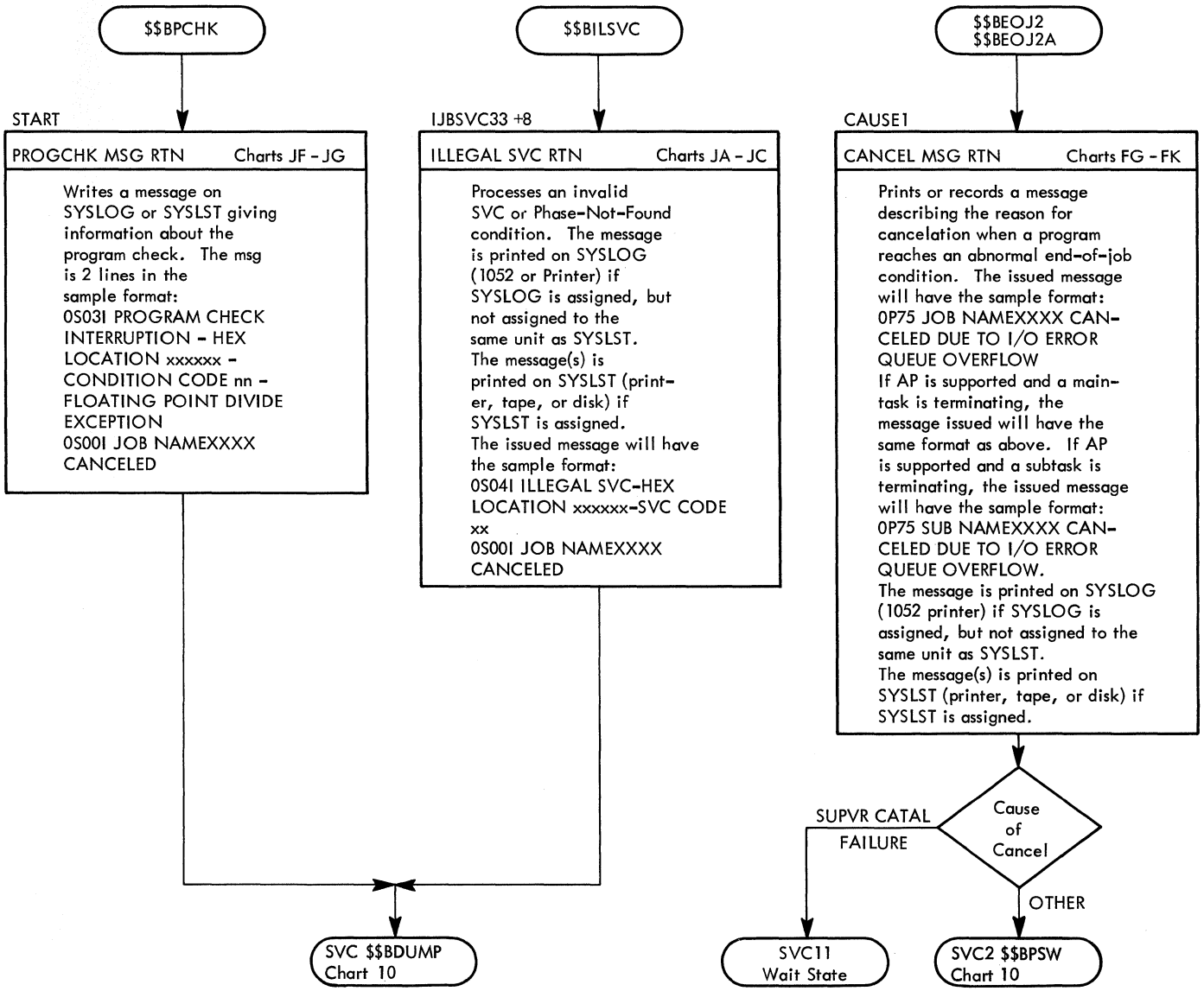


Chart 12. Logical Transient Terminator (Part 5 of 8)

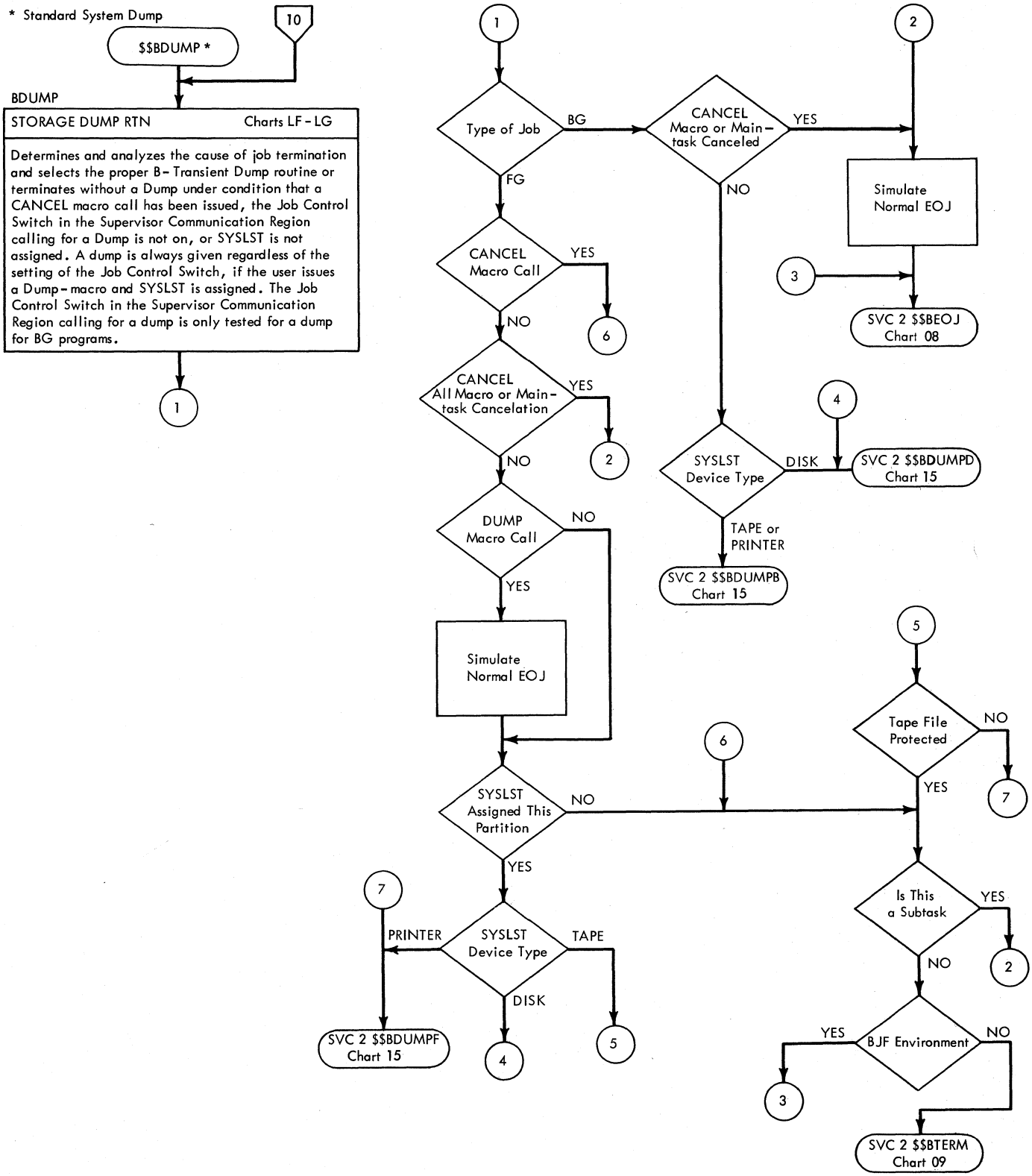


Chart 13. Logical Transient Terminator (Part 6 of 8)

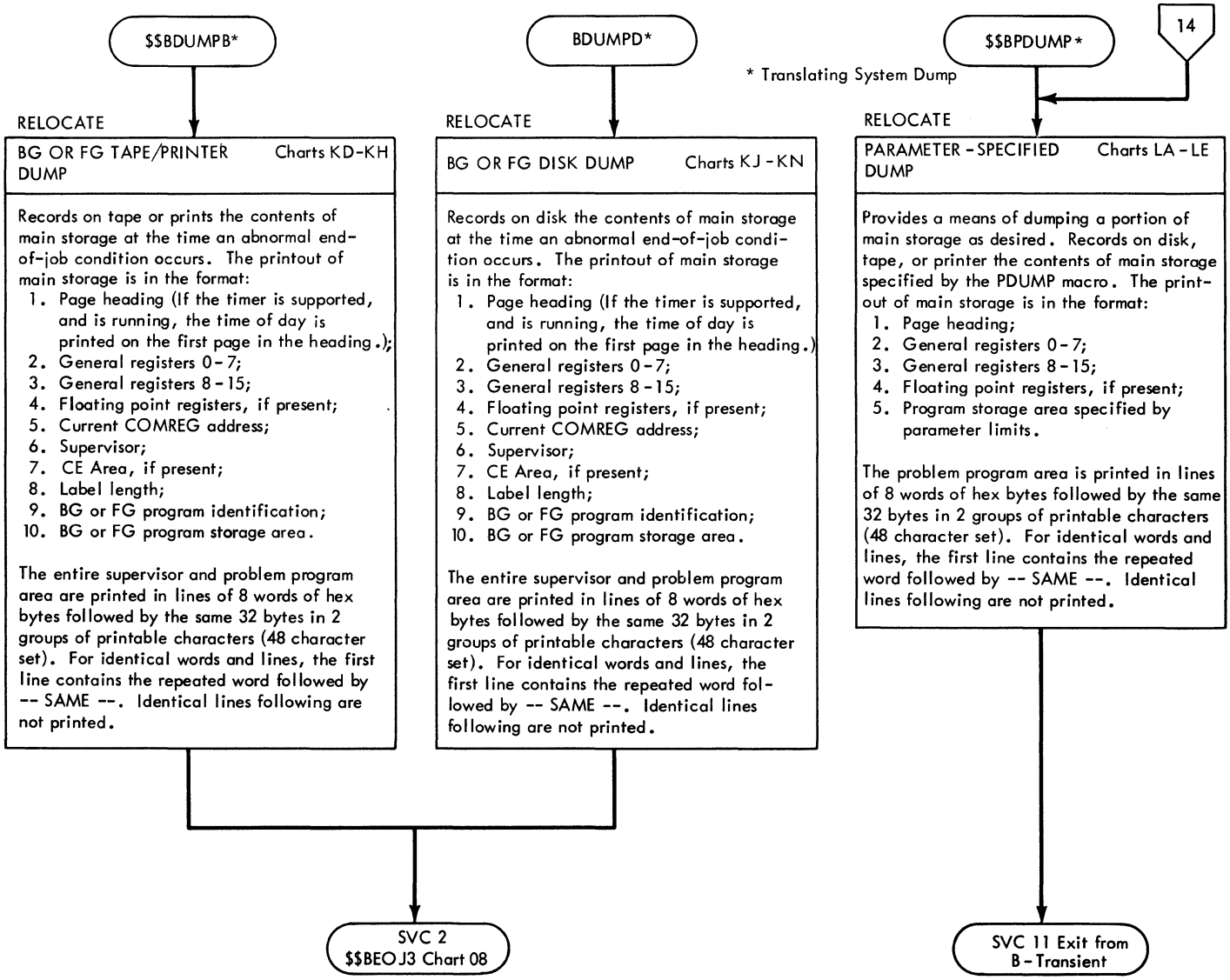
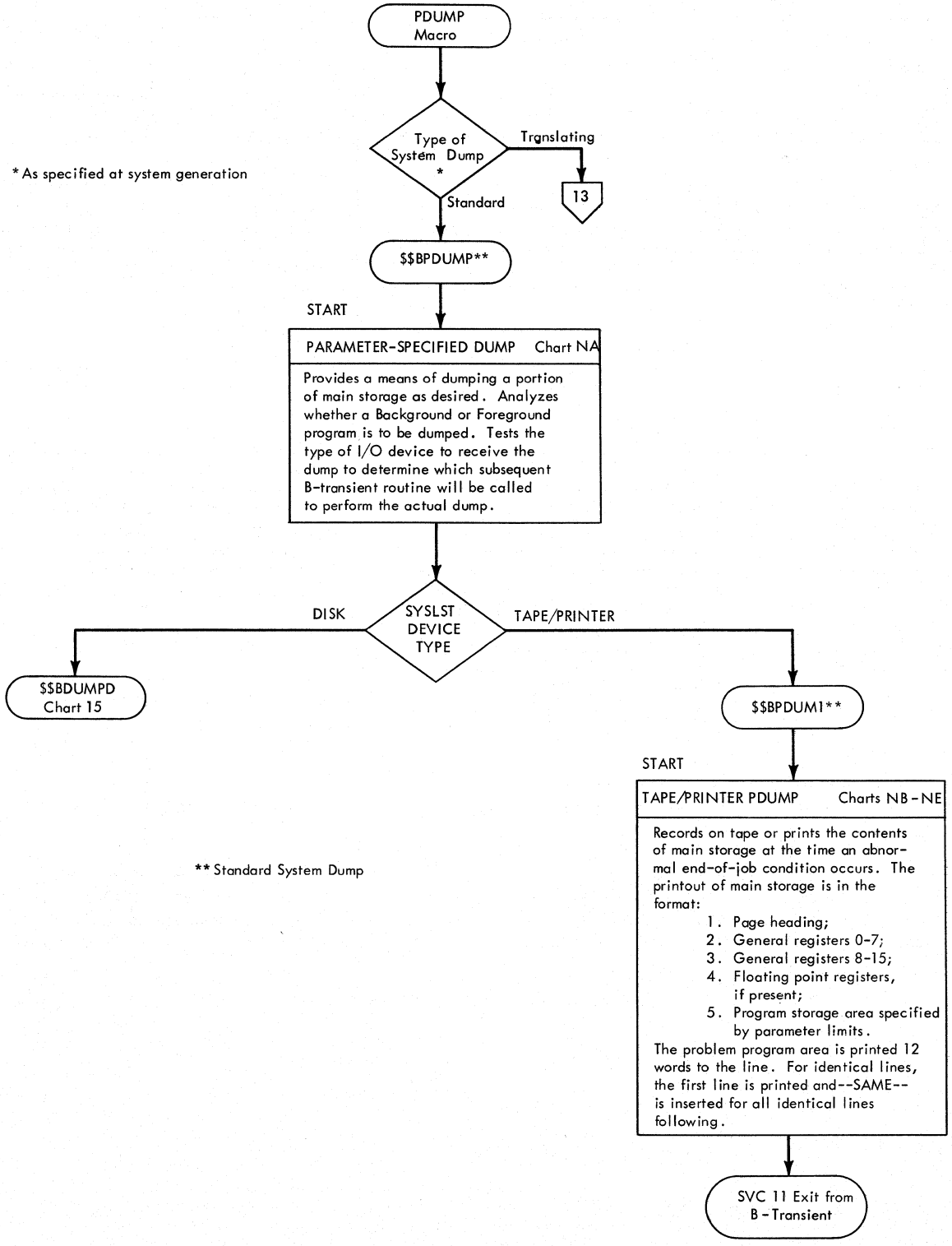


Chart 14. Logical Transient Terminator (Part 7 of 8)



* As specified at system generation

** Standard System Dump

Chart 15. Logical Transient Terminator (Part 8 of 8)

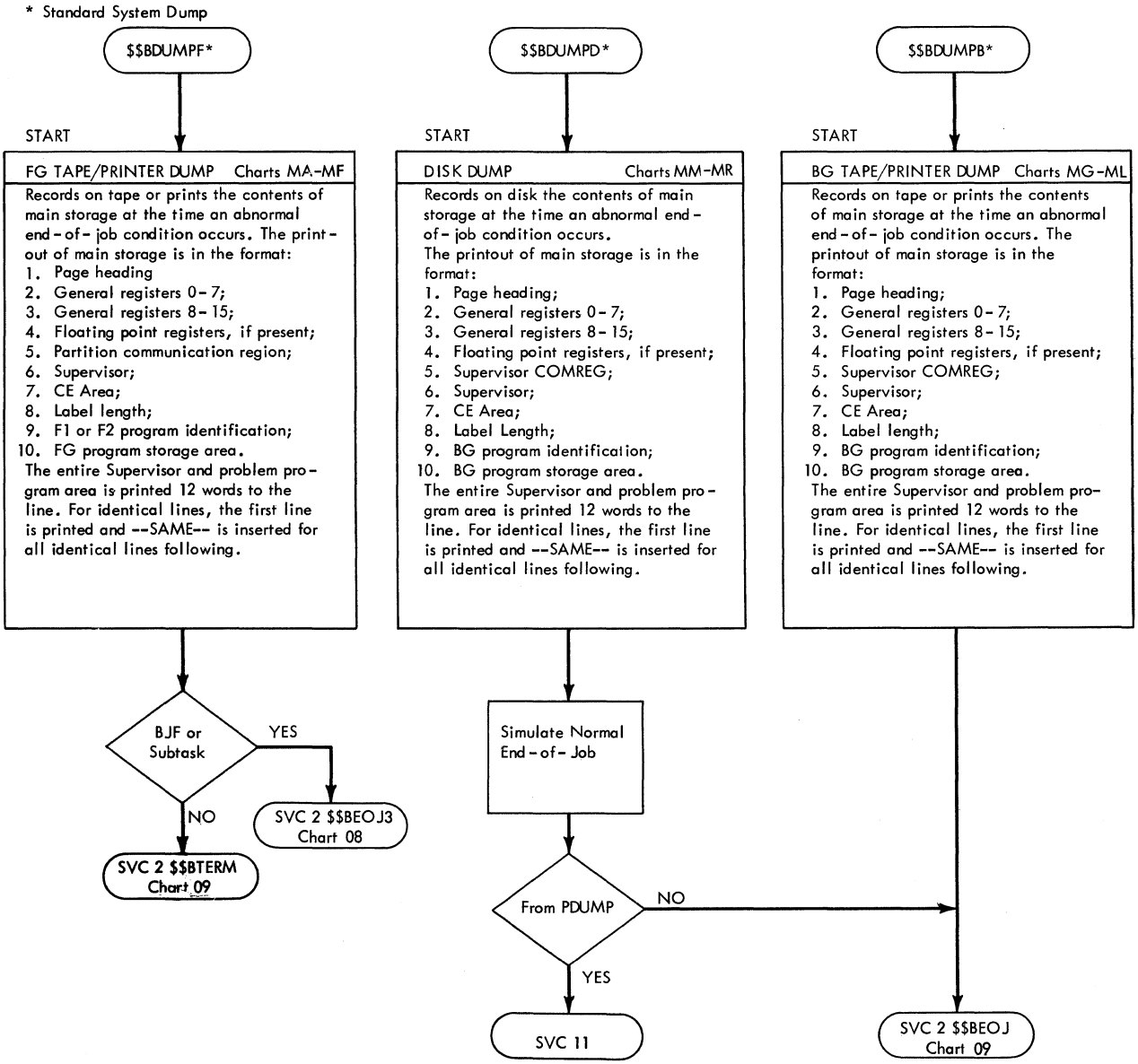


Chart AA. $$BATTNA - Nonresident Attention/Initiator Root Phase
Refer to Chart 02.

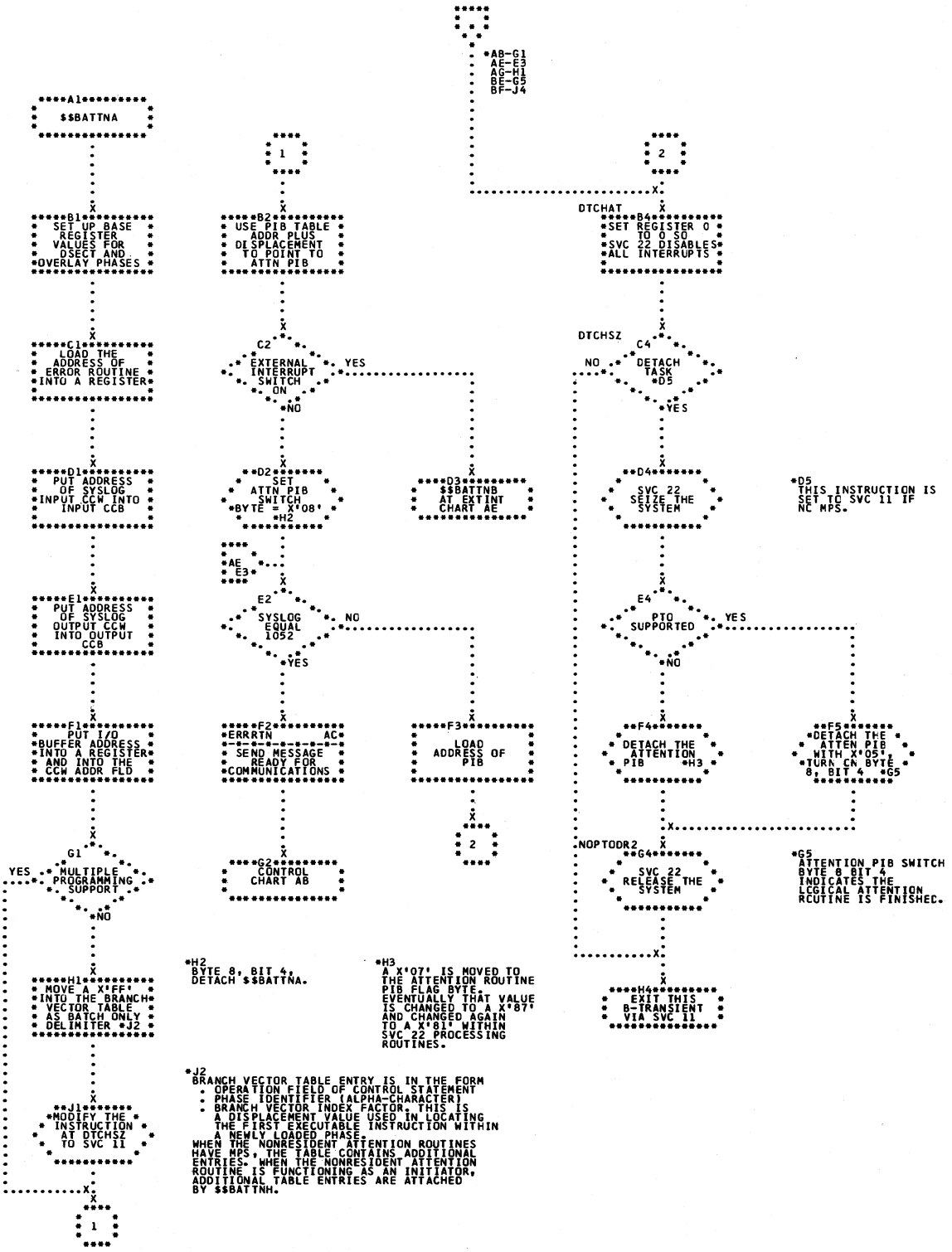


Chart AB. $$BATTNA - Control Routine
Refer to Chart 02.

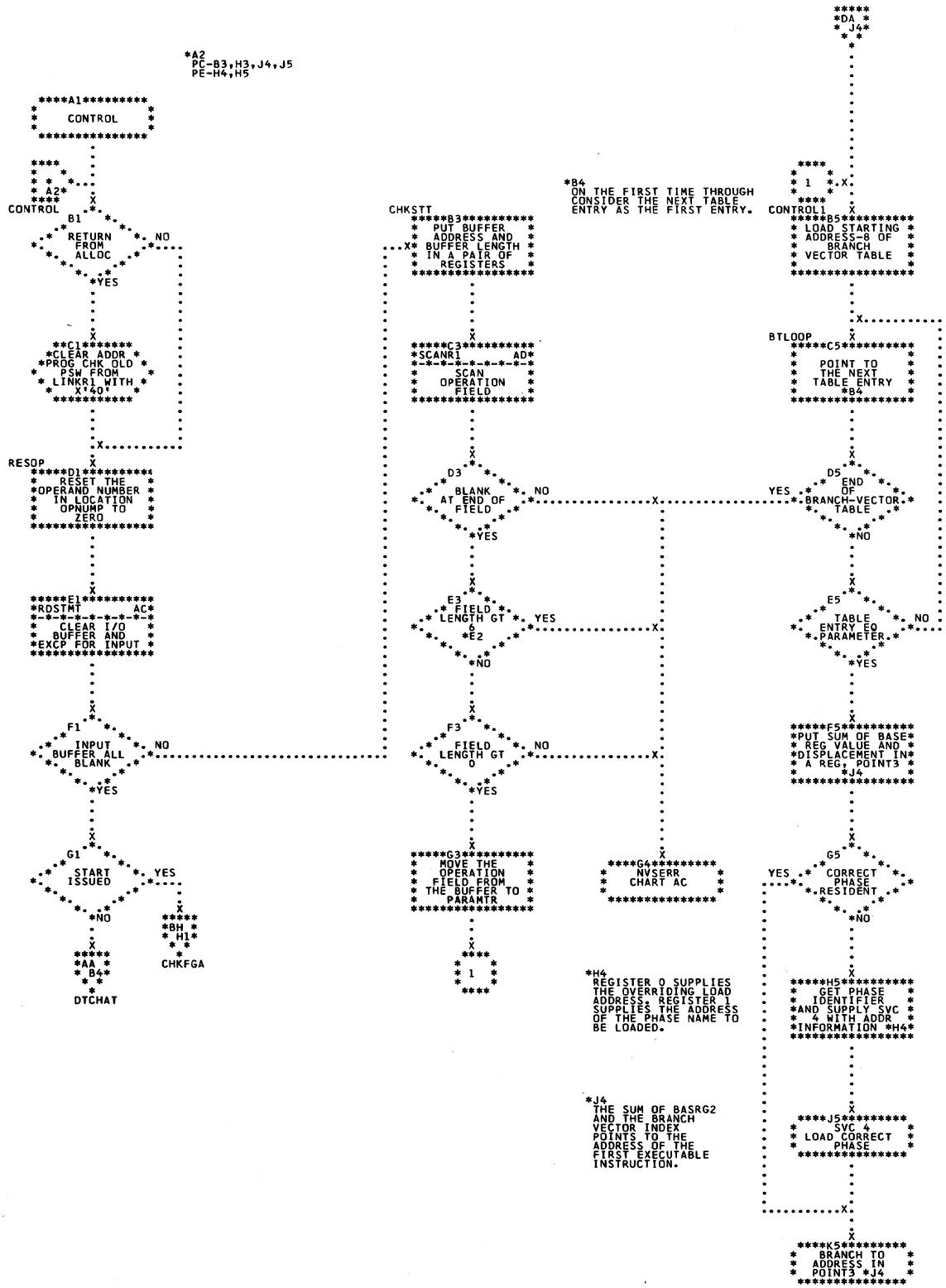


Chart AC. $$BATTNA - Root Phase Subroutines
Refer to Chart 02.

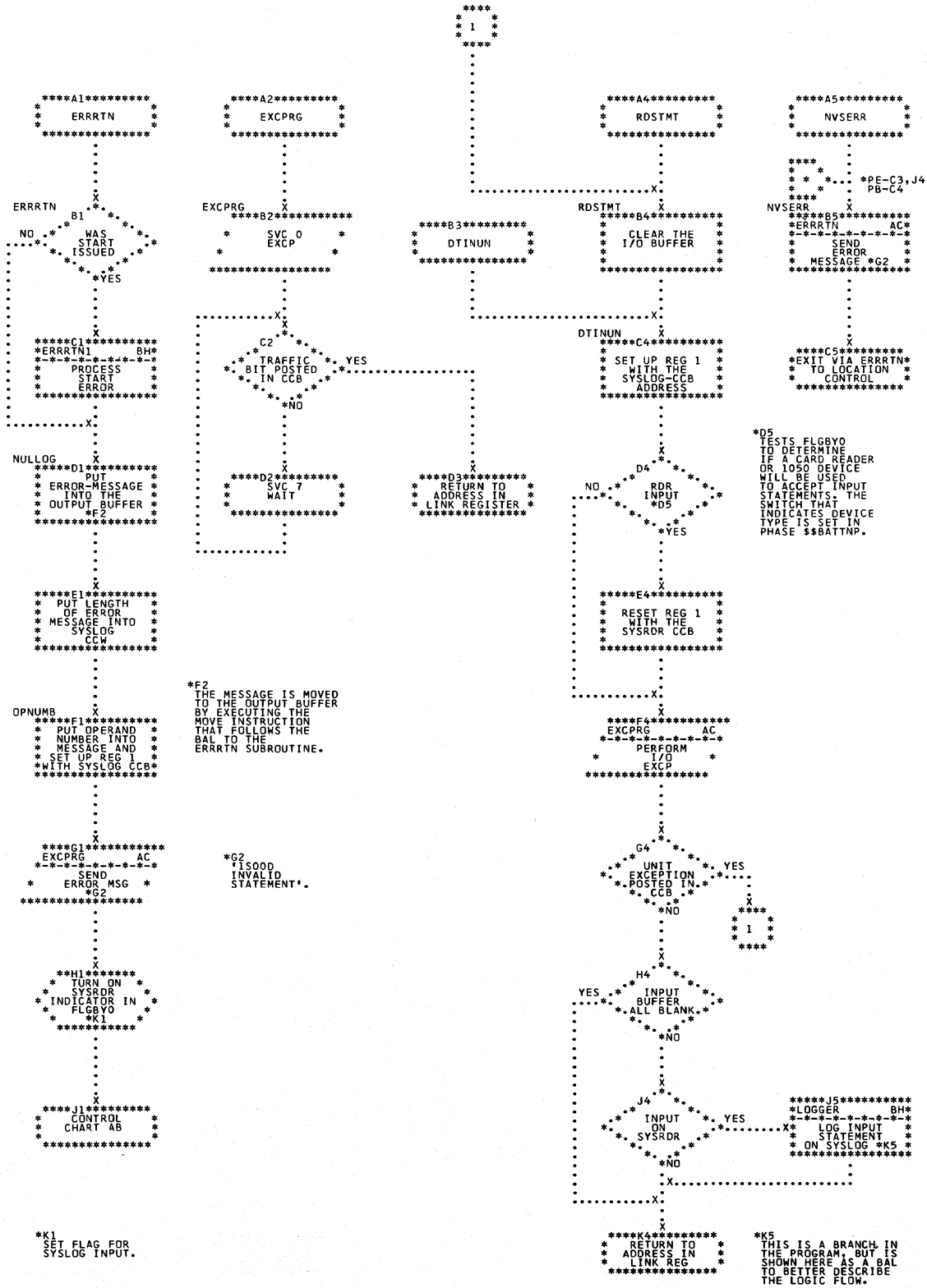


Chart AD. $$BATTNA - General Scan Routines
Refer to Chart 02.

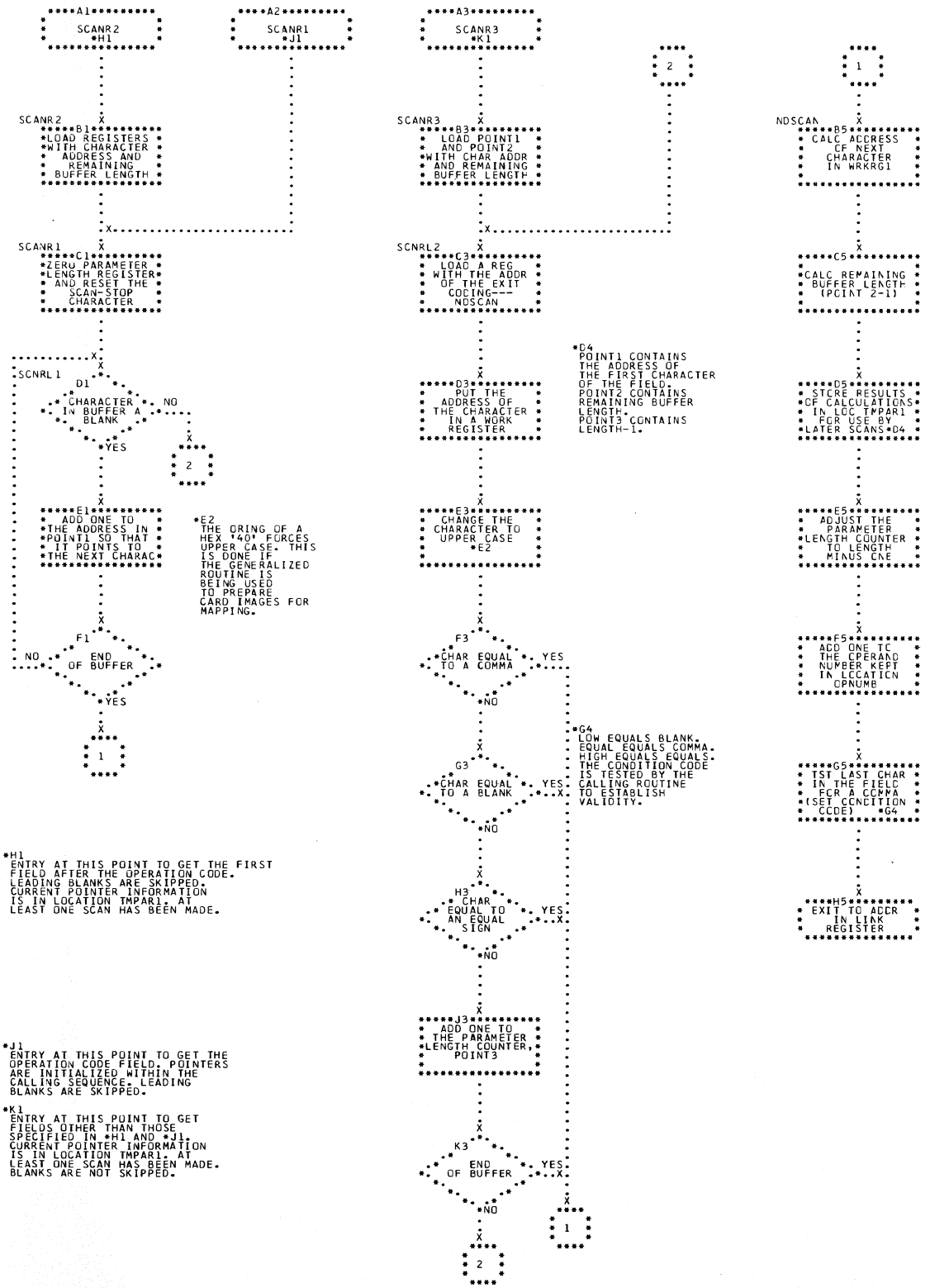


Chart AE. $$BATTNB - MSG Statement Processor
Refer to Chart 03.

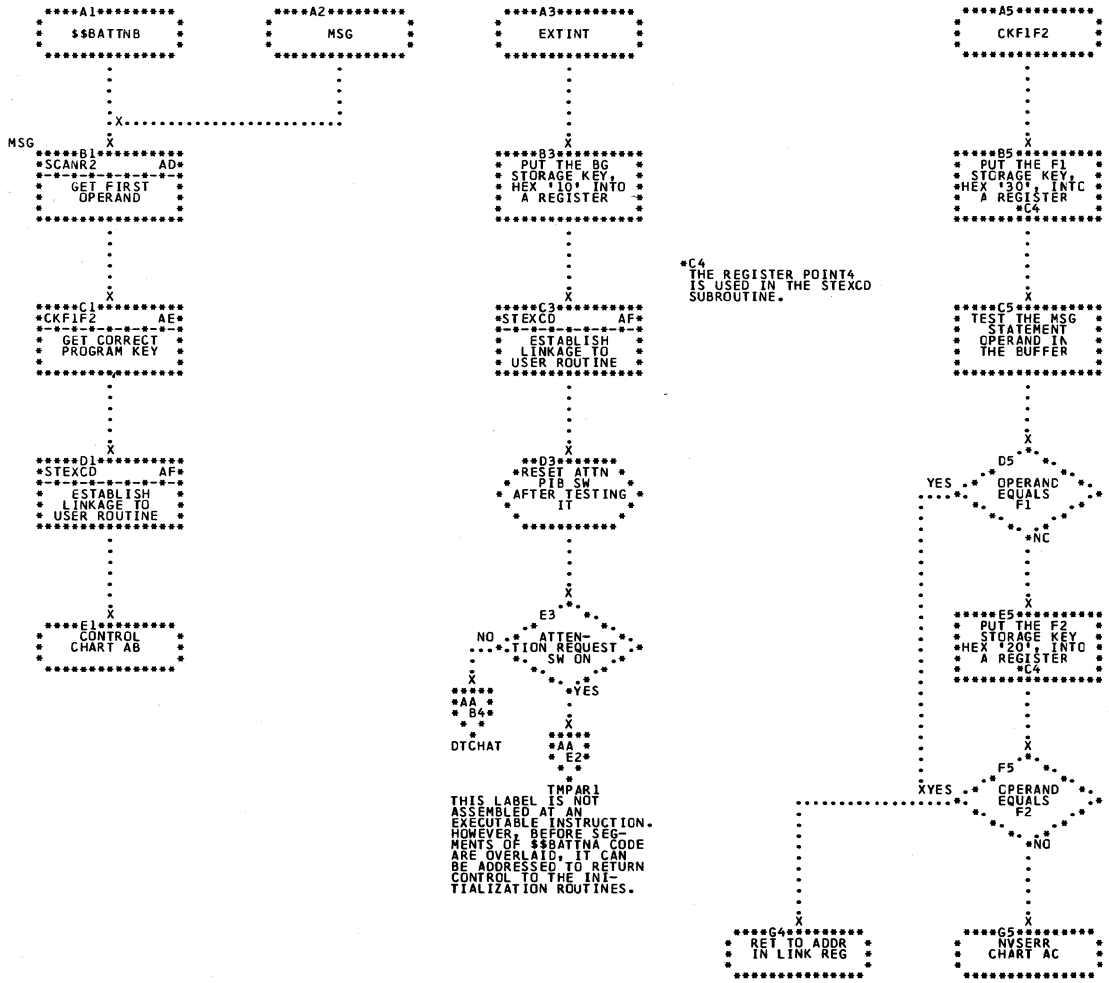


Chart AF. $$BATTNB - Set Operator Communications and Exit Table Linkage
Refer to Chart 03.

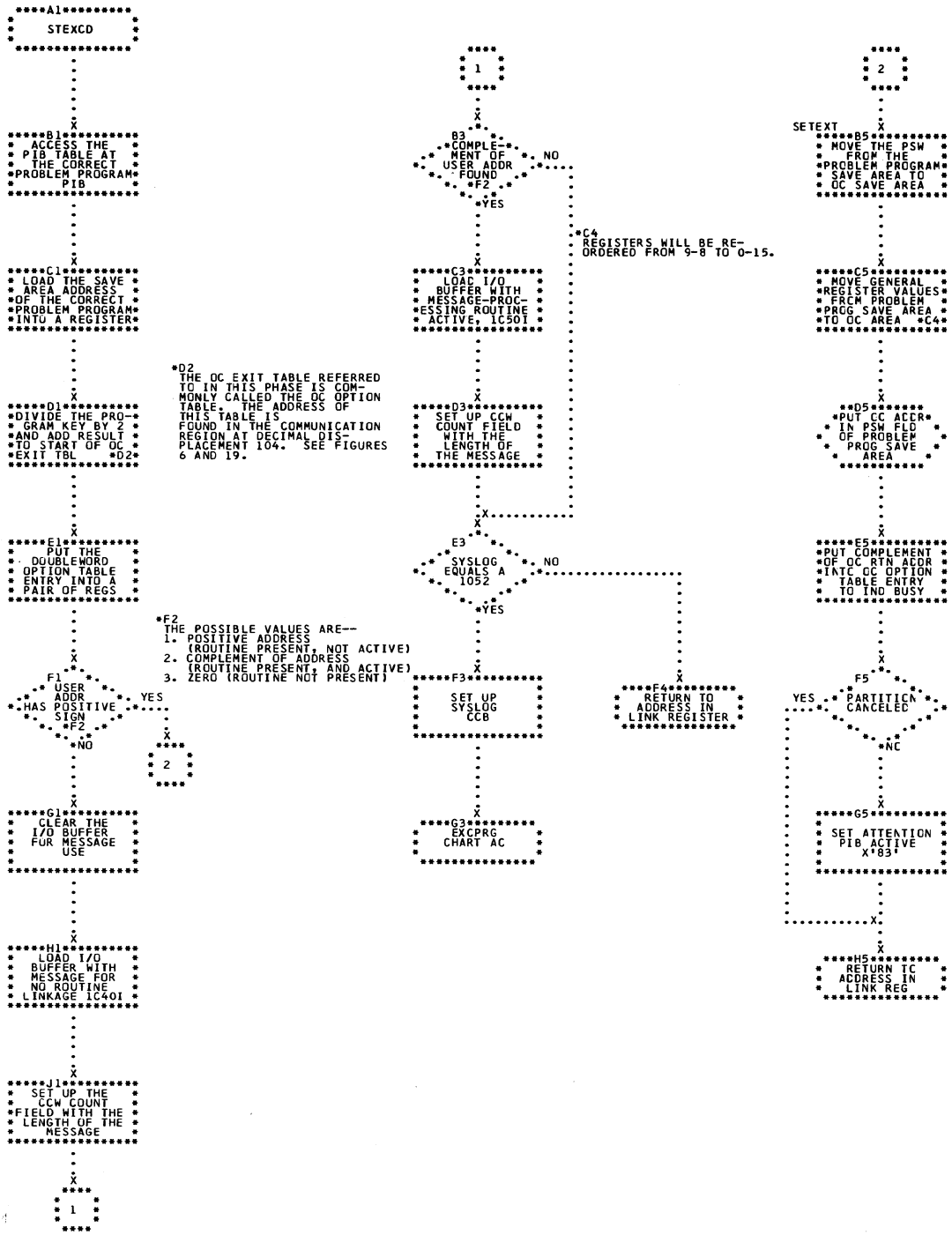


Chart AG. $$BATTNC - CANCEL Statement Processor (Part 1 of 2)
Refer to Chart 03.

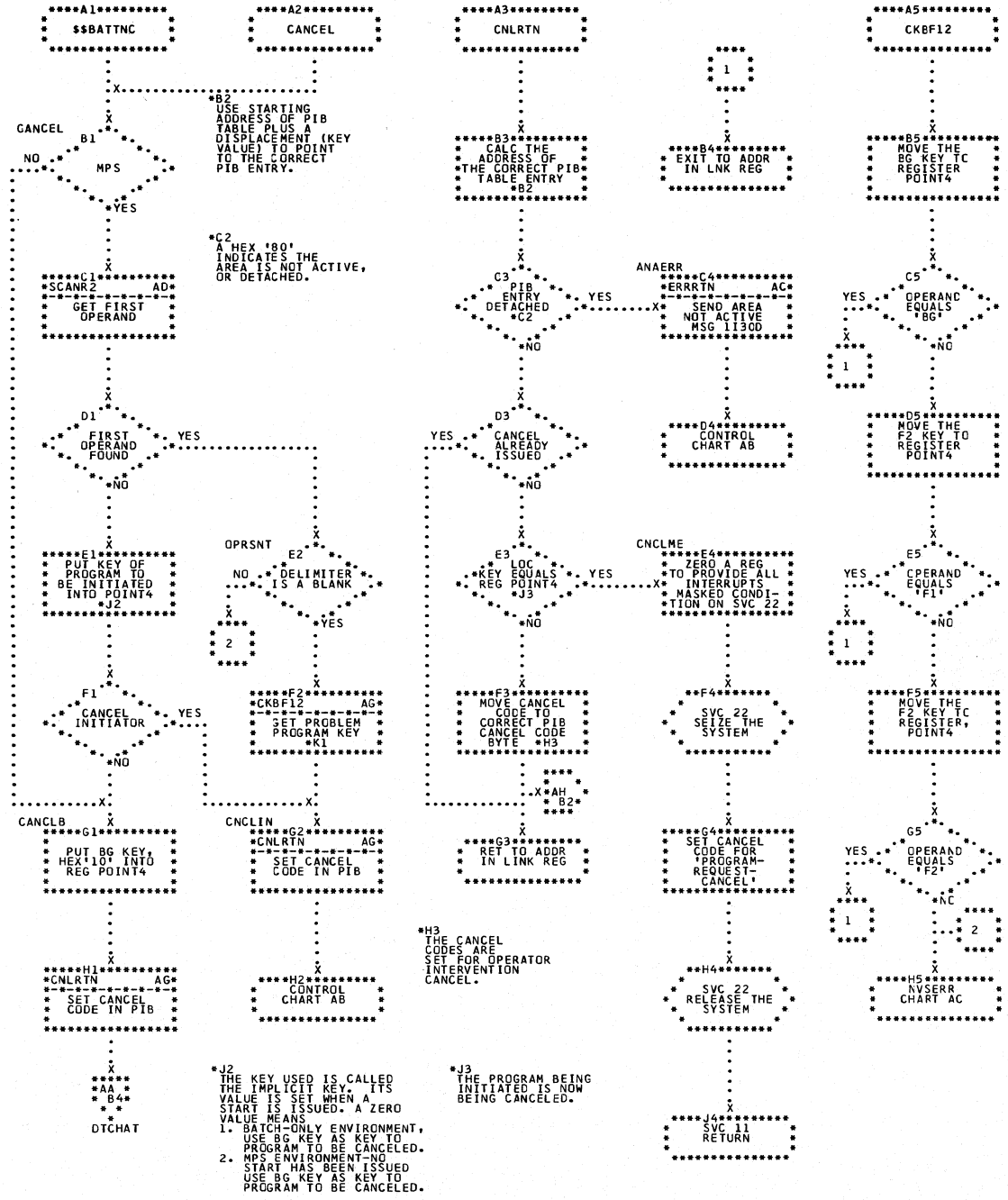


Chart AH. $$BATTNC - CANCEL Statement Processor (Part 2 of 2)
Refer to Chart 03.

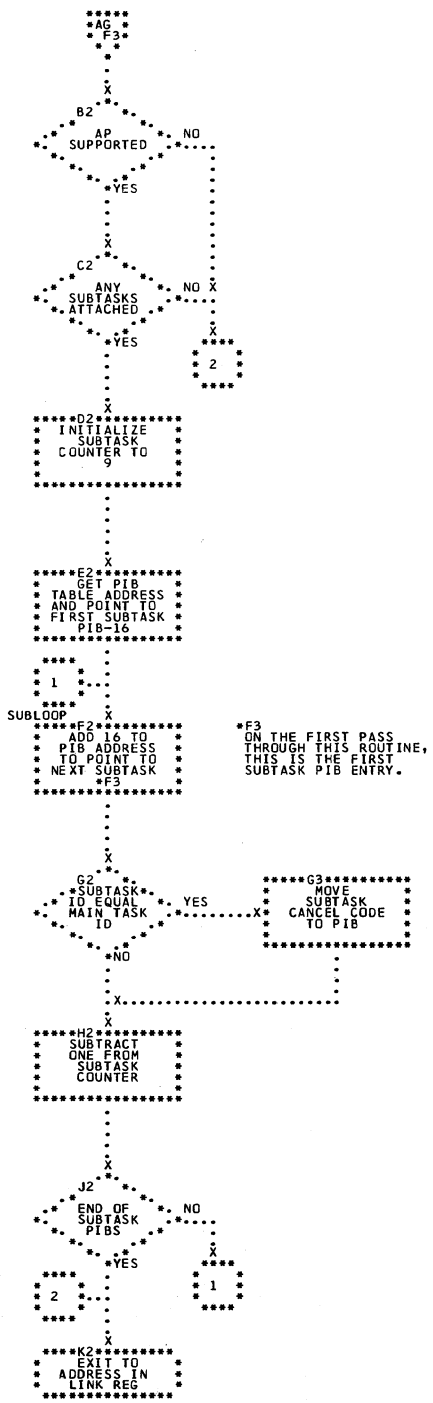


Chart AJ. $$BATTNC - PAUSE, LOG, NOLOG, and IGNORE Statement Processors
Refer to Chart 03.

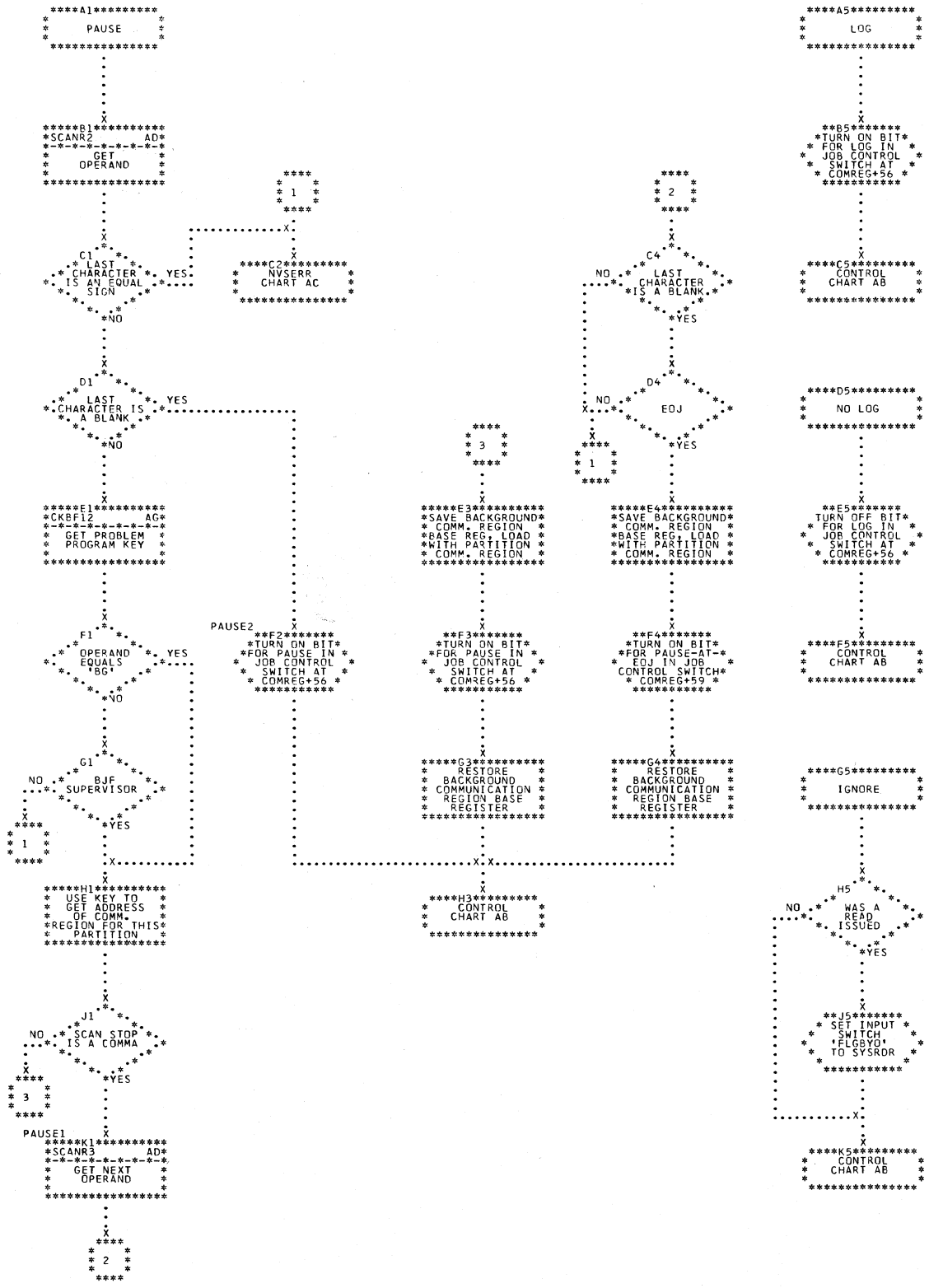


Chart AK. $$BATTND - MAP Statement Processor
Refer to Chart 03.

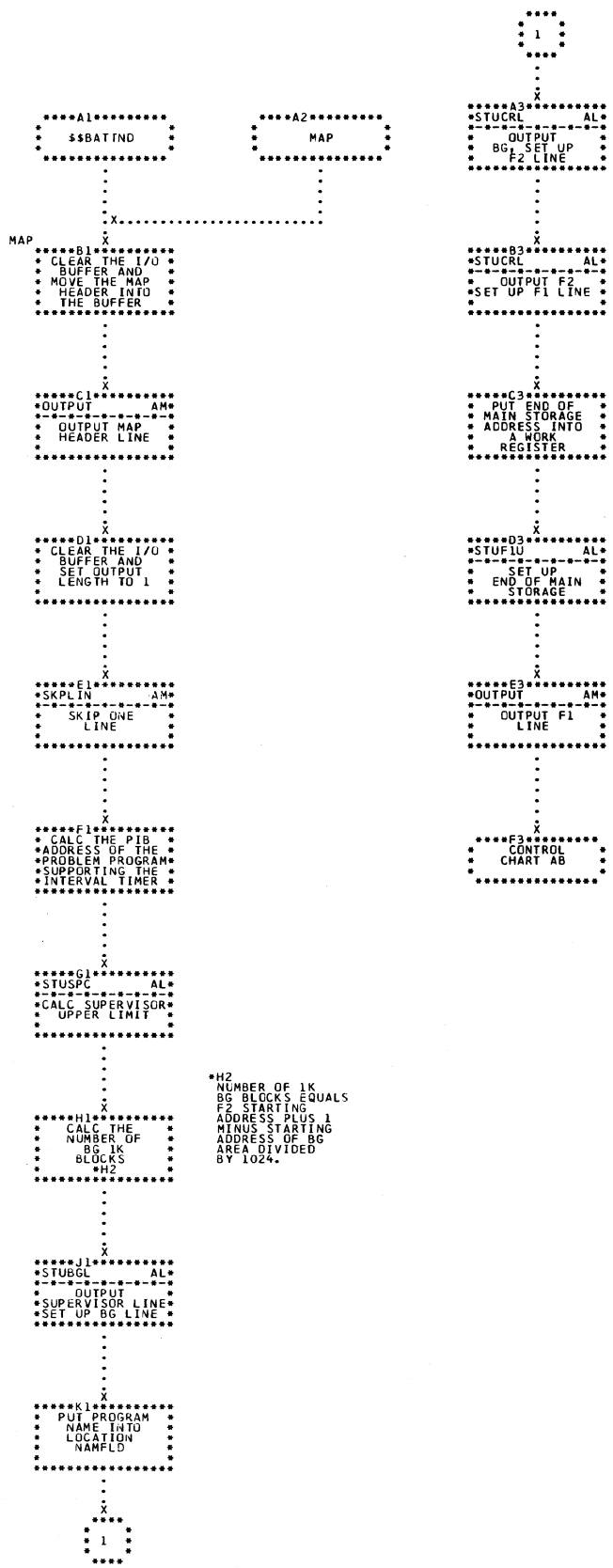


Chart AL. $$BATTND - Output MAP Subroutines (Part 1 of 2)
Refer to Chart 03.

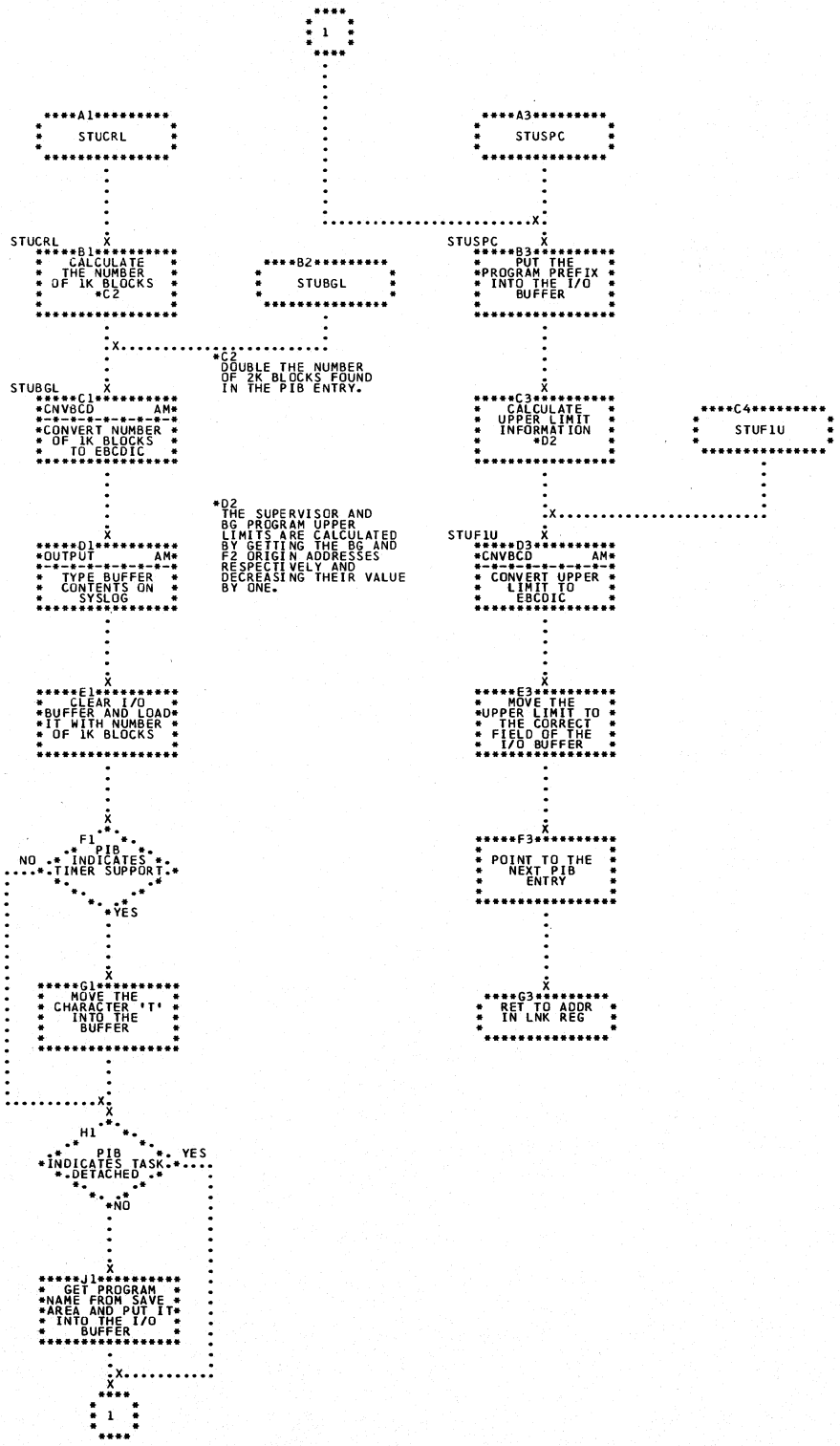


Chart AM. $$BATTND - Output MAP Subroutines (Part 2 of 2)
Refer to Chart 03.

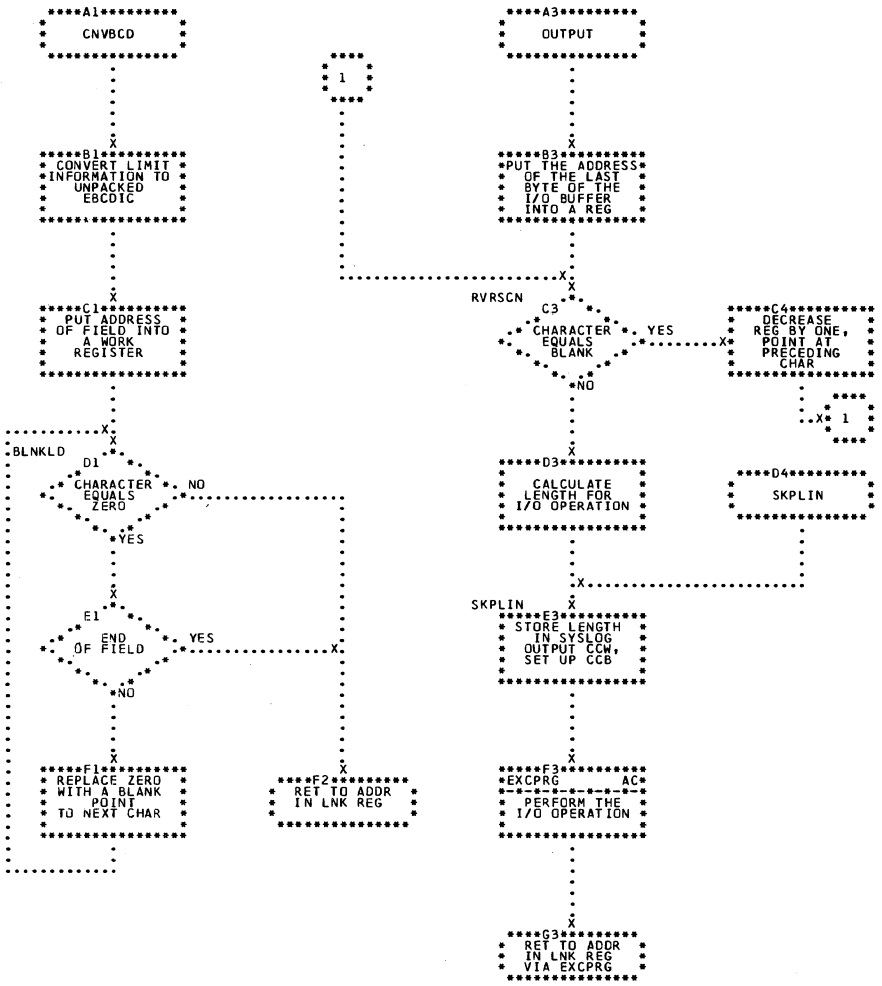


Chart BA. $$BATTNE - ALLOC Statement Processor (Part 1 of 4)
Refer to Chart 04.

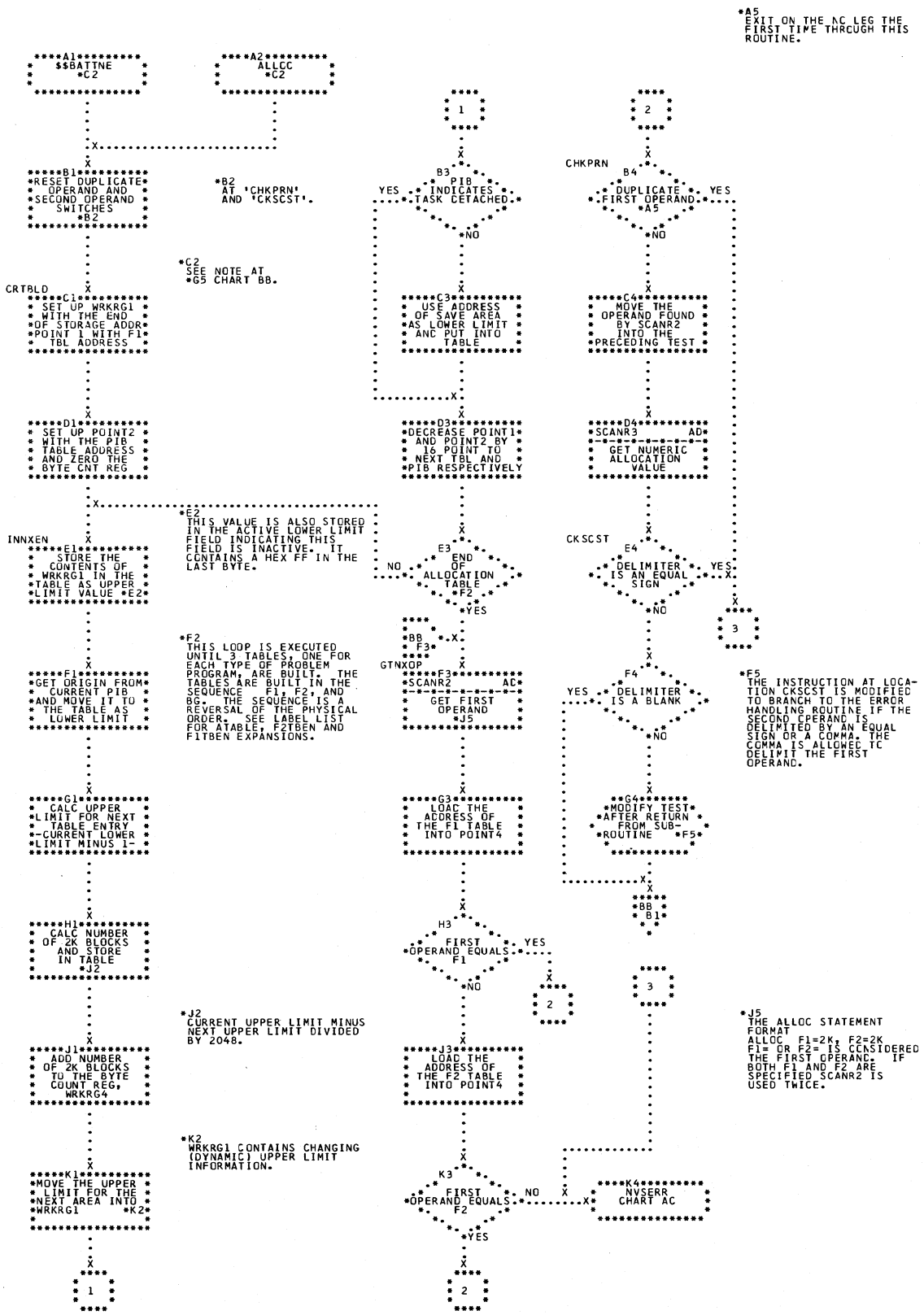


Chart BB. $$BATTNE - ALLOC Statement Processor (Part 2 of 4)
Refer to Chart 04.

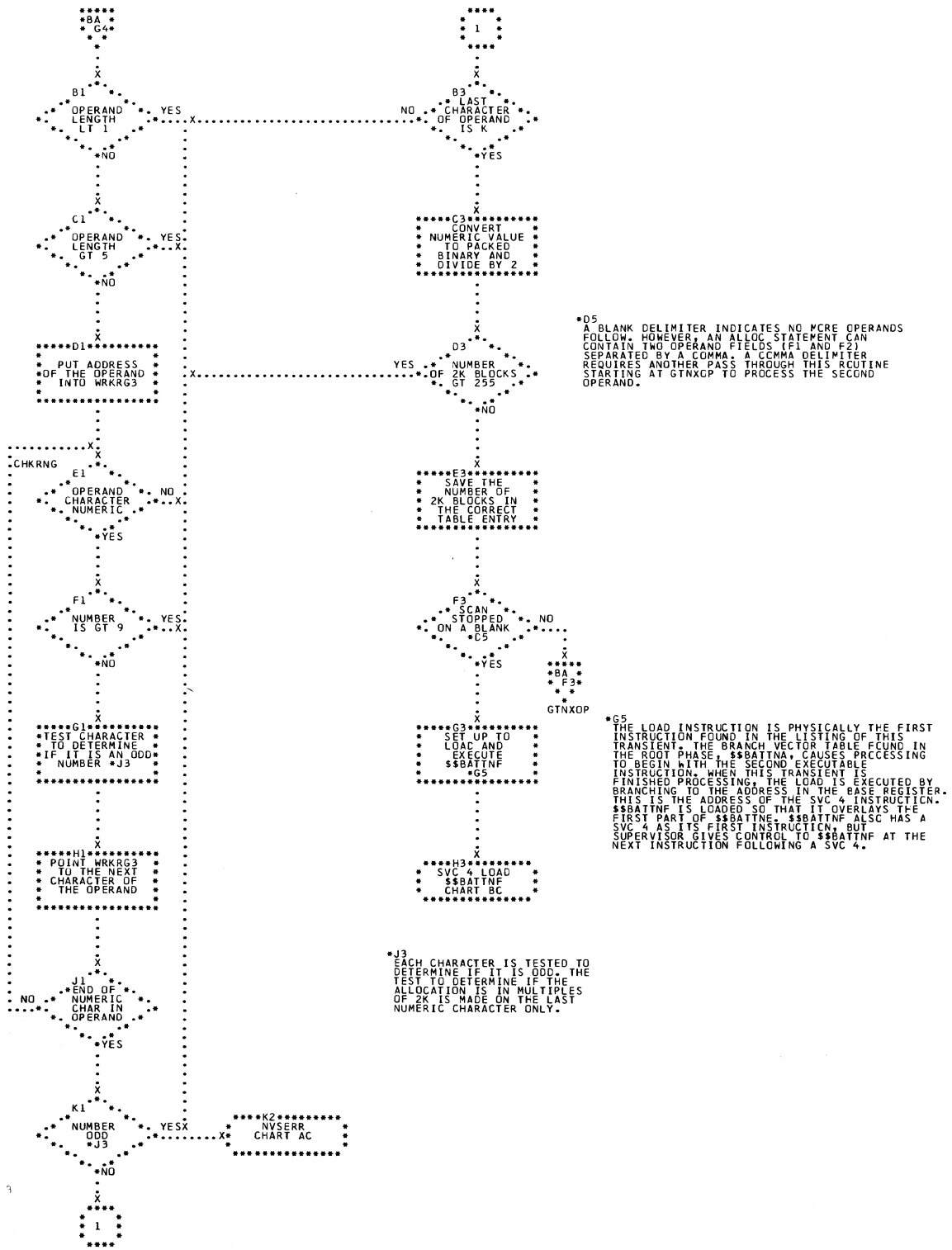


Chart BC. $$BATTNF - ALLOC Statement Processor (Part 3 of 4)
Refer to Chart 04.

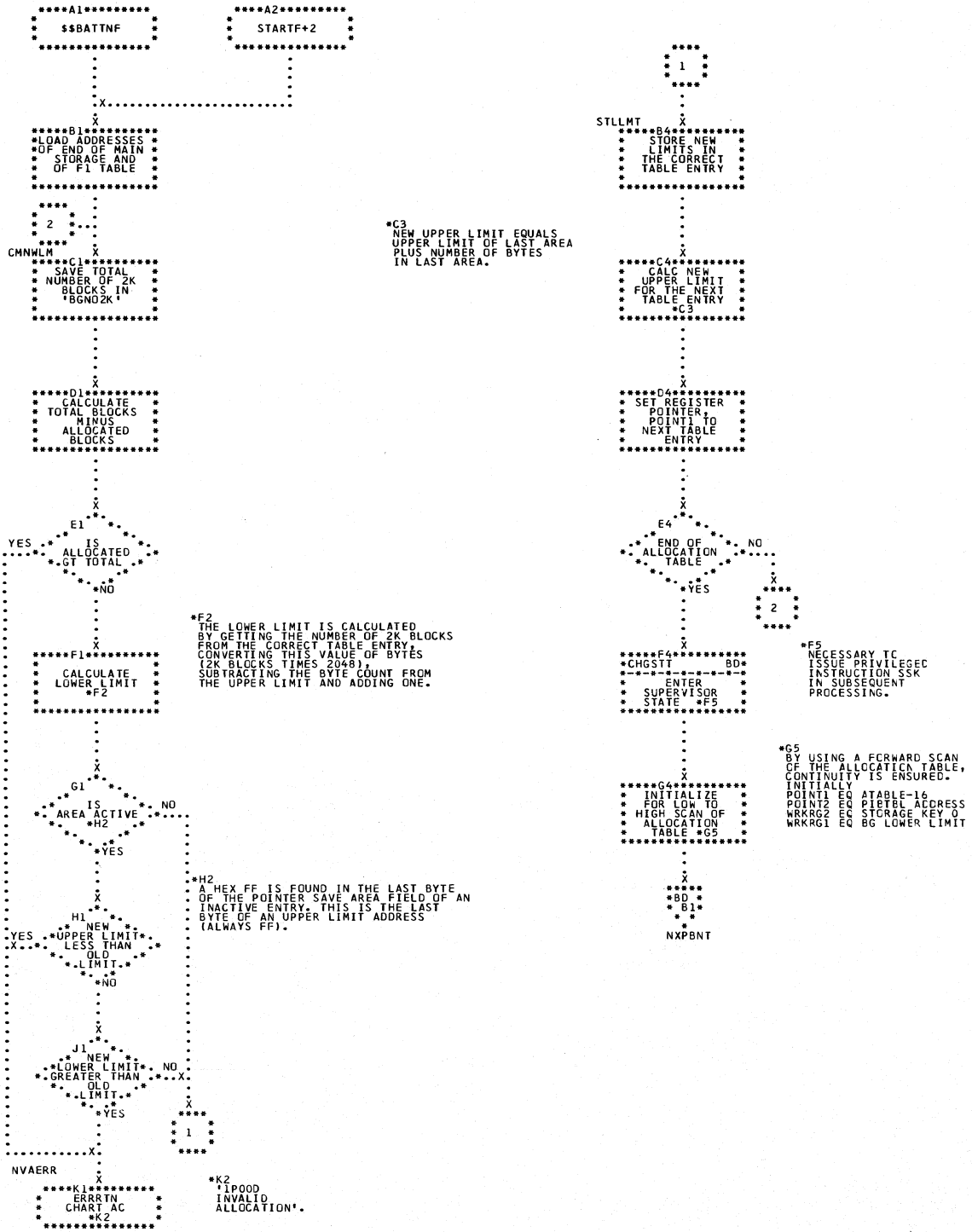


Chart BD. $$BATTNF - ALLOC Statement Processor (Part 4 of 4)
Refer to Chart 04.

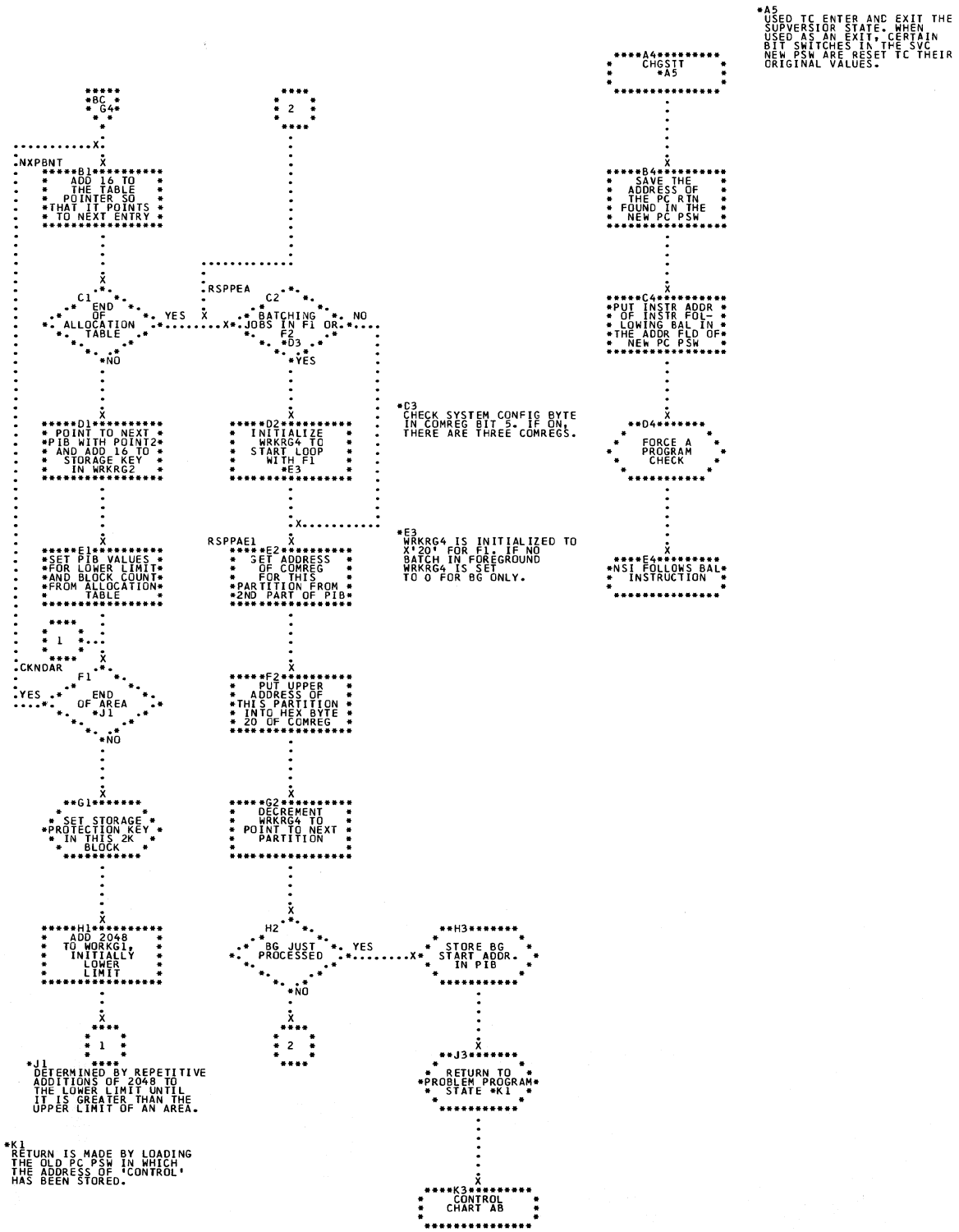


Chart BE. $$BATTNG - START and BATCH Statement Processors

Refer to Chart 04.

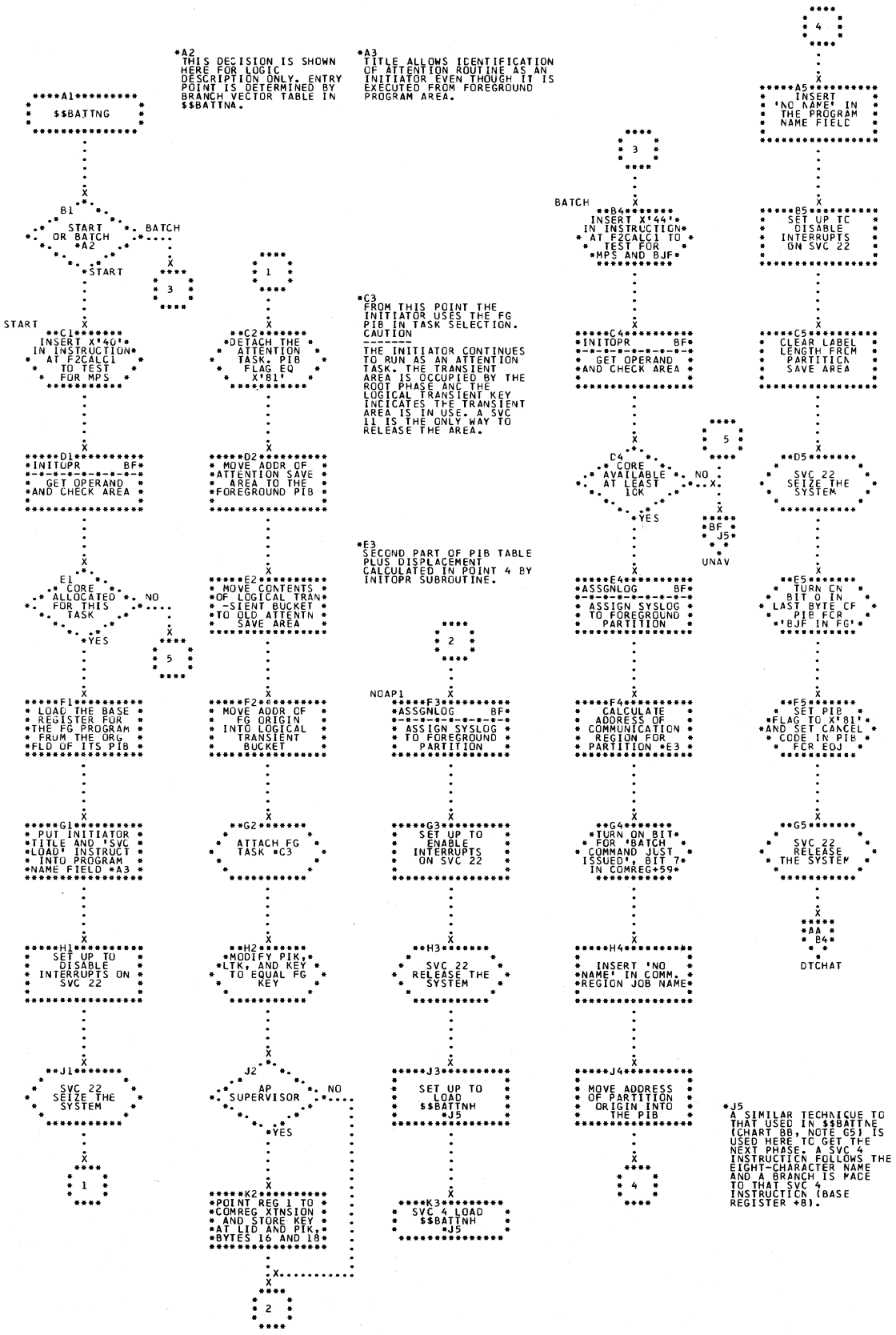


Chart BF. $$BATTNG - START and BATCH Subroutines
Refer to Chart 04.

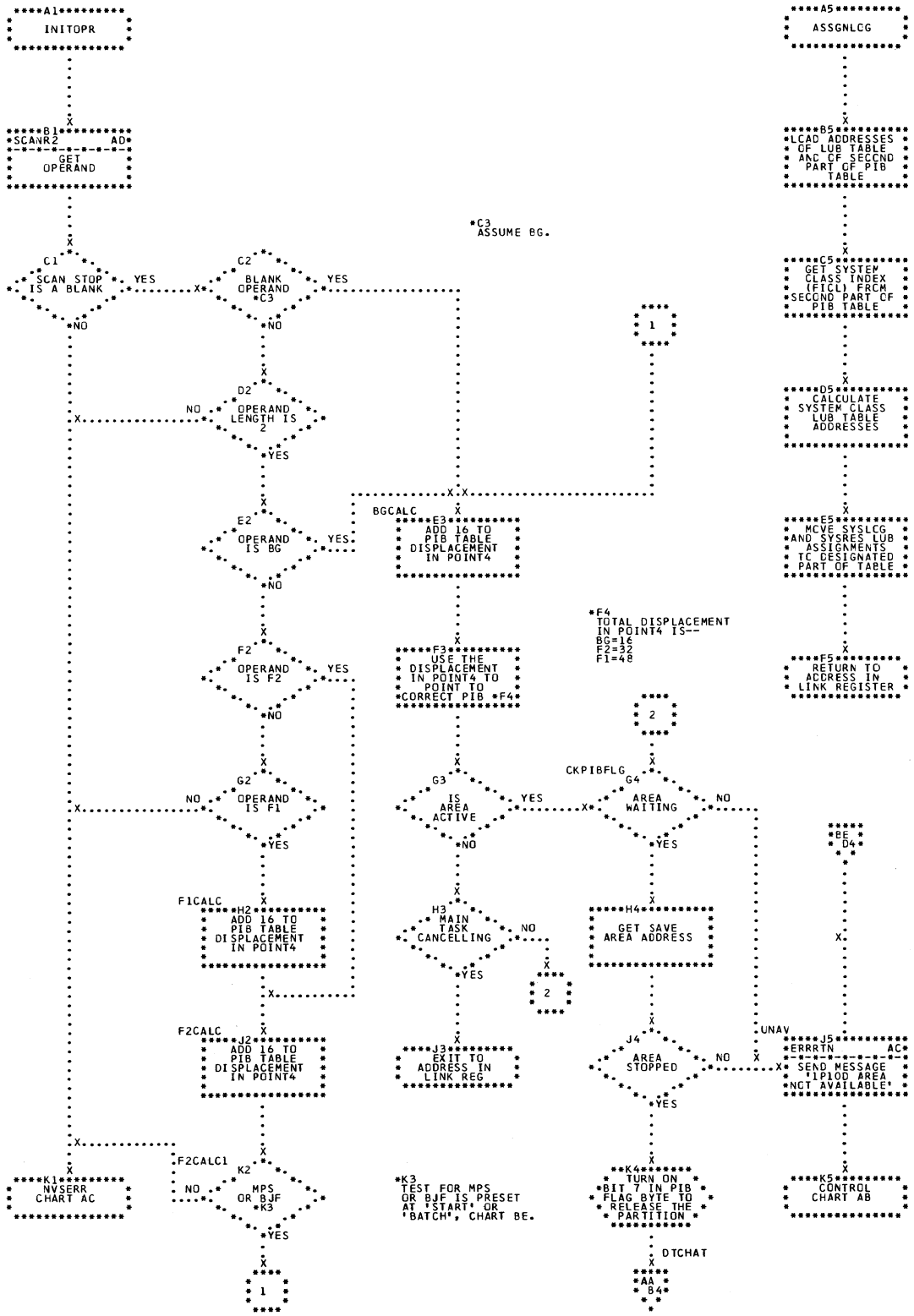


Chart BG. $$BATTNH - START Statement Processor Channel Program
Refer to Chart 04.

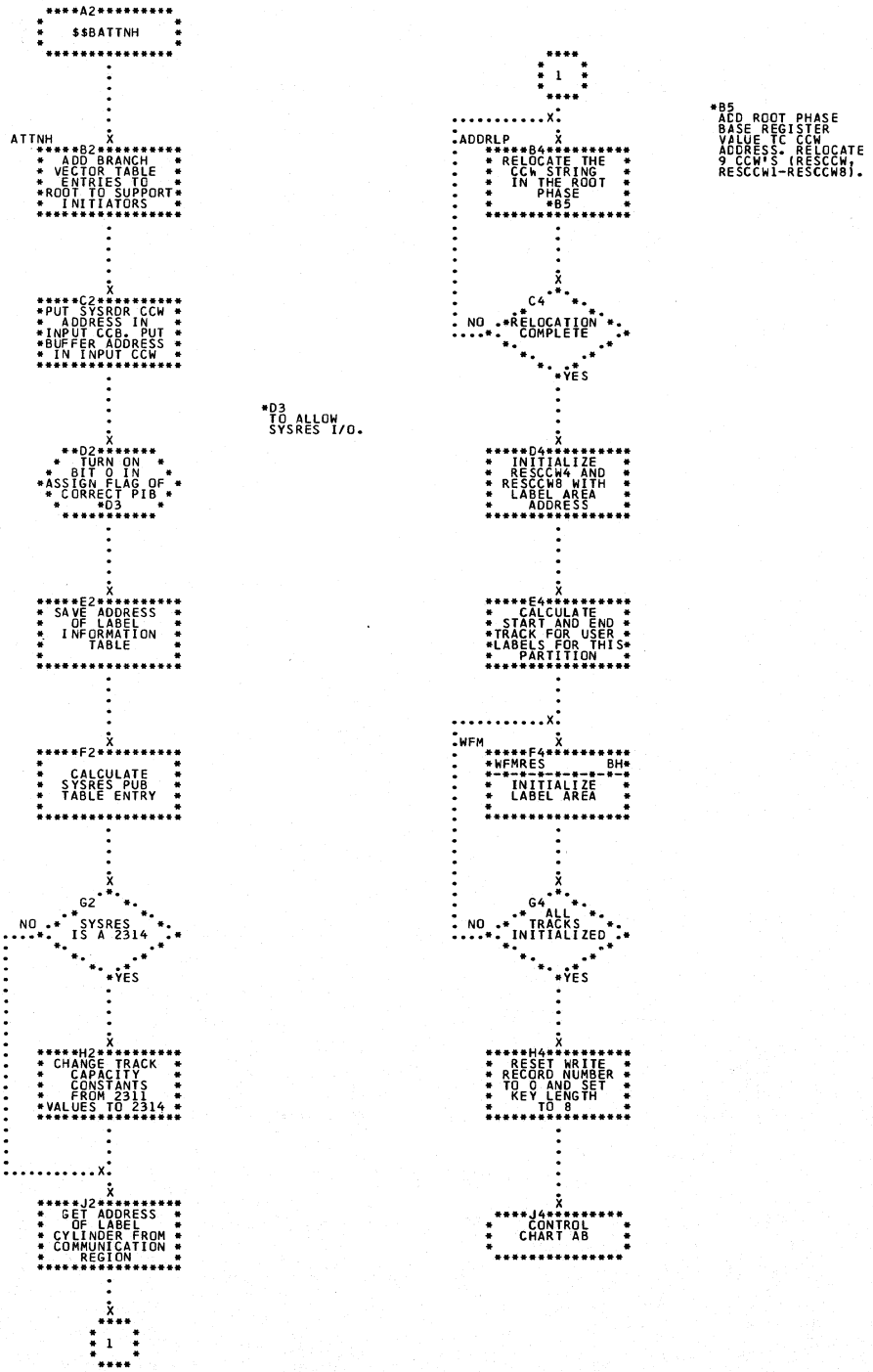


Chart BH. $$BATTNH - Subroutines
Refer to Chart 04.

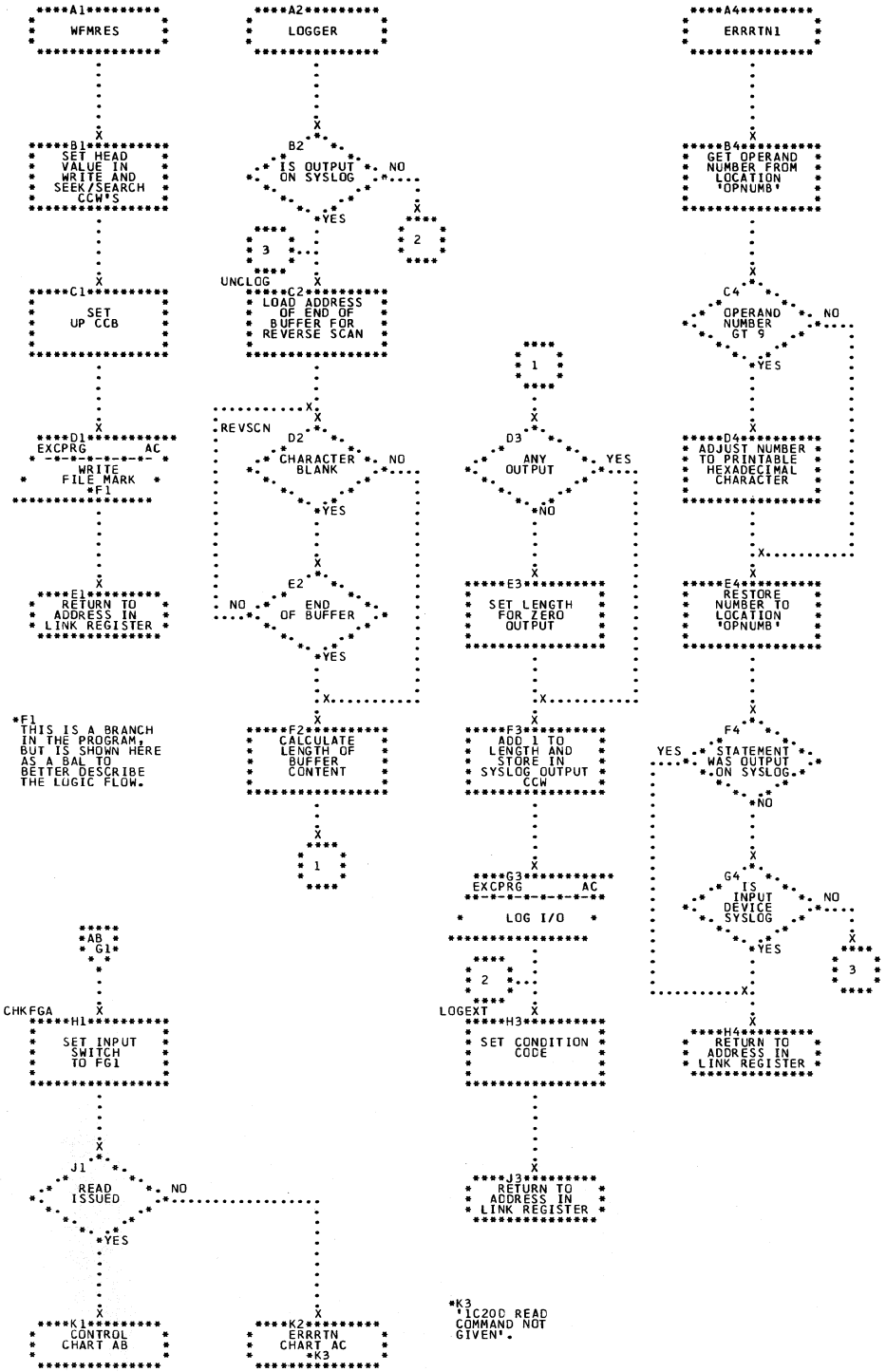


Chart CA. $$BATTNI - ASSGN Statement Processor (Part 1 of 2)
Refer to Chart 05.

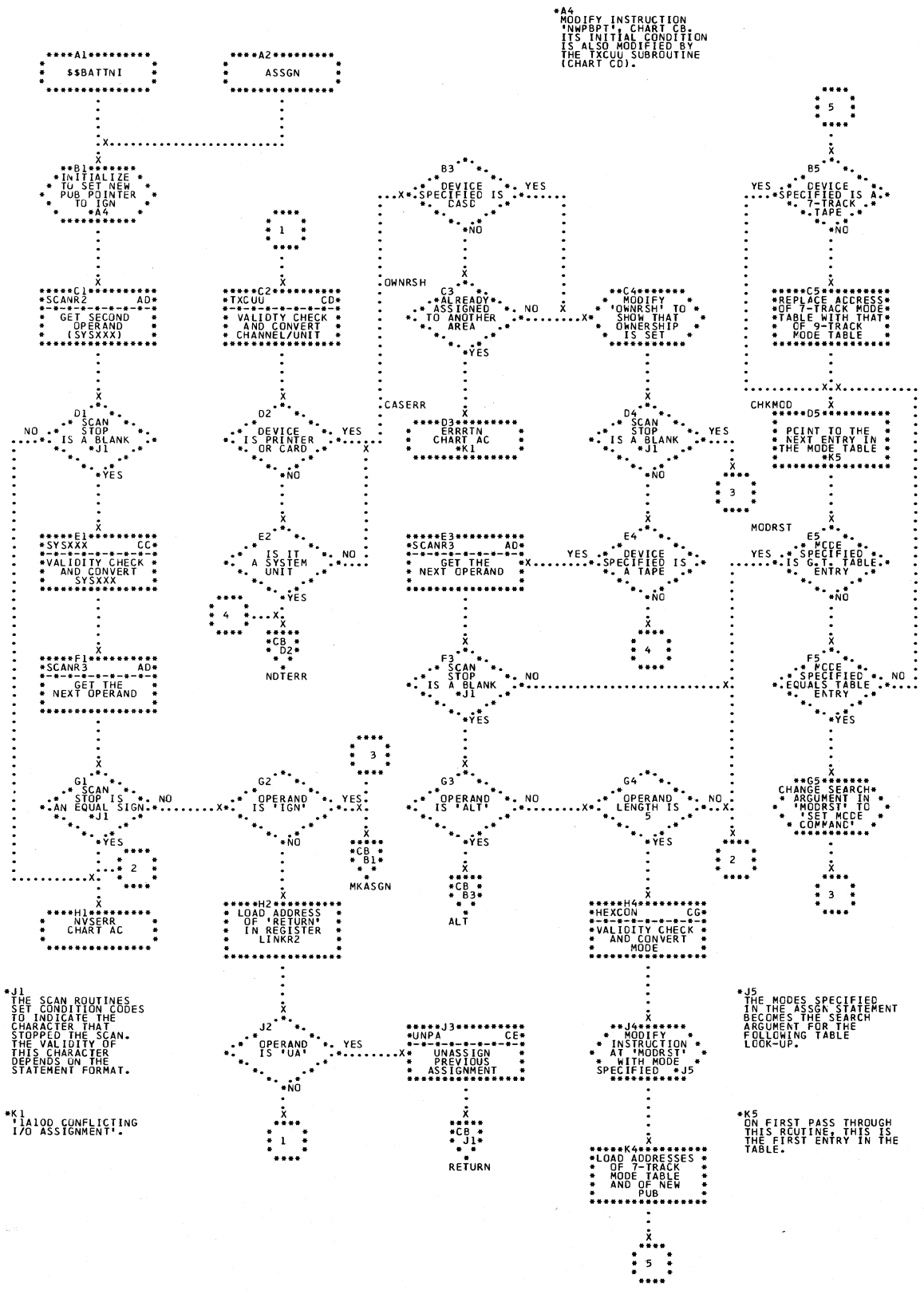


Chart CB. $$BATTNI - ASSGN Statement Processor (Part 2 of 2)
Refer to Chart 05.

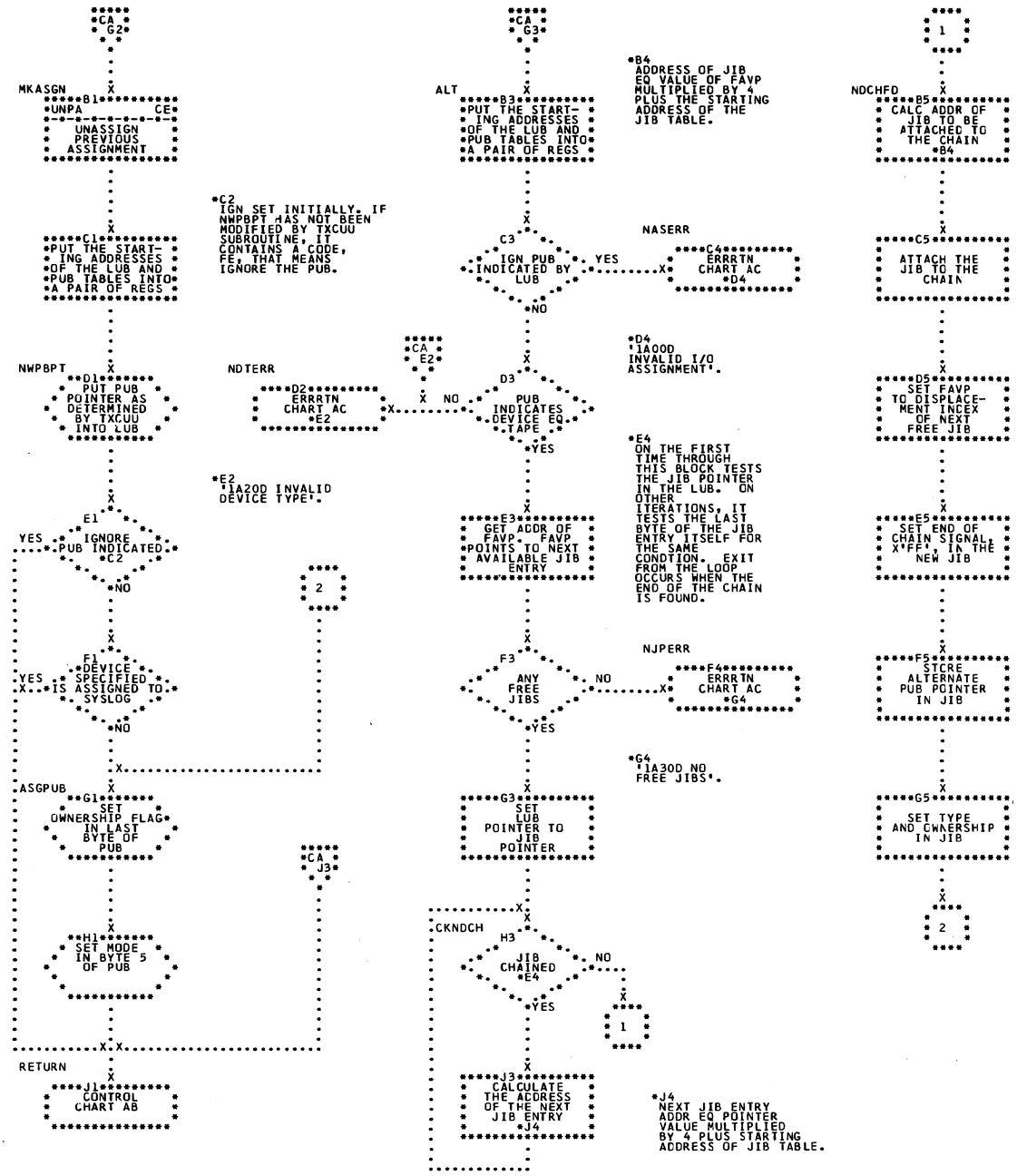


Chart CC. $$BATTNI - Validate SYSXXX Subroutine
Refer to Chart 05.

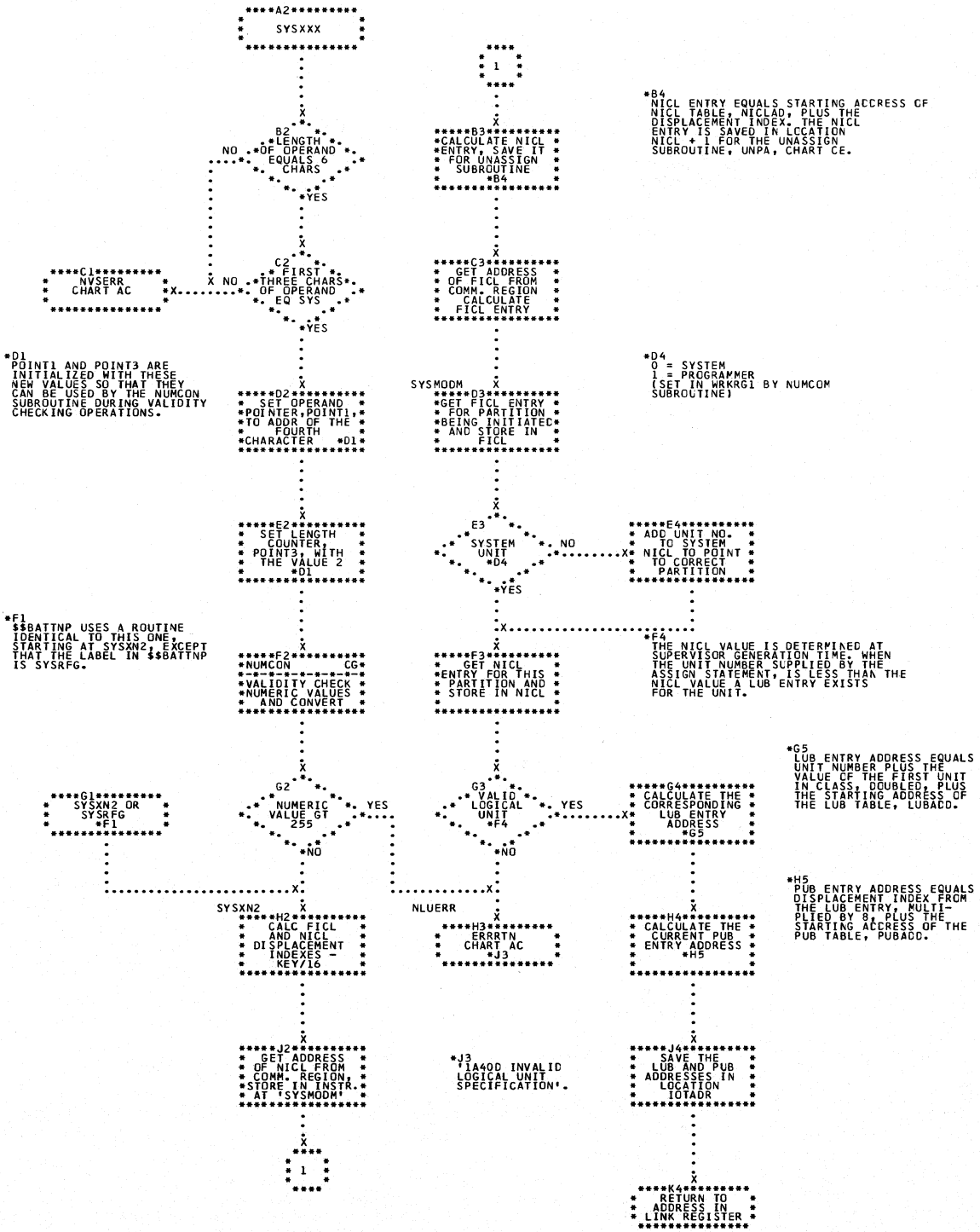


Chart CD. $$BATTNI - Validity Check Channel and Unit
Refer to Chart 05.

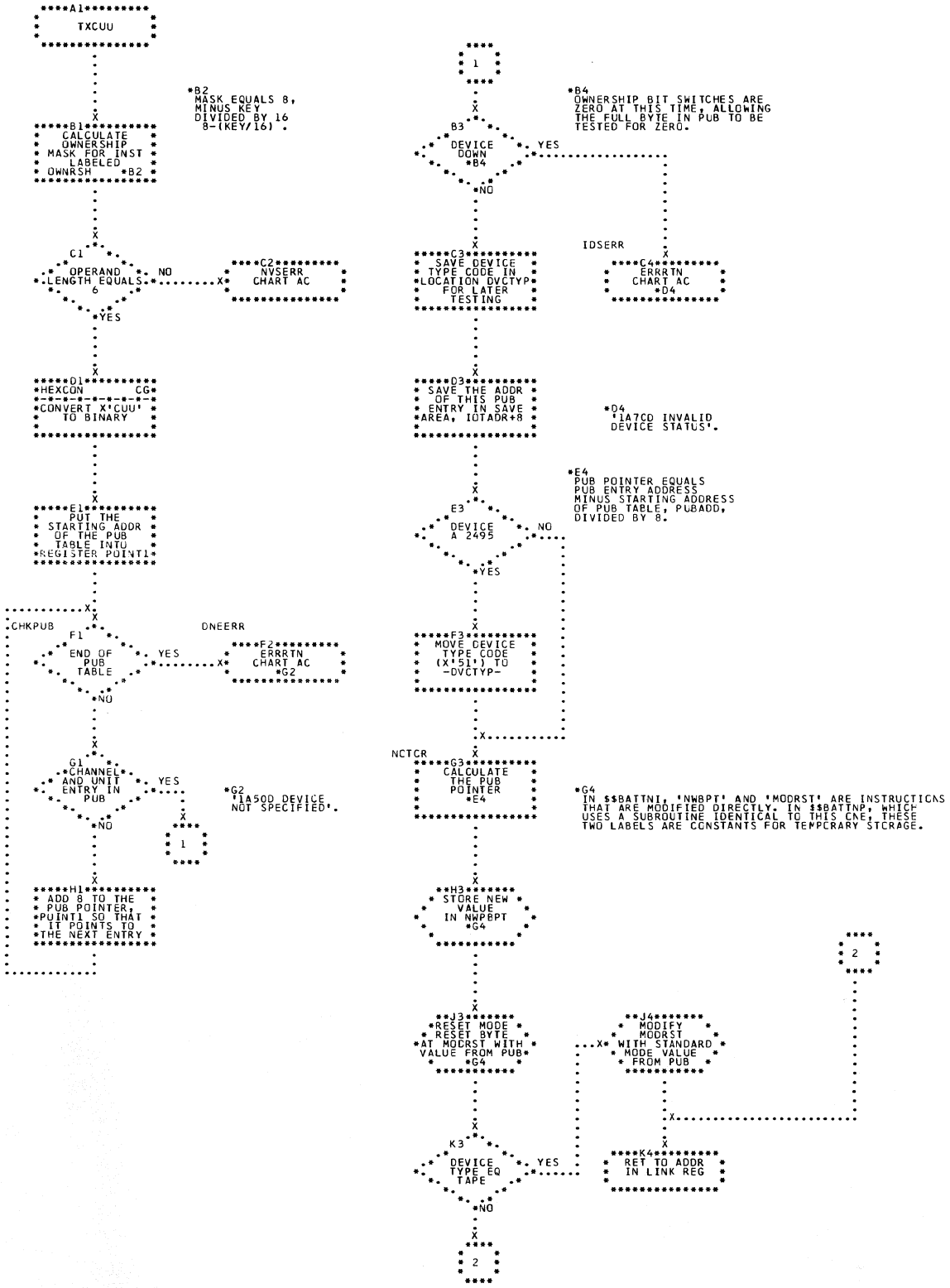


Chart CE. $$BATTNI - Unassign Subroutine
Refer to Chart 05.

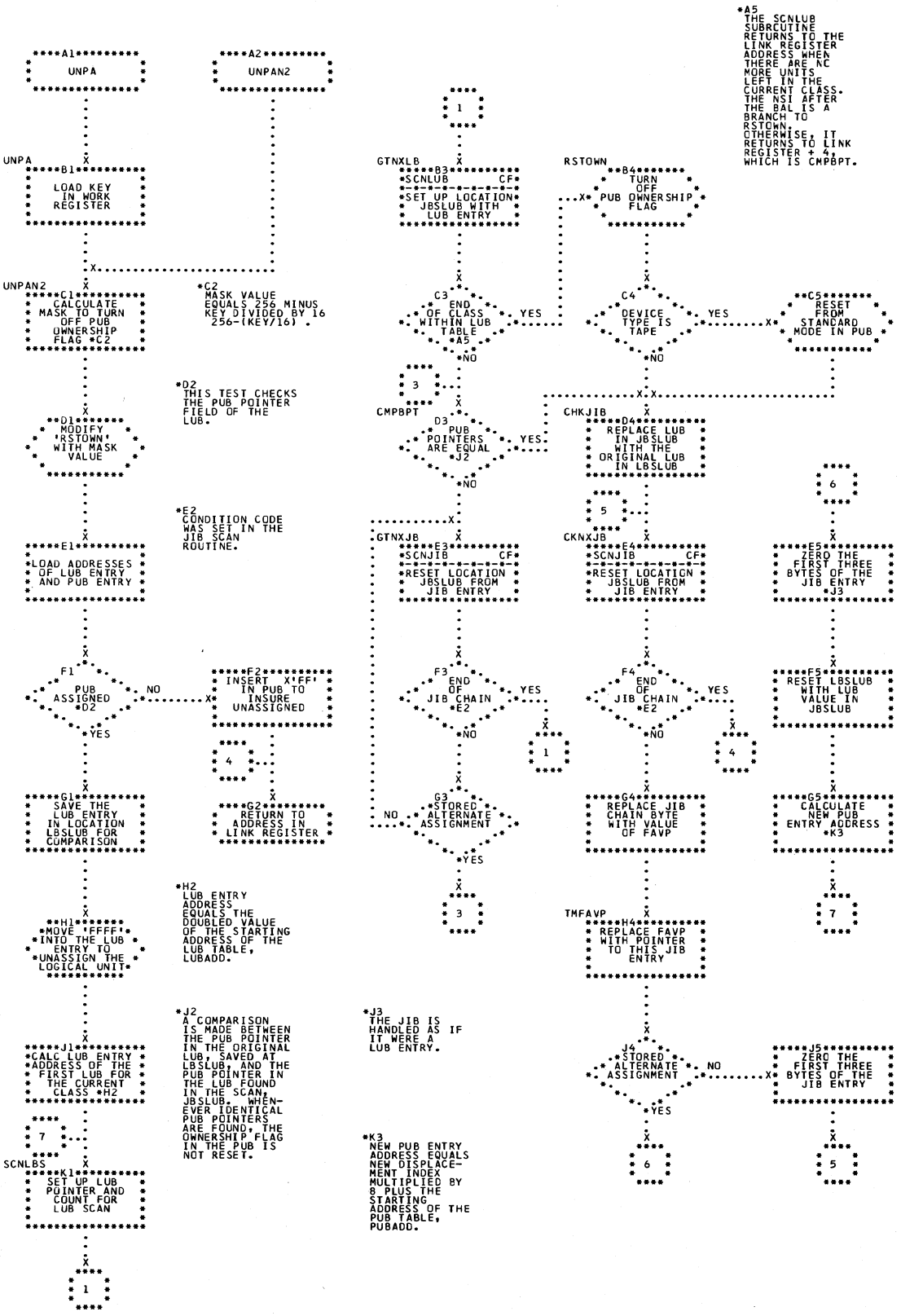


Chart CF. $$BATTNI - Scan LUBs and JIBs Subroutines
Refer to Chart 05.

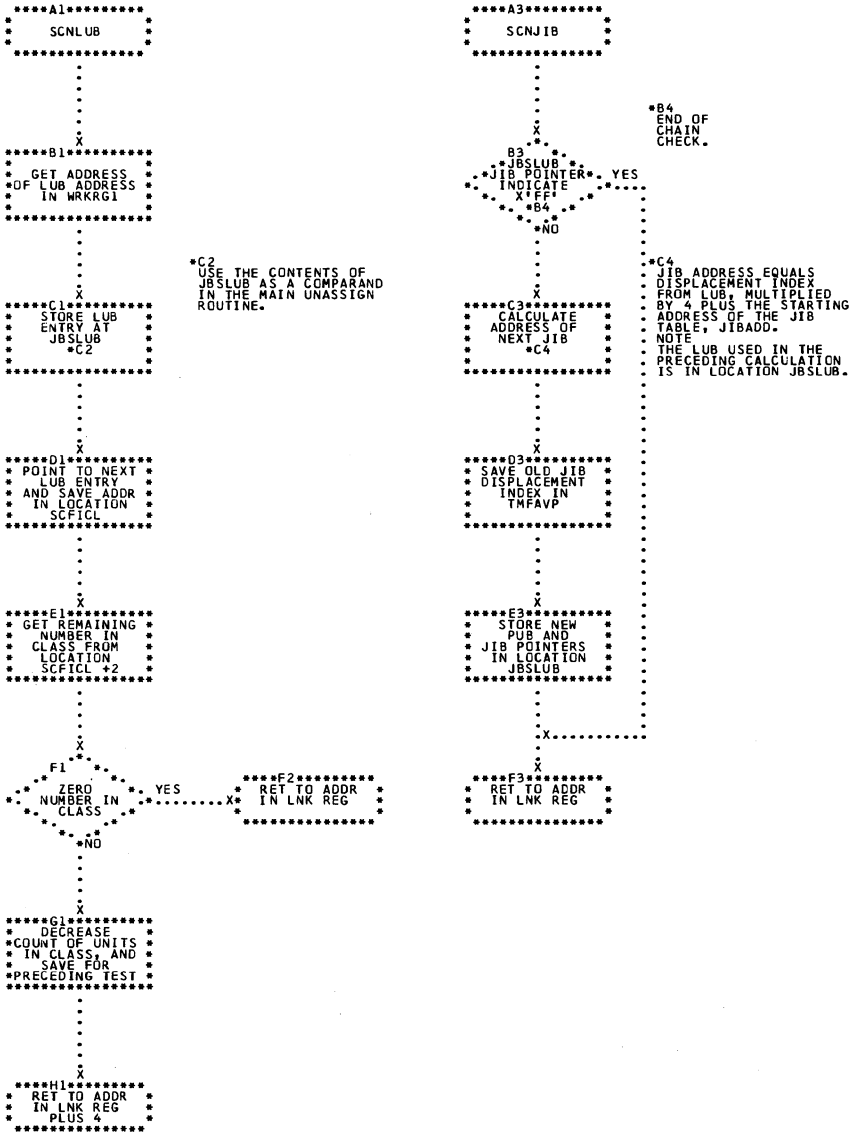


Chart CG. $$BATTNI - Conversion Subroutines
Refer to Chart 05.

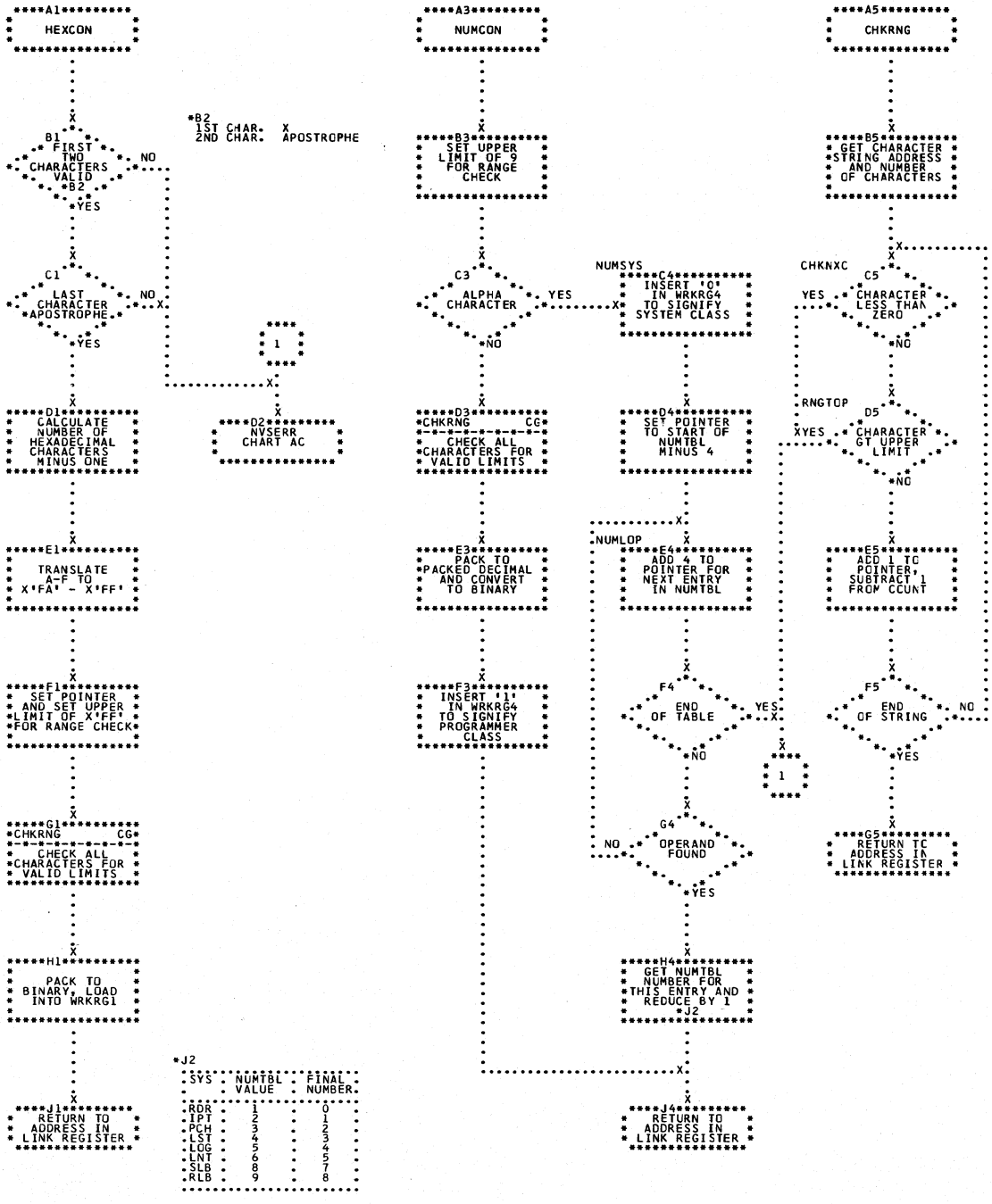


Chart CH. $$BATTNI - UNA Statement Processor
Refer to Chart 05.

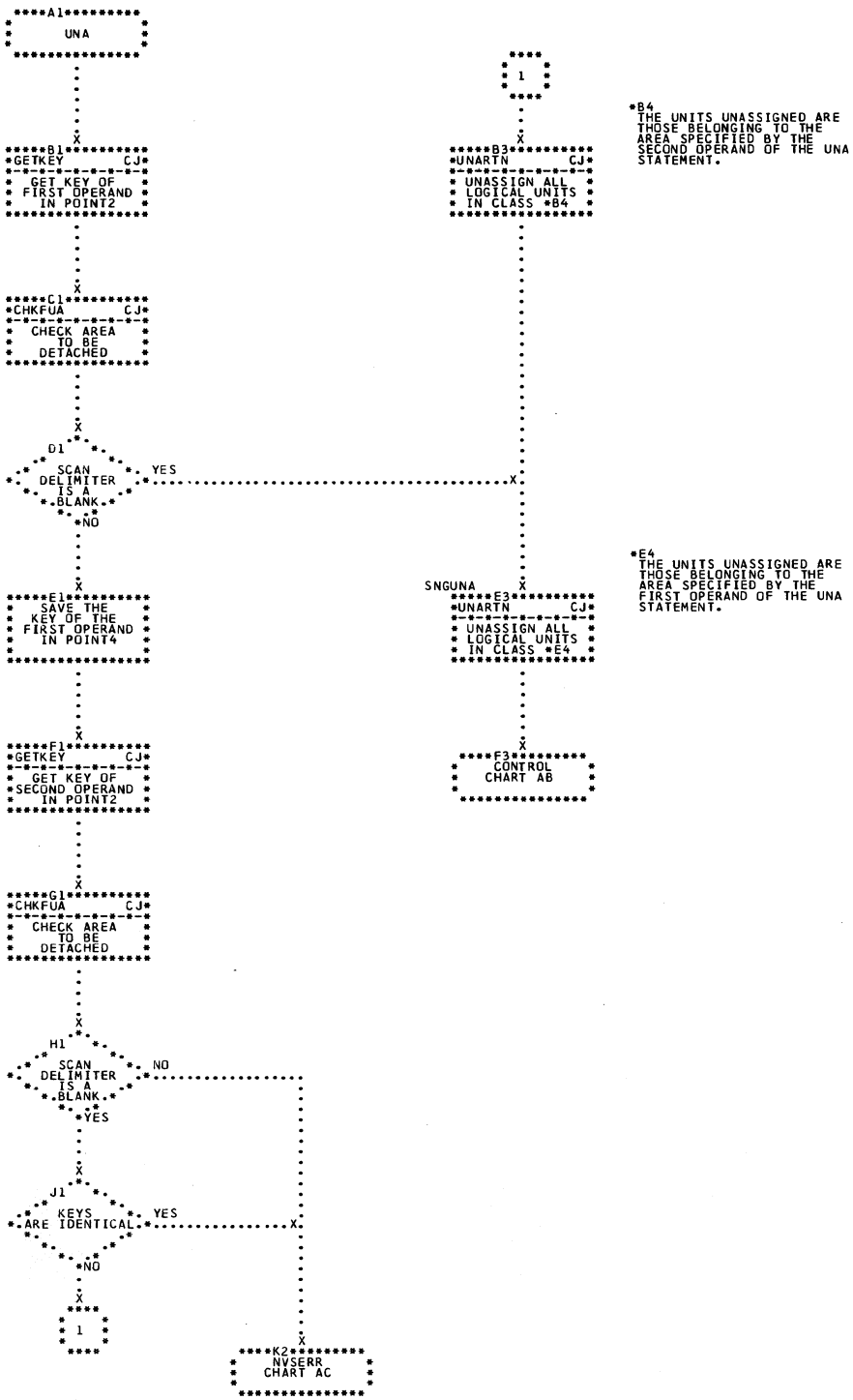


Chart CJ. $$BATTNI - Miscellaneous Subroutines
Refer to Chart 05.

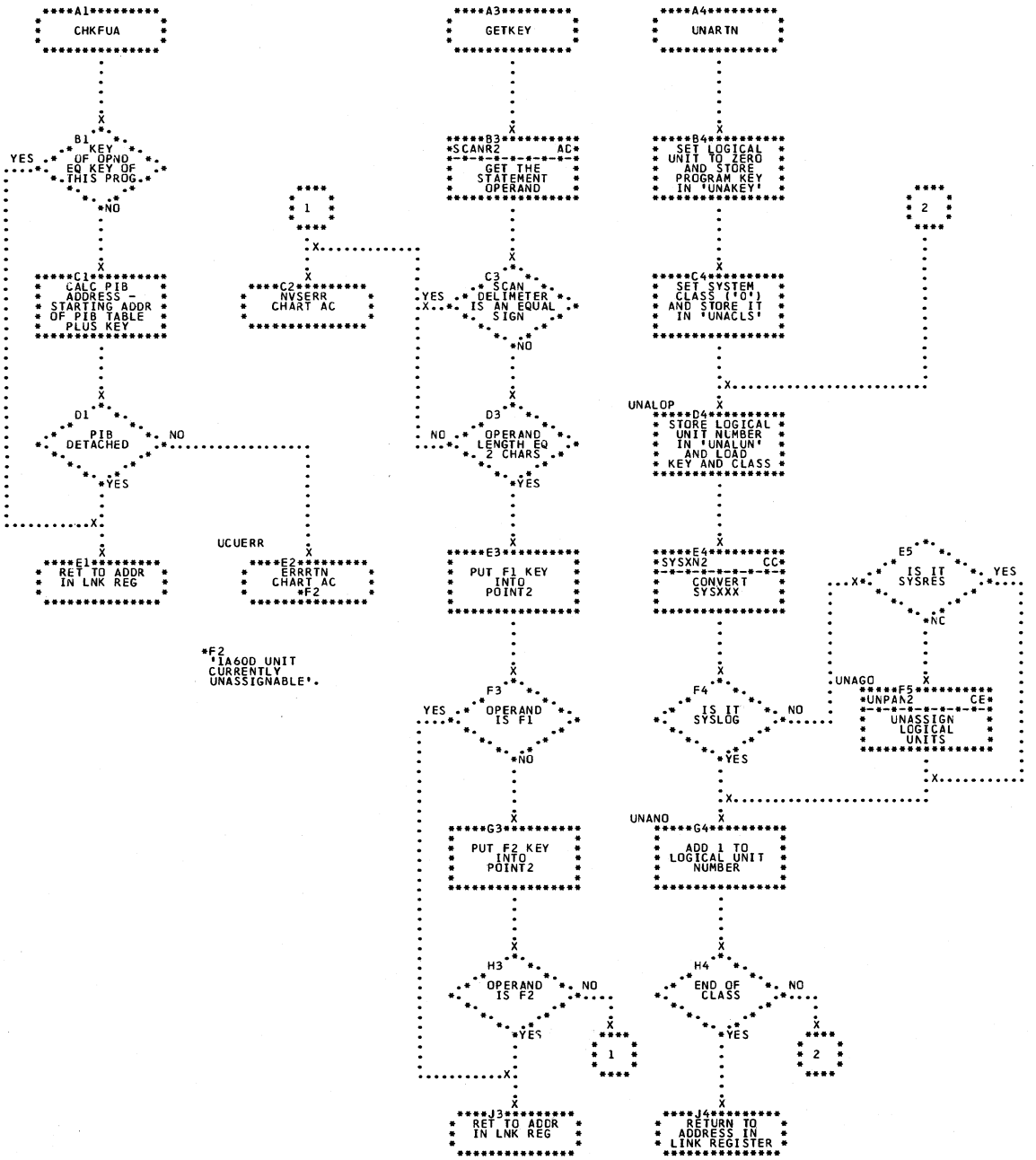


Chart CK. $$BATTNJ - LISTIO Statement Processor
Refer to Chart 05.

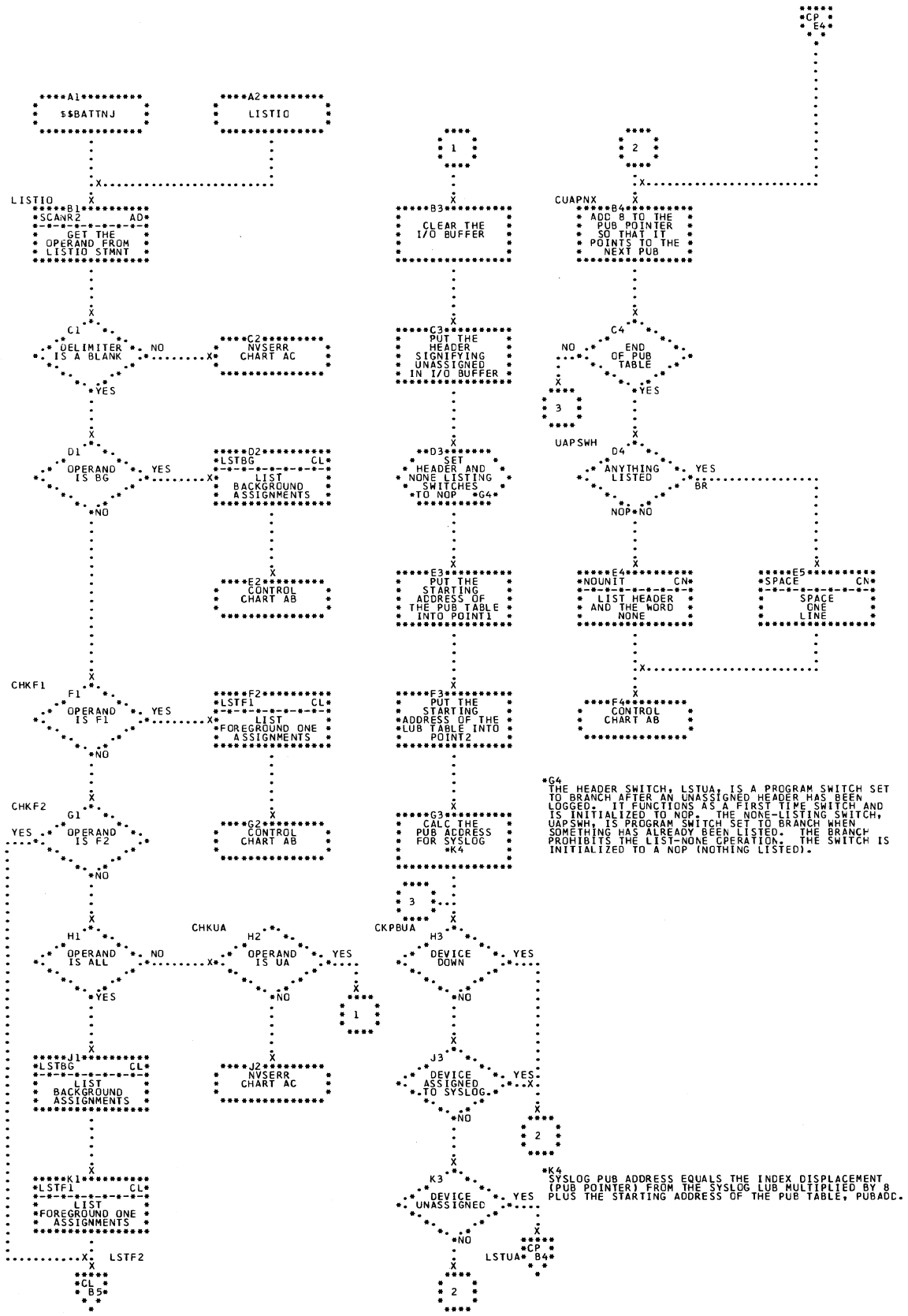


Chart CL. $$BATTNJ - Subroutines
Refer to Chart 05.

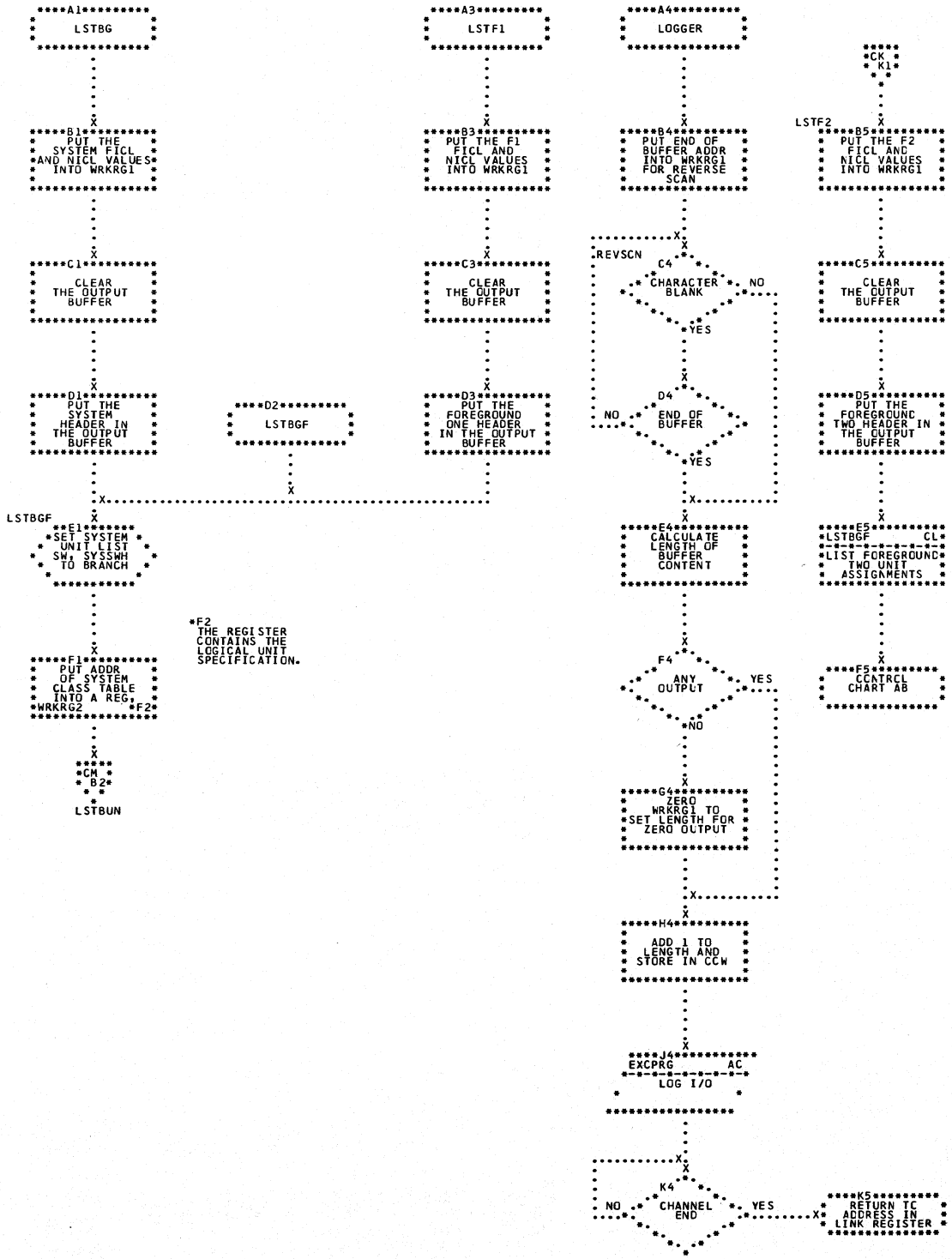


Chart CM. $$BATTNJ - Locate Assignment Routine
Refer to Chart 05.

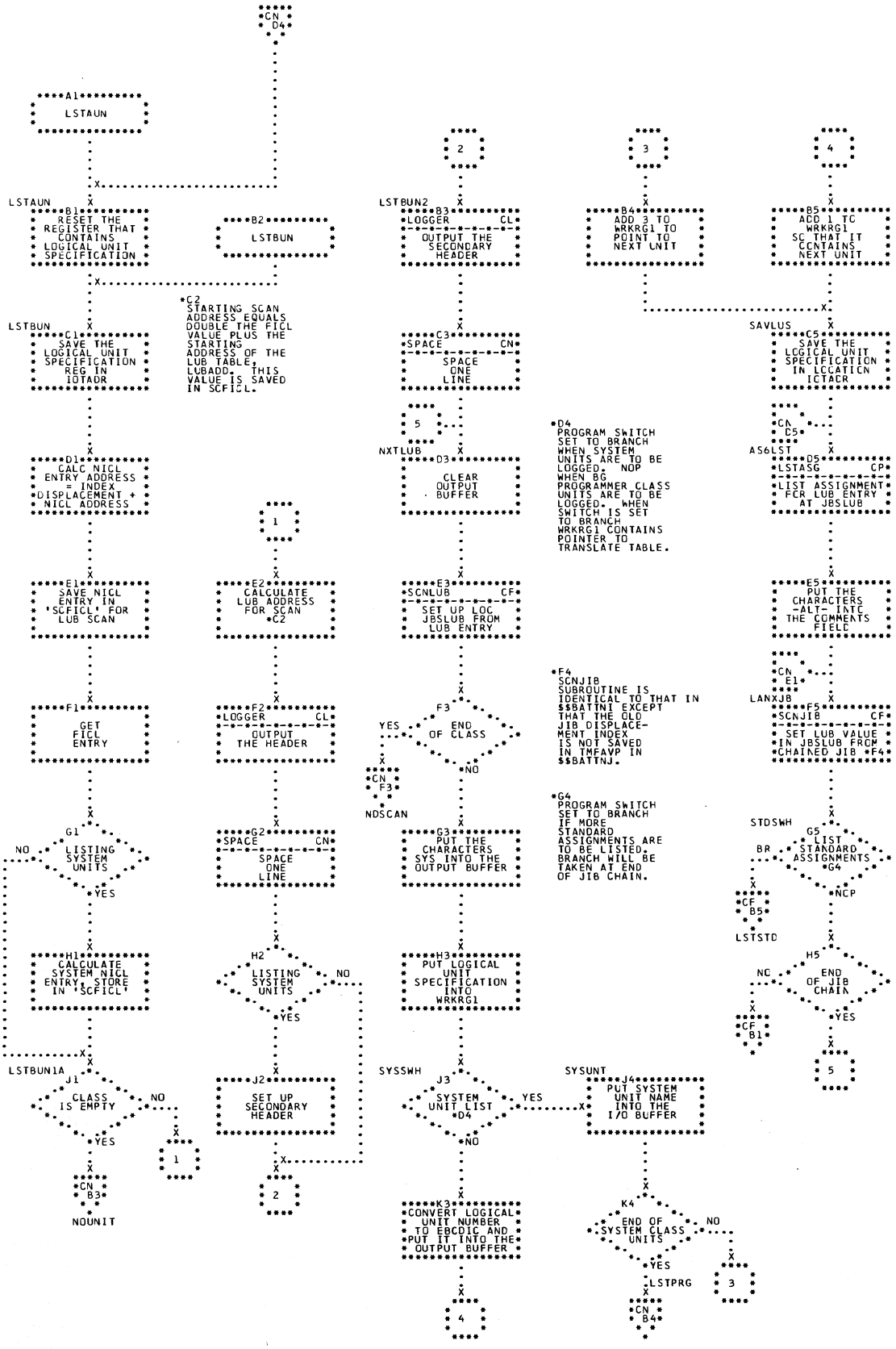


Chart CN. $$BATTNJ - Output List (Part 1 of 2)
Refer to Chart 05.

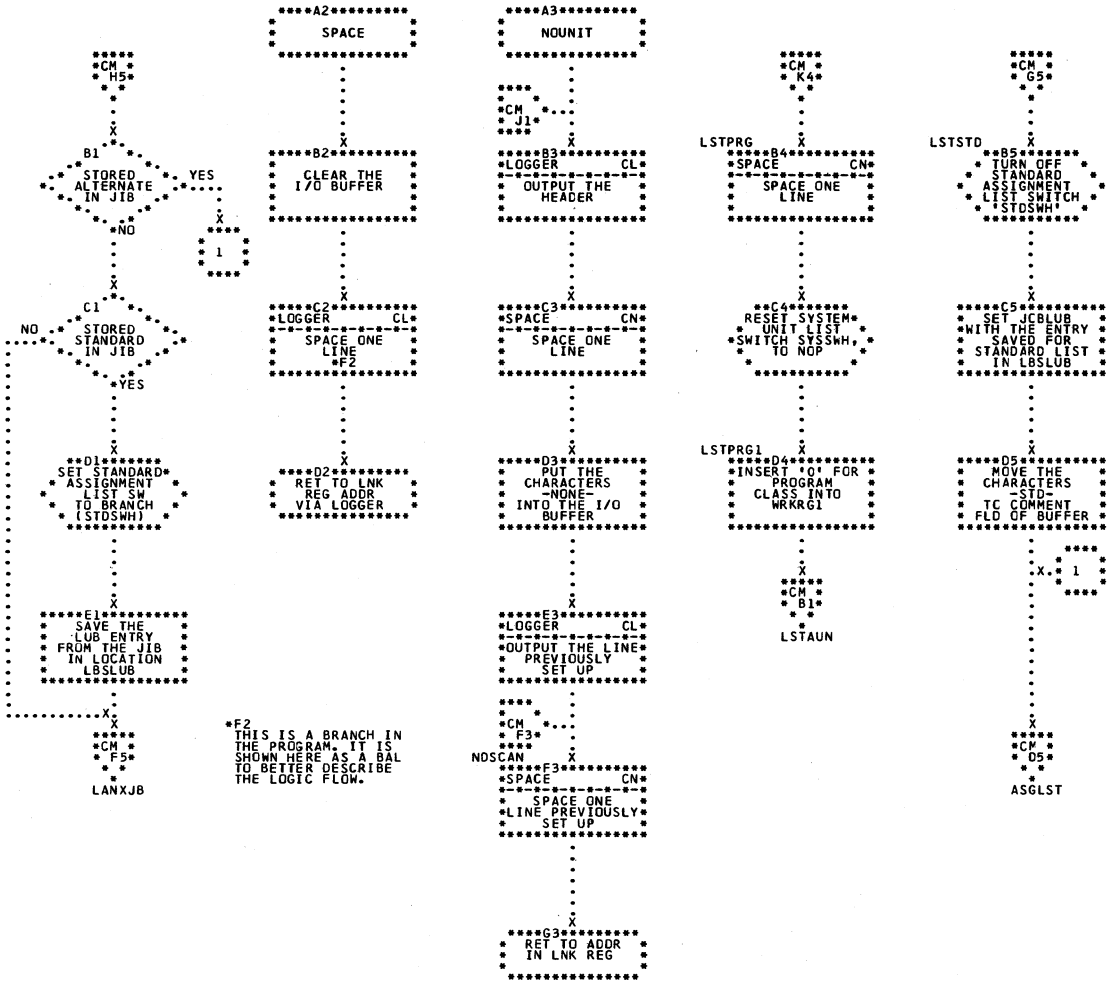


Chart CP. $$BATTNJ - Output List (Part 2 of 2)
Refer to Chart 05.

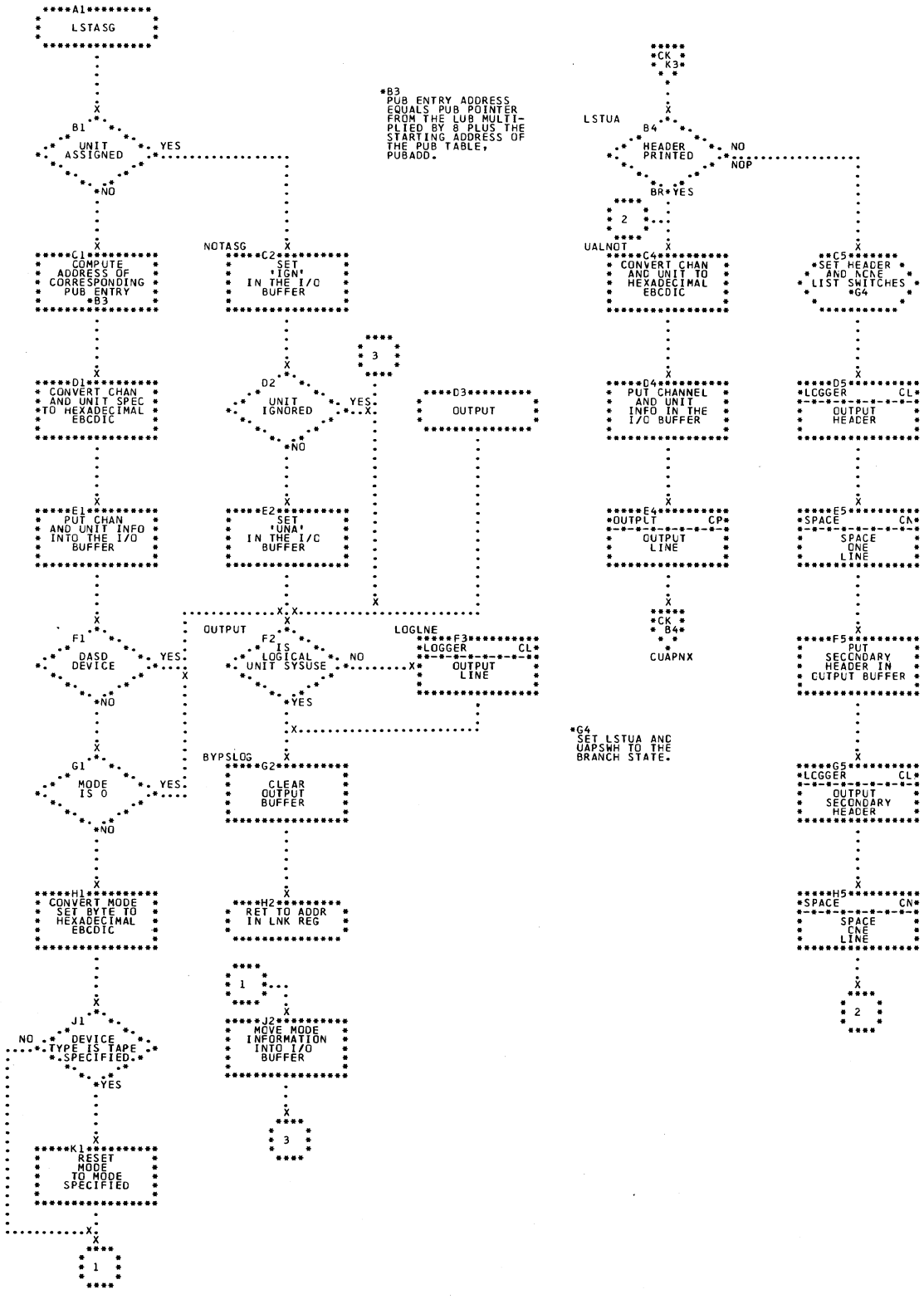


Chart DA. $$BATTNK - VOL Statement Processor
Refer to Chart 05.

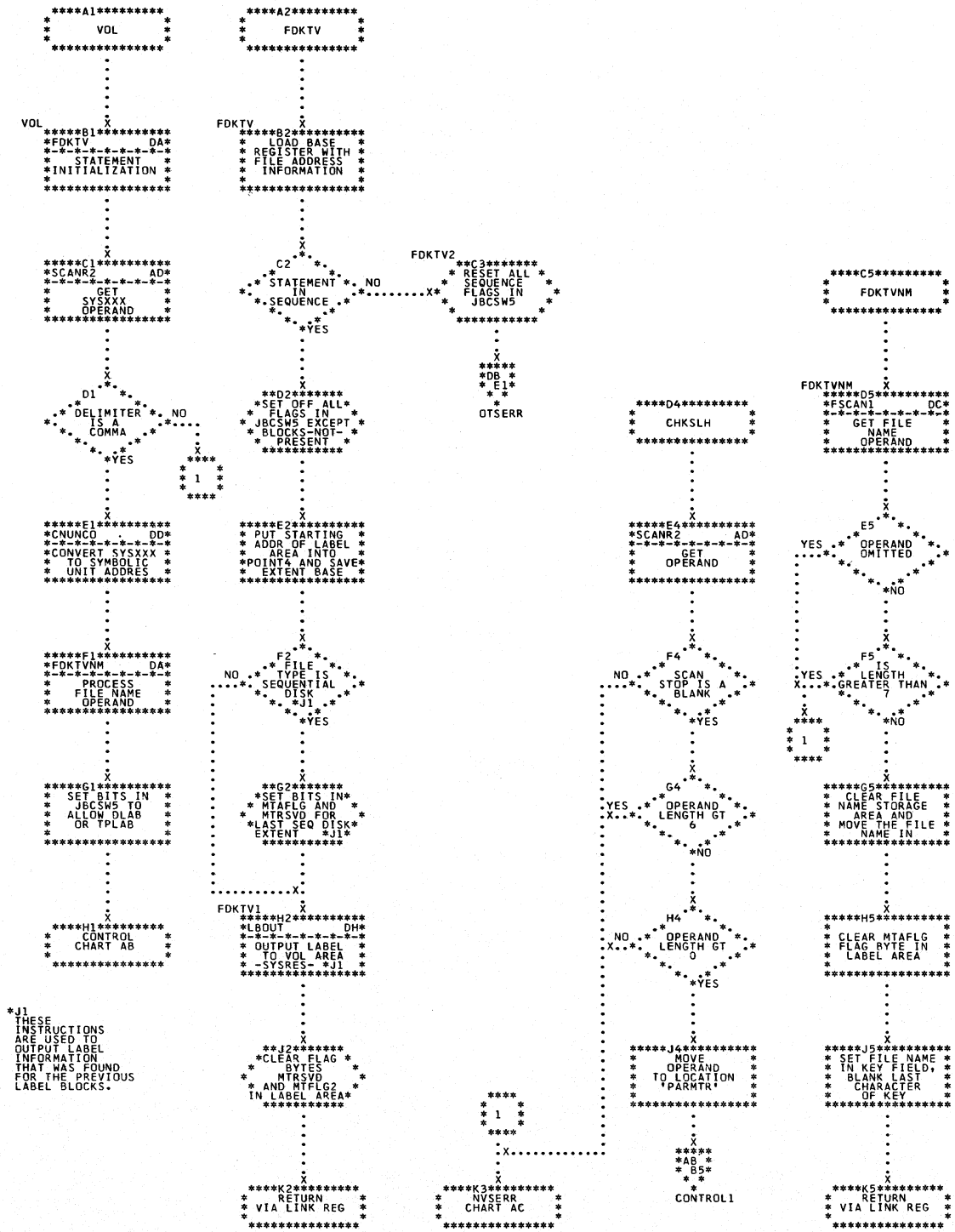


Chart DB. $$BATTNK - TPLAB Statement Processor
Refer to Chart 05.

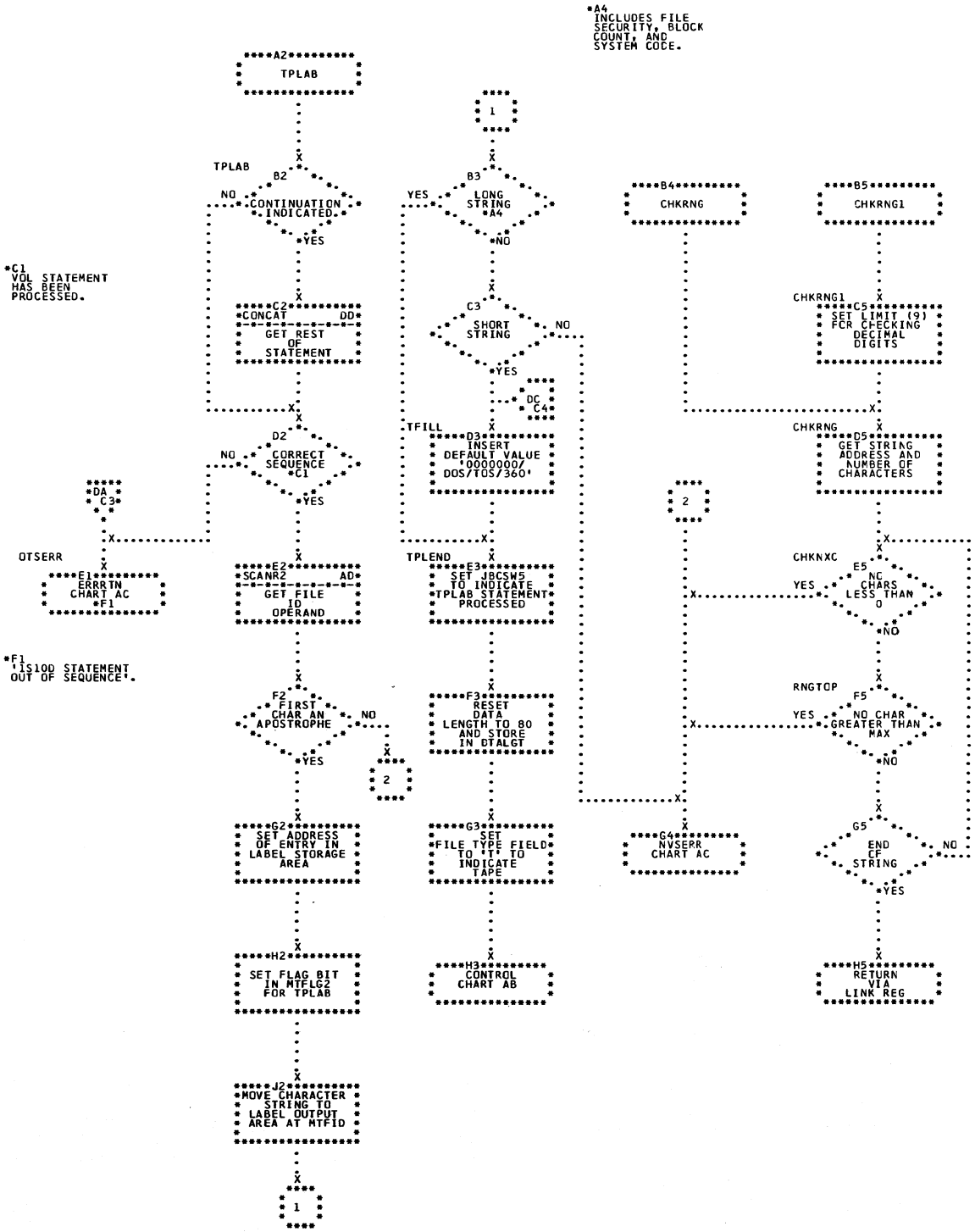


Chart DC. $$BATTNK - TLBL Statement Processor
Refer to Chart 05.

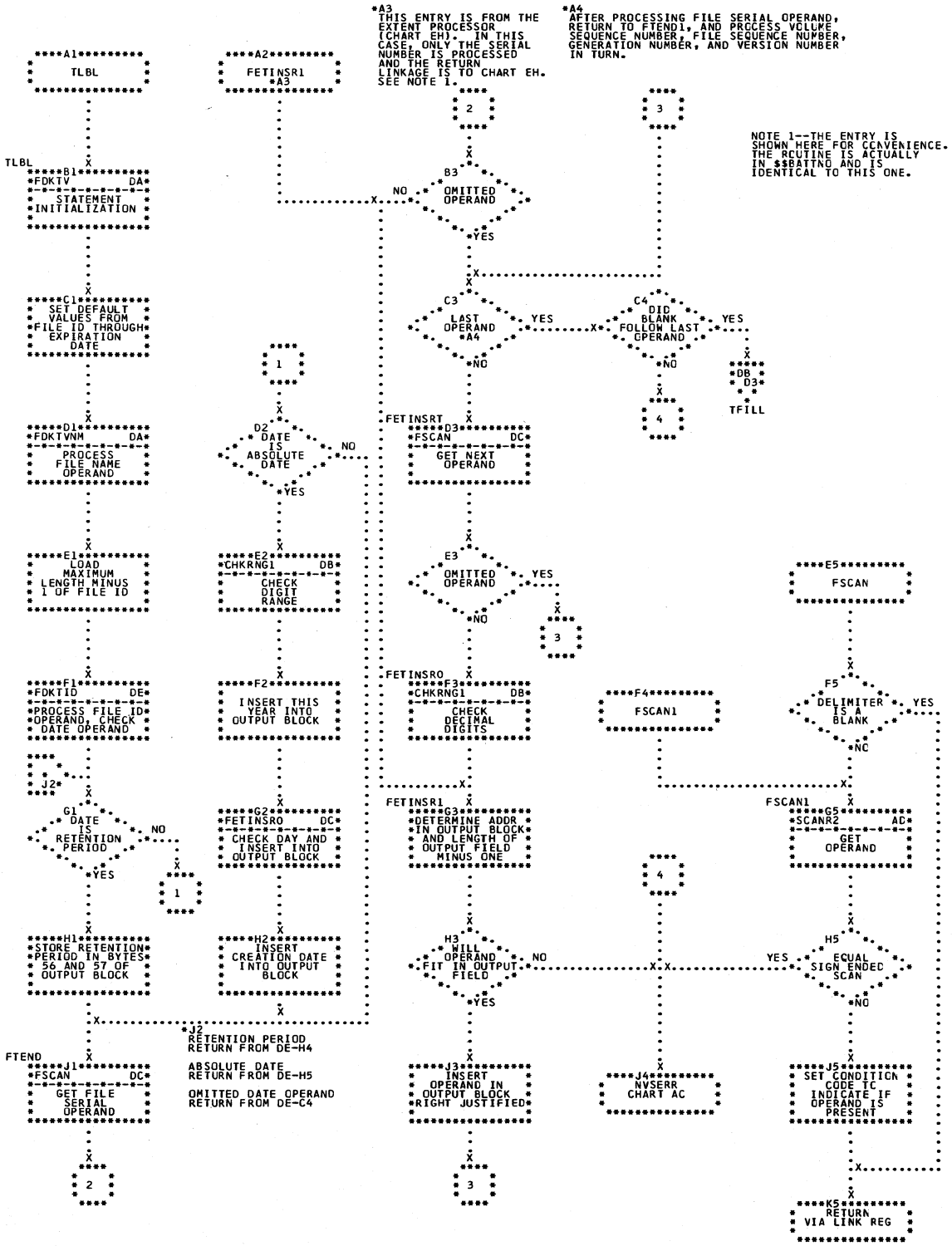


Chart DD. $$BATTNK - Check, Convert, and Concatenate Subroutines
Refer to Chart 05.

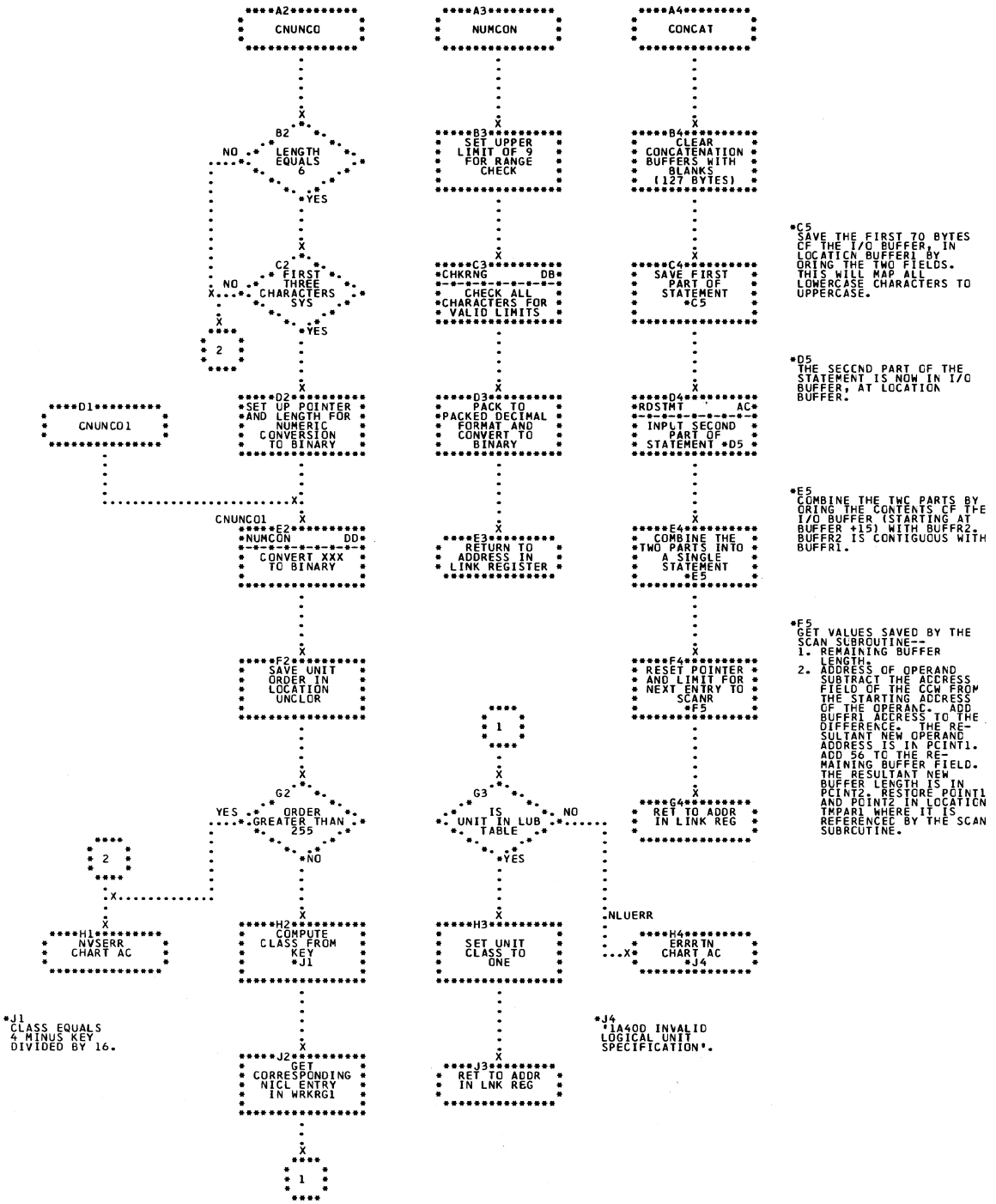


Chart DE. $$BATTNK - Process File ID and Date Operands
Refer to Chart 05.

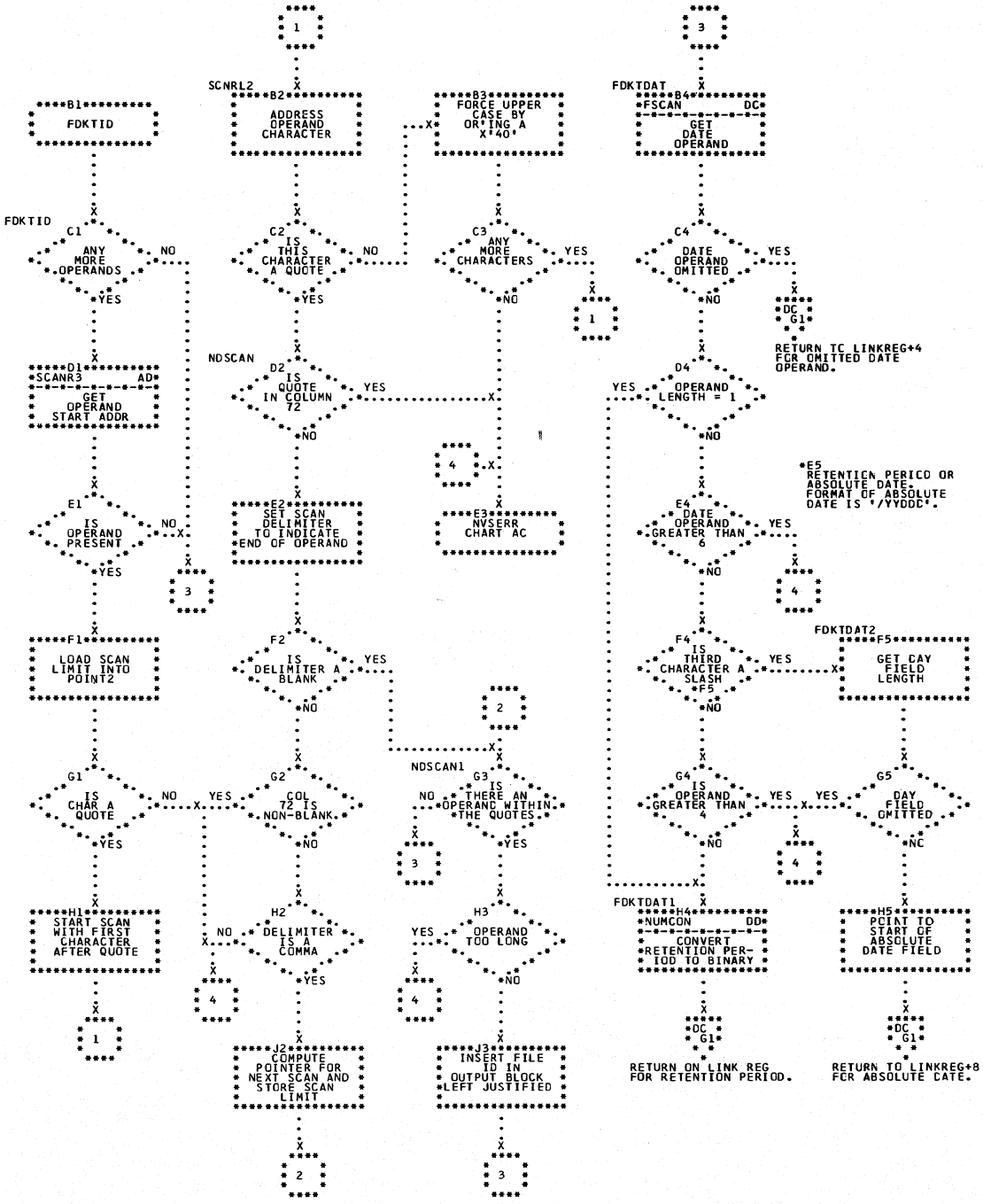


Chart DF. $$BATTNK - DLBL Statement Processor (Part 1 of 2)
Refer to Chart 05.

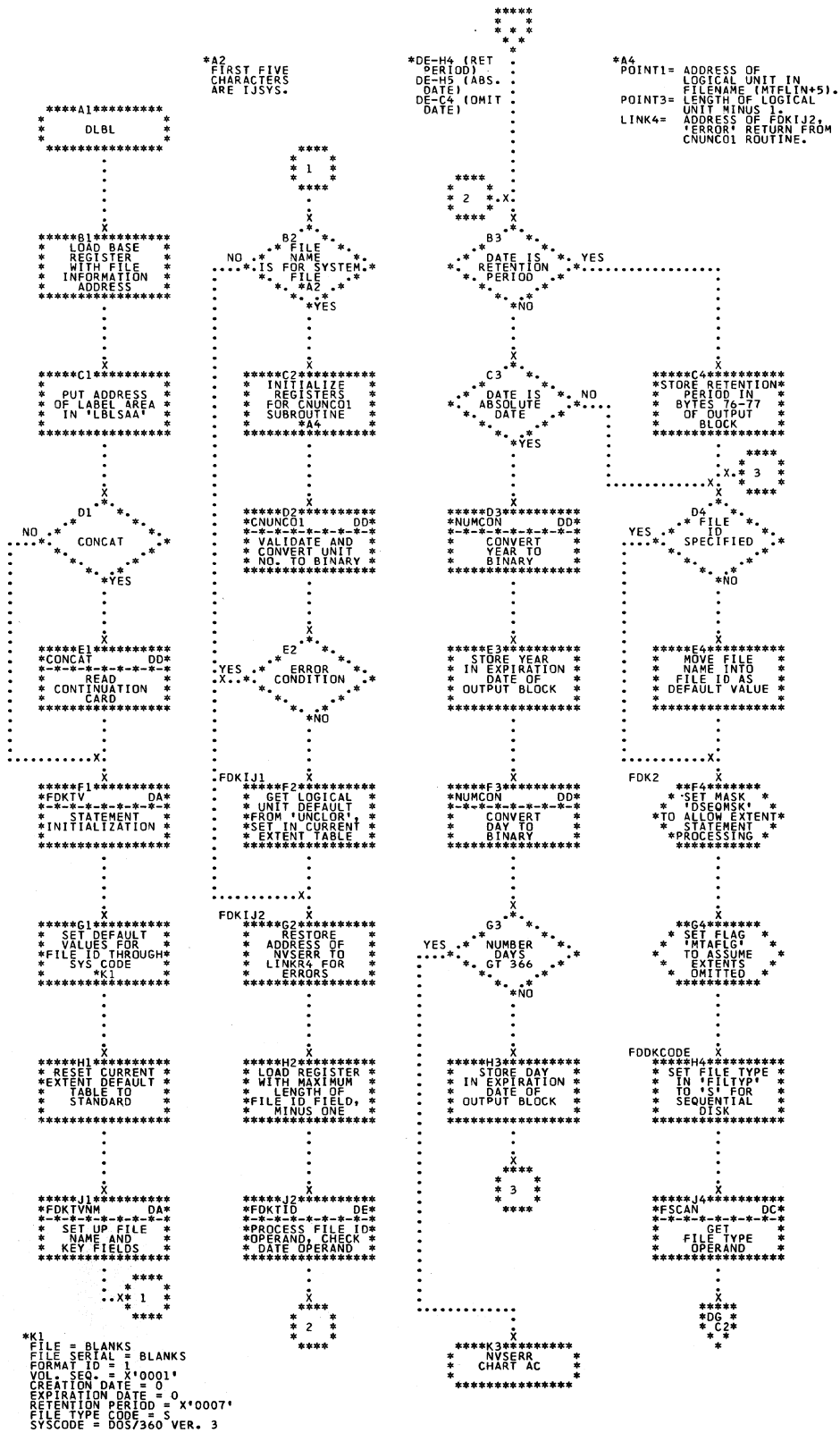


Chart DG. $$BATTNK - DLBL Statement Processor (Part 2 of 2)
Refer to Chart 05.

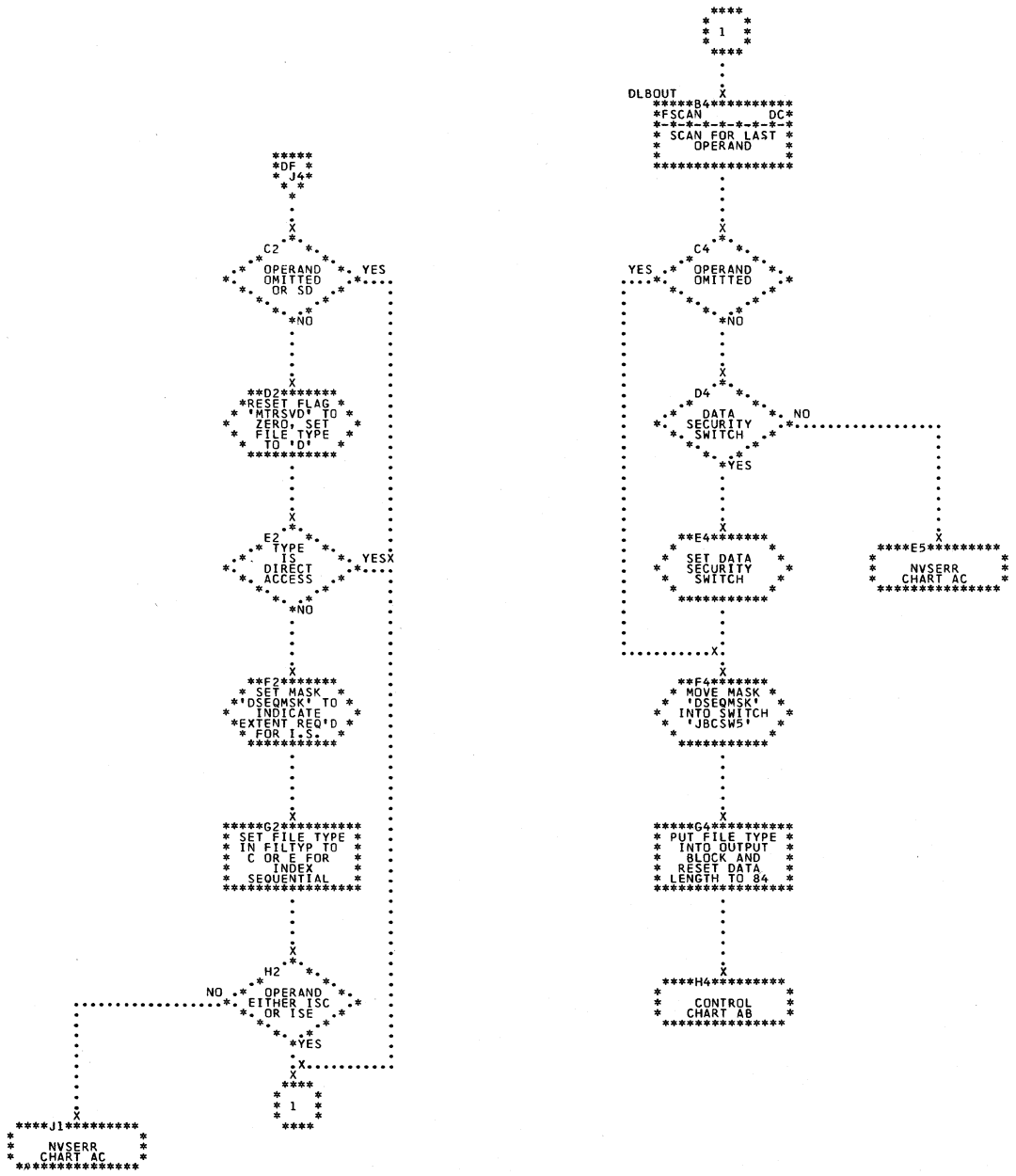


Chart DH. $$BATTNK - Output Label Data Subroutines
Refer to Chart 05.

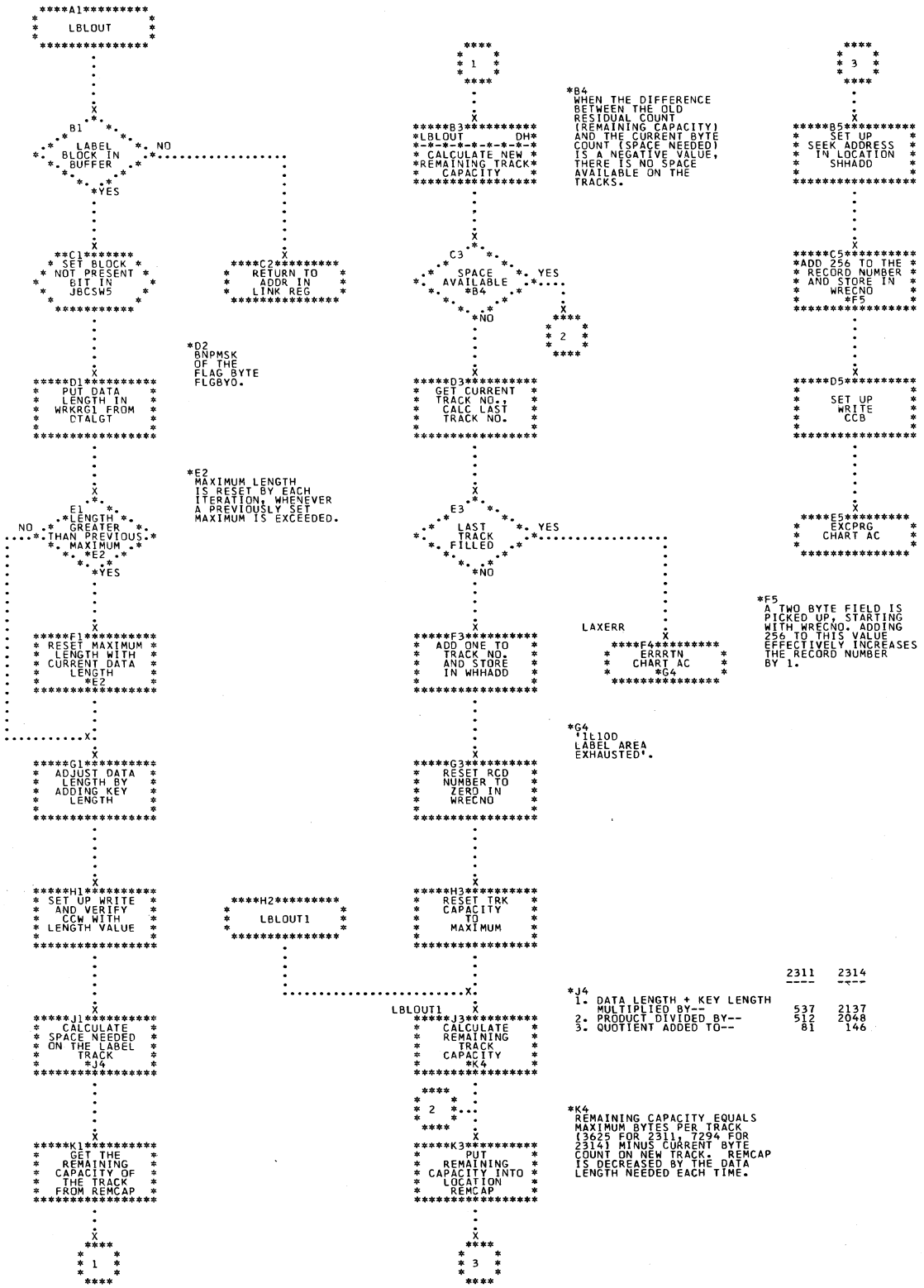


Chart DJ. $$BATTNL - DLAB Statement Processor
Refer to Chart 06.

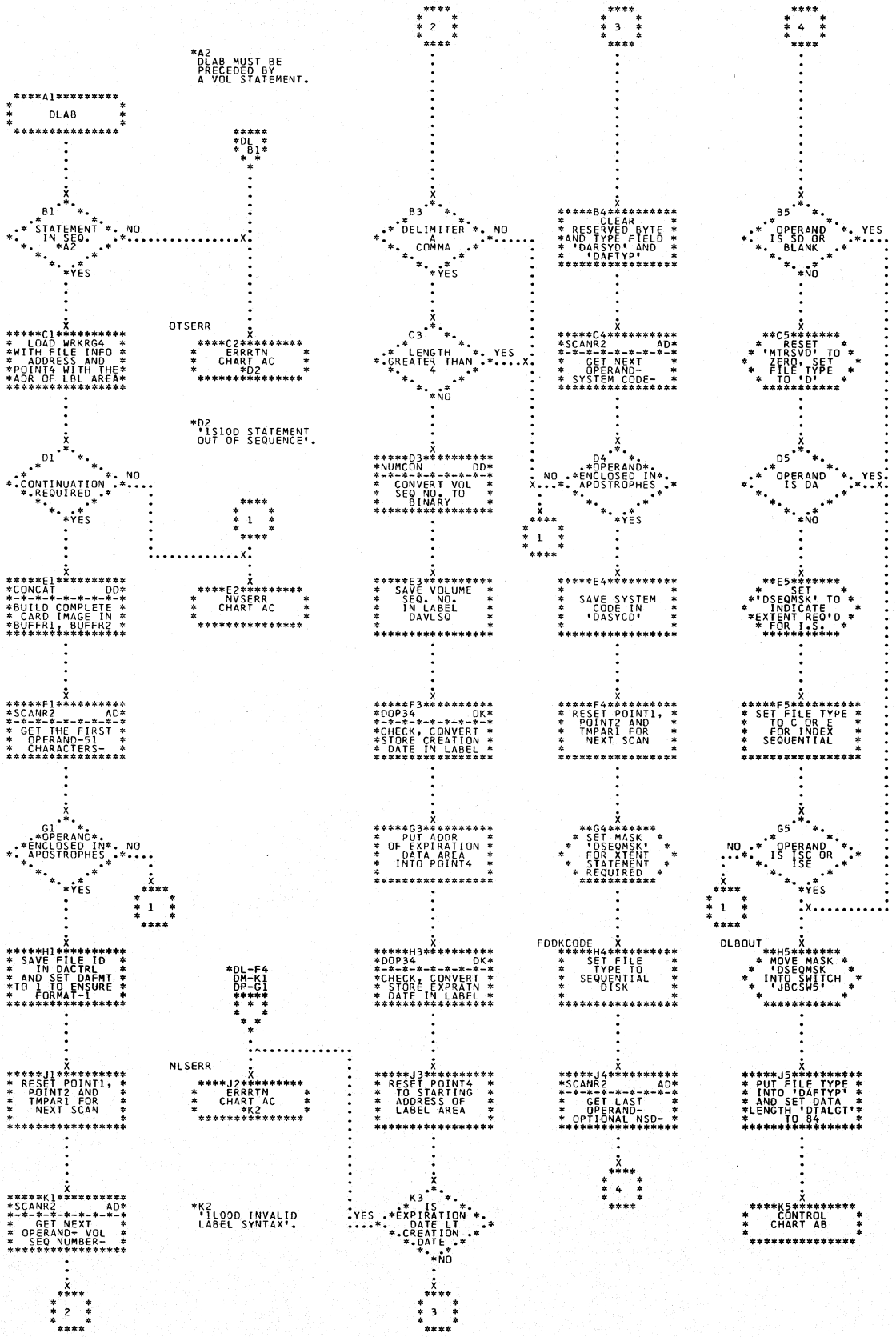


Chart DK. $$BATTNL - Extract Operand Routine
Refer to Chart 06.

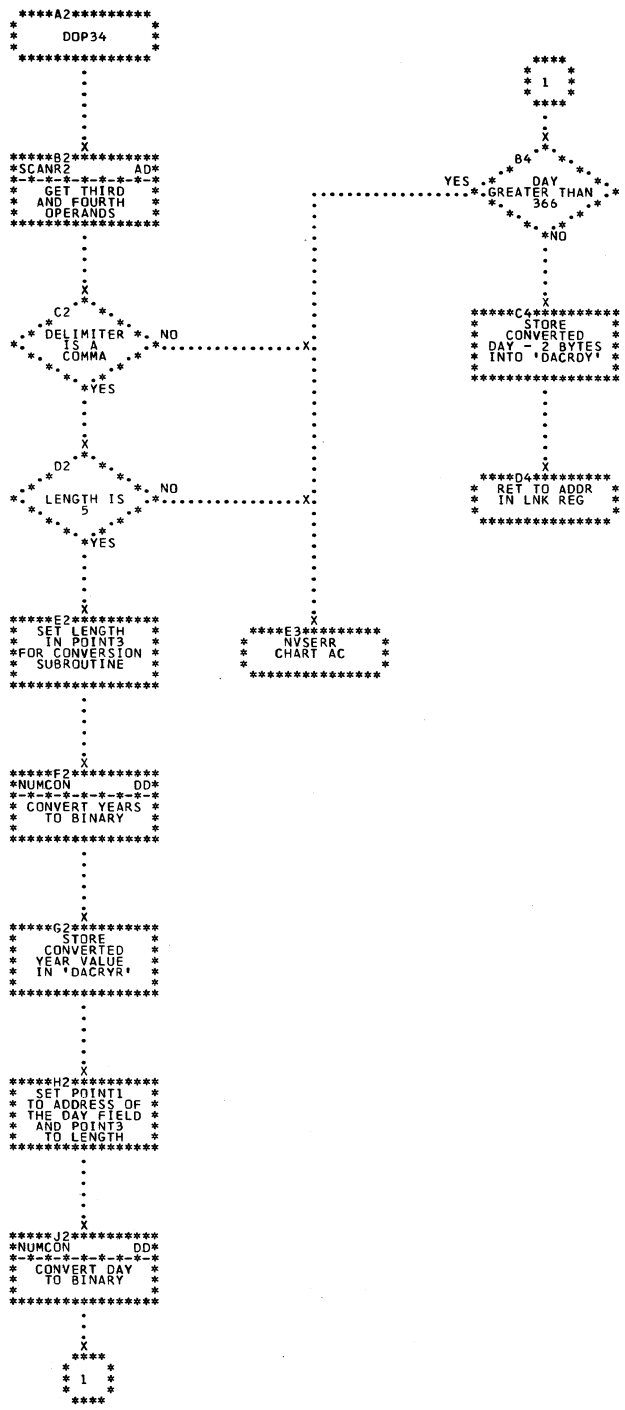


Chart DL. $$BATTNL - XTENT Statement Processor (Part 1 of 3)
Refer to Chart 06.

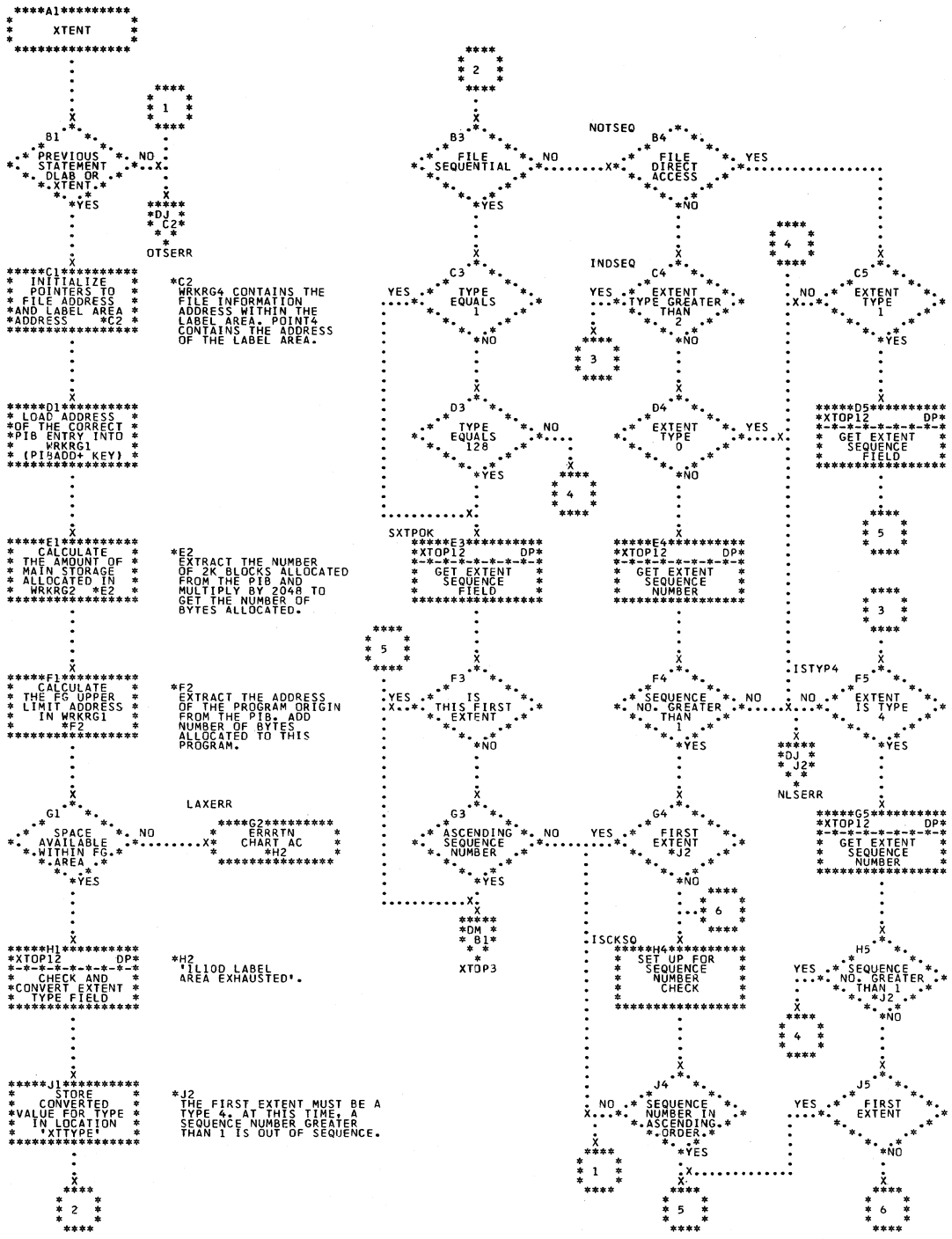


Chart DM. $$BATTNL - XTENT Statement Processor (Part 2 of 3)
Refer to Chart 06.

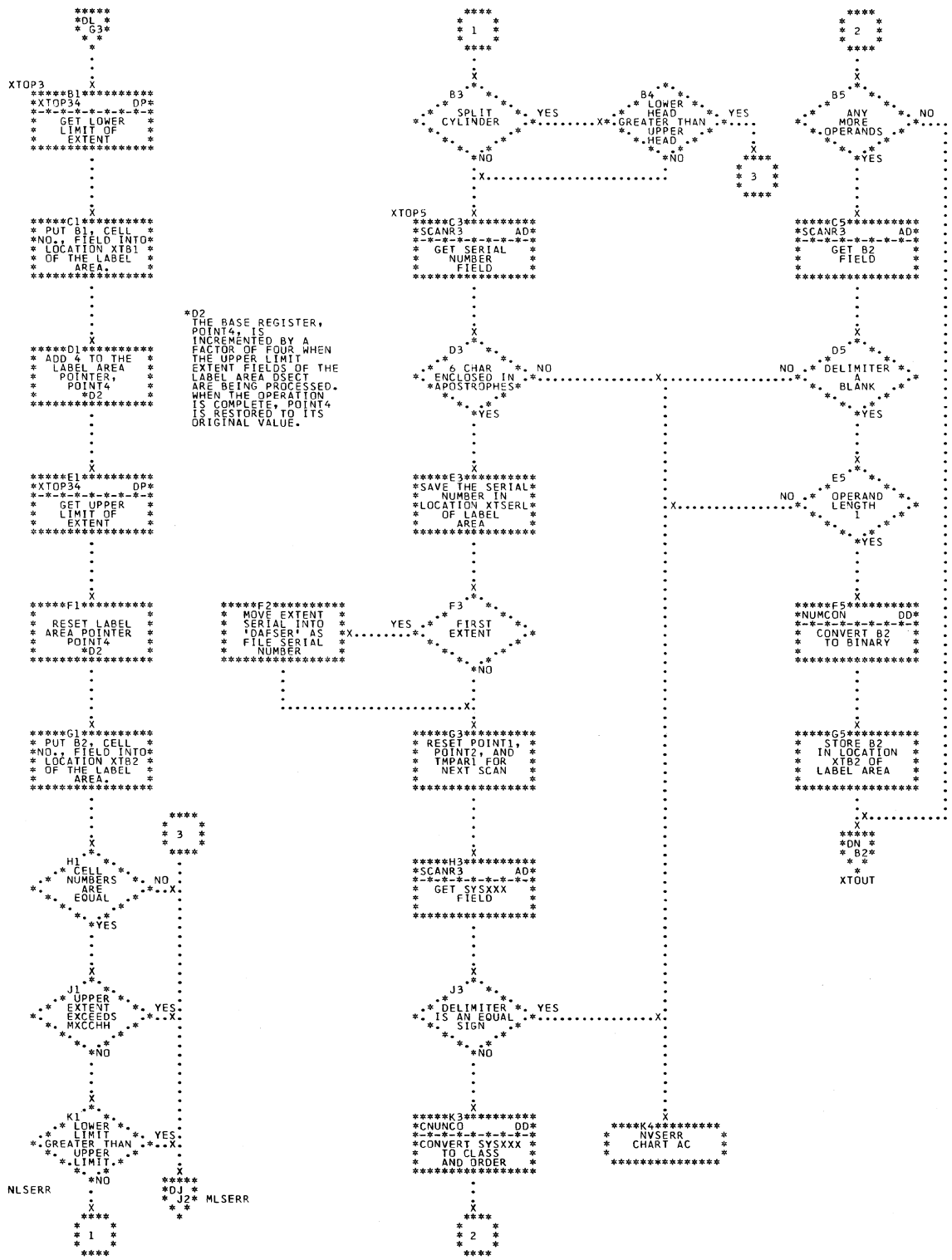


Chart DN. $$BATTNL - XTENT Statement Processor (Part 3 of 3)
Refer to Chart 06.

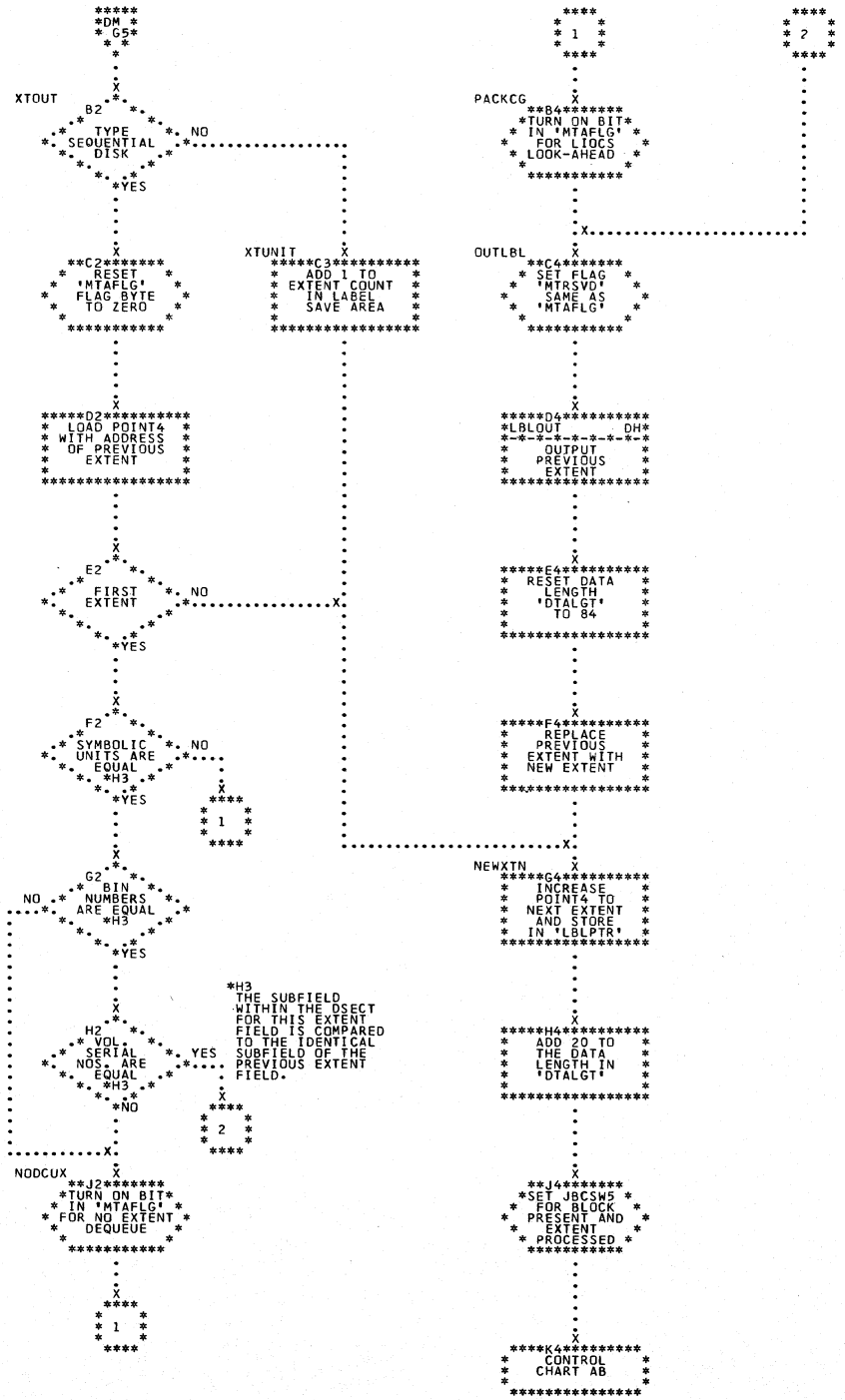


Chart DP. $$BATTNL - XTENT Processor Subroutines
Refer to Chart 06.

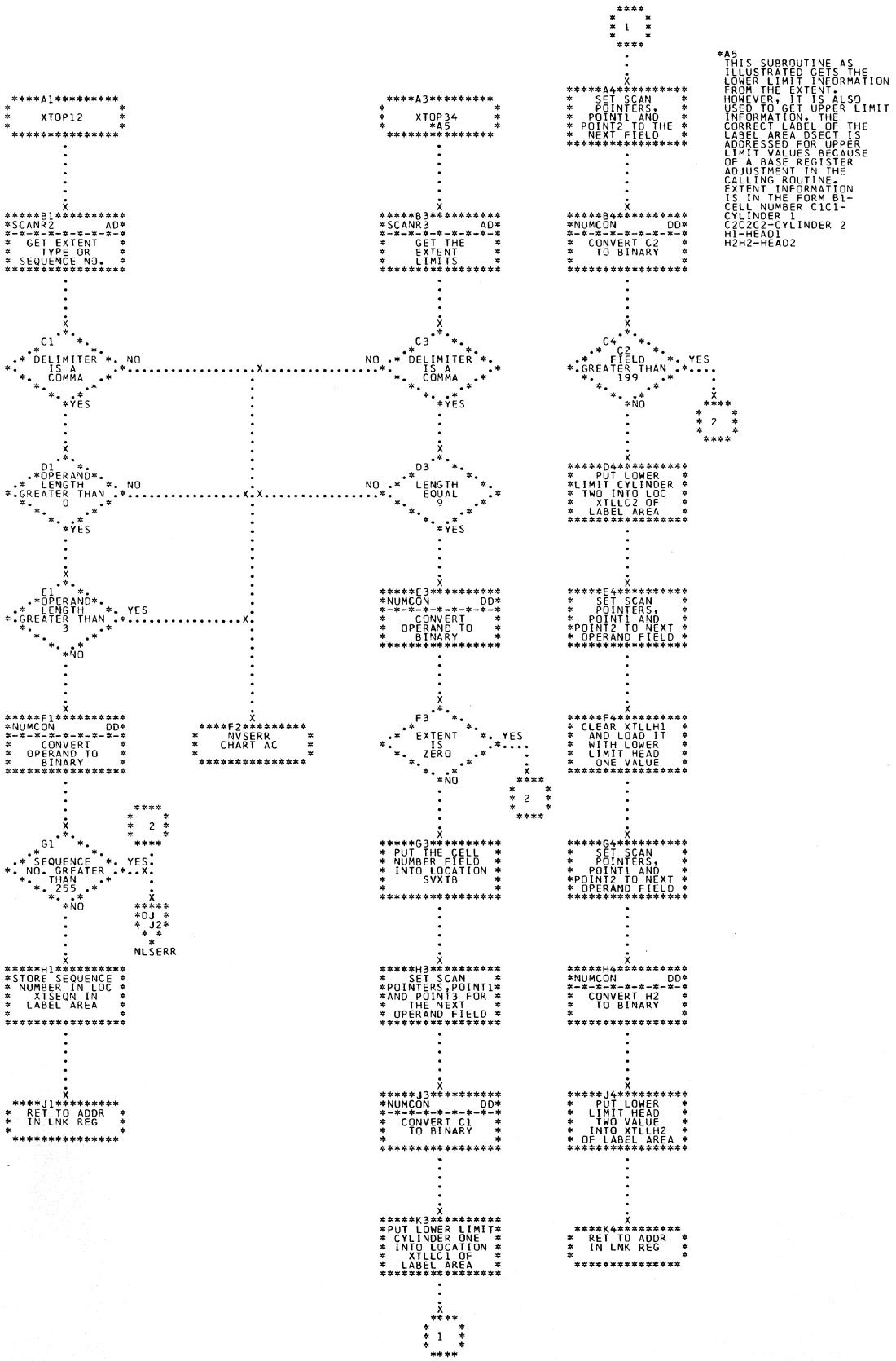


Chart EA. $$BATTNM - EXEC Statement Processor
Refer to Chart 06.

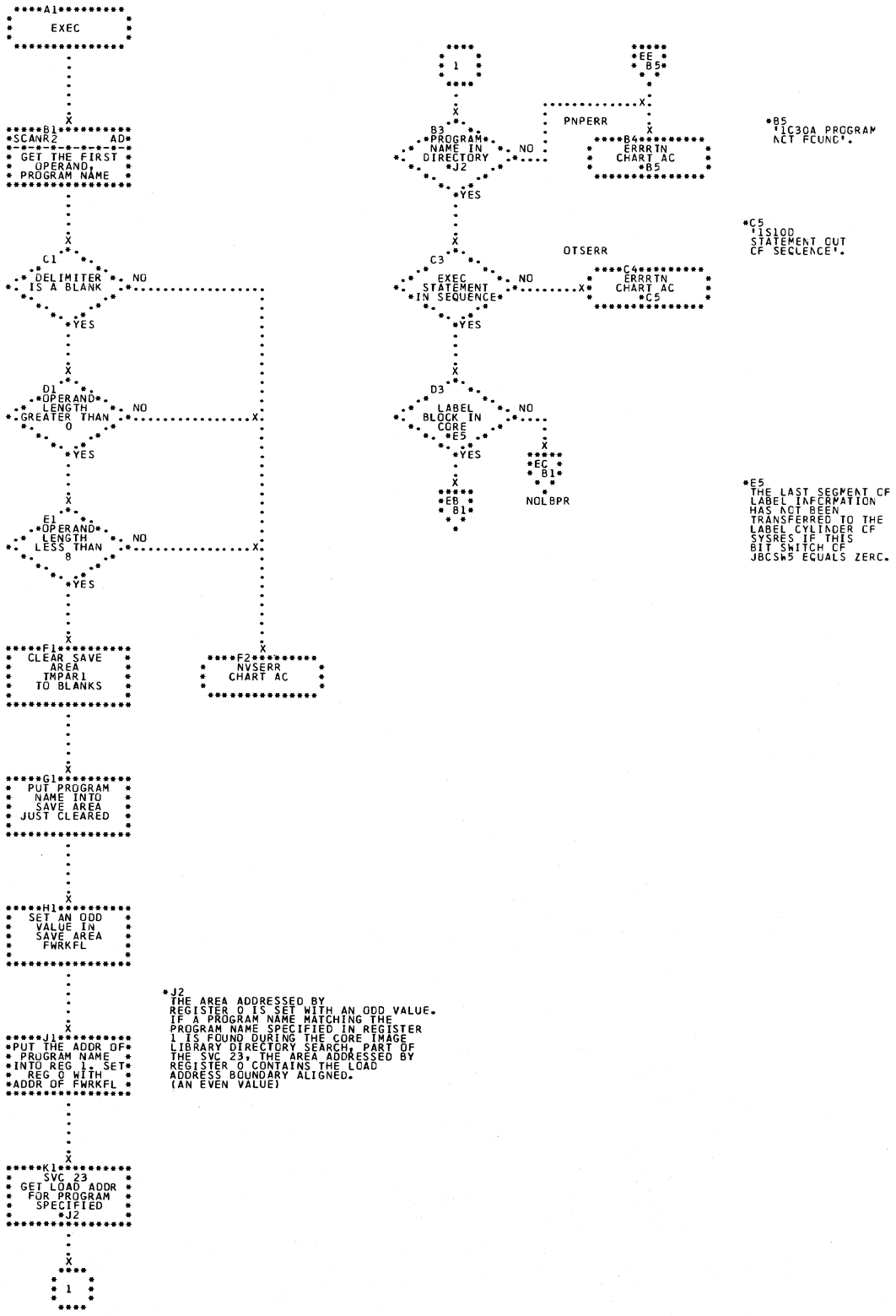


Chart EB. $$BATTNM - Output Last Block of Label Data
Refer to Chart 06.

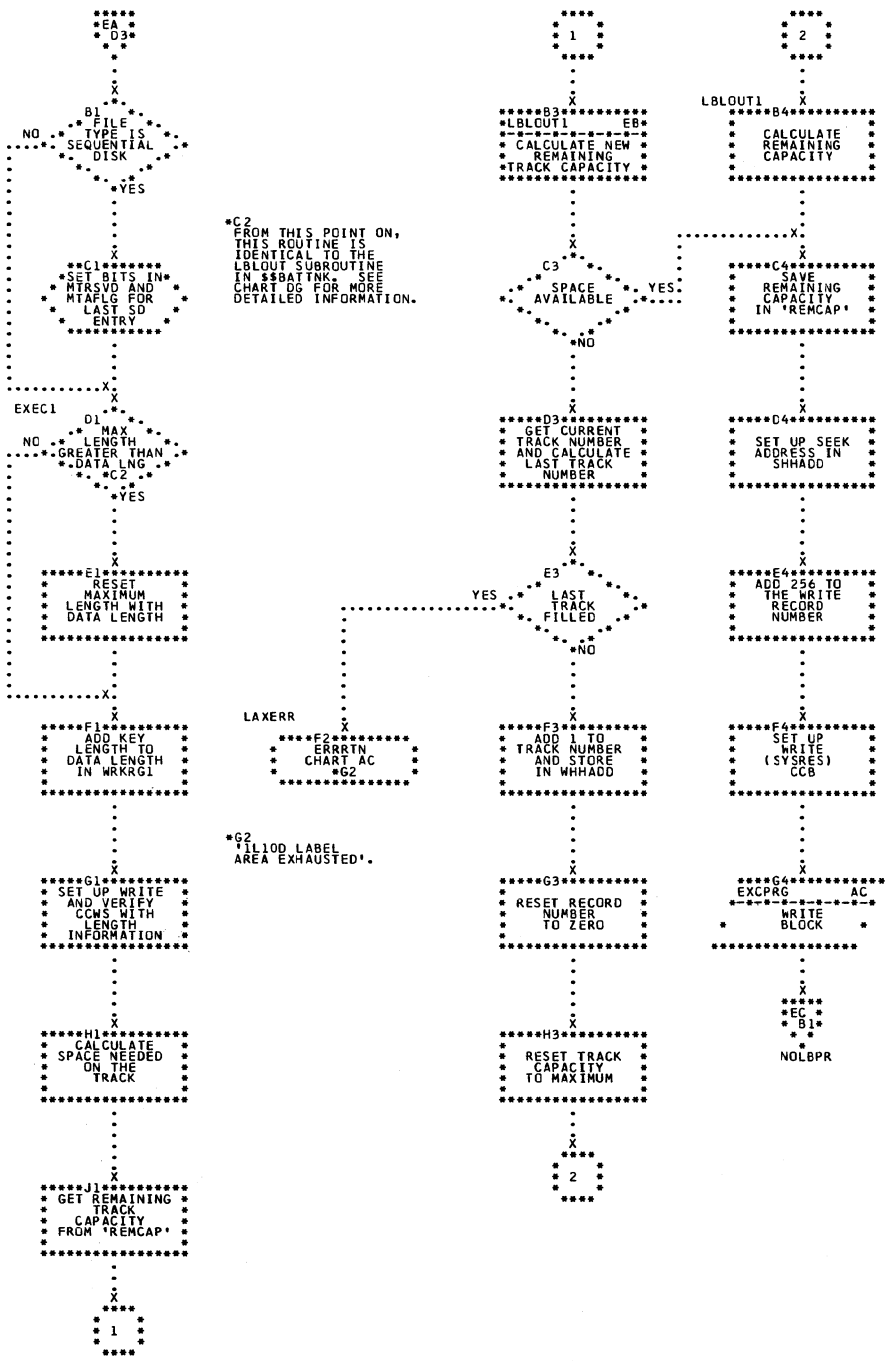


Chart EC. $$BATTNM - Move Last Block Routine
Refer to Chart 06.

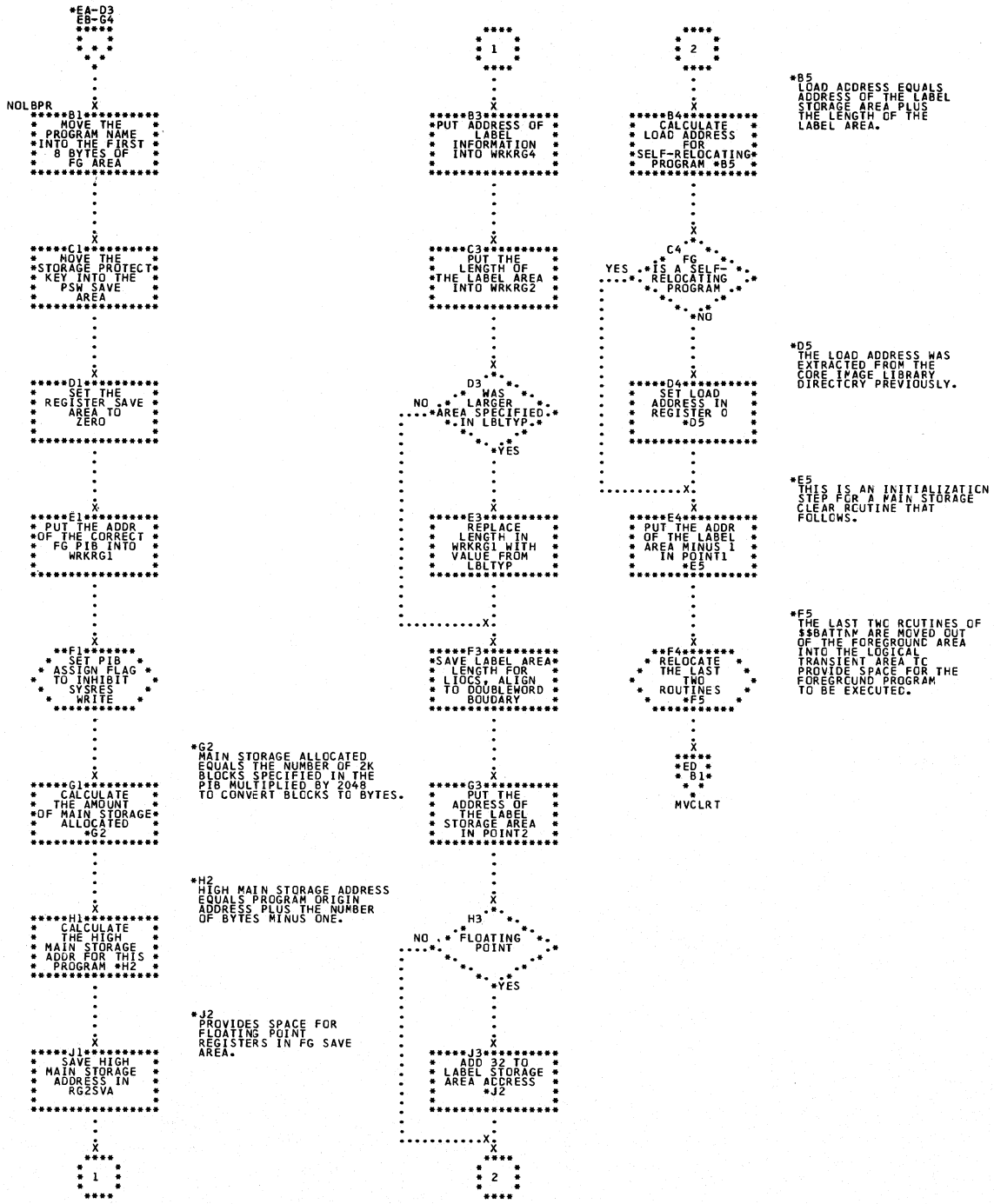


Chart ED. $$BATTNM - Move Subroutine and Initialize for FG Program Load Routine
Refer to Chart 06.

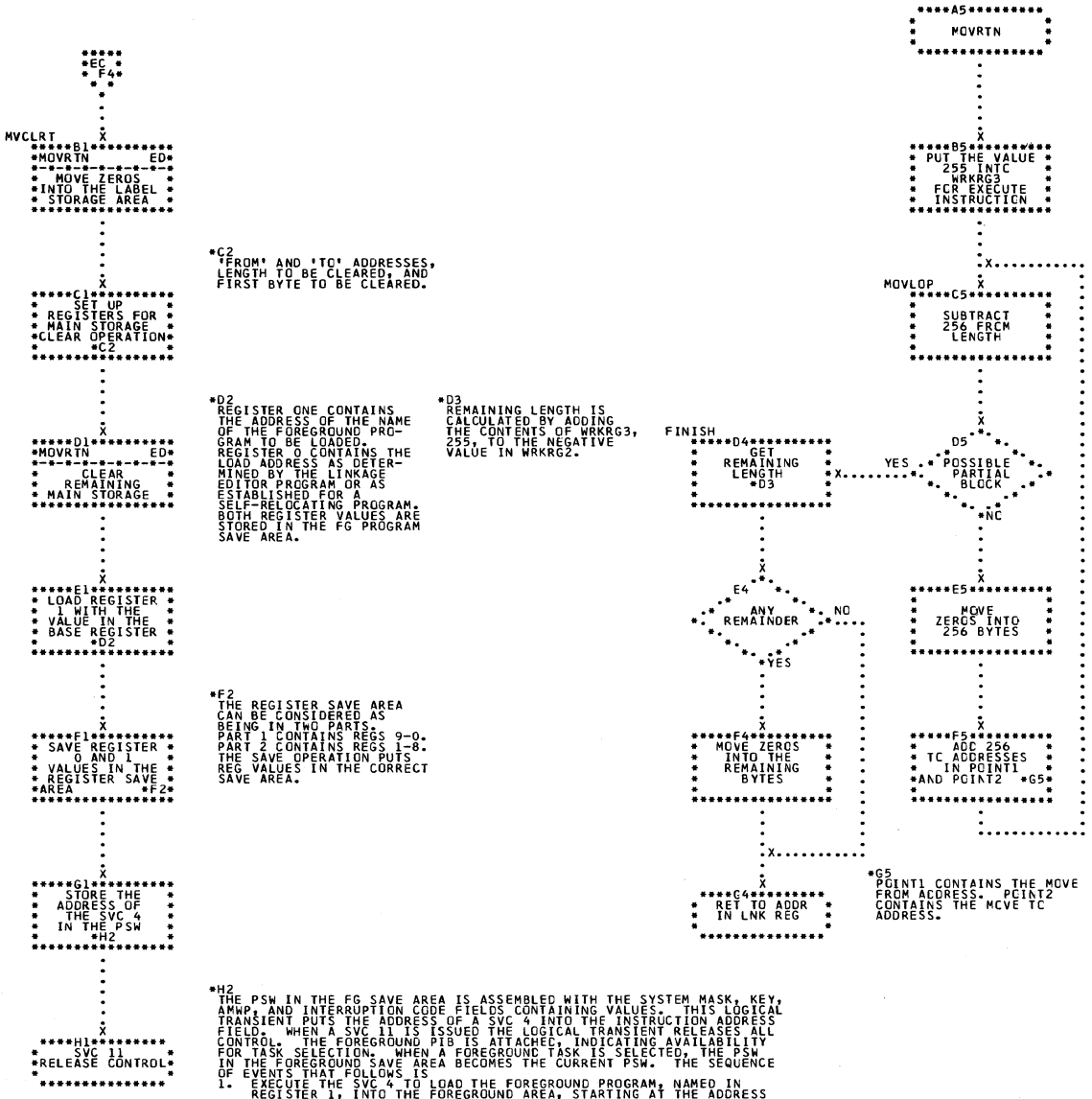


Chart EE. $$BATTNM - UCS Statement Processor
Refer to Chart 06.

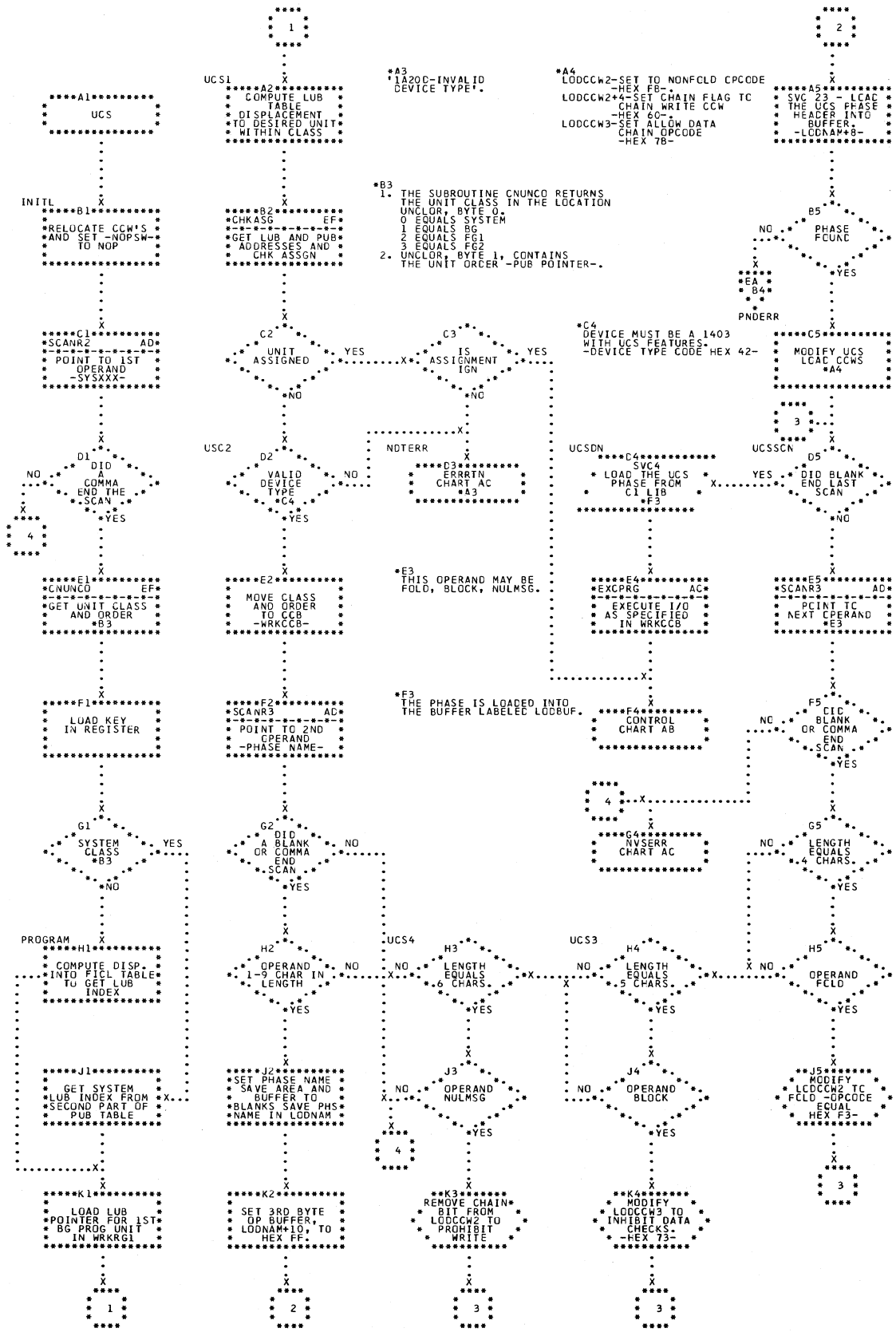


Chart EF. $$BATTNM - UCS Subroutines
Refer to Chart 06.

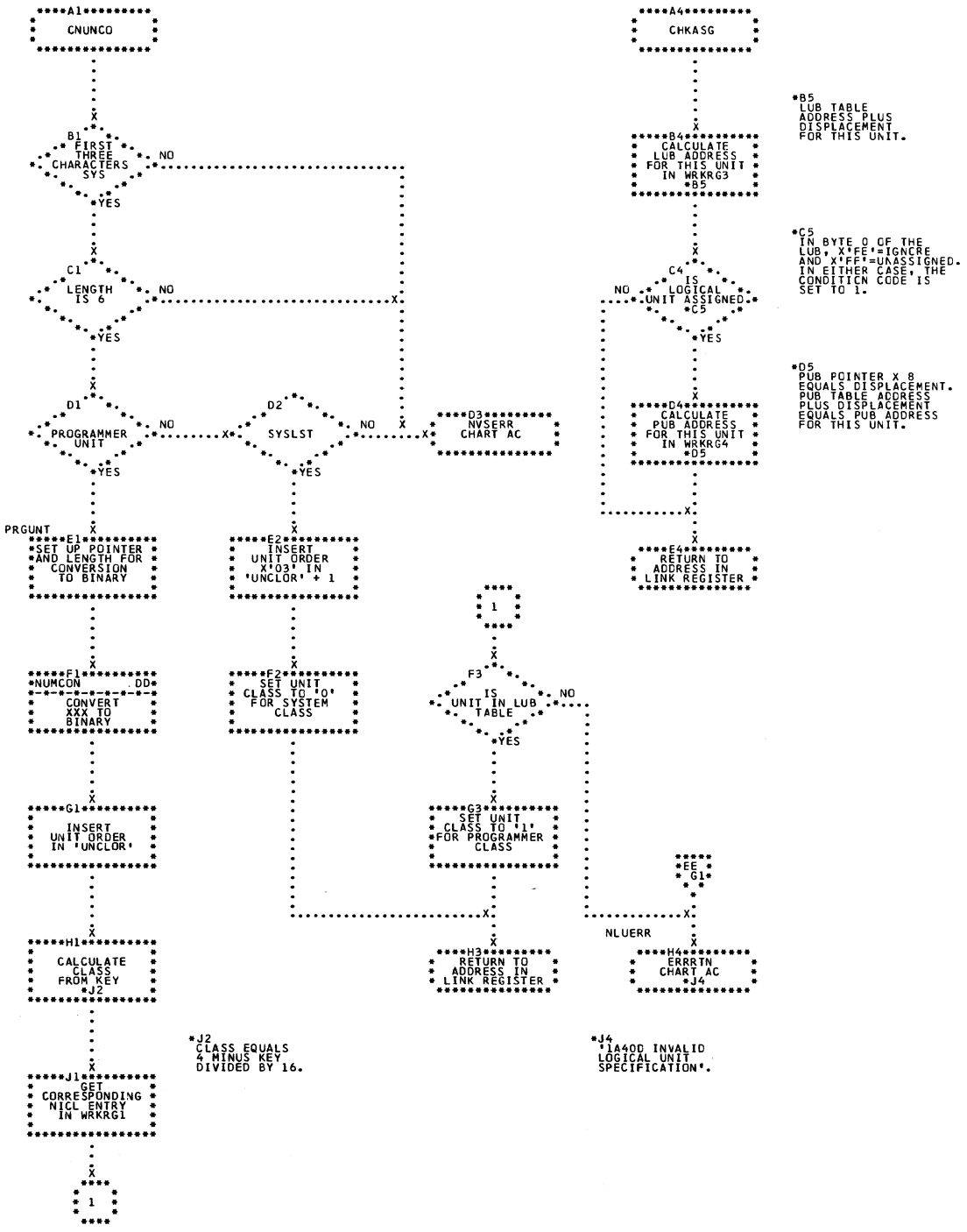


Chart EG. $$BATTNN - TIMER Statement Processor
Refer to Chart 06.

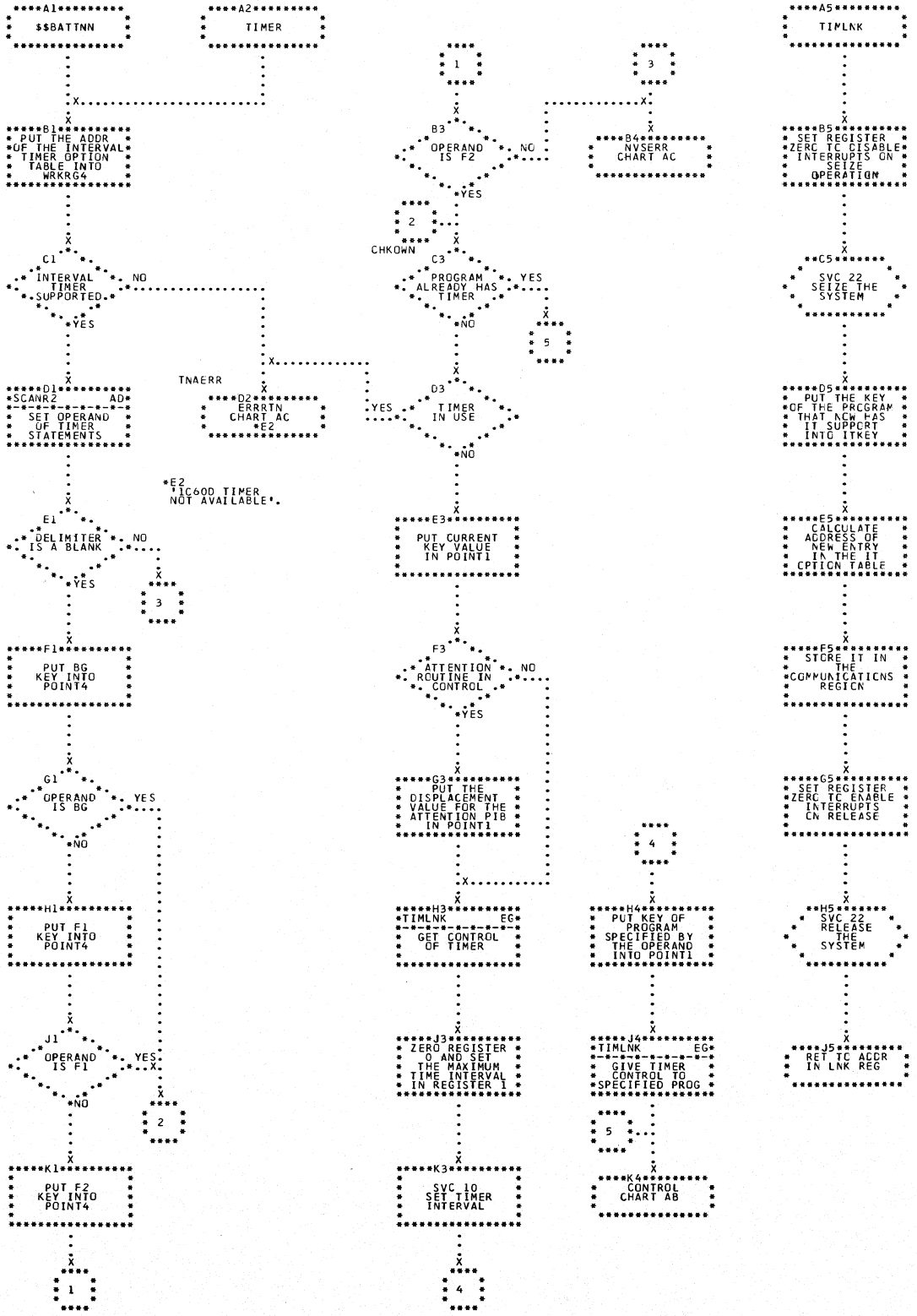
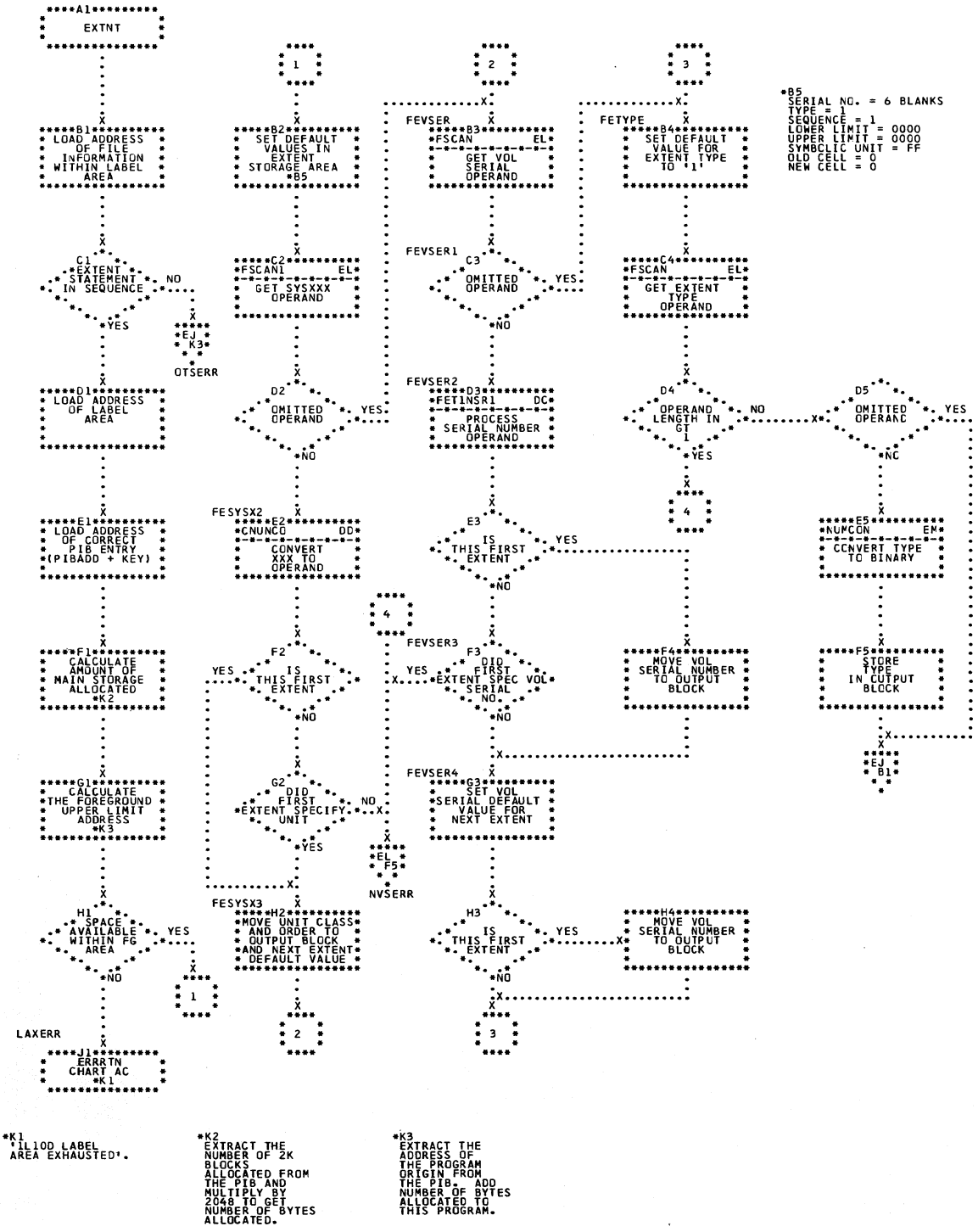


Chart EH. $$BATTNO - EXTENT Statement Processor (Part 1 of 3)
Refer to Chart 06.



*K1
'1L10D LABEL AREA EXHAUSTED'.

*K2
EXTRACT THE NUMBER OF 2K BLOCKS ALLOCATED FROM THE PIB AND MULTIPLY BY 2048 TO GET NUMBER OF BYTES ALLOCATED.

*K3
EXTRACT THE ADDRESS OF THE PROGRAM ORIGIN FROM THE PIB. ADD NUMBER OF BYTES ALLOCATED TO THIS PROGRAM.

Chart EJ. $$BATTNO - EXTENT Statement Processor (Part 2 of 3)
Refer to Chart 06.

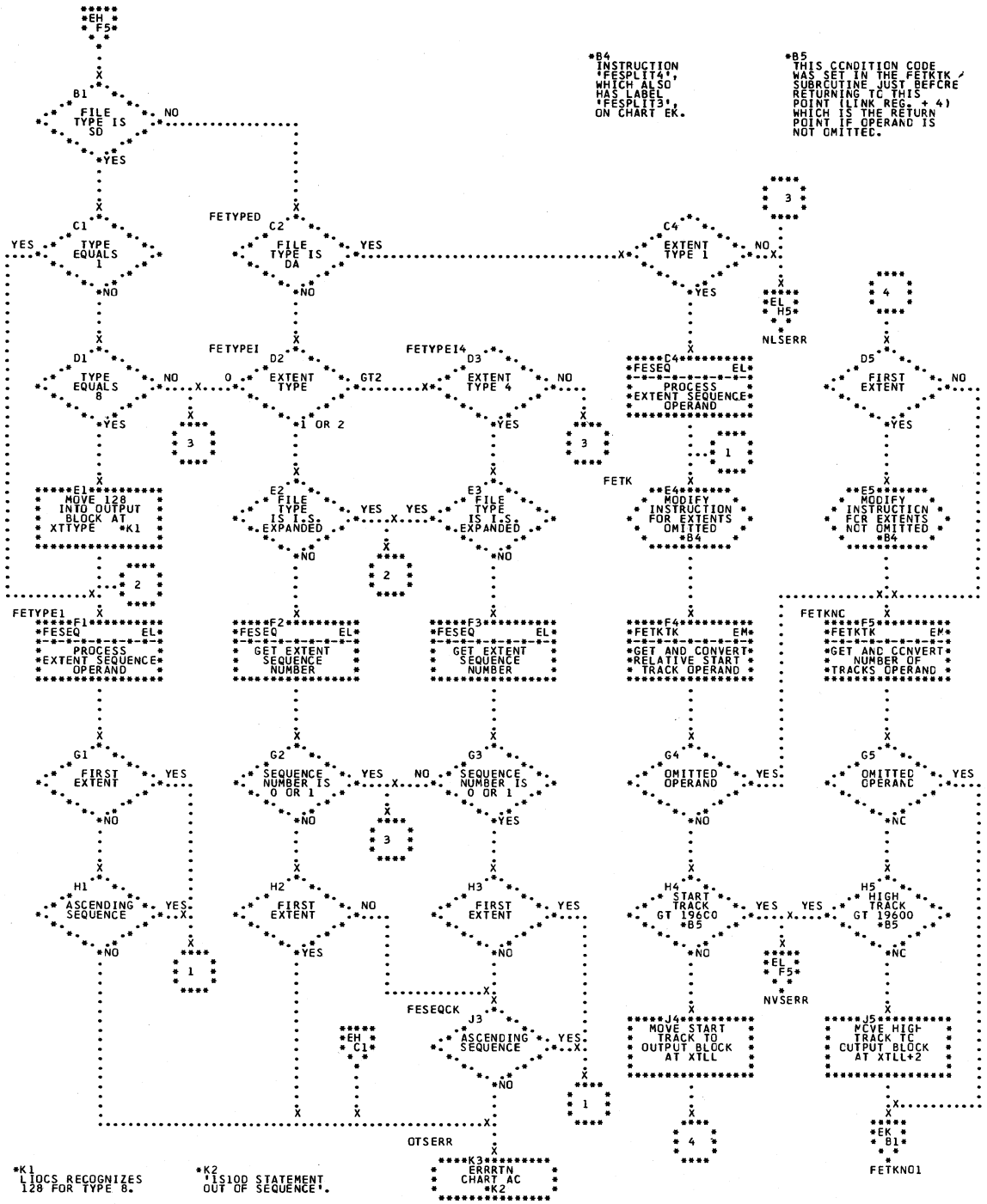


Chart EK. $$BATTNO - EXTENT Statement Processor (Part 3 of 3)
Refer to Chart 06.

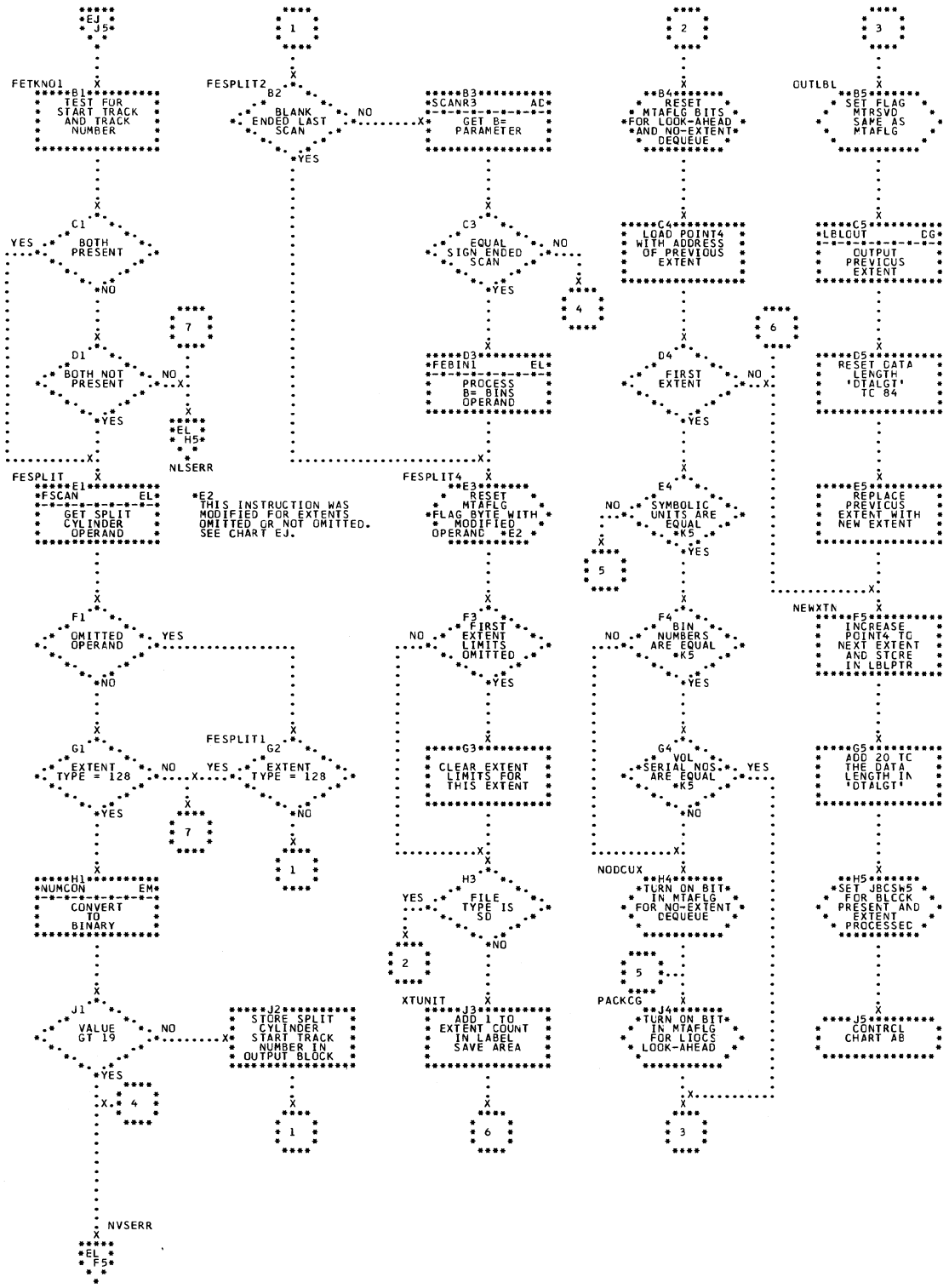


Chart EL. $$BATTNO - EXTENT Processor Subroutines
Refer to Chart 06.

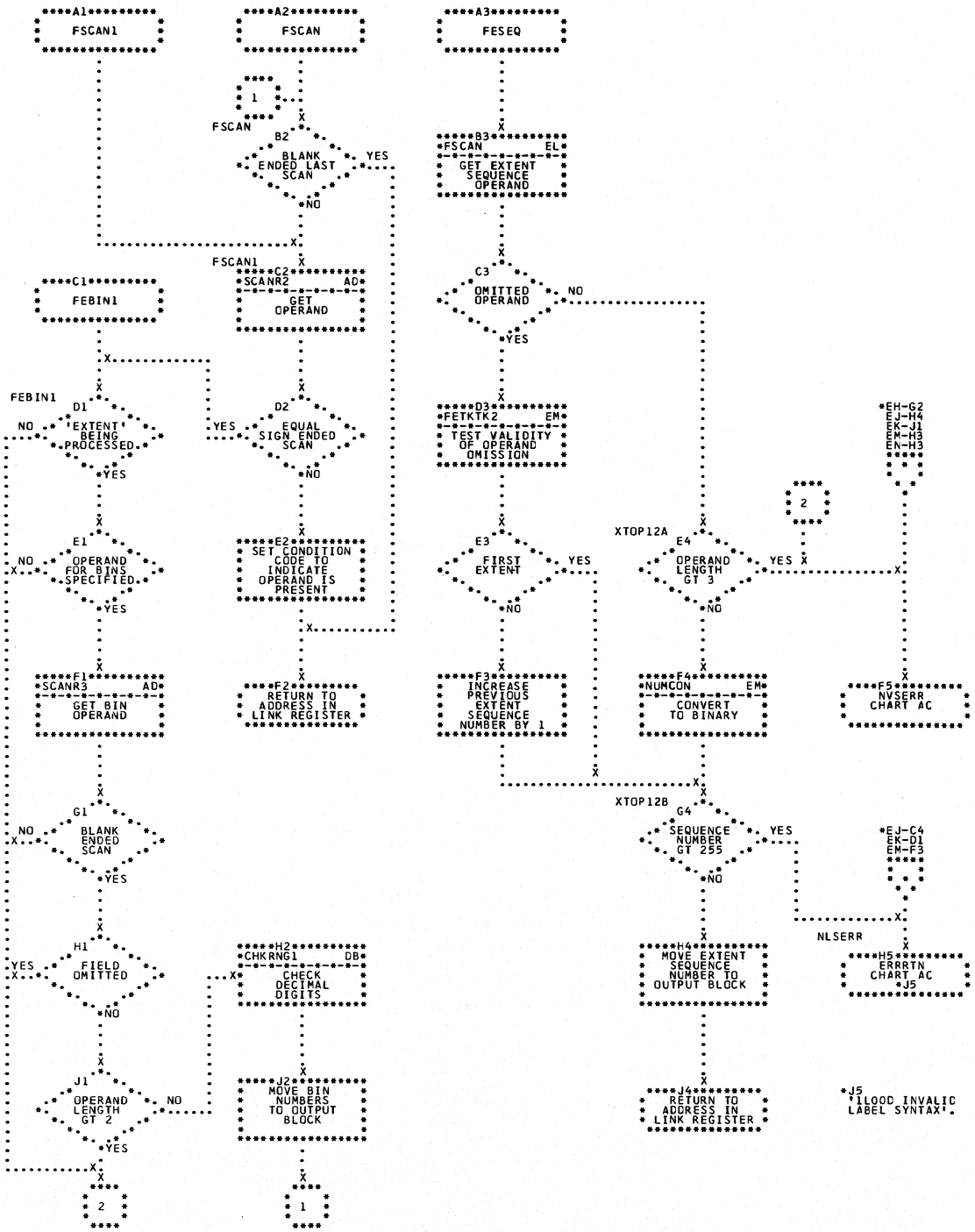


Chart EM. $$BATTNO - Process Track Operands
Refer to Chart 06.

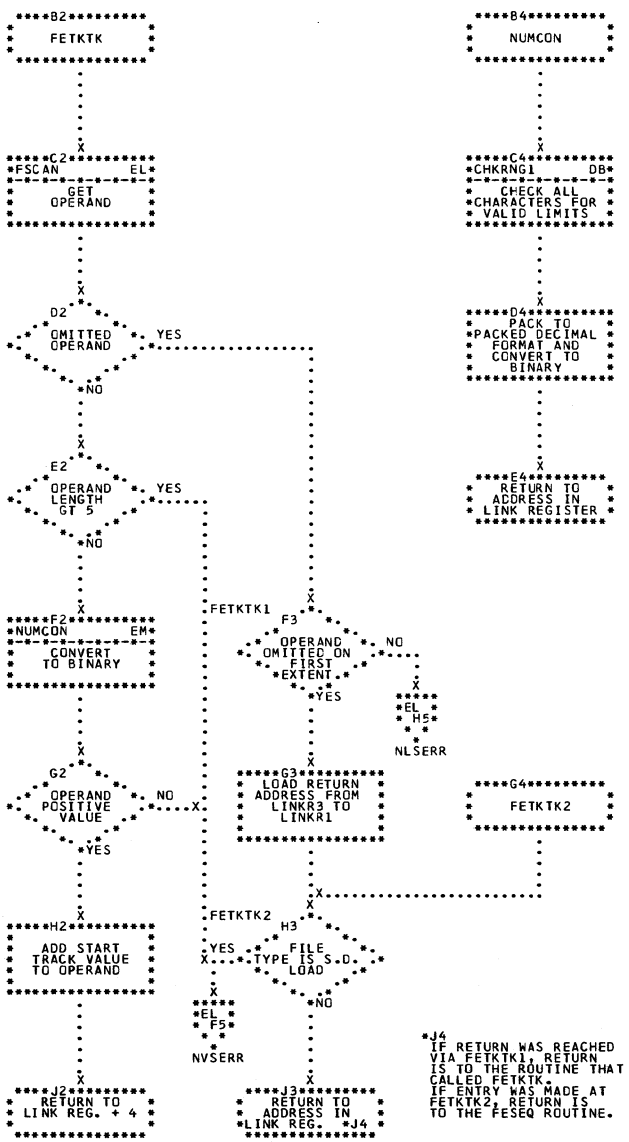


Chart EN. $$BATTNO - LBLTYP Statement Processor
Refer to Chart 06.

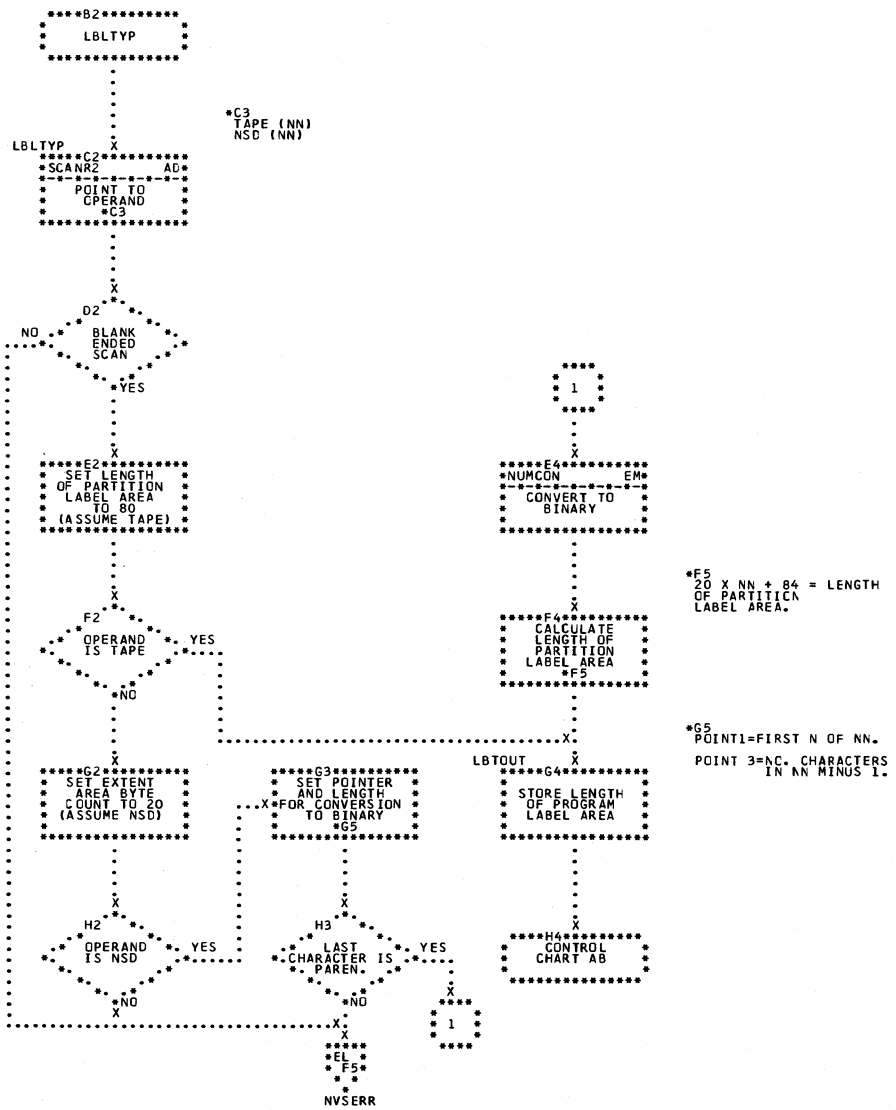


Chart EP. $$BATTNP - READ Statement Processor
Refer to Chart 06.

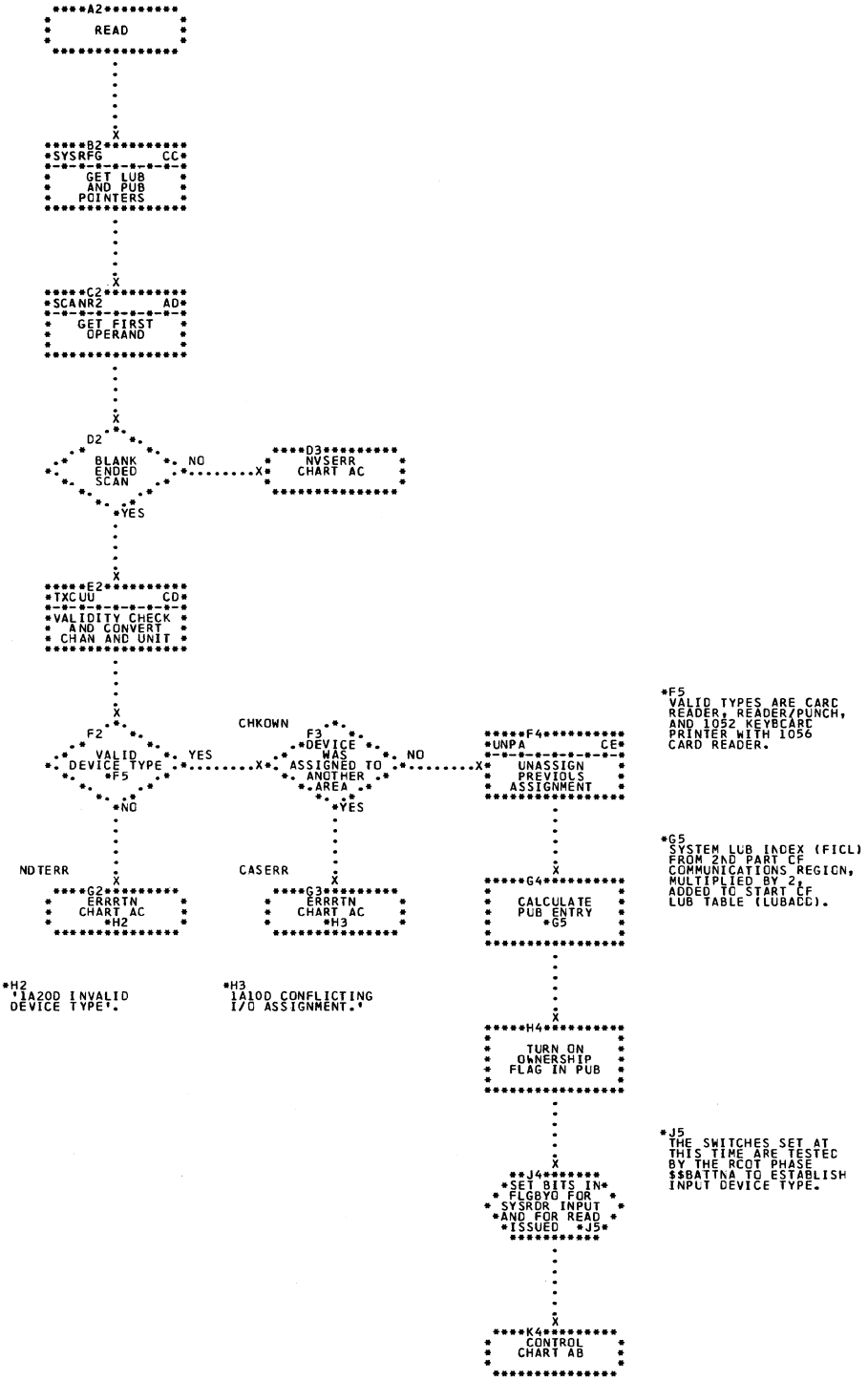


Chart EQ. $$BATTNP - HOLD or RELSE Statement Processor
Refer to Chart 06.

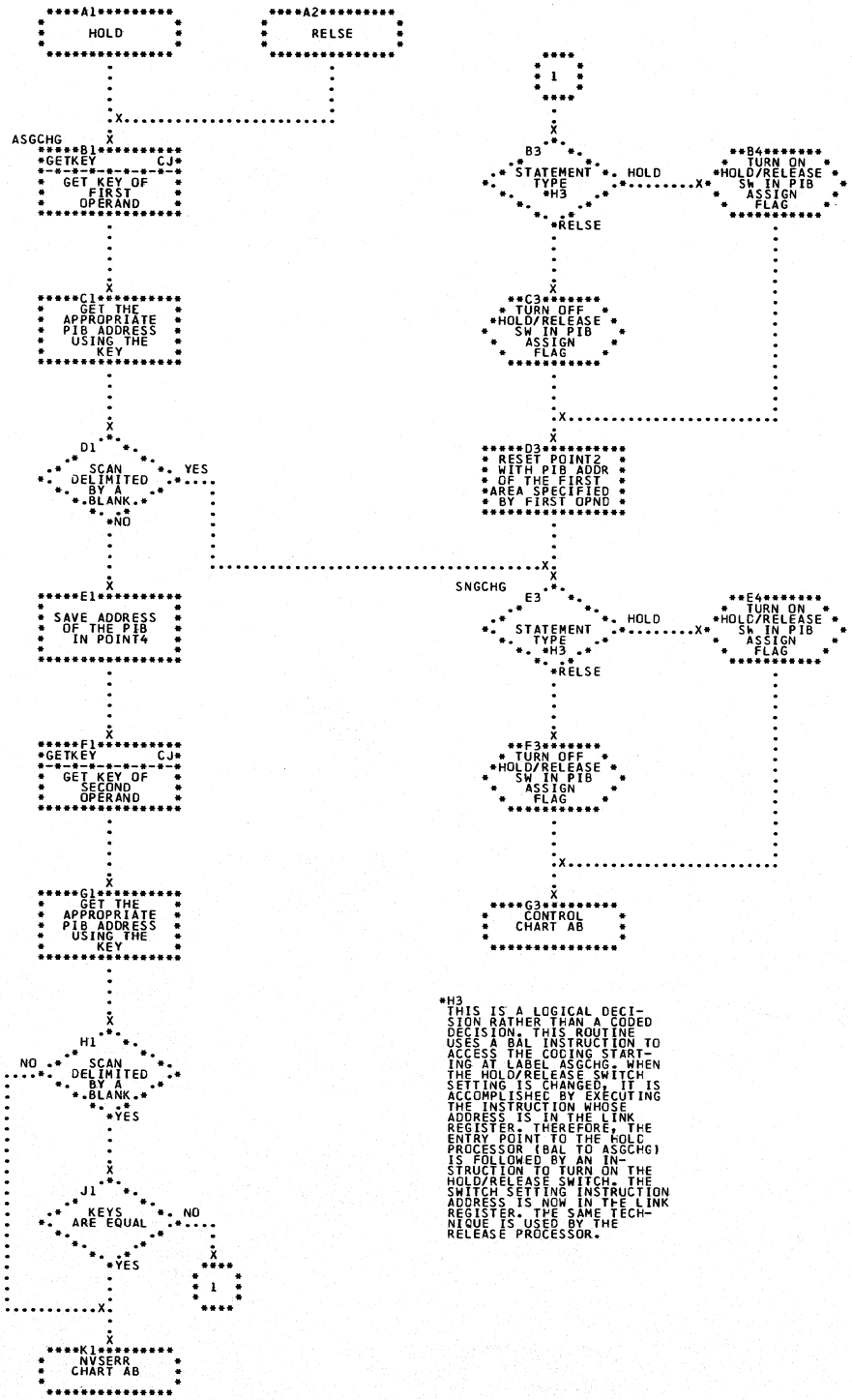


Chart FA. $$BEOJ - Terminated Program I/O Handling and EOJ Processing (Part 1 of 3)
Refer to Chart 09.

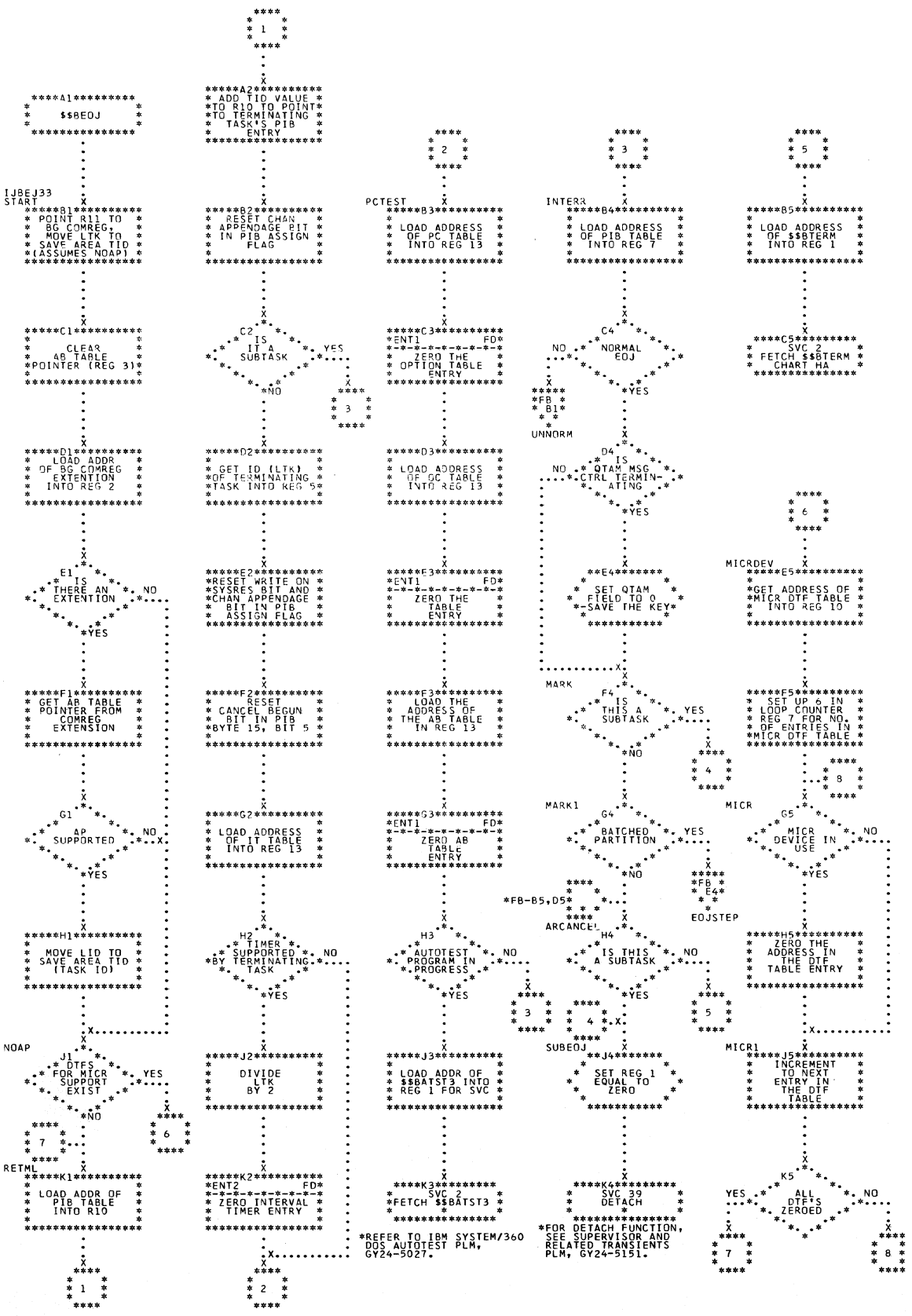


Chart FB. $$BEOJ - Terminated Program I/O Handling and EOJ Processing (Part 2 of 3)
Refer to Chart 09.

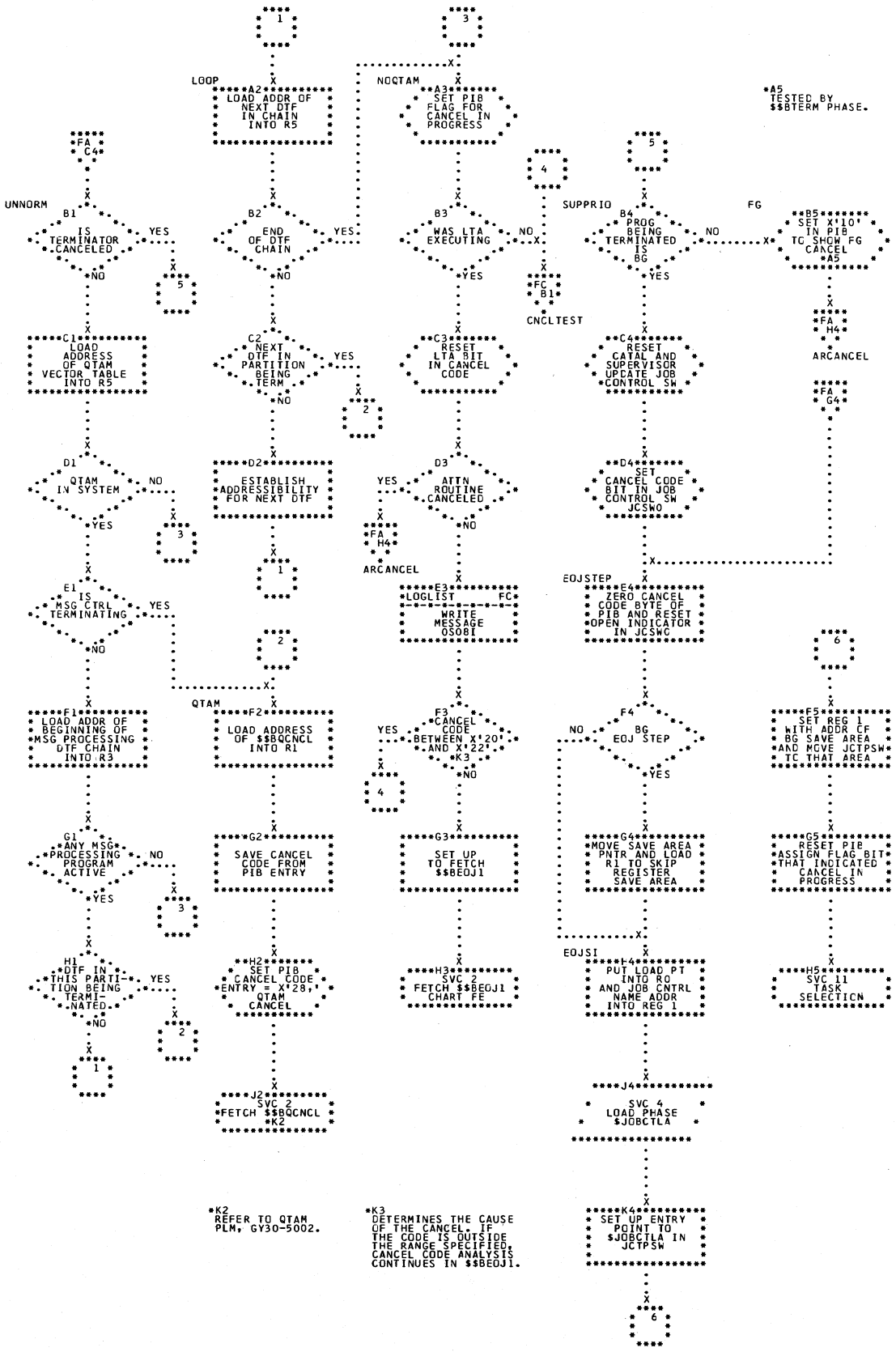


Chart FC. $$BEOJ - Terminated Program I/O Handling and EOJ Processing (Part 3 of 3)
Refer to Chart 09.

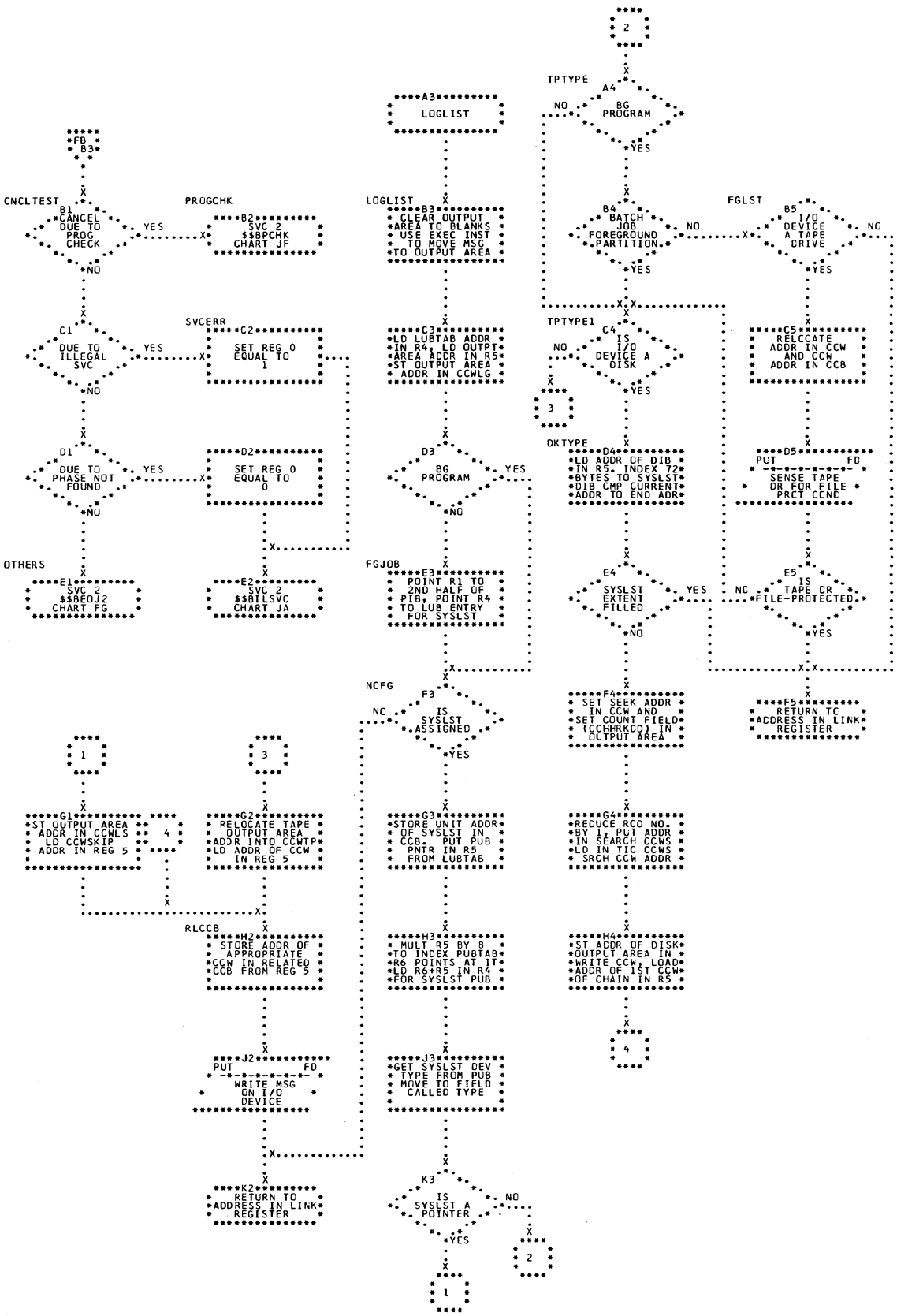


Chart FD. $$BEOJ - Message Output Subroutine and Zero Option Table Subroutine
Refer to Chart 09.

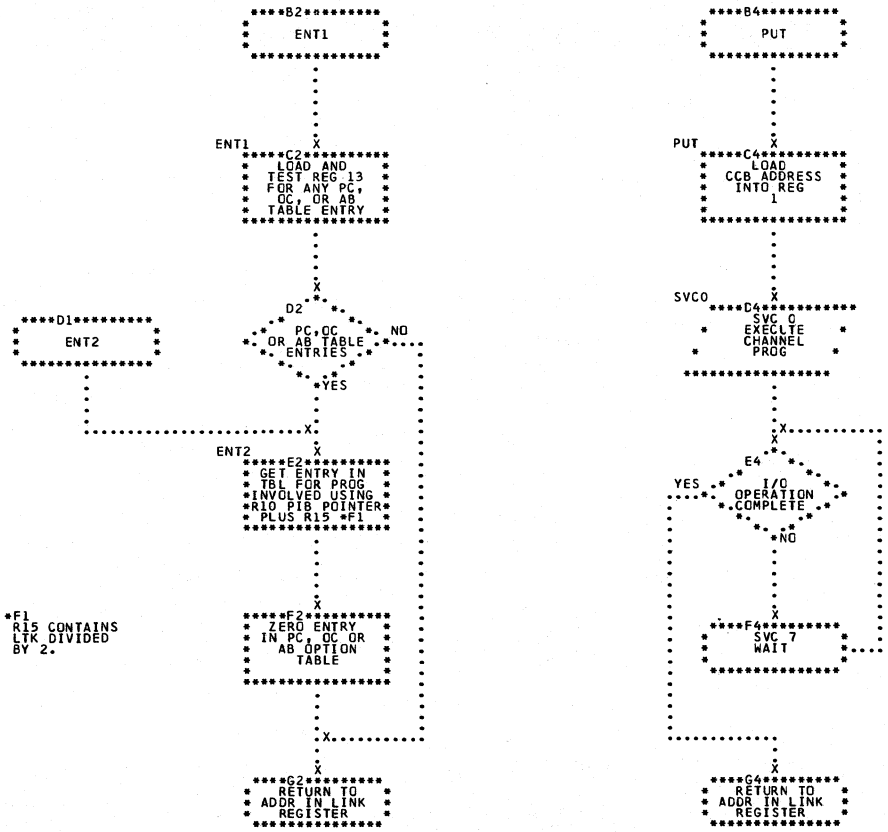


Chart FE. $$BEOJ1 - Prepare Cancel Cause Message
Refer to Chart 10.

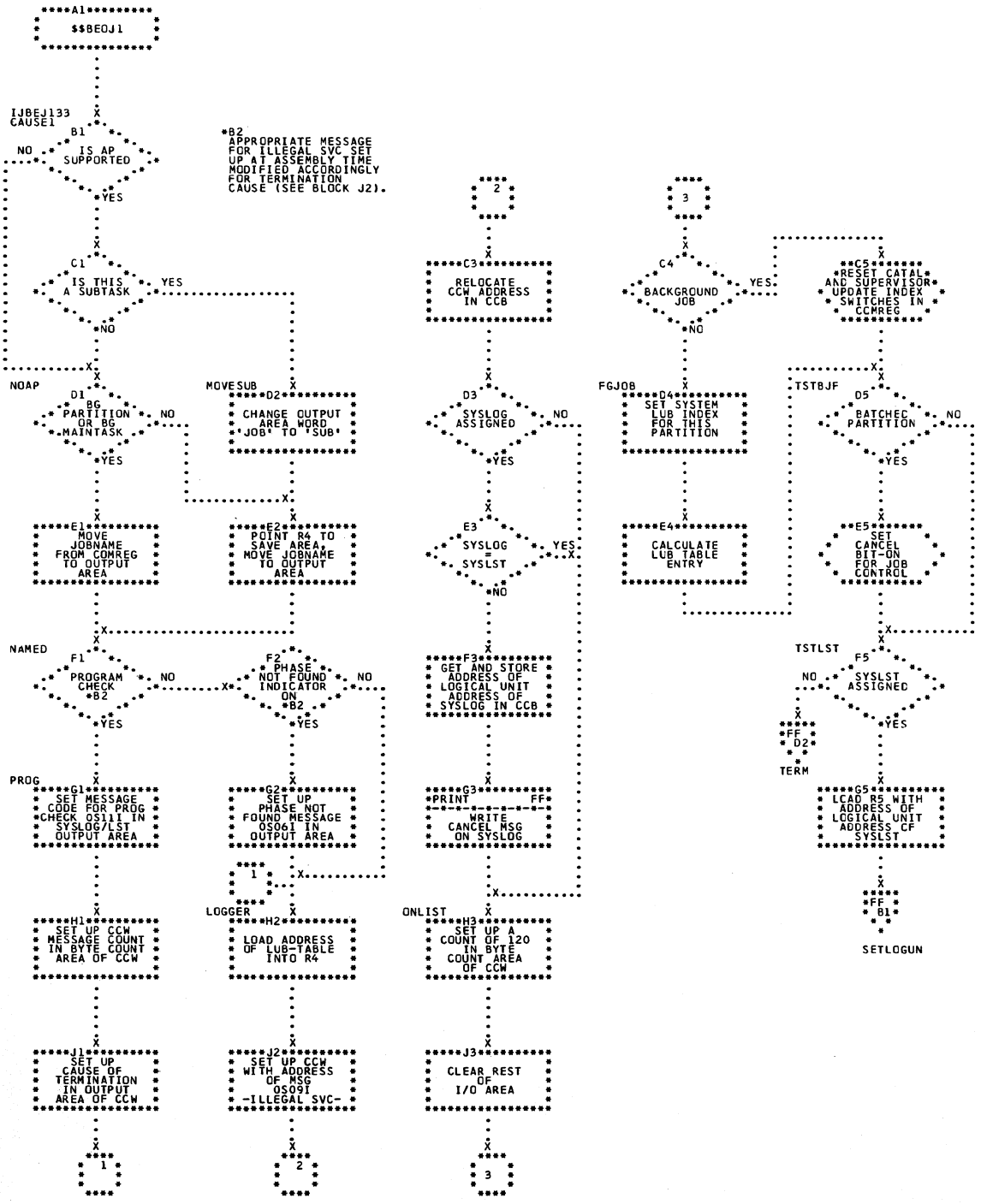


Chart FF. $$BEOJ1 - Output Cancel Message on SYSLST
Refer to Chart 10.

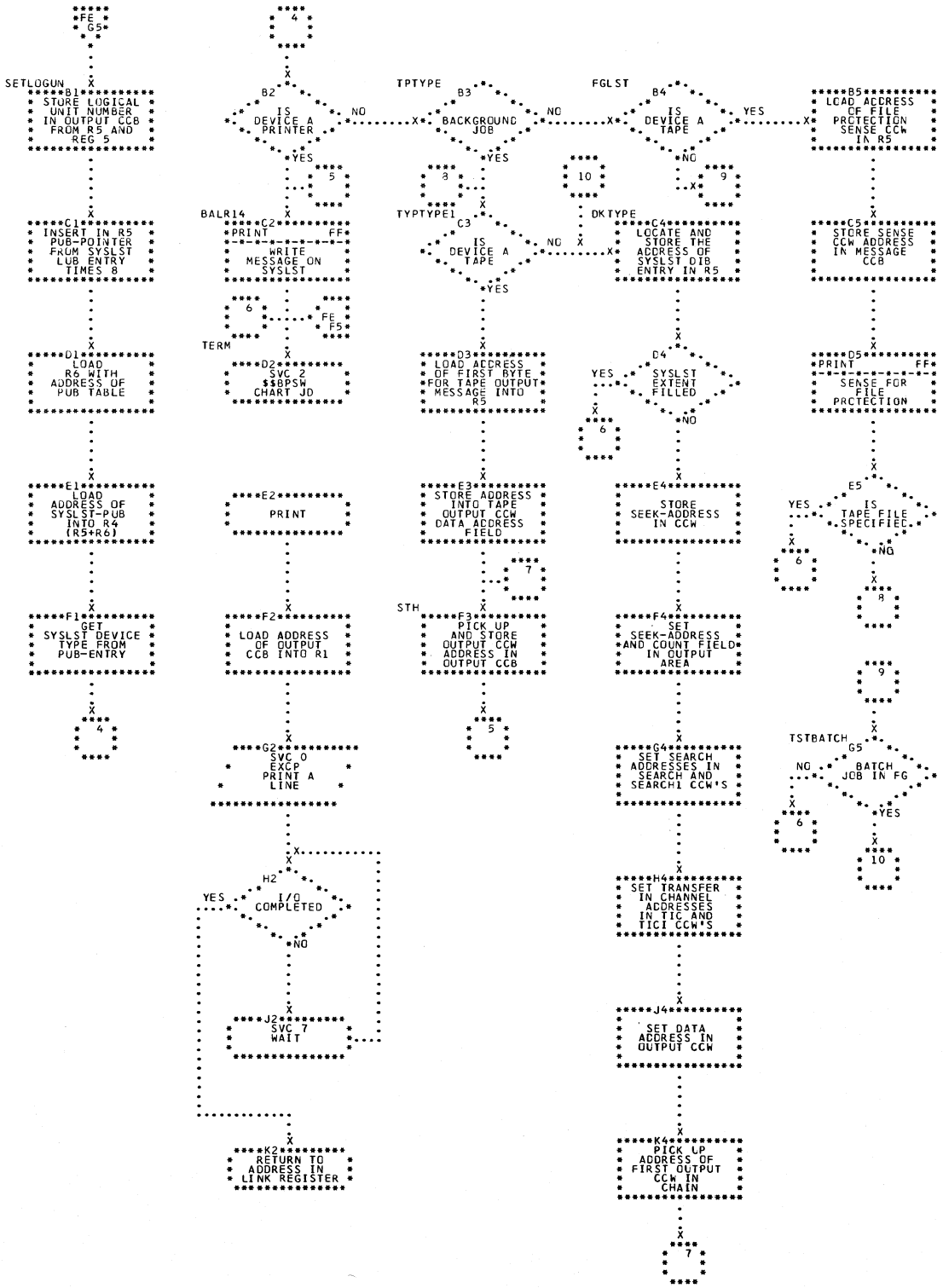


Chart FG. $$BEOJ2 - Select Cancel Message and Program/Task Identification
Refer to Chart 11.

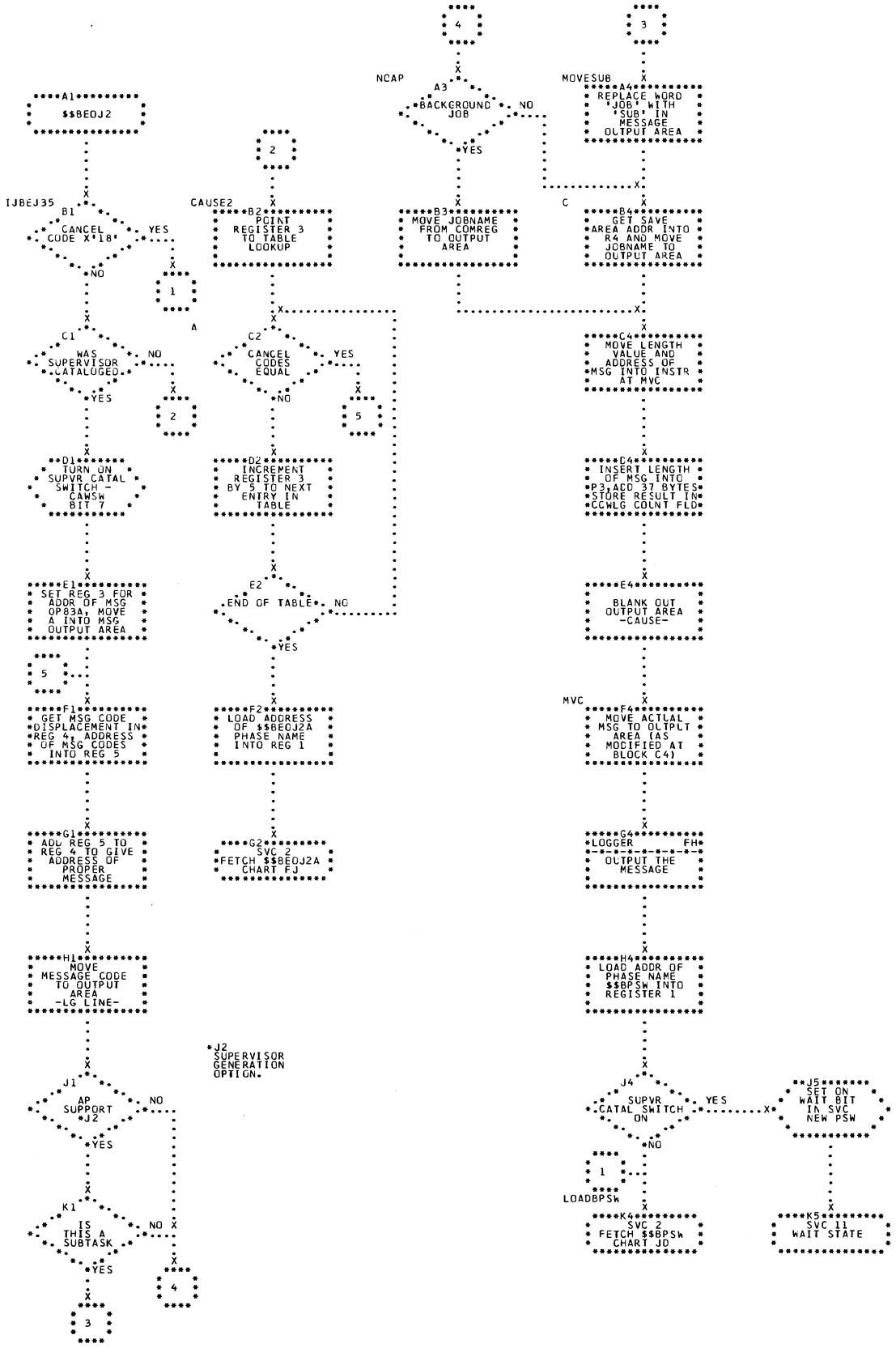


Chart FH. $$BEOJ2 - Select I/O Device and Output the Cancel Message
Refer to Chart 11.

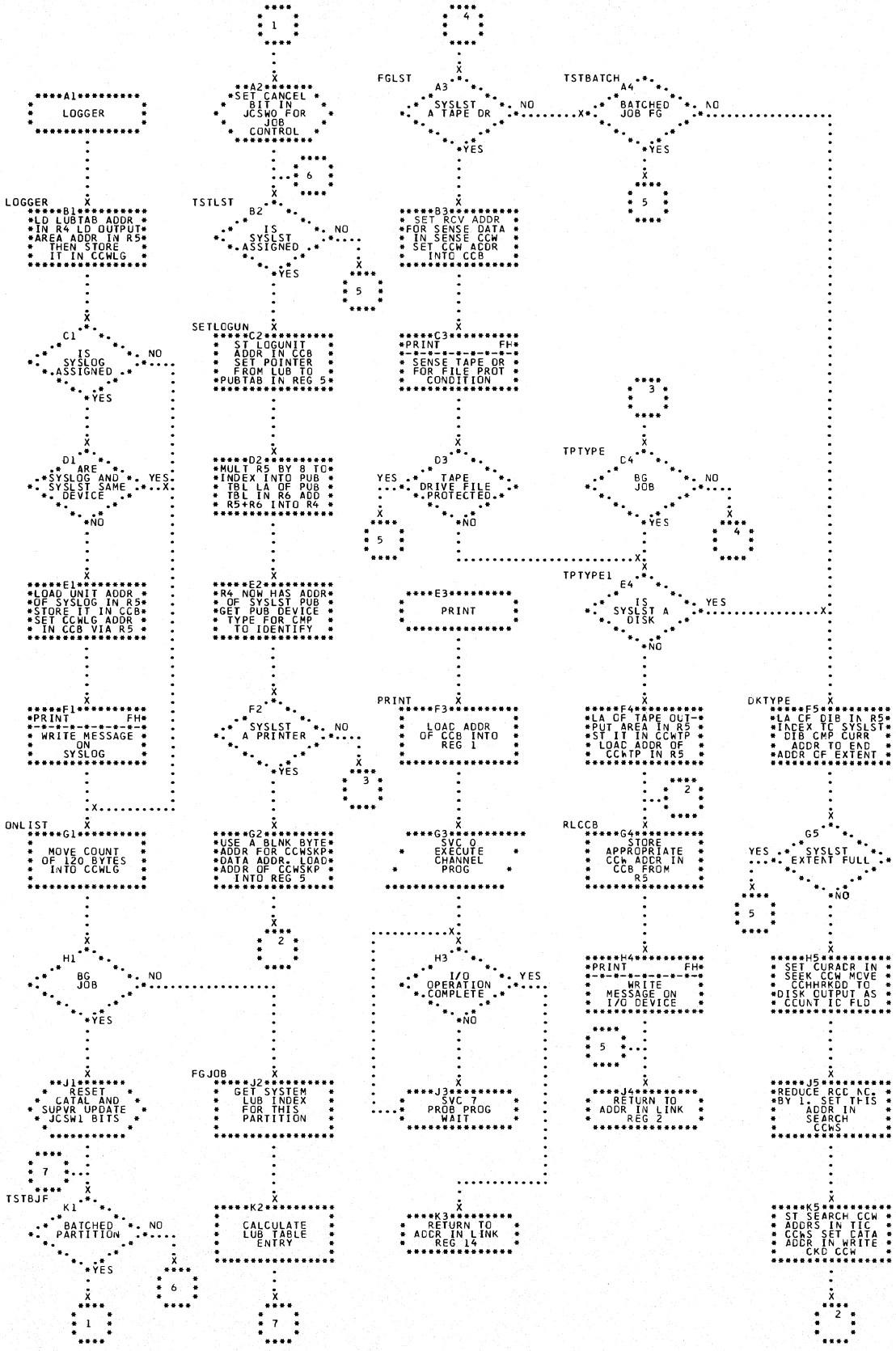


Chart FJ. $$BEOJ2A - Select Cancel Message and Program/Task Identification
Refer to Chart 11.

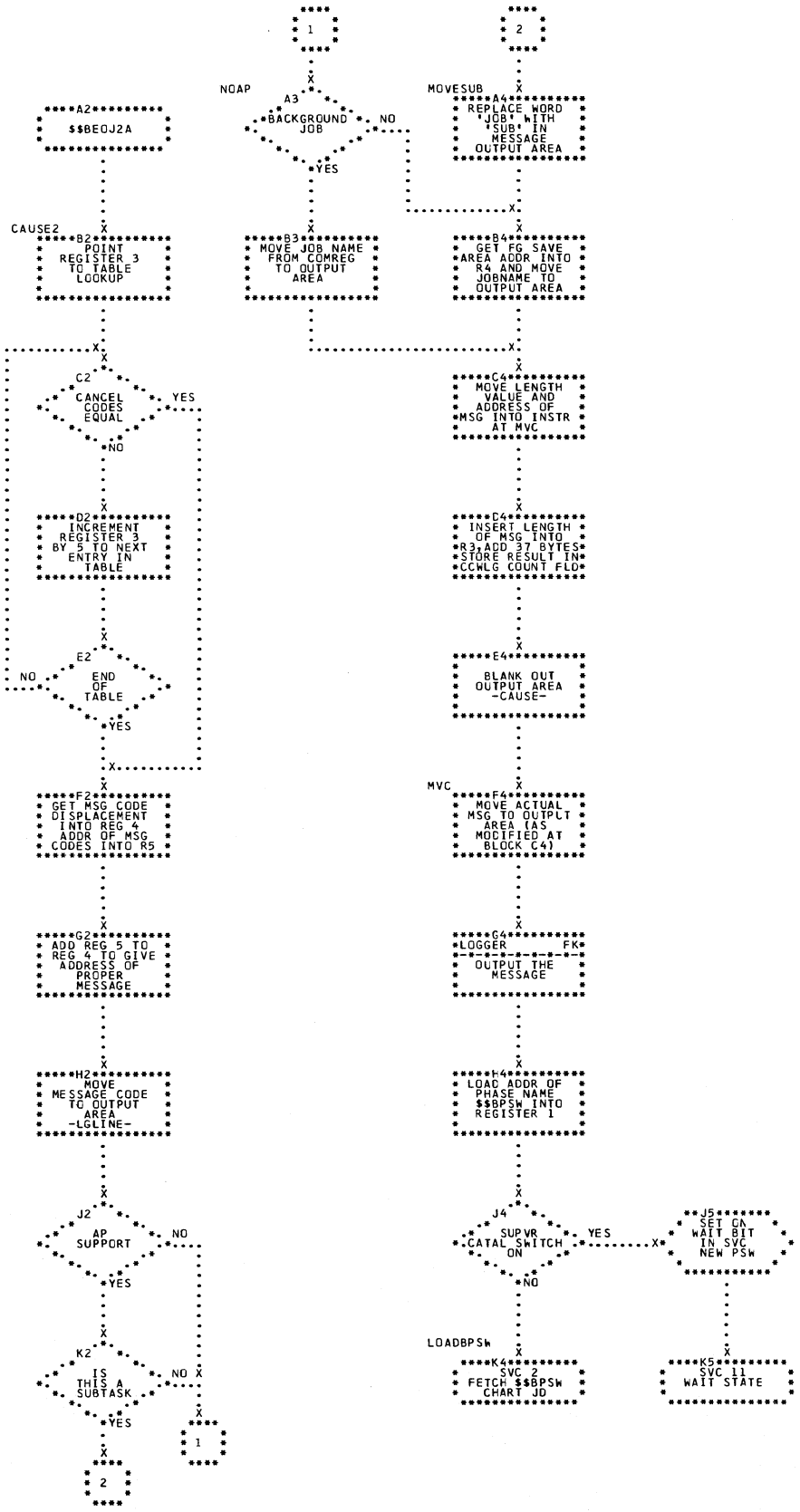


Chart FK. $$BEOJ2A - Select I/O Device and Output the Cancel Message
Refer to Chart 11.

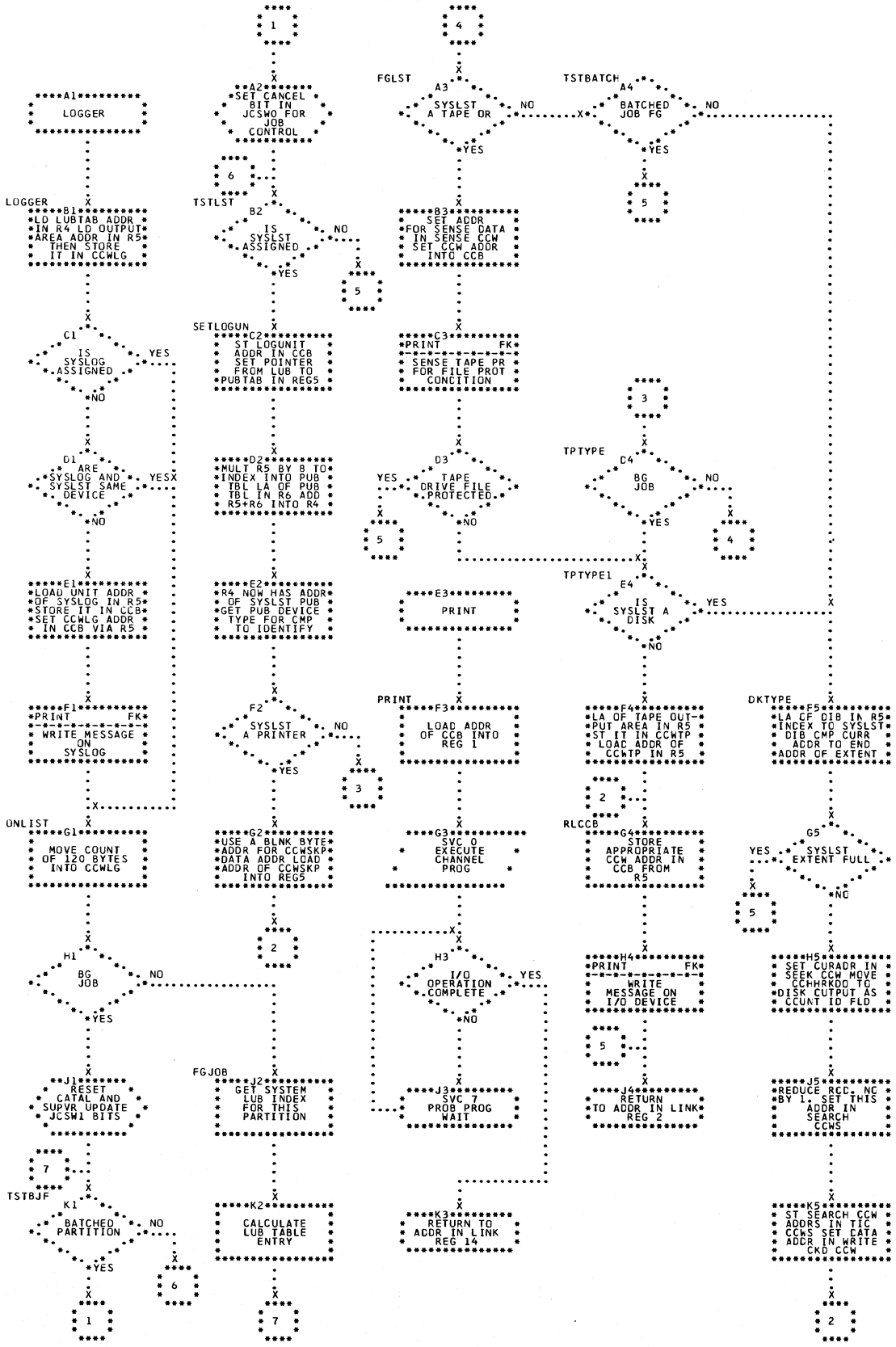


Chart FL. $$BEOJ3 - Quiesce I/O for TP Devices
Refer to Chart 08.

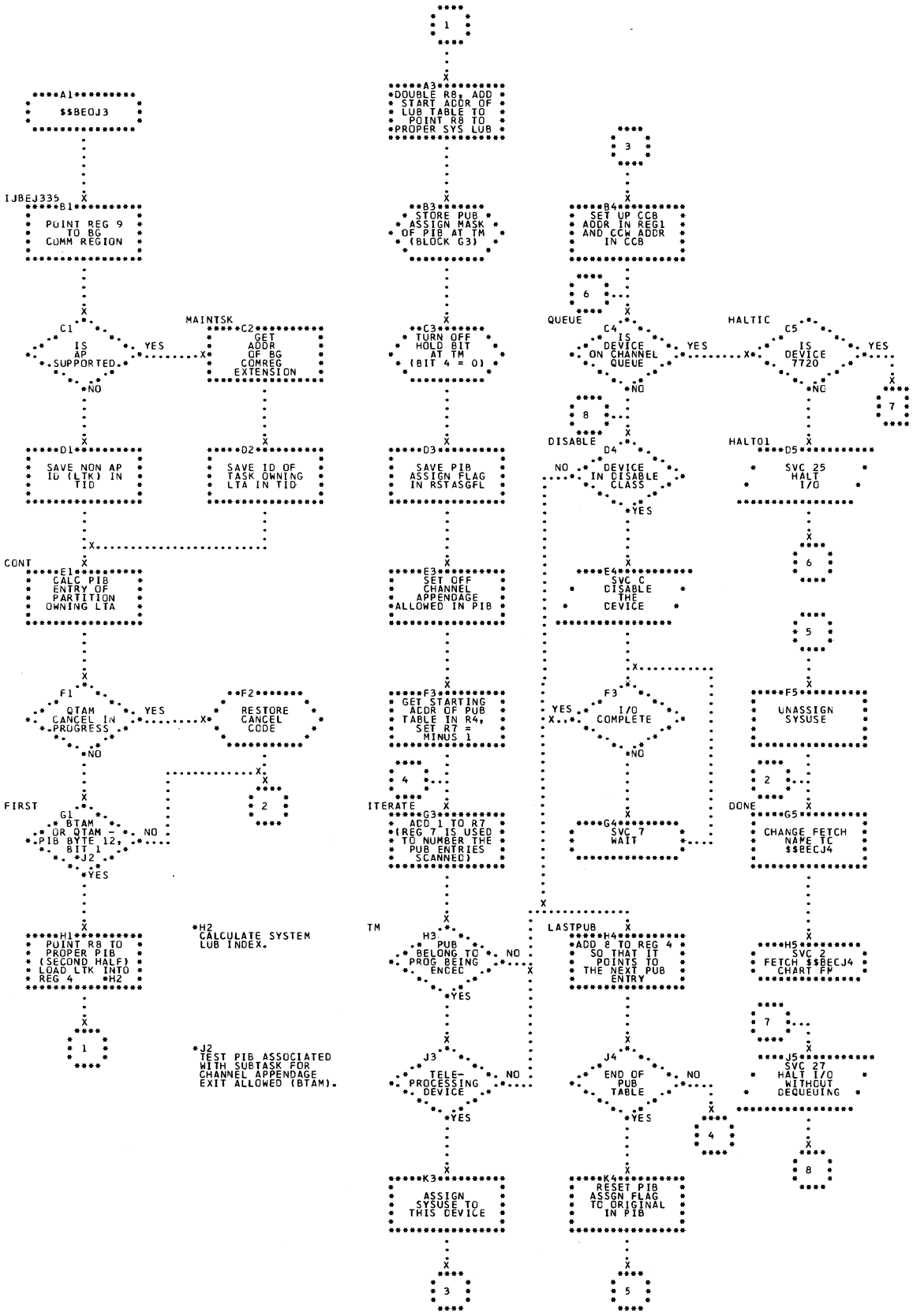


Chart FM. $$BEOJ4 - Quiesce I/O for Non-TP Devices
Refer to Chart 08.

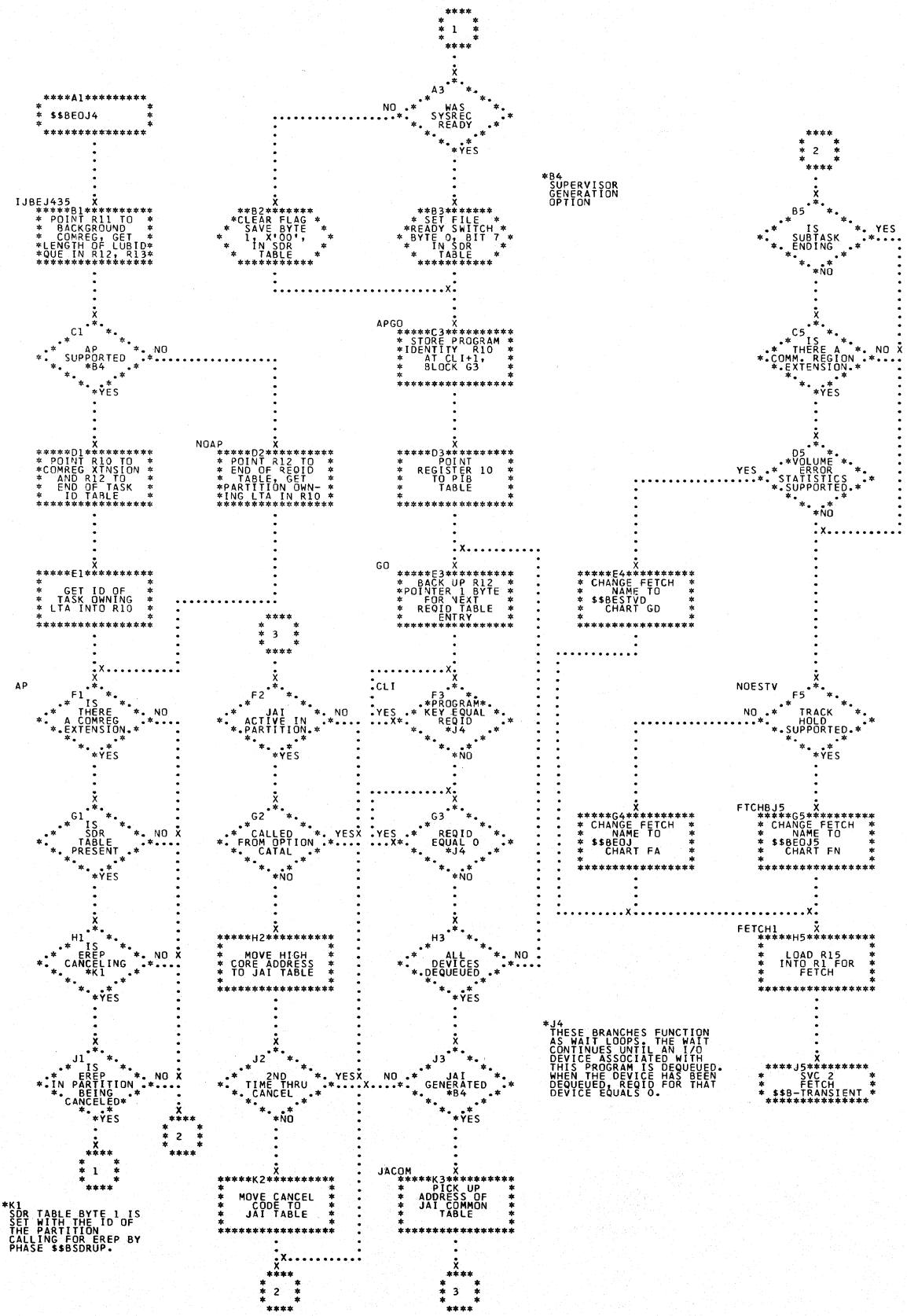


Chart FN. $$BEOJ5 - Release Tracks Held by Task/Partition
Refer to Chart 08.

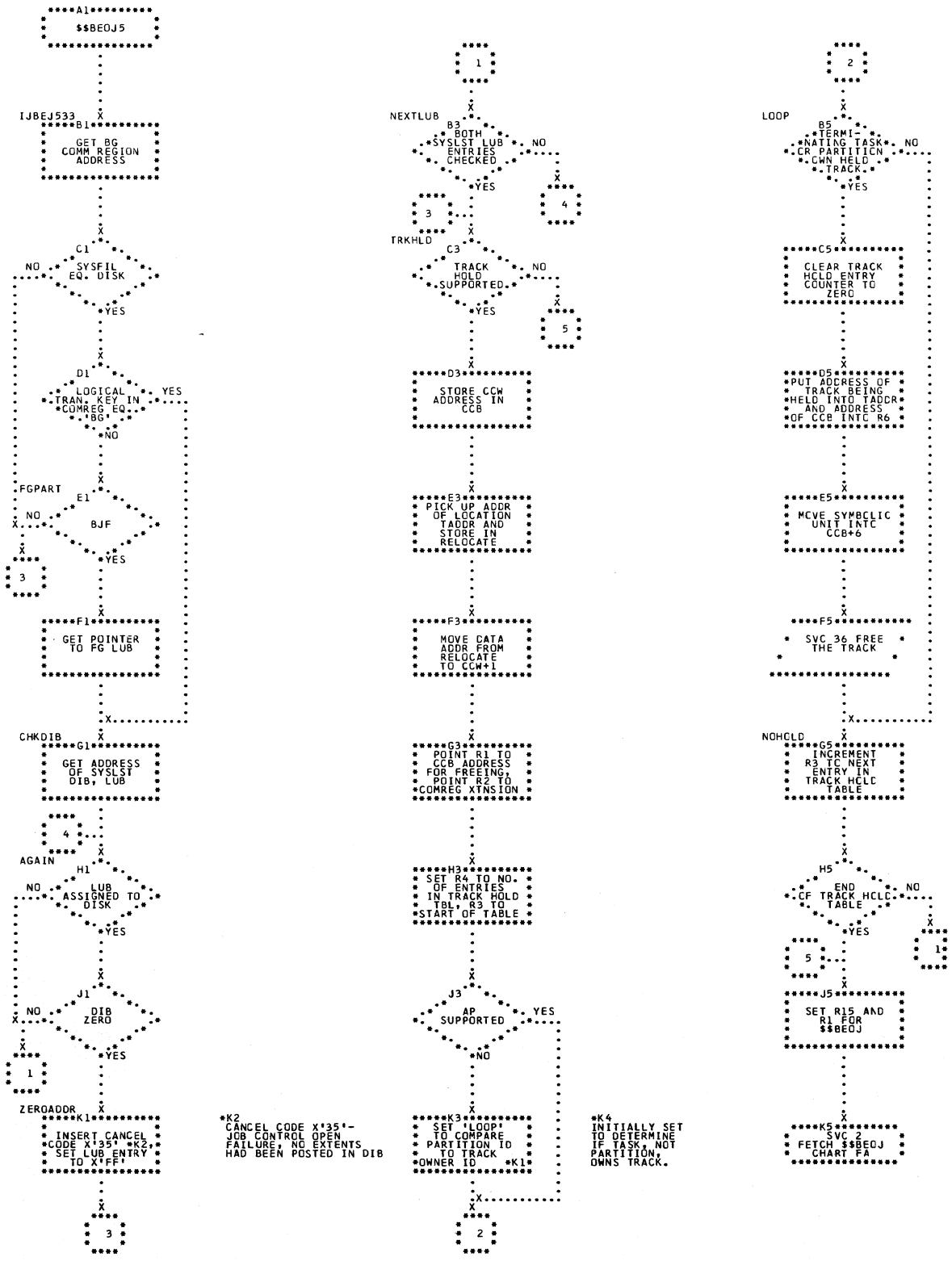


Chart GA. $$BESTVA - Phase 1 of Tape Volume Error Statistics
Refer to Chart 08.

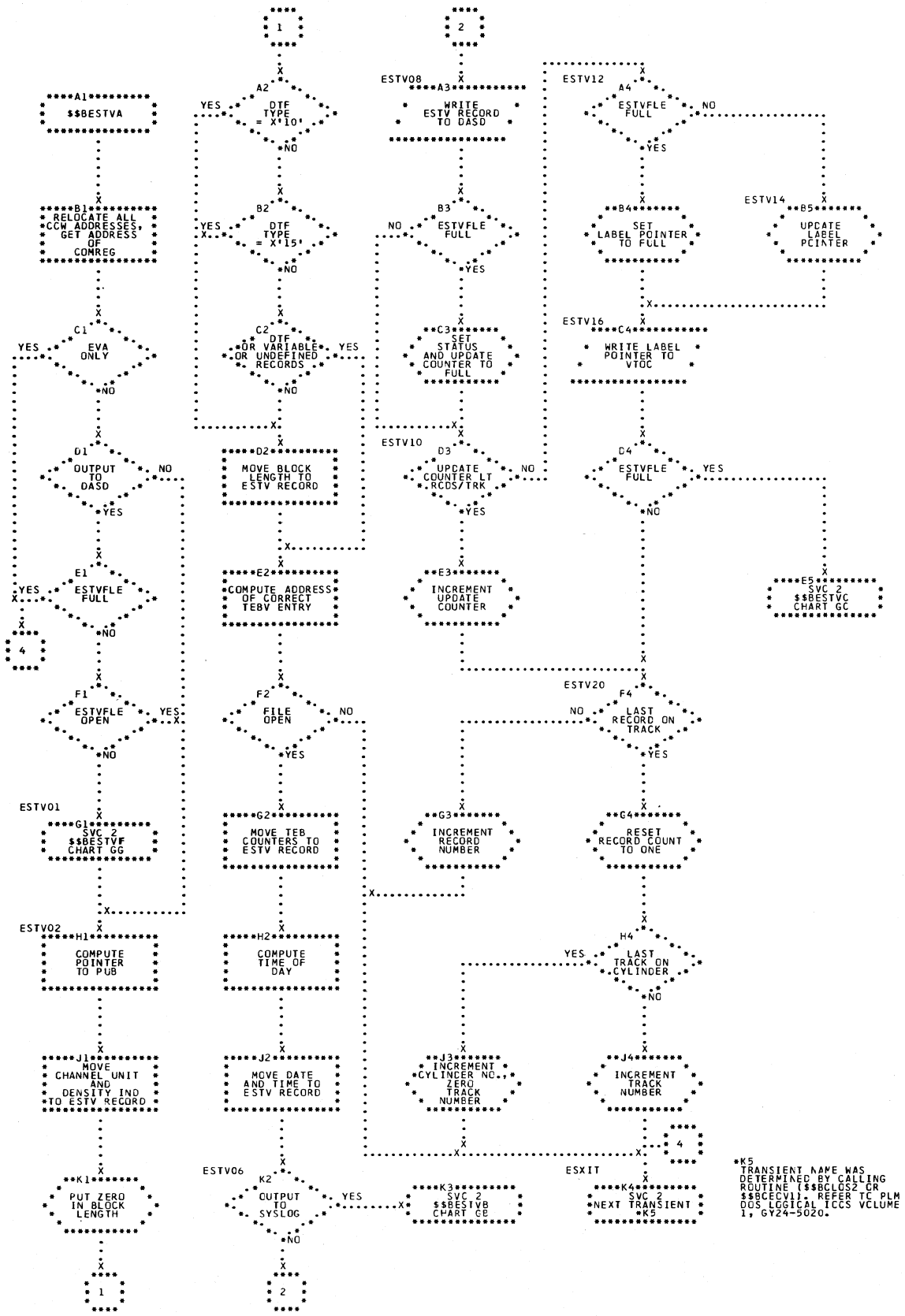


Chart GB. $$BESTVB - Phase 2 of Tape Volume Error Statistics
Refer to Chart 08.

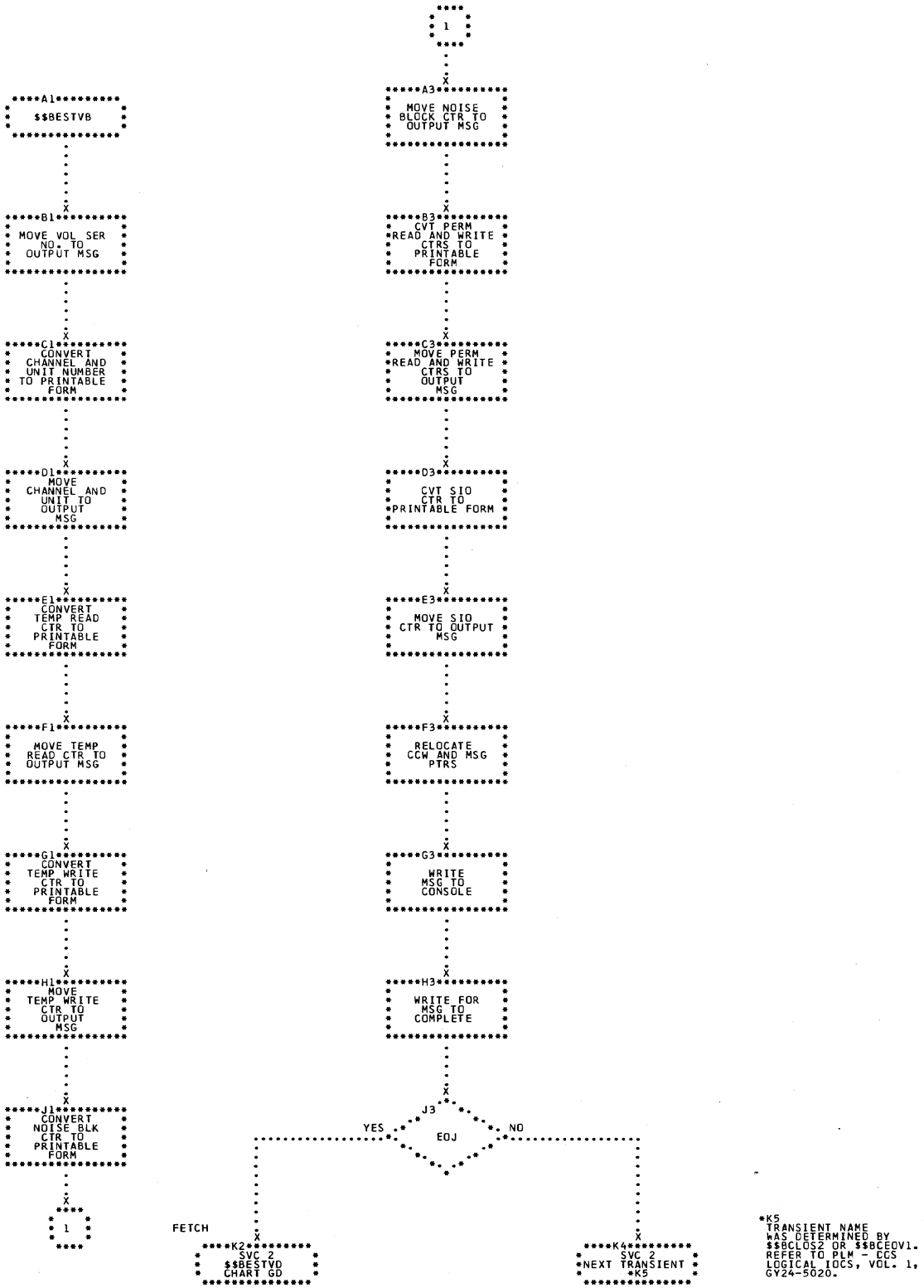


Chart GC. $$BESTVC - Phase 3 of Tape Volume Error Statistics
Refer to Chart 08.

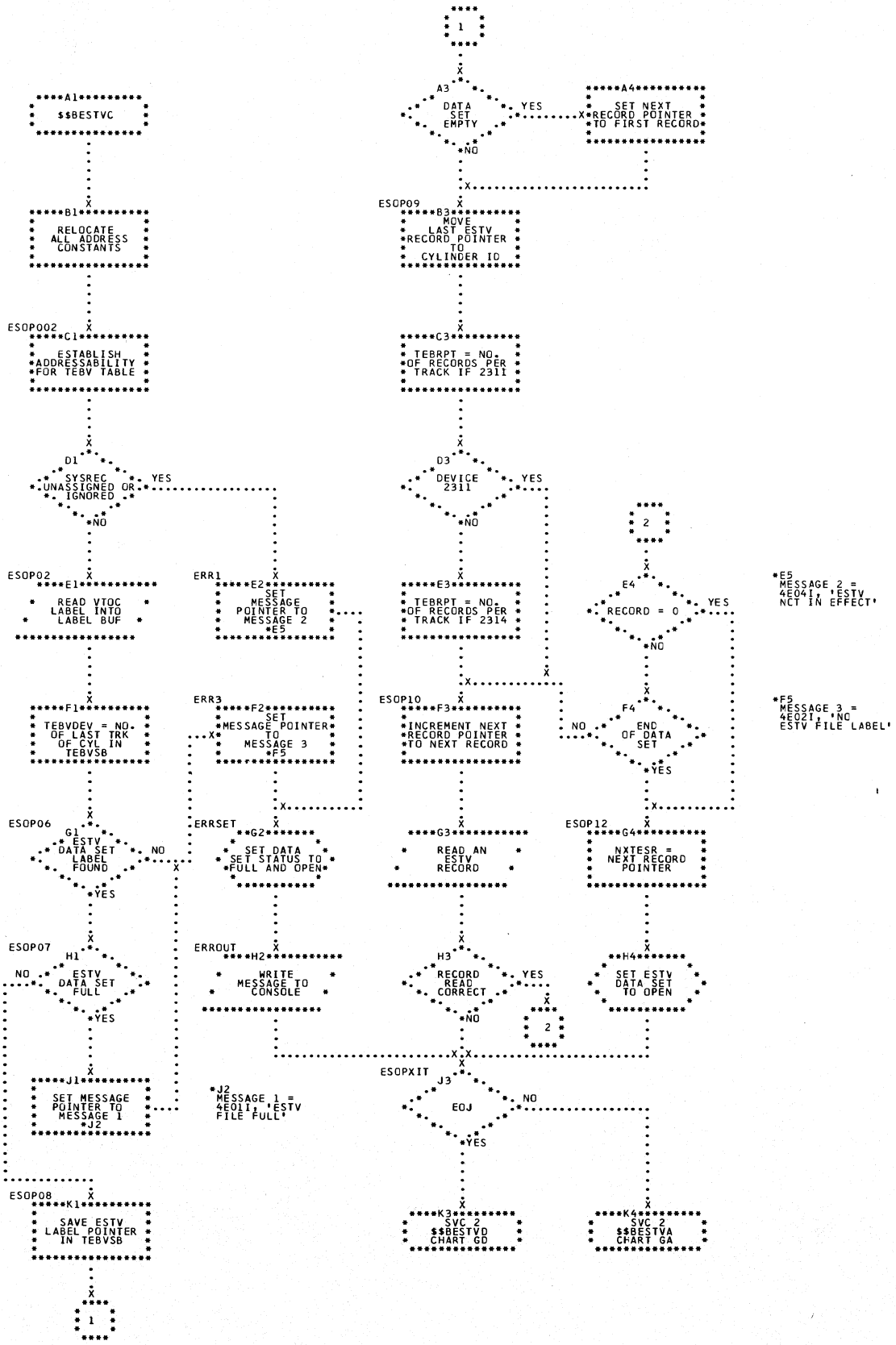


Chart GD. $$BESTVD - Phase 4 of Tape Volume Error Statistics (Part 1 of 2)
Refer to Chart 08.

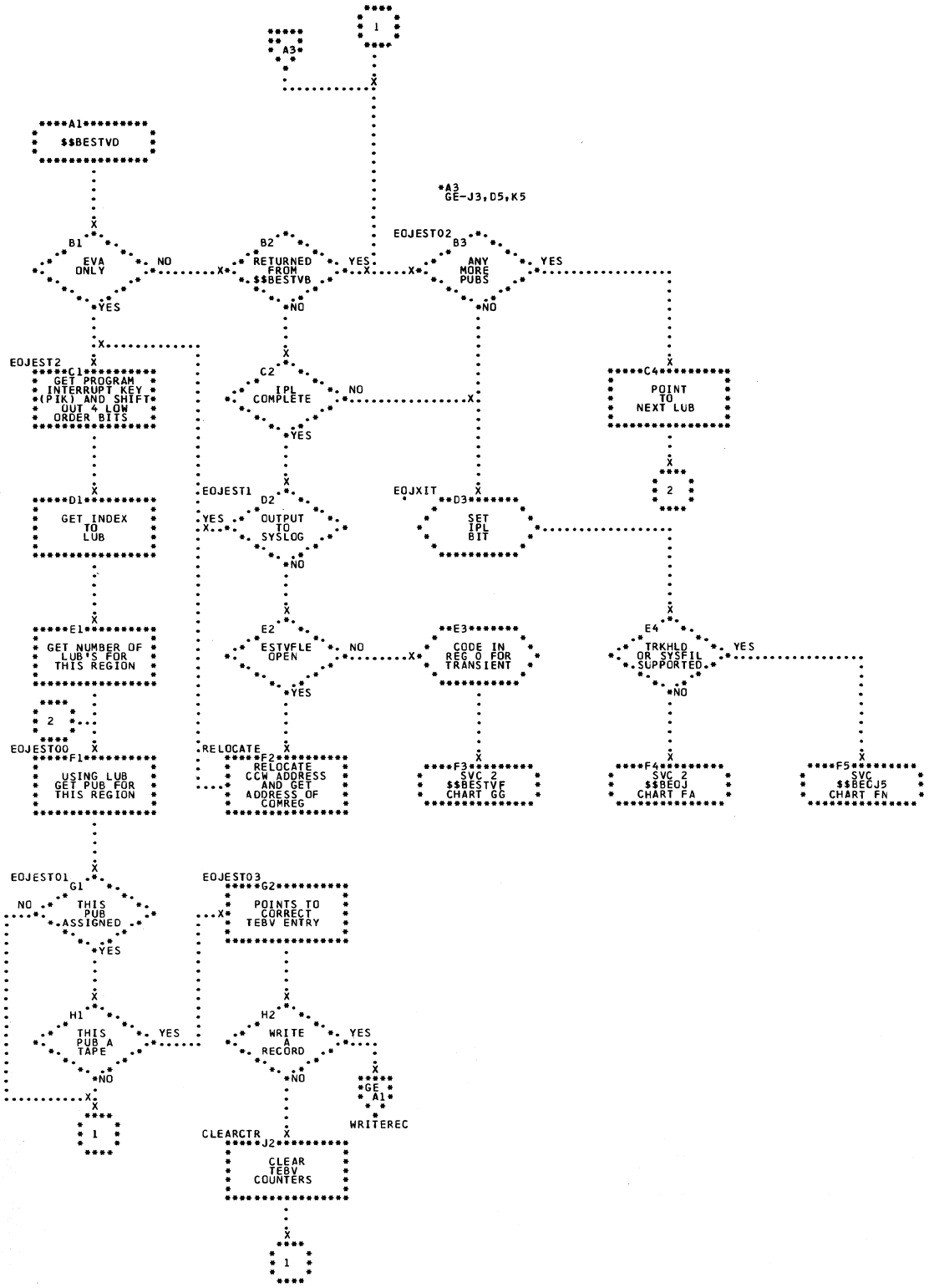


Chart GE. $$BESTVD - Phase 4 of Tape Volume Error Statistics (Part 2 of 2)
Refer to Chart 08.

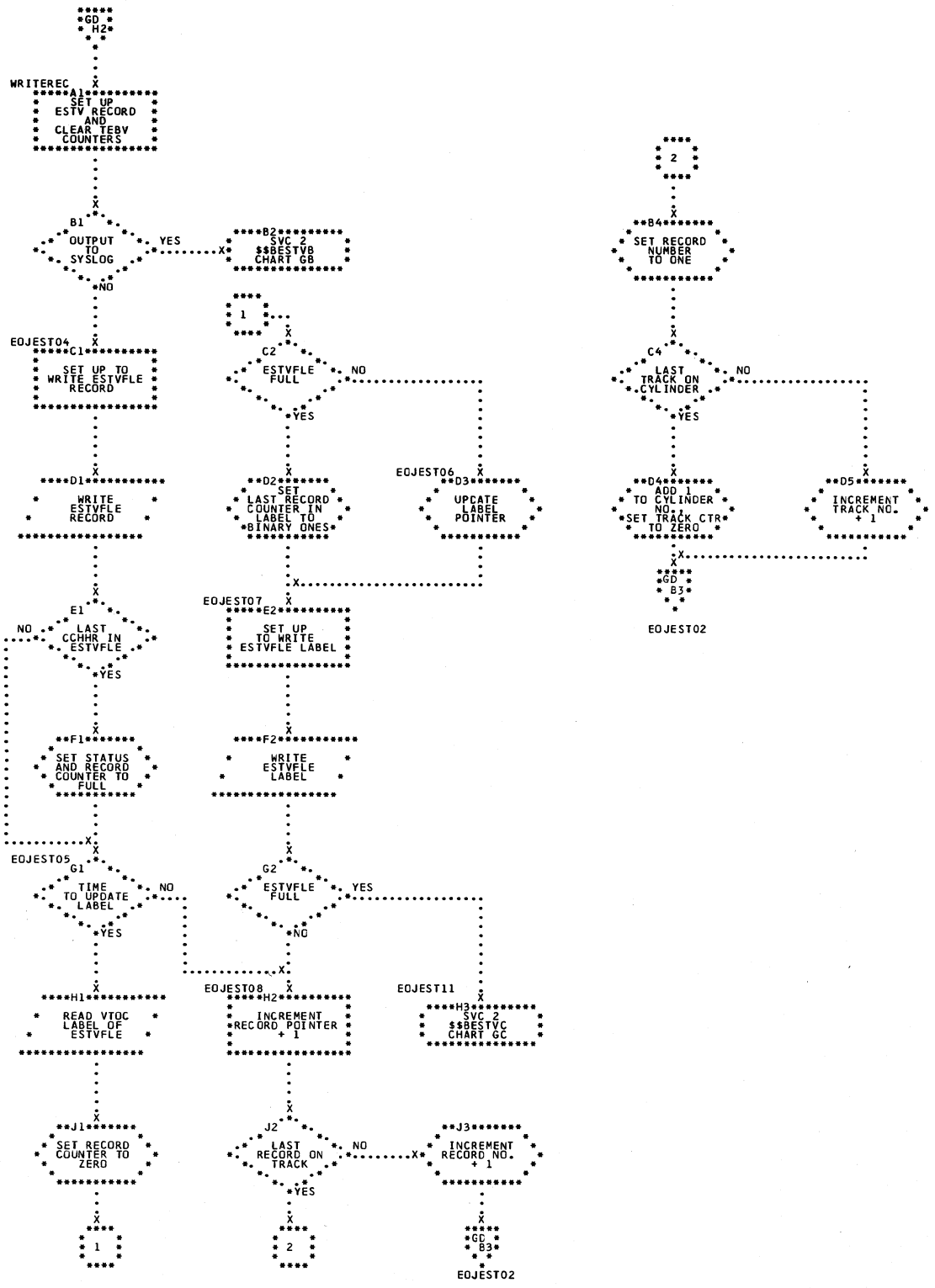


Chart GF. $$BESTVE - Phase 5 of Tape Volume Error Statistics
Refer to Chart 08.

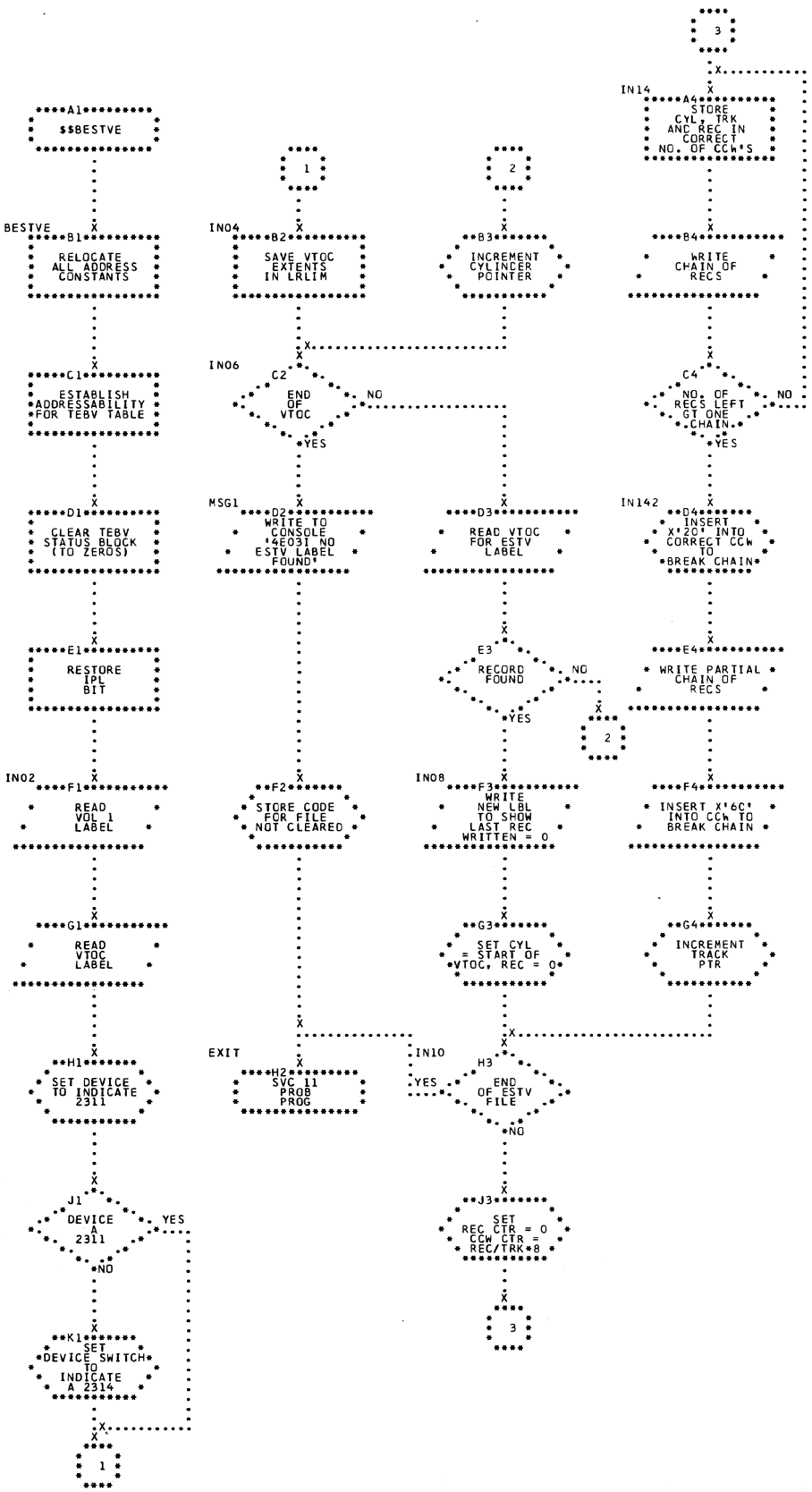


Chart GG. $$BESTVF - Phase 6 of Tape Volume Error Statistics
Refer to Chart 08.

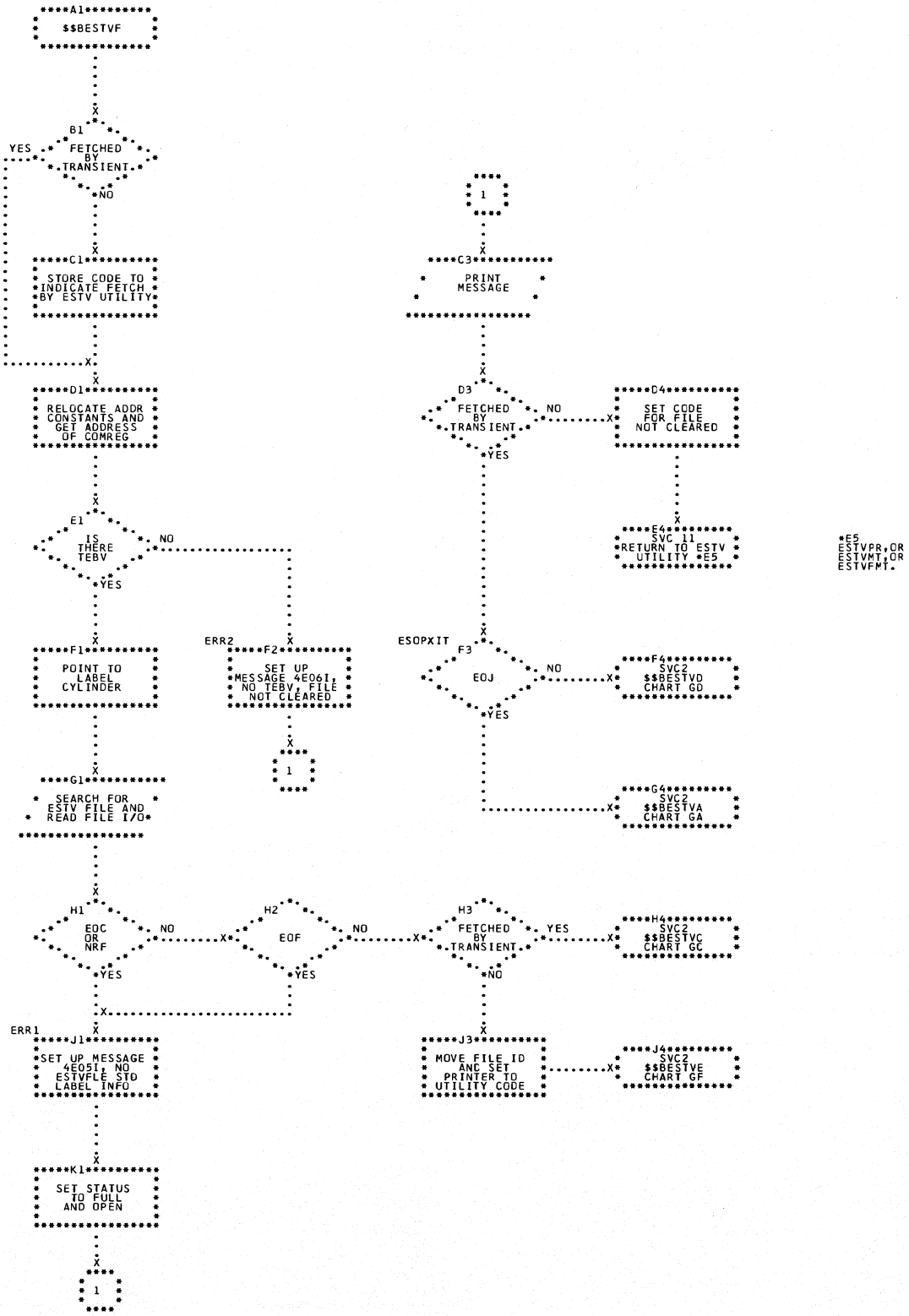


Chart HA. $$BTERM - Reset Foreground PUB Ownership and Detach Attention Routine
Refer to Chart 09.

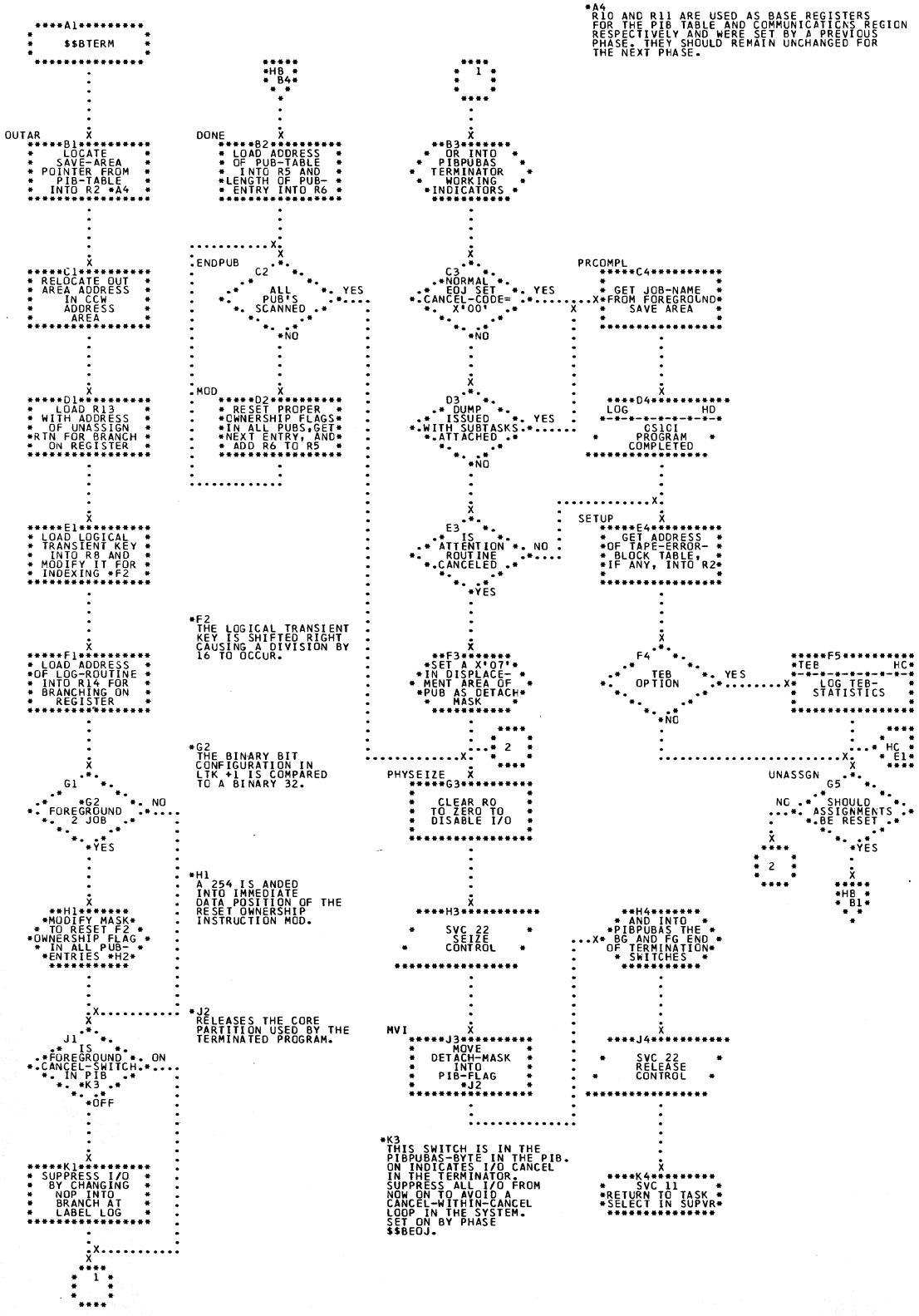


Chart HB. $$BTERM - Reset JIBs for I/O Device of Terminated Program
Refer to Chart 09.

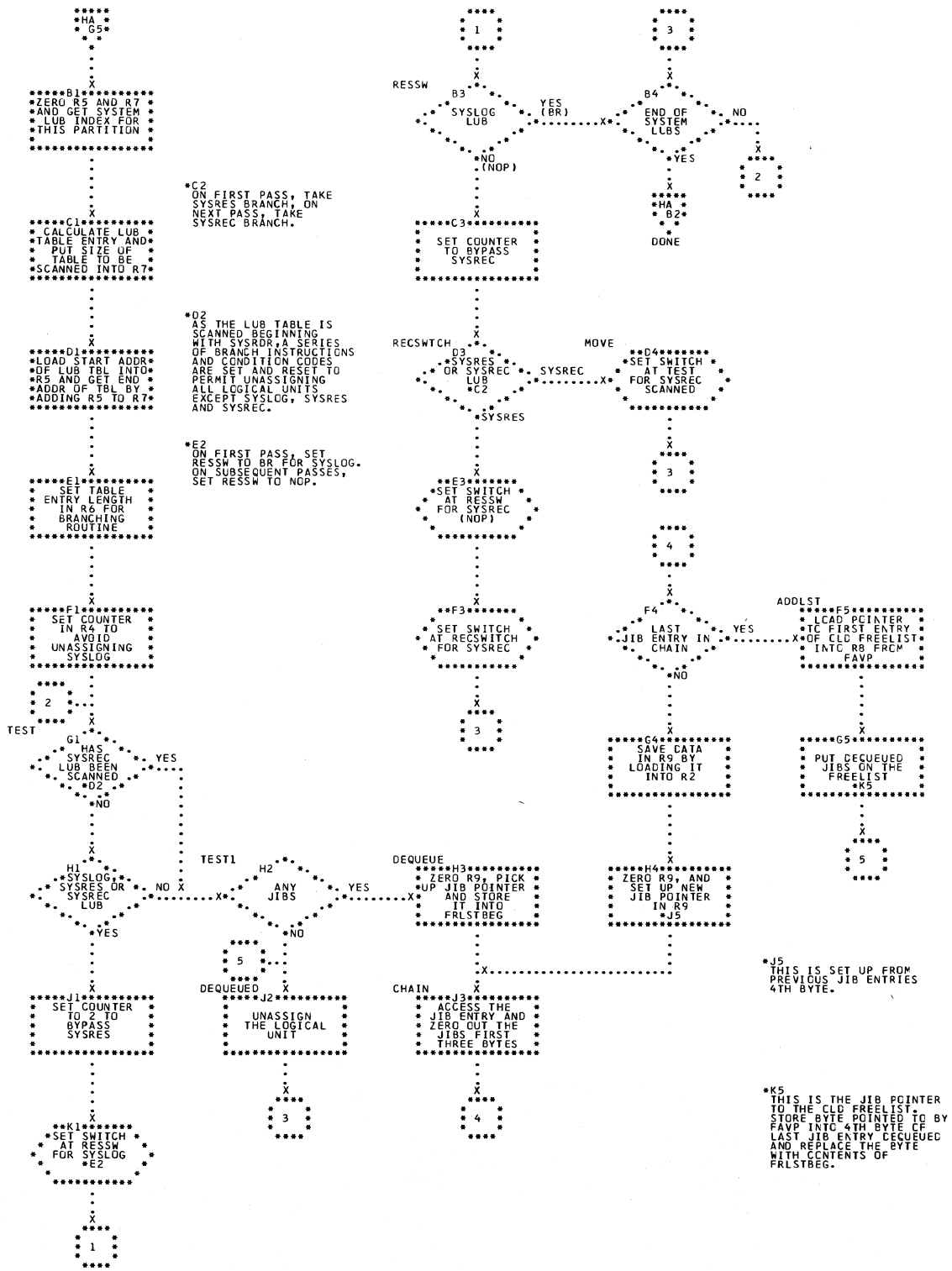


Chart HC. $$BTERM - Get TEB Statistics and Reset TEBs
Refer to Chart 09.

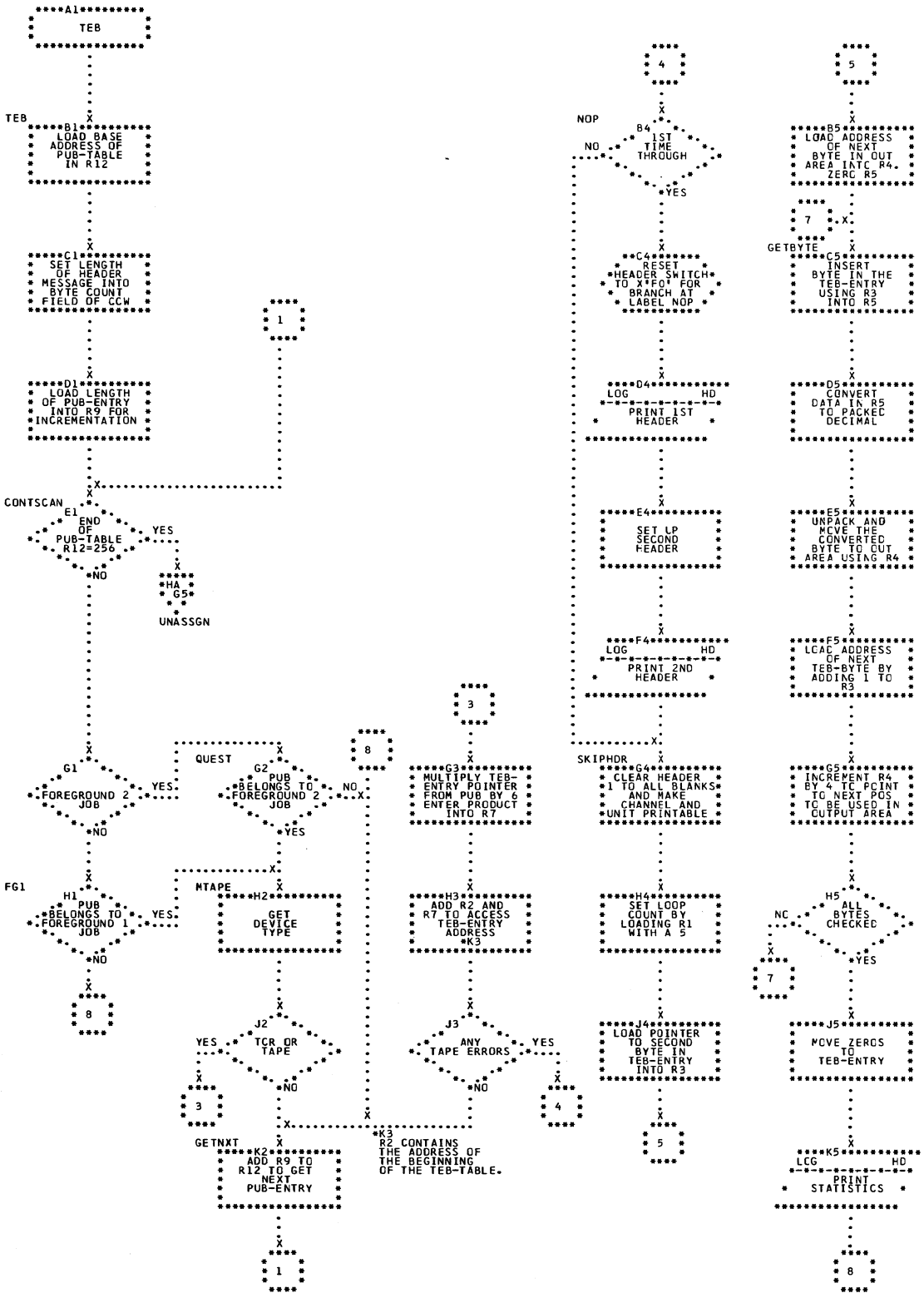


Chart HD. $$BTERM - Print Message and TEB Statistics Subroutine
Refer to Chart 09.

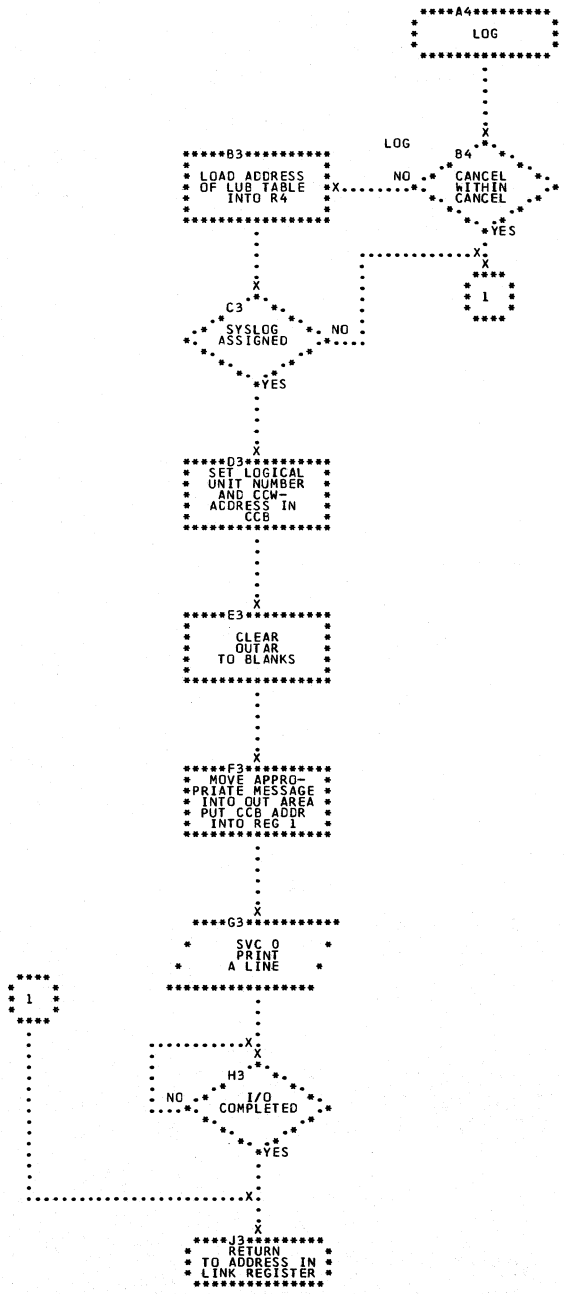


Chart JA. $$BILSVC - Prepare Information about Cancel Cause
Refer to Chart 11.

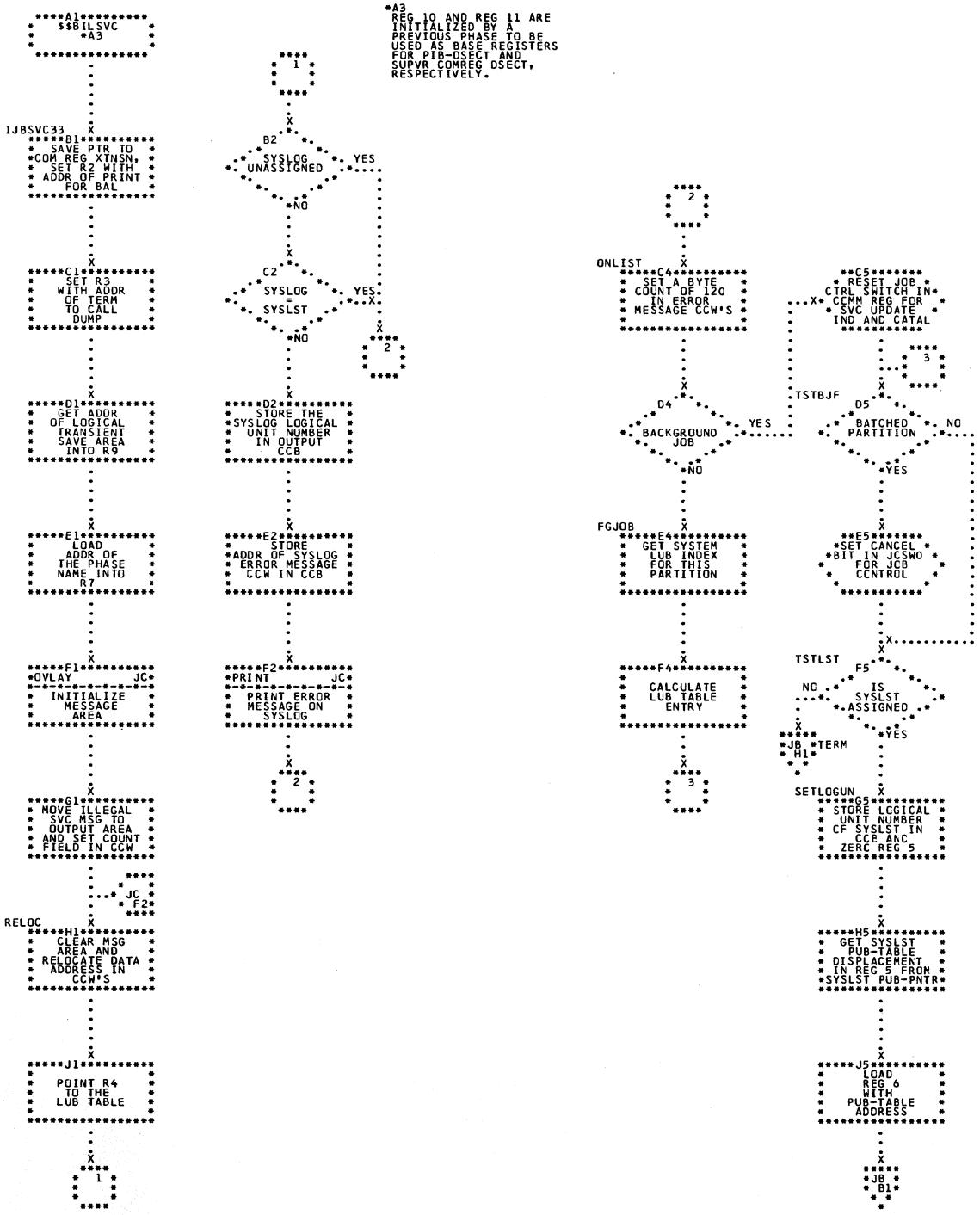


Chart JB. $$BILSVC - Select I/O Device and Prepare to Output a Message
Refer to Chart 11.

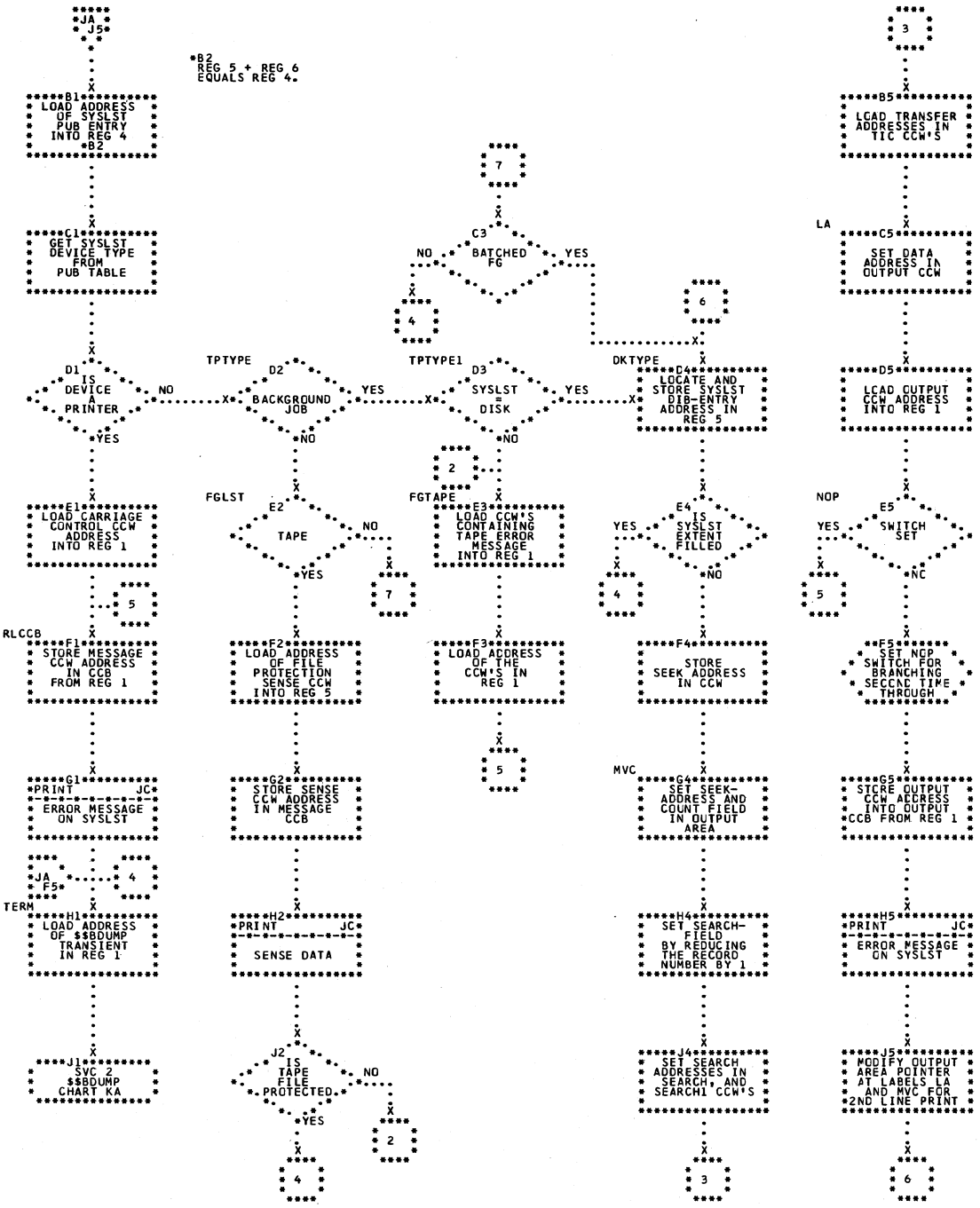


Chart JC. $$BILSVC - Message Initialization and Output Subroutines
Refer to Chart 11.

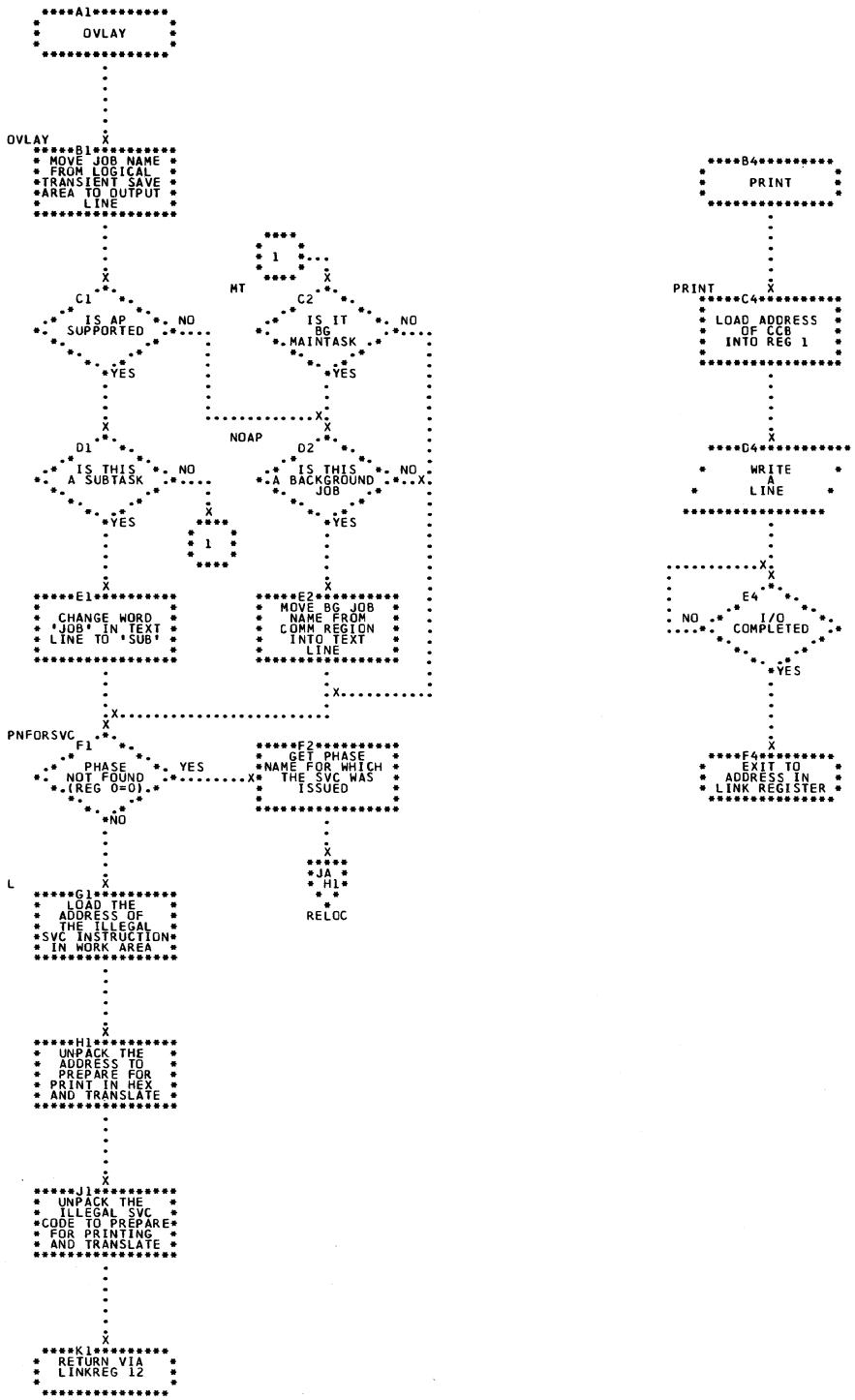


Chart JD. $$BPSW - Prepare Canceled Program's PSW for Output Message and PIOCS Subroutine

Refer to Chart 11.

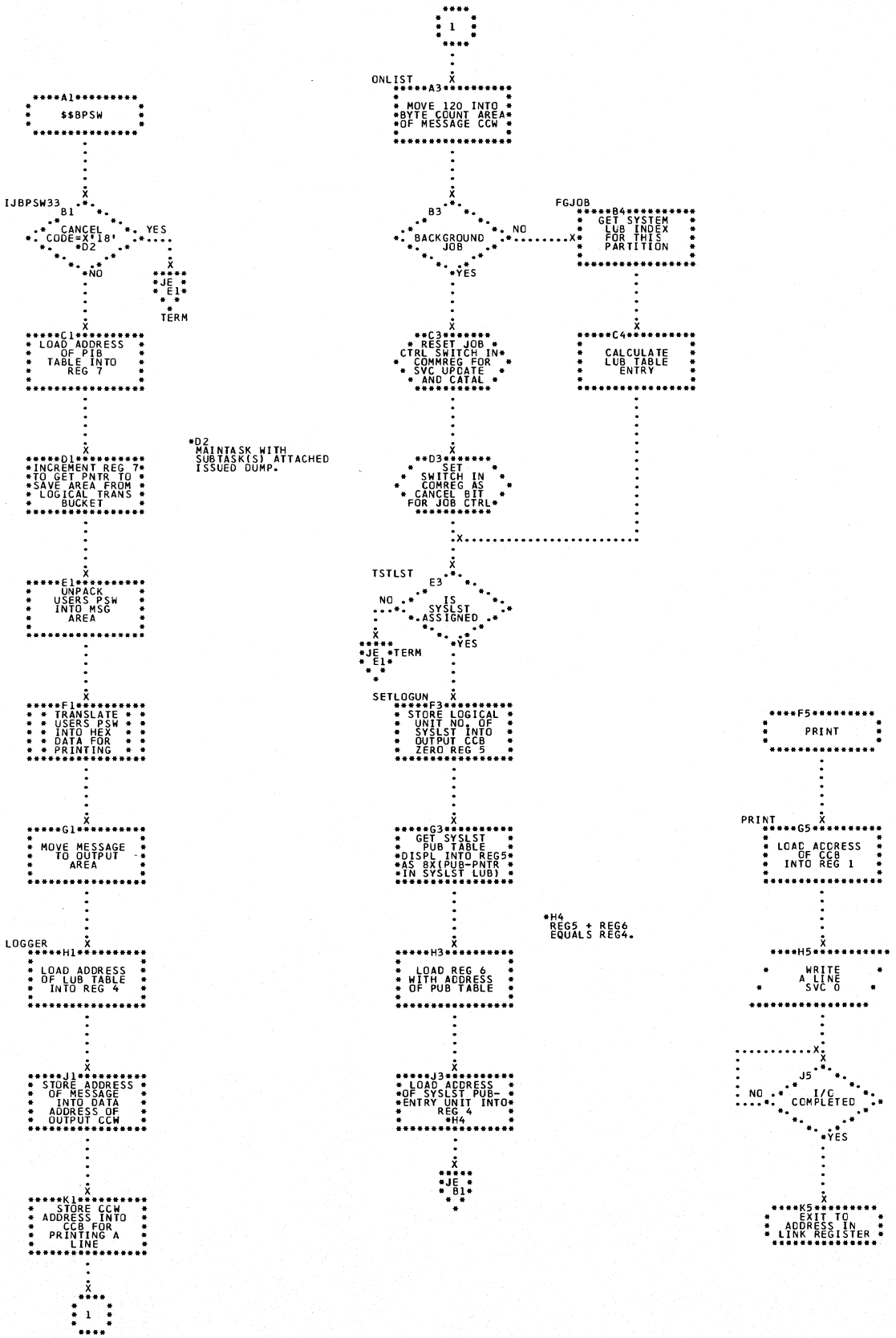


Chart JE. $$BPSW - Select I/O Device, and Prepare to Output a Message
Refer to Chart 11.

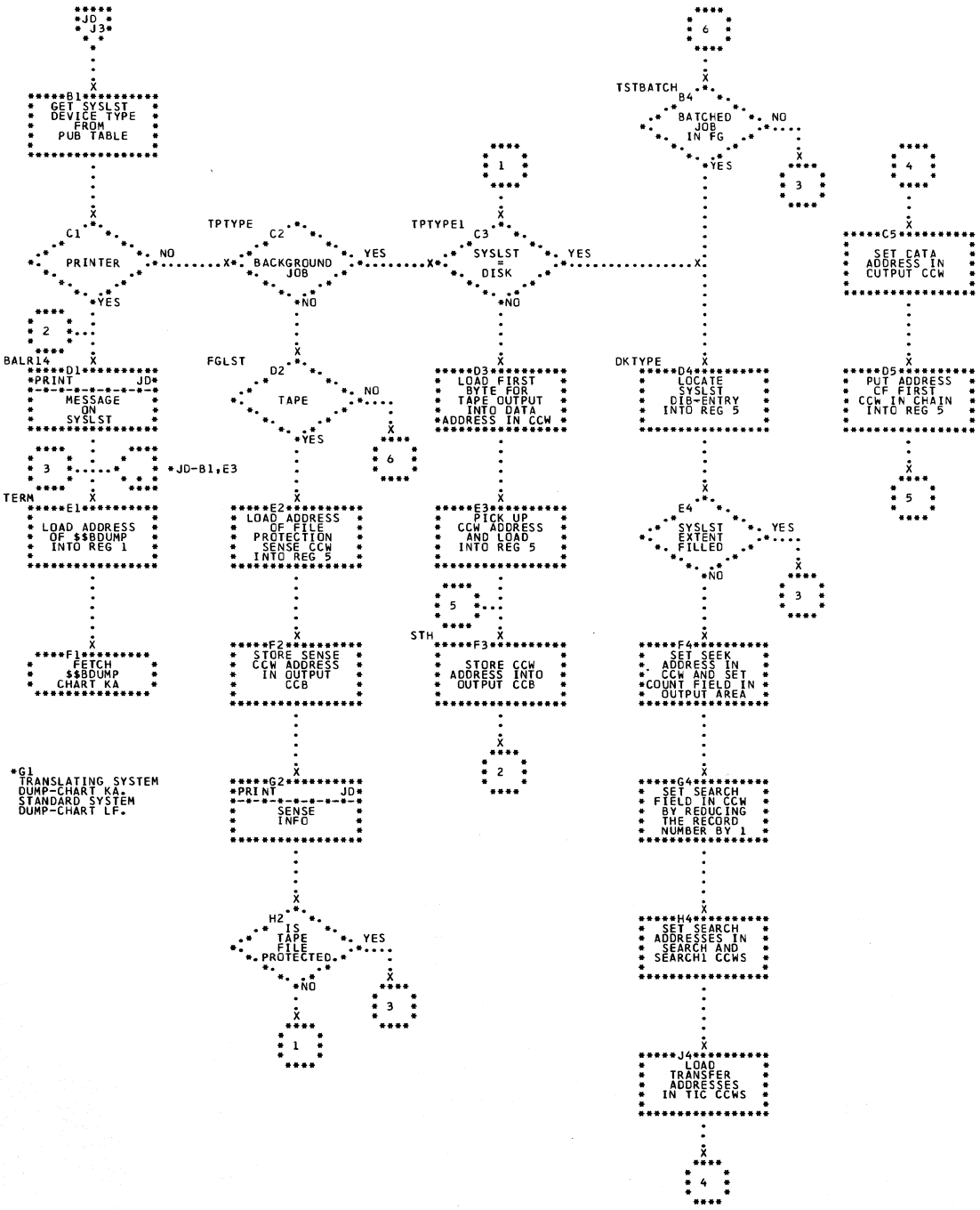


Chart JF. $$BPCHK - Prepare Information for Message about PC Cancel and Select I/O Device

Refer to Chart 11.

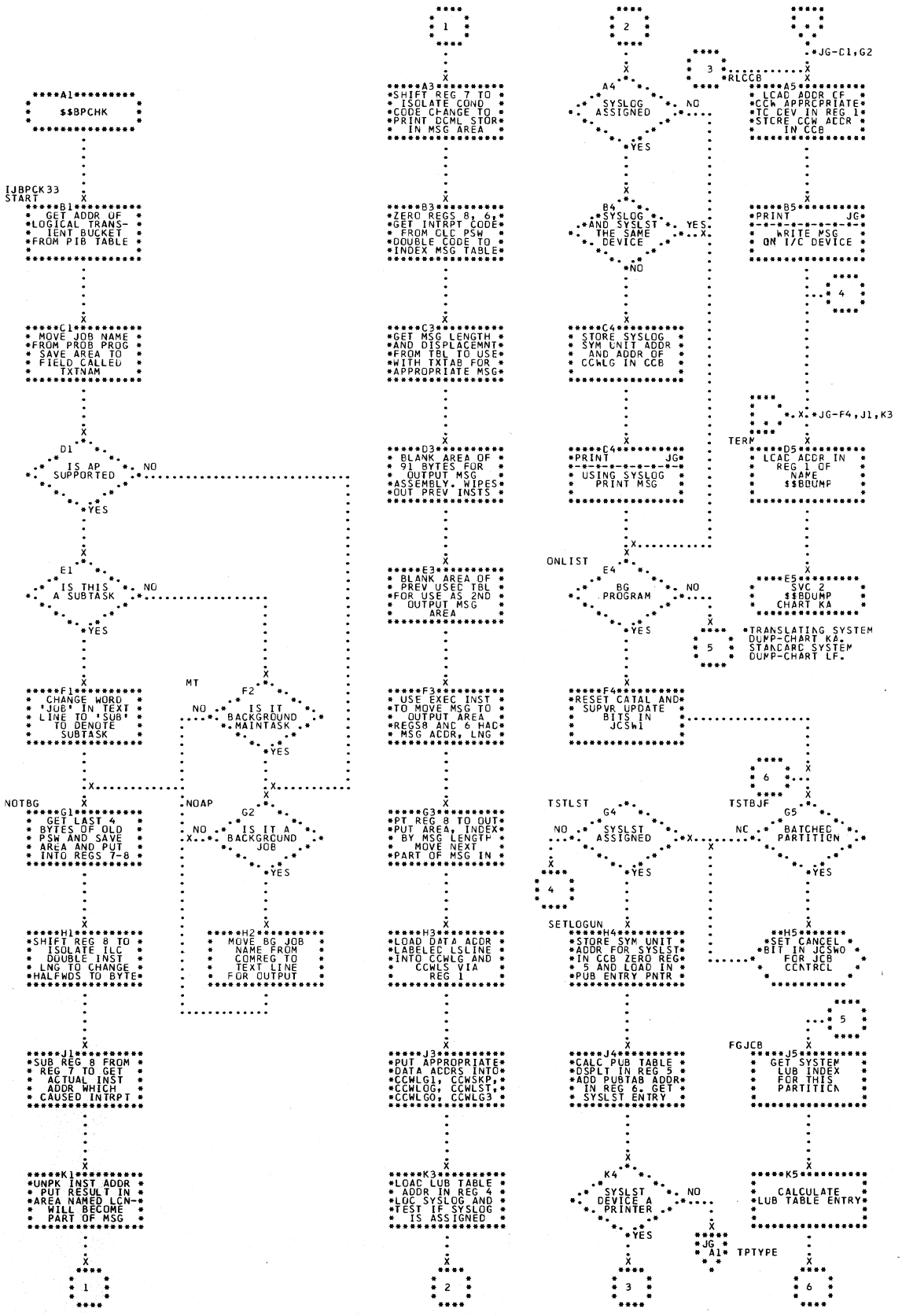


Chart JG. $$BPCHK - Set Up for I/O and Output the Message
Refer to Chart 11.

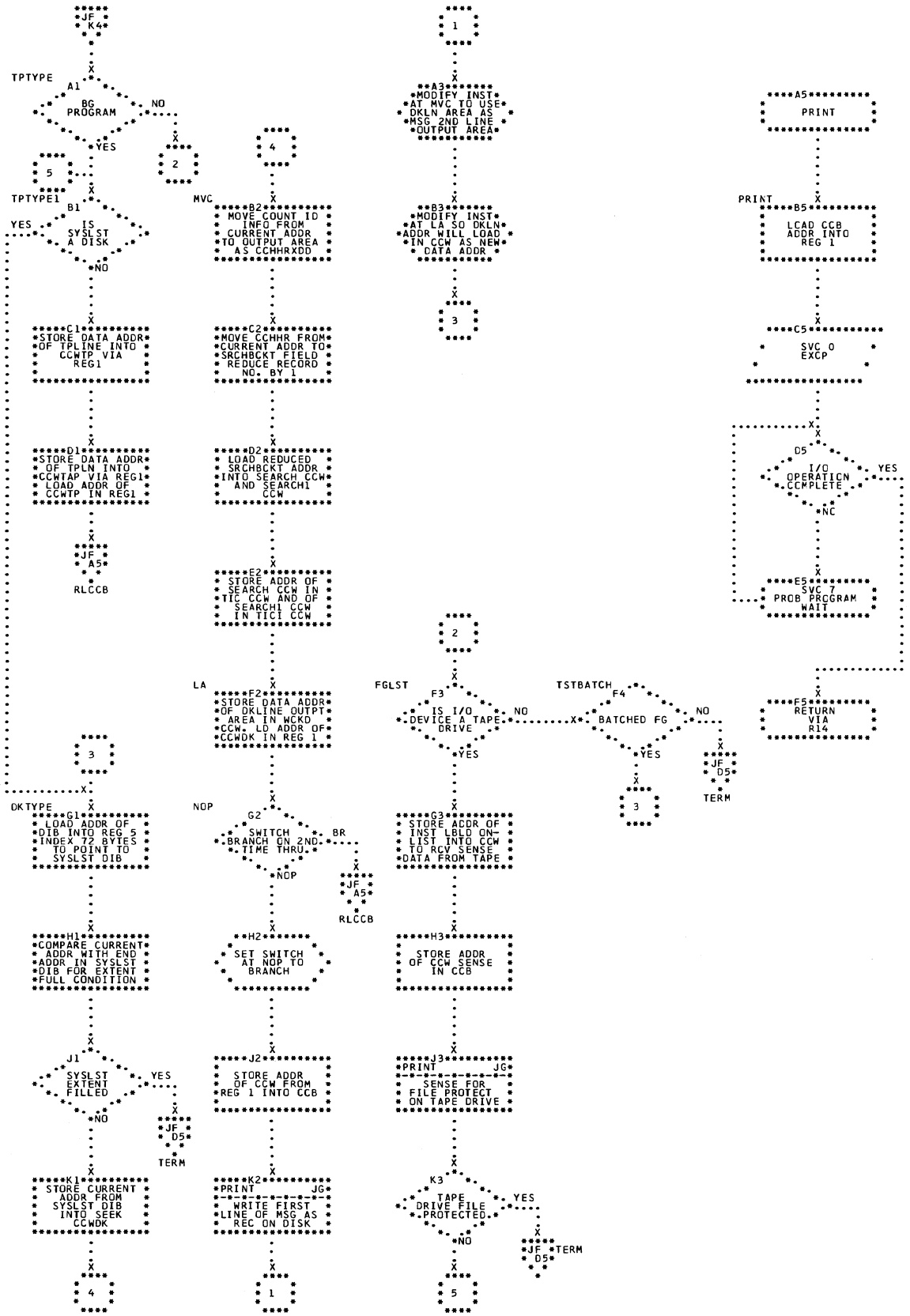


Chart KA. $$BDUMP - Translating System Dump, Monitor Background/Foreground Program Dump (Part 1 of 3)
Refer to Chart 10.

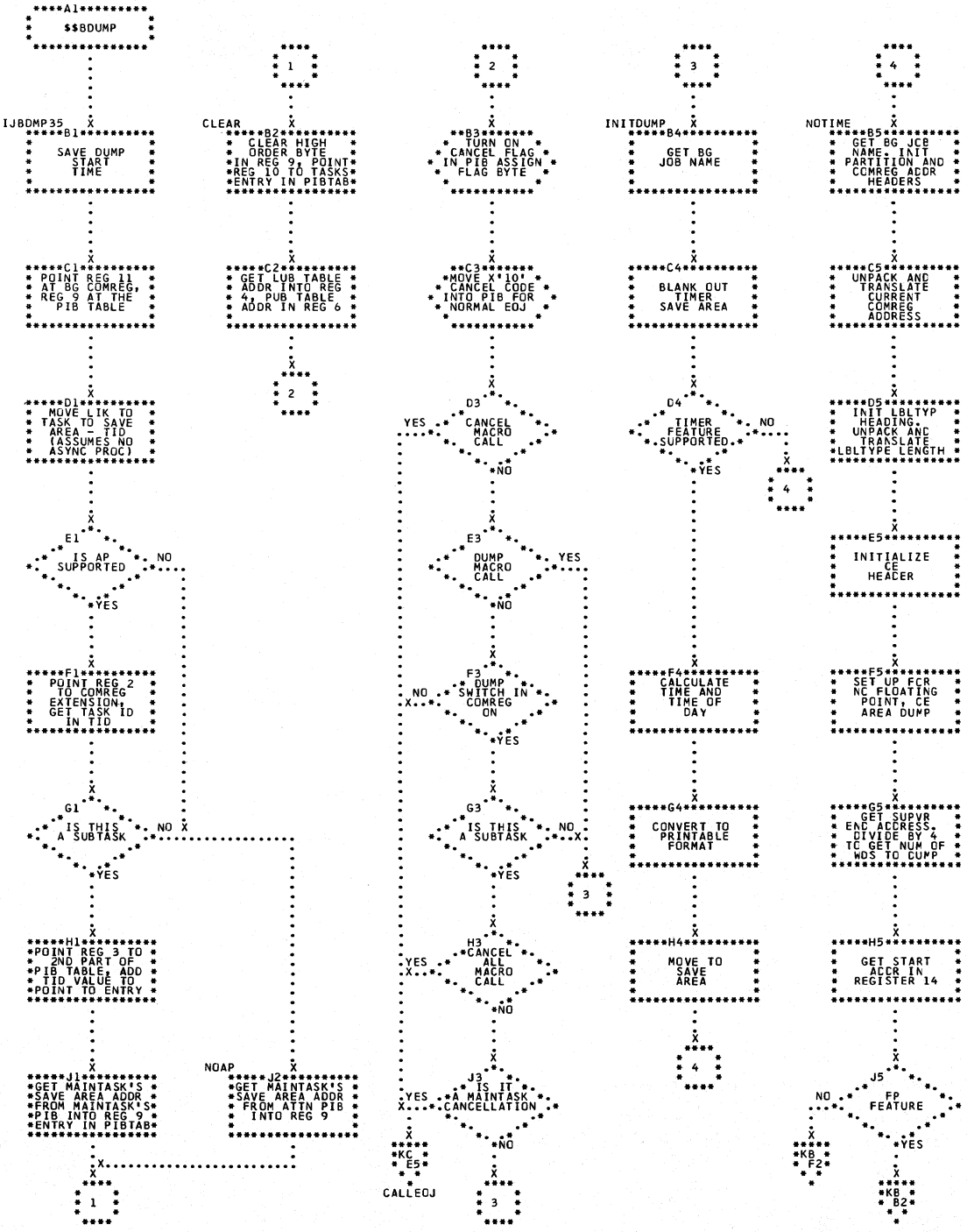


Chart KB. $$BDUMP - Translating System Dump, Monitor Background/Foreground Program Dump (Part 2 of 3)
Refer to Chart 10.

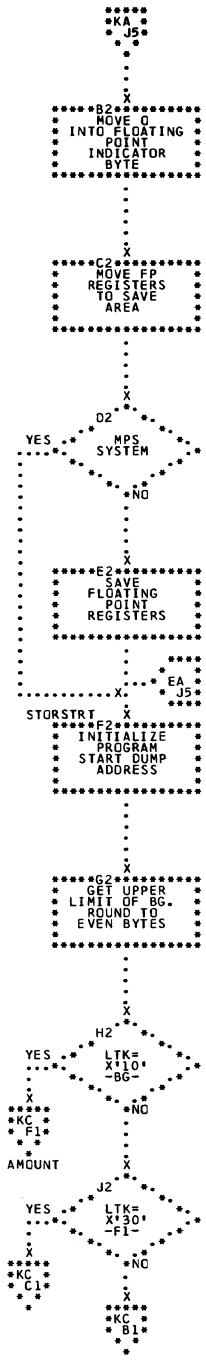


Chart KC. $$BDUMP - Translating System Dump, Monitor Background/Foreground Program Dump (Part 3 of 3)
Refer to Chart 10.

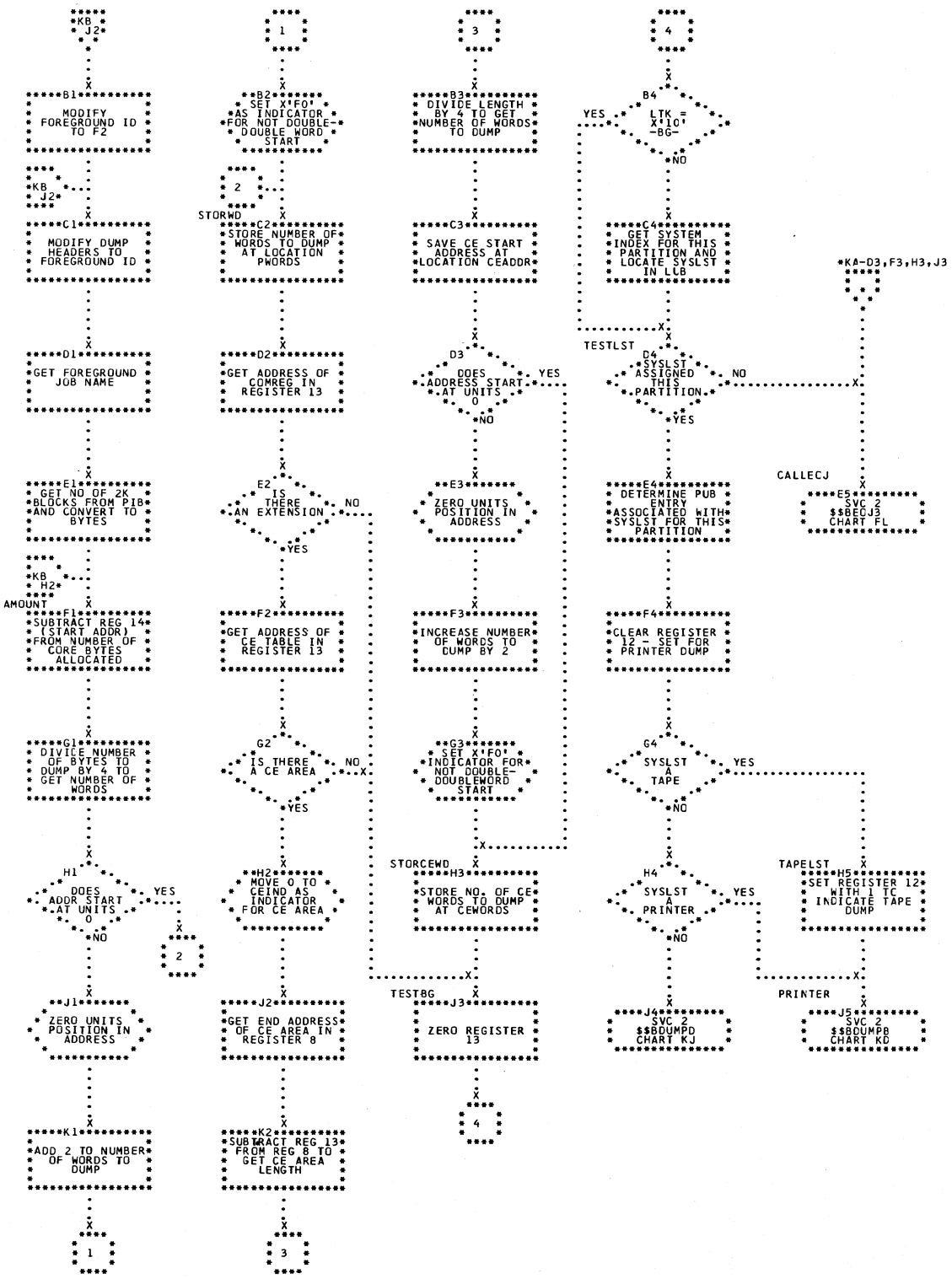


Chart KD. $$BDUMPB - Translating System Dump, Background/Foreground Dump on Printer or Tape (Part 1 of 2)
Refer to Chart 13.

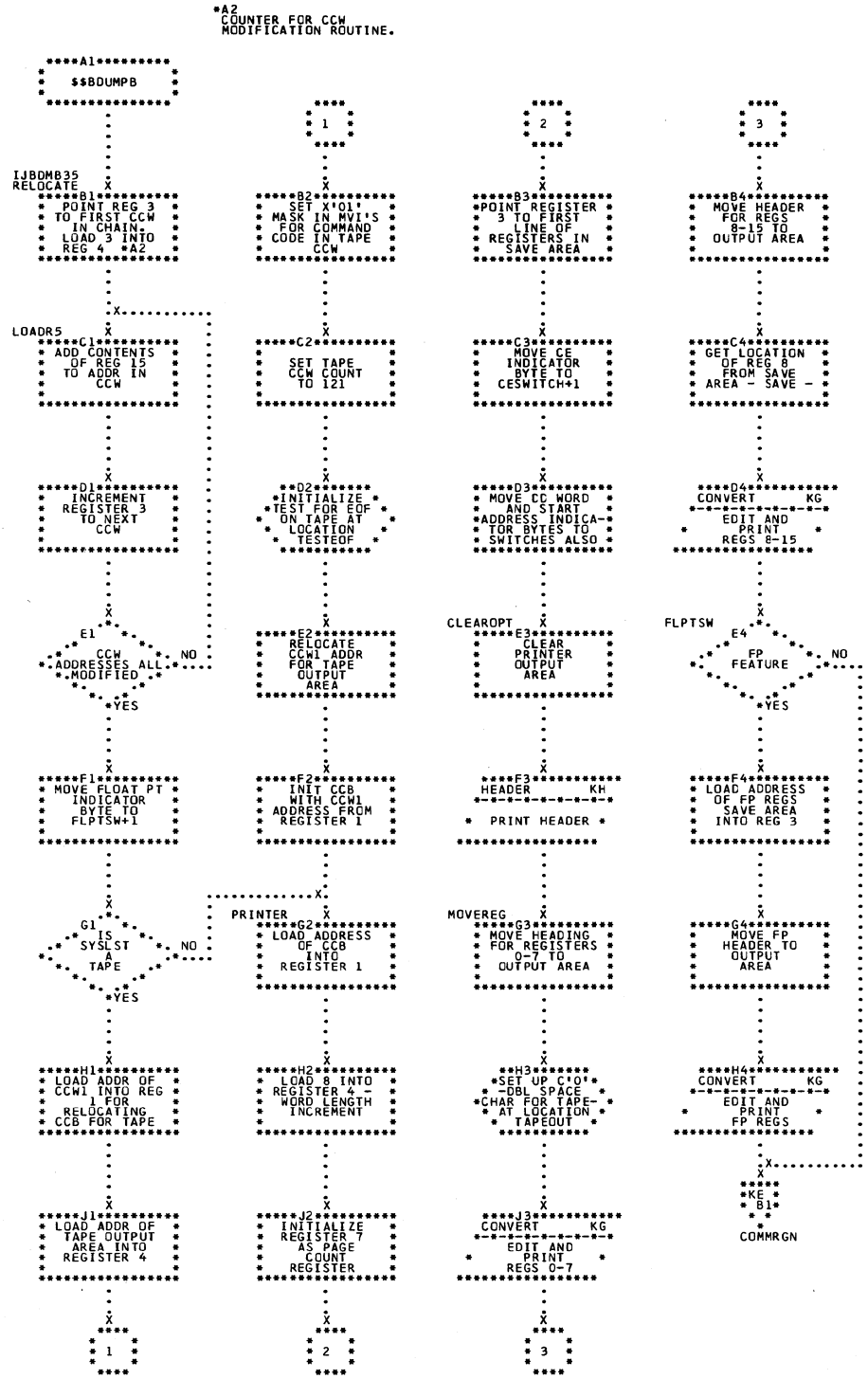


Chart KE. $$BDUMPB - Translating System Dump, Background/Foreground Dump on Printer or Tape (Part 2 of 2)
Refer to Chart 13.

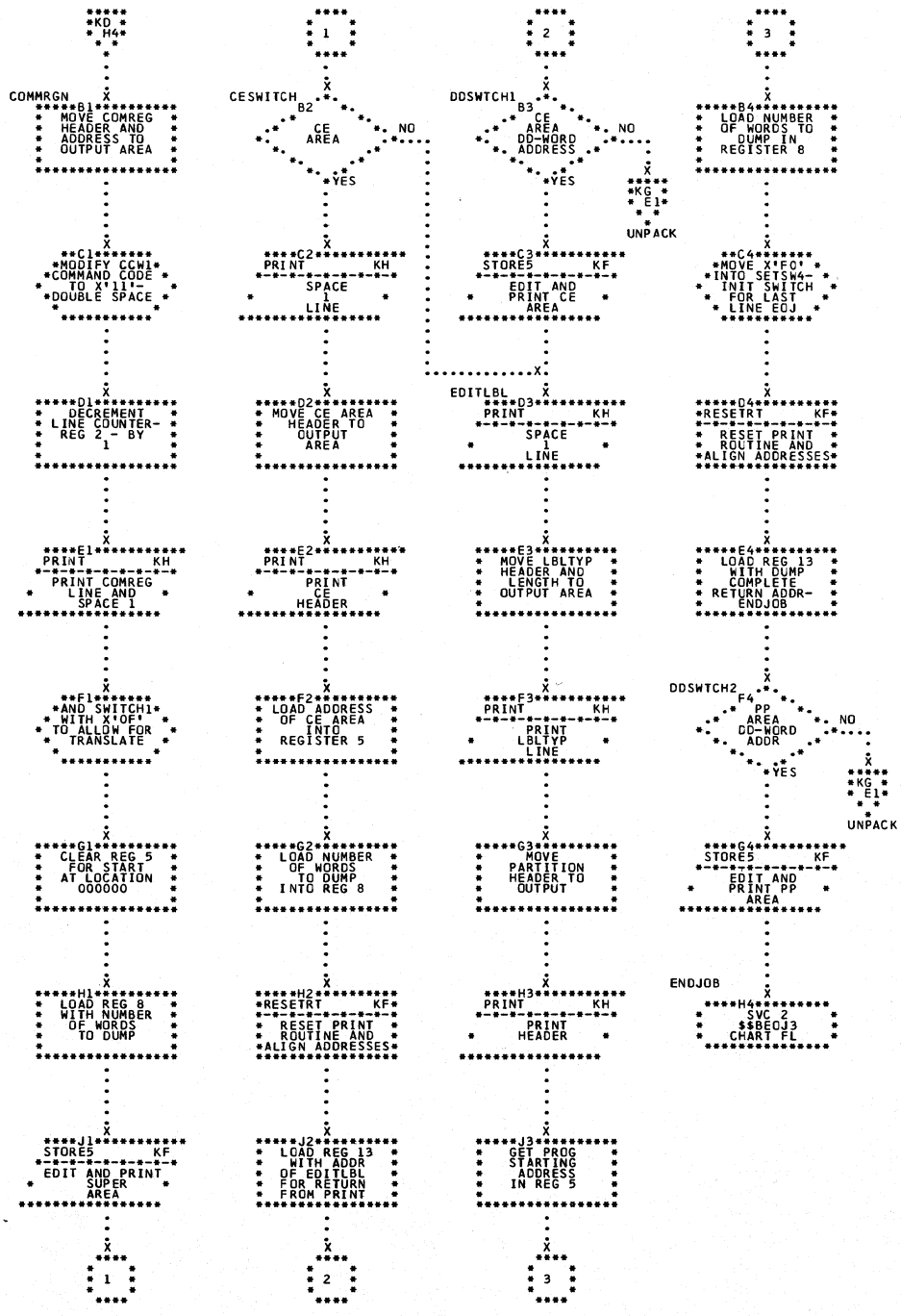


Chart KF. $$BDUMPB - Translating System Dump, Reset Storage Print Routine and Edit a Line Subroutines
Refer to Chart 13.

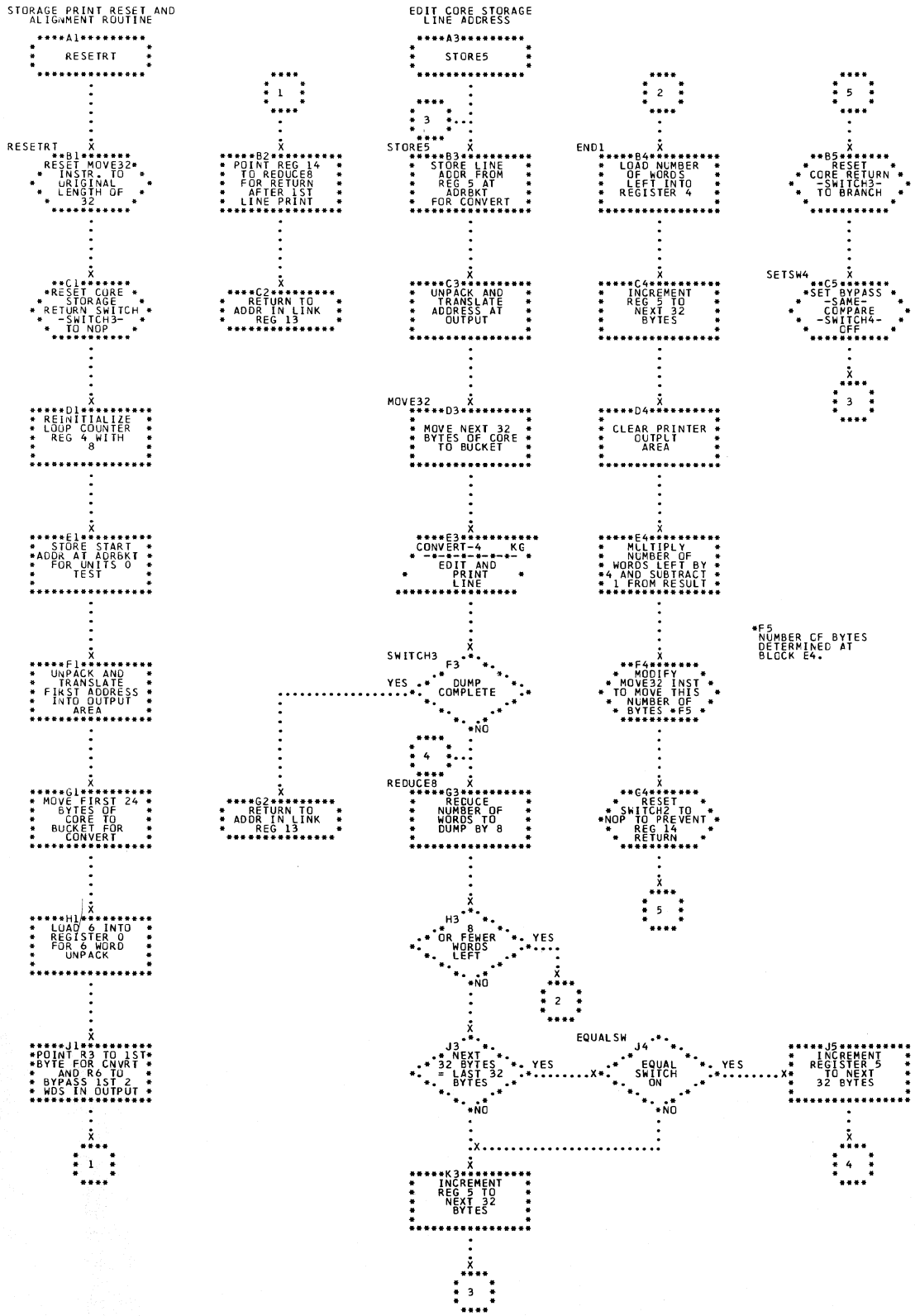


Chart KG. $$BDUMPB - Translating System Dump, Subroutine to Edit and Print a Line
Refer to Chart 13.

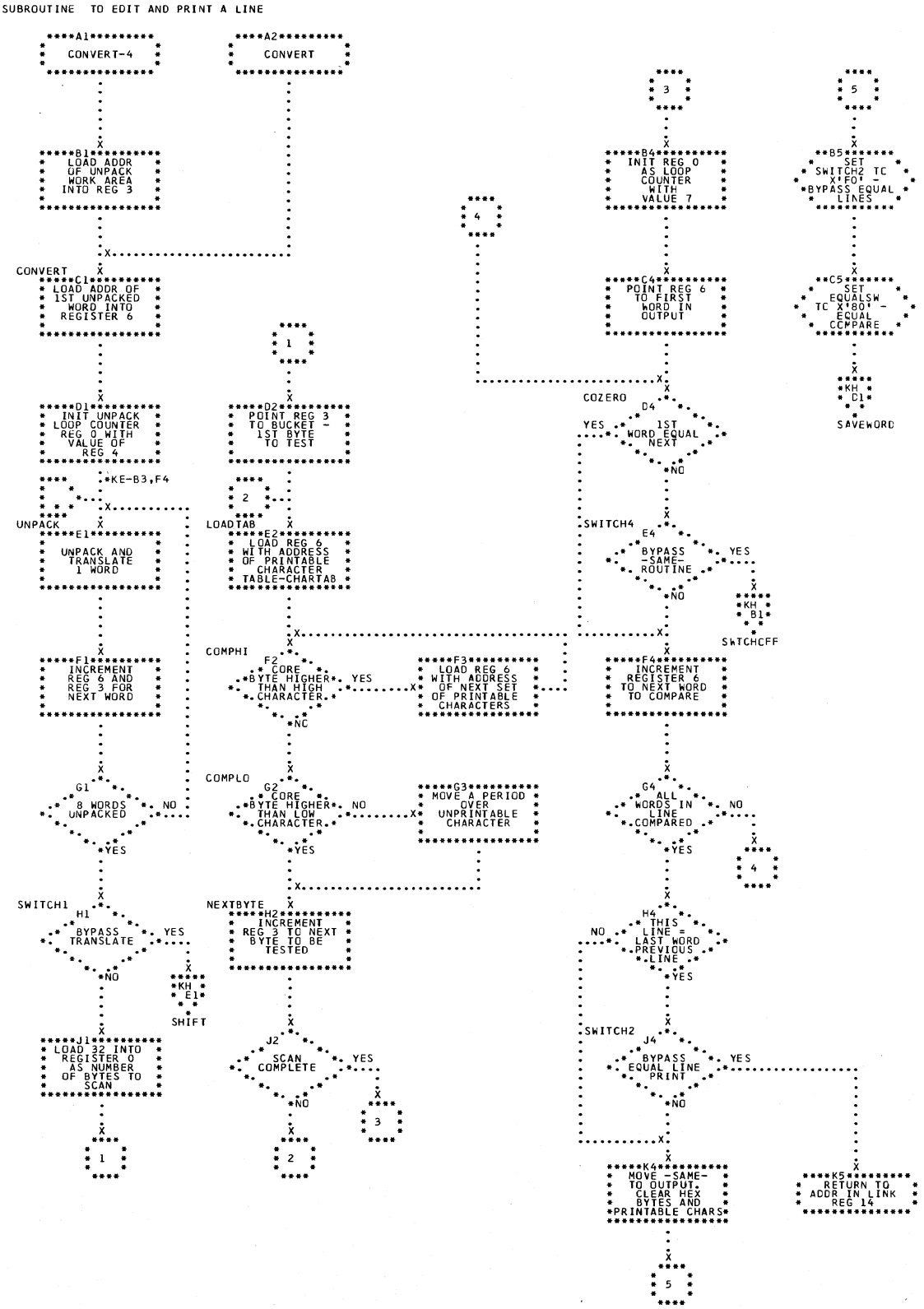


Chart KH. $$BDUMPB - Translating System Dump, Edit and Print a Line and Prepare Page Headings Subroutines
Refer to Chart 13.

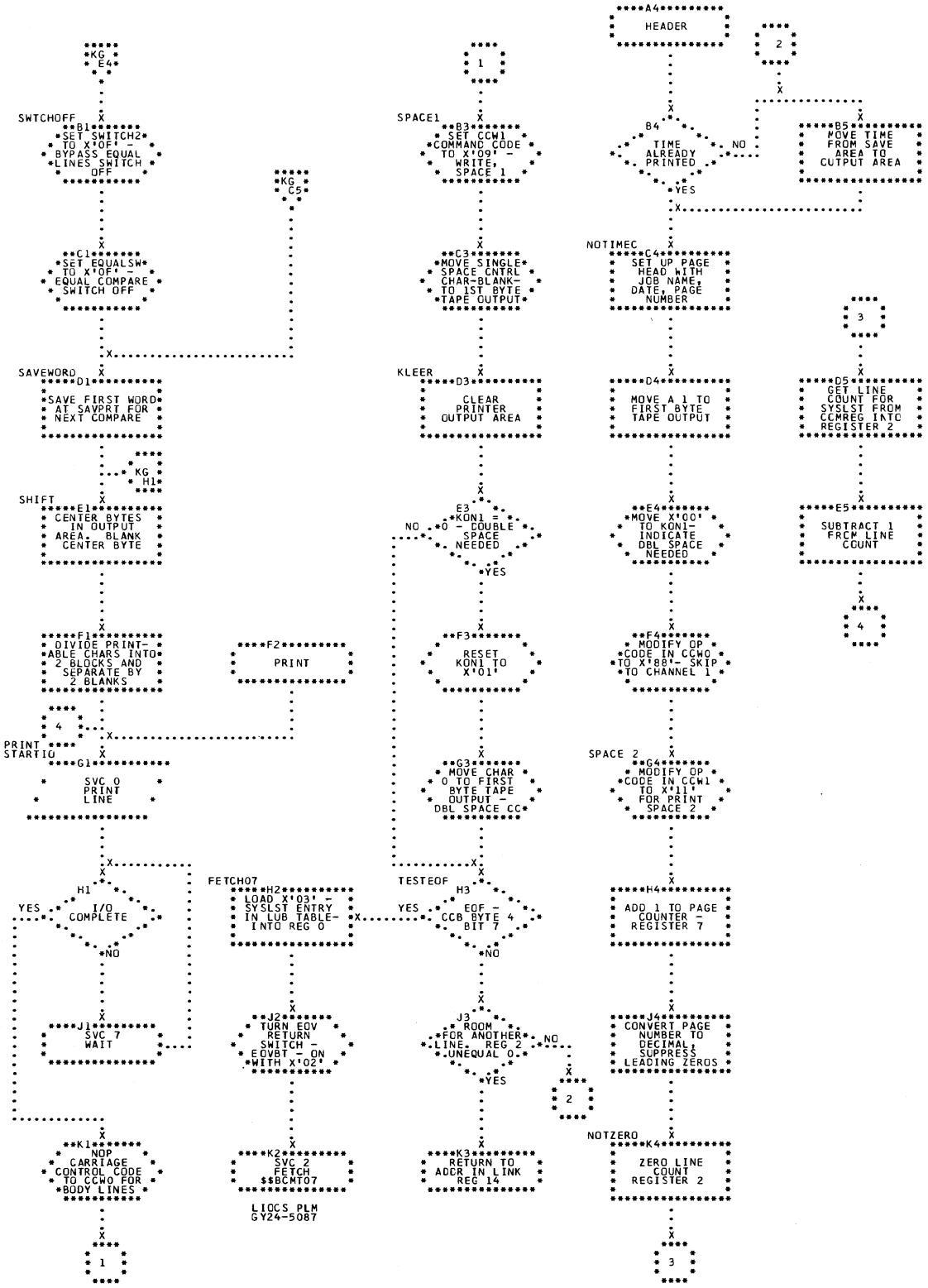


Chart KJ. $$BDUMPD - Translating System Dump, Background/Foreground Dump on Disk (Part 1 of 2)
Refer to Chart 13.

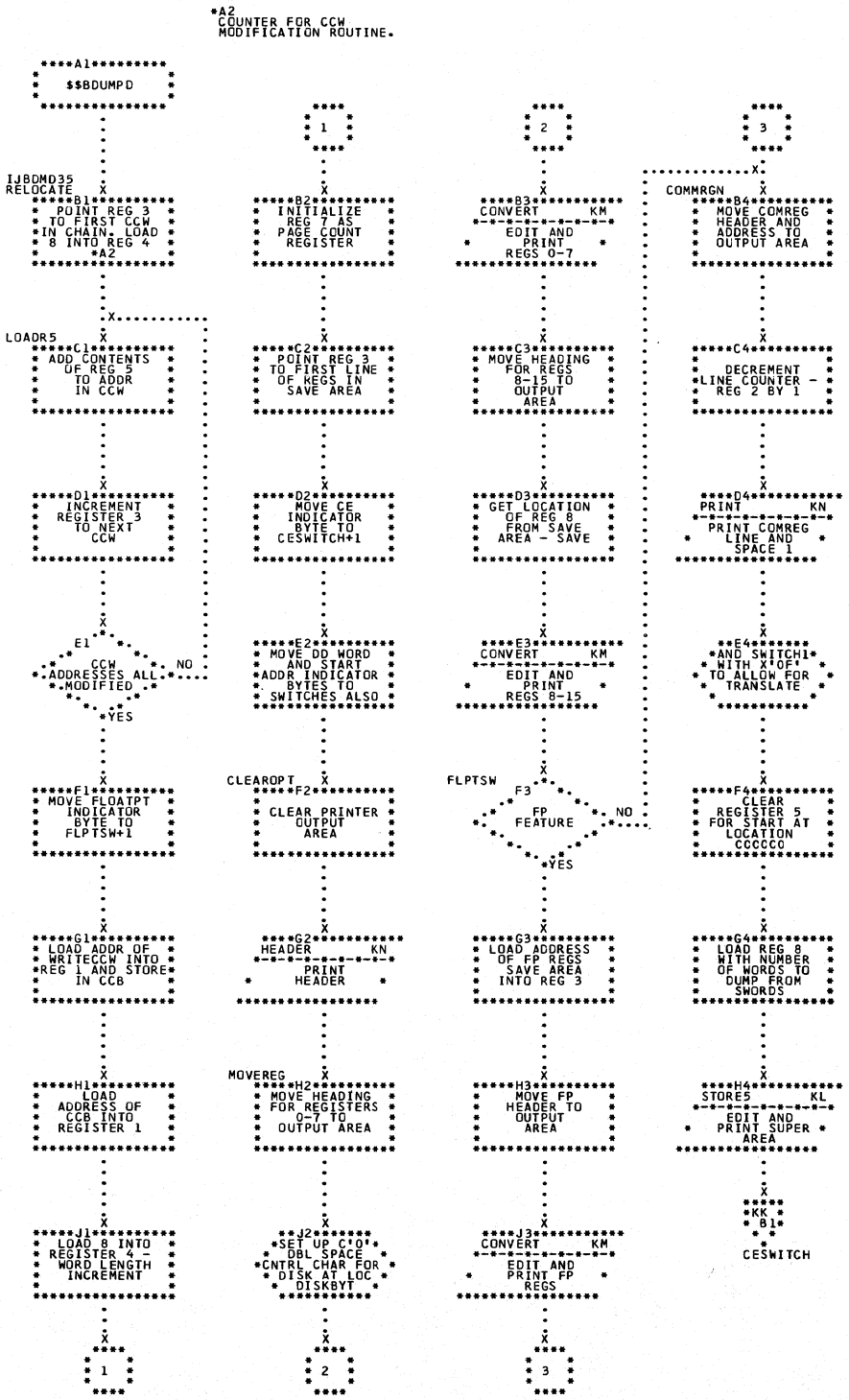


Chart KK. $$BDUMPD - Translating System Dump, Background/Foreground Dump on Disk (Part 2 of 2)
Refer to Chart 13.

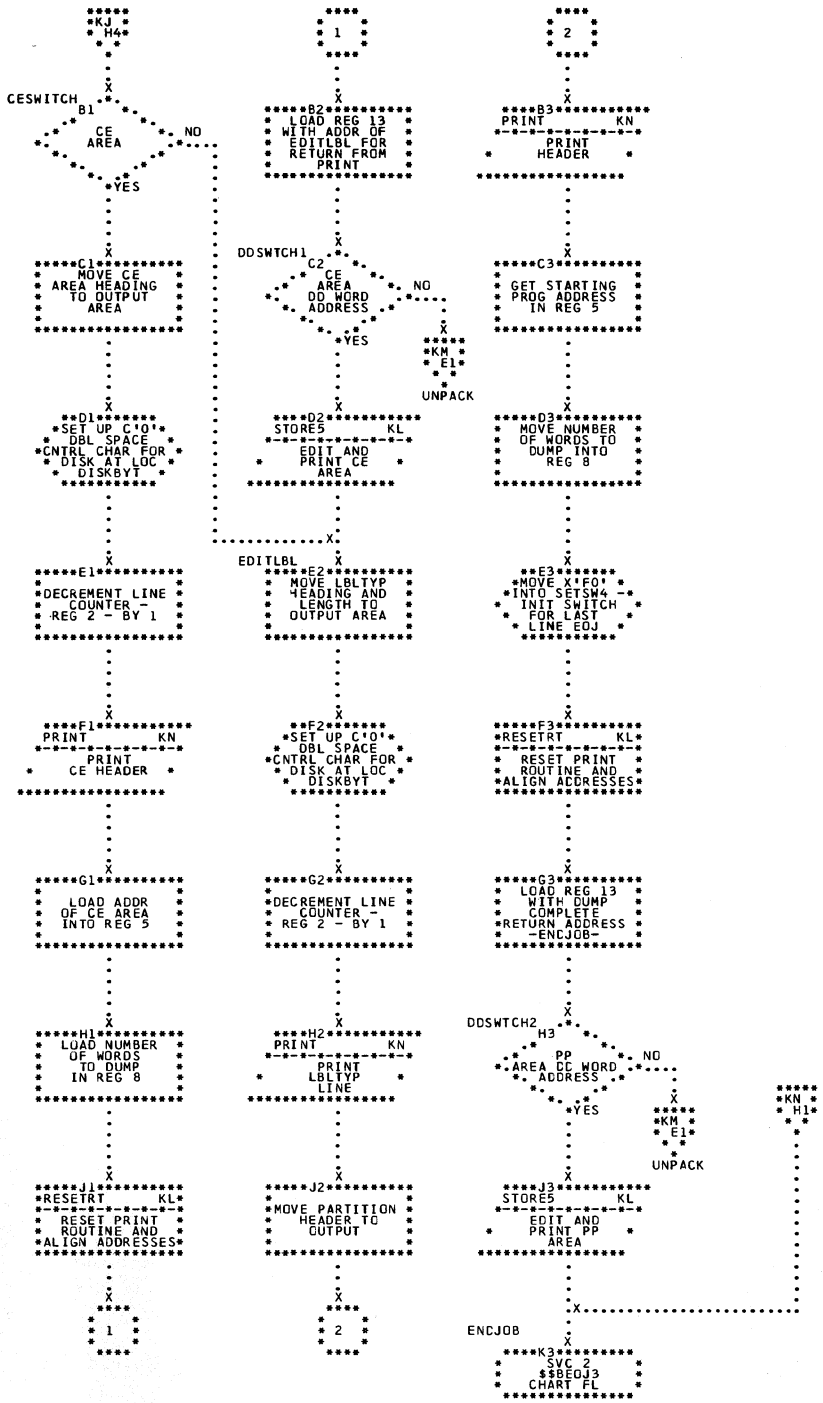


Chart KL. $$BDUMPD - Translating System Dump, Reset Storage Print Routine and Edit a Line Subroutines
Refer to Chart 13.

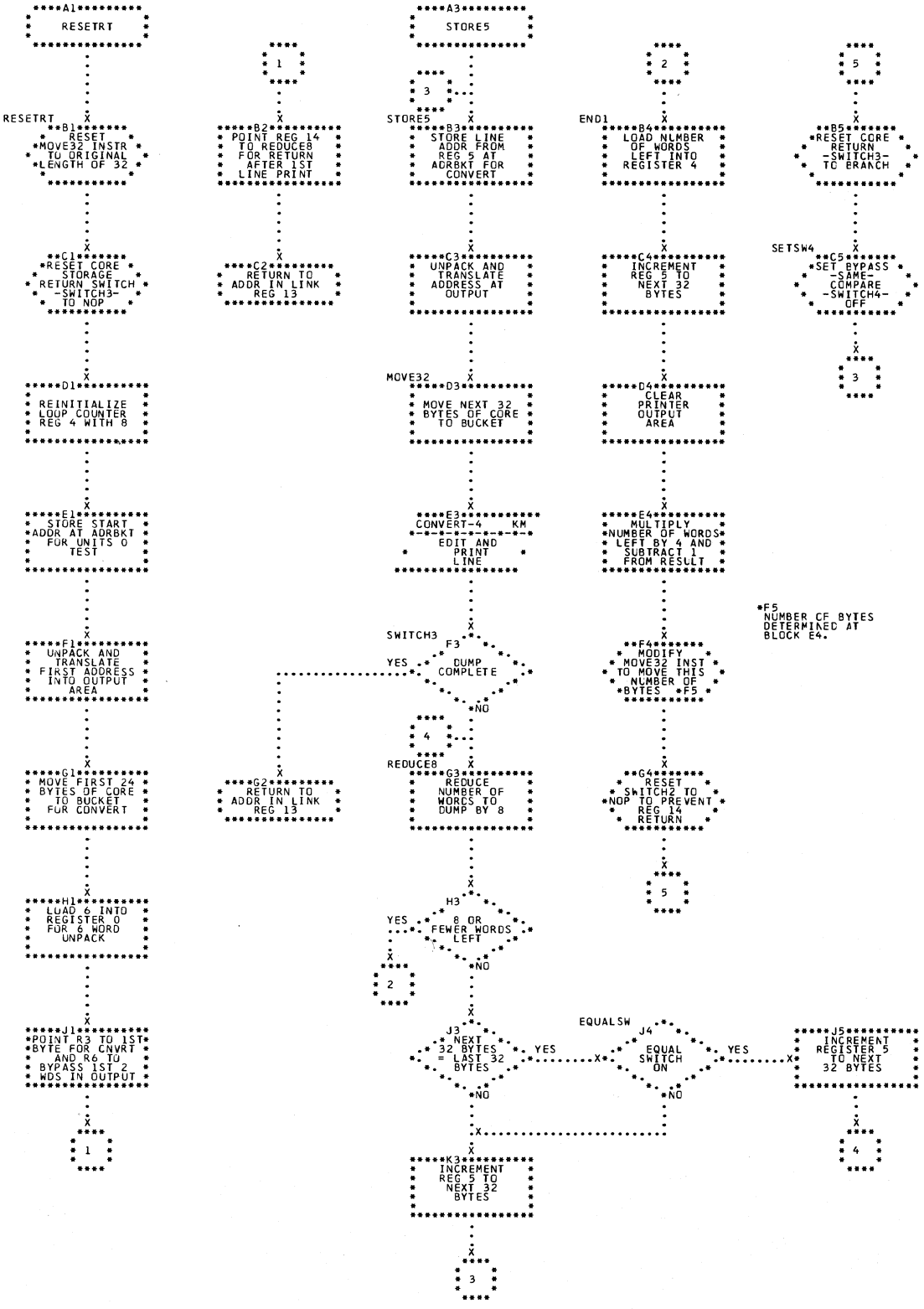


Chart KM. $$BDUMPD - Translating System Dump, Subroutines to Edit and Print a Line
Refer to Chart 13.

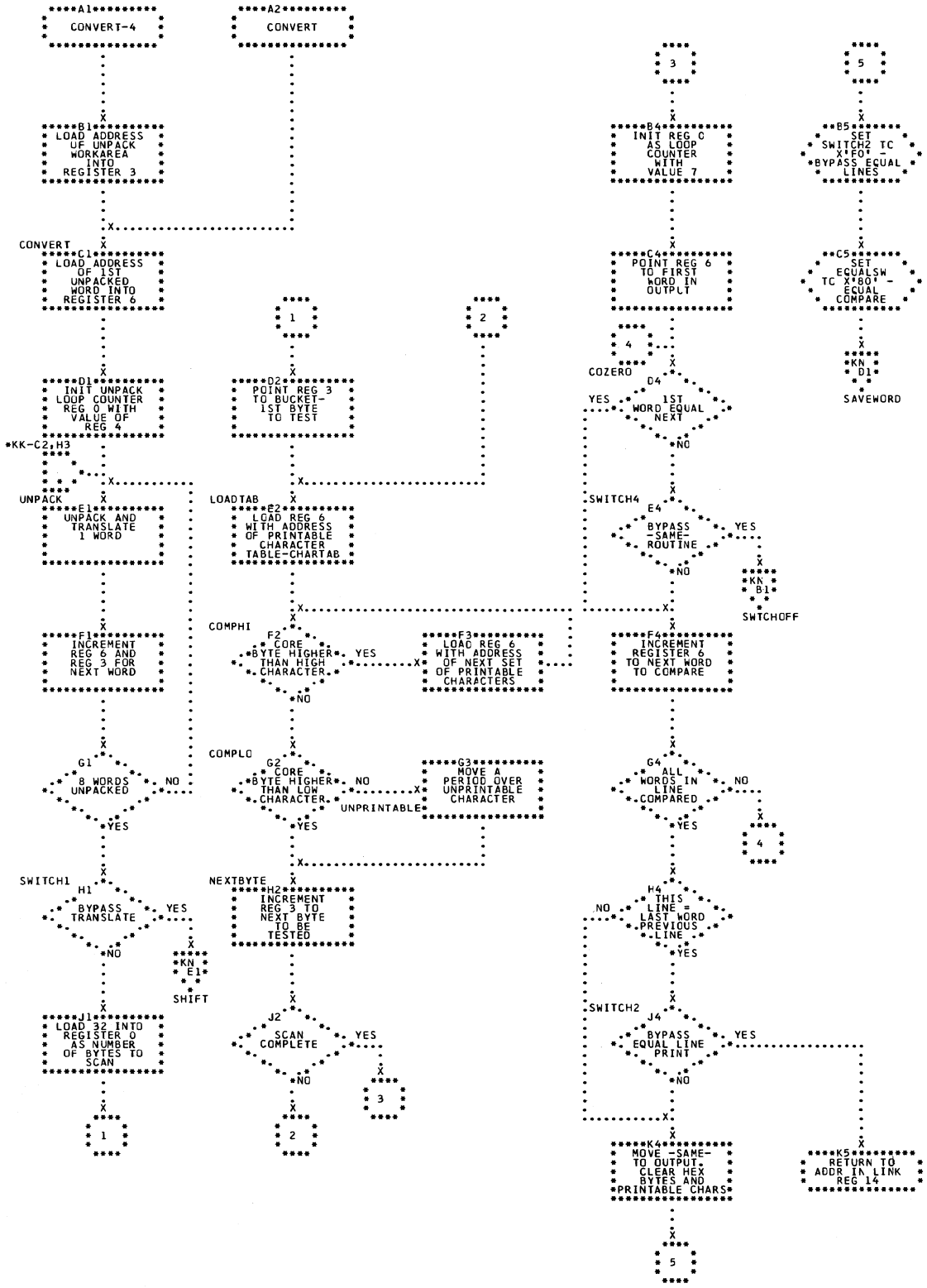


Chart KN. $$BDUMPD - Translating System Dump, Edit and Write a Line and Prepare Page Headings Subroutines
Refer to Chart 13.

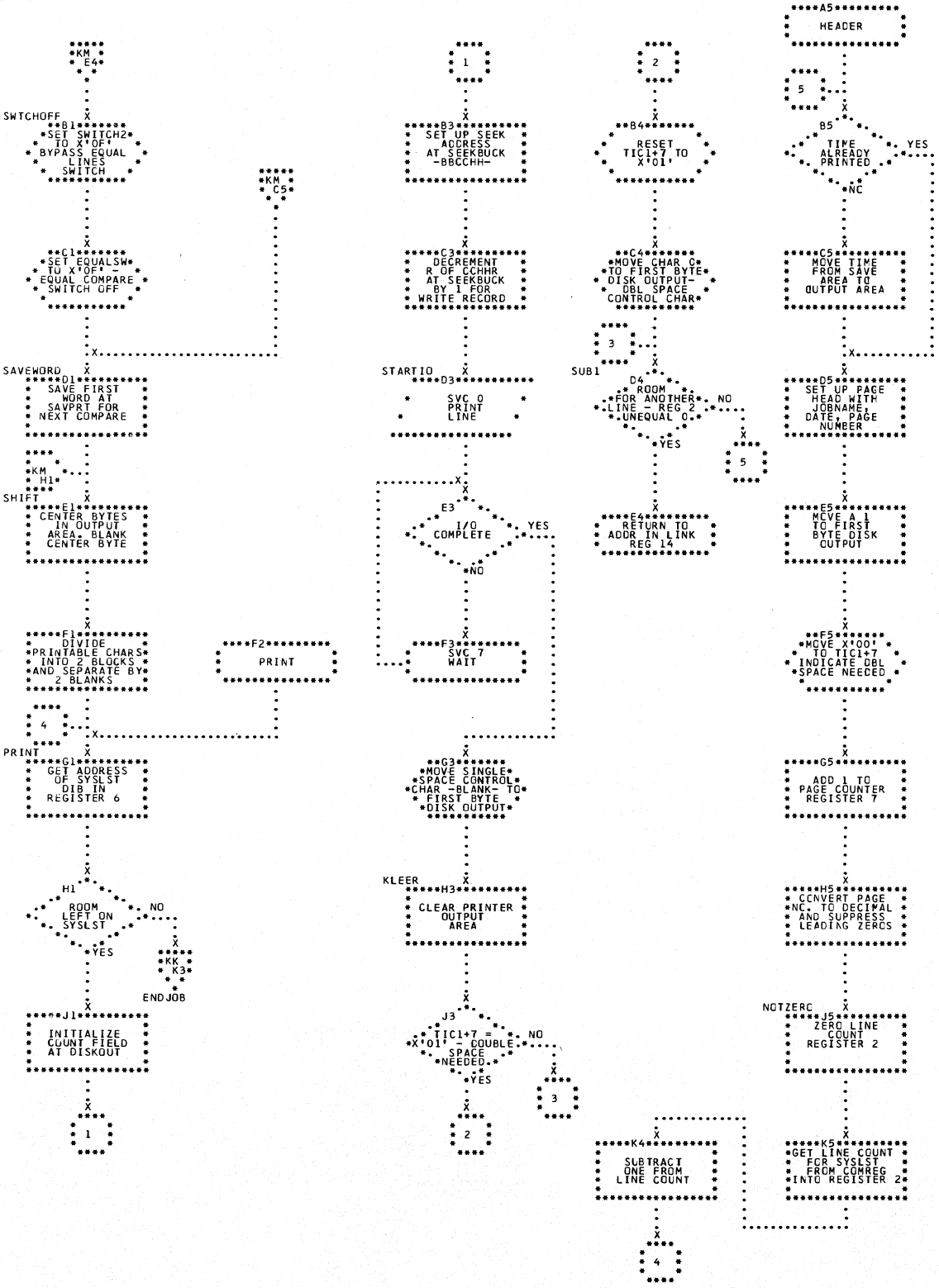


Chart LA. $$BPDUMP - Translating System Dump, Background/Foreground Parameter Dump (Part 1 of 3)
Refer to Chart 13.

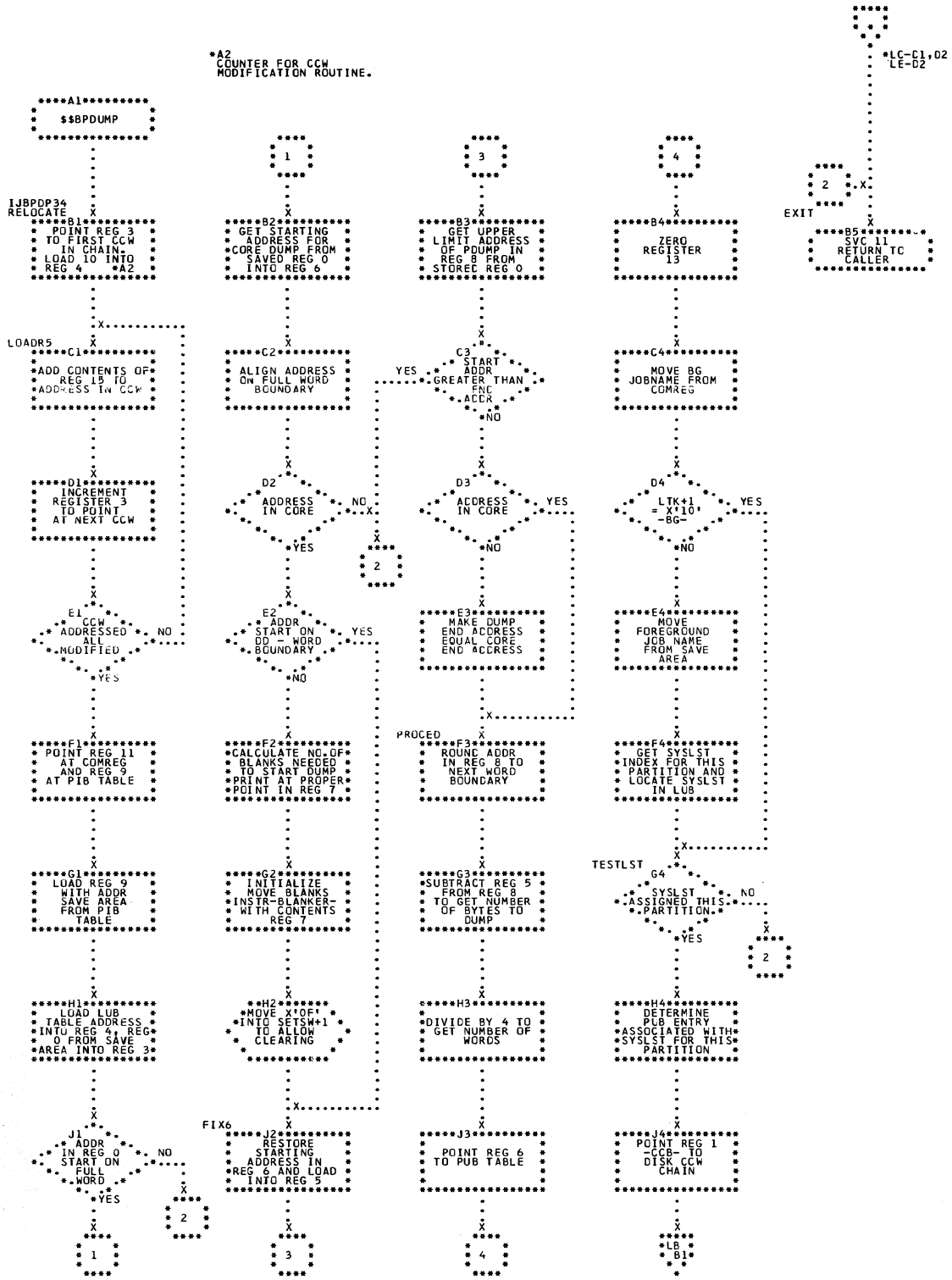


Chart LB. $$BPDUMP - Translating System Dump, Background/Foreground Parameter Dump (Part 2 of 3)
Refer to Chart 13.

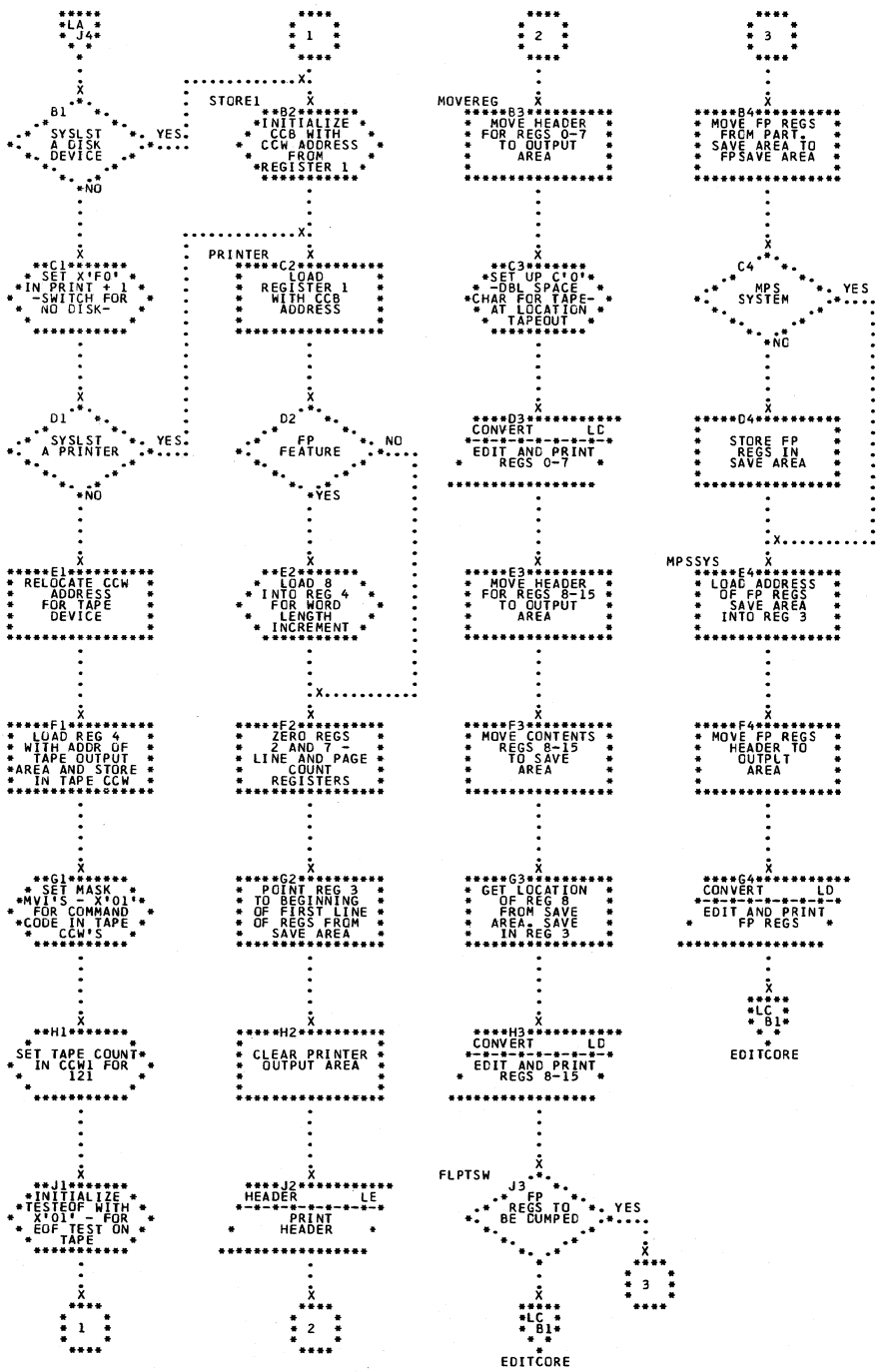


Chart LC. $$BPDUMP - Translating System Dump, Background/Foreground Parameter Dump (Part 3 of 3)
Refer to Chart 13.

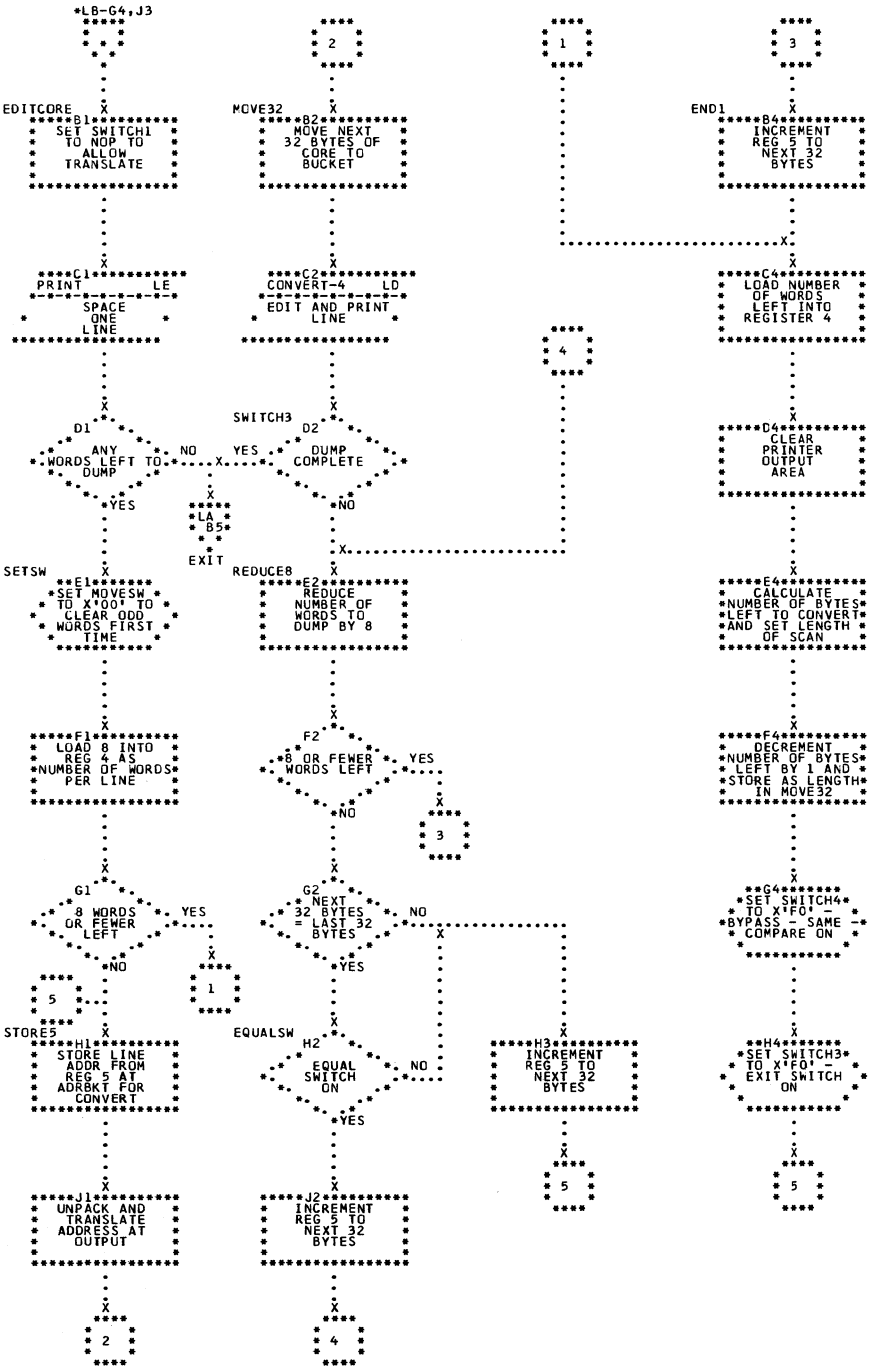


Chart LD. $$BPDUMP - Translating System Dump, Subroutine to Edit and Print a Line
Refer to Chart 13.

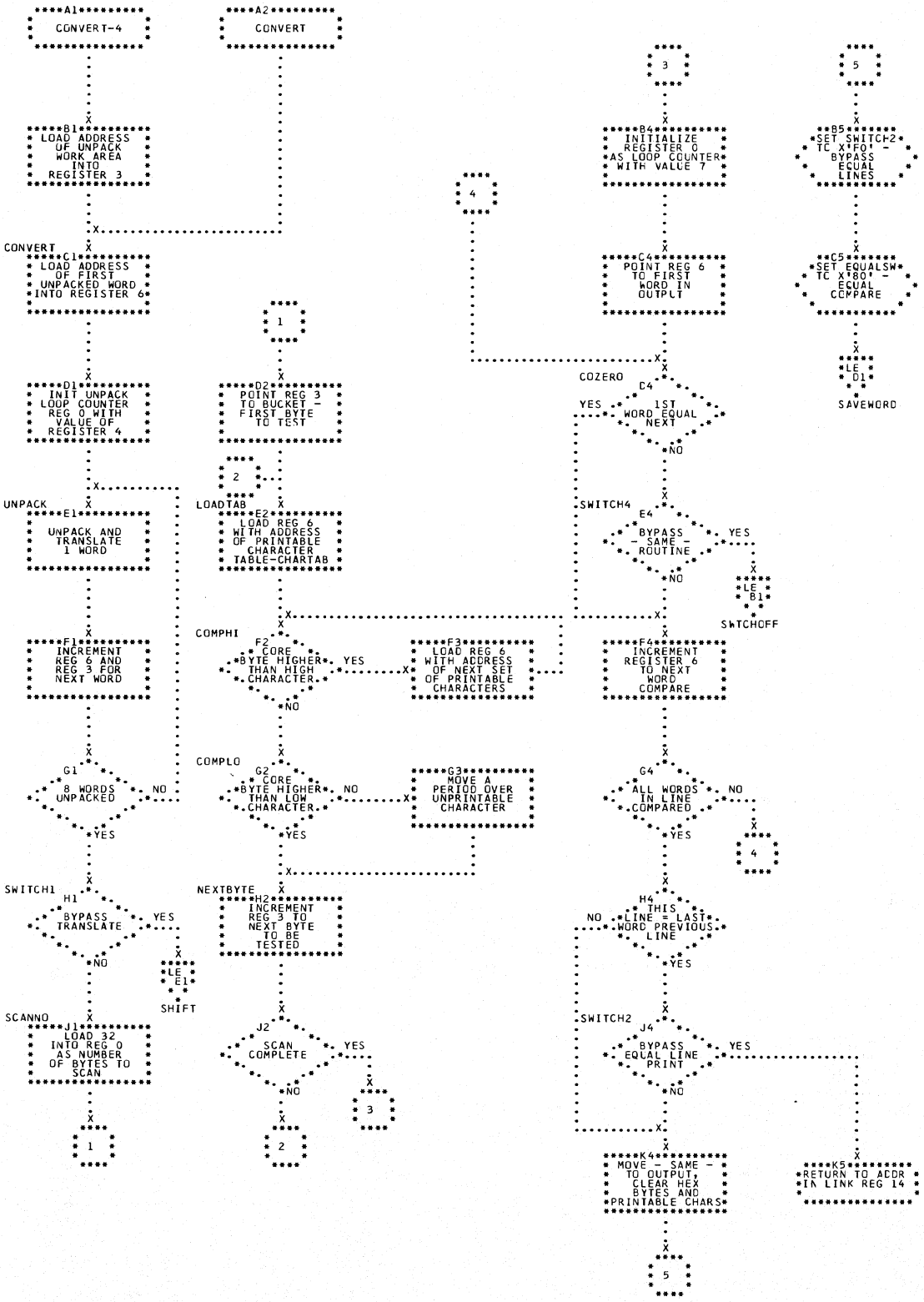


Chart LE. $$BPDUMP - Translating System Dump, Edit and Print a Line and Prepare Page Headings Subroutines
Refer to Chart 13.

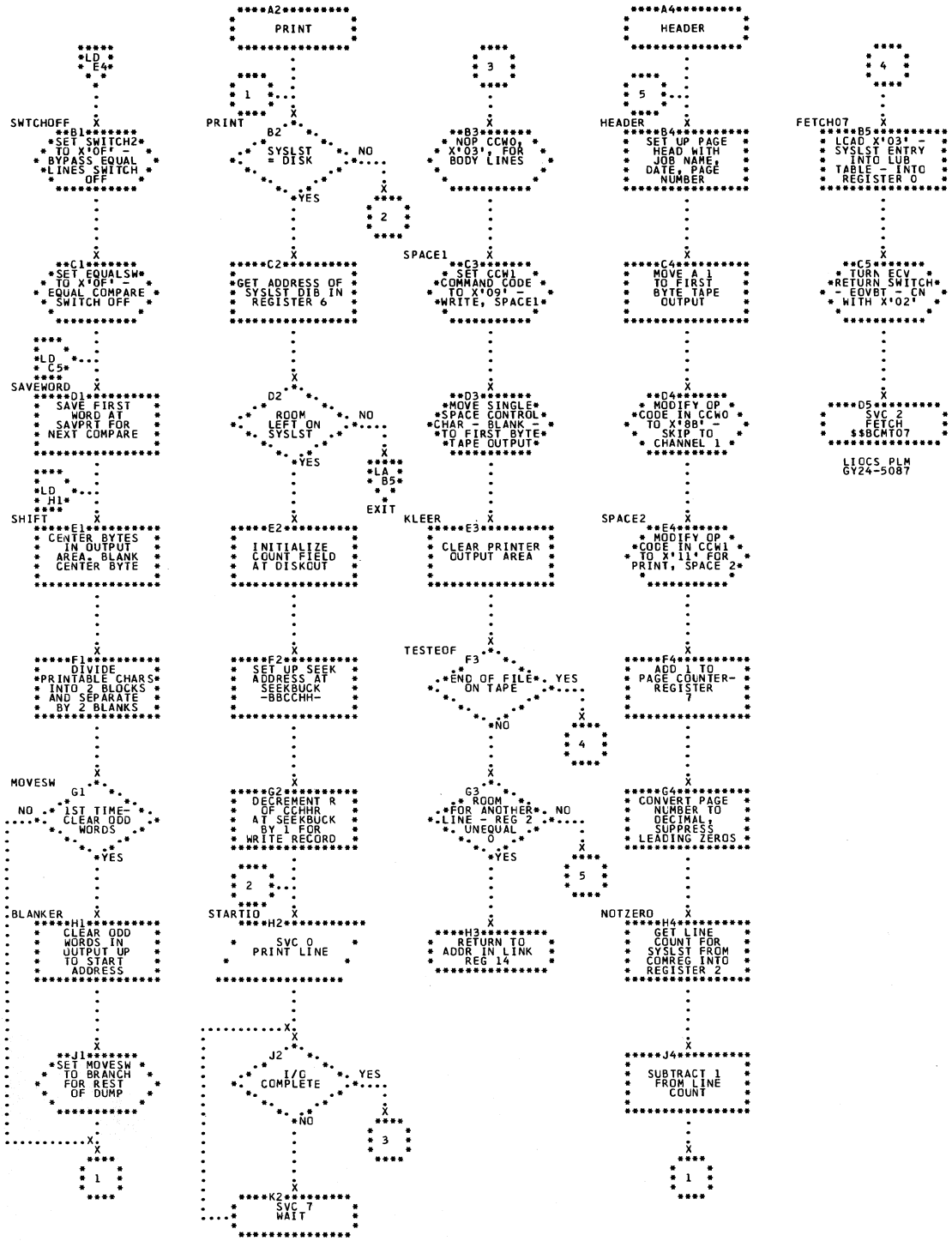


Chart LF. $$BDUMP - Standard System Dump, Monitor Background Program Dump
Refer to Chart 12.

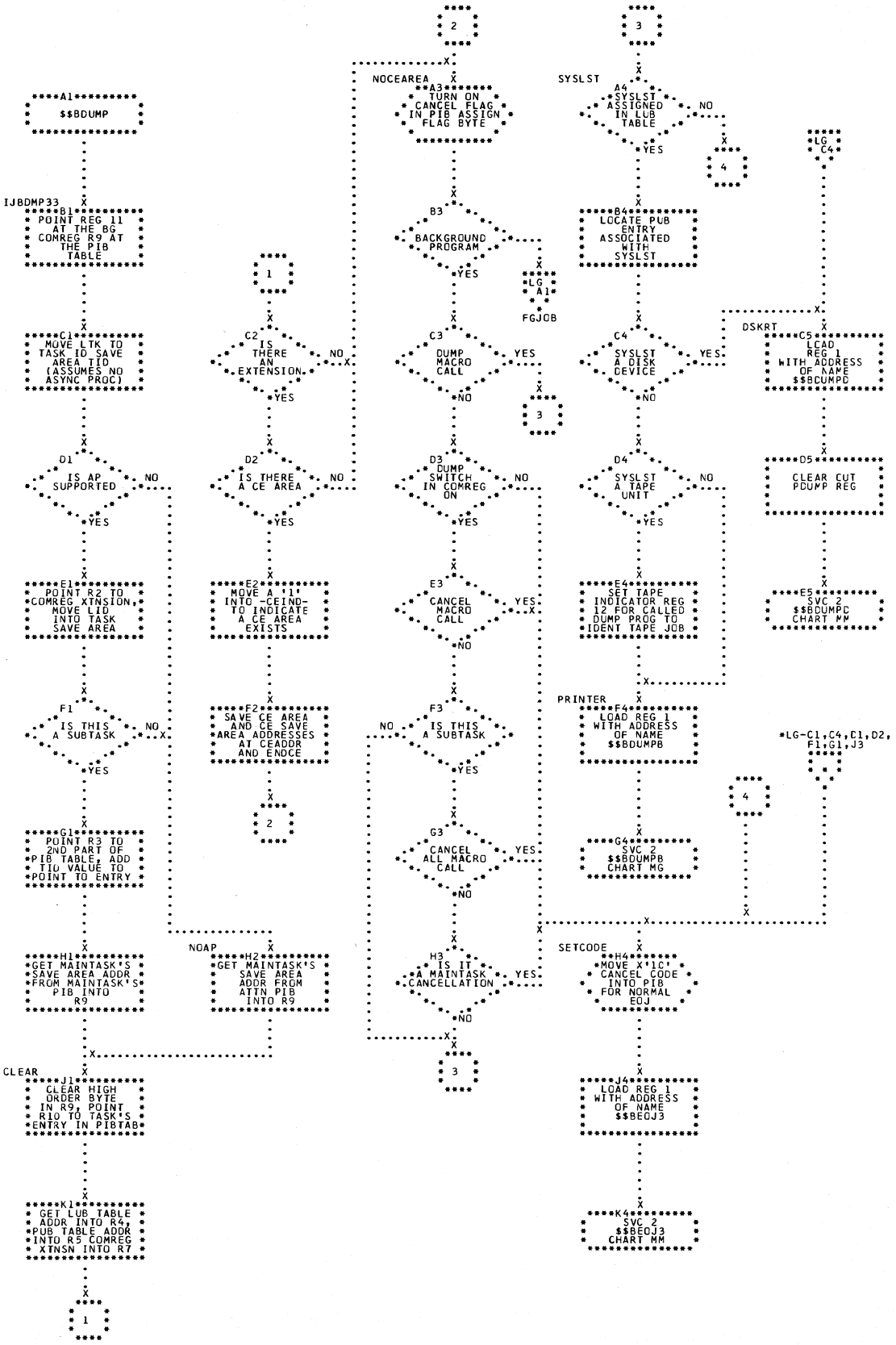


Chart LG. $$BDUMP - Standard System Dump, Monitor Foreground Program Dump
Refer to Chart 12.

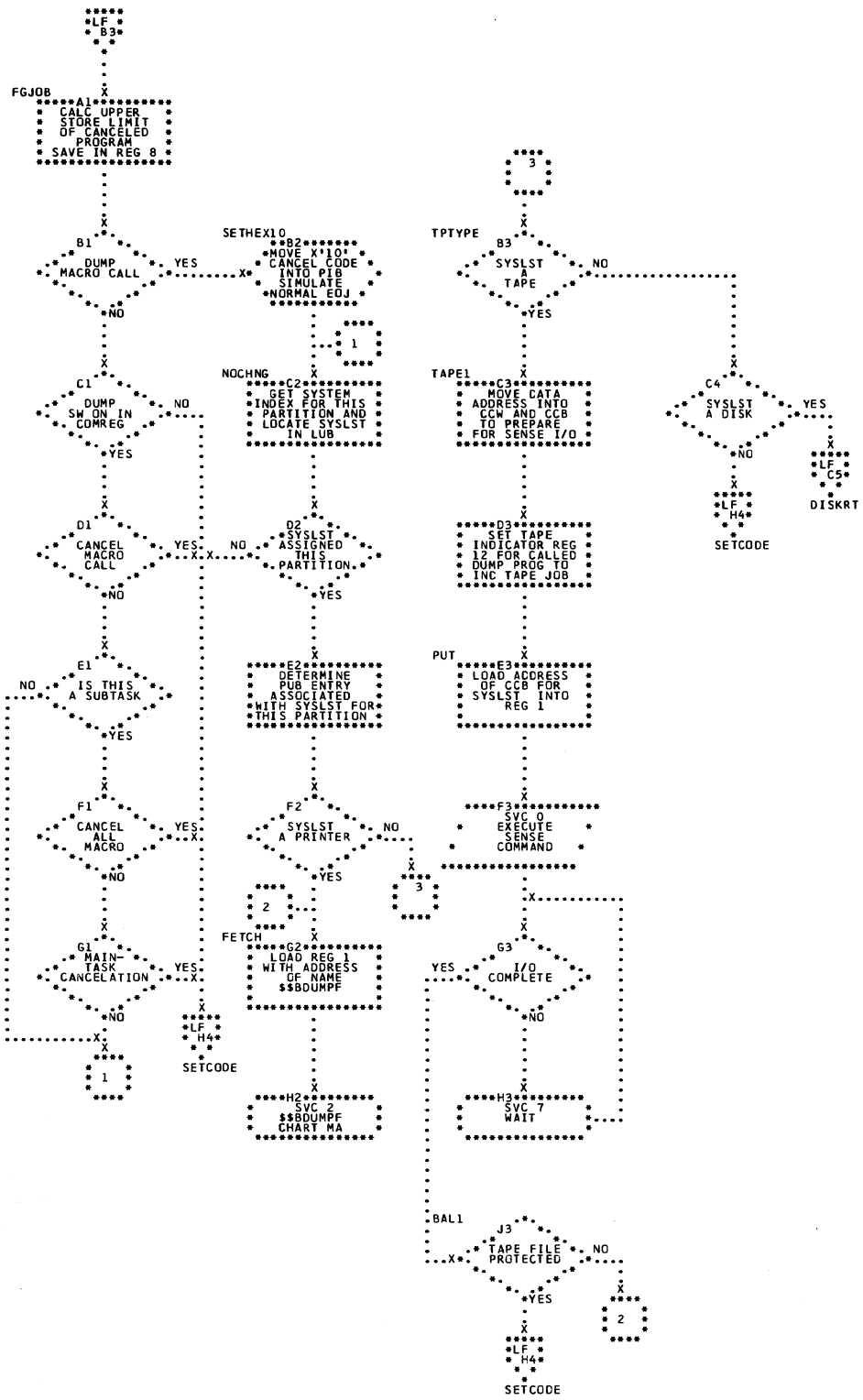


Chart MA. $$BDUMPF - Standard System Dump, Foreground Program Dump (Part 1 of 2)
Refer to Chart 14.

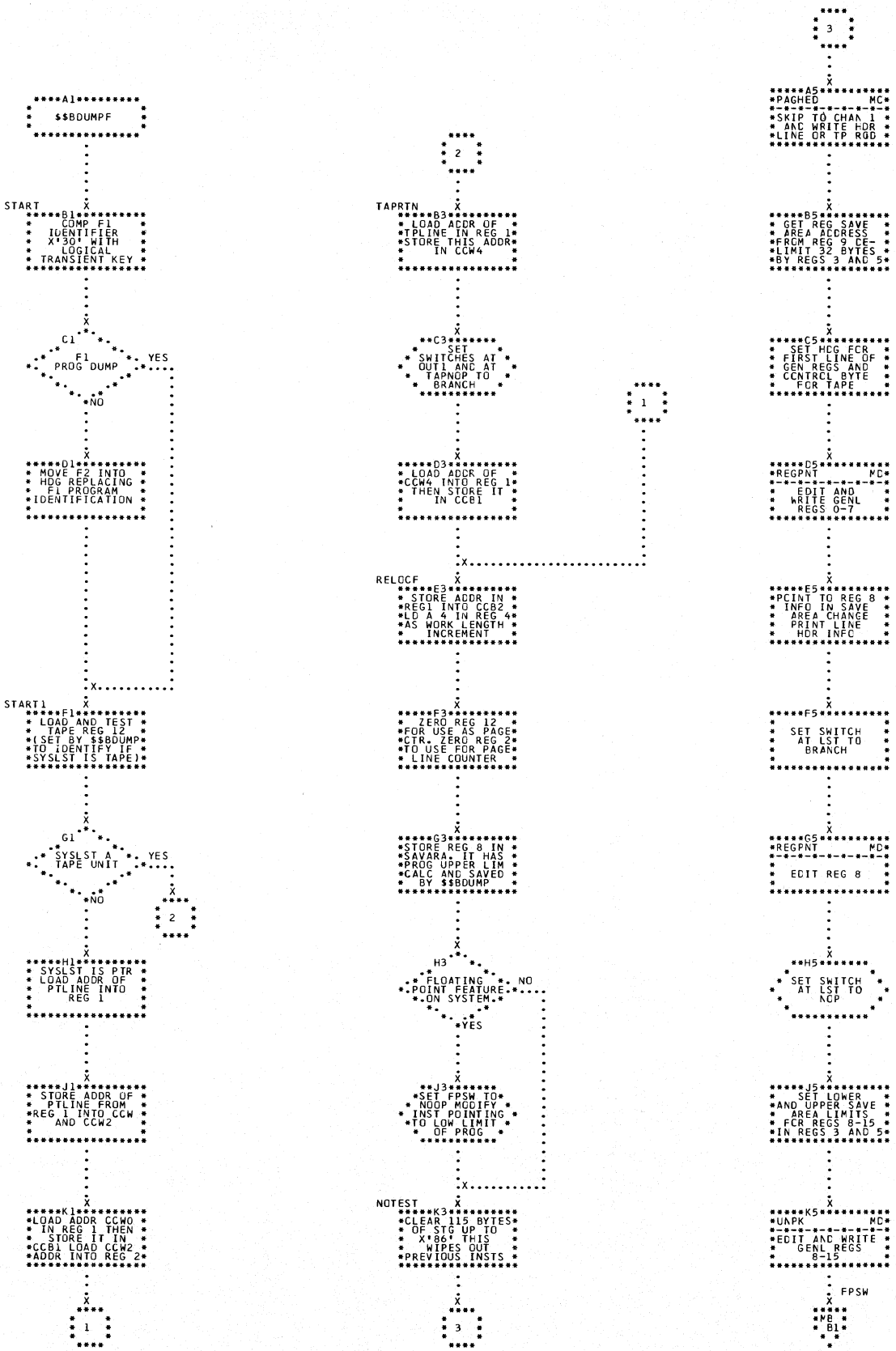


Chart MB. $$BDUMPF - Standard System Dump, Foreground Program Dump (Part 2 of 2)
Refer to Chart 14.

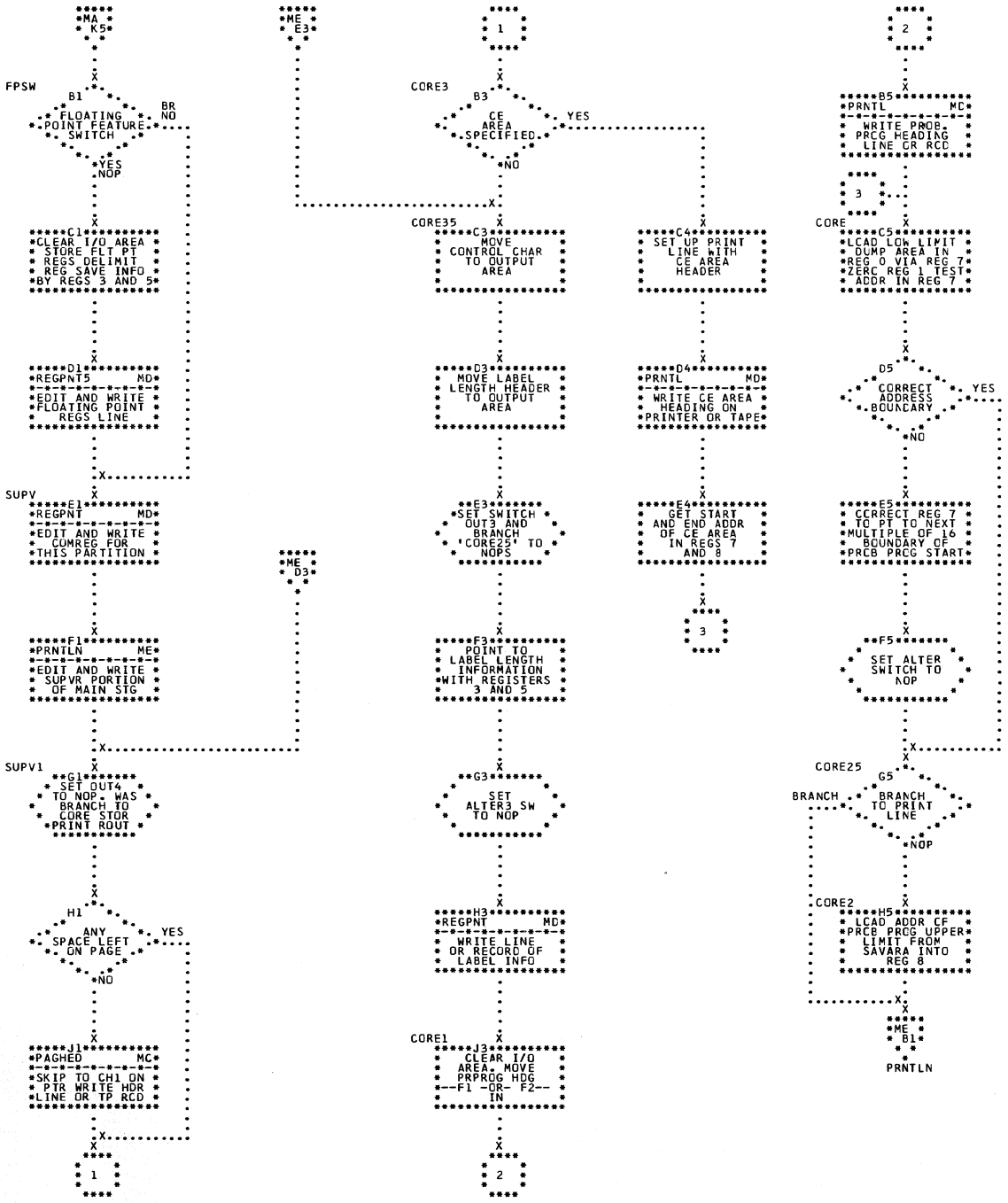


Chart MC. $$BDUMPF - Standard System Dump, Prepare Page Headings and PIOCS Subroutines
Refer to Chart 14.

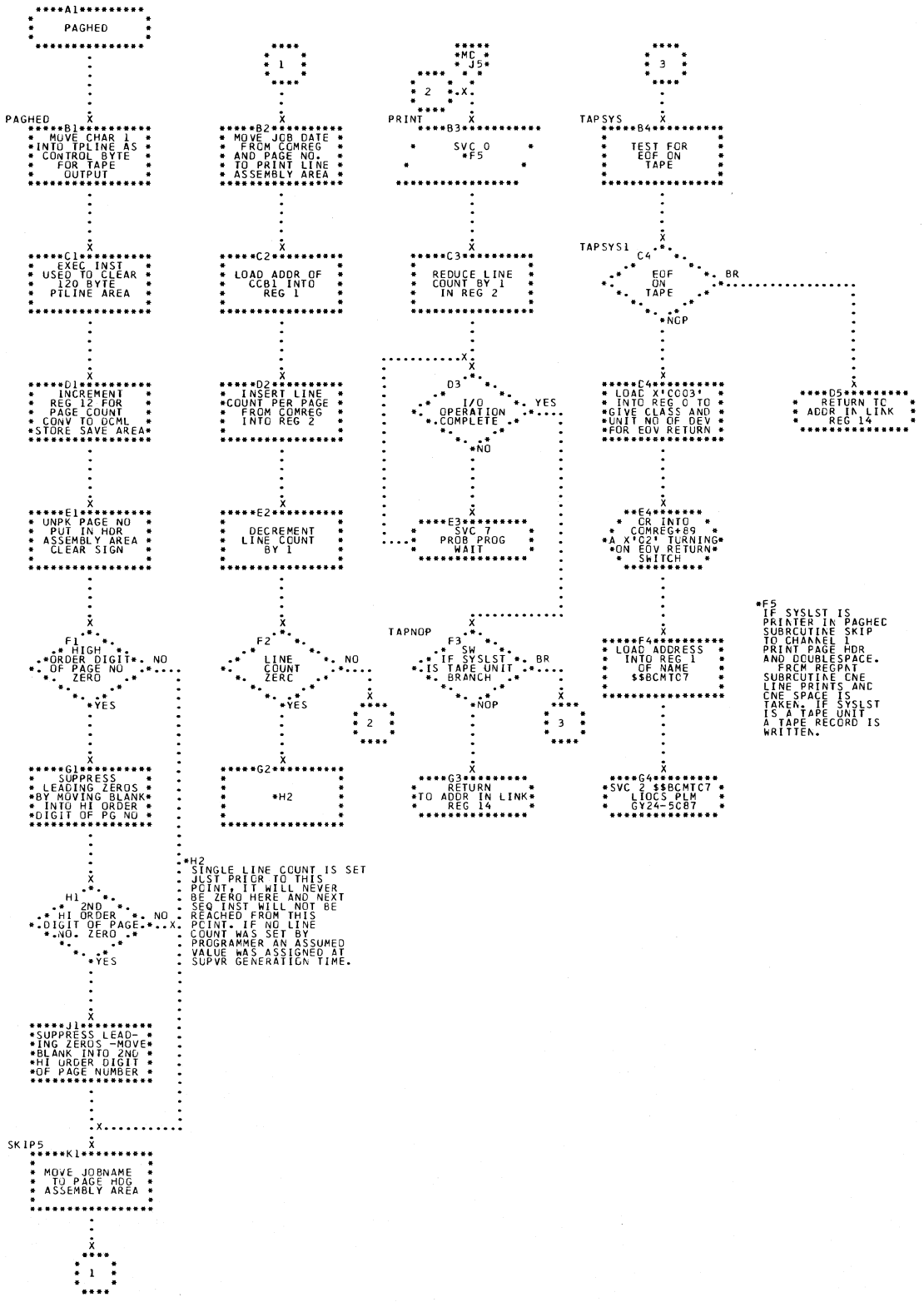


Chart MD. $$BDUMPF - Standard System Dump, Prepare and Edit a Line Subroutine
Refer to Chart 14.

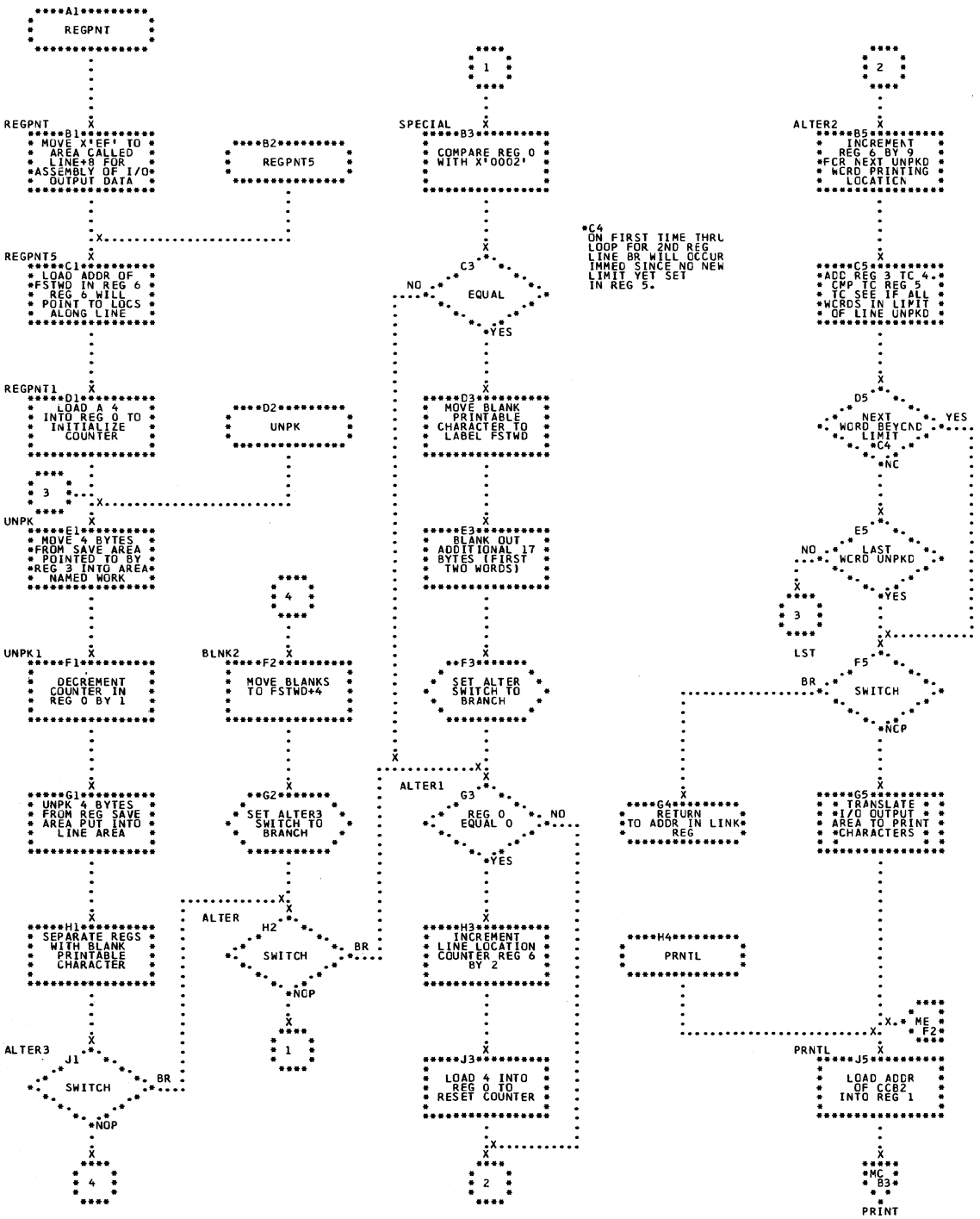


Chart ME. $$BDUMPF - Standard System Dump, Line Test Subroutines (Part 1 of 2)
Refer to Chart 14.

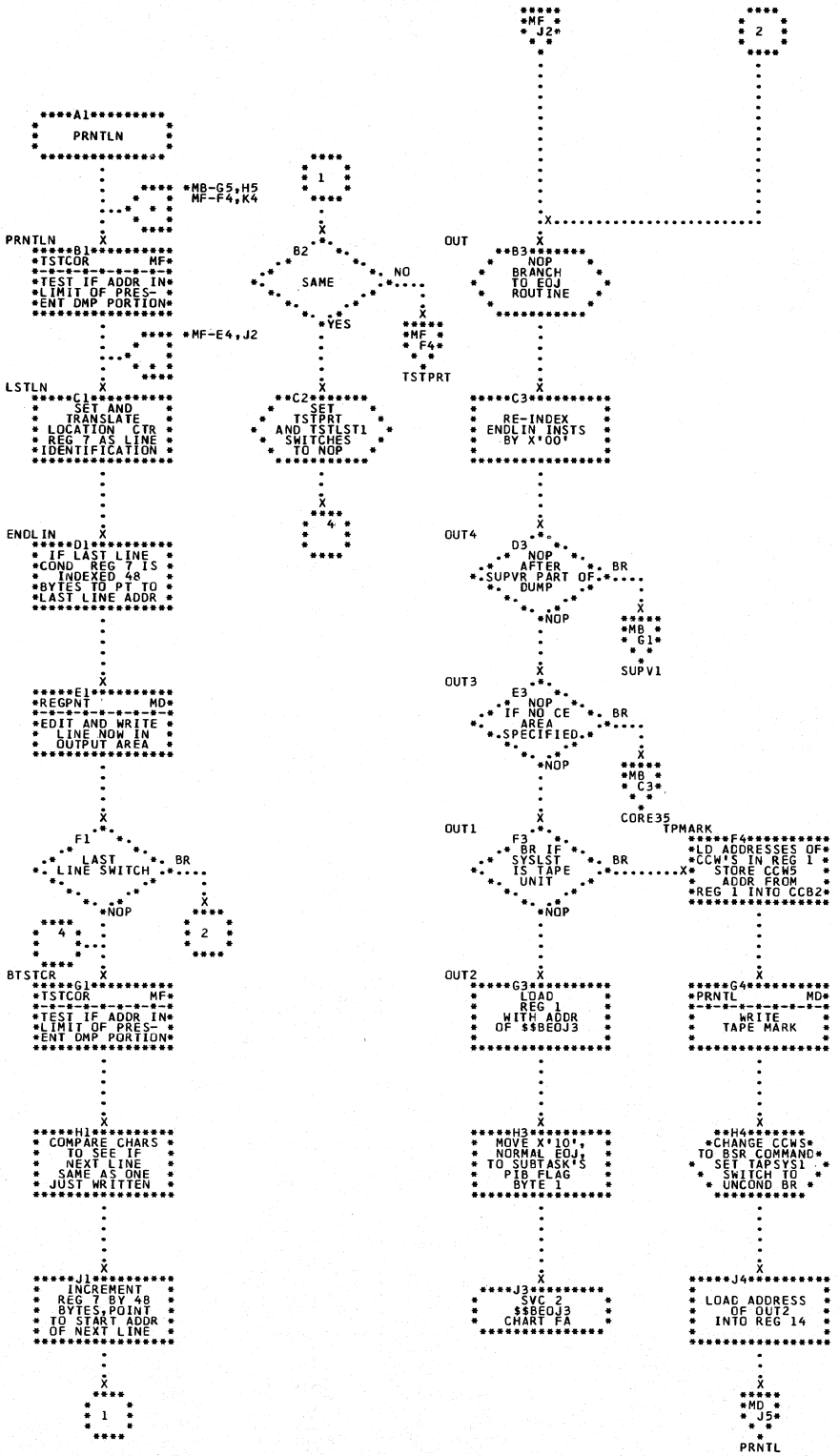


Chart MF. $$BDUMPF - Standard System Dump, Line Test Subroutines (Part 2 of 2)
Refer to Chart 14.

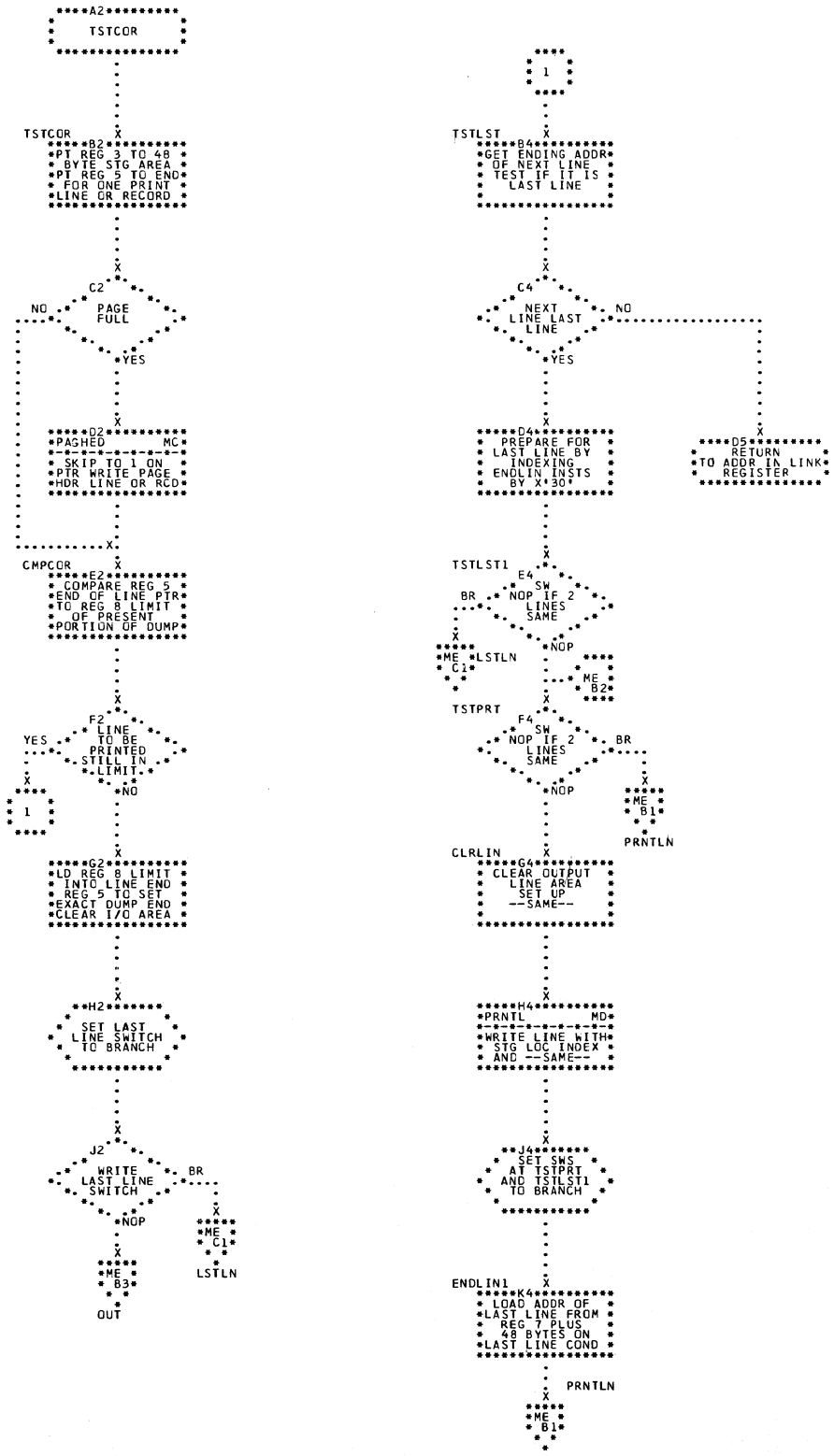


Chart MG. $$BDUMPB - Standard System Dump, Initialization for BG Storage Dump on Printer or Tape

Refer to Chart 14.

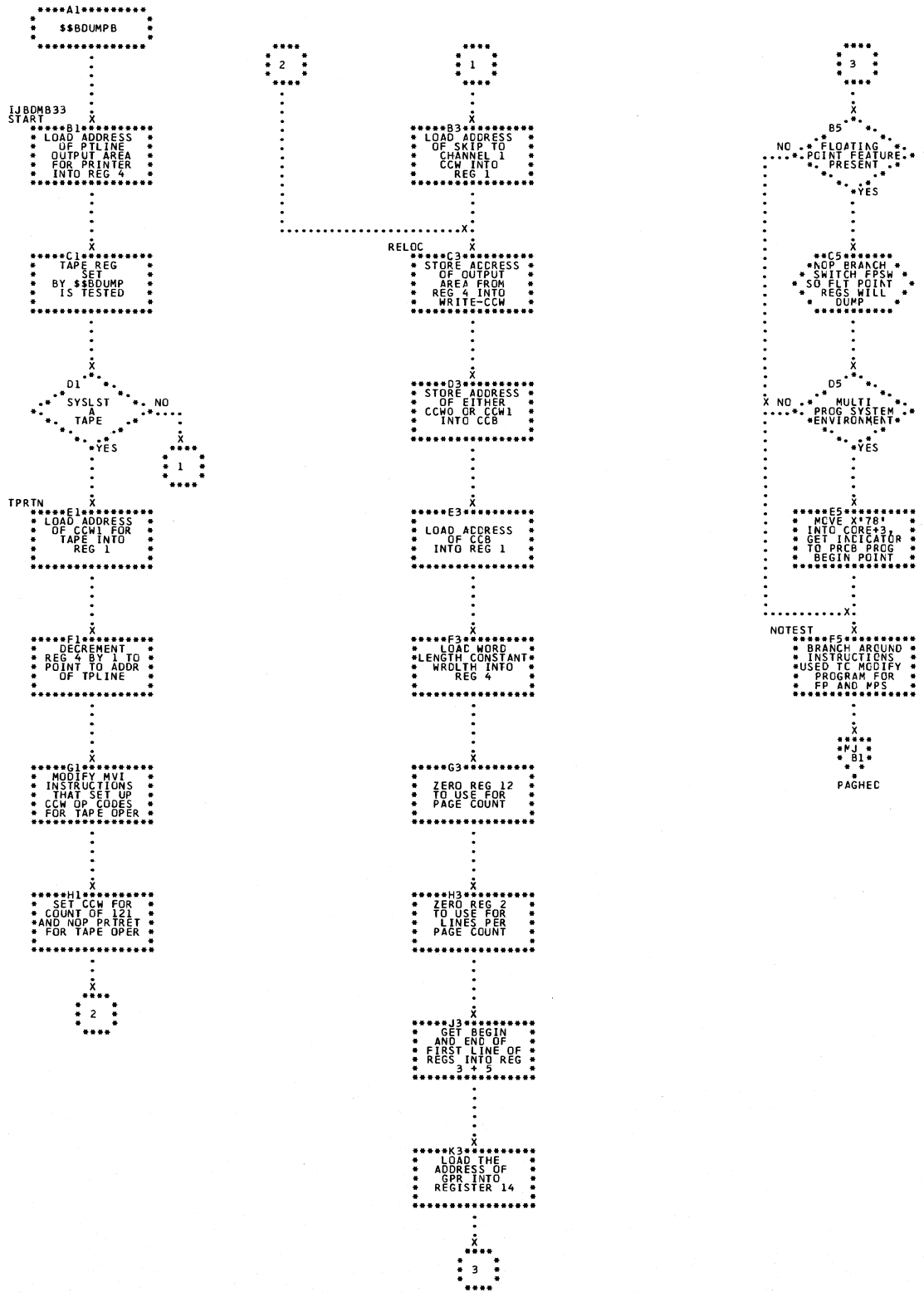


Chart MH. $$BDUMPB - Standard System Dump, BG Dump on Printer or Tape
Refer to Chart 14.

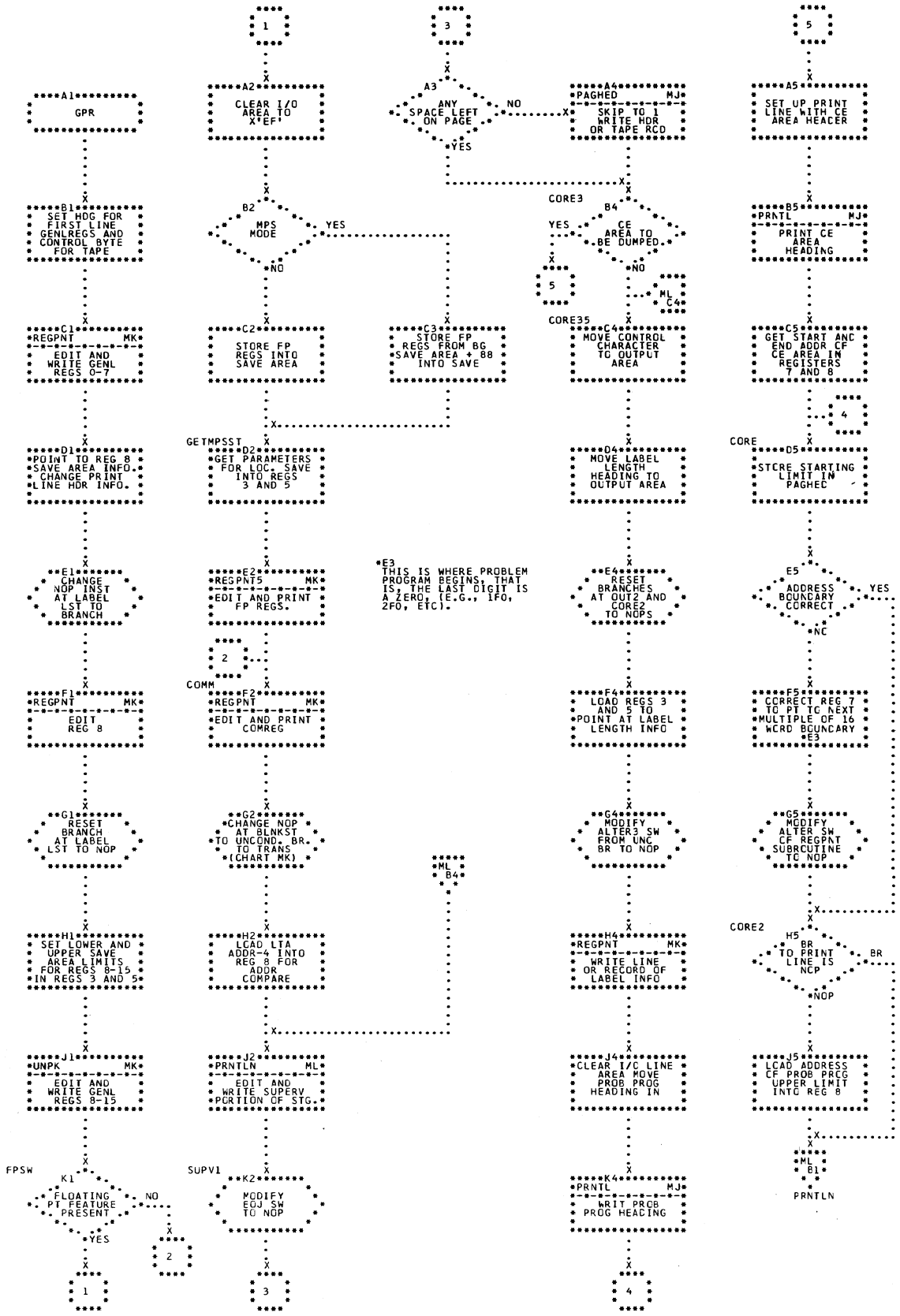


Chart MJ. $$BDUMPB - Standard System Dump, Prepare Page Headings and PIOCS Subroutines
Refer to Chart 14.

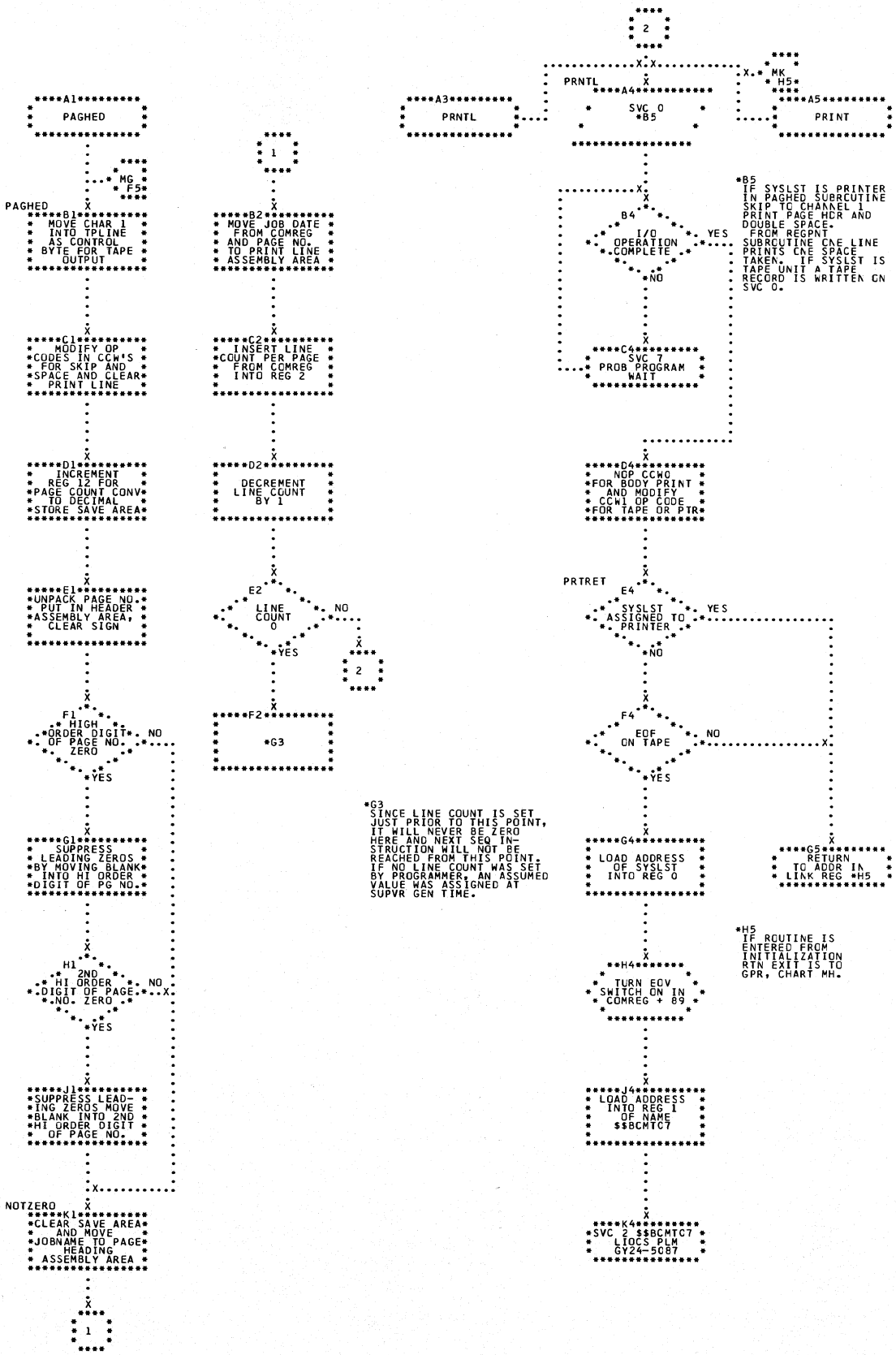


Chart MK. $$BDUMPB - Standard System Dump, Prepare and Edit a Line Subroutine
Refer to Chart 14.

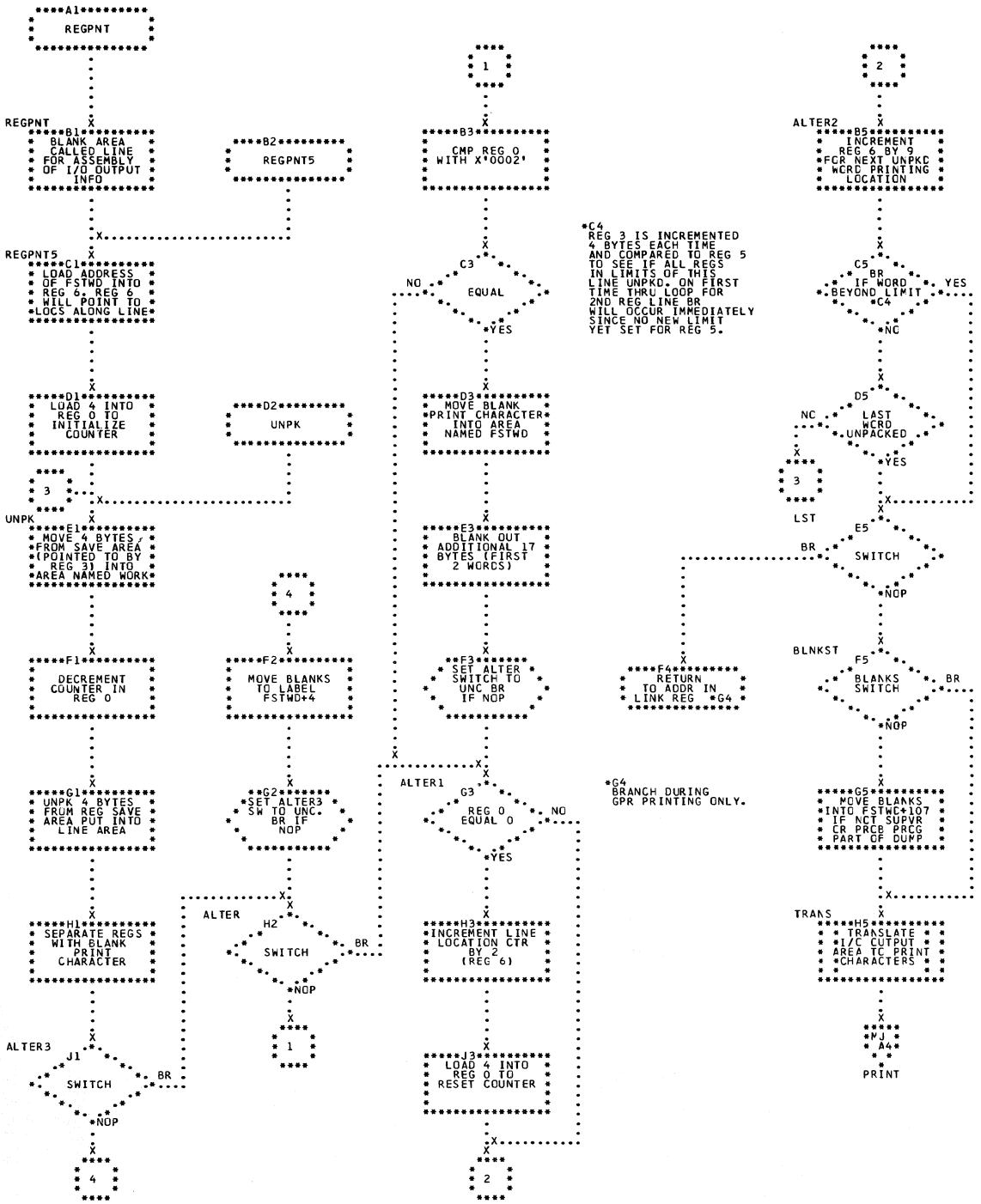


Chart ML. $$BDUMPB - Standard System Dump, Line Test Subroutines
Refer to Chart 14.

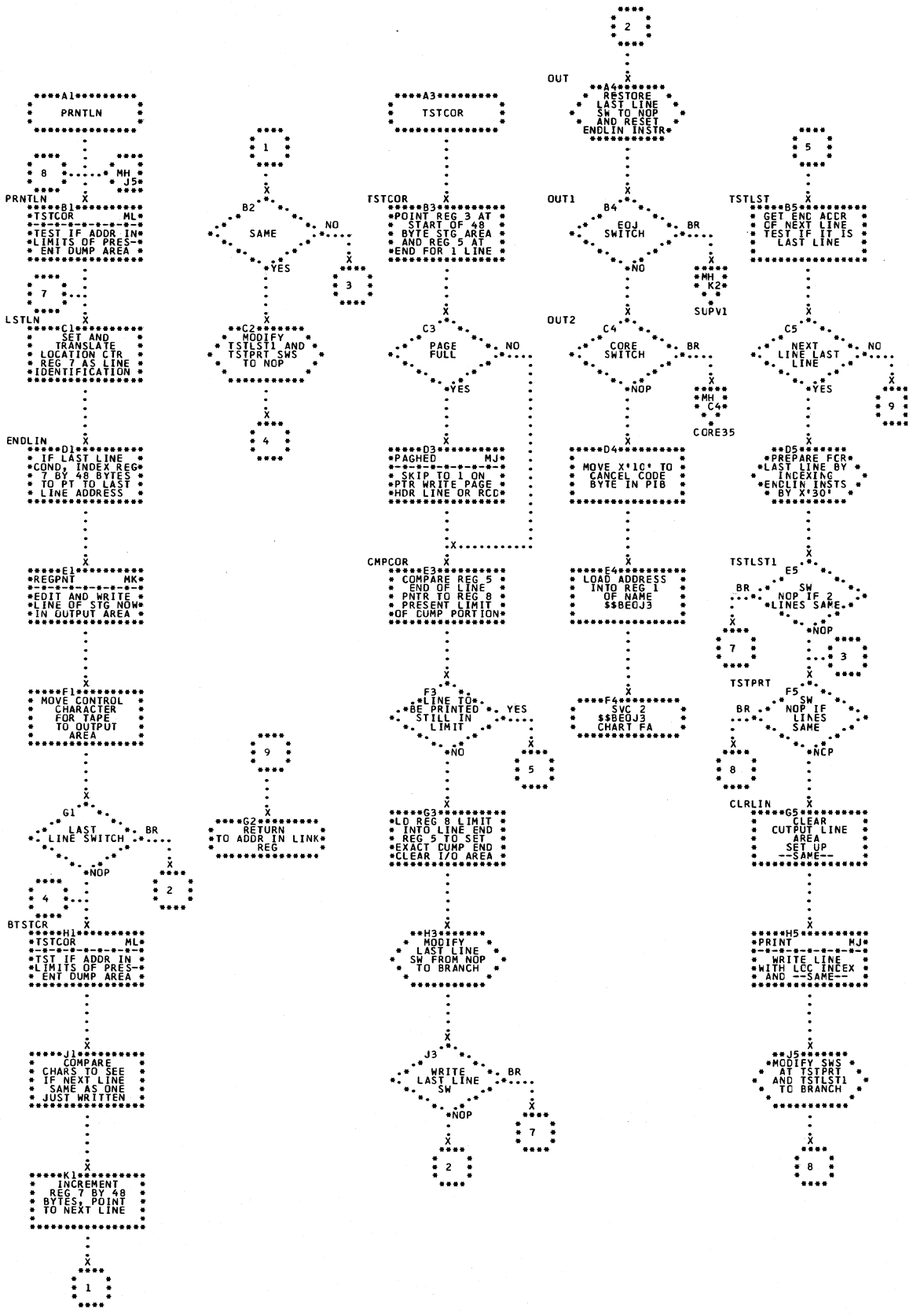


Chart MM. $$BDUMPD - Standard System Dump, Dump on Disk Device (Part 1 of 2)
Refer to Chart 14.

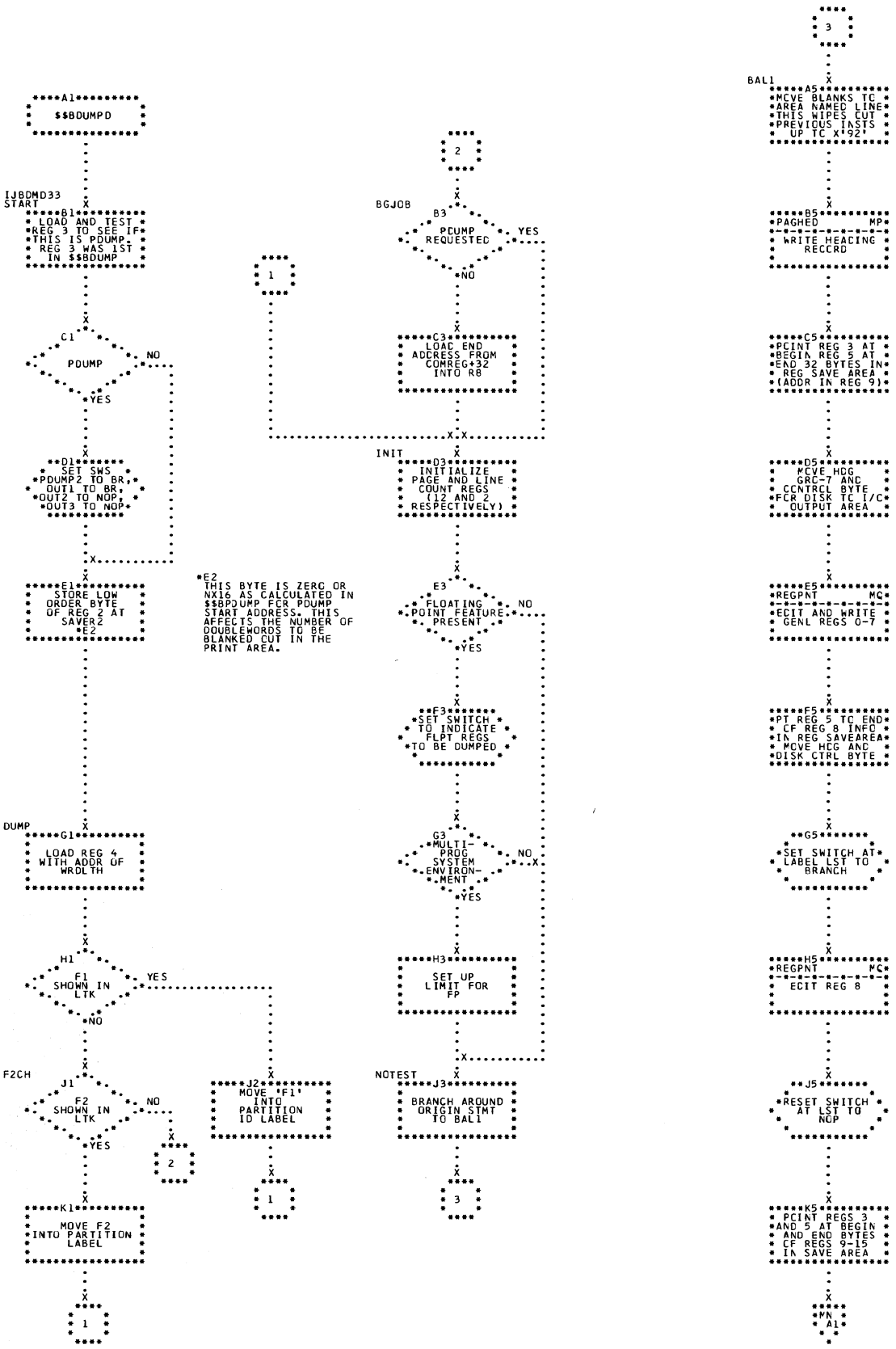


Chart MN. $$BDUMPD - Standard System Dump, Dump on Disk Device (Part 2 of 2)
Refer to Chart 14.

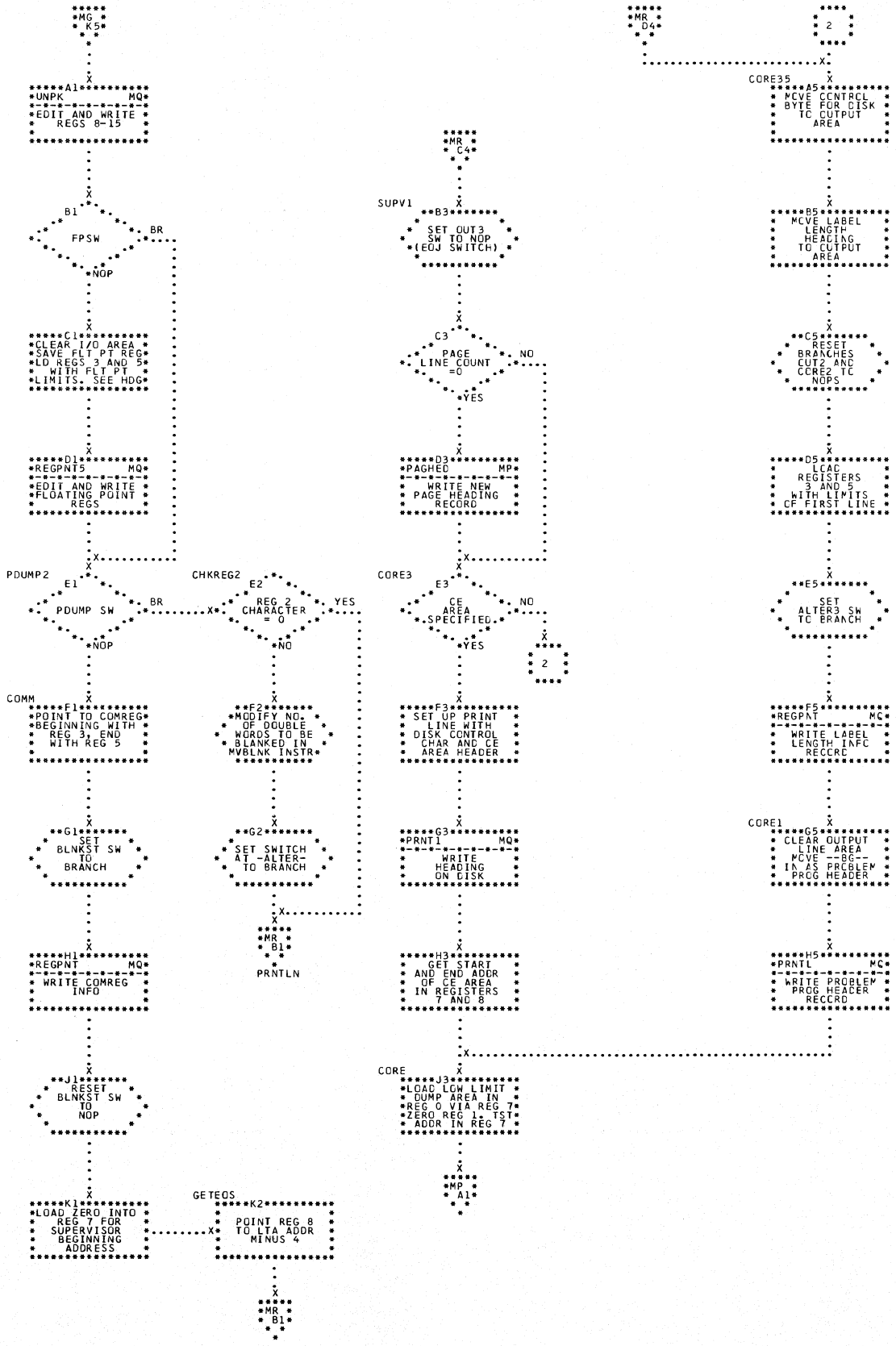


Chart MP. $$BDUMPD - Standard System Dump, Prepare Page Headings and PIOCS Subroutines
Refer to Chart 14.

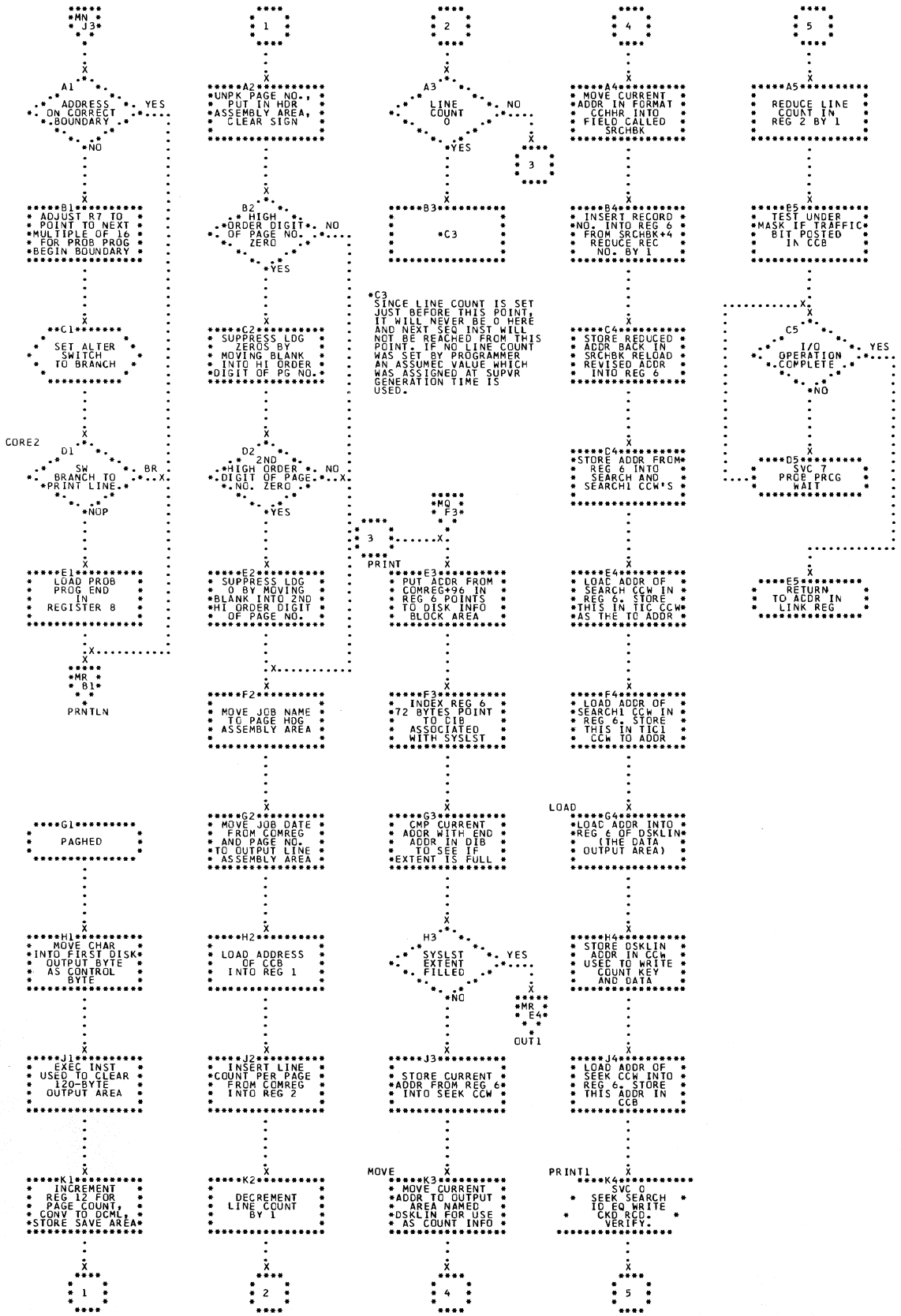


Chart MQ. $$BDUMPD - Standard System Dump, Prepare and Edit a Line Subroutine
Refer to Chart 14.

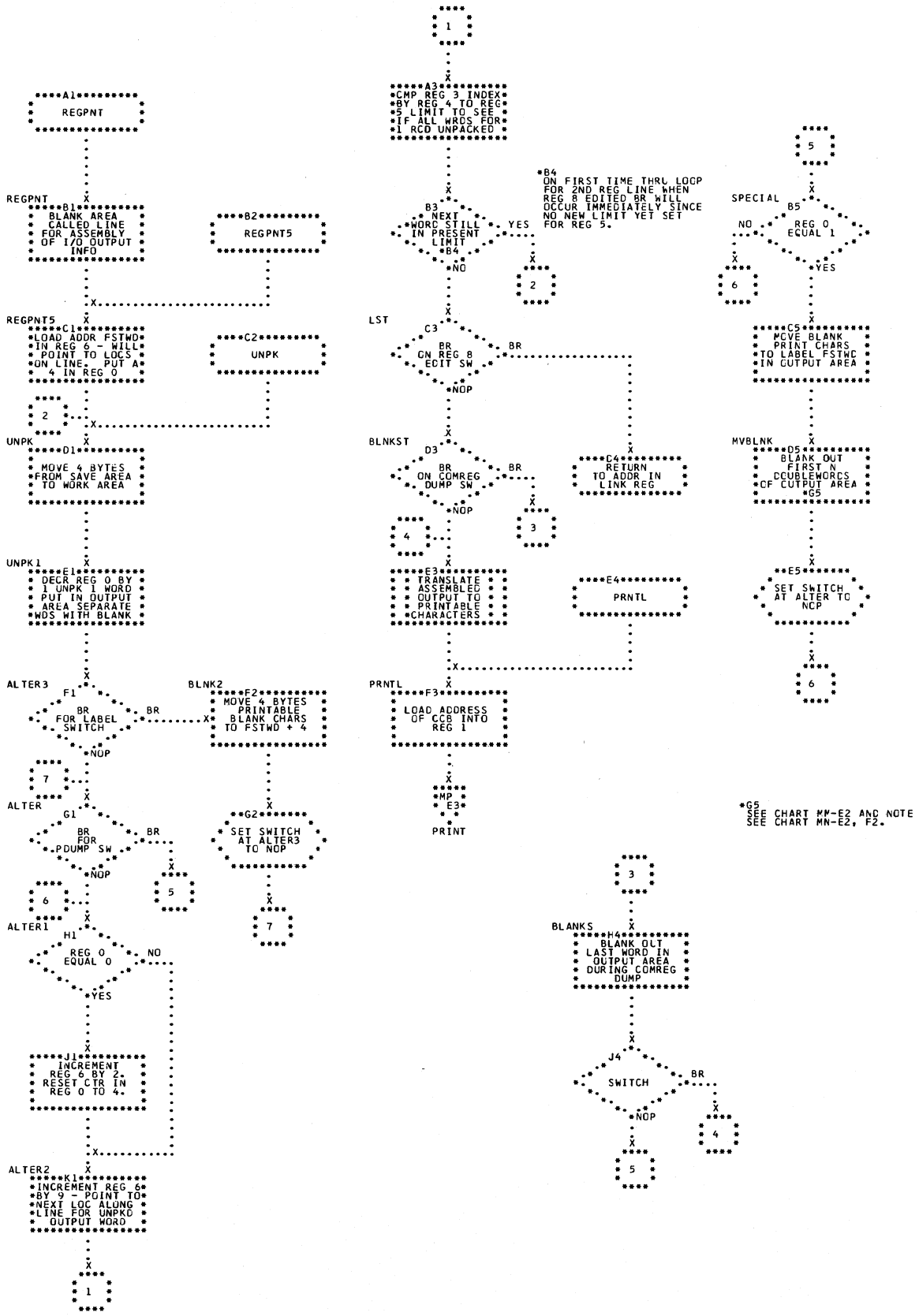


Chart MR. $$BDUMPD - Standard System Dump, Line Test Subroutines
Refer to Chart 14.

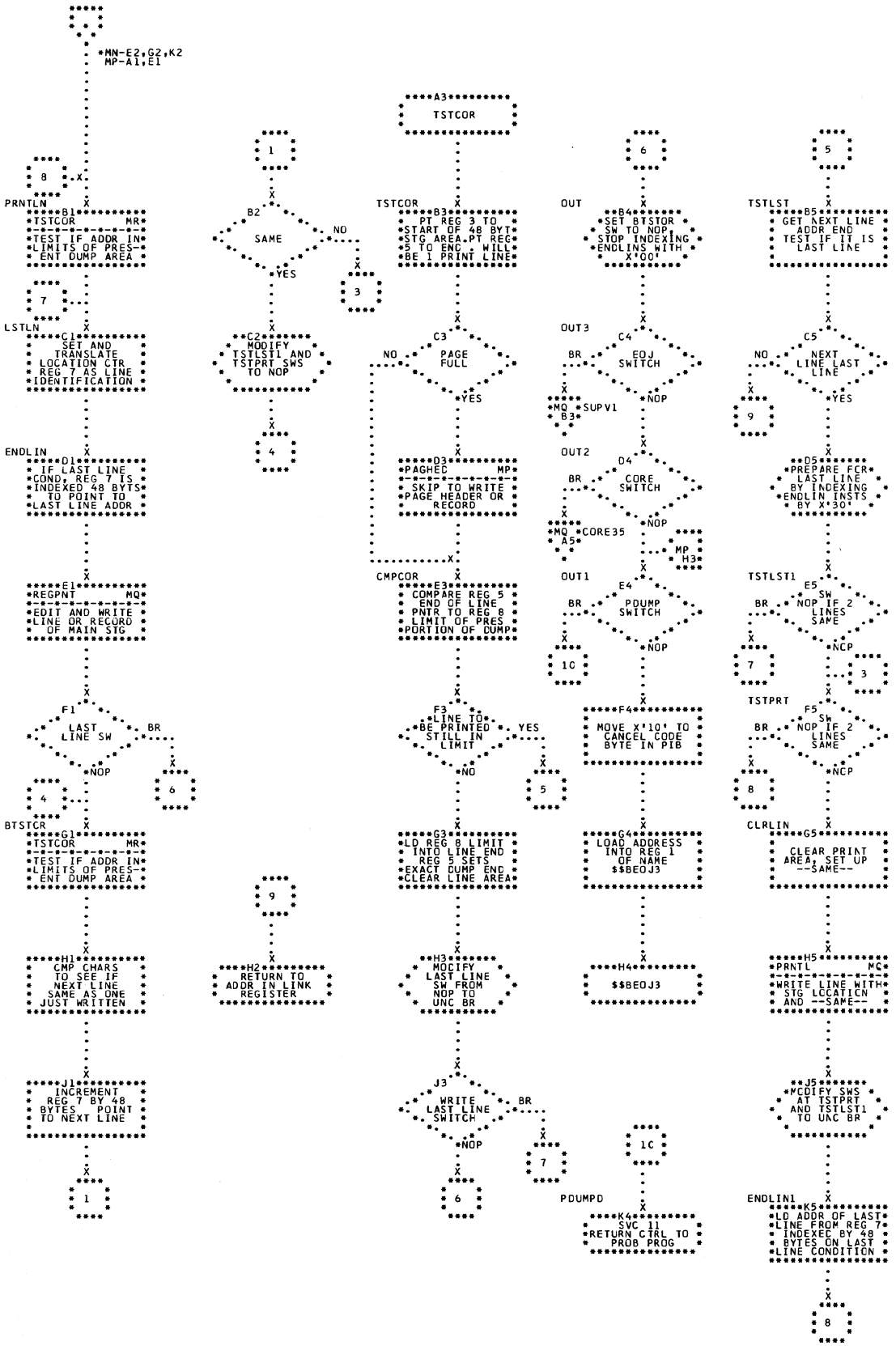


Chart NA. $$BPDUMP - Standard System Dump, Parameter Storage Dump Monitor
Refer to Chart 15.

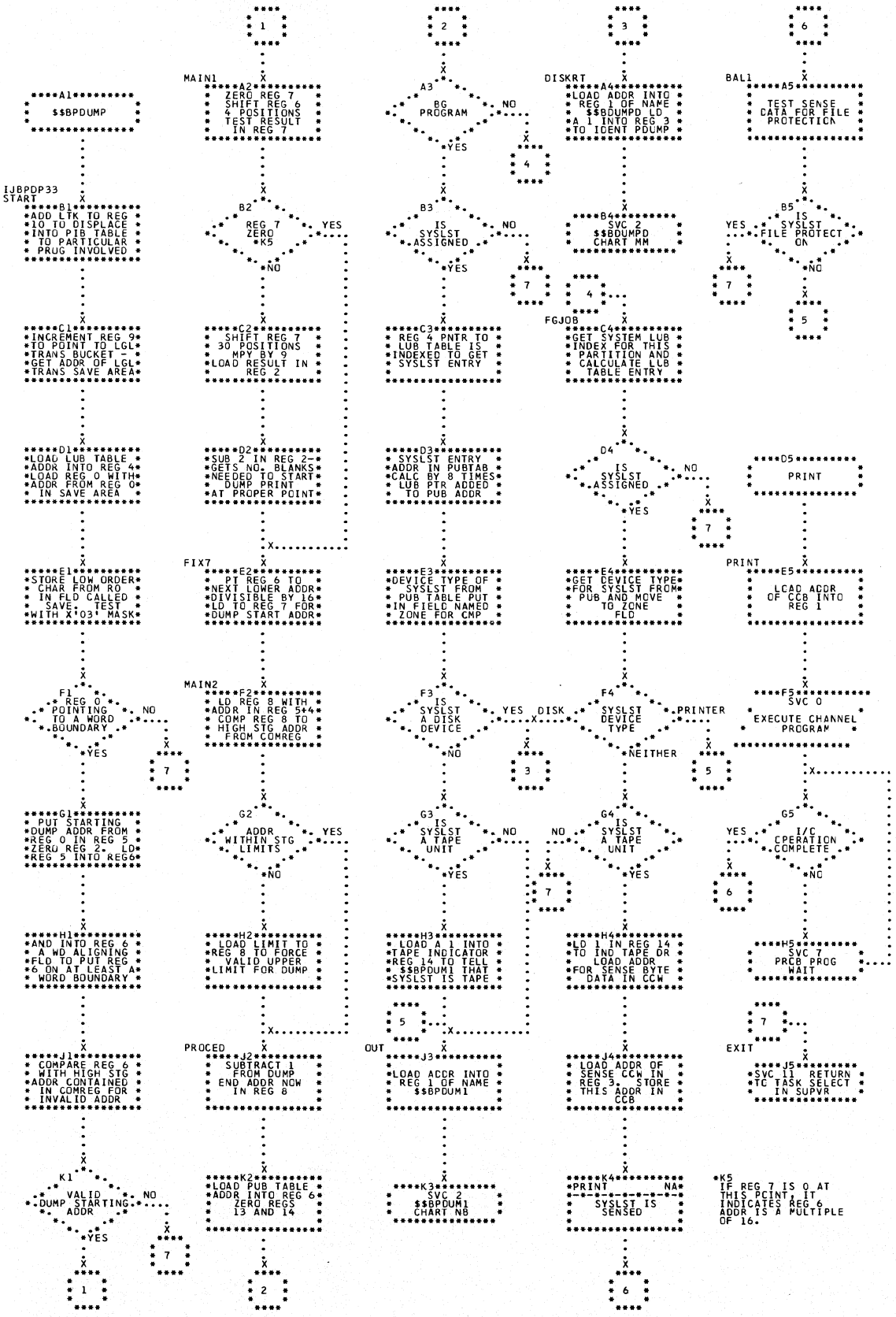


Chart NB. $$BPDUM1 - Standard System Dump, Initialize Parameter Dump on Printer or Tape
Refer to Chart 15.

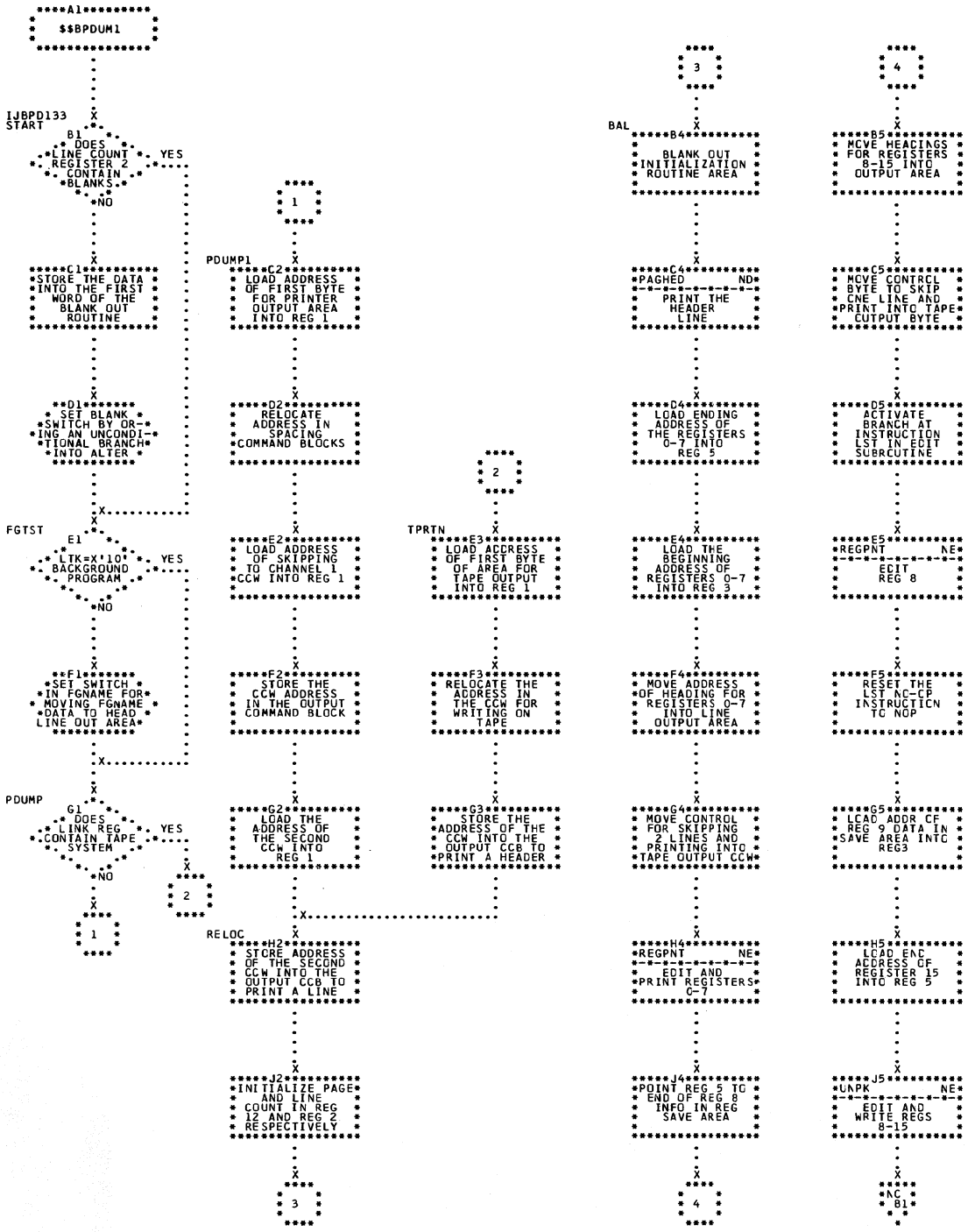


Chart NC. $$BPDUM1 - Standard System Dump, Parameter Storage Dump on Printer or Tape
Refer to Chart 15.

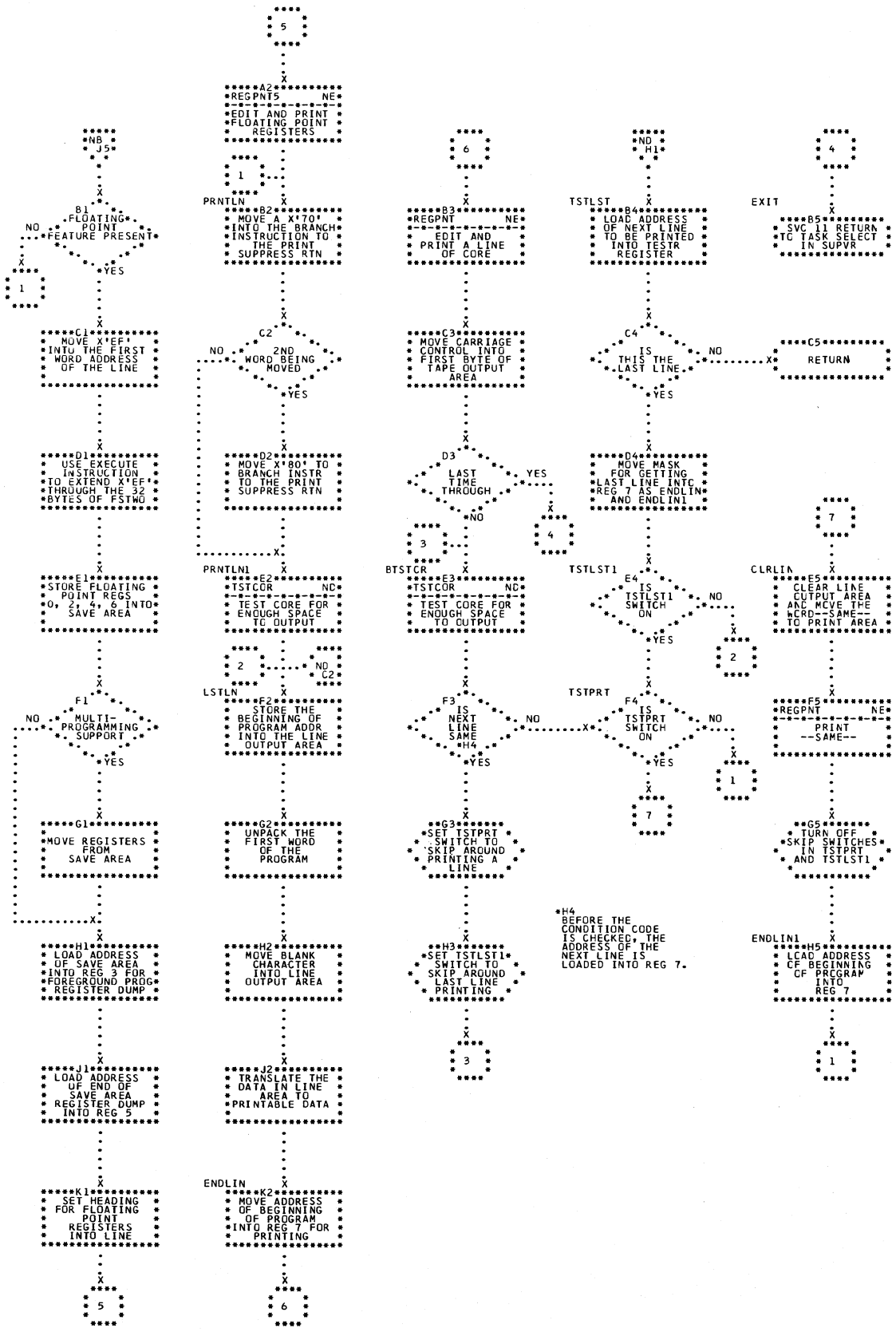


Chart ND. $$BPDUM1 - Standard System Dump, Line Test Subroutines
Refer to Chart 15.

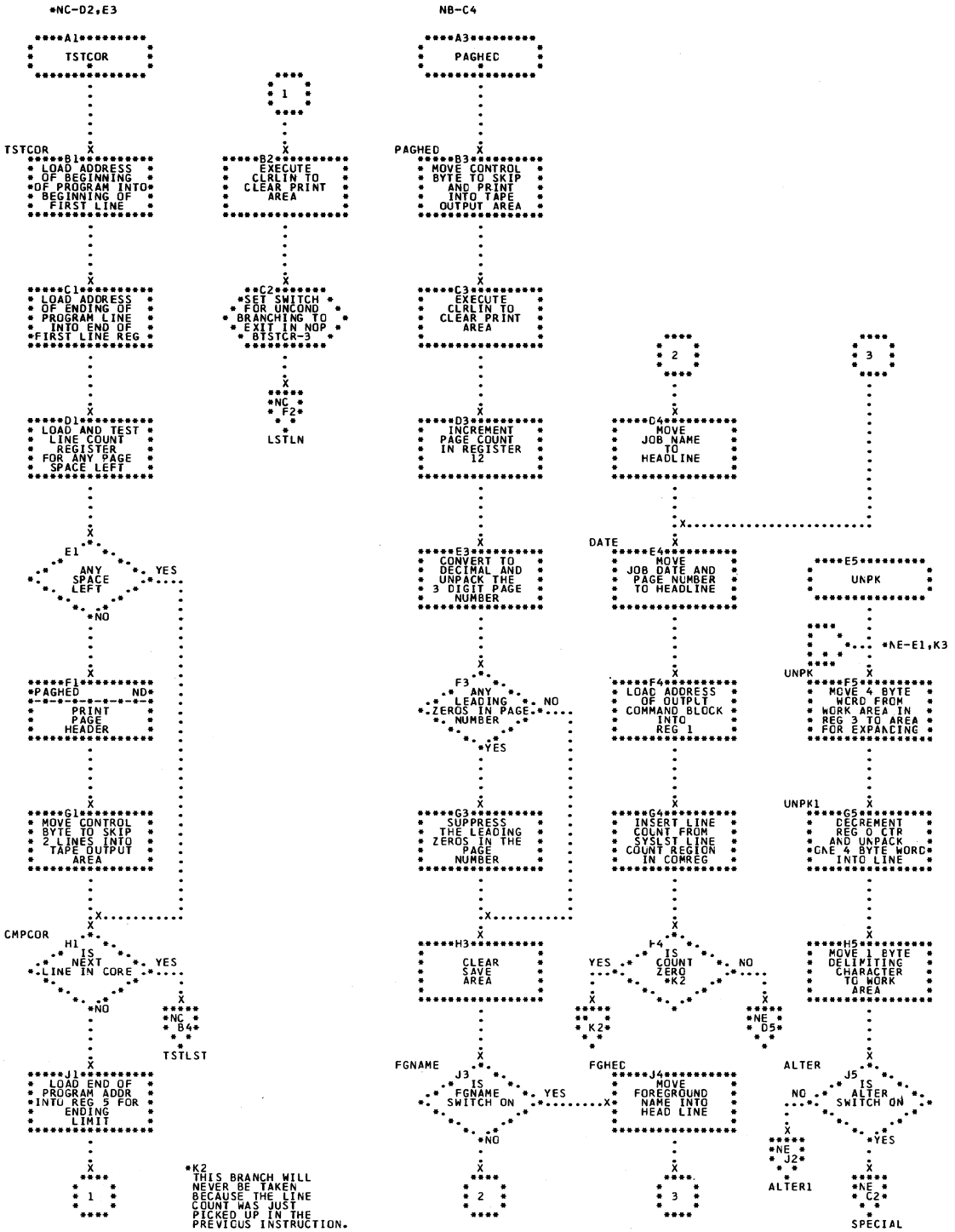


Chart NE. $$BPDUM1 - Standard System Dump, Prepare and Edit a Line Subroutine
Refer to Chart 15.

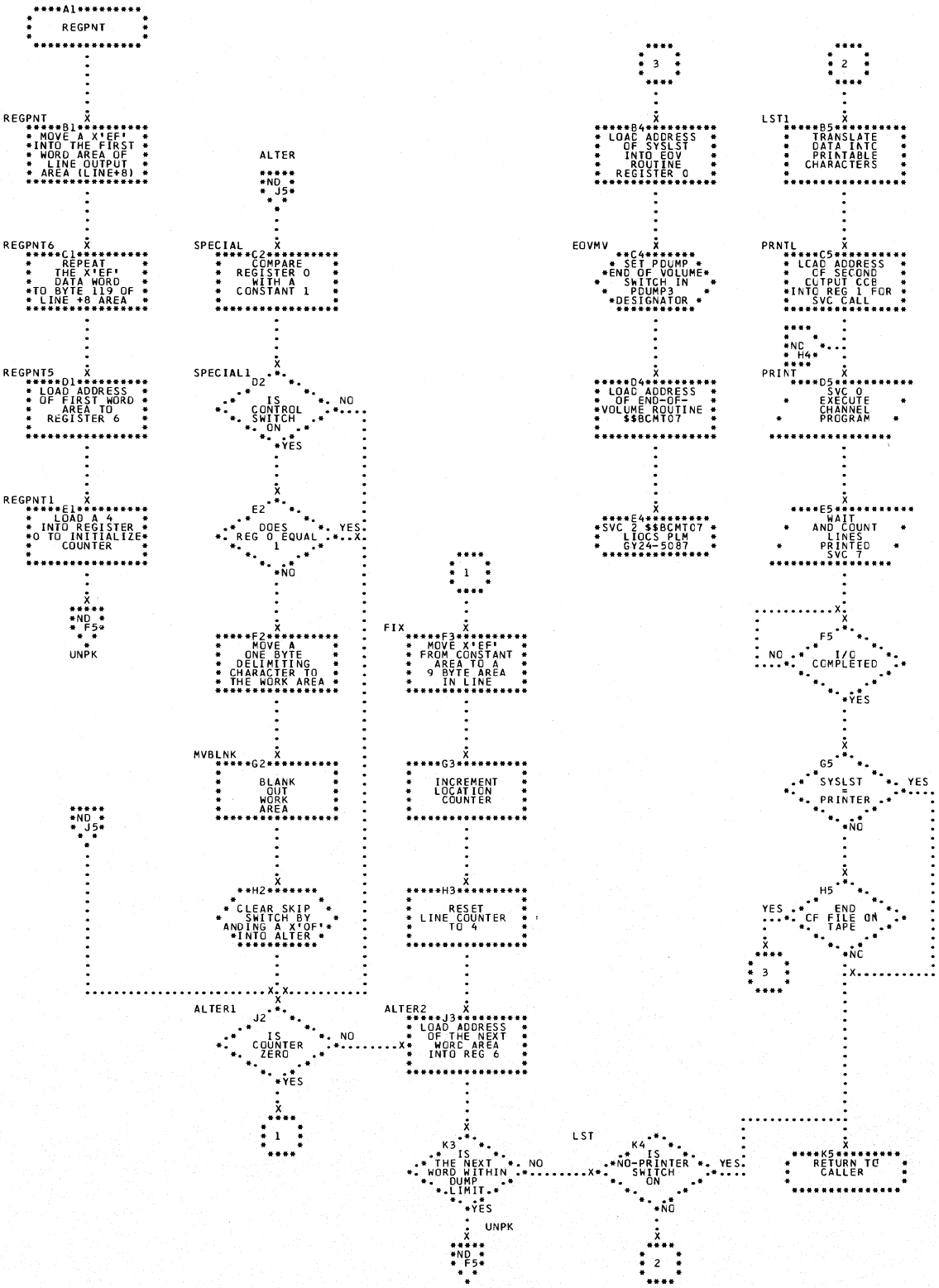


Chart NF. $$BSYSWR - Set up a Write on SYSRES Operation; Move Label Cylinder Address to COMREG

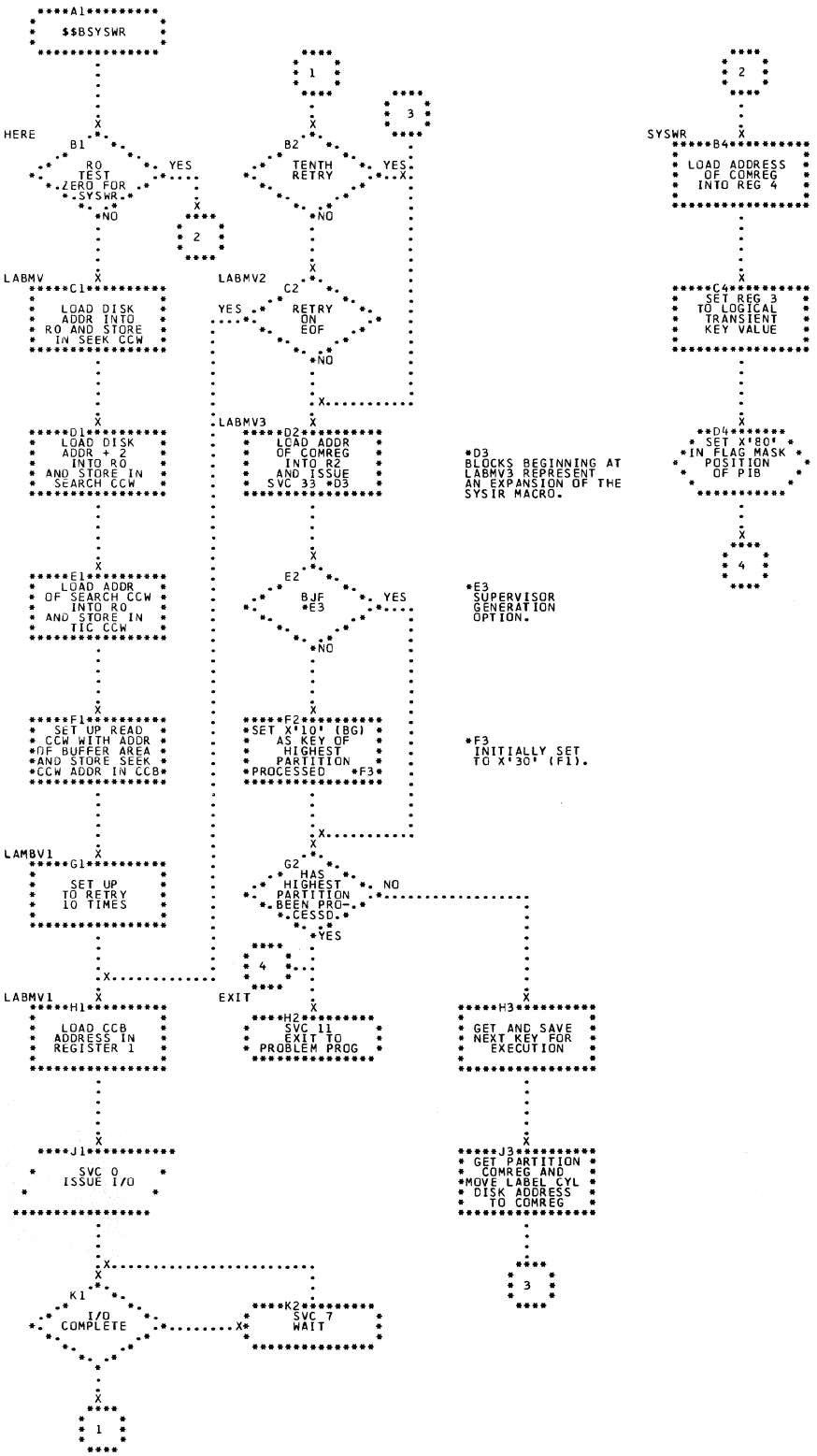


Chart NG. $$BSDRUP - EREP Processing, Suppress File-Ready-for-Recording Switch

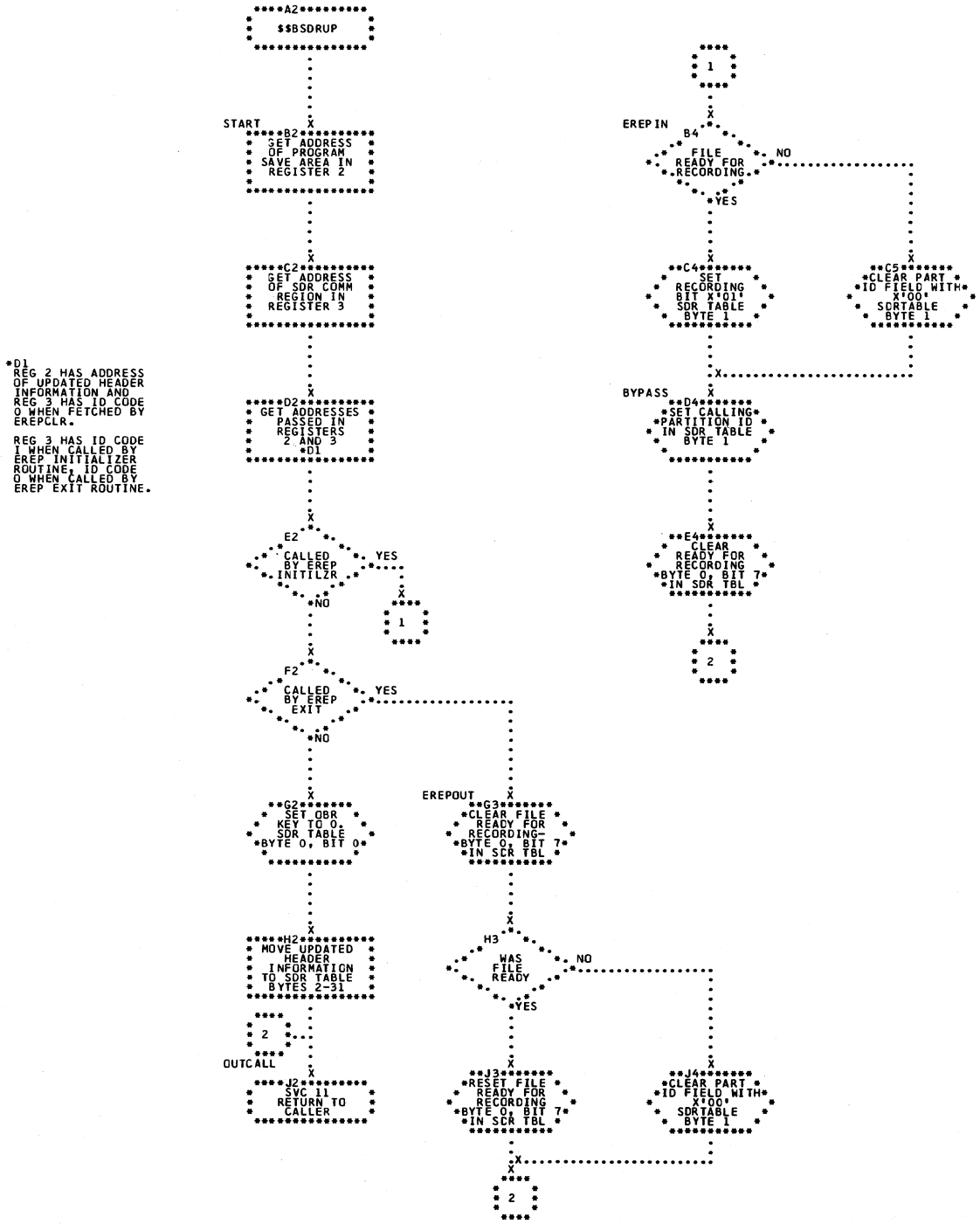


Chart NH. $$BCCHHR - Core Image Directory Scan (Part 1 of 3)

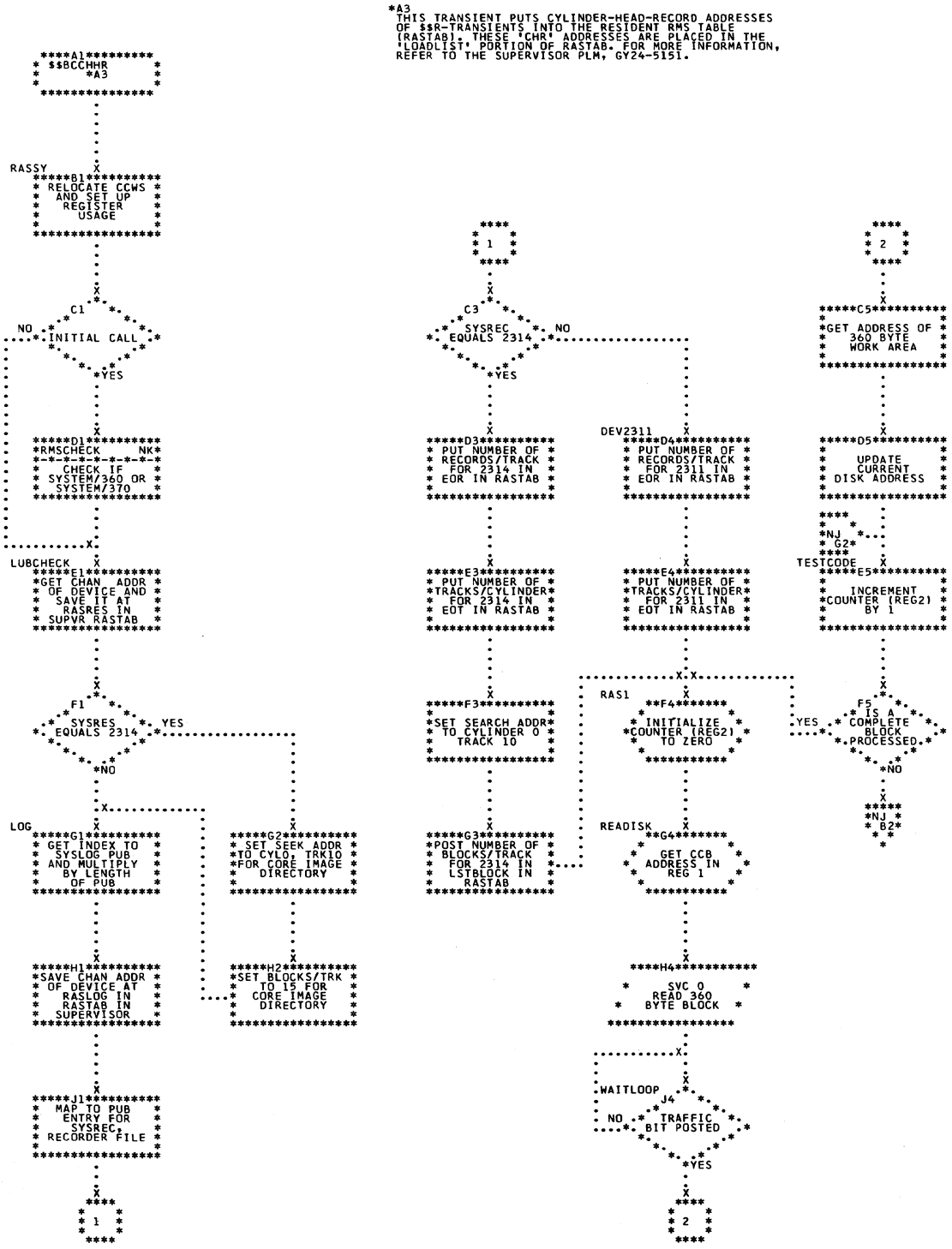


Chart NJ. $$BCCHHR - Core Image Directory Scan (Part 2 of 3)

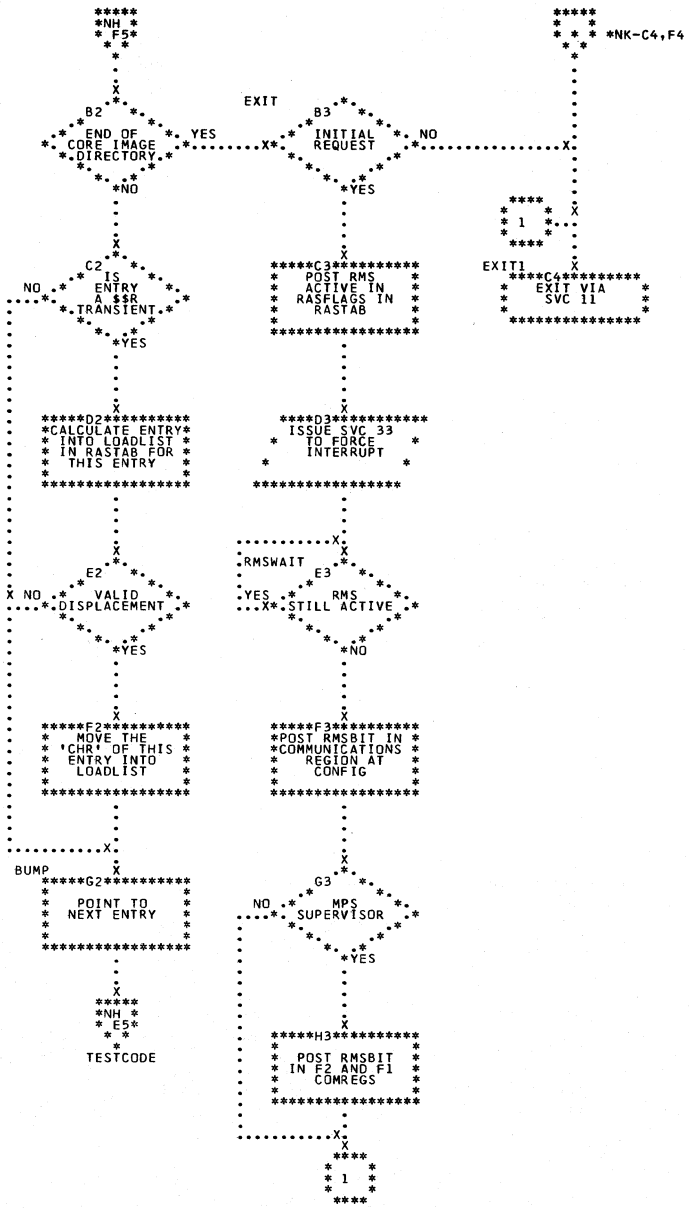


Chart NK. $$BCCHHR - Core Image Directory Scan (Part 3 of 3)

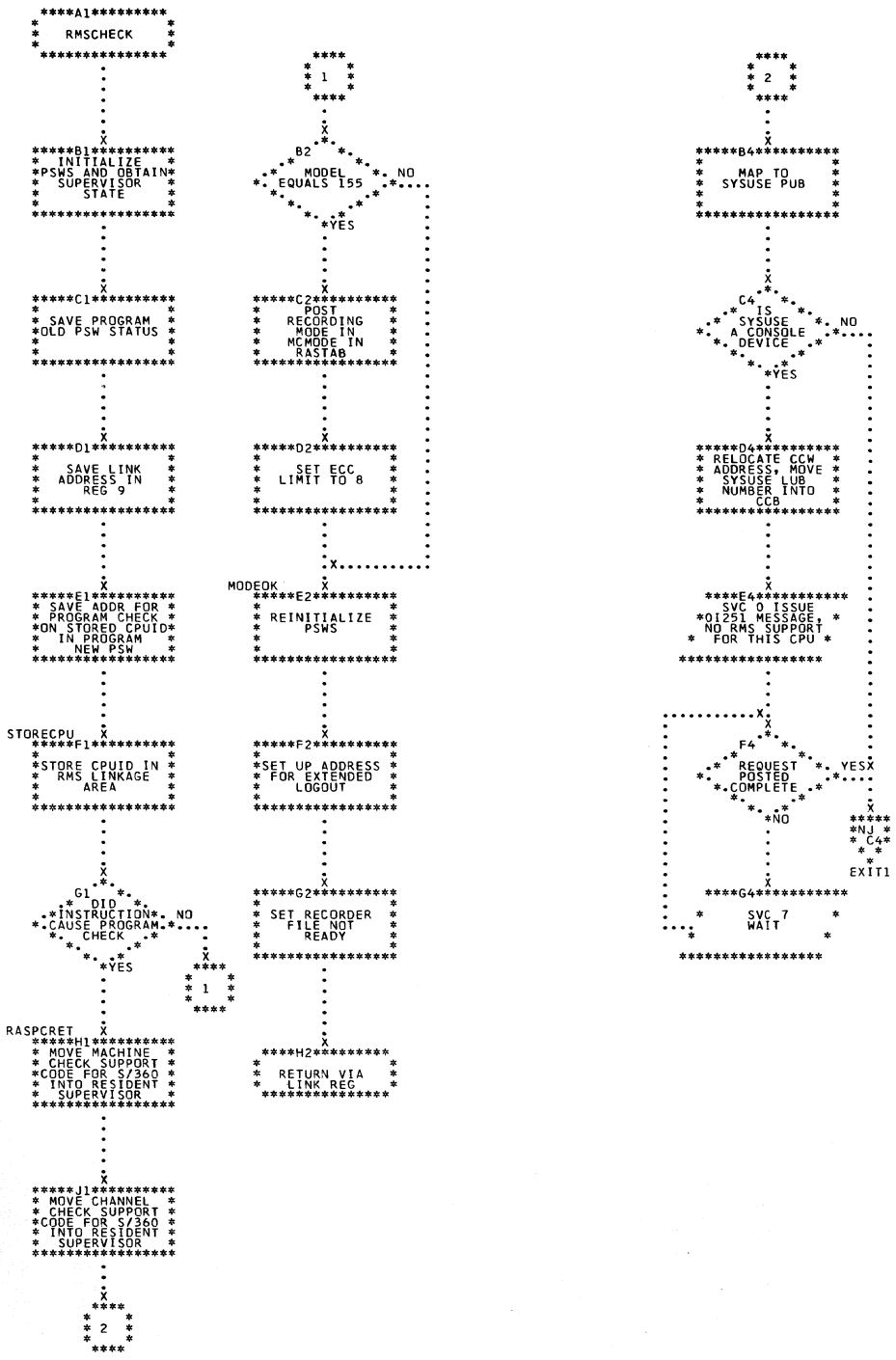


Chart PA. $$BATTNQ - MODE Command Parameter Processor (Part 1 of 2)
Refer to Chart 07.

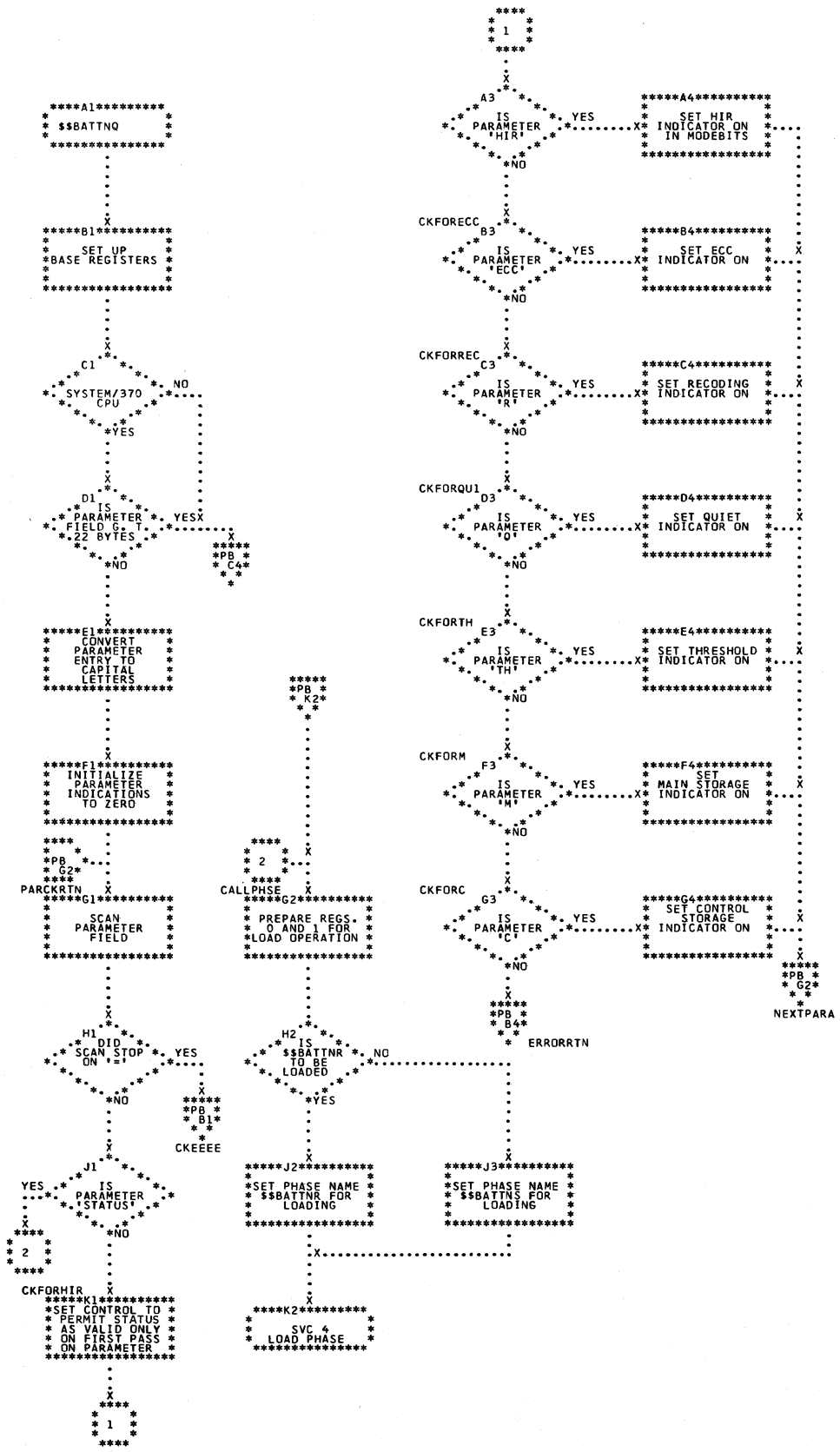


Chart PB. $$BATTNQ - MODE Command Parameter Processor (Part 2 of 2)
Refer to Chart 07.

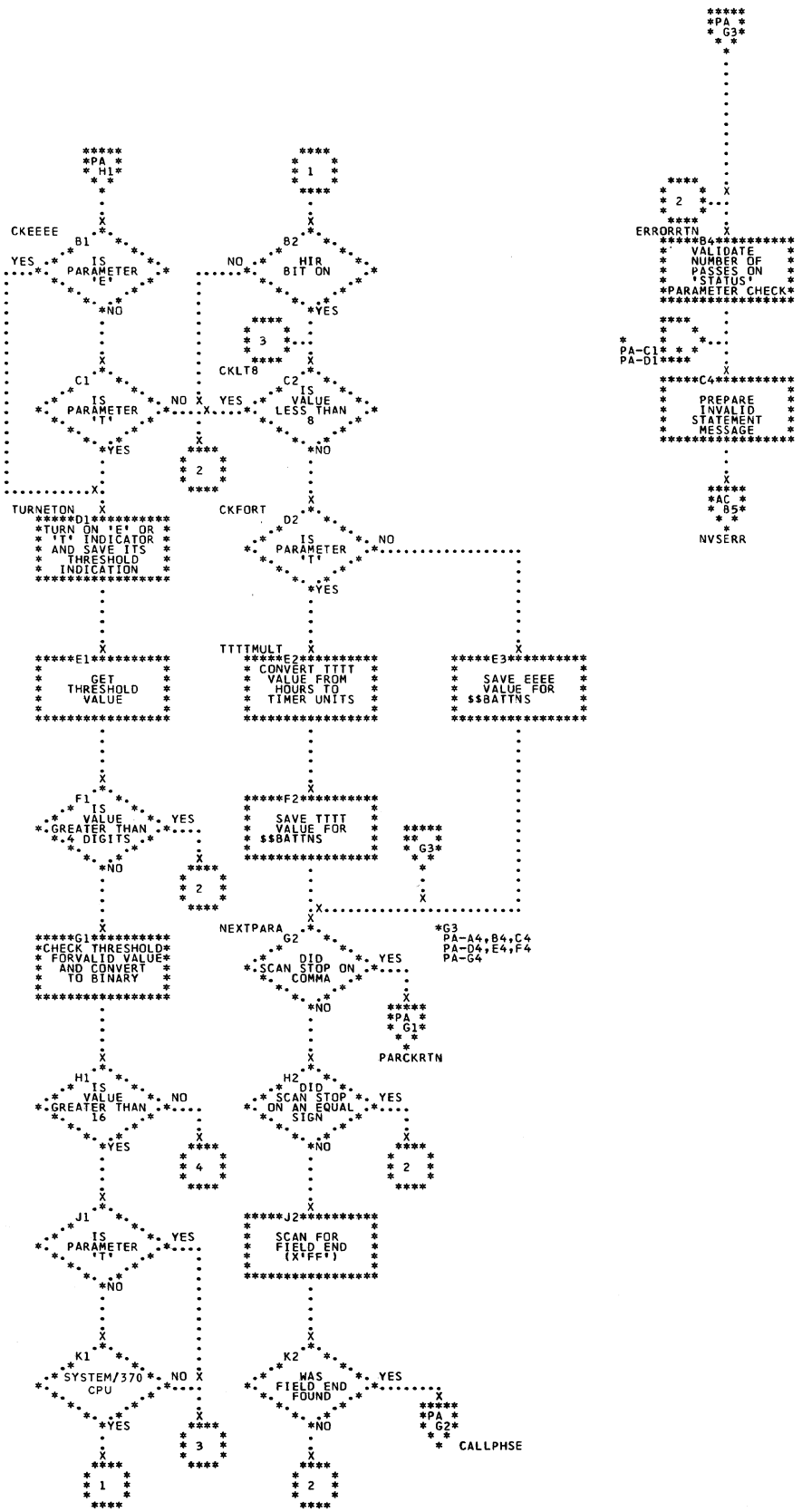


Chart PC. $$BATTNR - MODE Command Status Report Processor (Part 1 of 2)
Refer to Chart 07.

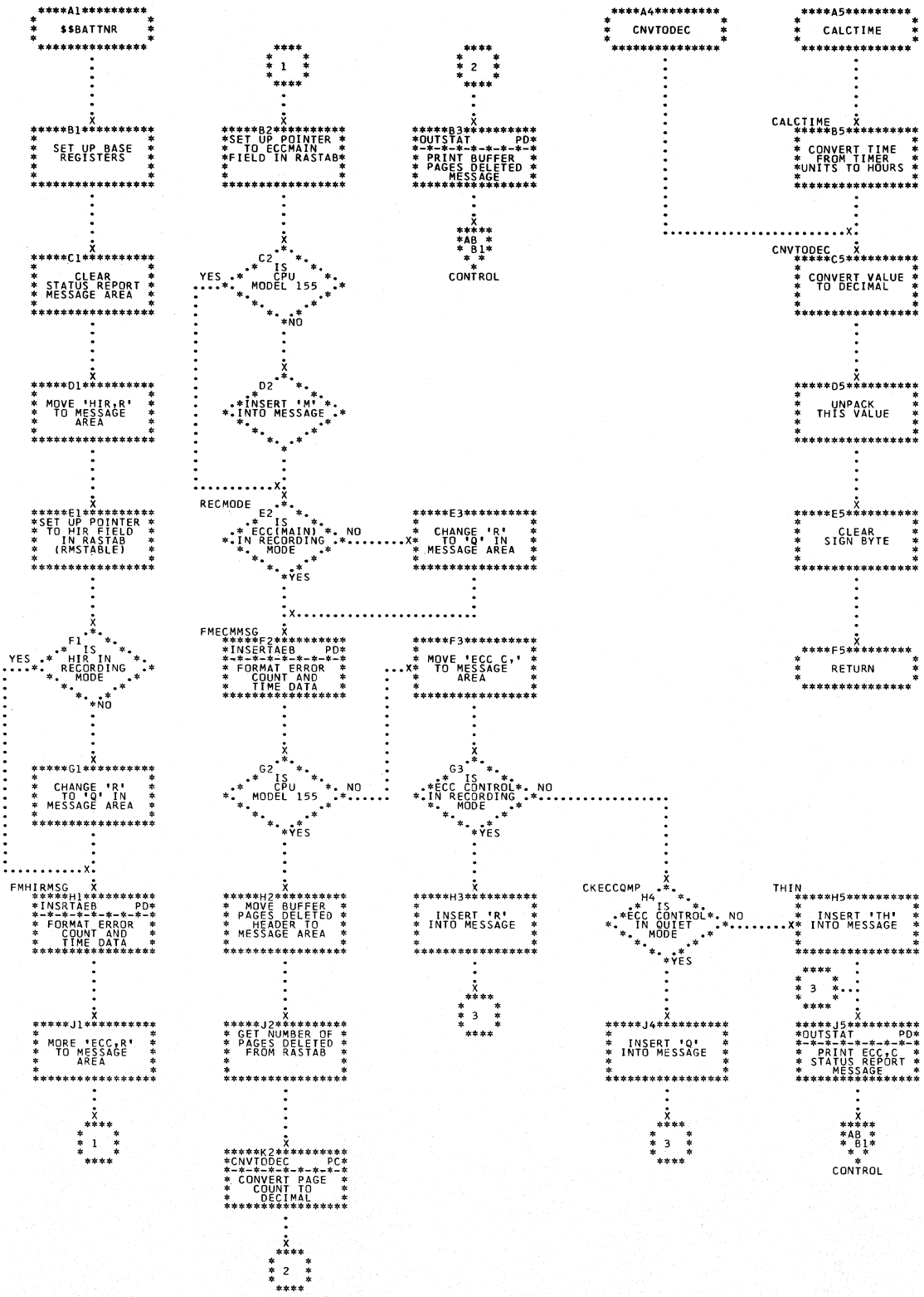


Chart PD. $$BATTNR - MODE Command Status Report Processor (Part 2 of 2)
Refer to Chart 07.

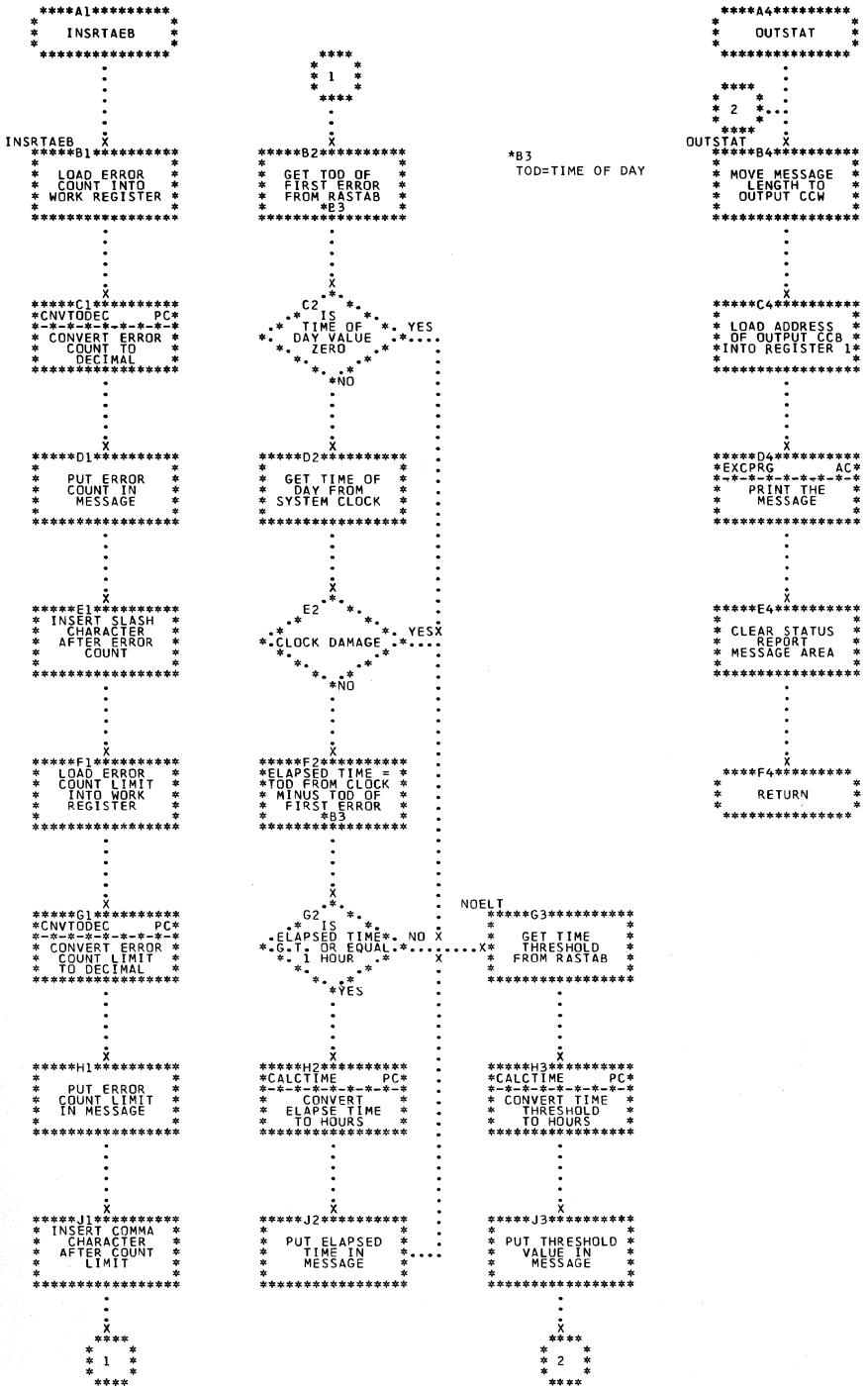


Chart PE. $$BATTNS - MODE Command Validity Checker (Part 1 of 2)
Refer to Chart 07.

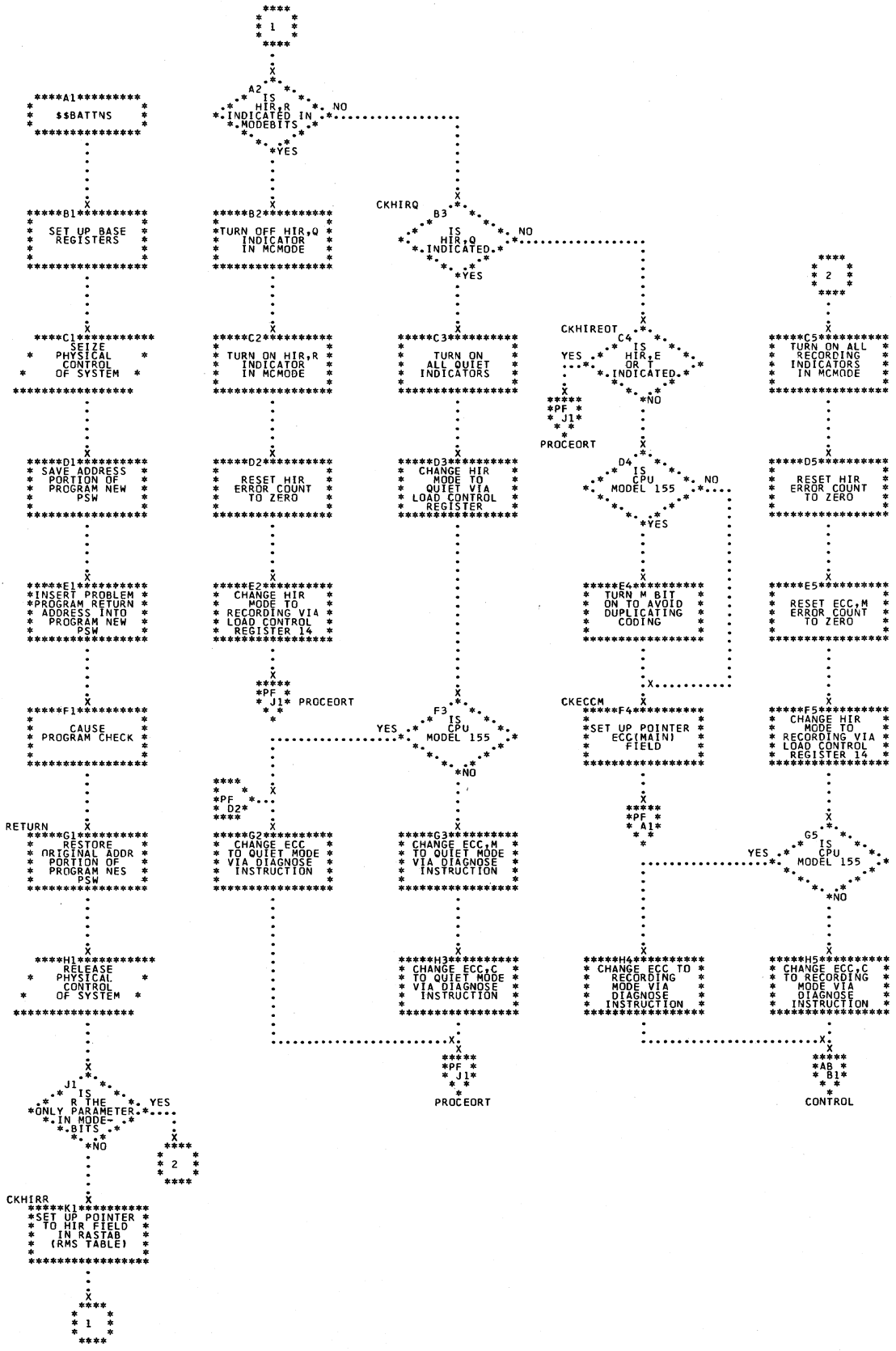
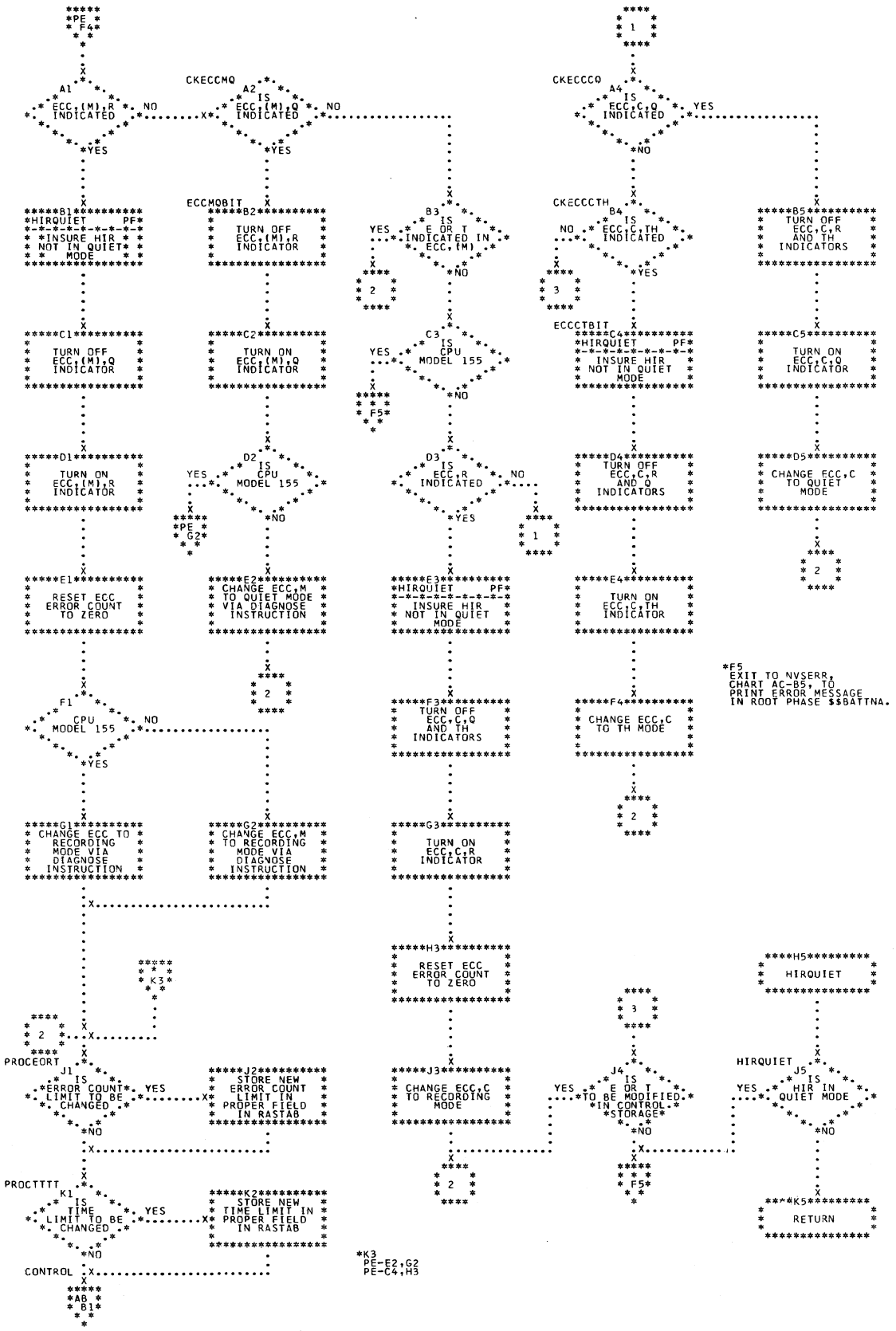


Chart PF. $$BATTNS - MODE Command Validity Checker (Part 2 of 2)
Refer to Chart 07.



APPENDIX A: LABEL LIST

Certain labels in this Appendix are defined by superscripts (e.g., $$BDUMPF[1]), which have these meanings:

[1] = Standard System Dumps

[2] = Translating System Dumps

[3] = Listing only

Label	Phase	Chart
A	$$BEOJ2	FG
ADDLST	$$BTERM	HB

The pointer from FAVP byte, which was pointing to the first available JIB before this terminating phase began, is put in the chain byte of the last-dequeued JIB (using register 8 as an intermediate storage). The second byte of the LUB has a pointer to the first JIB associated with that LUB; this pointer is now put in the FAVP byte.

ADDRLP	$$BATTNH	BG
ALLOC	$$BATTNE	BA
ALT	$$BATTNI	CB
ALTER	$$BDUMPF[1]	MD
	$$BDUMPB	MK
	$$BDUMPD	MQ
	$$BPDUM1	ND

Switch set either to enter or to bypass the SPECIAL routine. To understand the use of this SPECIAL routine, which blanks out printing of the first two storage data words, consider this example: the beginning address of a problem program or parameter dump falls between 3F8 and 3FF. To begin print of the dump at the nearest lower double-word boundary, it is necessary to blank out data from 3F0 through 3F7.

For a parameter dump, if the desired starting address is 3FC, an additional calculation is made to determine the number of additional blanks needed. This number is put in register 2 by the $$BPDUMP monitor phase and passed to the phase actually performing the dump. This switch is, therefore, a NOP only once (if needed) at the outset of the problem program portion of a dump or a parameter dump, and is normally set to a branch.

ALTER1	$$BDUMPF[1]	MD
	$$BDUMPB	MK
	$$BDUMPD	MQ
	$$BPDUM1	ND

This routine puts an extra two spaces between groups of four words, making a total of three spaces. This makes the dump easier to read, because storage locations such as 1B0, 1C0, 1D0, etc. stand out clearly. The word counter, register 0, used for this grouping function is reset to 4.

ALTER2	$$BDUMPF[1]	MD
	$$BDUMPB	MK
	$$BDUMPD	MQ
	$$BPDUM1	ND

This routine increments register 6, which points to locations along the print line where data information is being assembled. Register 6 is incremented by nine for each new word to be printed: one for the space between words and eight for the print positions of each unpacked word.

ALTER3	$$BDUMPF[1]	MD
	$$BDUMPB	MK
	$$BDUMPD	MQ

Switch to enter or bypass instructions that create two blank spaces between the location counter and first word of storage data. Switch is set to branch, except when preparing the first word of each new print line.

AMOUNT	$$BDUMP[2]	KC
ANAERR	$$BATTNC	AG
AP	$$BEOJ4	FM
APGO	$$BEOJ4	FM
ARCANCEL	$$BEOJ	FA
ASGCHG	$$BATTNP	EQ
ASGLST	$$BATTNJ	CM
ASGPUB	$$BATTNI	CB
ASSGN	$$BATTNI	CA
ASSGNLOG	$$BATTNG	BF
ATTNH	$$BATTNH	BG

B	$$BEOJ2	FG
	$$BEOJ2A	FJ
BAL	$$BPDUM1[1]	NB
BAL1	$$BDUMP[1]	LG
	$$BDUMPD	MM
	$$BPDUMP	NA

In $$BDUMPD: Routine to blank out initializing instructions of this phase so this portion of storage can be used as an I/O output area.

In $$BPDUMP: Sense data is tested for file-protect condition if SYSLST is a

tape unit. If it is protected, the dump cannot be taken, and this program phase returns to supervisor for selection of next task. If not protected, B-transient $$BPDUM1 is fetched to perform the actual parameter dump.

BALR14	$$BEOJ1	FF
	$$BPSW	JE
BATCH	$$BATTNG	BE
BESTVE	$$BESTVE	GF
BGCALC	$$BATTNG	BF
BGJOB	$$BDUMPD[1]	MM
BLANKER	$$BPDUMP[2]	LE
BLANKS	$$BDUMPD[1]	MQ

Blanks are used to blank out the unneeded high-order positions of the printline area when the registers and user's part of the communications region are printed.

BLNK2	$$BDUMPF[1]	MD
	$$BDUMPD	MQ
BLNKLD	$$BATTND	AM
BLNKST	$$BDUMPB[1]	MK
	$$BDUMPD	MQ

Switch that determines if BLANKS instruction will be used. Switch set to branch except under conditions given in the BLANKS label.

BLNKLD	$$BATTND	AM
BTLOOP	$$BATTNA	AB

Beginning of a table lookup, in the branch vector table, that finds the appropriate B-transient required for further processing.

BTSTCR	$$BDUMPB	ML
	BDUMPD	MR
	$$BDUMPF[1]	ME
	$$BPDUM1	NC

Branch and link to TSTCOR subroutine is followed by comparing characters of the next line to be printed with those of the line just printed. If the next line is identical, a switch is set to branch to the CLRLIN routine that suspends printing the identical line and prints---SAME---instead.

BUMP	$$BCCHHR	NJ
BYPASS	$$BSDRUP	NG
BYPSLOG	$$BATTNJ	CP
C	$$BEOJ2	FG
	$$BEOJ2A	FJ
CALCTIME	$$BATTNR	PC
CALLEOJ	$$BDUMP[2]	KC
CALLPHSE	$$BATTNQ	PA
CANCEL	$$BATTNC	AG
CANCLB	$$BATTNC	AG
CASERR	$$BATTNI	CA
	$$BATTNP	EP
CAUSE1	$$BEOJ1	FE
CAUSE2	$$BEOJ2	FG
	$$BEOJ2A	FJ
CESWITCH	$$BDUMPB[2]	KE
	$$BDUMPD	KK
CHAIN	$$BTERM	HB
CHGSTT	$$BATTNF	BD
CHKASG	$$BATTNM	EF
CHKF1	$$BATTNJ	CK
CHKF2	$$BATTNJ	CK
CHKFGA	$$BATTNH	BH
CHKFUA	$$BATTNI	CJ
CHKJIB	$$BATTNI	CE
CHKMOD	$$BATTNI	CA
CHKNXC	$$BATTNI	CG
	$$BATTNK	DB
CHKOWN	$$BATTNN	EG
	$$BATTNP	EP
CHKPRN	$$BATTNE	BA
CHKPUB	$$BATTNI	CD
CHKRNG	$$BATTNE	BB
	$$BATTNI	CG
	$$BATTNK	DB
CHKRNG1	$$BATTNK	DB
CHKSLH	$$BATTNK	DA
CHKSTT	$$BATTNA	AB
CHKUA	$$BATTNJ	CK
CKBF12	$$BATTNC	AG
CKECCCQ	$$BATTNS	PF
CKECCM	$$BATTNS	PE
CKECCMQ	$$BATTNS	PF
CKEEEE	$$BATTNQ	PB
CKFORC	$$BATTNQ	PA
CKFORECC	$$BATTNQ	PA
CKFORHIR	$$BATTNQ	PA
CKFORM	$$BATTNQ	PA
CKFORQUI	$$BATTNQ	PA
CKFORREC	$$BATTNQ	PA
CKFORT	$$BATTNQ	PB
CKFORTH	$$BATTNQ	PA
CKF1F2	$$BATTNB	AE
CKHIREOT	$$BATTNS	PE
CKHIRQ	$$BATTNS	PE
CKHIRR	$$BATTNS	PE
CKLT8	$$BATTNQ	PB
CKNDAR	$$BATTNF	BD
CKNDCH	$$BATTNI	CB
CKNXJB	$$BATTNI	CE

Exit point to the scan JIB subroutine, SCNJIB. The subroutine is entered to reset JBSLUB according to any JIB chained to the logical unit.

CKPBUA	$$BATTNJ	CK
CKPIBFLG	$$BATTNG	BF
CKSCST	$$BATTNE	BA
CLEAR	$$BDUMP[2]	KA
	$$BDUMP[1]	LF
CLEARCTR	$$BESTVD	GD
CLEAROPT	$$BDUMPB[2]	KD
	$$BDUMPD	KJ

CLI	$$BEOJ4	FM
CLRLIN	$$BDUMPF[1]	MF
	$$BDUMPB	ML
	$$BDUMPD	MR
	$$BPDUM1	NC
CMNWLM	$$BATTNF	BC
CMPBPT	$$BATTNI	CE

Test for identical PUB pointers. Identical PUB pointers indicate that another LUB is assigned to the physical unit pointed to by the LUB just unassigned. (See label UNPA in this list.) If there is no other LUB with a matching PUB, the ownership flag of the PUB indicated by the LUB in LBSLUB is reset so that the PUB is not assigned to any LUB.

CMPCOR	$$BDUMPF[1]	MF
	$$BDUMPB	ML
	$$BDUMPD	MR
	$$BPDUM1	ND

Register 5 contains the highest storage location that prints for any single line. Register 5 is compared to register 8 (which contains the upper storage limit of the dump) to see if limit of dump will be exceeded should the entire line be printed. If register 5 is higher than register 8, the value in register 8 is then loaded into register 5 and the printing ceases at the dump limit.

CNCLIN	$$BATTNC	AG
CNCLME	$$BATTNC	AG
CNCLTEST	$$BEOJ	FC
CNLRTN	$$BATTNC	AG
CNUNCO	$$BATTNK	DD
	$$BATTNM	EF
CNUNCO1	$$BATTNK	DD
CNVBCD	$$BATTND	AM
CNVTODEC	$$BATTNR	PC
COMM	$$BDUMPB[1]	MH
	$$BDUMPD	MN
COMMRGN	$$BDUMPB[2]	KG
	$$BDUMPD	KJ
COMPHI	$$BDUMPB[2]	KG
	$$BDUMPD	KM
	$$BPDUMP	LD
COMPLO	$$BDUMPB[2]	KG
	$$BDUMPD	KM
	$$BPDUMP	LD
CONCAT	$$BATTNK	DD

Entry point to a subroutine used to:

1. Read the second half of a statement.
2. Join the first and second parts of a statement forming a single statement. (This operation is called concatenation.)
3. Reset the address of the operand in the I/O area named BUFFER.
4. Reset the length of the operand.

CONT	$$BEOJ3	FL
CONTROL	$$BATTNA	AB
CONTROL1	$$BATTNA	AB
CONTSCAN	$$BTERM	HC
CONVERT	$$BDUMPB[2]	KG
	$$BDUMPD	KM
	$$BPDUMP	LD
CORE	$$BDUMPF[1]	MB
	$$BDUMPB	MH
	$$BDUMPD	MN

Register 7, containing the beginning storage address of the problem program area, is tested for proper boundary alignment. If register 7 is not on a boundary that is a multiple of 16, it is adjusted to a boundary such as 1B0, 1C0, 1D0, etc, and the switch at ALTER is set to NOP. See label ALTER.

CORE1	$$BDUMPF[1]	MB
	$$BDUMPD	MN
CORE2	$$BDUMPF[1]	MB
	$$BDUMPB	MH
	$$BDUMPD	MP
CORE25	$$BDUMPF[1]	MB
CORE3	$$BDUMPF[1]	MB
	$$BDUMPB	MH
	$$BDUMPD	MN
CORE35	$$BDUMPF[1]	MB
	$$BDUMPB	MH
	$$BDUMPD	MN
COZERO	$$BDUMPB[2]	KG
	$$BDUMPD	KM
	$$BPDUMP	LD
CRTBLD	$$BATTNE	BA
CUAPNX	$$BATTNJ	CK

D	$$BEOJ2	FG
	$$BEOJ2A	FJ
DATE	$$BPDUM1[1]	ND
DDSWTCH1	$$BDUMPB	KE
	$$BDUMPD	KK
DDSWTCH2	$$BDUMPB	KE
	$$BDUMPD	KK
DEQUEUE	$$BTERM	HB

The JIB pointer from the LUB is temporarily stored at label FRLSTBEG. The JIB pointed at by the LUB is addressed, and its first 3 bytes are zeroed. The chain byte (4th byte) of the JIB is checked for additional JIBs in the chain; if there are any, the first 3 bytes of these JIBs are zeroed until the end of the chain is reached.

DEQUEUED	$$BTERM	HB
DEV2311	$$BCCHHR	NH
DISABLE	$$BEOJ3	FL
DISKRT	$$BPDUMP[1]	NA
DKTYPE	$$BEOJ	FC
	$$BEOJ1	FF
	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BILSVC	JB
	$$BPSW	JE

	$$BPCHK	JG
DLAB	$$BATTNL	DJ
DLBL	$$BATTNK	DF
DLBOUT	$$BATTNK	DG
	$$BATTNL	DJ
DNEERR	$$BATTNI	CD
DONE	$$BEOJ3	FL
	$$BTERM	HA
DOP34	$$BATTNL	DK
DSKRT	$$BDUMP[1]	LF
DTCHAT	$$BATTNA	AA
DTCHSZ	$$BATTNA	AA
DTINUN	$$BATTNA	AC
DTSTCR	$$BDUMPB[1]	MF
DUMP	$$BDUMPD[1]	MM
ECCCTBIT	$$BATTNS	PF
EDITCORE	$$BPDUMP[2]	LC
EDITLBL	$$BDUMPB[2]	KE
	$$BDUMPD	KK
END1	$$BDUMPB[2]	KF
	$$BDUMPD	KL
	$$BPDUMP	LC
ENDJOB	$$BDUMPB[2]	KE
	$$BDUMPD	KK
ENDLIN	$$BDUMPF[1]	ME
	$$BDUMPB	ML
	$$BDUMPD	MR
	$$BPDUM1	NC
ENDLIN1	$$BDUMPF[1]	MF
	$$BDUMPD	MR
ENDMOD	$$BPDUM1	NC
ENDPUB	$$BTERM	HA
ENT1	$$BEOJ	FD
ENT2	$$BEOJ	FD
EOJEST00	$$BESTVD	GD
EOJEST01	$$BESTVD	GD
EOJEST02	$$BESTVD	GD
EOJEST03	$$BESTVD	GD
EOJEST04	$$BESTVD	GE
EOJEST05	$$BESTVD	GE
EOJEST06	$$BESTVD	GE
EOJEST07	$$BESTVD	GE
EOJEST08	$$BESTVD	GE
EOJEST1	$$BESTVD	GD
EOJEST11	$$BESTVD	GD
EOJEST2	$$BESTVD	GD
EOJXIT	$$BESTVD	GD
EOJS1	$$BEOJ	FB
EOJSTEP	$$BEOJ	FB
EOVMV	$$BPDUM1[1]	NE
EQUALSW	$$BDUMPB[2]	KF
	$$BDUMPD	KL
	$$BPDUMP	LC
ERR1	$$BESTVC	GC
	$$BESTVF	GG
ERR2	$$BESTVF	GG
ERR3	$$BESTVC	GC
ERRORRTN	$$BATTNQ	PB
ERROUT	$$BESTVC	GC
ERRRTN	$$BATTNA	AC
ERRRTN1	$$BATTNH	BH
ERRSET	$$BESTVC	GC
ESOPXIT	$$BESTVC	GC
	$$BESTVF	GG
ESOP002	$$BESTVC	GC
ESOP02	$$BESTVC	GC
ESOP06	$$BESTVC	GC
ESOP07	$$BESTVC	GC
ESOP08	$$BESTVC	GC
ESOP09	$$BESTVC	GC
ESOP092	$$BESTVC	GC
ESOP10	$$BESTVC	GC
ESOP12	$$BESTVC	GC
ESTV01	$$BESTVA	GA
ESTV02	$$BESTVA	GA
ESTV06	$$BESTVA	GA
ESTV08	$$BESTVA	GA
ESTV10	$$BESTVA	GA
ESTV12	$$BESTVA	GA
ESTV14	$$BESTVA	GA
ESTV16	$$BESTVA	GA
ESTV20	$$BESTVA	GA
ESXIT	$$BESTVA	GA
EXCPRG	$$BATTNA	AC
EXEC	$$BATTNM	EA

Entry point to the execute (EXEC) processor. This phase is the last processing phase of the foreground initiator. The foreground program will be loaded when this phase has finished executing and when the foreground program has been chosen by the task selection mechanism of the supervisor.

EXEC1	$$BATTNM	EB
	$$BESTVE	GF
	$$BPDUMP[2]	LA
	$$BPDUMP[1]	NA
	$$BPDUM1	NC
	$$BSYSWR	NF
EXIT	$$BCCHHR	NJ
EXIT1	$$BCCHHR	NJ
EXTINT	$$BATTNB	AE
EXTNT	$$BATTNO	EH
F1CALC	$$BATTNG	BF
F2CALC	$$BATTNG	BF
F2CALC1	$$BATTNG	BF
FDDKCODE	$$BATTNK	DF
	$$BATTNL	DJ

FDEOJ	$$BDUMPF[1]	ME
FDEOJ1	$$BDUMPF[1]	ME
FDK2	$$BATTNK	DF
FDKIJ1	$$BATTNK	DF
FDKIJ2	$$BATTNK	DF
FDKTDAT	$$BATTNK	DE
FDKTDAT1	$$BATTNK	DE
FDKTDAT2	$$BATTNK	DE
FDKTID	$$BATTNK	DE
FDKTV	$$BATTNK	DA
FDKTV1	$$BATTNK	DA
FDKTV2	$$BATTNK	DA
FDKTVNM	$$BATTNK	DA
FEBIN1	$$BATTNO	EL
FESEQ	$$BATTNO	EL
FESEQCK	$$BATTNO	EJ
FESPLIT	$$BATTNO	EK
FESPLIT1	$$BATTNO	EK
FESPLIT2	$$BATTNO	EK
FESPLIT4	$$BATTNO	EK
FESYSX2	$$BATTNO	EH
FESYSX3	$$BATTNO	EH
FETCH	$$BDUMP	LG
	$$BESTVB	GB
FETCH07	$$BDUMPB[2]	KH
	$$BPDUMP	LE
FETCH1	$$BEOJ4	FM
FETINSR0	$$BATTNK	DC
FETINSR1	$$BATTNK	DC
FETINSRT	$$BATTNK	DC
FETK	$$BATTNO	EJ
FETKNO	$$BATTNO	EJ
FETKNO1	$$BATTNO	EK
FETKTK	$$BATTNO	EM
FETKTK1	$$BATTNO	EM
FETKTK2	$$BATTNO	EM
FETYPE	$$BATTNO	EH
FETYPE1	$$BATTNO	EJ
FETYPED	$$BATTNO	EJ
FETYPEI	$$BATTNO	EJ
FETYPEI4	$$BATTNO	EJ
FEVSER	$$BATTNO	EH
FEVSER1	$$BATTNO	EH
FEVSER2	$$BATTNO	EH
FEVSER3	$$BATTNO	EH
FEVSER4	$$BATTNO	EH
FG	$$BEOJ	FB
FG1	$$BTERM	HC
FGHED	$$BPDUM1	ND
FGJOB	$$BEOJ1	FE
	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BILSVC	JA
	$$BPSW	JD
	$$BPCHK	JF
	$$BDUMP[1]	LG
	$$BPDUMP	NA
FGLST	$$BEOJ	FC
	$$BEOJ1	FF
	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BILSVC	JB
	$$BPSW	JE
	$$BPCHK	JG
FGNAME	$$BPDUM1	ND
FGTAPE	$$BILSVC	JB
FHMIRMSG	$$BATTNR	PC
FINISH	$$BATTNM	ED
FIRST	$$BEOJ3	FL
FIX	$$BPDUM1[1]	NE

When word counter reaches zero, two extra blanks are inserted between words so that locations such as 1B0, 1C0, 1D0, etc, will stand out, thus making the dump easier to read.

FIX6	$$BPDUMP[2]	LA
FIX7	$$BPDUMP[1]	NA
FLPTSW	$$BDUMPB[2]	KD
	$$BDUMPD	KJ
	$$BPDUMP	LB
FPSW	$$BDUMPF[1]	MB
	$$BDUMPB	MH
FSCAN	$$BATTNK	DC
	$$BATTNO	EL
FSCAN1	$$BATTNK	DC
	$$BATTNO	EL
FTCHBJ5	$$BEOJ4	FM
FTEND	$$BATTNK	DC
GETBYTE	$$BTERM	HC
GETKEY	$$BATTNI	CJ
GETMPSST	$$BDUMPB[1]	MH
GETNXT	$$BTERM	HC
GO	$$BEOJ4	FM
GPR	$$BDUMPB[1]	MH
GTNXJB	$$BATTNI	CE

Continues search for LUBs with a PUB pointer that matches the pointer in LBSLUB. However, the search is within the JIB table.

GTNXLB	$$BATTNI	CE

Start of a repetitive sequence of code to get each LUB of a given class and compare its PUB pointer with the PUB pointer of the LUB in LBSLUB.

GTNXOP	$$BATTNE	BA
HALTIO	$$BEOJ3	FL
HALTIO1	$$BEOJ3	FL
HEADER	$$BDUMPB[2]	KH
	$$BDUMPD	KN
	$$BPDUMP	LE
HERE	$$BSYSWR	NF
HEXCON	$$BATTNI	CG

HOLD	$$BATTNP	EQ

Entry point to the HOLD processor. This routine sets a switch in the appropriate PIB assign flag. This switch can be interrogated later by the job control program.

IDSERR	$$BATTNI	CD
IGNORE	$$BATTNC	AH
IJBDMB33	$$BDUMPB[1]	MG
IJBDMD33	$$BDUMPD[1]	MM
IJBDMF33	$$BDUMPF[1]	MA
IJBDMB35	$$BDUMPB[2]	KD
IJBDMD35	$$BDUMPD[2]	KJ
IJBDMP35	$$BDUMP[2]	KA
IJBEJ33	$$BEOJ	FA
IJBEJ35	$$BEOJ2	FG
IJBEJ133	$$BEOJ1	FE
IJBEJ235	$$BEOJ2A	FJ
IJBEJ335	$$BEOJ3	FL
IJBEJ435	$$BEOJ4	FM
IJBEJ533	$$BEOJ5	FN
IJBPCK33	$$BPCHK	JF
IJBPD133	$$BPDUM1[1]	NB
IJBPDP33	$$BPDUMP[1]	NA
IJBPDP34	$$BPDUMP[2]	LA
IJBPSW33	$$BPSW	JD
IJBSVC33	$$BILSVC	JA
IN02	$$BESTVE	GF
IN04	$$BESTVE	GF
IN06	$$BESTVE	GF
IN08	$$BESTVE	GF
IN10	$$BESTVE	GF
IN14	$$BESTVE	GF
IN142	$$BESTVE	GF
INDSEQ	$$BATTNL	DL
INIT	$$BDUMPD	MM
INITDUMP	$$BDUMP[2]	KA
INITL	$$BATTNM	EE
INITOPR	$$BATTNG	BF
INNXEN	$$BATTNE	BA
INSRTAEB	$$BATTNR	PD
INTERR	$$BEOJ	FA
ISCKSQ	$$BATTNL	DL
ISTYP4	$$BATTNL	DL
ITERATE	$$BEOJ3	FL
JACOM	$$BEOJ4	FM
KLEER	$$BDUMPB[2]	KH
	$$BDUMPD	KN
	$$BPDUMP	LE
LA	$$BILSVC	JB
	$$BPCHK	JG
LABMV	$$BSYSWR	NF
LABMV1	$$BSYSWR	NF
LABMV2	$$BSYSWR	NF
LABMV3	$$BSYSWR	NF
LANXJB	$$BATTNJ	CM
LASTPUB	$$BEOJ3	FL
LAXERR	$$BATTNK	DH
	$$BATTNL	DL
	$$BATTNM	EB
	$$BATTNO	EH
LBLOUT	$$BATTNK	DH

Entry point to the subroutine used to output the label information that has been accumulated in the I/O area, BUFFER. The subroutine:

1. Sets length information in the write and verify CCWs.
2. Determines if space is available on the label track within SYSRES.
3. Updates the disk address if necessary.
4. Checks to ensure label area extents on SYSRES are not exceeded.
5. Sets up the seek address and CCB.
6. Branches to the I/O subroutine (EXCPRG) to write and verify the label information on SYSRES. See Appendix D for format of labels on SYSRES.

LBLOUT1	$$BATTNK	DH
	$$BATTNM	EB
LBLTYP	$$BATTNO	EN
LBTOUT	$$BATTNO	EN
LISTIO	$$BATTNJ	CK
LOAD	$$BDUMPD[1]	MP
LOADBPSW	$$BEOJ2	FG
	$$BEOJ2A	FJ
LOADR5	$$BDUMPB[2]	KD
	$$BDUMPD	KJ
	$$BPDUMP	LA
LOADTAB	$$BDUMPB[2]	KG
	$$BDUMPD	KM
	$$BPDUMP	LD
LOG	$$BATTNC	AJ
	$$BCCHHR	NH
	$$BTERM	HD
LOGEXT	$$BATTNH	BH
LOGGER	$$BATTNH	BH
	$$BATTNJ	CL
	$$BEOJ1	FE
	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BPSW	JD
LOGLIST	$$BEOJ	FC
LOGLNE	$$BATTNJ	CP
LOOP	$$BEOJ	FB
	$$BEOJ5	FN
LST	$$BDUMPF[1]	MD
	$$BDUMPB	MK
	$$BDUMPD	MQ

	$$BPDUM1	NE

When last word of a printline has been unpacked and printed, this switch is used to return from REGPNT subroutine to prepare the next line. For printing of registers and user's communications region, LST is a NOP that permits entry to a routine that blanks out unneeded high-order positions of the printline.

LST1	$$BPDUM1[1]	NE
LSTASG	$$BATTNJ	CP
LSTAUN	$$BATTNJ	CM

Entry point to the subroutine that lists the assignments for either F1 or F2 system class and programmer class units. The subroutine sets up primary and secondary headers, calls the LUB scanning subroutine and the JIB scanning subroutine, and calls the final output subroutine.

LSTBG	$$BATTNJ	CL
LSTBGF	$$BATTNJ	CL
LSTBUN	$$BATTNJ	CM
LSTBUN1A	$$BATTNJ	CM
LSTBUN2	$$BATTNJ	CM
LSTF1	$$BATTNJ	CL
LSTF2	$$BATTNJ	CL
LSTLN	$$BDUMPF[1]	ME
	$$BDUMPB	ML
	$$BDUMPD	MR
	$$BPDUM1	NC

The location counter, register 7, is set and translated to identify the storage locations being printed on each line of the dump. This label is also used to enter the PRNTLN subroutine on a last line condition, thereby bypassing the TSTCOR subroutine.

LSTPRG	$$BATTNJ	CN
LSTPRG1	$$BATTNJ	CN
LSTSTD	$$BATTNJ	CN
LSTUA	$$BATTNJ	CP

Program switch set to NOP when a UA operand is found. The switch is reset to a branch when the header 'UNASSIGNED' has been printed.

LUBCHECK	$$BCCHHR	NH
MAIN1	$$BPDUMP[1]	NA

The starting address for the parameter dump, entered in register 6, is shifted right double logical 4 positions so that any value not a multiple of 16 is now in register 7. If value in register 7 is now zero, it indicates that the starting value in register 6 is on a double-word boundary. Register 6 is then restored by shifting left to the next lower doubleword boundary nearest the value specified by the dump parameter (label FIXT). If register 7 was not zero when tested, the value now in it is used to calculate the number of blank positions needed so printout starts at desired starting byte.

MAIN2	$$BPDUMP[1]	NA

The upper parameter address is incremented by a word length and tested against system's main storage capacity to see if requested dump is a valid address within core. If not, the upper storage limit is put in register 8 to impose a valid dump end limit.

MAINTSK	$$BEOJ3	FL
MAP	$$BATTND	AK
MARK	$$BEOJ	FA
MARK1	$$BEOJ	FA
MICR	$$BEOJ	FA
MICR1	$$BEOJ	FA
MICRDEV	$$BEOJ	FA
MKASGN	$$BATTNI	CB

Entry point to a routine that makes the actual assignment during ASSGN processing. The assignment is made by:

1. Establishing the PUB pointer in the LUB.
2. Setting the ownership byte in the PUB.
3. Setting the mode byte in the PUB. (For tape devices only.)

MOD	$$BTERM	HA
MODEOK	$$BCCHHR	NK
MODRST	$$BATTNI	CA
MOVE	$$BDUMPD[1]	MP

Current address taken from the Disk Information Block (DIB) for the appropriate symbolic disk device is put in output area to serve as the count ID information when count, key, and data are written. The current address record number is then reduced by one and put in the search CCW for writing the first dump record.

MOVE32	$$BDUMPB[2]	KF
	$$BDUMPD	KL
	$$BPDUMP	LC

MOVEREG	$$BDUMPB	KD
	$$BDUMPD	KJ
MOVESUB	$$BEOJ1	FE
	$$BEOJ2	FG
	$$BEOJ2A	FJ
MOVLOP	$$BATTNM	ED

Start of a repetitive sequence of code to move the last two routines of the EXEC processor to the main storage area occupied by the root phase, $$BATTNA. The root phase resides in the logical transient area of main storage. The two routines are moved 256 bytes at a time. The last time the move is executed, the remaining bytes (less then 256) are moved to the logical transient area.

MOVRTN	$$BATTNM	ED

Entry point to the subroutine that:

1. Moves any label information from the temporary label storage area to the label storage area.
2. Clears the remainder of main storage to initialize it for the foreground program being initiated.

MOVSW	$$BPDUMP[2]	LE
MPSSYS	$$BPDUMP[2]	LB
MSG	$$BATTNB	AE
MSG1	$$BESTVE	GF
MT	$$BILSVC	JC
	$$BPCHK	JF
MTAPE	$$BTERM	HC

The device type from the PUB table entry for the device is examined. If the device is not a tape drive, the PUB scan proceeds to the next entry in the table; if it is a tape drive, the Tape Error Block (TEB) for that particular drive is addressed and checked for any record of tape errors. If this tape drive has had no errors, the PUB scan resumes, and the next device in the PUB table is investigated.

MTRSVD	$$BATTNK[3]	
	$$BATTNL[3]	
	$$BATTNO[3]	

A one-byte switch used when the file type is sequential disk (SD):
Bit 0 = 1: Look-ahead flag for LIOCS.

Bit 1 = 1: Last extent for file.

Bit 2: Not used.

Bit 3 = 1: No extent dequeue.

Bit 4 = 1: Extent limits omitted.

Bit 5 = 1: Extent limits converted to address.

Bits 6, 7: Not used.

MVBLNK	$$BDUMPD[1]	MQ
	$$BPDUM1	NE
MVC	$$BEOJ2	FG
	$$BEOJ2A	FJ
	$$BILSVC	JB
	$$BPCHK	JG
MVCLRT	$$BATTNM	ED
MVI	$$BTERM	HA

A detach flag is posted in the PIB for the terminated program. The portion of core occupied by this program is now available for overlay. An End-of-Termination switch is set in the PIBPUBAS flag byte, an SVC 22 releases control of the system from this program, and an SVC 11 returns the system to the Task Selection routine of the supervisor.

NAMED	$$BEOJ1	FE
NASERR	$$BATTNI	CB
NDCHFD	$$BATTNI	CB
NDSCAN	$$BATTNA	AD
	$$BATTNJ	CN
	$$BATTNK	DE
NDSCAN1	$$BATTNK	DE
NDTERR	$$BATTNI	CB
	$$BATTNM	EE
	$$BATTNP	EP
NEWXTN	$$BATTNL	DN
	$$BATTNO	EK
NEXTBYTE	$$BDUMPB[2]	KG
	$$BDUMPD	KM
	$$BPDUMP	LD
NEXTPARA	$$BATTNQ	PB
NJPERR	$$BATTNI	CB
NLSERR	$$BATTNL	DJ
	$$BATTNO	EL
NLUERR	$$BATTNI	CC
	$$BATTNK	DD
	$$BATTNM	EF
NOAP	$$BDUMP[2]	KA
	$$BDUMP[1]	LF
	$$BEOJ	FA
	$$BEOJ1	FE
	$$BEOJ2	FG
	$$BEOJ2A	FJ
	$$BEOJ4	FM
	$$BILSVC	JC
	$$BPCHK	JF
NOAP1	$$BATTNG	BE
NOCHNG	$$BDUMP[1]	LG

Routine used, when a foreground program is to be dumped, to identify the physical I/O device associated with SYSLST. The type of device determines which B-transient dump program will be fetched to perform the actual dump.

NODCUX	$$BATTNL	DN
	$$BATTNO	EK
NOELT	$$BATTNR	PD
NOESTV	$$BEOJ4	FM
NOFG	$$BEOJ	FC

NOHOLD	$$BEOJ5	FN
NOLBPR	$$BATTNM	EC
NOLOG	$$BATTNC	AJ
NOP	$$BTERM	HC
	$$BILSVC	JB
	$$BPCHK	JD

In $$BTERM: Switch to enter or bypass the routine that prints headings prior to logging the Tape Error Block (TEB) statistics. Because only one set of headings is needed, this routine is used only for the first TEB statistics logged. Thereafter, this routine is bypassed by making this switch an unconditional branch.

In $$BILSVC and $$BPCHK: After first line of message has been output, this switch is set to branch. The next time through, the second line of the message is output and the branch causes the transient $$BDUMP to be fetched.

NOPTODR2	$$BATTNA	AA
NOQTAM	$$BEOJ	FB
NOTASG	$$BATTNJ	CP
NOTBG	$$BPCHK	JF
NOTCR	$$BATTNI	CD
NOTEST	$$BDUMPF[1]	MA
	$$BDUMPB	MG
	$$BDUMPD	MM

An area of storage used for phase initialization instructions is blanked out to be used as an output area for the dump. If needed, a branch is taken past the end of the cleared area to the next instruction.

NOTIME	$$BDUMP	KA
	$$BDUMPB	KH
	$$BDUMPD	KN
NOTSEQ	$$BATTNL	DL
NOTZERO	$$BDUMPB[2]	KH
	$$BDUMPD	KN
	$$BPDUMP	LE
	$$BDUMPB[1]	MJ
NOUNIT	$$BATTNJ	CN
NULLOG	$$BATTNA	AC
NUMCON	$$BATTNI	CG
	$$BATTNK	DD
	$$BATTNO	EM
NUMLOP	$$BATTNI	CG
NUMSYS	$$BATTNI	CG
NVAERR	$$BATTNF	BC
NVSERR	$$BATTNA	AC
NWPBPT	$$BATTNI	CB
NXPBNT	$$BATTNF	BD
NXTLUB	$$BATTNJ	CM

ONLIST	$$BEOJ1	FE
	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BILSVC	JA
	$$BPSW	JD
	$$BPCHK	JF

OPNUMB	$$BATTNA	AC
OPRSNT	$$BATTNC	AG
OTHERS	$$BEOJ	FC
OTSERR	$$BATTNK	DB
	$$BATTNL	DH
	$$BATTNM	EA
	$$BATTNO	EJ
OUT	$$BDUMPF[1]	ME
	$$BDUMPB	ML
	$$BDUMPD	MR
	$$BPDUMP	NA

Switch made a NOP when the supervisor portion of dump is completed. During the problem program portion of the dump, the switch permits exit from the dump phase by fetching $$BEOJ when the dump limit is reached.

OUT1	$$BDUMPF[1]	ME
	$$BDUMPB	ML
	$$BDUMPD	MR

If SYSLST is a tape unit, this switch is set to branch to write a tapemark following the record of the last line of the dump.

OUT2	$$BDUMPF[1]	ME
	$$BDUMPB	ML
	$$BDUMPD	MR
OUT3	$$BDUMPF[1]	ME
	$$BDUMPD	MR
OUT3	$$BDUMPF[1]	ME
OUTAR	$$BTERM	HA

Entry point to this program phase. The output area address is loaded into a CCW. Register 13 is loaded as a link register to the unassign routine. The partition of the terminated program is identified as F2 or not F2. If F2, the ownership flags are reset in the PUB entries of devices owned by this program.

If the program is not an F2 program, it must be an F1, because the $$BTERM phase is called to terminate foreground programs. The PIB assign flag byte is checked to see if the cancel switch is on, which indicates cancel occurred while in a terminator phase due to an I/O malfunction. To prevent a repetitive cancel-within-cancel loop, a branch is set in the switch at label LOG to suppress further I/O operations.

OUTCALL	$$BSDRUP	NG
OUTLBL	$$BATTNL	DN
	$$BATTNO	EK
OUTPUT	$$BATTND	AM
	$$BATTNJ	CP
OUTSTAT	$$BATTNR	PD
OVLAY	$$BILSVC	JC
OWNRSH	$$BATTNI	CA

PACKCG	$$BATTNL	DN
	$$BATTNO	EK
PAGHED	$$BDUMPF[1]	MC
	$$BDUMPB	MJ
	$$BDUMPD	MP
	$$BPDUM1	ND
PARCKRTN	$$BATTNQ	PA
PAUSE	$$BATTNC	AJ
PAUSE1	$$BATTNC	AJ
PAUSE2	$$BATTNC	AJ
PCTEST	$$BEOJ	FA
PDUMP	$$BPDUM1[1]	NB
PDUMP1	$$BPDUM1[1]	NB
PDUMP2	$$BDUMPD[1]	MN

Switch set to branch if it is a parameter dump. Bypasses printout of all parts of core, except the area specified in the parameter limits.

PDUMPD	$$BDUMPD[1]	MR
PHYSEIZE	$$BTERM	HA

Further I/O operations are disabled, and an SVC 22 is issued that disables multiprogramming and gives this program control over the system to complete its desired functions until another SVC 22 is issued to release control.

PNFORSVC	$$BILSVC	JC
PNPERR	$$BATTNM	EA
PRCOMPL	$$BTERM	HA
PRGUNT	$$BATTNM	EF
PRINT	$$BEOJ1	FF
	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BILSVC	JC
	$$BPSW	JD
	$$BPCHK	JG
	$$BDUMPB[2]	KH
	$$BDUMPD	KN
	$$BPDUMP	LE
	$$BDUMPF[1]	MC
	$$BDUMPB	MJ
	$$BDUMPD	MP
	$$BPDUMP	NA
	$$BPDUM1	NE
PRINT1	$$BDUMPD[1]	MP

Routine that uses PIOCS to seek, search ID equal, write count, key and data, verify, and wait for completion of the I/O operation.

PRINTER	$$BDUMP[2]	KC
	$$BDUMPB	KD
	$$BPDUMP	LB
	$$BDUMP[1]	LF
PRNTL	$$BDUMPF[1]	MD
	$$BDUMPB	MJ
	$$BDUMPD	MQ
	$$BPDUM1	NE
PRNTLN	$$BDUMPF[1]	ME
	$$BDUMPB	ML
	$$BDUMPD	MR
	$$BPDUM1	NC
PRNTLN1	$$BPDUM1[1]	NC
PROCED	$$BPDUMP[2]	LA
	$$BPDUMP[1]	NA
PROCTTTT	$$BATTNS	PF
PROG	$$BEOJ1	FE
PROGCHK	$$BEOJ	FC
PRTRET	$$BDUMPB[1]	MJ
PUT	$$BEOJ	FD
	$$BDUMP[1]	LG
QTAM	$$BEOJ	FB
QUEST	$$BTERM	HC
QUEUE	$$BEOJ3	FL
RAS1	$$BCCHHR	NH
RASPCRET	$$BCCHHR	NK
RASSY	$$BCCHHR	NH
RDSTMT	$$BATTNA	AC
READ	$$BATTNP	EP
READISK	$$BCCHHR	NH
RECMODE	$$BATTNR	PC
RECSWTCH	$$BTERM	HB
REDUCE8	$$BDUMPB[2]	KF
	$$BDUMPD	KL
	$$BPDUMP	LC
REGPNT	$$BDUMPF[1]	MD
	$$BDUMPB	MK
	$$BDUMPD	MQ
	$$BPDUM1	NE
REGPNT1	$$BDUMPF[1]	MD
	$$BPDUM1	NE
REGPNT5	$$BDUMPF[1]	MD
	$$BDUMPB	MK
	$$BDUMPD	MQ
	$$BPDUM1	NE
REGPNT6	$$BPDUM1[1]	NE
RELOC	$$BILSVC	JA
	$$BDUMPB[1]	MG
	$$BPDUM[1]	NB
RELOCATE	$$BDUMPB[2]	KD
	$$BDUMPD	KJ
	$$BESTVD	GD
RELOCF	$$BDUMPF[1]	MA
RELSE	$$BATTNP	EQ
RESTRT	$$BDUMPB[2]	KF
	$$BDUMPD	KL
RESOP	$$BATTNA	AB
RESSW	$$BTERM	HB

RETML	$$BEOJ	FA
RETURN	$$BATTNI	CB
REVSCN	$$BATTNH	BH
	$$BATTNJ	CL
RLCCB	$$BEOJ	FC
	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BILSVC	JB
	$$BPCHK	JF
RMSWAIT	$$BCCHHR	NJ
RNGTOP	$$BATTNI	CG
	$$BATTNK	DB
RSPPAE1	$$BATTNF	BD
RSPPEA	$$BATTNF	BD
RSTOWN	$$BATTNI	CE
RVRSCN	$$BATTND	AM
SAVEWORD	$$BDUMPB[2]	KH
	$$BDUMPD	KN
	$$BPDUMP	LE
SAVLUS	$$BATTNJ	CM
SCANNO	$$BPDUMP[2]	LD
SCANR1	$$BATTNA	AD
SCANR2	$$BATTNA	AD
SCANR3	$$BATTNA	AD
SCNJIB	$$BATTNI	CF

Entry point to a subroutine that:

- Initializes JBSLUB with the first and last bytes of the JIB chained to the current pseudo LUB entry of JBSLUB.
- Returns immediately to the calling sequence when an end-of-JIB-chain condition is found.

SCNLBS	$$BATTNI	CE

Entry point to a subroutine that:

- Returns, sequentially, each LUB entry in a given class to the calling routine.
- Returns immediately to the calling routine when there are no more entries in a given class.

SCNLUB	$$BATTNI	CF
SCNRL1	$$BATTNA	AD
SCNRL2	$$BATTNA	AD
	$$BATTNK	DE
SETCODE	$$BDUMP[1]	LF
SETEXT	$$BATTNB	AF
SETHEX10	$$BDUMP[1]	LG
SETLOGUN	$$BEOJ1	FF
	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BILSVC	JA
	$$BPSW	JD
	$$BPCHK	JF

Routine that sets logical unit address for SYSLST in CCB after determination of symbolic device to be used for message output.

SETSW	$$BPDUMP[2]	LC
SETSW4	$$BDUMPB[2]	KF
	$$BDUMPD	KL
SETUP	$$BTERM	HA
SHIFT	$$BDUMPB[2]	KH
	$$BDUMPD	KN
	$$BPDUMP	LE
SKIP5	$$BDUMPF[1]	MC
SKIPHDR	$$BTERM	HC
SKPLIN	$$BATTND	AM
SNGCHG	$$BATTNP	EQ
SNGUNA	$$BATTNI	CH
SPACE	$$BATTNJ	CN
SPACE1	$$BDUMPB[2]	KH
	$$BPDUMP	LE
SPACE2	$$BDUMPB[2]	KH
	$$BPDUMP	LE
SPECIAL	$$BDUMPF[1]	MD
	$$BDUMPD	MQ
	$$BPDUM1	NF

See discussion of this label under ALTER.

SPECIAL1	$$BPDUM1[1]	NE
START	$$BATTNG	BE
	$$BDUMPF[1]	MA
	$$BDUMPB	MG
	$$BDUMPD	MM
	$$BPDUMP	NA
	$$BPDUM1	NB
	$$BEOJ	FA
	$$BPCHK	JF
START1	$$BDUMPF[1]	MA
STARTF	$$BATTNF	BC
STARTIO	$$BDUMPB[2]	KH
	$$BDUMPD	KN
	$$BPDUMP	LE
STDSWH	$$BATTNJ	CM

Program switch set to branch when stored standard assignments are to be logged. The branch is taken at the end of the JIB table scan. The scan finds any stored standard assignments. The switch is reset at location LSTSTD.

STEXCD	$$BATTNB	AF
STH	$$BEOJ1	FF
	$$BPSW	JE
STLLMT	$$BATTNF	BC
STORCEWD	$$BDUMP[2]	KC
STORE1	$$BPDUMP[2]	LB
STORE5	$$BDUMPB[2]	KF
	$$BDUMPD	KL
	$$BPDUMP	LC
STORECPU	$$BCCHHR	NK
STORSTRT	$$BDUMP[2]	KA
STORWD	$$BDUMP[2]	KC
STUBGL	$$BATTND	AL
STUCRL	$$BATTND	AL

STUF1U	$$BATTND	AL
STUSPC	$$BATTND	AL
SUB1	$$BDUMPD[2]	KN
SUBEOJ	$$BEOJ	FA
SUBLOOP	$$BATTNC	AH
SUPPRIO	$$BEOJ	FB

This routine is entered if an abnormal end-of-job condition occurs while transient $$BTERM is executing. An I/O unrecoverable error would cause a cancel of $$BTERM itself, resulting in an unending loop. Therefore, I/O operation is bypassed and $$BTERM is recalled.

SUPV	$$BDUMPF[1]	MB
	$$BDUMPD	MN
SUPV1	$$BDUMPF[1]	MB
	$$BDUMPB	MH
	$$BDUMPD	MN
SVC0	$$BEOJ	FD
SVC2	$$BEOJ	FC
SVCERR	$$BEOJ	FC
SWITCH1	$$BDUMPB[2]	KG
	$$BDUMPD	KM
	$$BPDUMP	LD
SWITCH2	$$BDUMPB[2]	KG
	$$BDUMPD	KM
	$$BPDUMP	LD
SWITCH3	$$BDUMPB[2]	KF
	$$BDUMPD	KL
	$$BPDUMP	LC
SWITCH4	$$BDUMPB[2]	KG
	$$BDUMPD	KM
	$$BPDUMP	LD
SWTCHOFF	$$BDUMPB[2]	KH
	$$BDUMPD	KN
SXTPOK	$$BATTNL	DL
SYSMODM	$$BATTNI	CC
SYSRFG	$$BATTNI	CC
SYSSWH	$$BATTNJ	CM

Program switch set to branch when system units are to be logged. The switch is set to branch by the list BG routine (Chart PF). The switch is reset to NOP after the system class units have all been logged.

SYSTST	$$BDUMP[1]	LF

Routine similar to the NOCHNG routine. Identifies the physical device assigned to SYSLST for a background program dump.

SYSUNT	$$BATTNJ	CM
SYSWR	$$BYSWR	NF
SYSXN2	$$BATTNI	CC
SYSXXX	$$BATTNI	CC

TAPE1	$$BDUMP[1]	LG

When SYSLST is a tape unit, the CCB and CCW are modified accordingly to perform a sense operation for a file-protect condition. Register 12 signals the fetched dump program that a tape unit is to receive the storage dump.

TAPELST	$$BDUMP[2]	KC
TAPNOP	$$BDUMPF[1]	MC
TAPRTN	$$BDUMPF[1]	MA

Data address is stored in the tape CCW, and CCB is furnished with the CCW address. Switches at OUT1 and TAPNOP are set to branch to perform functions necessary for output on tape unit.

TAPSYS	$$BDUMPF[1]	MC
TAPSYS1	$$BDUMPF[1]	MC
TEB	$$BTERM	HC
TEBVDPL	$$BEOJ4	FM
TERM	$$BEOJ1	FF
	$$BILSVC	JB
	$$BPSW	JE
	$$BPCHK	JF
	$$BDUMP[1]	LG
	$$BTERM	HB
TEST1	$$BTERM	HB
TESTBG	$$BDUMP	KC
TESTCODE	$$BCCHHR	NH
TESTEOF	$$BDUMPB	KH
	$$BPDUMP	LE
TESTLST	$$BDUMP	KC
	$$BPDUMP	LA
TFILL	$$BATTNK	DB
TIMER	$$BATTNN	EG
TIMLNK	$$BATTNN	EG
TLBL	$$BATTNK	DC
TM	$$BEOJ3	FL
TMFAVP	$$BATTNI	CE
TNAERR	$$BATTNN	EG
TPLAB	$$BATTNK	DB
TPLEND	$$BATTNK	DB
TPMARK	$$BDUMPF[1]	ME
TPRTN	$$BDUMPB[1]	MG
	$$BPDUM1	NB
TPTYPE	$$BEOJ	FC
	$$BEOJ1	FF
	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BILSVC	JB
	$$BPSW	JE
	$$BPCHK	JG
	$$BDUMP[1]	LG
TPTYPE1	$$BEOJ	FC
	$$BEOJ1	FF
	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BILSVC	JB
	$$BPSW	JE
	$$BPCHK	JG
TRANS	$$BDUMPB[1]	MK
TSTBATCH	$$BEOJ1	FF

	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BPSW	JE
	$$BPCHK	JG
TSTBJF	$$BEOJ1	FE
	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BILSVC	JA
TSTCOR	$$BDUMPF[1]	MF
	$$BDUMPB	ML
	$$BDUMPD	MR
	$$BPDUM1	ND

Entry to subroutine that tests whether storage area to be printed on a line is in dump limits and whether the next line is the last line. Register 3 points to the storage address of the first byte of a line to be printed, and register 5 points to the last byte of the line. See discussion of the CMPCOR label, which is part of this subroutine.

TSTLST	$$BEOJ1	FE
	$$BEOJ2	FH
	$$BEOJ2A	FK
	$$BILSVC	JA
	$$BPSW	JD
	$$BPCHK	JF
	$$BDUMPF[1]	MF
	$$BDUMPB	ML
	$$BDUMPD	MR
	$$BPDUM1	NC
TSTLST1	$$BDUMPF[1]	MF
	$$BDUMPB	ML
	$$BDUMPD	MR
	$$BPDUM1	NC

Switch that is set to a branch on last line of dump. If a portion of core is identical to the previous line, this switch is set to NOP and the identical data is shown by printing a line with---SAME---.

TSTPRT	$$BDUMPF[1]	MF
	$$BDUMPB	ML
	$$BDUMPD	MR
	$$BPDUM1	NC
TTTTMULT	$$BATTNQ	PB
TURNETON	$$BATTNQ	PB
TXCUU	$$BATTNI	CD
TYPTYPE1	$$BEOJ1	FF

UALNOT	$$BATTNJ	CP
UAPSWH	$$BATTNJ	CK

Program switch set to branch after unit assignments have been listed. This switch is initialized in the NOP state. It is set to branch just before the 'UNASSIGNED' header is logged.

UCS	$$BATTNM	EE
UCS1	$$BATTNM	EE
UCS2	$$BATTNM	EE
UCS3	$$BATTNM	EE
UCS4	$$BATTNM	EE
UCSDN	$$BATTNM	EE
UCSSCN	$$BATTNM	EE
UCUERR	$$BATTNI	CJ
UNA	$$BATTNI	CH
UNAGO	$$BATTNI	CJ
UNALOP	$$BATTNI	CJ
UNANO	$$BATTNI	CJ
UNARTN	$$BATTNI	CJ
UNASSGN	$$BTERM	HA

Test for resetting symbolic device assignments and, if required, continue to the next chart, where the LUBNDX from the PIB of this program (F1 or F2) is inserted in register 5. In the case of F1, for example, LUBNDX is equal to the sum of the LUBs assigned to devices owned by the system programs, the background program, and the foreground 2 program. This index is doubled because there are two bytes per LUB entry. The result is the displacement from the LUB table starting address, where this foreground program's LUBs begin. By adding this displacement to the LUB table starting address, the actual address for the first LUB is obtained in register 5.

The number of LUBs assigned to this type of foreground program is obtained from the NICL (Number-in-Class); this value is adjusted and doubled.

UNAV	$$BATTNG	BF
UNCLOG	$$BATTNH	BH
UNNORM	$$BEOJ	FB

Routine entered when abnormal end-of-job condition exists. Investigation of cause of cancel and type of program executing is made, to determine which B-transient of the terminating phases to call next.

UNPA	$$BATTNI	CE

Entry point to a routine that unassigns currently assigned logical units. The subroutine saves, in location LBSLUB, the LUB entry of the LUB to be unassigned. It then unassigns the LUB in the LUB table. It checks the LUB table and JIB table for other LUBs that point to the same physical unit as that of the LUB just unassigned. It resets the ownership flag in the PUB if no other LUBs point to that physical unit. Any stored alternate assignments found in the JIB table are treated as LUBs (unassigned, then checked for matching PUB pointers).

UNPACK	$$BDUMPB	KG
	$$BDUMPD	KM
	$$BPDUMP	LD
UNPAN2	$$BATTNI	CE
UNPK	$$BDUMPF[1]	MD

	$$BDUMPB	MK
	$$BDUMPD	MQ
	$$BPDUM1	ND
UNPK1	$$BDUMPF[1]	MD
	$$BDUMPD	MQ
	$$BPDUM1	ND
VOL	$$BATTNK	DA
WAITLOOP	$$BCCHHR	NH
WFM	$$BATTNH	BG
WFMRES	$$BATTNH	BH
WRITEREC	$$BESTVD	GE
XTENT	$$BATTNL	DL
XTOP12	$$BATTNL	DP

Entry point to a subroutine that extracts and validity checks the first two operands (type and sequence number) of an XTENT statement. It converts the operand to binary, and stores it in the label area, DSECT (I/O area).

XTOP12A	$$BATTNO	EL
XTOP12B	$$BATTNO	EL
XTOP3	$$BATTNL	DM

XTOP34 $$BATTNL DP

Entry point to a subroutine that extracts limit information from the XTENT statement, performs initial validity checks, converts the numeric EBCDIC limit data to binary, and puts the limits into the label area, DSECT (I/O area).

XTOP5 $$BATTNL DM

Entry point to the routine that:

1. Gets and checks the serial number, and stores it in the label area DSECT (I/O area).
2. Converts the SYSXXX field of the extent to class and displacement.
3. Gets the B2 field of an extent, converts it to binary, and stores it in the label area, DSECT (I/O area).

XTOUT	$$BATTNL	DN
XTUNIT	$$BATTNO	EK
ZERTEBV	$$BEOJ4	FM

APPENDIX B: ERROR MESSAGE CROSS REFERENCE

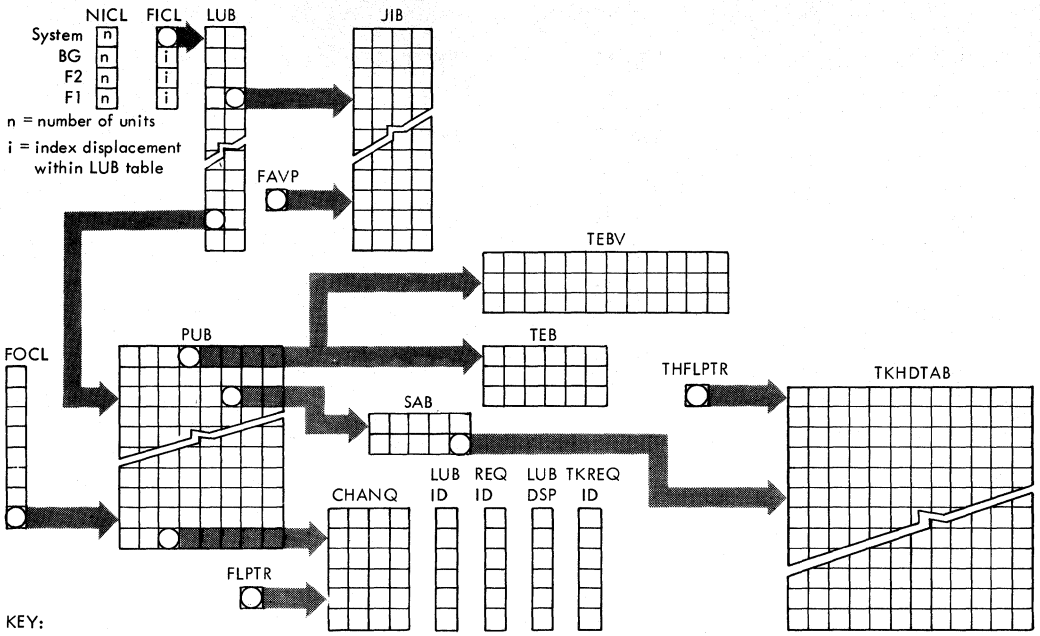
Message	Phase	Chart
0P70I	$$BEOJ2	FG*
0P71I	$$BEOJ2	FG*
0P72I	$$BEOJ2	FG*
0P73I	$$BEOJ2A	FJ*
0P74I	$$BEOJ2A	FJ*
0P75I	$$BEOJ2	FG*
0P76I	$$BEOJ2	FG*
0P77I	$$BEOJ2	FG*
0P78I	$$BEOJ2A	FJ*
0P79I	$$BEOJ2A	FJ*
0P81I	$$BEOJ2A	FJ*
0P82I	$$BEOJ2A	FJ*
OP83A	$$BEOJ2	FG*
0P83I	$$BEOJ2	FG*
0P84I	$$BEOJ2A	FJ*
0P85I	$$BEOJ2A	FJ*
0S00I	$$BPCHK	JF
	$$BILSVC	JA
0S01I	$$BEOJ2	FG*
0S02I	$$BEOJ2	FG*
0S03I	$$BPCHK	JF*
0S04I	$$BILSVC	JA*
0S05I	$$BILSVC	JA*
0S06I	$$BEOJ1	FE
0S07I	$$BPSW	JD
0S08I	$$BEOJ	FA
0S09I	$$BEOJ1	FE*
0S10I	$$BTERM	HA
0S11I	$$BEOJ1	FE*
0S12I	$$BEOJ2A	FJ*
0S13I	$$BEOJ2A	FJ*
0S14I	$$BEOJ2A	FJ*
1A00D	$$BATTNI	CB
1A10D	$$BATTNI	CA
	$$BATTNP	EP
1A20D	$$BATTNI	CA,CB
	$$BATTNM	EE
	$$BATTNP	EP
1A30D	$$BATTNI	CB
1A40D	$$BATTNI	CC
	$$BATTNK	DD
	$$BATTNM	EE,EF
1A50D	$$BATTNI	CD
1A60D	$$BATTNI	CJ
1A70D	$$BATTNI	CD
1C20D	$$BATTNH	BH
1C30A	$$BATTNM	EA,EE
1C40I	$$BATTNB	AF
1C50I	$$BATTNB	AF
1C60D	$$BATTNN	EG
1I30D	$$BATTNC	AG
1I60A	$$BATTNA	AA
1I80I	$$BTERM	HA
1L00D	$$BATTNL	DH,DK,DL,DN
	$$BATTNO	EJ,EK,EL,EM
1L10D	$$BATTNK	DG
	$$BATTNL	DK
	$$BATTNM	EB
	$$BATTNO	EH
1P00D	$$BATTNF	BC
1P10D	$$BATTNG	BF

---

*Also refer to Cancel Codes and Messages Figure.

1S00D	$$BATTNA	AB,AC	4E00I	$$BESTVB	GB
	$$BATTNB	AE			
	$$BATTNC	AG			
	$$BATTNE	BA,BB	4E01I	$$BESTVC	GC
	$$BATTNG	BF			
	$$BATTNI	CA,CC,CD,			
		CG,CH,CJ	4E02I	$$BESTVC	GC
	$$BATTNJ	CK			
	$$BATTNK	DA,DB,DC,			
		DD,DE,DF	4E03I	$$BESTVE	GF
	$$BATTNL	DH,DJ,DL			
	$$BATTNM	EA,EE,EF			
	$$BATTNN	EG	4E04I	$$BESTVC	GC
	$$BATTNO	EH,EJ,EK,			
		EL,EM,EN			
	$$BATTNP	EP,EQ	4E05I	$$BESTVF	GG
1S10D	$$BATTNK	DA,DB			
	$$BATTNL	DH,DK	4E06I	$$BESTVF	GG
	$$BATTNM	EA			
	$$BATTNO	EH,EJ			

APPENDIX C: SUPERVISOR REFERENCE FIGURES



KEY:

NICL (Number in Class) : The first byte contains the number of system class units. The second, third, and fourth bytes contain the number of programmer class units (BG, F2, F1) (Figure 24).

FICL (First in Class) : The first byte points to the first system class unit in the LUB table. (Always the first LUB table entry.) The second byte points to the first programmer class unit in the LUB table BG area. The third points to the first programmer class unit in the LUB table F2 area. The fourth points to the first programmer class unit in the LUB table F1 area (Figure 24).

LUB (Logical Unit Block) Table : The first byte points to a PUB table entry (if the logical unit is assigned) or contains X'FF'. The second byte points to a JIB table entry or contains X'FF' (Figure 24).

PUB (Physical Unit Block) Table : The first two bytes contain the channel and unit address of the physical device; the third a CHANQ pointer; the fourth a TEB pointer; the fifth device type codes; the sixth a device characteristic code or a SAB pointer; the seventh the channel scheduler flag; and the eighth has the job control flag (See Figure 19).

FOCL (First on Channel List) : The first byte points to the first PUB (highest priority) on channel zero. The next byte points to the first PUB (highest priority) on channel one, etc. A hexadecimal FF indicates the associated channel is not supported.

TEB (Tape Error Block by Unit) : One TEB is built for each tape unit at supervisor generation time if tape error statistics by unit are required (Figure 21).

TEBV (Tape Error Block by Volume) : One TEBV is built for each tape unit at supervisor generation time if tape error statistics by volume are required (Figure 22).

FAVP (First Available Pointer) : A one-byte pointer to the next available JIB entry.

JIB (Job Information Block) : The first two bytes contain extent or LUB information. The third contains ownership and JIB flags. The fourth contains JIB chaining information (Figure 20).

CHANQ (Channel Queue) Table : The first byte contains the chain field (a pointer to the next in queue). The last three bytes contain the CCB address (Figure 18).

LUBID (LUB Identification) : A one-byte pointer to the LUB making the I/O request.

REQID (Requestor Identification) : A one-byte pointer to the program containing the CCB (Figure 18).

LUBDSP (LUB Displacement) : A one-byte value equal to the absolute LUB number (CCB byte 7).

FLPTR (Free List Pointer) : A one-byte pointer to the next free entry in the channel queue (Figure 18).

SAB (Seek Address Block) : A four-byte (BCCH) address that is the current disk address of the device plus a fifth byte that contains a Track Hold Table pointer of X'FF'. If the Track Hold function is not supported, the fifth byte contains X'00'.

TKHDTAB (Track Hold Table) : The first byte contains a pointer to the next available entry (or X'FF'); bytes 2 - 4 have CCB address of the requesting task; bytes 5 - 10 have a disk address (BBCCHH) of track being held; byte 11 has key of owning track; and byte 12 has two uses: bit 0=1 means a task is waiting for the track, and bits 4 - 7 count the number of holds on the track. (Figure 25). Note: The number of holds is one more than the value of bits 4 - 7 of the last byte.

THFLPTR (Track Hold Free List Pointer) : A one-byte pointer to the next free entry in the Track Hold Table.

TKREQID (Track Requestor Identification) : A one-byte pointer to the PIB of the task requesting I/O.

Figure 8. I/O Table Interrelationship

COMREG*

Displacement hexadecimal	0	8	0A	0C	17	18	20	24	28	2C
Displacement decimal	0	8	10	12	23	24	32	36	40	44
	Date	Address of PPBEG	Address of EOSSP	Problem Program Use	UPSI Byte	Job Name	Highest Storage Address of the Partition	End Address of Last Phase Fetched or Loaded	Address of Uppermost Byte of Phase With Highest Ending Address	Label Area Length
	XXXXXXXX	XX	XX	XXXXXXXXXXX	X	XXXXXXXX	XXXX	XXXX	XXXX	XX

Displacement hexadecimal	2E	30	34	35	36	37	38	39	3A	3B	3C	3E
Displacement decimal	46	48	52	53	54	55	56	57	58	59	60	62
	PIK (PID)	End of Storage Address	Machine Confg. Byte	System Confg. Byte	Standard Language Translator I/O Options	Dump, Log and ASCII Options	Job Control Byte	Linkage Control Byte	Language Translator Control Byte	Job Duration Indicator Byte	Disk Address of Label Cylinder	Address of FOCL
	XX	XXXX	X	X	X	X	X	X	X	X	XX	XX

Job Control Switches

Displacement hexadecimal	40	42	44	46	48	4A	4C	4E	4F	58	5A	5C
Displacement decimal	64	66	68	70	72	74	76	78	79	88	90	92
	Address of PUB	Address of FAVP	Address of JIB	Address of TEB	Address of FICL	Address of NICL	Address of LUB	Line Count for SYSLST	System Date	LIOCS Comm. Bytes	Address of 1st Part of PIB Table	ID Number of Last Checkpoint
	XX	XX	XX	XX	XX	XX	XX	X	XXXXXXXXX	XX	XX	XX

Displacement hexadecimal	5E	60	62	64	66	68	6A	6C	6E
Displacement decimal	94	96	98	100	102	104	106	108	110
	Length of LUB ID Queue = No. of Channel Queue Entries	Address of Disk Information Block (DIB)	Address of Error Recovery Block	Address of PC Option Table less 8 bytes	Address of IT Option Table less 8 bytes	Address of OC Option Table less 8 bytes	Key of Program with Timer Support	Address of the LUBID Queue	Logical Transient Key
	XX	XX	XX	XX	XX	XX	XX	XX	XX

Displacement hexadecimal	70	7C	7E	80	84	86	87	88
Displacement decimal	112	124	126	128	132	134	135	136
	Supervisor Constants	Address of 2nd Part of PIB Table	Address of MICR DTF Table (PDTABB)	Address of QTAM Vector Table	Address of BG Comm. Region	Option Indicator	System Configuration Byte 2	Pointer to Comm. Region Extension
	XXXXXXXXXXXX	XX	XX	XXXX	XX	X	X	XXXX

* The address of the communications region is in fixed location X'14' - X'17'.

Displacement values illustrated can be used to access the listing and/or the key that follows the figure. The key offers more detailed information about each area when necessary.

Figure 9. Supervisor Communications Region (Part 1 of 5)

Key to Communications Region Displacements:

0 MM/DD/YY or DD/MM/YY obtained from the job control date statement. Format controlled by COMREG + 53 (System Configuration Byte, date convention bit 0).

8 Address of the problem program area.

10 Address of the beginning of the problem program area. Y (EOSSP)=Y (PPBEG) if the storage protection option has not been selected. Y (EOSSP) equals the first main storage location with a storage protection key of 1, if storage protection is supported.

12 User area. If seek separation option is specified, bytes 12 and 13 are used at IPL time for the address of the seek address block.

23 User program switch indicator.

24 Job name set by the job control program from information found in the job statement.

32 Address of the uppermost byte of the problem program area as determined by the IPL program (Clear storage routine determines the address, ENDRD routine of $$A$IPL2 stores it.), or the address of the uppermost byte of the partition as determined during processing of the ALLOC statement.

36 Address of the uppermost byte of the last phase of the problem program fetched or loaded. The initial value (as shown) is overlaid by the first fetch or load to the problem program area.

40 Highest ending main-storage address of the phase among all the phases having the same first four characters as the operand on the EXEC statement. For the background partition only, job control builds a phase directory of these phases. The address value may be incorrect if the program loads any of these phases above its link-edited origin address. If the EXEC statement has no operand, job control places in this location the ending address of the program just link-edited.

44 Length of the problem program label area.

46 Program Interrupt Key - PIK (if asynchronous processing is not supported): Value is equal to the displacement from the start of the PIB table to the PIB for the task.

OR

Partition Identifier - PID (if asynchronous processing is supported): Value is hex 10, 20, or 30 to identify the partition in which a maintask or a subtask is running. (See the communications region extension, displacement 18, for the PIK in an asynchronous processing supervisor.)

First byte - always zero.
Second byte - contains the key of the program that was last enabled for interrupts, or the partition identifier in an AP supervisor.

Task	PIK (PID) Value
*All Bound	X'00'
BG	X'10'
*F2	X'20'
*F1	X'30'
Attn Rtn	X'40'
Quiesce I/O	X'50'
Supervisor	X'60'

*These tasks do not exist in a non-MPS supervisor.

48 Logical end of main storage address.

Figure 9. Supervisor Communications Region (Part 2 of 5)

Key to Communications Region Displacements:

52 Machine Configuration Byte (Values set at supervisor generation time.)

Bit 0: 1 = Storage protect feature
0 = No storage protect feature
1: 1 = Decimal feature
0 = No decimal feature
2: 1 = Floating-point feature
0 = No floating-point feature
3: 1 = Physical transient overlap option
0 = No physical transient overlap option
4: 1 = Timer feature
0 = No timer feature
5: 1 = Channel switching device
0 = No channel switching device
6: 1 = Burst mode on multiplex channel support
0 = No burst mode on multiplex channel support
7: Reserved

53 System Configuration Byte

Bit 0: 1 = DDMMYYJJ (Set at generation time by STDJC)
0 = MMDDYYJJ
1: 1 = Multiprogramming environment
0 = Batch job environment
2: 1 = DASD file-protect supported
0 = No file-protect support for DASD
3: 1 = DASD SYSIN - SYSOUT
0 = No DASD SYSIN - SYSOUT
4: 1 = Teleprocessing
0 = No teleprocessing
5: 1 = Batch job in foreground
0 = No BJF
6: 1 = Asynchronous processing
0 = No AP
7: 1 = Track Hold
0 = No Track Hold

54 This byte contains the standard language translator I/O options (set by the STDJC macro).

Bit 0:	DECK option	1 = yes, output object modules on SYSPCH
1:	LIST option	1 = yes, output source module listings and diagnostics on SYSLST
2:	LISTX option	1 = yes, output hexadecimal object module listings on SYSLST (compilers only)
3:	SYM option	1 = yes, output symbol tables on SYSLST/SYSPCH
4:	XREF option	1 = yes, output symbolic cross reference list on SYSLST
5:	ERRS option	1 = yes, output diagnostics on SYSLST (compilers only)
6:	CHARSET option	1 = 48, input on SYSIPT is 48 or 60 character set
7:	Reserved	

55 This byte contains the standard supervisor options for abnormal EOJ and control statement display, and the indicator for the presence of the ASCII-EBCDIC and EBCDIC-ASCII translation tables.

Bit 0:	Always on	
1:	DUMP option	1 = yes, dump registers and storage on SYSLST
2:	Reserved	
3:	LOG option	1 = yes, list all control statements on SYSLST
4-6:	Reserved	
7:	ASCII option	1 = yes, ASCII supported

Figure 9. Supervisor Communications Region (Part 3 of 5)

Key to Communications Region Displacement:

56 Job control byte

Bit 0: 1 = Job Accounting Interface (JA) not supported
0 = Job Accounting Interface (JA) is supported
1: 1 = Return to caller on LIOCS disk open failure
0 = Do not return to caller on LIOCS disk open failure
2: 1 = Job control input from SYSRDR
0 = Job control input from SYSLOG
3: 1 = Job control output on SYSLOG
0 = Job control output not on SYSLOG
4: 1 = Cancel job
0 = Do not cancel job
5: 1 = Pause at end-of-job step
0 = No pause at end-of-job step
6: 1 = SYSLOG is not a 1052
0 = SYSLOG is a 1052
7: 1 = SYSLOG is assigned to the same device as SYSLST
0 = SYSLOG is not assigned to the same device as SYSLST

57 Linkage control byte

Bit 0: 1 = SYSLNK open for output
0 = SYSLNK not open for output
1: 1 = $ or FG program phase deleted, renamed, or cataloged (flag bit for $MAINEOJ)
2: 1 = Allow EXEC
0 = Suppress EXEC
3: 1 = Catalog linkage editor output
0 = Do not catalog linkage editor output
4: 1 = Supervisor has been updated
0 = Supervisor has not been updated
5: 1 = Executing in AUTOTEST mode
0 = Not executing in AUTOTEST mode
6: 1 = Reallocate or condense in progress
7: 1 = Fetch $MAINEOJ at end of job to update system directory
0 = Do not fetch $MAINEOJ at end of job for update

58 Language processor control byte. This is a set of switches used to specify nonstandard language translator options. The switches within the byte are controlled by job control OPTION statements and when set to 1, override standard options. The format of this byte is identical to the stnadard option byte (displacement 54) with one exception: Bit 7 in this byte is used to indicate to LIOCS that the rewind and unload option has been specified.

59 Job duration indicator byte

Bit 0: 1 = Within a job condition
0 = Outside a job condition
1: 1 = Dump on an abnormal end-of-job condition
0 = No dump on abnormal EOJ
2: 1 = Pause at EOJ step
0 = No pause at EOJ
} Set by Attention Routine for Job Control
3: 1 = Job control output on SYSLST
0 = Output not on SYSLST
4: 1 = Job is being run out of sequence with a temporary assignment for SYSRDR
0 = Conditions for 1 setting not met
5: 1 = PCIL is being condensed
0 = PCIL is not being condensed
6: Reserved
7: 1 = Batch command just issued
0 = Condition for 1 setting did not occur

Figure 9. Supervisor Communications Region (Part 4 of 5)

Key to Communications Region Displacements:

60 Binary disk address of the volume label area (label cylinder).

62 → 76 As illustrated (Figures for information blocks, I/O tables, and pointers begin at Figure 7 which refers to more detailed Figures).

78 Set to the value nn specified in the LINES = nn parameter of the STDJC macro.

79 The format of the system date contained within this field is determined by the IPL program from information supplied in the date convention byte (displacement 53). Bytes 85-87 contain the day count.

88 Bytes reserved for use by LIOCS. Transient dump programs insert a key to indicate to the LIOCS end-of-volume routine, $$BCMT07, that it was called by a B-transient.

90 Address of the first part of the program information block (PIB) table. (See Figures 16, and 17).

92 ID number of the last checkpoint. Temporary indicator of file protected DASD. Used at IPL time, when DASDFP is specified.

94 Length of the LUBID queue (in bytes). This equals the number of channel queue entries. It can also be used to access the REQID, LUBDSP, and TKREQID queues: (See Figure 18.)

96 Address of disk I/O position data. This is the starting address of the disk information block (DIB) table (See Figure 12).

98 Address of the beginning of the error recovery block. The error recovery block contains addresses of error recovery exits, error recovery queue information that can be used by physical transients routines, and defines storage for the error queue entries (See Figure 13).

100 → 104 As illustrated (See Figure 14).

106 Key of the program (BG, F2, or F1) that has timer support.

108 As illustrated (See Figure 18).

110 Logical Transient Key (LTK) contains the same value as the PIK (PID) (Displacement 46) when the logical transient is requested. When the transient area is not in use, LTK is equal to zero. The SVC 2 routine sets the LTK. The SVC 11 routine resets the LTK.

112 Supervisor constants:

DOLLARBO	(4 bytes) = C'$$BO'
SSKADR	(5 bytes) = XL5'0'
LTAREA	(3 bytes) = Adcon of LTSVPT, logical transient save pointer

124 Address of second part of program information block (PIB) table (See Figure 15).

126 Address of PDTABB, table of DTF addresses for MICR support (See Figure 11).

128 Address of QTAM vector table (IJLQTTAD).

132 Address of background communications region.

134 Option Indicator Byte

Bit 0:	1 = MCRR indicated for OBR writer
	0 = No MCRR indicated for OBR writer
1:	1 = EU interface active
	0 = EU interface not active
2:	1 = TP request
	0 = No TP request
3:	1 = Supervisor support tor only 9-track tape
	0 = Supervisor does not support 9-track tape exclusively
4-7:	Reserved

135 System Configuration Byte 2

Bit 0:	1 = PCIL supported
	0 = PCIL not supported
1-7:	Reserved

136 Pointer to communications region extension (See Figure 10).

Figure 9. Supervisor Communications Region (Part 5 of 5)

BGXTNSN (See Note)

0 (Hexadecimal Displacement) 0 (Decimal Displacement)	4 4	8 8	0C 12	10 16	12 18	14 20	18 24	1C 28	20 32
CE Table Address	Track Hold Table Address (THTABAD)	Difference Between 1st and 2nd Part of PIB Table (PIBDIFF)	AB Termination Table Address -8 (ABPTR)	ID of Task Owning LTA (LID)	ID of Task Running (PIK)	Task Requester ID Table Address (TKIDPTR)	Address Used by QTAM (MVCFLD)	SDR Table Address (SDRTABLE)	TEBV Table Address (TEBVTAB)
XXXX	XXXX	XXXX	XXXX	XX	XX	XXXX	XXXX	XXXX	XXXX

24 (Hexadecimal Displacement) 36 (Decimal Displacement)	28 40	2C 44	30 48	34 52	38 56	3C 60
OLTEP Linkage Address	RMS Linkage Address (RASLINK)	ASCII-EBCDIC Translation Table Address	(Reserved)	JAI Common Table Address (ACCTCOMN)	JAI Partition Table Address (ACCTxx)	&SYSPARM Field Address
XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX

Key to displacements:

0 CE Table Address.

4 Track Hold Table Address (THTABAD).

8 Difference between addresses of first part of PIB table and second part of PIB table (PIBDIFF).

12 Abnormal Termination Table Address (minus 8) (ABPTR).

16 Identification (LID) of the task owning the Logical Transient Area. Contains same value as PIK (displacement 18) when LTA is in use. Contains zero when LTA is not in use.

18 Program Interrupt Key (PIK) if asynchronous processing is supported. Value is equal to the displacement of the start of the PIB table to the PIB of the main task or subtask being selected (running).
First byte - zero
Second byte - contains the displacement into the PIB table for a maintask or a subtask.
Maintask - PIK value is hex 10, 20, or 30.
Subtask - PIK valve is hex 70, 80, 90, . . . F0.

20 Task Requester ID Table Address (TKIDPTR).

24 MVCFLD address used by QTAM.

28 Statistical Data Recorder Table Address (SDRTABLE).

32 Tape Error Blocks by Volume Table Address (TEBVTAB).

36 Pointer to OLTEP Linkage Addresses

40 RMS Linkage Area Address (RASLINK)

44 ASCII-EBCDIC Translation Table Address.

48 (Reserved)

52 JAI Common Table Address (ACCTCOMN)

56 JAI Partition Table Address (ACCTxx; where xx = BG, F2, or F1).

60 Address of &SYSPARM Field.

Note: If communications regions are generated for the foreground partitions, the labels in those extensions will be F2XTNSN and F1XTNSN. The extensions, wherever used, are generated by the COMMNEX macro. Following the background extension (and immediately preceding the MCRR Linkage Table) is a six-byte area. The first four bytes are the address of the background save area (BGSAV), and the last two bytes are the value 4,096, used to restore base registers.

Figure 10. Communications Region Extensions

The table of DTF addresses (PDTABB) contains six 8-byte entries; one for each external line of the direct control feature on the system.

PDTABB

Byte	0	1	2	3	4	5	6	7
						DTF address for MICR:		
0	NI	PDSTAT +1,	X'FE'		Ownership Flags	Device on line 7		
8	NI	PDSTAT +1,	X'FD'			Device on line 6		
16	NI	PDSTAT +1,	X'FB'			Device on line 5		
24	NI	PDSTAT +1,	X'F7'			Device on line 4		
32	NI	PDSTAT +1,	X'EF'			Device on line 3		
40	NI	PDSTAT +1,	X'DF'			Device on line 2		

Background = 10
Foreground 2 = 20
Foreground 1 = 30

- Bytes 0-3 --Contain an 'AND' instruction that is executed in main line coding to turn off the external line status after its detection.

  PDSTAT +1 will contain one or more of the following interrupt codes:

PSW Interrupt Code Bit	Interrupt Code (PSW Bits 26-31)*	External Interrupt Cause
31	nnnnnnn1	External signal 7
30	nnnnnn1n	External signal 6
29	nnnnn1nn	External signal 5
28	nnnn1nnn	External signal 4
27	nnn1nnnn	External signal 3
26	nn1nnnnn	External signal 2

- Byte 4 --Contains the flag of the partition containing the DTF.
- Bytes 5-7 --Contain the address of the DTF table.

Table of pointers (PDTABA) to DTF addresses associated with the external interrupt line. The table is set up to handle the status in descending order from Bit 31 to Bit 26 of the external old PSW.

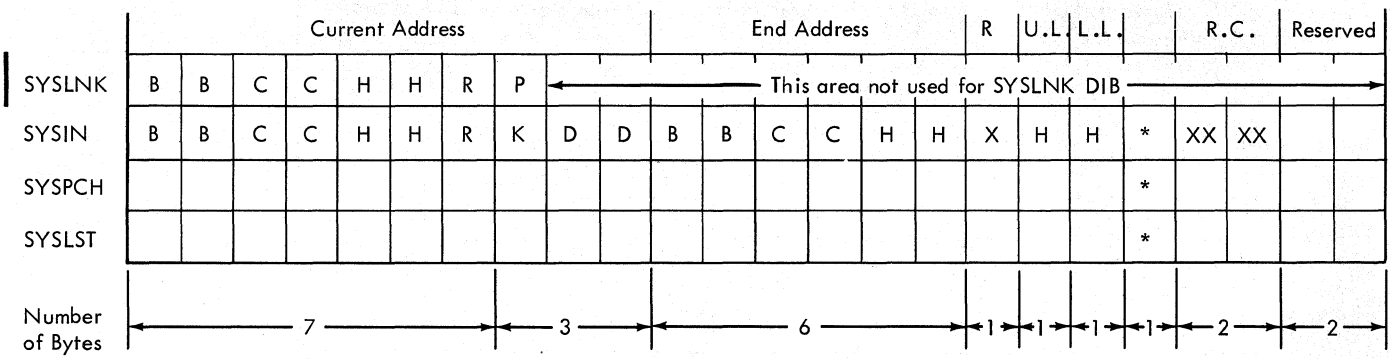
PDTABA

Byte	0	1	2	3	4	5	6	7
0	00	08	00	10	00	08	00	18
8	00	08	00	10	00	08	00	20
16	00	08	00	10	00	08	00	18
24	00	08	00	10	00	08	00	28
32	00	08	00	10	00	08	00	18
40	00	08	00	10	00	08	00	20
48	00	08	00	10	00	08	00	18
56	00	08	00	10	00	08	00	

*n = other external-interrupt conditions.

Bytes 126 and 127 (X'7E'-'7F') of the communications region contain the address of these tables. Label PDTABB identifies the first byte of the first table. These tables are also used for optical reader/sorters.

Figure 11. Tables for MICR DTF Addresses and Pointers



KEY: Current Address: The next address to be used (for both input and output).
End Address : The last address within the limits of the extent.
R : Maximum number of records per track.
U.L. : Upper head limit
L.L. : Lower head limit
R.C. : Record Count - residual capacity for beginning of operator notification. This is set at system generation time with the SYSFIL parameter, or after IPL with the SET statement (RCLST and/or RCPCH operands). A warning message is issued by job control after end-of-job step when the minimum number of remaining records has been reached or exceeded during the previous job.
P : Starting cylinder of Private Core Image Library, if PCIL is assigned.
KDD : Key and data length for the symbolic device.

KDD for SYSIN = X'000050'
KDD for SYSPCH = X'000051'
KDD for SYSLST = X'000079'

Bytes 96 and 97 (X'60' - '61') of the communications region contain the address of the SYSLNK entry.
Label DSKPOS identifies the first byte of the table.

Figure 12. Disk Information Block (DIB) Table

Displacement from ERBLOC in Decimal	Length in Bytes	Label	Description
-2	2	ERRQ	Address of first error queue entry in table (ERQUE).
0	2	ERBLOC	Address of retry ERP exit (EXRTY).
2	2		Address of ignore ERP exit (EXIGN).
4	2		Address of DISWHY (retry) ERP exit (EXWHY).
6	2	YCHANQ	Address of the channel queue table (CHANQ).
8	2		Address of cancel ERP exit (EXCAN).
10	2	FULQUE	Address of last entry in error queue table.
12	2	ERQPTR	Address of last entry queued to table (initially ERQUE - 22).
14	2	RIK	Requestor I/O key (RIK).
16	2		Address of cancel attention exit (ATNCNL).
18	2		Address of attention dequeue exit (PUBDEQ).
20	2	ATNEXT	Address of attention exit (EXT02).
22	8	SVC3NM	A - transient phase name field ($$ANERRx, or $$ANERAx) where x = any alphabetic character.
30	5 x 22	ERQUE	Five 22 bytes error queue entries. (See below)

Error Queue Entry (22 bytes)

Bytes	CSW 0-7	Pub Address of Device in Error 8-9	Flag Byte (see below) 10	Message Number * 11	Sense Data * 12-17	Disk Seek Address 18-21
	XXXXXXXX	XX	X	X	XXXXXX	XXXX

Flag Byte

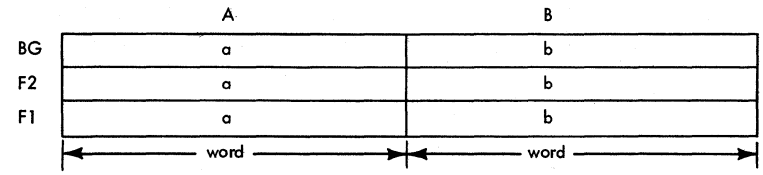
Bit	Designation
0	Unused
1	Intervention required
2	Passback ◄— *
3	Allow ignore
4	DASD in error
5	Allow retry
6	No CCB available
7	Unused

* For information on how ERBLOC is used by physical transients, see Supervisor and Physical Transients, listed in the Preface.

Bytes 98 and 99 (X'62' - X'63') of the communications region contain the address of the error block. Label ERBLOC identifies the first byte of the table. The address of the first error queue entry is at ERBLOC - 2.

Figure 13. Error Recovery Block (ERBLOC) and Error Queue Entry

PC Option Table and OC Option Table:



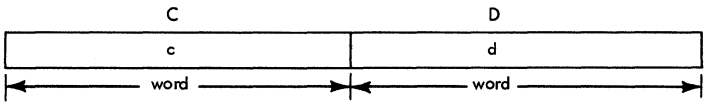
A

No STXIT given: a = 0
STXIT issued: a = address of the user program check (operator communications) routine
STXIT issued when the user routine is already in use: a = complement of user program check (Operator communications) routine address

B

No STXIT given: b = 0
STXIT issued: address of the user save area

IT Option Table:



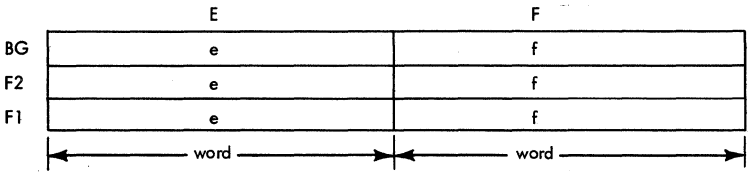
C

No TECB or STXIT issued: c = 0
TECB issued: c = address of the timer event control block
STXIT issued: c = address of the user interval timer routine
STXIT issued when user routine is already in use: c = complement of the user interval time routine address

D

No TECB or STXIT issued: d = 0
TECB issued: d = complement of the TECB address
STXIT issued: d = address of the user save area

AB Option Table:



E

No STXIT given: e = 0
STXIT issued and rtnaddr parameter passed: e = address of entry point of user's abnormal termination routine. If AP (asynchronous processing) is supported, the maintask and subtasks may have the same or different AB routines. When a subtask is ATTACHed after a STXIT AB macro has been issued by the maintask, the subtask will receive the AB routine address specified by the maintask only if the ATTACH macro for that subtask has the ABSAVE parameter specified. The subtask can override this by issuing its own STXIT AB macro.

F

No STXIT given or no save area parameter passed: f = 0
STXIT issued and save area parameter passed: f = address of a 72-byte save area used by the supervisor to store the old PSW and general registers 0-15.

Each table address (less 8 bytes) is found in the communications region at the byte locations specified below. The labels shown identify the first byte of the corresponding table.

Table	Bytes in COMREG	Label
PC	100-101 (X'64' - '65')	PCTAB
IT	102-103 (X'66' - '67')	ITTAB
OC	104-105 (X'68' - '69')	OCTAB
AB	12-13 (X'0C - '0D') of extension	ABTAB

Figure 14. Option Tables

Byte Number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	= 16 Byte Length
All Bound PIB			Reserved				H'16' Priority of All Bound PIB (Lowest)			Reserved			H'0' All Bound PIB Displacement		Reserved		
Background PIB	Address of BG Comm. Region		System LUB Index		Reserved		Priority of BG PIB (Note 4)		Address of Termination ECB, if any, or F'0'				X'0010' BG PIB Displacement		Reserved		
FG2 PIB (Note 1)	Address of Area Comm. Region (Note 2)		System LUB Index		Reserved		Priority of F2 PIB (Note 4)		Address of Termination ECB, if any, or F'0'				X'0020' F2 PIB Displacement		Reserved		
FG1 PIB (Note 1)	Address of Area Comm. Region (Note 2)		System LUB Index		Reserved		Priority of F1 PIB (Note 4)		Address of Termination ECB, if any, or F'0'				X'0030' F1 PIB Displacement		Reserved		
Attention PIB	Address of BG Comm. Region		0	0	Reserved		H'3' Priority of Attention PIB			F'0'			X'0040' Attention PIB Displacement		Reserved		
Quiesce I/O PIB			Reserved				H'2' Priority of Quiesce I/O PIB			F'0'			X'0050' Quiesce PIB Displacement		Reserved		
Supervisor PIB			Reserved				H'1' Priority of Supervisor PIB (Highest)			F'0'			X'0060' Supervisor PIB Displacement		Reserved		
Subtask PIB (Note 3)	Address of Area Comm. Region		System LUB Index		Reserved		Priority of Subtask (Note 4)		ECB Address for Subtask, or F'0'				PIB Displacement of Maintask		Reserved		

Note 1. Generated only if MPS is specified.

Note 2. Always background communications region except when MPS = BJF.

Note 3. Total of nine subtasks generated, and only when AP is specified.

Note 4. Will be filled in with halfword indicating the relative priority of task in the system (range H'4' to H'15', the lower the number the higher the priority).

Bytes 124 and 125 (X'7C'-'7D') of the communications region contain the address of the second part of the PIB table. Label PIB2AD identifies the first byte of the table. The second part of PIB table comes before the first part in storage allocation. Refer to Figure 1.

Figure 15. Second Part of Program Information Block (PIB) Table

PIB TABLE

Byte Number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	= 16 Byte Length
All Bound PIB	Flag Byte See A *	Reserved	SP Prefix		Branch Instruction to the All Bound Routine							Reserved					
Problem Program PIB (Note 1)	Flag Byte See B *	Cancel Code (Fig. 2)	SYSLOG ID (BG, F2, or F1)		NOP Instruction (CR)	Address of the Partition Save Area			Number of Core Blocks (Note 2)	Address of the Origin of the Partition			PIB Assign Flag See D *	User LUB Index	Number of Program LUBs	Flag Byte See C *	
Attention PIB	Flag Byte See E *	Cancel Code (Fig. 2)	SYSLOG ID (AR)		Branch Code (BC)	Active=Address of Save Area Inactive = Remainder of BC Instruction			Switch Byte See F *	Logical Transient Bucket (contains save area address)			X'07' See D *	Reserved	Address of the Logical Transient		
Quiesce PIB	Flag Byte See A *	Cancel Code (Fig. 2)	C'/&'		Branch Instruction to Quiesce I/O Routine				Scratch Byte X'00'	X'00'	Channel PUB Table Index Values X'04'	X'08'	X'0C'	X'10'	X'14'	X'18'	
Supervisor PIB	Flag Byte See A *	Cancel Code (Fig. 2)	SP Prefix		Branch Instruction to General Exit Routine				Address of SYSRES PUB		Length of Error Queue Entry		Constants to Clear Bytes 2-5 of CCB X'1F'	X'05'	X'00'	X'00'	
Subtask PIB for AP (Note 3)	Flag Byte See B *	Cancel Code (Fig. 2)	SYSLOG ID (BG, F2, or F1)		NOP Instruction	Address of the Save Area			Number of Core Blocks (Note 2)	Address of the Origin of the Main Task			PIB Assign Flag See D *	User LUB Index	Number of LUBs	Flag Byte See C *	

Note 1: Three problem program PIBs are built in this sequence when the MPS or BJF feature is selected as a generation option: { Background PIB, Foreground 2 PIB, Foreground 1 PIB }

When a batch-only environment is established at generation time, the All Bound and Foreground PIBs are excluded from the table, and only one (BG) problem program PIB is built. However, the X'20' bytes that F2 and F1 PIBs normally occupy (between PIBBG and PIBAR) are filled with 32 bytes of DIBs data.

Note 2: Number is in multiples of 2K for F2 and F1. BG is always 10K (X'0A').

Note 3: Total of nine subtask PIBs are generated, and only when AP is specified at generation time.

* See Figure 17 for flag byte expansions A, B, C, D, E,and F.

Bytes 90 and 91 (X'5A' - '5B') of the communications region contain the address of the first part of the PIB Table. Label PIBTAB identifies the first byte of the table.

Figure 16. First Part of Program Information Block (PIB) Table (See Figure 15 for Second Part)

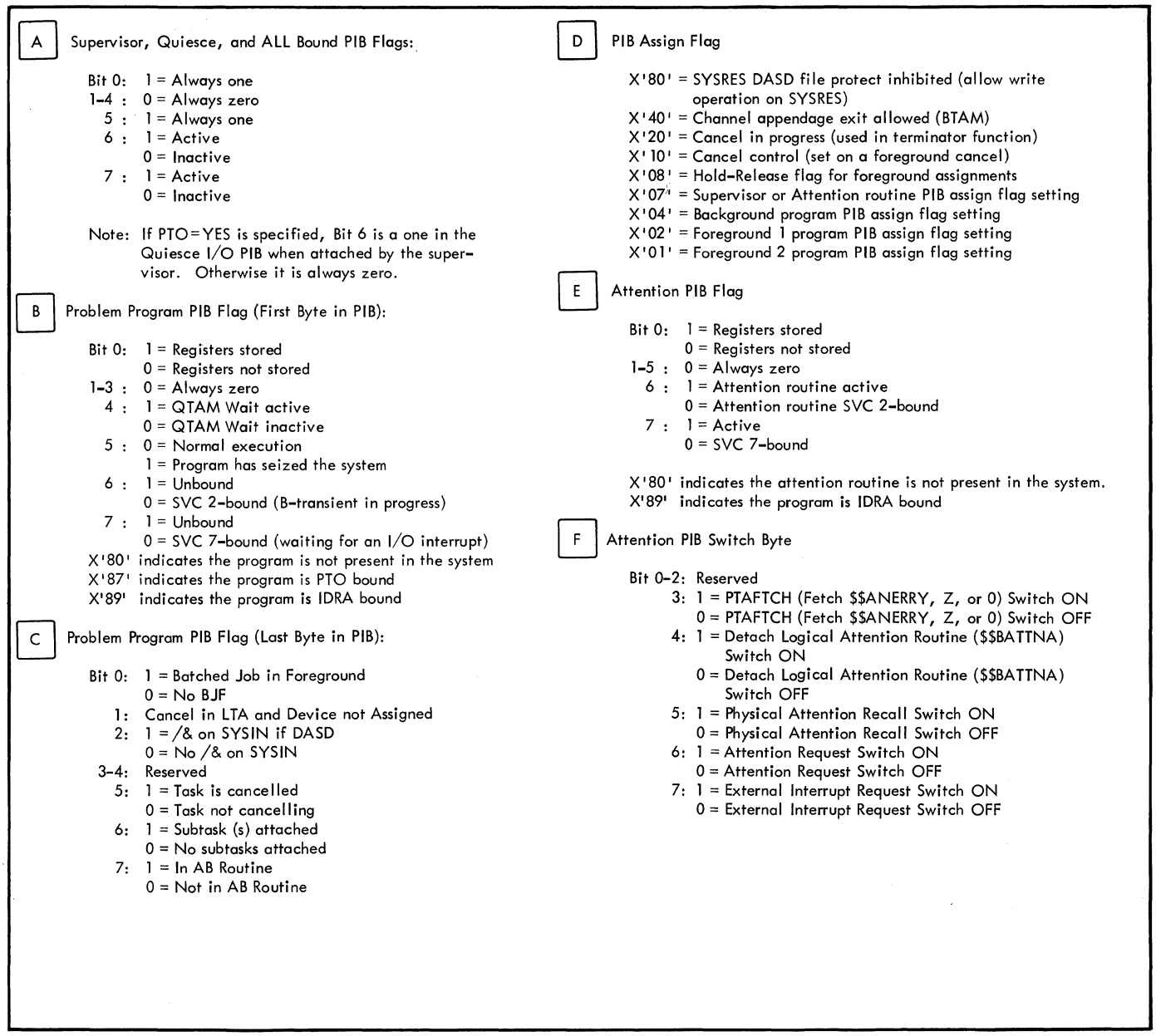
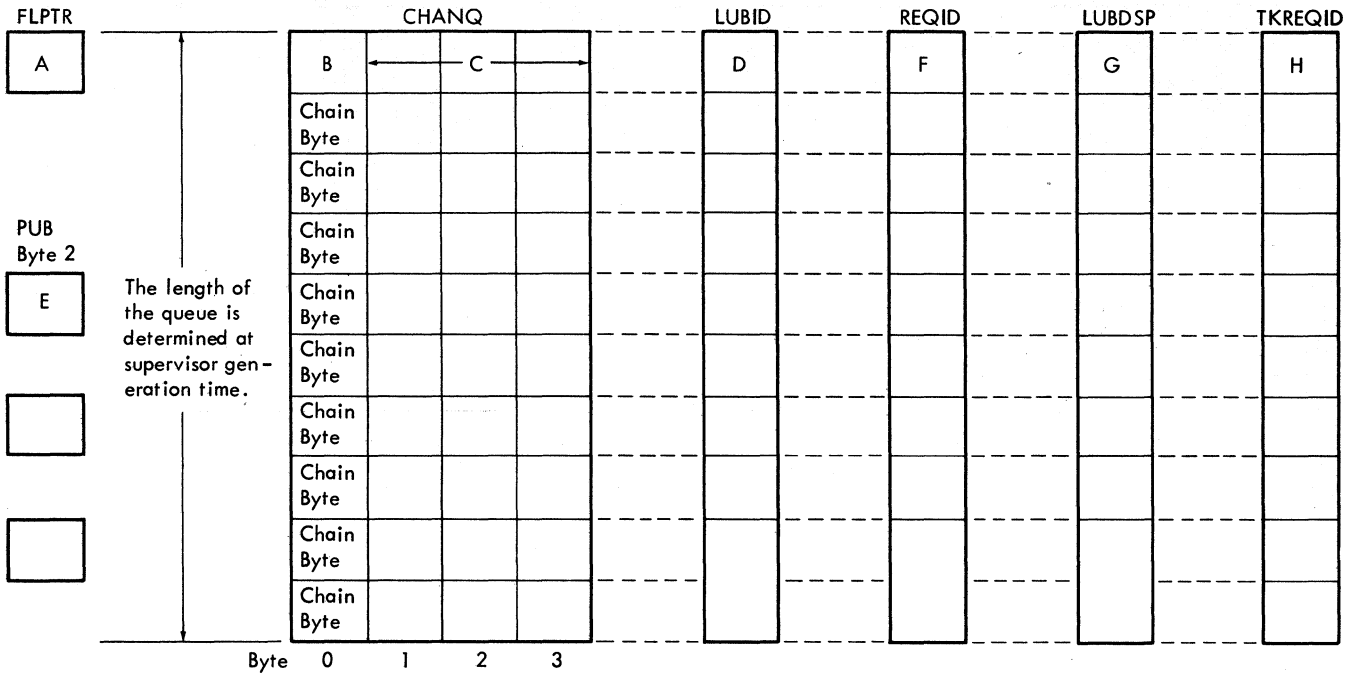


Figure 17. PIB Flag Expansions



KEY

A The free list pointer contains a displacement index to a free list entry within the channel queue. The free list is a group of entries that function in essentially the same manner as a device queue. When the free list pointer contains a hexadecimal FF, it indicates that no more free list entries are available.

B The first byte of the channel queue entry (chain byte) contains a pointer (displacement index) to the next channel queue entry for that device. A hexadecimal FF indicates the last channel queue entry for that device. New requests on a given device are queued at the end of a given device queue.

C CCB address for the specified device.

D A pointer (displacement index) to the entire LUB table identifying the logical unit making the I/O request. This is doubled to get the actual displacement into the full LUB table.

E Contains a pointer (displacement index) to the first channel queue entry for a specific device (Figure 19).

F Contains a code identifying the program making the I/O request. The one-byte entry is called a RID (Requestor Identification). The RID indicates what program the CCB belongs to. The RID is in the form X'nk'.

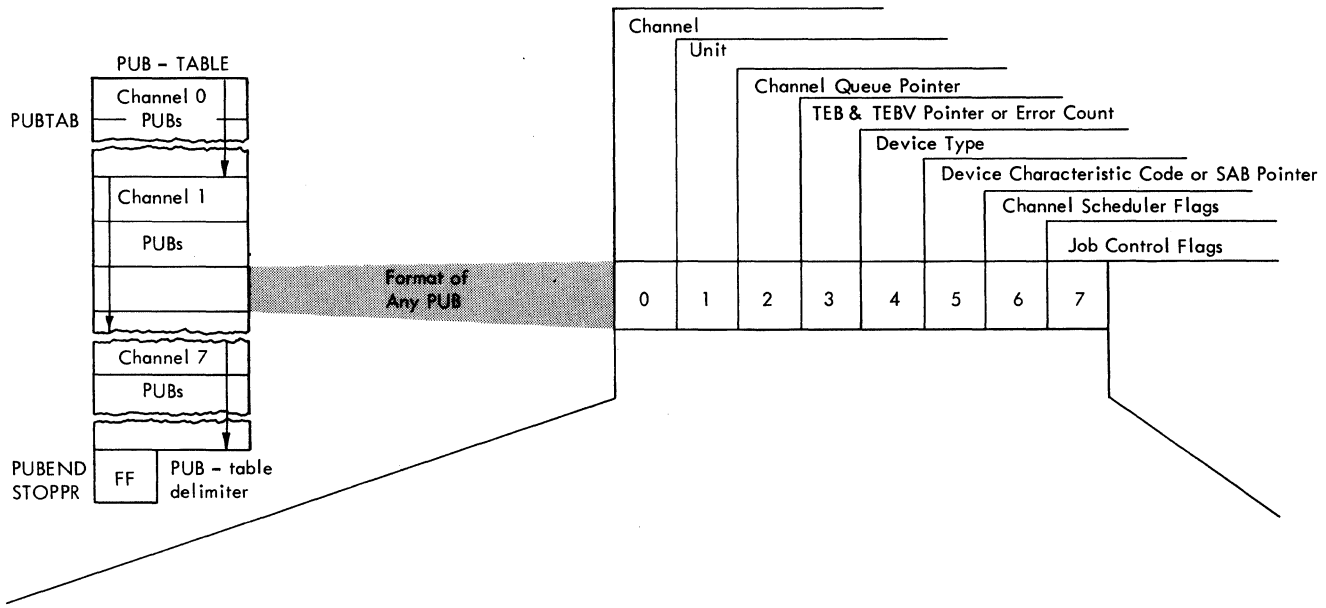
n = user-storage protection key (supervisor = 0, BG = 1, F2 = 2, F1 = 3).
k = 0 for all user requests and all supervisor CCBs, where n = 0.
k = 1 for supervisor CCBs to SYSLOG that bypass ID prefix.
k = 2 for a fetch CCB.
nk = FF for any unused channel queue entries.

G Contains X'FF' if the LUB is nonsystem class, or contains the displacement index within the partition LUB if it is a system class LUB.

H Contains X'FF', or the displacement into the PIB table for the PIB of the task requesting I/O.

Bytes 108-109 (X'6C'-'6D') of the communications region contain the address of the LUBID Table. Label LUBIDTAB identifies the first byte of the table. The addresses of the other tables are not at fixed locations. They can be found in the program listing cross-reference by using the labels CHANQ, REQIDTAB, LUBDSPTB, and TSKIDTAB.

Figure 18. CHANQ, LUBID, REQID, LUBDSP, and TKREQID Tables



BYTE 0 - Channel number. (Hex 0-7, FF=NULL)

BYTE 1 - I/O device unit number. (HEX 1F= 1052, HEX 80=magnetic tape unit 0 ...)

BYTE 2 - HEX 0, 1, 2, ... points to the first channel queue entry for this device.

BYTE 3 - If device is a magnetic tape unit* and TEBs and/or TEBVs are specified, this byte is a TEB/TEBV pointer (HEX 1, 2, 3 ...)

If device is a magnetic tape unit* but neither TEBs nor TEBVs are specified, this byte is an error counter.

If device is not a magnetic tape unit*, this byte is an error counter.

BYTE 4 - See Figure 26 for device type codes.

BYTE 5 - SS of the MODE=parameter in the DVCGEN macro for tape unit. (See Figure 27.)

For DASD with seek separation, this byte is used as the SAB pointer. With track hold but not seek separation supported, this byte contains a pointer to the Track Hold Table entry or X'FF' (with both SKSEP and TRKHLD specified, the track hold pointer is found in the SAB entry).

For MICR type devices, this byte indicates the external interrupt line is in use.

BYTE 6 -
Bit 0: 1 = Device busy
1: 1 = Switchable device
2: 1 = EOJ for SYSRDR or SYSIPT
3: 1 = I/O error queued for recovery
4: 1 = Operator intervention required
5: 1 = Device end posting required
6: 1 = Burst device on MPX
7: 1 = 7-track tape unit

BYTE 7 -
Bit 0-4: standard MODE assignment for 7-track tape (all ones if not tape, all zeros if device is down).
5: device is assigned to a background job
6: device is assigned to a foreground 1 job
7: device is assigned to a foreground 2 job

NOTE: A null is generated for each device to be supported by the supervisor. Standard physical unit assignments are made to the PUB table at supervisor generation time. PUBs are ordered by channel and priority within a channel.

Bytes 64 and 65 (X'40' - '41') of the communications region contain the address of the PUB table entry. Label PUNTAB identifies the first byte of the table.
*2400-series or 3420 Magnetic Tape Units or 2495 Tape Cartridge Reader (TEBs); 2400-series or 3420 Magnetic Tape Units only (TEBVs)

Figure 19. Physical Unit Block (PUB) Table

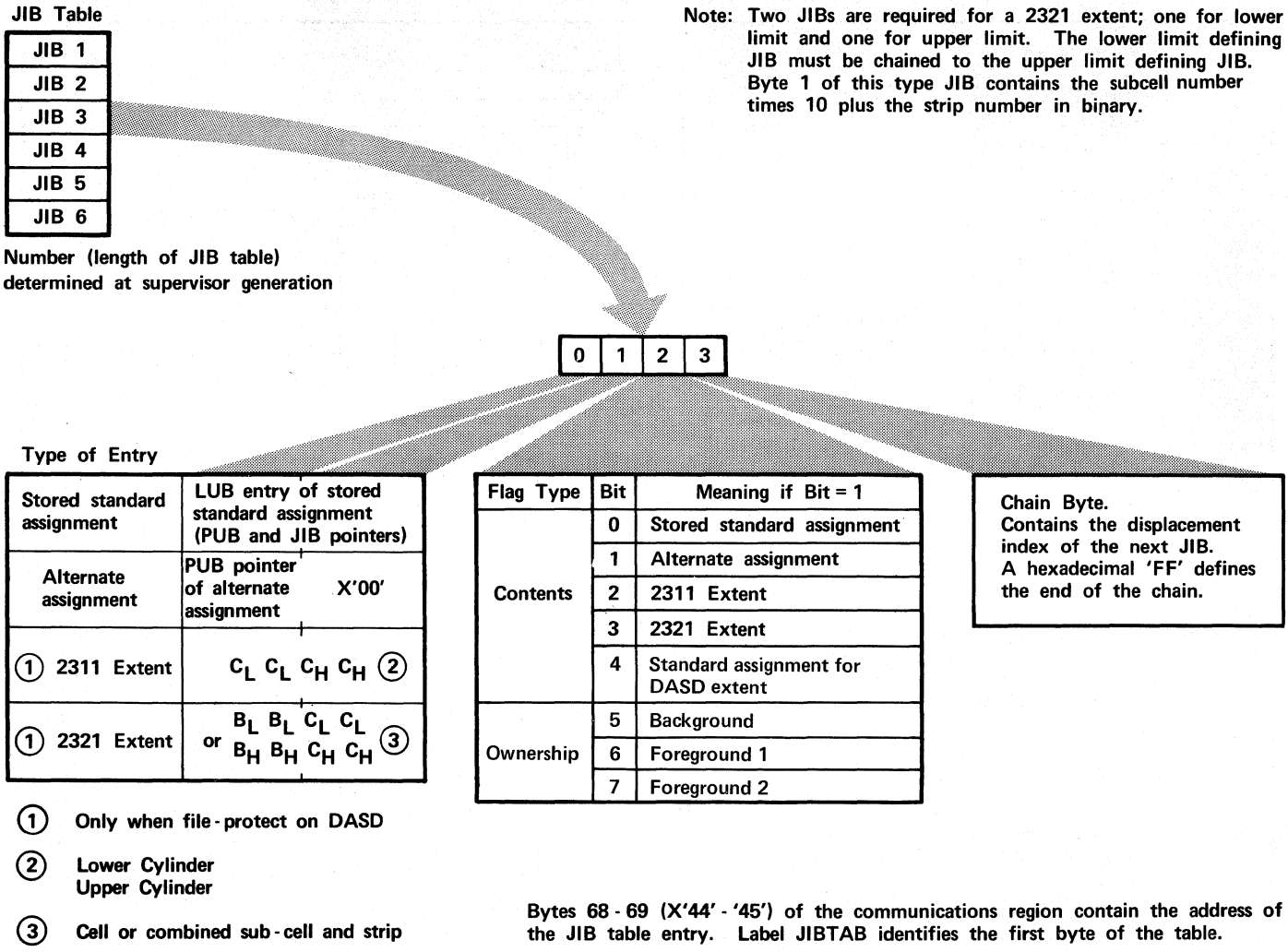
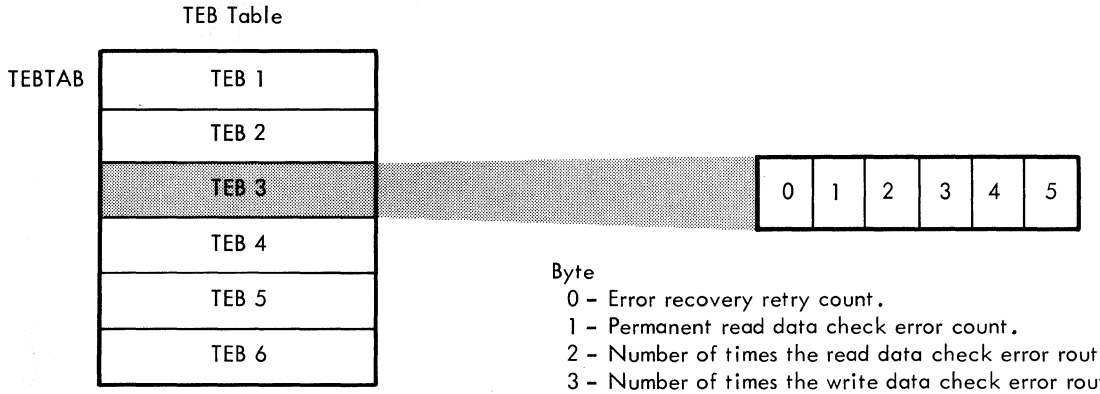


Figure 20. Job Information Block (JIB) Table



One TEB is generated for each 2400 series or 3420 magnetic tape or 2495 Tape Cartridge Reader unit if the FOPT macro contains the TEB = n parameter. Job control resets each TEB at normal or abnormal End-of-Job. An unused TEB contains HEX'FF0000000000'. A TEB is referenced from byte 3 of a magnetic tape unit PUB.

Bytes 70 and 71 (X'46'- '47') of the communications region contain the address of the TEB table entry. Label TEBTAB identifies the first byte of the table.

Figure 21. Tape Error Block (TEB) Table

Decimal Displacement	Label	Byte Length	Description
			(TEBV Status Block portion of TEBV Table, see Note 1)
0	TEBLEN	1	Length of TEBV Error Block (for each Error Block generated)
1	TSBLEN	1	Length of TEBV Status Block (4, 6, or 22 bytes, see Note 1)
2	EVARTH	1	EVA Read Error Threshold
3	EVAWTH	1	EVA Write Error Threshold
...	.....	...	... ...
4	TEBSTAT	1	DASD ESTV File Status
5	TEBUDC	1	ESTVFLE Label Update Counter
...	.....	...	... ...
6	TEBDEV	1	Data Set Device Code
7	UPXTNT	4	Disk Address of Upper Extent of Data Set (cchh)
11	TEBRPT	1	Number of Records per Track
12	NXTESR	5	Disk Address of Next Available Space for Data Record (cchhr)
17	ESTVLABL	5	Pointer to ESTVFLE Label in VTOC (cchhr)
...	.....	...	... ...
			(TEBV Error Block portion of TEBV Table, see Note 2)
22	TEBV	1	Status Indicator (giving status of posting and writing error conditions)
23		1	Usage Indicator (X'00'=TEBV Error Block in use, X'FF'=Error Block generated but not serving any tape unit)
24		1	Retry Counter
25		1	Permanent Read Errors
26		1	Temporary Read Errors
27		1	Temporary Write Errors
28		1	Erase Gaps
29		1	Noise Blocks
30		1	Permanent Write Errors
31		1	Cleaner Actions
32		2	Number of Start I/Os
34		6	Volume Serial Number (volume ID)
...	.....	...	... ...
40 (repeat bytes 22-39 for each TEBV Error Block)			

Figure 22. TEBV Table Showing Status Block and Error Blocks (Part 1 of 2)

Note 1: The TEBV (Tape Error Block by Volume) Table is composed of one Status Block and (n) Error Blocks, and is addressed symbolically by label TEBVTAB.

Supervisor generation options in the FOPT macro determine the size of the TEBV Status Block at generation time:

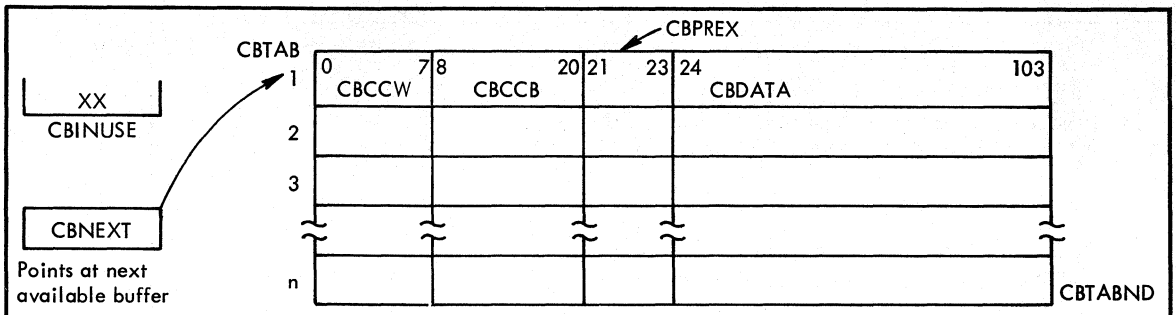
- When EVA is chosen without ESTV, the TEBV Status Block is four bytes long (bytes 0-3), followed by TEBV Error Blocks, so that bytes 4-21 are omitted.

- When ESTV output is to SYSLOG, the TEBV Status Block is six bytes long (bytes 0-5), followed by TEBV Error Blocks, so that bytes 6-21 are omitted.

- When ESTV output is to DASD, the TEBV Status Block is 22 bytes long (bytes 0-21, such as shown in this Figure), followed by TEBV Error Blocks.

Note 2: The number of TEBV Error Blocks generated corresponds to the (n) parameter of the FOPT macro for TEB, TEBV, or EVA options. A TEBV Error Block always contains 18 bytes, as shown in bytes 22-39 of this Figure. Therefore, the TEBV Table is composed of one TEBV Status Block (with its byte length dependent on supervisor generation options, as described in Note 1), followed by (n) number of 18-byte TEBV Error Blocks.

Figure 22. TEBV Table Showing Status Block and Error Blocks (Part 2 of 2)



The Buffer Table is a 104 multiplied-by-n byte area of main storage, where n is 1 - 50. Each buffer entry in the table is 104 bytes. CBNEXT is a half-word constant that points at the next available buffer entry. CBNEXT is initialized with the address of CBTAB and is incremented by 104 every time a buffer is used, so that it points at the next entry. When its value becomes greater than CBTABND, CBNEXT is reinitialized with the value CBTAB.

CBINUSE is a one-byte counter that contains the number of entries currently in use. It is incremented whenever a buffer is used and is decremented at dequeue time when the buffer becomes free.

Each buffer entry contains the following fields:

Displacement:	0	CCW moved from requestor core. The data address portion of the CCW is modified to point at the data portion of the buffer.
	8	CCB moved from requestor core. The CCW address in the CCB is modified to point at the CCW in the buffer.
	21	Prefix. SYSLOG ID moved from problem program PIB. The prefix is printed with the data to identify BG, F1, or F2 partitions.
	24	Data moved from requestor core.

Figure 23. Console Buffering (CBTAB) Table and Work Areas

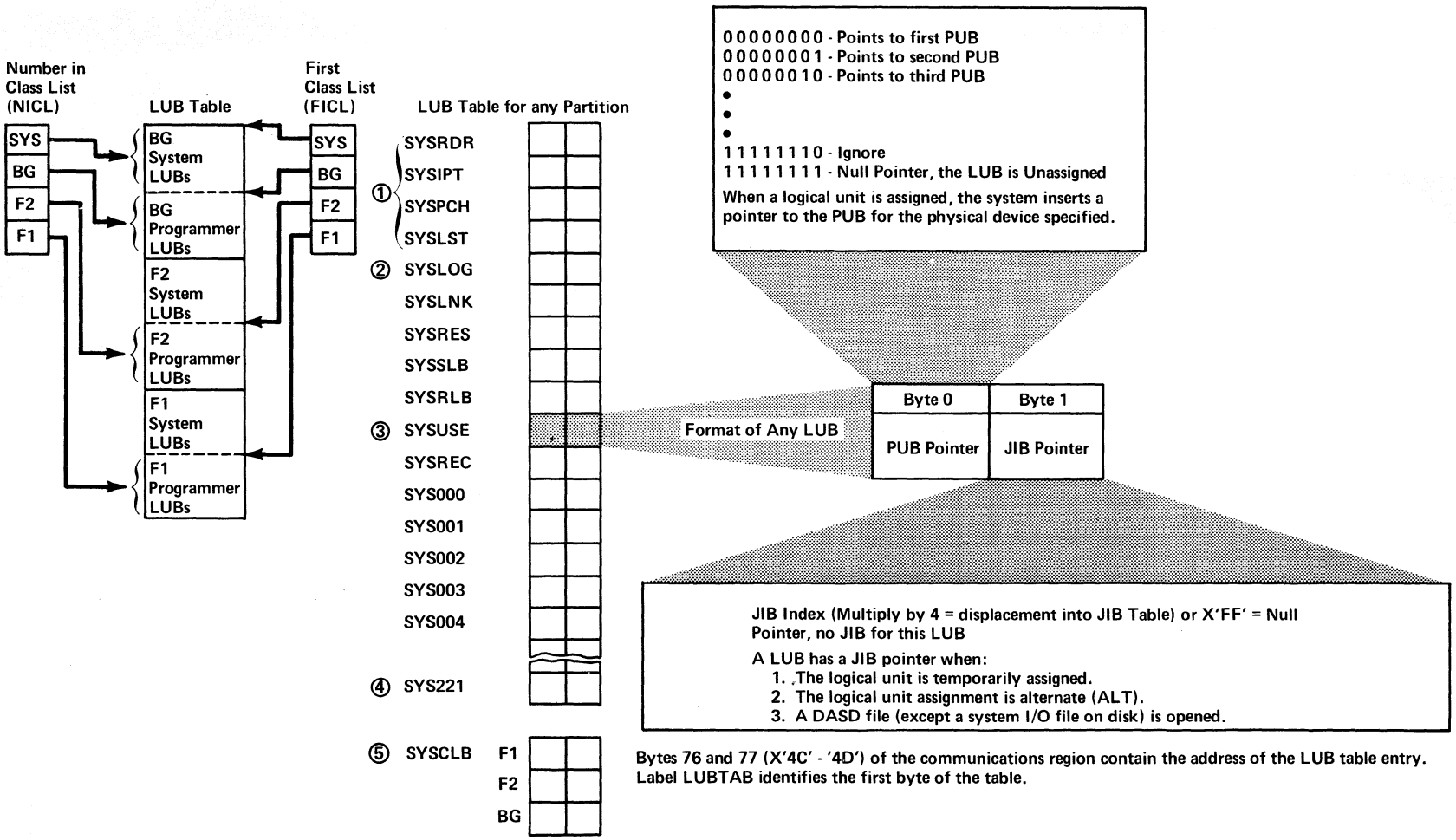


Figure 24. Logical Unit Block (LUB) Table

① When in Single Program Initiation mode (Foreground 1 or 2): Must be unit record device and can be referenced by the program.

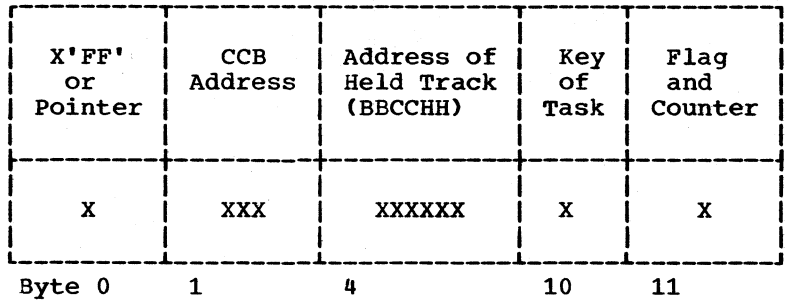
② When in Single Program Initiation mode (Foreground 1 or 2): Can be referenced by the program.

③ SYSUSE may be called SYSCTL in error recovery messages.

④ The maximum number of programmer logical units in the system is 222 if MPS = BJF, or 244 if MPS = YES or NO.

⑤ The SYSCLB (Private Core Image Library) LUB entry functions the same as other LUB entries, but is not part of the LUB Table. To locate the SYSCLB LUB in supervisor, perform the following steps:

1. Divide the PIK by 8.
2. Subtract the result in step 1 from the address of the PIB extension block.
3. If option AP = YES, the result of step 2 is the location of SYSCLB LUB. If option AP = NO, add 16 (for the all-bound PIBX) to the result of step 2.



X'FF' or Pointer	CCB Address	Address of Held Track (BBCCHH)	Key of Task	Flag and Counter
X	XXX	XXXXXX	X	X
Byte 0	1	4	10	11

Byte	Explanation
0	X'FF' or pointer to next available entry in the table. This is also placed in the PUB table, byte 5.
1-3	Address of CCB associated with the task requesting the hold.
4-9	Disk address of the track being held (in the form BBCCHH).
10	Key of the task owning the track.
11	Bit 0 on indicates a task is waiting for this track. 1-3 Unused 4-7 Counter of number of holds on the track.

Figure 25. Track Hold (TKHDTAB) Table

Card Code	Actual Device	Dev.Type X'nn'	Device Type
2400T9	9 - track 2400 Series Magnetic Tape Units	50	Magnetic Tape Units
	9 - track 3420 Magnetic Tape Units		
2400T7	7 - track 2400 Series Magnetic Tape Units		
	7 - track 3420 Magnetic Tape Units		
2495TC	2495 Tape Cartridge Reader	51	Tape Cartridge Reader
1442N1	1442N1 Card Read Punch	30	Card Readers - Punches
2520B1	2520B1 Card Read Punch	31	
2501	2501 Card Reader	10	Card Readers
2540R	2540 Card Reader	11	
2540P	2540 Card Punch	21	Card Punches
2520B2	2520B2 Card Punch	20	
1442N2	1442N2 Card Punch	22	
2520B3	2520B3 Card Punch	20	
1403	1403 Printer	40	Printers
1403U	1403 Printer with UCS Feature	42	
1404	1404 Printer	40	
1443	1443 Printer	41	
1445	1445 Printer	41	
1050A	1052,3210,or 3215 Printer - Keyboard	00	
UNSP	Unsupported Device	FF	Unsupported. No burst mode on mulitplexor channel
UNSPB	Unsupported Device	FF	Unsupported with burst mode on multiplexor channel
2311	2311 Disk Storage Drive	60	DASD
2314	2314 Direct Access Storage Facility	62	
	2319 Disk Storage Facility		
2321	2321 Data Cell Drive	61	
1412**	1412 Magnetic Character Reader	75	MICR - Magnetic Ink Character Recognition Devices and Optical Reader/Sorters
1419**	1419 Magnetic Character Reader	72	
	1255 Magnetic Character Reader		
	1259 Magnetic Character Reader		
1419P**	1419 Dual Address Adapter Primary Control Unit	73	
1419S**	1419 Dual Address Adapter Secondary Control Unit	74	
2701*	2701 Data Adapter Unit	D0	Teleprocessing lines A = SAD0 command when enabling the line B = SAD1 command when enabling the line C = SAD2 command when enabling the line D = SAD3 command when enabling the line
2702 (A, B, C, D)	2702 Transmission Control Unit	D1	
2703	2703 Transmission Control	D2	
2671	2671 Paper Tape Reader	70	Paper Tape Reader
1285	1285 Optical Reader	76	Optical Readers
1287	1287 Optical Reader	77	
1288	1288 Optical Page Reader		
1017	1017 Paper Tape Reader with 2826 Control Unit Model 1	78	Paper Tape Reader
1018	1018 Paper Tape Punch with 2826 Control Unit Model 1	79	Paper Tape Punch
2260	2260 or 2265 Display Station	C0	Display Station
7770	7770 Audio Response Unit	D3	Audio Response Units
7772	7772 Audio Response Unit	D4	
1017TP	1017 Paper Tape Reader with 2826 Control Unit Model 2	D5	Paper Tape Reader
1018TP	1018 Paper Tape Punch with 2826 Control Unit Model 2	D6	Paper Tape Punch

Note: The codes used in the DVCGEN macros are the same codes used in IPL statements.

* For other teleprocessing devices, see IBM System/360, DOS BTAM and QTAM PLMs, GY30-5001 and GY30-5002.

** This device type code is also used for the 1270/1275 optical reader/sorters.

Figure 26. Device Type Codes

Density (Byte per Inch)	Parity	Convert Feature	Translate	* SS Code
200	odd	on	off	10
200	odd	off	off	30
200	odd	off	on	38
200	even	off	off	20
200	even	off	on	28
556	odd	on	off	50
556	odd	off	off	70
556	odd	off	on	78
556	even	off	off	60
556	even	off	on	68
800	odd	on	off	90
800	odd	off	off	B0
800	odd	off	off	B8
800	even	off	off	A0
800	even	off	on	A8
800	dual density nine-track			C8
1600	dual density nine-track			C0

*Refer to PUB table, byte 5 (Figure 19).

Figure 27. Density Data

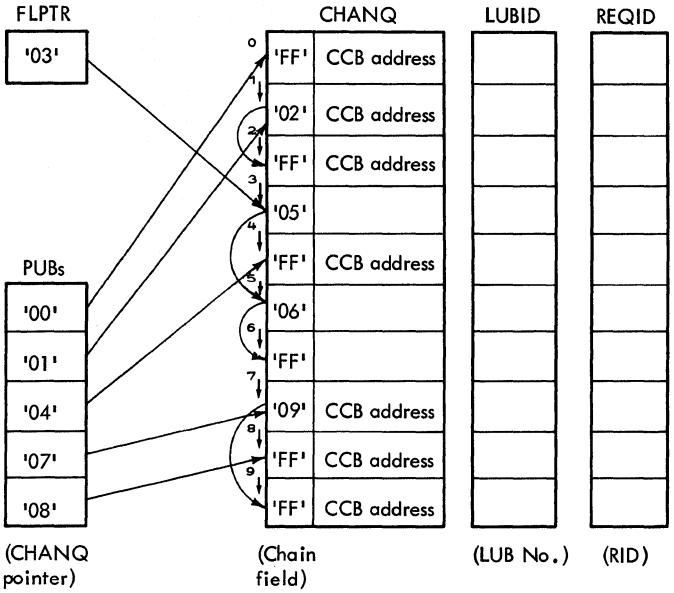


Figure 28. Example of the CHANQ Table Operation

Priority Table		
Sample Status	PIB Tables	MVCFLD
X'84'	Supervisor task PIB	X'60'
X'84'	Quiesce I/O task PIB	X'50'
X'80'	Attention task PIB	X'40'
X'83'	† Foreground 1 program PIB	X'30'
X'82'	† Foreground 2 program PIB	X'20'
X'83'	Background program PIB	X'10'
X'85'	†All bound PIB	X'00'

1. Test status flags in order specified by priority table.
2. Select 1st PIB for which the TRT function is not X'00'.

PIB Flags During Task Selection		Table of Selection Criteria	
Meaning of Status	Flag	Label	TRT Function
Detached	X'80'	TRTMSK	X'00'
Waiting for B-transient area	X'81'	TRTLTK	X'00' or X'03' (Note 1)
Waiting for CCB or TECB	X'82'		X'00'
Ready to run	X'83'	TRTRUN	X'03' or X'00' (Note 2)
Inactive SUPVR or Quiesce I/O	X'84'		X'00'
Active SUPVR, Quiesce I/O, or All bound	X'85'		X'05'

Note 1: X'00' when the B-transient area is in use as indicated by the Logical Transient Key (LTK).

Note 2: X'00' when a task has seized the system. That task's status flag will equal X'84' or X'85'.

† These PIBs are generated for MPS option only.

Figure 29. Task Selection Procedure

Displacement	Label	Description
0-15	(ACCTCOMN) ACCTSVRG	Temporary register save area.
16-17	ACCTSVRX	Save area for remainder of overhead counter times distributed by partition on exit.
18-19	ACCTSVRE	Save area for remainder of all-bound counter times distributed by partition on entry.
20-23	ACCTPCNT	Count of partitions using JAI.
24	ACCTSAID	Owner of physical transient area*.
25	ACCTFAID	Interrupted program*.
26	ACCTRAID	Active program*.
27	ACCTSWCH	Accounting switches: if bit = 1, true; if bit = 0, not true. bit 0 - cancel accounting bit 1 - no active partitions bit 2 - catalog in process bit 3 - alternate label area bit 4 - IPL indicator bit 5 - $JOBACCT in F1 bit 6 - $JOBACCT in F2 bit 7 - $JOBACCT in BG
28-31	ACCTIME	Start time of current accounting interval, in complement format.
32-33	ACCTRESC	Reserved.
34-35	ACCTUSEP	Address of user save area (ACCTUSER).
36-39	ACCTBLES	Address of BG Job Accounting Table.
40-43	- - - - -	Address of F2 Job Accounting Table if BJF; otherwise zero.
44-47	- - - - -	Address of F1 Job Accounting Table if BJF; otherwise zero.
48-53	ACCTSEAS	Seize blocks; serve as overlapped Event Control Blocks. TS Bit | (reserved) | Wait Bit 1 | PIK 1 | Wait Bit 2 | PIK 2 1st ECB 2nd ECB TS Bit: X'00' = no $JOBACCT running X'FF' = $JOBACCT active
54-55	ACCTUSEL	Length of user save area, set with 4th operand of global AG39.

*Note: X'00' = all bound, X'10' = BG, X'20' = F2, X'30' = F1, X'40' = overhead and FG if SPI.

Figure 30. Job Accounting Interface Common Table (ACCTCOMN)

Displacement	Label	Description
0-3	ACCTWK1* (ACCTABLE)*	Work area used in SIO update.
4-7	ACCTWK2	Work area used with ACCTWK1 in start/stop time routine
8-11	ACCTSVPT	Job card pointer; address of job card field following jobname.
12	ACCTPART	ID of partition in charge (partition switch name).
13	ACCTRES2	Reserved.
14-15	ACCTLEN	Length of SIO area = 6n + 1, where n = number of devices for this partition in SYSGEN option JA = (n1, n2, n3).
16-21	ACCTLOAD	Label area instruction; moves JAI label area address to OPEN/CLOSE transients.
22-23	ACCTRES3	Reserved.
24-27	ACCTLADD	Address of alternate label area.
28-31	ACCTCPUT	Counter for CPU time elapsed in a jobstep, counted in 300ths of a second.
32-35	ACCTOVHT	Counter for overhead time; time not charged to any partition.
36-39	ACCTBNDT	Counter for all-bound time; system wait state time divided between running partitions.
40-47	ACCTSVJN	Save area for job name during simulated EOJ.
		JOB ACCOUNTING TABLE (user's portion of Partition Table)
48-55	ACCTJBNM	Job name; taken from job card.
56-71	ACCTUSRS	User information; 16 bytes from Job card.
72-73	ACCTPTID	Partition ID; 'BG', 'F2', or 'F1' in EBCDIC format.
74	ACCTCNCL	Cancel code; see Cancel Codes and Messages (Figure 32).
75	ACCTYPER	Type of record: 'S' = job step, 'L' = last step of job.
76-83	ACCTDATE	Date in format specified at SYSGEN (MM/DD/YY or DD/MM/YY).
84-87	ACCTSTRT	Start time of job, in packed decimal (OHHMMSSF; F = sign).
88-91	ACCTSTOP	Stop time of job, in same format as ACCTSTRT
92-95	ACCTRES	Reserved.
96-103	ACCTEXEC	Phase name; taken from execute card.
104-107	ACCTHICR	High core address of active program phase, from COMREG.
108-111	ACCTIMES	CPU time elapsed in a job step; counted in 300ths of a second.
112-115	------	Overhead time; elapsed time not charged to any partition, in 300ths of a second.
116-119	------	All-bound time; system wait state time divided between running partitions, in 300ths of a second.
120	ACCTSIOS	SIO tables: 6 bytes for each device specified by SYSGEN options, as follows: 2 bytes for device address (Ocuu), 4 bytes for count of SIOs in current jobstep.
———	------	Overflow byte: normally X'20', but is X'30' if more devices are used within a partition than specified by SYSGEN options.

*Note: DSECT ACCTABLE symbolically addresses the JAI Partition Tables with labels as shown. Each partition in which JAI is supported has its own JAI Partition Table, labeled ACCTBG, ACCTF2, ACCTF1, for active partitions BG, F2, and F1 respectively.

Figure 31. Job Accounting Interface Partition Table (ACCTxx*)

0 LD00SLOT ($$RAST00) (1) (2)	4 LD01SLOT ($$RAST01) (3)	8 LD02SLOT ($$RAST02) (5)	12 LD03SLOT ($$RAST03) (5)	16 LD04SLOT ($$RAST04) (5)	20 LD05SLOT ($$RAST05) (5)	24 LD06SLOT ($$RAST06) (5)	28 LD07SLOT ($$RAST07) (5)	32 LD08SLOT ($$RAST08) (5)
36 LD09SLOT ($$RAST09) (5)	40 LD10SLOT ($$RAST10) (4)	44 LD11SLOT ($$RAST11) (5)	48 RASCCB Residual Count	50 RASTIB Transmission information	52 - - - - CCW Status bytes	54 - - - - SYSRES LUB	56 RASCCBF RAS CCB indicator	57 - - - - RAS Fetch CCWs address
60 - - - - CCW stored address	64 RASCCWS RAS seek CCW	72 RASRCG RAS search CCW	80 RASTIC TIC CCW	88 RASREAD CCW to read module into RTA	96 RASEEK Seek Address	103 RTAOWN Index into load list for RTA owner	104 RASRETR RTA return address after I/O operation	106 MCPIK PIK of task interrupted by machine check
108 RASIOA I/O routine address	110 RASFCHA RTA fetch routine address	112 RTAID RTA I/O request ID	113 ERPID Return load index for WTOR request	114 ERPIBA ERPIB queue address	116 RASDEQA CCB DEQ routine address	118 XCANRASA RAS cancel routine address	120 CCENTADR Channel Check entry address	122 RASRES SYSRES I/O address
124 RASREC SYSREC I/O address	126 RASLOG SYSLOG device address	128 RASEMIOA Emergency SIO address	130 RASCQDSP CCB look-up routine address	132 SUPRETR Save area for registers 9 and 10	140 SUPBB Base address (X'1000') for supervisor	144 SUPBC Base address (X'2000') for supervisor	148 SUPBD Base address (X'3000') supervisor	
152 (HIR - Hardware Instruction Retry accumulators)				164 (ECCMAIN - Main storage error accumulators)				
152 HIRACNT HIR accumulated count	154 HIR1CNT Count threshold value	156 HIR1TME Time of day for first error of group	160 HIRLTME Time threshold in clock units	164 ECMACNT Accumulated ECC count for main storage	166 ECMLCNT Count threshold value	168 ECM1TME Time of day for first error of count	172 ECMLTME Time threshold in clock units	
176 RESTARTA Disk restart address	178 RESTARTP PUB address of unit to be restarted	180 MCMODE Mode status for machine checks	181 BUFDEL Count of buffers deleted	182 RASMSG1 Message byte 1 (6)	183 RASMSG2 Message byte 2 (7)	184 EOR Records/track for SYSREC	185 EOT Tracks/cylinder for SYSREC	186 CCDEVT X'00'

Figure 32. RMS Monitor Table (RASTAB) (Part 1 of 2)

Notes:

(1) Areas labeled LDxxSLOT (bytes 0 - 47) are called the Load List and each of the 12 entries are formatted as follows:

BYTE	0	1 2 3
	Flag Byte	Cylinder - Head - Record (disk address of R-transient in the core image directory)

(2) LD00SLOT flag byte:

Bit	Flag	Description
0	X'80'	$$RAST00 module activated.
1	X'40'	Machine check analysis to be performed.
2	X'20	Channel check analysis to be performed.
3	X'10'	Active I/O units are valid.
4	X'08'	System termination situation.
5	X'04	Reserved.
6	X'02'	Reserved.
7	X'01'	Attempt made to record in system termination situation.

(3) LD01SLOT flag byte:

Bit	Flag	Description
0	X'80'	$$RAST01 module activated.
1	X'40'	Build and record channel check records.
2-7	- -	Reserved.

(4) LD10SLOT flag byte:

Bit	Flag	Description
0	X'80'	$$RAST10 module activated.
1	X'40'	Refetch calling module after issuing message.
2-7	- -	Reserved.

(5) LDxxSLOT flag byte:

Bit	Flag	Description
0	X'80'	$$RASTxx module activated; that is, should be fetched.
1-7	- -	Reserved.

(6) RASMSG1:

Bit	Flag	Description
0-3	- -	Reserved.
4	X'08'	Timer damage.
5	X'04'	ECC in Quiet mode.
6	X'02'	Reserved.
7	X'01'	MCAR repair failed.

(7) RASMSG2:

Bit	Flag	Description
0	X'80'	Check damage.
1	X'40'	Last track on SYSREC.
2	X'20'	C40 buffer pages deleted.
3	X'10'	Soft machine checks disabled.
4	X'08'	ECC MCI disabled.
5	X'04'	SYSREC full-run EREP.
6	X'02'	Error on SYSREC at BBCCHHR.
7	X'01'	Soft machine check.

Figure 32. RMS Monitor Table (RASTAB) (Part 2 of 2)

APPENDIX D: LABEL INFORMATION FORMAT ON SYSRES

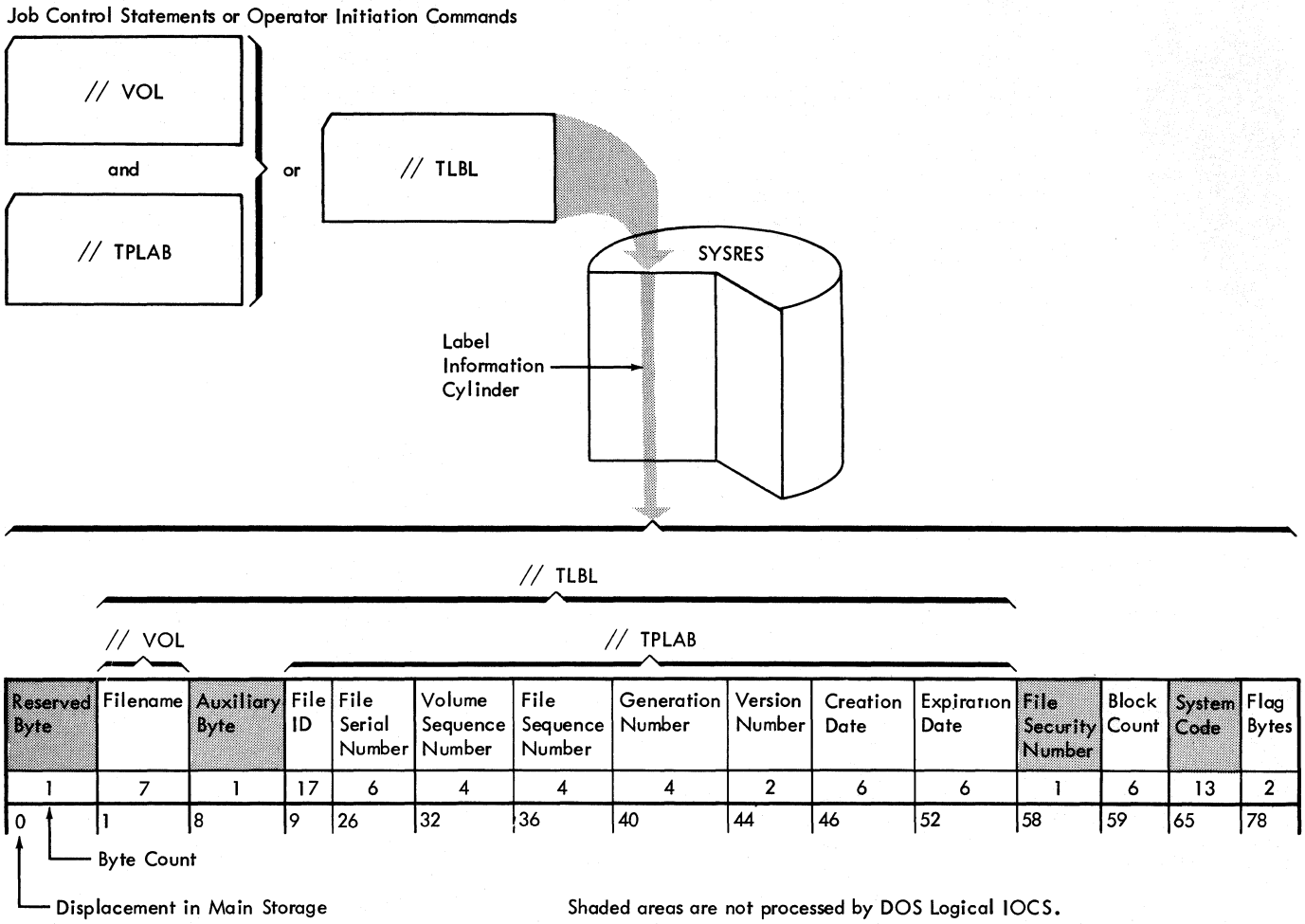


Figure 33. Format of SYSRES Tape Label Information

1	2	3		4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19		Field
DLBL-EXTENT Indicator	Filename	DA/IS Switch	File ID		Format ID	File Serial Number	Volume Seq. No.	Creation Date	Expiration Date	Reserved	Open Code	System Code	Volume Serial Number	EXTENT Type	EXTENT Seq. No.	Extent Lower Limit	Extent Upper Limit	Logical (Symbolic) Unit Address	2321 Lower Cell	2321 Upper Cell	Another Extent if DA or ISFMS
1	7	1	44		1	6	2	3	3	2	1	13	6	1	1	4	4	2	1	1	Bytes
0	1	8	9		53	54	60	62	65	68	70	71	84	90	91	92	96	100	102	103	Displacement

Field	Name	Description
1.	DLBL-EXTENT	SD Bit 0: 1 = Next extent on a new pack. Bit 1: 1 = Last extent. Bit 2: 1 = Bypass extent. Bit 3: 1 = New volume on same unit. Bit 4: 1 = Extent limits omitted. Bit 5: 1 = Extent converted to DASD address. Bit 6: 1 = No EXTENT/XTENT card. Bit 7: 1 = Unused. DA or ISFMS Number of extents.
2.	Filename	
3.	DA/IS Switch	Bits 0-3: Unused. Bit 4: 1 = Extent limits omitted. Bit 5: 1 = Extent converted to DASD address. Bit 6 & 7: Unused.
4.	File ID	File identifier including generation and version numbers. If field is missing on DLBL card, Filename padded with blanks is inserted.
5.	Format ID	Numeric 1 is inserted.
6.	File Serial No.	Volume serial number from first extent.
7.	Volume Seq. No.	Always initialized to X'0001'.
8.	Creation Date	Initialized with 3 bytes of X'00'.
9.	Expiration Date	If date is in the form YYDDD, it is converted to YDD. If date is in retention period form, 1 to 4 characters, the field is padded with binary zeros.
10.	Reserved	The retention period, if specified is converted to a 2-byte number and inserted in this field.
11.	Open Code	DLBL type: S = Sequential D = Direct Access C or E = Indexed sequential File Management System where: C = Load create function E = Load extend function
12.	System Code	Initialized to contain DOS/360 VER 3. This field is not processed by DOS.
13.	Volume Serial No.	Volume serial number for extent.
14.	Extent Type	Same codes as in Format - 1 label: X'00' = Next three fields do not indicate any extent. X'01' = Prime data area (ISFMS) or consecutive area, etc., (that is the extent containing the user's data records). X'02' = Overflow area of an ISFMS file. X'04' = Cylinder index or master index of an ISFMS file. X'40' = User label track area. X'8n' = Shared cylinder indicator, where n = 1, 2, or 4.
15.	Extent Seq. No.	Number of extent as determined by the extent card sequence.
16. & 17.	Extent Lower and Upper Limits	Before the OPEN, DLBL/EXTENT information is in the relative track form of HHNNT followed by three bytes of binary zeros. HH = Relative (to 0) start address in tracks. NN = Number of tracks. T = 0 or upper track number for split cylinder in SD files. Following an OPEN on DLBL/EXTENT cards, or whenever DLAB/XTENT cards are used, the extent lower and upper limits are each in the CCHH format.
18.	Logical (Symbolic) Unit Address	This 2-byte field identifies the logical unit with the same code as that used in a CCB. The first byte identifies the unit class: X'00' = System Logical Unit X'01' = Programmer Logical Unit The second byte identifies the logical unit within its class. Thus X'0003' denotes SYSLST and X'0103' denotes SYS003.
19.	2321 Lower Cell 2321 Upper Cell	2321 extent lower and upper cell limit. This 2-byte field contains zeros for 2311/2314/2319 disk.

Note: For Sequential Disk files, a complete 104-byte block is repeated for each new EXTENT.
For Direct Access and ISFMS files, only fields 13 through 18 are repeated for each EXTENT.

Figure 34. Format of SYSRES DASD Label Information

APPENDIX E: PROGRAM KEY DEFINITIONS

PID (Partition Identifier)

The PID is a halfword long, consisting of a zero value in the high-order byte and the key value in the low-order byte. The key value is the key of the partition that was last enabled for interrupts. Key values are X'00', X'10', X'20',...X'60' for All Bound, BG, F2, F1, Attention, Quiesce I/O and Supervisor, respectively.

PID is defined in an AP (asynchronous processing) supervisor only and is found at byte-displacement 46 in the communications region. (This halfword is named PIK in a non-AP supervisor. See PIK below.)

PIK (Program Interrupt Key)

The PIK is a halfword long, consisting of a zero value in the high-order byte and the key value in the low-order byte. The key value is the key of the program that was last enabled for interrupts.

When an interrupt occurs, the value of the PIK indicates to the supervisor which program (task) was interrupted. It can also be used by transient programs and problem programs to determine if they are running as BG, F1, or F2, or, in the AP supervisor, if the last interrupted task was a maintask or a subtask.

The value of the PIK equals the displacement from the beginning of the PIB table to the PIB entry for the program (task). For BG, F2, and F1 tasks, this value equals the storage protect key multiplied by sixteen.

In a non-AP supervisor, PIK is found at displacement 46 in the communications region. In an AP supervisor PIK is found at displacement 18 in the communication region extension. (Also see PID.)

Task	PIK Value
All Bound*	X'00'
BG	X'10'
F2*	X'20'
F1*	X'30'
AR	X'40'
Quiesce I/O	X'50'
Supervisor	X'60'
Subtask**	X'70' - X'F0'

*Multiprogramming generation option only.

**Asynchronous processing generation option only. A total of nine subtask PIBs is generated, thus the displacement hex 70-F0 indicates the maximum range.

The PIK is set by task selection within the general exit routine. The fetch routine sets the PIK to X'60', because it enables itself for interrupts and because it gets control directly from the SVC interrupt routines. Like other completely disabled supervisor routines, the SVC interrupt routines do not change the PIK from the value it had when the interrupt occurred that transferred control.

LID (Logical Transient Identification)

The LID contains the same value as the PIK when the logical transient area is in use (i.e., the LID identifies ownership of the logical transient area). When this transient area is free, the halfword LID contains zeros. The SVC 2 routine sets the LID, and the SVC 11 routine resets it to zero. LID is defined only in an AP supervisor. See also LTK, PIK, and PID.

LTK (Logical Transient Key)

The LTK has the same value as the PIK when the logical transient area is in use. When the transient area is free, the LTK equals zero. The SVC 2 routine sets the LTK, and the SVC 11 routine resets it to zero.

RID (Requestor Identification)

See Figure 18, REQID (Item F).

RIK (Requestor I/O Key)

When a supervisor routine (fetch or physical transient) issues an SVC 0 or SVC 15, the routine puts the value to be used in the CAW storage protect key into the high-order digit of the second byte of the RIK halfword. When this value is zero, the low order digit has these special meanings:

RIK	Meaning
X'01'	This is a SYSLOG I/O request. The channel scheduler is not to type a SYSLOG ID prefix.
X'02'	This has been a fetch I/O request. This special code is required by ERP to recognize fetch requests.

Fetch always sets a X'02' in the RIK. ERP transients put the key of the program requiring ERP into the RIK, when the ERP is a retry of a user EXCP and the ERP transient requires control to return to itself.

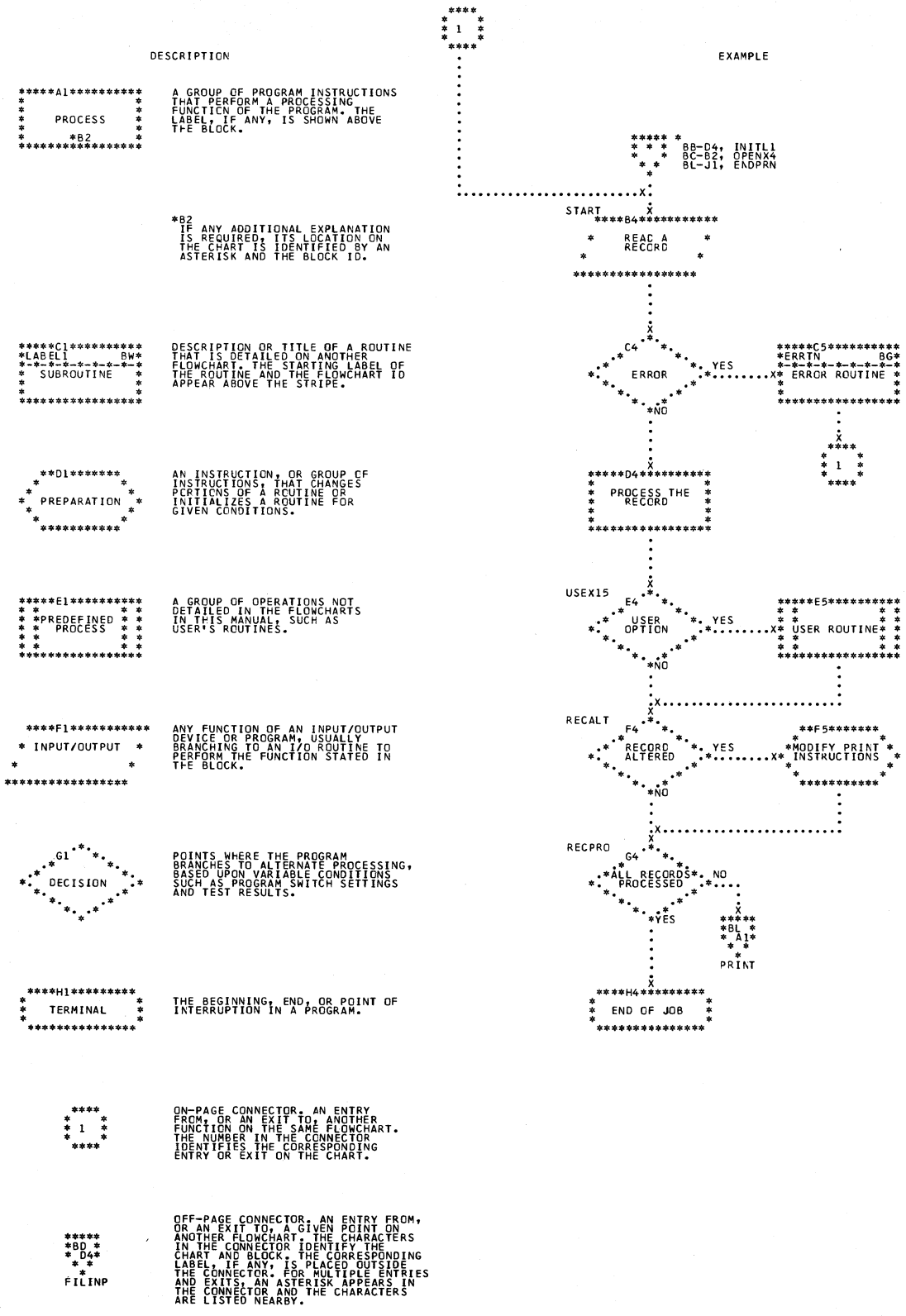
Physical transients put a X'01' into the RIK when they are doing a SYSLOG I/O. The PIK for physical transients has a value of X'06', therefore the channel scheduler would type "SP" (supervisor ID) as the SYSLOG ID. The physical transients put the ID of the program referred to by the message into the message.

FIK (Fetch I/O Key)

Used by the fetch to validate the phase name address and load address. FIK has the following values:

SVC	Contents
1	Key of the problem program requestor.
2	0
3	0
4	0 if the transient issued the SVC 4. Key of the problem program if not a transient.

APPENDIX F: EXPLANATION OF FLOWCHART SYMBOLS



The index gives the relationship of core-image phase names, relocatable module names, microfiche labels, and microfiche identification numbers.

An asterisk indicates the microfiche label. If the microfiche label differs from both the phase and the module name, it is so indicated in parentheses.

When a phase or module takes up more than one microfiche card, the identification number of only the first card is shown.

For the complete microfiche cross-reference index, see Introduction to DOS Logic, listed in the Preface.

Core Image Phase Name	Relocatable Module Name	Card ID
$$BATTNA*	None	CTL.044.00
$$BATTNB ($$BATTNA)	None	CTL.044.00
$$BATTNC ($$BATTNA)	None	CTL.044.00
$$BATTND ($$BATTNA)	None	CTL.044.00
$$BATTNE ($$BATTNA)	None	CTL.044.00
$$BATTNF ($$BATTNA)	None	CTL.044.00
$$BATTNG ($$BATTNA)	None	CTL.044.00
$$BATTNH ($$BATTNA)	None	CTL.044.00
$$BATTNI*	None	CTL.045.00
$$BATTNJ*	None	CTL.046.00
$$BATTNK*	None	CTL.047.00
$$BATTNL*	None	CTL.048.00
$$BATTNM*	None	CTL.049.00
$$BATTNN*	None	CTL.050.00
$$BATTNO*	None	CTL.051.00
$$BATTNP*	None	CTL.052.00
$$BATTNQ*	None	CTL.052.50
$$BATTNR*	None	CTL.052.60
$$BATTNS*	None	CTL.052.70
$$BCCHHR*	None	CTL.052.80
$$BDUMP*	None	CTL.072.00
$$BDUMP*	IJBDUMPS	CTL.072.00
$$BDUMP*	IJBDUMPT	CTL.072.01
$$BDUMPB*	IJBDMPBS	CTL.073.00
$$BDUMPB*	IJBDMPBT	CTL.073.01
$$BDUMPD*	IJBDMPDS	CTL.074.00
$$BDUMPD*	IJBDMPDT	CTL.074.01
$$BDUMPF*	IJBDMPFS	CTL.075.00
$$BEOJ*	None	CTL.076.00
$$BEOJ1*	None	CTL.077.00
$$BEOJ2*	None	CTL.078.00
$$BEOJ2A*	None	CTL.079.00
$$BEOJ3*	None	CTL.080.00
$$BEOJ4*	None	CTL.081.00
$$BEOJ5*	None	CTL.082.00
$$BESTVA*	None	CTL.085.00
$$BESTVB*	None	CTL.086.00
$$BESTVC*	None	CTL.087.00
$$BESTVD*	None	CTL.088.00
$$BESTVE*	None	CTL.089.00
$$BESTVF*	None	CTL.089.50
$$BILSVC*	None	CTL.090.00
$$BPCHK*	None	CTL.172.00
$$BPDUMP*	IJBPDMPS	CTL.173.00
$$BPDUMP*	IJBPDMPT	CTL.173.01
$$BPDUM1*	IJBPDUMS	CTL.174.00
$$BPSW*	None	CTL.175.00
$$BSDRUP*	None	CTL.185.00
$$BSYSWR*	None	CTL.186.00
$$BTERM*	None	CTL.187.00

For a more complete list of data processing terms, refer to IBM Data Processing Techniques, A Data Processing Glossary, GC20-1699.

ASCII (American National Standard Code for Information Interchange): A 128-character, 7-bit code. The high-order bit in the System/360 8-bit environment is zero.

CCH (Channel Check Handler): A feature that assesses System/370 channel errors to determine if the system can continue operations.

channel inboard error: An error that occurs between one I/O device and the central processing unit.

core image library: A SYSRES area (or a device of the same type as SYSRES) that stores programs processed by the linkage editor. Each program is in a form that can be executed in main storage.

core wrap mode: The method of operation that records the events of a trace in main storage. It is the default process when no output device for the trace has been specified. The contents can be displayed by either a dump program or manually from the console.

data set security: A feature that provides protection for disk files. A data secured file cannot be accidentally accessed by a problem program.

Disk Operating System: A disk resident system that provides capabilities for 16K and larger IBM System/360 and System/370 systems.

DOS Volume Statistics: A facility that monitors and records the number of temporary read and write errors on currently accessed tape volumes. This facility has two options, Error Statistics by Tape Volume (ESTV) and Error Volume Analysis (EVA).

EREP (Environmental Recording, Editing, and Printing): A program that processes the data contained on the system recorder file.

ESTV (Error Statistics by Tape Volume): One of the two options of the DOS Volume Statistics. With ESTV support, the system collects data on tape errors by volume for any tape volumes used by the system.

EVA (Error Volume Analysis): One of the two options of the DOS Volume Statistics. With this option, the system issues a message to the operator when a number of temporary read or write errors (specified by the user at system generation time) has been exceeded on a currently accessed tape volume.

fetch:

1. To bring a program phase into main storage from a core image library for immediate execution.

2. The routine that retrieves requested phases and loads them into main storage.

3. The name of a macro instruction (FETCH) used to transfer control to the system loader.

4. To transfer control to the system loader.

F/L Trace (Fetch/Load Trace): A program that records information about phases and transients as they are called from a core image library.

GSVC Trace (Generalized Supervisor Calls Trace): A program that records SVC interrupts as they occur. All or a selected group of SVCs can be traced.

IDRA (Independent Directory Read-in Area): A resident area, created by a supervisor option, into which the system reads core image library directories for fetch and load operations. Using IDRA frees the physical transient area to perform error recovery procedures.

I/O (Input/Output) error logging: The process of recording OBR and SDR records on the system recorder file.

I/O Trace (Input/Output Trace): A program that records I/O device activity for all or a selected group of I/O devices.

job accounting interface: A function that accumulates accounting information for each job step to: charge usage of the system, help plan new applications, and help supervise system operation more efficiently.

MCAR (Machine Check Analysis and Recording): A feature that records System/370 machine check interrupt error information on the system recorder file and

then attempts to recover from the interrupt.

MCI (Machine Check Interrupt): The interrupt that occurs if the central processing unit fails to operate.

MCRR (Machine Check Recording and Recovery): The recording of pertinent data on the system recorder file after either a machine check interrupt or a channel inboard error occurred on System/360 Model 30, Model 40, or Model 50.

object module: One or more control sections in relocatable, nonexecutable form. An object module must be processed by the linkage editor before it can be executed in the system.

OBR (Outboard Recorder): A feature that records pertinent data on the system recorder file when an unrecoverable I/O error occurs.

overlay: A program segment (phase) that is loaded into main storage. It replaces all or part of a previously retrieved section.

PCIL (Private Core Image Library): A file referenced in the same manner and for the same purposes as the system core image library, but distinct from the system core image library. PCIL increases available core image library space to enable compiling, linkage editing, and executing in the foreground partition, when a private core image library is assigned to that foreground partition.

PDAID (Problem Determination Aids): Programs that trace a specified event when it occurs during the operation of a program. The traces provided are QTAM Trace, I/O Trace, F/L Trace, and GSVC Trace.

phase: The smallest complete unit that can be referenced in a core image library. Each program overlay is a complete phase. If the program has no overlays, the program itself is a complete phase.

private library: A relocatable, core image, or source statement library that is separate and distinct from the system library.

problem determination: A procedure or process (provided by IBM) that the user can follow after an error message to determine the cause of that error. (See PDAID)

QTAM Trace: A routine that records certain supervisor and I/O activities on tape or in main storage.

RMS (Recovery Management Support): A feature for System/370 that consists of the MCAR (Machine Check Analysis and Recording) and CCH (Channel Check Handler) functions. RMS gathers information about System/370 hardware reliability and attempts certain error recovery operations. RMS is a part of the entire reliability, availability, and serviceability support for System/370.

SDR (Statistical Data Recorder): A feature that records the cumulative error status of an I/O device on the system recorder file.

system recorder file: The file that is used to record hardware reliability data.

Indexes to systems reference library manuals are consolidated in the publication DOS Master Index, GC24-5063. For additional information about any subject listed below, refer to other publications listed for the same subject in the Master Index.

A (physical) transients 11
AB (abnormal termination) table 212
abnormal termination (AB) table 212
ALLOC statement processor
  detail chart 52
  general chart 28
  text 11
ASCII 238
ASSGN statement processor
  detail chart 60
  general chart 29
attention interrupts
  (also see statement processors)
  logical 11
  physical 11
attention routines
  logical transient 11
  physical transient 11

B (logical) transients 11
BATCH statement processor
  detail chart 56
  general chart 28
  text 11

cancel codes and messages 15
CANCEL statement processor
  detail chart 46
  general chart 27
  text 11
CBF (console buffering) table 222
CCH (channel check handler) 238
channel inboard error 238
channel queue (CHANQ) table 202,216
channel queue table operation 227
CHANQ (channel queue) table 202,216
CHANQ table operation 227
charts
  detail (see phases or statement processors)
  general (see phases or statement processors)
communications region
  additional 208,212
  extensions 208,212
console buffering (CBF) table 222
core image library 238
core wrap mode 238

data set security
  definition 238
  general chart 30
density data 226
detail charts (see phases or statement processors)
device type codes 225
DIB (disk information block) table 210
disk information block (DIB) table 210
Disk Operating System 238
  program flow 10
DLAB statement processor
  detail chart 82
  general chart 30
DLBL statement processor
  detail chart 79
  general chart 29
DOS Volume Statistics 238
DOS, see Disk Operating System
DTF addresses and pointers, (MICR) 208
dumps
  standard system 19
  translating 19

ERBLOC (error recovery block) 211
EREP 238
error message cross reference 200
error queue entry 211
error recovery block (ERBLOC) 211
ESTV 238
EVA (Error Volume Analysis) 238
EXEC statement processor
  detail chart 88
  general chart 30
EXTENT statement processor
  detail chart 95
  general chart 30

F/L Trace (Fetch/Load Trace) 238
FAVP (first available pointer) 202
fetch 238
fetch I/O key (FIK) 235
FICL (first-in-class-list) table 202,223
FIK (fetch I/O key) 235
first available pointer (FAVP) 202
first-in-class-list (FICL) table 202,223
first-on-channel-list (FOCL) 202
flowchart symbols, explanation of 236
flowcharts
  detail 40
  general 25
FLPTR (free list pointer) 202
FOCL (first-on-channel-list) 202
free list pointer (FLPTR) 202

general charts (see phases or statement processors)
glossary 238

GSVC Trace (Generalized SVC Trace) 238

HOLD statement processor
detail chart 102
general chart 30

I/O error logging 238
I/O table interrelationship 202
I/O Trace (Input/Output Trace) 238
IDRA
definition 238
PIB table support 215
IGNORE statement processor
detail chart 48
general chart 27
text 11
Independent Directory Read-in Area, see IDRA
initiator, program 13
interphase communication area (for $$ANERRZY,0) 11,16
interrupts
logical attention 11
physical attention 11
supervisor call (SVC) 21
interval timer (IT) table 212
IT (interval timer) table 212

JA
common table (ACCTCOMN) 228
definition 238
partition table (ACCTXX) 229
user's table 229
JIB (job information block) table 202, 218
job accounting interface, see JA
job information block (JIB) table 202, 218

label information format on SYSRES
DASD 233
tape 232
label list 186
LBLTYP statement processor
detail chart 95
general chart 30
LID (logical transient identification) 234
LISTIO statement processor
detail chart 69
general chart 29
sample output 17
LOG statement processor
detail chart 48
general chart 27
text 11
logical ($$B) transients
area 11
attention 11
initiator 13
logical ($$B) transients (CONT.)
storage location 12
terminator 13
logical transient identification (LID) 234
logical transient key (LTK) 234
logical unit block (LUB) table 202,223
LTK (logical transient key) 234
LUB (logical unit block) table 202,223
LUBDSP table 216
LUBID table 202,216

main storage map, supervisor 12
MAP output 11,16
MAP statement processor
detail chart 49
general chart 27
text 11
MCAR (Machine Check Analysis and Recording) 238
MCI (Machine Check Interrupt) 239
MCRR (Machine Check Recording and Recovery) 239
MICR
DTF addresses and pointers tables 208
microfiche index 237
MODE statement processor
detail chart 180
general chart 31
text 11
MSG statement processor
detail chart 44
general chart 27
text 11

next-in-class-list (NICL) table 202,223
NICL (next-in-class-list) table 202,223
NOLOG statement processor
detail chart 48
general chart 27
text 11

object module 239
OBR (Outboard Recorder) 239
OC (operator communications) table 212
operator communications (OC) table 212
option tables
AB (abnormal termination) 212
IT (interval timer) 212
OC (operator communications) 212
PC (program check) 212
overlay 239

partition identifier (PID) 234
PAUSE statement processor
detail chart 47
general chart 27
text 11
PC (program check) table 212

PCIL
definition 239
SVC usage 21
PDAID (Problem Determination Aids) 239
phase 239
phases
$$BATTNA, detail chart 40
$$BATTNA, general chart 26
$$BATTNA, text 11
$$BATTNB, detail chart 44
$$BATTNB, general chart 27
$$BATTNB, text 11
$$BATTNC, detail chart 46
$$BATTNC, general chart 27
$$BATTNC, text 11
$$BATTND, detail chart 49
$$BATTND, general chart 27
$$BATTND, text 11
$$BATTNE, detail chart 52
$$BATTNE, general chart 28
$$BATTNE, text 11
$$BATTNF, detail chart 54
$$BATTNF, general chart 28
$$BATTNF, text 11
$$BATTNG, detail chart 56
$$BATTNG, general chart 28
$$BATTNG, text 11
$$BATTNH, detail chart 58
$$BATTNH, general chart 28
$$BATTNH, text 11
$$BATTNI, detail chart 60
$$BATTNI, general chart 29
$$BATTNI, text 13
$$BATTNJ, detail chart 69
$$BATTNJ, general chart 29
$$BATTNJ, text 13
$$BATTNK, detail chart 74
$$BATTNK, general chart 29
$$BATTNK, text 13
$$BATTNL, detail chart 82
$$BATTNL, general chart 30
$$BATTNL, text 13
$$BATTNM, detail chart 88
$$BATTNM, general chart 30
$$BATTNM, text 13
$$BATTNN, detail chart 94
$$BATTNN, general chart 28
$$BATTNN, text 11
$$BATTNO, detail chart 95
$$BATTNO, general chart 30
$$BATTNO, text 13
$$BATTNP, detail chart 101
$$BATTNP, general chart 30
$$BATTNP, text 13
$$BATTNQ, detail chart 180
$$BATTNQ, general chart 31
$$BATTNQ, text 11
$$BATTNR, detail chart 182
$$BATTNR, general chart 31
$$BATTNR, text 11
$$BATTNS, detail chart 184
$$BATTNS, general chart 31
$$BATTNS, text 11
$$BCCHHR, detail chart 177
phases (CONT.)
$$BDUMP, detail chart, standard dump 152
$$BDUMP, detail chart, translating dump 134
$$BDUMP, general chart, standard dump 36
$$BDUMP, general chart, translating dump 34
$$BDUMP, text 13
$$BDUMPB, detail chart, standard dump 160
$$BDUMPB, detail chart, translating dump 137
$$BDUMPB, general chart, standard dump 38
$$BDUMPB, general chart, translating dump 37
$$BDUMPB, text 13
$$BDUMPD, detail chart, standard dump 165
$$BDUMPD, detail chart, translating dump 142
$$BDUMPD, general chart, standard dump 38
$$BDUMPD, general chart, translating dump 37
$$BDUMPD, text 13
$$BDUMPF, detail chart, standard dump 154
$$BDUMPF, general chart, standard dump 38
$$BDUMPF, text 13
$$BEOJ, detail chart 103
$$BEOJ, general chart 33
$$BEOJ, text 13
$$BEOJ1, detail chart 107
$$BEOJ1, general chart 34
$$BEOJ1, text 13
$$BEOJ2, detail chart 109
$$BEOJ2, general chart 35
$$BEOJ2, text 13
$$BEOJ2A, detail chart 111
$$BEOJ2A, general chart 35
$$BEOJ2A, text 13
$$BEOJ3, detail chart 113
$$BEOJ3, general chart 32
$$BEOJ3, text 13
$$BEOJ4, detail chart 114
$$BEOJ4, general chart 32
$$BEOJ4, text 13
$$BEOJ5, detail chart 115
$$BEOJ5, general chart 32
$$BEOJ5, text 13
$$BESTVA, detail chart 116
$$BESTVA, general chart 32
$$BESTVB, detail chart 117
$$BESTVB, general chart 32
$$BESTVC, detail chart 118
$$BESTVC, general chart 32
$$BESTVD, detail chart 119
$$BESTVD, general chart 32
$$BESTVE, detail chart 121
$$BESTVE, general chart 32

phases (CONT.)
$$BESTVF, detail chart 122
$$BESTVF, general chart 32
$$BILSVC, detail chart 127
$$BILSVC, general chart 35
$$BILSVC, text 13
$$BPCHK, detail chart 132
$$BPCHK, general chart 35
$$BPCHK, text 109
$$BPDUMP, detail chart, standard dump 170
$$BPDUMP, detail chart, translating dump 147
$$BPDUMP, general chart, standard dump 39
$$BPDUMP, general chart, translating dump 37
$$BPDUMP, text 13
$$BPDUM1, detail chart, standard dump 171
$$BPDUM1, general chart, standard dump 39
$$BPDUM1, text 13
$$BPSW, detail chart 130
$$BPSW, general chart 34
$$BPSW, text 13
$$BSDRUP, detail chart 176
$$BSYSWR, detail chart 175
$$BTERM, detail chart 123
$$BTERM, general chart 33
$$BTERM, text 13
physical ($$A) transients attention routines 11
physical unit block (PUB) table 202,217
PIB (program information block) 214
flag expansions 215
table 213,214
PID (partition identifier) 234
PIK (program interrupt key) 234
Private Core Image Library, see PCIL
private library 239
problem determination 239
program check (PC) table 212
program
information block (see PIB)
initiator 13
interrupt key (PIK) 234
key definitions 234
terminator 13
terminator phases 18
PUB (physical unit block) table 202,217

QTAM Trace 239

READ statement processor
detail chart 101
general chart 30
record format, label cylinder
DASD 233
tape 232
Recovery Management Support, see RMS
RELSE statement processor
detail chart 102
RELSE statement processor (CONT.)
general chart 30
REQID table 202,216
requestor I/O key (RIK) 235
requestor identification (RID) 216,234
RID (requestor identification) 216,234
RIK (requestor I/O key) 235
RMS
definition 239
monitor table (RASTAB) 230
use in MODE command 11

SAB (seek address block) 202
SDR (Statistical Data Recorder) 239
seek address block (SAB) 202
standard system dump 19
START statement processor
detail chart 56
general chart 28
text 11
statement processors
ALLOC, detail chart 52
ALLOC, general chart 28
ASSGN, detail chart 60
ASSGN, general chart 29
BATCH, detail chart 56
BATCH, general chart 28
CANCEL, detail chart 46
CANCEL, general chart 27
DLAB, detail chart 82
DLAB, general chart 30
DLBL, detail chart 79
DLBL, general chart 29
EXEC, detail chart 88
EXEC, general chart 30
EXTENT, detail chart 95
EXTENT, general chart 30
HOLD, detail chart 102
HOLD, general chart 30
IGNORE, detail chart 48
IGNORE, general chart 27
LBLTYP, detail chart 100
LBLTYP, general chart 30
LISTIO, detail chart 69
LISTIO, general chart 29
LOG, detail chart 48
LOG, general chart 27
MAP, detail chart 49
MAP, general chart 27
MODE, detail chart 180
MODE, general chart 31
MSG, detail chart 44
MSG, general chart 27
NOLOG, detail chart 48
NOLOG, general chart 27
PAUSE, detail chart 48
PAUSE, general chart 27
READ, detail chart 101
READ, general chart 30
RELSE, detail chart 102
RELSE, general chart 30
START, detail chart 56
START, general chart 28

statement processors (CONT.)
TIMER, detail chart 94
TIMER, general chart 28
TLBL, detail chart 76
TLBL, general chart 29
TPLAB, detail chart 75
TPLAB, general chart 29
UCS, detail chart 92
UCS, general chart 30
UNA, detail chart 67
UNA, general chart 29
VOL, detail chart 74
VOL, general chart 29
XTENT, detail chart 84
XTENT, general chart 30
supervisor call interrupt (SVC), see SVC
supervisor
calls 21,23
cancel codes 15
communications regions 203
storage map 12
SVC (supervisor call interrupt)
list 23
text 21
SYSRES label information format
DASD 233
tape 232
system dump, standard 19
system recorder file 239

tables
AB 212
CBF 222
CHANQ 202,216,227
DIB 210
FICL 202,223
I/O 202
IT 212
JA Common 228
JA Partition 229
JA User's 229
JIB 202,218
LUB 202,223
LUBID 202,216
MICR DTF addresses 208
NICL 202,223
OC 212
option 212
PC 212
PIB 214
PUB 202,217
REQID 202,216
RMS Monitor 230
TEB 202,219
TEBV 202,220
THFLPTR 202
TKHDTAB 202,224
TKREQID 202,216
tape error block by volume, see TEBV
tape error block, see TEB
task selection procedure 227
TEB
address pointer 203
TEB (CONT.)
I/O tables, storage allocation 12, 202
tape error block 202,219
TEB table format 219
TEBV
address pointer 208
I/O tables, storage allocation 12, 202
tape error block by volume 202,220
TEBV table format 220
terminator phases 18
terminator, program 13
THFLPTR table 202
TIMER statement processor
detail chart 94
general chart 28
text 11
TKHDTAB (track hold) table 67,202
TKREQID table 202,216
TLBL statement processor
detail chart 76
general chart 29
TPLAB statement processor
detail chart 75
general chart 29
track hold (TKHDTAB) table 67,202
transient attention routines
logical 11
physical 11
translating dump 19

UCS statement processor
detail chart 92
general chart 30
UNA statement processor
detail chart 67
general chart 29

version and modification level (see intro. PLM index)
VOL statement processor
detail chart 74
general chart 29

XTENT statement processor
detail chart 84
general chart 30

Reader's Comments Form

DOS Logical Transients GY24-5152-1

Your comments, accompanied by answers to the following questions, help us produce better publications for your use. If your answer to a question is "No" or requires qualification, please explain in the space provided below. Please give specific page and line references with your comments when appropriate. All comments will be handled on a non-confidential basis. Copies of this and other IBM publications can be obtained through IBM branch offices.

	Yes	*No*
• Does the publication meet your needs?	☐	☐
• Did you find the material:		
Easy to read and understand?	☐	☐
Organized for convenient use?	☐	☐
Complete?	☐	☐
Well illustrated?	☐	☐
Written for your technical level?	☐	☐

• What is your occupation? _____

• How do you use this publication:

As an introduction to the subject?	☐	As an instructor in a class?	☐
For advanced knowledge of the subject?	☐	As a student in a class?	☐
For information about operating procedures?	☐	As a reference manual?	☐

Your comments:

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. *If you would like a reply, please supply your name and address on the reverse side of this form.*

Staple

GY24-5152-1

Your comments, please . . .

This publication is one of a series that serves as a reference source for systems analysts, programmers, and operators of IBM systems. Your answers to the questions on the back of this form, together with your comments, help us produce better publications for your use. Each reply is carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

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

Cut Along Line

Fold Fold

FIRST CLASS
PERMIT NO. 170
ENDICOTT, N. Y.

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . .

IBM Corporation
P. O. Box 6
Endicott, N. Y. 13760

Attention: Programming Publications, Dept. G60

Fold Fold

If you would like a reply, *please print:*

Your Name _____

Company Name _____

Department _____

Street Address _____

City _____

State _____ *Zip Code* _____

IBM

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

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

DOS Logical Transients (S360-36) Printed in U.S.A. GY24-5152-1

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

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