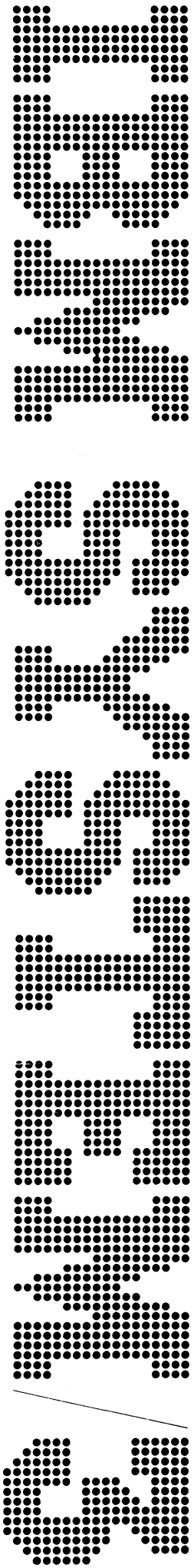


IBM System/3
Basic Assembler
Program Logic Manual

Program Numbers:

5702-AS1 (Models 8 and 10)
5704-AS1 (Model 15)
5704-AS2 (Model 15)
5705-AS1 (Model 12)

LY21-0504-3
File No. S3-21

Fourth Edition (March, 1974)

This is a major revision of, and obsoletes, LY21-0504-2. Changes are made to reflect the addition of System/3 Model 15. This manual has been extensively revised; therefore, the entire edition should be reviewed.

This edition applies to version 10, modification 00 of IBM System/3 Model 10 Disk System, Program Product Number 5702-AS1, to version 01, modification 00 of IBM System/3 Model 15, Program Product Number 5704-AS1, and to all subsequent versions and modifications until otherwise indicated in new editions or Technical Newsletters. Changes are continually made to the specifications herein; before using this publication in connections with the operation of IBM Systems, consult the latest IBM System/3 Newsletter, GN20-2228 for the editions that are applicable and current.

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

A Reader's Comment Form is at the back of the publication. If the form is gone, address your comments to IBM Corporation, Department 245, Rochester, Minnesota 55901.

Preface

This program logic manual describes the internal logic of the IBM System/3 Disk Assembler. The main function of the assembler is to translate source program statements into machine language for execution by the computer. There are five sections within the PLM, organized as follows:

1. *Introduction* contains general information about the operational, environmental, and physical characteristics of the assembler.
2. *Method of Operation* describes the functional flow of logic and data in the assembler.
3. *Program Organization* describes the function and storage allocation of each phase in the assembler.
4. *Directory* contains information needed for quick reference to the program listings.
5. *Data Area Formats* describes the formats of all data areas used by two or more phases.

System/3 Model 8

The System/3 Model 8 is supported by System/3 Model 10 Disk System Control Programming and Program Products. The facilities described in this publication for the Model 10 are also applicable to the Model 8, although the Model 8 is not referenced. It should be noted that not all devices and features which are available on the Model 10 are available on the Model 8. Therefore, Model 8 users should be familiar with the contents of *IBM System/3 Model 8 Introduction*, GC21-5114.

Related Publications

- *IBM System/3 Model 12 System Data Areas and Diagnostic Aids Handbook,* SY21-0045
- *IBM System/3 Model 15 System Data Areas and Diagnostic Aids Handbook,* SY21-0032

Prerequisite Publications

Effective use of this program logic manual requires an understanding of the information presented in the *IBM System/3 Basic Assembler Reference Manual,* SC21-7509, as well as topics discussed in the program logic manuals shown below, which are referenced in this manual. In the flowcharts, references are made indirectly, by using function/module names in library blocks (see *Appendix B. Flowcharting Techniques,* for an example). The following chart shows which PLM to refer to for a discussion of each function/module name used in the flowcharts.

Model 10 Disk System	Function/Module Name											
	IOS	DM	Open	Close	Allocate	SYSLOG	SYSPCH	SYSPRT	SYSLST	EOJ	FETCH	Supv
IBM System/3 Disk Systems Data Management and Input/Output Supervisor Logic Manual, SY21-0512.	X	X	X	X								
IBM System/3 Disk Systems System Control Program Logic Manual, SY21-0502.		X			X	X	X	X		X	X	X
Model 12												
IBM System/3 Model 12 System Control Program Logic Manual, SY21-0046.	X	X	X	X	X	X	X		X	X	X	X
Model 15												
IBM System/3 Model 15 Supervisor and IOS Logic Manual, SY21-0033.	X										X	X
IBM System/3 Model 15 Data Management Logic Manual, SY21-0034.		X	X	X								
IBM System/3 Model 15 Scheduler Logic Manual, SY21-0035.					X	X	X	X		X		

Section 1. Introduction

The IBM System/3 Model 10 Disk System Basic Assembler Language, IBM System/3 Model 12 Basic Assembler Language, and IBM System/3 Model 15 Basic Assembler Language provide a convenient way to write programs for IBM System/3. The IBM System/3 Basic Assembler processes instructions written in assembler language, translates them into machine instructions, assigns storage locations, and performs other functions necessary to produce an executable, machine-language program. Input to the assembler programs consists of source records on disk storage. The source records are processed by the following phases:

- Phase 05 Assembler Initialization Phase. Initializes the assembler; processes OPTIONS, HEADERS, and/or ICTL statements.
- Phase 10 Compression Phase. Reads source file; generates intermediate text on the work file.
- Phase 20 Symbol Processing Phase. Reads the intermediate text and builds symbol table in core storage.
- Phase 21 Symbol Table Overflow Processing Phase. Called only if symbol table overflows. Tests intermediate text following the overflow for previously defined symbols.
- Phase 22 Symbol Substitution Phase. Places values from last (or only) symbol table into the intermediate text term records. Builds an ESL (External Symbol List) table.
- Phase 29 ESL Output Phase. Puts ESL records in the object file. Prints the ESL.
- Phase 30 Source/Object Output Phase. Generates object code and source object listing, and puts the object code out to an object file.
- Phase 35 Diagnostic Phase. Writes a diagnostic message list for assembly errors.
- Phase 40 Cross-Reference Build Phase. Builds a cross-reference file (if XREF is specified on the OPTIONS card).
- Phase 41 Cross-Reference Sort Phase. Sorts the file built in Phase 40, generates the cross reference listing, and fetches the Overlay Linkage Editor if object output is required (DECK and/or OBJ specified on OPTIONS statement).

A detailed account of each phase is contained in *Section 3, Program Organization.*

STORAGE REQUIREMENTS

The Model 10 Disk System Basic Assembler (5702-AS1) requires 8,192 bytes of main storage for execution, exclusive of control program requirements.

The Model 12 Basic Assembler (5705-AS1) and the Model 15 Basic Assembler (5704-AS1 or 5704-AS2) require 10,240 bytes of main storage for execution, exclusive of control program requirements.

Section 2. Method of Operation

This section describes the functional flow of logic and data through the various phases of the assembler.

Phase-to-Phase Communication

Phase-to-phase communication during execution of the assembler is accomplished by using two communication modules:

1. $ASROT – Assembler Communications Module. This is the basic communication module for the entire assembler. It resides in the low address area of the partition throughout the assembly process. (See *Data Area Formats* for a detailed description of $ASROT.) The $ASROT module contains the following sections:
   a. Communications Vector (COMVEC). Contains a transfer vector for all data management routines, address tables for any working storage that must be accessed outside of an executing module, and the name of the currently executing module. Disk DTFs and IOBs for source file, work file, and object files also reside in COMVEC.
   b. Assembler Common (ASMCOM). A working storage/constant block used by all assembler phases for intraphase and interphase communication. See *Data Area Formats* for a detailed description of ASMCOM.

   Note: The Compiler Access Method (CAM) resides in $ASROT. See Section 3. *Program Organization* for a description of CAM.

2. $ASPRC – Printer Communications Module. This module contains the data areas necessary to print the assembler listing, in conjunction with a print module. These data areas are:
   a. Header Area. Data area used for passing the header of the listing from phase to phase. Accessed through COMVEC.
   b. Printer Buffer. Print buffer with proper alignment (X'7C' byte boundary).
   c. Print Module Interface. In the Model 10 Assembler, this interface is the Printer DTF; in the Model 12 Assembler, this interface is the SYSLST parameter list; in the Model 15 Assembler, this interface is the SYSPRT parameter list. Accessed through COMVEC.
   d. Print Module Load Area (Model 15 Assembler only). The SYSPRT module is loaded into this area, which is accessed through COMVEC. See *Data Area Formats* for a detailed description of $ASPRC.

Register Conventions

1. The assembler uses Index Register 1, hereafter referred to as XR1, as the communications register. At all times XR1 will be pointing either to the Communications Vector (COMVEC) or to Assembler Common (ASMCOM). Each of these two areas contains the displacement of the other, so that XR1 can be switched with a single instruction. At the entry to each phase (after Phase 05), XR1 is pointing to COMVEC. The first instruction of each phase moves the phase's module name to COMVEC, so that COMVEC always contains the name of the currently executing module.

2. Index Register 2, hereafter referred to as XR2, is used as a DTF pointer and a work register throughout the assembly. The contents of XR2 at the entry to a phase are unpredictable.

3. Communications Vector initialization. $ASROT is loaded only once with Phase 05. Therefore, any communication tables or routines which are loaded with a later phase must have their addresses moved to COMVEC. This is normally accomplished during the initialization routine for the phase, using an EXTRN statement, an address constant, and a MVC.

Work File ($WORK2)

The work file is a scratch file used by the assembler for intermediate disk storage.

Data is passed back and forth between the work file and the phases throughout the assembly process. See *Appendix A* and *Data Area Formats* for a detailed description of the work file.

Source File ($SOURCE)

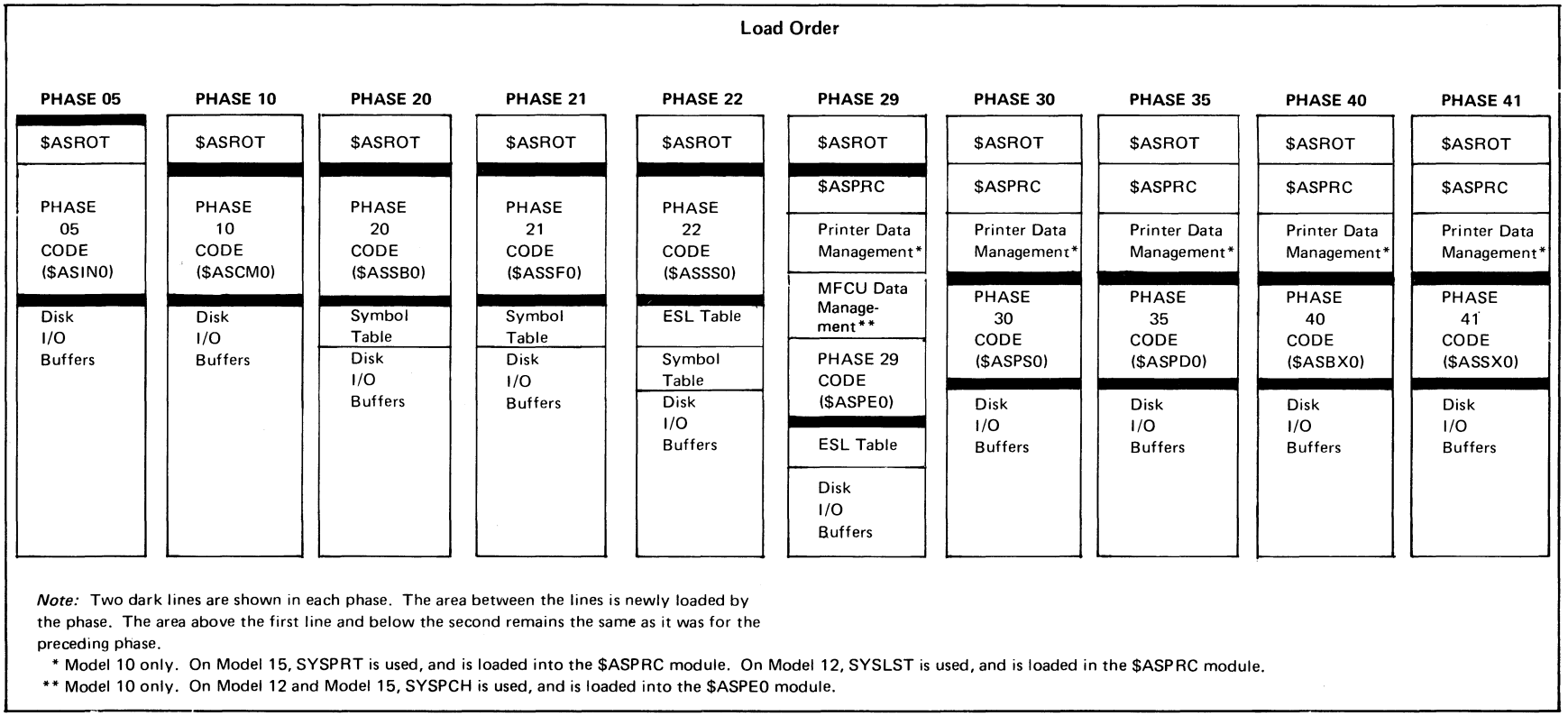
The source file is used by the assembler for storing the source program. It provides source records for phases 05, 10, 29, and 30. The source file is loaded by the Scheduler or the macro processor before the assembler begins executing.

Object File ($WORK)

The object program produced by the assembler is put out on the object file during phases 29 and 30. After assembly, Phase 41 passes control to the overlay linkage editor if object output is required. The object file then becomes an input file for the linkage editor.

Figure 1 shows the main storage load structure for each phase in the assembler program. It also indicates which areas remain the same from one phase to the next, and which areas are overlaid with new coding.

Figure 2 illustrates the overall flow of logic and data through the assembler program. The logic flow is traced from phase to phase, with the major functions of each phase listed under the phase's numerical title. The data flow is followed between the phases and ASMCOM, the Work File, the Source, and the Object File. Any output produced by a phase is also indicated as data flow.



Note: Two dark lines are shown in each phase. The area between the lines is newly loaded by the phase. The area above the first line and below the second remains the same as it was for the preceding phase.

* Model 10 only. On Model 15, SYSPRT is used, and is loaded into the $ASPRC module. On Model 12, SYSLST is used, and is loaded in the $ASPRC module.

** Model 10 only. On Model 12 and Model 15, SYSPCH is used, and is loaded into the $ASPE0 module.

Figure 1. Main Storage Load Structure for the Assembler

Page of LY21-0504-3
Issued 17 March 1976
By TNL: LN21-5382

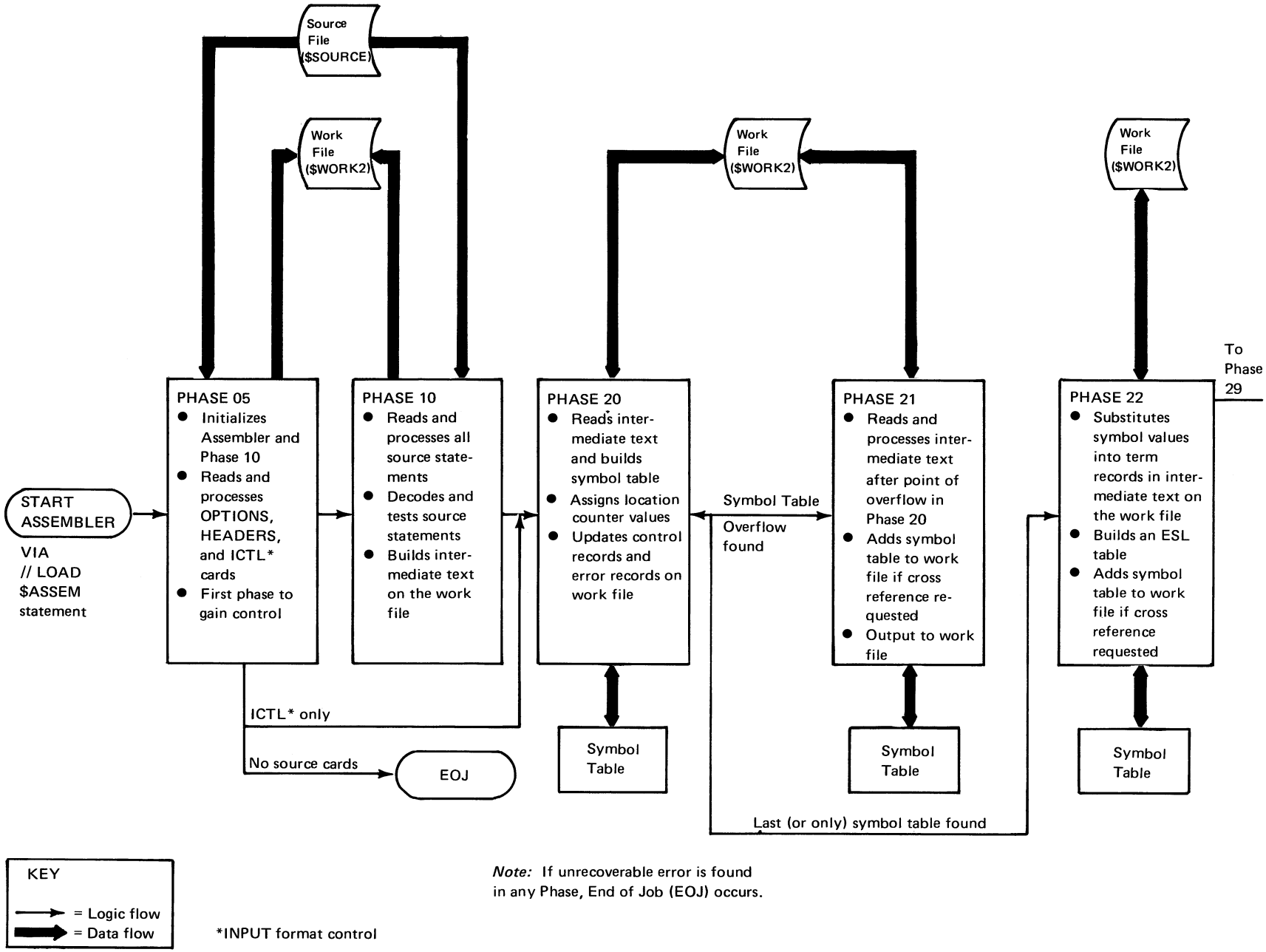


Figure 2 (Part 1 of 2). Data and Logic Flow for the Assembler

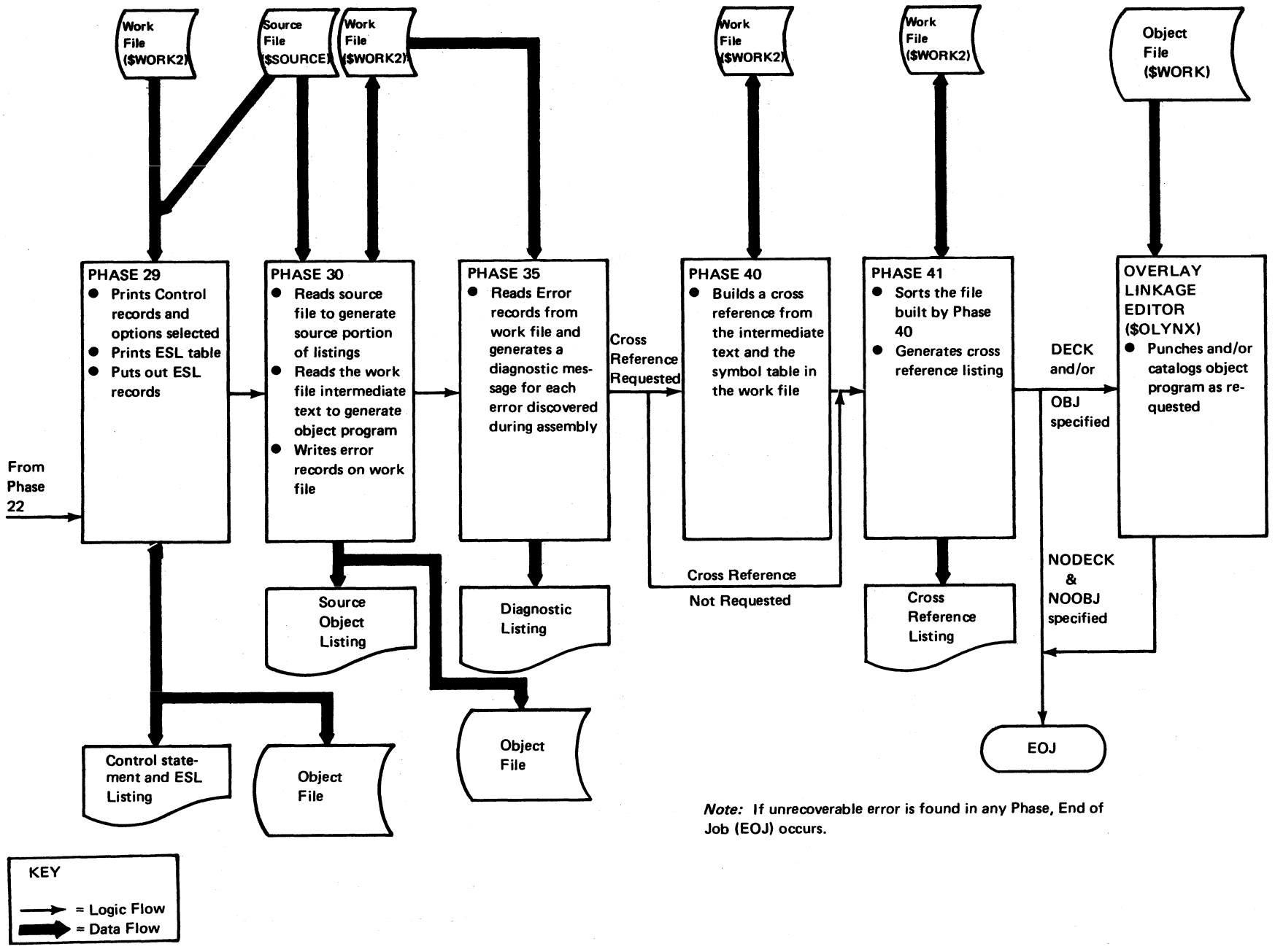


Figure 2 (Part 2 of 2). Data and Logic Flow for the Assembler

Note: If unrecoverable error is found in any Phase, End of Job (EOJ) occurs.

This section describes in detail each of the phases and routines that perform the functions of the assembler. The description of each phase includes a main storage map and appropriate flowcharts. (Some of the simpler routines within each phase do not have flowcharts.) For a description of the ASMCOM data area used by the individual phases, see *Section 5, Data Area Formats.*

MODULE $ASIN0 – ASSEMBLER INITIALIZATION (PHASE 05)

Main Storage Map: Figure 3.

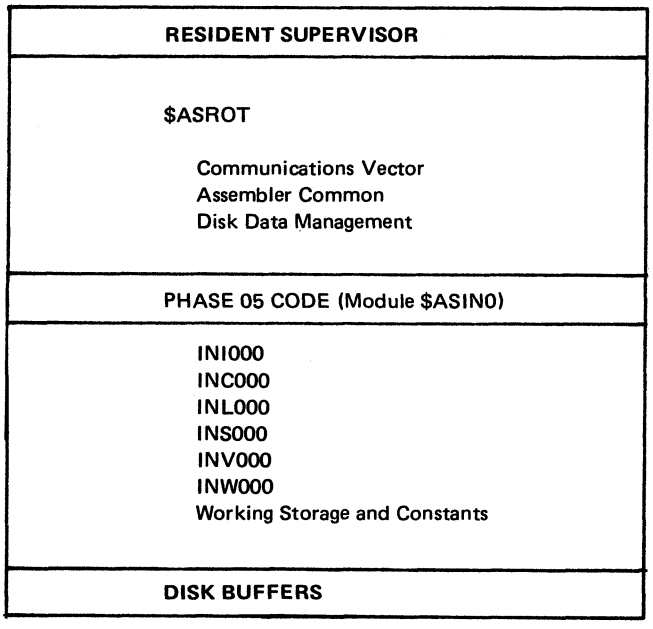


Figure 3. Phase 05 Main Storage Map

Entry Point: $ASIN1 – entered from the scheduler. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Scheduler Logic Manual,* SY21-0035.)

Charts: AA-AD.

Functions:

- Initializes the assembler (INI000–Chart AA).
- Processes three types of records which, if used, must precede all other source statements.
  1. OPTIONS–assembler control options and/or HEADERS–assembler control information.
  2. ICTL–source record boundaries (INC000–Chart AB).
- Fetches Phase 10 (INC000–Chart AB).
- Searches an operand field until a comma or a blank is found (INL000–no chart).
- Searches a source record and places the following information into ASMCOM:
  1. Length of name (if present).
  2. Column number of rightmost byte of operation code.
  3. Length of operation code.
  4. Column number of leftmost byte of operand.
  5. Length of operand field.
  6. Column number of rightmost byte of operand (INS000–Chart AC).
- Converts zoned decimal strings to binary equivalents (INV000–no chart).
- Provides interfaces with disk data management (INW000, INW010, and INW100–Chart AD).

Input:

- Source File

  1. OPTIONS, HEADERS, and ICTL (Input Format Control) cards.

  2. Assembler language cards.

Output:

- OPTIONS switches set as requested on OPTIONS records. (If none used, default options are DECK, LIST, XREF, REL, OBJ, NOXBUF.) Valid options and control record count stored in ASMCOM.

- HEADERS information stored in ASMCOM.

- Beginning and end column values set as specified on ICTL (Input Format Control) record. (If none used, default values of 1 and 87 are assumed. Sequence ID field is assumed to be in columns 89-96.)

- One or more records written into work file.

  1. Intermediate text control record.

  2. Error records.

  3. Last intermediate text control record if end-of-source or invalid ICTL is found.

Exits:

- Normal

  Control passed to Phase 10 (Module $ASCMO). Exception: If end-of-source is found, control passed to Phase 20 (Module $ASSBO). If there are no source cards, control is passed to EOJ.

- Error

  Control is passed to scheduler via Halt/Syslog or EOJ transients. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Scheduler Logic Manual,* SY21-0035.)

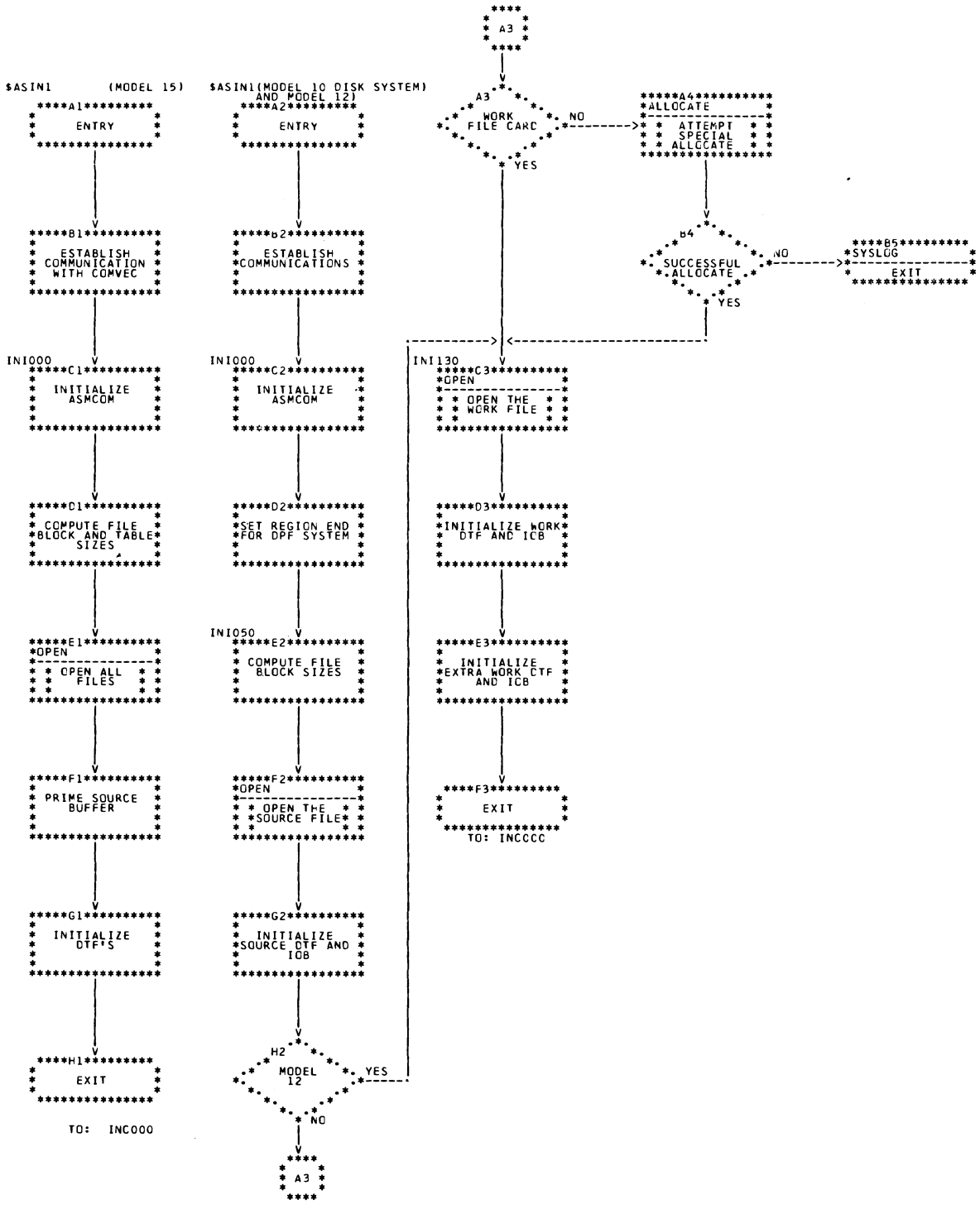


Chart AA. Initialization Routine (INI000) — Model 10/Model 12

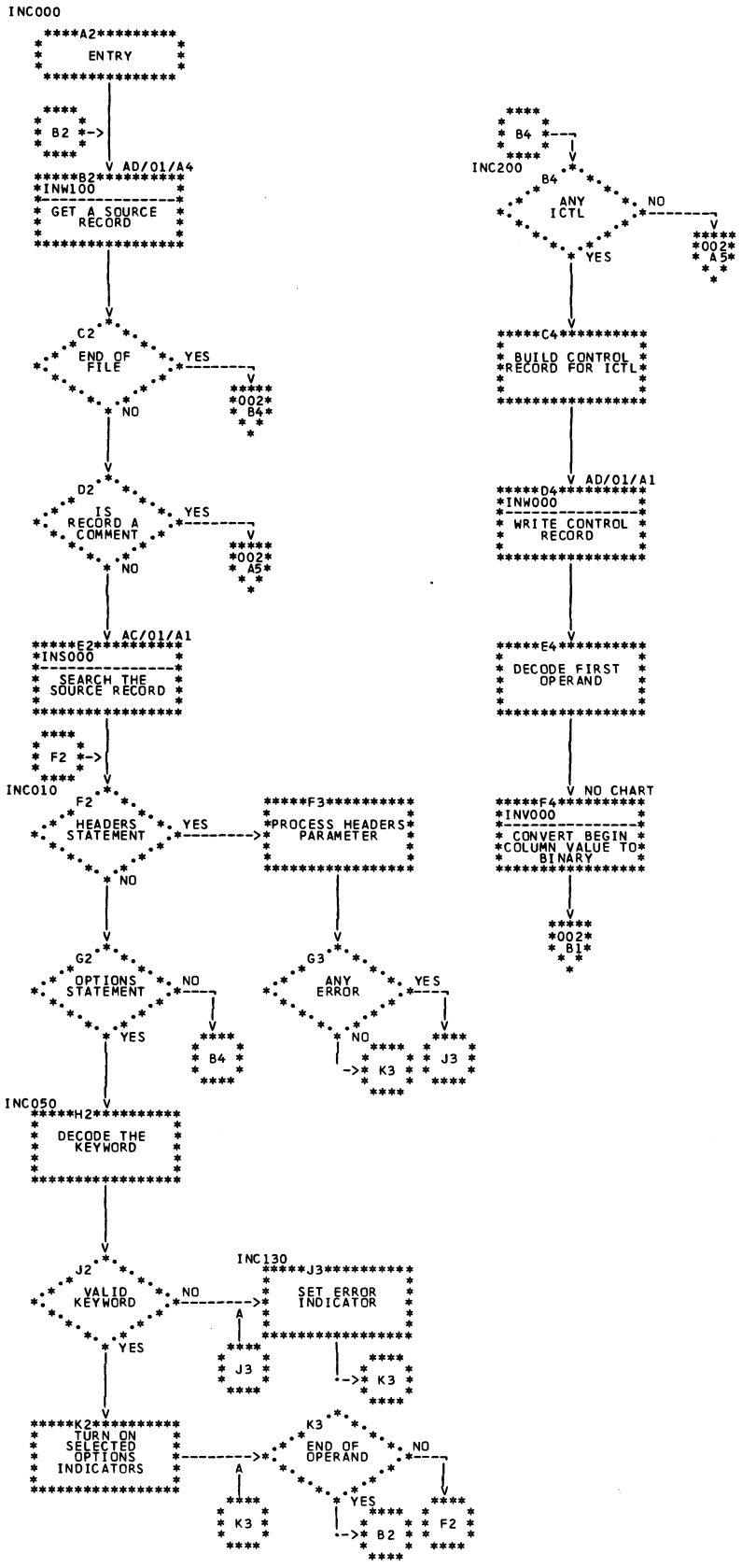


Chart AB. (Part 1 of 2). Options and/or ICTL Instruction Processing Routine (INC000)

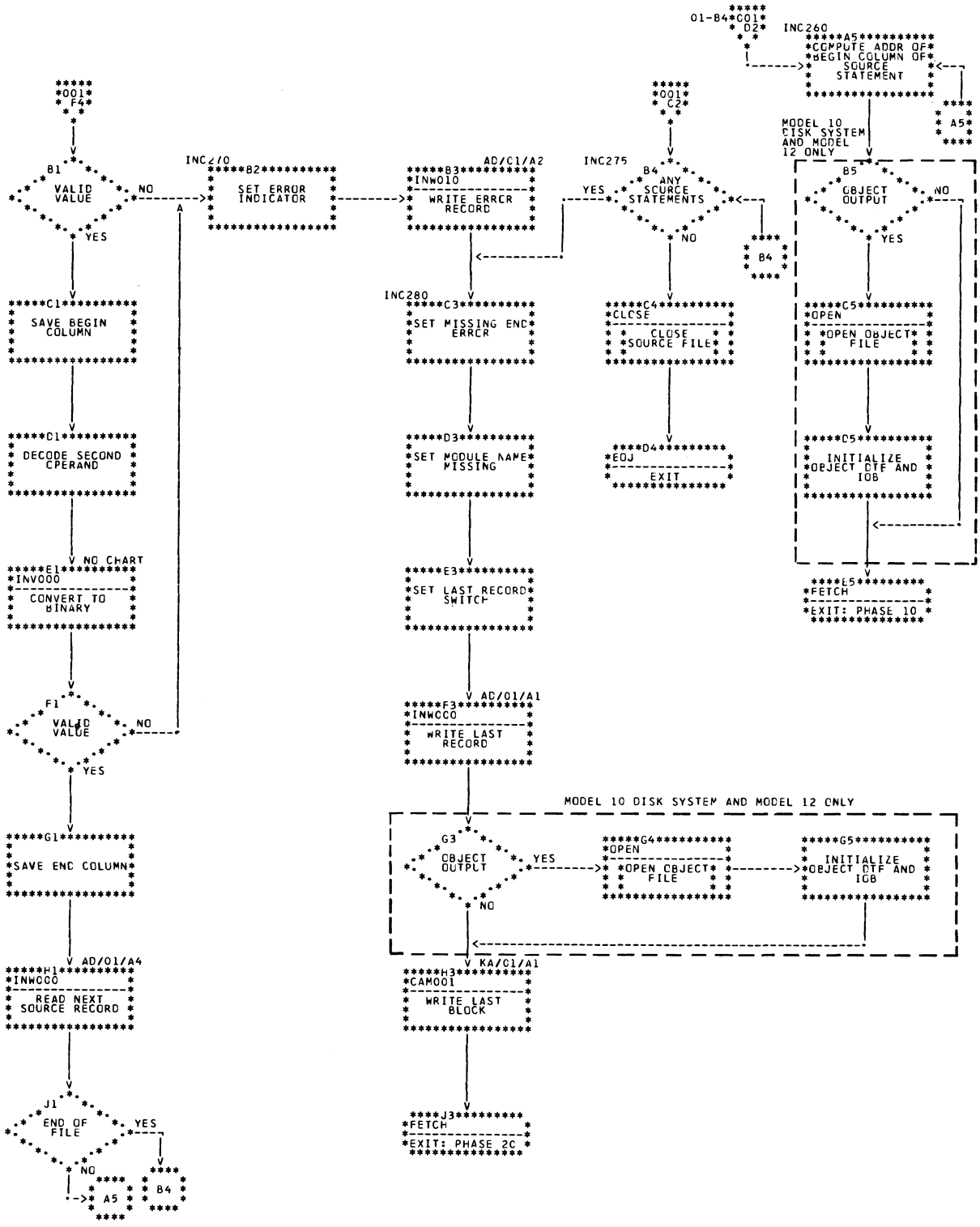


Chart AB. (Part 2 of 2). Options and/or ICTL Instruction Processing Routine (INC000)

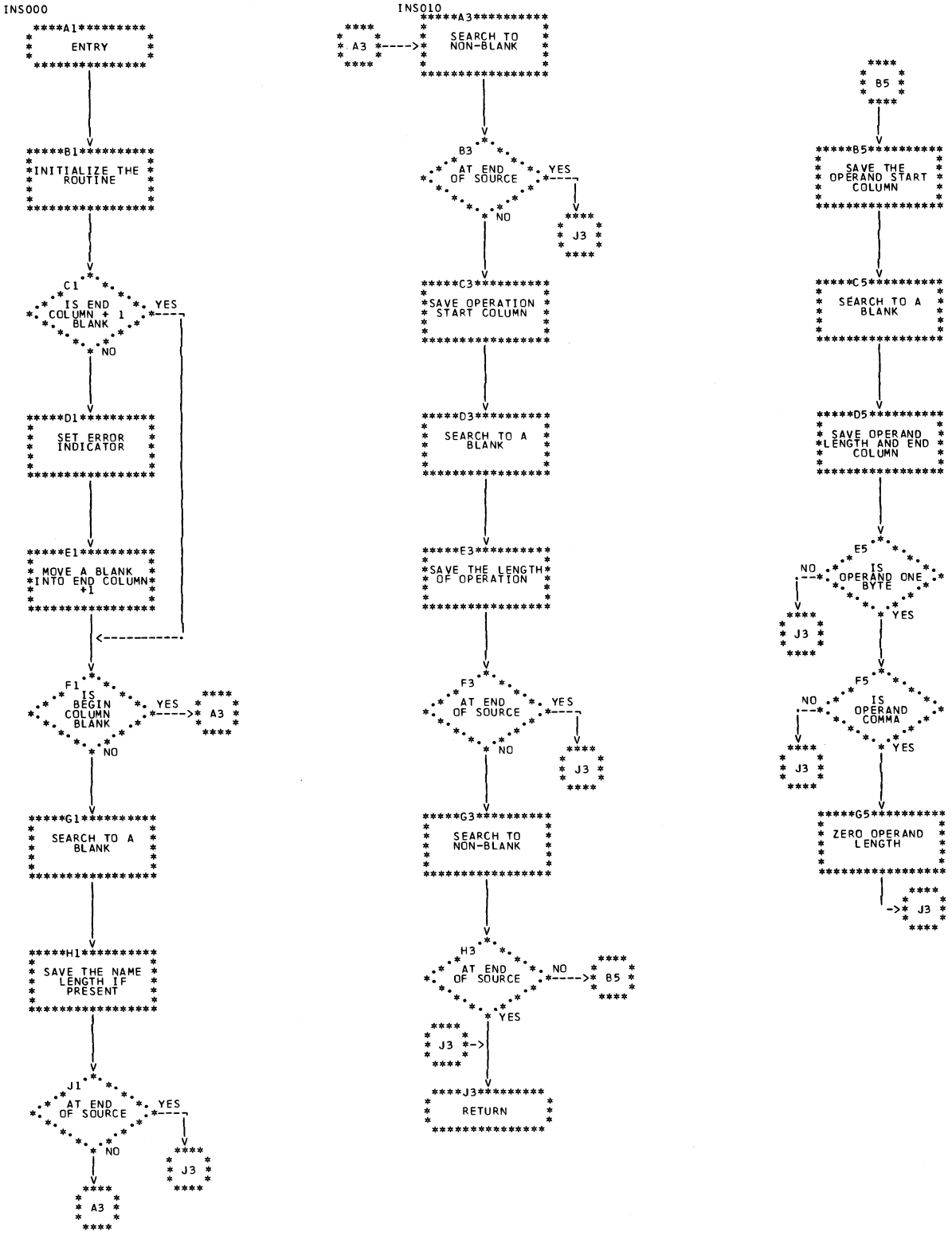


Chart AC. Source Record Search Routine (INS000)

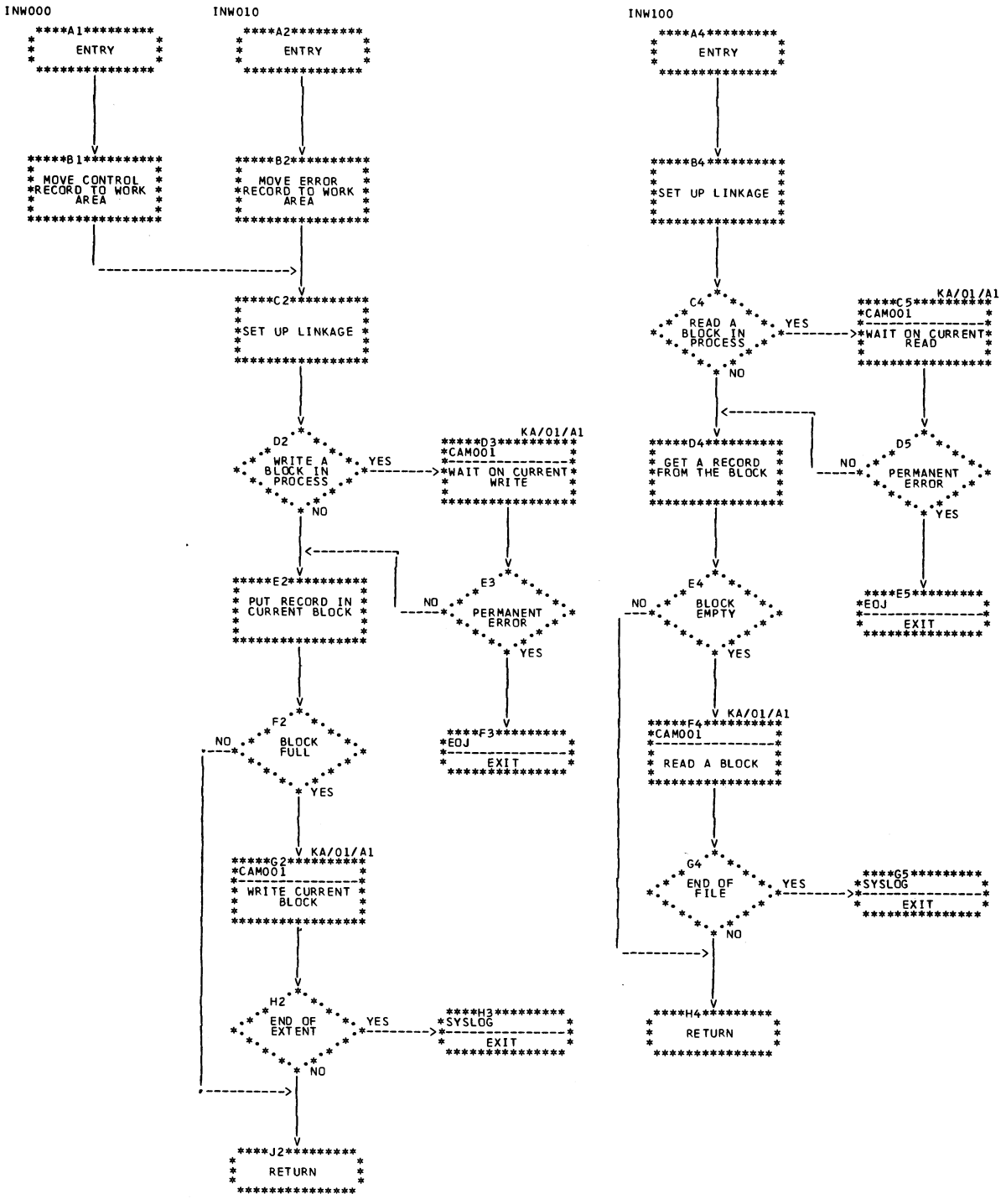


Chart AD. Disk Data Management Interfaces (INW000)

MODULE $ASCM0—COMPRESSION (PHASE10)

Main Storage Map: Figure 4.

Entry Point: $ASCM1—entered from Phase 05 via FETCH. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Supervisor and IOS Logic Manual,* SY21-0033.)

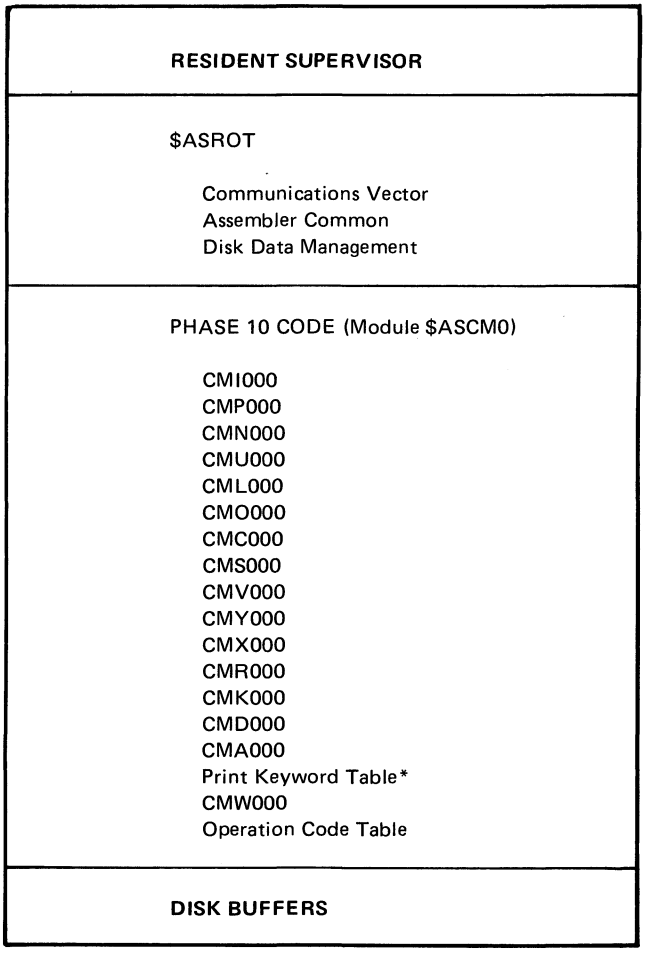
Charts: BA-BH.

Functions:

- Initializes Phase 10 for processing (CMI000—Chart BA).
- Controls Phase 10 processing of source records, including sequence checking if requested (CMP000—Chart BB).
- Tests for name field (CMN000—no chart).
- Puts out control record and a name record if a valid symbol is present (CMU000—no chart).
- Performs actions required when Phase 10 assembly processing is complete (CML000—Chart BG).
- Fetches Phase 20 (CML000—Chart BG).
- Determines the operation code specified on the current source statement (CMO000—no chart).
- Determines if a referenced byte contains either a valid alphabetic or numeric character (CMC000—no chart).
- Searches a source record; records following information in ASMCOM:

  1. Length of name (if present).
  2. Column number of rightmost byte of operation code.
  3. Length of operation byte.
  4. Column number of leftmost byte of operand.
  5. Length of operand field.
  6. Column number of rightmost byte of operand (CMS000—no chart; functionally the same as INS000 in Phase 05).

- Converts zoned decimal strings to binary equivalents (CMV000—no chart).
- Decodes and tests the syntax of the operand field of the current source statement for proper syntactical coding (CMY000—Chart BC; see Figure 5. Syntax Testing).
- Determines if the operand format set by the Syntax Routine CMY000 is valid for the current type of source statement (CMX000—no chart).
- If operand format is valid, sets up operand byte in the intermediate text control record build area (CMX000—no chart).
- Determines object length—in bytes—of current machine source statement (CMR000—no chart).
- Prepares each term in the operand field for the CMD000 routine by searching for the end of the current term and temporarily placing a blank in that byte (CMK000—Chart BE).
- Creates intermediate text term record for each valid term in the operand field of the current source record (CMD000—Chart BF).
- Processes all assembler operations (CMA000—Chart BD).
- Creates intermediate text term records for the operand field of the TITLE and DC instructions (CMA000—Chart BD).
- Provides interfaces with data management for disk operations (CMW000, CMW010, CMW020, CMW030, CMW100—Chart BH).



* Model 12 and Model 15 only

Figure 4. Phase 10 Main Storage Map

Branch Table (2)	Acceptable Syntactical Possibilities for Currently Loaded Branch Tables (1) / Possible Expression (3)	Any Non-Syntactical Element (4)	End of Field	Left Parentheses	Comma	Right Parentheses
1	START	Record element start. Load branch table 2	Error	Error	Error	Error
2	X	Continue	Compute and store element length return	Compute and store element length load branch table 3	Compute and store element length load branch table 1	Error
3	X(	Record element start. Load branch table 7	Error	Error	Load branch table 4	Error
4	X(, or X(X,	Record element start. Load branch table 5	Error	Error	Error	Error
5	X(X,X or X(,X	Continue	Error	Error	Error	Compute and store element length. Load branch table 6
6	X(X,X) or X(,X) or X(X)	Error unless DC/DS	End of operand, end of field	Error unless DC/DS	End of operand, not end of field	Error
7	X(X	Continue	Error	Error	Compute and store element length. Load branch table 4	Compute and store element length. Load branch table 6

Notes:

(1) Acceptable syntactical possibilities are listed across the top of the table.

(2) Reading across from each Branch Table number, the action taken is indicated in the case of each syntactical possibility. (Example: In Branch Table 1, only a non-syntactical element is acceptable. All other possibilities will cause an error bit to be turned on in the error record build areas.

(3) For each Branch Table number, the Possible Expression column shows the format that the operand being tested will be in up to that point.

(4) A non-syntactical element in an operand is a character or series of characters other than the parentheses and commas which form the syntactical elements. (Example: X(X,X) ... The Xs are non-syntactical elements.)

Figure 5. Syntax Testing

Input:

- Source file ($SOURCE). Source records are read from the source file by disk data management.
- Assembler Common (ASMCOM). This area in the module $ASROT has been completely initialized in phase 05.
- Operation Code Table. Contains information about all instructions supported by the assembler. This table is divided into five sections, each of which can be accessed by means of pointers containing the address of the right-most byte of the first mnemonic in that section. Each pointer also contains a one-byte count of the number of mnemonics in the section.)

  1. Section 1, all 1-character mnemonics.
  2. Section 2, all 2-character mnemonics.
  3. Section 3, all 3-character mnemonics.
  4. Section 4, all 4-character mnemonics.
  5. Section 5, all 5-character mnemonics.

The operation code table is loaded with Phase 10. It is included as input here because it is used by the phase as a model to which source records are compared.

Output:

- Intermediate text in the work file ($WORK2). Intermediate text records written into the work file by disk data management.
- Parameters stored in ASMCOM:

  1. TITLEN – name from the first title statement.
  2. MODULE – module name from the start statement.
  3. SEQCNT – count of sequence errors.
  4. ESLCNT – count of ESL table entries.

Exits:

- Normal – control is passed to Phase 20 (module $ASSBO).
- Error – control is passed to the scheduler via Halt/Syslog or EOJ transients. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Programming Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Scheduler Logic Manual,* SY21-0035.)

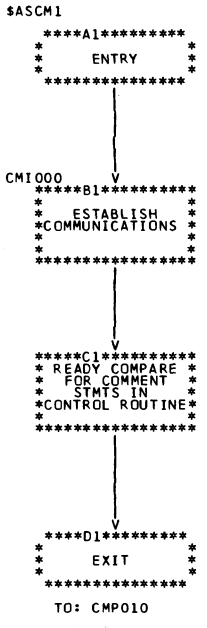


Chart BA. Initialization Routine (CMI000)

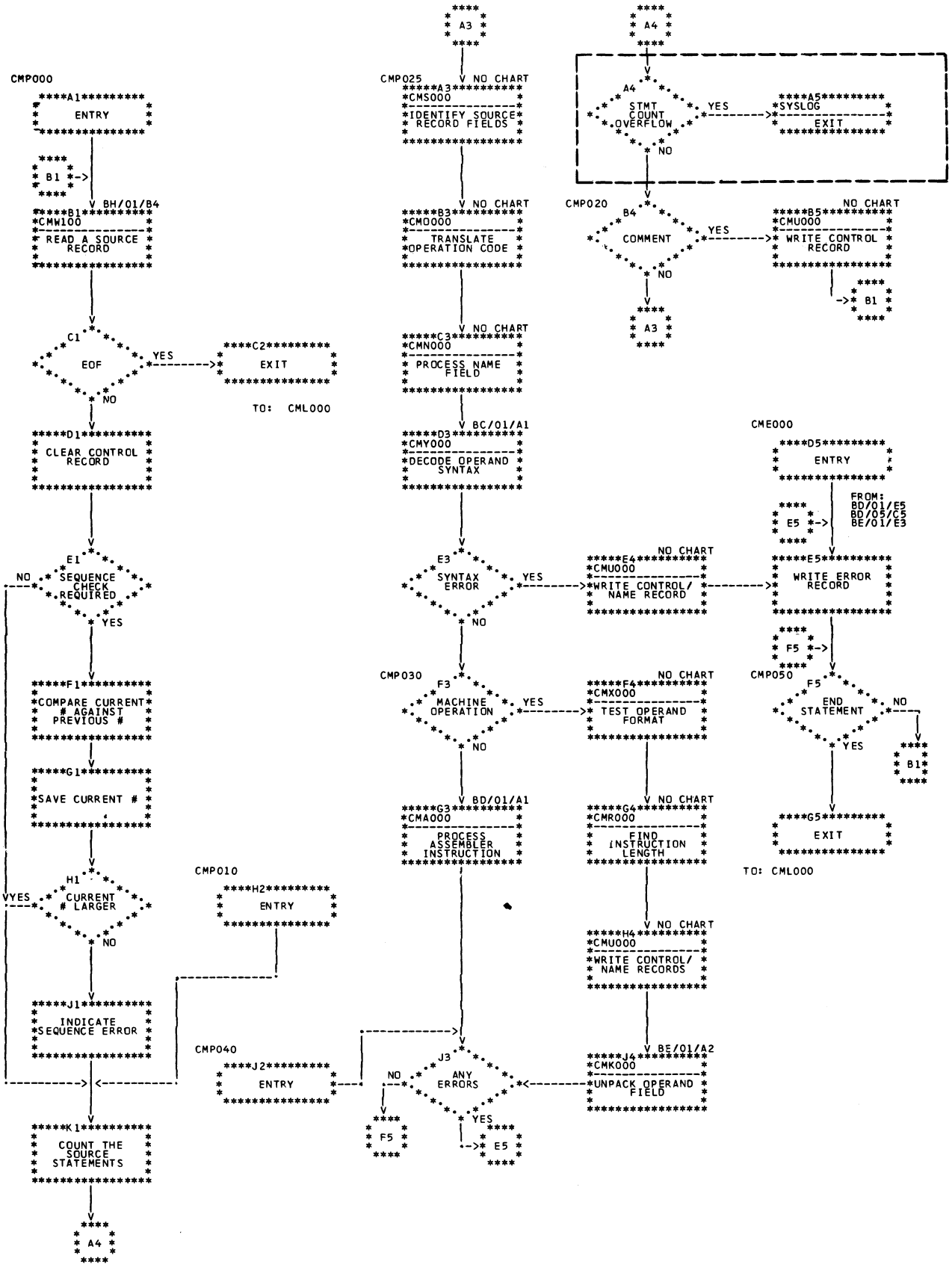


Chart BB. Main Control Routine (CMP000)

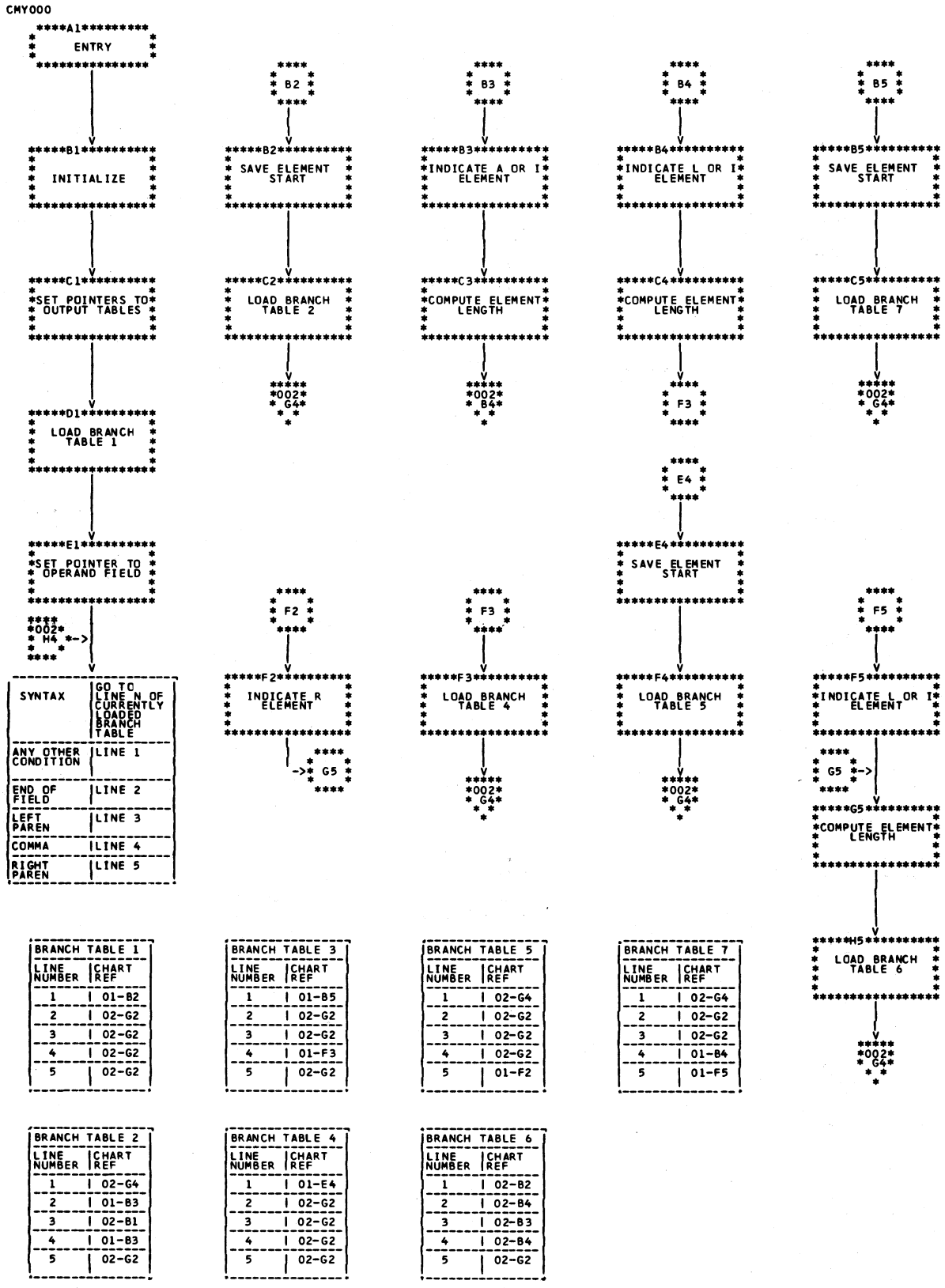


Chart BC (Part 1 of 2). Operand Syntax Testing Routine (CMY000)

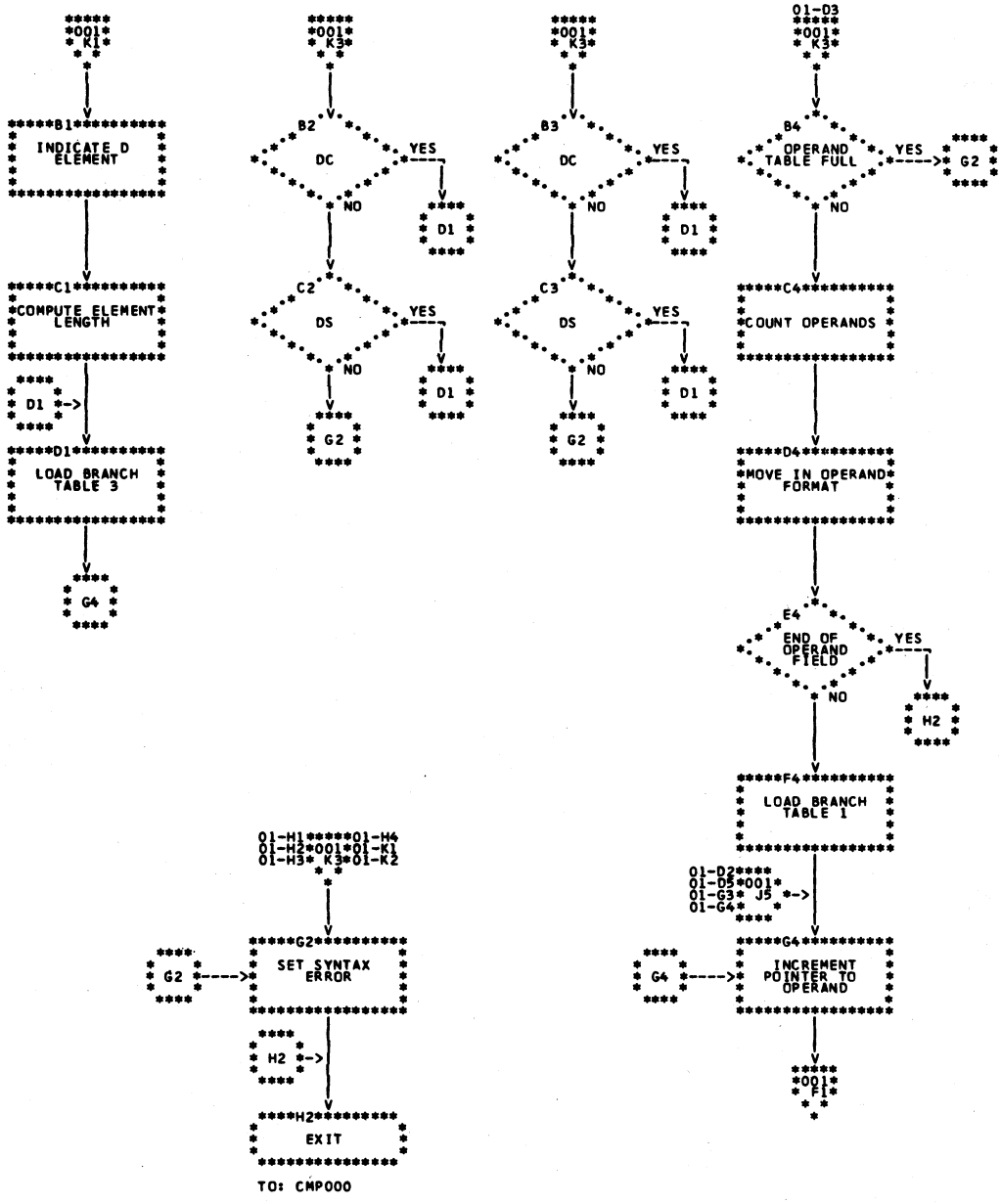


Chart BC (Part 2 of 2). Operand Syntax Testing Routine (CMY000)

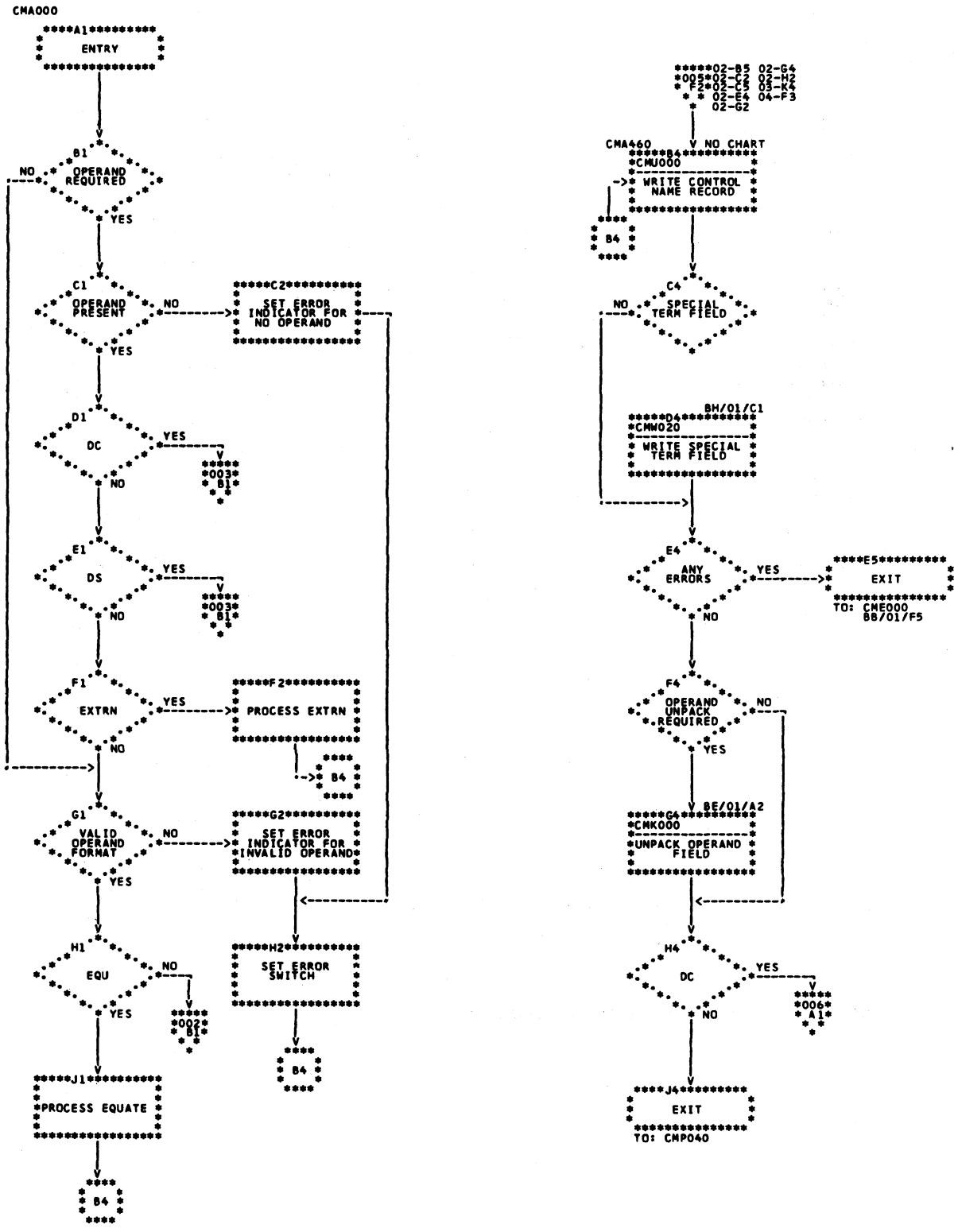


Chart BD (Part 1 of 6). Assembler Instruction Processing Routine (CMA000)

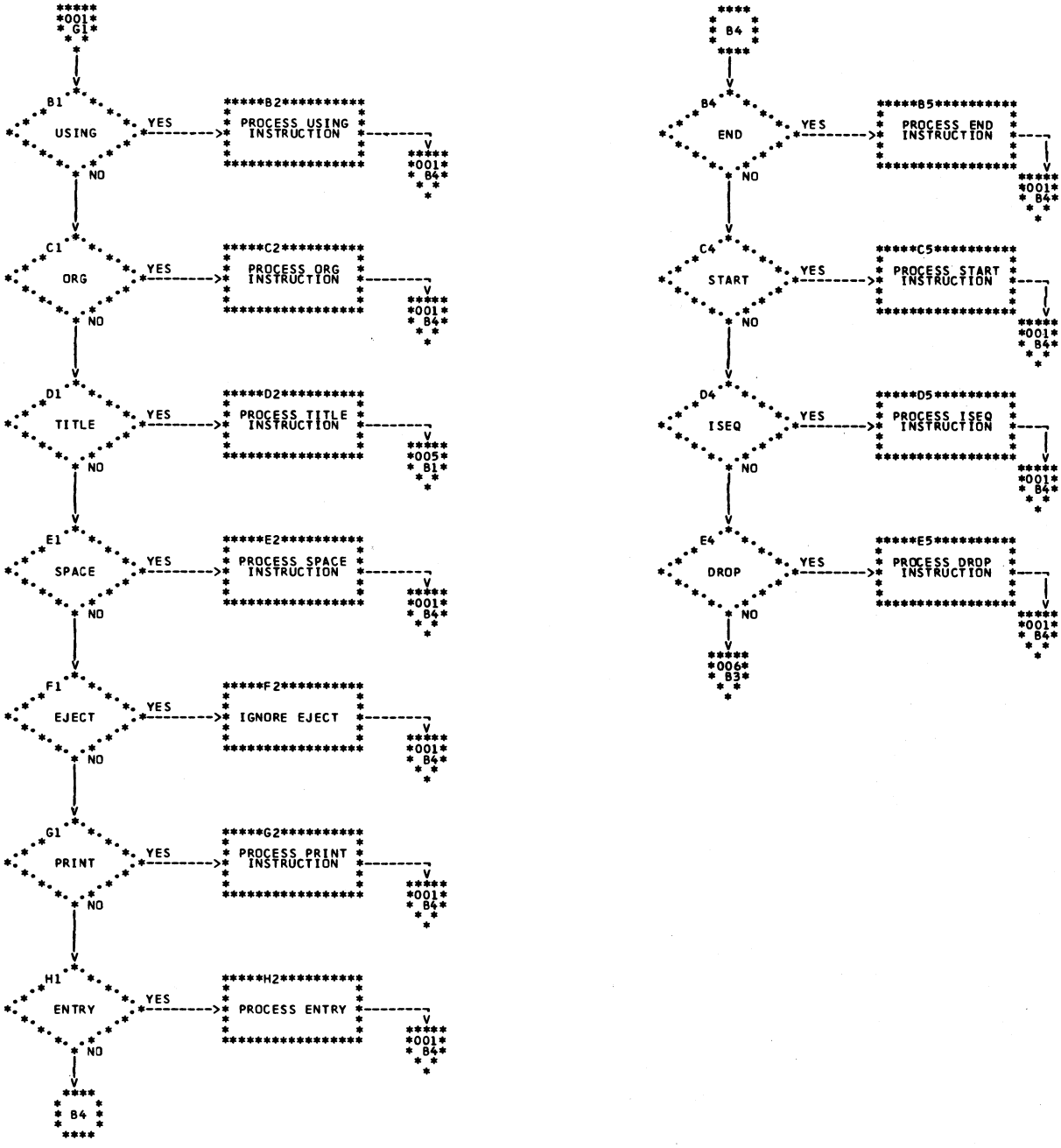


Chart BD (Part 2 of 6). Assembler Instruction Processing Routine (CMA000)

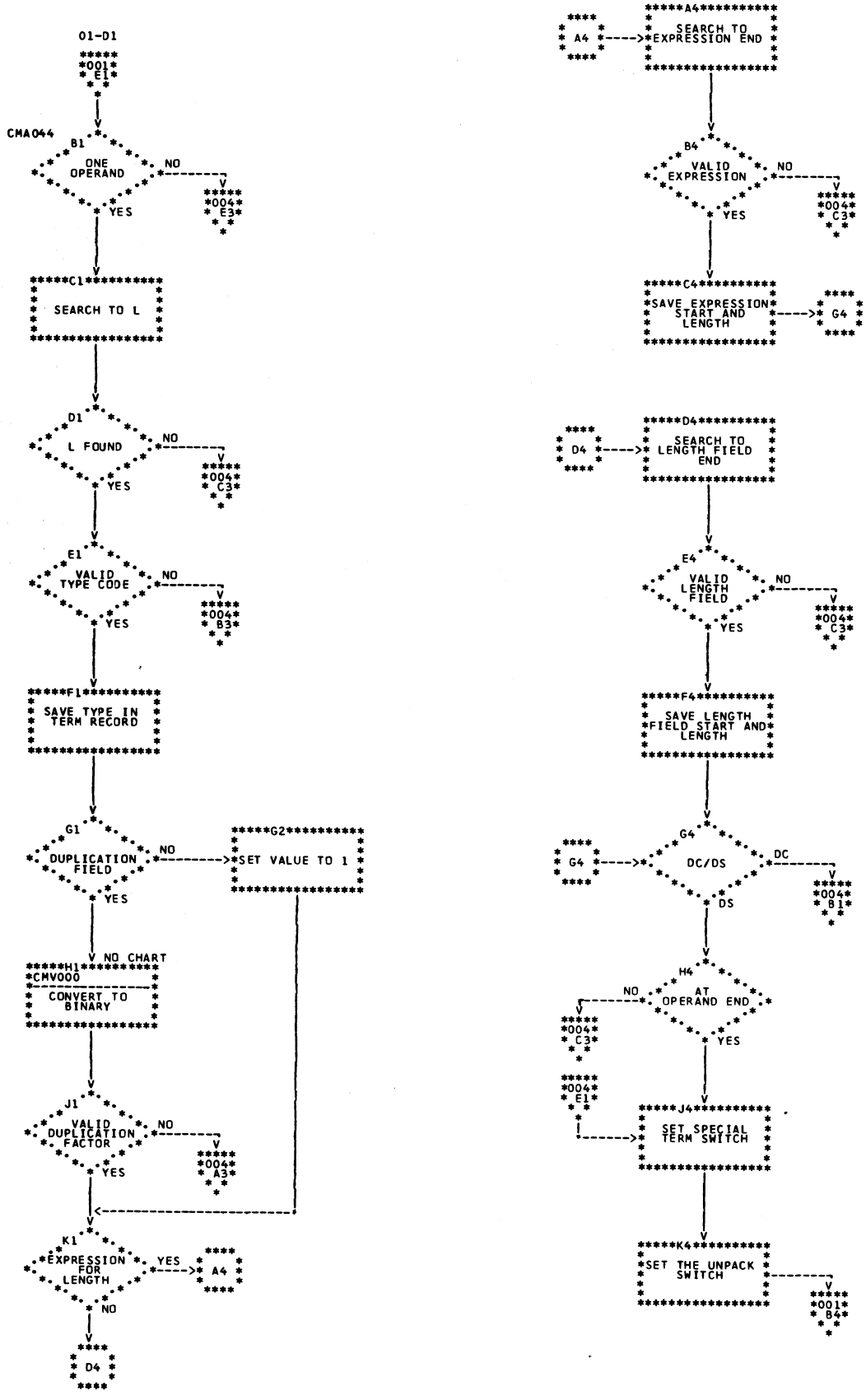


Chart BD (Part 3 of 6). Assembler Instruction Processing Routine (CMA000)

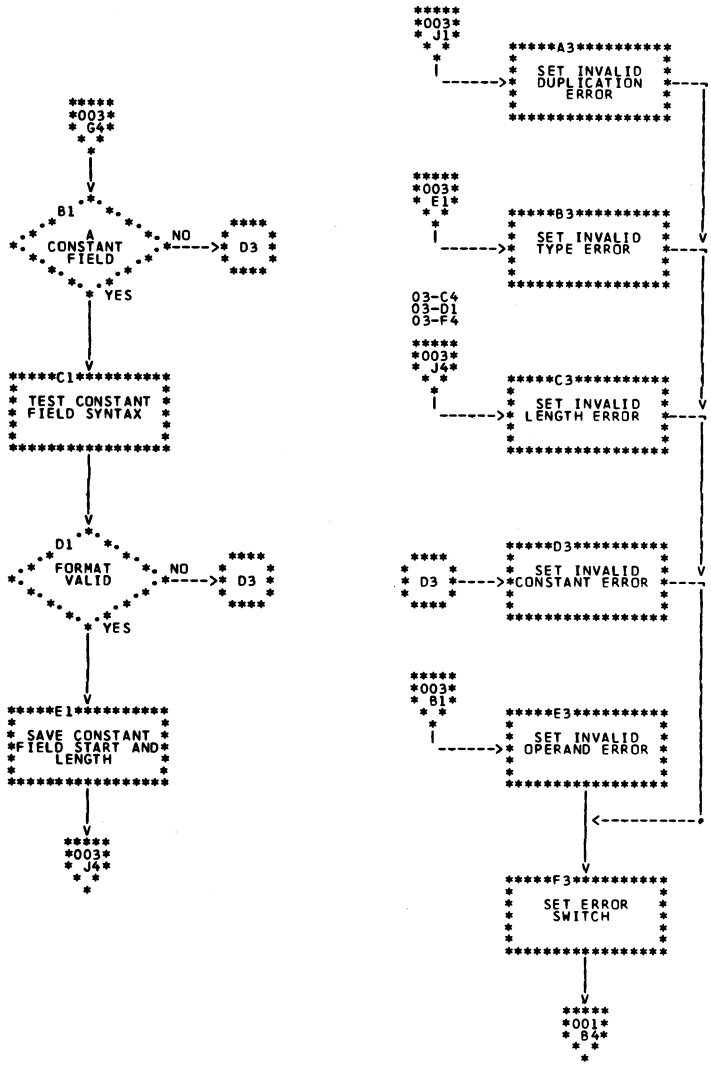


Chart BD (Part 4 of 6). Assembler Instruction Processing Routine (CMA000)

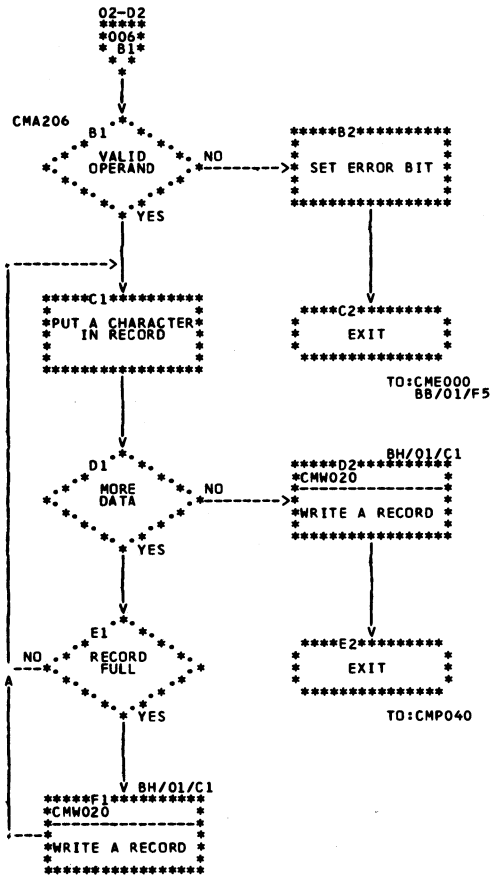


Chart BD (Part 5 of 6). Assembler Instruction Processing Routine (CMA000)

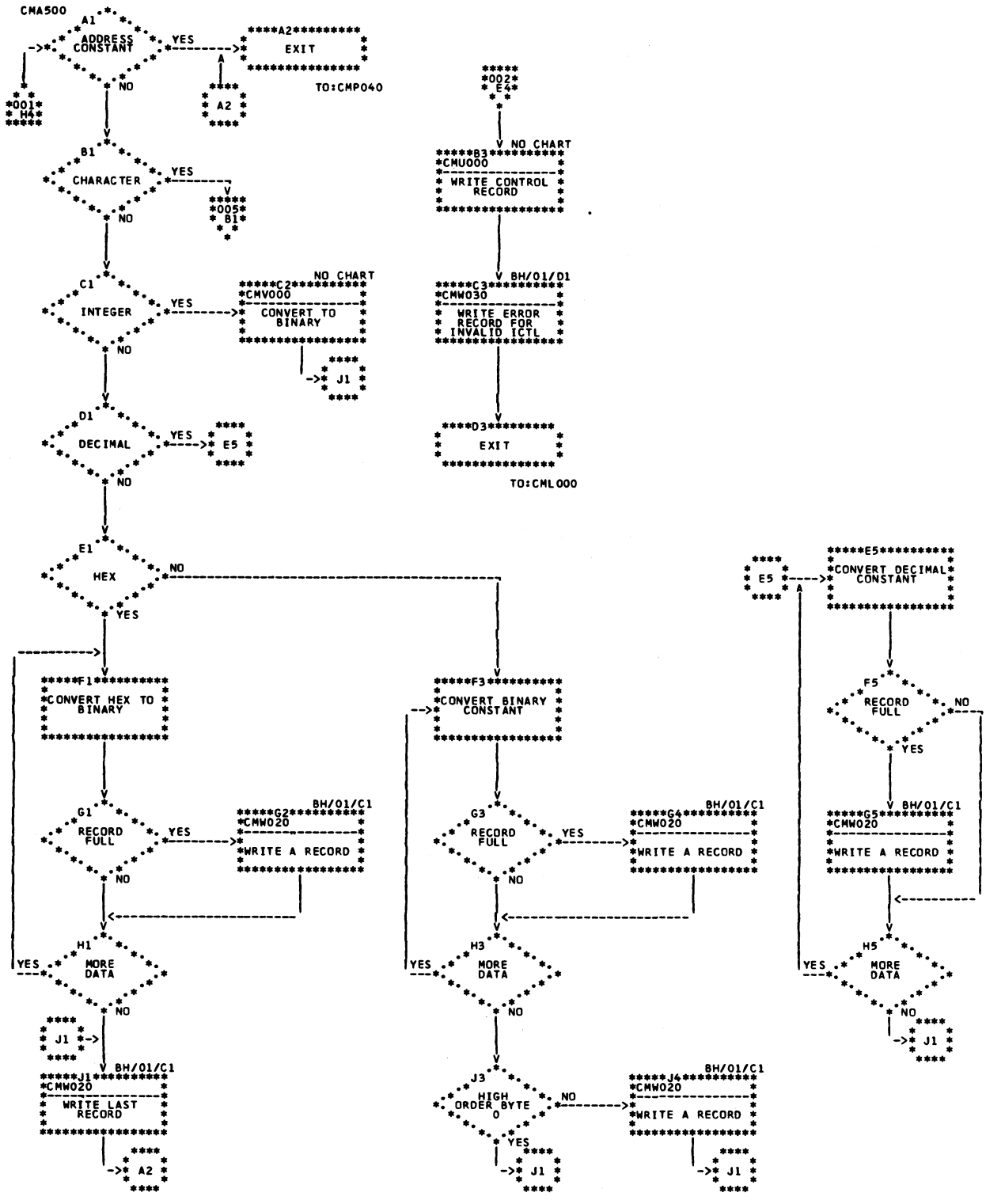


Chart BD (Part 6 of 6). Assembler Instruction Processing Routine (CMA000)

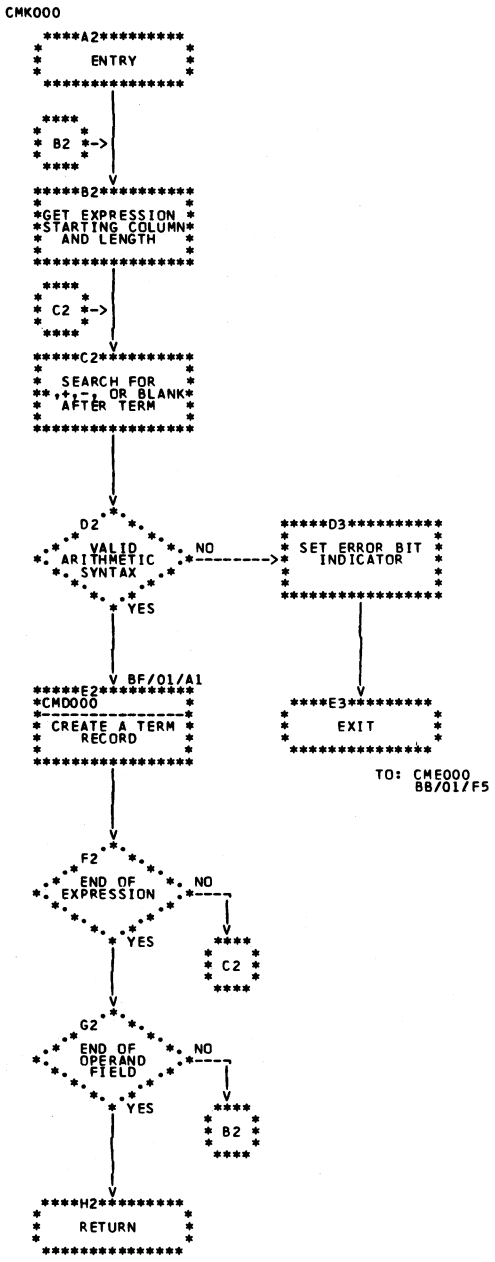


Chart BE. Prepare Operand for Unpack Routine (CMK000)

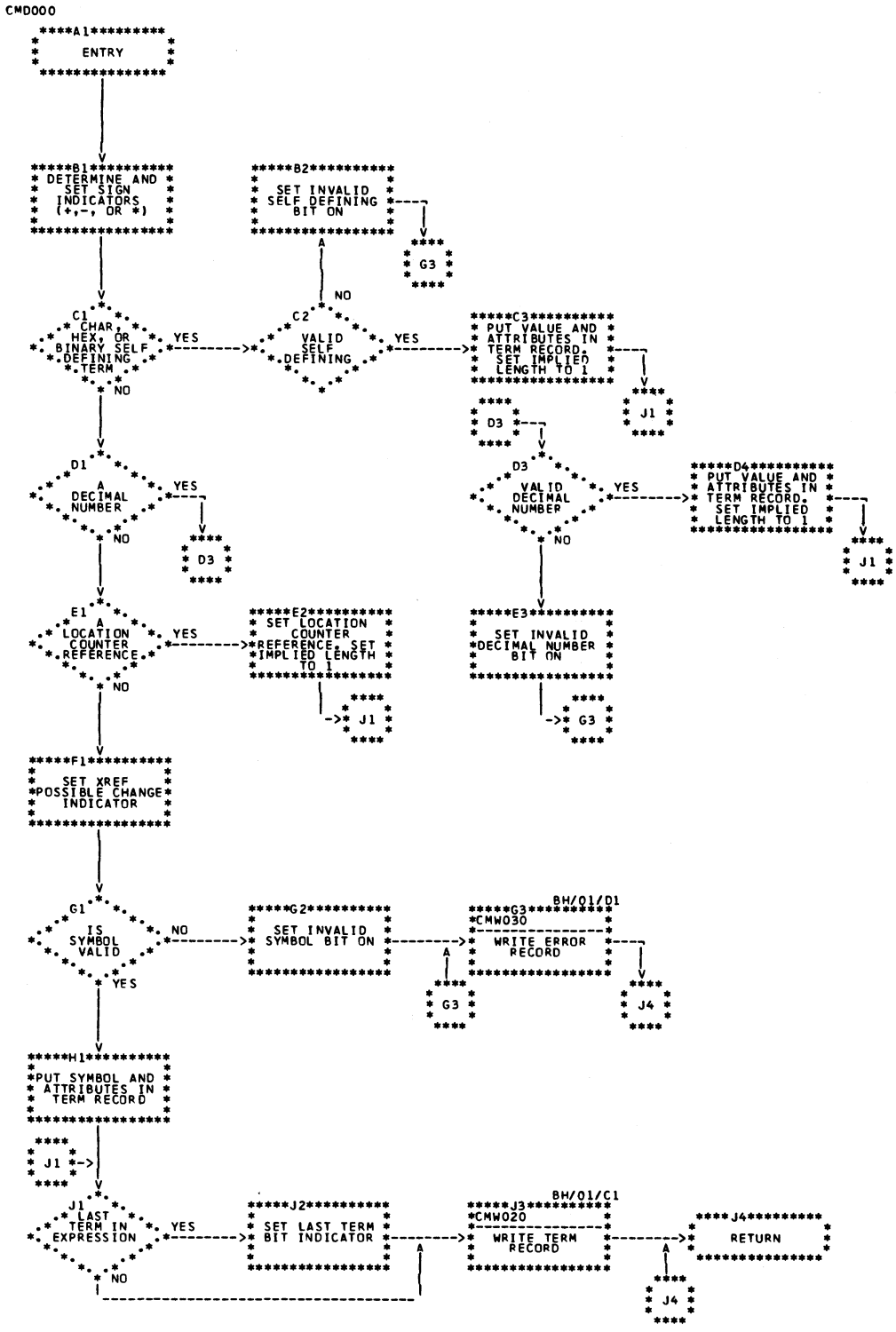


Chart BF. Operand Unpack Routine (CMD000)

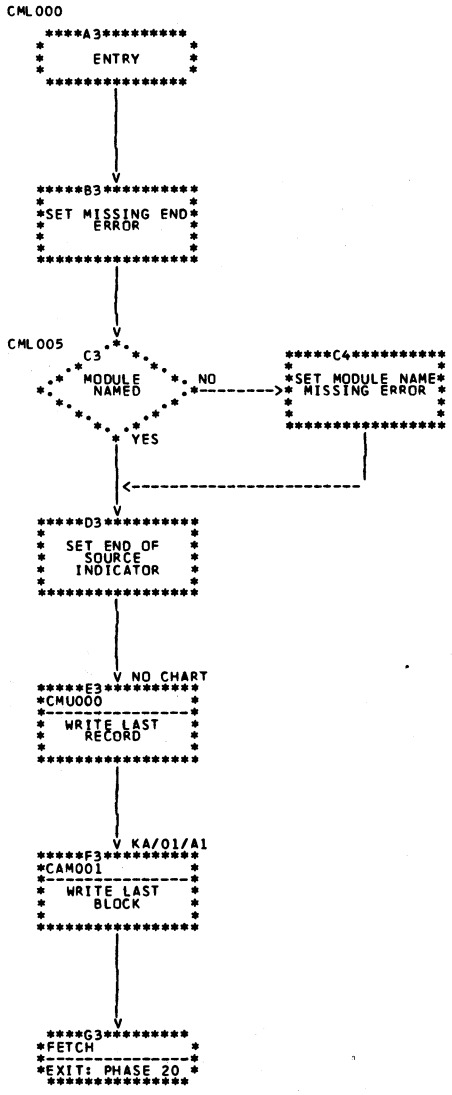


Chart BG. Termination Routine (CML000)

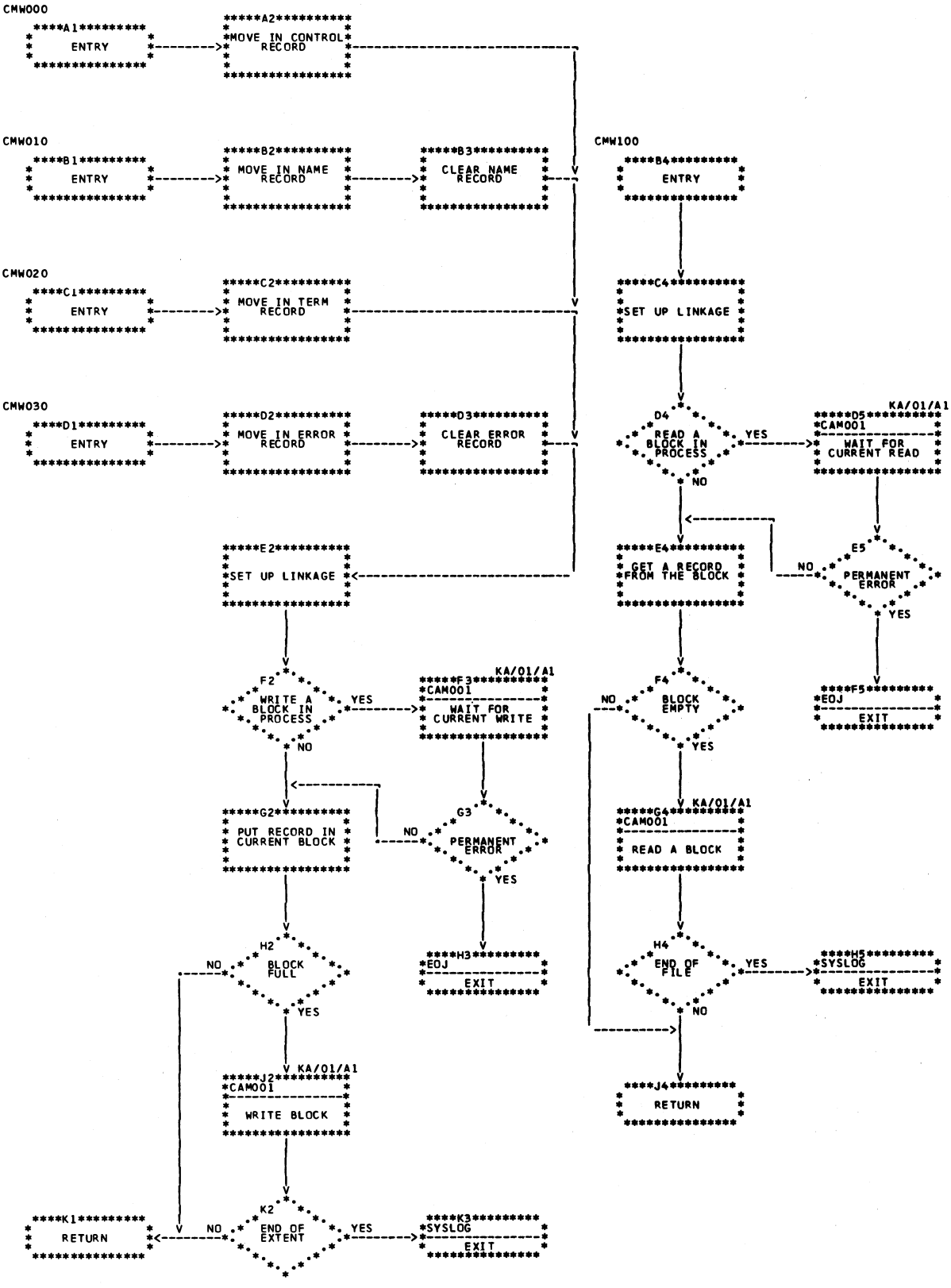


Chart BH. Disk Data Management Interfaces (CMW000)

MODULE $ASSB0–SYMBOL PROCESSING (PHASE 20)

Main Storage Map: Figure 6.

Entry Point: $ASSB1 – entered from Phase 05, Phase 10 or Phase 21 via FETCH. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Supervisor and IOS Logic Manual,* SY21-0033.)

Charts: CA-CK.

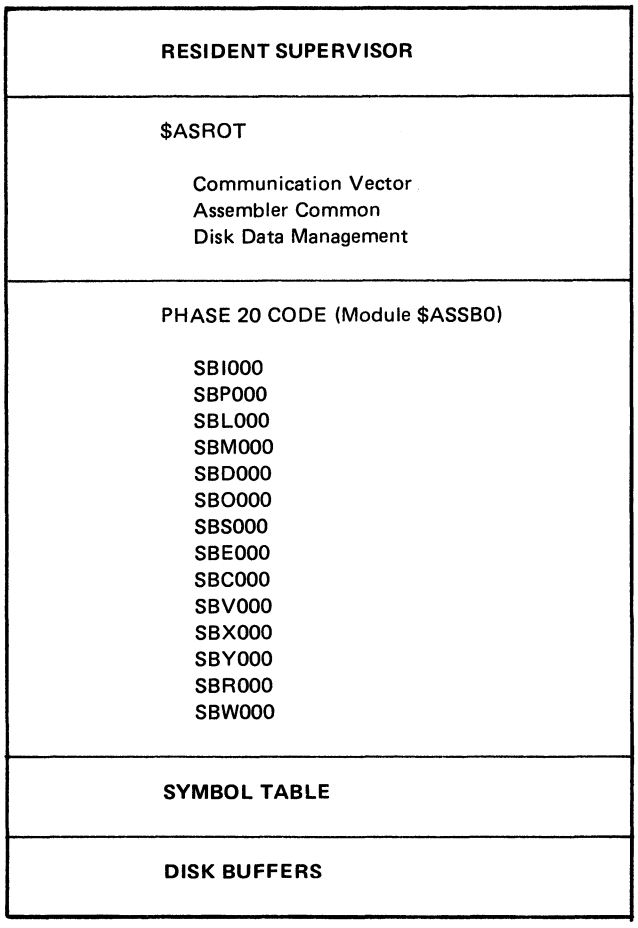


Figure 6. Phase 20 Main Storage Map

Functions:

- Initializes Phase 20 for processing (SBI000–Chart CA).
- Controls Phase 20 processing of the intermediate text file until end of data is found (SBP000–Chart CB).
- Performs tasks associated with Phase 20 termination (SBL000–no chart).
- Performs Phase 20 processing for machine instructions. Symbols are placed in symbol table (SBM000–Chart CC).
- Tests DC/DS specification for valid length and for duplication. Stores symbol (if present) in table (SBD000–Chart CD).
- Performs Phase 20 processing of ORG and EQU statements (SBO000–Chart CE).
- Processes START instruction (SBS000–Chart CF).
- Processes EXTRN instruction (SBE000–Chart CJ).
- Updates location counter by object length of current statement (SBC000–no chart).
- Checks for overflow of symbol table (SBC000–no chart).
- Checks maximum location counter value (SBC000–no chart).
- Evaluates all expressions in the operand field of the current source record (SBV000–Chart CG).
- Performs multiplication calculations (SBX000–no chart).
- Moves symbol and its attributes to symbol table (SBY000 –Chart CH).
- Counts symbols in table; controls calling of Phase 21 when table overflows (SBY000–Chart CH).
- Searches symbol table for a given symbol (SBR000–Chart CI).
- Provides interfaces with data management for disk operations (SBW000, SBW010, SBW100–Chart CK).

Input:

- Intermediate Text on the Work File

  1. Name Records – used to build symbol table.

  2. Control Records – used to obtain lengths for location counter assignments and to assign symbol lengths, attributes, and values.

  3. Term Records – used to obtain length allocation for DC/DS statements, location counter changes for ORG and symbol length, attributes, and values for EQU statements.

Output:

- Symbol Table – built in lower end of main storage; designed to fill all space between Phase 20 coding and disk buffers.

- Symbol table parameters in ASMCOM – used by Phases 21 and 22 for accessing the symbol table.

Exits:

- Normal

  1. Control passed to Phase 21 on symbol table overflow.

  2. Control passed to Phase 22 when last (or only) symbol table is processed.

- Error – control returned to scheduler via Halt/Syslog or EOJ transients. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Scheduler Logic Manual,* SY21-0033.)

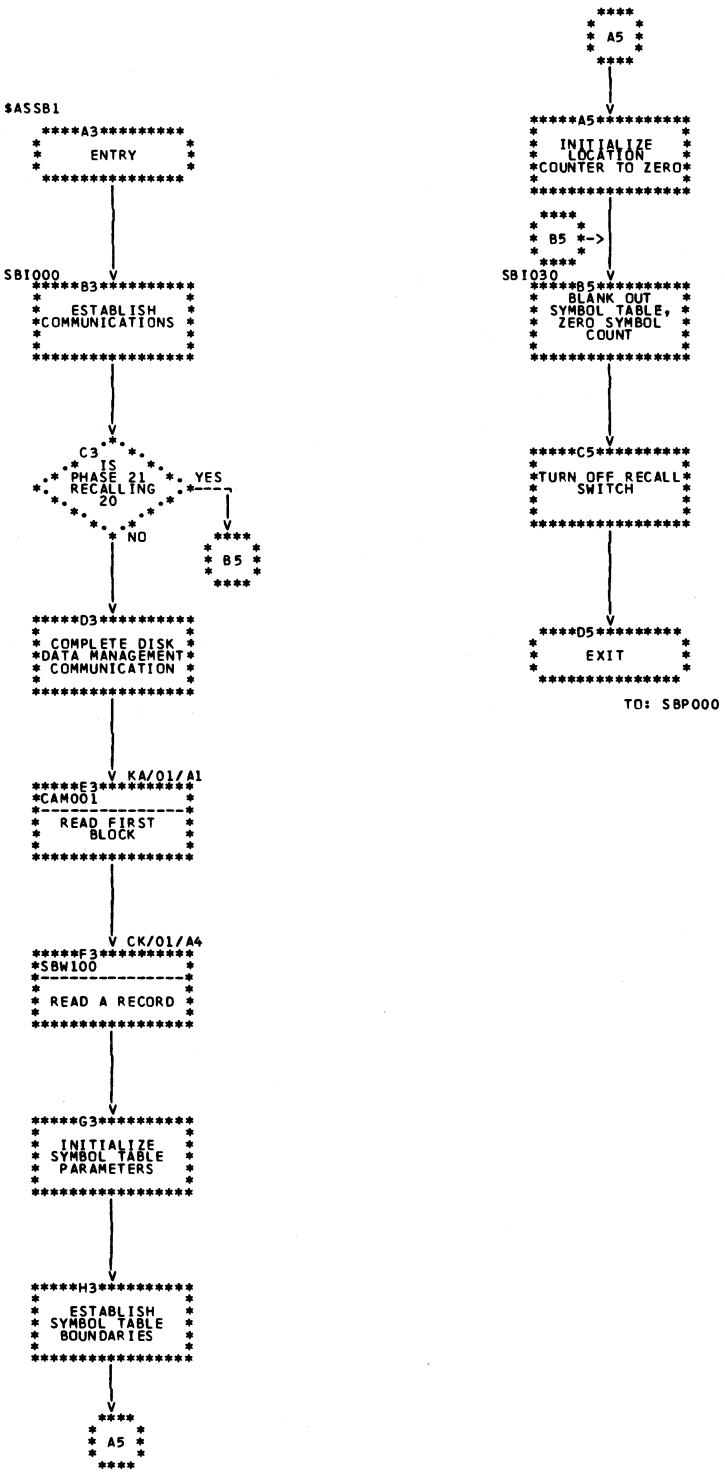


Chart CA. Initialization Routine (SBI000)

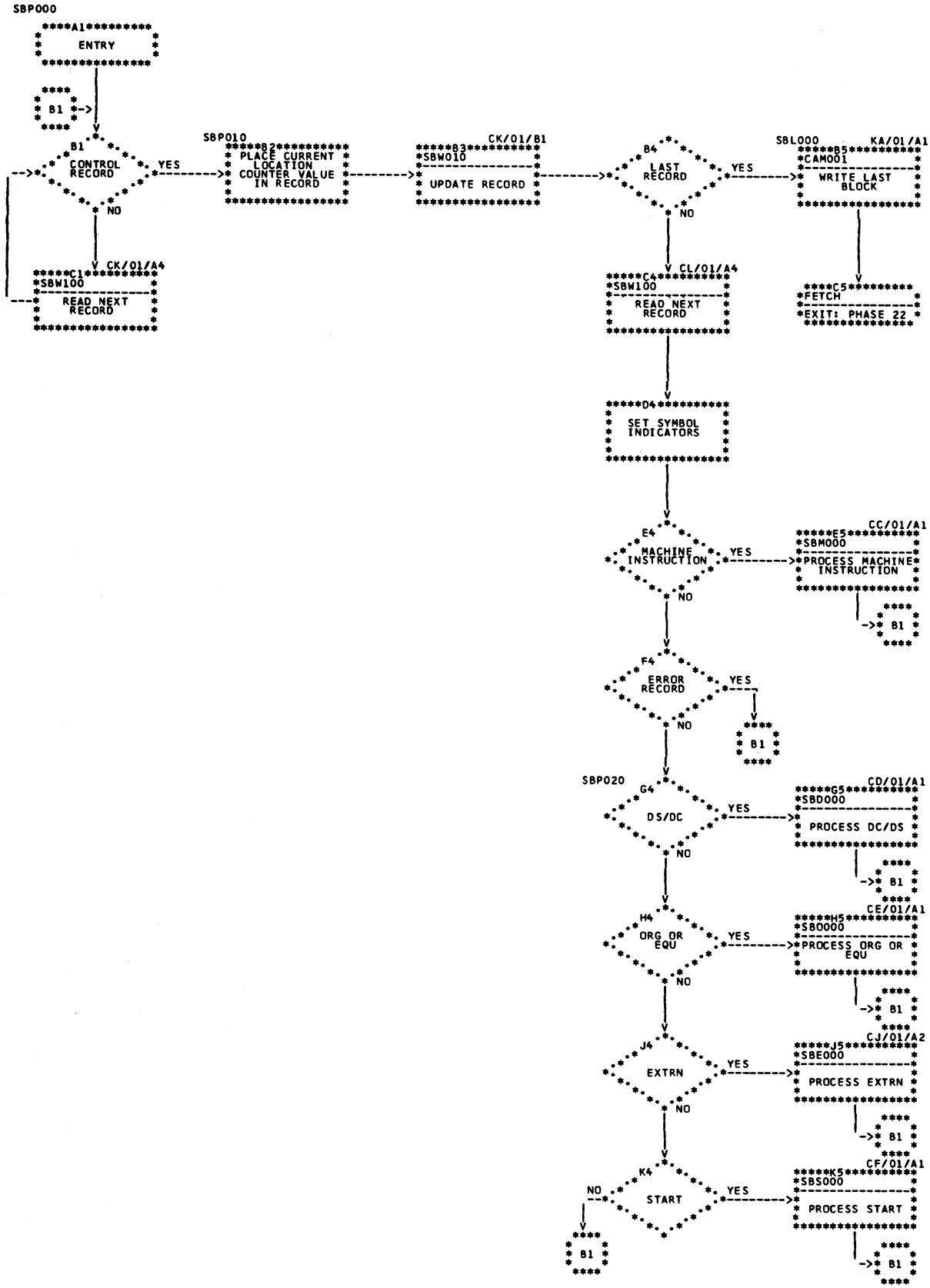


Chart CB. Main Control Routine (SBP000)

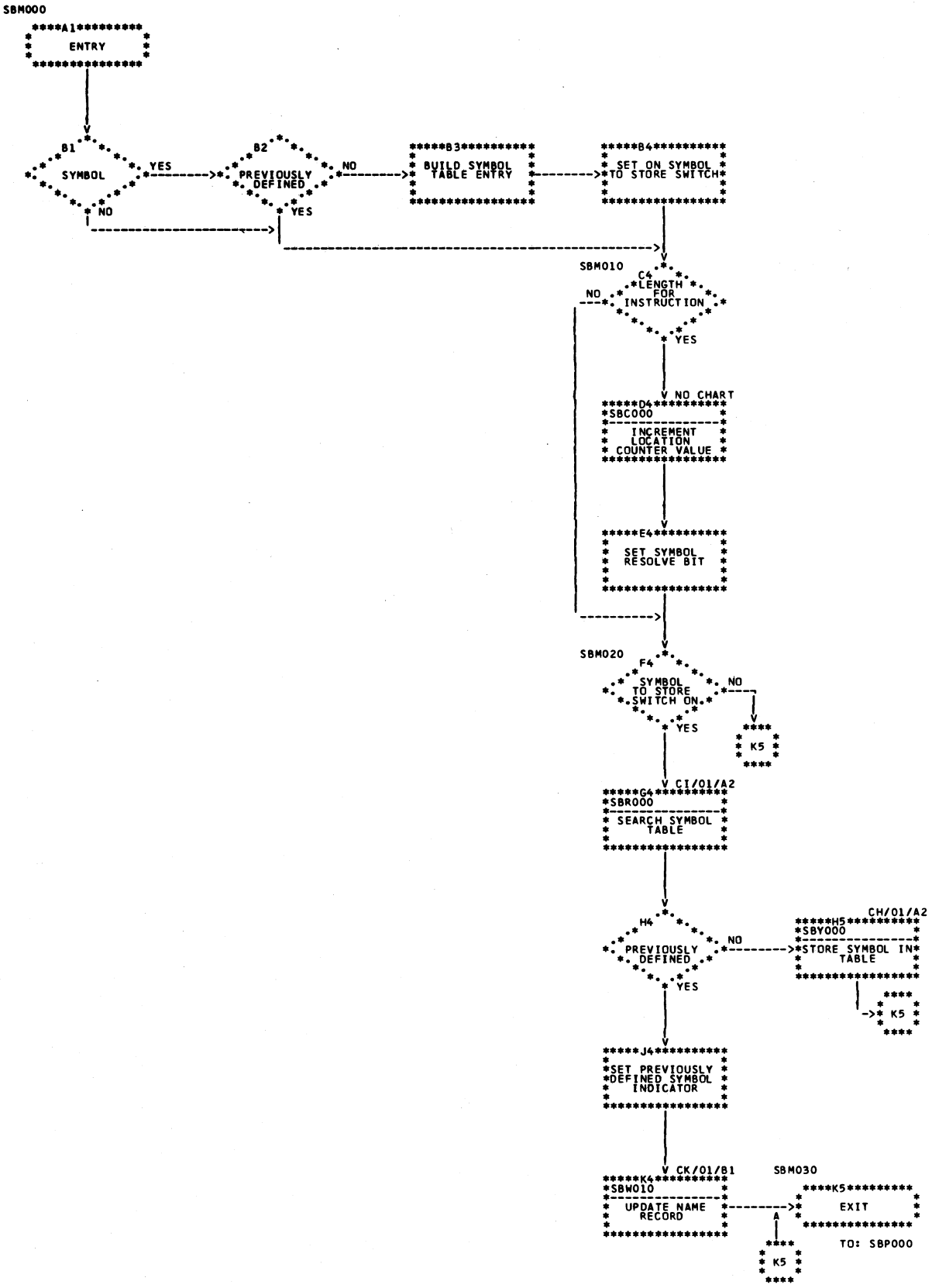


Chart CC. Machine Instruction Processing Routine (SBM000)

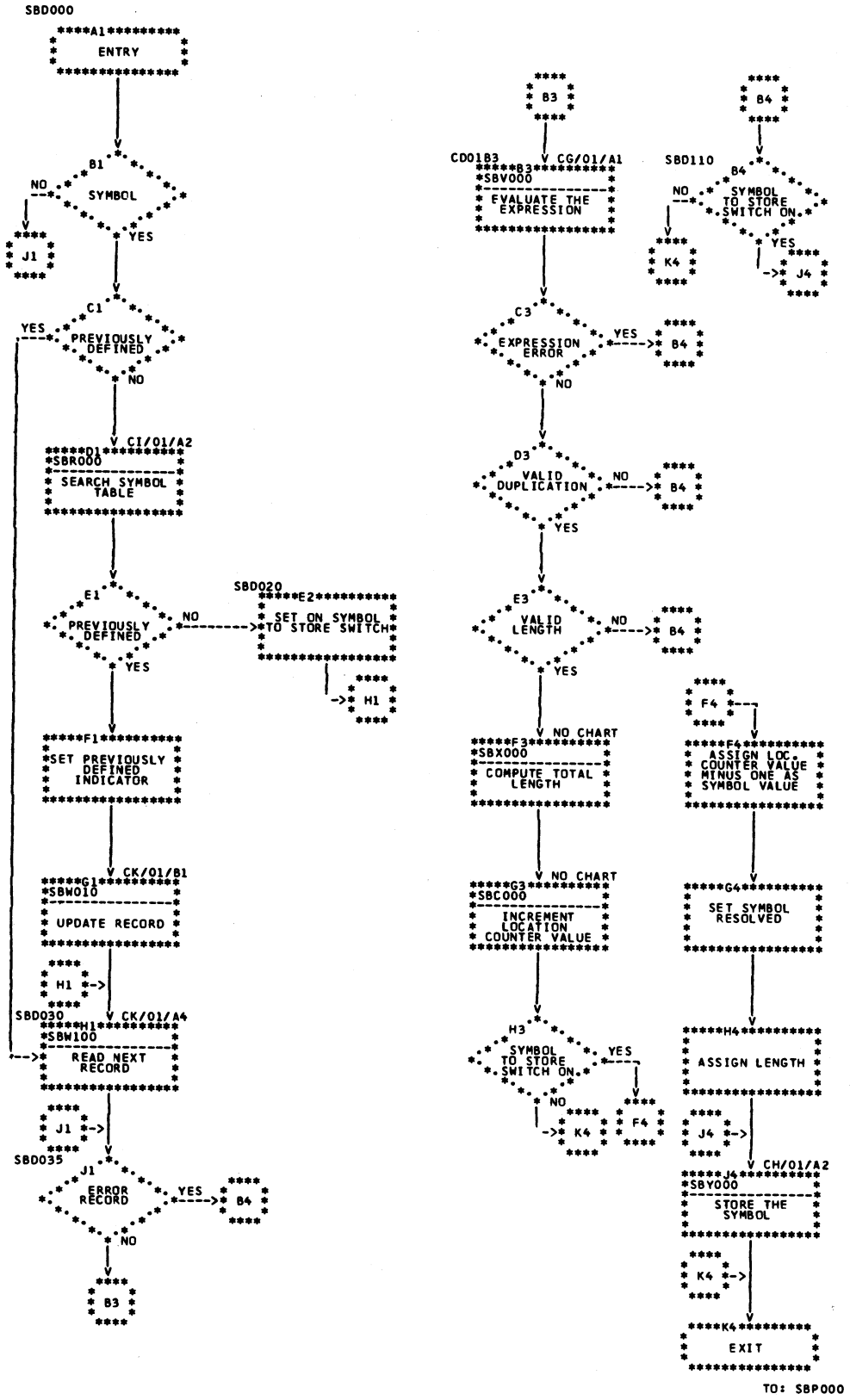


Chart CD. DC/DS Processing Routine (SBD000)

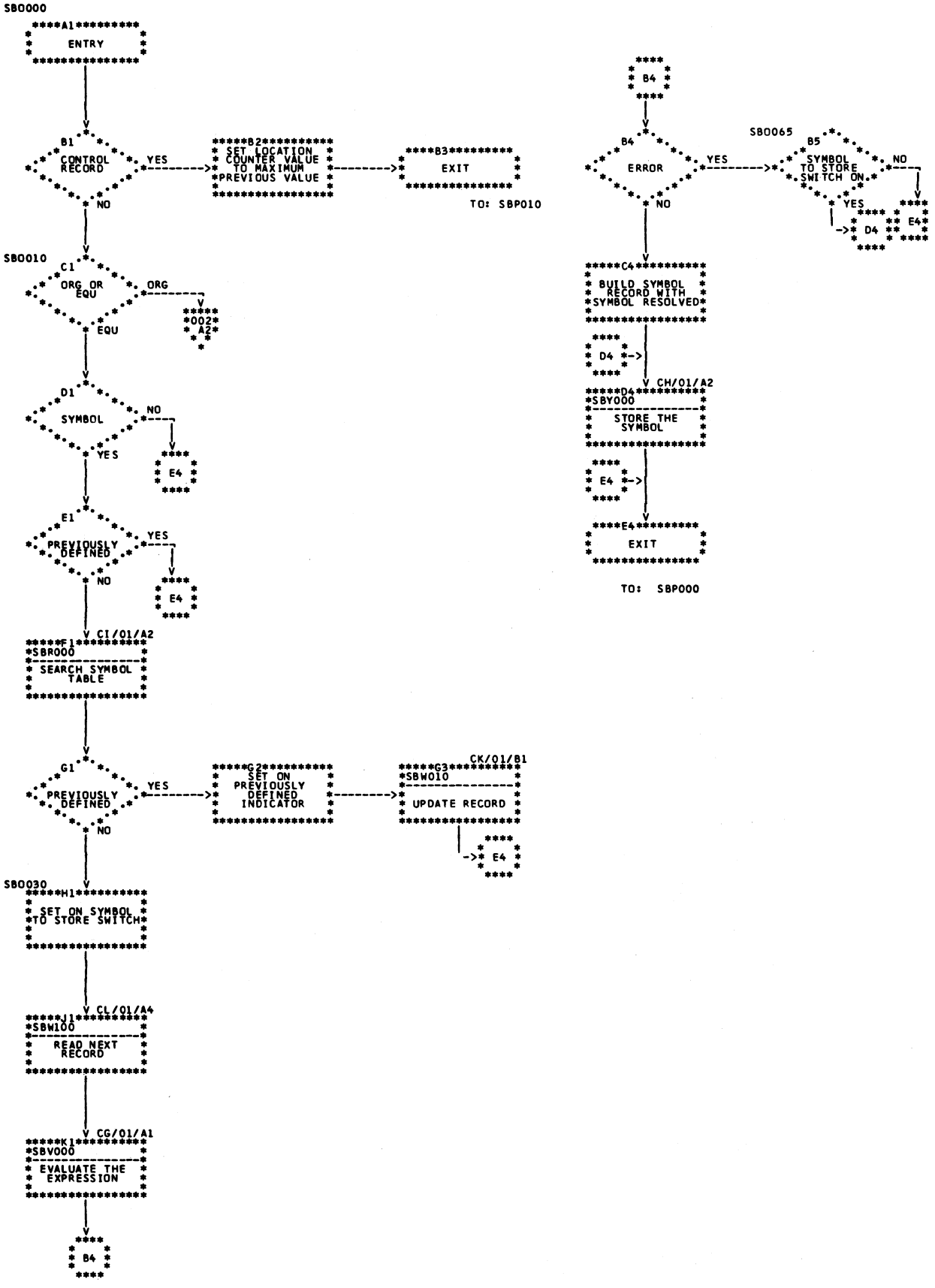


Chart CE (Part 1 of 2). ORG/EQU Processing Routine (SBO000)

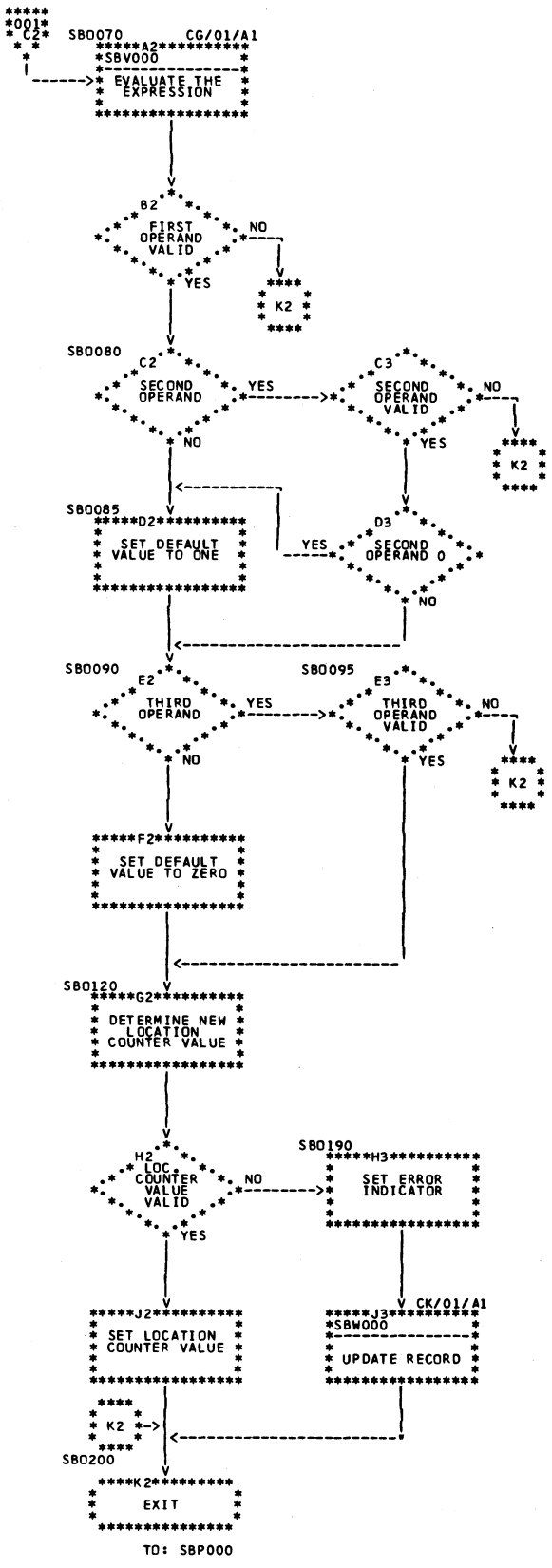


Chart CE (Part 2 of 2). ORG/EQU Processing Routine (SBO000)

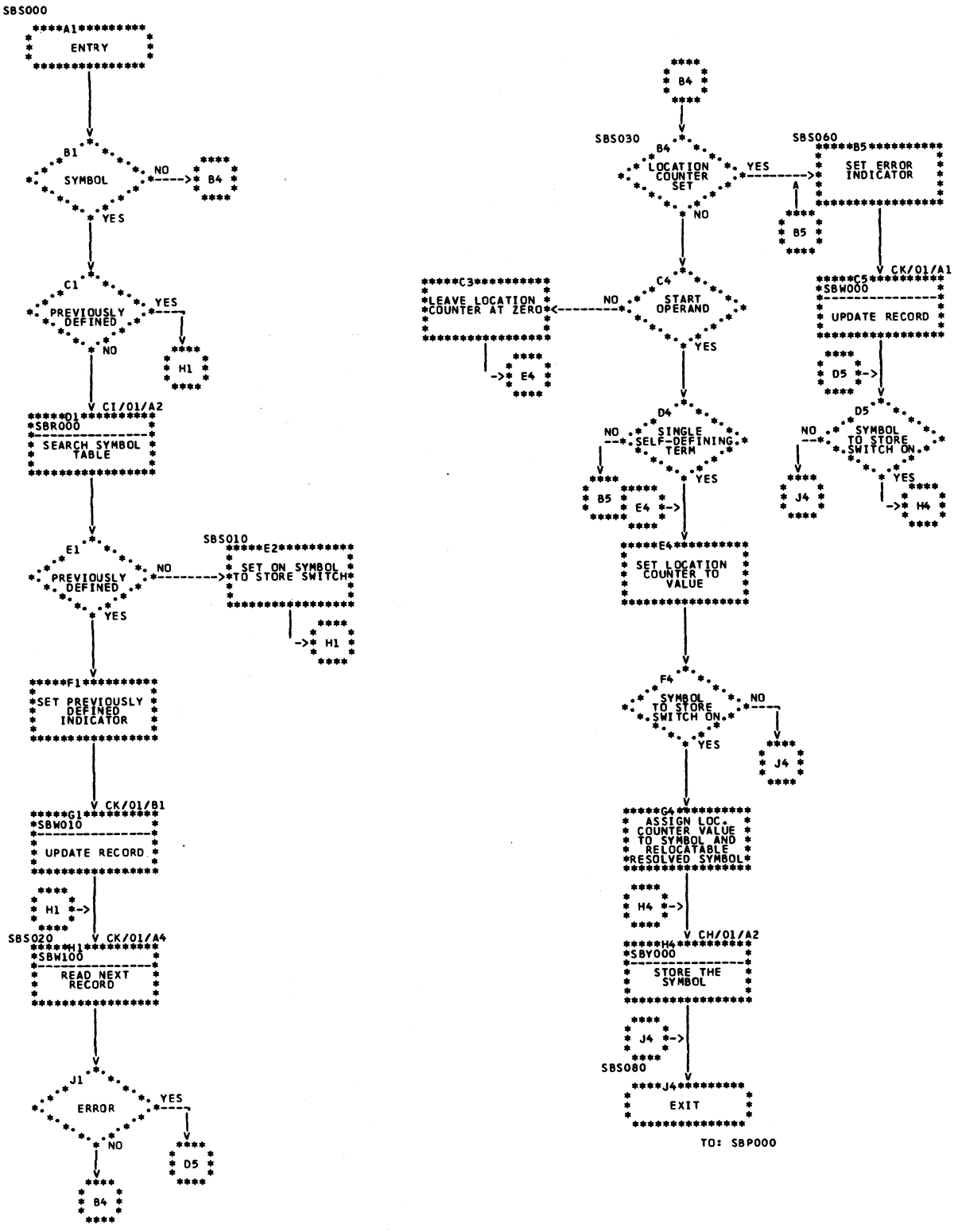


Chart CF. START Instruction Processing Routine (SBS000)

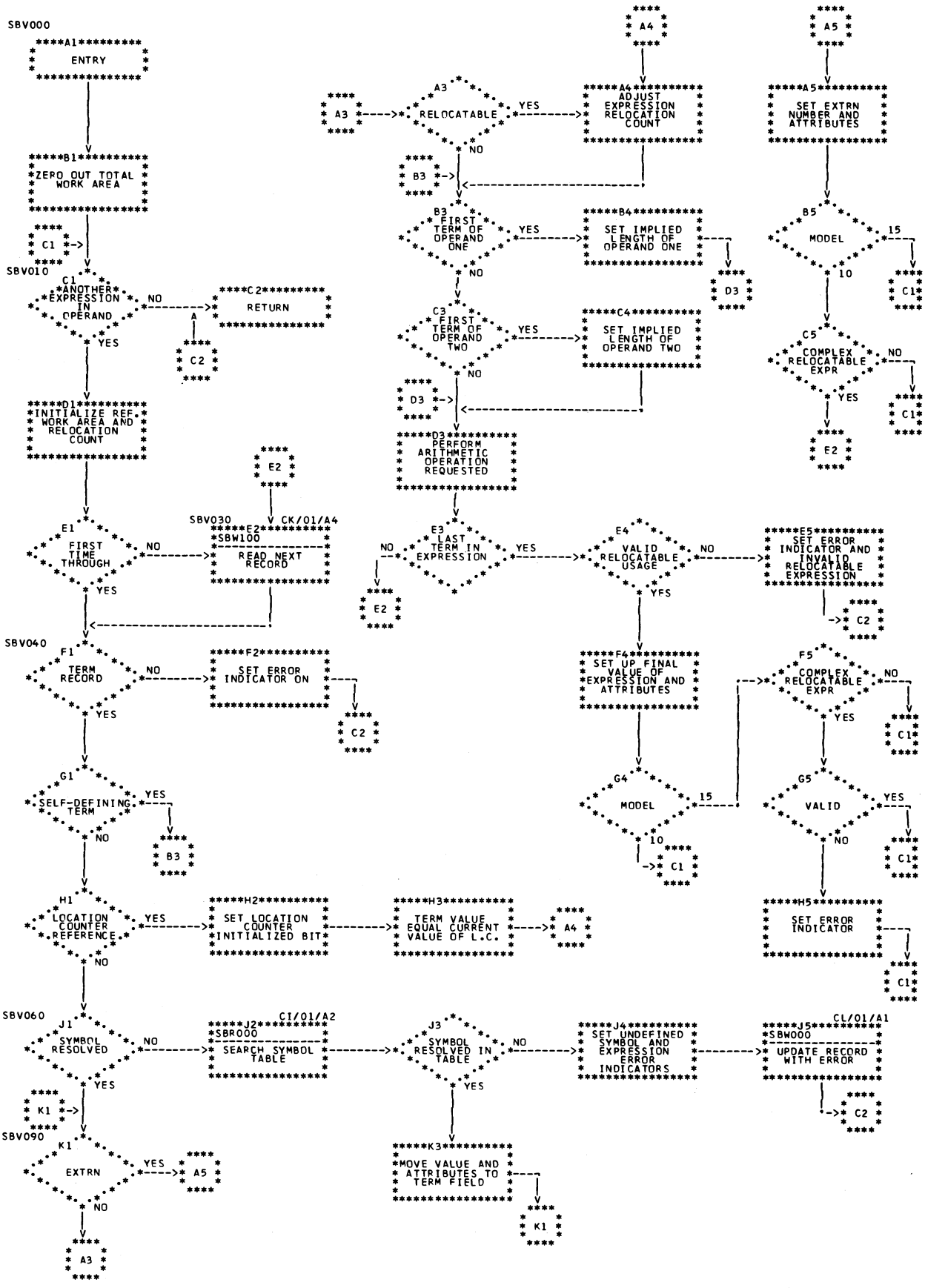


Chart CG. Expression Evaluation Routine (SBV000)

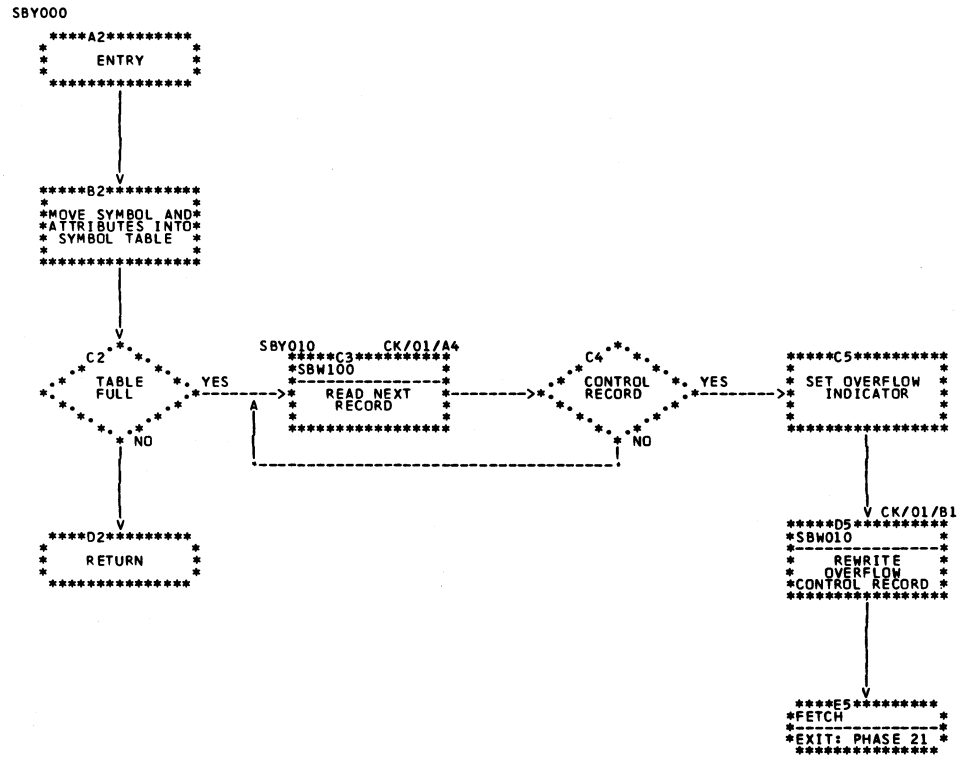


Chart CH. Store and Count Symbol Routine (SBY000)

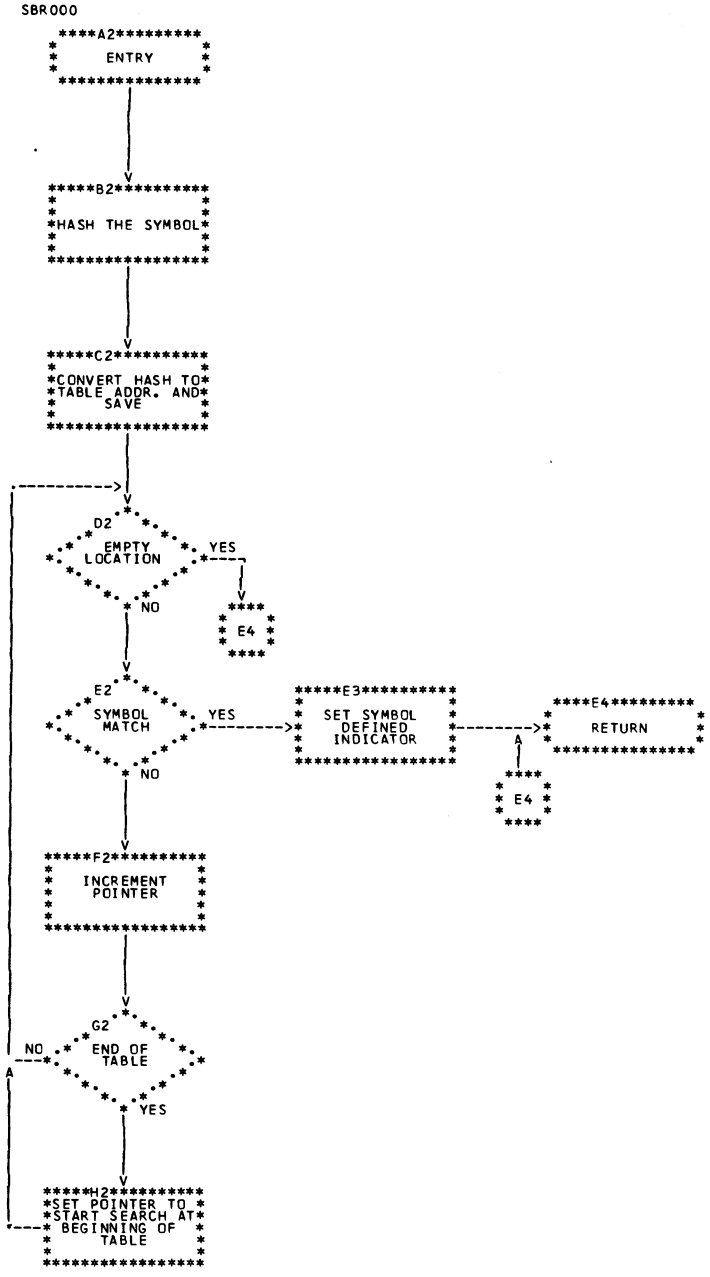


Chart CI. Search Symbol Table Routine (SBR000)

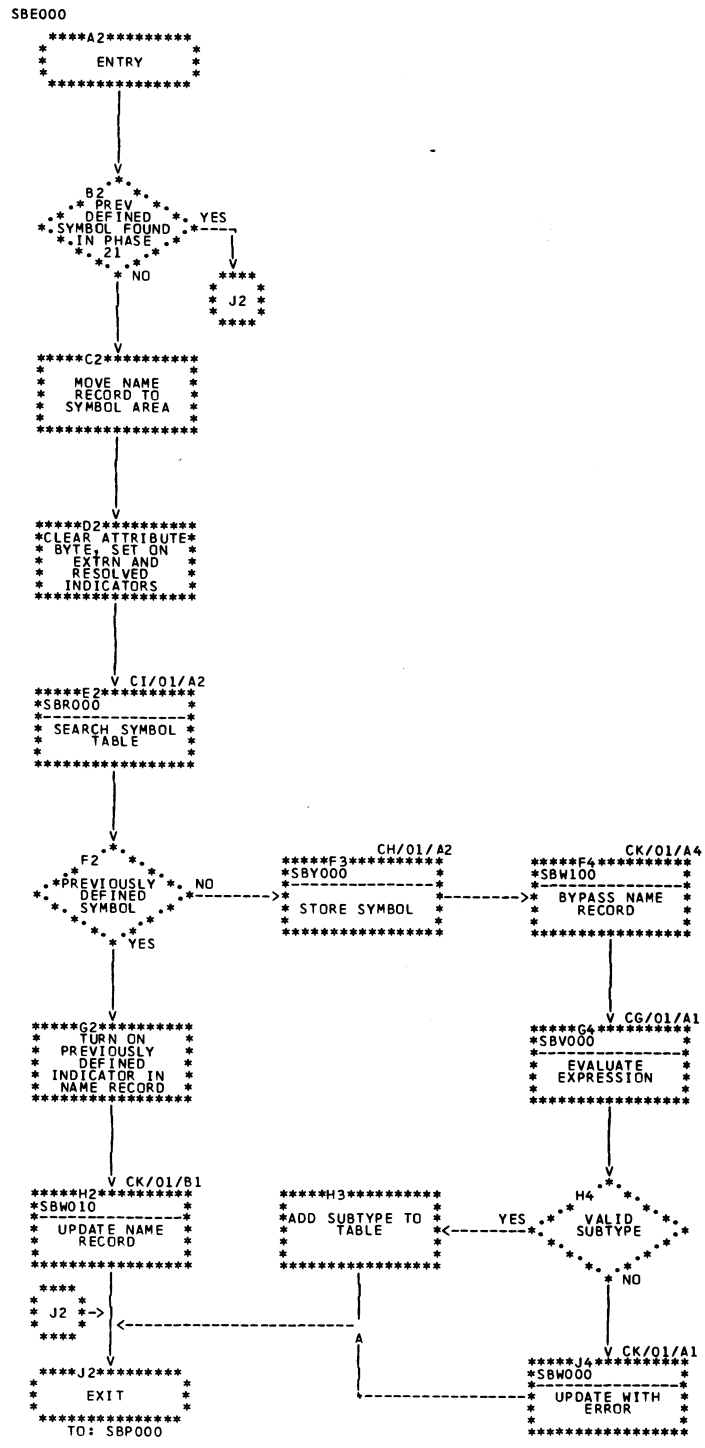


Chart CJ. EXTRN Instruction Processing Routine (SBE000)

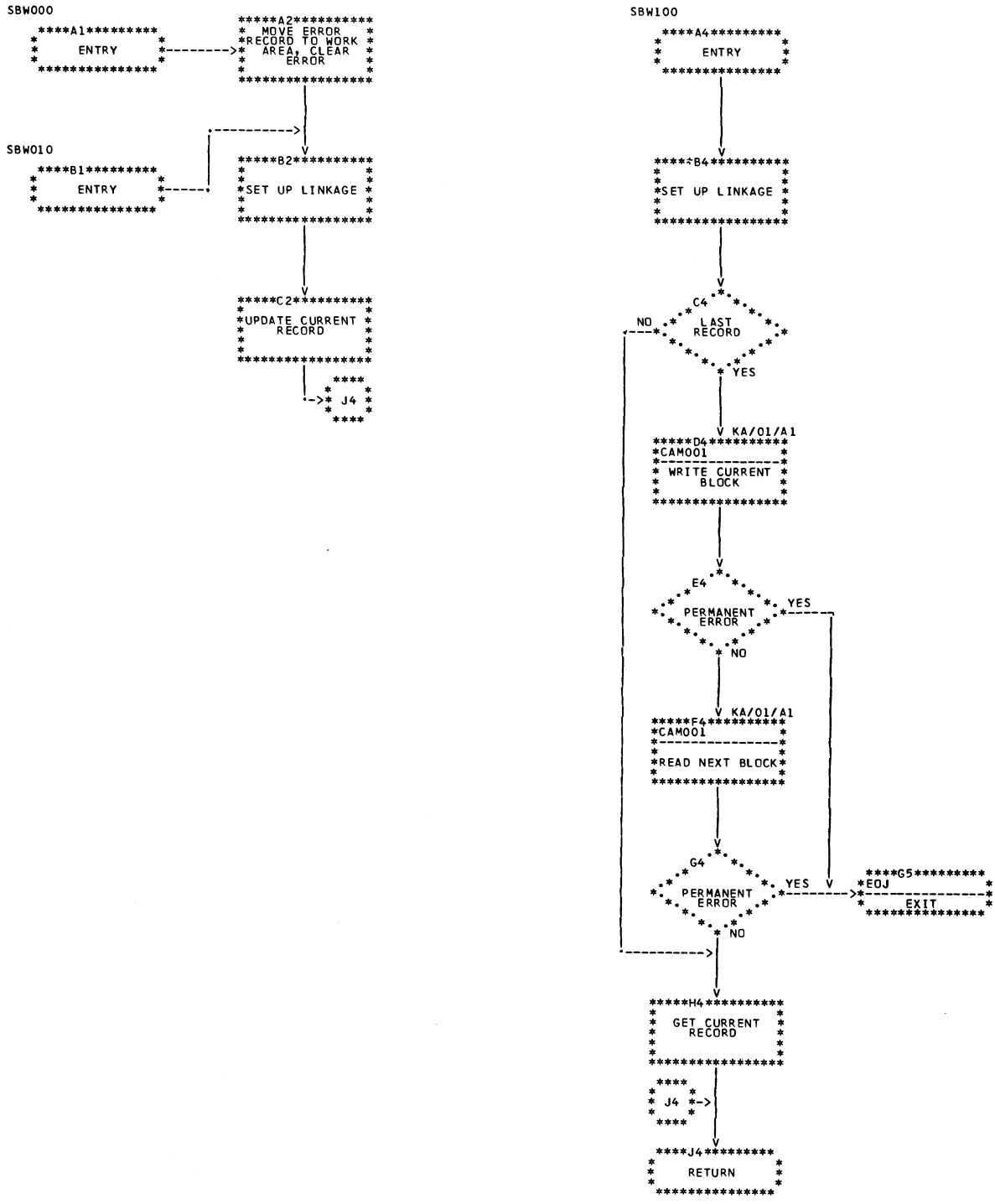


Chart CK. Disk Data Management Interfaces (SBW000)

MODULE $ASSF0—SYMBOL TABLE OVERFLOW PROCESSING (PHASE 21)

Main Storage Map: Figure 7.

Entry Point: $ASSF1—entered from Phase 20 via FETCH. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Supervisor and IOS Logic Manual,* SY21-0033.)

Charts: DA-DD.

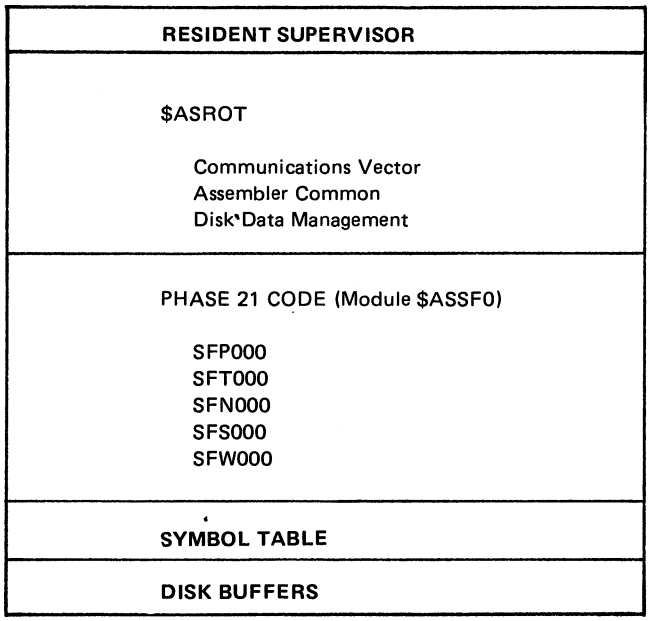


Figure 7. Phase 21 Main Storage Map

Functions:

- Controls Phase 21 processing of the intermediate text in $WORK2 (SFP000—Chart DA).
- Processes intermediate text term records, resolving all symbols defined in the current symbol table (SFT000—Chart DB).
- Tests name records for previously defined symbols (SFN000—Chart DC).
- Searches the symbol table for a given symbol (SFS000—no chart; functionally the same as SBR000 routine in Phase 20).
- Provides interface with data management for disk operations (SFW000—Chart DD).

Input:

- Intermediate text in $WORK2.
- Symbol table.
- ASMCOM—symbol table parameters.

Output:

- All output is to $WORK2

1. Intermediate Text
   a. Symbols in the term records that are defined in the present symbol table are updated with the symbol's value and attributes.
   b. Name records containing symbols which are previously defined are updated with a previously defined symbol indicator.
2. Symbol table entries — if a cross reference has been requested, all symbol table entries are added to the end of $WORK2.

Exits:

- Normal — control passed to Phase 20 for continued symbol processing.
- Error — control returned to scheduler via Halt/Syslog or EOJ transients. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Scheduler Logic Manual,* SY21-0033.)

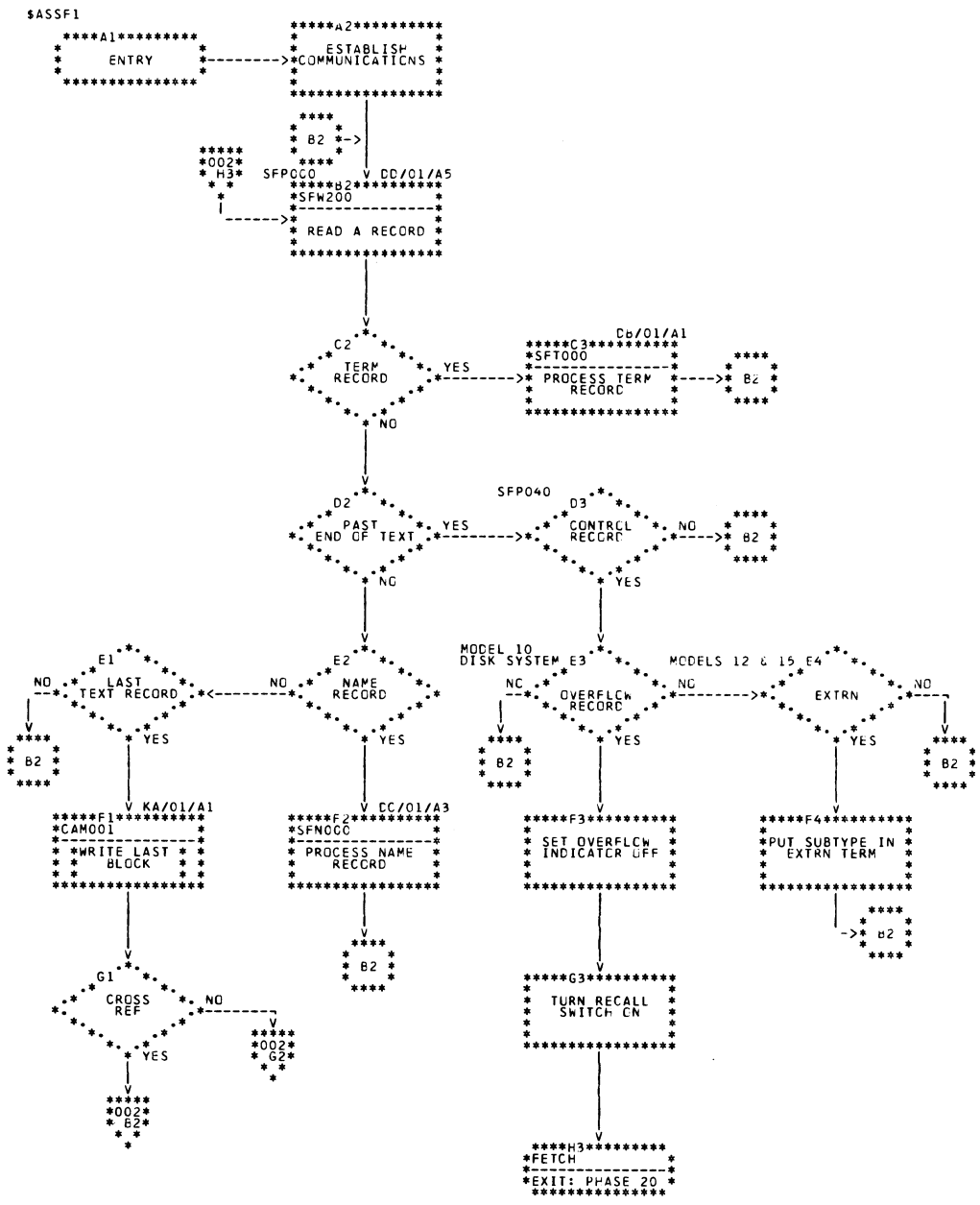


Chart DA (Part 1 of 2). Main Control Routine (SFP000)

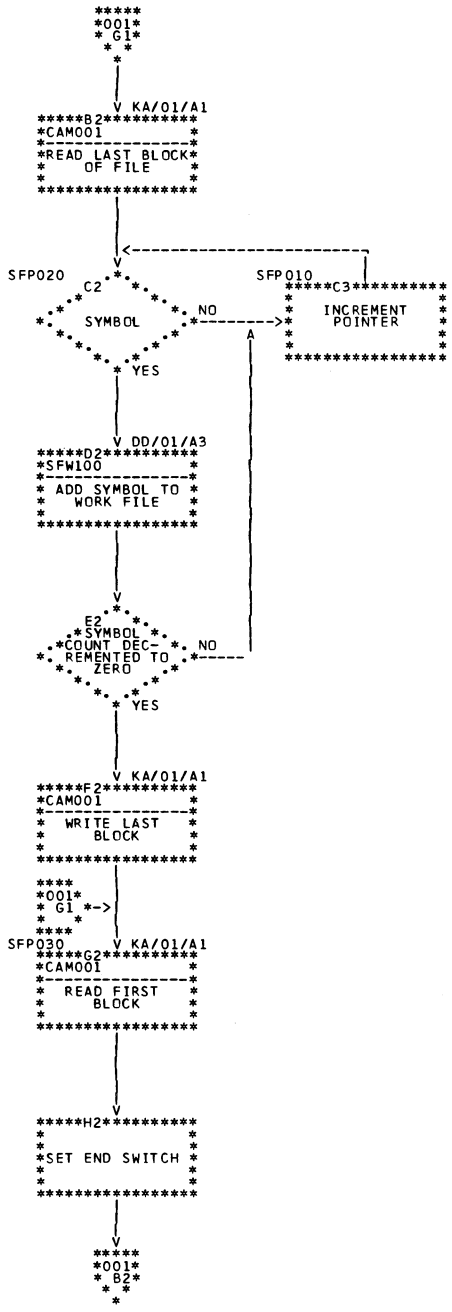


Chart DA (Part 2 of 2). Main Control Routine (SFP000)

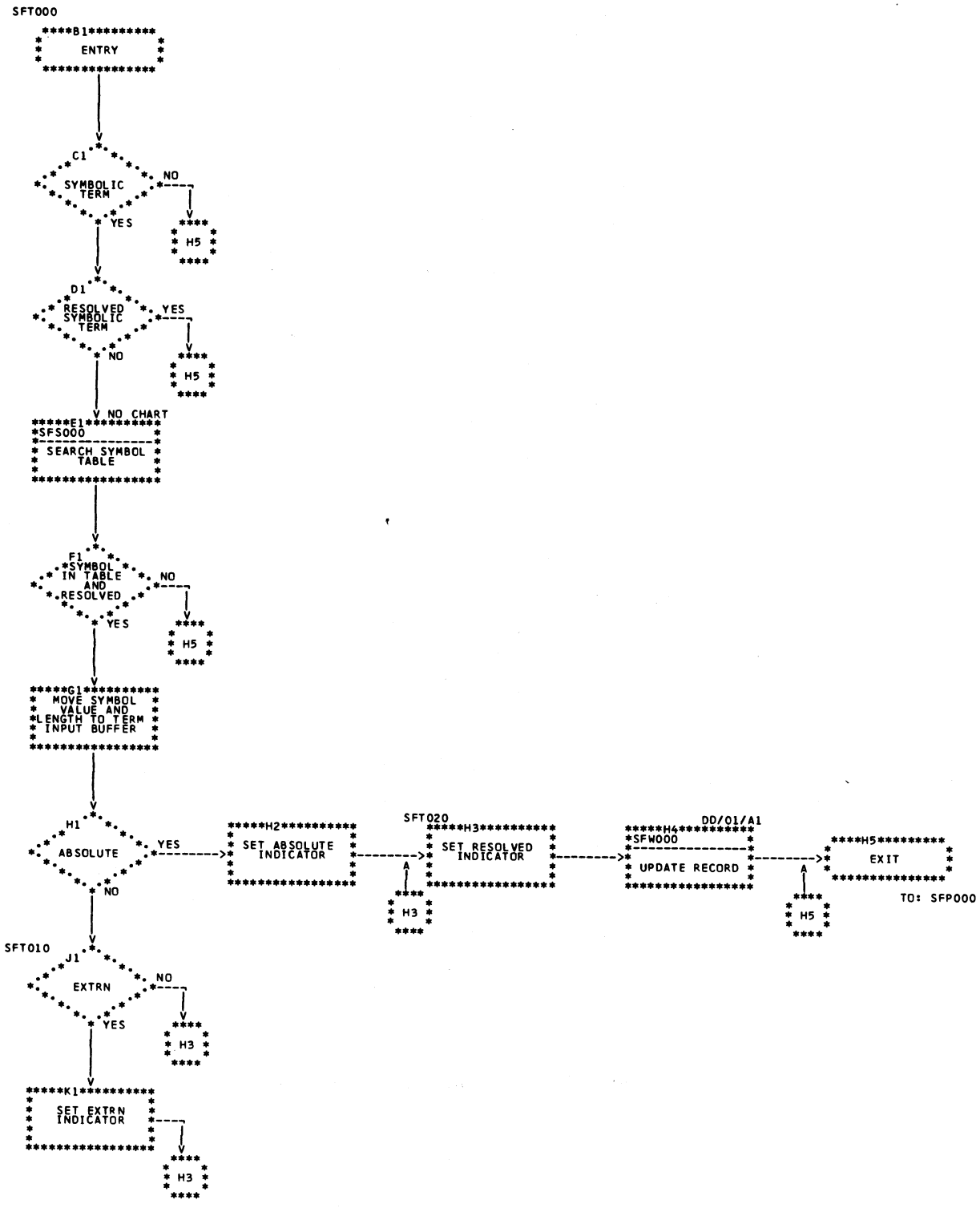


Chart DB. Term Record Processing Routine (SFT000)

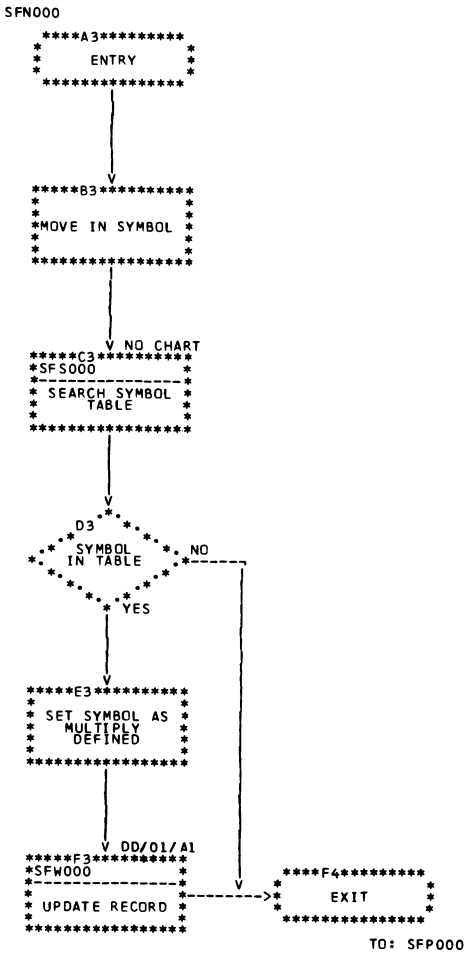


Chart DC. Name Record Processing Routine (SFN000)

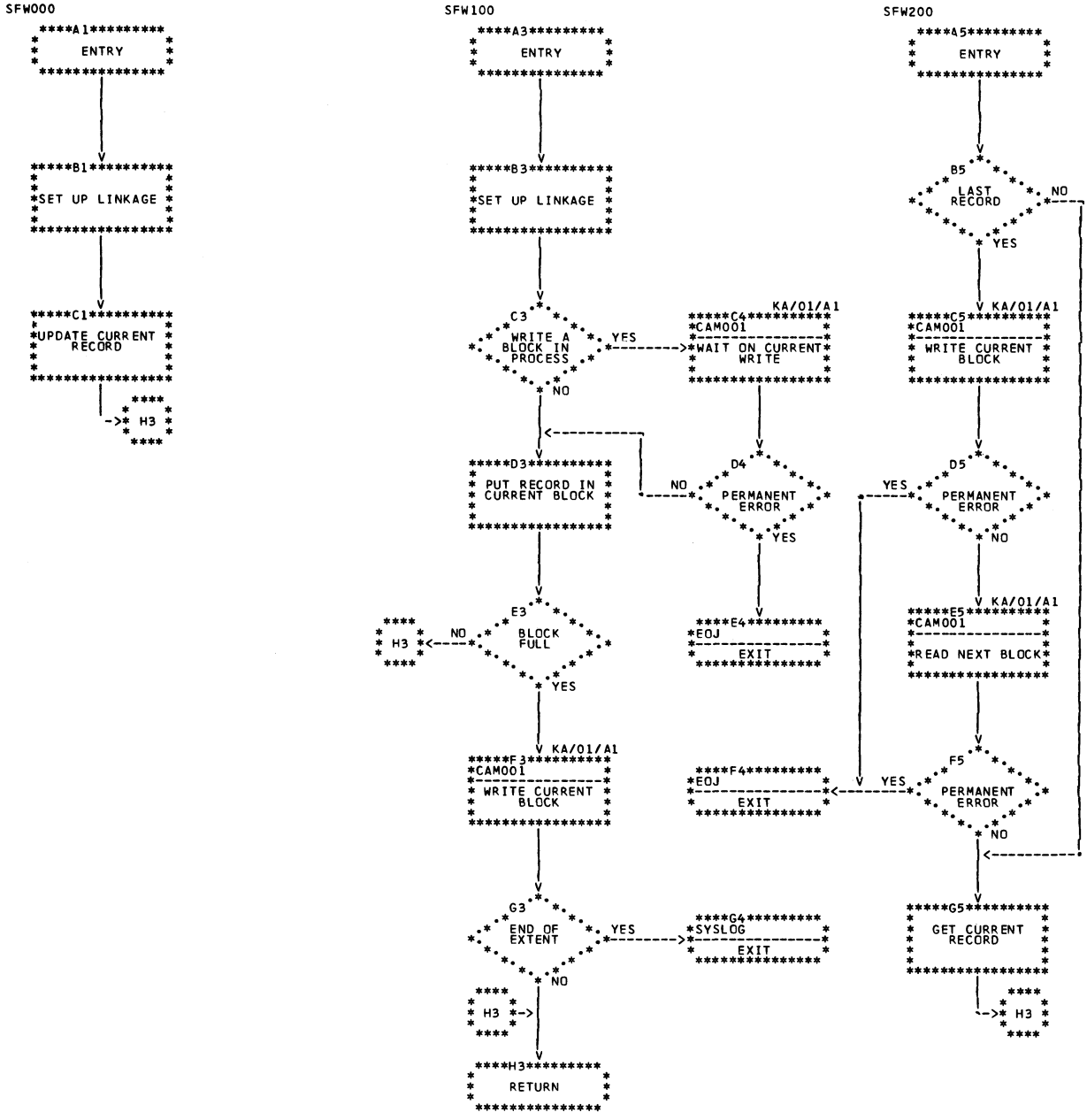


Chart DD. Disk Data Management Interfaces (SFW000)

MODULE $ASSS0—SYMBOL SUBSTITUTION (PHASE 22)

Main Storage Map: Figure 8.

Entry Point: $ASSS1 — entered from Phase 20 via FETCH. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Supervisor and IOS Logic Manual,* SY21-0033.)

Charts: EA-EE

RESIDENT SUPERVISOR
$ASROT Communication Vector Assembler Common Disk Data Management
PHASE 22 CODE (Module $ASSS0) SSI000 SSP000 SSL000 SST000 SSE000 SSU000 SSS000 SSW000
ESL Table
SYMBOL TABLE
DISK BUFFERS

Figure 8. Phase 22 Main Storage Map

Functions:

- Initializes Phase 22 for processing (SSI000—Chart EA).
- Controls Phase 22 processing (SSP000—Chart EB).
- Performs functions associated with termination of Phase 22 (SSL000—Chart EC).
- Tests each term record for an unresolved symbol. Resolves those symbols in the present symbol table (SST000—no chart; functionally the same as SFT000 routine in Phase 21).
- Builds ESL (External Symbol List) table entries for valid EXTRN and ENTRY statements (SSE000—Chart ED).
- Adds EXTRN and ENTRY entries to ESL (SSU000—no chart).
- Searches symbol table for a given symbol (SSS000—no chart; functionally the same as SBR000 routine in Phase 20).
- Provides interface for disk data management (SSW000—Chart EE).

Input:

- Symbol table in main storage.
- Associated symbol table parameters in ASMCOM.
- Intermediate text in $WORK2.

Output:

- All term symbols that are defined in the current symbol table are updated with their values and attributes in the intermediate text.
- Symbol table entries — if a cross-reference has been requested, all symbol table entries are added to the end of $WORK2.
- ESL table passed to Phase 29.

Exits:

- Normal — control is passed to Phase 29, ESL output.
- Error — control is returned to scheduler via Halt/Syslog or EOJ transients. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Supervisor and IOS Logic Manual,* SY21-0033.)

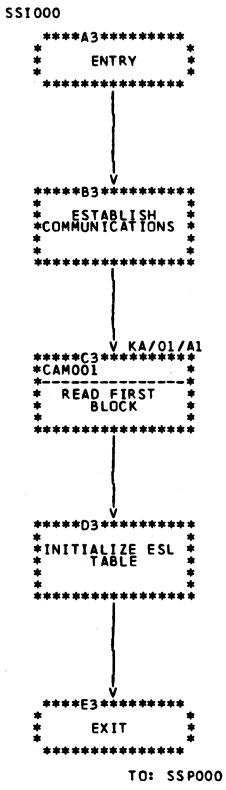


Chart EA. Initialization Routine (SSI000)

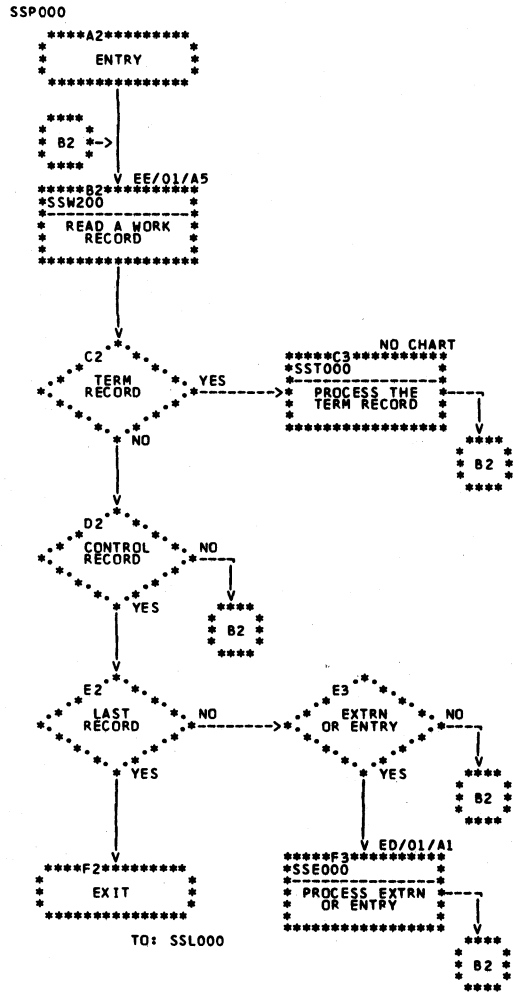


Chart EB. Main Control Routine (SSP000)

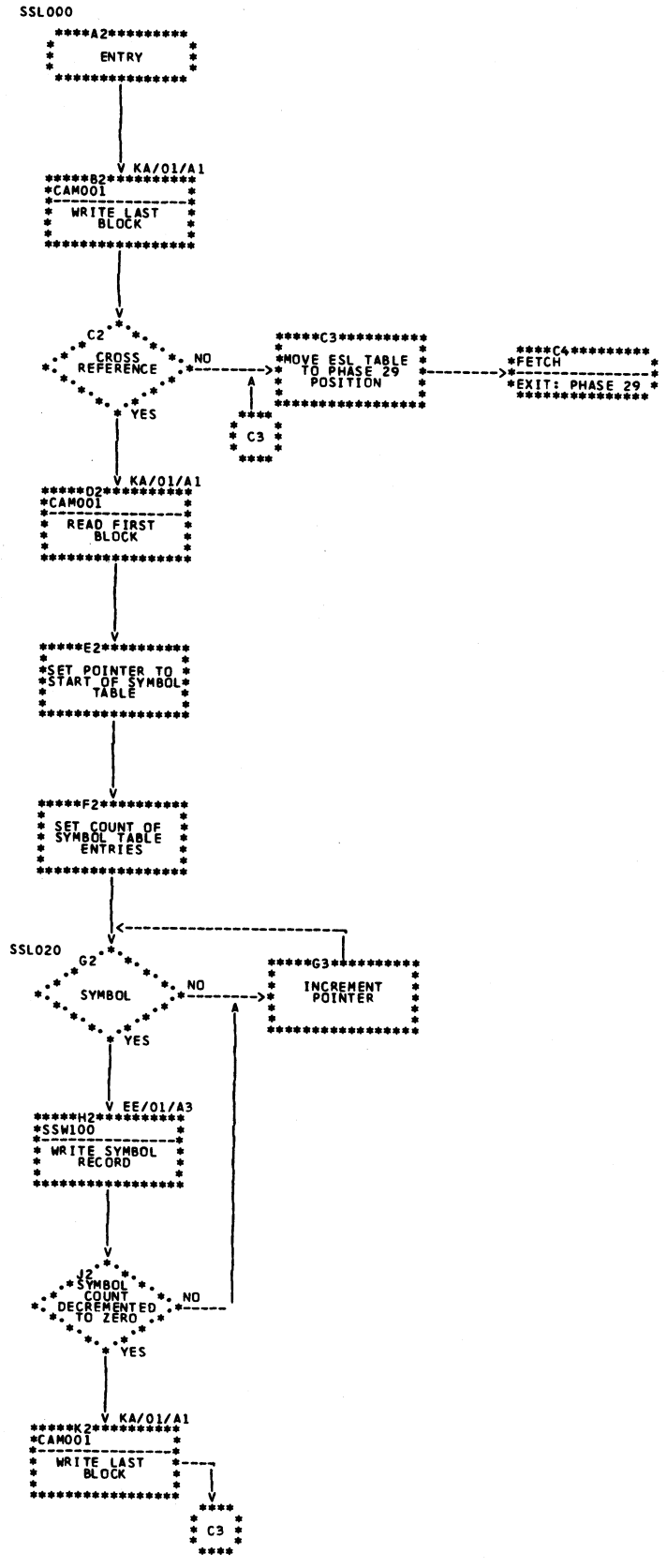


Chart EC. Termination Routine (SSL000)

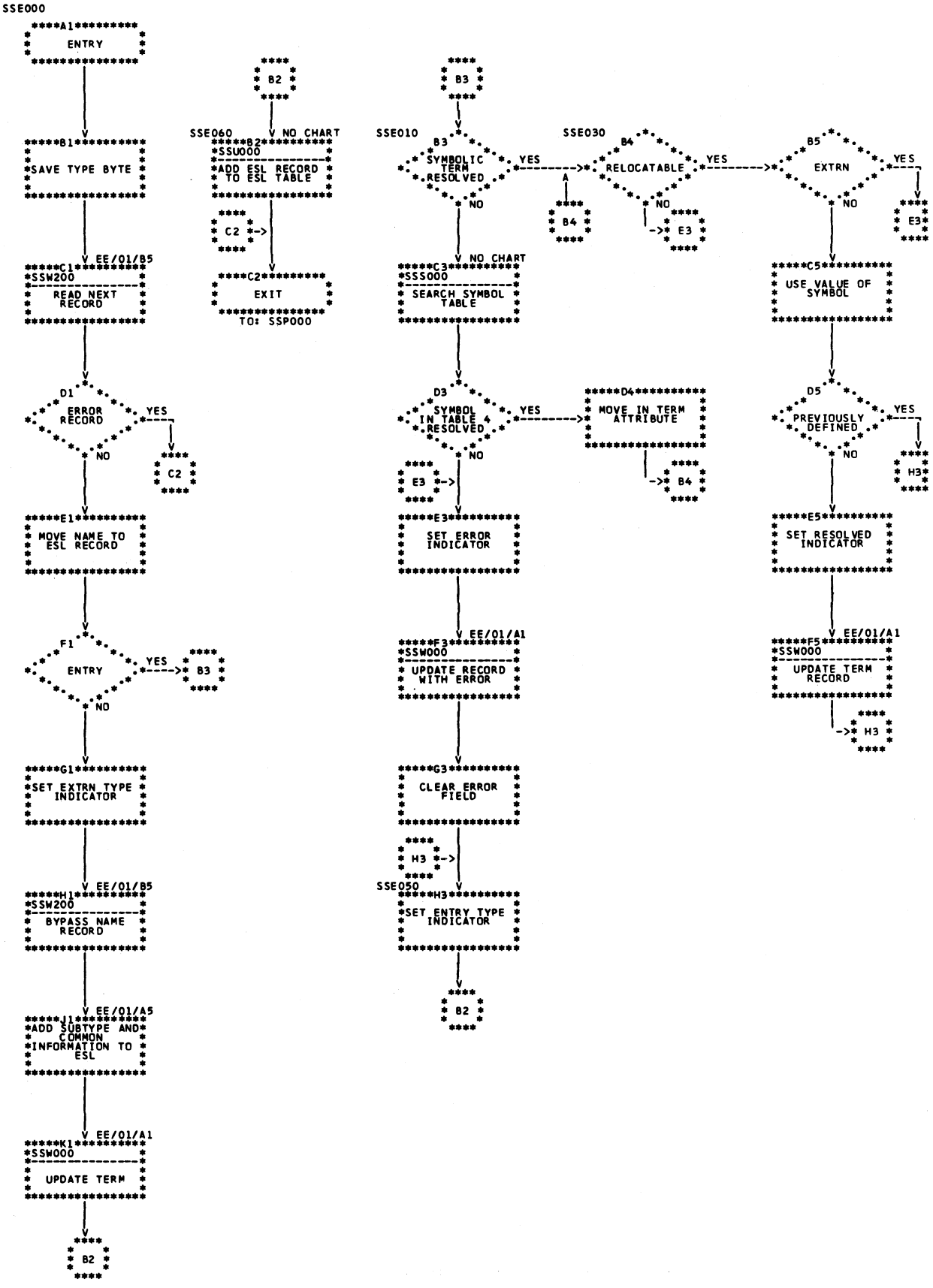


Chart ED. EXTRN/ENTRY Processing Routine (SSE000)

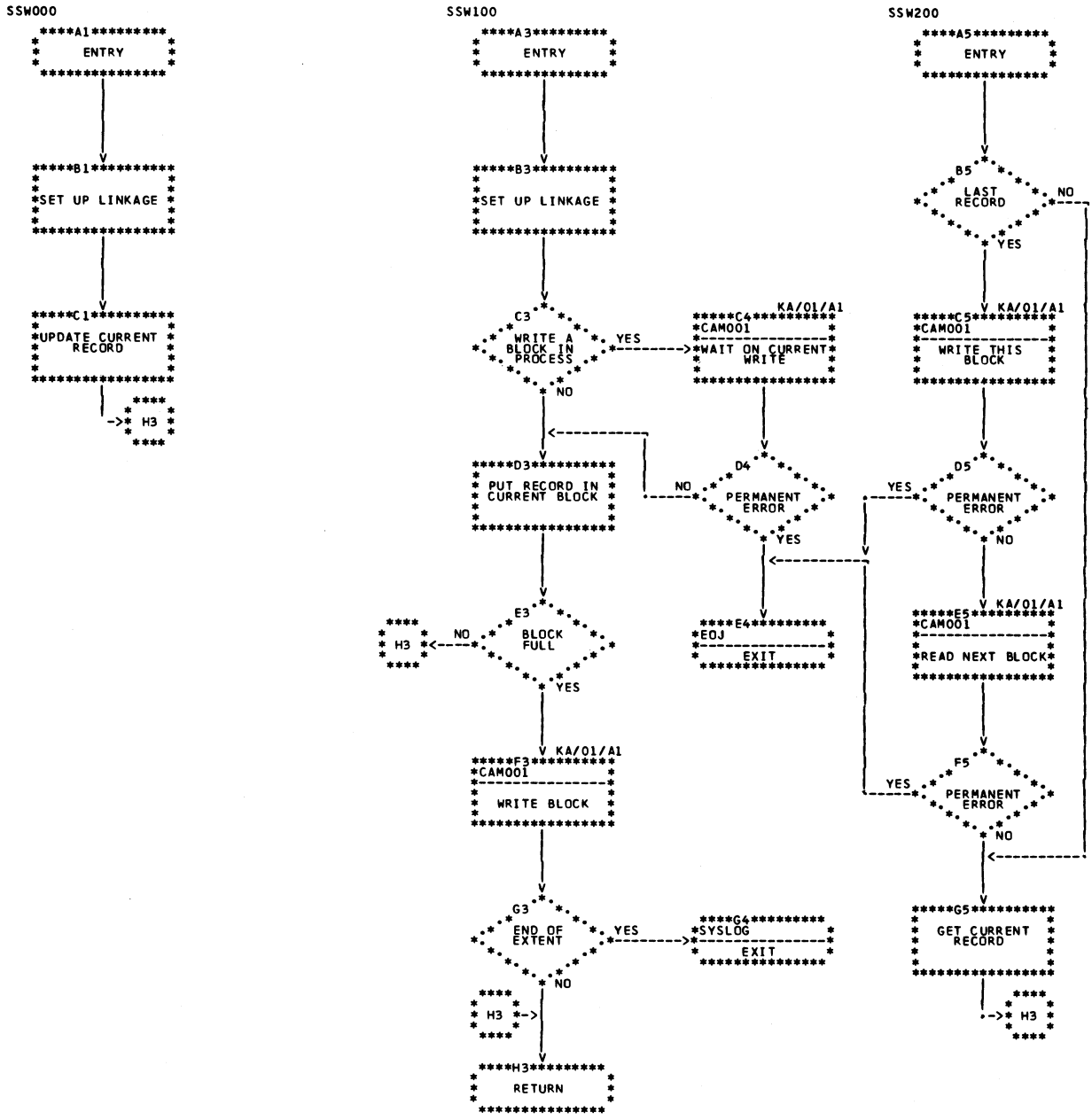


Chart EE. Disk Data Management Interfaces (SSW000)

MODULE $ASPE0 – ESL OUTPUT (PHASE 29)

Main Storage Map: Figure 9.

Entry Point: $ASPE1 – entered from Phase 22 via FETCH. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Supervisor and IOS Logic Manual,* SY21-0033.)

RESIDENT SUPERVISOR
$ASROT Communications Vector Assembler Common Disk Data Management
$ASPRC (see *Section 5* for description)
Printer Data Management *
MFCU Data Management *
PHASE 29 CODE (Module $ASPE0) PEI000 PEO000 PEP000 PES000 PEL000 PEG000 PER000 PEW000 PEX000 Working Storage
ESL TABLE AND DISK BUFFERS

* Model 10 only.

Figure 9. Phase 29 Main Storage Map

Charts: FA-FI.

Functions:

- Initializes Phase 29 (PEI000–Chart FA).
- Prints OPTIONS record(s) and a summary of selected options (PEO000–Chart FB).
- Punches an absolute loader (PEP000–Chart FC).
- Sorts, builds, and puts out ESL (External Symbol List) object records (PEP000–Chart FC).
- Prints the ESL table (PES000–Chart FD).
- Performs functions associated with termination of Phase 29 (PEL000–Chart FE).
- Prints the printer listing header and counts pages (PEG000–Chart FF).
- Provides disk data management interface (PEW000–Chart FG).
- Provides print module interface (PEX000–Chart FH) and punch module interface (PER000–Chart FI).

Input:

- ESL table.
- Parameters in ASMCOM.
  1. ESLCNT – contains a count of ESL table entries.
  2. OPTONS – contains LIST/NOLIST,DECK/NODECK, REL/NOREL, and OBJ/NOOBJ options.
  3. OPTCNT – count of control statement records.
  4. MODULE – module name.
  5. TITLEN – TITLE name.
  6. ESLTBA – ESL table address.

Output:

- If DECK and NOREL are specified on the OPTIONS statement, an absolute loader is punched.
- If DECK and/or OBJ is specified, the ESL object records are placed in the $WORK file.
- If LIST is specified, the ESL table is sorted, then printed.
- The following information is passed to Phase 30 via ASMCOM:

  1. PAGCNT – current printer page size.
  2. LPSIZE – current printer line size.
  3. Other miscellaneous areas of ASMCOM are initialized.

- The listing header in the printer communication module $ASPRC is initialized for the source/object listing done in Phase 30.

Exits:

- Normal – control is passed to Phase 30 (module $ASPS0).
- Error – control is returned to scheduler via Halt/Syslog or EOJ transients. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Scheduler Logic Manual,* SY21-0035.)

Working Storage Block:

- Contains all working storage and data constants required for Phase 29 processing which are not in the communications modules ($ASROT, $ASPRC).
- Contains equates for ASMCOM references.

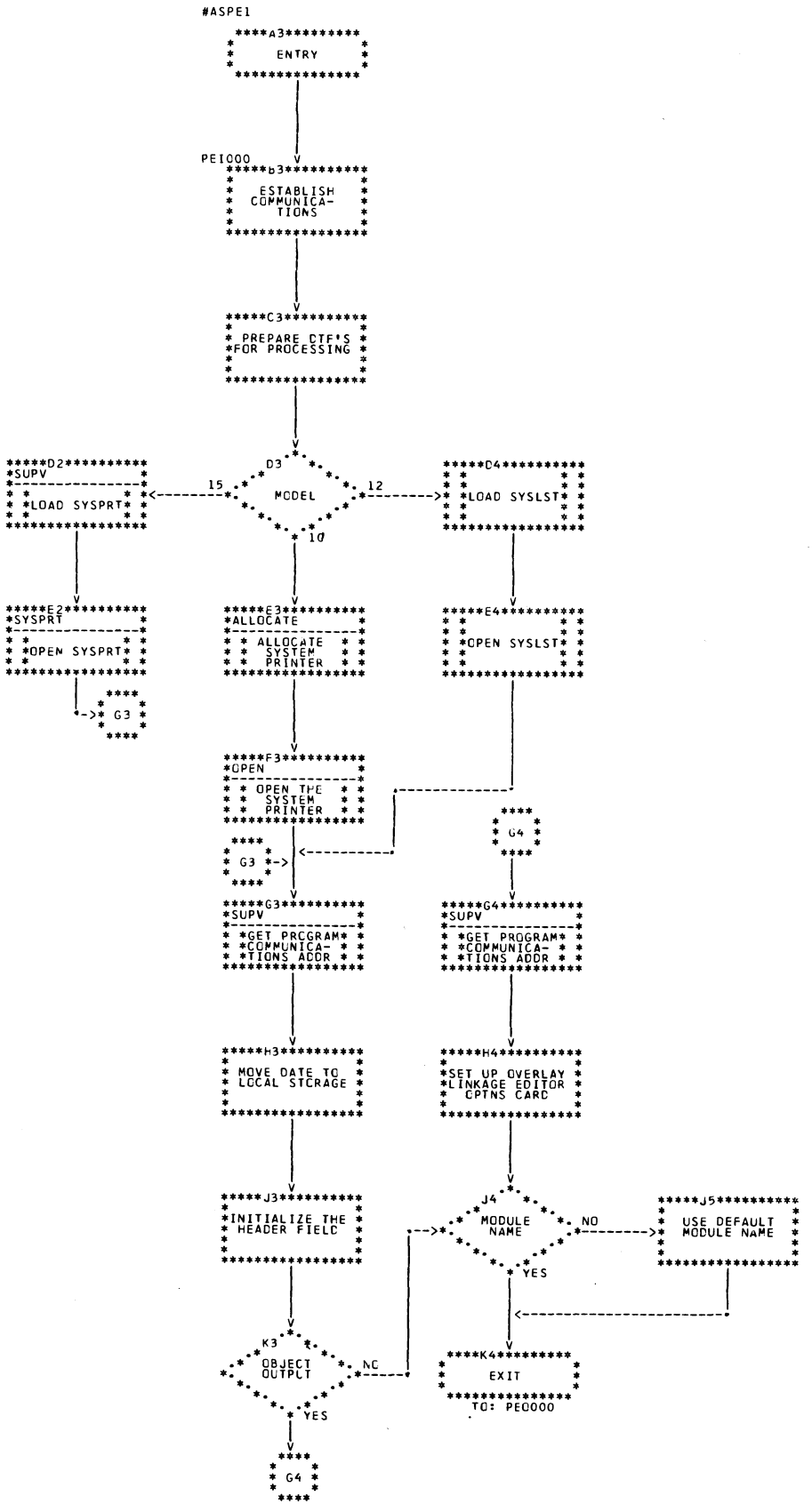


Chart FA. Initialization Routine (PEI000)

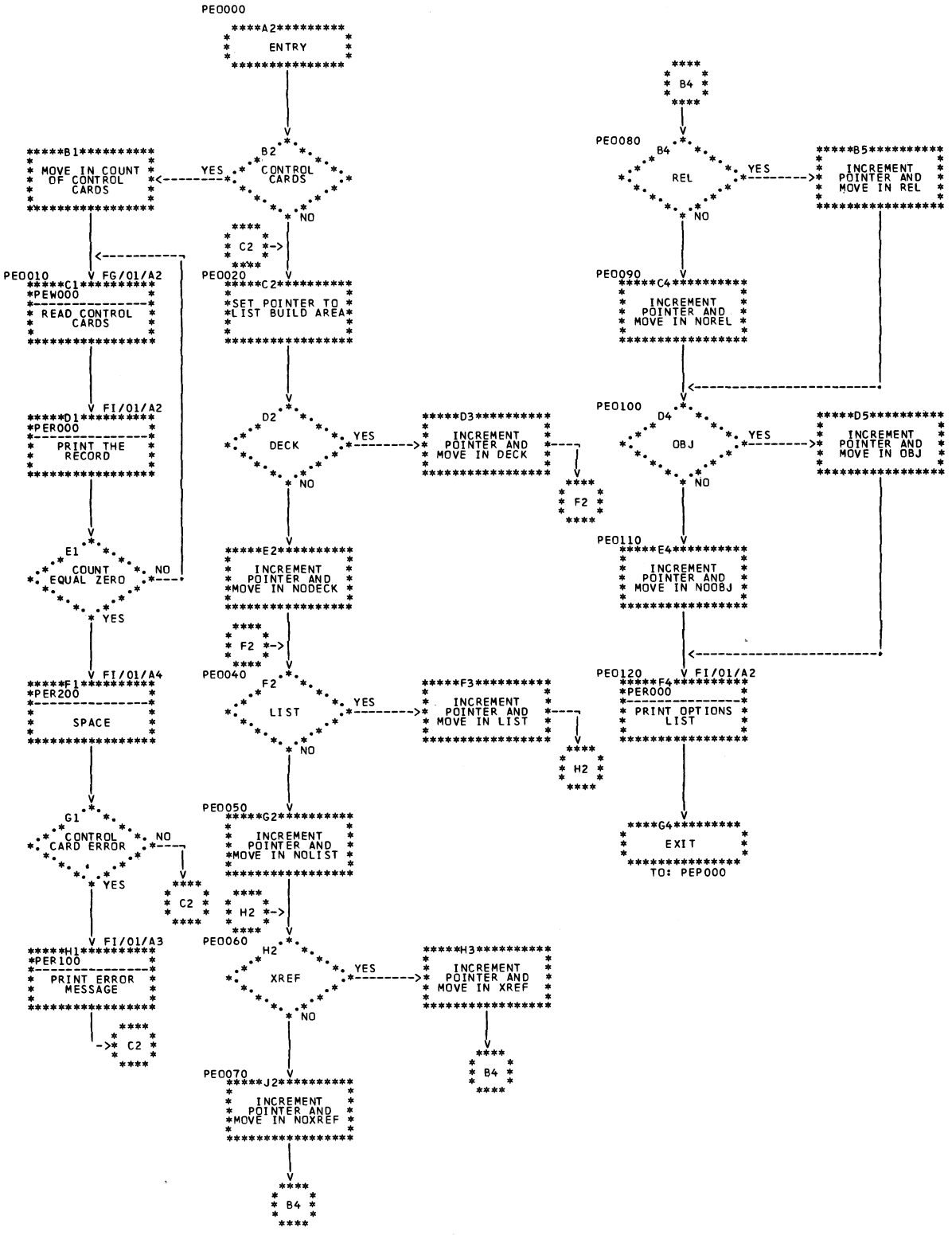


Chart FB. Options Processing Routine (PEO000)

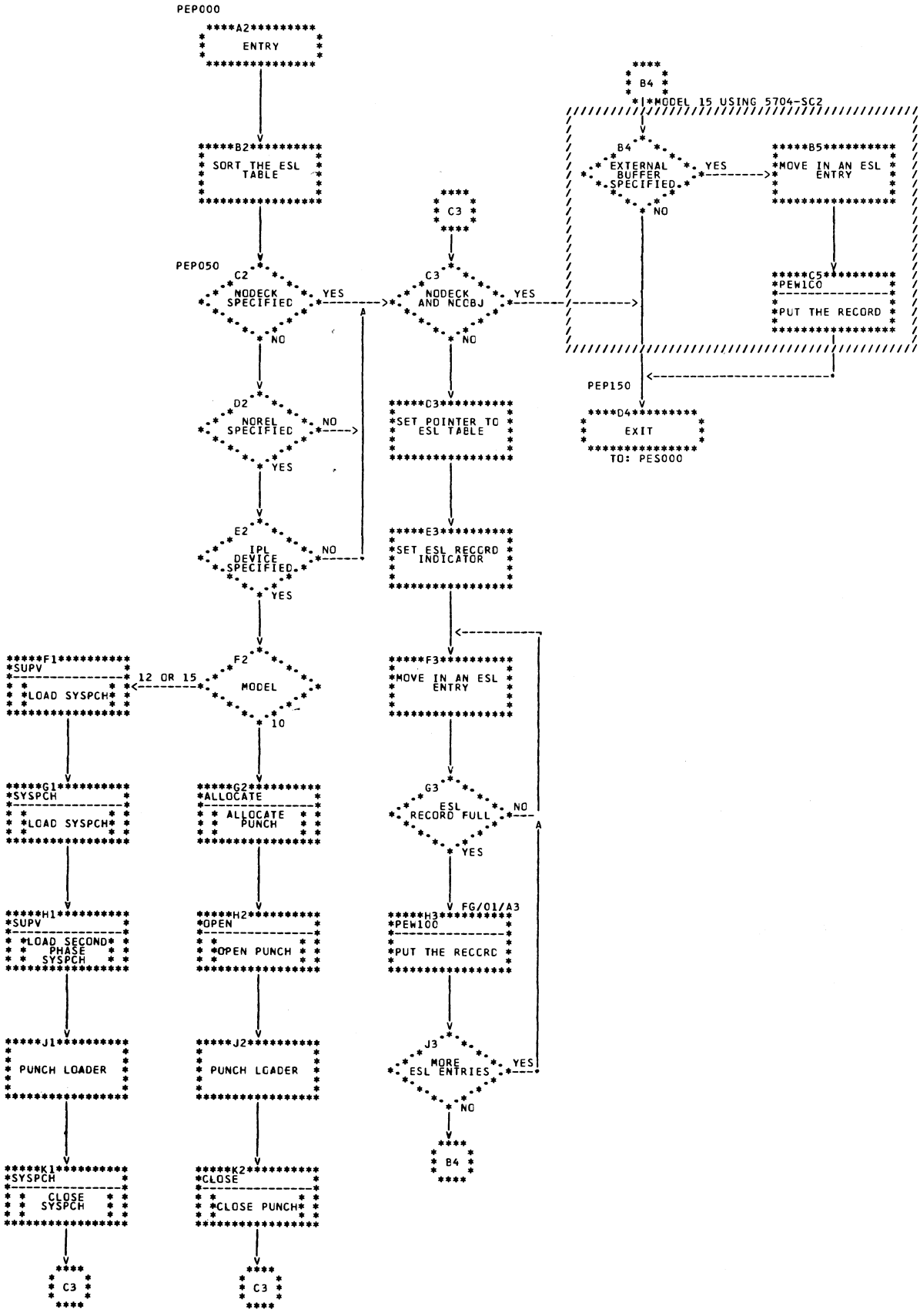


Chart FC. ESL Object Output Routine (PEP000)

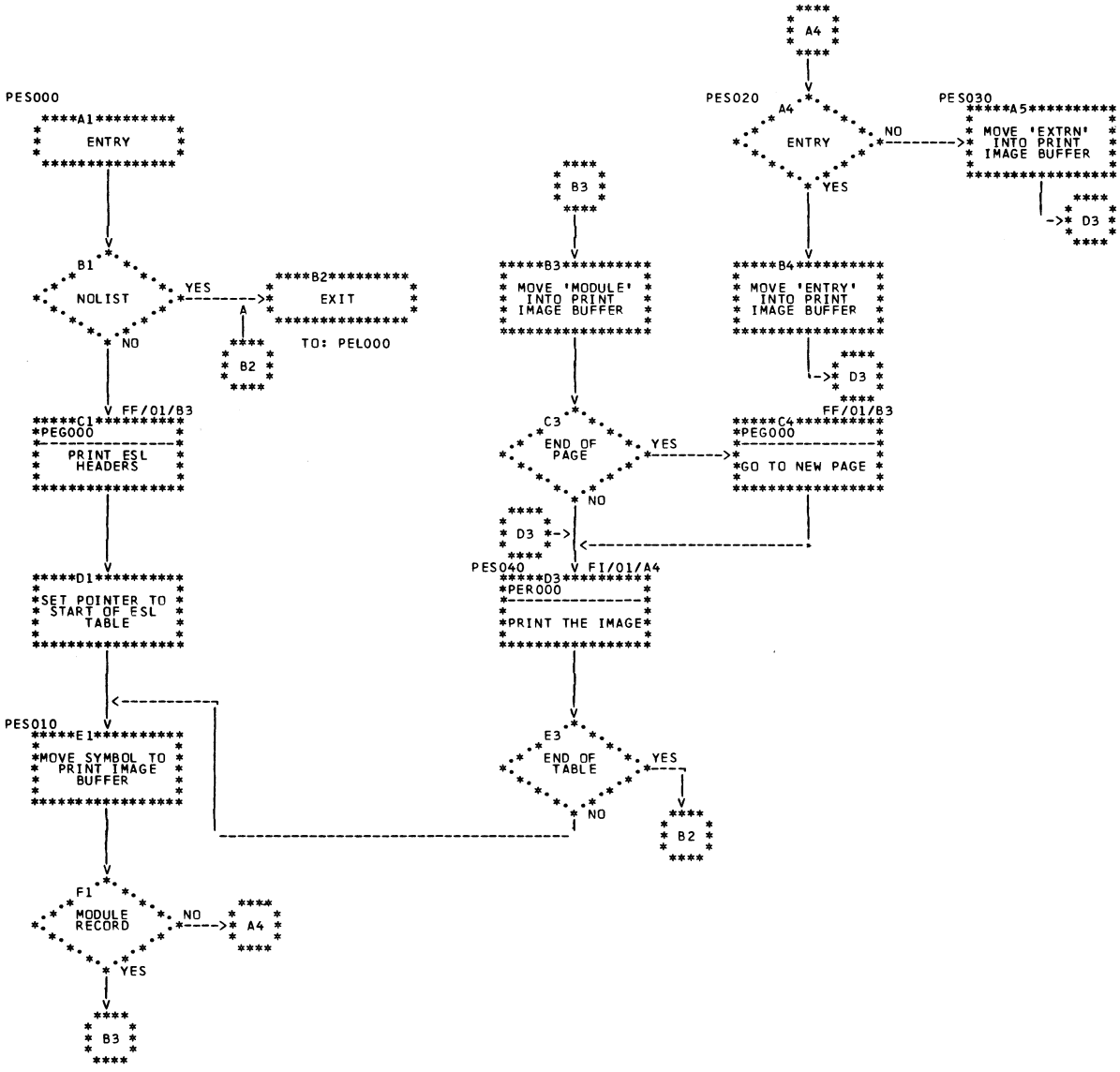


Chart FD. ESL Table Print Routine (PES000)

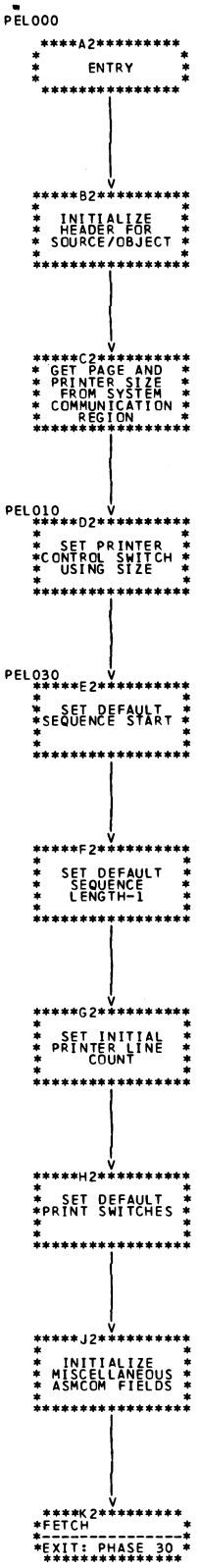


Chart FE. Termination Routine (PEL000)

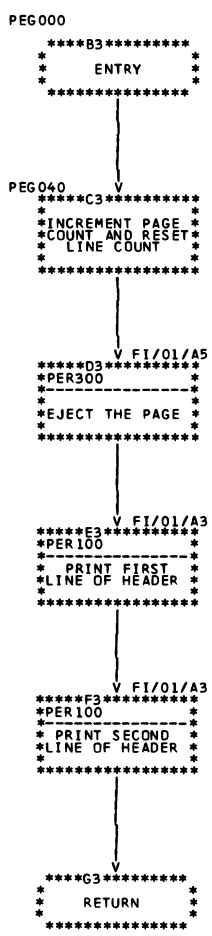


Chart FF. Page Header Control Routine (PEG000)

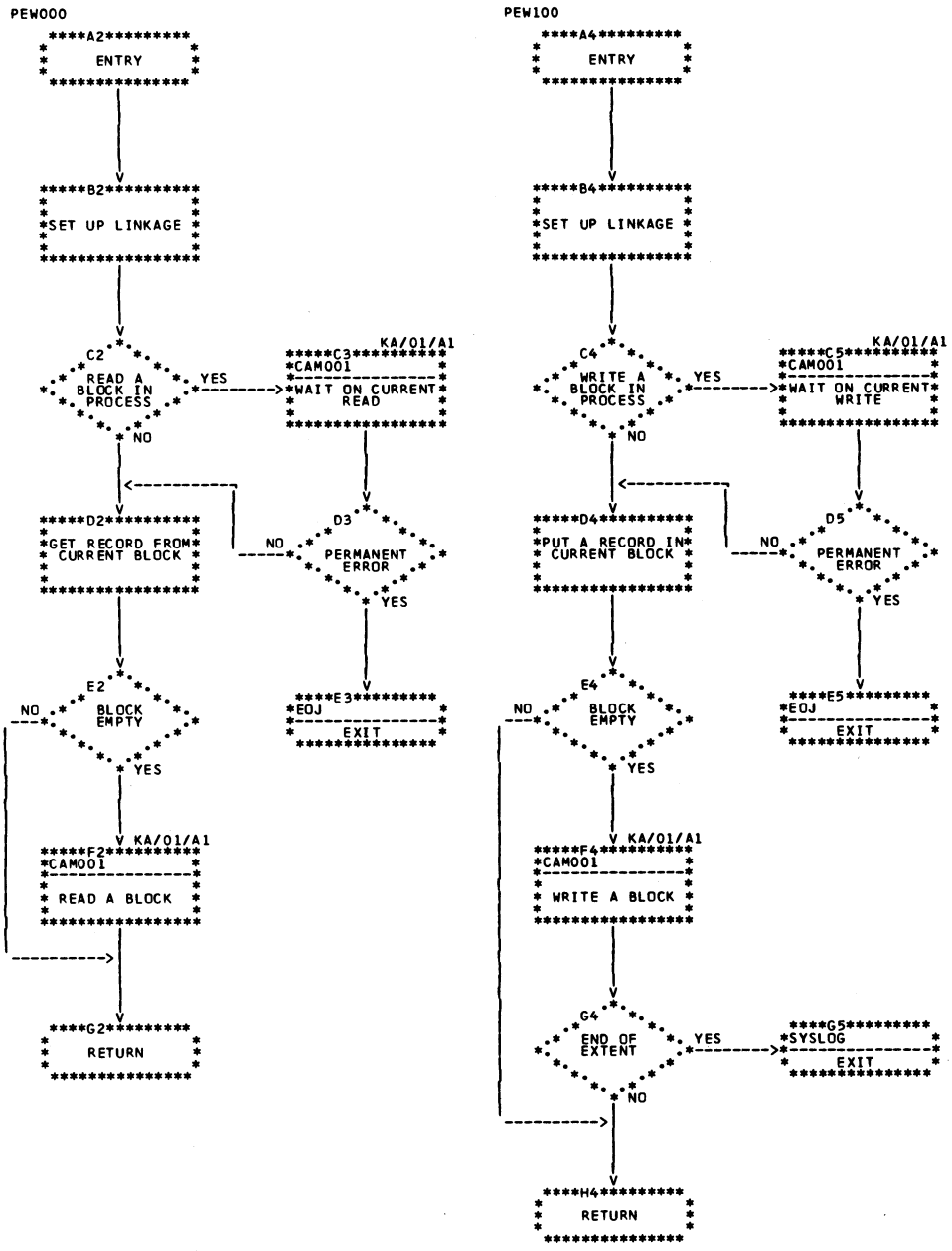


Chart FG. Disk Data Management Interfaces (PEW000)

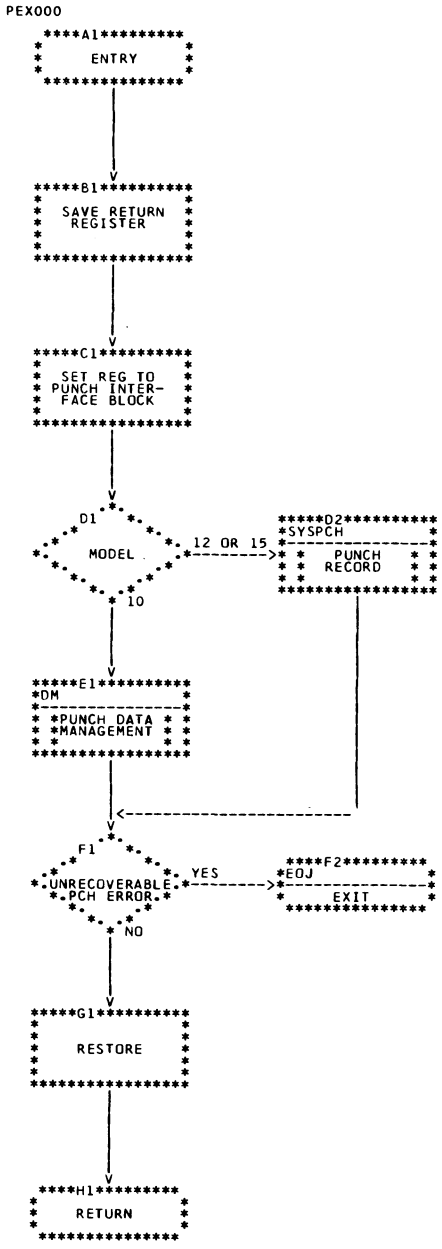


Chart FH. Punch Data Management Interface (PEX000)

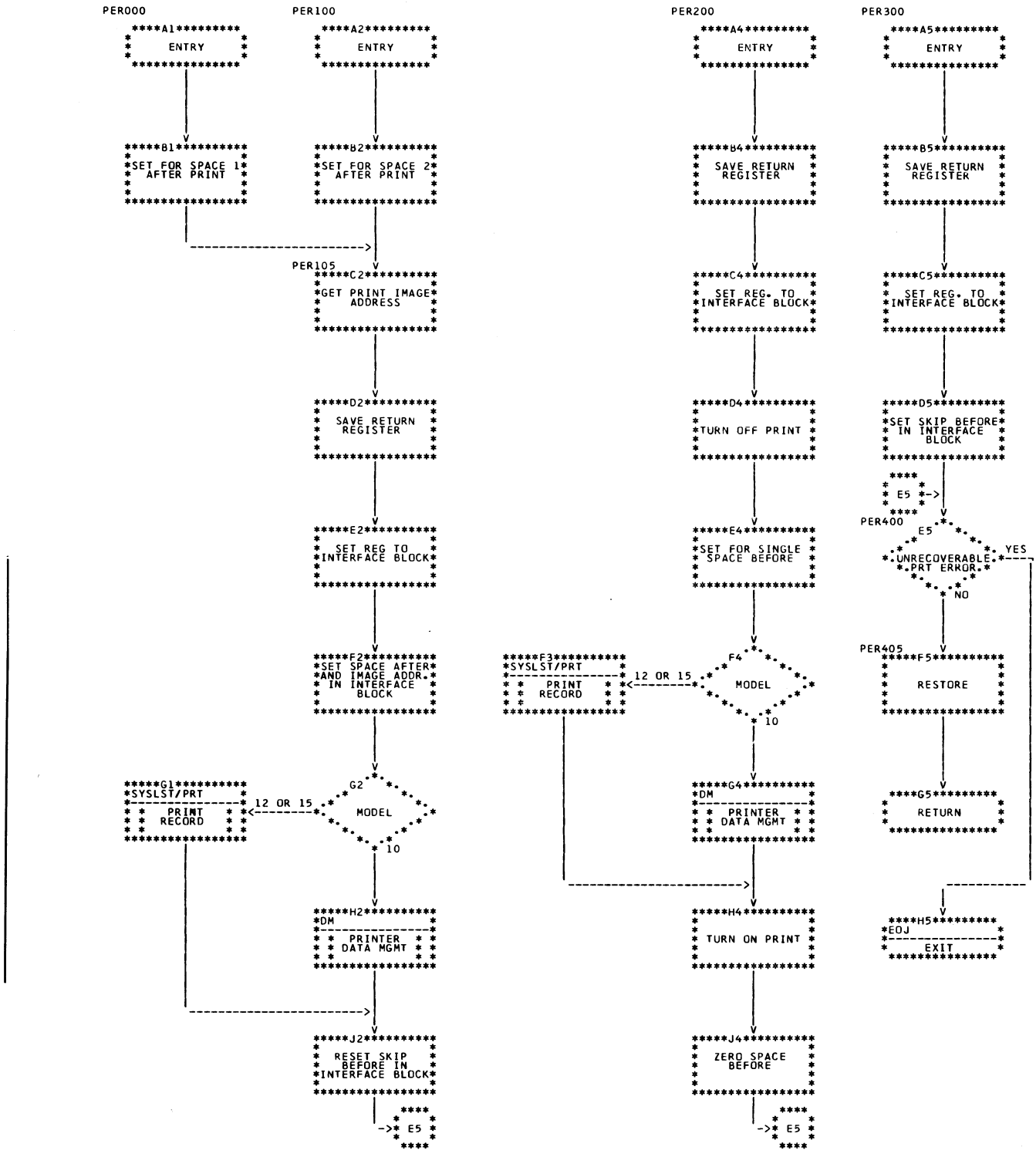


Chart FI. Print Data Management Interface (PER000)

MODULE $ASPS0—SOURCE/OBJECT OUTPUT (PHASE 30)

Main Storage Map: Figure 10.

Entry Point: $ASPS1 — entered from Phase 29 via FETCH. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Supervisor and IOS Logic Manual,* SY21-0033.)

Charts: GA-GK.

Functions:

- Initializes Phase 30 for processing (PSI000—Chart GA).
- Controls Phase 30 processing of intermediate text file until end of data or end of file is reached (PSC000—Chart GB).
- Searches for control records; puts error number in listing if any errors are found (PSK000—no chart).
- Performs tasks associated with Phase 30 termination (PST000—Chart GC).
- Puts object code in $WORK (PSH000—Chart GD).
- Converts binary data to hexadecimal representation (PSX000—no chart).
- Evaluates the expressions in the operand field of the current source record (PSE000—no chart; functionally the same as SBV000 routine in Phase 20).
- Initializes the Print Edit Routine at the start of the Output Phase and for every ISEQ start request (PSY000—Chart GF).
- Performs all editing of the print image required by the various sized line printers — 132 column printer, 120 column printer, and 96 column printer (PSZ000—Chart GC).
- Controls page size of printed output (PSP000—no chart).
- Counts pages (increment page count parameter) and prints current header (PSG000—no chart).
- Performs binary multiplication operations (PSM000—no chart; functionally the same as SBX000 routine in Phase 20).
- Provides linkage to overlay routines, $ASOV1 and $ASOV2 (PSLNK1, PSLNK2—Chart GE) — Model 10 Disk System only.
- Provides interface for printer data management (PSR000 —no chart; functionally the same as PER000 routine in Phase 29).
- Provides interface for disk data management (PSW000—Chart GK).

Functions: Module $ASOV1 (Part of Module $ASPS0, for Model 15)

- Performs Phase 30 processing for machine instructions, translating them into object code (PSS000—Chart GH).
- Processes Group 1 assembler instructions, DROP, ENTRY, EQU, ORG, START, USING, (PSB000—Chart GI).

Functions: Module $ASOV2 (Part of Module $ASPS0, for Model 15)

- Processes Group 2 assembler instructions, DC, DS, EJECT, END, EXTRN, ISEQ, PRINT, SPACE, TITLE, (PSA000—Chart GJ).

Input:

- $SOURCE.
- Intermediate text in $WORK2.

  1. Fixed Length, 10-byte records.

  2. Four types of records.
     a. Control Record — one created for every source statement.
     b. Name Record — one for every valid name entry.
     c. Term Record — one for each term in a valid operand, and one or more, as required, to specify the operand field of the TITLE and DC assembler instructions.
     d. Error Record — created when there are one or more errors in a statement.

Output:

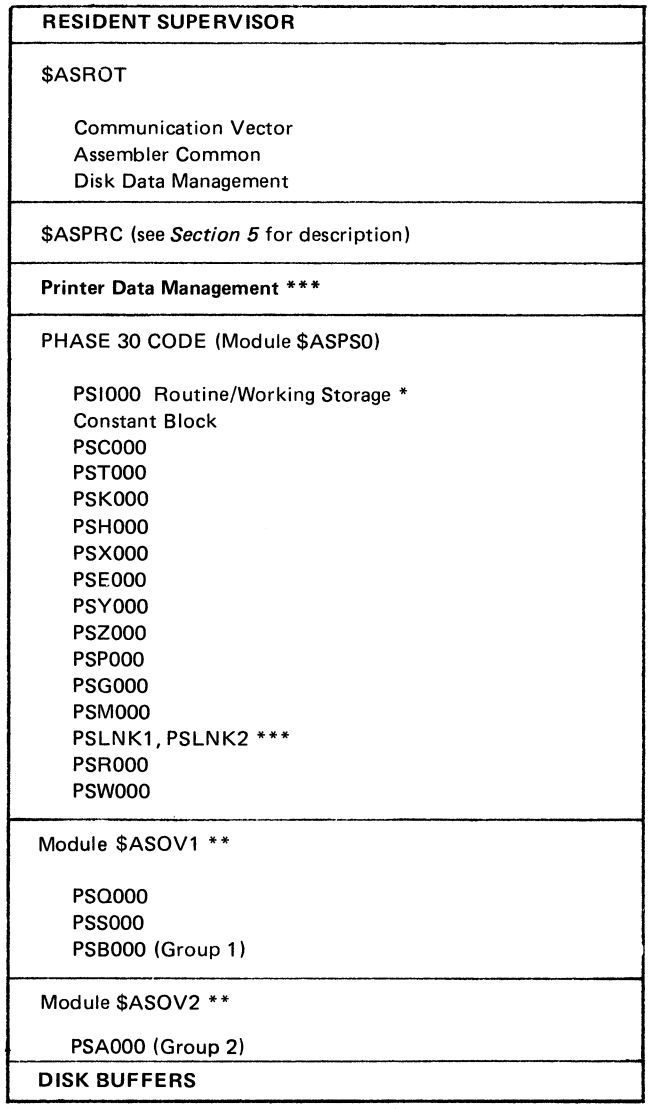
- Printer listing of the source statements and any object code generated by them if LIST is specified.
- Object deck is placed in $WORK.

Exits:

- Normal — control is passed to Phase 35 (module $ASPD0).
- Error — control is passed to scheduler via Halt/Syslog or EOJ transients. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Scheduler Logic Manual,* SY21-0035.)

Working Storage: All required Phase 30 working storage that is not in the communication modules ($ASROT, $ASPRC) resides in this block.

Constant Block: The Constant Block is loaded in the area immediately following the initialization routine (PSI000); it is moved into Assembler Common when the phase is initialized.



* PSI000 and its associated contant block occupy the same main storage as Phase 30 Working Storage.

** $ASOV1, $ASOV2 - The subroutines to perform machine and assembler instruction processing reside in these two modules. These modules are loaded into main storage at the end of $ASPS0. The amount of main storage available to Phase 30 determines whether they will both be resident or will both be loaded on call (load). The initialization routine (PSI000) determines the amount of main storage available and assigns load addresses and switches to control the loading. If there is sufficient main storage, $ASOV1 and $ASOV2 are loaded only once. On the Model 12 and Model 15, these routines are part of $ASPS0.

*** Model 10 only.

Figure 10. Phase 30 Main Storage Map

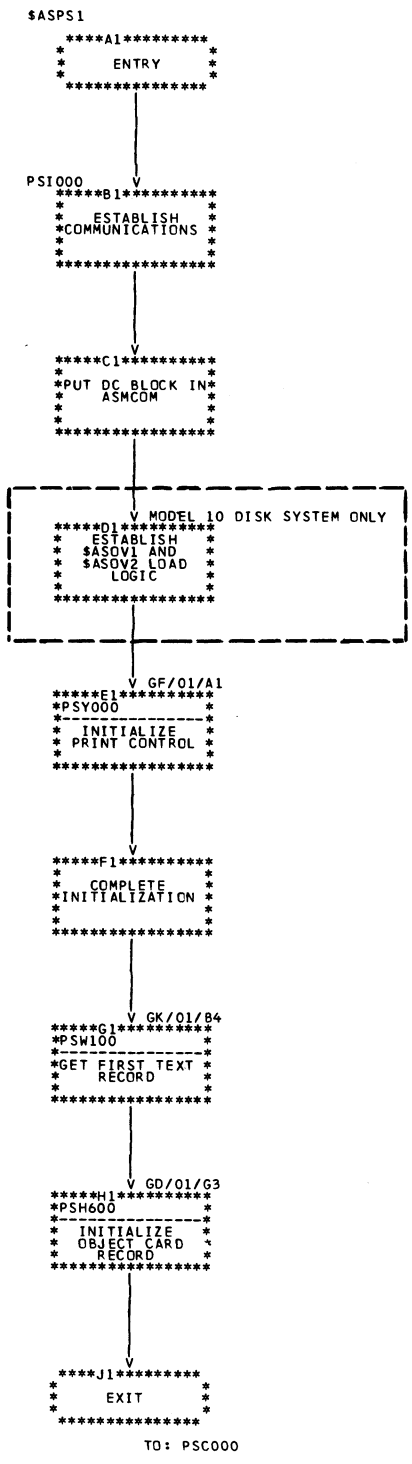


Chart GA. Initialization Routine (PSI000)

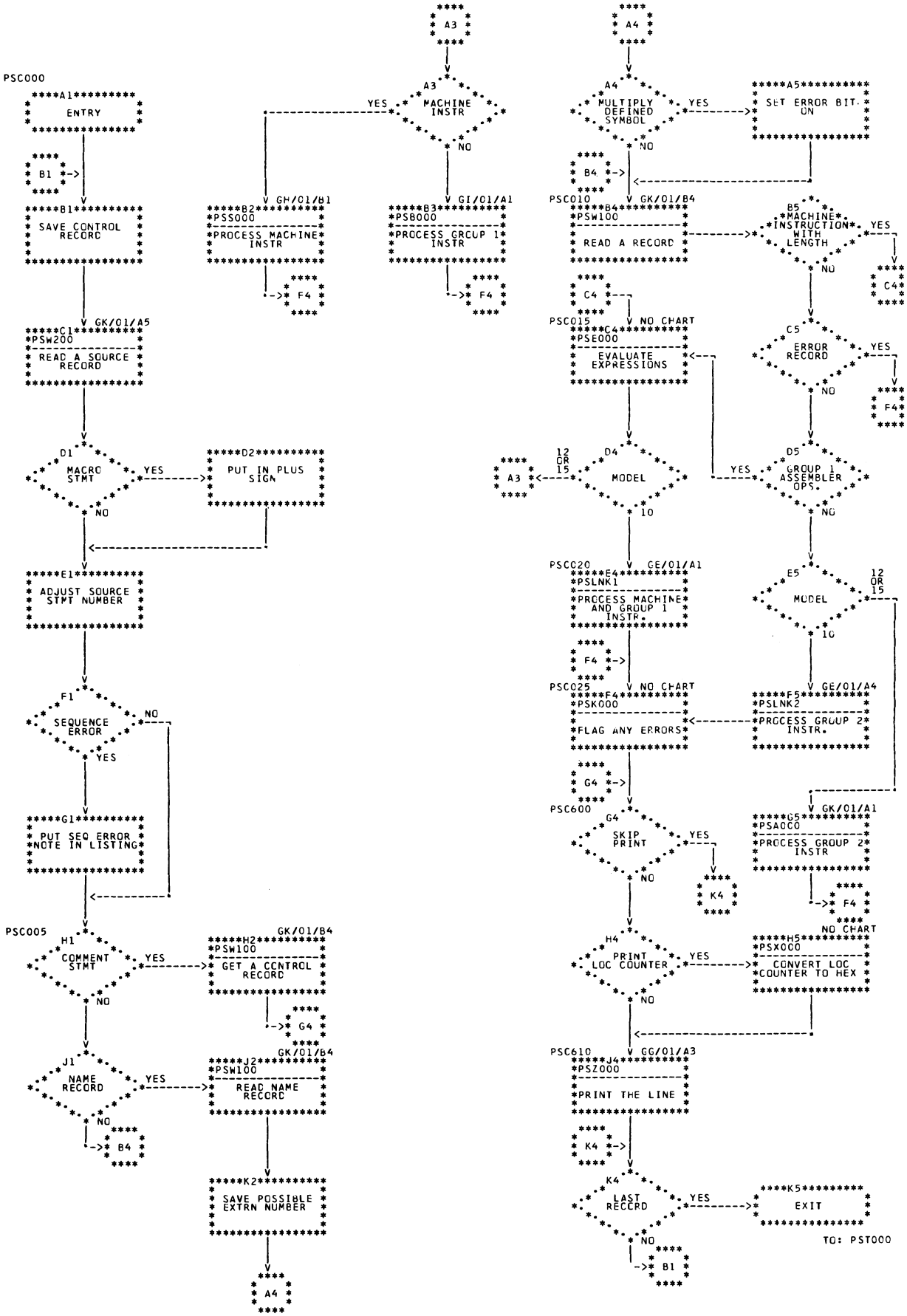


Chart GB. Main Control Routine (PSC000)

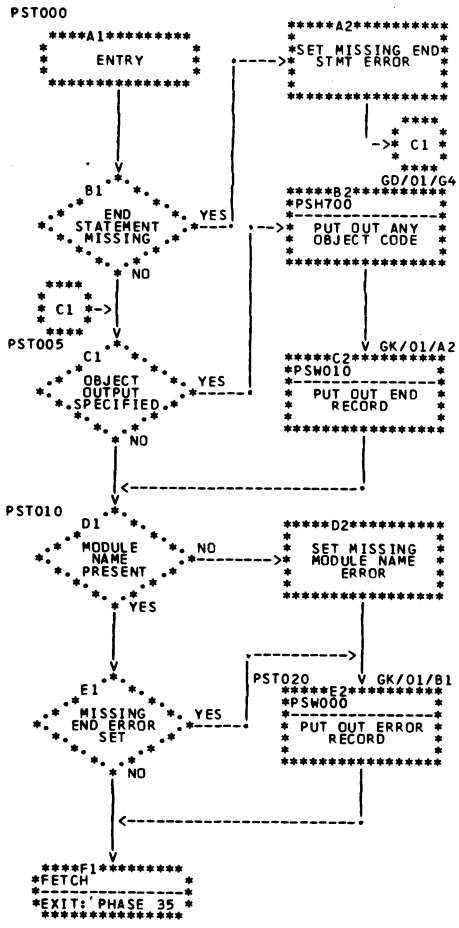


Chart GC. Termination Routine (PST000)

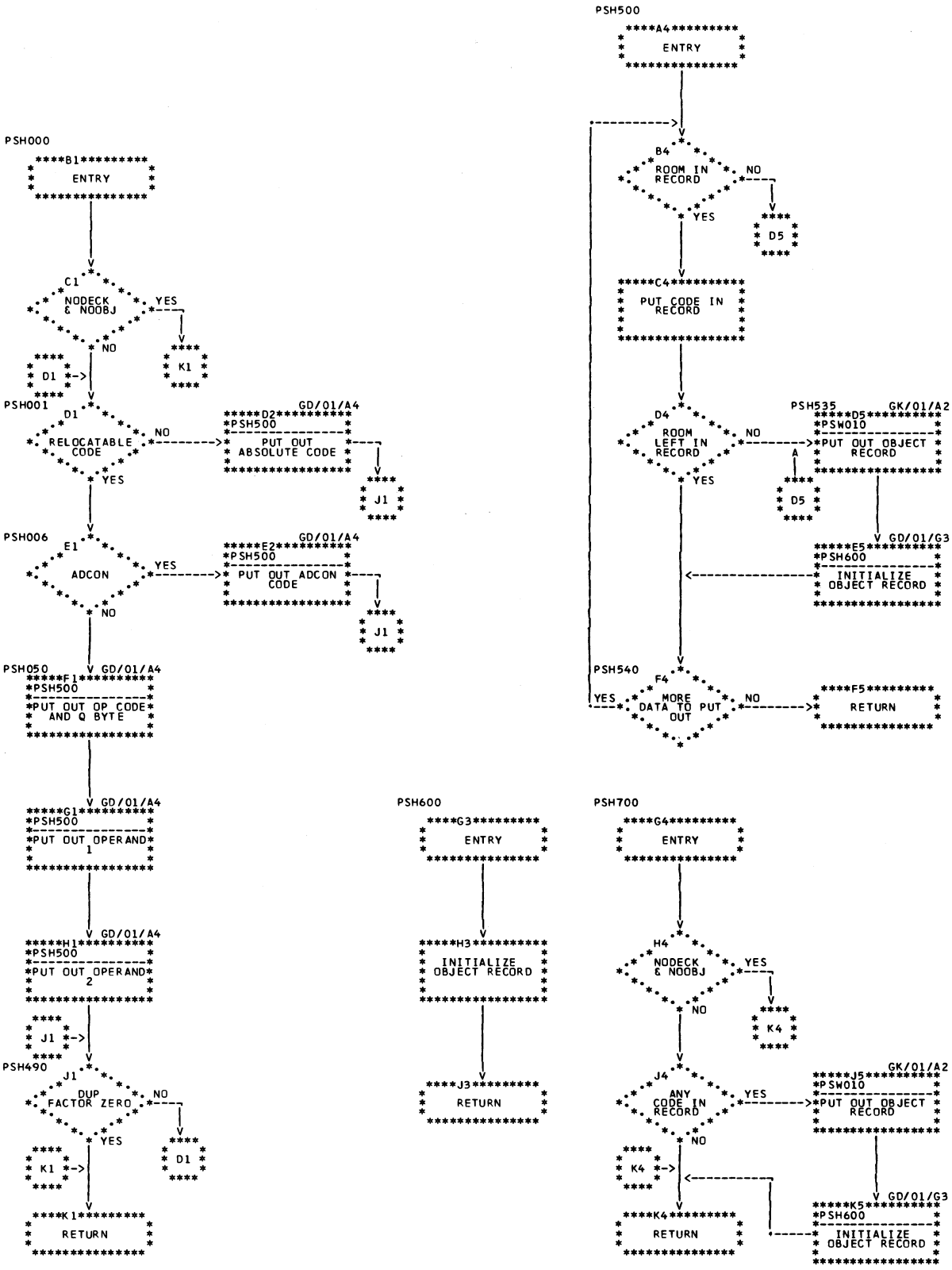


Chart GD. Put Object Code in Object File Routine (PSH000)

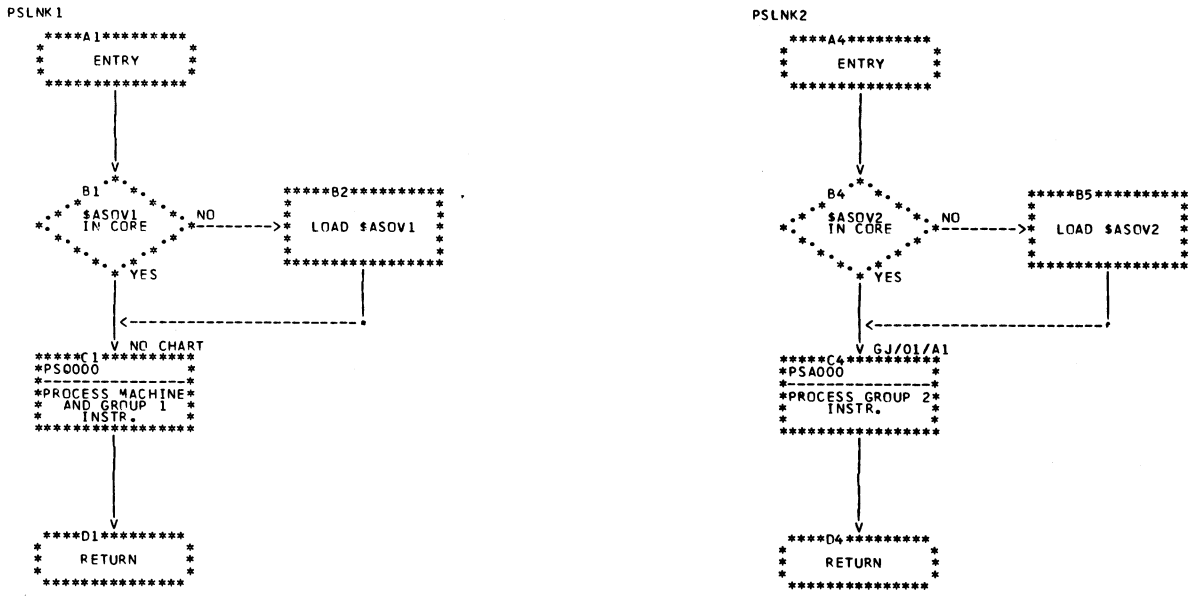


Chart GE. Linkage to Overlay Routines $ASOV1 and $ASOV2 (PSLNK1 and PSLNK2) — Model 10 only

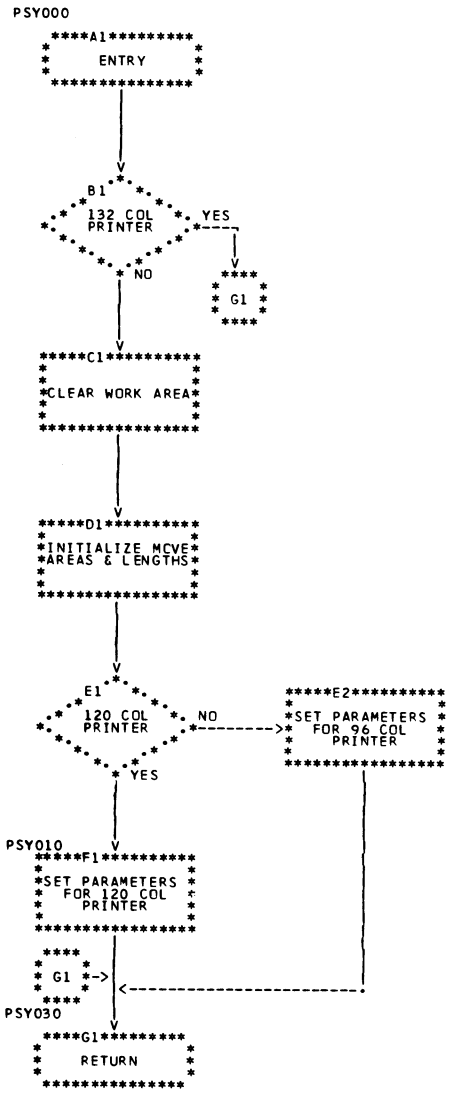
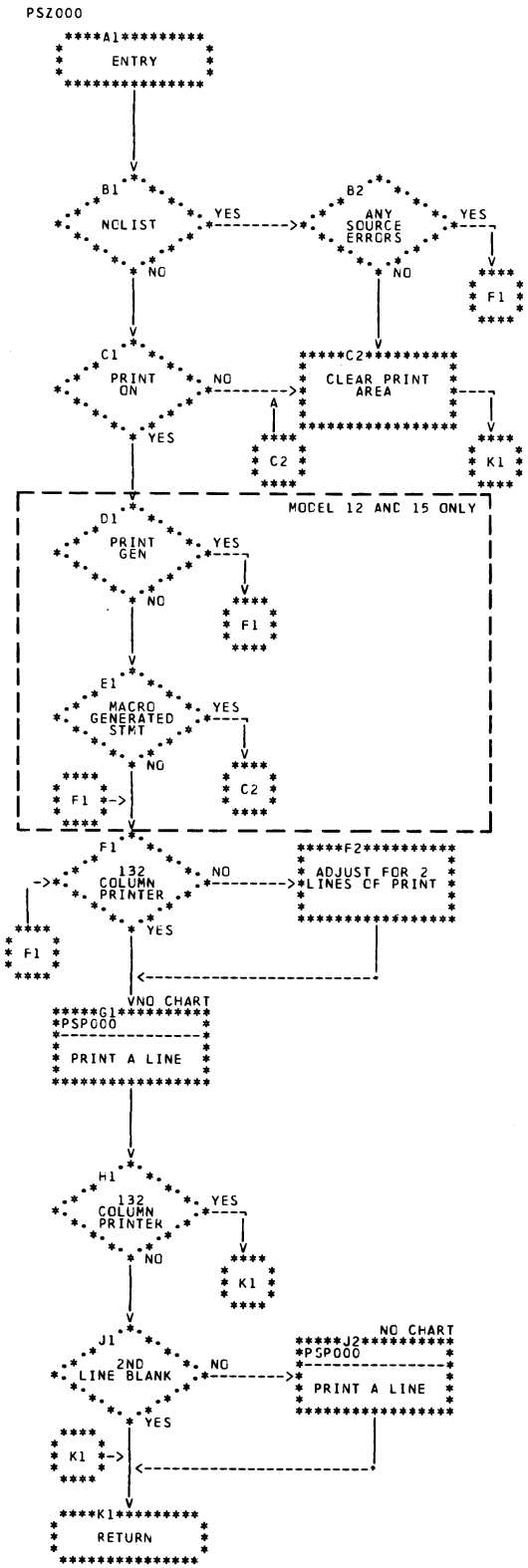


Chart GF. Print Image Edit Initialization Routine (PSY000)



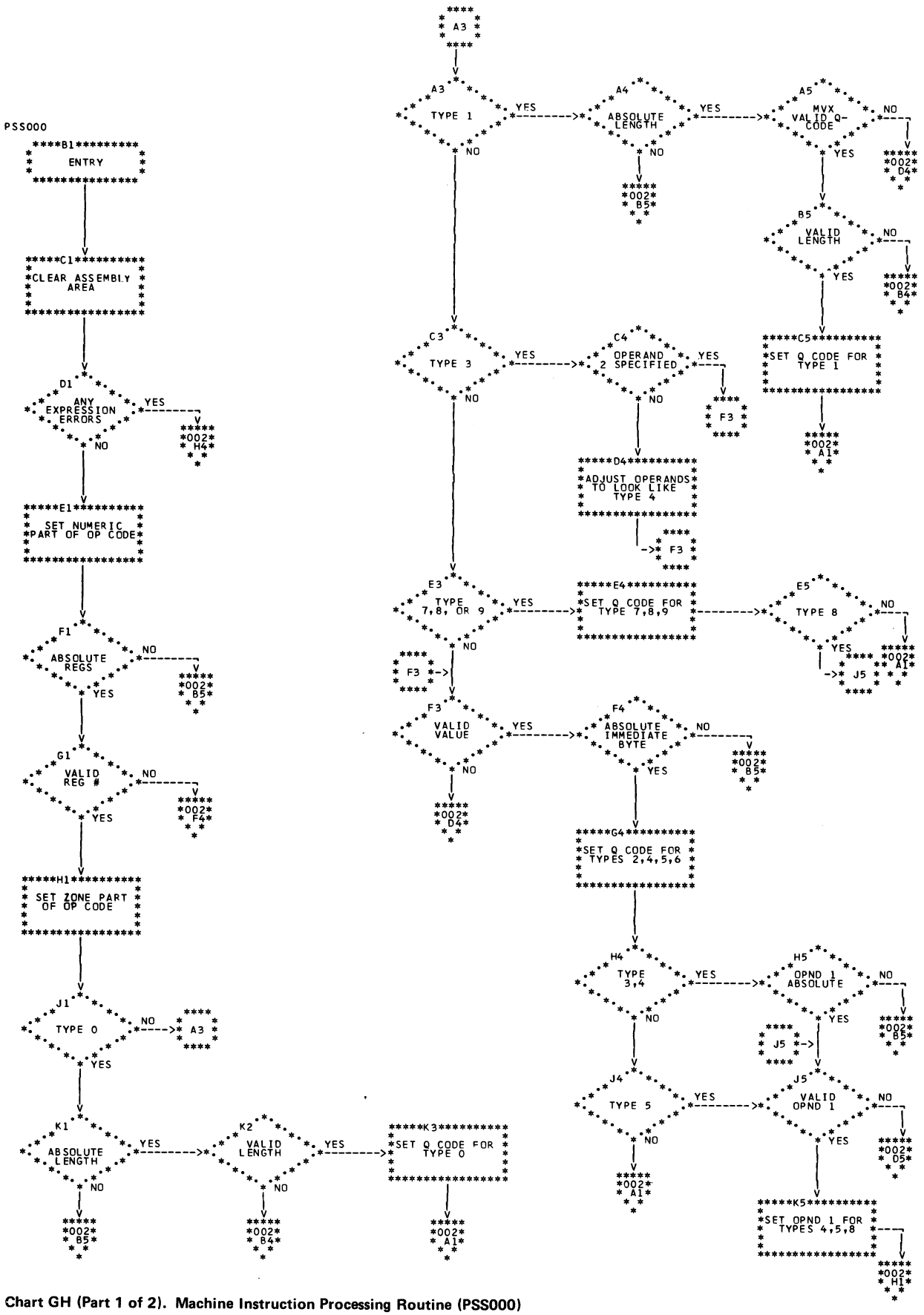


Chart GH (Part 1 of 2). Machine Instruction Processing Routine (PSS000)

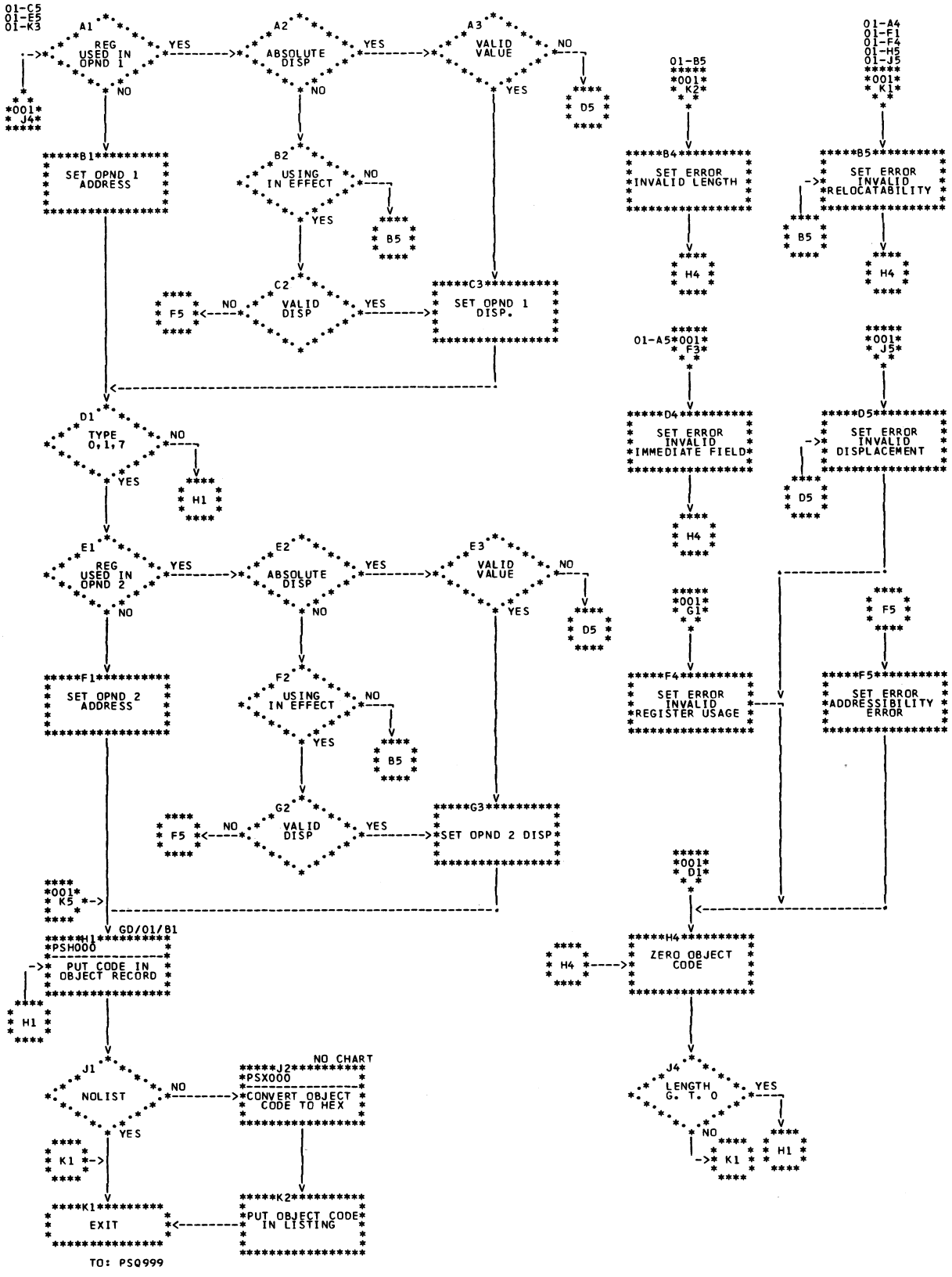


Chart GH (Part 2 of 2). Machine Instruction Processing Routine (PSS000)

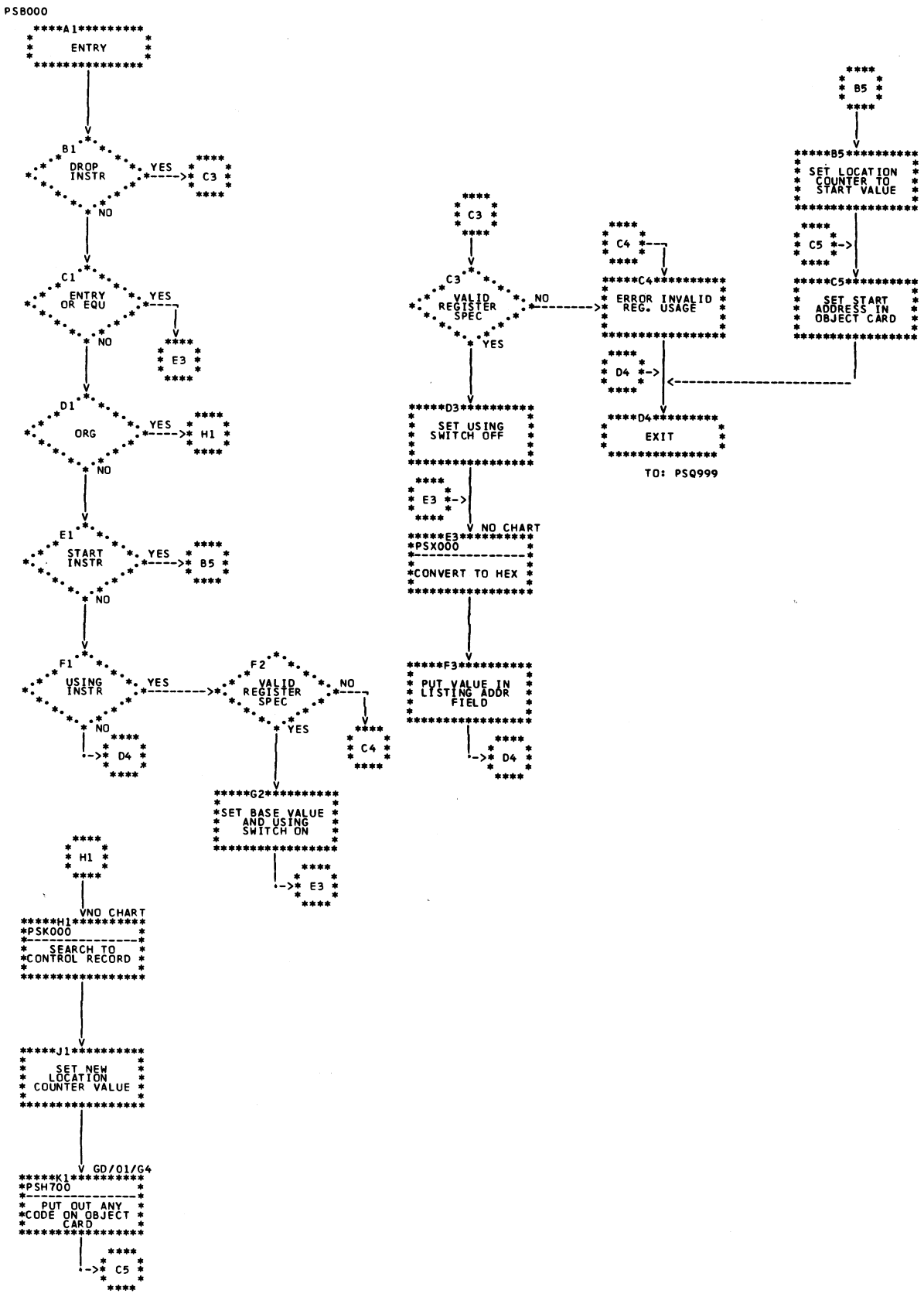


Chart GI. Group 1 Assembler Instruction Processing Routine (PSB000)

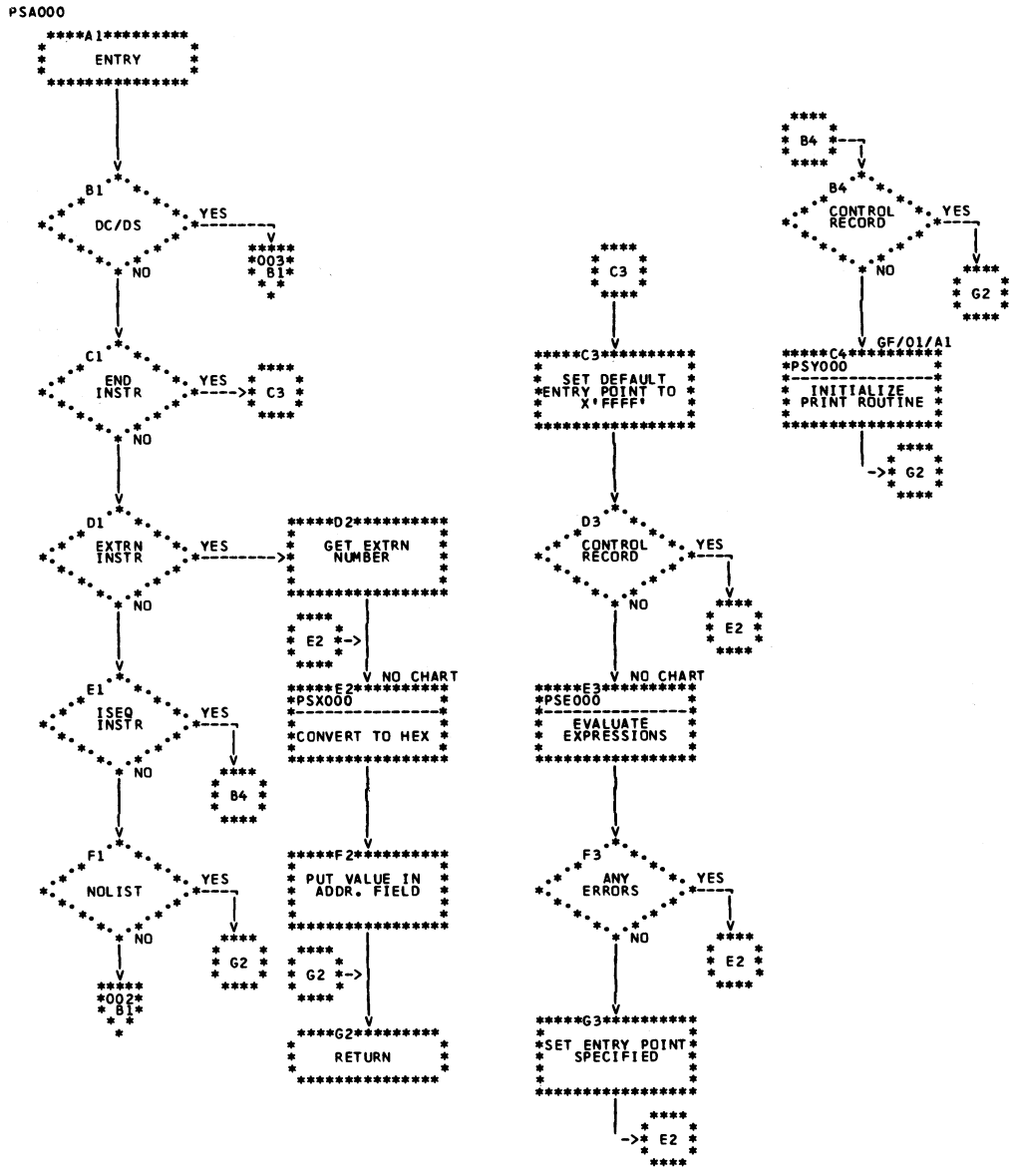


Chart GJ (Part 1 of 3). Group 2 Assembler Instruction Processing Routine (PSA000)

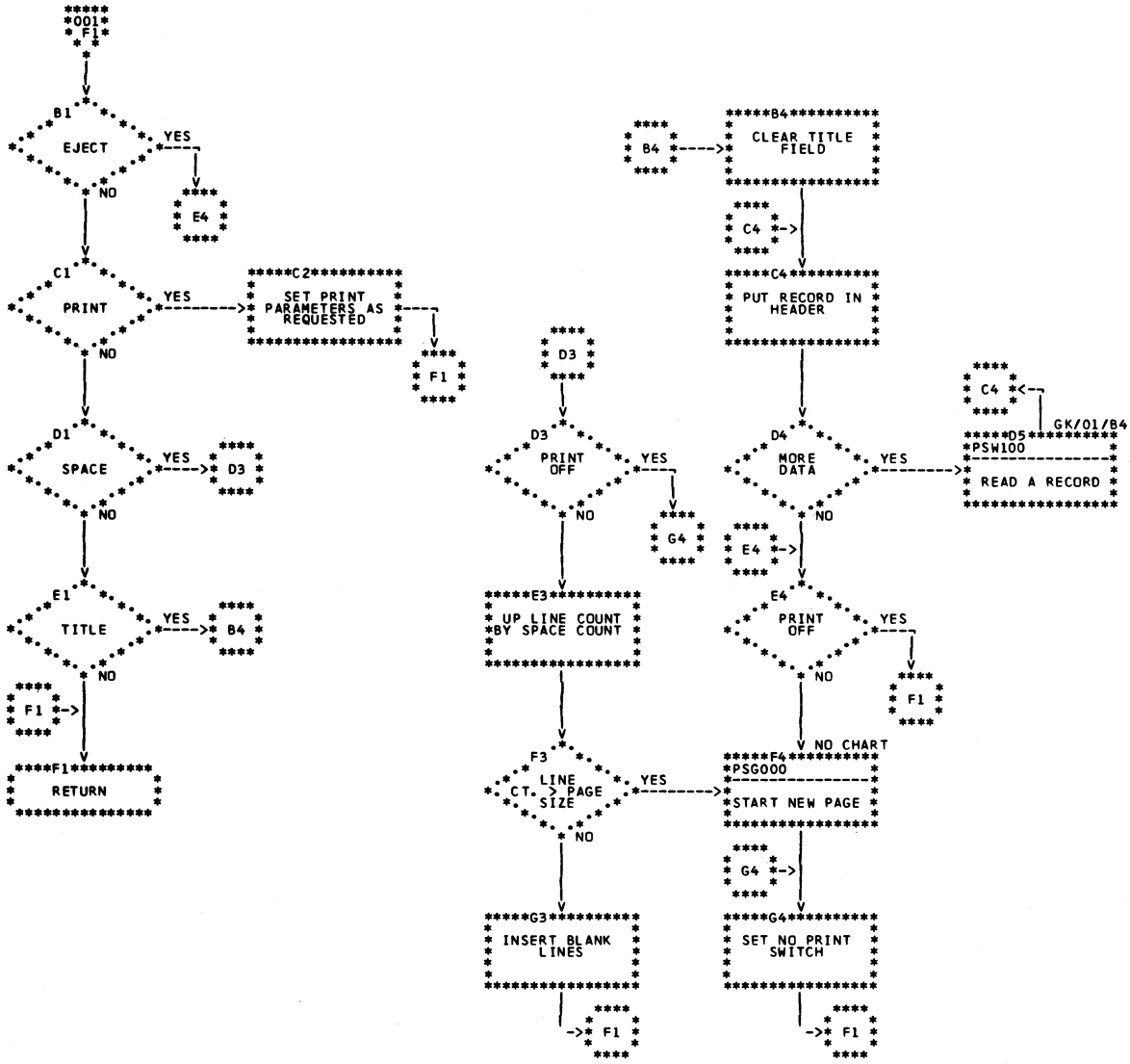


Chart GJ (Part 2 of 3). Group 2 Assembler Instruction Processing Routine (PSA000)

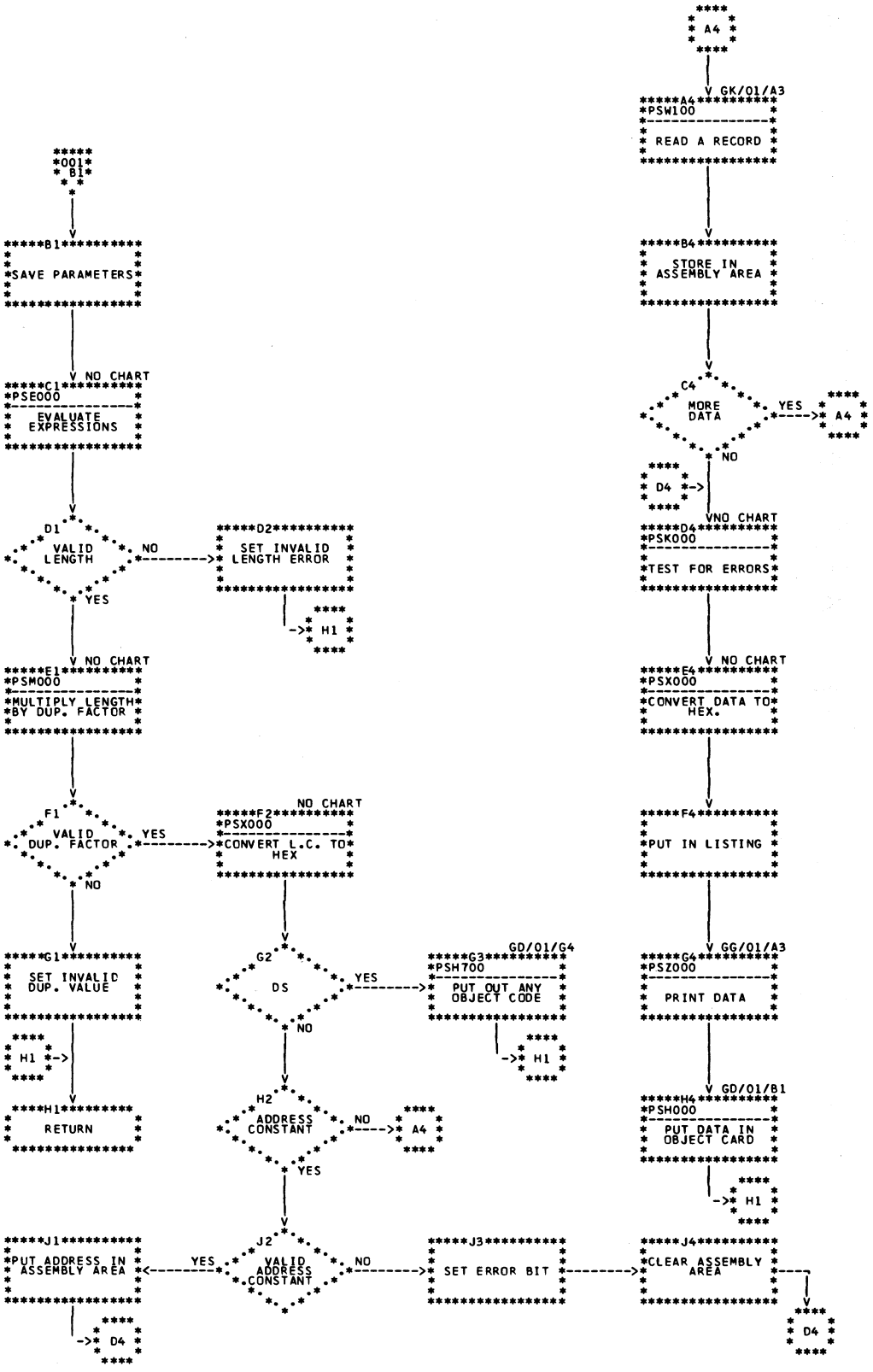


Chart GJ (Part 3 of 3). Group 2 Assembler Instruction Processing Routine (PSA000)

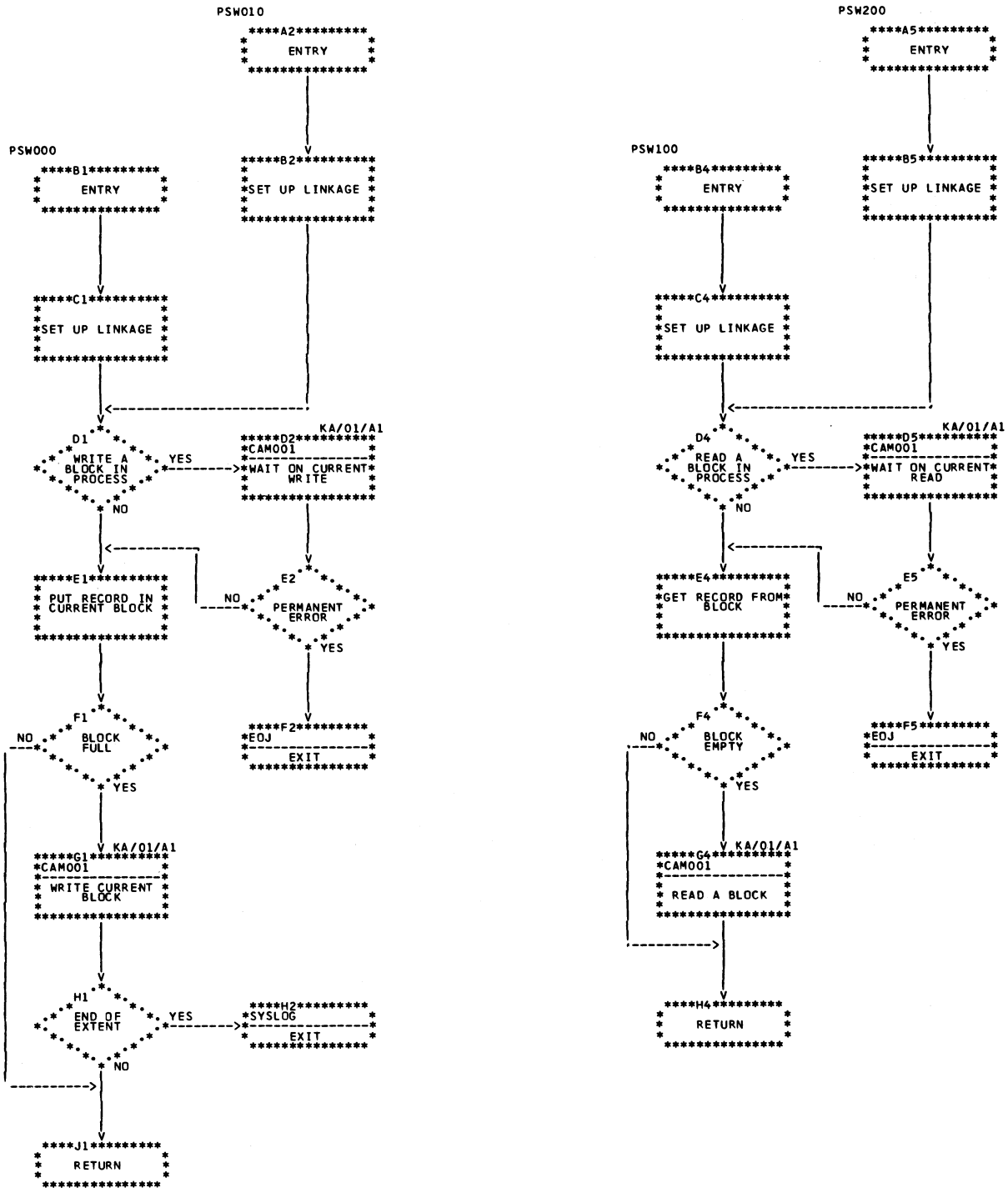


Chart GK. Disk Data Management Interface (PSW000)

MODULE $ASPD0 – PRINT DIAGNOSTICS (PHASE 35)

Main Storage Map: Figure 11.

Entry Point: $ASPD1 – entered from Phase 30 via FETCH. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Supervisor and IOS Logic Manual,* SY21-0033.)

Charts: HA-HE.

Functions:

- Controls Phase 35 processing of error records from error file (PDM000–Chart HA).
- Performs functions associated with termination of Phase 35 (PDL000–Chart HB).
- Converts the error field into error code; prints the associated codes and diagnostics (PDC000–Chart HC).
- Prints the printer listing header and counts the pages (PDG000–no chart; functionally the same as PEG000 routine in Phase 29).
- Controls page size of printed output (PDP000–Chart HD).
- Converts binary numbers to decimal (PDV000–no chart).
- Provides print module interface for print operations (PDR000–no chart, functionally the same as PER000).
- Provides interface with disk data management (PDW000–Chart HE).

Input:

- Error counts in ASMCOM.
- Error records written in $WORK2 in Phase 30.

Output:

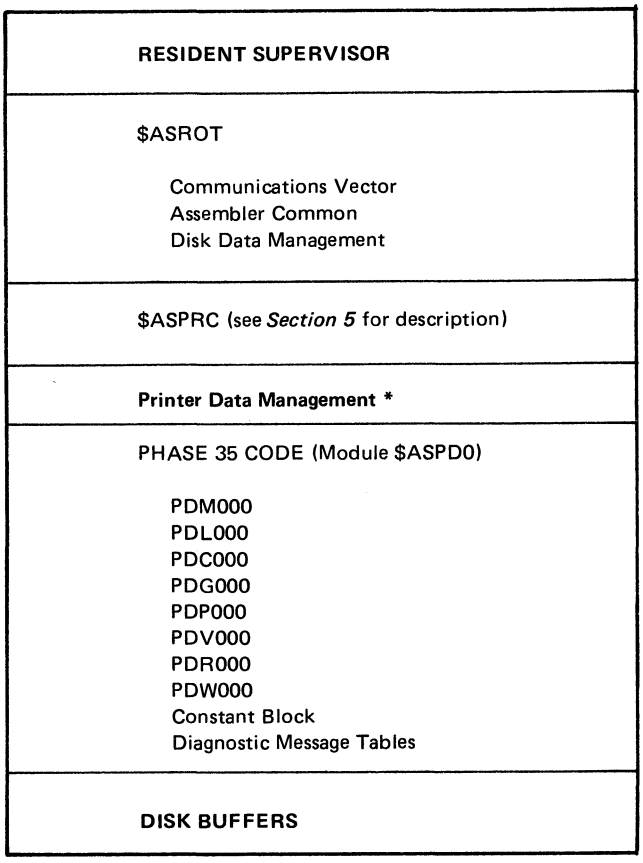
- Printed listings – diagnostic messages and error summary statements.

Exits:

- Normal
  1. If cross-reference is required, Phase 40 is fetched.
  2. If cross-reference is not required, Phase 41 is fetched.
- Error – control is returned to scheduler via Halt/Syslog or EOJ transients. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Supervisor and IOS Logic Manual,* SY21-0033.)

Diagnostic Messages Tables:

- Contain error codes and diagnostic message text for all assembler diagnostics.
- Used to convert error bit strings to diagnostic messages.
- A diagnostic pointer table contains addresses to the diagnostic table. There is a pointer for each bit position in the error field.



* Model 10 only

Figure 11. Phase 35 Main Storage Map

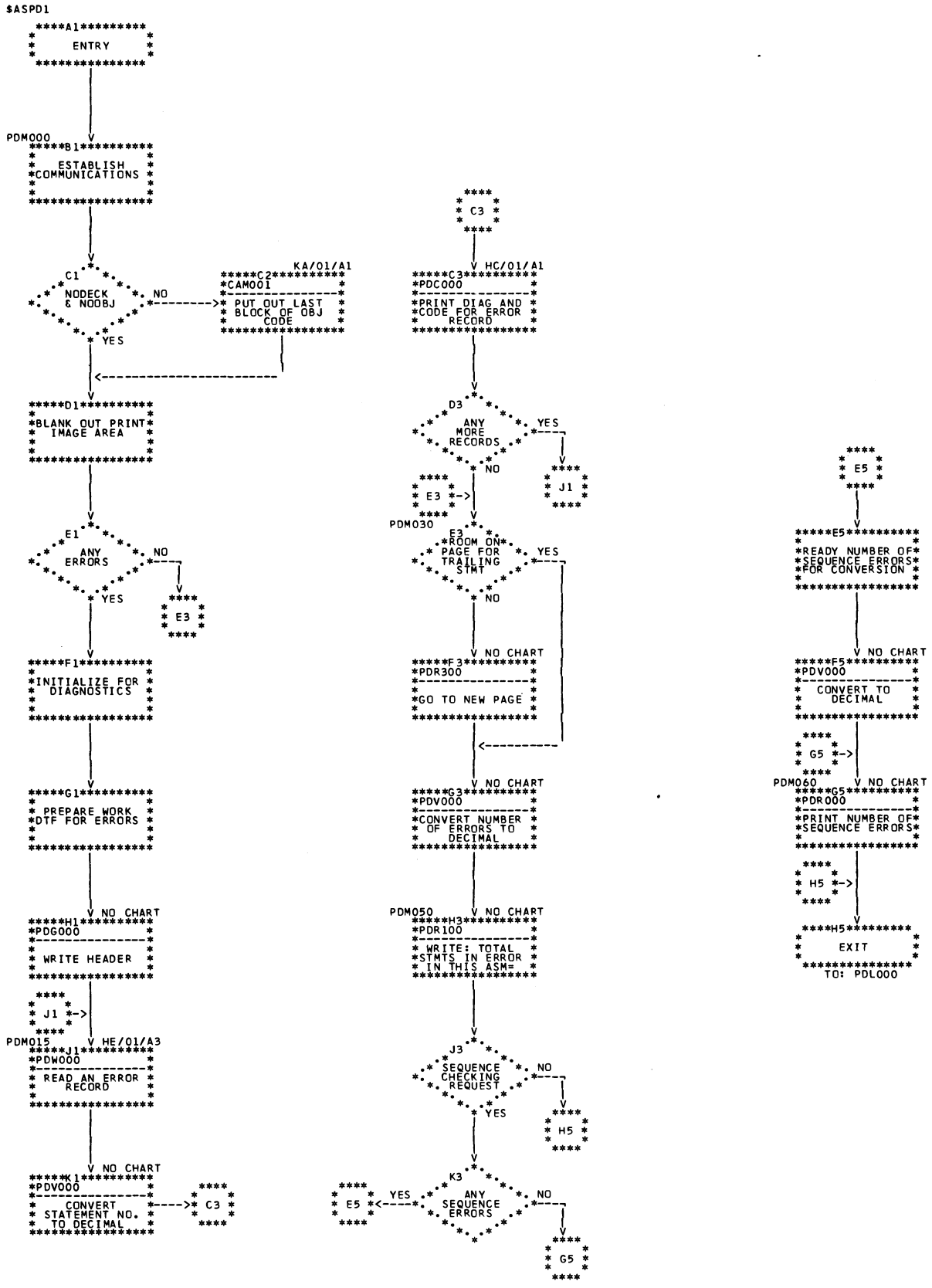


Chart HA. Main Control Routine (PDM000)

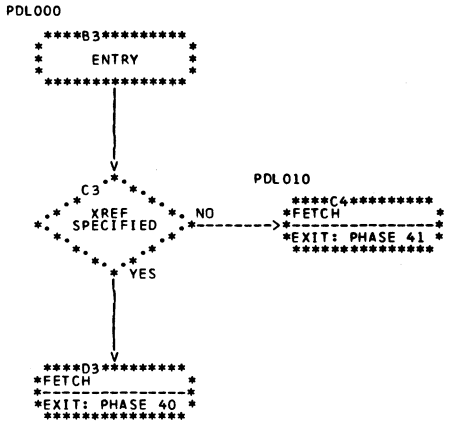


Chart HB. Termination Routine (PDL000)

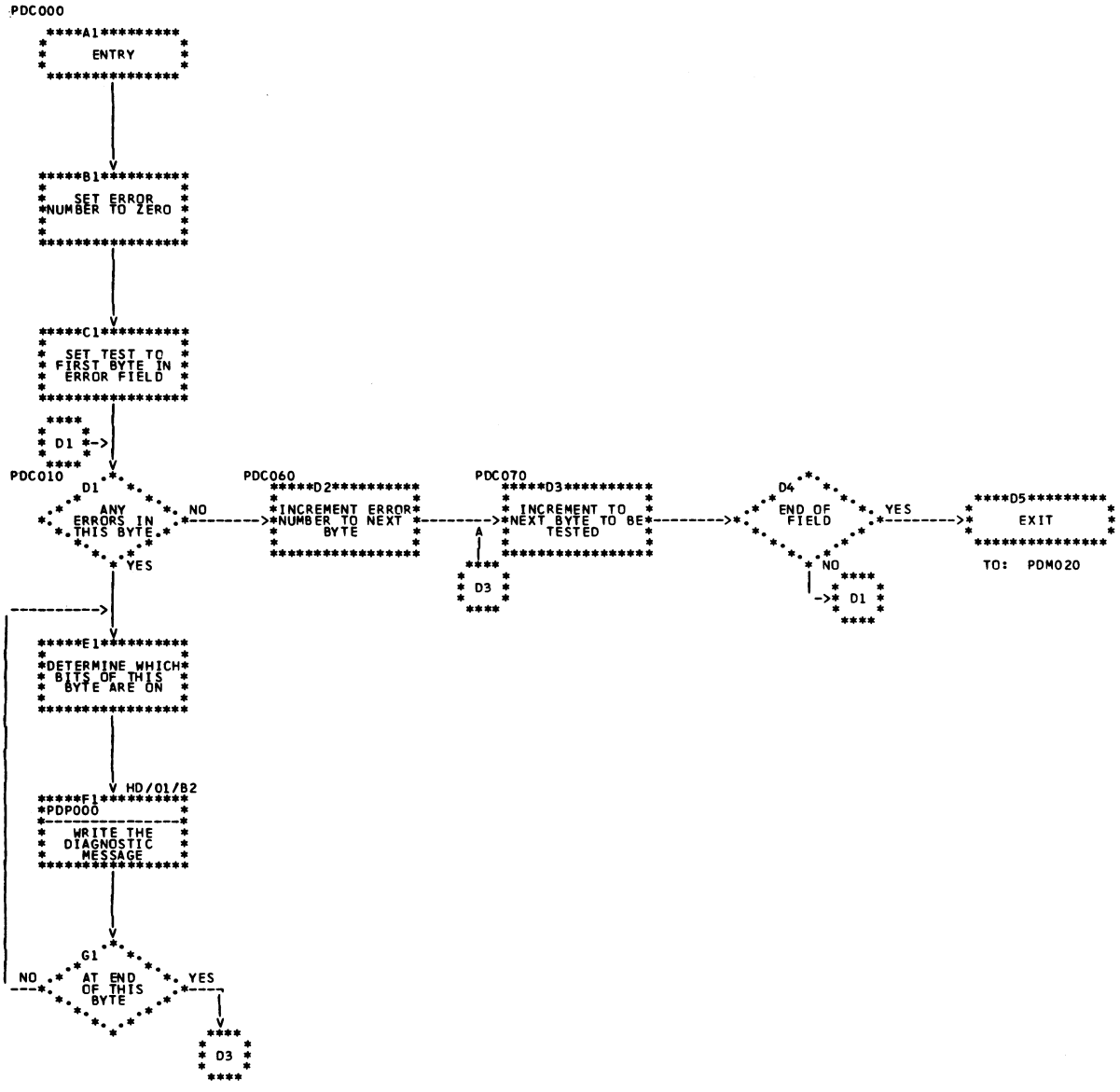


Chart HC. Diagnostic Output Routine (PDC000)

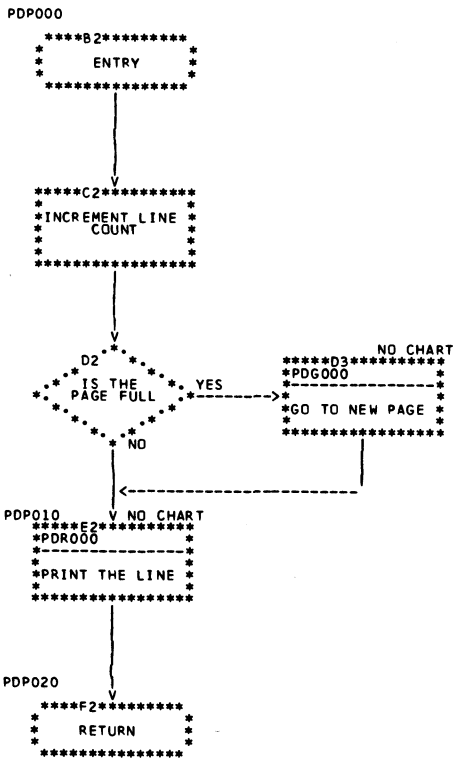


Chart HD. Print and Count Line Routine (PDP000)

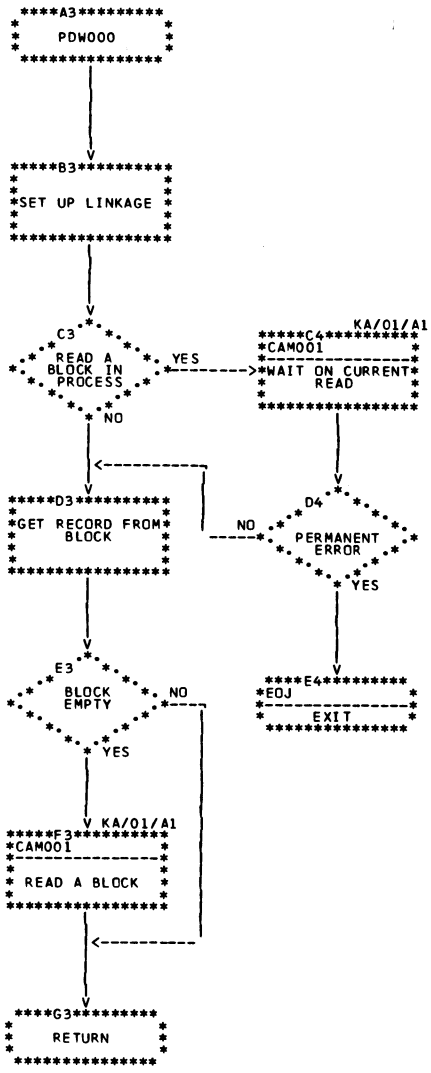


Chart HE. Disk Data Management Interface (PDW000)

MODULE $ASBX0 – BUILD CROSS REFERENCE (XREF) FILE (PHASE 40)

Main Storage Map: Figure 12.

Entry Point: $ASBX1 – entered from Phase 35 via FETCH. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Supervisor and IOS Logic Manual,* SY21-0033.)

Charts: IA-IF.

Functions:

- Initializes Phase 40 for processing (BXI000–Chart IA).
- Controls the building of the XREF sort file (BXP000–Chart IB).
- Completes Phase 40 processing when end of $WORK2 is found (BXL000–Chart IC).
- Initializes Phase 41 (BXL000–Chart IC).
- Moves records from the work area to the XREF file build area (BXM000–Chart ID).
- When XREF file build area is full, branches to XREF file block sort routine (BXS000) for sorting, then branches to BXW100 to write the blocks into the XREF sort file (BXM000–Chart ID).
- Sorts the contents of current block in XREF file build area (BXS000–Chart ID).
- Provides interface for disk data management (BXW000–Chart IF).

Input:

- Intermediate text contained in $WORK2.
- Symbol tables contained in $WORK2.

Output:

- XREF sort file.
- Assembler – common parameters
  1. XRPIMG – print image area is blanked.
  2. XRBLKC – count of blocks in sort file.
  3. XRPASC – count of passes required to merge the file.
  4. XRFRST – relative sector address of first block in file.
  5. XRLAST – relative sector address of last block in file.
  6. XRAVL1 – sort file availability table is initialized.
- Listing header area in $ASPRC is initialized for cross reference listing.

Exits:

- Normal – Phase 41 is fetched.
- Error – control is returned to scheduler via EOJ transient. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Scheduler Logic Manual,* SY21-0035.)

RESIDENT SUPERVISOR
$ASROT Communications Vector Assembler Common Disk Data Management
$ASPRC (see *Section 5* for description)
Printer Data Management *
PHASE 40 CODE (Module $ASBX0) BXI000 BXP000 BXL000 BXM000 BXS000 BXW000 Constant Block Working Storage
DISK BUFFERS

* Model 10 only

Figure 12. Phase 40 Main Storage Map

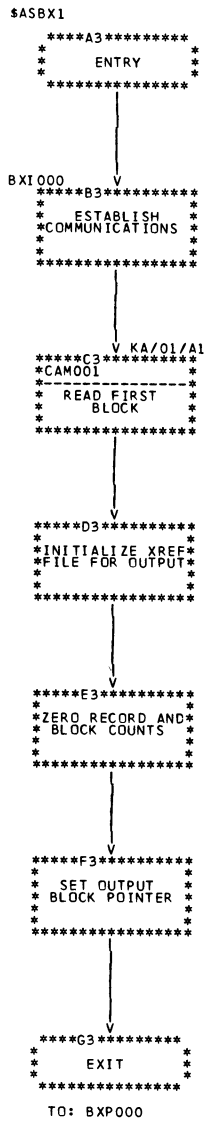


Chart IA. Initialization Routine (BXI000)

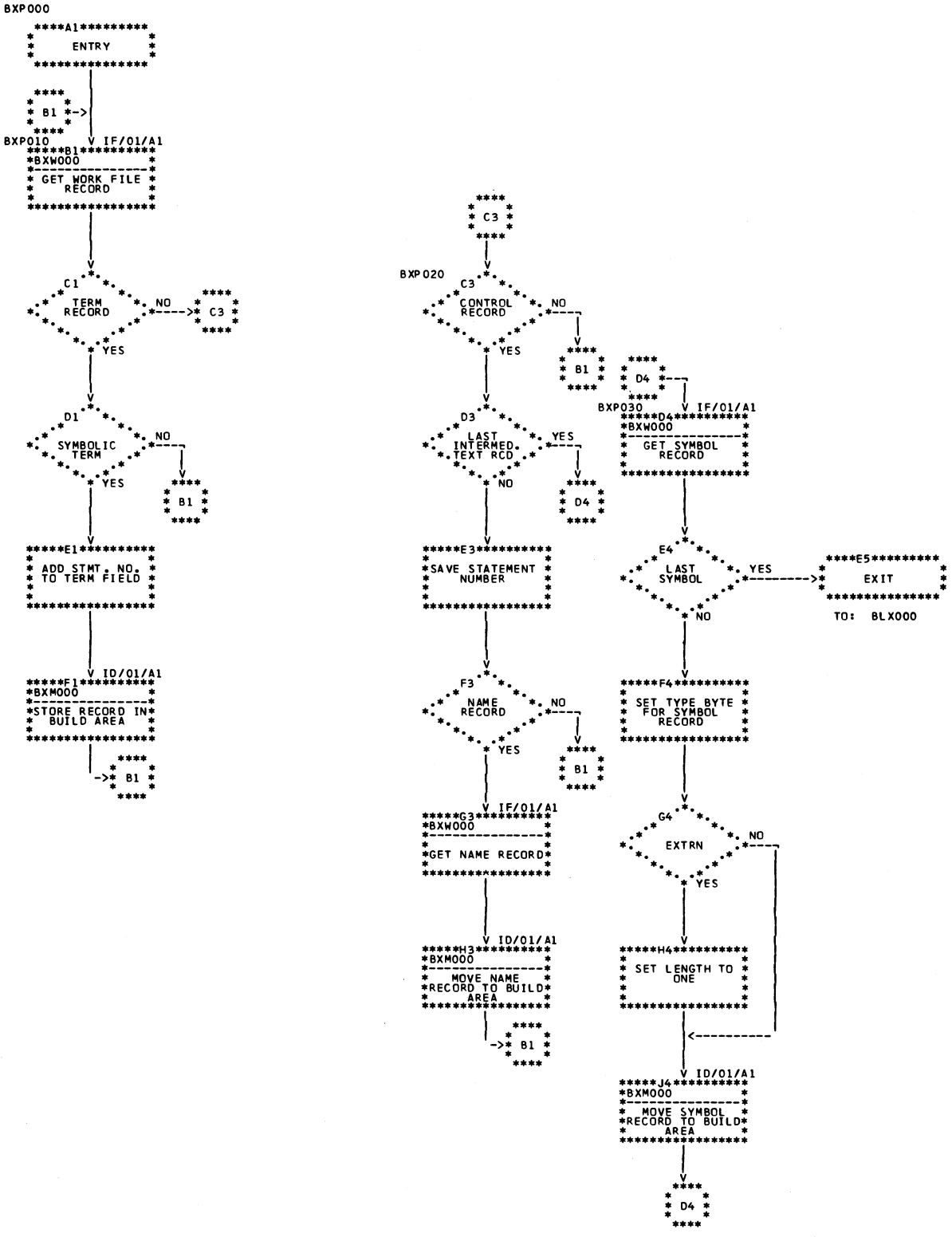


Chart IB. Cross Reference Build Routine (BXP000)

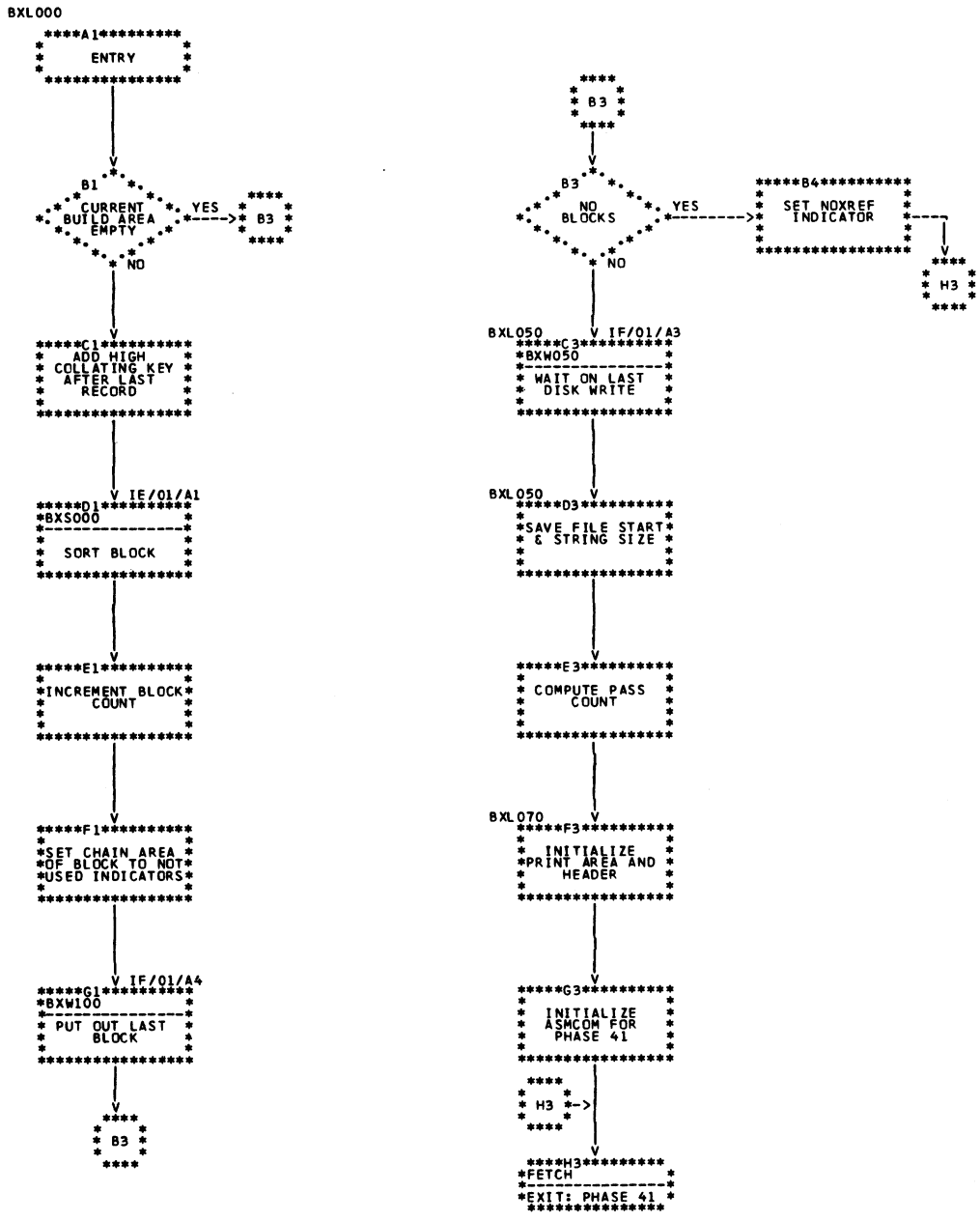


Chart IC. Termination Routine (BXL000)

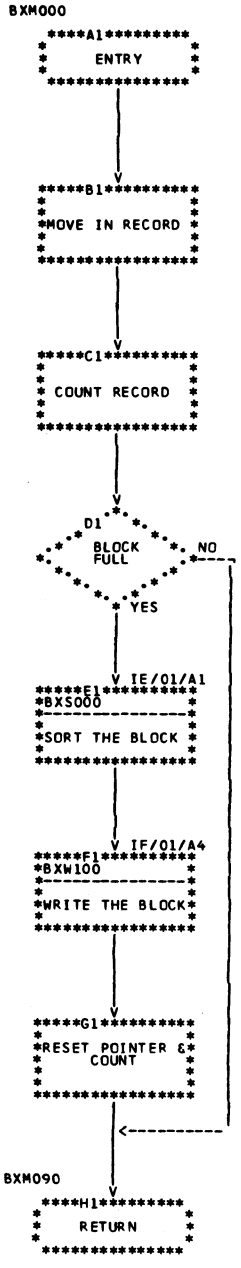


Chart ID. Build Area Move and Output Routine (BXM000)

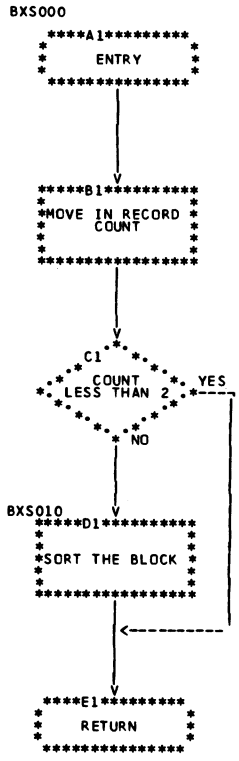


Chart IE. Cross Reference File Block Sort Routine (BXS000)

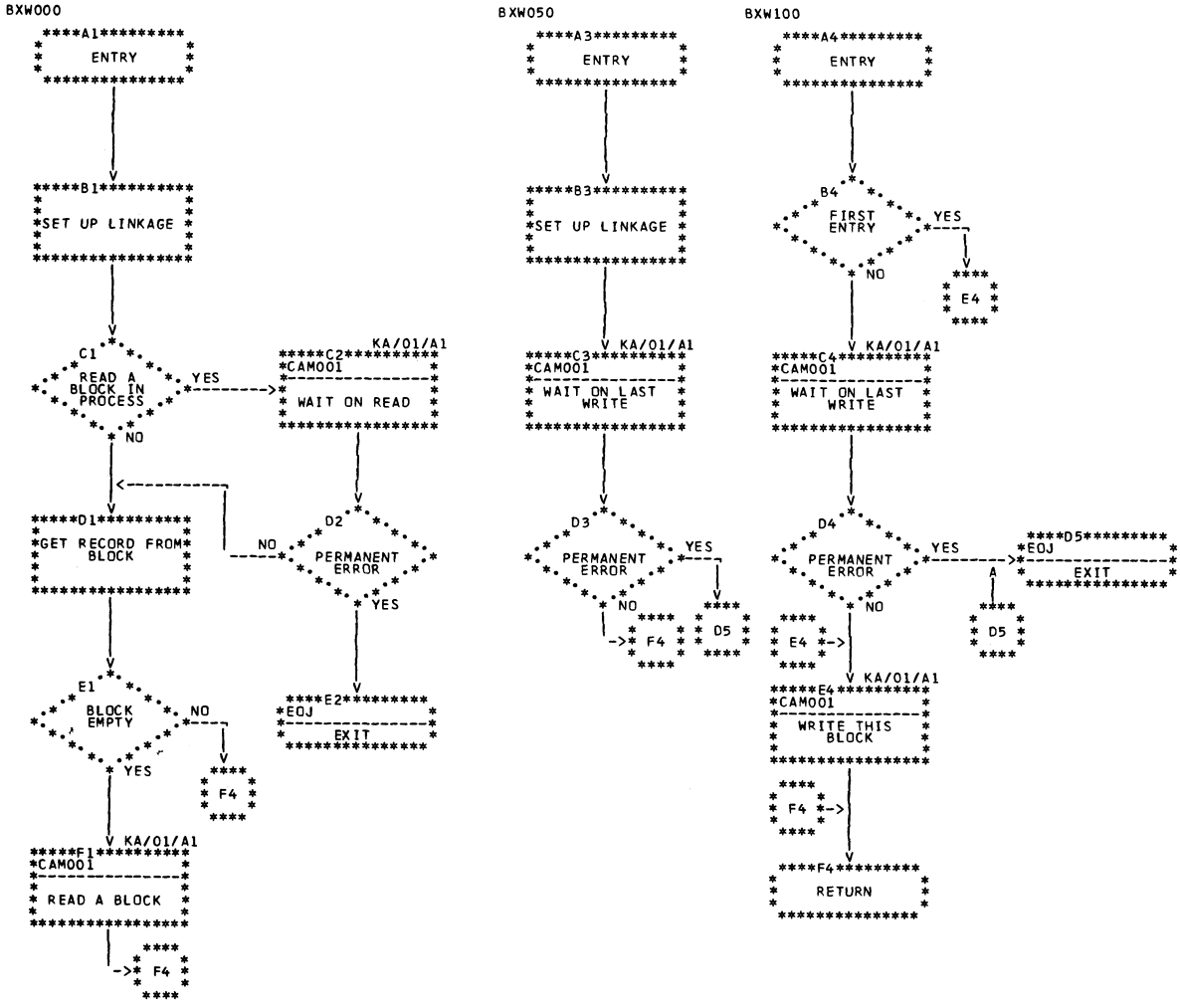


Chart IF. Disk Data Management Interface (BXW000)

MODULE $ASSX0 – MERGE AND LIST CROSS REFERENCE (PHASE 41)

Main Storage Map: Figure 13.

Entry Point: $ASSX1 – entered from Phase 40 via FETCH. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Supervisor and IOS Logic Manual,* SY21-0033.)

Charts: JA-JE.

RESIDENT SUPERVISOR
$ASROT Communications Vector Assembler Common Disk Data Management
$ASPRC (see *Section 5* for description)
Printer Data Management *
PHASE 41 CODE (Module $ASSX0) SXI000 Routine SXS000 Routine SXO000 Routine SXL000 Routine SXT000 Routine SXW000 Routine SXG000 Routine SXP000 Routine SXV000 Routine SXR000 Routine Constant Block Working Storage
DISK BUFFERS

* Model 10 only

Figure 13. Phase 41 Main Storage Map

Functions:

- Completes Phase 41 initialization (SXI000–Chart JA).
- Controls the merging of the XREF sort file (SXS000–Chart JB).
- Controls the moving of records from the input areas to the output area (SXO000–Chart JC).
- Controls the writing and chaining together of output strings and the reading of chained input strings (SXO000–Chart JC).
- Creates the XREF listing at the last pass of the sort file merge (SXL000–Chart JD).
- Closes out Phase 41 processing (SXT000–Chart JC).
- Provides interface for reading to and writing from the sort file (SXW000–Chart JE).
- Prints the printer-listing header and counts pages (SXG000–no chart).
- Controls size of page on printed output (SXP00–no chart).
- Converts binary numbers to decimal (SXV000–no chart).
- Provides print module interface for print operations (SXR000–no chart, functionally the same as PER000).

Input:

- XREF records in work $WORK2 file.
- Assembler Common parameters
  1. XRPIMG – print image area all blanks.
  2. XRBLKC – count of blocks in file.
  3. XRPASR – count of passes required to merge file.
  4. XRFRST – relative sector address of first block in file.
  5. XRLAST – relative sector address of last block in file.
- Listing header in $ASPRC module.

Output:

- Cross reference listing.
- Error summary statements of error counts.

Exits:

- Normal
  1. If object output exists, control is passed to Phase $OLYNX of the overlay linkage editor.
  2. If no object output exists, control is returned to the scheduler via EOJ. (For Model 10 Disk System, see *IBM System/3 Disk Systems System Control Program Logic Manual,* SY21-0502; for Model 12, see *IBM System/3 Model 12 System Control Program Logic Manual,* SY21-0046; for Model 15, see *IBM System/3 Model 15 Scheduler Logic Manual,* SY21-0035.)
- Error – EOJ.

Constant Block:

- All required for Phase 41 that are not stored in assembler common reside in this block.

Working Storage:

- All required Phase 41 working storage not in ASMCOM resides in this block.

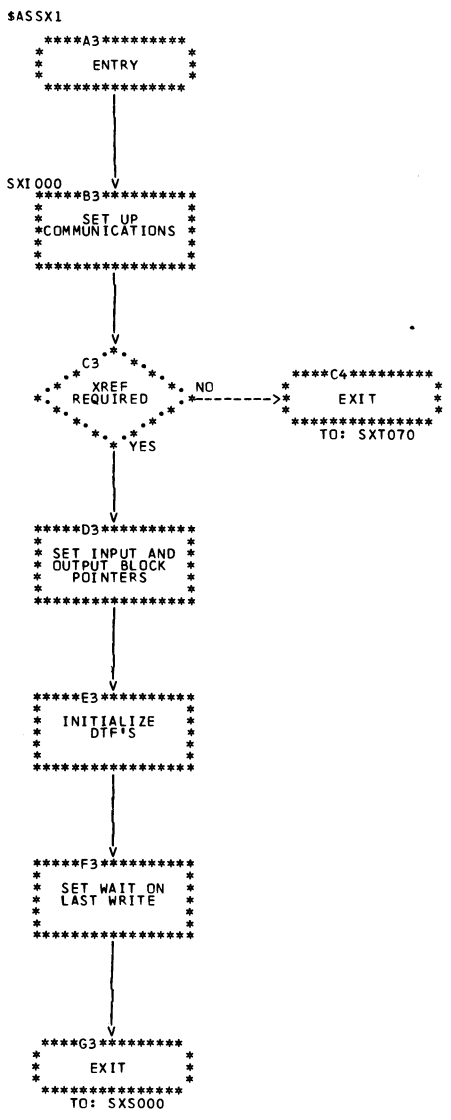


Chart JA. Initialization Routine (SXI000)

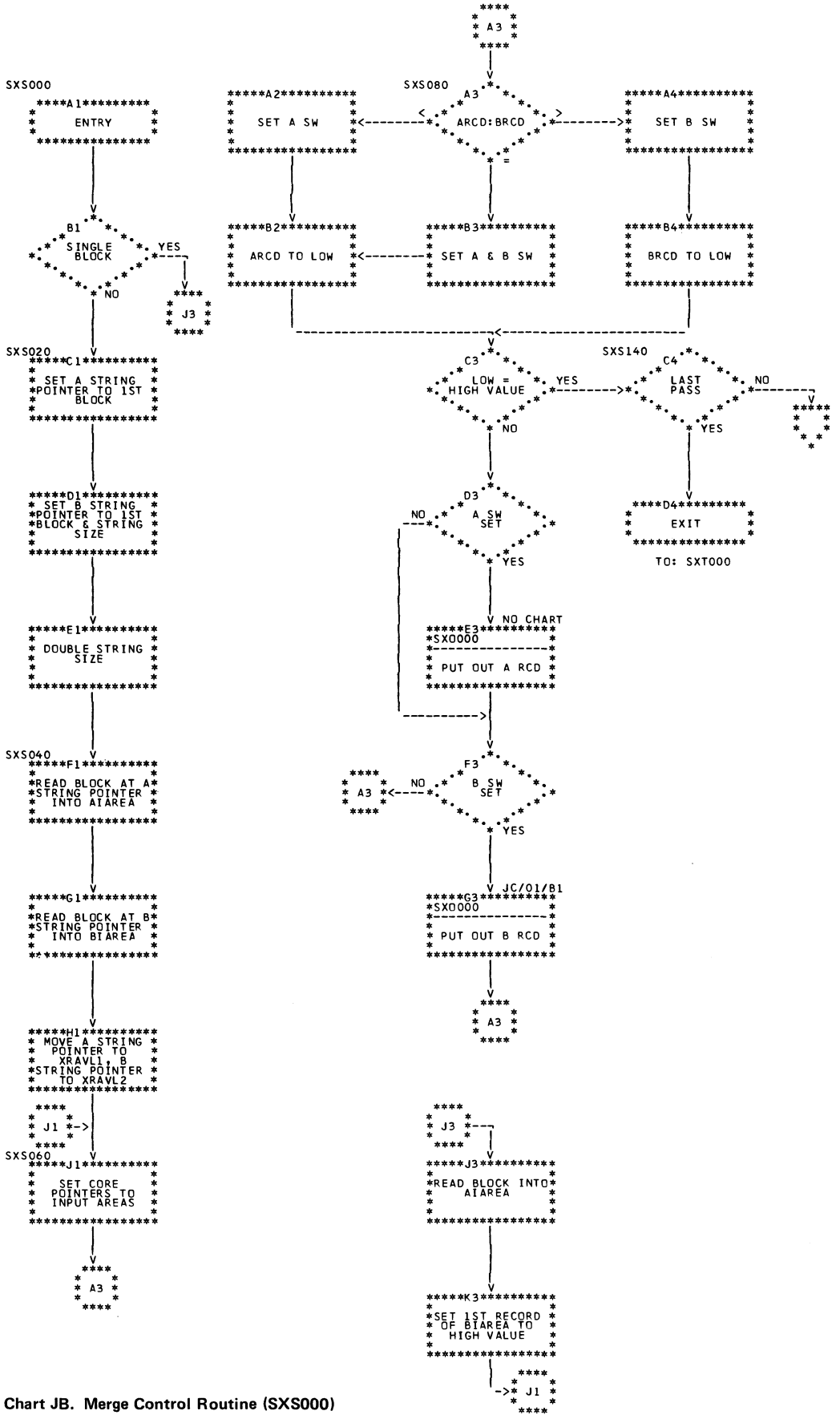


Chart JB. Merge Control Routine (SXS000)

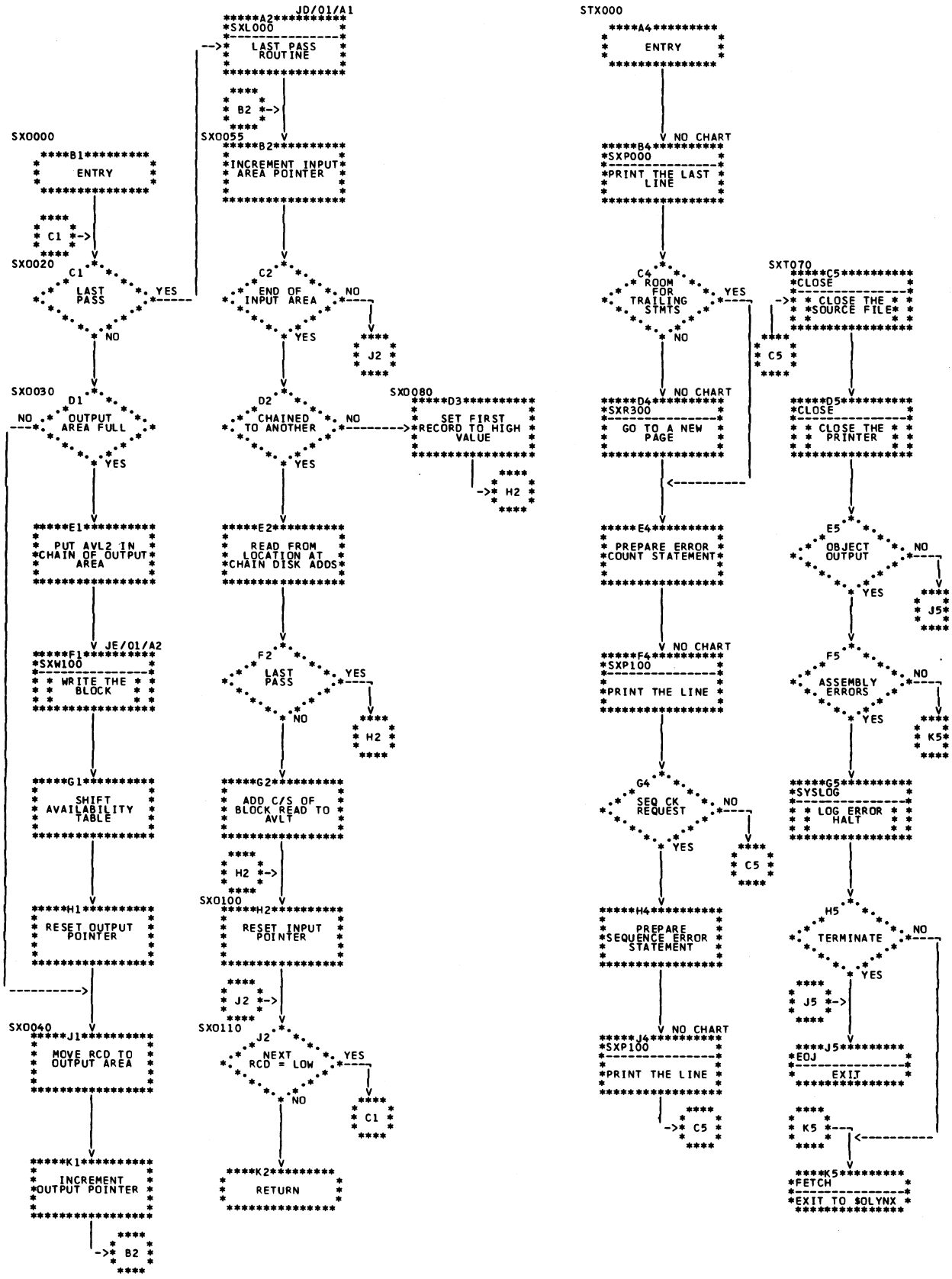


Chart JC. Merge Output Routine and Termination Routine (SXO000 and SXT000)

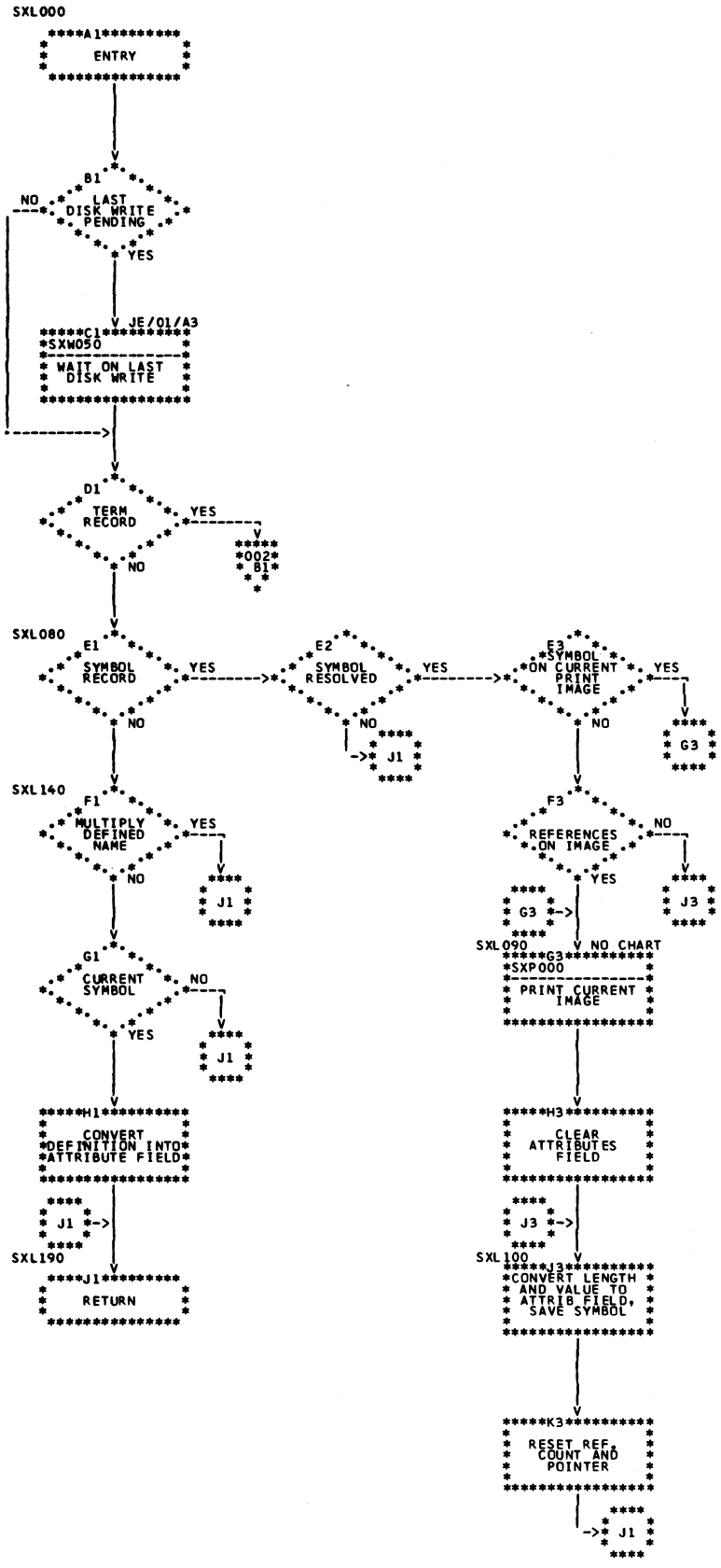


Chart JD (Part 1 of 2). Last Pass Output Routine (SXL000)

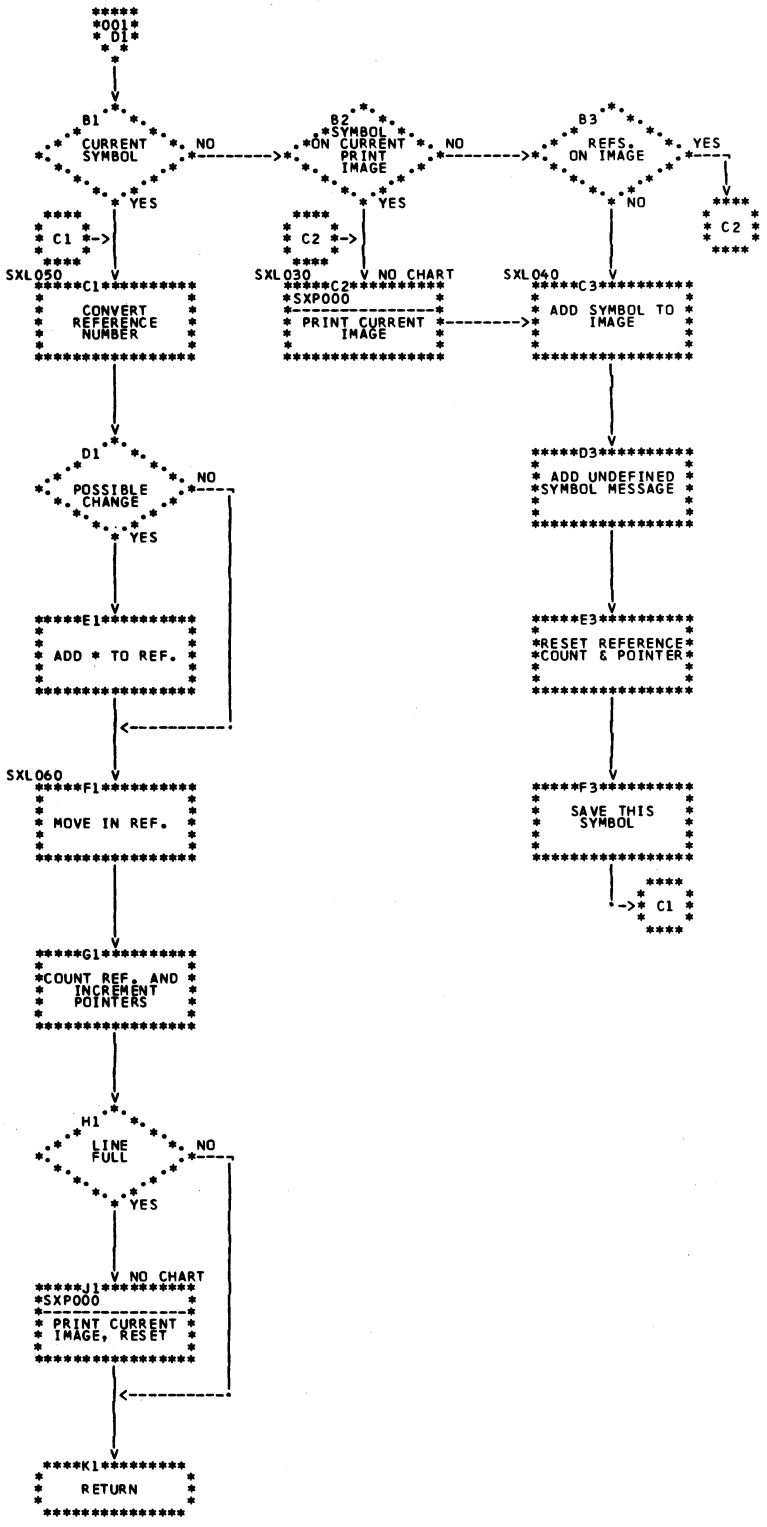


Chart JD (Part 2 of 2). Last Pass Output Routine (SXL000)

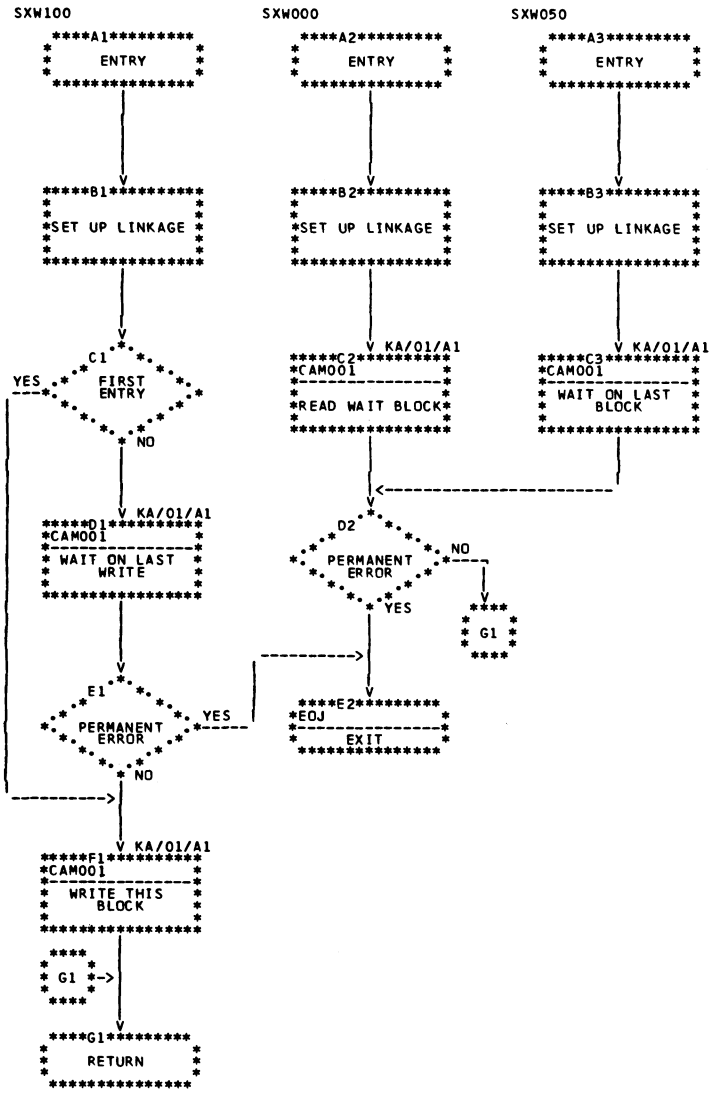


Chart JE. Disk Data Management Interfaces (SXW000)

$CAM – COMPILER ACCESS METHOD

Entry Point: CAM001 – can be called from any module.

Chart: KA.

Functions:

- Retrieves up to 255 sectors at a time according to a binary relative sector number.
- Loads up to 255 sectors at a time according to a binary relative sector number.

Input: $WORK, $WORK2, $SOURCE file.

Output: $WORK, $WORK 2 file.

Exit: Return to calling module.

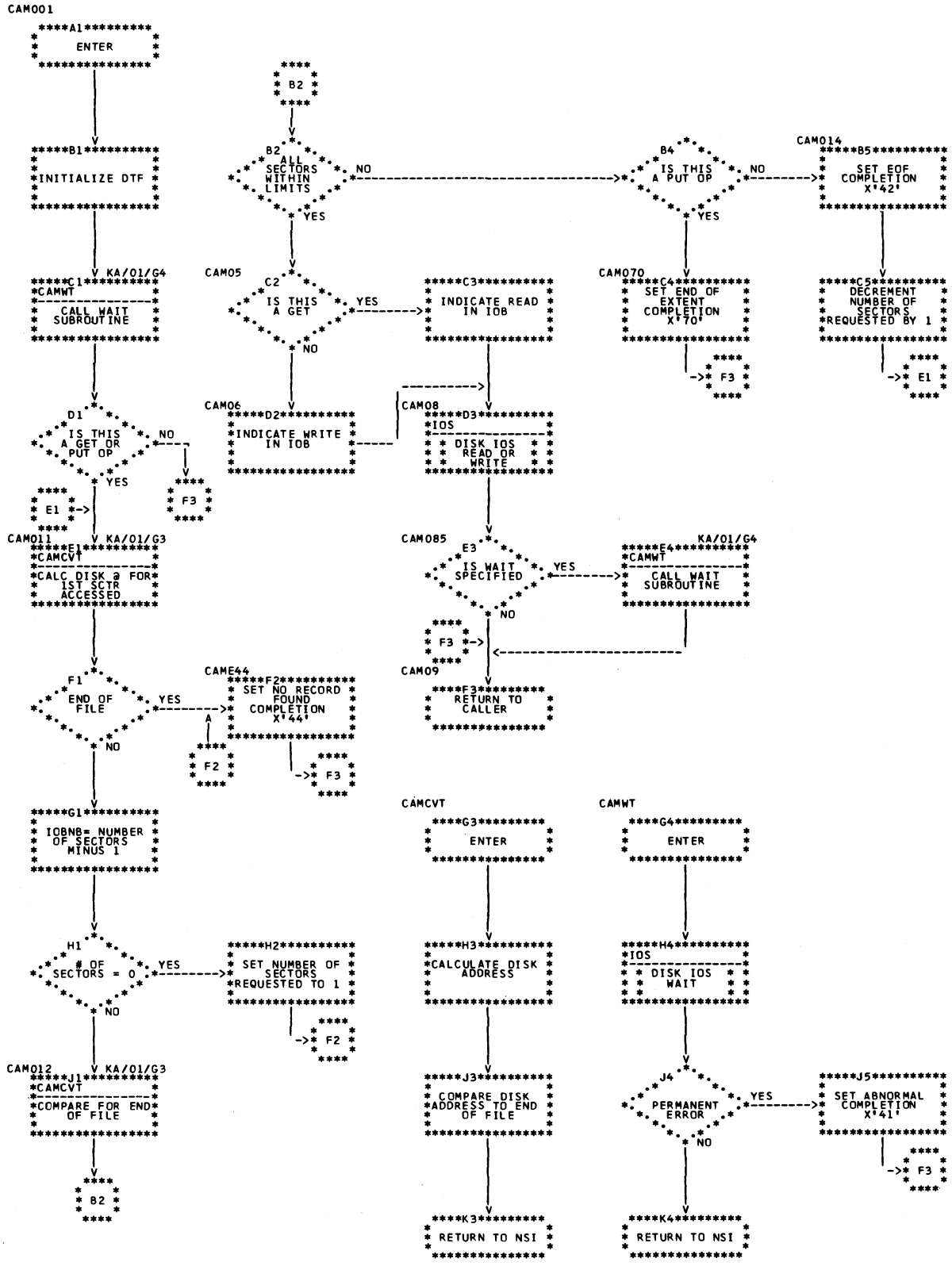


Chart KA. Compiler Access Method ($CAM001)

This section contains a quick-reference table (Figure 14) that may be used in finding phases of the assembler on microfiche. The routine names are indexed on the microfiche records.

The table consists of the following:

- Module name
- Descriptive name
- Entry Point
- Routines within each phase
- Synopsis of the functional description

MODULE NAME	DESCRIPTIVE NAME	ENTRY POINT	ROUTINES	CHART	FUNCTIONALLY IDENTICAL ROUTINE, CHART	SYNOPSIS
$ASIN0 (Phase 05)	ASSEMBLER INITIALIZATION	$ASIN1	INI000 INC000 INL000 INS000 INV000 INW000	AA AB none AC none AD		Performs assembler and Phase 10 initialization. Processes OPTIONS, HEADERS, and input format (ICTL) control cards.
$ASCM0 (Phase 10)	COMPRESSION	$ASCM1	CMI000 CMP000 CMN000 CMU000 CML000 CMO000 CMC000 CMS000 CMV000 CMY000 CMX000 CMR000 CMK000 CMD000 CMA000 CMW000	BA BB none none BG none none none none BC none none BE BF BD BH	INS000, AC	Processes all the source statements after the control cards. Builds the intermediate text file and decodes and tests the source statements.

Figure 14 (Part 1 of 3). Assembler Directory

MODULE NAME	DESCRIPTIVE NAME	ENTRY POINT	ROUTINES	CHART	FUNCTIONALLY IDENTICAL ROUTINE, CHART	SYNOPSIS
$ASSB0 (Phase 20)	SYMBOL PROCESSING	$ASSB1	SBI000 SBP000 SBL000 SBM000 SBD000 SBO000 SBS000 SBE000 SBC000 SBV000 SBX000 SBY000 SBR000 SBW000	CA CB none CC CD CE CF CJ none CG none CH CI CK		Assigns the location counter values and builds the symbol table.
$ASSF0 (Phase 21)	SYMBOL TABLE OVERFLOW	$ASSF1	SFP000 SFT000 SFN000 SFS000 SFW000	DA DB DC none DD	SBR000, CI	Tests the intermediate text from the statement following the overflow to the end of the text for symbols that have been previously defined. Previously defined symbols are marked. Resolves all references to symbols that are defined in the present symbol table. Resolved symbols are marked.
$ASSS0 (Phase 22)	SYMBOL SUBSTITUTION	$ASSS1	SSI000 SSP000 SSL000 SST000 SSE000 SSU000 SSS000 SSW000	EA EB EC none ED none none EE	SFT000, DB SBR000, CI	Assigns the symbol value and attribute from the last (or only) symbol table to that symbol in the intermediate text Term records. Builds an ESL table for module name and EXTRN and ENTRY statements.
$ASPE0 (Phase 29)	ESL OUTPUT	$ASPE1	PEI000 PEO000 PEP000 PES000 PEL000 PEG000 PEW000 PER000 PEX000	FA FB FC FD FE FF FG FI FH		Prints the OPTIONS and HEADERS records and the options selected. Punches absolute loader if DECK and NOREL are specified. Sorts and prints the ESL listing if LIST is specified. Initializes Phase 30.
$ASPS0 (Phase 30)	SOURCE/OBJECT OUTPUT	$ASPS1	PSI000 PSC000 PST000 PSK000 PSH000 PSX000 PSE000 PSY000 PSZ000 PSP000 PSG000 PSM000 PSLNK1** PSLNK2** PSR000 PSW000	GA GB GC none GD none none GF GG none none none GE GE none GK	 SBV000, CG PER000, FI	Generates object code for machine instructions and DCs. Puts out object code. Builds printer image. Prints source statements and object code generated. Performs assembly time operations requested by assembler statements.

Figure 14 (Part 2 of 3). Assembler Directory

MODULE NAME	DESCRIPTIVE NAME	ENTRY POINT	ROUTINES	CHART	FUNCTIONALLY IDENTICAL ROUTINE, CHART	SYNOPSIS
$ASOV1 * (Phase 30 - overlay 1)	MACHINE INSTRUCTION PROCESSOR AND GROUP 1 ASSEMBLER OPERATION PROCESSOR	PSQ000	PSQ000** PSS000 PSB000	none GH GI		Processes machine instructions and Group 1 Assembler instructions.
$ASOV2 * (Phase 30 - overlay 2)	GROUP 2 ASSEMBLER OPERATION PROCESSOR	PSA000	PSA000	GJ		Processes Group 2 Assembler instructions.
$ASPD0 (Phase 35)	PRINT DIAGNOSTICS	$ASPD1	PDM000 PDL000 PDC000 PDG000 PDP000 PDV000 PDR000 PDW000	HA HB HC none HD none none HE	 PEG000, FF PER000, FI	Reads the Error records and prints diagnostic messages. Prints the error summary statements.
$ASBX0 (Phase 40)	BUILD XREF FILE	$ASBX1	BXI000 BXP000 BXL000 BXM000 BXS000 BXW000	IA IB IC ID IE IF		Reads the work file and creates the cross reference sort file. Initializes working storage, print area, and listing header for Phase 41.
$ASSX0 (Phase 41)	MERGE AND LIST XREF	$ASSX1	SXI000 SXS000 SXO000 SXL000 SXT000 SXW000 SXG000 SXP000 SXV000 SXR000,	JA JB JC JD JC JE none none none none	 PER000, FI	Merges the XREF sort file. Generates the XREF listing. Prints the error summary statements. Fetches Overlay Linkage Editor if object output is required.
$CAM	COMPILER ACCESS METHOD	CAM001		KA		Interface to Disk IOS to read or write blocks of one or more sectors.

* On Model 12 and Model 15, these routines are part of $ASPS0.
** On Model 10 only; not present for Model 12 and Model 15.

Figure 14 (Part 3 of 3). Assembler Directory

All data areas used by more than one phase of the Assembler are included in this section:

1. $ASROT – Assembler Communications Module. Provides for interphase communication through a table of indirect linkages, table pointers, and interphase storage areas (see Figure 15).

2. Communications Vector (COMVEC) – a part of $ASROT. Contains the transfer vector and an address table for all other tables in the $ASROT module (see Figure 16).

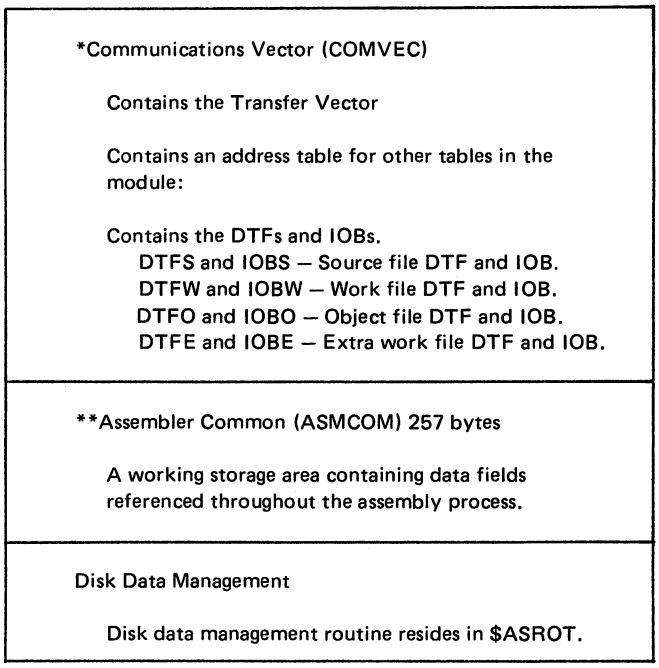
3. Assembler Common (ASMCOM) – a part of $ASROT. Used for interphase constants, working storage, and communication (see Figure 17).

4. Assembler Common Interphase Parameter Usage Chart – shows which fields in ASMCOM are used by each phase (see Figure 18).

5. $ASPRC Module – contains printer module interface, printer buffer, and listing header (see Figure 19).

6. Symbol Table – a table containing symbols found in the source program (see Figures 20 and 21).

7. Work File ($WORK2) – a scratch file on disk used for intermediate storage throughout the assembly process (see Figures 22 and 23).



*See Figure 16 for detailed description of COMVEC.

**See Figure 17 for detailed description of ASMCOM.

Figure 15. $ASROT – Assembler Communications Module

MODEL 10				MODEL 12 and MODEL 15			
BYTES (310 total) From	To	Hexadecimal Displacement	Contents	BYTES (532 total) From	To	Hexadecimal Displacement	Contents
		COMVEC Entry at 00				COMVEC Entry at 00	
0	5	05	Current module name.	-5	0	00	Current module name.
6	7	07	Displacement from COMVEC to ASMCOM.	1	2	02	Displacement from COMVEC to ASMCOM.
TRANSFER VECTOR				TRANSFER VECTOR			
8	11	08	Transfer to CAM.	3	6	03	Transfer to CAM.
12	15	0C	Transfer to printer module.	7	10	07	Transfer to SYSLST or SYSPRT.
ADDRESS TABLE				ADDRESS TABLE			
16	17	–	CAM transfer address.	11	12	–	CAM transfer address.
18	19	13	Printer data management transfer address.	13	14	–	SYSLST or SYSPRT routine address.
20	21	15	Printer DTF address.	15	16	10	SYSLST or SYSPRT parameter list address.
22	23	17	Header area start address.	17	18	12	Header area start address.
24	25	19	$ASROT module end address +1.	19	20	14	Work DTF address.
				21	22	16	Source DTF address.
				23	24	18	Object DTF address.
				25	26	1A	Extra work DTF address.
DTFs and IOBs				DTFs and IOBs			
26	74	1A	Source file ($SOURCE) DTF.	27	121	1B	Work file ($WORK2) DTF and extension.
75	96	4B	Source file ($SOURCE) IOB.	122	151	–	Work file ($WORK2) IOB.
97	145	61	Work file $WORK2 DTF.	152	246	–	Source file ($SOURCE) DTF and extension.
146	167	92	Work file $WORK2 IOB.	247	276	–	Source file ($SOURCE) IOB.
168	216	A8	Object file $WORK DTF.	277	371	–	Object file ($WORK) DTF and extension.
217	238	D9	Object file $WORK IOB.	372	401	–	Object file ($WORK) IOB.
239	287	EF	Extra work file DTF.	402	496	–	Extra work file DTF and extension.
288	309	–	Extra work file IOB.	497	526	–	Extra work file IOB.

Figure 16. Communications Vector (COMVEC)

ASSEMBLER COMMON (ASMCOM) (phases 05 and 10)

Field Name	Field Length	Hexadecimal Displacement Model 10	Hexadecimal Displacement Models 12 and 15	Contents
COMVDP	2	00	00	Communications vector displacement
I0	2	02	02	Binary zero constant
I1	2	04	04	Binary one constant
I2	2	06	06	Binary two constant
OPTONS	1	07	07	Indicates options selected
SEQCNT	2	09	09	Count of sequence errors in the source file ($SOURCE)
INTRCD	10	13	13	Work file disk data management work area
BGNCOL	2	15	15	Source record begin column
OPTCNT	1	16	16	Count of OPTIONS records
MODULE	6	1C	1C	Module name
TITLEN	6	22	22	Title name
ESLCNT	1	23	23	Count of ESL records
CATGVL	1	24	24	Module category level
COML4V	2	26	26	EXTRN subtype 4 common length
COML5V	2	28	28	EXTRN subtype 5 common length
SYTBSZ *	1	—	29	Symbol table size in sectors
NAMLNG	2	2A	2B	Name field length
OPTCOL	2	2C	2D	Right-most column number of operation code
OPTLNG	2	2E	2F	Operation code length
OPDCOL	2	30	31	Left-most column number of operand field
ORDLNG	2	32	33	Operand field length
OPDEND	2	34	35	Right-most column number of operand field
PRECOL	2	36	37	Pointer to current source column
OPFTEN	3	39	3A	Operand format table end
NOFOPD	1	3A	3B	Number of operands
EPOSEN	12	46	47	Operand element position table
NOFELE	1	47	48	Number of elements in operand
INTEND	10	51	52	Control record build area
NAMEND	10	5B	5C	Name record build area
TRMEND	10	65	66	Term record build area
ERREND	10	6F	70	Error record build area
SOURFD	97	D0	D1	Source record input area
LASTSQ	8	D8	D9	Last source sequence field
CTLSWS	1	D9	DA	Control switches
AAWORK	2	DB	DC	Work area A
ABWORK	2	DD	DE	Work area B
ACWORK	2	DF	E0	Work area C
PACI	1	E0	E1	Pseudo accumulator switches
PAC	4	E4	E5	Pseudo accumulator one
TPAC	4	E8	E9	Pseudo accumulator two
SWITCH1	1	E9	EA	Temporary bit switches
I3	2	EB	EC	Binary three constant
EXTCNT	1	EC	ED	EXTRN count
ESLMAX *	1	—	EE	Maximum ESL record count
ON **	2	EE	—	Print key word
OFF **	3	F1	—	Print key word
DATA **	4	F5	—	Print key word
NODATA**	6	FB	—	Print key word
ENDCOL	2	FD	F0	Source record end column
CMXBUF***	5	—	F5	External buffer
ASMCEN		FF	FF	ASMCOM end

*Model 12 and Model 15 only.
**Model 10 only.
***Model 15 using 5704-AS2 only.

Figure 17 (Part 1 of 6). Assembler Common (ASMCOM)

ASSEMBLER COMMON (ASMCOM)
(Phases 20 and 21)

Field Name	Field Length	Hexadecimal Displacement	Contents
ASMCOM is the same as in Phases 05 and 10 through COML5V for Model 10 or SYTBSZ for Model 15.			
LOCNTR	3	2C	Current location counter value
LCRCHG	3	2F	Location counter increment
LOCRSW	1	30	Location counter switch
SYMOVS	1	31	Symbol table overflow switches
NSTENT	2	33	Number of table entries
MAXSTE	2	35	Maximum number of entries
SYTBSA	2	37	Symbol table start address
SYTBEA	2	39	Symbol table end address
SYMCNT	2	3B	Current symbol count
SYMIND	1	3C	Symbol table indicators
SYMEND	10	46	Symbol entry build/search area
INTEND	10	50	Control record save area
TRMEND	10	5A	Term record save area
ERREND	10	64	Error record build area
AAWORK	2	66	Work area A
ABWORK	2	68	Work area B
ACWORK	2	6A	Work area C
PACI	1	6B	Pseudo accumulator switches
PAC	4	6F	Pseudo accumulator one
TPAC	4	73	Pseudo accumulator two
INTLCR	3	76	Initial location counter
MAXLCR	3	79	Maximum location counter
EXPVTB	44	A5	Expression evaluation tables
EXPWRK	14	B3	Expression evaluation work area
MWORK	4	B7	Multiplication work area
SYMSAV	6	BD	Symbol save area
.	.	.	.
.	.	.	.
. (not used)	.	.	.
.	.	.	.
ASMCEN		FF	ASMCOM end

ASSEMBLER COMMON (ASMCOM)
(Phase 22)

Field Name	Field Length	Hexadecimal Displacement	Contents
ASMCOM is the same as in Phases 20 and 21 through MAXLCR.			
ESLINC	2	7B	ESL record length constant
ESLPTR	2	7D	Pointer to ESL table
ESLREC	12	89	ESL record build area
.	.	.	.
.	.	.	.
. (not used)	.	.	.
.	.	.	.
ASMCEN		FF	ASMCOM end

Figure 17 (Part 2 of 6). Assembler Common (ASMCOM)

ASSEMBLER COMMON (ASMCOM)
(Phase 29)

Field Name	Field Length	Hexadecimal Displacement	Contents
ASCOM is the same as in Phases 05, 10, 20, 21 and 22 through ESLCNT.			
ESLINC	2	25	ESL record increment
	2	27	Not used
ESLTBA	2	29	Address of ESL table
ESLREC	12	35	ESL record save area
	2	37	Not used
AAWORK	2	39	Work area A
ABWORK	2	3B	Work area B
ACWORK	2	3D	Work area C
PACI	1	3E	Pseudo accumulator switches
PAC	4	42	Pseudo accumulator one
TPAC	4	46	Pseudo accumulator two
.	.	.	.
.	.	.	.
. (not used)	.	.	.
CMXBUF*	5	F5	External buffer size
ASMCEN		FF	ASMCOM end

*Model 15 using 5704-AS2 only.

ASSEMBLER COMMON (ASMCOM)
(Phase 30)

Field Name	Field Length	Hexadecimal Displacement	Contents
ASMCOM is the same as in previous phases through BGNCOL.			
ERRCNT	2	17	Count of source statements in error
PAGCNT	1	18	Printer page size
LINCNT	1	19	Printer line count
LPSIZE	1	1A	Printer column size
LPCLSW	1	1B	Print size switches
INTEND	10	25	Control record area
ERREND	11	30	Error Record area
PRNTSW	1	31	Print control switches
USWTCH	1	32	Temporary bit switches
LDADDR	2	34	Object code load address
DCWORK	3	37	DC Work area
AAWORK	2	39	Work area A
ABWORK	2	3B	Work area B
ACWORK	2	3D	Work area C
PACI	1	3E	Pseudo accumulator switches
PAC	4	42	Pseudo accumulator one
TPAC	4	46	Pseudo accumulator two
RELIND	1	47	Relocation and EXTRN switches
OBJECT	8	4F	Object code assembly area
VALUE1	2	51	Base value for register one
VALUE2	2	53	Base value for register two
HEXOBJ	16	63	Hex object conversion area
EXPVTB	44	8F	Expression evaluation tables
MWORK	4	93	Multiplication work area
OBDADF	2	95	Duplication factor for DC code
OBDAAD	2	97	Address of DC code

Figure 17 (Part 3 of 6). Assembler Common (ASMCOM)

Field Name	Field Length	Hexadecimal Displacement	Contents	
OBDLTH	2	99	Object code length	
MOVELH (Model 10 Disk System)	2	9B	Object code move length	
EXTNUM (Model 12 and Model 15)	2	9B	Object code move length	
RELDSP	1	9C	RLD value	
BYTAVL	2	9E	Bytes left on object record	
LOADSW (Model 10 Disk System only)	1	9F	$ASOV1/$ASOV2 Load switch	
FX96LN	1	A0	Fold column for 96 column printer	
F120LN	1	A1	Fold column for 120 column printer	
DCZERO	1	A2	Decimal zero	
SEQALP	3	A5	Sequence error code	
OPCDNO	1	A6	Operation code error constant	
OPNDNO	1	A7	Operand error constant	
	2	A9	Binary eight	
	2	AB	Binary minus eight	
	2	AD	Binary sixteen	
	1	AE	Binary fifty seven	
I256	2	B0	Binary two hundred fifty six	
QCDMAX	2	B2	Maximum MVX instruction Q-code value	
JPDEC3	2	B4	Jump instruction length	
HEXSAD	2	B6	Start address of Hex conversion	
HEXEAD	2	B8	End address of Hex conversion	
D1	1	B9	Decimal one	
TXTTXA	2	BB	Object text address	
TXTRLA	2	BD	Object RLD field address	
ASEMAD	2	BF	Assembly area address	
SOURAD	2	C1	Source start address	
OBOADR	2	C3	Object input area address	
SRCADR	2	C5	Source input area address	
STMTNO	4	C9	Source statement number	
MINUS1	2	CB	Binary minus one	
NAMEXN	2	CD	EXTRN number	
OPADEC	2	CF	Output area decrement	
OBJDEC	2	D1	Input area decrement	
PRTOAD	2	D3	Print area start address	
PRTOCA	2	D5	Print object field address	
ASEMSA	2	D7	Assembly area start address	
HEADRA	2	D9	Header area address	
PSE0ID	2	DB	PSE000 address	Model 10 Disk System
PSG0ID	2	DD	PSG000 address	
PSH0ID	2	DF	PSH000 address	
PSH7ID	2	E1	PSH700 address	
PSK0ID	2	E3	PSK000 address	
PSM0ID	2	E5	PSM000 address	
PSR2ID	2	E7	PSR200 address	
PSW1ID	2	E9	PSW100 address	
PSX0ID	2	EB	PSX000 address	
PSY0ID	2	ED	PSY000 address	
PSZ0ID	2	EF	PSZ000 address	
	.	F0-FF	Not used	
MOVELH	2	DB	Object code move length	(Model 12 and Model 15)
RELOPD	1	DC	Operand relocation indicators	
EXTN01	1	DD	EXTRN number for operand 1	
EXTN02	1	DE	EXTRN number for operand 2	
		DF-FF	Not used	
ASMCEN		FF	ASMCOM end	

Figure 17 (Part 4 of 6). Assembler Common (ASMCOM)

ASSEMBLER COMMON (ASMCOM)
(Phase 35)

Field Name	Field Length	Hexadecimal Displacement	Contents
ASMCOM is the same as in Phase 30 through LPCLSW.			
AAWORK	2	1D	Work area A
ABWORK	2	1F	Work area B
ACWORK	2	21	Work area C
PACI	1	22	Pseudo accumulator switches
PAC	4	26	Pseudo accumulator one
TPAC	4	2A	Pseudo accumulator two
ERRNUM	2	2C	Error number within error table
ERNBIN	1	2D	Error number increment per byte
ERNOIN	1	2E	Error number increment per bit
ADIAGP	2	30	Address of diagnostic field
PRNIEN	96	90	Print image field
.	.	.	.
.	.	.	.
. (Not Used)	.	.	.
.	.	.	.
ASMCEN		FF	ASMCOM end

ASSEMBLER COMMON (ASMCOM)
(Phases 40 and 41)

Field Name	Field Length	Hexadecimal Displacement	Contents
ASMCOM is the same as in Phases 30 and 35 through LPCLSW.			
XRATAB	12	27	"A" merge string table
XRBTAB	12	33	"B" merge string table
XRAVLT	8	3B	Cross-reference file availability table
XRPASR	2	3D	Merge passes to last pass
XRPASC	2	3F	Current pass count
XRFRST	2	41	C/S address of 1st XREF file block
XRLAST	2	43	C/S address of last XREF file block
XRBSST	2	45	Block string length in sectors
XRRCSZ	2	47	XREF file record size
XRBLKC	2	49	Blocks in XREF file
XRCTAB	14	57	XREF file output block table
XRPIME	96	B7	Print image area
XRREFC	1	B8	Reference count
XRREFP	2	BA	Reference pointer
XRRFPS	2	BC	Start reference pointer
XRRDSW	1	BD	A/B string switch
XRSYMS	6	C3	Current symbol save area
XRKYLC	2	C5	Length of sort key minus 1
XRNKRL	2	C7	Length of record not in key
AAWORK	2	C9	Work area A
ABWORK	2	CB	Work area B
ACWORK	2	CD	Work area C
PACI	1	CE	Pseudo accumulator switches

Figure 17 (Part 5 of 6). Assembler Common (ASMCOM)

ASSEMBLER COMMON (ASMCOM)
(Phases 40 and 41 Cont.)

Field Name	Field Length	Hexadecimal Displacement	Contents
PAC	4	D2	Pseudo accumulator one
TPAC	4	D6	Pseudo accumulator two
.	.	.	.
.	.	.	.
. (Not Used)	.	.	.
.		.	.
ASMCEN		FF	ASMCOM end

Figure 17 (Part 6 of 6). Assembler Common (ASMCOM)

ASSEMBLER COMMON INTERPHASE PARAMETER USAGE

Only data fields used by more than one phase are shown. Fields which are merely initialized for the next phase are not included. The letter X indicates the phase in which a data field is initialized and first used; the letter Y indicates a phase in which a data field is used (other than that in which it is initialized or terminated); the letter Z indicates the phase in which a data field is used and then terminated. If there is no letter, the field is not used by that phase.

Field	Phase 05	Phase 10	Phase 20	Phase 21	Phase 22	Phase 29	Phase 30	Phase 35	Phase 40	Phase 41
OPTONS	X	Y	Y	Y	Y	Y	Y	Y	Y	Z
BGNCOL	X	Y					Z			
OPTCNT	X					Z				
ENDCOL	X	Y								
SEQCNT		X						Y		Z
MODULE		X			Y	Z				
TITLEN		X				Z				
ESLCNT		X			Y	Z				
ESLMAX *	X	Z								
CATGVL	X				Z					
COML4V	X				Z					
COML5V	X				Z					
SYTBSZ *	X		Z							
INTLCR			X		Z					
MAXLCR			X		Z					
SYMOVS			X	Z						
NSTENT			X	Y	Z					
MAXSTE			X	Y	Z					
SYTBSA			X	Y	Z					
SYTBEA			X	Y	Z					
SYMCNT			X	Y	Z					
ESLTBA					X	Z				
PAGCNT						X	Y	Y		Z
LPSIZE						X	Z			
LPCLSW						X	Z			
SQFDST						X	Z			
SQFDLG						X	Z			
MNESCT							X	Y		Z
ERRCNT							X	Y		Z
XRPASR									X	Z
XRFRST									X	Z
XRLAST									X	Z
XRBLKC									X	Z

* Model 12 and Model 15 only

Figure 18. Assembler Common Interphase Usage Chart

$ASPRC MODULE

The $ASPRC Module is loaded by Phase 29 and remains in main storage through the end of the assembly process. The contents of this module are illustrated in Figure 19.

<table>
<tr><th>Model 10 $ASPRC</th><th>Model 12 and Model 15 $ASPRC</th></tr>
<tr><td rowspan="2">PRINTER DTF</td><td>PRINTER DTF</td></tr>
<tr><td>HEADER AREA</td></tr>
<tr><td rowspan="2">PRINT BUFFER
(on X'7C' boundary)</td><td>PRINT BUFFER
(on X'7C' boundary)</td></tr>
<tr><td rowspan="3">HEADER AREA</td><td>SYSPRT/SYSLST PARM LIST</td></tr>
<tr><td>PRINT IOB</td></tr>
<tr><td>SYSPRT module load area</td></tr>
</table>

Figure 19. $ASPRC Module

SYMBOL TABLE

The Symbol Table consists of 10-byte entries for each symbol found. It is built by Phase 20 in the lower end of main storage and is designed to fill the entire space between Phase 20 coding and the work file disk buffer. See Figure 20. The table is used by Phase 21 (if overflow occurs) and Phase 22.

See Figure 21 for a detailed description of each 10-byte entry in the symbol table.

<table>
<tr><td>$ASROT</td></tr>
<tr><td>PHASE 20 CODE</td></tr>
<tr><td>SYMBOL TABLE</td></tr>
<tr><td>DISK BUFFERS</td></tr>
</table>

Figure 20. Phase 20 Main Storage Diagram

SYMBOL TABLE DATA AREA FORMAT

Name	Bytes (10 total)	Description
SYMATR	1	Symbol attributes are indicated in this field. Possible settings are: 1. ABSIND (X'01') indicates an absolute attribute 2. RELIND (X'00') indicates a relocatable attribute 3. EXTIND (X'30') indicates an EXTRN symbol 4. SYMRES (X'02') indicates symbol is resolved 5. EXTDEF (X'80') indicates EXTRN definition symbol (Model 12 and Model 15 only)
SYMBOL	2-7	Symbol itself placed in this field
SYMLNG	8	Length attribute of symbol (Length -1)
SYMVAL	9-10	Symbol value

Figure 21. Data Area Format for each 10-byte entry in the Symbol Table

WORK FILE ($WORK2)

The Work File is used by the assembler for intermediate disk storage throughout the assembly process. Figure 22 shows the relative locations of the data in the work file. (Note that the cross reference data is written into the same area as the intermediate text and symbol table, and requires no additional space.)

See *Appendix A* for a description of record formats for the four types of data in the work file.

Figure 23 is a work file activity chart.

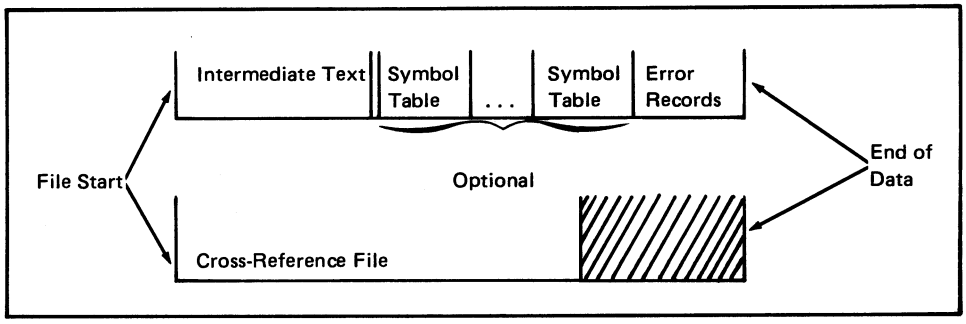


Figure 22. Work File

Phase	Function	Data Activity
05	Initialize	Put intermediate text
10	Compression	Put intermediate text
20	Build symbol table; assign location counter	Update intermediate text (on disk) with location counter value and error records
21	Process overflow symbol table	Update symbolic terms Add symbol table entries Update symbolic terms
22	Substitute symbols	Update symbolic terms Add symbol table entries
29	ESL processing; initialize Phase 30	Prime buffer
30	Assemble object code	Get intermediate text Add error records
35	Diagnostics	Get error records
40	XREF build	Get intermediate text Write XREF blocks
41	XREF merge	Read XREF blocks Write XREF blocks

Figure 23. Work File ($WORK2) Activity Chart

The work file contains four types of data (intermediate text, symbol table entries, error records, and cross-reference data). A description of the record formats for the four types of work file data follows:

Intermediate Text Data

This data is used by the assembler to translate source assembler language to either a relocatable or an absolute object program. The source code is translated to object code during a series of passes. At any point, the intermediate text contains the object program information at its current level of translation.

The intermediate text data is made up of variable length strings of fixed length (10-byte) records. There are four types of records in the file:

1. Control Records – one produced for each instruction in the program. Delimits the variable length string.

2. Name Records – one produced for each name field entry and for all EXTRN names.

3. Term Records – produced from the contents of the operand field. Contain expression terms, constants, and miscellaneous information.

4. Error Records – contain bit settings indicating errors that have been diagnosed.

The size of a string for a source statement is dependent on the statement type and the contents of the operand field. The following rules can be used to determine intermediate text requirements. (The rules apply only to error free source programs. A statement that contains errors generally requires less storage space.)

1. All Instructions
   a. One record for each machine or assembler instruction or comment statement in the program.
   b. One record if there is a name field entry.

2. Machine Instructions – one additional record for each term in the operand field.

3. Assembler Instructions
   a. END, ENTRY, EQU, EXTRN, ORG, USING – one additional record for each term in the operand field.
   b. EXTRN, ISEQ, PRINT, SPACE, START – one additional record for each instruction.
   c. TITLE – additional records = N/8 (plus one for any nonzero remainder), where N is the number of characters in the title operand field.
   d. DS/DC
      1) One additional record for duplication factor (default or specified value).
      2) One additional record for each term in the length specification.
   e. DC
      1) Address Constant – one record for each term in address expression.
      2) All other constants – additional records = N/8 (plus one for any nonzero remainder), where N is the number of bytes required to contain the converted constant plus one.

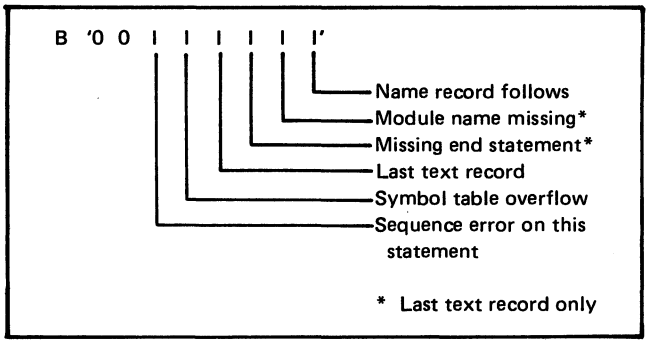
Figure 24 is a sample list of instructions together with the intermediate text space requirements for each.

Label	Instruction	Text Space
DECK	START 0	3
ENTRY	SLC A(2),A	5
	MVC A(2),CON1	4
	ALC A(2),CON2	4
	HPL X'FF',X'FF'	3
A	DS CL2	4
CON1	DC IL2'500'	5
CON2	DC IL2'320'	5
	END ENTRY	2

Figure 24. Intermediate Text Space Requirements

Control Record Format

Byte 1 Record TYPE and indicators



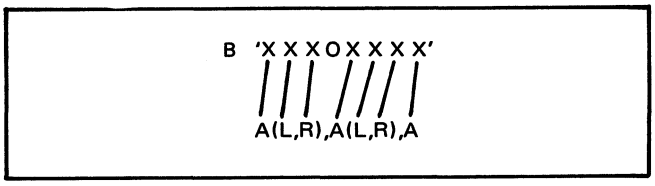
Byte 2 Reserved

Byte 3 Object code length for machine instructions

Byte 4 Operation code (machine code for machine instructions)

Byte 5 Q code (machine code for machine instruction)

Byte 6 Operand format. Bit on indicates operand element as shown.

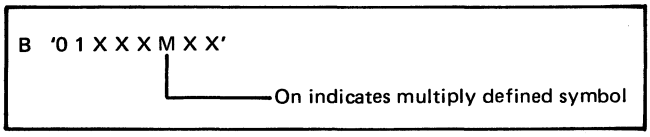


Byte 7, 8 Location counter value assigned before this instruction

Byte 9, 10 Statement number of this instruction

Name Record Format

Byte 1 Record type and indicators



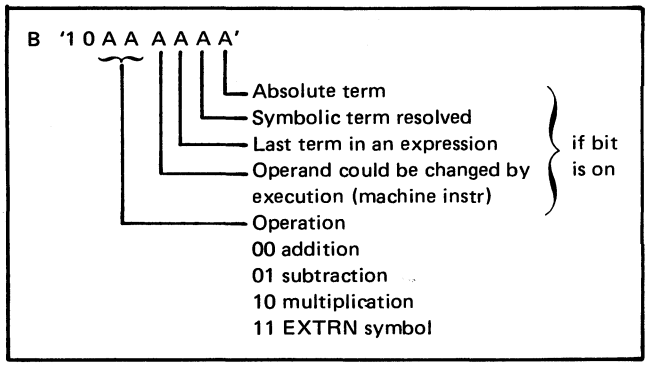
Byte 2, 7 Symbol padded with blanks

Byte 8, 9 Statement number where symbol is defined

Byte 10 EXTRN number for external symbol, zero otherwise

Term Record

Byte 1 Record type and term attributes



If byte 2 contains X'1E', a DC constant term, then bytes 3, 10 contain converted constants.

If byte 2 contains X'1F', a TITLE operand term, then bytes 3, 10 contain TITLE operand character string.

If byte 2 contains X'0F', a location counter reference, then bytes 3, 7 are not used, byte 8 contains length attribute (0), and bytes 9, 10 contain term value.

If byte 2 contains X'3F', a self-defining term, then bytes 3, 7 have miscellaneous uses, byte 8 contains length attribute (0), and bytes 9, 10 contain term value.

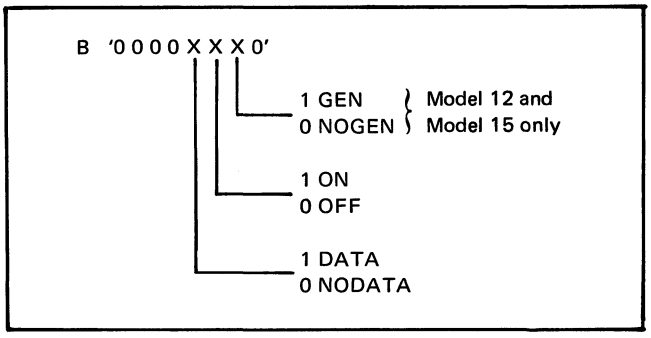
If bytes 2-7 contain a symbolic term padded with blanks, then byte 8 contains length attribute of term, and bytes 9, 10 contain term value.

The following special terms are created with a self-defining term indicator.

DC/DS — 1st term record contains constant type in byte 5 and duplication factor in byte 10.

SPACE — Space operand value in byte 10.

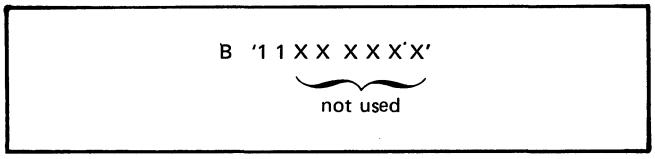
PRINT — Current print control settings in byte 8.



ISEQ — Sequence ISEQ specification
Bytes 7, 8—Sequence field length minus 1
Bytes 9, 10—Sequence field start column

Error Record

Byte 1 — Record Type



Byte 2 — Name field error indicator bits

Byte 3 — Operation field error indicator bits

Byte 4, 6 — Operand field error indicator bits

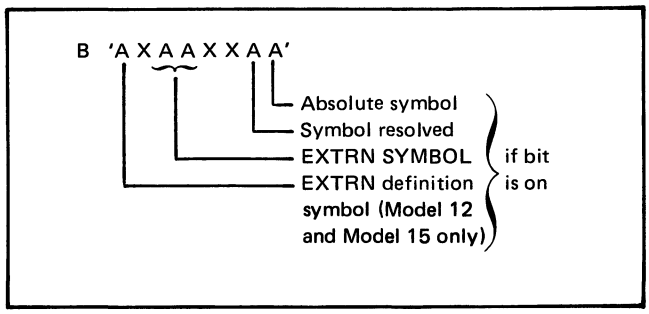
Byte 7, 10 — Reserved

Symbol Table Entries

Symbol table entries are added to the work file after the intermediate text only if a cross reference listing is requested. The symbol table entries are also 10-byte records.

Record format for symbol table entries:

Byte 1 — Symbol attributes



Bytes 2, 7 — Symbol padded with blanks

Byte 8 — Length attribute of symbol

Bytes 9, 10 — Value of symbol

Error Records

Error records are added to the work file as they are generated in Phase 30. They are also 10-byte records. The error-record format is:

Bytes 1, 2 — Statement number in error

Bytes 3, 8 — Error field bytes 2 through 6 of intermediate text error record

Bytes 9, 10 — Reserved

Cross-Reference Data

The cross-reference data is created from the intermediate text and symbol table entries in the work file. The blocks of the cross-reference data are rewritten into the work file as they are created. There are three types of records in the cross-reference data: symbol attribute records, symbol definition records, and symbol reference records.

Record formats for the three cross-reference data entries are as follows:

Symbol Attribute Record (translated symbol table entry)

Bytes 1, 6	Symbol padded with blanks
Byte 7	Type and attribute byte X'00AAXXAA'. The attributes are the same as for symbol table entries.
Byte 8	Length attribute
Bytes 9, 10	Symbol value

Symbol Definition Record (translated name record)

Bytes 1, 6	Symbol padded with blanks
Byte 7	Type byte X'01XXXXXX'
Bytes 8, 9	Statement number where defined
Byte 10	Not used

Symbol Reference Record (translated term record)

Bytes 1, 6	Symbol padded with blanks
Byte 7	Type byte X'01XXXXXX'
Bytes 8, 9	Statement number of the reference shifted one bit to the left. The rightmost bit on indicates a machine operand which could be altered by execution of the instruction.
Byte 10	Term type and attribute byte.

CHART NUMBERING

Flowcharts are identified in this publication in the following manner:

- A flowchart consisting of a single page is identified by a unique pair of letters.

  Example: AA, AB, AC

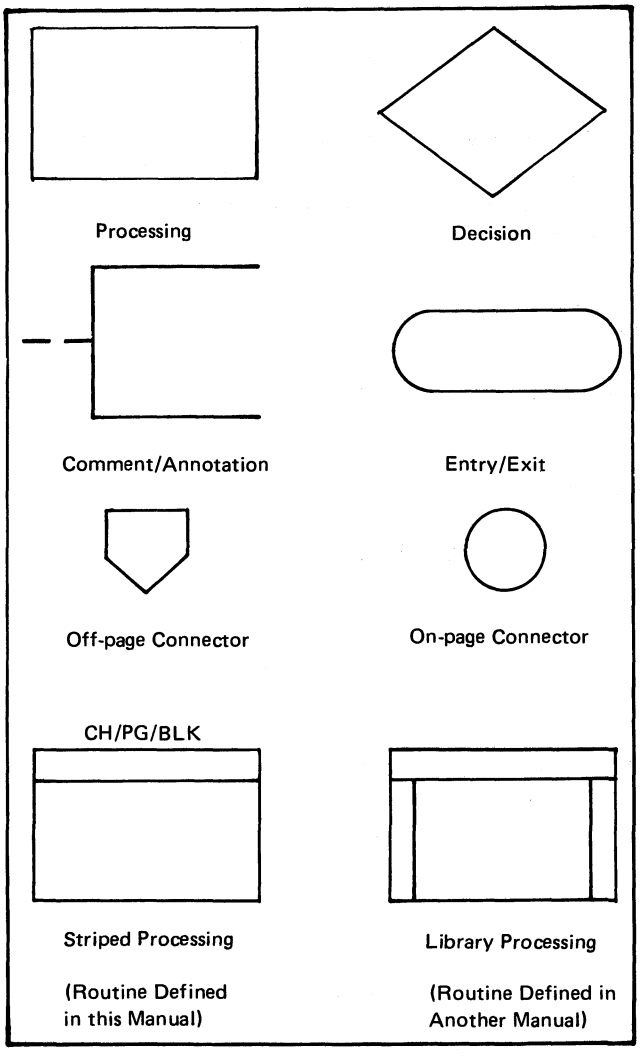
- If a flowchart consists of multiple pages, each page is identified by the same pair of letters, but each page has a unique number.

  Example: First page, CA-01
  Second page, CA-02
  Third page, CA-03, etc.

- Each part has been assigned two sets of flowchart identifying letters. Only after the first set has been completely used, i.e., AA-AZ, is the second set used.

SYMBOLS

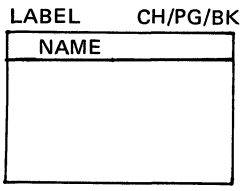
The flowchart symbols used in this PLM are:



Striped Processing Blocks

The striped processing block indicated entry to a module or routine which is flowcharted and/or described in this logic manual.

Example:

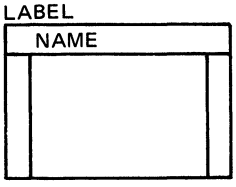


CH/PG/BK indicates the flowchart, page, and block identification where the module or routine is flowcharted. If it is not flowcharted, see the index for the location of the description of that routine.

Library Blocks

A library block indicates a function or module is documented in another manual.

Example:

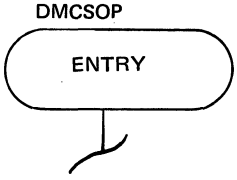


The NAME of the function/module is listed in the Preface of this manual under the name of the manual that contains the description of this function/module.

Entry Block

The label in the upper lefthand corner, just above the symbol, is the entry point in the listing for that part of the program.

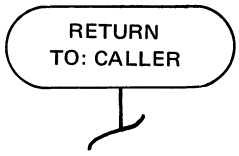
Example:



Exit Block

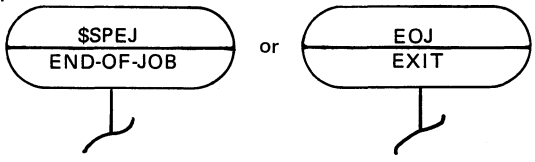
This block indicates that control is leaving this chart.

Example:



If control is being passed to a known function/module, which is documented in another manual, a striped exit block is used. The manual can be found via the Preface as with library blocks.

Example:



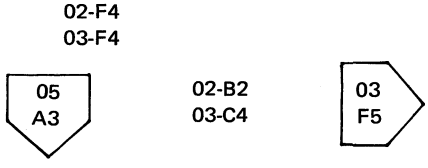
Connectors

Off-page connectors are used to reference between different pages of the same chart ID. Off-page connectors leaving a page contain the page number and block number of their destination.

Example:

Off-page connectors contain the page and block number of their origin. If the entry point referenced by the off-page connector is referenced from more than one origin, all origins are given.

Example:



On-page connectors contain the location of a block on the same page. On-page connectors always contain the location of the destination block.

IBM

Technical Newsletter

This Newsletter No.	LN21-5263
Date	17 April 1975
Base Publication No.	LY21-0504-3
File No.	S3-21
Previous Newsletters	None

IBM System/3 Basic Assembler Program Logic Manual
Program Numbers: 5702-AS1 Model 10 Disk System
5704-AS1 Model 15

This Technical Newsletter, a part of version 12 modification 00 of IBM System/3 Model 10 Disk System Basic Assembler, Program Product Number 5702-AS1, and version 03 modification 00 of IBM System/3 Model 15 Basic Assembler, Program Product Number 5704-AS1, provides replacement pages for the subject publication. These replacement pages remain in effect for subsequent versions and modifications unless specifically altered. Pages to be inserted and/or removed are:

iii, iv
1, 2

Changes to text and illustrations are indicated by a vertical line at the left of the change; new or extensively revised illustrations are denoted by the symbol • at the left of the caption

Summary of Amendments

This Technical Newsletter introduces information concerning the IBM System/3 Model 8.

Note: Please file this cover letter at the back of the manual to provide a record of changes.

IBM Corporation, Publications, Department 245, Rochester, Minnesota 55901

IBM **Technical Newsletter**

This Newsletter No.	LN21-5382
Date	17 March 1976
Base Publication No.	LY21-0504-3
File No.	S3-21
Previous Newsletters	LN21-5263

IBM System/3
Basic Assembler Program
Logic Manual

This technical newsletter, a part of version 01, modification 00 of IBM System/3 Model 12 Basic Assembler (Program Number 5705-AS1) also applies to the IBM System/3 Model 8, IBM System/3 Model 10 Disk System and the IBM System/3 Model 15 basic assembler programs. This technical newsletter provides replacement pages for the subject publication. These replacement pages remain in effect for subsequent versions and modifications unless specifically altered. Pages to be replaced are:

Cover, edition notice	65 through 70
iii, iv	75, 76
1 through 6	83, 84
9 through 20	89, 90
33, 34	95 through 98
45 through 48	107 through 114
51, 52	117, 118
57 through 62	123, 124

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

Summary of Amendments

- Adds Model 12 Basic Assembler Program information (Program Number 5705-AS1)
- Several minor changes

Note. Please file this cover letter at the back of the manual to provide a record of changes.

IBM Corporation, Publications, Department 245, Rochester, Minnesota 55901

Printed in U.S.A.

Technical Newsletter

This Newsletter No. LN21-5469
Date 31 December 1976

Base Publication No. LY21-0504-3
File No. S3-21

Previous Newsletters LN21-5263
LN21-5382

IBM System/3
Basic Assembler Program
Logic Manual

This technical newsletter, a part of version 05, modification 00 of IBM System/3 Model 15 Basic Assembler Program 5704-AS1 and version 01, modification 00 of IBM System/3 Model 15 Basic Assembler Program 5704-AS2, provides replacement pages for the subject publication. These replacement pages remain in effect for subsequent versions and modifications unless specifically altered. Pages to be inserted and/or removed are:

Cover, Edition Notice
1, 2
9, 10
61, 62
111 through 114
Cover
Reader's Comment, Business Reply

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

Summary of Amendments

- Support of IBM System/3 Model 15 Basic Assembler Program 5704-AS2
- Miscellaneous changes

Note: Please file this cover letter at the back of the manual to provide a record of changes.

IBM Corporation, Publications, Department 245, Rochester, Minnesota 55901