SY33-8567-1

Systems

DOS/VSE Assembler Logic

Program Number 5745-SC-ASM

IBM

Second Edition (March 1979)

This is a major revision of, and obsoletes, SY33-8567-0 and all subsequent TNLs. Changes to the text and to illustrations are indicated by a vertical line to the left of the change.

This edition applies to DOS/VSE and to all other releases until otherwise indicated in new editions or Technical Newsletters.

Changes are continually made to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest *IBM System/370 Bibliography,* Order No. GC20-0001 for the editions that are applicable and current.

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

A form is provided at the back of this publication for reader's comments. If the form has been removed, comments may be addressed to *IBM Nordic Laboratory, Product Communications, Box 962, S-181 09 Lidingö 9, Sweden.* IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Preface

Purpose of the Manual

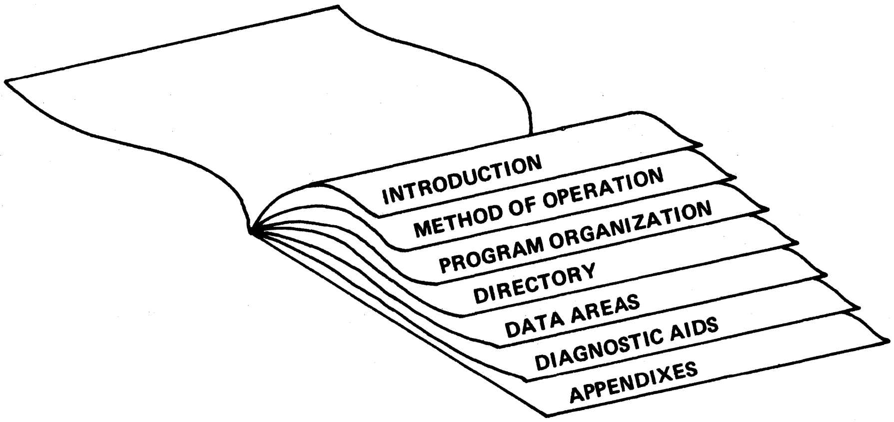
The purpose of the manual is to aid Programming Systems Representatives locate and circumvent faults in the DOS/VSE Assembler, and to assist system programmers with fixing or altering the program design. The manual describes the logic, structure, and operation of the assembler and is to be used as a complement to the program listings.

HOW THE MANUAL IS ORGANIZED

The manual is divided as follows:

- Part 1 - describes the logic of the DOS/VSE Assembler.
- Part 2 - describes the logic of the ESERV (De-edit) program.

Each part of the manual is divided into sections and appendixes as shown below:



Note: Part 2 of the manual has no Directory section.

The following is a brief description of the various sections.

"Introduction" contains a summary of general information about the program, such as size, purpose, environmental characteristics, physical considerations, and operational considerations.

"Method of Operation" describes the logical functions of the program.

Diagrams are used to show input, processing, and output of the functions and subfunctions; each diagram is accompanied by an extended description and cross-references to the program listings.

"Program Organization" describes how the program is divided up into units. This section contains a phase/control section/object module directory, a summary of the functions of each phase, control and data flow, allocation of main storage for the phases, main storage layouts of the phases, and the common data area for the assembler.

"Directory" contains cross-references between the program's control sections, entry points, routine names, module names, and method-of-operation diagrams.

"Data Areas" contains detailed layouts of the program's data areas. It also describes table and dictionary formats.

"Diagnostic Aids" contains information on debugging the assembler, I/O activity and workfile formats for each phase, and register usage.

"Appendixes" includes information on error messages, macro and COPY code usage, reverse Polish notation element formats, pseudo operation codes, internal character set, edited text flags, edited statement formats, statements modifying data areas, and APAR documentation.

USING THE MANUAL FOR THE FIRST TIME

Read through the sections in the following order:

- Read the Introduction.
- Read the introductory material for the "Method of Operation" section in order to get a good idea of how to read the method-of-operation diagrams and extended descriptions; then study the main functional flow of the program through the diagrams and descriptions.
- Study the figures in the "Program Organization" section to learn how the program is physically structured.
- Continue reading the remaining sections in order to orient yourself for quick reference to the pertinent information on the manual.

PREREQUISITE READING

Effective use of this manual requires the reader to have an understanding of the material in the following publications:

IBM System/370 Principles of Operation, Order No. GA22-7000, which contains information on IBM System/370 machine operations, storage and register addressing, and the functions and formats of machine language instructions.

OS/VS-DOS/VSE-VM/370 Assembler Language, Order No. GC33-4010, which contains information on the functions and formats of assembler language instructions, and the coding of macro definitions and instructions.

Guide to the DOS/VSE Assembler, Order No. GC33-4024, which contains information on the assembler options, program listings, complete descriptions of the input and output, and shows how to execute the assembler.

Part 1: DOS/VSE Assembler Logic

Organization of Part 1

- Introduction
- Method of Operation
- Program Organization
- Directory
- Data Areas
- Diagnostic Aids
- Appendixes

Contents

Table of Contents for Method of Operation Diagrams

(see Part II page 51)

Figures

Introduction

The DOS/VSE Assembler is the system control program assembler language translator for DOS/VSE. The language processed is a subset of the language supported by System/370 OS Assemblers and is documented in OS/VS - DOS/VSE - VM/370 Assembler Language. All System/370 instructions are supported by the DOS/VSE Assembler.

Size of the Assembler

The minimum virtual partition size required by the DOS/VS Assembler is 24K.

Purpose and Function of the Assembler

The purpose of the DOS/VSE Assembler is to translate source programs written in the DOS assembler language into object modules suitable for processing by the DOS Linkage Editor. The assembler performs three major functions in processing source programs: (1) expansion of macro definitions called by macro instructions, (2) assembly of machine instructions into object code, and (3) processing of assembler instructions.

The assembler also produces edited macro definitions suitable for cataloging on a source statement library.

Environmental Characteristics

SYSTEM CONFIGURATION

The minimum configuration required by the assembler is the same as that for the DOS control program: one disk drive, one card reader/punch, and a printer. The following data sets are used by the assembler:

SYSRES	Disk
SYSIPT	Card, Tape, or Disk
SYSSLB (optional)	Disk
SYSLST (optional)	Printer, Tape, or Disk
SYSPCH (optional)	Card, Tape, or Disk
SYSLNK (optional)	Disk
SYS001, SYS002, SYS003	Disk

DEVICE NEEDS

The assembler requires devices for SYSRES, SYSIPT, and the three workfiles SYS001, SYS002, and SYS003. Other devices are needed only if the data sets are specified by their corresponding assembler option.

SYSTEM INTERFACES

System-dependent functions and operations of the assembler (interfaces between the assembler and the system) are centralized in interface modules to allow relative ease of modification for new features of the Disk Operating System. The names and functions of these interface modules are listed below.

IPKAA	Basic system interface routines (workfile I/O and subroutine call routines), common data area (COMMON)
IPKAB	SYSSIPT, SYSSLB input routines
IPKAC	SYSPCH routine for EDECK output
IPKAD	SYSSLB logic module (DTFSL)
IPKAE	SYSSLB routines for reading edited macros from SYSSLB
IPKAF	SYSPCH/SYSLNK output routines
IPKAG	SYSIPT logic module (CPMOD)
IPKAH	SYSPCH/SYSLNK/SYSLST logic module (CPMOD)
IPKAI	SYSLST output routine

Interface macros used by the assembler to provide service functions and to call for functions from interface modules are described in Appendix C, "Macro Usage."

Physical Considerations

The assembler is made up of 19 phases residing on a core image library. See "Program Organization" for a table showing the phases, control sections, and object modules of the assembler.

Operational Considerations

INPUT

Input to the assembler is as follows:

Source code	SYSIPT
COPY code (sublibrary A on a source statement library)	SYSRES/SYSSLB
Edited macro definitions (sublibrary E on a source statement library)	SYSRES/SYSSLB

Sublibraries A and E are optional. One private library may be used in addition to the system library. For a complete description of the input see Guide to the DOS/VSE Assembler.

OUTPUT

Assembler output is as follows:

Object modules	SYSLNK/SYSPCH
Source macro definitions (in edited format)	SYSPCH
Program listing	SYSLST

The output is controlled by specifying assembler options. For a complete description of the output see Guide to the DOS/VSE Assembler.

CONTROL INFORMATION FOR THE ASSEMBLER

The user specifies options for the assembler in the OPTION job control statement. For a description of the assembler options see Guide to the DOS/VSE Assembler.

Special Features of the Assembler

MACRO LIBRARY AND COPY LIBRARY

The DOS/VSE Assembler uses two sublibraries of the source statement library: (1) the macro library, containing macro definitions in an edited format, and (2) the COPY library, containing sequences of assembler language instructions and/or macro definitions in source format. Because the macro definitions in the macro library are edited, the assembler is relieved from editing and syntax checking the macro definitions when they are called by macro instructions from an assembler program.

ESERV (DE-EDIT) PROGRAM

The ESERV program translates edited macros back into their source format. This "de-editing" may be optionally combined with an update of the macro. The logic of the ESERV program is described in Part 2 of this manual. For a complete description of how to use the program see Guide to the DOS/VSE Assembler.

Method of Operation

Purpose of the Section

The purpose of this section is to give a functional description of the assembler and to provide a cross-reference from any given diagram to other parts of the manual and the program listings.

HOW THE SECTION IS ORGANIZED

This section consists of diagrams showing the functions and subfunctions of the assembler. These diagrams are arranged in a hierarchy as illustrated in the foldout, Figure 46, at the back of the manual. **(Please open Figure 46 and use it as a guide to the diagrams.)** With each diagram is an "Extended Description" containing detailed information about the function or subfunction.

HOW TO READ THE DIAGRAMS AND DESCRIPTIONS

Each diagram illustrates:

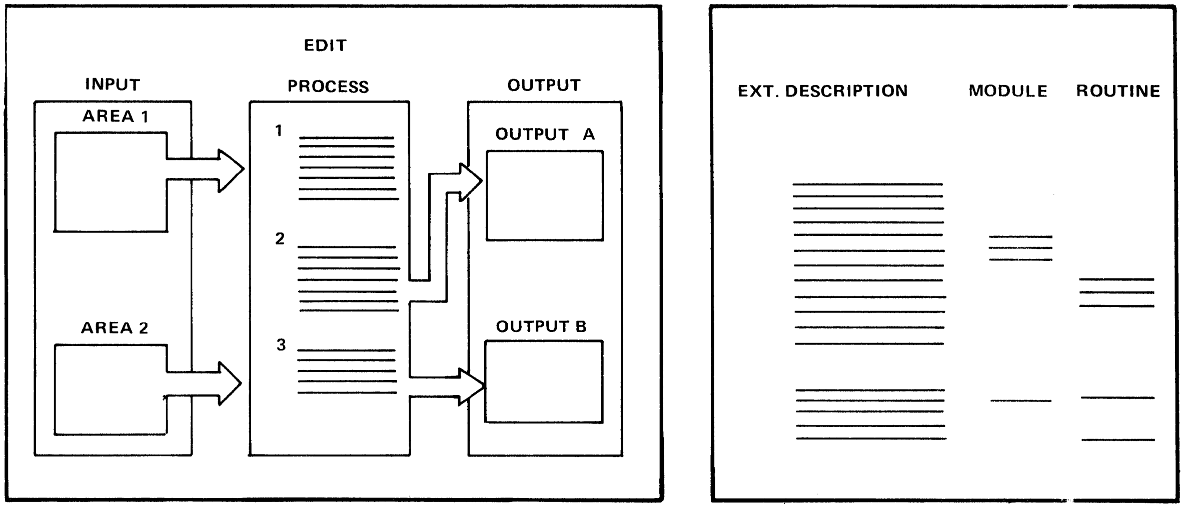
- Input - showing what the data is and where it is from
- Process - describing how the data is processed by the assembler
- Output - showing where the data goes

Data areas are identified on the diagrams in two ways: main-storage address (upper case, parenthesis), and by DSECT name (upper case, underlined). Data areas as shown on the diagrams are highly schematic. For complete and accurate data area layouts see "Data Areas".

The extended descriptions are related to the diagrams by numbered process steps. In addition, the extended descriptions give the names of the module and routine(s) which perform the function.

Many of the data areas and routines are mentioned in two or more diagrams. For a cross-reference of these data areas, the diagrams in this section, and the program listings use the "Directory". The Directory also cross-references the appropriate microfiche card if you wish to go directly to the listings.

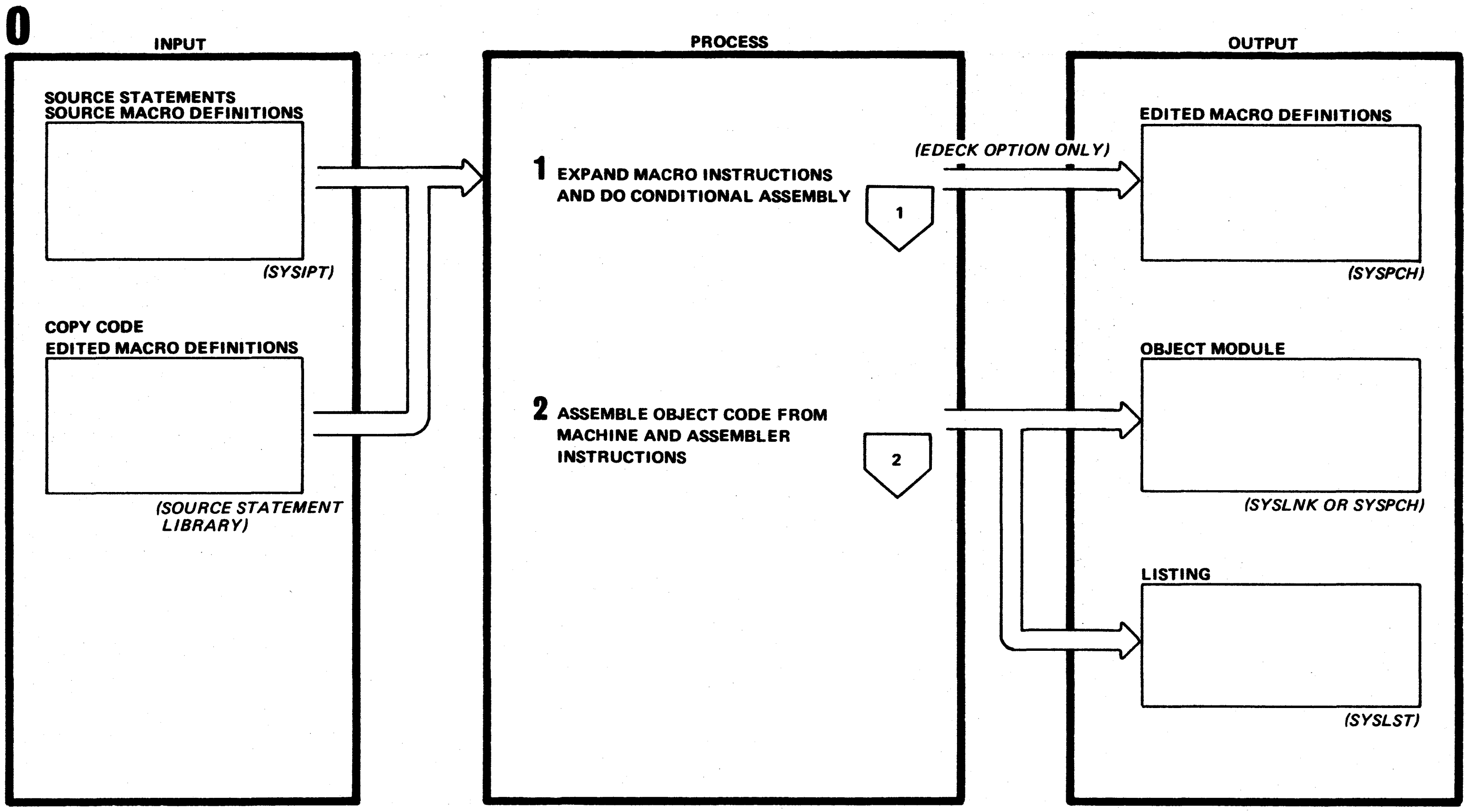
Start reading the process block and refer to the input and output as you proceed through the diagram. Use the extended descriptions if you require more detailed information.



The following symbols are used in the diagrams.

⇒	Data flow	- - - - →	Data reference
→	Pointer	⬠	Reference to another diagram
(MNAENT)	Main storage address		
➡	Control flow	*PSTRINGS*	DSECT name

Translate Source Code into Object Code



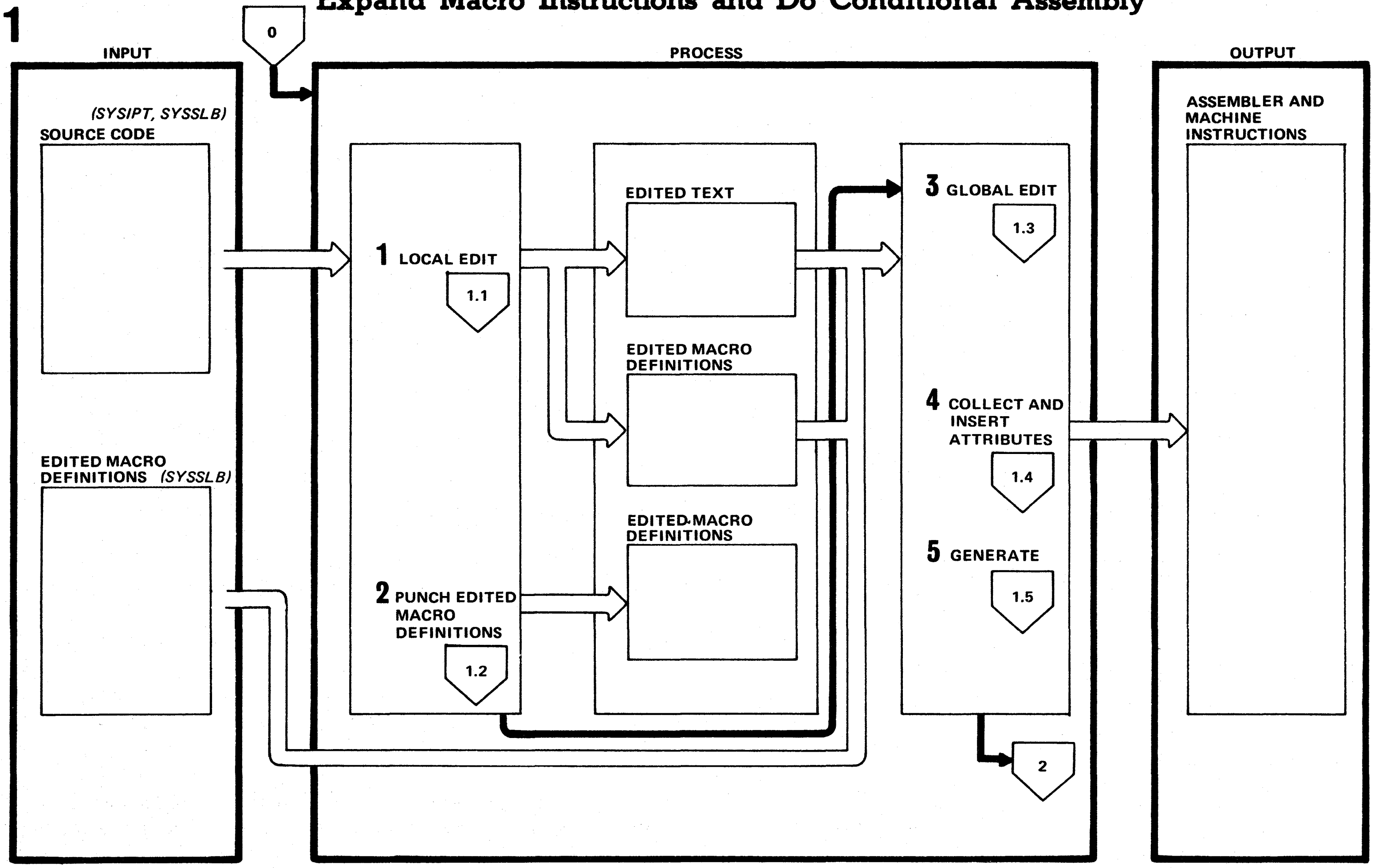
0

EXTENDED DESCRIPTION

1. Source statements are read and macro instructions expanded. Conditional assembly in open code is performed. If the EDECK option has been specified, edited macro definitions can be obtained on SYSPCH.

2. After all macro instructions have been expanded, the assembler and machine instructions are assembled into object code.

1 Expand Macro Instructions and Do Conditional Assembly



1

EXTENDED DESCRIPTION

Because the assembler accepts edited macros, editing proceeds in two stages: local editing and global editing. Local editing involves only local variable symbols. Global editing involves global variable symbols and therefore cannot be done until edited macro definitions have been read in (if they are called). After both local and global editing have been done, the macro instructions can be expanded according to their definitions; conditional assembly in open code is also performed.

1. Source code is read and macro definitions and instructions edited locally. Some editing of machine and assembler instructions is also done.

2. If the EDECK option has been specified, locally edited source macro definitions are punched.

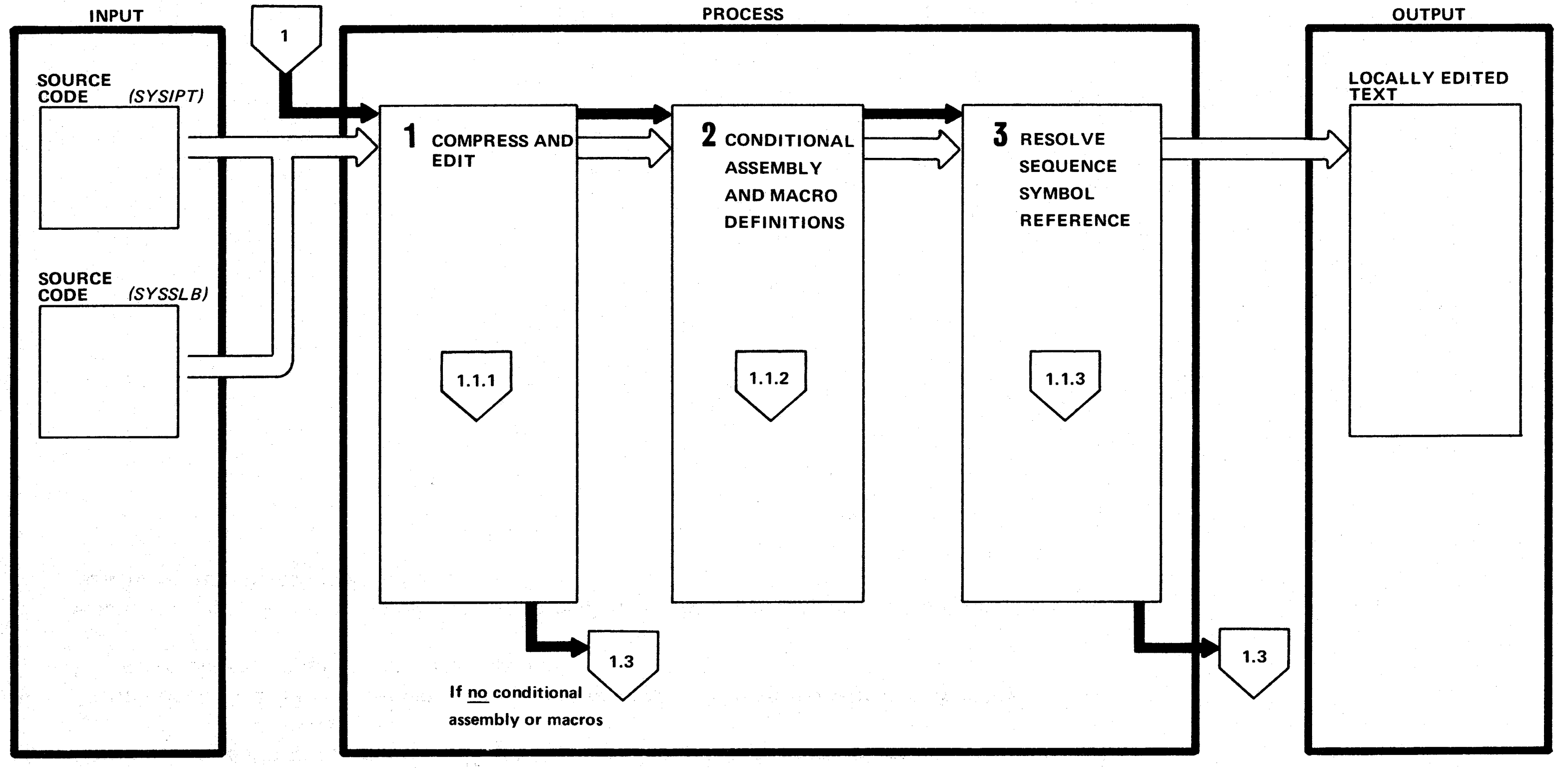
3. Edited macro definitions are read in from the macro library and global editing done.

4. Attributes needed for conditional assembly are collected. The edited text is now ready for macro expansion and conditional assembly.

5. Macro instructions are expanded and conditional assembly is performed. The output is now ready for the assembler phases.

1.1

Local Edit

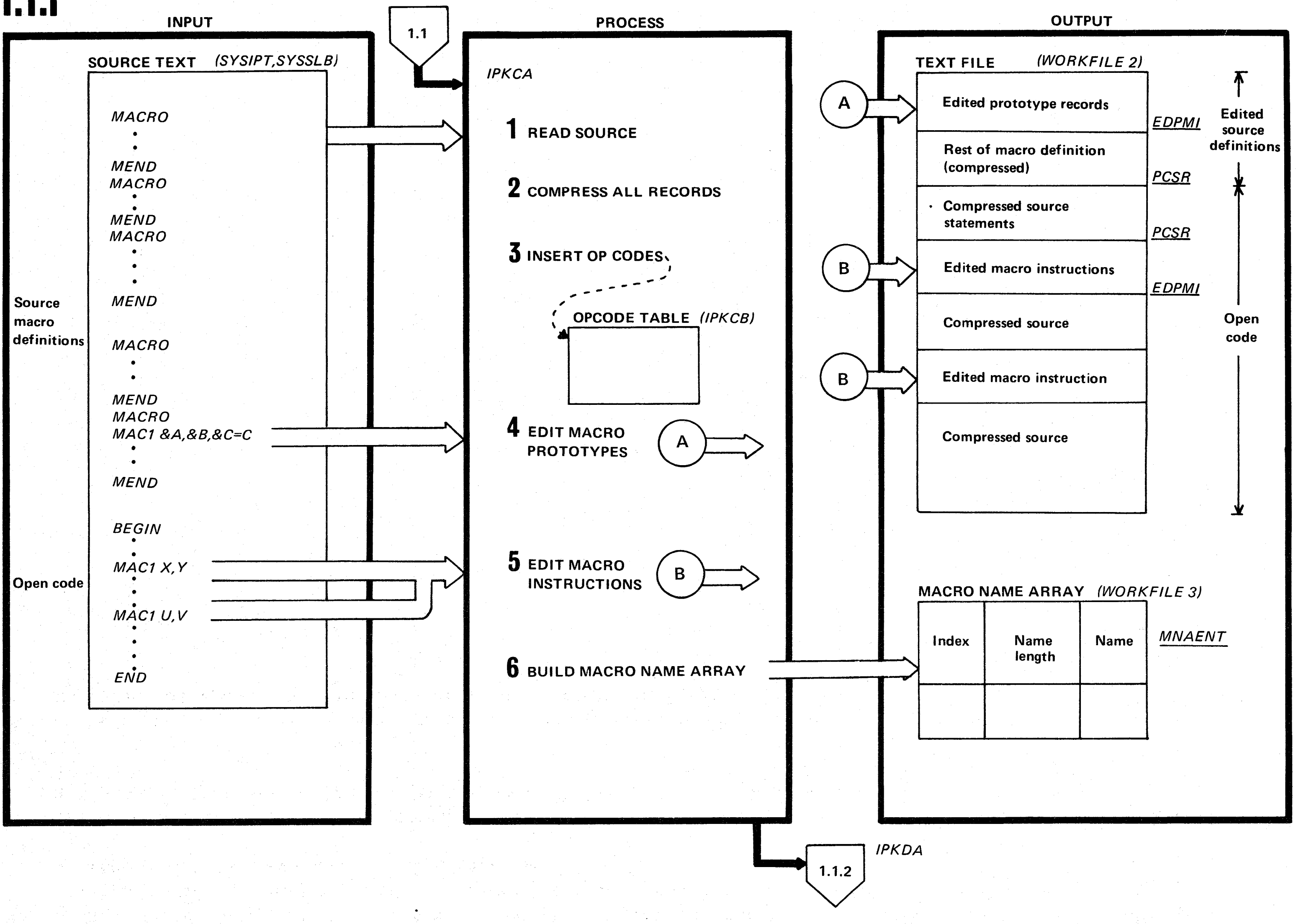


1.1

	EXTENDED DESCRIPTION	MODULE	ROUTINE
1.	In the first pass over the text file, macro instructions and prototypes are edited, opcodes inserted in all records, and all records are compressed. Some editing of macro definitions is also done (see Diagram 1.1.1).	IPKCC IPKCA	MIROUT
2.	In the second pass, macro definitions are edited (see Diagram 1.1.2). Expressions involving conditional assembly are translated to reverse Polish notation (see Diagram 1.1.2.1).	IPKCC	PROROUT
3.	In the third pass, sequence symbol references are resolved (replaced by addresses) and the edited text separated from the compressed source records (see Diagram 1.1.3).	IPKEA	

1.1.1

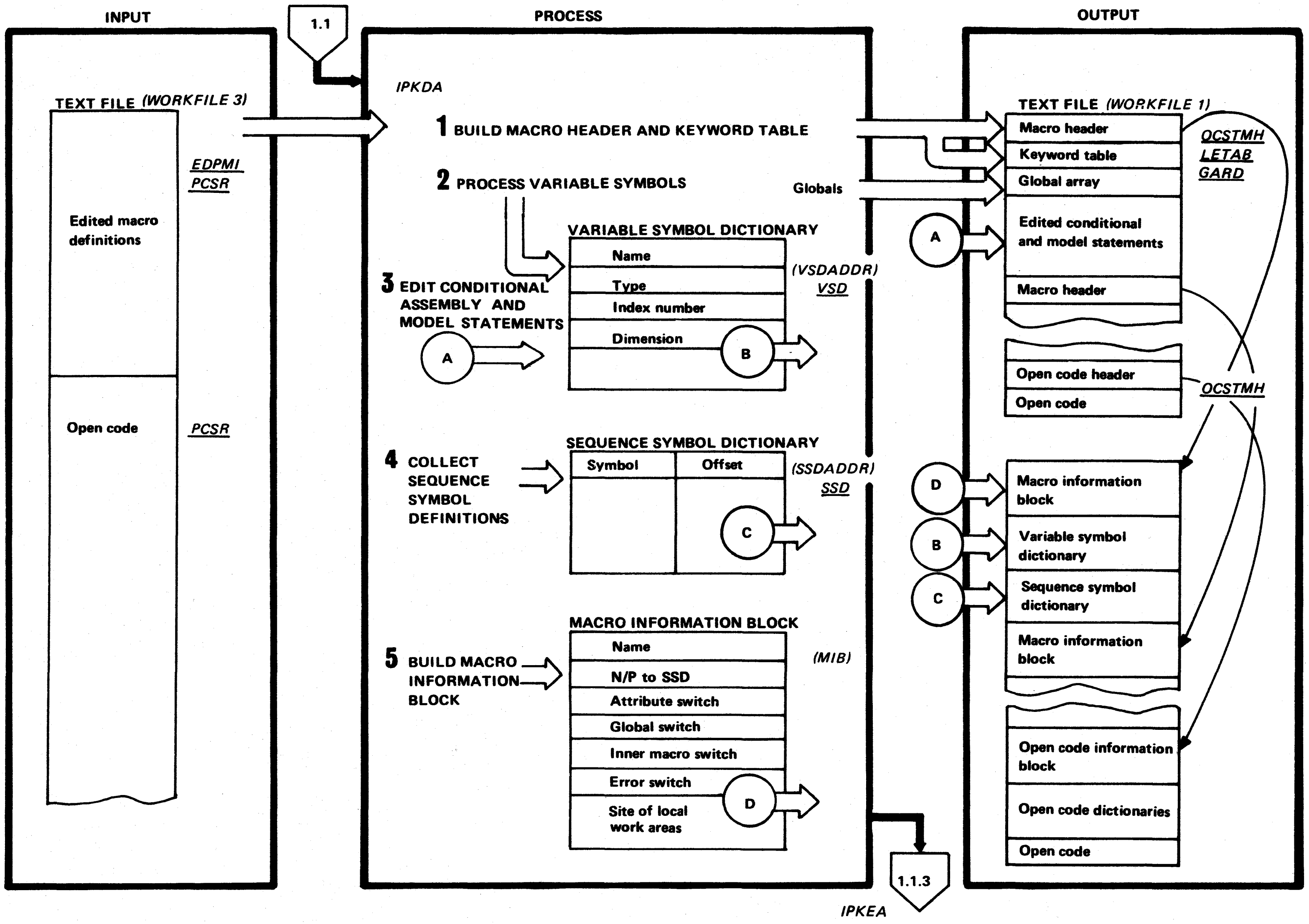
Compress and Edit



1.1.1

	EXTENDED DESCRIPTION	MODULE	ROUTINE
1.	Source statements are read from the system input device (SYSIPT) and from the source statement library. SYSIPT contains source macro definitions and "open code" (in that order) and may contain COPY statements which cause library COPY books to be brought in from sublibrary A on the source statement library.	IPKCA	DRIVER
2.	All records are compressed. Normally the whole statement is contained in one compressed source record.	IPKCB	
3.	At the same time as compression is done, pseudo opcodes are inserted in the record (see Appendix E for the pseudo opcodes).		
4.	Editing of source macro definitions is begun. At least two edited statements are created for each macro type: a header and a text record. The edited prototype records contain the macro name, positional and keyword parameters, and a number of "items".	IPKCC	PROROUT
5.	Macro instructions both in open code and within source macro definitions are partially edited; operands needing substitution are not completely processed until the next text pass (see Diagram 1.1.2).	IPKCC	MIROUT
6.	As macro instructions are edited, a macro name array is built for open code only (inner macro instructions are handled in Global Edit -- Diagram 1.3). The macro name array is built in main storage in blocks whose size is determined by initialization; the blocks are written on workfile 3 when filled. Before a macro is entered it is checked for previous entry.	IPKCC	MIROUT

1.1.2 Edit Macro Definitions and Conditional Assembly

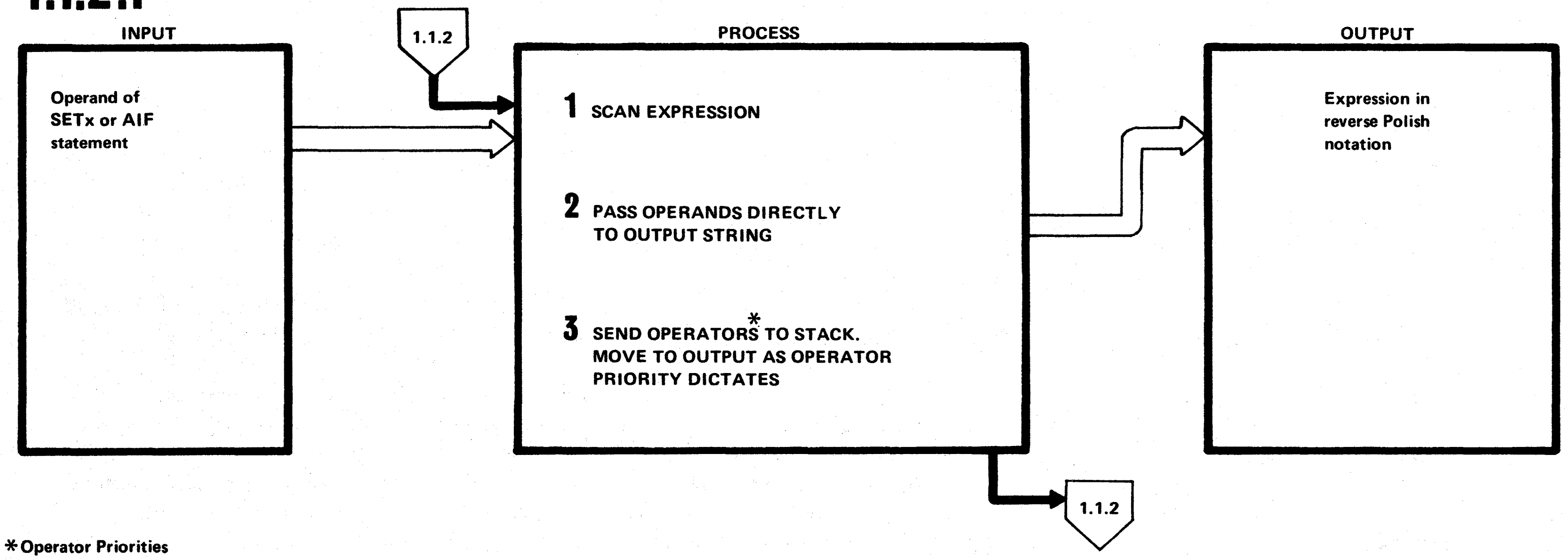


1.1.2

EXTENDED DESCRIPTION	MODULE	ROUTINE
1. The macro header contains an index number for the macro definition and the N/P address of the definition's macro information block (see step 5). The keyword table, consisting of keyword names and their default values, is built from the edited prototype statements for the definition.	IPKDA	NEWST
2. Variable symbols (local, global, and macro parameters) are entered in the variable symbol dictionary (one dictionary for each definition, and one for open code). Global definitions are also entered in a global array for the definition. The global array will later be used in global processing (see Diagram 1.3). The variable symbol dictionary is used in editing the conditional assembly and model statements in the macro definition (step 3). It is also written on workfile 3 for use by the ESERV program if required.	IPKDA	VSDLKP
3. Editing consists of two main functions: all variable symbol references (except sequence symbols -- see step 4) are replaced by their index numbers and all conditional assembly expressions are translated into reverse Polish notation (see Diagram 1.1.2.1). At this point macro instructions requiring substitution in their operand fields are fully edited.	IPKDA	INTERPR
4. Sequence symbols, together with the offset in bytes from the end of the global array to the statement with the symbol in its name field, are collected in the sequence symbol dictionary. References to sequence symbols will be replaced by the offsets in the next text pass (see Diagram 1.1.3). At the end of the macro definition or open code, the dictionary is written on workfile 3 and its position noted in the macro information block or open code information block.	IPKDA	SSDENT
5. A macro information block is built and written on workfile 3 at the conclusion of processing for a macro definition.	IPKDA	

1.1.2.1

Convert Pre-Assembly to Reverse Polish Notation



*Operator Priorities

0	generate, (
1	AIF, AGO, SETx, ACTR
2	OR
3	AND
4	NOT
5	GT, LT, EQ, NE, LE, GE
6	+,-
7	* , /
8	unary minus
9	attributes
10	concatentation of quoted strings
11	concatentation, substrings
12	binary or character parameter, conversion (arithmetic-character, binary character, character-arithmetic, binary-arithmetic)
13	index, subscript, suboperand
15	)

1.1.2.1 EXTENDED DESCRIPTION

Conditional assembly expressions are translated into reverse Polish notation. This is a form which makes it easier for the assembler to evaluate the expression.

MODULE	ROUTINE
IPKDA	INTERPR

1. Expressions are scanned.
2. Operands are assigned identifying flags and are inserted immediately in the output string. Variable symbols are processed as described in Diagram 1.1.2 and pointers to generation-time value areas (dictionaries) entered.
3. Operators are put into a stack according to their priority. The higher the priority, the sooner the operator is inserted into the output string. The first operator encountered is always entered into the stack. All succeeding operators are entered in the stack after a comparision to the previous entered operator. If the priority is lower than the previous entry, the operator is placed in the stack and the next operator is compared. If the priority is higher than or equal to that of the previous entry, the previous operator is removed and placed in the output string; the comparision is then continued with the next operator in the stack.

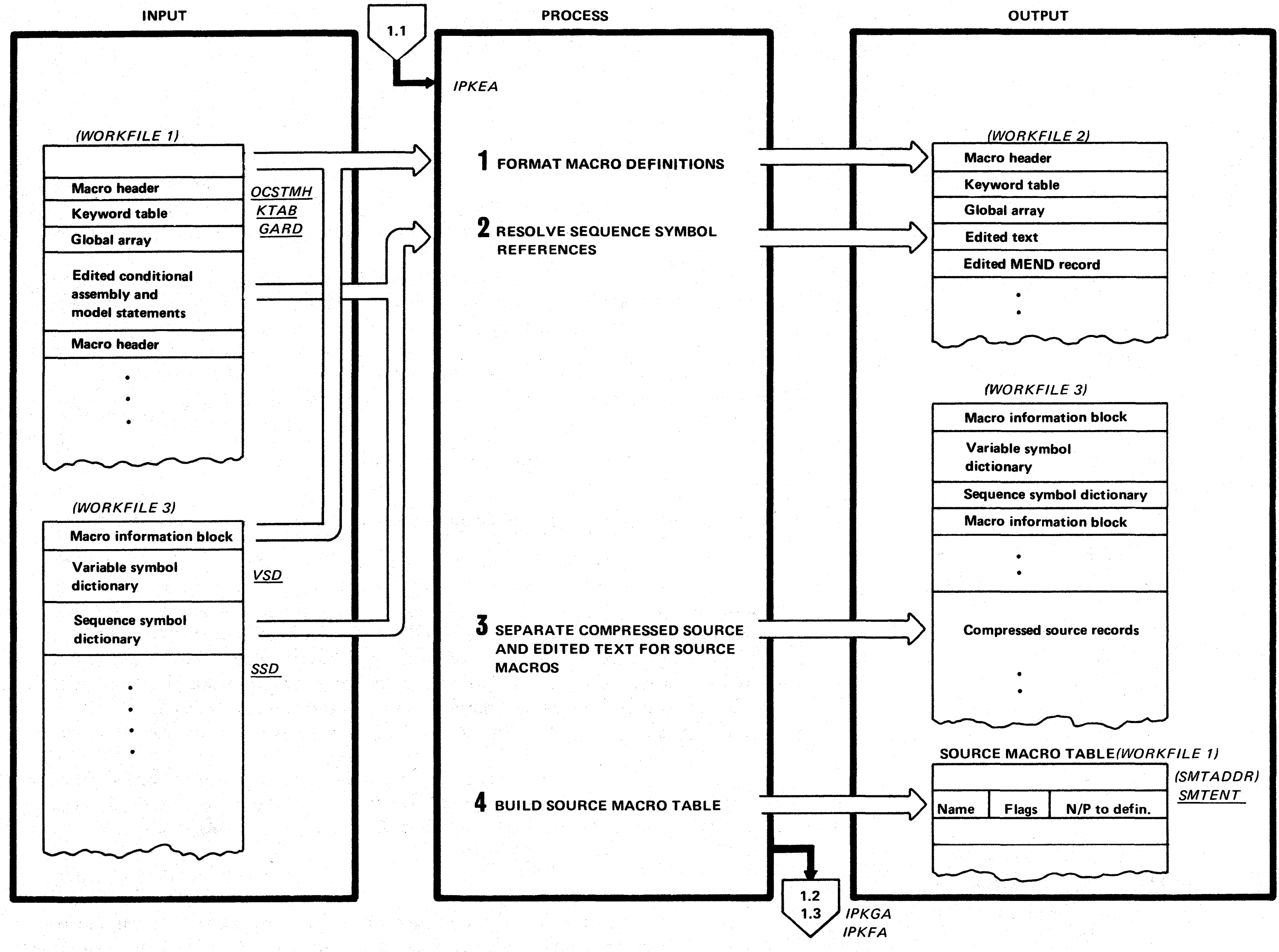
   "Start character mode" and "end character mode" operators are placed immediately into the string.

   There are several exceptions to the processing method:

   Unary plus. Not entered in the stack.
   Unary minus. Before entering in the stack, the previous stack operator is checked. If it is a unary minus, both operators are discarded. Otherwise, the unary minus is entered as normal.
   Left parenthesis. Placed in the stack without comparison of priority.
   Right parenthesis. Causes the stack to be emptied until a left parenthesis is found in the stack. The left parenthesis is also removed.

For evaluation of reverse Polish notation, see Appendix D and Diagram 1.5.2.
For examples of expressions in reverse Polish, see Diagnostic Aids.
For flags, see Appendix G.

1.1.3 Resolve Sequence Symbol References

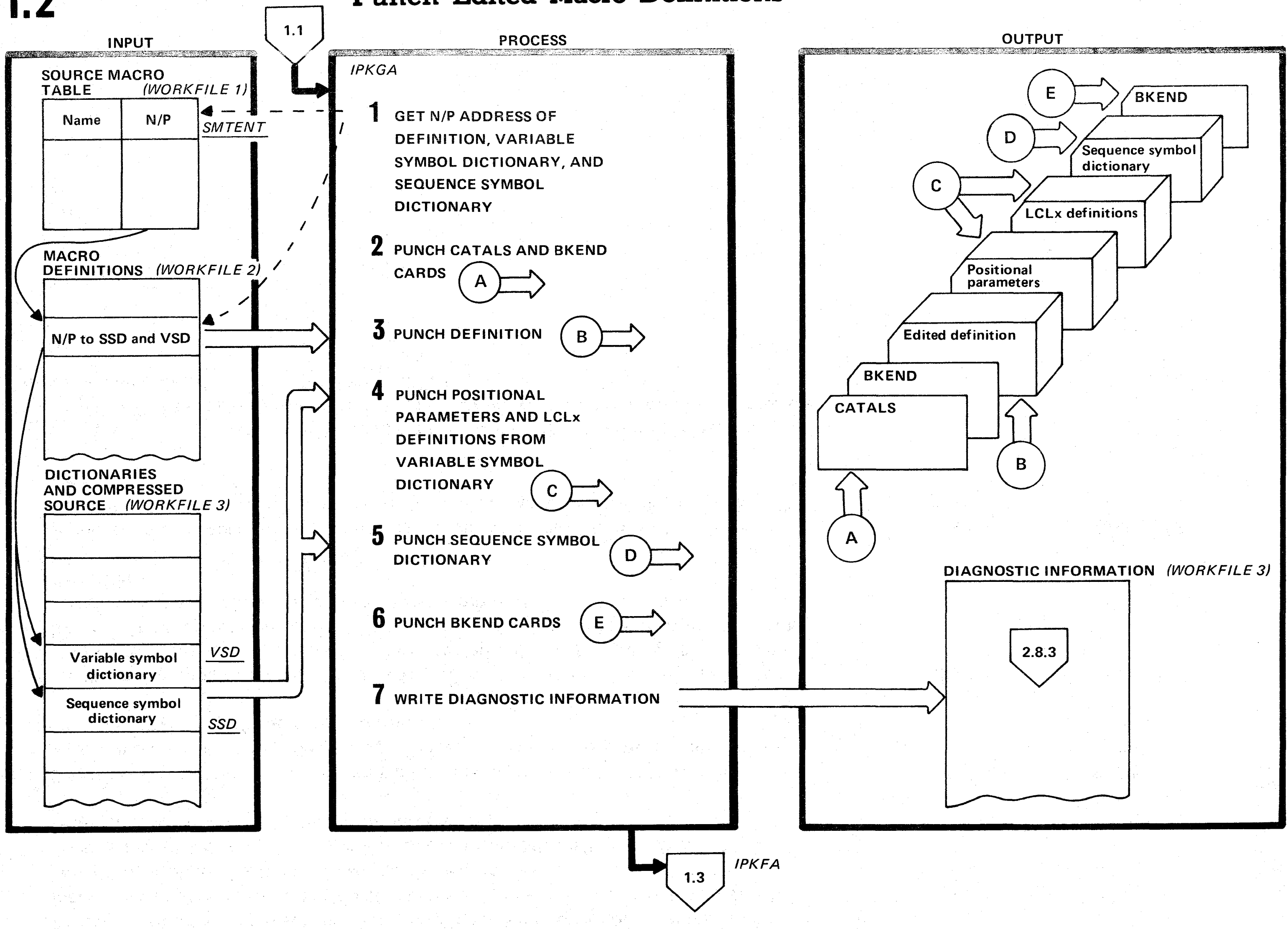


1.1.3

EXTENDED DESCRIPTION	MODULE	ROUTINE
1. The edited macro definitions are put in a form suitable for global editing and/or EDECK output. This involves adding flags (attribute collection required, inner macro instructions present, keyword parameters present, global variables present) and the size of the local value areas to the macro header. The information is obtained from the macro information block(s) on workfile 3. An edited MEND record is also added to the macro definition.	IPKEA	
2. Sequence symbol references are resolved by reading the macro definition together with the sequence symbol dictionary. References are replaced by the byte offset from the beginning of the edited definition.	IPKEA	SSDLKP
3. At this point compressed source records, which have been mixed with the edited records on workfile 1, are separated from the edited text for later printing on the listing. They are written on workfile 3 directly after the macro information blocks, variable symbol dictionaries, and sequence symbol dictionaries for all the macro definitions.	IPKEA	MOVEPUT
4. A source macro table is built for use by Global Edit (and EDECK) (see Diagrams 1.3 and 1.2). It contains the names of all source macro definitions and their N/P addresses.	IPKEA	SMTENTR

Note: The macro information block is kept on workfile 3, but is no longer needed after step 1. The variable symbol and sequence symbol dictionaries are kept for possible use in punching edited macro definitions (EDECK).

1.2 Punch Edited Macro Definitions



1 GET N/P ADDRESS OF DEFINITION, VARIABLE SYMBOL DICTIONARY, AND SEQUENCE SYMBOL DICTIONARY

2 PUNCH CATALS AND BKEND CARDS

3 PUNCH DEFINITION

4 PUNCH POSITIONAL PARAMETERS AND LCLx DEFINITIONS FROM VARIABLE SYMBOL DICTIONARY

5 PUNCH SEQUENCE SYMBOL DICTIONARY

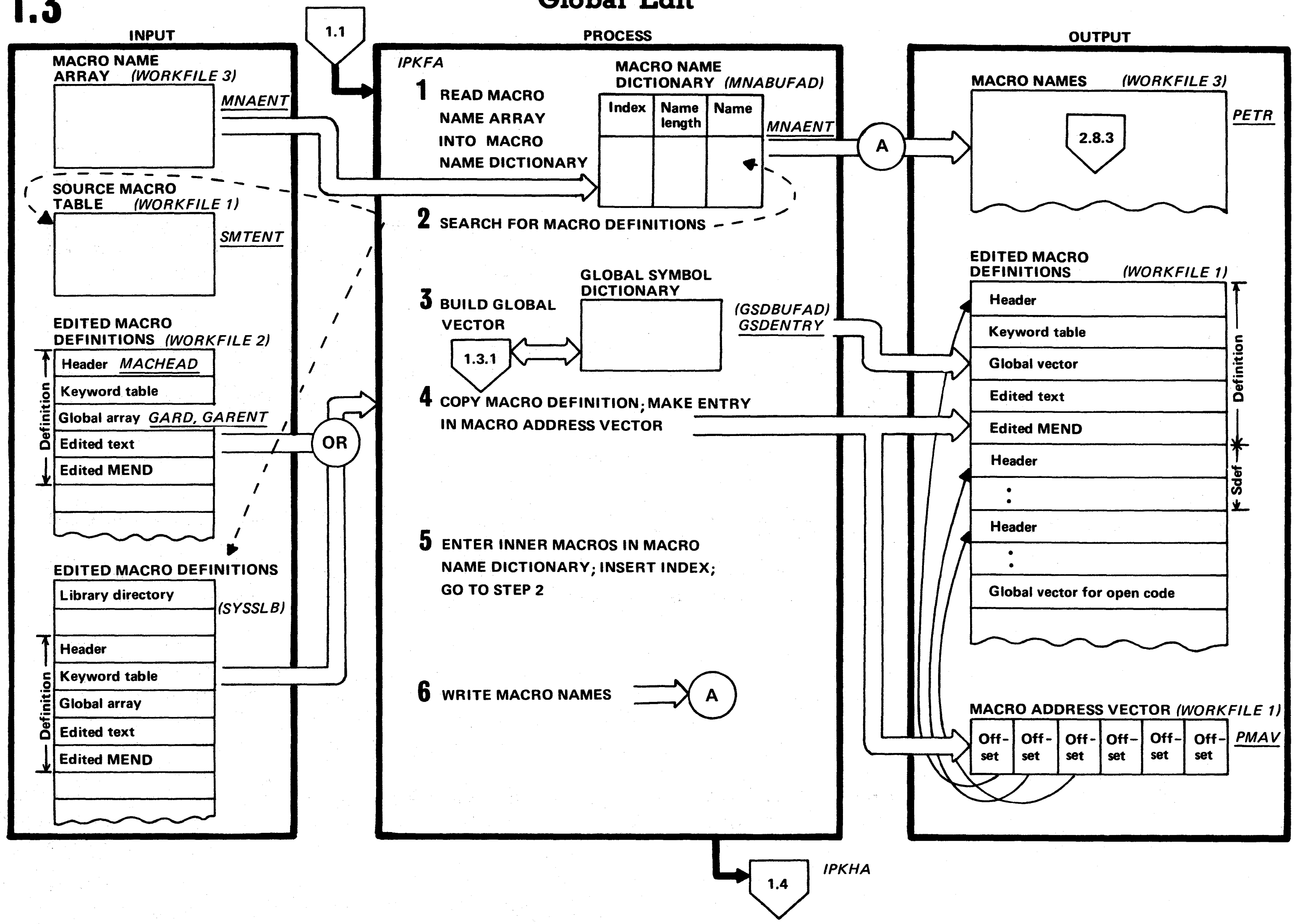
6 PUNCH BKEND CARDS

7 WRITE DIAGNOSTIC INFORMATION

1.2

EXTENDED DESCRIPTION	MODULE	ROUTINE
Immediately after resolution of sequence symbol references, the macro definition is ready to be punched and cataloged, if so desired. (In other words, at this stage it is "pre-edited.")		
1. When the EDECK option of the assembler is in effect, macro definitions without editing errors are punched in their edited form. The source macro table is used to locate the definition and its corresponding variable symbol and sequence symbol dictionaries.	IPKGA	STSMTGET
2. A CATALS and a BKEND card are punched so that the resulting deck can be submitted directly for cataloging in the macro library.	IPKGA	CATALBKE
3. The edited definition is copied from workfile 2.	IPKGA	MPUNCH STGET
4. Immediately after the definition, positional parameters and local variable symbol declarations are punched from the variable symbol dictionary. These are needed so that the macro definition can be reconstructed, if necessary, by the ESERV program.	IPKGA	MPUNCH STGET
5. The sequence symbol dictionary is also punched for the same reasons as above.	IPKGA	MPUNCH STGET
6. A BKEND card is punched.	IPKGA	
7. Diagnostic information (macro name and number of cards punched) is written on workfile 3.	IPKGA	STDIAG

1.3 Global Edit



1.3

The two main functions of global edit are

- Build a global vector for each macro definition and open code
- Build the macro address vector

The global vector is a series of indexes used by Generate (ASSEIA) to compute the address of the global symbol's value in the generation-time global symbol value area (see Diagram 1.3.1).
The macro address vector is a series of offsets used by Generate (ASSEIA) to find a macro definition.

	EXTENDED DESCRIPTION	MODULE	ROUTINE
1.	A workarea -- the macro name dictionary -- is used to account for all macro instructions used in the source program. Initially the macro name dictionary is identical to the macro name array built during the first pass (see Diagram 1.1.1). Thus at first it contains only names of those macro instructions in open code and not the names of those occuring within macro definitions ("inner" macros).	IPKFA	GEINIT
2.	The macro name dictionary is scanned and the corresponding macro definition looked for, first in the source macro table (which contains pointers to source definitions) and then in SYSSLB.	IPKFA	SMTSRCH MLIBSRCH
3.	The global array of each macro definiton is used to build the global vector (see Diagram 1.3.1).	IPKFA	FINDGS GSDENT GVENT CHECKGS
4.	The global vector is then written, along with the macro definition, onto workfile 1. The N/P address (in byte-offset form) of the macro definiton is entered in the macro address-vector. The macro address vector is later used to locate the edited macro definition for expansion (see Diagram 1.5).	IPKFA	MAVENTRY
5.	If a macro instruction is present within a macro definition, its name is entered at the end of the macro name dictionary (if not already there). Processing then continues from step 2. Inner macro instructions are given some editing and are assigned index numbers.	IPKFA	MNDENT MNDSRCH
6.	All names in the macro name dictionary are written on workfile 3. They will be printed later (see Diagram 2.8.3).	IPKFA	GEFIN

Overflow and search techniques

The following techniques are used when there is not enough space in the partition for the macro name dictionary, the source macro table, and the global symbol dictionary.

Macro Name Dictionary (MND)

The MND is operated on in three ways:

1. MNDGET reads macro names from the MND sequentially.
2. MNDSRCH searches the MND for the names of inner macro instructions.
3. MNDENT makes new entries in the MND.

MNDGET reads macro names from MND blocks (overflow blocks on workfile 2). When it has taken all the entries from a block, it reads in the next block.

MNDSRCH searches the MND blocks as follows: if MNDGET is reading block C and the MND is made up of blocks A,B,C, and D, MNDSRCH searches C, reads and searches A, reads and searches B, reads and searches C, reads and searches D. If the name is found, MNDSRCH inserts the index to the macro instruction in the edited text and returns control to the main program. If the name is not found, MNDSRCH calls MNDENT.

MNDENT will always have the last MND block in main storage at MNDENT start. If there is space on this block, MNDENT makes the new entry. If the MND buffer is full, MNDENT writes the last block, creates a new block where it places the new entry, and writes the new block. If necessary, MNDENT then reads into main storage the block that MNDGET was reading.

Source Macro Table (SMT)

The assembler reads the first block of the SMT into main storage at ASSEFA initialization. The SMTSRCH routine searches the block for a macro name corresponding to the entry taken from the MND. If the name is not found, SMTSRCH reads the next block of the SMT and continues the search until the name is found or all the blocks have been read. For example, if SMT block C is in main storage and the SMT is made up of blocks A, B, C, and D, SMTSRCH searches C, reads and searches D, reads and searches A, and reads and searches B. If the macro name is found, SMTSRCH returns to the main program with a pointer to the SMT entry. If the macro name is not found, SMTSRCH returns to the main program with the pointer set to zero.

Global Symbol Dictionary (GSD)

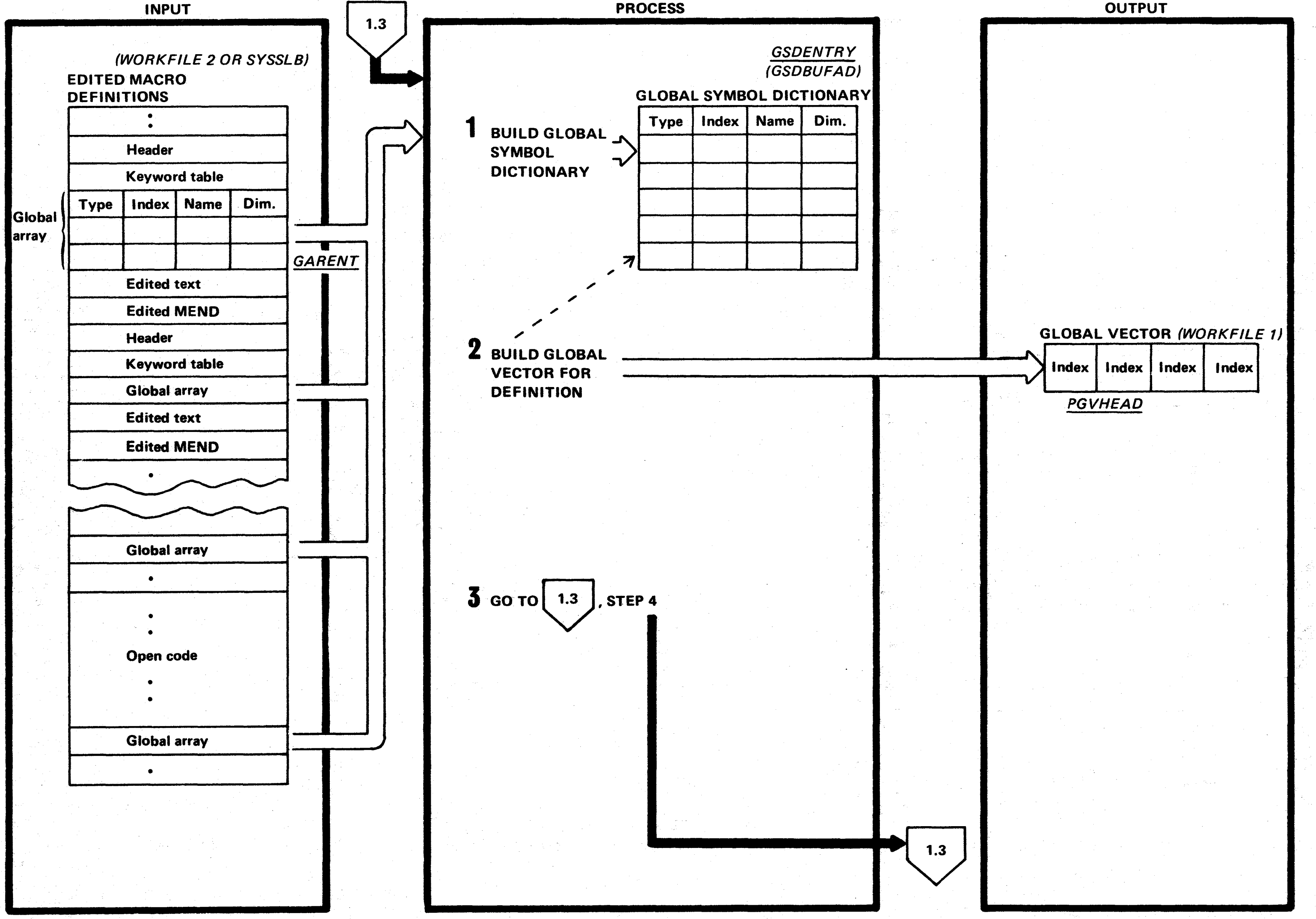
The GSD is handled in almost the same way as the MND except that there is no counterpart to MNDGET for the GSD. Therefore, it is not necessary to read back the block that was in main storage at the beginning of the search.

Macro Address Vector (MAV)

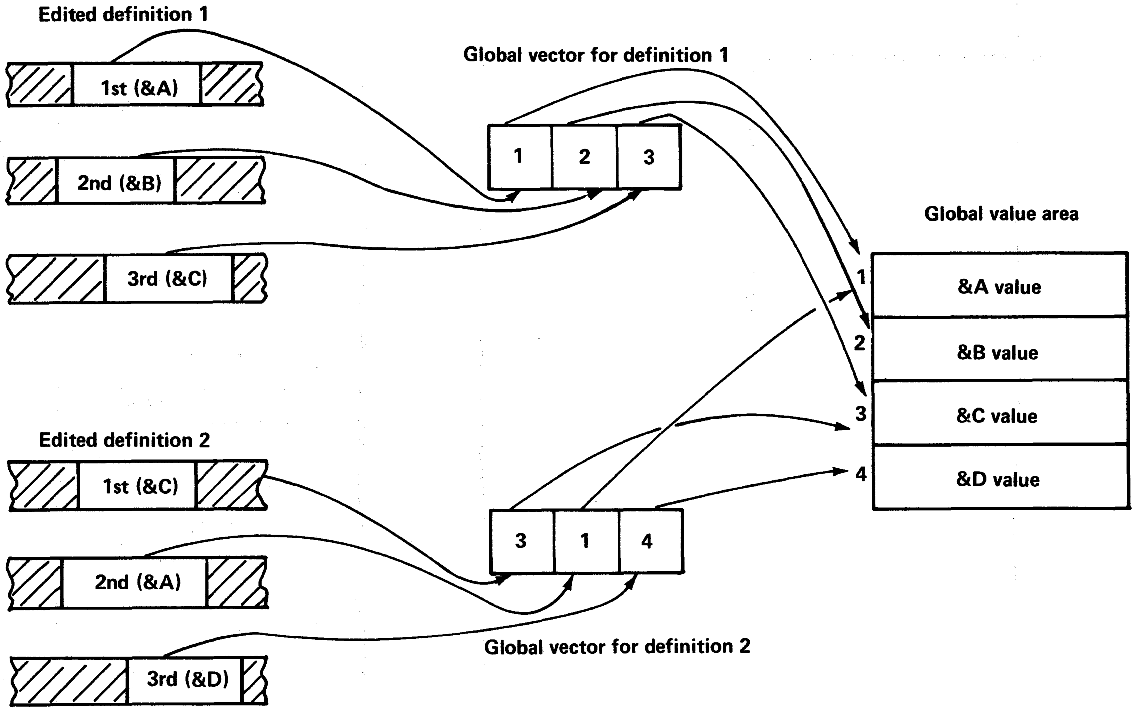
MAV blocks are written on workfile 2 among the MND and GSD blocks. At the end of ASSEFA, the blocks are copied from workfile 2 to workfile 1 in order to make them contiguous.

Note: It is necessary to have a note/point table for the MND, GSD, and MAV in order to keep track of the blocks.

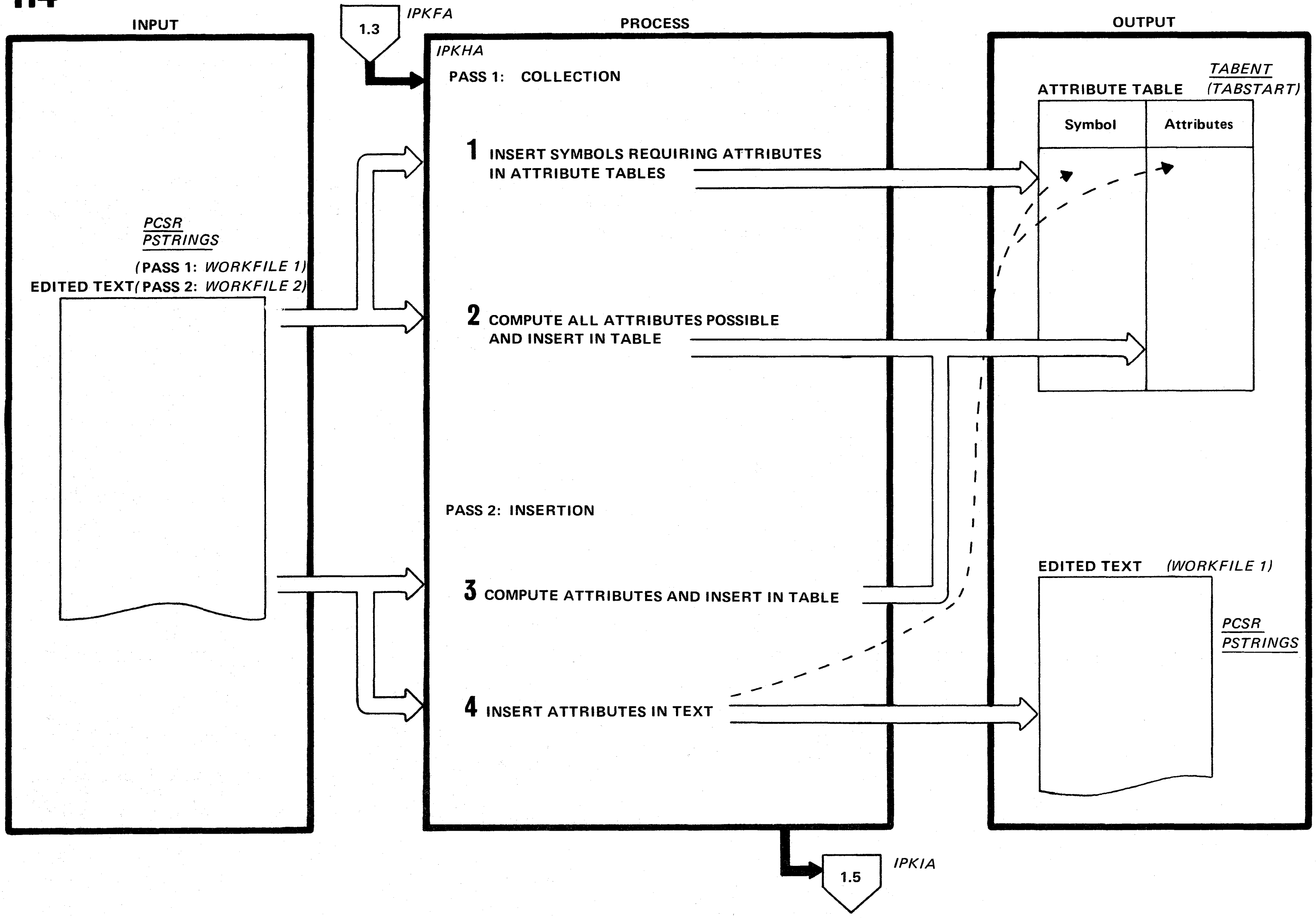
1.3.1 Build Global Vector



1.3.1

EXTENDED DESCRIPTION	MODULE	ROUTINE
Each macro definition with GBLx declarations (and open code, if it contains global symbols) has a global vector (GV). The relation among global symbol references, global vectors, and the global value area is shown below:		
		
1. The global symbol dictionary, a workarea used to assign index numbers (positions in the global dictionary), is first built from the global array of the macro definition being edited. For each new declaration a check for type and dimension is made if the symbol is already present. Contradictory information gives an error message.		FINDGS GSENT CHECKGS GEERR
2. The global vector is then built from the index numbers assigned in the global symbol dictionary. The <u>position</u> in the global vector reflects the order of occurrence of the symbol in the definition (and thus the index number in the reference). The <u>value</u> at that position in the vector gives the index of the symbol's value in the global value area. There is a separate series of indexes for GBLA, GBLB, and GBLC symbols.	IPKFA	GVENT
A global vector for open code is built from the open code global array.	IPKFA	GEOC

1.4 Collect and Insert Attributes



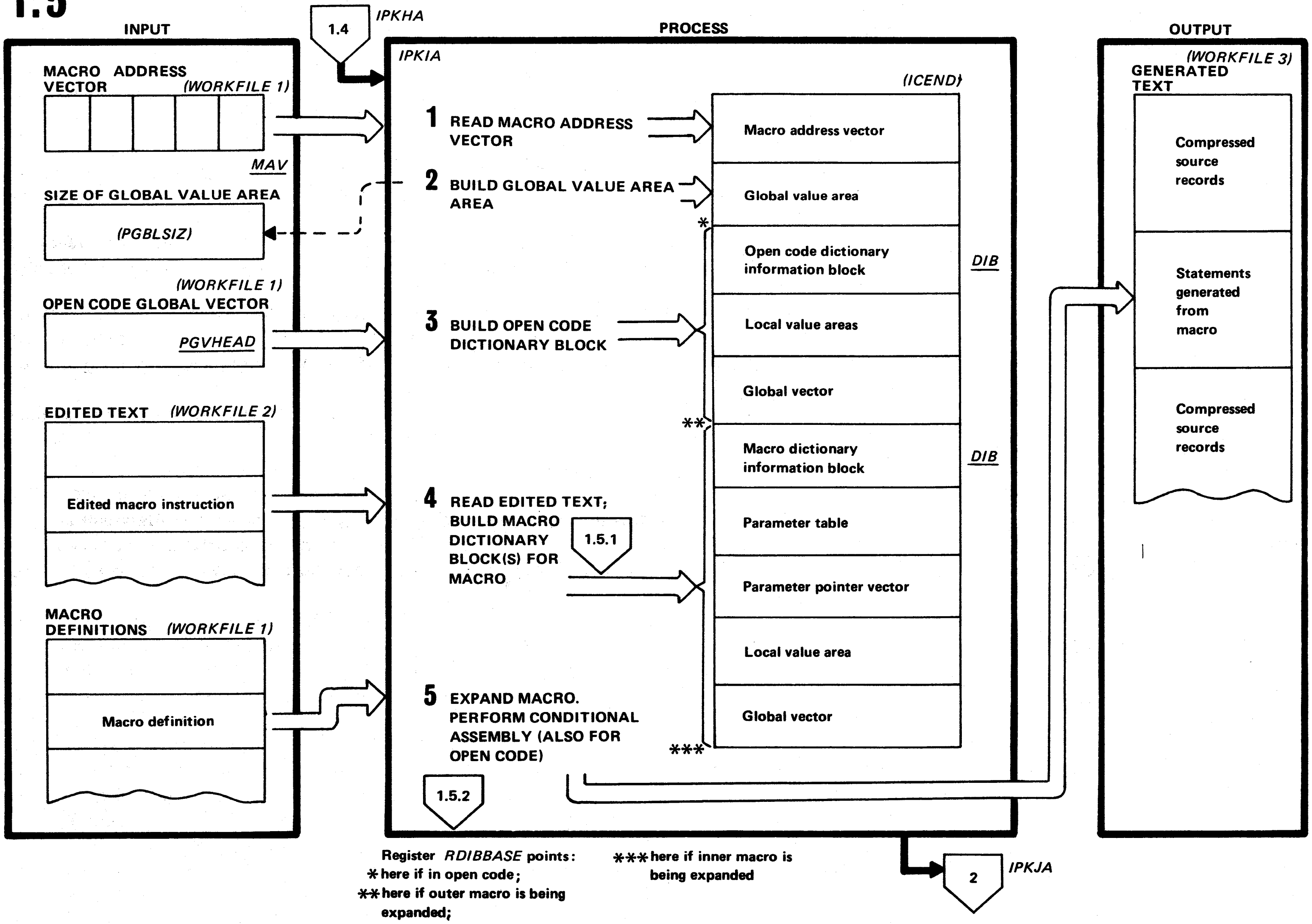
1.4

EXTENDED DESCRIPTION	MODULE	ROUTINE
1. In the first pass over the edited text file, ordinary symbols are collected from operands of macro instructions and conditional assembly statements with attribute references in open code, and placed in the attribute table.	IPKHA	MACINS CAED
2. Attributes of those symbols defined after the macro instruction or conditional assembly statements are collected and placed in the table.	IPKHA	NAMSCAN ATSCAN
3. In the second pass, attributes of symbols defined before the macro instruction or conditional assembly statement are placed in the table.		
4. Attributes are inserted into the macro instruction or conditional assembly statement with the help of the attribute table.	IPKHA	INSERT1 INSERT2

Overflow technique: If the attribute table overflows, no more symbols are collected in either of the passes. The attributes for all symbols in the table are computed and then inserted into the macro instructions and conditional assembly statements where required. A new attribute table can now be built up from text, starting where the old table overflowed. Two more text passes are needed for each overflow.

Workfile usage: Pass 1: read file 1, write file 2
Pass 2: read file 2, write file 1

1.5 Generate



Register *RDIBBASE* points:
*here if in open code;
**here if outer macro is being expanded;
***here if inner macro is being expanded

1.5

	EXTENDED DESCRIPTION	MODULE	ROUTINE
1.	The phase is initialized by loading in a number of reference elements and workareas. The macro address vector is read from workfile 1.	IPKIA	INIT
2.	The global value area is built (this is a workarea used to hold the values of global symbols during generation).		
3.	The open code dictionary block, consisting of a header, local variable symbol areas, and the global vector for open code, is built directly under the global value area.		
4.	Text is read from workfile 2, starting at open code. Compressed source records are passed directly to the generated text file (workfile 3). When a macro instruction is encountered, a macro dictionary block (a work and reference area for expansion of the macro) is built (see Diagram 1.5.1).	IPKIA IPKIA	DRIVER IMIEDIT
5.	The macro definition is processed instruction by instruction and the expanded instruction written on workfile 3. Conditional assembly and substitution are performed as necessary (see Diagram 1.5.2). Conditional assembly and substitution are also done for open code as necessary.	IPKIA IPKIA	DRIVER CAEVAL

Dynamic allocation of SETC variables

For SETC variables the value areas do not contain the values directly. Instead each SETC value area has an associated String Storage Area (SSA), where space is allocated dynamically to the values. The value area then contains for each variable its length and its offset in the SSA.

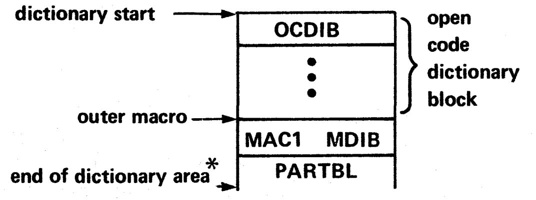
The SSA grow and shrink dynamically during macro processing as dictated by the SETC statements executed. The initial size of each SSA is zero.

The global SSA is located at the top of the dictionary area and expands downwards. Each local area is located at the top of the corresponding dictionary block and expands upwards.

Thus, overflow is possible during macro expansion. This is handled in a manner similar to that used for overflow when building the dictionary block.

Dictionary Block Overflow

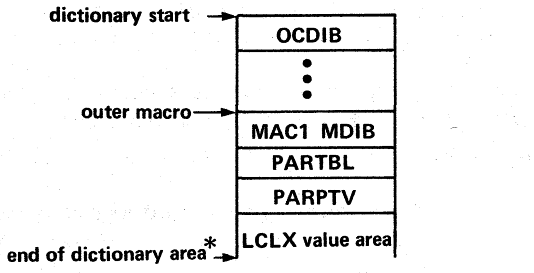
Overflow in parameter table (outer macro)

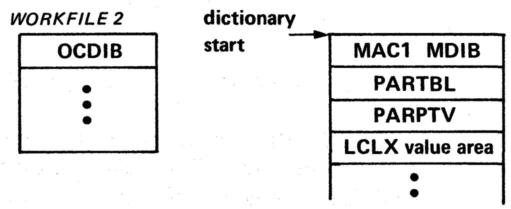


Error message IPK101 "DICTIONARY SPACE FOR VARIABLES EXHAUSTED"

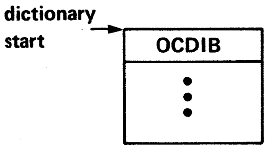
Open code dictionary block contains information necessary for PARTBL build and cannot be written on workfile until PARTBL is completed.
The macro must be abandoned.

Overflow after parameter table complete (outer macro)**



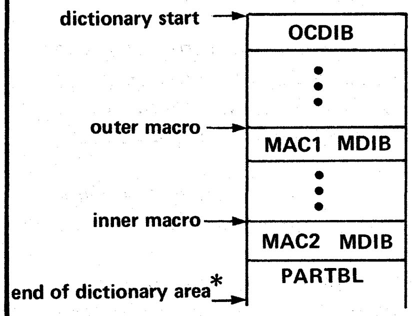


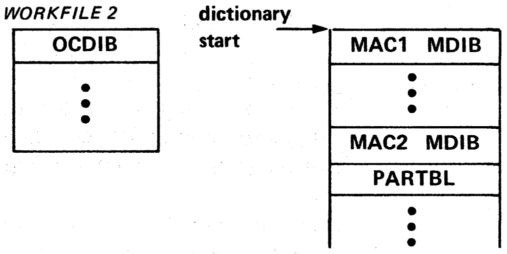
Open code dictionary block written on workfile 2.
Outer macro is moved to dictionary start
and processing continues.



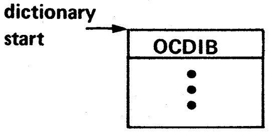
When MEND for MAC1 is encountered, open code dictionary block is read into dictionary start and processing continues.

Overflow in parameter table (inner macro)





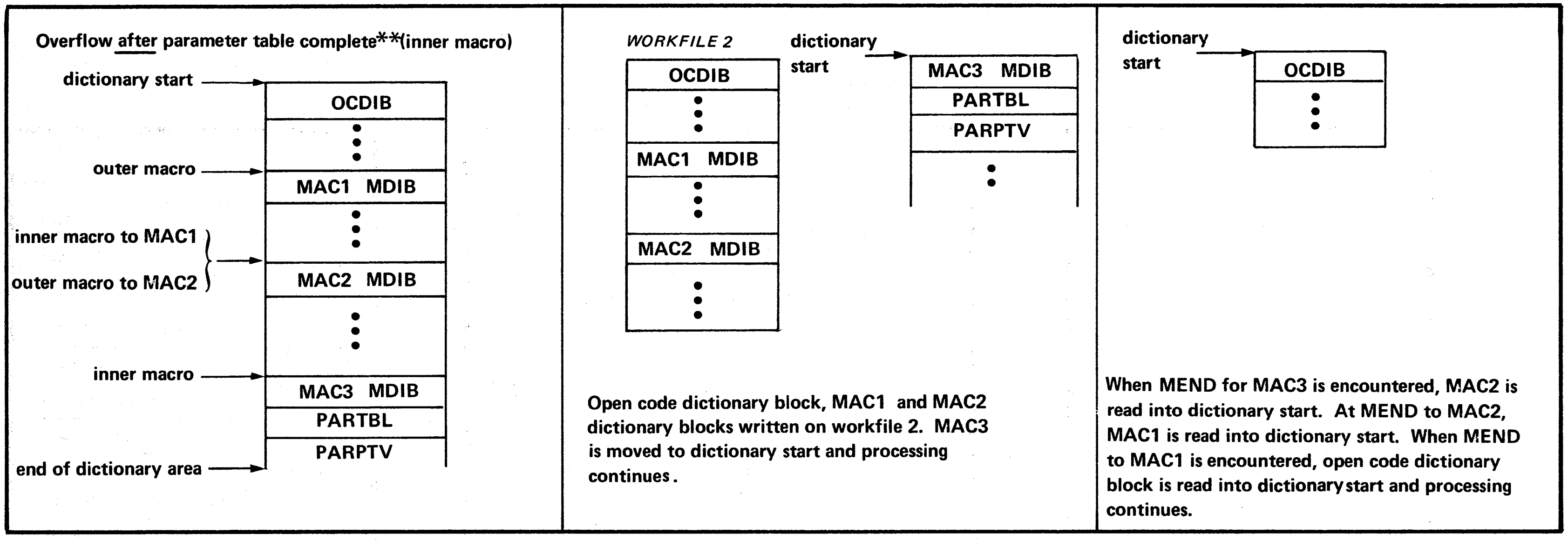
Open code dictionary block written on workfile 2.
Outer macro is moved to dictionary start.
MAC1 dictionary block contains information necessary for MAC2 PARTBL build.



At MEND to MAC2, MAC1 is read into dictionary start. When MEND for MAC1 is encountered, open code dictionary block is read into dictionary start and processing continues.

* or the last entry in the Keyword Name Array

** overflow in PARPTV, LCLX value area, and Global Vector is treated in a similar way

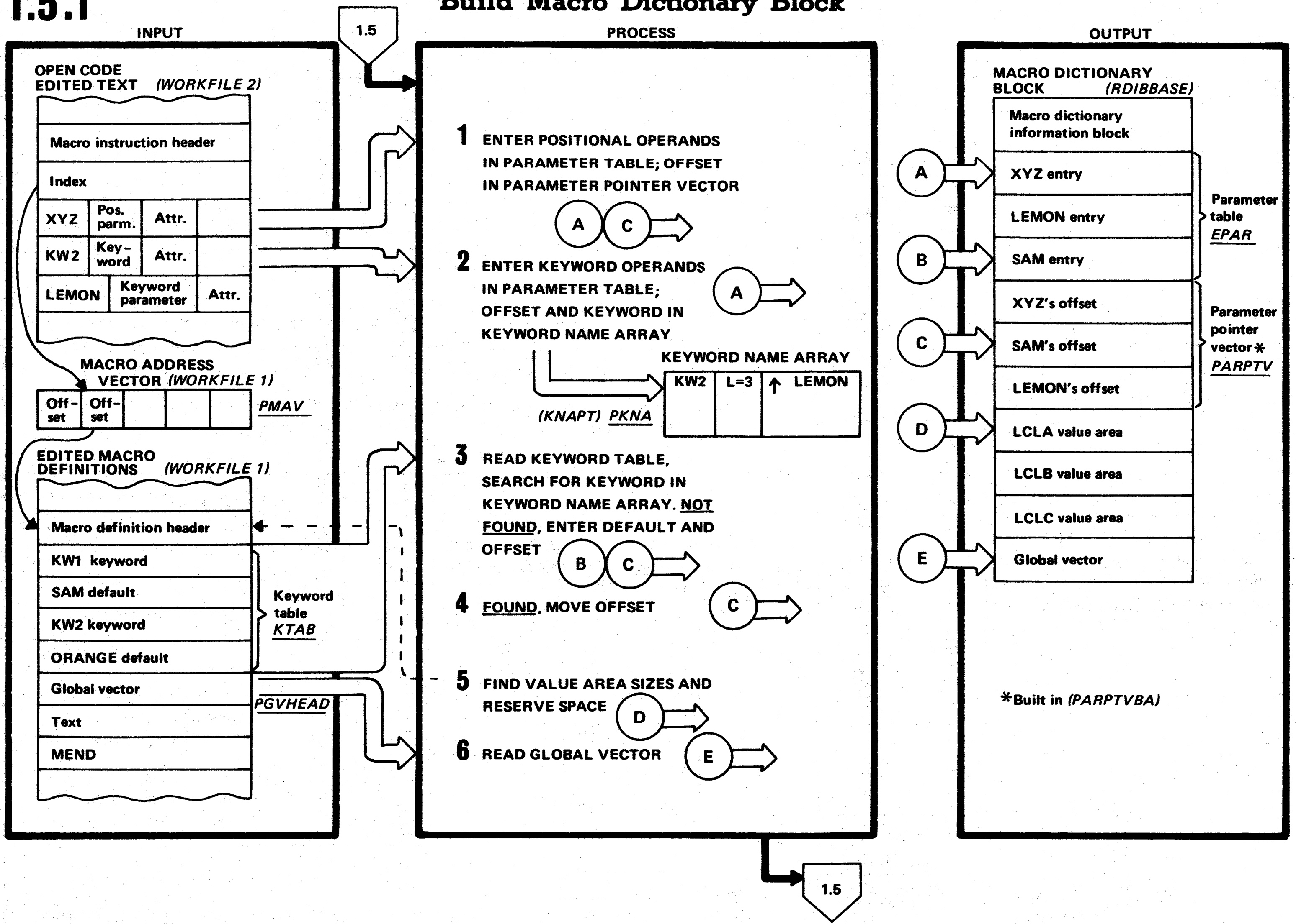


Open code dictionary block, MAC1 and MAC2 dictionary blocks written on workfile 2. MAC3 is moved to dictionary start and processing continues.

When MEND for MAC3 is encountered, MAC2 is read into dictionary start. At MEND to MAC2, MAC1 is read into dictionary start. When MEND to MAC1 is encountered, open code dictionary block is read into dictionary start and processing continues.

**Overflow in PARPTV, LCLX value area, and Global Vector is treated in a similar way.
Overflow in the String Storage Area during macro expansion is also treated in a similar way.

1.5.1 Build Macro Dictionary Block



1 ENTER POSITIONAL OPERANDS IN PARAMETER TABLE; OFFSET IN PARAMETER POINTER VECTOR

2 ENTER KEYWORD OPERANDS IN PARAMETER TABLE; OFFSET AND KEYWORD IN KEYWORD NAME ARRAY

3 READ KEYWORD TABLE, SEARCH FOR KEYWORD IN KEYWORD NAME ARRAY. NOT FOUND, ENTER DEFAULT AND OFFSET

4 FOUND, MOVE OFFSET

5 FIND VALUE AREA SIZES AND RESERVE SPACE

6 READ GLOBAL VECTOR

1.5.1

The macro dictionary block is an in-storage workarea used in expanding macros and performing conditional assembly. It consists of:

- Header (the macro dictionary information block)
- Parameter table (contains values and attributes of keyword and positional parameters used in the definition, as well as name field parameters and the current values of SYSNDX and SYSECT).
- Parameter pointer vector (contains offsets of the parameter entries in the parameter table. The parameter pointer vector is analogous to the global vector in that it is a table of pointers to another table containing the actual values).
- Local variable symbol value areas.
- Global vector.

In the following description it is assumed that the macro definition prototype is

```
MAC1  &PP,&KW1=SAM,&KW2=ORANGE
```

and that the macro instruction being expanded is

```
MAC1  XYZ,KW2=LEMON
```

Thus the prototype has one positional parameter and two keyword parameters; the first keyword parameter operand has been omitted in the instruction.

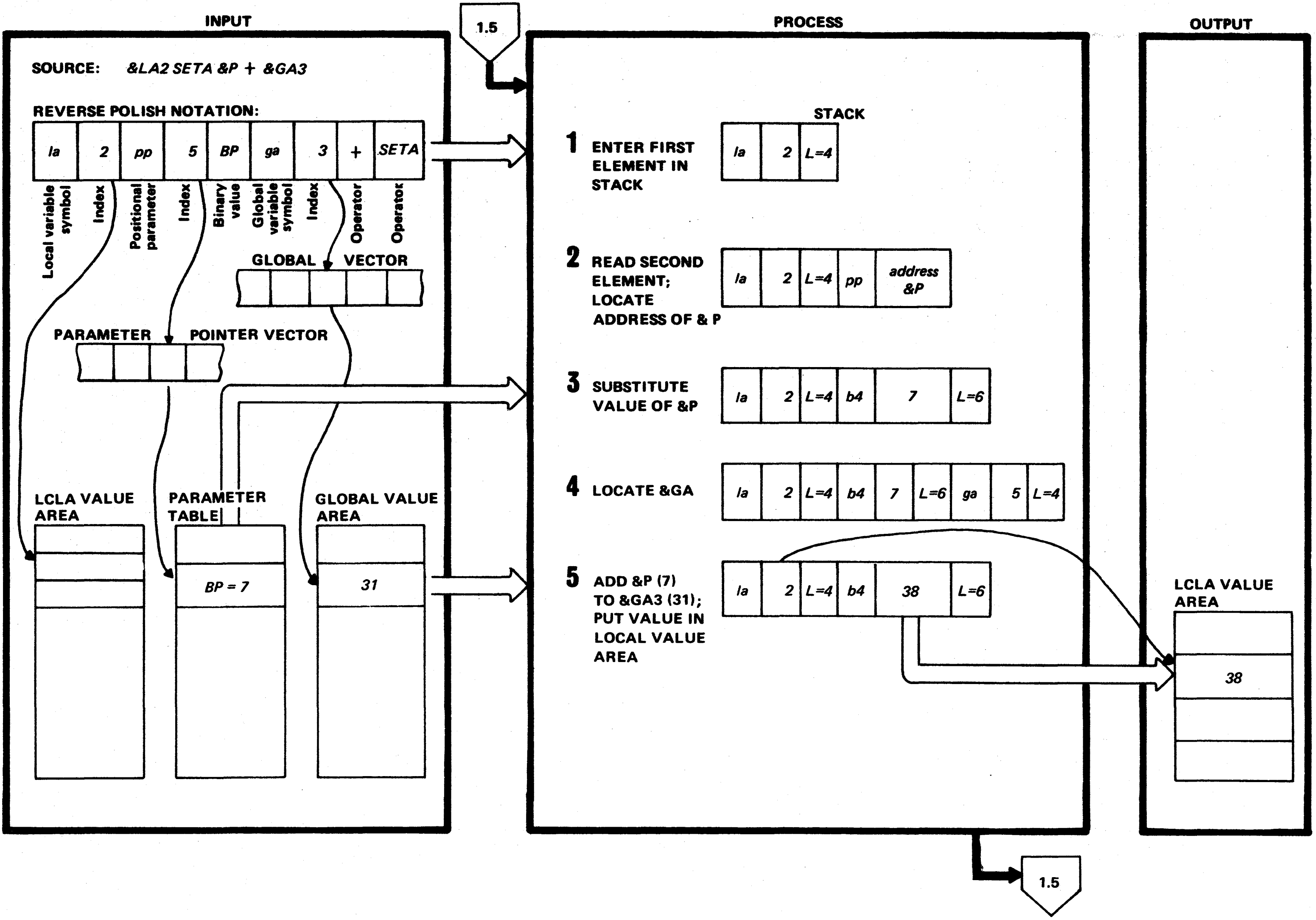
EXTENDED DESCRIPTION	MODULE	ROUTINE
1. The edited macro instruction is read and the positional parameter operand (XYZ) placed in the parameter table. The offset of the positional parameter is placed in the parameter pointer vector. (Positional parameters come first in the table, directly after SYSNDX, SYSECT, and the macro name field parameter. Then come the keyword parameters.)	IPKIA	MAIN10
2. Keyword parameters are then read in from the edited macro instruction and entered in the parameter table. At this point only "LEMON" is entered, since the first keyword parameter has been omitted. The offset is <u>not</u> entered in the parameter pointer vector, but in the keyword name array, along with the keyword's name and the length of the parameter.	IPKIA	MAINK

1.5.1 EXTENDED DESCRIPTION (continued)

	MODULE	ROUTINE
(The keyword name array is used to keep track of which keyword parameters have been omitted in the macro instruction so that default values can be inserted if necessary.)		
3. The keyword table of the macro definition is then read. Each keyword in the table is searched for in the keyword name array. If the keyword (in this case KW1) is <u>not</u> present, a default value (SAM) is placed in the parameter table. If found (as is the case with KW2), its offset in the parameter table, which had been kept in the keyword name array, is moved to the parameter pointer vector. Thus the offsets of the keyword parameters have the same order as they do in the prototype. *During steps 1-3 the keyword name array is built in high-address storage while the parameter table is built in low-address storage, towards it. The parameter pointer vector was built elsewhere in main storage. The parameter pointer vector is now written over the keyword name array, directly under the parameter table.*	IPKIA	MAIN30
4. The sizes of the local value areas are obtained from the macro definition header and space reserved for them.	IPKIA	IMIEDIT
5. The global vector is read into main storage directly under the local value areas.	IPKIA	IMIEDIT

This page intentionally left blank

1.5.2 Evaluate Reverse Polish Notation



1 ENTER FIRST ELEMENT IN STACK

2 READ SECOND ELEMENT; LOCATE ADDRESS OF & P

3 SUBSTITUTE VALUE OF &P

4 LOCATE &GA

5 ADD &P (7) TO &GA3 (31); PUT VALUE IN LOCAL VALUE AREA

1.5.2 EXTENDED DESCRIPTION

MODULE	ROUTINE
IPKIA	CAEVAL

Expressions in reverse Polish notation are sent to a routine for evaluation. The expression elements (operands and operators) are scanned from left to right. Operands are placed in stack with a length byte on top. Operators are one-byte elements acting on from zero to three stack elements and give a result element in the stack or an exit from the evaluating routine.

Actions taken by the evaluating routine when it encounters different operators and operands is summarized in Appendix D.

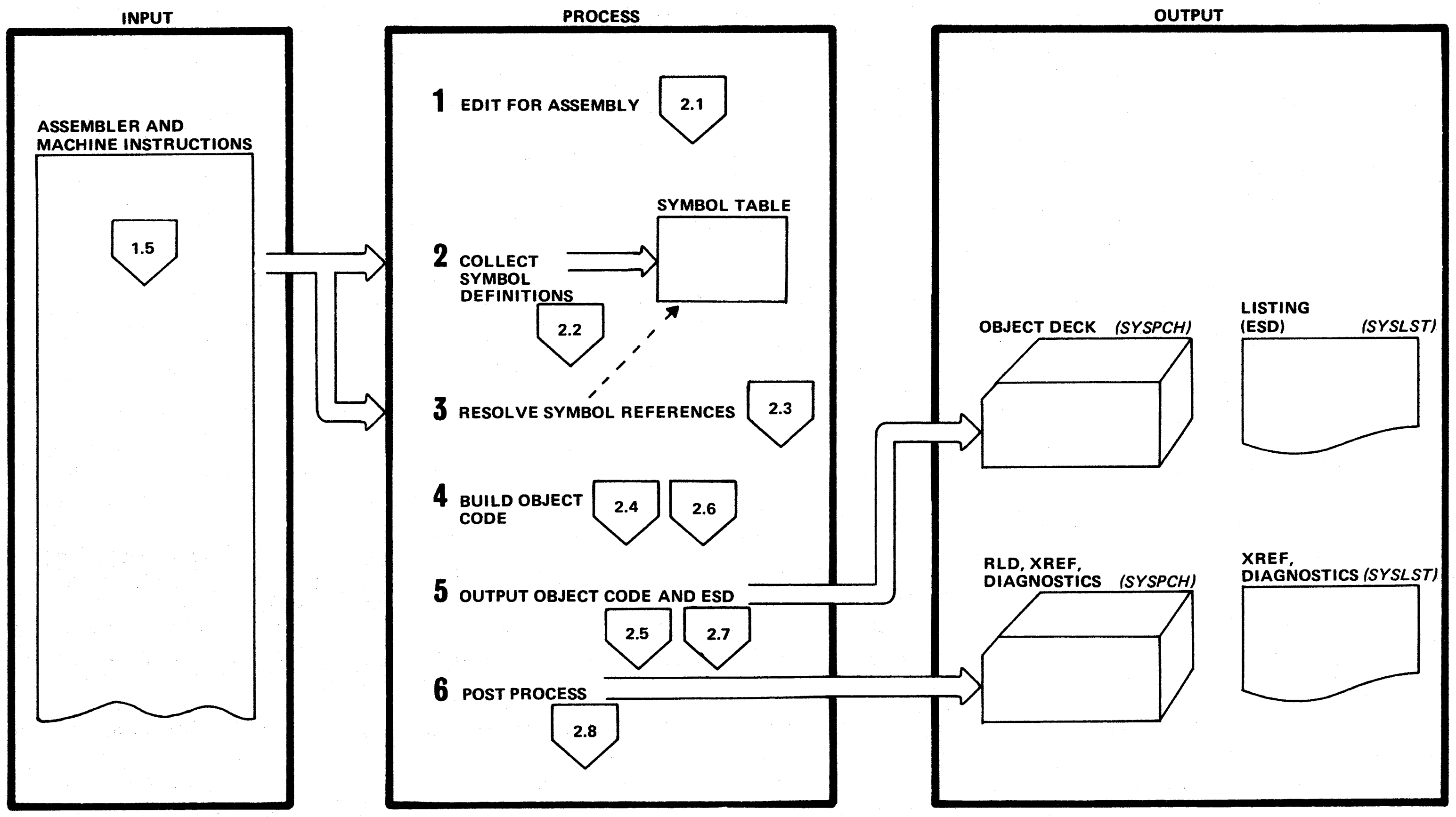
In the example in the diagram, the statement being evaluated is

&LA2 SETA &P + &GA3

where &LA2 is a local arithmetic variable symbol (the second defined in the macro definition); &P is a positional parameter (the fifth in the macro definition, with a value of 7); and &GA3 is a global variable symbol (the third declared in the definition). In the reverse Polish record, "la", "pp", etc., stand for flags (see Diagram 1.1.2.1 and Appendix D).

2

Assemble

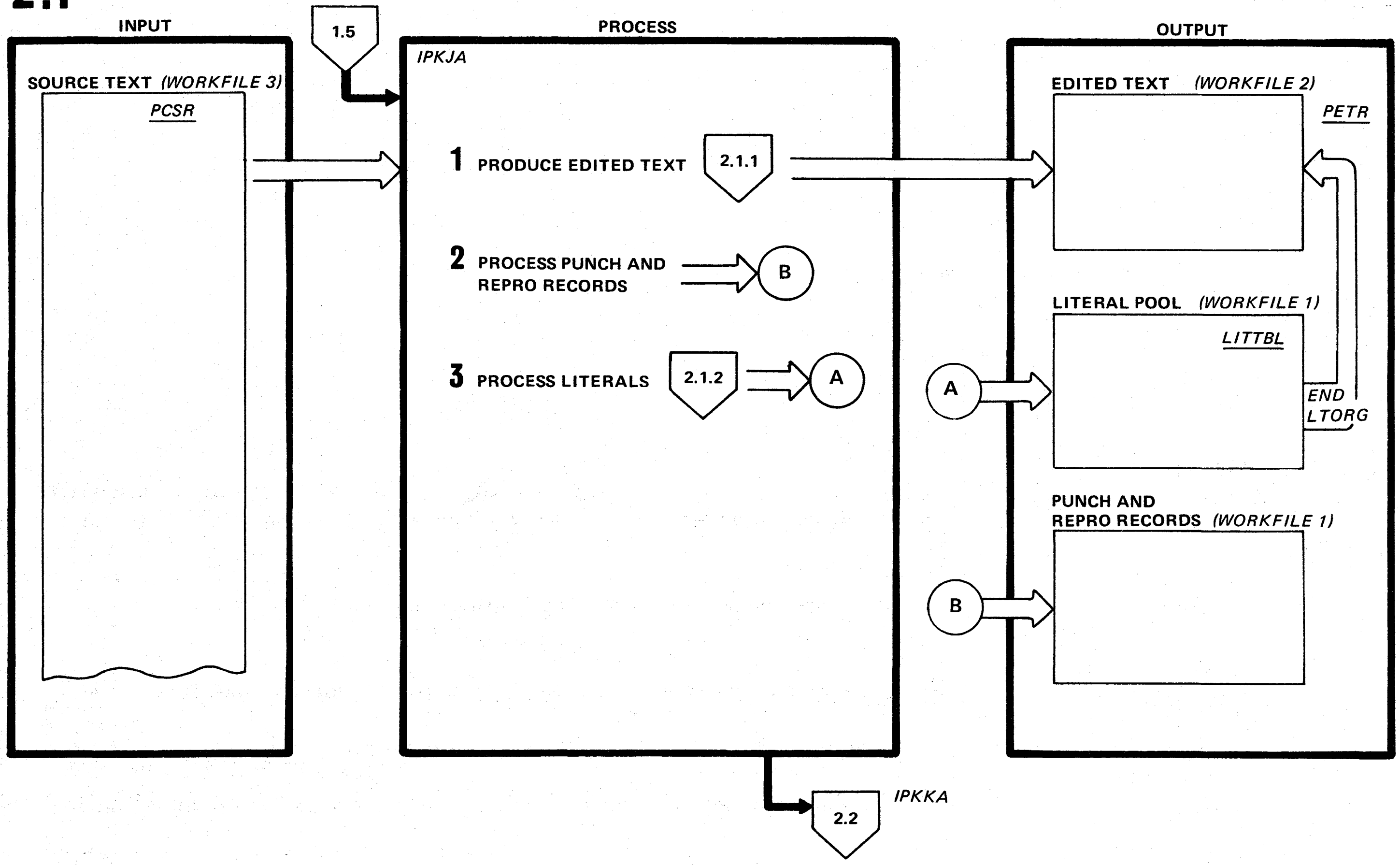


INPUT
ASSEMBLER AND MACHINE INSTRUCTIONS
1.5
PROCESS
1 EDIT FOR ASSEMBLY
2.1
SYMBOL TABLE
2 COLLECT SYMBOL DEFINITIONS
2.2
3 RESOLVE SYMBOL REFERENCES
2.3
4 BUILD OBJECT CODE
2.4
2.6
5 OUTPUT OBJECT CODE AND ESD
2.5
2.7
6 POST PROCESS
2.8
OUTPUT
OBJECT DECK (SYSPCH)
LISTING (ESD) (SYSLST)
RLD, XREF, DIAGNOSTICS (SYSPCH)
XREF, DIAGNOSTICS (SYSLST)

2

EXTENDED DESCRIPTION	MODULE
At this stage the edited text is free from macro instructions and conditional assembly statements (that is, it consists only of machine and assembler instructions). The second main function of the assembler is to translate these statements into object code and to produce a listing and other related information.	
1. Editing of statements is done (see Diagram 2.1).	IPKIC IPKJA
2. Symbol definitions are collected and stored in a symbol table (see Diagram 2.2).	IPKKA
3. References to symbols are resolved with the help of the symbol table (see Diagram 2.3).	IPKLA
4. Object code is generated for the different instruction types (see Diagram 2.4 and 2.6).	IPKNA IPKOA
5. Object code is put out, together with a listing and the external symbol dictionary (see Diagrams 2.5 and 2.7).	IPKMA IPKPA
6. In the post-process phases, the relocation dicitionary, cross-reference dictionary, and diagnostic information is put out (see Diagram 2.8).	IPKQA IPKRA-RC IPKSA-SB

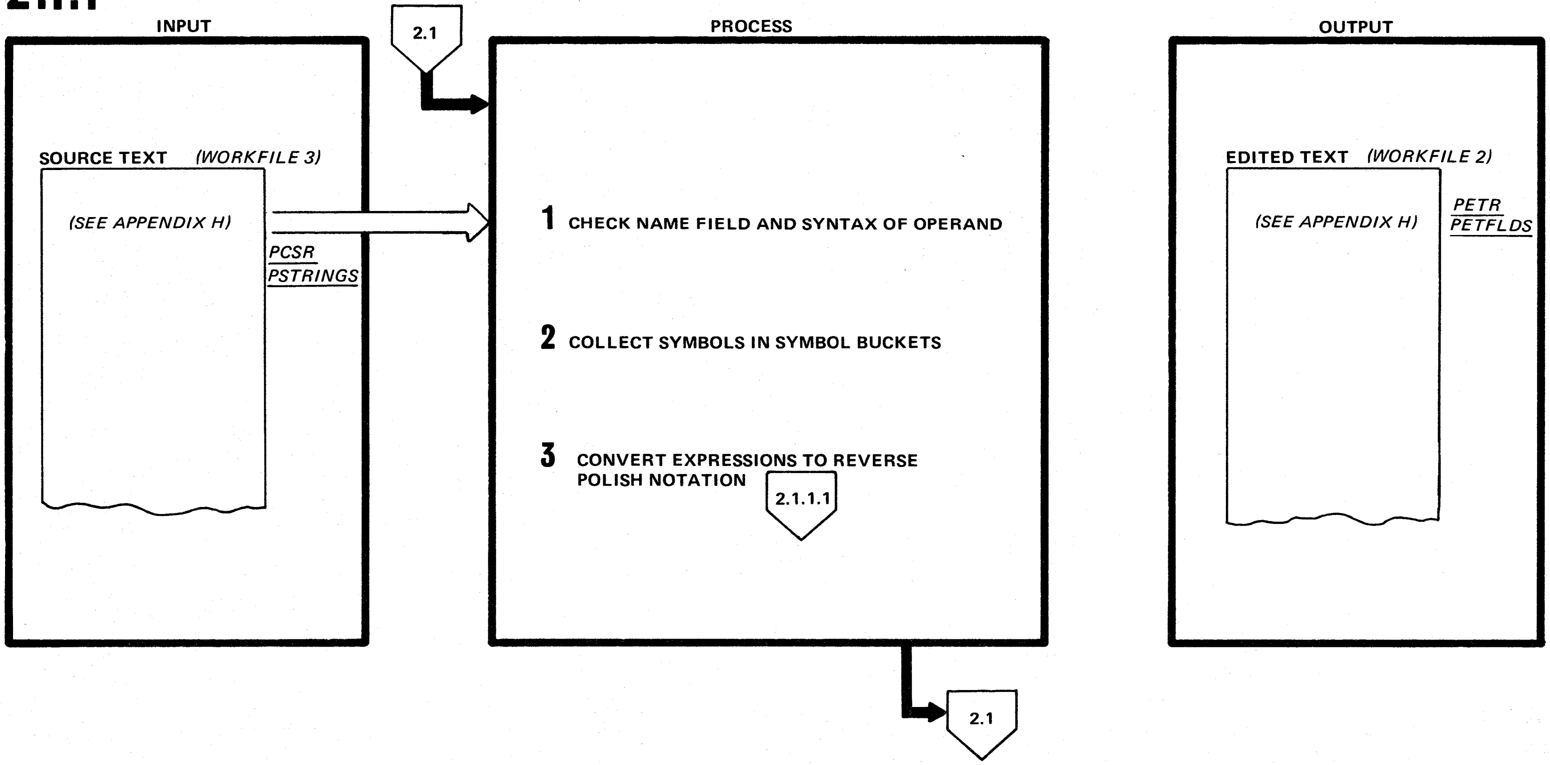
2.1 Edit for Assembler and Machine Instructions



2.1

EXTENDED DESCRIPTION	MODULE	ROUTINE
1. Editing consists of checking, converting the record into a suitable format for later processing, collecting symbols into "symbol buckets" and converting expressions into reverse Polish notation (see Diagram 2.1.1).	IPKJA	
2. PUNCH and REPRO records found before the first control section are processed and written on workfile 1. See Diagram 2.5 for later processing.	IPKJA	PUNCHR REPROEDR
3. Literals are put into a literal pool and processed when an END or LTORG expression is encountered (see Diagram 2.1.2).	IPKJA	LITERAL LTORGR LITDRV LITMN LITSRCE

2.1.1 Edit



1 CHECK NAME FIELD AND SYNTAX OF OPERAND

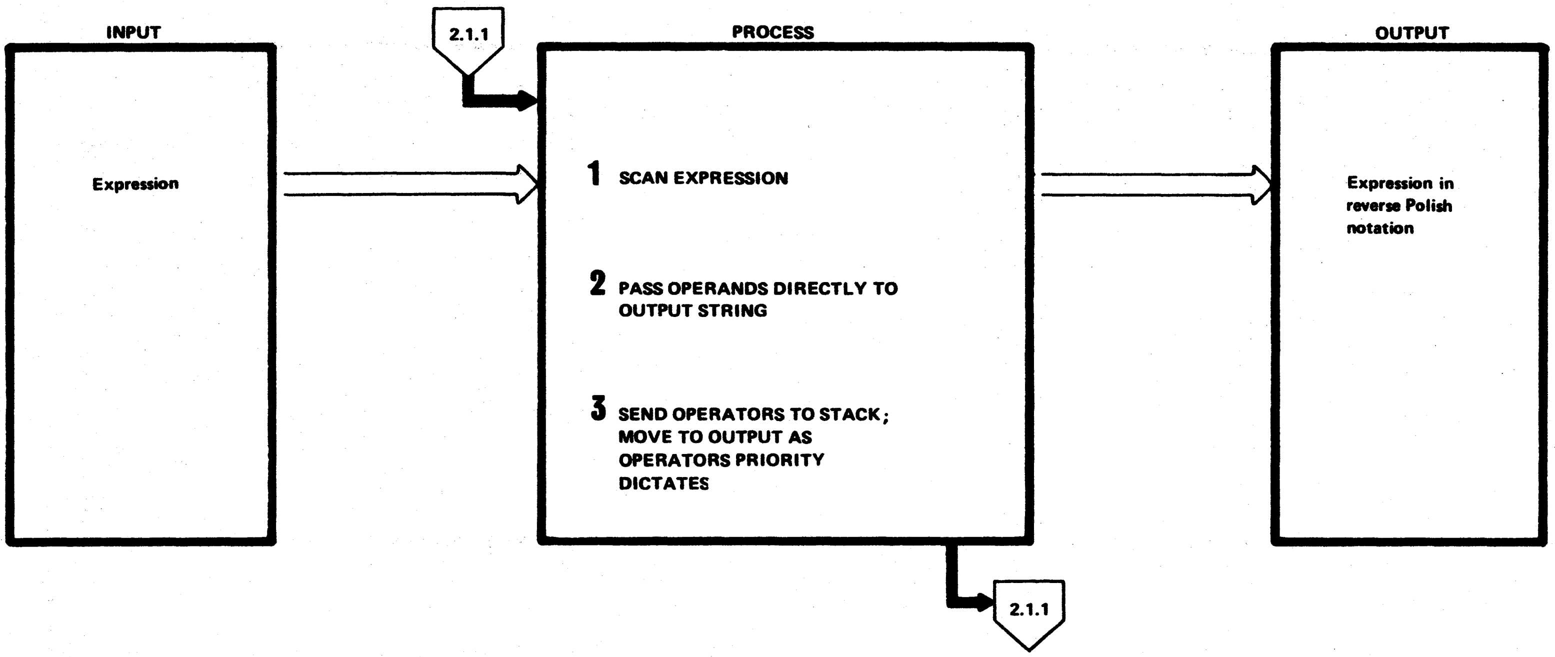
2 COLLECT SYMBOLS IN SYMBOL BUCKETS

3 CONVERT EXPRESSIONS TO REVERSE POLISH NOTATION

2.1.1

EXTENDED DESCRIPTION	MODULE	ROUTINE
Records which have been partially edited at an earlier stage (see Diagram 1.1.1) are more fully edited for assembly. The length attribute, ESDID, and location counter fields are created but not filled until symbol resolution (see Diagram 2.3).		
1. The name field and operand syntax are checked.	IPKJA	CHKNAME
2. All symbols are placed in "symbol buckets" in the edited text. Symbols in expressions are replaced by flags referring to the buckets; the buckets appear in the same order as the symbols in the expressions. If the same symbol appears more than once in an expression, there will be more than one bucket for it (see "Diagnostic Aids" for examples of symbol buckets).	IPKJA	OPERAND
3. Expressions are converted to reverse Polish notation (see Diagram 2.1.1.1).	IPKJA	POLIFY

2.1.1.1 Convert Assembly Expressions to Reverse Polish Notation



1 SCAN EXPRESSION

2 PASS OPERANDS DIRECTLY TO OUTPUT STRING

3 SEND OPERATORS TO STACK; MOVE TO OUTPUT AS OPERATORS PRIORITY DICTATES

2.1.1.1

EXTENDED DESCRIPTION

MODULE	ROUTINE
IPKJA	POLIFY

1. Expressions are scanned from left to right.

2. Operands are assigned identifying flags and are inserted immediately into the output string (see Appendix G for the flags).

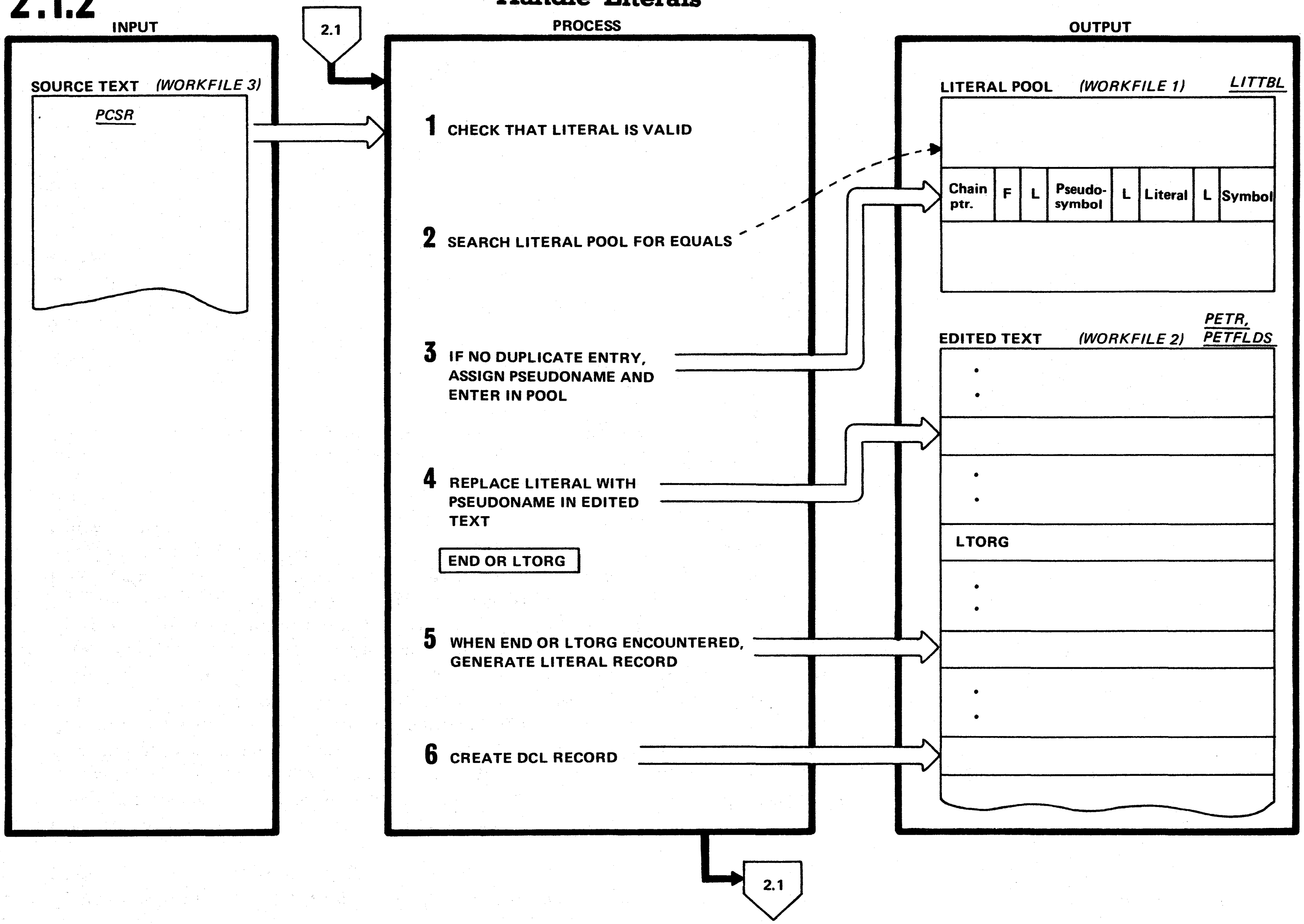
3. Operators are put in a stack according to their priority. The higher the priority, the sooner the operator is inserted into the output string. The first operator encountered is always entered into the stack. For all other operators, the operator's priority is compared to that of the previous operator entered in the stack. If the priority is lower than that of the previous entry, the operator is placed in the stack. If the priority is higher than or equal to that of the previous entry, the previous operator is removed from the stack and placed in the output string. The operator's priority is then compared with that of the next operator in the stack, and so on.
   There are two exceptions to this processing method:
   Left parenthesis: placed in the stack without comparision of priority
   Right parenthesis: causes the stack to be emptied until a left parenthesis is found. The left parenthesis is also removed.
   Operator priorities:

   2 + –

   3 / *

   For examples of expressions in reverse Polish, see "Diagnostic Aids".

2.1.2 Handle Literals



1 CHECK THAT LITERAL IS VALID

2 SEARCH LITERAL POOL FOR EQUALS

3 IF NO DUPLICATE ENTRY, ASSIGN PSEUDONAME AND ENTER IN POOL

4 REPLACE LITERAL WITH PSEUDONAME IN EDITED TEXT

5 WHEN END OR LTORG ENCOUNTERED, GENERATE LITERAL RECORD

6 CREATE DCL RECORD

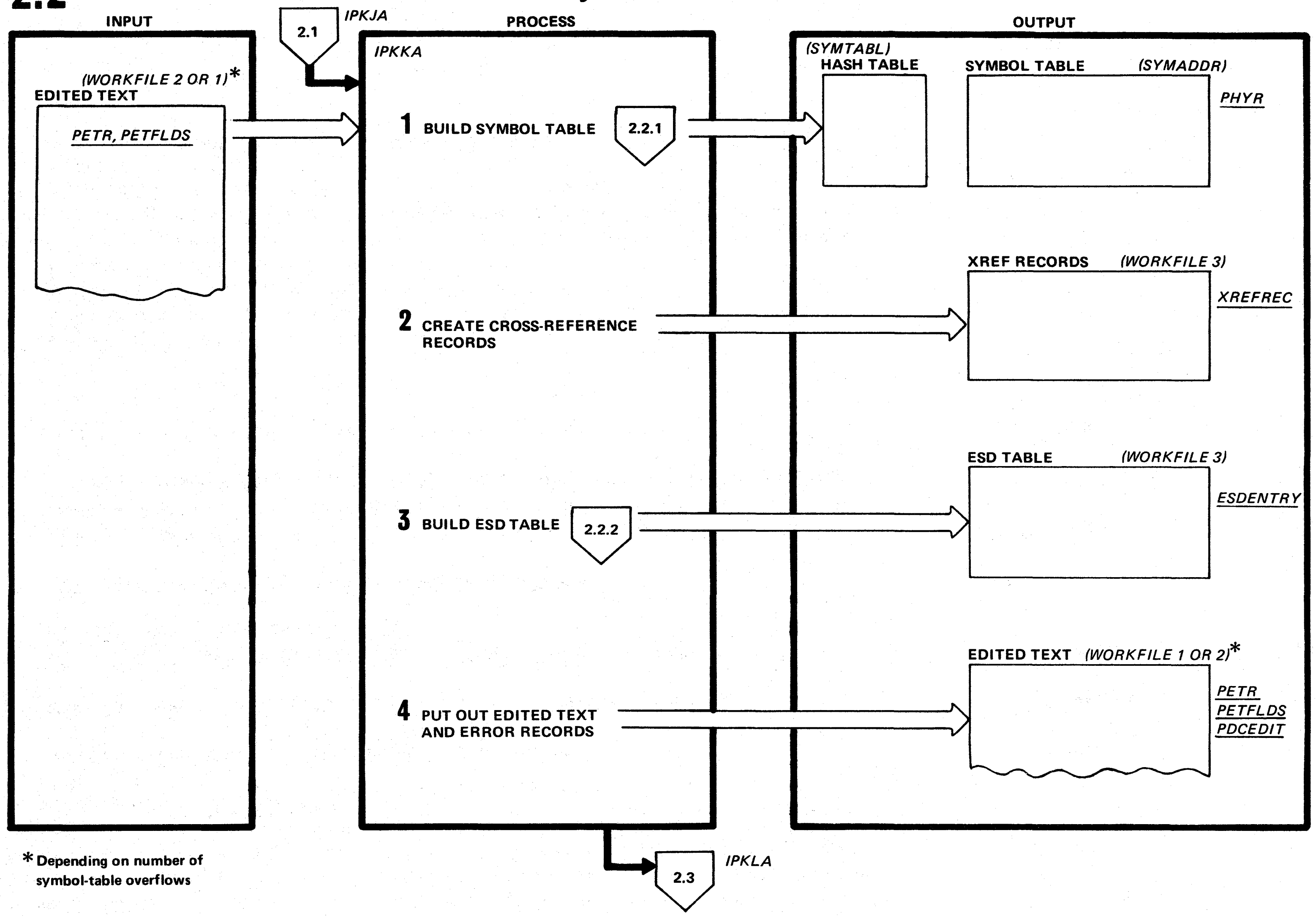
2.1.2

EXTENDED DESCRIPTION	MODULE	ROUTINE
A literal pool contains all literals occuring from the start of the first control section or the previous LTORG. The entries are in chains according to the length required in main storage when assembled (8, 4, 2, or 1 bytes). The literals are assigned symbolic names so that they can be handled as DC instructions: when a LTORG or END statement is encountered, literal DC instructions are generated for each entry so that the literals can later be handled as regular DC instructions during symbol resolution.		
1. The literal is checked for validity (using the same routine as for DCs).	IPKJA	DCR
2. The literal pool (in main storage and on workfile 1) is searched for equals.	IPKJA	DCR
3. If not already entered in the literal pool, the literal is assigned a symbolic name* and entered in its appropriate chain.	IPKJA	LITERAL
4. The literal is replaced by its symbolic name in the edited text.	IPKJA	LITMN
5. When a LTORG or END statement is encountered, the literal pool is read and literals retrieved from the 8-, 4-, 2-, and 1-byte literal chains; a literal record (in the form of a compressed source record) is constructed for each and written on workfile 2 for the listing.	IPKJA	LITSRCE
6. A DCL instruction is generated for each literal and written on workfile 2.	IPKJA	DCR FIXUP DRIVER

*A symbolic name for literals is generated with
- an identification byte (internal code = 1) in the first byte
- five characters copied from the literal definition in bytes 2-6
- a sequence number for generated names in bytes 7-8

When a location counter reference is made in a literal, a name is generated for the statement if it has not already got a name field. The name generated is similar to that for literals with
- an identification byte (internal code = 8) in the first byte
- five blank characters
- a sequence number for generated name in byte 7-8

2.2 Collect Symbol Definitions

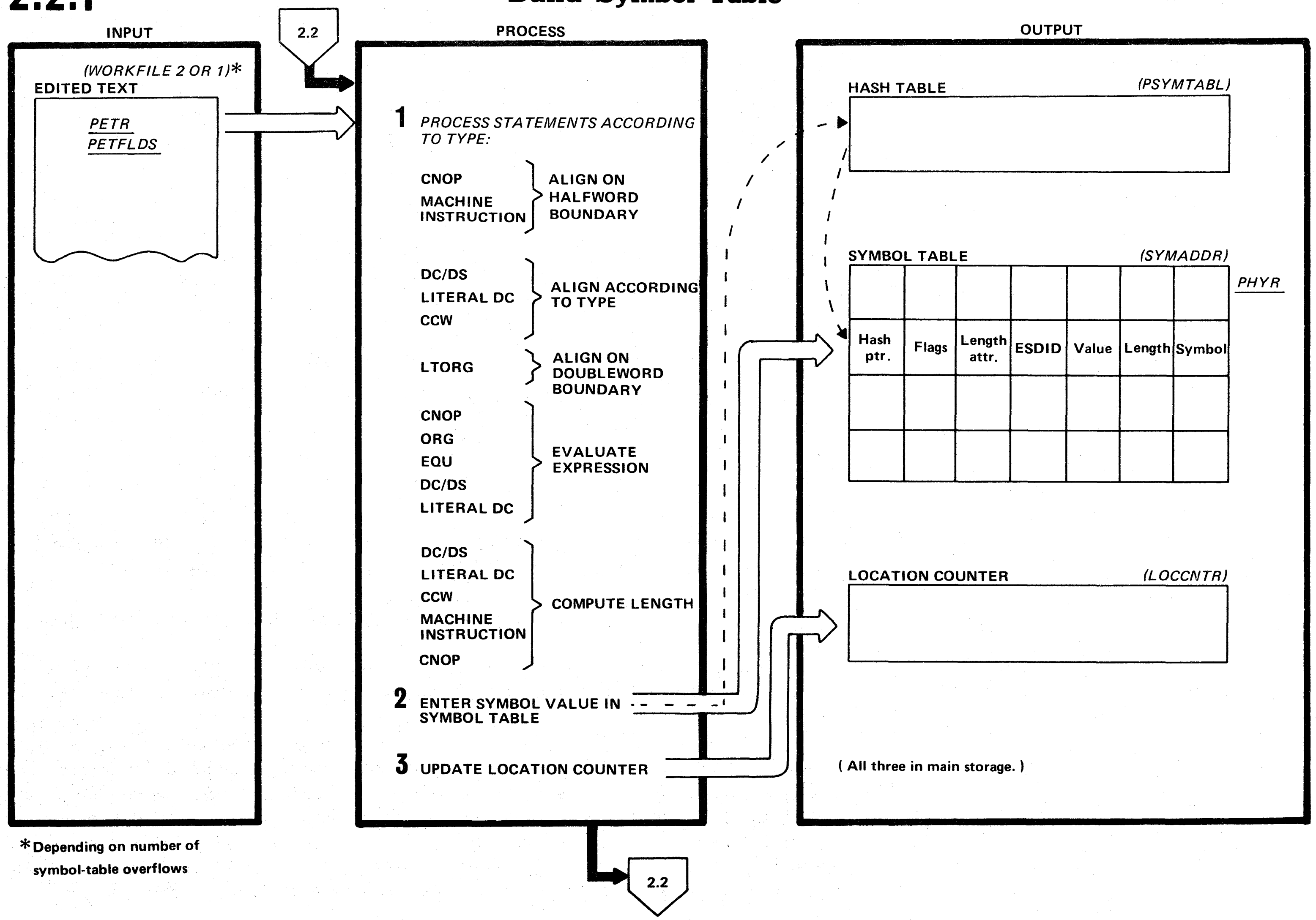


* Depending on number of symbol-table overflows

2.2

EXTENDED DESCRIPTION	MODULE	ROUTINE
1. Symbol definitions are collected in a symbol table for later use in resolving symbol references.	IPKKA	
<u>Overflow technique</u>: If the symbol table overflows, the substitution phase (ASSELA) is called in to process the currently built symbol table segment. The substitution phase will resolve all references to symbols in the symbol table. When this is done, IPKKA will be called back in and will start processing the text where it left off when the overflow occurred. It will discard the old symbol table and start building a new one. This process will continue until the text is all processed or the symbol table overflows again. If it does, the substitution phase will be called and that segment processed, and so on.		
2. Cross-reference records are created for both symbol definitions and references (and duplicates). These will later be put out by the post-processor (see Diagram 2.8).	IPKKA	CROSSREF
3. The external symbol dictionary table is built (see Diagram 2.2.2).	IPKKA	
4. The text is edited: expressions in the operands of CNOP, ORG, EQU, and END statements are evaluated and length and duplication factors calculated for DC, DS, and DCL instructions.	IPKKA	

2.2.1 Build Symbol Table

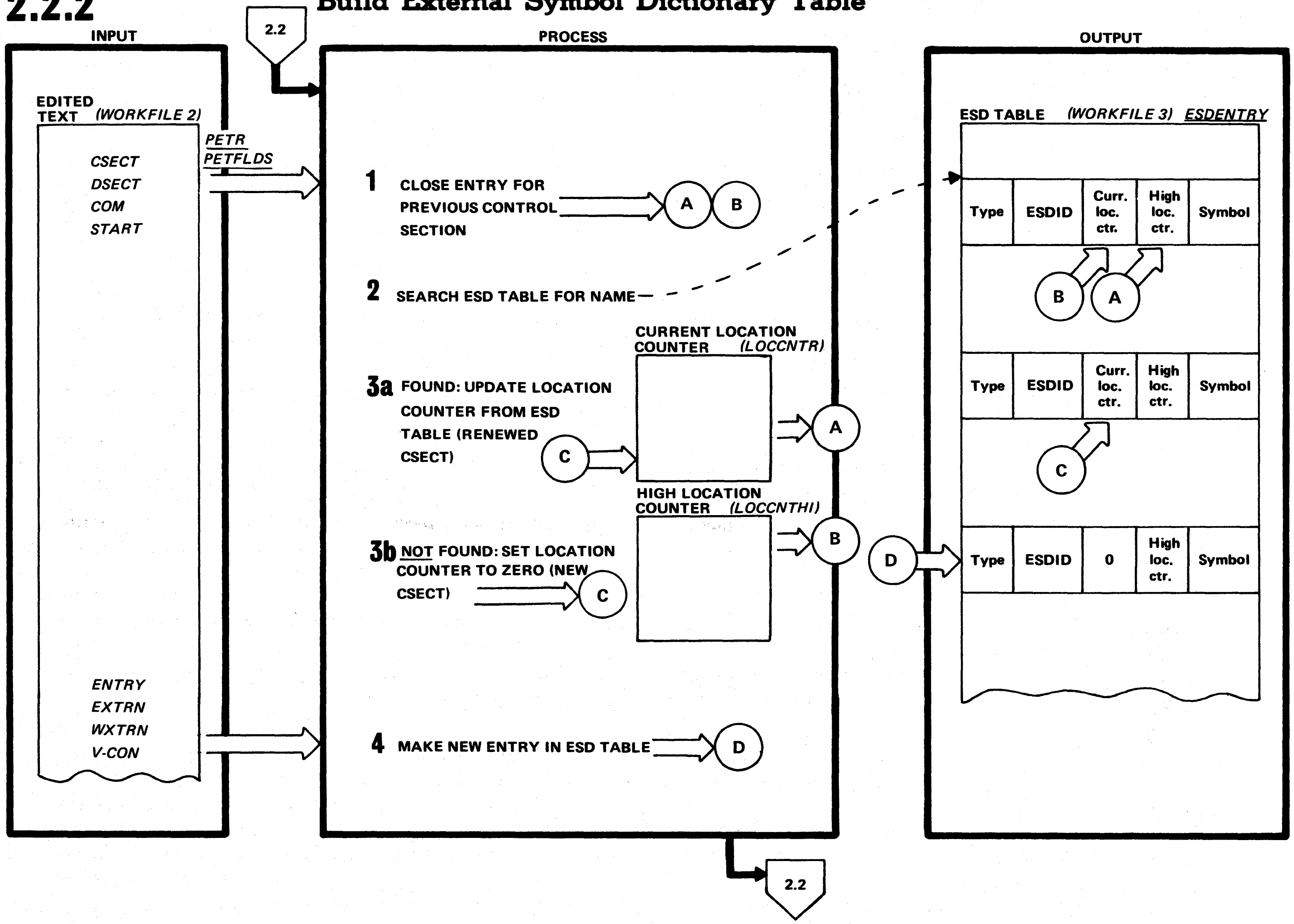


*Depending on number of symbol-table overflows

2.2.1

EXTENDED DESCRIPTION	MODULE	ROUTINE
Symbol definitions are collected in the symbol table. In order to define a symbol, the corresponding value of the location counter must be computed. This value, in turn, depends on the lengths of the instructions and their alignment in main storage when assembled.	IPKKA	
1. Instructions are processed according to type. They are aligned, expressions are evaluated, and their lengths computed. The routine names for the instructions given below are shown in the column to the right:		
Machine instructions		MACHINOP
EQU		EQUR
CNOP		CNOPR
ORG		ORGPROC
DC		DCR
DS		DCR
DCL		DCR
CCW		CCWR
LTORG		LTORGR
END		ENDR
2. Symbols are entered in the symbol table, together with their values, length attributes, and ESDID.	IPKKA	
3. The location counter is updated for those instructions requiring it.	IPKKA	

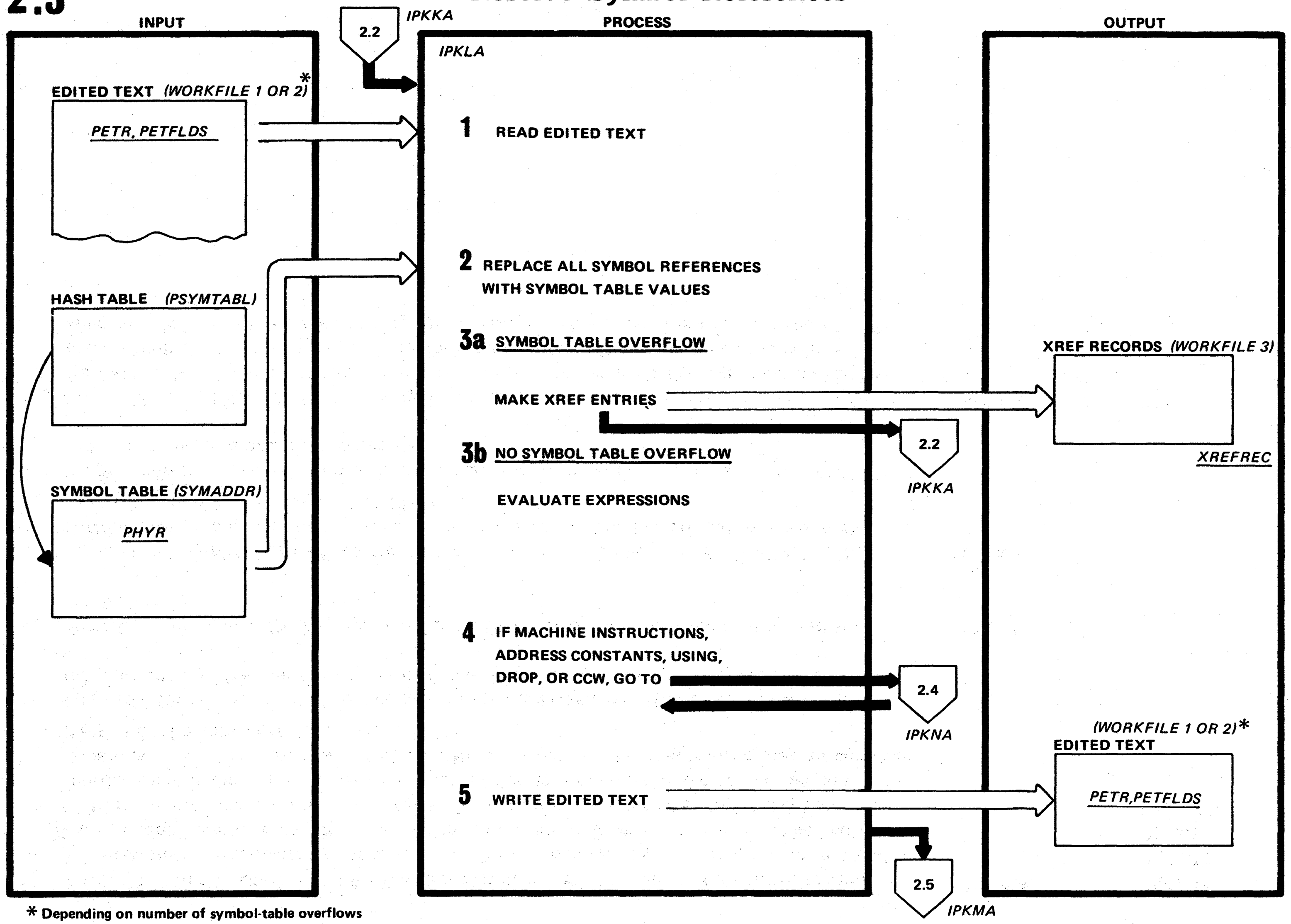
2.2.2 Build External Symbol Dictionary Table



2.2.2

EXTENDED DESCRIPTION	MODULE	ROUTINE
The external symbol dictionary table saves external symbols which will later be printed out in the ESD output; the table is also a control for CSECTs.		
1. A CSECT, DSECT, COM, or START instruction means the beginning of a control section. The previous control section's entry in the ESD table is closed by retrieving the current and high location counter values from COMMON and inserting them in the entry. (The "current" location counter value is simply the value of the location counter; the "high" location counter value is the highest value that occurred during processing of the control section -- this value may later have been lowered by an ORG statement.) The high value is used to compute the size of the control section. The entry for the last control section will be updated when the END statement is read. If there are literals after the END statement, the first control section will be updated after them.	IPKKA	CSECTR DSECTR COMR STARTR
2. The label associated with the CSECT, DSECT, COM, or START instruction is looked for in the ESD table.	IPKKA	
3 a. If found, the symbol has been entered before, and the CSECT is a resumed CSECT. The current location counter value is retrieved from the entry (ESDLCTR) and inserted in the current location counter value (LOCCNTR).	IPKKA	
3 b. If <u>not</u> found, the location counter is set to zero (or to the address specified in the START operand) to begin a new control section.		
4. An ENTRY, EXTRN, WXTRN, or V-type address constant causes an entry to be made in the ESD table. The current location counter value is the same as the high location counter value -- both are the actual value of the location counter. The current location counter value for ENTRY is the value at which the symbol is defined, not the value for the ENTRY statement itself.	IPKKA	PEEX

2.3 Resolve Symbol References



1 READ EDITED TEXT

2 REPLACE ALL SYMBOL REFERENCES WITH SYMBOL TABLE VALUES

3a SYMBOL TABLE OVERFLOW

MAKE XREF ENTRIES

3b NO SYMBOL TABLE OVERFLOW

EVALUATE EXPRESSIONS

4 IF MACHINE INSTRUCTIONS, ADDRESS CONSTANTS, USING, DROP, OR CCW, GO TO

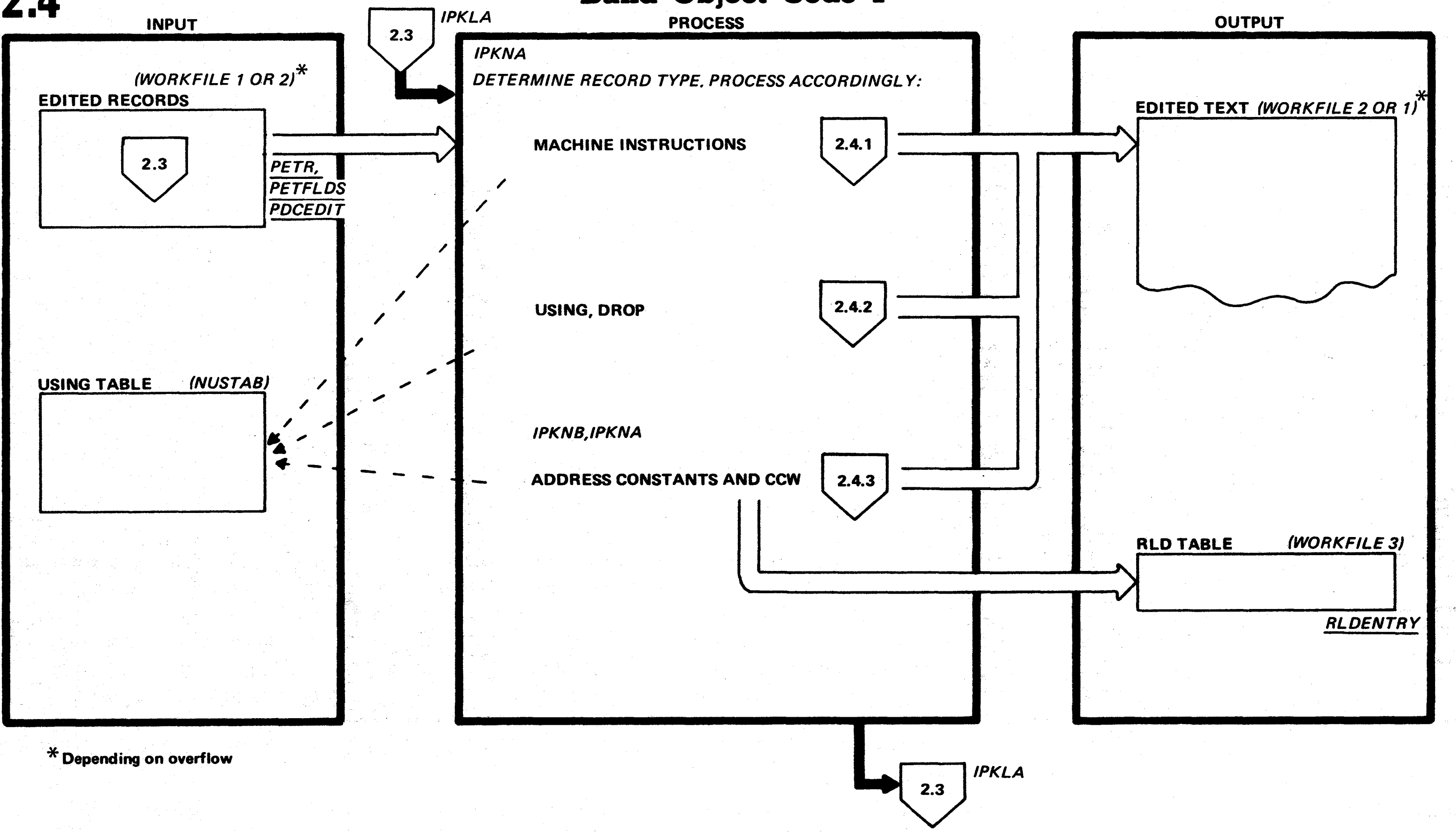
5 WRITE EDITED TEXT

*** Depending on number of symbol-table overflows**

2.3

EXTENDED DESCRIPTION	MODULE	ROUTINE
1. Edited text containing unresolved symbol references is read from workfile 1 (or from workfile 2, depending on symbol table overflow).	IPKLA	
2. Symbol references are resolved by hashing the symbol and finding its value in the symbol table.		
3 a. If there has been symbol table overflow		
ESD entries are made from those symbol references which are resolved before they are read by IPKKA.	IPKKA	CROSSREF
3 b. If there has been no symbol table overflow or if this is the last time IPKLA has been called		
Expressions involving symbols can now be evaluated (expressions involving CNOP, ORG, END, and EQU, in addition to duplication and length modifiers in DC and DS instructions, have already been evaluated (see Diagram 2.2.1)).	IPKKA	EVALUATE
4. If the instruction is a machine instruction, address constant, USING, DROP, or CCW, IPKNA is called.		
5. Otherwise the edited text is written on workfile 1 or 2.		

2.4 Build Object Code 1

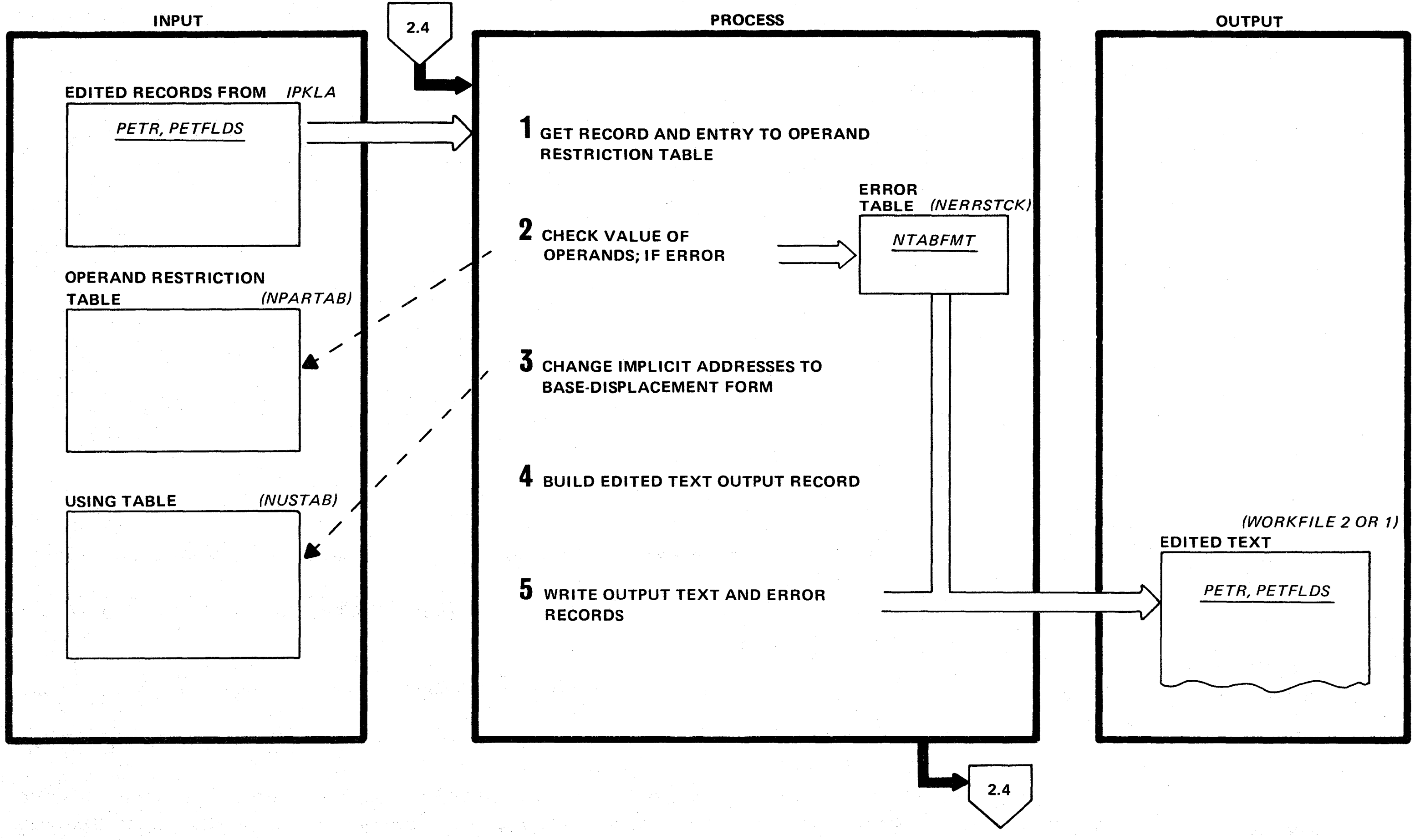


* Depending on overflow

2.4

EXTENDED DESCRIPTION	MODULE	ROUTINE
Input: edited records from IPKLA Output: object code and edited records on workfile 2 (or 1, depending on symbol table overflow).		
Processing proceeds according to record type. The following records are processed:		
Machine Instructions (Diagram 2.4.1)		
Each instruction is processed according to its length and type code (included in the pseudo-opcode). Implicit addresses are resolved by means of the using table.	IPKNA	NMACHOP
USING, DROP (Diagram 2.4.2)		
These instructions, which influence the using table, are processed.	IPKNA	NUSING NDROP
Address Constants and CCW (Diagram 2.4.3)		
Address constants and CCW instructions are processed at this stage. Implicit addresses are resolved by means of the using table.	IPKNA IPKNB	NSADDR

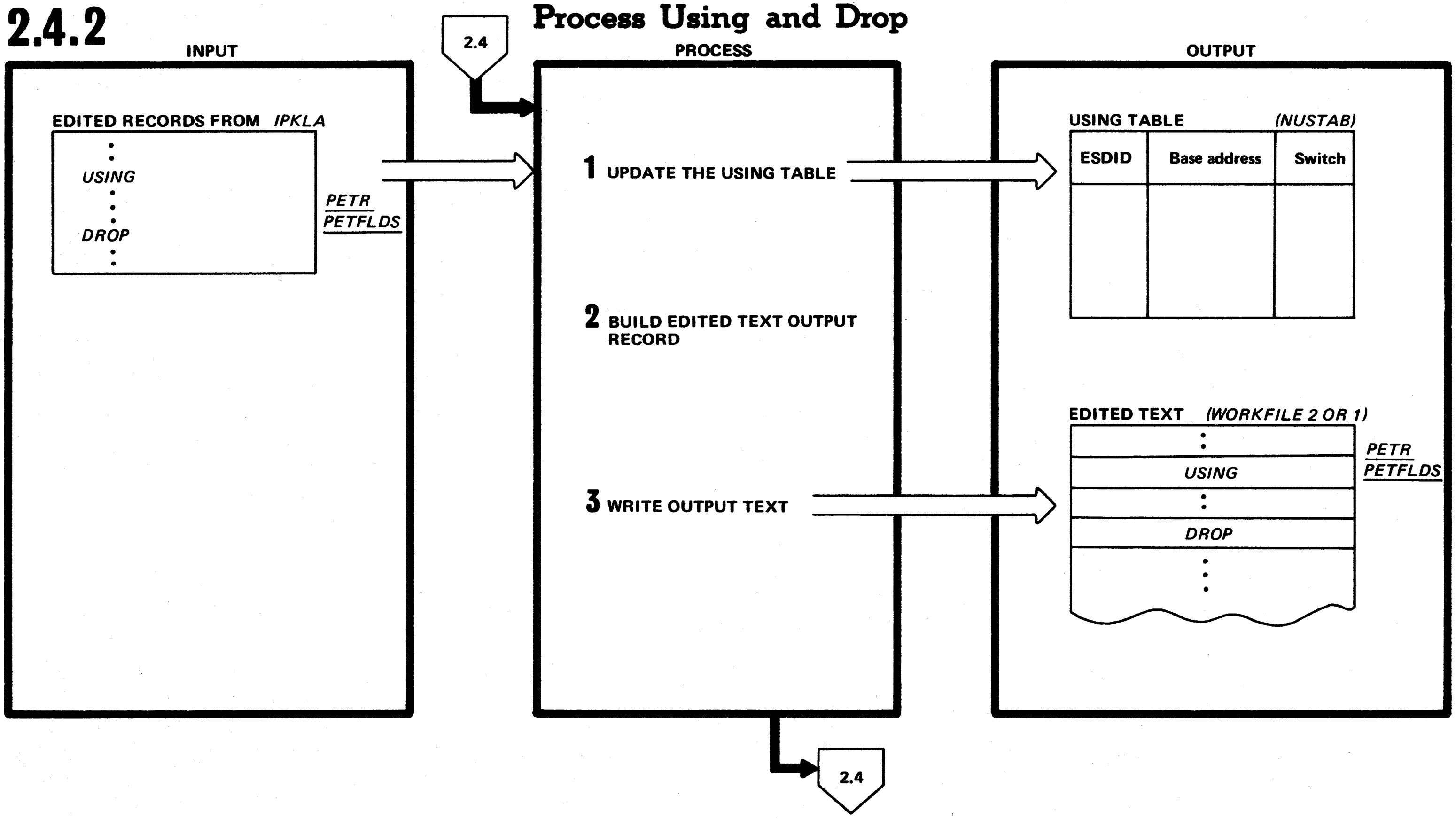
2.4.1 Process Machine Instructions



2.4.1

EXTENDED DESCRIPTION	MODULE	ROUTINE
1. The length of the instructions and the type code are used to determine which routine will handle the instruction and also to find an entry in the operand restriction table. The table contains information about the type of operand, allowed values, and where the value should be stored.	IPKNA	NMACHOP
2. The values of the operands are checked; if an error is detected, the error number and operand number are stored in an error table.	IPKNA	NTESTST1 NTESTST2
3. Implicit addresses are decomposed to base-displacement form by means of the using table. The table is searched for the register giving the smallest displacement among those available. If two registers give the same displacement, the higher-numbered register is used.	IPKNA	NADDRSPL
4. Object code for the instruction is built together with listing information and inserted in the edited text.	IPKNA	NMACHEND
5. The edited text record is written onto workfile 2, followed by error records, if any.	IPKNA	NMACHEND

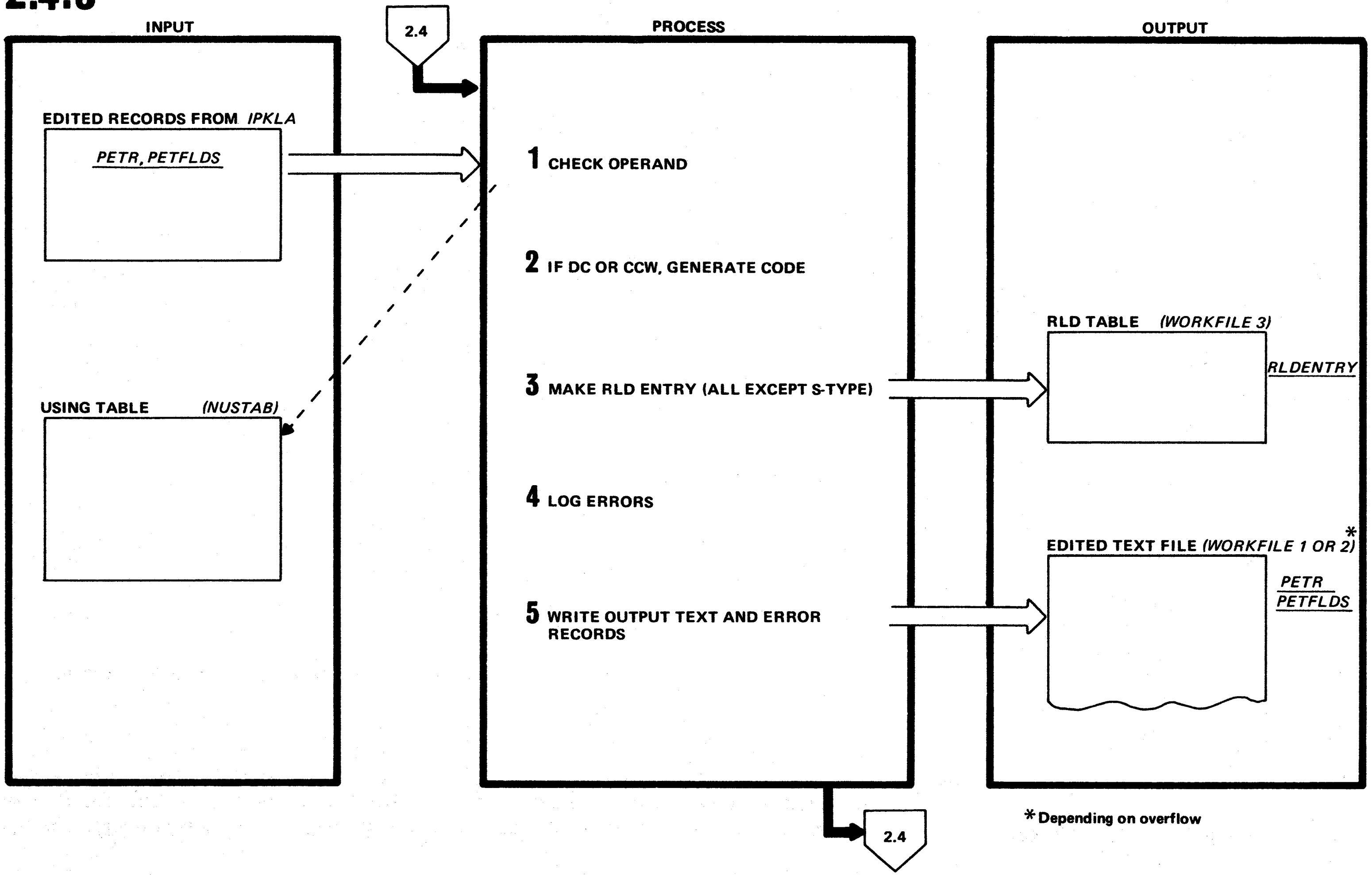
2.4.2 Process Using and Drop



2.4.2

EXTENDED DESCRIPTION	MODULE	ROUTINE
The using table has 17 entries. There is one entry each for registers 1-15 and two for register 0 (this is necessary because the absolute and relocatable case can occur simultaneously for register 0). Each entry consists of the ESDID, base address, and a switch indicating if the entry is used or not.		
<u>USING.</u> The operands are checked and the ESDID and the base address stored in the table.	IPKNA	NUSING
<u>DROP.</u> The operands are checked and the corresponding switches in the table are set to indicate that the registers are no longer used as base registers. If there are no operands, all entries are indicated to be free.	IPKNA	NDROP

2.4.3 Process Address Constants and CCWs

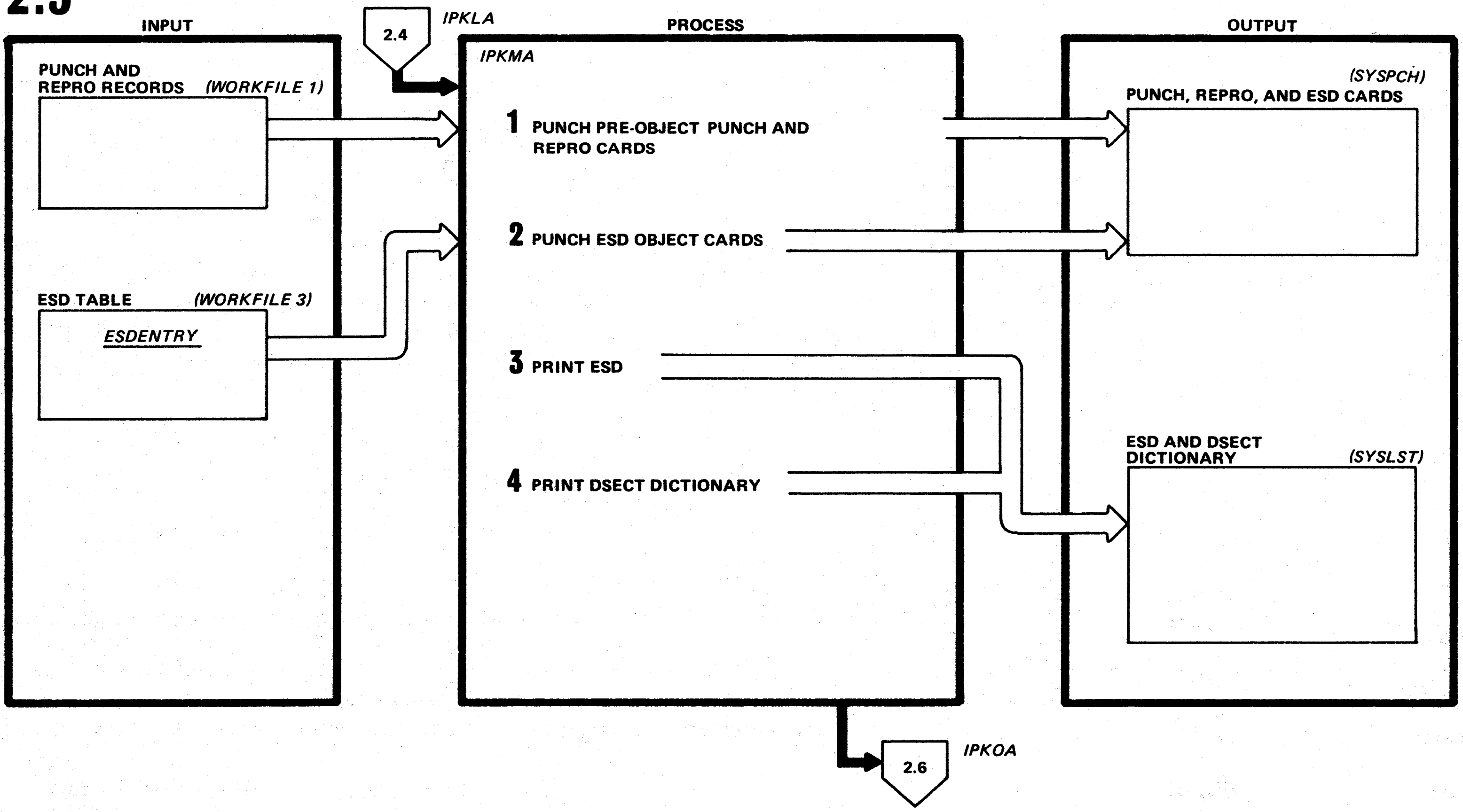


2.4.3

EXTENDED DESCRIPTION	MODULE	ROUTINE
1. In the case of an implicit address, the using table is used to split the address into base-displacement form; otherwise, the validity of the register and displacement are checked.	IPKNA IPKNB	NSADDR DCPROC
2. If the record is a DC or CCW, object code for the output edited record is generated.		
CCW	IPKNB	CCWPROC
S-type address constant.	IPKNA	
Other address constant.	IPKNB	DCPROC
3. An RLD entry is made for all instructions except S-type address constants.	IPKNB	WRITERLD
4. Errors are logged.	IPKOA	ERRLOG
5. Output records and error records are written.	IPKOA IPKNA	ERRPUT

2.5

Print/Punch the External Symbol Dictionary

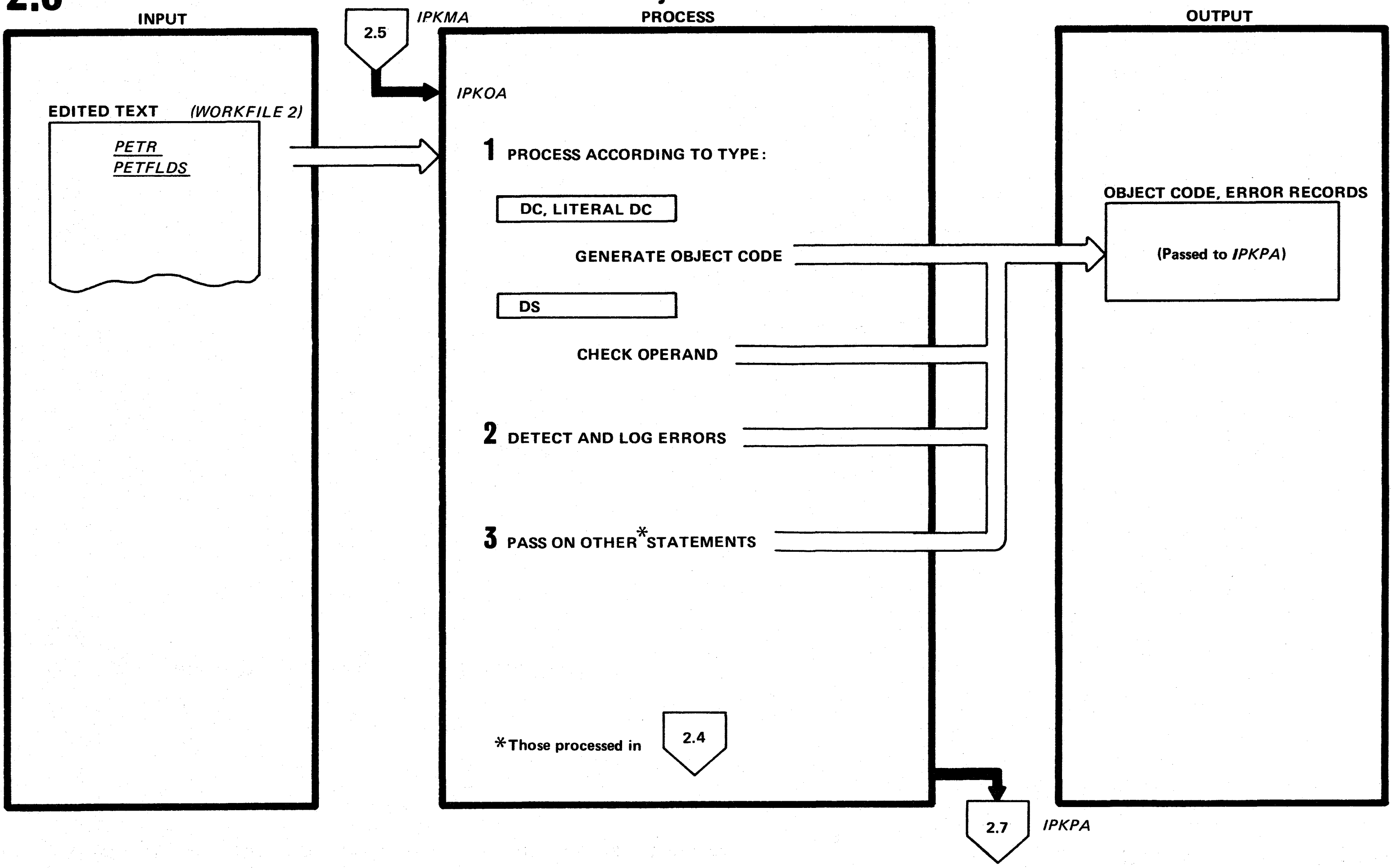


2.5

EXTENDED DESCRIPTION	MODULE	ROUTINE
1. PUNCH and REPRO records previously written on workfile 1 (see Diagram 2.1) are punched.	IPKMA	
2, 3. The ESD table is translated, punched, and printed.	IPKMA	ESDROUT
4. The DSECT dictionary is printed after the external symbol dictionary.	IPKMA	DSROUT

2.6

Build Object Code 2

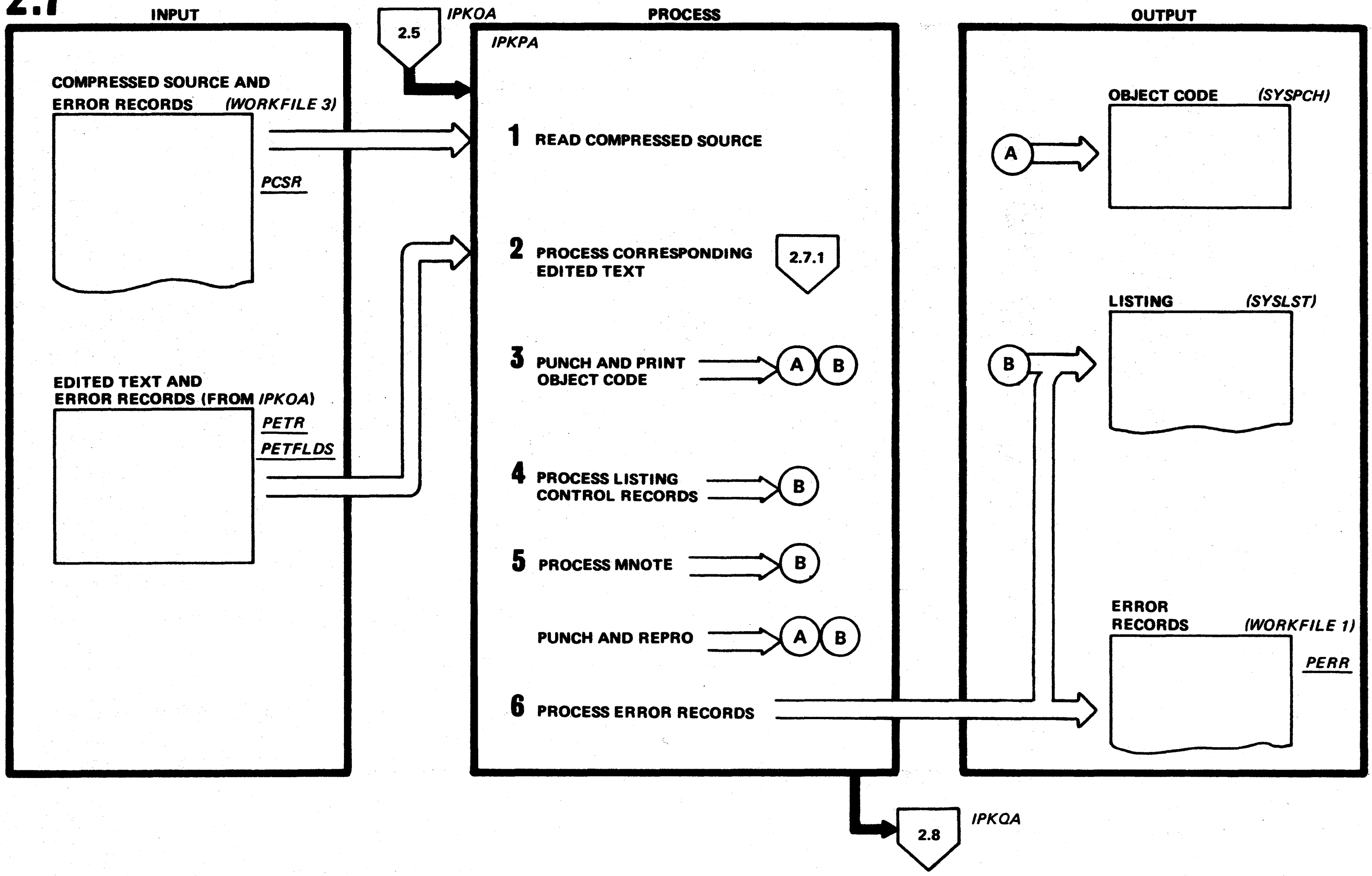


2.6

EXTENDED DESCRIPTION	MODULE	ROUTINE
1. Object code is built for DC and DS instructions		
DC, LITERAL DC	IPKOA	DCPROC
Object code is generated.		
DS	IPKOA	DCPROC
Operands are checked.		
2. Errors are logged and put out after the statement in error.	IPKOA	ERRLOG ERRPUT

2.7

Output


INPUT
COMPRESSED SOURCE AND ERROR RECORDS (WORKFILE 3)
PCSR
EDITED TEXT AND ERROR RECORDS (FROM IPKOA)
PETR
PETFLDS
2.5
IPKOA
PROCESS
IPKPA
1 READ COMPRESSED SOURCE
2 PROCESS CORRESPONDING EDITED TEXT
2.7.1
3 PUNCH AND PRINT OBJECT CODE
A
B
4 PROCESS LISTING CONTROL RECORDS
B
5 PROCESS MNOTE
B
PUNCH AND REPRO
A
B
6 PROCESS ERROR RECORDS
2.8
IPKQA
OUTPUT
OBJECT CODE (SYSPCH)
A
LISTING (SYSLST)
B
ERROR RECORDS (WORKFILE 1)
PERR

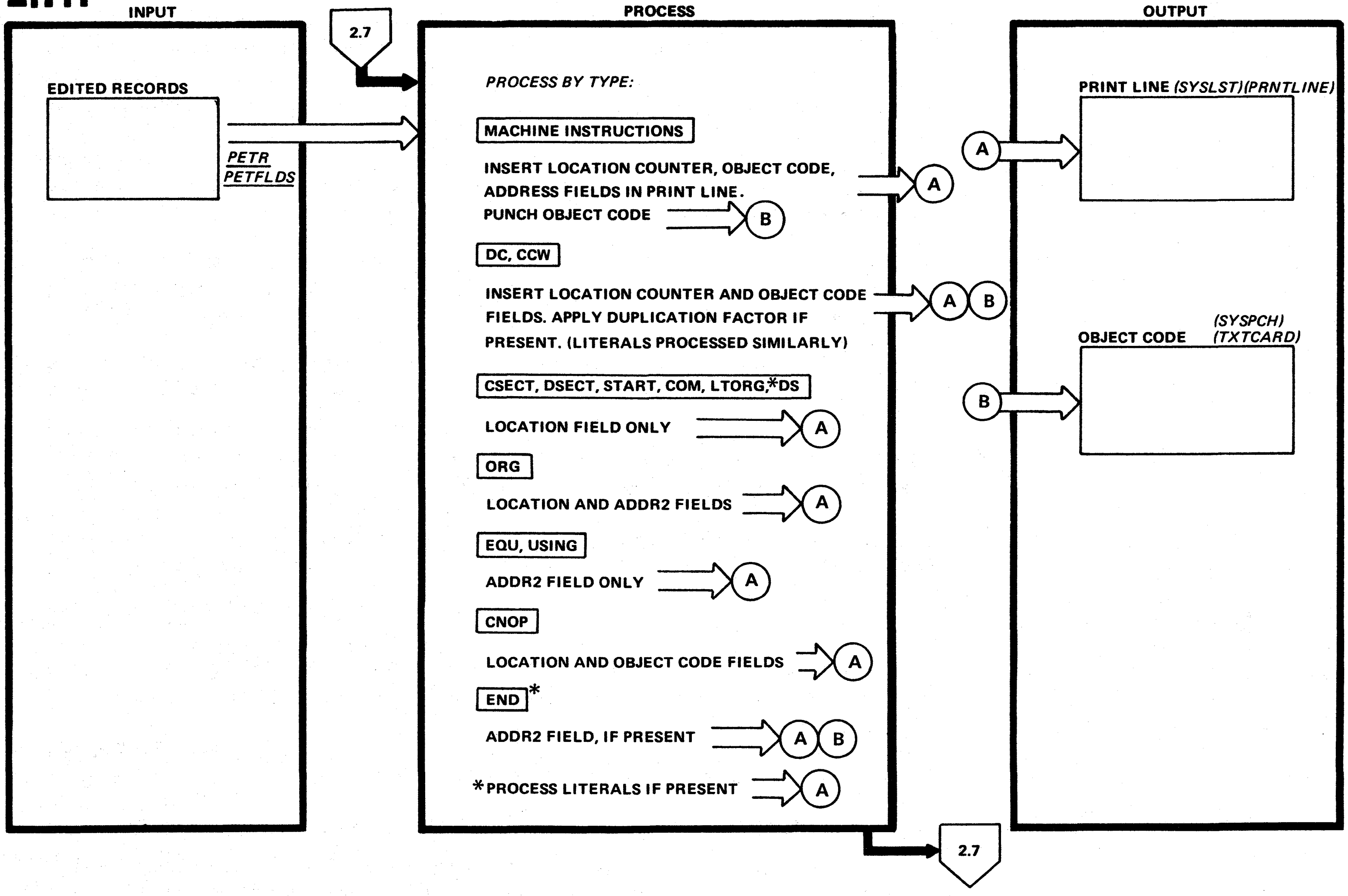
2.7

	EXTENDED DESCRIPTION	MODULE	ROUTINE
1.	Compressed source records are read from workfile 3.	IPKPA	GETSRC
2.	The compressed source records are checked against the last edited text record read and passed from IPKOA. If there is no corresponding edited text record (for example, if the compressed source record is for an erroneous machine instruction), the record is printed and the next one read.	IPKPA	
	When there is edited text for the record, it is processed as shown on Diagram 2.7.1.		EDTEXT
3.	The code developed in the previous step (Diagram 2.7.1) is printed for the listing and punched for the object deck.	IPKPA	PUNCHOUT DUMP PRINTER
4.	Listing control statements are processed according to type:		
	TITLE. The heading is replaced and the page eject control character is set.* The symbol name is inserted if it appears on the first title statement.	IPKPA	TITLEOP
	EJECT. The page eject control character is set.*		EJECTOP
	SPACE. The control character for spacing** is set according to the number of lines indicated by the operand. When the space operand exceeds two, blank lines are printed until fewer than three spaces remain before the next print line.		SPACEOP
	PRINT: The print switches that control the printing of all statements, generated statements, and object code, are updated when a PRINT statement is encountered. The PRINT statement itself is always printed regardless of the status of the print switches.		PRINTOP
	Errors in List Control Statements. An error record is built and written on workfile 1. The statement in error is printed. The requested operation is performed if the error is a minor one.		WRTERROR
5.	**MNOTE.** The message is printed on SYSLST and an error message written on workfile 1.	IPKPA	MNOTEOP
	PUNCH, REPRO. The statement(s) is printed on SYSLST and the card(s) punched on SYSPCH. These cards are intermixed with object deck cards.		PUNCHOP REPROOP
6.	The text ***ERROR*** is printed on a separate line after each statement that contains one or more errors. An error record containing the statement number is built for the diagnostic phase and written on workfile 1.	IPKPA	COPERROR

* When the page eject control character is set, a page eject will first be performed before the next line is printed.

**The next printed line will be preceeded by one or two extra blank lines.

2.7.1 Process Edited Text

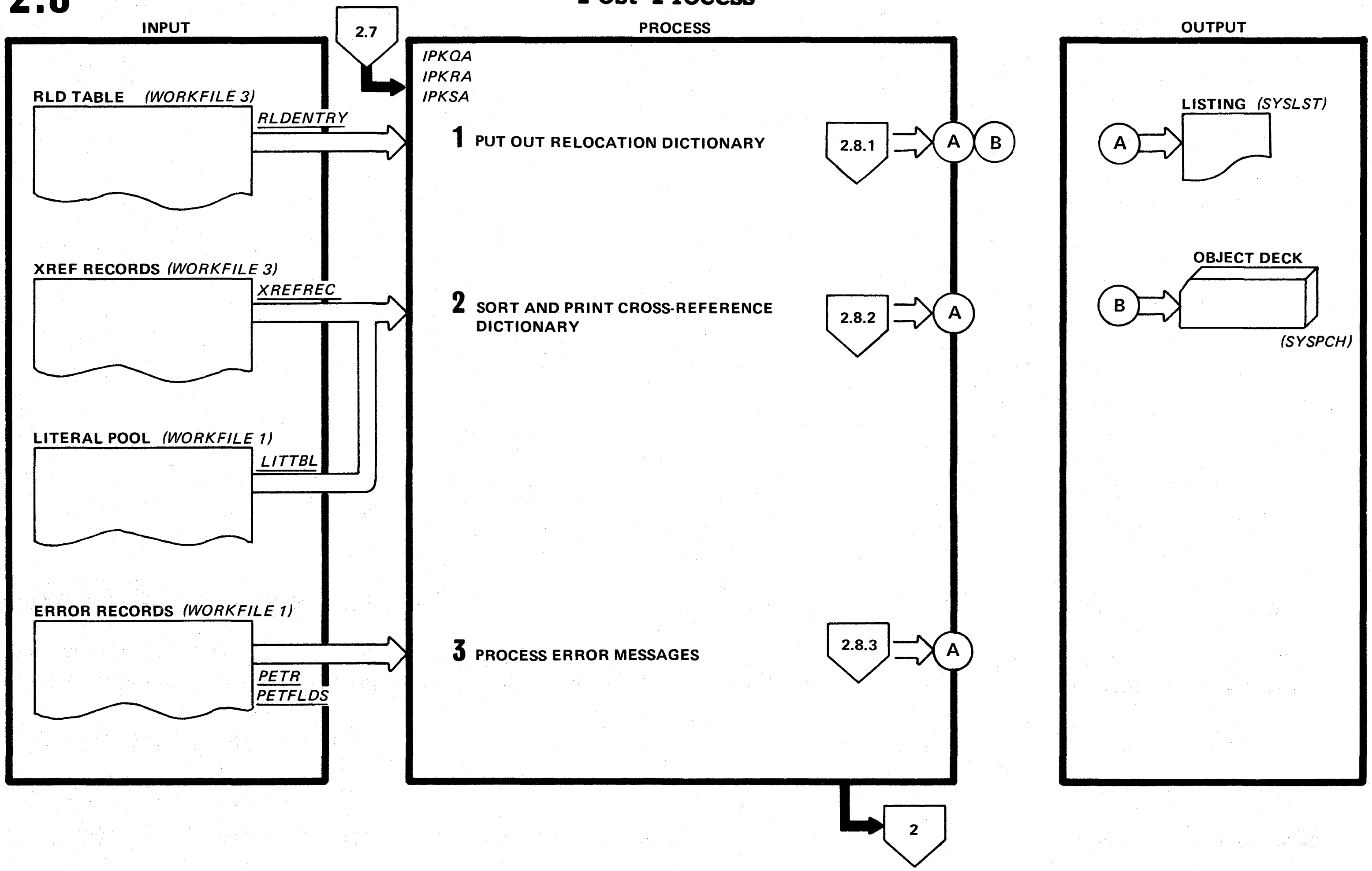


2.7.1

EXTENDED DESCRIPTION	MODULE	ROUTINE
MACHINE INSTRUCTIONS		
The location counter value, object code, and address fields are inserted in the print line.	IPKOA	MACHOPS
Object code is punched.	IPKPA	DUMP
ALL OTHER INSTRUCTION TYPES		
All other instructions are handled by type according to a branching table. Literals are formatted for the listing.	IPKOA IPKOA	PSEUTBL LITCOPE
Object code is punched for some instructions.	IPKPA	DUMP

2.8

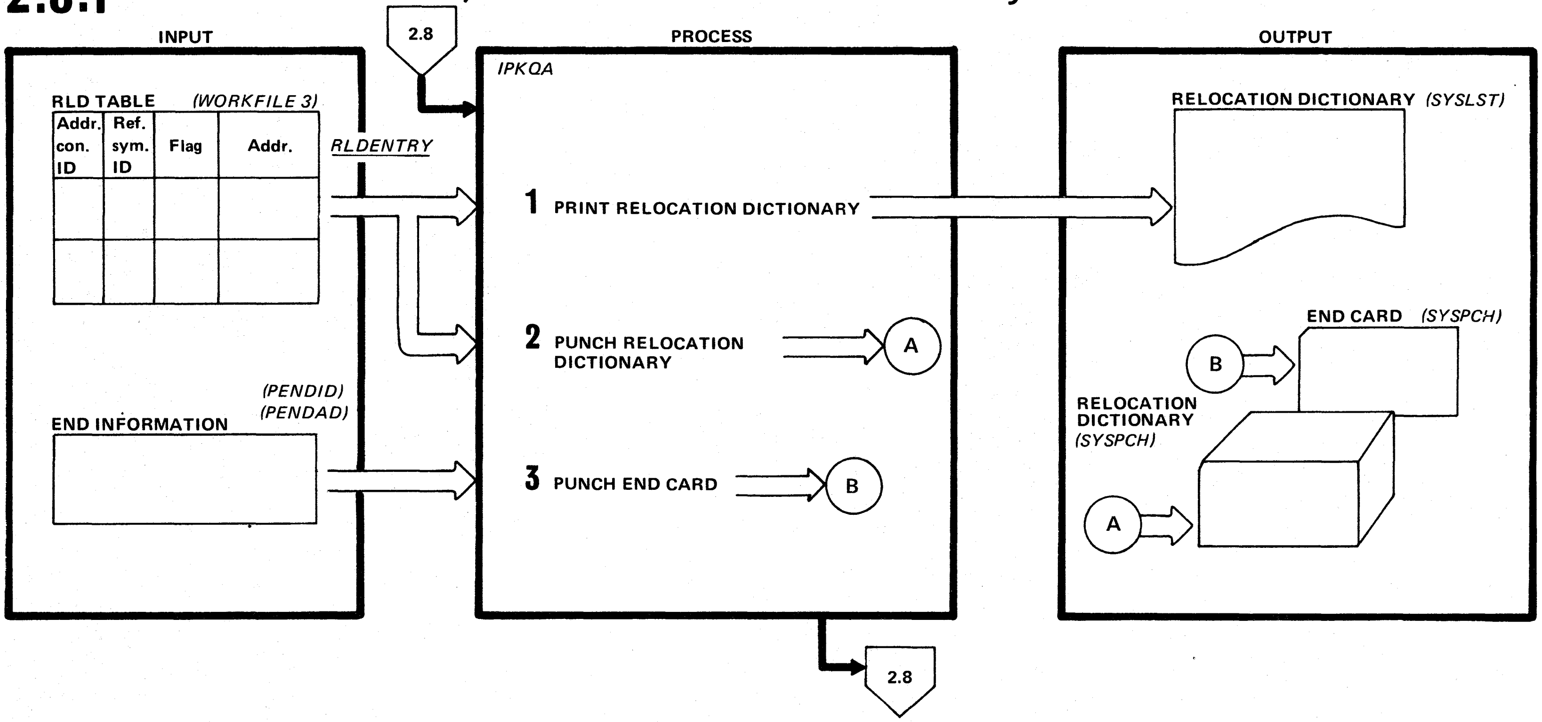
Post Process



2.8

EXTENDED DESCRIPTION	MODULE	ROUTINE
1. The relocation dictionary is printed and punched (see Diagram 2.8.1).	IPKQA	RLDROUT
2. The literal pool and cross reference records are merged; they are then sorted and printed on SYSLST (see Diagram 2.8.2).	IPKRA	
3. Error messages are processed and printed (see Diagram 2.8.3).	IPKSA	DECODE PRINTER

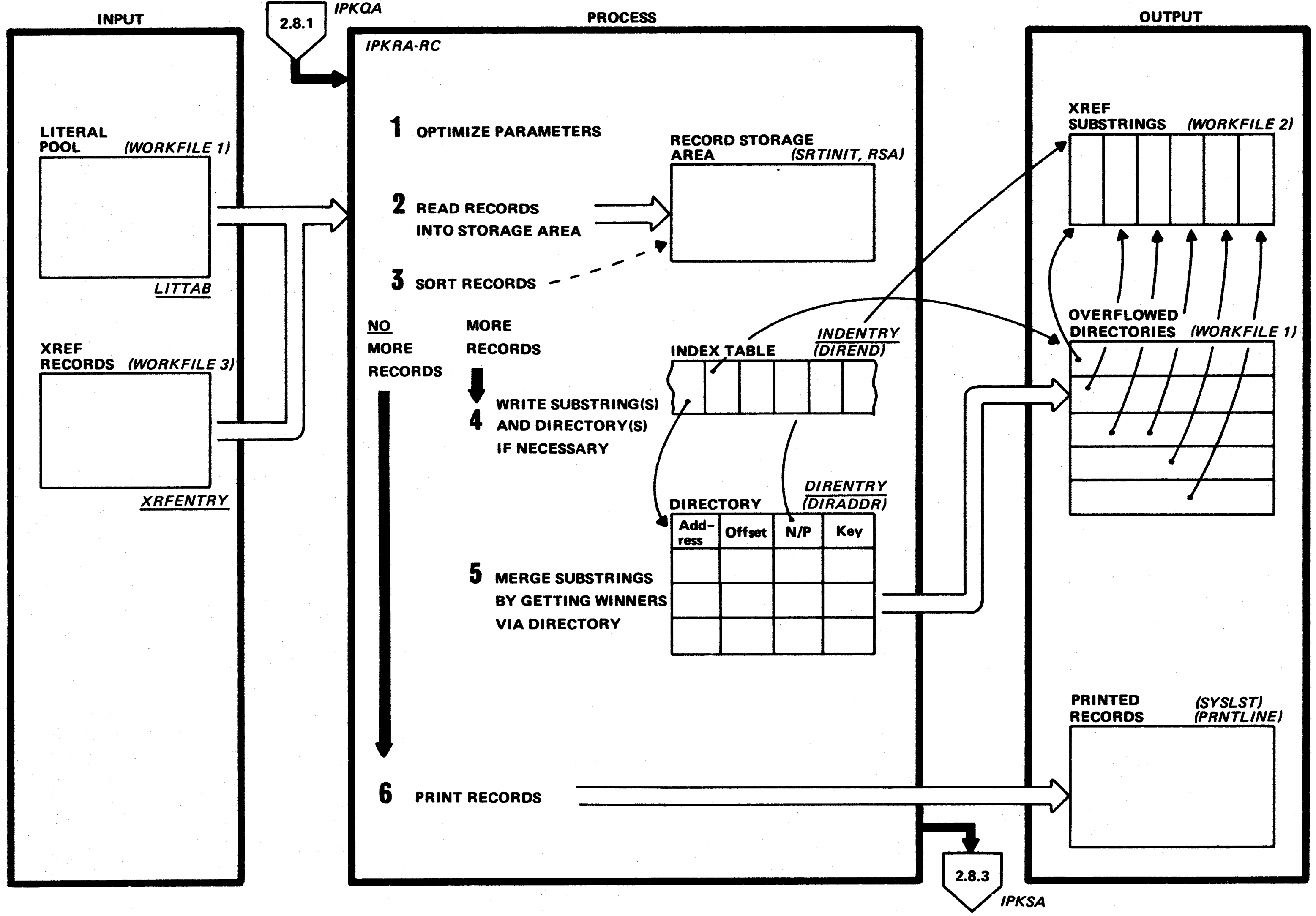
2.8.1 Print/Punch the Relocation Dictionary



2.8.1

EXTENDED DESCRIPTION	MODULE	ROUTINE
1. The RLD information is translated to output format and printed two columns per page.	IPKQA	RLDROUT
2. The RLD object cards are punched.	IPKQA	RLDPCH
3. An END card is punched. If there was a name in the operand field of the END statement, its ID and address is punched.	IPKQA	

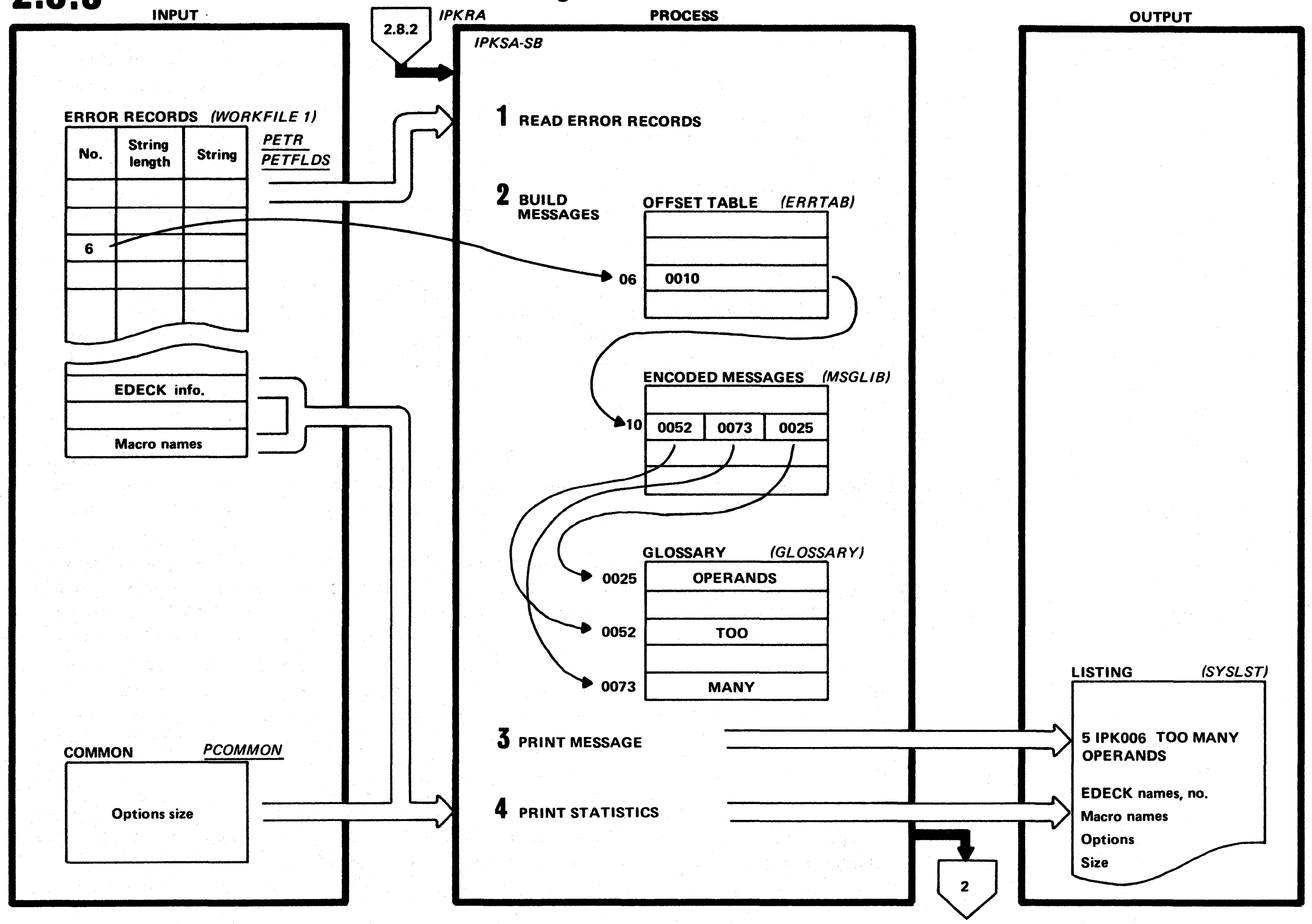
2.8.2 Sort and Print the Cross-Reference Dictionary



2.8.2

EXTENDED DESCRIPTION	MODULE	ROUTINE
The cross-reference sort is done in two phases. In the first phase, one or more sorted substrings are built. If there is more than one string, they are written on workfile 2. A directory entry is created for each substring, containing the physical disk address and the lowest key number in the string. If the directory overflows, an entry is made in the index table, consisting of the lowest key in the directory and the physical disk address for the directory.		
In the second phase, "M" blocks are read and the records retrieved in ascending order using the directory, which is then updated according to ascending keys. When necessary, the next directory is read into main storage and merged with the first one. If there is only one string, the print module is fetched and the records passed to the print buffer one by one.		
1. The total number of bytes to be sorted is checked against available storage to determine if an "in-storage" sort is possible. If it is not, the internal sort block size "B" and merge order "M" are calculated with the respect to the number of strings to be sorted. Finally, all addresses to the I/O buffers, directories, index table, and record storage are initialized.	IPKRA	SRTINIT
2. Since the literal cross-reference records are generated with a pseudoname instead of a symbol name, the literal pool is read and the corresponding pseudoname is built and concatenated with each actual literal.	IPKRA	SRTLIT
3. The internal sort technique used is Shell's sort.	IPKRA	SRTRSA
4. If the sort is not an " in-storage" sort, the Conner merge technique is used. Each sorted substring is written and a directory entry containing the lowest key of the substring and its physical address on disk is created. If the directory overflows, it is written on workfile 1 and a new directory built. Each time the directory overflows, an entry is made in the index table. The entries in the directory are in ascending order according to key number.	IPKRA	SRTOUT SRTDIR
5. Phase 2 is now loaded and "M" blocks read into storage together with the first (or only) directory block. The winner pointed to by the first directory entry is passed to the print routine and the directory is updated according to the next key. If the new winner is not in storage, the corresponding block will be read and the winner put out. If a winner record is not pointed to by a directory entry in main storage, the next directory block will be read by using the index table and the two directories merged.	IPKRB	MRGMAIN MRGDIR
6. The records are put out.	IPKRA, RB or RC	MRGPRT

2.8.3 Diagnostics and Statistics

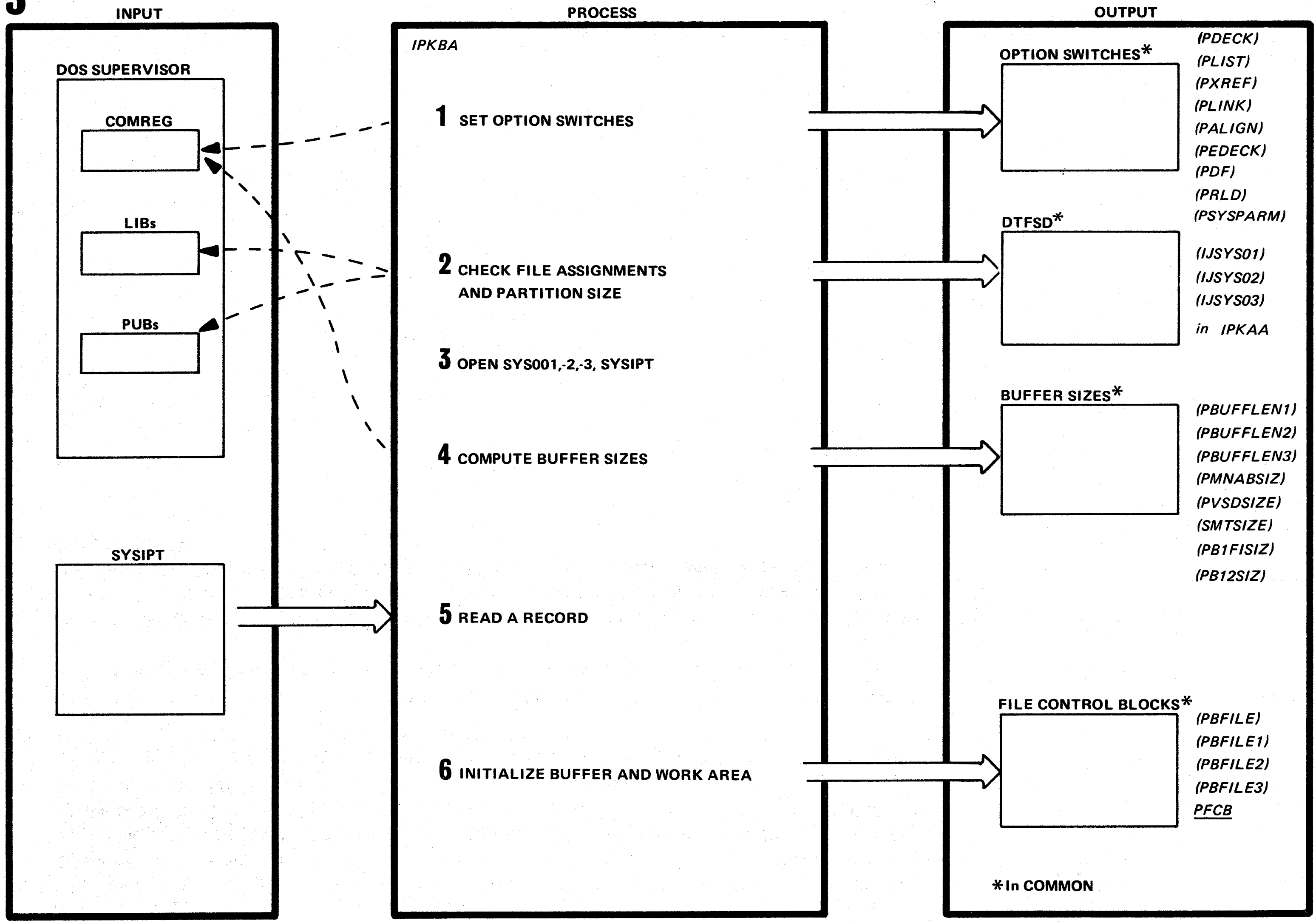


2.8.3

	EXTENDED DESCRIPTION	MODULE	ROUTINE
1.	Error records are read from workfile 1.	IPKSA	
2.	The message is built up in the following way: the error record contains a number which corresponds to an entry in the offset table. This entry points to a corresponding "encoded message". The messages are coded so that each word in the message is represented by a two-byte code; this code is the offset of the actual English word in a "glossary", where all the different words are kept in EBCDIC code, each word preceded by one byte that contains its length in bytes. Error records that contain strings are handled by inserting the string into a special area in the glossary. It (the string) is then handled like an ordinary glossary entry.	IPKSA IPKSB	DECODE
3.	The messages are printed on SYSLST immediately after the statement in error.	IPKSA	PRINTER
4.	Statistics are printed. A summary of errors found in the assembly is printed from workfile 1; the macro name and number of cards punched for the EDECK option are printed from workfile 1. The names of macros called is printed from workfile 1. Assembler options in effect and the partition size are printed from COMMON.	IPKSB	

3

Initialize

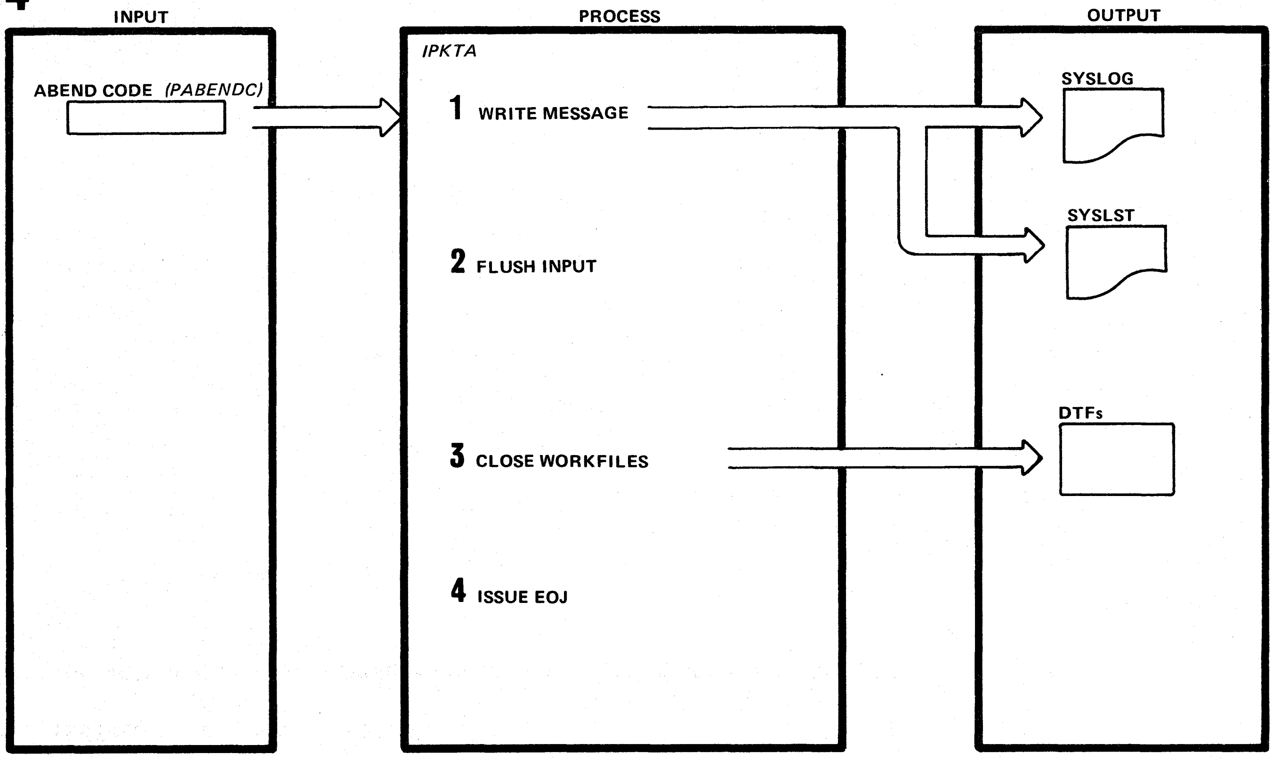


3

	EXTENDED DESCRIPTION	MODULE	ROUTINE
1.	The assembler options, which have been passed to the communications region of the DOS supervisor from ASSGN cards and from the options chosen at system generation, are used to set option switches in COMMON.	IPKBA	INOPT
2.	File assignments and partition size are checked and used to set values in the DTFSD. Errors cause an ABEND.		INFILE INPARSIZ
3.	The workfiles and the input files are opened.		INOPEN
4.	Buffer sizes are computed from information in the DOS supervisor communication region, the DTFSD, and from the overlay switches.		INBUFSZ
5.	The first record is read from SYSIPT.		INREC
6.	The buffer and work area addresses are initialized for the first three phases.		INITCDE

4

ABEND


INPUT
ABEND CODE (PABENDC)
PROCESS
IPKTA
1 WRITE MESSAGE
2 FLUSH INPUT
3 CLOSE WORKFILES
4 ISSUE EOJ
OUTPUT
SYSLOG
SYSLST
DTFs

4

EXTENDED DESCRIPTION	MODULE	ROUTINE
The ABEND function is called when any of the following have been detected:		
● Workfile I/O error	IPKTA	
● End of workfile extent		
● Too many macros or global symbols for the partition		
● I/O unit required but not assigned		
● Incorrect workfile unit type		BADASGN
● Too small a partition		
1. The ABEND routine writes the appropriate error message on SYSLST and SYSLOG before terminating the job.	IPKTA	MESSROUT
2. Further input is flushed.		
3. All workfiles are closed.		
4. An end-of-job command is issued.		

Program Organization

Purpose of the Section

The purpose of this section is to describe the structure of the assembler: how it is divided into phases, the order in which these phases are loaded into main storage and given control, and how control and data are passed from one phase to another.

This section contains:

- Phase/control section/object module directory
- Summary of the functions of each phase
- Control and data flow between phases
- Allocation of main storage for the phases
- Main storage layouts
- Common data area for the assembler

Phase/Control Section/Object Module Directory

Phase	Control section	Object module	Description of the object module
ASSEMBLY	IPKAD000	IPKAD	SYSSLB logic module (DTFSL)
	IPKAJ000	IPKAJ	Assembler identifier
	IPKBA000	IPKBA	Initializer
	IPKAB000	IPKAB	SYSIPT and SYSSLB input
	IPKAG000	IPKAG	SYSIPT logic module (CPMOD)
	IPKAA000	IPKAA	Basic interface routines and common data area
	IPKAA002	IPKAA	Workfile logic module
ASSECA	IPKCA001	IPKCA	Input for assembler
	IPKCB000	IPKCB	Op-code table and op-code look-up
	IPKCC000	IPKCC	Macro instruction and macro prototype editor
	IPKCD001	IPKCD	Overlay for TITLE, ISEQ, COPY, BKEND, and EOF on SYSSLB
ASSEDA	IPKDA000	IPKDA	Conditional assembly editor
	IPKDB000	IPKDB	Variable symbol declaration processor
ASSEEA	IPKEA000	IPKEA	Sequence symbol resolution
ASSEGA	IPKGA000	IPKGA	EDECK output
	IPKAC000	IPKAC	Punch routine
	IPKAH000	IPKAH	SYSPCH logic module (CPMOD)
ASSEFA	IPKFA000	IPKFA	Global edit
	IPKAE000	IPKAE	SYSSLB input routines
	IPKAD000	IPKAD	SYSSLB logic module (DTFSL)

Figure 1. Phase/Control Section/Object Module Directory. This figure shows how the phases are divided into control sections and object modules.
(Part 1 of 2)

Phase	Control section	Object module	Description of the object module
ASSEHA	IPKHA000	IPKHA	Attribute collection
ASSEIA	IPKIA000 IPKAA001 IPKAA003 IPKCB000 IPKIC000	IPKIA IPKAA IPKAA IPKCB IPKIC	Generate POINT with byte offset routine POINT with byte offset routine for FBA Op-code table and op-code look-up Op-code substitution
ASSEJA	IPKJA000	IPKJA	Assembler pre-processor and literal processor
ASSEKA	IPKKA001 IPKKA000	IPKKA IPKKA	Assignment initializer Assignment
ASSELA	IPKLA000 IPKNA000 IPKNB000	IPKLA IPKNA IPKNA	Substitution Build code 1 for machine instructions and S-type constants Build code 1 for address constants (A,V, and Y) and CCWs
ASSEMA	IPKMA000 IPKAF000 IPKAI000 IPKAH000	IPKMA IPKAF IPKAI IPKAH	External symbol dictionary output (ESD) Punch routine Print routine SYSPCH/SYSLST/SYSLNK logic module (CPMOD)
ASSEOA	IPKOA000 IPKPA000	IPKOA IPKPA	Build code 2 for constants (except A, V, Y, and S-type) Text output
ASSEQA	IPKQA000	IPKQA	Relocation dictionary output (RLD)
ASSERA	IPKRA000	IPKRA	Cross-reference sort and print (XREF)
ASSERB	IPKRB000	IPKRB	Cross-reference merge and print (XREF)
ASSERC	IPKRC000	IPKRC	Cross-reference print (XREF)
ASSESA	IPKSA000 IPKSB000	IPKSA IPKSB	Diagnostics output Diagnostics and statistics output
ASSETA	IPKTA000	IPKTA	ABEND routine

Figure 1. Phase/Control Section/Object Module Directory. (Part 2 of 2)

Summary of the Functions of Each Phase

The following figure lists the functions accomplished in each phase of the assembler. Some of these functions are broken down into subfunctions. For a description of how the phases work see "Method of Operation".

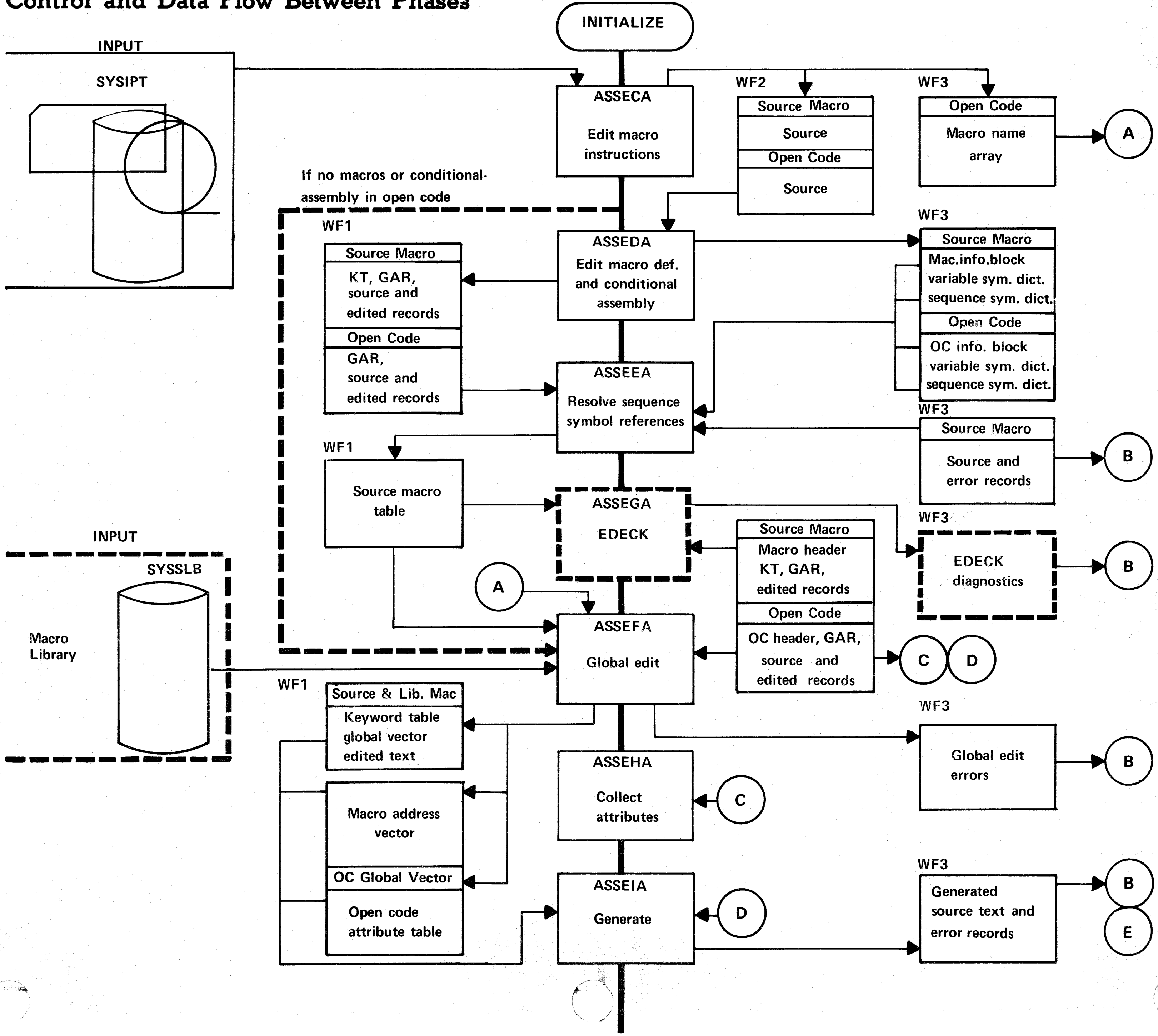
Phase	Diagram	Function
ASSEMBLY	none	• Check file assignments • Open workfiles and SYSIPT • Compute buffer sizes • Perform I/O
ASSECA	1.1	• Read all source and compress text • Look up operation codes • Build macro name array (MNA) • Edit macro instructions and prototypes
ASSEDA	1.1.2	• Process variable symbol declarations • Edit conditional assembly statements • Collect sequence symbol declarations • Complete macro instruction editing
ASSEEA	1.1.3	• Resolve all sequence symbol references • Set up source macro header and tables • Separate compressed source records (CSR) and edit text for source macros • Build source macro table (SMT)
ASSEGA	1.2	• Punch source macros in edited format
ASSEFA	1.3	• Build a global vector (GV) • Build the macro address vector (MAV)
ASSEHA	1.4	• Look up attributes for all parameters and all symbols with attribute references in open code
ASSEIA	1.5	• Expand macro instructions • Evaluate conditional assembly expressions • Perform conditional assembly • Perform substitution
ASSEJA	2.1	• Edit all machine and assembler instructions • Build literal pools after each LTORG and END • Output cross-reference records for all literals • Write on workfile 1 any PUNCH and REPRO records found before first control section
ASSEKA	2.2	• Assign values to all symbols • Build symbol table • Build external symbol dictionary (ESD) table • Evaluate length of EQU,CNOP,ORG, and END expressions • Output cross-reference records for all symbol definitions, references, and duplicates

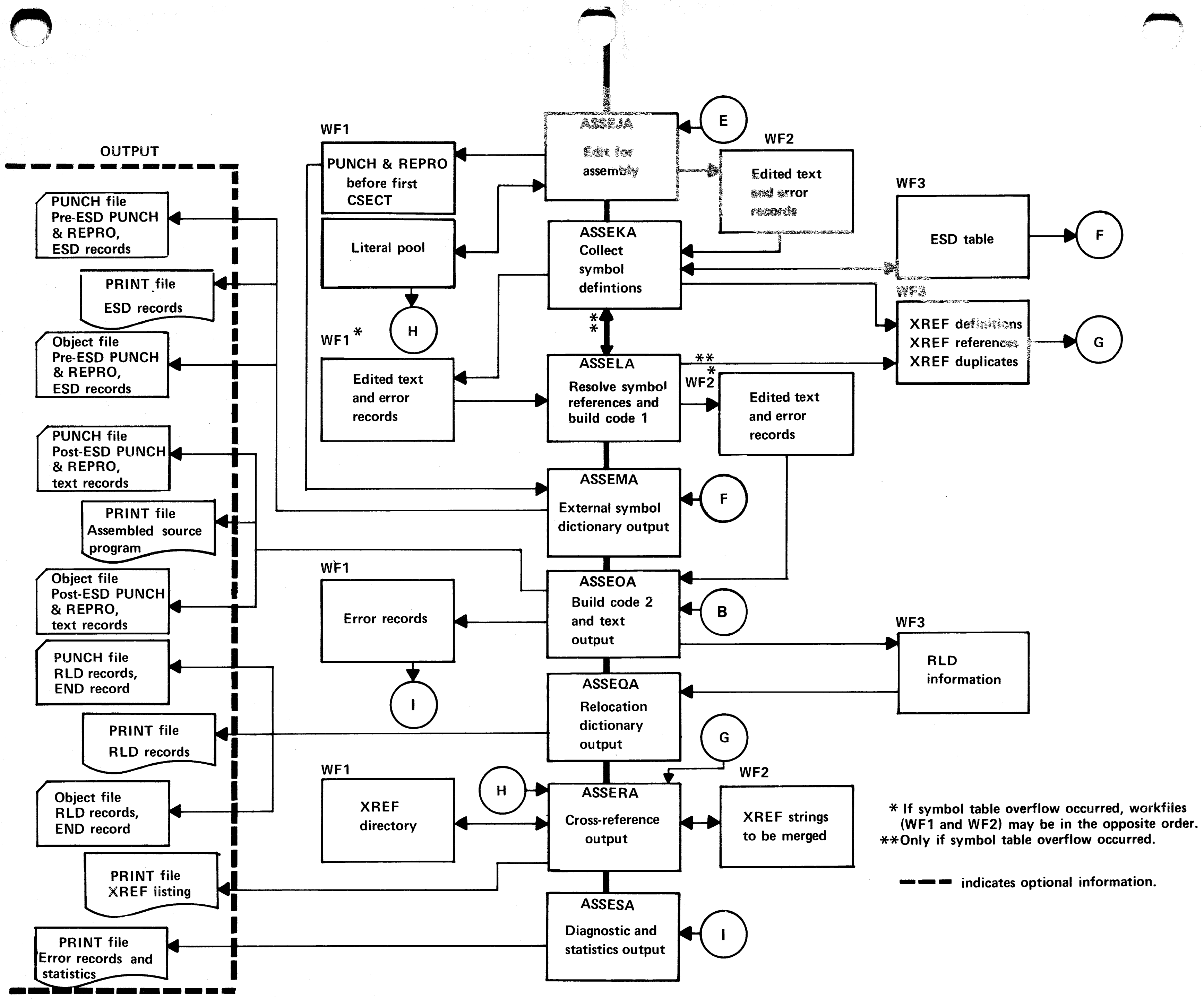
Figure 2. Summary of the Functions of Each Phase. (Part 1 of 2)

Phase	Diagram	Function
ASSELA	2.3	• Substitute values for each symbol • Evaluate all expressions • Handle USING and DROP statements • Convert implicit addresses into base-displacement form • Build all object output for machine instructions and S-type constants • Collect relocation dictionary (RLD) information for RLD output • Build all object output for address constants (A, V, and Y) and CCWs
ASSEMA	2.5	• Output cards for PUNCH and REPRO records found <u>before</u> first control section • Output ESD cards, ESD, and DSECT listing
ASSEOA	2.4	• Build object output for all constants (except A, V, Y and S-type) • Merge source and edited text • Output text listing • Output text cards
ASSEQA	2.8.1	• Output RLD cards and listing • Output END card
ASSERA-RC	2.8.2	• Sort cross-reference (XREF) records and print XREF listing
ASSESA	2.8.3	• Output error messages and statistics

Figure 2. Summary of the Functions of Each Phase. (Part 2 of 2)

Control and Data Flow Between Phases





* If symbol table overflow occurred, workfiles (WF1 and WF2) may be in the opposite order.
**Only if symbol table overflow occurred.

▬ ▬ ▬ indicates optional information.

Figure 3. Control and Data Flow Between Phases

Allocation of Main Storage for the Phases

The vertical axis of the diagram below represents the amount of main storage available to the partition. The horizontial axis represents time (the order in which the phases are loaded and executed in main storage). Certain parts of the ASSEMBLY phase, (for example, basic interface routines and the workfile logic module) are in main storage throughout execution. Certain parts of the ASSEMA phase (the print routine and the SYSPCH/SYSLST/SYSLNK logic module (CPMOD)) are loaded into storage by ASSEMA and remain in storage to the end of execution. The shaded portion of the diagram represents the area of the partition occupied by the work areas, buffers, dictionaries, tables, etc., of the phases; the size of this part of storage is variable depending upon the size of the partition.

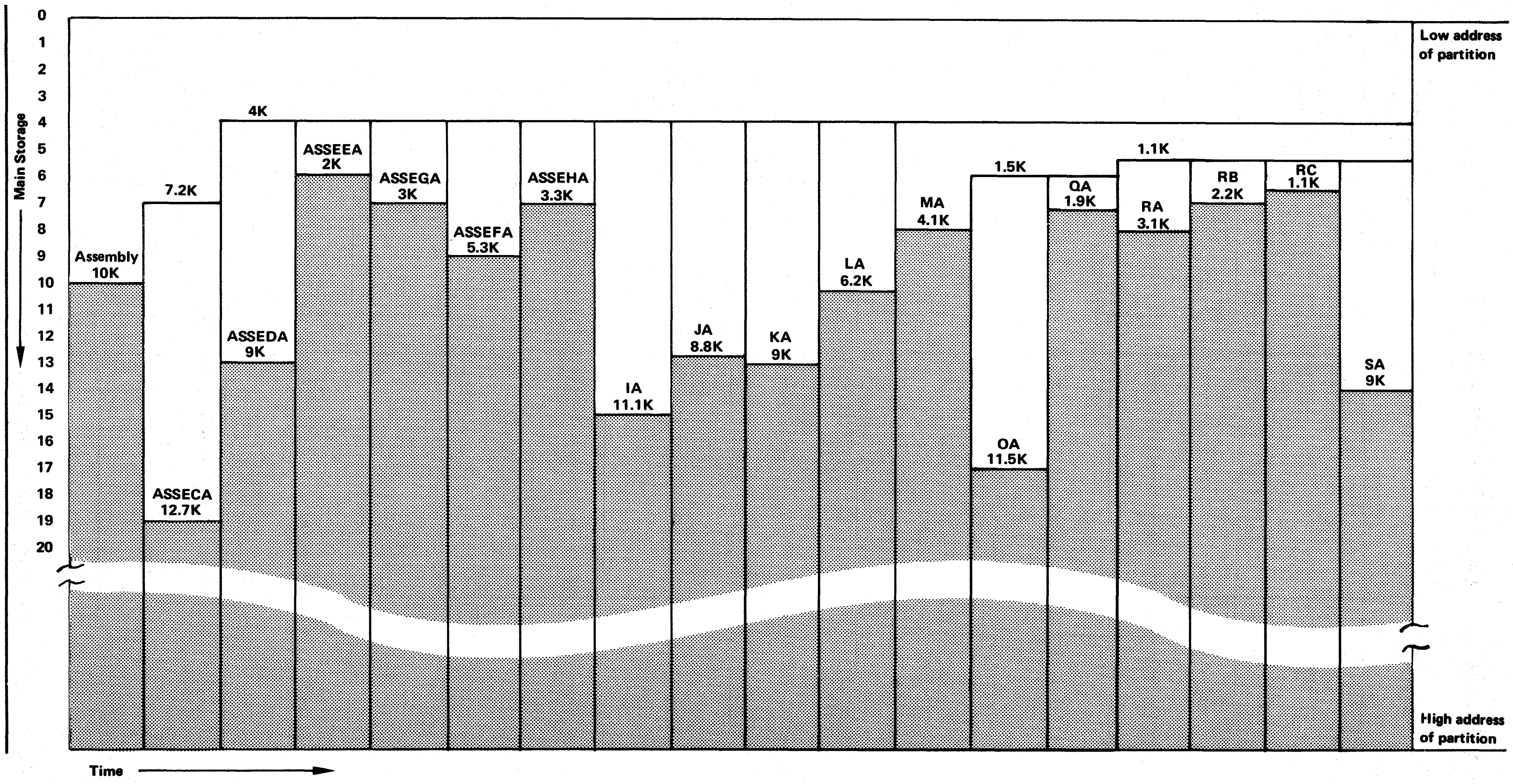


Figure 4. Allocation of Main Storage for the Phases

Main Storage Layouts of the Phases

The following figures illustrate the contents and layouts of the phases as they are loaded into main storage. For a cross-reference to the order in which they are loaded and their various sizes, see Figure 4. The contents of the COMMON interface phase "ASSEMBLY" are shown in Figure 1. Workareas, buffers, etc., generally begin at the high storage address and work downwards using only as much of the available storage as they require.

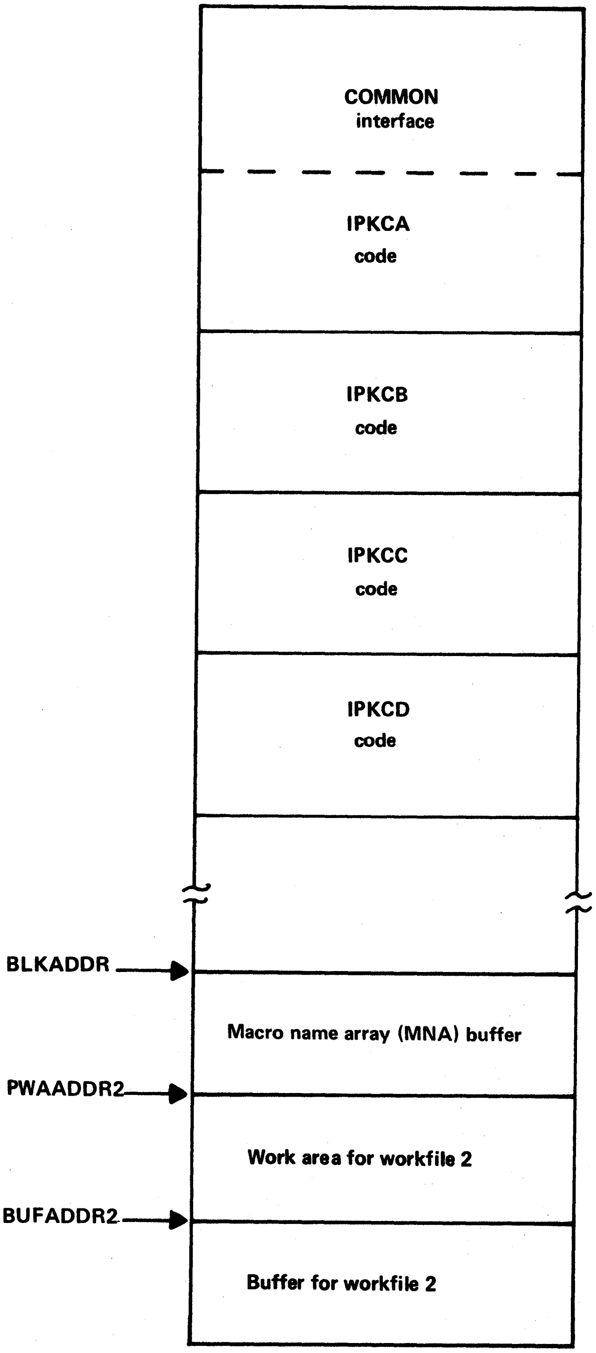
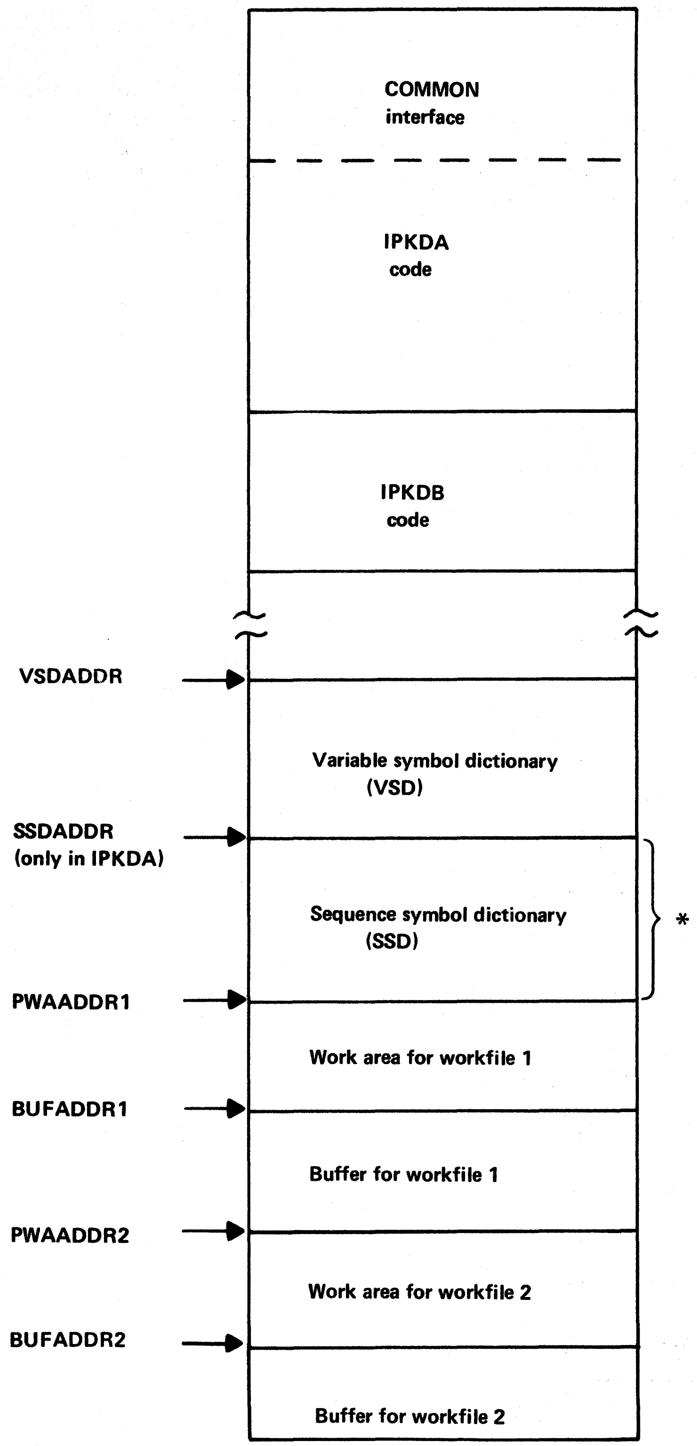


Figure 5. ASSECA Main Storage Layout.



* This area is overlaid by the VSD in module IPKDB

Figure 6. ASSEDA Main Storage Layout.

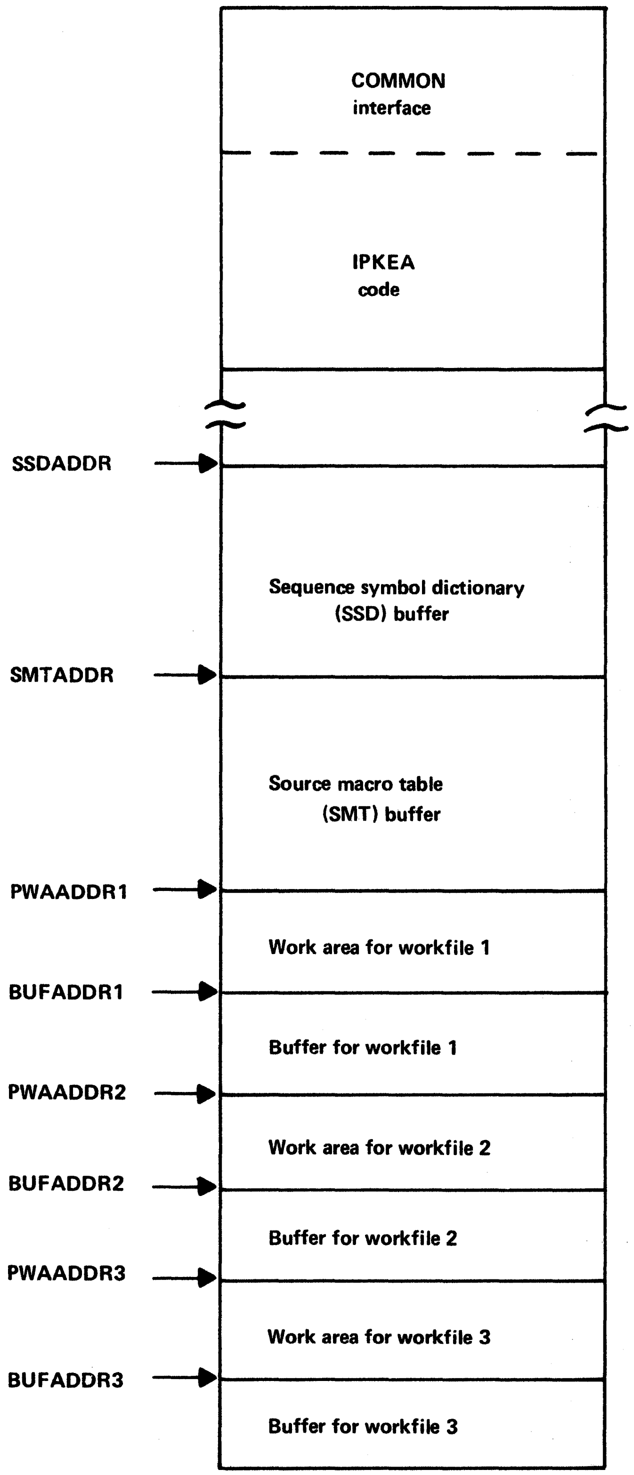


Figure 7. ASSEEA Main Storage Layout.

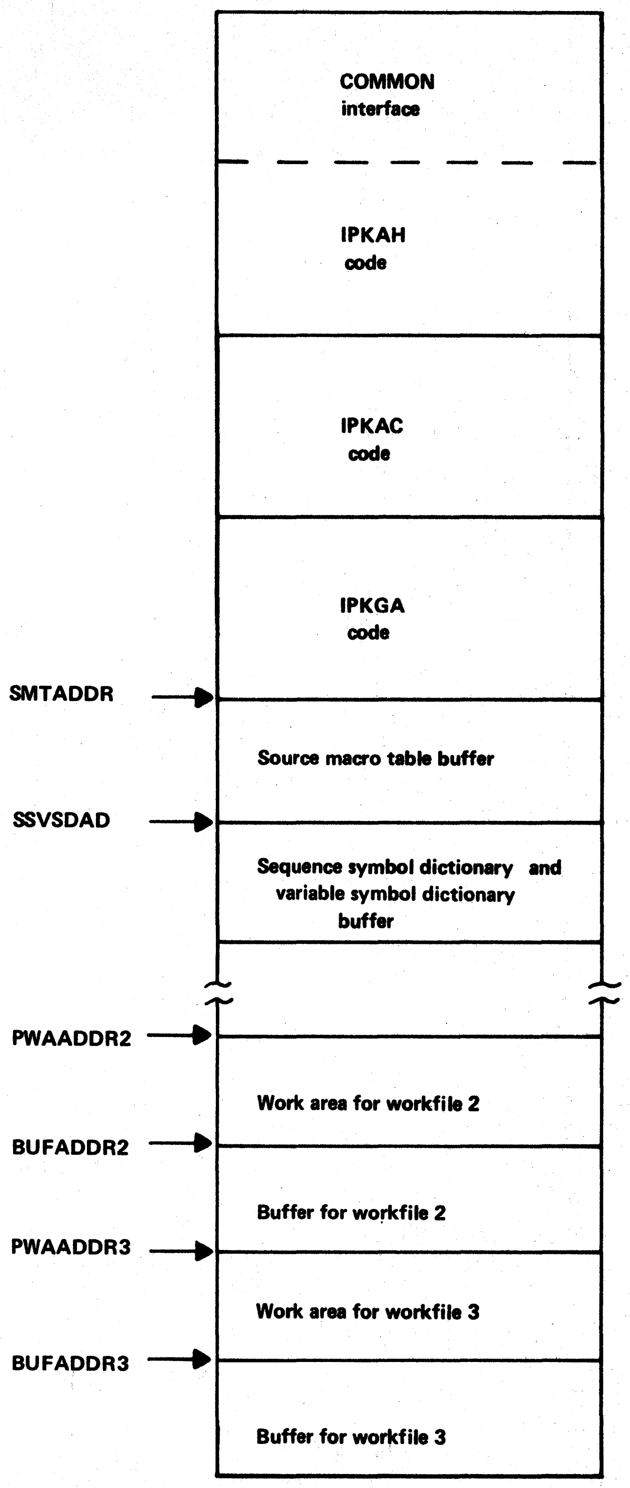
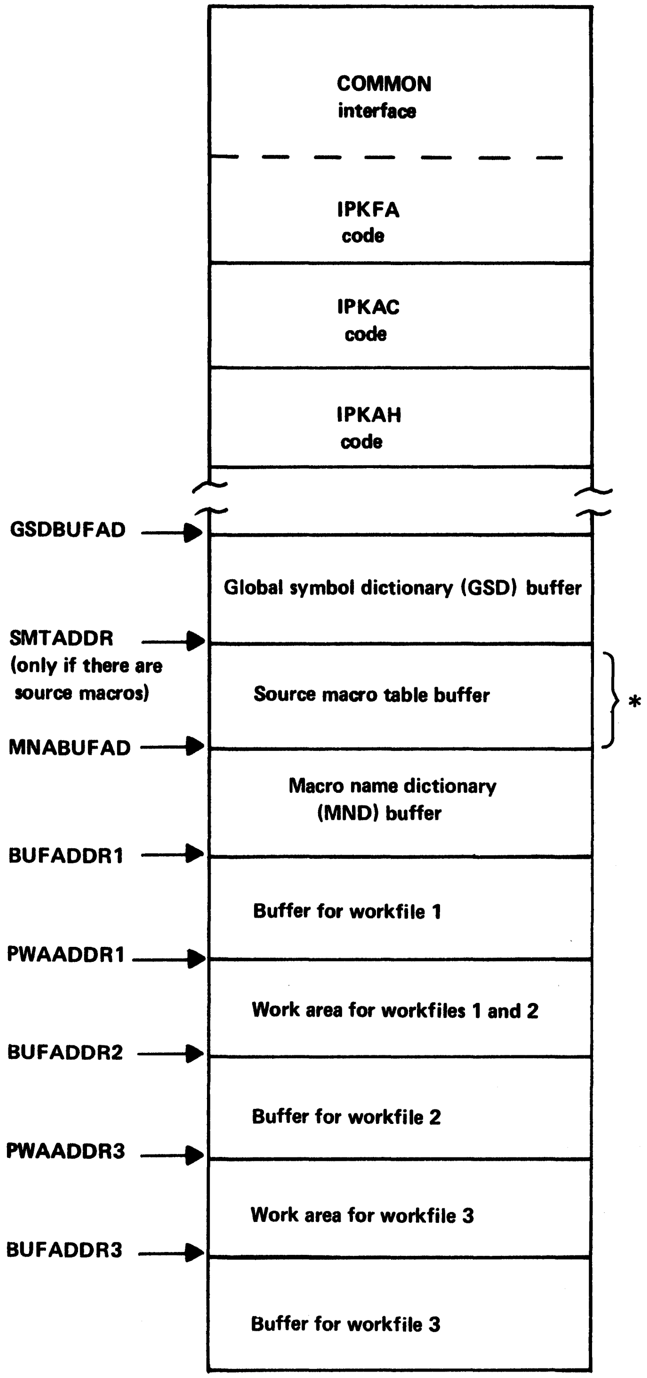


Figure 8. ASSEGA Main Storage Layout.



* This area is overlaid by GSD buffer if there are no source macros.

Figure 9. ASSEFA Main Storage Layout.

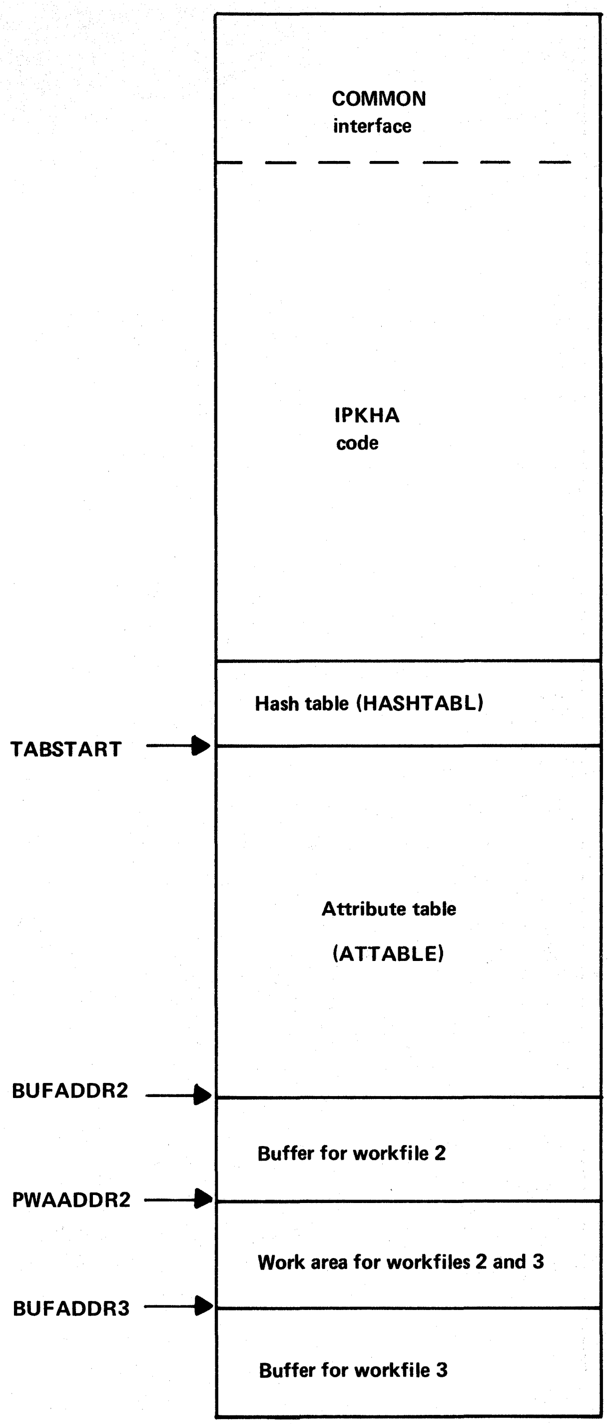


Figure 10. ASSEHA Main Storage Layout.

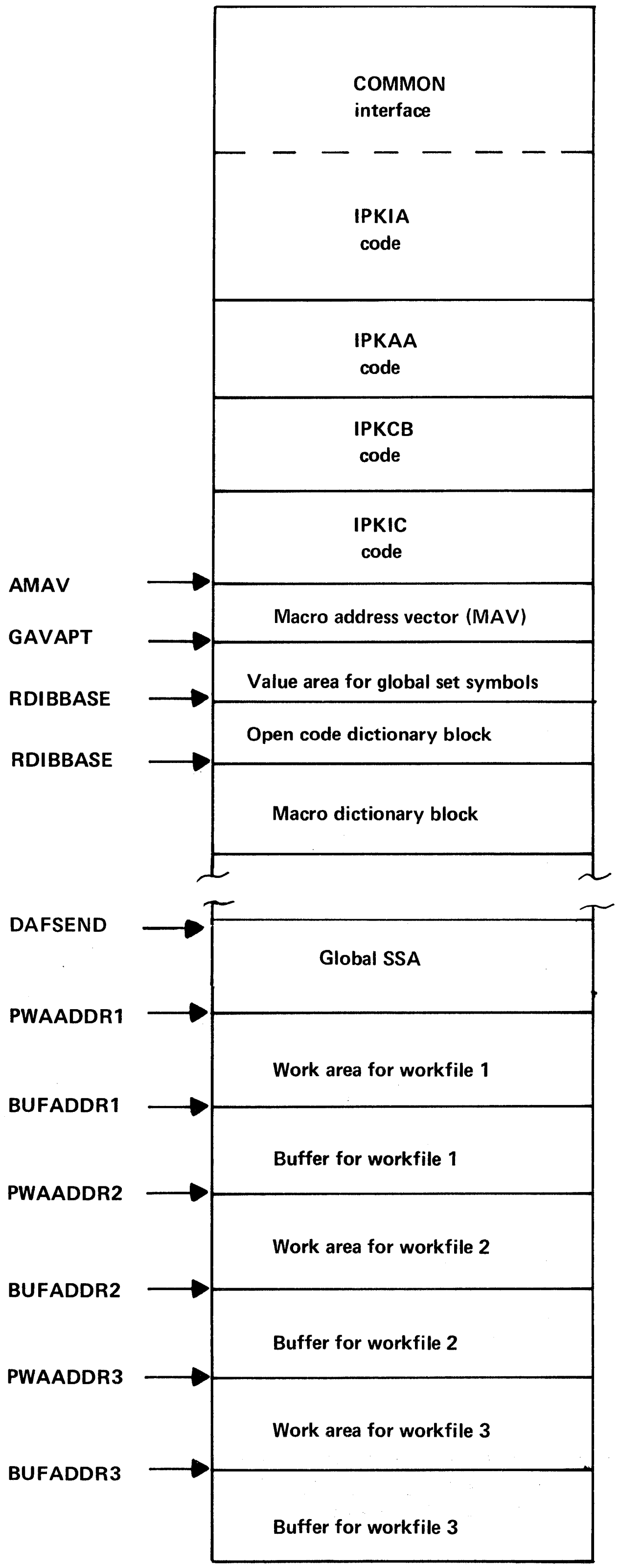
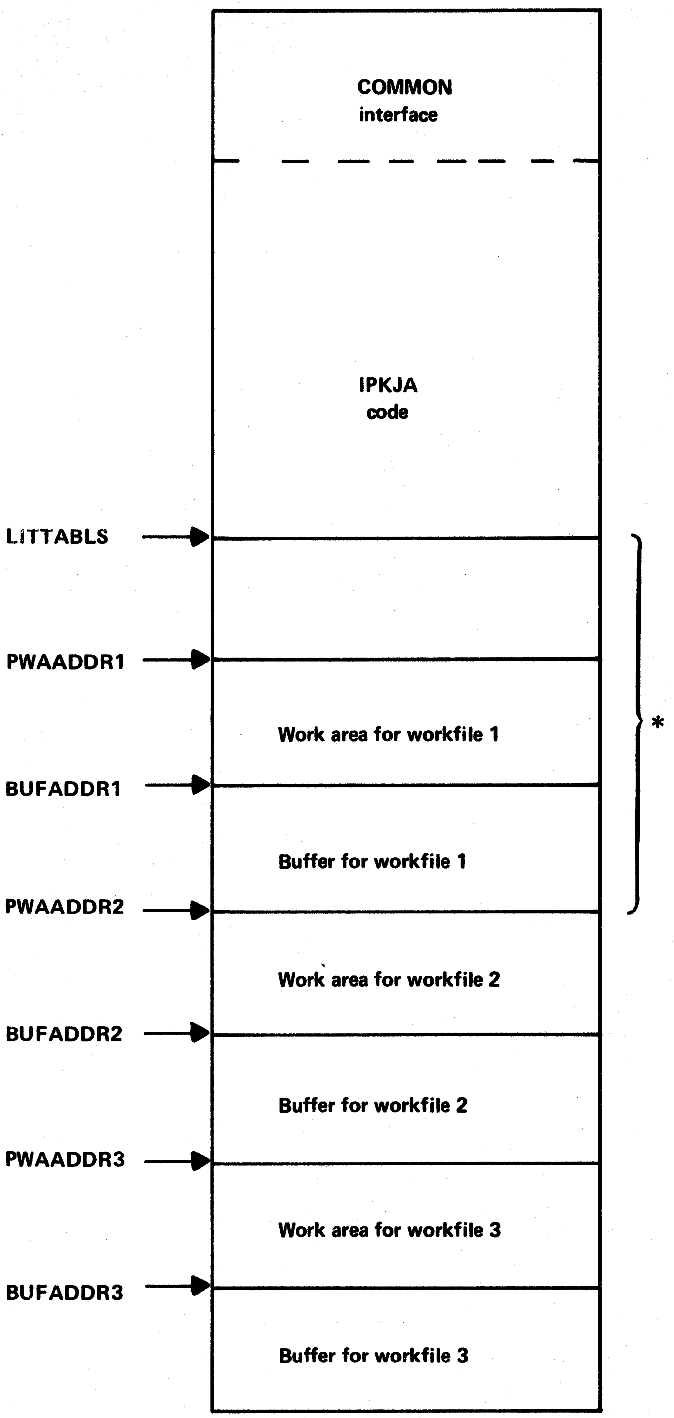


Figure 11. ASSEIA Main Storage Layout.



* If there are PUNCH and/or REPRO records before the first control section, this area is first used by workfile 1, as shown here, and then overlaid by the Literal pool. If there are no PUNCH or REPRO records before the first control section, workfile 1 is not used and this area is only used by the Literal pool.

Figure 12. ASSEJA Main Storage Layout.

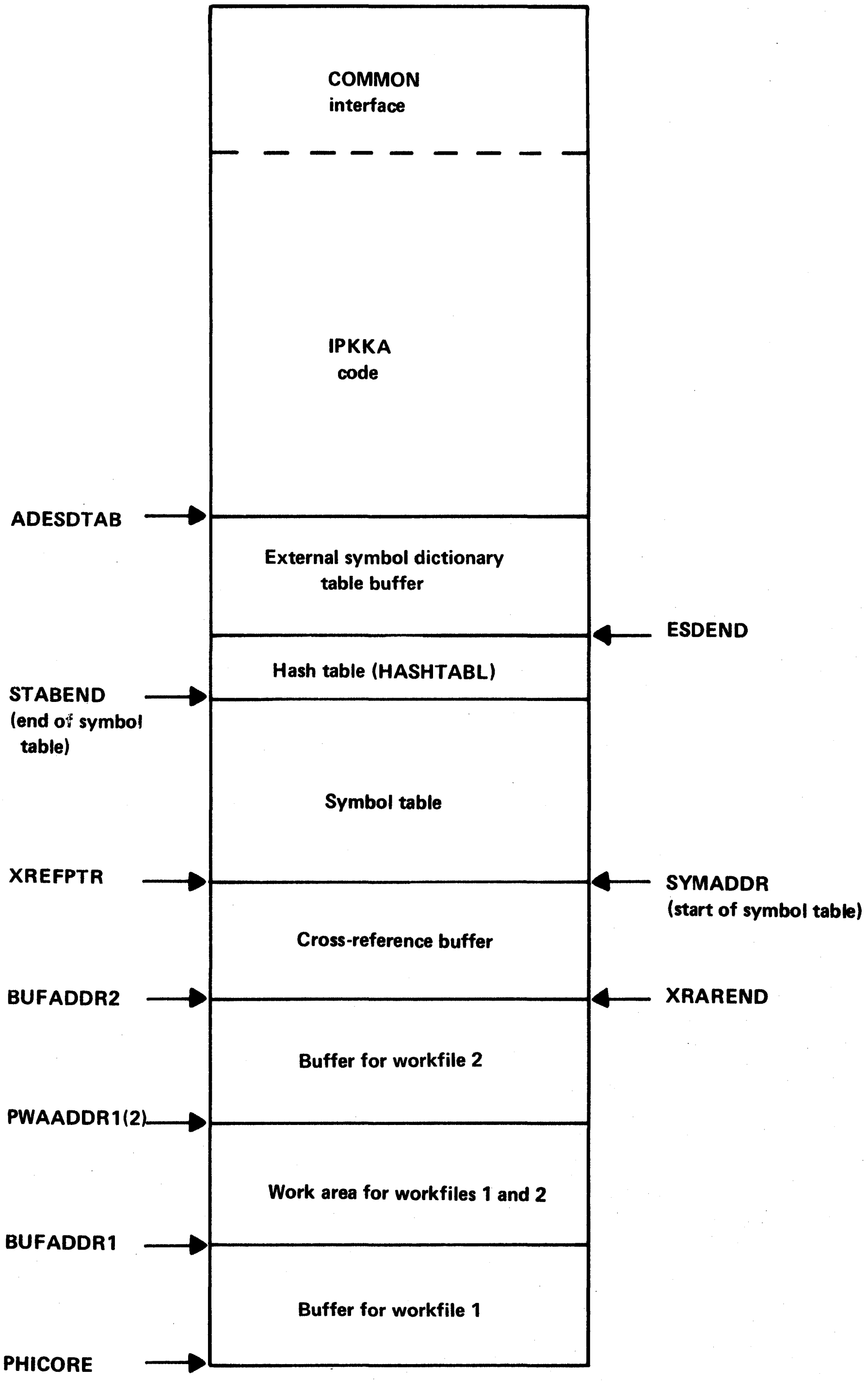


Figure 13. ASSEKA Main Storage Layout.

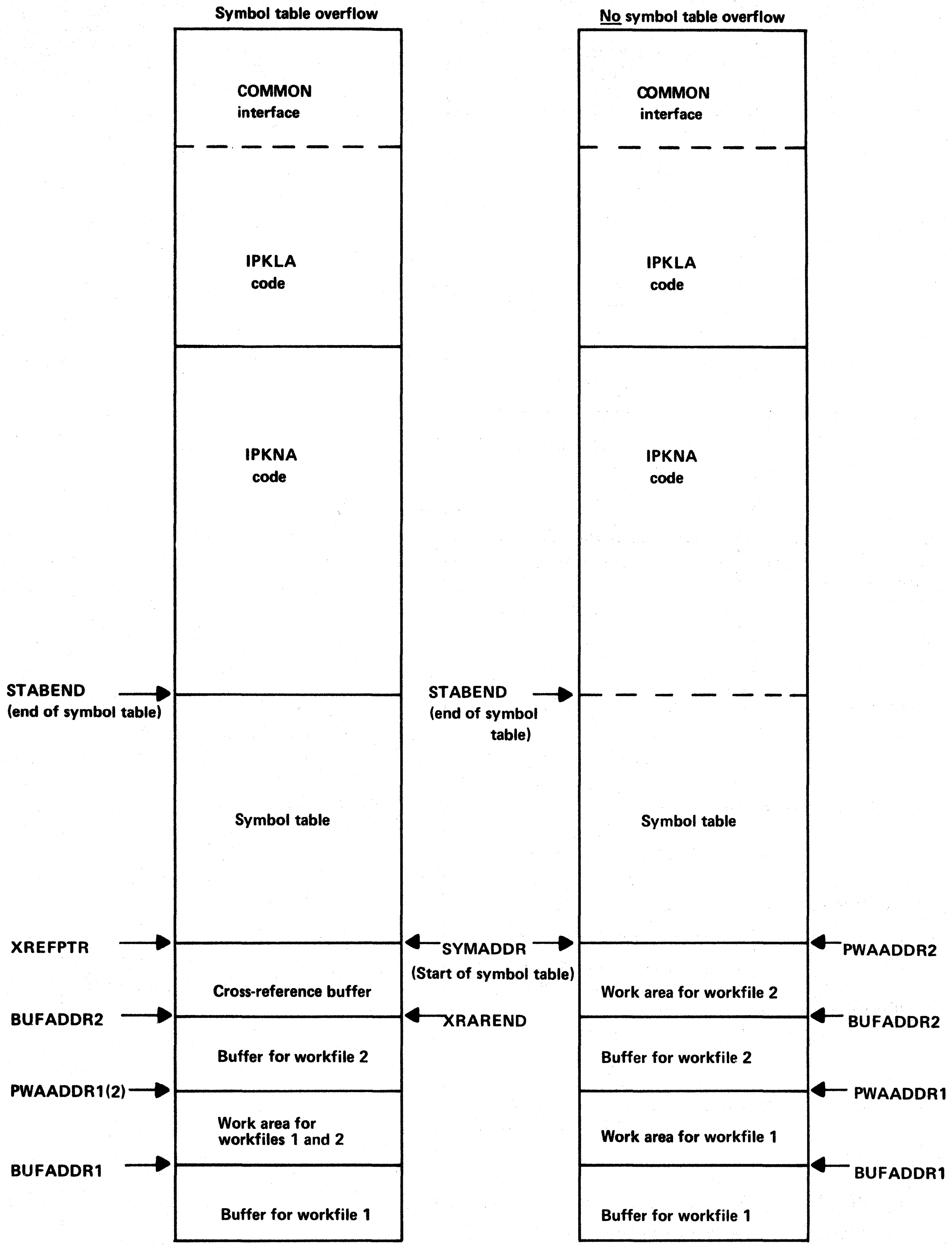
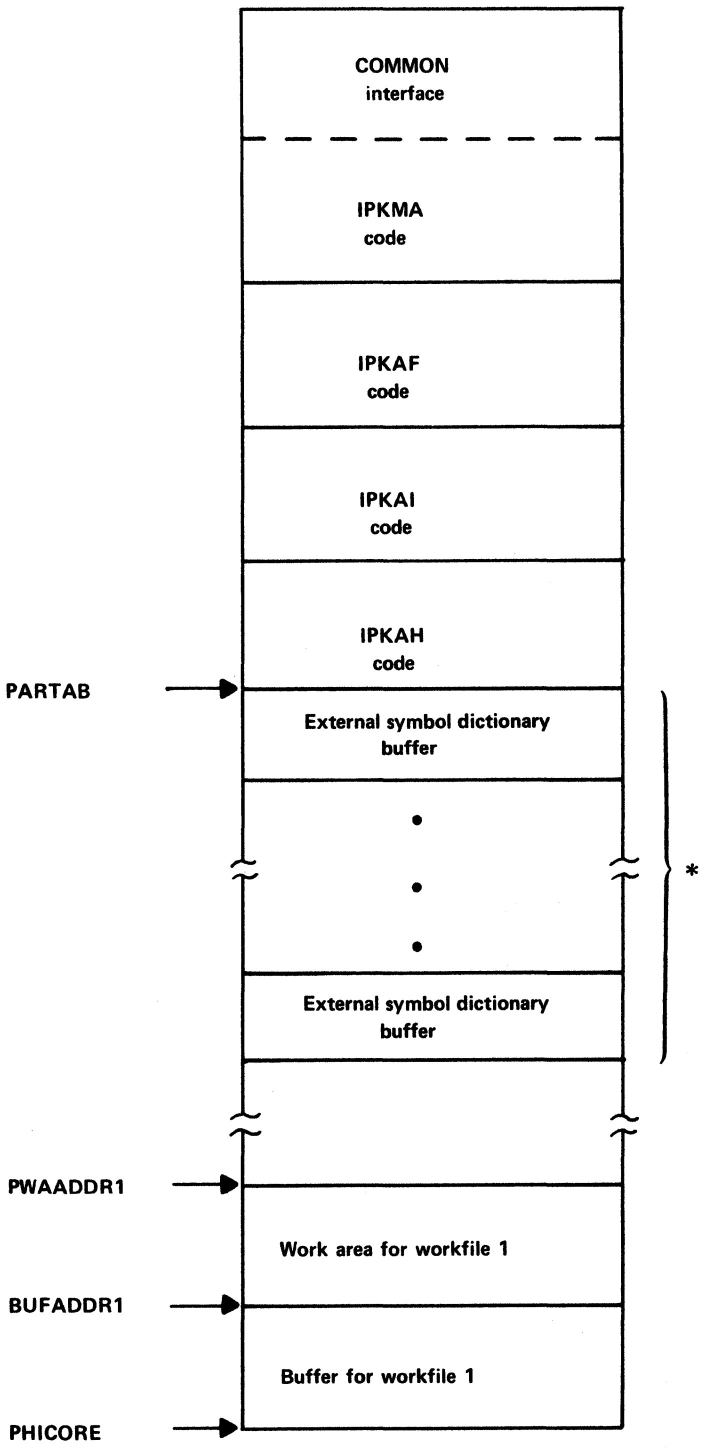


Figure 14. ASSELA Main Storage Layout.



* The maximum number of ESD buffers is seven.

Figure 15. ASSEMA Main Storage Layout.

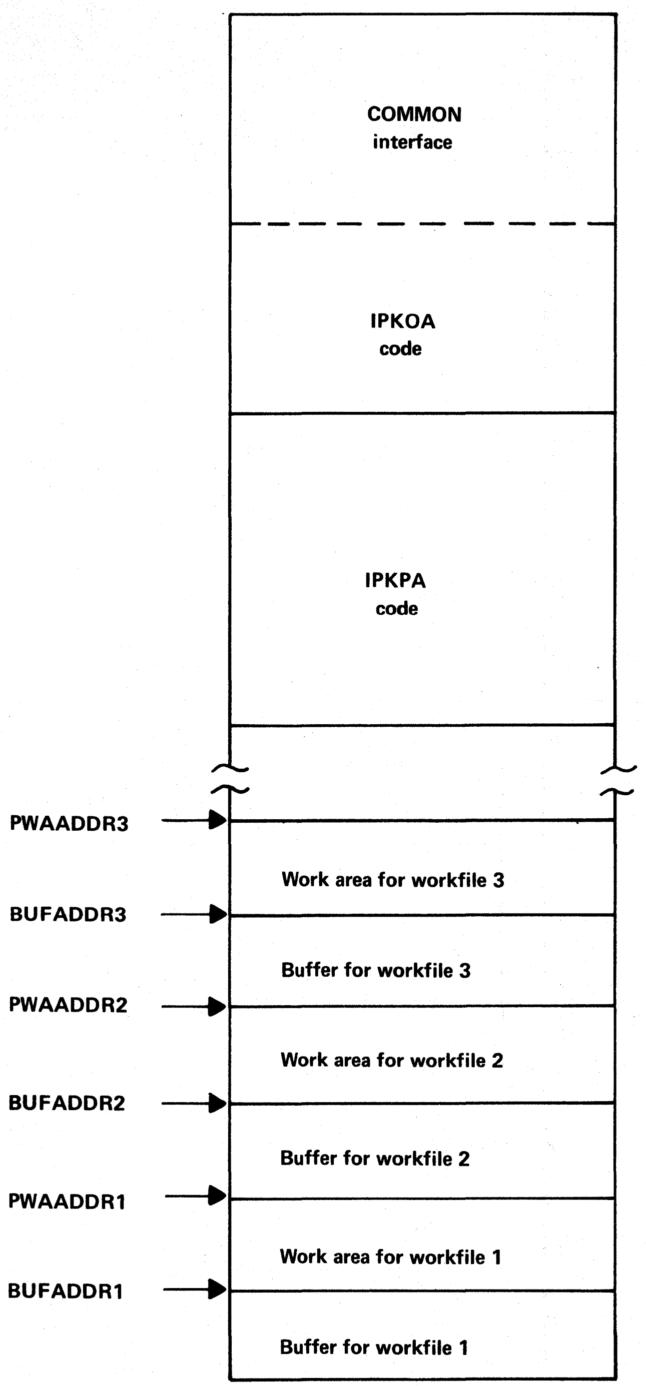
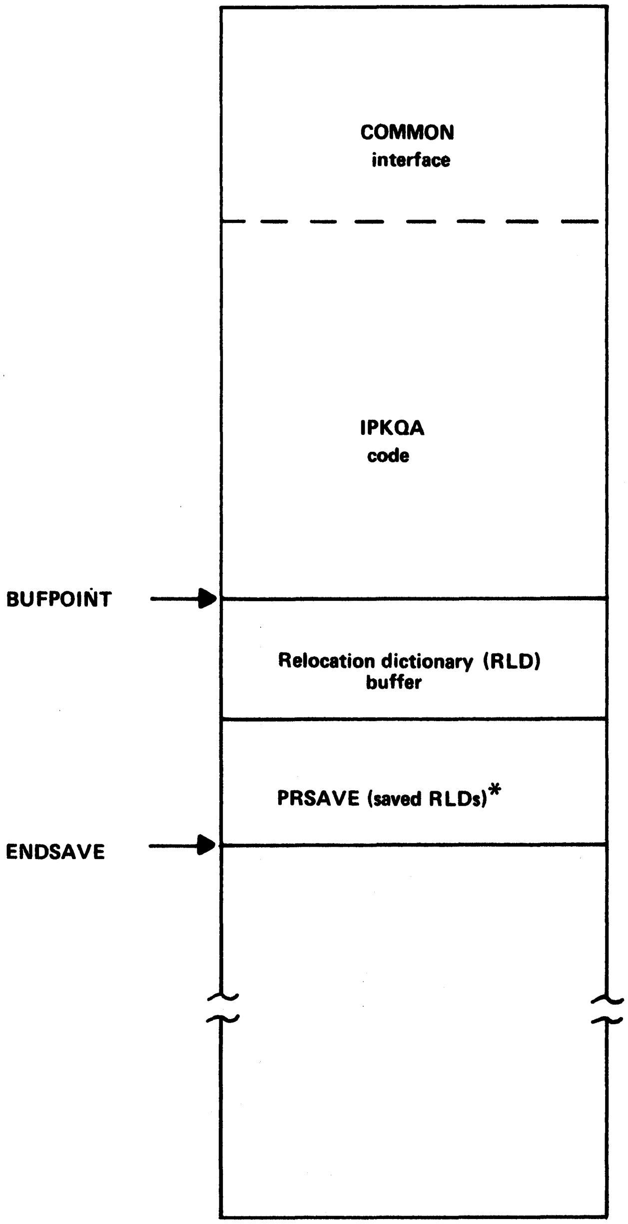
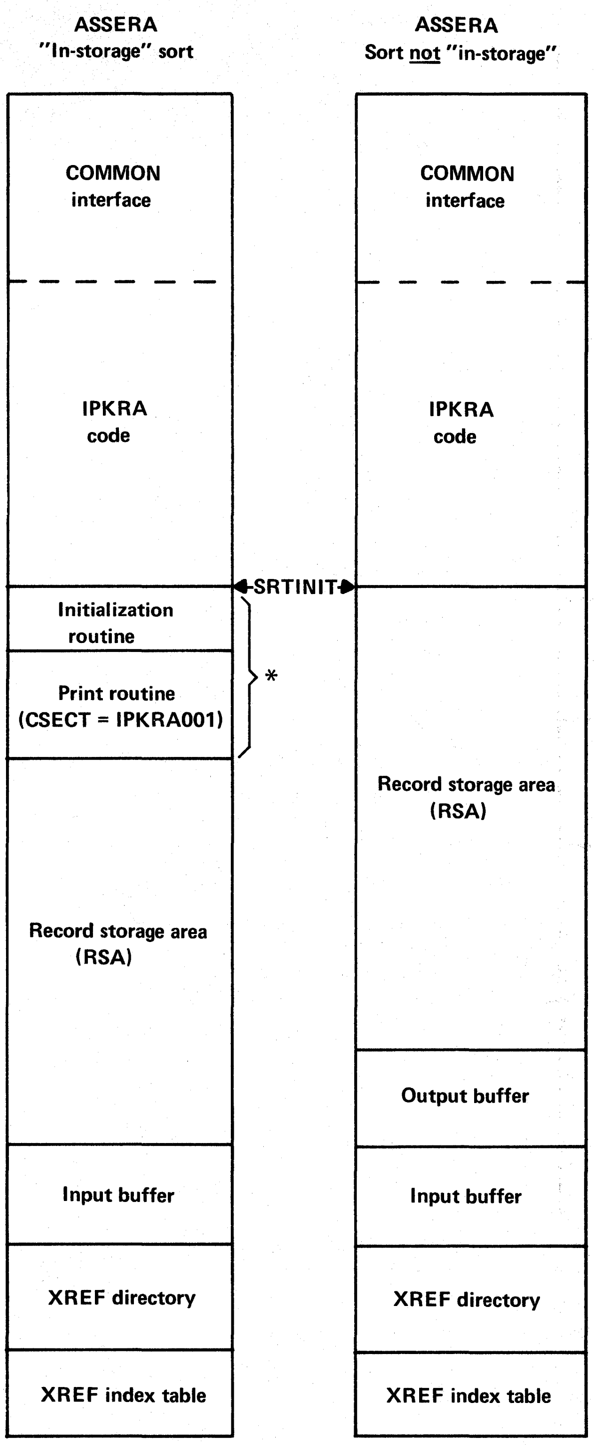


Figure 16. ASSEOA Main Storage Layout.



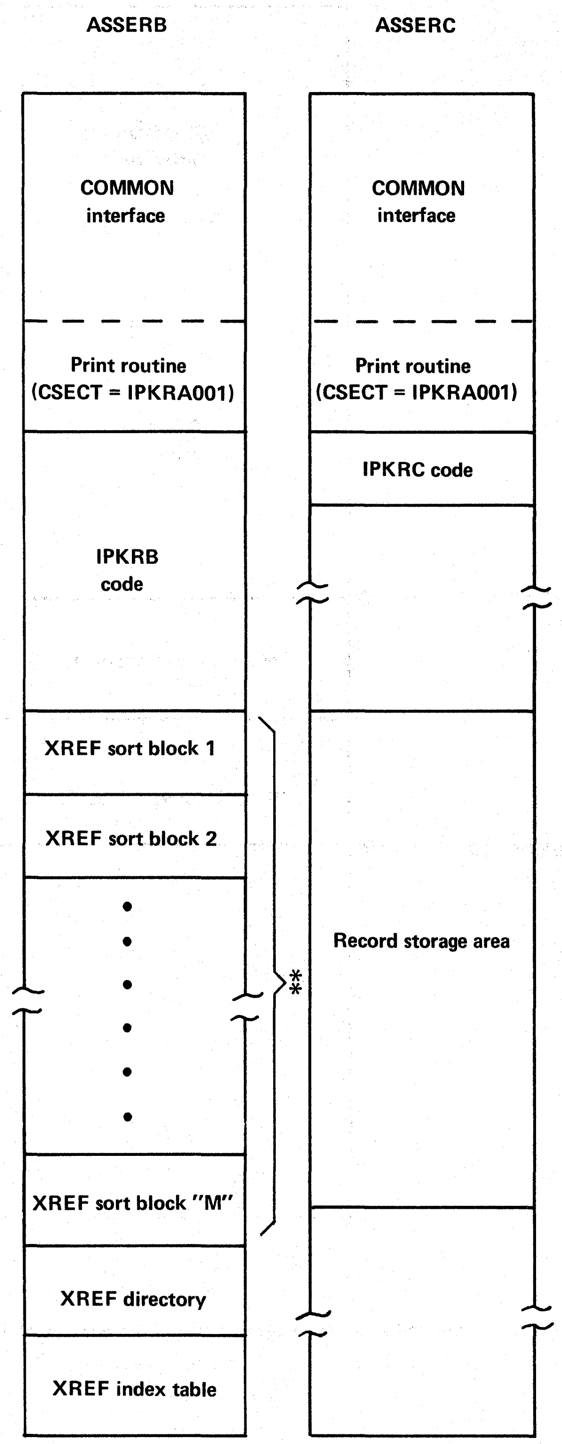
* Maximum number of RLDs saved = maximum number of lines.

Figure 17. ASSEQA Main Storage Layout.



* If the record storage area starts at label SRTINIT, then both the initialization and print routines are overlaid by the RSA and the output printing is handled by ASSERC.

Figure 18. ASSERA Main Storage Layout.



** The number of XREF sort blocks is 2≤M≤19.

Figure 19. ASSERB and ASSERC Main Storage Layout.

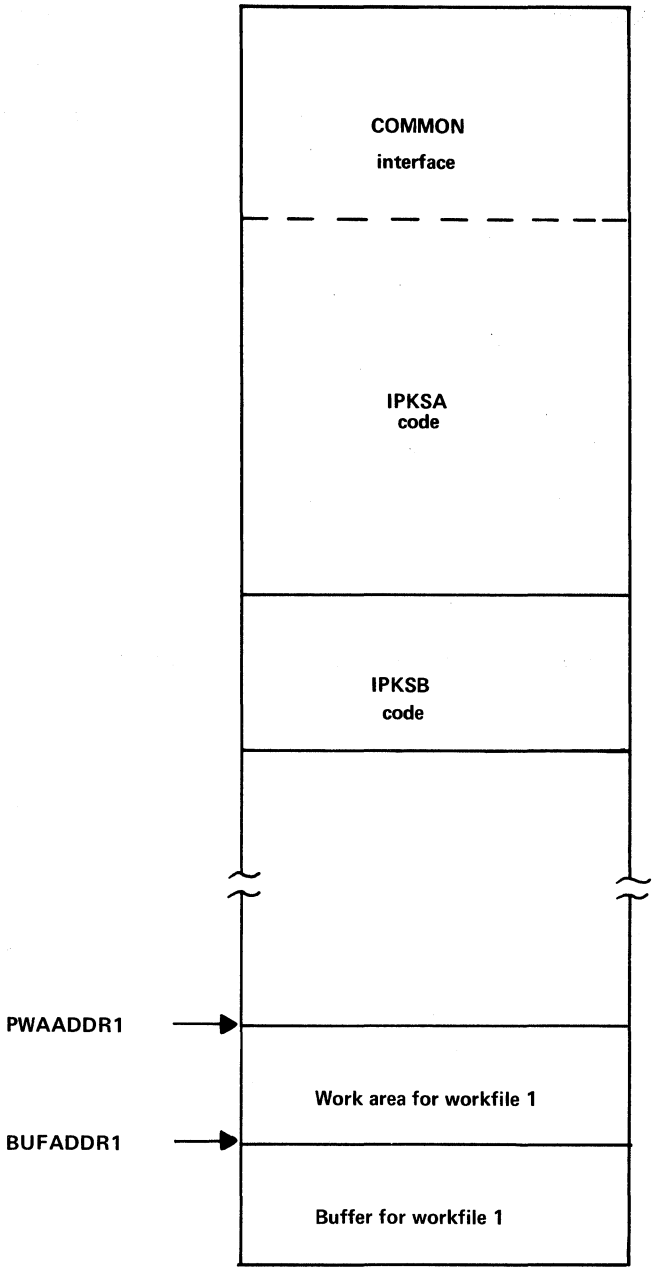


Figure 20. ASSESA Main Storage Layout.

Common Data Area for the Assembler

The interface phase ASSEMBLY contains the common data area COMMON. This data area is included in all other modules in the DSECT "PCOMMON". PCOMMON is divided up into seven parts, each part a COPY book, as follows:

IBRTAB	Branch table (branches to interface routines)
PCOM1	Equates and data areas used by the assembler
PCOM2,3,5,6,7	Data areas used by the phases of the assembler at different times during execution

The different modules COPY those parts of COMMON that they need. PCOM1,2,3,5,6,7 overlay each other by means of ORGs. For example:

PCOM2 starts with `ORG7 EQU *`

then

PCOM3 starts with `ORG ORG7`

See "Data Areas" for a complete description of the DSECT PCOMMON.

Directory

Purpose of the Section

The purpose of this section is to assist you in getting from the information in the manual to the pertinent code in the program listings and/or from the listings to the relevant information in the manual. The directory relates each module, entry point, and control section name in the program to the corresponding microfiche card.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
BEC	DSECT NAME; SOURCE AND OBJECT TEXT OUTPUT		BEC	IPKPA
CAED	COL.MODE AND OVERFLOW ?, ATTRIBUTE PHASE	1.4	IPKHA000	IPKHA
CAEDIT	DSECT NAME; ATTRIBUTE PHASE		CAEDIT	IPKHA
CAEVAL	SAVE REGISTERS, PHKGEN	1.5,1.5.2	IPKIA000	IPKIA
CATALBKE	EDECK OUTPUT	1.2	IPKGA000	IPKGA
CCWCODE	DSECT NAME; CCW OUTPUT, CONSTANT AND CCW CODE BUILD		CCWCODE	ADDRES
CCWCODE	DSECT NAME; CCW OUTPUT, ADDRESS CONSTANT AND CCW CODE BUILD		CCWCODE	IPKNA
CCWR	STORE LENGTH OF CCW, ASSIGNMENT PHASE	2.2.1	IPKKA000	IPKKA
CHECKGS	GLOBAL EDIT	1.3,1.3.1	IPKFA000	IPKFA
CHKNAME	PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1.1	IPKJA000	IPKJA
CNOPR	PUT VALUES IN SYMBOL BUCKETS, ASSIGNMENT PHASE	2.2.1	IPKKA000	IPKKA
CODE	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		CODE	
COMR	GIVE IT A NAME, ASSIGNMENT PHASE	2.2.2	IPKKA000	IPKKA
COMREG	DSECT NAME; ONE TIME INITIALIZER		COMREG	IPKBA
CONXXXX	DSECT NAME; ONE TIME INITIALIZER		CONXXXX	IPKBA
COPERROR	SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
CROSSREF	XREF LISTING ?, ASSIGNMENT PHASE	2.3,2.2	IPKKA000	IPKKA
CROSSREF	XREF LISTING ?, SUBSTITUTION PHASE	2.3	IPKLA000	IPKLA
CSECTR	INDICATE THAT IN CSECT, ASSIGNMENT PHASE	2.2.2	IPKKA000	IPKKA
DCEDIT	DSECT NAME; *** THIS DSECT, ASSIGNMENT PHASE		DCEDIT	IPKKA
DCPROC	DC AND DS CODE BUILD	2.6	IPKOA000	IPKOA
DCR	ZEROS TO POPNUMB IN OUTPUT RCD, PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1.2	IPKJA000	IPKJA
DCR	ASSIGNMENT PHASE	2.2.1	IPKKA000	IPKKA
DIB	DSECT NAME; OC OR MACRO DIB, PHKGEN	1.5.1,1.5	DIB	IPKIA
DICTINFO	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		DICTINFO	
DIRADDR	ADDRESS TO DIRECTORY BUFFER, POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	PCOMMON	IPKRA
DIREND	END OF DIRECTORY, POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	PCOMMON	IPKRA
*DIRENTRY	DSECT NAME; DESCRIBES ONE ENTRY IN THE, POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	DIRENTRY	IPKRA
*DIRENTRY	DSECT NAME; DESCRIBES ONE ENTRY IN THE, POST PROCESSOR; XREF SORTING		DIRENTRY	IPKRB
DRIVER	OP-CODE LOOKUP AND STMT COMPRESS	4	IPKCA001	IPKCA

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.
**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
DRIVER	PHKGEN	1.5	IPKIA000	IPKIA
DRIVER	GET POINTER TO OUTPUT AREA, PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1.2	IPKJA000	IPKJA
DSECTR	WAS THERE A NAME ?, ASSIGNMENT PHASE	2.2.2	IPKKA000	IPKKA
DSROUT	SAVE RETURN REG, ESD INTERLUDE PHASE	2.5	IPKMA000	IPKMA
DUMP	SAVE, SOURCE AND OBJECT TEXT OUTPUT	2.7,2.7.1	IPKPA000	IPKPA
*EDPMI	DSECT NAME; EDITED PROTOTYPE AND M-I, EXTERNAL SYMBOL DICTIONARY		EDPMI	
*EDPMI	DSECT NAME; EDITED PROTOTYPE AND M-I, M-I AND PROTOTYPE EDITOR		EDPMI	IPKCC
*EDPMI	DSECT NAME; EDITED PROTOTYPE AND M-I, VARIABLE SYMBOL DECLARATION PROCESSOR		EDPMI	IPKDB
*EDPMI	DSECT NAME; EDITED PROTOTYPE AND M-I, GLOBAL EDIT		EDPMI	IPKFA
*EDPMI	DSECT NAME; EDITED PROTOTYPE AND M-I, ATTRIBUTE PHASE		EDPMI	IPKHA
*EDPMI	DSECT NAME; EDITED PROTOTYPE AND M-I, PHKGEN		EDPMI	IPKIA
EDTEXT	ENTRY POINT; SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
EINFO	DSECT NAME; ERROR ITEM DESCRIPTOR, EXTERNAL SYMBOL DICTIONARY		EINFO	
EJECTOP	SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
ENDCARD	DSECT NAME; ***, RLD OUTPUT PHASE		ENDCARD	IPKQA
ENDR	PUT VALUES IN SYMBOL BUCKETS, ASSIGNMENT PHASE	2.2.1	IPKKA000	IPKKA
*EPAR	DSECT NAME; ENTRY IN PARAMETER TABLE, PHKGEN	1.5.1	EPAR	IPKIA
EQUR	EQUR (3705) @DL29301, ASSIGNMENT PHASE	2.2.1	PCOMMON	IPKKA
ERRBYTES	DSECT NAME; *** THIS DSECT, ASSIGNMENT PHASE		ERRBYTES	IPKKA
ERRBYTES	DSECT NAME; *** THIS DSECT, SUBSTITUTION PHASE		ERRBYTES	IPKLA
ERRCALL	DSECT NAME; SOURCE AND OBJECT TEXT OUTPUT		ERRCALL	IPKPA
*ERRENT	DSECT NAME; ,ENTRY IN ERROR STACK, EXTERNAL SYMBOL DICTIONARY		ERRENT	
*ERRENT	DSECT NAME; ENTRY IN ERROR STACK, VARIABLE SYMBOL DECLARATION PROCESSOR		ERRENT	IPKDB
ERRLOG	DC AND DS CODE BUILD	2.4.3,2.6	IPKOA000	IPKOA
ERRPUT	ENTRY POINT; DC AND DS CODE BUILD	2.6,2.4.3	IPKOA000	IPKOA
ERSTACKM	DSECT NAME; PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES		ERSTACKM	IPKJA

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.
**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
ESDCOL17	DSECT NAME; ***, ESD INTERLUDE PHASE		ESDCOL17	IPKMA
*ESDENTRY	DSECT NAME; THIS DSECT DESCRIBES AN ENTRY, ASSIGNMENT PHASE	2.2,2.2.2	ESDENTRY	IPKKA
*ESDENTRY	DSECT NAME; ***, ESD INTERLUDE PHASE	2.5	ESDENTRY	IPKMA
ESDLCTR	* CURRENT LOCCNTR OF CONT. SEC, ASSIGNMENT PHASE	2.2.2	ESDENTRY	IPKKA
ESDROUT	ESD INTERLUDE PHASE	2.5	IPKMA000	IPKMA
ESDTAB	DSECT NAME; ***, ESD INTERLUDE PHASE		ESDTAB	IPKMA
*EVALSTCK	DSECT NAME; ENTRY IN THE EVAL ROUT STACK, ASSIGNMENT PHASE		EVALSTCK	IPKKA
*EVALSTCK	DSECT NAME; ENTRY IN THE EVAL ROUT STACK, SUBSTITUTION PHASE		EVALSTCK	IPKLA
EVALUATE	SAVE RETURN REGISTER, ASSIGNMENT PHASE	2.3	IPKKA000	IPKKA
EVALUATE	SUBSTITUTION PHASE	2.3	IPKLA000	IPKLA
FINDGS	GLOBAL EDIT	1.3.1,1.3	IPKFA000	IPKFA
FIXUP	INSERT END OF OPERAND FLAG, PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1.2	IPKJA000	IPKJA
*GARD	DSECT NAME; GLOBAL ARRAY RECORD DSECT, VARIABLE SYMBOL DECLARATION PROCESSOR		GARD	IPKDB
*GARD	DSECT NAME; GLOBAL ARRAY RECORD DSECT, GLOBAL EDIT		GARD	IPKFA
*GARENT	DSECT NAME; GLOBAL ARRAY ENTRY DSECT, VARIABLE SYMBOL DECLARATION PROCESSOR		GARENT	IPKDB
*GARENT	DSECT NAME; GLOBAL ARRAY ENTRY DSECT, GLOBAL EDIT	1.3.1	GARENT	IPKFA
GEERR	GLOBAL EDIT	1.3.1	IPKFA000	IPKFA
GEFIN	GLOBAL EDIT	1.3	IPKFA000	IPKFA
GEINIT	GLOBAL EDIT	1.3	IPKFA000	IPKFA
GEOC	GLOBAL EDIT	1.3.1	IPKFA000	IPKFA
GETSRC	GET A SOURCE RECORD, SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
GSDBUFAD	ADDRESS OF GSD BUFFER, GLOBAL EDIT	1.3,1.3.1	IPKFA000	IPKFA
GSDENT	GLOBAL EDIT	1.3.1,1.3	IPKFA000	IPKFA
*GSDENTRY	DSECT NAME; GSD ENTRY DSECT, GLOBAL EDIT	1.3.1,1.3	GSDENTRY	IPKFA
GSDNPDST	DSECT NAME; DSECT FOR GSD N/P TABLE, GLOBAL EDIT		GSDNPDST	IPKFA
GVENT	GLOBAL EDIT	1.3,1.3.1	IPKFA000	IPKFA
ICEND	ALIAS FOR IPKIC999. PHKGEN	1.5	IPKIA000	IPKIA
IELEM	DSECT NAME; INPUT ELEMENT FORMAT, PHKGEN		IELEM	IPKIA
IJJCPDV2	CSECT NAME; SYSIPT LOGIC MODULE		IJJCPDV2	IPKAG

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.
**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
IJJCPD0	CSECT NAME; SYSPCH/SYSLNK/SYSLST LOGIC MODULE		IJJCPD0	IPKAH
IJJCPD1N	CSECT NAME; EXTERNAL SYMBOL DICTIONARY		IJJCPD1N	
IJJCPD2	CSECT NAME; 3-3, SYSIPT LOGIC MODULE		IJJCPD2	IPKAG
*IJJCPTAB	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		IJJCPTAB	
*IJJCPTAB	DSECT NAME; SYSIPT LOGIC MODULE		IJJCPTAB	IPKAG
*IJJCPTAB	DSECT NAME; SYSPCH/SYSLNK/SYSLST LOGIC MODULE		IJJCPTAB	IPKAH
IJSYS01	BASIC INTERFACE ROUTINES AND PCOMMON	1.1	IPKAA000	IPKAA
IJSYS02	BASIC INTERFACE ROUTINES AND PCOMMON	1.1	IPKAA000	IPKAA
IJSYS03	BASIC INTERFACE ROUTINES AND PCOMMON	1.1	IPKAA000	IPKAA
IJ2M0074	CSECT NAME; 3-3, EXTERNAL SYMBOL DICTIONARY		IJ2M0074	
IJ2T0074	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		IJ2T0074	
IMIEDIT	EDITED M-I, PHKGEN	1.5.1,1.5	IPKIA000	IPKIA
INBUFSZ	ONE TIME INITIALIZER	3	IPKBA000	IPKBA
*INDENTRY	DSECT NAME; ENTRY IN INDEX TABLE, POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	INDENTRY	IPKRA
*INDENTRY	DSECT NAME; DESCRIBES ENTRY IN INDEX TABLE, POST PROCESSOR; XREF SORTING		INDENTRY	IPKRB
INFILE	ONE TIME INITIALIZER	3	IPKBA000	IPKBA
INIT	PHKGEN	1.5	IPKIA000	IPKIA
INITCDE	ONE TIME INITIALIZER	3	IPKBA000	IPKBA
INOPEN	ONE TIME INITIALIZER	3	IPKBA000	IPKBA
INOPT	ONE TIME INITIALIZER	3	IPKBA000	IPKBA
INPARSIZ	ONE TIME INITIALIZER	3	IPKBA000	IPKBA
INREC	ONE TIME INITIALIZER	3	IPKBA000	IPKBA
INSERT1	SAVE RETURN REG, ATTRIBUTE PHASE	1.4	IPKHA000	IPKHA
INSERT2	ATTRIBUTE PHASE	1.4	IPKHA000	IPKHA
IPKAA000	CSECT NAME; BASIC INTERFACE ROUTINES AND PCOMMON		IPKAA000	IPKAA
IPKAA001	CSECT NAME; INTERFACE ROUTINE FOR MACRO PROCESSING		IPKAA001	IPKAA
IPKAA002	CSECT NAME; BASIC INTERFACE ROUTINES AND SDMODW		IPKAA002	IPKAA
IPKAA003	CSECT NAME; INTERFACE ROUTINE FOR MACRO PROCESSING (FBA VERSION)		IPKAA003	IPKAA
IPKAB000	CSECT NAME; SYSIPT AND SYSSLB ROUTINES		IPKAB000	IPKAB
IPKAC000	CSECT NAME; PUNCH ROUTINE FOR EDECK		IPKAC000	IPKAC
IPKAD000	CSECT NAME; DTFSL		IPKAD000	IPKAD

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.
**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
IPKAD100	CSECT NAME; DTFSL		IPKAD100	IPKAD
IPKAE000	CSECT NAME; SYSSLB ROUTINES FOR GLOBAL EDIT		IPKAE000	IPKAE
IPKAF000	CSECT NAME; SYSPCH/SYSLNK OUTPUT		IPKAF000	IPKAF
IPKAG000	CSECT NAME; SYSIPT LOGIC MODULE		IPKAG000	IPKAG
IPKAH000	CSECT NAME; SYSPCH/SYSLNK/SYSLST LOGIC MODULE		IPKAH000	IPKAH
IPKAI000	CSECT NAME; PRINT ROUTINE		IPKAI000	IPKAI
IPKAJ000	CSECT NAME; EMBEDDED IDENTIFIER		IPKAJ000	IPKAJ
IPKBA000	CSECT NAME; ONE TIME INITIALIZER		IPKBA000	IPKBA
IPKCA001	CSECT NAME; OP-CODE LOOKUP AND STMT COMPRESS		IPKCA001	IPKCA
IPKCB000	CSECT NAME; EXTERNAL SYMBOL DICTIONARY		IPKCB000	
IPKCC000	CSECT NAME; M-I AND PROTOTYPE EDITOR		IPKCC000	IPKCC
IPKCD001	CSECT NAME; OVERLAY FOR ICTL,ISEQ,TITLE,COPY,BKEND,EOF		IPKCD001	IPKCD
IPKDA000	CSECT NAME; EXTERNAL SYMBOL DICTIONARY		IPKDA000	
IPKDB000	CSECT NAME; VARIABLE SYMBOL DECLARATION PROCESSOR		IPKDB000	IPKDB
IPKEA000	CSECT NAME; SEQ SYM REFERENCE PROCESSOR		IPKEA000	IPKEA
IPKFA000	CSECT NAME; GLOBAL EDIT		IPKFA000	IPKFA
IPKGA000	CSECT NAME; EDECK OUTPUT		IPKGA000	IPKGA
IPKHA000	CSECT NAME; ATTRIBUTE PHASE		IPKHA000	IPKHA
IPKIA000	CSECT NAME; PHKGEN		IPKIA000	IPKIA
IPKIC000	CSECT NAME; LOOKUP AND CHECK OF GENERATED OPCODES		IPKIC000	IPKIC
IPKJA000	CSECT NAME; PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES		IPKJA000	IPKJA
IPKKA000	CSECT NAME; ASSIGNMENT PHASE		IPKKA000	IPKKA
IPKKA001	CSECT NAME; ASSIGNMENT PHASE		IPKKA001	IPKKA
IPKLA000	CSECT NAME; SUBSTITUTION PHASE		IPKLA000	IPKLA
IPKMA000	CSECT NAME; ESD INTERLUDE PHASE		IPKMA000	IPKMA
IPKNA000	CSECT NAME; USING,DROP,MACHIN OP AND S-CONST CODE BUILD		IPKNA000	IPKNA
IPKNB000	CSECT NAME; ADDRESS CONSTANT AND CCW CODE BUILD		IPKNB000	IPKNA
IPKOA000	CSECT NAME; DC AND DS CODE BUILD		IPKOA000	IPKOA
IPKPA000	CSECT NAME; SOURCE AND OBJECT TEXT OUTPUT		IPKPA000	IPKPA
IPKQA000	CSECT NAME; RLD OUTPUT PHASE		IPKQA000	IPKQA

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.

**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
IPKRA000	CSECT NAME; POST PROCESSOR; XREF SORTING AND PRINTING		IPKRA000	IPKRA
IPKRA001	CSECT NAME; POST PROCESSOR; XREF SORTING AND PRINTING		IPKRA001	IPKRA
IPKRB000	CSECT NAME; POST PROCESSOR; XREF SORTING		IPKRB000	IPKRB
IPKRC000	CSECT NAME; POST PROCESSOR; XREF PRINTING		IPKRC000	IPKRC
IPKSA000	CSECT NAME; EXTERNAL SYMBOL DICTIONARY		IPKSA000	
IPKSB000	CSECT NAME; EXTERNAL SYMBOL DICTIONARY		IPKSB000	
IPKTA000	CSECT NAME; EXTERNAL SYMBOL DICTIONARY		IPKTA000	
*KEYTAB	DSECT NAME; KEYWORD TABLE DSECT, VARIABLE SYMBOL DECLARATION PROCESSOR	1.1.2	KEYTAB	IPKDB
*KEYTAB	DSECT NAME; KEYWORD TABLE DSECT, GLOBAL EDIT		KEYTAB	IPKFA
*KEYTAB	DSECT NAME; KEYWORD TABLE DSECT, PHKGEN		KEYTAB	IPKIA
KNAPT	-> ENTRY IN KNA, PHKGEN	1.5.1	IPKIA000	IPKIA
LBUF	DSECT NAME; DESCRIBES EDECK CARD IMAGE, GLOBAL EDIT		LBUF	IPKFA
LITCOPE	ENTRY POINT; SOURCE AND OBJECT TEXT OUTPUT	2.7.1	IPKPA000	IPKPA
LITDRV	PICK UP ACTUAL LIT BLK ADDR, PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1	IPKJA000	IPKJA
LITERAL	PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1.2,2.1	IPKJA000	IPKJA
LITMN	LOAD OFFSET TO 8-CHAIN, PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1.2,2.1	IPKJA000	IPKJA
LITSRCE	PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1,2.1.2	IPKJA000	IPKJA
*LITTAB	DSECT NAME; THIS DESCRIBES AN ENTRY IN, POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	LITTAB	IPKRA
*LITTBL	DSECT NAME; ENTRY IN LITERAL TABLE, PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1.2,2.1	LITTBL	IPKJA
LOCCNTHI	HIGHEST LOCCNTR OF THIS SEC., ASSIGNMENT PHASE	2.2.2	PCOMMON	IPKKA
LOCCNTR	LOCATION COUNTER, ASSIGNMENT PHASE	2.2.2	PCOMMON	IPKKA
LTORG	LTORG, PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1	PCOMMON	IPKJA
LTORGR	ENTRY POINT; PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1	IPKJA000	IPKJA
LTORGR	INSERT LENGTH ATTRIBUTE, ASSIGNMENT PHASE	2.2.1	IPKKA000	IPKKA
LUB	DSECT NAME; LOGICAL UNIT BLKS FOR PARTITION, DTFSL		LUB	IPKAD
*MACHEAD	DSECT NAME; MACRO HEADER RECORD OUT, SEQ SYM REFERENCE PROCESSOR		MACHEAD	IPKEA
*MACHEAD	DSECT NAME; MACRO HEADER DSECT, GLOBAL EDIT		MACHEAD	IPKFA

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.
**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
*MACHEAD	DSECT NAME; MACRO HEADER DSECT, EDECK OUTPUT		MACHEAD	IPKGA
*MACHEAD	DSECT NAME; MACRO HEADER DSECT, PHKGEN		MACHEAD	IPKIA
MACHINOP	PUT RIGHT VALUES, ASSIGNMENT PHASE	2.2.1	IPKKA000	IPKKA
MACHOPS	ENTRY POINT; DC AND DS CODE BUILD	2.7.1	IPKOA000	IPKOA
MACINS	TEST IF COLLECTION MODE, ATTRIBUTE PHASE	1.4	IPKHA000	IPKHA
MAINK	PHKGEN	1.5.1	IPKIA000	IPKIA
MAIN10	GET OPERAND RECORD, PHKGEN	1.5.1	IPKIA000	IPKIA
MAIN30	READ KT RECORD, PHKGEN	1.5.1	IPKIA000	IPKIA
MAVENTRY	GLOBAL EDIT	1.3	IPKFA000	IPKFA
MESSAGE	DSECT NAME; MESSAGE LAYOUT IN POOL, EXTERNAL SYMBOL DICTIONARY		MESSAGE	
MIB	VARIABLE SYMBOL DECLARATION PROCESSOR	1.1.2	PCOMMON	IPKDB
MIROUT	SAVE MNEMONIC OP-CODE, M-I AND PROTOTYPE EDITOR	1.1,4	IPKCC000	IPKCC
MLIBSRCH	GLOBAL EDIT	1.3	IPKFA000	IPKFA
MNABUFAD	ADDRESS OF MND BUFFER, GLOBAL EDIT	1.3	IPKFA000	IPKFA
*MNAENT	DSECT NAME; MACRO NAME ARRAY ENTRY, OP-CODE LOOKUP AND STMT COMPRESS	4	MNAENT	IPKCA
*MNAENT	DSECT NAME; MACRO NAME ARRAY OR MACRO NAME, M-I AND PROTOTYPE EDITOR		MNAENT	IPKCC
*MNAENT	DSECT NAME; MACRO NAME ARRAY, OVERLAY FOR ICTL,ISEQ,TITLE,COPY,BKEND,EOF		MNAENT	IPKCD
*MNAENT	DSECT NAME; MACRO NAME ARRAY OR MACRO NAME, GLOBAL EDIT	1.3	MNAENT	IPKFA
MNDENT	GLOBAL EDIT	1.3	IPKFA000	IPKFA
MNDNPDST	DSECT NAME; DSECT FOR MND N/P TABLE, GLOBAL EDIT		MNDNPDST	IPKFA
MNDSRCH	GLOBAL EDIT	1.3	IPKFA000	IPKFA
MNOTEOP	SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
MOVEPUT	FROM FIELD, SEQ SYM REFERENCE PROCESSOR	1.1.3	IPKEA000	IPKEA
MPUNCH	SAVE RETURN ADDRESS, EDECK OUTPUT	1.2	IPKGA000	IPKGA
MRGDIR	POST PROCESSOR; XREF SORTING	2.8.2	IPKRB000	IPKRB
MRGMAIN	POST PROCESSOR; XREF SORTING	2.8.2	IPKRB000	IPKRB
MRGPRT	ENTRY POINT, POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	IPKRA001	IPKRA
NADDRSPL	USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4.1	IPKNA000	IPKNA
NAMSCAN	CLEAR WORK REGISTER, ATTRIBUTE PHASE	1.4	IPKHA000	IPKHA
NDROP	USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4,2.4.2	IPKNA000	IPKNA

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.
**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
NERRSTCK	ERROR NUMBER STACK, USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4.1	IPKNA000	IPKNA
NMACHEND	TOO MANY OPERANDS, USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4.1	IPKNA000	IPKNA
NMACHOP	INITIATE GENERATED CODE, USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4,2.4.1	IPKNA000	IPKNA
NPARTAB	HERE, USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4.1	IPKNA000	IPKNA
NSADDR	INDICATE S-TYPE CONSTANT, USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4,2.4.3	IPKNA000	IPKNA
*NTABFMT	DSECT NAME; USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4.1	NTABFMT	IPKNA
NTESTST1	ENTRY POINT; USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4.1	IPKNA000	IPKNA
NTESTST2	ENTRY POINT; USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4.1	IPKNA000	IPKNA
NUSING	SET LOOP COUNTER, USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4,2.4.2	IPKNA000	IPKNA
NUSTAB	USING TABLE, USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4.3	IPKNA000	IPKNA
*OCSTMH	DSECT NAME; OC START AND MACRO HEADER REC, VARIABLE SYMBOL DECLARATION PROCESSOR	1.1.2	OCSTMH	IPKDB
*OCSTMH	DSECT NAME; OC START AND MACRO HDR REC IN, SEQ SYM REFERENCE PROCESSOR	1.1.3	OCSTMH	IPKEA
OPENTRY	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		OPENTRY	
OPERAND	PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1.1	IPKJA000	IPKJA
OPERANDS	ZERO WORKREGISTER, SUBSTITUTION PHASE	2.3	IPKLA000	IPKLA
OPSTACKM	DSECT NAME; PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES		OPSTACKM	IPKJA
ORGPROC	INSERT CUR VALUES IN STMT, ASSIGNMENT PHASE	2.2.1	IPKKA000	IPKKA
PALIGN	ALIAS FOR BIT1. ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
PARM	DSECT NAME; DESCRIBES PREAD/PWRITE PARM, BASIC INTERFACE ROUTINES AND PCOMMON		PARM	IPKAA
PARPTV	DSECT NAME; PARAMETER POINTER VECTOR DSECT, PHKGEN	1.5.1	PARPTV	IPKIA
PBUFLEN1	BUFFER LENGTH, ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
PBUFLEN2	BUFFER LENGTH, ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
PBUFLEN3	BUFFER LENGTH, ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
PB1FISIZ	WF1 BUFSIZE IN F AND I, ONE TIME INITIALIZER	3	PCOMMON	IPKBA
PB12SIZ	WF1,WF2 BUFSIZ, ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
*PCOMMON	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		PCOMMON	
*PCOMMON	DSECT NAME; SYSIPT AND SYSSLB ROUTINES		PCOMMON	IPKAB

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.

**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
*PCOMMON	DSECT NAME; PUNCH ROUTINE FOR EDECK		PCOMMON	IPKAC
*PCOMMON	DSECT NAME; SYSSLB ROUTINES FOR GLOBAL EDIT		PCOMMON	IPKAE
*PCOMMON	DSECT NAME; SYSPCH/SYSLNK OUTPUT		PCOMMON	IPKAF
*PCOMMON	DSECT NAME; PRINT ROUTINE		PCOMMON	IPKAI
*PCOMMON	DSECT NAME; ONE TIME INITIALIZER		PCOMMON	IPKBA
*PCOMMON	DSECT NAME; OP-CODE LOOKUP AND STMT COMPRESS		PCOMMON	IPKCA
*PCOMMON	DSECT NAME; M-I AND PROTOTYPE EDITOR		PCOMMON	IPKCC
*PCOMMON	DSECT NAME; OVERLAY FOR ICTL,ISEQ,TITLE,COPY,BKEND,EOF		PCOMMON	IPKCD
*PCOMMON	DSECT NAME; VARIABLE SYMBOL DECLARATION PROCESSOR		PCOMMON	IPKDB
*PCOMMON	DSECT NAME; SEQ SYM REFERENCE PROCESSOR		PCOMMON	IPKEA
*PCOMMON	DSECT NAME; GLOBAL EDIT		PCOMMON	IPKFA
*PCOMMON	DSECT NAME; EDECK OUTPUT		PCOMMON	IPKGA
*PCOMMON	DSECT NAME; ATTRIBUTE PHASE		PCOMMON	IPKHA
*PCOMMON	DSECT NAME; PHKGEN		PCOMMON	IPKIA
*PCOMMON	DSECT NAME; LOOKUP AND CHECK OF GENERATED OPCODES		PCOMMON	IPKIC
*PCOMMON	DSECT NAME; PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES		PCOMMON	IPKJA
*PCOMMON	DSECT NAME; ASSIGNMENT PHASE		PCOMMON	IPKKA
*PCOMMON	DSECT NAME; SUBSTITUTION PHASE		PCOMMON	IPKLA
*PCOMMON	DSECT NAME; ESD INTERLUDE PHASE		PCOMMON	IPKMA
*PCOMMON	DSECT NAME; USING,DROP,MACHIN OP AND S-CONST CODE BUILD		PCOMMON	IPKNA
*PCOMMON	DSECT NAME; DC AND DS CODE BUILD		PCOMMON	IPKOA
*PCOMMON	DSECT NAME; SOURCE AND OBJECT TEXT OUTPUT		PCOMMON	IPKPA
*PCOMMON	DSECT NAME; RLD OUTPUT PHASE		PCOMMON	IPKQA
*PCOMMON	DSECT NAME; POST PROCESSOR; XREF SORTING AND PRINTING		PCOMMON	IPKRA
*PCOMMON	DSECT NAME; POST PROCESSOR; XREF SORTING		PCOMMON	IPKRB
*PCOMMON	DSECT NAME; POST PROCESSOR; XREF PRINTING		PCOMMON	IPKRC
*PCSR	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		PCSR	
*PCSR	DSECT NAME; OP-CODE LOOKUP AND STMT COMPRESS	4	PCSR	IPKCA
*PCSR	DSECT NAME; M-I AND PROTOTYPE EDITOR		PCSR	IPKCC
*PCSR	DSECT NAME; OVERLAY FOR ICTL,ISEQ,TITLE,COPY,BKEND,EOF		PCSR	IPKCD

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.
**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
*PCSR	DSECT NAME; VARIABLE SYMBOL DECLARATION PROCESSOR		PCSR	IPKDB
*PCSR	DSECT NAME; SEQ SYM REFERENCE PROCESSOR		PCSR	IPKEA
*PCSR	DSECT NAME; GLOBAL EDIT		PCSR	IPKFA
*PCSR	DSECT NAME; EDECK OUTPUT		PCSR	IPKGA
*PCSR	DSECT NAME; ATTRIBUTE PHASE		PCSR	IPKHA
*PCSR	DSECT NAME; PHKGEN		PCSR	IPKIA
*PCSR	DSECT NAME; LOOKUP AND CHECK OF GENERATED OPCODES		PCSR	IPKIC
*PCSR	DSECT NAME; PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1.2,2.1	PCSR	IPKJA
*PCSR	DSECT NAME; ASSIGNMENT PHASE		PCSR	IPKKA
*PCSR	DSECT NAME; SUBSTITUTION PHASE		PCSR	IPKLA
*PCSR	DSECT NAME; SOURCE AND OBJECT TEXT OUTPUT	2.7	PCSR	IPKPA
*PDCEDIT	DSECT NAME; EDITED RECORD FOR DC AND DS, PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES		PDCEDIT	IPKJA
*PDCEDIT	DSECT NAME; EDITED RECORD FOR DC AND DS, ASSIGNMENT PHASE	2.2	PDCEDIT	IPKKA
*PDCEDIT	DSECT NAME; EDITED RECORD FOR DC AND DS, SUBSTITUTION PHASE		PDCEDIT	IPKLA
*PDCEDIT	DSECT NAME; EDITED RECORD FOR DC AND DS, USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4	PDCEDIT	IPKNA
*PDCEDIT	DSECT NAME; EDITED RECORD FOR DC AND DS, DC AND DS CODE BUILD		PDCEDIT	IPKOA
*PDCEDIT	DSECT NAME; EDITED RECORD FOR DC AND DS, SOURCE AND OBJECT TEXT OUTPUT		PDCEDIT	IPKPA
*PDCOUT	DSECT NAME; DC OUTPUT, CONSTANT AND CCW CODE BUILD		PDCOUT	ADDRES
*PDCOUT	DSECT NAME; DC OUTPUT, ADDRESS CONSTANT AND CCW CODE BUILD		PDCOUT	IPKNA
*PDCOUT	DSECT NAME; DC OUTPUT, DC AND DS CODE BUILD		PDCOUT	IPKOA
PDECK	ALIAS FOR BIT3. ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
PEDECK	ALIAS FOR BIT2. ONE TIME INITIALIZER	3	PCOMMON	IPKBA
PEEX	ZERO WORK REGISTER, ASSIGNMENT PHASE	2.2.2	IPKKA000	IPKKA
PENDAD	ADDRESS OF END OPERAND, RLD OUTPUT PHASE	2.8.1	PCOMMON	IPKQA
PENDID	OF END OPERAND, RLD OUTPUT PHASE	2.8.1	PCOMMON	IPKQA
*PERR	DSECT NAME; SOURCE AND OBJECT TEXT OUTPUT	2.7	PERR	IPKPA
*PETFLDS	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		PETFLDS	
*PETFLDS	DSECT NAME; OP-CODE LOOKUP AND STMT COMPRESS		PETFLDS	IPKCA
*PETFLDS	DSECT NAME; M-I AND PROTOTYPE EDITOR		PETFLDS	IPKCC

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.
**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
*PETFLDS	DSECT NAME; OVERLAY FOR ICTL,ISEQ,TITLE,COPY,BKEND,EOF		PETFLDS	IPKCD
*PETFLDS	DSECT NAME; VARIABLE SYMBOL DECLARATION PROCESSOR		PETFLDS	IPKDB
*PETFLDS	DSECT NAME; SEQ SYM REFERENCE PROCESSOR		PETFLDS	IPKEA
*PETFLDS	DSECT NAME; GLOBAL EDIT		PETFLDS	IPKFA
*PETFLDS	DSECT NAME; EDECK OUTPUT		PETFLDS	IPKGA
*PETFLDS	DSECT NAME; PHKGEN		PETFLDS	IPKIA
*PETFLDS	DSECT NAME; PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1.1	PETFLDS	IPKJA
*PETFLDS	DSECT NAME; ASSIGNMENT PHASE	2.2,2.2.2	PETFLDS	IPKKA
*PETFLDS	DSECT NAME; SUBSTITUTION PHASE	2.3	PETFLDS	IPKLA
*PETFLDS	DSECT NAME; USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4.3,2.4	PETFLDS	IPKNA
*PETFLDS	DSECT NAME; DC AND DS CODE BUILD	2.6,2.7.1	PETFLDS	IPKOA
*PETFLDS	DSECT NAME; SOURCE AND OBJECT TEXT OUTPUT	2.7	PETFLDS	IPKPA
*PETR	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		PETR	
*PETR	DSECT NAME; OP-CODE LOOKUP AND STMT COMPRESS		PETR	IPKCA
*PETR	DSECT NAME; M-I AND PROTOTYPE EDITOR		PETR	IPKCC
*PETR	DSECT NAME; OVERLAY FOR ICTL,ISEQ,TITLE,COPY,BKEND,EOF		PETR	IPKCD
*PETR	DSECT NAME; VARIABLE SYMBOL DECLARATION PROCESSOR		PETR	IPKDB
*PETR	DSECT NAME; SEQ SYM REFERENCE PROCESSOR		PETR	IPKEA
*PETR	DSECT NAME; GLOBAL EDIT	1.3	PETR	IPKFA
*PETR	DSECT NAME; EDECK OUTPUT		PETR	IPKGA
*PETR	DSECT NAME; PHKGEN		PETR	IPKIA
*PETR	DSECT NAME; PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1.2,2.1	PETR	IPKJA
*PETR	DSECT NAME; ASSIGNMENT PHASE	2.2,2.2.1	PETR	IPKKA
*PETR	DSECT NAME; SUBSTITUTION PHASE	2.3	PETR	IPKLA
*PETR	DSECT NAME; USING,DROP,MACHIN OP AND S-CONST CODE BUILD	2.4.2,2.4	PETR	IPKNA
*PETR	DSECT NAME; DC AND DS CODE BUILD	2.7.1,2.6	PETR	IPKOA
*PETR	DSECT NAME; SOURCE AND OBJECT TEXT OUTPUT	2.7	PETR	IPKPA
PEVOPND	DSECT NAME; AFTER EVALUATION THE EDITED TEXT CONSISTS, SUBSTITUTION PHASE		PEVOPND	IPKLA
*PFCB	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		PFCB	

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.
**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
*PFCB	DSECT NAME; BASIC INTERFACE ROUTINES AND PCOMMON	1.1	PFCB	IPKAA
*PFCB	DSECT NAME; ONE TIME INITIALIZER		PFCB	IPKBA
*PFCB	DSECT NAME; OP-CODE LOOKUP AND STMT COMPRESS		PFCB	IPKCA
*PFCB	DSECT NAME; M-I AND PROTOTYPE EDITOR		PFCB	IPKCC
*PFCB	DSECT NAME; OVERLAY FOR ICTL,ISEQ,TITLE,COPY,BKEND,EOF		PFCB	IPKCD
*PFCB	DSECT NAME; VARIABLE SYMBOL DECLARATION PROCESSOR		PFCB	IPKDB
*PFCB	DSECT NAME; SEQ SYM REFERENCE PROCESSOR		PFCB	IPKEA
*PFCB	DSECT NAME; GLOBAL EDIT		PFCB	IPKFA
*PFCB	DSECT NAME; EDECK OUTPUT		PFCB	IPKGA
*PFCB	DSECT NAME; ATTRIBUTE PHASE		PFCB	IPKHA
*PFCB	DSECT NAME; PHKGEN		PFCB	IPKIA
*PFCB	DSECT NAME; LOOKUP AND CHECK OF GENERATED OPCODES		PFCB	IPKIC
*PFCB	DSECT NAME; PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES		PFCB	IPKJA
*PFCB	DSECT NAME; ASSIGNMENT PHASE		PFCB	IPKKA
*PFCB	DSECT NAME; SUBSTITUTION PHASE		PFCB	IPKLA
*PFCB	DSECT NAME; ESD INTERLUDE PHASE		PFCB	IPKMA
*PFCB	DSECT NAME; USING,DROP,MACHIN OP AND S-CONST CODE BUILD		PFCB	IPKNA
*PFCB	DSECT NAME; DC AND DS CODE BUILD		PFCB	IPKOA
*PFCB	DSECT NAME; SOURCE AND OBJECT TEXT OUTPUT		PFCB	IPKPA
*PFCB	DSECT NAME; RLD OUTPUT PHASE		PFCB	IPKQA
*PFCB	DSECT NAME; POST PROCESSOR; XREF SORTING AND PRINTING		PFCB	IPKRA
*PFCB	DSECT NAME; POST PROCESSOR; XREF SORTING		PFCB	IPKRB
*PFCB	DSECT NAME; POST PROCESSOR; XREF PRINTING		PFCB	IPKRC
PFILE1	FILE CONTROL BLOCK FOR FILE 1, ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
PFILE2	FILE CONTROL BLOCK FOR FILE 2, ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
PFILE3	FILE CONTROL BLOCK FOR FILE 3, ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
PGBLSIZ	SIZE OF GLOBAL WORK AREAS, PHKGEN	1.5	PCOMMON	IPKIA
*PGVHEAD	DSECT NAME; GLOBAL VECTOR HEADER DSECT, GLOBAL EDIT	1.3.1	PGVHEAD	IPKFA
*PGVHEAD	DSECT NAME; GLOBAL VECTOR HEADER DSECT, PHKGEN	1.5.1,1.5	PGVHEAD	IPKIA

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.
**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
*PHYR	DSECT NAME; ASSIGNMENT PHASE	2.2.1,2.2	PHYR	IPKKA
*PHYR	DSECT NAME; SUBSTITUTION PHASE		PHYR	IPKLA
*PKNA	DSECT NAME; KEYWORD NAME ARRAY DSECT, PHKGEN	1.5.1	PKNA	IPKIA
PLINK	ALIAS FOR BIT4. ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
PLINKBFR	DSECT NAME; SYSPCH/SYSLNK OUTPUT		PLINKBFR	IPKAF
PLIST	ALIAS FOR BIT5. ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
*PMAV	DSECT NAME; MACRO ADDRESS VECTOR DSECT, PHKGEN	1.5.1,1.5	PMAV	IPKIA
PMNABSIZ	LENGTH OF MNA BLOCK, ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
PNPVAL	DSECT NAME; DESCRIBES N/P VALUE, BASIC INTERFACE ROUTINES AND PCOMMON		PNPVAL	IPKAA
POINTBCK	DSECT NAME; BASIC INTERFACE ROUTINES AND PCOMMON		POINTBCK	IPKAA
POLEXP	DSECT NAME; ATTRIBUTE PHASE		POLEXP	IPKHA
POLIFY	SAVE RETURN REGISTER, PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1.1	IPKJA000	IPKJA
PPCARD	DSECT NAME; SYSPCH/SYSLNK OUTPUT		PPCARD	IPKAF
PPCHBUF	DSECT NAME; DESCRIBES PUNCH BUFFER, PUNCH ROUTINE FOR EDECK		PPCHBUF	IPKAC
PRINTER	SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
PRINTOP	SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
PRLINE	DSECT NAME; ***, RLD OUTPUT PHASE		PRLINE	IPKQA
PRNTLINE	PRINT BUFFER FOR XREF DATA, POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	IPKRA001	IPKRA
PROROUT	SAVE OP-CODE MNEMONIC, M-I AND PROTOTYPE EDITOR	1.1,4	IPKCC000	IPKCC
PSEUTBL	SOURCE AND OBJECT TEXT OUTPUT	2.7.1	IPKPA000	IPKPA
*PSTRINGS	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		PSTRINGS	
*PSTRINGS	DSECT NAME; OP-CODE LOOKUP AND STMT COMPRESS		PSTRINGS	IPKCA
*PSTRINGS	DSECT NAME; M-I AND PROTOTYPE EDITOR		PSTRINGS	IPKCC
*PSTRINGS	DSECT NAME; OVERLAY FOR ICTL,ISEQ,TITLE,COPY,BKEND,EOF		PSTRINGS	IPKCD
*PSTRINGS	DSECT NAME; VARIABLE SYMBOL DECLARATION PROCESSOR		PSTRINGS	IPKDB
*PSTRINGS	DSECT NAME; SEQ SYM REFERENCE PROCESSOR		PSTRINGS	IPKEA
*PSTRINGS	DSECT NAME; GLOBAL EDIT		PSTRINGS	IPKFA
*PSTRINGS	DSECT NAME; EDECK OUTPUT		PSTRINGS	IPKGA
*PSTRINGS	DSECT NAME; ATTRIBUTE PHASE	1.4	PSTRINGS	IPKHA
*PSTRINGS	DSECT NAME; PHKGEN		PSTRINGS	IPKIA

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.

**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
*PSTRINGS	DSECT NAME; LOOKUP AND CHECK OF GENERATED OPCODES		PSTRINGS	IPKIC
*PSTRINGS	DSECT NAME; PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES		PSTRINGS	IPKJA
*PSTRINGS	DSECT NAME; ASSIGNMENT PHASE		PSTRINGS	IPKKA
*PSTRINGS	DSECT NAME; SUBSTITUTION PHASE		PSTRINGS	IPKLA
*PSTRINGS	DSECT NAME; SOURCE AND OBJECT TEXT OUTPUT		PSTRINGS	IPKPA
PSYMTABL	START OF HASH TABLE, ASSIGNMENT PHASE	2.2.1	PCOMMON	IPKKA
PSYMTABL	START OF HASH TABLE, SUBSTITUTION PHASE	2.3	PCOMMON	IPKLA
PSYSPARM	ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
PUNCHOP	SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
PUNCHOUT	SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
PUNCHR	ALIAS FOR REPROEDR. PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1	IPKJA000	IPKJA
PVSDSIZE	VSDSIZE, ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
PXREF	ALIAS FOR BIT6. ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
RDIBBASE	ALIAS FOR R15. -> CURRENT DICTIONARY BLOCK, PHKGEN	1.5.1	IPKIA000	IPKIA
REPROEDR	BR IF PRIVATE CODE HAS STARTED, PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES	2.1	IPKJA000	IPKJA
REPROOP	SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
RLDCOL17	DSECT NAME; ***, RLD OUTPUT PHASE		RLDCOL17	IPKQA
*RLDENTRY	DSECT NAME; ONE RLD ENTRY, CONSTANT AND CCW CODE BUILD		RLDENTRY	ADDRES
*RLDENTRY	DSECT NAME; ONE RLD ENTRY, ADDRESS CONSTANT AND CCW CODE BUILD		RLDENTRY	IPKNA
*RLDENTRY	DSECT NAME; ***, RLD OUTPUT PHASE	2.8.1,2.8	RLDENTRY	IPKQA
RLDPCH	SAVE RETURN REG, RLD OUTPUT PHASE	2.8.1	IPKQA000	IPKQA
RLDROUT	SAVE RETURN VALUE, RLD OUTPUT PHASE	2.8,2.8.1	IPKQA000	IPKQA
RLDTAB	DSECT NAME; ***, RLD OUTPUT PHASE		RLDTAB	IPKQA
RSA	POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	IPKRA001	IPKRA
RTBL	ALIAS FOR R2. -> NEXT ENTRY IN PARTBL, PHKGEN	1.5.1	IPKIA000	IPKIA
SEQENT	DSECT NAME; SEQ SYM REFERENCE PROCESSOR		SEQENT	IPKEA
SMTADDR	ADDR OF SMT, SEQ SYM REFERENCE PROCESSOR	1.1.3	IPKEA000	IPKEA
*SMTENT	DSECT NAME; SOURCE MACRO TABLE ENTRY, SEQ SYM REFERENCE PROCESSOR	1.1.3	SMTENT	IPKEA
*SMTENT	DSECT NAME; SOURCE MACRO TABLE ENTRY DSECT, GLOBAL EDIT	1.3	SMTENT	IPKFA

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.
**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
*SMTENT	DSECT NAME; SOURCE MACRO TABLE ENTRY DSECT, EDECK OUTPUT		SMTENT	IPKGA
SMTENTR	SEQ SYM REFERENCE PROCESSOR	1.1.3	IPKEA000	IPKEA
SMTSIZE	SIZE OF SMT BLOCK, ONE TIME INITIALIZER	1.1	PCOMMON	IPKBA
SMTSRCH	GLOBAL EDIT	1.3	IPKFA000	IPKFA
SPACEOP	SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
SRTDIR	POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	IPKRA000	IPKRA
SRTINIT	ENTRY POINT; POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	IPKRA000	IPKRA
SRTLIT	ENTRY POINT, POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	IPKRA000	IPKRA
SRTOUT	ENTRY POINT, POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	IPKRA000	IPKRA
SRTRSA	POST PROCESSOR; XREF SORTING AND PRINTING	2.8.2	IPKRA000	IPKRA
*SSD	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		SSD	
*SSD	DSECT NAME; SEQUENCE SYMBOL DICTIONARY, VARIABLE SYMBOL DECLARATION PROCESSOR	1.1.2	SSD	IPKDB
*SSD	DSECT NAME; SEQ SYM REFERENCE PROCESSOR	1.1.3	SSD	IPKEA
*SSD	DSECT NAME; DSECT FOR SSD ITEM, EDECK OUTPUT		SSD	IPKGA
SSDADDR	DA ADDR OF SSD, VARIABLE SYMBOL DECLARATION PROCESSOR	1.1.2	PCOMMON	IPKDB
SSDLKP	SEQ SYM REFERENCE PROCESSOR	1.1.3	IPKEA000	IPKEA
STACKEL	DSECT NAME; STACK ELEMENT FORMAT, PHKGEN		STACKEL	IPKIA
STACKENT	DSECT NAME; STACK ENTRY, EXTERNAL SYMBOL DICTIONARY		STACKENT	
STARTR	SAVE INPUT POINTER, ASSIGNMENT PHASE	2.2.2	IPKKA000	IPKKA
STDIAG	EDECK OUTPUT	1.2	IPKGA000	IPKGA
STGET	EDECK OUTPUT	1.2	IPKGA000	IPKGA
STSMTGET	EDECK OUTPUT	1.2	IPKGA000	IPKGA
SYMADDR	POINTER TO START OF SYMBOL TAB, ASSIGNMENT PHASE	2.2.1	PCOMMON	IPKKA
SYMADDR	POINTER TO START OF SYMBOL TAB, SUBSTITUTION PHASE	2.3	PCOMMON	IPKLA
TABENT	DSECT NAME; ATTRIBUTE PHASE	1.4	TABENT	IPKHA
TABOP	DSECT NAME; INTERPRETER TABLE, EXTERNAL SYMBOL DICTIONARY		TABOP	
TABSTART	START OF ATTRIBUTE TABLE, ATTRIBUTE PHASE	1.4	IPKHA000	IPKHA
TITLEOP	SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
TXTCARD	SOURCE AND OBJECT TEXT OUTPUT	2.7.1	IPKPA000	IPKPA

*DATA AREA. SEE DATA AREA SECTION FOR DETAILED LAYOUT.

**EXPLANATION OF PLM NUMBERED REFERENCES:
A SINGLE NUMERAL REFERS TO AN OPERATIONS DIAGRAM IN THE METHOD OF OPERATIONS SECTION.
'F', FOLLOWED BY A NUMERAL, REFERS TO A FIGURE IN THE PROGRAM ORGANIZATION SECTION.

SYMBOLIC NAME	DESCRIPTION: NAME AND USE	PLM REF**	CSECT/ DSECT	MODULE/ MCROFCH
VASIZE	DSECT NAME; VALUE AREA SIZE DSECT, PHKGEN		VASIZE	IPKIA
*VSD	DSECT NAME; EXTERNAL SYMBOL DICTIONARY		VSD	
*VSD	DSECT NAME; VARIABLE SYMBOL DECLARATION PROCESSOR		VSD	IPKDB
*VSD	DSECT NAME; DSECT FOR VSD ITEM, EDECK OUTPUT		VSD	IPKGA
VSDADDR	DA ADDR OF VSD IN CORE, VARIABLE SYMBOL DECLARATION PROCESSOR	1.1.2	PCOMMON	IPKDB
WORKAREA	DSECT NAME; WORKAREA WHERE ED.TXT IS BUILT, PRE-PROCESSOR PHASE TO THE ASSEMBLER PHASES		WORKAREA	IPKJA
*WORKDTF	DSECT NAME; DTFSD DSECT FOR OI LOGIC, BASIC INTERFACE ROUTINES AND PCOMMON		WORKDTF	IPKAA
*WORKDTF	DSECT NAME; DTFSD DSECT FOR OI LOGIC, ONE TIME INITIALIZER		WORKDTF	IPKBA
WRTERROR	SOURCE AND OBJECT TEXT OUTPUT	2.7	IPKPA000	IPKPA
*XREFREC	DSECT NAME; THIS IS A DSECT TO DESCRIBE, ASSIGNMENT PHASE	2.2	XREFREC	IPKKA
*XREFREC	DSECT NAME; THIS IS A DSECT TO DESCRIBE, SUBSTITUTION PHASE	2.3	XREFREC	IPKLA
*XRFENTRY	DSECT NAME; XREF RECORD DESCRIPTION, POST PROCESSOR; XREF SORTING AND PRINTING		XRFENTRY	IPKRA
*XRFENTRY	DSECT NAME; ENTRY IN BLOCK, POST PROCESSOR; XREF SORTING		XRFENTRY	IPKRB
*XRFENTRY	DSECT NAME; ENTRY IN BLOCK, POST PROCESSOR; XREF PRINTING		XRFENTRY	IPKRC
XRFTAB	DSECT NAME; XREF BLOCK IDENTIFIERS, POST PROCESSOR; XREF SORTING AND PRINTING		XRFTAB	IPKRA

Data Areas

Purpose of the Section

The purpose of this section is to assist you in interpreting data areas in a storage dump. The entries are listed in alphabetical order and after each entry is a cross-reference of the various fields and their displacements in the data area.

The section contains those data areas referenced by two or more modules of the assembler plus any others which appear in "Method of Operation". The method-of-operation diagrams show how the data areas are used by the assembler.

Immediately following the data areas is a cross-reference listing of all the fields referred to in this section, the name of the DSECT in which they are located, and their displacement within the DSECT.

DATA AREA: **CODE**

SIZE: 2

CREATED BY:
} IPKSA,IPKSB
UPDATED BY:

FUNCTION: Description of an entry in the error message table.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	2	CODEHWD	THE TWO-BYTE CODE CONTAINS FOUR FLAG BITS PLUS A 12-BIT OFFSET IN GLOSSARY:
0	(0)	1	CODESW	FOUR FLAG BITS + FOUR OFFSET BITS
		1...	SFLAG	1=S MODIFIER
		.1..	COMMAFLG	1=COMMA MODIFIER
		..1.	EDFLAG	1=ED MODIFIER
		...1	UNFLAG	1=UN MODIFIER
		 xxxx		OFFSET
1	(1)	1		OFFSET
2	(2)	2	NEXTCODE	

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
CODEHWD	0	(0)
CODESW	0	(0)
COMMAFLG	0	(0)
EDFLAG	0	(0)
NEXTCODE	2	(2)
SFLAG	0	(0)
UNFLAG	0	(0)

*POINTER

DATA AREA: **DIRENTRY**

SIZE: 26

CREATED BY:
UPDATED BY: IPKRA,IPKRB

FUNCTION: Description of an entry in the directory table.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	22	DIRENT	DIRECTORY TABLE
0	(0)	3	CORADDR	ADDR TO MERGE BUFF IN CORE
3	(3)	2	OFFS	OFFSET TO RECORD IN BLOCK
5	(5)	6	BLKNP1	DISK ADDR TO THIS BLOCK
11	(B)	11	KEY	SORT CONTROLFIELD
11	(B)	6	BLKNP2	DISK ADDR TO NEXT BLOCK
17	(11)	2	FILL	NOT USED
19	(13)	1	FLAG	PROGRAM SWITCH
		.... 1...	BLKINSW	1=BLOCK IS IN MAIN STORAGE
20	(14)	6	DEFVAL	BLOCK INFORMATION

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
BLKINSW	19	(13)
BLKINSW	19	(13)
BLKNP1	5	(5)
BLKNP2	11	(B)
*CORADDR	0	(0)
DIRENT	0	(0)
FILL	17	(11)
FLAG	19	(13)
KEY	11	(B)
OFFS	3	(3)

*POINTER

DATA AREA: **EPDMI**

SIZE: Variable (depending upon different types of records)

CREATED BY:
UPDATED BY: IPKCC,IPKDA,IPKDB,IPKFA,IPKHA,IPKIA

FUNCTION: Description of an edited prototype or a macro instruction.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	1		COMMON HEADER (LNG=PCSRHEAD)
6	(6)	1	JSW0023	PROGRAM SWITCH
		1...	SWPMI1	1:ST RECORD OF PROTO OR M-I
		.1..	SWATRINS	ATTRIBUTES TO BE INSERTED (M-I)
		..1.	SWSUBST	SUBSTITUTION IN RECORD (M-I)
		.... .111	EDPMIORG	ONE OF THE FOLLOWING
			FIRST RECORD OF PROTOTYPE	
7	(7)	1	LPNAME	LENGTH OF MACRO NAME
8	(8)	8	PRONAME	MACRO NAME LEFT ADJUSTED
		...1	LEDPROT1	LENGTH OF 1:ST PROTO RECORD
			FIRST RECORD OF M-I	
				BACK LOCATION COUNTER
7	(7)	1	LMNAME	LENGTH OF MACRO NAME
8	(8)	8	MACNAM	MACRO NAME
16	(10)	2	INDEX	M-I INDEX NUMBER
		...1 ..1.	LOMIED1	LENGTH OF 1:ST OUTER M-I REC
			FIRST RECORD OF INNER M-I	
18	(12)	8	SEQFLD	SEQUENCE FIELD FROM CSR
		...1 1.1.	LIMIED1	LENGTH OF 1:ST INNER M-I REC
			SUBSEQUENT	PROTO AND M-I RECORDS
				BACK LOCATION COUNTER
7	(7)	1	ITEMT	TYPE FLAG
8	(8)	1	ITEMFLAG	PROGRAM SWITCH
		1...	NALTSRC	NEW OPD AFTER ALT STMT FORM
		.1..	ITEM1ST	1ST ITEM OF MI OR PROTOTYPE
		..1.	ITEMLISW	END ITEM SINGLE ON NEXT RCD
		.... 1...	ITEMLONG	ITEM LONGER THAN 255
		.... .1..	ITEMATSW	ATTRIBUTES INSERTED (HA)
		.... ..1.	ITEMKWSW	KEYWORD OPERAND SWITCH
		1	ITEMSLSW	SUBLIST OPERAND SWITCH
9	(9)	1	ITEML	ITEM LENGTH MODULO ITEMLADD
		....	ITEMLADD	LENGTH TO BE ADDED IF ITEMLONG
10	(A)	1	ITEM	ITEM CONTENT

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
		.... 1.1.	ITEMORG	IS ONE OF THE FOLLOWING
			TYPES1	SYMBOL WITH T',L',S' AND I' ATTRIBUTE
				BACK LOCATION COUNTER
10	(A)	1	SYM1ITEM	SYM1 ITEM
10	(A)	1	S1ITEMT	TYPE ATTRIBUTE
11	(B)	2	S1ITEML	LENGTH ATTRIBUTE
13	(D)	2	S1ITEMS	SCALE ATTRIBUTE
15	(F)	2	S1ITEMI	INTEGER ATTRIBUTE
17	(11)	1	S1ITEMK	COUNT ATTRIBUTE (STRING LEN)
18	(12)	1	S1ITEMST	STRING
			TYPES2	SYMBOL WITH T' AND L' ATTRIBUTE
				BACK LOCATION COUNTER
10	(A)	1	SYM2ITEM	SYM2ITEM
10	(A)	1	S2ITEMT	TYPE ATTRIBUTE
11	(B)	2	S2ITEML	LENGTH ATTRIBUTE
13	(D)	1	S2ITEMK	COUNT ATTRIBUTE (STRING LEN)
14	(E)	1	S2ITEMST	STRING
			TYPECHAR	SYMBOL/CHARACTER STRING WITH
				T' ATTRIBUTE ONLY
				BACK LOCATION COUNTER
10	(A)	1	CHARITEM	CHAR STRING ITEM AFTER ATTR
10	(A)	1	CHITEMT	TYPE ATTRIBUTE
11	(B)	1	CHITEMK	COUNT ATTRIBUTE
12	(C)	1	CHITEMST	STRING
			TYPESDEF	SELFDEFINING TERM
				BACK LOCATION COUNTER
10	(A)	1	SDEFITEM	SELF-DEFINING TERM ITEM
10	(A)	1	SDITEMT	TYPE ATTRIBUTE
11	(B)	3	SDITEMB	BINARY VALUE
14	(E)	1	SDITEMK	COUNT ATTRIBUTE (STRING LEN)
15	(F)	1	SDITEMST	STRING
			TYPEBC	BASIC CHARACTER EXPRESSION
				(STRING WITH SUBSTITUTION)
				BACK LOCATION COUNTER
10	(A)	1	BCITEM	BASIC CHARACTERITEM
10	(A)	1	BCDUMMY	DUMMY BYTE
		1111 1111	BCCOLUMN	COLUMN VALUE IN BC ITEM
11	(B)	1	BCITEMST	STRING
			TYPESUBS	SUBLIST START
				BACK LOCATION COUNTER
10	(A)	1	SUBSITEM	SUBLIST START ITEM
10	(A)	1	SSITEMK	LENGTH OF SUBLIST OPD (K')
			TYPESUBE	SUBLIST END
				BACK LOCATION COUNTER
10	(A)	1	SUBEITEM	SUBLIST END ITEM
10	(A)	1	SEITEMN	NO OF SUBOPERANDS (N')
			TYPEOM	OMITTED OPERAND OUTSIDE SUBLIST
				BACK LOCATION COUNTER
10	(A)	1	OMITEM	OMITTED OPERAND ITEM
			TYPEER	ERROR. FORMAT DESCRIBED IN OPDERAR (CC)
				BACK LOCATION COUNTER
10	(A)	1	ERITEM	ERROR RECORD
			TYPEPP	POSITIONAL PARAMETER (PROTOTYPE)
				BACK LOCATION COUNTER
10	(A)	1	PITEM	POSITIONAL PARAMETER
			TYPEKP	KEYWORD PARAMETER (PROTOTYPE)
			TYPEK	KEYWORD (M-I)
				BACK LOCATION COUNTER
10	(A)	1	KITEM	KEYWORD NAME

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
			TYPEEND	END OF OPERAND ITEMS
				BACK LOCATION COUNTER
10	(A)	2	ENDITEM	END ITEM
10	(A)	1	EITEMNP	NO OF POSITIONAL OPERANDS
11	(B)	1	EITEMNK	NO OF KEYWORD OPERANDS
			ITEM TYPE FLAGS	
		1.1. ...1	TYPEPP	POSITIONAL PROTOTYPE ITEM
		1.1. ..1.	TYPEKP	KEYWORD PROTOTYPE ITEM
		1.1. ..11	TYPES1	SYMBOL WITH ALL ATTRIBUTES
		1.1. .1..	TYPES2	SYMBOL WITH T' AND L' ATTR
		1.1. .1.1	TYPECHAR	CHARACTER STRING (T' ATTR ONLY)
		1.1. .11.	TYPESDEF	SELFDEFINING TERM ITEM
		1.1. .111	TYPESUBS	SUBLIST START ITEM
		1.1. 1...	TYPESUBE	SUBLIST END ITEM
		1.1. 1..1	TYPEBC	BASIC CHARACTER EXPR M-I ITEM
		1.1. 1.1.	TYPEOM	OMITTED OPERAND OUTSIDE SUBLIST
		1.1. 1.11	TYPEER	ERROR RECORD ITEM
		1.1. 11..	TYPEK	KEYWORD M-I ITEM
		1.1. 11.1	TYPEEND	END OF OPERANDS ITEM

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
BCCOLUMN	10	(A)
BCDUMMY	10	(A)
BCITEM	10	(A)
BCITEMST	11	(B)
CHARITEM	10	(A)
CHITEMK	11	(B)
CHITEMST	12	(C)
CHITEMT	10	(A)
EITEMNK	11	(B)
EITEMNP	10	(A)
ENDITEM	10	(A)
ERITEM	10	(A)
INDEX	16	(10)
ITEM	10	(A)
ITEMATSW	8	(8)
ITEMFLAG	8	(8)
ITEMKWSW	8	(8)
ITEML	9	(9)
ITEMLISW	8	(8)
ITEMLONG	8	(8)
ITEMSLSW	8	(8)
ITEMT	7	(7)
ITEM1ST	8	(8)
KITEM	10	(A)
LMNAME	7	(7)
LPNAME	7	(7)
MACNAM	8	(8)
NALTSRC	8	(8)
OMITEM	10	(A)
PITEM	10	(A)
PRONAME	8	(8)
SDEFITEM	10	(A)
SDITEMB	11	(B)

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
SDITEMK	14	(E)
SDITEMST	15	(F)
SDITEMT	10	(A)
SEITEMN	10	(A)
SEQFLD	18	(12)
SSITEMK	10	(A)
SUBEITEM	10	(A)
SUBSITEM	10	(A)
SWATRINS	6	(6)
SWPMI1	6	(6)
SWSUBST	6	(6)
SYM1ITEM	10	(A)
SYM2ITEM	10	(A)
S1ITEMI	15	(F)
S1ITEMK	17	(11)
S1ITEML	11	(B)
S1ITEMS	13	(D)
S1ITEMST	18	(12)
S1ITEMT	10	(A)
S2ITEMK	13	(D)
S2ITEML	11	(B)
S2ITEMST	14	(E)
S2ITEMT	10	(A)
TYPEBC	11	(B)
TYPECHAR	11	(B)
TYPEEND	11	(B)
TYPEER	11	(B)
TYPEK	11	(B)
TYPEKP	11	(B)
TYPEOM	11	(B)
TYPEPP	11	(B)
TYPESDEF	11	(B)
TYPESUBE	11	(B)
TYPESUBS	11	(B)
TYPES1	11	(B)
TYPES2	11	(B)

*POINTER

DATA AREA: **EPAR**

SIZE: Variable (depending upon different parameters)

CREATED BY:
IPKIA
UPDATED BY:

FUNCTION: Description of an entry in the parameter table.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	1	EPARFLAG	ENTRY FLAG
		.... .111	FLAGPMSK	SIGNIFICANT BITS OF EPARFLAG
1	(1)	1	EPART	TYPE ATTRIBUTE
2	(2)	1	EPARREST	REST OF ENTRY (LNG=VARIABLE)
				NEW TYPE OF ENTRY

```
*******************************************
* FLAG *  BINARY VALUE * LENGTH * STRING *        SYSINDEX
*******************************************
```

0	(0)	1	INDXENT	SYSNDX ENTRY
0	(0)	1	INDXFLAG	ENTRY FLAG A1
1	(1)	4	INDXB	BINARY VALUE
5	(5)	1	INDXCL	LENGTH OF CHARACTER STRING
6	(6)	4	INDXC	CHARACTER STRING
		.... 1.1.	INDXLEN	ENTRY LENGTH

```
*******************************************
*  FLAG  *  T'  *  K'  +  STRING        *        SYSECT
*******************************************
```

10	(A)	1	SECTENT	SYSECT ENTRY
10	(A)	1	SECTFLAG	ENTRY FLAG A5
11	(B)	1	SECTCT	TYPE ATTRIBUTE
12	(C)	1	SECTCL	LENGTH OF CHARACTER STRING
13	(D)	8	SECTC	CHARACTER STRING
		.... 1.1.	SECTLEN	ENTRY LENGTH
		...1 .1..	NAMEENT	NAME ENTRY FOLLOWS
				NEW TYPE OF ENTRY

```
****************************************************
* FLAG *  T'  *  L'  *  S'  *  I'  *  K'  * STRING *  SYMBOL WITH
****************************************************  ALL ATTRIBUTES
```

0	(0)	1	SYM1ENT	SYMBOL WITH ALL ATTRIBUTES
0	(0)	1	SYM1FLAG	ENTRY FLAG A3

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
1	(1)	1	SYM1T	TYPE ATTRIBUTE
2	(2)	2	SYM1L	L' ATTRIBUTE
4	(4)	2	SYM1S	S' ATTRIBUTE
6	(6)	2	SYM1I	I' ATTRIBUTE
8	(8)	1	SYM1K	K' ATTRIBUTE
9	(9)	1	SYM1C	CHARACTER STRING (LMG=K') (LNG=VARIABLE) NEW TYPE OF ENTRY

```
*****************************************
* FLAG *  T'  *  L'  *  K'  *  STRING   *        SYMBOL WITHOUT S'
*****************************************         AND I' ATTRIBUTES
```

0	(0)	1	SYM2ENT	SYMBOL WITH SOME ATTRIBUTES
0	(0)	1	SYM2FLAG	ENTRY FLAG A4
1	(1)	1	SYM2T	TYPE ATTRIBUTE
2	(2)	2	SYM2L	L' ATTRIBUTE
4	(4)	1	SYM2K	K' ATTRIBUTE
5	(5)	1	SYM2C	CHARACTER STRING (LNG=K') (LNG=VARIABLE) NEW TYPE OF ENTRY

```
************************************
* FLAG *  T'  *  K'  *   STRING     *          CHARACTER STRING
************************************
```

0	(0)	1	CHARENT	CHARACTER STRING
0	(0)	1	CHARFLAG	ENTRY FLAG A5
1	(1)	1	CHART	TYPE ATTRIBUTE
2	(2)	1	CHARK	K' ATTRIBUTE
3	(3)	1	CHARC	CHARACTER STRING (LNG = K') (LNG=VARIABLE) NEW TYPE OF ENTRY

```
*********************************************
* FLAG *  T'  * BINARY VALUE *  K'  * STRING *   SELFDEFINING TERM
*********************************************
```

0	(0)	1	SDEFENT	SELFDEFINING TERM ENTRY
0	(0)	1	SDEFFLAG	ENTRY FLAG A6
1	(1)	1	SDEFT	TYPE ATTRIBUTE
2	(2)	3	SDEFB	BINARY VALUE
5	(5)	1	SDEFK	K' ATTRIBUTE
6	(6)	1	SDEFC	CHARACTER STRING (LNG = K') (LNG=VARIABLE) NEW TYPE OF ENTRY

```
******************************************************************
* FLAG * ENTRY LENGTH * N' * K' * START FLAG * SUBENTRY * SUBEFLAG *
******************************************************************
*
*************       ********************************
* SUBENTRY  *       * SUBEFLAG * SUBENTRY * END FLAG * SUBLIST
*************       ********************************
```

0	(0)	1	SUBLENT	SUBLIST ENTRIES
0	(0)	6	SUBLHEAD	HEADER
0	(0)	1	SUBLFLAG	ENTRY FLAG A7

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
3	(3)	1	SUBLN	N' ATTRIBUTE
4	(4)	1	SUBLK	K' ATTRIBUTE
5	(5)	1	SUBEFLAG	SUBENTRY FLAG
6	(6)	1	SUBEL	SUBENTRY LENGTH
7	(7)	1	SUBE	SUB ENTRY TYPE A3,A4,A5,A6 (LNG=VARIABLE)

```
*****************************************************
* SUBEFLAG DS    CL1           NEXT                 *
* SUBEL    DS    CL1            SUB                 *
* SUBE     DS    XL(VARIABLE)   ENTRY (LNG=VARIABLE) *
*                                                   *
*  ETC                                              *
*                                                   *
* SUBLEND DS     CL1           SUBLIST END FLAG (A8) *
*****************************************************
```

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
CHARC	3	(3)
CHARFLAG	0	(0)
CHARK	2	(2)
CHART	1	(1)
EPARFLAG	0	(0)
EPARREST	2	(2)
EPART	1	(1)
FLAGPMSK	0	(0)
INDXB	1	(1)
INDXC	6	(6)
INDXCL	5	(5)
INDXFLAG	0	(0)
SDEFB	2	(2)
SDEFC	6	(6)
SDEFFLAG	0	(0)
SDEFK	5	(5)
SDEFT	1	(1)
SECTC	12	(C)
SECTCL	11	(B)
SECTFLAG	10	(A)
SUBE	7	(7)
SUBEFLAG	5	(5)
SUBEL	6	(6)
SUBLFLAG	0	(0)
SUBLHEAD	0	(0)
SUBLK	4	(4)
SUBLL	1	(1)
SUBLN	3	(3)
SYM1C	9	(9)
SYM1FLAG	0	(0)
SYM1I	6	(6)
SYM1K	8	(8)
SYM1L	2	(2)
SYM1S	4	(4)
SYM1T	1	(1)
SYM2C	5	(5)
SYM2FLAG	0	(0)
SYM2K	4	(4)
SYM2L	2	(2)
SYM2T	1	(1)

*POINTER

DATA AREA: **ERRBYTES**

SIZE: 11

CREATED BY:
UPDATED BY: IPKKA,IPKLA

FUNCTION: Description of an entry in the error stack.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	11	ERRAREA	
0	(0)	2	ERRCONST	
0	(0)	1	ERRNO	ERROR NUMBER
1	(1)	1	ERRSW	PROGRAM SWITCH
		1...	TEXT	TAKE STRING FROM SAVE AREA
		.1..	OPND	TAKE OPERAND NUMBER
2	(2)	1	ERRLNG	RINPT OR RWJA
3	(3)	8	ERRTXT	

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
ERRAREA	0	(0)
ERRCONST	0	(0)
ERRLNG	2	(2)
ERRNO	0	(0)
ERRSW	1	(1)
ERRTXT	3	(3)
OPND	1	(1)
TEXT	1	(1)

*POINTER

DATA AREA: **ERRENT**

SIZE: 8

CREATED BY:
UPDATED BY: IPKDA,IPKDB

FUNCTION: Description of an entry in the error stack.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	2	ERRINFO	
0	(0)	1	JSW0025	PROGRAM SWITCH
		1	SWSTR	STRING WANTED XSWITCH
0	(0)	1	ERRNO	ERROR NUMBER
2	(2)	3	STRPTR	STRING POINTER (RINPT)
5	(5)	3	EOFLDPTR	CURRENT END OF FIELD (EOPPTR)
		.... 1...	ERRENTL	STACK ENTRY LENGTH
		....	NOSTR	OMITTED STRING IN ERROR RECORD
		STRING REQUIRED/NOT REQUIRED INDICATORS IN ERROR CALL		
		1	STRING	STRING REQUIRED
		....	NOSTRING	STRING NOT REQUIRED

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
EOFLDPTR	5	(5)
ERRINFO	0	(0)
ERRNO	0	(0)
NOSTRING	5	(5)
STRING	5	(5)
STRPTR	2	(2)
SWSTR	0	(0)

*POINTER

DATA AREA: **ESDENTRY**

SIZE: 16

CREATED BY:
UPDATED BY: } IPKKA,IPKMA

FUNCTION: Description of an entry in the ESD table.

DISPLMNT			FIELD	DESCRIPTION: CONTENTS,
DEC	(HEX)	SIZE	NAME	MEANING/USE
				IN THE ESD TABLE
0	(0)	1	ESDTYPE	
		....	SDTYPE	
0	(0)	1	VSDTYPE	
		1	LDTYPE	
0	(0)	1	VLDTYPE	
		.... ..1.	ERTYPE	
0	(0)	1	VERTYPE	
		.... .1..	PCTYPE	
0	(0)	1	VPCTYPE	
		.... .1.1	CMTYPE	
0	(0)	1	VCMTYPE	
		.... 1...	DSTYPE	
0	(0)	1	VDSTYPE	
		.... 1..1	VCTYPE	
0	(0)	1	VVCTYPE	
		.... 1.1.	WXTYPE	
0	(0)	1	VWXTYPE	
0	(0)	1		
1	(1)	2	ESDESDID	* ESDID
3	(3)	3	ESDLCTR	* CURRENT LOCCNTR OF CONT. SEC
6	(6)	3	ESDHILC	* HIGHEST LOCCNTR OF CONT. SEC
9	(9)	8	ESDSYM	* SYMBOL
17	(11)	1	ESDNXT	* NEXT ENTRY

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
CMTYPE	0	(0)
DSTYPE	0	(0)
ERTYPE	0	(0)
ESDESDID	1	(1)
*ESDHILC	6	(6)
*ESDLCTR	3	(3)
ESDNXT	17	(11)
ESDSYM	9	(9)
ESDTYPE	0	(0)
LDTYPE	0	(0)
PCTYPE	0	(0)
SDTYPE	0	(0)
VCMTYPE	0	(0)
VCTYPE	0	(0)
VDSTYPE	0	(0)
VERTYPE	0	(0)
VLDTYPE	0	(0)
VPCTYPE	0	(0)
VSDTYPE	0	(0)
VVCTYPE	0	(0)
VWXTYPE	0	(0)
WXTYPE	0	(0)

*POINTER

DATA AREA: **EVALSTCK**

SIZE: 7-9

CREATED BY: }
UPDATED BY: } IPKKA,IPKLA

FUNCTION: Description of an entry in the evaluate routine stack.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	1	RANR	NUMBER OF REL ATTR 0=ABSOLUTE
1	(1)	4	EVALUE	VALUE
5	(5)	2	EVLENGTH	LENGTH OF ABSOLUTE ENTRY
7	(7)	1	EVVARY	END OF ABSOLUTE ENTRY
				ORG BEFORE LENGTH FOR VAR ENTRY
5	(5)	1	EVPLUS	SIGN
6	(6)	2	EVRELOC	RELOCATION ATTRIBUTE
8	(8)	2	RELLEN	LENGTH OF ENTRY IN STACK
10	(A)	1	EVNXT	NEXT ENTRY IN STACK
				SAME BUT WITH DISPLACEMENT=0
0	(0)	1	PLORMIN	SIGN
1	(1)	2	RAONE	RELOCATION ATTRIBUTE
3	(3)	1	NXTRA	NEXT PAIR OF SIGN AND RELATTR
				DISPLACEMENT = 0
0	(0)	2	STLENGTH	LENGTH OF STACK ENTRY
				AT END OF ENTRY

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
EVALUE	1	(1)
EVLENGTH	5	(5)
EVNXT	10	(A)
EVPLUS	5	(5)
EVRELOC	6	(6)
EVVARY	7	(7)
NXTRA	3	(3)
PLORMIN	0	(0)
RANR	0	(0)
RAONE	1	(1)
RELLEN	8	(8)
STLENGTH	0	(0)

*POINTER

DATA AREA: **GARD**

SIZE: 7

CREATED BY: }
UPDATED BY: } IPKDB,IPKFA

FUNCTION: Description of the global array.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	2	GARLEN	RECORD LENGTH
2	(2)	1		GLOBAL ARRAY IOP (LNG=PCSRIOP)
6	(6)	1	GARFLAG	NOT USED
		.... .111	GARITEM	DESCRIBED BY DSECT GARENT

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
GARFLAG	6	(6)
GARLEN	0	(0)

*POINTER

DATA AREA: **GARENT**

SIZE: 4-6

CREATED BY:
UPDATED BY: IPKDB,IPKFA

FUNCTION: Description of a global array item.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	1	GARTYPE	TYPE OF GLOBAL SYMBOL
1	(1)	2	GARNDX	GLOBAL SYMBOL INDEX
3	(3)	1	GARLGTH	LENGTH OF SYMBOL NAME
		.... .1..	GARSYM	SYMBOL NAME, VARIABLE LENGTH
4	(4)	2	GARDIM	DIMENSION
		1111 1111	GAREND	INDICATES LAST GAR

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
GARDIM	4	(4)
GAREND	4	(4)
GARLGTH	3	(3)
GARNDX	1	(1)
GARTYPE	0	(0)

*POINTER

DATA AREA: **GSDENTRY**

SIZE: 5-11

CREATED BY: } IPKFA
UPDATED BY: }

FUNCTION: Description of an entry in the global symbol dictionary.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	1	GSDTYPE	SYMBOL TYPE
1	(1)	2	GSDNDX	INDEX NUMBER
3	(3)	1	GSDLEN	LENGTH OF SYMBOL NAME
4	(4)	7	GSDSYM	SYMBOL NAME
4	(4)	2	GSDDIM	DIMENSION
				LENGTH AND FLAG HAVE SAME ADDR.
3	(3)	1	GSDFLG	ENDFLAG
		1111 1111	EGSD	END OF GSD
		1111 1.1.	EGSDB	END OF GSD BLOCK
		.... ..1.	DIMBIT	BIT6 IN GSDTYPE IS ON IF SYMBOL DIMENSIONED

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
DIMBIT	3	(3)
EGSD	3	(3)
EGSDB	3	(3)
GSDDIM	4	(4)
GSDFLG	3	(3)
GSDLEN	3	(3)
GSDNDX	1	(1)
GSDSYM	4	(4)
GSDTYPE	0	(0)

*POINTER

DATA AREA: **IJCPTAB**

SIZE: 232

CREATED BY:
UPDATED BY: IPKAF,IPKAG,IPKTA

FUNCTION: Table used by CPMOD.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	30	IJJCPCCB	CCB, ADDR OF LOGIC, DTF TYPE, OPEN SW, FILE NAME
		.... ..1.	IJJCOMBT	1ST COMMUNICATION BYTE
		11	IJJCMBT2	2ND COMMUNICATION BYTE
		.... .1..	IJJCPCB4	CSW STATUS
		.... 1...	IJJCPTCB	CCW ADDR
		...1 .11.	IJJFLNME	FILE NAME
30	(1E)	10	IJJCPF1	F1 ADDR, WORK AREA OR FLAG & SEQ. NO. & OPEN SW
		..1.	IJJCPLD2	LOAD INSTRUCTION FOR SYSTEM UNIT
40	(28)	2	IJJCPXTN	XTNT SEQ. NO. OF LAST XTNT & WORK AREA
		..1. 1...	IJJCPHCD	FOR ADJUSTMENT OF PUNCH CODE
42	(2A)	1	IJJCPSWS	
43	(2B)	1	IJJC2NSW	INDICATOR FOR OPEN AND LOGIC
44	(2C)	1	IJJALSW	LOGICAL INDICATORS
45	(2D)	3	IJJCP2ND	I/O AREA
48	(30)	4	IJJCPSCW	CCW OR WORK AREA
52	(34)	2	IJJLOHED	HH LOWER HEAD LIMIT
54	(36)	6	IJJCPMAX	CCHH UPPER LIMIT & BB SEEK ADDR
		..11 1...	IJJCCWE1	PUNCH ERROR CCW IF DEVICE IS 2540 PUNCH
60	(3C)	4	IJJCPSEK	CCHH
64	(40)	4	IJJCPREC	
		.1..	IJJCCWE2	PUNCH ERROR CCW2 IF DEVICE IS 2540 PUNCH
		.1..	IJJCPREAD	EOF ADD
68	(44)	4	IJJCPUPP	UPPER LIMIT
72	(48)	1	IJJCPRMX	NO. OF RECORDS/TRACK
		.1.. 1...	IJJCPSV1	80 BYTE CARD IMAGE SAVEAREA

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
73	(49)	1	IJJFRSTR	1 FIRST REC FOR INPUT OR OUTPUT
74	(4A)	2	IJJCPADJ	ADJUSTMENT FOR CCW ADDRESS
76	(4C)	4	IJJCPCNT	CCHH COUNT FIELD
80	(50)	2	IJJCPCTR	RECORD & KEY LENGTH
82	(52)	2	IJJCPDAT	DATA LENGTH
84	(54)	4	IJJLOAD	
88	(58)	24	IJJCPSST	SEEK, SEARCH, TIC CCW'S
112	(70)	16	IJJCPCCW	CCW'S FOR INPUT AND OUTPUT
128	(80)	24	IJJCPVER	VERIFY CCW'S
152	(98)	8	IJJECCW1	2540 PUNCH ERROR CCW1
		1..1 1...	IJJCPSV2	80 BYTE CARD IMAGE SAVEAREA
160	(A0)	8	IJJECCW2	2540 PUNCH ERROR CCW2
168	(A8)	64	IJJSAVEA	RESERVED FOR SAVE AREA

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
IJJALSW	44	(2C)
IJJCPADJ	74	(4A)
IJJCPCCB	0	(0)
IJJCPCCW	112	(70)
IJJCPCNT	76	(4C)
IJJCPCTR	80	(50)
IJJCPDAT	82	(52)
IJJCPF1	30	(1E)
IJJCPMAX	54	(36)
IJJCPREC	64	(40)
IJJCPRMX	72	(48)
IJJCPSCW	48	(30)
IJJCPSEK	60	(3C)
IJJCPSST	88	(58)
IJJCPSWS	42	(2A)
IJJCPUPP	68	(44)
IJJCPVER	128	(80)
IJJCPXTN	40	(28)
IJJCP2ND	45	(2D)
IJJC2NSW	43	(2B)
IJJECCW1	152	(98)
IJJECCW2	160	(A0)
IJJFRSTR	73	(49)
IJJLOAD	84	(54)
IJJLOHED	52	(34)
IJJSAVEA	168	(A8)

*POINTER

DATA AREA: **INDENTRY**

SIZE: 17

CREATED BY: }
UPDATED BY: } IPKRA,IPKRB

FUNCTION: Description of an entry in an index table.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	17	INDENT	
0	(0)	11	INDKEY	LOWEST KEY IN A DIRECTORY BLK
11	(B)	6	INDNP	NOTE VALUE TO DIR BLOCK

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
INDENT	0	(0)
INDKEY	0	(0)
INDNP	11	(B)

*POINTER

DATA AREA: **KEYTAB**

SIZE: 7

CREATED BY:
UPDATED BY: IPKDB,IPKFA,IPKIA

FUNCTION: Description of the keyword table.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	2	KWTLEN	RECORD LENGTH
2	(2)	1		KEYWORD TABLE IOP (LNG=PCSRIOP)
6	(6)	1	JSW0032	PROGRAM SWITCH
		1...	SWLASTKW	INDICATES LAST RECORD
		.... .111	KWTITEM	DESCRIBED BY DSECT KWTENT

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
KWTLEN	0	(0)
SWLASTKW	6	(6)

*POINTER

(This page intentionally left blank.)

DATA AREA: **MNAENT**

SIZE: 4-11

CREATED BY:
UPDATED BY: } IPKCA,IPKCC,IPKCD,IPKFA

FUNCTION: Description of an entry in the macro name array.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	2	MNANDX	INDEX NUMBER OF MACRO
2	(2)	1	MNALEN	LENGTH OF NAME
3	(3)	8	MAN	NAME 1-8 CHARS
				BACK TO LENGTH
2	(2)	1	EFLG	END OF BLOCK END OF ARRAY FLAG
		.... 111.	MAXENTL	MAX ENTRY PLUS END FLAG
		1111 1111	EMNA	END OF MNA FLAG
		1111 1.1.	EMNAB	END OF MNA BLOCK FLAG

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
EFLG	2	(2)
EMNA	2	(2)
EMNAB	2	(2)
MAN	3	(3)
MNALEN	2	(2)
MNANDX	0	(0)

*POINTER

DATA AREA: **NTABFMT**

SIZE: 2

CREATED BY: UPDATED BY:	IPKNA

FUNCTION: Description of an entry in an error table.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
		....	NENTRY1	FIRST ENTRY
0	(0)	1	NERNUMB	ERROR NUMBER
		1	NENTRY2	SECOND ENTRY
1	(1)	1	NOPRNUMB	OPERAND NUMBER

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
NERNUMB	0	(0)
NOPRNUMB	1	(1)

*POINTER

DATA AREA: **OCSTMH**

SIZE: 13

CREATED BY: } IPKDB,IPKEA
UPDATED BY: }

FUNCTION: Description of the open code start record.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	1		HEADER AND FLAG A (LNG=
7	(7)	1	NPMIB	N/P TO INFO-BLOCK (LNG=PNPRW)
		..1. .1.1	LOCSTMH	EA LENGTH OF OCST MHR

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
LOCSTMH	7	(7)
NPMIB	7	(7)

*POINTER

DATA AREA: **PCOMMON**

SIZE: Variable

CREATED BY:
UPDATED BY: } All modules except: IPKAC,IPKAF,IPKAG,IPKAJ

FUNCTION: Description of all common data areas and equates between the modules.

```
 DISPLMNT                     FIELD        DESCRIPTION: CONTENTS,
 DEC   (HEX)      SIZE        NAME         MEANING/USE


**************************************************************
*        WHOLE ASSEMBLER                                     *
**************************************************************
***************************************************************
*                                                             *
*        BRANCH TABLE FOR THE INTERFACE MACROS                *
*                                                             *
***************************************************************


                               CODE FOR LINKAGE TO TRACE PROGRAM

84      (54)       6                          NO COMMENTS

              .1.1 1.1.      PABEND
                                              BYPASS INTERFACE ROUTINES


***************************************************************
*                                                             *
*        EQUATES FOR WORK AREA LENGTHS. PWA$X MEANS LENGTH OF *
*        WORK AREA FOR FILE $ IN PHASE X                      *
*                                                             *
***************************************************************


             ..1. 11..      PWA2C        PUT COMPRESSED SOURCE RECORDS
             1..1 ....      PWA1D        PUT CSR AND EDITED TEXT RCDS
             ..1. 11..      PWA2D        GET CSR FROM C
             1..1 ....      PWA1E        GET CSR AND EDITED TXT FROM D
             1..1 ....      PWA2E        PUT OC CSR AND EDITED TEXT
             ..1. 11..      PWA3E        PUT MACRO CSR
             1..1 ....      PWA2F        GET OC CSR AND EDITED TXT FR E
             ..1. 11..      PWA3F        PUT ERROR RCDS
             1..1 ....      PWA1I        GET MACRO CSR AND EDITED TXT
             1..1 ....      PWA2I        GET OC CSR AND EDITED TEXT
             ..1. 11..      PWA3I        PUT GENERATED CSR
             ...1 1.1.      PWA2J        PUT EDITED RCD
             ..1. 11..      PWA3J        GET  COMPRESSED SOURCE RCD
             ...1 1.1.      PWA1K
             ...1 1.1.      PWA1L        TRANSFER RCD FROM L TO N
             ...1 1.1.      PWA2L        GET ED RCD WRITTEN BY J
             ...1 1.1.      PWA1N        RECEIVE RCD FROM L
```

```
DISPLMNT                        FIELD        DESCRIPTION:  CONTENTS,
DEC             SIZE            NAME         MEANING/USE

             ...1 1.1.          PWA2N
             ...1 1.1.          PWA1O        TRANSFER RCD FROM O TO P
             ...1 1.1.          PWA2O        GET RCD WRITTEN BY N
             ...1 1.1.          PWA1P        RECEIVE RCD FROM O
             ...1 1.1.          PWA2P
             ..1. 11..          PWA3P        GET COMPRESSED SOURCE RCD


****************************************************************
*                                                              *
*        INTERNAL CODE EQUATES                                 *
*                                                              *
****************************************************************

             .... ....          P0           NUMBER 0
             .... ...1          P1           1
             .... ..1.          P2           2
             .... ..11          P3           3
             .... .1..          P4           4
             .... .1.1          P5           5
             .... .11.          P6           6
             .... .111          P7           7
             .... 1...          P8           8
             .... 1..1          P9           9
             .... 1.1.          A            LETTER A
             .... 1.11          B            B
             .... 11..          C            C
             .... 11.1          D            D
             .... 111.          E            E
             .... 1111          F            F
             ...1 ....          G            G
             ...1 ...1          H            H
             ...1 ..1.          I            I
             ...1 ..11          J            J
             ...1 .1..          K            K
             ...1 .1.1          L            L
             ...1 .11.          M            M
             ...1 .111          N            N
             ...1 1...          O            O
             ...1 1..1          P            P
             ...1 1.1.          Q            Q
             ...1 1.11          R            R
             ...1 11..          S            S
             ...1 11.1          T            T
             ...1 111.          U            U
             ...1 1111          V            V
             ..1. ....          W            W
             ..1. ...1          X            X
             ..1. ..1.          Y            Y
             ..1. ..11          Z            Z
             ..1. .1..          DOLLAR       $
             ..1. .1.1          NUMBER       #
             ..1. .11.          AT           @
             ..1. .111          EQUAL        SPEC.  =
             ..1. 1...          LPARN        (
             ..1. 1..1          PLUS         +
             ..1. 1.1.          MINUS        -
             ..1. 1.11          ASTER        *
             ..1. 11..          SLASH        /
             ..1. 11.1          RPARN        )
```

DISPLMNT DEC (HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
	..1. 111.	COMMA	,
	..1. 1111	BLANK	BLANK
	..11	QUOTE	'
	..11 ...1	AMPER	&
	..11 ..1.	DOT	.
	....	NUMMIN	MINIMUM NUMBER CODE
	.... 1..1	NUMMAX	MAXIMUM NUMBER CODE
	.... 1.1.	ALFAMIN	MINIMUM ALPHA CODE
	..1. .11.	ALFAMAX	MAXIMUM ALPHA CODE
	..1. 1..1	OPERMIN	MINIMUM OPERATOR CODE
	..1. 11..	OPERMAX	MAXIMUM OPERATOR CODE

```
**************************************************************
*                                                            *
*            PSEUDO OPERATION CODE EQUATES                   *
*                                                            *
*                                                            *
*      NOTE: THE ASSEMBLER CODE IS DEPENDENT ON THE ORGANI-  *
*            ZATION OF THE OP-CODES. SHOULD IT BE CHANGED    *
*            THE CODE MAY HAVE TO BE UPDATED                 *
*                                                            *
**************************************************************
```

DISPLMNT DEC (HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
	....	SUBST	SUBSTITUTED OP-CODE
	1	REPROED	REPROED STATEMENT
	.... ..1.	PUNCH	PUNCH
	11	CNOP	CNOP
	.... .1..	ORG	ORG
	.... .1.1	END	END
	.... .11.	ENTRY	ENTRY
	.... .111	EXTRN	EXTRN
	.... 1...	WXTRN	WXTRN
	.... 1..1	USING	USING
	.... 1.1.	DROP	DROP
	.... 1.11	DC	DC
	.... 11..	DS	DS
	.... 11.1	DCL	LITERAL DC
	.... 111.	EQU	EQU
	 1111	EQUL	LITERAL EQU
	...1	CCW	CCW
	...1 ...1	START	START
	...1 ..1.	CSECT1ST	START OF 1ST CSECT
	...1 ..11	LTORG	LTORG
	...1 .1..	CSECT	CSECT
	...1 .1.1	DSECT	DSECT
	...1 .11.	COM	COM
	...1 .111	REPRO	REPRO
	...1 1...	EJECT	EJECT
	...1 1..1	PRINT	PRINT
	...1 1.1.	SPACE	SPACE
	...1 1.11	TITLE	TITLE

DISPLMNT DEC (HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
	...1 11..	ICTL	ICTL
	...1 11.1	ISEQ	ISEQ
	...1 111.	CMENT	* COMMENT
	...1 1111	COPY	COPY
	..1.	MI	M-I SOURCE 1ST CARD
	..1. ...1	ERROR	ERROR RECORD
	..1. ..1.	MIC	MACRO INSTR. SOURCE CONT. CARDS
	..1. ..11	MNOTE	MNOTE
	..1. .1..	MIED	M-I EDITED RECORD
	..1. .1.1	MCMENT	.* COMMENT
	..1. .11.	MACRO	MACRO
	..1. .111	MEND	MEND
	..1. 1...	MEXIT	MEXIT
	..1. 1..1	ANOP	ANOP
	..1. 1.1.	SETA	SETA
	..1. 1.11	SETB	SETB
	..1. 11..	SETC	SETC
	..1. 11.1	ACTR	ACTR
	..1. 111.	AIF	ACTR
	..1. 111.	AIFB	AIFB
	..1. 1111	AGO	AGO
	..1. 1111	AGOB	AGOB
	..11	GBLA	GBLA
	..11 ...1	GBLB	GBLB
	..11 ..1.	GBLC	GBLC
	..11 ..11	LCLA	LCLA
	..11 .1..	LCLB	LCLB
	..11 .1.1	LCLC	LCLC
	..11 .11.	PROTO	PROTOTYPE SOURCE
	..11 .111	PROTOED	PROTOTYPE EDITED REC
	..11 1...	MHR	MACRO HEADER RECORD
	..11 1..1	KT	KEYWORD TABLE RECORD
	..11 1.1.	GAR	GLOBAL ARRAY RECORD
	..11 1.11	MNA	MAC NAME ARRAY RECORD
	..11 11..	OCST	OPEN CODE START RECORD
	..11 11.1	UNDEF	UNDEFINED OPCODE
	..11 111.	LITSRC	LITERAL SOURCE RECORD

```
***************************************************************
*                                                             *
*        REGISTER EQUATES                                     *
*                                                             *
***************************************************************
```

	....	R0	REGISTER 0
	1	R1	1
	.... ..1.	R2	2
	11	R3	3
	.... .1..	R4	4
	.... .1.1	R5	5
	.... .11.	R6	6
	.... .111	R7	7
	.... 1...	R8	8
	.... 1..1	R9	9
	.... 1.1.	R10	10
	.... 1.11	R11	11
	.... 11..	R12	12
	.... 11.1	R13	13

```
DISPLMNT       SIZE          FIELD          DESCRIPTION:  CONTENTS,
DEC   (HEX)                  NAME           MEANING/USE


             .... 111.       R14            14
             .... 1111       R15            15
             .... .11.       ROFFS          PARAMETER REGS
             .... .111       RPARM          *FOR
             .... 1...       RFILE          *INTERFACE MACROS
             .... 1..1       RINPT          PGETL RECORD POINTER
             .... 1.1.       ROUTPT         PPUTL RECORD POINTER
             .... 1.11       RBA            BASE REGISTERS
             .... 11.1       RBIF           INTERFACE BASE REGISTER
             .... 111.       RBR            STANDARD BRANCH REGISTER
             .... .11.       RBRSAVE        BRANCH REGISTER FOR PSAVE

***************************************************************
*                                                             *
*        BIT EQUATES FOR BIT HANDLING MACROS                  *
*                                                             *
***************************************************************

             1... ....       BIT0           1000 0000
             .1.. ....       BIT1           0100 0000
             ..1. ....       BIT2           0010 0000
             ...1 ....       BIT3           0001 0000
             .... 1...       BIT4           0000 1000
             .... .1..       BIT5           0000 0100
             .... ..1.       BIT6           0000 0010
             .... ...1       BIT7           0000 0001
             1111 1111       BITFF          1111 1111

***************************************************************
*                                                             *
*        MASK EQUATES                                         *
*                                                             *
***************************************************************


             .... .111       ADDR           'ICM MASK' FOR ADDRESS
             .... 1...       HIGHBYTE       HIGH ORDER BYTE OF REGISTER
             .... ...1       LOWBYTE        LOW ORDER BYTE OF REGISTER

***************************************************************
*                                                             *
*        MISCELLANEOUS                                        *
*                                                             *
***************************************************************



1658    (67A)        1          JSW0005        PROGRAM SWITCH

              1... ....         PNWTRKSW       NEXT PWRITE STARTS ON A NEW TRK
              .1.. ....         POPSW          TELLS PRETURN TO POP THE SAVE
              ..1. ....         PEOBSW         '1' MEANS END OF BOOK ON SYSSLB
```

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
		...1	PLBEOFSW	'1' MEANS EOF ON SYSSLB
		.... 1...	PNOBKSW	'1' MEANS BOOK NOT FND
		.... .1..	PRLD	OPTION RLD
		.... ..1.	PSXREF	OPTION SXREF
1659	(67B)	1	JSW0006	PROGRAM SWITCH
		1...	PINEOFSW	'1' MEANS EOF ON SYSIPT
		.1..	PALIGN	OPTION ALIGN
		...1	PDECK	OPTION DECK
		.... 1...	PLINK	OPTION LINK OR CATALS
		.... .1..	PLIST	OPTION LIST
		.... ..1.	PXREF	OPTION XREF
		1	PDF	OPTION SUBLIB
1660	(67C)	6	NPTEMP **	TEMPORARY STORAGE FOR READNEXT NOTE VALUE
1660	(67C)	2	NPTEMPCC	CYLINDER
1662	(67E)	1	NPTEMPH	HEAD
1663	(67F)	1	NPTEMPR	RECORD
1664	(680)	2	NPTEMPTB	REMAINING TRACK CAPACITY
1666	(682)	3	PHICORE	ADDRESS OF HIGHEST BYTE, THAT IS NOT USED FOR BUFFERS
1669	(685)	1	PABENDC	ABEND CODE
1670	(686)	2		
1670	(686)	2	PLRECLN	LENGTH OF RECORD LENGTH
1672	(688)	4		
1672	(688)	16	IFSAVE	INTERFACE ROUTINE SAVE AREA

```
*************************************************************
*                                                           *
*        ASSEMBLER FLAGS                                    *
*                                                           *
*************************************************************
```

		..11 ..11	UNMINUS	UNARY MINUS
		..11 .1..	LATTR	LENGTH ATTRIBUTE
		..11 .1.1	SYMFLAG	SYMBOL FLAG
		..11 .11.	LOCCTR	LOCATION COUNTER FLAG
		..11 .111	SELFLAGL	SELFDEFINING TERM LARGE
		..11 1...	SELFLAGS	SELFDEFINING TERM SMALL
		..11 1..1	ENDEXPR	END OF EXPRESSION
		..11 1.1.	EFLOPD	END OF OPERAND FIELD
		..11 1.11	OPRDFLAG	BEGINNING OF OPERAND
		..11 11..	ERROPND	ERRONEOUS OPERAND
		..11 11.1	AD1FLAG	ADDR1 FIELD PRESENT
		..11 111.	AD2FLAG	ADDR2 FIELD PRESENT
		..11 1111	CODEFLAG	GEN. CODE FIELD PRESENT

```
*************************************************************
*                                                           *
*        FILE CONTROL BLOCKS                                *
*                                                           *
*************************************************************
```

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
1688	(698)	35	PFILE1	FILE CONTROL BLOCK FOR FILE 1
1688	(698)	3	PDTFADR1	ADDRESS OF DTFSD
1691	(69B)	3	BUFPT1	POINTER TO NEXT RCD IN BUFFER
1694	(69E)	3	BUFADDR1	ADDRESS OF BUFFER
1697	(6A1)	2	PBUFLEN1	BUFFER LENGTH
1699	(6A3)	3	PENDBUF1	ADDRESS OF LAST BYTE OF BUFFER
1702	(6A6)	3	PWAADDR1	ADDRESS OF WORKAREA
1705	(6A9)	1		SWITCHES (SEE DSECT PFCB)
1706	(6AA)	3	PEOFADR1	ADDRESS OF END-OF-FILE ROUTINE
1709	(6AD)	8	PNPOINT1	NOTE/POINT VALUE
1717	(6B5)	6	PNEXTNP1	N/P VALUE FOR NEXT BLOCK
1723	(6BB)	35	PFILE2	FILE CONTROL BLOCK FOR FILE 2
1723	(6BB)	3	PDTFADR2	ADDRESS OF DTFSD
1726	(6BE)	3	BUFPT2	POINTER TO NEXT RCD IN BUFFER
1729	(6C1)	3	BUFADDR2	ADDRESS OF BUFFER
1732	(6C4)	2	PBUFLEN2	BUFFER LENGTH
1734	(6C6)	3	PENDBUF2	ADDRESS OF LAST BYTE OF BUFFER
1737	(6C9)	3	PWAADDR2	ADDRESS OF WORK AREA
1740	(6CC)	1		SWITCHES (SEE DSECT PFCB)
1741	(6CD)	3	PEOFADR2	ADDRESS OF END-OF-FILE ROUTINE
1744	(6D0)	8	PNPOINT2	NOTE/POINT VALUE
1752	(6D8)	6	PNEXTNP2	N/P VALUE FOR NEXT BLOCK
1758	(6DE)	35	PFILE3	FILE CONTROL BLOCK FOR FILE 3
1758	(6DE)	3	PDTFADR3	ADDRESS OF DTFSD
1761	(6E1)	3	BUFPT3	POINTER TO NEXT RCD IN BUFFER
1764	(6E4)	3	BUFADDR3	ADDRESS OF BUFFER
1767	(6E7)	2	PBUFLEN3	BUFFER LENGTH
1769	(6E9)	3	PENDBUF3	ADDRESS OF LAST BYTE OF BUFFER
1772	(6EC)	3	PWAADDR3	ADDRESS OF WORK AREA
1775	(6EF)	1		SWITCHES (SEE DSECT PFCB)
1776	(6F0)	3	PEOFADR3	ADDRESS OF END-OF-FILE ROUTINE
1779	(6F3)	8	PNPOINT3	NOTE/POINT VALUE
1787	(6FB)	6	PNEXTNP3	N/P VALUE FOR NEXT BLOCK
		..11 111.	PMAXBSIZ	MAX BLOCK LENGTH OFFSET IN DTF

*

EQUATES FOR CONTROL NUMBERS FOR PRINT *

*

		1111 ...1	PEJ	EJECT, THEN PRINT
		.1..	PSS	SINGLE SPACE, THEN PRINT
		1111	PDS	DOUBLE SPACE, THEN PRINT
		.11.	PTS	TRIPLE SPACE, THEN PRINT

*

PUSH-DOWN SAVE-AREA DEFINITION *

*

		.... 1.1.	PSAVELVL	MAXIMUM NUMBER OF LEVELS
		.... .1..	PSAVESZ	SIZE OF EACH LEVEL
1796	(704)	4		
1796	(704)	0	PSAVETBL	SAVE AREA
		..1. 1..1	PSAVEND	END OF SAVE AREA
1836	(72C)	2		
1836	(72C)	2	PSAVPT	CURRENT SAVE AREA INDEX

```
 DISPLMNT                         FIELD       DESCRIPTION:  CONTENTS,
 DEC   (HEX)       SIZE           NAME        MEANING/USE


1840    (730)        4
1840    (730)        4            PSAVTEMP    SAVES RWAA FOR PSAVE & PRETURN
1844    (734)        4            STEP        USED BY TRACE PROGRAM
1848    (738)        2
1848    (738)        2            PLINECNT    LINE COUNT
1850    (73A)        4            PROGID      PROGRAM IDENTIFICATION
1850    (73A)        4                        FROM FIRST TITLE STMNT

*************************************************************
*        C - P                                              *
*************************************************************
*        ICTL CONSTANTS                                     *

1854    (73E)        3            PICTL
1854    (73E)        1            PICTLST     START COLUMN
1855    (73F)        1            PICTLEND    END COLUMN
1856    (740)        1            PICTLCNT    CONTINUE COLUMN

                ..11 11..         ORG1        *


*************************************************************
*        E - P                                              *
*************************************************************


1857    (741)        8            NPTXT3      N/P TO WF3 TEXT


*************************************************************
*        G - S                                              *
*************************************************************

1865    (749)        3            PPUNCHPT    ADDRESS OF PUNCH ROUTINE

1868    (74C)        1            JSW0007     PROGRAM SWITCH

                1... ....         PNOSEQSW    1 INDICATES NO CARD SEQUENCING
                .1.. ....         PGFMSGSW    1 MEANS MESSAGE FROM GA OR FA
                                              ON FILE 3
*************************************************************
*        E - J                                              *
*************************************************************


1869    (74D)        2            OCSTMTNO    AT ENTRY OF JA CONTAINS NUMBER
                                              OF LAST STMT OUTPUT ON SOURCE
                                              FILE(WF3)  BY THE EDITOR

                .1.. 1.1.         ORG101      DELIMITS AREA THAT IS OVERLAID


*************************************************************
*        C - K                                              *
*************************************************************
```

```
 DISPLMNT                       FIELD        DESCRIPTION:  CONTENTS,
 DEC   (HEX)       SIZE         NAME         MEANING/USE


1871    (74F)        1          POVLSW       PROGRAM SWITCH

                 1... ....      PCSWOVL      OVERLAY SWITCHES
                 .1.. ....      PDSWOVL      *
                 ..1. ....      PISWOVL      *
                 ...1 ....      PJSWOVL      *
                 .... 1...      PKSWOVL      *
                 .1.. 1.11      ORG2         *


*****************************************************************
*        C - J                                                  *
*****************************************************************

1872    (750)        8          NPOCST       N/P TO OPEN CODE TEXT ON WF2

                 .1.1 ..11      ORG3         *


*****************************************************************
*        C - I                                                  *
*****************************************************************

1880    (758)        9          PSYSPARM
1880    (758)        1          PSYSPLEN     LENGTH OF FIELD
1881    (759)        8          PSYSPSTR     FIELD

1889    (761)        1          JSW0009      PROGRAM SWITCH

                 1... ....      SWCAOC       C/A IN OPEN CODE
                 .1.. ....      SWSM         SOURCE MACROS PRESENT
                 ..1. ....      SWMIOC       M-I:S IN OPEN CODE
                 ...1 ....      SWCAFT       C/A IN FIRST TITLE STMT

1890    (762)        2          PB1FISIZ     WF1 BUFSIZE IN F AND I
1892    (764)        2          PB12SIZ      WF1,WF2 BUFSIZ

                 .11. ...1      ORG4         *

*****************************************************************
*        C - F                                                  *
*****************************************************************

1894    (766)        2          PMNABSIZ     LENGTH OF MNA BLOCK
1896    (768)        3          PFINDPT      ADDRESS OF PFIND ROUTINE
1899    (76B)        3          PINPUTPT     ADDRESS OF PINPUT ROUTINE
1902    (76E)        2          SMTSIZE      SIZE OF SMT BLOCK
1904    (770)        2          PVSDSIZE     VSDSIZE

                 .11. 11.1      ORG5         *
```

```
DISPLMNT                     FIELD          DESCRIPTION:  CONTENTS,
DEC   (HEX)      SIZE        NAME           MEANING/USE


************************************************************
*        E - F                                             *
************************************************************


1906    (772)       6          NPSMT          N/P TO SMT START
               .111 ..11       ORG6           *


************************************************************
*        C - E                                             *
************************************************************


1912    (778)       8          NPEOTXT2       NP TO CA END OF TXT WF2

1920    (780)       1          JSW0010        PROGRAM SWITCH

               1... ....       SWREPRO        PROCESS NEXT AS REPROED
               .1.. ....       SWPROTO        PROTOTYPE EXPECTED
               ..1. ....       SWMACRO        PROCESSING MACRO DEF
               ...1 ....       SWOC           PROCESSING OPEN CODE
               .... 1...       SWCOPY         PROCESSING COPY CODE
               .... .1..       SWFLUSH        FLUSHUSHING AFTER END STMT


************************************************************
*        C - D                                             *
************************************************************


1921    (781)       1          CCCALL         TYPE OF CALL TO OVLY CC

               .111 11.1       ORG7           *


************************************************************
*        C                                                 *
************************************************************


1922    (782)       3          BLKADDR        ADDRESS OF MNA BUFFER
1925    (785)       2          CMNASIZE       MNA BLOCK SIZE

               1... .111       ORGSAVE1       SAVE LOCATION COUNTER
                                              OVERLAY AREA NOT USED ANY MORE
1857    (741)       1          PFETCHDF       PFETCH SWITCH FOR CD
```

```
DISPLMNT                          FIELD          DESCRIPTION:  CONTENTS,
DEC   (HEX)      SIZE             NAME           MEANING/USE

1858   (742)       1              JSW0011        PROGRAM SWITCH

               1... ....          SWCMARK        CARD HAS CONTINUE MARK
               .1.. ....          SWCC           LEGAL CONTINUE CARD
               ..1. ....          SWCILL         CONTINUE CARD IS EXCESSIVE
               ...1 ....          SWNOBLK        NONBLANKS BEFORE CONT COL
               .... 1...          SWCARD1        1ST GET IN IDENT
               .... .1..          SWMNA1         FIRST CALL TO CC
               .... ..1.          SWILICTL       ICTL NOW ILLEGAL
               .... ...1          SWCC2          2ND CONT CARD IN PROCESS


1859   (743)       1              JSW0012        PROGRAM SWITCH

               1... ....          SWQM           IN QUOTE MODE
               .1.. ....          SWAMP          POSSIBLE VARIABLE SYMBOL

1860   (744)       6              NPLASTW        WRITE NP FOR LAST BLOCK
1866   (74A)       2
1866   (74A)       2              NDXCNT         M-I INDEX COUNTER

               .1.. .11.          PCHECK1        ORGCHECK
                                                 RESTORE LOCATION COUNTER
1927   (787)           0                         1-ORG1-(PCHECK1-ORGSAVE1),(R0))
     THE PRECEDING INSTRUCTION IS FLAGGED IF THE OVERLAY ORG IS IN ERROR

1928   (788)       2
1928   (788)       2              CURBEG         CURRENT BEG COL
1930   (78A)       2              CURCNT         CURRENT CONTINUE COL

               1... .11.          ORGSAVE2       SAVE LOCATION COUNTER
                                                 *
1906   (772)       2
1906   (772)       2              CUREND         CURRENT END COL
1908   (774)       2              CONTEND        END COL OF 1ST CONT CARD
1910   (776)       2              ENDCNT         NO OF COL CUREND TO CURCNT
1912   (778)       4
1912   (778)       4              EOCARD1        END OF 1ST CARD STMT FIELD

               .111 .11.          PCHECK2

1916   (77C)           0                         ORG5-(PCHECK2-ORGSAVE2),(R0))
     THE PRECEDING INSTRUCTION IS FLAGGED IF THE OVERLAY ORG IS IN ERROR
                                                 RESTORE LOCATION COUNTER
1932   (78C)       4
1932   (78C)       4              BASESAVE       SAVE AREA FOR 1 BASEREG
1936   (790)       4              SCNEND         END OF STMT FIELD IN WORKAREA
1940   (794)       4              STREND         END OF STRING FIELDS IN WRKAREA
1944   (798)       4              SCNBEG         SAVE AREA FOR BEGIN OF SCAN
1948   (79C)       3              BLANKS         BLANKS FOR WORKING PURPOSE
1951   (79F)       264            INPWRKAR       3-CARD WORKAREA
2216   (8A8)       4
2216   (8A8)       4              STRAR          BEGIN OF STRING FIELDS
2220   (8AC)       80             CC1AREA        SAVE 1ST CONT CARD OF MACRO

                                      PHASE C ERROR STACK


               .... .1..          MAXERRNO       MAX ERRORS THAT IS LOGGED
               .... ....          NOSTRING       STRING NOT REQUIRED
```

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
2300	(8FC)	1	ERRST	ERROR STACK (LNG=MAXERRNO)
2304	(900)	2	ERRCNT	ERROR COUNTER
			ISEQ PARAMETERS	
2306	(902)	1	PISEQST	ISEQ START COL
2307	(903)	1	PISEQEND	ISEQ END COL
2308	(904)	39	PISEQAR	ISEQ FIELD AREA
2347	(92B)	1	JSW0013	PROGRAM SWITCH
		1...	SWISEQ	ISEQ CHECK REQUIRED
		.1..	SWTITLE1	1ST TITLE SWITCH
		..1.	SWISEQER	ON IF 1ST CARD HAS ISEQ ERROR
		...1	SWISEQE1	ON IF 1ST CONT CARD HAS ISEQ ERROR
		.... 1...	SWISEQE2	ON IF 2ND CONT CARD HAS ISEQ ERROR
			AREAS FOR COPY	
		11	MAXCNEST	MAX NEST LEVEL FOR COPY
2348	(92C)	1	MACSTAT	MACRO STATUS IN COPY BOOOK
		1111 .111	MACON	MACRO STM READ IN THE BOOK
		....	MACOFF	NO MACRO STMT READ IN THE BOOK
2349	(92D)	1	STATSTK	STACK FOR MACRO STATUS (LNG=MAXCNEST)
2352	(930)	1	JSW0014	PROGRAM SWITCH
		1...	SWMSEAR1	1ST SEARCH
		.1..	SWMOVFL	MNA OVERFLOW
		....	PIPT	INPUT FROM SYSIPT
2353	(931)	2	PCOPY	COPY NESTING DEPTH. IF ZERO,
2355	(933)	1	PRESCNT	SYSIPT RESIDUAL COUNT. IS
2360	(938)	2	CLINK	LINK ADDRESS FOR C

FIELD NAME	DISPLACEMENT	
	DECIMAL	(HEX)
ASSGNSW	1922	(782)
BEGOFIN	2044	(7FC)
BEGOFOUT	2040	(7F8)
BLKINCNO	1995	(7D3)
BOTTHALF	2035	(7F3)
*BUFADDR1	1694	(69E)
*BUFADDR2	1729	(6C1)
*BUFADDR3	1764	(6E4)
*BUFPT1	1691	(69B)
*BUFPT2	1726	(6BE)
*BUFPT3	1761	(6E1)
*BYTEOFFS	1953	(7A1)
COMMODSW	1926	(786)
CROSSNP	1961	(7A9)
CROSSW	1922	(782)
*CURESD	1992	(7C8)
CURNP	1967	(7AF)
CURSECT	1932	(78C)
CURSECTL	1931	(78B)
DBCORE	1951	(79F)
DBFULLSW	1926	(786)
DBVSDADR	2000	(7D0)
DLINK	2024	(7E8)
DSCOMSW	2031	(7EF)
EAWF2END	1920	(780)
ELINK	1976	(7B8)
ENDID	2011	(7DB)
ENDSW	1963	(7AB)
ENTRYCNT	2008	(7D8)
EQORGSW	1922	(782)
ERRCOUNT	2036	(7F4)
ERRINOPD	1926	(782)
ESDIDHI	1955	(743)
ESDIDLO	2013	(7DD)
*ESDPTR	1987	(7C4)
ESDTABSW	1963	(7AB)
FILE1NP	1880	(758)
FILE1NPR	1886	(75E)
FILE12EX	1904	(770)
FILE2NP	1892	(764)
FILE2NPR	1898	(76A)
FLINK	1960	(7A8)
*GAVAPT	1940	(794)
*GBVAPT	1943	(797)
*GCVAPT	1946	(79A)
GLINK	1913	(779)
HLINK	1949	(79D)
IATESTSW	1926	(786)
IFSAVE	1672	(688)
ILINK	1984	(760)
JLINK	1928	(788)
KLINK	2072	(818)
LBARADDR	1994	(76A)
LCLASIZ	1932	(78C)
LCLBSIZ	1935	(78F)
LCLCSIZ	1935	(792)
LITSW	1963	(7AB)
LLINK	2072	(818)
*LOCCNTHI	2028	(7EC)
*LOCCNTR	2025	(7E9)
LOCLATR	2021	(7E5)
LOCRATR	2023	(7E7)
*POINTER		

FIELD	DISPLACEMENT	
NAME	DECIMAL	(HEX)
LOCTYPE	2031	(7EF)
MOCROSW	1926	(786)
MFLAGS	1922	(782)
MIB	1922	(782)
*MIBADDR	1948	(79C)
MLEVEL	1951	(79D)
MNAM	1924	(784)
NLINK	1960	(7A8)
NPLITBEG	1913	(779)
NPSSDR1	1963	(7AB)
NPSSDWL	1970	(7B2)
NPTEMP	1660	(67C)
NPTEMPCC	1660	(67C)
NPTEMPH	1662	(67E)
NPTEMPR	1663	(67F)
NPTEMPTB	1664	(680)
NPVSD	1947	(79B)
NPVSDR1	1987	(7C3)
*NXTENTRY	2004	(7D4)
OLINK	1960	(7A8)
OVFLADDR	1997	(7CD)
PABENDC	1669	(685)
PALIGN	1659	(67B)
PASSGNSW	2035	(7F3)
*PBUFLEN1	1697	(6A1)
*PBUFLEN2	1732	(6C4)
*PBUFLEN3	1767	(6E7)
PB1FISIZ	1890	(762)
PB12SIZ	1892	(764)
PCONTSW	1904	(770)
PCSWOVL	1871	(74F)
PDECK	1659	(67B)
PDF	1659	(67B)
PDSWOVL	1871	(74F)
*PDTFADR1	1688	(698)
*PDTFADR2	1723	(6BB)
*PDTFADR3	1758	(6DE)
PEDECK	1659	(67B)
*PENDBUF1	1699	(6A3)
*PENDBUF2	1734	(6C6)
*PENDBUF3	1769	(6E9)
PENTDEF	1922	(782)
PEOBSW	1658	(67A)
*PEOFADR1	1706	(6AA)
*PEOFADR2	1741	(6CD)
*PEOFADR3	1776	(6F0)
*PFETCHDA	2008	(7D8)
*PFETCHDB	2012	(7DC)
*PFETCHIA	1956	(7A4)
*PFETCHIB	1960	(7A8)
*PFETCHIC	1964	(7A7)
PFILE1	1688	(698)
PFILE2	1723	(6BB)
PFILE3	1758	(6DE)
*PFINDPT	1896	(768)
PFRSTASG	1922	(782)
PFRSTOV	2035	(7F3)
PGBLASIZ	1945	(799)
PGBLBSIZ	1948	(79C)
PGBLCSIZ	1951	(79F)
*POINTER		

FIELD	DISPLACEMENT	
NAME	DECIMAL	(HEX)
PGBLSIZ	1945	(799)
PGENSW	1926	(786)
PGEN5SW	1926	(786)
PGFMSGSW	1863	(747)
*PHICORE	1666	(682)
PIBSW	1926	(786)
PICSW	1926	(786)
*PICTLCNT	1856	(740)
*PICTLEND	1855	(73F)
*PICTLST	1854	(73E)
PIERCNT	1974	(7B6)
PIERSTK	1976	(7B8)
PINEOFSW	1650	(67B)
*PITPUTPT	1899	(76B)
PISWOVL	1871	(74F)
PJSWOVL	1871	(74F)
PKSWOVL	1871	(74F)
PLASTSUB	2035	(7F3)
PLBEOFSW	1658	(67A)
PLINECNT	1848	(738)
PLINENUM	1923	(783)
PLINK	1659	(67B)
PLIST	1659	(67B)
PLITBLK	1909	(775)
PLITLEN	1911	(777)
PLRECLN	1670	(686)
*PMAVBSIZ	1941	(795)
*PMAVNO	1943	(797)
PMAVNP	1935	(78F)
PMAXBSIZ	1787	(6FB)
*PMIBLEN	1956	(7A4)
PNEXTNP1	1717	(6B5)
PNEXTNP2	1752	(6D8)
PNEXTNP3	1787	(6FB)
PNOBKSW	1658	(67A)
PNOSEQSW	1863	(747)
PNPMAC1	1912	(778)
PNPOCGV	1927	(787)
PNPOINT1	1709	(6AD)
PNPOINT2	1744	(6D0)
PNPOINT3	1779	(6F3)
PNWTRKSW	1658	(67A)
POPSW	1658	(67A)
PPAGENO	1925	(785)
PREFCNT	1936	(790)
PRLD	1658	(67A)
PROGID	1850	(73A)
PSAVETBL	1796	(704)
PSAVPT	1836	(72C)
PSAVTEMP	1840	(730)
PSPILLA	1968	(7B0)
PSTCNAM	1912	(778)
PSTMCSEQ	1915	(77B)
PSXREF	1658	(67A)
*PSYMTABL	2032	(7F0)
PSYSNDX	1927	(787)
PSYSPLEN	1880	(758)
PSYSPSTR	1881	(759)
PUNDEFSW	1926	(786)
PVSDSIZE	1904	(770)
*PWAADDR1	1702	(6A6)
*POINTER		

FIELD	DISPLACEMENT	
NAME	DECIMAL	(HEX)
*PWAADDR2	1737	(6C9)
*PWAADDR3	1772	(6EC)
PXREF	1659	(67B)
P3705SW	1658	(67A)
*SAVADDR	2048	(800)
SAVAR	2051	(803)
SAVESDNP	1967	(7B5)
SAVREG1	2060	(80C)
SAVREG2	2064	(810)
SECTSW	1963	(7AB)
SELFLAGL	1672	(688)
SELFLAGS	1672	(688)
SMTSIZE	1902	(76E)
SSDADDR	1953	(7A1)
SSDBLK1	1969	(7B1)
*SSDEND	2019	(7E3)
SSDINFO	1958	(7A6)
SSDNP	1941	(795)
SSDSIZE	1961	(7A9)
*STABEND	2000	(7D0)
*STARTLOC	1952	(7A0)
STARTSW	1922	(782)
STEP	1844	(734)
STMTNR	1921	(781)
SWATTR	1922	(782)
SWCAFT	1889	(761)
SWCAOC	1889	(761)
SWCOPY	1920	(780)
SWDS	1922	(782)
SWFLUSH	1920	(780)
SWGBLX	1922	(782)
SWINM	1922	(782)
SWKT	1922	(782)
SWLA	1922	(782)
SWMACRO	1920	(780)
SWMIOC	1889	(761)
SWNOED	1922	(782)
SWNOGEN	1922	(782)
SWNOSTOR	1922	(782)
SWOC	1920	(780)
SWPROTO	1920	(780)
SWREPRO	1920	(780)
SWSM	1889	(761)
SWSTART	1963	(7AB)
SW2	1963	(7AB)
*SYMADDR	1996	(7CC)
VSADDR	1982	(7BE)
VSDBLK1	1993	(7C9)
*VSDEND	2016	(7E0)
VSDINFO	1982	(7BE)
VSDSIZE	1985	(7C1)
*XRAREND	1982	(7BE)
*XREFADDR	2016	(7E0)
XREFLEN	2019	(7E3)
XREFPARM	2013	(7DF)
*XREFPTR	1979	(7BB)

(This page intentionally left blank.)

(This page intentionally left blank.)

(This page intentionally left blank.)

DATA AREA: **PCSR**

SIZE: 7

CREATED BY:
UPDATED BY: IPKCA,IPKCB,IPKCC,IPKCD,IPKDA,IPKDB,IPKEA,IPKGA, IPKFA,IPKHA,IPKIA,IPKIC,IPKJA,IPKKA,IPKLA,IPKPA

FUNCTION: Description of a compressed source record.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	6	PCSRHEAD	2 4
0	(0)	2	PCSRLEN	**************
2	(2)	4	PCSRIOP	* LEN * IOP *
				**************
2	(2)	1	PCSROP0	PROGRAM SWITCH
		1...	PCSRMCOP	
		.1..	PCSRLIT1	THREE IOP BYTE0 FLAGS
		..1.	PCSRLIT2	
		...1	PCSRLIT3	
		.... 1...	SWSFILE	SOURCE FILE DESTINATION
		.... .1..	SWEFILE	EDITED FILE DESTINATION
		.... ..1.	PNOSTNO	
		.1..	PCSRCOM	COMMENTS BEFORE MACRO TEXT
3	(3)	1	PCSROP	IOP BYTE1 MCOP
4	(4)	1	PCSROPX	IOP BYTE2 OP EXTENSIONS
5	(5)	1	PCSROP3	PROGRAM SWITCH
		1...	PCSRDEAD	IOP BYTE3 FLAGS,
6	(6)	1	PCSRFLGA	PROGRAM SWITCH
		1...	PCASRC	
		.1..	PCAEDTXT	
		..1.	PGENSTMT	
		...1	PFROMLIB	
		.... 1...	PSMACDEF	
		.... .1..	PSBSTOPD	
		.... ..1.	PSBSTOP	
		1	PSBSTNAM	
7	(7)	1	PCSRSTR1	

SOURCE CODE STRINGS

THEY ALL LOOK LIKE THIS

```
* 1 BYTE * 1 BYTE *LENGTH-1*
******************************
* LENGTH * COLUMN * SOURCE *
******************************
```

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
PCAEDTXT	6	(6)
PCASRC	6	(6)
PCSRDEAD	5	(5)
PCSRFLGA	6	(6)
PCSRHEAD	0	(0)
PCSRIOP	2	(2)
PCSRLEN	0	(0)
PCSRLIT1	2	(2)
PCSRLIT2	2	(2)
PCSRLIT3	2	(2)
PCSRMCOP	2	(2)
PCSROP	3	(3)
PCSROPX	4	(4)
PCSROP0	2	(2)
PCSROP3	5	(5)
PCSRSTR1	7	(7)
PFROMLIB	6	(6)
PGENSTMT	6	(6)
PNOSTNO	2	(2)
PSBSTNAM	6	(6)
PSBSTOP	6	(6)
PSBSTOPD	6	(6)
PSMACDEF	6	(6)
SWEFILE	2	(2)
SWSFILE	2	(2)

*POINTER

DATA AREA: **PDCEDIT**

SIZE: 10

CREATED BY:
UPDATED BY: } IPKJA,IPKKA,IPKLA,IPKNA,IPKOA,IPKPA

FUNCTION: Decsription of the operand part of the edited text for DC, DS, and literal DC statements.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	3	PDUPEXP	DUPLICATION FACTOR
3	(3)	1	PNOCONST	NUMBER OF CONSTANTS
4	(4)	1	PDCTYPE	TYPE BYTE
		....	CTYPE	CHARACTER CONSTANT
		1	XTYPE	HEXADECIMAL CONSTANT
		.... ..1.	BTYPE	BINARY CONSTANT
		11	PTYPE	PACKED DECIMAL CONSTANT
		.... .1..	ZTYPE	ZONED DECIMAL CONSTANT
		.... .1.1	FTYPE	FIXED POINT FULLWORD
		.... .11.	HTYPE	FIXED POINT HALFWORD
		.... .111	LTYPE	FLOATING POINT LONG CONSTANT
		.... 1...	DTYPE	FLOATING POINT DOUBLE WORD
		.... 1..1	ETYPE	FLOATING POINT FULL WORD
		.... 1.1.	ATYPE	A-TYPE ADDRESS CONSTANT
		.... 1.11	VTYPE	V-TYPE ADDRESS CONSTANT
		.... 11..	YTYPE	Y-TYPE ADDRESS CONSTANT
		.... 11.1	STYPE	S-TYPE ADDRESS CONSTANT
5	(5)	1	PLENFLAG	
		..11	BITLEN	
5	(5)	1	VBITLEN	
		.1..	EXPLEN	
5	(5)	1	VEXPLEN	
		.1.. ...1	IMPLEN	
5	(5)	1	VIMPLEN	
5	(5)	1		
6	(6)	1	PDCFLAG	PROGRAM SWITCH
		1...	PDUPCONT	ADDR CONST CONT
		.1..	PTRUNRHT	TRUNCATE RIGHT I
7	(7)	1	PBITALGN	NUMBER OF BITS TO SHIFT OR TRUNCATE
8	(8)	1		NOT USED
9	(9)	1	PMODIFS	START OF MODIFIER FIELDS *
0	(0)	1	PMODFLAG	
		.1.. ..1.	SCALE	

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	1	VSCALE	
		.1.. ..11	EXPON	
0	(0)	1	VEXPON	
		.1.. .1..	DCEXP	
0	(0)	1	VDCEXP	
		.1.. .1.1	DCVAL	
0	(0)	1	VDCVAL	
0	(0)	1		
1	(1)	3	PMODEXP	MODIFIER EXPRESSION
4	(4)	1	PNXTMOD	
				*
1	(1)	1	PDCLEN	LENGTH OF CONSTANT FIELD
2	(2)	1	PDCFLD	BEGINNING OF CONSTANT FIELD

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
ATYPE	4	(4)
BITLEN	5	(5)
BTYPE	4	(4)
CTYPE	4	(4)
DCEXP	0	(0)
DCVAL	0	(0)
DTYPE	4	(4)
ETYPE	4	(4)
EXPLEN	5	(5)
EXPON	0	(0)
FTYPE	4	(4)
HTYPE	4	(4)
IMPLEN	5	(5)
LTYPE	4	(4)
PBITALGN	7	(7)
PDCFLAG	6	(6)
PDCFLD	2	(2)
PDCLEN	1	(1)
PDCTYPE	4	(4)
PDUPCONT	6	(6)
PDUPEXP	0	(0)
PLENFLAG	5	(5)
PMODEXP	1	(1)
PMODFLAG	0	(0)
PMODIFS	9	(9)
PNOCONST	3	(3)
PNXTMOD	4	(4)
PTRUNRHT	6	(6)
PTYPE	4	(4)
SCALE	0	(0)
STYPE	4	(4)
VBITLEN	5	(5)
VDCEXP	0	(0)
VDCVAL	0	(0)

*POINTER

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
VEXPLEN	5	(5)
VEXPON	0	(0)
VIMPLEN	5	(5)
VSCALE	0	(0)
VTYPE	4	(4)
XTYPE	4	(4)
YTYPE	4	(4)
ZTYPE	4	(4)

* POINTER

(This page intentionally left blank.)

DATA AREA: **PERR**

SIZE: Variable (depending upon different error text)

CREATED BY:
UPDATED BY: } IPKPA

FUNCTION: Description of an error record.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	1	PERRHD	RECORD HEADER (LNG=PERLEN-PETHEAD)
0	(0)	2	PERRLEN	RECORD LENGTH
2	(2)	1	PERRIOP	IOP (LNG=PETIOP)
2	(2)	1	PERRIOP0	FLAGS
3	(3)	1	PERRIOP1	ERROR ID
4	(4)	1	PERRIOP2	ERROR NUMBER
5	(5)	1		NOT USED
6	(6)	2	PERRSTNR	STMNT NO. INSERTED BY OUTPUT
8	(8)	1	PERRSTRL	LENGTH OF TEXT
9	(9)	1	PERREXPT	TEXT EXCERPT

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
PERREXPT	9	(9)
PERRHD	0	(0)
PERRIOP	2	(2)
PERRIOP0	2	(2)
PERRIOP1	3	(3)
PERRIOP2	4	(4)
PERRLEN	0	(0)
PERRSTNR	6	(6)
PERRSTRL	8	(8)

*POINTER

DATA AREA: **PETFLDS**

SIZE: Variable

CREATED BY: } IPKCA,IPKCC,IPKCD,IPKDA,IPKDB,IPKEA,IPKGA,IPKFA,
IPKIA,IPKJA,IPKKA,IPKLA,IPKNA,IPKOA,IPKPA,IPKSA,
UPDATED BY: } IPKSB

FUNCTION: Description of the variable fields of edited text records.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	1	PNAMLNG	THE NAME FIELD LENGTH
1	(1)	1	PNAME	NAME
				THE SYMBOL WORK BUCKETS BEFORE SUBSTITUTION:
0	(0)	1	PAFLAG2	PROGRAM SWITCH
		1...	DONESYM	VALUE HAS BEEN SUBSTITUTED
		.1..	ENTVALUE	VALUE IS IN PENTVAL
		..1.	ERDSCOM	ENTRY DISALLOWED (FOR EQU)
		1111	PSYMFLAG	***********************
		.... .111	PSYMLEN	* FLAG * LEN * SYMBOL *
				***********************
1	(1)	8	PSYMBOL	
9	(9)	5	PENTVAL	BUCKET EXTENSION FOR ENTRY VALUE
9	(9)	9	PNXTBKT	
				AFTER SUBSTITUTION: 1 2 2 3 2
0	(0)	1	PSFLAG2	********************************
1	(1)	2	PSLENATR	* FLAG * LA * RA * VALUE * *
3	(3)	2	PSRELATR	********************************
5	(5)	3	PSVALUE	
				OPERAND FIELDS LOOK LIKE:
0	(0)	1	POPFLAG	OPERAND FLAG
1	(1)	1	PEXPFLAG	PROGRAM SWITCH
		.... .1..	NEXPF1	EXPRESSION FLAG
		.... ..1.	NEXPF2	
		1	NEXPF3	

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
2	(2)	1	POLSTR	POLISH STRING
				AFTER EVALUATE
				VALUES WILL BE INSERTED
2	(2)	2	PLA	LENGTH ATTRIBUTE
4	(4)	1	PRA	NUMBER OF RELOCATION FACTORS
5	(5)	4	PEXPVAL	VALUE OF EXPRESSION
9	(9)	1	PSIGN	SIGN OF RELOCATION FACTOR
10	(A)	1	PRATT	RELOCATION ATTRIBUTE
				USED BY OUTPUT
0	(0)	1	PAFLG	FIELD TYPE FLAG
1	(1)	3	PAFLD	ADDR FIELD
4	(4)	1	PANXT	NEXT FLAG
				BEGINNING OF OBJ CODE FIELD
1	(1)	2	PCODE	OBJ CODE, TWO BYTES
3	(3)	2	PCNXT	NEXT TWO BYTES
0	(0)	1	THISELEM	CURRENT CHARACTER
		1	NXTELEM	NEXT CHARACTER
		.... .1.1	ERROPLEN	LENGTH OF ERROR OPERAND

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
DONESYM	0	(0)
ENTVALUE	0	(0)
ERDSCOM	0	(0)
NEXPF1	1	(1)
NEXPF2	1	(1)
NEXPF3	1	(1)
PAFLAG2	0	(0)
*PAFLD	1	(1)
PAFLG	0	(0)
PANXT	4	(4)
PCNXT	3	(3)
PCODE	1	(1)
PENTVAL	9	(9)
PEXPFLAG	1	(1)
PEXPVAL	5	(5)
PLA	2	(2)
PNAME	1	(1)
PNAMLNG	0	(0)
PNXTBKT	9	(9)
POLSTR	2	(2)
POPFLAG	0	(0)
PRA	4	(4)
PRATT	10	(A)
PSFLAG2	0	(0)
PSIGN	9	(9)
PSLENATR	1	(1)
PSRELATR	3	(3)
*PSVALUE	4	(4)
PSYMBOL	1	(1)
PSYMFLAG	0	(0)
PSYMLEN	0	(0)
SREG	0	(0)
THISELEM	0	(0)

*POINTER

DATA AREA: **PETR**

SIZE: Variable

CREATED BY:
UPDATED BY: IPKCA,IPKCC,IPKCD,IPKDA,IPKDB,IPKEA,IPKGA,IPKFA,IPKIA,
IPKJA,IPKKA,IPKLA,IPKNA,IPKOA,IPKPA,IPKSA,IPKSB

FUNCTION: Decription of an edited text record

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	6	PETHEAD	2 4
0	(0)	2	PETLEN	*************
				* LEN * IOP *
2	(2)	4	PETIOP	*************
2	(2)	1	PETOP0	PROGRAM SWITCH
		1...	PETMCOP	IOP BYTE0, MACH.OP FLAG
		.1..	PER2STR	2 STRINGS IN ERROR RCD FLAG
		..1.	PERNOQ	NO QUOTES AROUND ERROR STRING
		.... 1...	PDCSTYPE	S-TYPE CONST,CODE ALREADY BUILT
		.... .1..	PDCCONT	DC CONTINUATION RECORD FOLLOWS
		.... ..1.	PDUPSYM	
		1	PDEFENT	

PDCSTYPE ALSO USED TO INDICATE OVERFLOW POINT IN SUBSTITUTION

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
		.111	PALGNBIT	NUMBER OF ALGN BYTES
3	(3)	1	PETOP	IOP BYTE1 MAGHINEOP
4	(4)	1	PETEPX	IOP BYTE2 OP EXTENSIONS
				ALSO USED AS
4	(4)	1	POPNUMB	OPERAND NUMBER IN A DC/DS
5	(5)	1	PETOP3	PROGRAM SWITCH
		1...	PETDEAD	IOP BYTE3 FLAGS
		.111	PINSTRLN	INSTRUCTION LENGTH
		.... .111	PCBTYPE	INSTRUCTION TYPE
6	(6)	2	PETSTNO	STATEMENT NUMBER
8	(8)	2	PLENATTR	LENGTH ATTRIBUTE
10	(A)	2	PLRELATR	2 3 1
12	(C)	3	PLOCCNTR	****************
15	(F)	1	PSYMNO	* RA * LC * S# *
16	(10)	1	PNAMFLD	****************
				AN ERROR RECORD
				ONLY CONSISTS
6	(6)	1	PERLNG	OF 6 BYTE HEADER AND A
7	(7)	1	PERSRC	VARIABLE STRING FIELD

A STMT NUMBER IS INSERTED BY THE OUTPUT PHASE.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
				USED BY OUTPUT
6	(6)	2	PERRSTNO	STMNT NO. INSERTED BY OUTPUT
8	(8)	1	PERRSTR	LENGTH OF TEXT
9	(9)	1	PERREXC	TEXT EXCERPT

FIELD	DISPLACEMENT	
NAME	DECIMAL	(HEX)
PALGNBIT	2	(2)
PCBTYPE	5	(5)
PDCCONT	2	(2)
PDCSTYPE	2	(2)
PDEFENT	2	(2)
PDUPSYM	2	(2)
PERLNG	6	(6)
PERNOQ	2	(2)
PERREXC	9	(9)
PERRSTNO	6	(6)
PERRSTR	8	(8)
PERSRC	7	(7)
PER2STR	2	(2)
PETDEAD	5	(5)
PETEPX	4	(4)
PETHEAD	0	(0)
PETIOP	2	(2)
PETLEN	0	(0)
PETMCOP	2	(2)
PETOP	3	(3)
PETOP0	2	(2)
PETOP3	5	(5)
PETSTNO	6	(6)
PINSTRLN	5	(5)
PLENATTR	8	(8)
*PLOCCNTR	11	(B)
PLRELATR	10	(A)
PNAMFLD	16	(10)
POPNUMB	4	(4)
PSYMNO	15	(F)

*POINTER

DATA AREA: **PFCB**

SIZE: 35

CREATED BY:
UPDATED BY: } All modules except: IPKAB-AJ,IPKCB,IPKTA

FUNCTION: Description of a file control block.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	3	PDTFADDR	ADDRESS OF DTF
3	(3)	3	BUFPT	POINTER TO RECORD IN BUFFER
6	(6)	3	BUFADDR	ADDRESS OF BUFFER
9	(9)	2	PRECLEN	MAX RECORD LENGTH
11	(B)	3	ENDBUF	ADDRESS OF LAST BYTE OF BUFFER
14	(E)	3	PWAADDR	ADDRESS OF WORKAREA
17	(11)	1	PFCBSW	PROGRAM SWITCH
		1...	OPENSW	IF 1, FILE OPEN
		.1..	READSW	IF 1, READ, IF 0, WRITE
		..1.	UPDSW	IF 1, WRITE UPDATE
		...1	BUF2SW	IF 1, TWO BUFFERS
		.... 1...	UPD2SW	IF 1, UPDATE MODE
		.... .1..	FIRSTSW	IF 1, FIRST I/O OPERATION
		.... ..1.	PFCBSW1	*
		1	PFCBSW2	*
18	(12)	3	PEOF	EOF ADDRESS
21	(15)	8	PNOTEPNT	NOTE POINT VALUES
21	(15)	6	PNPRW	
21	(15)	4	PCCHR	CYLINDER, HEAD, RECORD
25	(19)	2	PTRKBAL	TRACK BALANCE
27	(1B)	2	PNPOFFS	RECORD OFFSET FROM BUFFER START
29	(1D)	1	PNEXTNP	N/P VALUE FOR NEXT BLOCK (LNG=PNPRW)

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
*BUFADDR	6	(6)
*BUFPT	3	(3)
BUF2SW	17	(11)
*ENDBUF	11	(B)
FIRSTSW	17	(11)
OPENSW	17	(11)
PCCHR	21	(15)
*PDTFADDR	0	(0)
*PEOF	18	(12)
PFCBSW	17	(11)
PFCBSW1	17	(11)
PFCBSW2	17	(11)

*POINTER

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
PNEXTNP	29	(1D)
PNOTEPNT	21	(15)
PNPOFFS	27	(1B)
PNPRW	21	(15)
PRECLEN	9	(9)
PTRKBAL	25	(19)
*PWAADDR	14	(E)
READSW	17	(11)
UPDSW	17	(11)
UPD2SW	17	(11)

*POINTER

DATA AREA: **PGVHEAD**

SIZE: 9

CREATED BY:
UPDATED BY: } IPKFA,IPKIA

FUNCTION: Description of the global vector header.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	2	PGVHLEN	RECORD LENGTH
2	(2)	1	PGVHIOP	GLOBAL VECTOR IOP (LNG=PCSRIOP)
6	(6)	1	JSW0028	PROGRAM SWITCH
		1...	SWGVLST	LAST RECORD SWITCH
7	(7)	2	PGVENT1	GLOBAL VECTOR ITEM

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
PGVENT1	7	(7)
PGVHIOP	2	(2)
PGVHLEN	0	(0)
SWGVLST	6	(6)

*POINTER

DATA AREA: **PHYR**

SIZE: 11-18

CREATED BY:
UPDATED BY: IPKKA,IPKLA

FUNCTION: Description of an entry in the symbol table.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	2	HASHPTR	HASH PINTER
2	(2)	1	SYMFLAGS	PROGRAM SWITCH
		1...	ENTESD	FLAGS: ENTRY EXITS IN ESD TAB
		.1..	LXFLAG	ENTRY SYM WITHOUT DEF
		..1.	ENTRYER	ENTRY NOT ALLOWED
		...1	LDNAME	ENTRY SYMBOL
3	(3)	2	SYMLATTR	LENGTH ATTRIBUTE OF SYMBOL
5	(5)	2	SYMESDID	RELOC ATTRIBUTE OF SYMBOL
7	(7)	3	SYMVALUE	VALUE OF SYMBOL
10	(A)	1	SYMLENG	MOVE LENGTH OF SYMBOL SOURCE
11	(B)	8	SYMSRC	SYMBOL SOURCE 1-8 CHARACTERS
19	(13)	1	SYMNXT	BEGINNING OF NEXT ENTRY

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
ENTESD	2	(2)
ENTRYER	2	(2)
HASHPTR	0	(0)
LDNAME	2	(2)
LXFLAG	2	(2)
SYMBREG	2	(2)
SYMESDID	5	(5)
SYMFLAGS	2	(2)
SYMLATTR	3	(3)
SYMLENG	10	(A)
SYMNXT	19	(13)
SYMSRC	11	(B)
*SYMVALUE	7	(7)

*POINTER

DATA AREA: **PSTRINGS**

SIZE: Variable

CREATED BY: } IPKCA,IPKCB,IPKCC,IPKCD,IPKDA,IPKDB,IPKEA,IPKGA,
UPDATED BY: } IPKFA,IPKHA,IPKIA,IPKJA,IPKKA,IPKLA,IPKPA

FUNCTION: Description of the different fields in source records.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	1	PNAMLEN	LENGTH OF NAME
1	(1)	1	PNAMCOL	START COLUMN OF NAME
2	(2)	1	PNAMSRC	NAME
				OP-CODE
0	(0)	1	POPLEN	LENGTH OF OP-CODE
1	(1)	1	POPCOL	START COLUMN OF OP-CODE
2	(2)	1	POPSRC	OP-CODE
				OPERAND
0	(0)	1	POPDLEN	LENGTH OF OPERAND
1	(1)	1	POPDCOL	START COLUMN OF OPERAND
2	(2)	1	POPDSRC	OPERAND
				STRING
0	(0)	1	PSTRLEN	STRING LENGTH
1	(1)	1	PSTRCOL	START COLUMN OF STRING
2	(2)	1	PSTRSRC	STRING
				FIELD (ANY OF ABOVE TYPES)
0	(0)	1	PFLDLEN	FIELD LENGTH
1	(1)	1	PFLDCOL	START COLUMN OF FIELD
2	(2)	1	PFLDSRC	FIELD
				*
		....	THISCHAR	EQUATES FOR
		1	NXTCHAR	SCANNING

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
PFLDCOL	1	(1)
PFLDLEN	0	(0)
PFLDSRC	2	(2)
PNAMCOL	1	(1)
PNAMLEN	0	(0)
PNAMSRC	2	(2)
POPCOL	1	(1)
POPDCOL	1	(1)
POPDLEN	0	(0)
POPDSRC	2	(2)
POPLEN	0	(0)
POPSRC	2	(2)
PSTRCOL	1	(1)
PSTRLEN	0	(0)
PSTRSRC	2	(2)

*POINTER

DATA AREA: **RLDENTRY**

SIZE: 6

CREATED BY:
UPDATED BY: IPKOA, IPKQA

FUNCTION: Description of an entry in the relocation dictionary table.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DECSRIPTION: CONTENTS, MEANING/USE
0	(0)	2	RLADID	ID OF CONSTANT
2	(2)	2	RLREFID	ID OF REFERENCE
4	(4)	1	RLFLAG	PROGRAM SWITCH
		1...	ACONSW	A CONSTANT
		.1..	YCONSW	Y CONSTANT
		..1.	CCWSW	CCW CONSTANT
		...1	VCONSW	V CONSTANT
		.... 1...	LEN1SW	BIT 4 USED AS LENGTH
		.... .1..	LEN2SW	BIT 5 USED AS LENGTH
		.... ..1.	SIGNSW	0-POS. REL. FAC. 1-NEG. REL. FAC.
5	(5)	3	RLADDR	RELOCATION ADDRESS
8	(8)	1	RLDNXT	NEXT ENTRY

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
ACONSW	4	(4)
CCWSW	4	(4)
LEN1SW	4	(4)
LEN2SW	4	(4)
* RLADDR	5	(5)
RLADID	0	(0)
RLDNXT	8	(8)
RLFLAG	4	(4)
RLREFID	2	(2)
SIGNSW	4	(4)
VCONSW	4	(4)
YCONSW	4	(4)

* POINTER

DATA AREA: **SMTENT**

SIZE: 10

CREATED BY: }
UPDATED BY: } IPKEA,IPKGA,IPKFA

FUNCTION: Description of the source macro table.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANINg/USE
0	(0)	1	SWSMT	PROGRAM SWITCH
		1...	SWSMTATR	ATTRIBUTES
		.1..	SWSMTINM	INNER MACROS
		..1.	SWSMTGBL	GLOBAL VARIABLES
		...1	SWSMTNST	DONT STORE
		.... 1...	SWSMTNED	DONT EDIT
		.... .1..	SWSMTNGN	DONT GENERATE
1	(1)	1	SMTNP	N/P TO MACRO HEADER (LNG=PNOTEPNT)
9	(9)	1	SMTLEN	LENGTH OF MACRO NAME
10	(A)	8	SMTNAME	MACRO NAME
		.... .1.1	SMTENTL	
				OR INSTEAD OF LENGTH BYTE
9	(9)	1	SMTEFLG	END FLAG
		1111 .111	ESMT	END OF TABLE FLAG
		1111 .1.1	ESMTB	END OF BLOCK FLAG

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
ESMT	9	(9)
ESMTB	9	(9)
SMTEFLG	9	(9)
SMTLEN	9	(9)
SMTNAME	10	(A)
SMTNP	1	(1)
SWSMT	0	(0)
SWSMTATR	0	(0)
SWSMTGBL	0	(0)
SWSMTINM	0	(0)
SWSMTNED	0	(0)
SWSMTNGN	0	(0)
SWSMTNST	0	(0)

*POINTER

DATA AREA: **SSD**

SIZE: 4

CREATED BY:
UPDATED BY: } IPKDA,IPKDB,IPKEA,IPKGA

FUNCTION: Description of the sequence symbol dictionary.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	3	SSDOFFS	OFFSET IN EDITED TEXT
3	(3)	1	SSDSYML	LENGTH OF SYMBOL
4	(4)	1	SSDSYM	SYMBOL
				OR INSTEAD OF LENGTH BYTE
3	(3)	1	SSDFLAG	EO BLOCK/EO DICTIONARY FLAG
		1111 .111	ESSD	END OF DICTIONARY
		1111 .1.1	ESSDB	END OF BLOCK
		.... .111	SSDMAXL	MAX ENTRY LENGTH

VSD AND SSD INFO BLOCKS

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
ESSD	3	(3)
ESSDB	3	(3)
SSDFLAG	3	(3)
SSDOFFS	0	(0)
SSDSYM	4	(4)
SSDSYML	3	(3)

*POINTER

DATA AREA: **VSD**

SIZE: 4

CREATED BY: } IPKDA,IPKDB,IPKGA
UPDATED BY: }

FUNCTION: Description of the variable symbol dictionary.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	1	VSDTYPE	TYPE
1	(1)	2	VSDNDX	INDEX NUMBER
3	(3)	1	VSDSYML	LENGTH OF SYMBOL
4	(4)	1	VSDSYM	SYMBOL
4	(4)	2	VSDDIM	MAX DIM,ONLY SUBSCRIPTEDS OR INSTEAD OF LENGTH BYTE
3	(3)	1	VSDFLAG	END SEGMENT/END DICTIONARY
		1111 .111	EVSD	END OF DICTIONARY
		1111 .1.1	EVSDB	END OF SEGMENT

SSD ENTRY FORMAT

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
EVSD	3	(3)
EVSDB	3	(3)
VSDDIM	4	(4)
VSDFLAG	3	(3)
VSDNDX	1	(1)
VSDSYM	4	(4)
VSDSYML	3	(3)
VSDTYPE	0	(0)

*POINTER

DATA AREA: **WORKDTF**

SIZE: 152

CREATED BY:
UPDATED BY: IPKAA,IPKBA

FUNCTION: Description of the DTFSD.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	2		RESIDUAL COUNT
2	(2)	1	JSW0049	PROGRAM SWITCH
		1...	WIOCOMPL	THE COMPLETION BI
3	(3)	1	JSW0050	PROGRAM SWITCH
		1...	W1	I/O ERROR BITS
		...1	W2	
		.... ..1.	W3	
4	(4)	1	WCSW	PROGRAM SWITCH
		1	WEOF	CSW STATUS BYTE 1
5	(5)	11		CSWSTAT BYTE 2,REST OF CCB
16	(10)	4	WMODAD	ADDR OF LOGIC MODULE
20	(14)	9		DTF TYPE ETC
29	(1D)	1	WDEVTYP	DTF DEVICE TYP-INIT BY BA
30	(1E)	2	WPCTY	DRMAINING BYTES ON TRACK
32	(20)	6		-NOT USED BY IO ROUT
38	(26)	1	WLOWHEAD	LOWER HEAD LIMIT
39	(27)	1	WUPHEAD	UPPER HEAD LIMIT
40	(28)	2	WRECLEN	RECORD LENGTH
42	(2A)	4	WILOWEXT	INIT EXTENT LOWER LIMIT
46	(2E)	4	WCLOWEXT	CURRENT EXTENT LOWER LIMIT
50	(32)	4	WCHIEXT	EXTENT HIGH LIMIT
50	(32)	4	WEXTLIM	EXTENT HIGH LIMIT
54	(36)	2		SEEK-SEARCH BUCKET -BB
56	(38)	5	WCCHHR	CCHHR PART,WHICH IS
56	(38)	4	WCCHH	SEEK-SEARCH BUCKET
56	(38)	2	WCC	CYLINDER
56	(38)	4	WCURBLK	CURRENT POSITION (FBA BLOCK NO)
58	(3A)	1		PART OF HEAD NUMBER,ALWAYS 0
59	(3B)	1	WH	HEAD , FOLLOWED BY
60	(3C)	1	WREC	RECORD
			SWITCH BYTE	TO INDICATE
61	(3D)	1	JSW0052	PROGRAM SWITCH
		1...	WFRST	FIRST REC ON TRACK WRITTEN
		.... 1...	WTRKLM	TRACK LIMIT REACHED AT READ
		1	WPNT	POINT

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
62	(3E)	2	WMPCTY	TRACK CAPACITY CONSTANT
64	(40)	8		RESERVED SPACE FOR RPS
			CHANNEL PROGRAM	
72	(48)	6		SEEK-SEARCH-TIC CCW
96	(60)	16	CCWS	2 CCWS FOR READ,WRITE OR WRUPD
96	(60)	2	CCWS1	1
		.... .11.	RDAT	
96	(60)	1	VRDAT	
		.... .1.1	WDAT	
96	(60)	1	VWDAT	
		...1 .11.	WCKD	
96	(60)	1	VWCKD	
96	(60)	1		
97	(61)	7		REST OF 1ST CCW
104	(68)	2	CCWS2	2
				THE ASSEMBLED CCW:S IN DTF:S HAVE BEEN ADJUSTED BY INIT TO- CCW:S TO WRITE
112	(70)	2	WCCW	WRITE CCW 1
120	(78)	2	WCCW2	WRITE CCW 2
120	(78)	1		COMMAND (0)
121	(79)	3	WDATAD	DATA ADDRESS
124	(7C)	2		CHAIN BYTE AND
126	(7E)	2	WDATLEN	RECORD LENGTH CCW:S TO READ OR UPDATE WRITE
128	(80)	2	RCCW	
		.... .11.	RDA	
128	(80)	1	VRDA	
		.... .1.1	WDA	
128	(80)	1	VWDA	
128	(80)	1		
129	(81)	3	RDATAD	DATA ADDRESS
132	(84)	2		CHAIN BYTE AND
134	(86)	2	RDATLEN	RECORD LENGTH
136	(88)	2	RCCW1	CCW TO READ COUNT BUCKET
144	(90)	2	WCOUNT	COUNT BUCKET
144	(90)	5	WCNTCHR	DISK ADDRESS - CCHHR
144	(90)	4	WCNTCH	CCHH
148	(94)	1	WCNTR	R
		1..1 ..11	WCNTH	(TO ADDRESS HEAD NR)
149	(95)	1		RECORD LENGTH 1ST BYTE,0
150	(96)	2	WCNTDL	RECORD LENGTH

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
152	(98)	1	WVER3FL1	VERSION 3 FLAGS
153	(99)	3	WVER3FL2	VERSION 3 FLAGS
156	(9C)	4	WCISIZE	CISIZE IF PRESENT
		0000 1001	WSHCON	CONSTANTS FOR
160	(A0)	2	WMCOM	EFFECTIVE RECORD LENGTH
162	(A2)	2	WADCON	CALCULATION
THE FOLLOWING CONSTANTS ARE INIT FOR AND USED BY PPOINTGN ONLY				
164	(A4)	2	WEFRLEN	EFFECTIVE RECORD LENGTH
166	(A6)	1	PBLKPTRK	BLOCKS/TRACK
166	(A6)	1	PBLKPCI	BLOCKS/CONTROL INTERVAL
167	(A7)	1	WTRKPCYL	TRACKS/CYL FOR POINTGN ROUT
168	(A8)	2	CIMAXCAP	CI MAX CAPACITY
170	(AA)	2	WRECLEN1	LENGTHS OF TWO PRECEDING BLOCKS
172	(AC)	2	WRECLEN2	LENGTHS OF TWO PRECEDING BLOCKS
174	(AE)	1	NOTESW	SWITCH FOR CONTR. PNOTE
175	(AF)	1	NOTESW1	SWITCH FOR CONTR. PNOTE
176	(B0)	1	CKDEBASW	CKD/FBA SWITCH
		0000 0000	CKDTYPE	CODE FOR CKD DEVICE
		1001 0000	FBATYPE	CODE FOR FBA DEVICE

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
CCWS	96	(60)
CCWS1	96	(60)
CCWS2	104	(68)
CIMAXCAP	168	(A8)
CKDFBASW	176	(B0)
NOTESW	174	(AE)
NOTESW1	175	(AF)
PBLKPCI	166	(A6)
PBLKPTRK	166	(A6)
RCCW	128	(80)
RCCW1	136	(88)
RDA	128	(80)
RDAT	96	(60)
*RDATAD	129	(81)
*RDATLEN	134	(86)
VRDA	128	(80)
VRDAT	96	(60)
VWCKD	96	(60)
VWDA	128	(80)
VWDAT	96	(60)
WADCON	162	(A2)
WCC	56	(38)
WCCHH	56	(38)
WCCHHR	56	(38)
WCCW	112	(70)
WCCW2	120	(78)
WCHIEXT	50	(32)
WCKD	96	(60)
WCLOWEXT	46	(2E)
WCNTCH	144	(90)
WCNTCHR	144	(90)
WCNTDL	150	(96)

*POINTER

FIELD	DISPLACEMENT	
NAME	DECIMAL	(HEX)
WCNTR	148	(94)
WCOUNT	144	(90)
WCSW	4	(4)
WCURBLK	56	(38)
WDA	128	(80)
WDAT	96	(60)
*WDATAD	121	(79)
*WDATLEN	126	(7E)
WDEVTYP	29	(1D)
WEFFRLEN	164	(A4)
WEOF	4	(4)
WEXTLIM	50	(32)
WFRST	61	(3D)
WH	59	(3B)
WILOWEXT	42	(2A)
WIOCOMPL	2	(2)
WLOWHEAD	38	(26)
WMCON	160	(A0)
WMODAD	16	(10)
WMPCTY	62	(3E)
WPCTY	30	(1E)
WPNT	61	(3D)
WREC	60	(3C)
WRECLEN	40	(28)
WRECLEN1	170	(AA)
WRECLEN2	172	(AC)
WTRKLM	61	(3D)
WTRKPCYL	167	(A7)
WUPHEAD	39	(27)
WVER3FL1	152	(98)
WVER3FL2	153	(99)
W1	3	(3)
W2	3	(3)
W3	3	(3)

*POINTER

DATA AREA: **XREFREC**

SIZE: 11-17

CREATED BY:
UPDATED BY: IPKKA,IPKLA

FUNCTION: Description of an entry in the cross-reference table.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	8	XRSYMBOL	XREF ENTRIES FOR DEFN AND REF
8	(8)	1	XRFLAG	CROSSREF FLAG
		1.1.	DUPL	DUPLICATE
		.1..	REF	REFERENCE
		..1.	DEFIN	DEFINITION
9	(9)	2	XRSN	STATEMENT NUMBER
11	(B)	1	XREFEND	END OF REFERENCE RECORD
11	(B)	2	XRLATTR	LENGTH ATTRIBUTE
13	(D)	2	XRESDID	ESDID
15	(F)	3	XRVALUE	SYMBOL VALUE
18	(12)	1	XDEFEND	END OF DEF OR DUPL RECORD

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
DEFIN	8	(8)
DUPL	8	(8)
REF	8	(8)
XDEFEND	18	(12)
XREFEND	11	(B)
XRESDID	13	(D)
XRFLAG	8	(8)
XRLATTR	11	(B)
XRSN	9	(9)
XRSYMBOL	0	(0)
*XRVALUE	15	(F)

*POINTER

DATA AREA: **XRFENTRY**

SIZE: Variable (depending upon the literal source)

CREATED BY: } IPKRA,IPKRB,IPKRC
UPDATED BY: }

FUNCTION: Description of a cross-reference record.

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
0	(0)	1	XRFBYTE1	
		1111 .111	XRFEOB	XREF END OF BLOCK
0	(0)	1	VXRFEOB	
		.... 1..1	XRFLIT	LITERAL RECORD IF NUMERIC
0	(0)	1	VXRFLIT	
0	(0)	1		
				FIRST BYTE IN XREF RECORD
0	(0)	11	XRFREF	
0	(0)	6	XRFPSEUD	PSEUDO NAME FOR LITERAL XREF
				FIRST BYTE IN XREF RECORD
0	(0)	8	XRFSYM	SYMBOL NAME
8	(8)	1	XRFLAG	PROGRAM SWITCH
		1...	DUPSW	DUPLICATE DEFINITION RECORD
		.1..	REFSW	REFERENCE RECORD
		..1.	DEFSW	LITERAL DEFINITION RECORD
9	(9)	2	XRFSN	XREF STATEMENT NUMBER
11	(B)	7	XRFVAL	
11	(B)	2	XRFLATTR	LENGTH ATTR OF SYMBOL
13	(D)	2	XRFESDID	ESD ID
15	(F)	3	XRFVALUE	VALUE OF SYMBOL
				DEFINITION RECORD FIELD
11	(B)	1	XRFLITLN	LENGTH OF LIT XREF
12	(C)	1	XRFLSRC	BEGINNING OF LITERAL SOURCE

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
DEFSW	8	(8)
DUPSW	8	(8)
REFSW	8	(8)
VXRFEOB	0	(0)
VXRFLIT	0	(0)
XRFBYTE1	0	(0)
XRFEOB	0	(0)
XRFESDID	13	(D)

*POINTER

FIELD NAME	DISPLACEMENT DECIMAL	(HEX)
XRFLAG	8	(8)
XRFLATTR	11	(B)
XRFLIT	0	(0)
XRFLITLN	11	(B)
XRFLSRC	12	(C)
XRFPSEUD	0	(0)
XRFREF	0	(0)
XRFSN	9	(9)
XRFSYM	0	(0)
XRFVAL	11	(B)
XRFVALUE	15	(F)

*POINTER.

DATA AREA FIELD CROSS-REFERENCE

The following is a directory of field entries in the data areas illustrated in this section. The list includes the field name, DSECT name, and the field displacement in decimal and hexadecimal. For a cross-reference of the statements modifying or referencing these fields, see Appendix I.

FIELD	DSECT	DISPLACEMENT	
		DECIMAL	(HEX)
ASSGNSW	PCOMMON	1922	(782)
BCCOLUMN	EDPNI	10	(A)
BCDUMMY	EDPMI	10	(A)
BCITEM	EDPMI	10	(A)
BCITEMST	EDPMI	11	(B)
BEGOFIN	PCOMMON	2044	(7FC)
BEGOFOUT	PCOMMON	2040	(7F8)
BITLEM	PDCEDIT	5	(5)
BLKINCNO	PCOMMON	1995	(7D3)
BLKINSW	DIRENTRY	19	(13)
BLKNP1	DIRENTRY	5	(5)
BLKNP2	DIRENTRY	11	(B)
BOTTHALF	PCOMMON	2035	(7F3)
BTYPE	PDCEDIT	4	(4)
*BUFADDR	PFCB	6	(6)
*BUFADDR1	PCOMMON	1694	(69E)
*BUFADDR2	PCOMMON	1729	(6C1)
*BUFADDR3	PCOMMON	1764	(6E4)
*BUFPT	PFCB	3	(3)
*BUFPT1	PCOMMON	1691	(69B)
*BUFPT2	PCOMMON	1726	(6BE)
*BUFPT3	PCOMMON	1761	(6E1)
BUF2SW	PFCB	17	(11)
*BYTEOFFS	PCOMMON	1953	(7A1)
CCWS	WORKDTF	96	(60)
CCWSW	RLDENTRY	4	(4)
CCWS1	WORKDTF	96	(60)
CCWS2	WORKDTF	104	(68)
CHARC	EPAR	3	(3)
CHARFLAG	EPAR	0	(0)
CHARITEM	EDPMI	10	(A)
CHARK	EPAR	2	(2)
CHART	EPAR	1	(1)
CHITEMK	EDPMI	11	(B)
CHITEMST	EDPMI	12	(C)
CHITEMT	EDPMI	10	(A)
CIMAXCAP	WORKDTF	168	(A8)
CKDFBASW	WORKDTF	172	(AC)
CMTYPE	ESDENTRY	0	(0)
CODEHWD	CODE	0	(0)
CODESW	CODE	0	(0)
COMMAFLAG	CODE	0	(0)
COMMODSW	PCOMMON	1926	(786)
*CORADDR	DIRENTRY	0	(0)
CROSSNP	PCOMMON	1961	(7A9)
CROSSW	PCOMMON	1922	(782)
CTYPE	PDCEDIT	4	(4)
*CURESD	PCOMMON	1992	(7C8)
CURNP	PCOMMON	1967	(7AF)
CURSECT	PCOMMON	1932	(78C)
CURSECTL	PCOMMON	1931	(78B)
DBCORE	PCOMMON	1951	(79F)
DBFULLSW	PCOMMON	1926	(786)
DBVSDADR	PCOMMON	2000	(7D0)
DCEXP	PDCEDIT	0	(0)
DCVAL	PDCEDIT	0	(0)
DIMBIT	GSDENTRY	3	(3)
DEFIN	XREFREC	8	(8)
DEFSW	XRFENTRY	8	(8)
DIRENT	DIRENTRY	0	(0)
DLINK	PCOMMON	2024	(7E8)

*POINTER.

FIELD	DSECT	DISPLACEMENT	
		DECIMAL	(HEX)
DONESYM	PETFLDS	0	(0)
DSCOMSW	PCOMMON	2031	(7EF)
DSTYPE	ESDENTRY	0	(0)
DTYPE	PDCEDIT	4	(4)
DUPL	XREFREC	8	(8)
DUPSW	XRFENTRY	8	(8)
EAWF2END	PCOMMON	1920	(780)
EDFLAG	CODE	0	(0)
EDPMIORG	EDPMI	6	(6)
EFLG	MNAENT	2	(2)
EGSD	GSDENTRY	3	(3)
EGSDB	GSDENTRY	3	(3)
EITEMNK	EDPMI	11	(B)
EITEMNP	EDPMI	10	(A)
ELINK	PCOMMON	1976	(7B8)
EMNA	MNAENT	2	(2)
EMNAB	MNAENT	2	(2)
*ENDBUF	PFCB	11	(B)
ENDID	PCOMMON	2011	(7DB)
ENDITEM	EDPMI	10	(A)
ENDSW	PCOMMON	1963	(7AB)
ENTESD	PHYR	2	(2)
ENTRYCNT	PCOMMON	2008	(7D8)
ENTRYER	PHYR	2	(2)
ENTVALUE	PETFLDS	0	(0)
EOFLDPTR	ERRENT	5	(5)
EPARFLAG	EPAR	0	(0)
EPARREST	EPAR	2	(2)
EPART	EPAR	1	(1)
EQORGSW	PCOMMON	1922	(782)
ERDSCOM	PETFLDS	0	(0)
ERITEM	EDPMI	10	(A)
ERRAREA	ERRBYTES	0	(0)
ERRCONST	ERRBYTES	0	(0)
ERRCOUNT	PCOMMON	2036	(7F4)
ERRINFO	ERRENT	0	(0)
ERRINOPD	PCOMMON	1926	(782)
ERRLNG	ERRBYTES	2	(2)
ERRNO	ERRBYTES	0	(0)
ERRNO	ERRENT	0	(0)
ERRSW	ERRBYTES	1	(1)
ERRTXT	ERRBYTES	3	(3)
ERTYPE	ESDENTRY	0	(0)
ESDESDID	ESDENTRY	1	(1)
*ESDHILC	ESDENTRY	6	(6)
ESDIDHI	PCOMMON	1955	(7A3)
ESDIDLO	PCOMMON	2013	(7DD)
*ESDLCTR	ESDENTRY	3	(3)
ESDNXT	ESDENTRY	17	(11)
*ESDPTR	PCOMMON	1987	(7C4)
ESDSYM	ESDENTRY	9	(9)
ESDTABSW	PCOMMON	1963	(7AB)
ESDTYPE	ESDENTRY	0	(0)
ESMT	SMTENT	9	(9)
ESMTB	SMTENT	9	(9)
ESSD	SSD	3	(3)
ESSDB	SSD	3	(3)
ETYPE	PDCEDIT	4	(4)
EVALUE	EVALSTCK	1	(1)
EVLENGTH	EVALSTCK	5	(5)
EVNXT	EVALSTCK	10	(A)
EVPLUS	EVALSTCK	5	(5)

FIELD	DSECT	DISPLACEMENT	
		DECIMAL	(HEX)
EVRELOC	EVALSTCK	6	(6)
EVSD	VSD	3	(3)
EVSDB	VSD	3	(3)
EVVARY	EVALSTCK	7	(7)
EXPLEN	PDCEDIT	5	(5)
EXPON	PDCEDIT	0	(0)
FILE1NP	PCOMMON	1880	(758)
FILE1NPR	PCOMMON	1886	(75E)
FILE12EX	PCOMMON	1904	(770)
FILE2NP	PCOMMON	1892	(764)
FILE2NPR	PCOMMON	1898	(76A)
FILL	DIRENTRY	17	(11)
FIRSTSW	PFCB	17	(11)
FLAG	DIRENTRY	19	(13)
FLAGPMSK	EPAR	0	(0)
FLINK	PCOMMON	1960	(7A8)
FTYPE	PDCEDIT	4	(4)
GARDIM	GARENT	4	(4)
GAREND	GARENT	4	(4)
GARFLAG	GARD	6	(6)
GARLEN	GARD	0	(0)
GARLGTH	GARENT	3	(3)
GARNDX	GARENT	1	(1)
GARTYPE	GARENT	0	(0)
*GAVAPT	PCOMMON	1940	(794)
*GBVAPT	PCOMMON	1943	(797)
*GCVAPT	PCOMMON	1946	(79A)
GLINK	PCOMMON	1913	(779)
GSDDIM	GSDENTRY	4	(4)
GSDFLG	GSDENTRY	3	(3)
GSDLEN	GSDENTRY	3	(3)
GSDNDX	GSDENTRY	1	(1)
GSDSYM	GSDENTRY	4	(4)
GSDTYPE	GSDENTRY	0	(0)
HASHPTR	PHYR	0	(0)
HLINK	PCOMMON	1949	(79D)
HTYPE	PDCEDIT	4	(4)
IATESTSW	PCOMMON	1926	(786)
IFSAVE	PCOMMON	1672	(688)
IJJALSW	IJJCPTAB	44	(2C)
IJJCPADJ	IJJCPTAB	74	(4A)
IJJCPCCB	IJJCPTAB	0	(0)
IJJCPCCW	IJJCPTAB	112	(70)
IJJCPCNT	IJJCPTAB	76	(4C)
IJJCPCTR	IJJCPTAB	80	(50)
IJJCPDAT	IJJCPTAB	82	(52)
IJJCPF1	IJJCPTAB	30	(1E)
IJJCPMAX	IJJCPTAB	54	(36)
IJJCPREC	IJJCPTAB	64	(40)
IJJCPRMX	IJJCPTAB	72	(48)
IJJCPSCW	IJJCPTAB	48	(30)
IJJCPSEK	IJJCPTAB	60	(3C)
IJJCPSST	IJJCPTAB	88	(58)
IJJCPSWS	IJJCPTAB	42	(2A)
IJJCPUPP	IJJCPTAB	68	(44)
IJJCPVER	IJJCPTAB	128	(80)
IJJCPXTN	IJJCPTAB	40	(28)
IJJCP2ND	IJJCPTAB	45	(2D)
IJJC2NSW	IJJCPTAB	43	(2B)
IJJECCW1	IJJCPTAB	152	(98)
IJJECCW2	IJJCPTAB	160	(A0)
IJJFRSTR	IJJCPTAB	73	(49)
OINTER.			
IJJLOAD	IJJCPTAB	84	(54)
IJJLOHED	IJJCPTAB	52	(34)
IJJSAVEA	IJJCPTAB	168	(A8)
ILINK	PCOMMON	1984	(760)
IMPLEN	PDCEDIT	5	(5)
INDENT	INDENTRY	0	(0)
INDEX	EDPMI	16	(10)
INDKEY	INDENTRY	0	(0)
INDNP	INDENTRY	11	(B)
INDXB	EPAR	1	(1)
INDXC	EPAR	6	(6)
INDXCL	EPAR	5	(5)
INDXFLAG	EPAR	0	(0)
ITEM	EDPMI	10	(A)
ITEMATSW	EDPMI	8	(8)
ITEMFLAG	EDPMI	8	(8)
ITEMKWSW	EDPMI	8	(8)
ITEML	EDPMI	9	(9)
ITEMLISW	EDPMI	8	(8)
ITEMLONG	EDPMI	8	(8)
ITEMSLSW	EDPMI	8	(8)
ITEMT	EDPMI	7	(7)
ITEM1ST	EDPMI	8	(8)
JLINK	PCOMMON	1928	(788)
KEY	DIRENTRY	11	(B)
KITEM	EDPMI	10	(A)
KLINK	PCOMMON	2072	(818)
KWTLEN	KEYTAB	0	(0)
LBARADDR	PCOMMON	1994	(76A)
LCLASIZ	PCOMMON	1932	(78C)
LCLASZ	MACHEAD	16	(10)
LCLBSIZ	PCOMMON	1935	(78F)
LCLBSZ	MACHEAD	19	(13)
LCLCSIZ	PCOMMON	1935	(792)
LCLCSZ	MACHEAD	22	(16)
LDNAME	PHYR	2	(2)
LDTYPE	ESDENTRY	0	(0)
LEN1SW	RLDENTRY	4	(4)
LEN2SW	RLDENTRY	4	(4)
LITSW	PCOMMON	1963	(7AB)
LLINK	PCOMMON	2072	(818)
LMNAME	EDPMI	7	(7)
*LOCCNTHI	PCOMMON	2028	(7EC)
*LOCCNTR	PCOMMON	2025	(7E9)
LOCLATR	PCOMMON	2021	(7E5)
LOCRATR	PCOMMON	2023	(7E7)
LOCSTMH	OCSTMH	7	(7)
LOCTYPE	PCOMMON	2031	(7EF)
LPNAME	EDPMI	7	(7)
LTYPE	PDCEDIT	4	(4)
LXFLAG	PHYR	2	(2)
MACNAM	EDPMI	8	(8)
MACROSW	PCOMMON	1926	(786)
MAN	MNAENT	3	(3)
MFLAGS	PCOMMON	1922	(782)
MIB	PCOMMON	1922	(782)
*MIBADDR	PCOMMON	1948	(79C)
MLEVEL	PCOMMON	1951	(79D)
MNALEN	MNAENT	2	(2)
MNAM	PCOMMON	1924	(784)
MNAME	MACHEAD	8	(8)
MNAMEL	MACHEAD	7	(7)

FIELD	DSECT	DISPLACEMENT DECIMAL	(HEX)
MNANDX	MNAENT	0	(0)
NALTSRC	EDPMI	8	(8)
NERNUMB	NTABFMT	0	(0)
NEXPF1	PETFLDS	1	(1)
NEXPF2	PETFLDS	1	(1)
NEXPF3	PETFLDS	1	(1)
NEXTCODE	CODE	2	(2)
NLINK	PCOMMON	1960	(7A8)
NOPRNUMB	NTABFMT	1	(1)
NOSTRING	ERRENT	5	(5)
NOTESW	WORKDTF	170	(AA)
NOTESW1	WORKDTF	171	(AB)
NPLITBEG	PCOMMON	1913	(779)
NPMIB	OCSTMH	7	(7)
NPSSDR1	PCOMMON	1963	(7AB)
NPSSDWL	PCOMMON	1970	(7B2)
NPTEMP	PCOMMON	1660	(67C)
NPTEMPCC	PCOMMON	1660	(67C)
NPTEMPH	PCOMMON	1662	(67E)
NPTEMPR	PCOMMON	1663	(67F)
NPTEMPTB	PCOMMON	1664	(680)
NPVSD	PCOMMON	1947	(79B)
NPVSDR1	PCOMMON	1987	(7C3)
*NXTENTRY	PCOMMON	2004	(7D4)
NXTRA	EVALSTCK	3	(3)
OFFS	DIRENTRY	3	(3)
OLINK	PCOMMON	1960	(7A8)
OMITEM	EDPMI	10	(A)
OPENSW	PFCB	17	(11)
OPND	ERRBYTES	1	(1)
OVFLADDR	PCOMMON	1997	(7CD)
PABENDC	PCOMMON	1669	(685)
PAFLAG2	PETFLDS	0	(0)
*PAFLD	PETFLDS	1	(1)
PAFLG	PETFLDS	0	(0)
PALGNBIT	PETR	2	(2)
PALIGN	PCOMMON	1659	(67B)
PANXT	PETFLDS	4	(4)
PASSGNSW	PCOMMON	2035	(7F3)
PBITAL	PDCOUT	7	(7)
PBITALGN	PDCEDIT	7	(7)
PBLKPCI	WORKDTF	166	(A6)
PBLKPTRK	WORKDTF	166	(A6)
*PBUFLEN1	PCOMMON	1697	(6A1)
*PBUFLEN2	PCOMMON	1732	(6C4)
*PBUFLEN3	PCOMMON	1767	(6E7)
PB1FISIZ	PCOMMON	1890	(762)
PB12SIZ	PCOMMON	1892	(764)
PCAEDTXT	PCSR	6	(6)
PCASRC	PCSR	6	(6)
PCBTYPE	PETR	5	(5)
PCCHR	PFCB	21	(15)
PCI	PFCB	21	(15)
PCIBAL	PFCB	25	(19)
PCNXT	PETFLDS	3	(3)
PCODE	PETFLDS	1	(1)
PCONTSW	PCOMMON	1904	(770)
PCSREAD	PCSR	5	(5)
PCSRFLGA	PCSR	6	(6)
PCSRHEAD	PCSR	0	(0)
PCSRIOP	PCSR	2	(2)
PCSRLEN	PCSR	0	(0)
PCSRLIT1	PCSR	2	(2)
PCSRLIT2	PCSR	2	(2)
PCSRLIT3	PCSR	2	(2)
PCSRMCOP	PCSR	2	(2)
PCSROP	PCSR	3	(3)
PCSROPX	PCSR	4	(4)
PCSROP0	PCSR	2	(2)
PCSROP3	PCSR	5	(5)
PCSRSTR1	PCSR	7	(7)
PCSWOVL	PCOMMON	1871	(74F)
PCTYPE	ESDENTRY	0	(0)
PDCCODE	PDCOUT	11	(B)
PDCCONT	PETR	2	(2)
PDCED	PDCOUT	3	(3)
PDCFL	PDCOUT	6	(6)
PDCFLAG	PDCEDIT	6	(6)
PDCFLD	PDCEDIT	2	(2)
PDCLEN	PDCEDIT	1	(1)
PDCSTYPE	PETR	2	(2)
PDCTYPE	PDCEDIT	4	(4)
PDECK	PCOMMON	1659	(67B)
PDEFENT	PETR	2	(2)
PDF	PCOMMON	1659	(67B)
PDSWOVL	PCOMMON	1871	(74F)
*PDTFADDR	PFCB	0	(0)
*PDTFADR1	PCOMMON	1688	(698)
*PDTFADR2	PCOMMON	1723	(6BB)
*PDTFADR3	PCOMMON	1758	(6DE)
PDUPC	PDCOUT	6	(6)
PDCUPCONT	PDCEDIT	6	(6)
PDUPEXP	PDCEDIT	0	(0)
PDUPLFAC	PDCOUT	0	(0)
PDUPSYM	PETR	2	(2)
PEDECK	PCOMMON	1659	(67B)
*PENDBUF1	PCOMMON	1699	(6A3)
*PENDBUF2	PCOMMON	1734	(6C6)
*PENDBUF3	PCOMMON	1769	(6E9)
PENTDEF	PCOMMON	1922	(782)
PENTVAL	PETFLDS	9	(9)
PEOBSW	PCOMMON	1658	(67A)
*PEOF	PFCB	18	(12)
*PEOFADR1	PCOMMON	1706	(6AA)
*PEOFADR2	PCOMMON	1741	(6CD)
*PEOFADR3	PCOMMON	1776	(6F0)
PERLNG	PETR	6	(6)
PERNOQ	PETR	2	(2)
PERREXC	PETR	9	(9)
PERREXPT	PERR	9	(9)
PERRHD	PERR	0	(0)
PERRIOP	PERR	2	(2)
PERRIOPC	PERR	2	(2)
PERRIOP1	PERR	3	(3)
PERRIOP2	PERR	4	(4)
PERRLEN	PERR	0	(0)
PERRSTNO	PETR	6	(6)
PERRSTNR	PERR	6	(6)
PERRSTR	PETR	8	(8)
PERRSTRL	PERR	8	(8)
PERSRC	PETR	7	(7)
PER2STR	PETR	2	(2)
PETDEAD	PETR	5	(5)
PETEPX	PETR	4	(4)

*POINTER.

FIELD	DSECT	DISPLACEMENT DECIMAL	(HEX)
PETHEAD	PETR	0	(0)
PETIOP	PETR	2	(2)
PETLEN	PETR	0	(0)
PETMCOP	PETR	2	(2)
PETOP	PETR	3	(3)
PETOP0	PETR	2	(2)
PETOP3	PETR	5	(5)
PETSTNO	PETR	6	(6)
PEXPFLAG	PETFLDS	1	(1)
PEXPVAL	PETFLDS	5	(5)
PFCBSW	PFCB	17	(11)
PFCBSW1	PFCB	17	(11)
PFCBSW2	PFCB	17	(11)
*PFETCHDA	PCOMMON	2008	(7D8)
*PFETCHDB	PCOMMON	2012	(7DC)
*PFETCHIA	PCOMMON	1956	(7A4)
*PFETCHIB	PCOMMON	1960	(7A8)
*PFETCHIC	PCOMMON	1964	(7AC)
PFILE1	PCOMMON	1688	(698)
PFILE2	PCOMMON	1723	(6BB)
PFILE3	PCOMMON	1758	(6DE)
*PFINDPT	PCOMMON	1896	(768)
PFLDCOL	PSTRINGS	1	(1)
PFLDLEN	PSTRINGS	0	(0)
PFLDSRC	PSTRINGS	2	(2)
PFROMLIB	PCSR	6	(6)
PFRSTASG	PCOMMON	1922	(782)
PFRSTOV	PCOMMON	2035	(7F3)
PGBLASIZ	PCOMMON	1945	(799)
PGBLBSIZ	PCOMMON	1948	(79C)
PGBLCSIZ	PCOMMON	1951	(79F)
PGBLSIZ	PCOMMON	1945	(799)
PGENSTMT	PCSR	6	(6)
PGENSW	PCOMMON	1926	(786)
PGEN5SW	PCOMMON	1926	(786)
PGFMSGSW	PCOMMON	1863	(747)
PGVENT1	PGVHEAD	7	(7)
PGVHIOP	PGVHEAD	2	(2)
PGVHLEN	PGVHEAD	0	(0)
*PHICORE	PCOMMON	1666	(682)
PIBSW	PCOMMON	1926	(786)
PICSW	PCOMMON	1926	(786)
*PICTLCNT	PCOMMON	1856	(740)
*PICTLEND	PCOMMON	1855	(73F)
*PICTLST	PCOMMON	1854	(73E)
PIERCNT	PCOMMON	1974	(7B6)
PIERSTK	PCOMMON	1976	(7B8)
PINEOFSW	PCOMMON	1659	(67B)
*PINPUTPT	PCOMMON	1899	(76B)
PINSTRLN	PETR	5	(5)
PISWOVL	PCOMMON	1871	(74F)
PITEM	EDPMI	10	(A)
PJSWOVL	PCOMMON	1871	(74F)
PKSWOVL	PCOMMON	1871	(74F)
PLA	PETFLDS	2	(2)
PLASTSUB	PCOMMON	2035	(7F3)
PLBEOFSW	PCOMMON	1658	(67A)
PLENATTR	PETR	8	(8)
PLENFLAG	PCEDIT	5	(5)
PLINECNT	PCOMMON	1848	(738)

*POINTER

FIELD	DSECT	DISPLACEMENT DECIMAL	(HEX)
PLINENUM	PCOMMON	1923	(783)
PLINK	PCOMMON	1659	(67B)
PLIST	PCOMMON	1659	(67B)
PLITBLK	PCOMMON	1909	(775)
PLITLEN	PCOMMON	1911	(777)
*PLOCCNTR	PETR	12	(C)
PLORMIN	EVALSTCK	0	(0)
PLRECLN	PCOMMON	1670	(686)
PLRELATR	PETR	10	(A)
*PMAVBSIZ	PCOMMON	1941	(795)
*PMAVNO	PCOMMON	1943	(797)
PMAVNP	PCOMMON	1935	(78F)
PMAXBSIZ	PCOMMON	1787	(6FB)
*PMIBLEN	PCOMMON	1956	(7A4)
PMODEXP	PCEDIT	1	(1)
PMODFLAG	PCEDIT	0	(0)
PMODIFS	PCEDIT	9	(9)
PNAMCOL	PSTRINGS	1	(1)
PNAME	PETFLDS	1	(1)
PNAMFLD	PETR	16	(10)
PNAMLEN	PSTRINGS	0	(0).
PNAMLNG	PETFLDS	0	(0)
PNAMSRC	PSTRINGS	2	(2)
PNEXTNP	PFCB	29	(1D)
PNEXTNP1	PCOMMON	1717	(6B5)
PNEXTNP2	PCOMMON	1752	(6D8)
PNEXTNP3	PCOMMON	1787	(6FB)
PNOBKSW	PCOMMON	1658	(67A)
PNOCONST	PCEDIT	3	(3)
PNOSEQSW	PCOMMON	1863	(747)
PNOSTNO	PCSR	2	(2)
PNOTEPNT	PFCB	21	(15)
PNPMAC1	PCOMMON	1912	(778)
PNPOCGV	PCOMMON	1927	(787)
PNPOFFS	PFCB	27	(1B)
PNPOINT1	PCOMMON	1709	(6AD)
PNPOINT2	PCOMMON	1744	(6D0)
PNPOINT3	PCOMMON	1779	(6F3)
PNPRW	PFCB	21	(15)
PNWTRKSW	PCOMMON	1658	(67A)
PNXTBKT	PETFLDS	9	(9)
PNXTMOD	PCEDIT	4	(4)
POLSTR	PETFLDS	2	(2)
POPCOL	PSTRINGS	1	(1)
POPDCOL	PSTRINGS	1	(1)
POPDLEN	PSTRINGS	0	(0)
POPDSRC	PSTRINGS	2	(2)
POPFLAG	PETFLDS	0	(0)
POPLEN	PSTRINGS	0	(0)
POPNUMB	PETR	4	(4)
POPSRC	PSTRINGS	2	(2)
POPSW	PCOMMON	1658	(67A)
POVLSW	PCOMMON	1871	(74F)
PPAGENO	PCOMMON	1925	(785)
PRA	PETFLDS	4	(4)
PRATT	PETFLDS	10	(A)
PREC	PFCB	24	(18)
PRECLEN	PFCB	9	(9)
PREFCNT	PCOMMON	1936	(790)
PRLD	PCOMMON	1658	(67A)
PROGID	PCOMMON	1850	(73A)

FIELD	DSECT	DISPLACEMENT DECIMAL	(HEX)
PRONAME	EDPMI	8	(8)
PSAVETBL	PCOMMON	1796	(704)
PSAVPT	PCOMMON	1836	(72C)
PSAVTEMP	PCOMMON	1840	(730)
PSBSTNAM	PCSR	6	(6)
PSBSTOP	PCSR	6	(6)
PSBSTOPD	PCSR	6	(6)
PSFLAG2	PETFLDS	0	(0)
PSIGN	PETFLDS	9	(9)
PSLENATR	PETFLDS	1	(1)
PSMACDEF	PCSR	6	(6)
PSPILLA	PCOMMON	1968	(7B0)
PSRELATR	PETFLDS	3	(3)
PSTMCNAM	PCOMMON	1912	(778)
PSTMCSEQ	PCOMMON	1915	(77B)
PSTRCOL	PSTRINGS	1	(1)
PSTRLEN	PSTRINGS	0	(0)
PSTRSRC	PSTRINGS	2	(2)
*PSVALUE	PETFLDS	5	(5)
PSXREF	PCOMMON	1658	(67A)
PSYMBOL	PETFLDS	1	(1)
PSYMFLAG	PETFLDS	0	(0)
PSYMLEN	PETFLDS	0	(0)
PSYMNO	PETR	15	(F)
*PSYMTABL	PCOMMON	2032	(7F0)
PSYSNDX	PCOMMON	1927	(787)
PSYSPLEN	PCOMMON	1880	(758)
PSYSPSTR	PCOMMON	1881	(759)
PTRKBAL	PFCB	25	(19)
PTRUNR	PDCOUT	6	(6)
PTRUNRHT	PDCEDIT	6	(6)
PTYPE	PDCEDIT	4	(4)
PUNDEFSW	PCOMMON	1926	(786)
PVSDSIZE	PCOMMON	1904	(770)
*PWAADDR	PFCB	14	(E)
*PWAADDR1	PCOMMON	1702	(6A6)
*PWAADDR2	PCOMMON	1737	(6C9)
*PWAADDR3	PCOMMON	1772	(6EC)
PXREF	PCOMMON	1659	(67B)
RANR	EVALSTCK	0	(0)
RAONE	EVALSTCK	1	(1)
RCCW	WORKDTF	128	(80)
RCCW1	WORKDTF	136	(88)
RDA	WORKDTF	128	(80)
RDAT	WORKDTF	96	(60)
*RDATAD	WORKDTF	129	(81)
*RDATLEN	WORKDTF	134	(86)
READSW	PFCB	17	(11)
REF	XREFREC	8	(8)
REFSW	XRFENTRY	8	(8)
RELLEN	EVALSTCK	8	(8)
*RLADDR	RLDENTRY	5	(5)
RLADID	RLDENTRY	0	(0)
RLDNXT	RLDENTRY	8	(8)
RLFLAG	RLDENTRY	4	(4)
RLREFID	RLDENTRY	2	(2)
*SAVADDR	PCOMMON	2048	(800)
SAVAR	PCOMMON	2051	(803)
SAVESDNP	PCOMMON	1967	(7B5)
SAVREG1	PCOMMON	2060	(80C)

*POINTER

FIELD	DSECT	DISPLACEMENT DECIMAL	(HEX)
SAVREG2	PCOMMON	2064	(810)
SCALE	PCEDIT	0	(0)
SDEFB	EPAR	2	(2)
SDEFC	EPAR	6	(6)
SDEFFLAG	EPAR	0	(0)
SDEFITEM	EDPMI	10	(A)
SDEFK	EPAR	5	(5)
SDEFT	EPAR	1	(1)
SDITEMB	EDPMI	11	(B)
SDITEMK	EDPMI	14	(E)
SDIEMST	EDPMI	15	(F)
SDITEMT	EDPMI	10	(A)
SDTYPE	ESDENTRY	0	(0)
SECTC	EPAR	12	(C)
SECTCL	EPAR	11	(B)
SECTFLAG	EPAR	10	(A)
SECTSW	PCOMMON	1963	(7AB)
SEITEM	EDPMI	10	(A)
SEQFLD	EDPMI	18	(12)
SFLAG	CODE	0	(0)
SIGNSW	RLDENTRY	4	(4)
SMTEFLG	SMTENT	9	(9)
SMTLEN	SMTENT	9	(9)
SMTNAME	SMTENT	10	(A)
SMTNP	SMTENT	1	(1)
SMTSIZE	PCOMMON	1902	(76E)
SREG	PETFLDS	0	(0)
SSDADDR	PCOMMON	1953	(7A1)
SSDBLK1	PCOMMON	1969	(7B1)
*SSDEND	PCOMMON	2019	(7E3)
SSDFLAG	SSD	3	(3)
SSDINFO	PCOMMON	1958	(7A6)
SSDNP	PCOMMON	1941	(795)
SSDNPT	MACHEAD	25	(19)
SSDOFFS	SSD	0	(0)
SSDSIZE	PCOMMON	1961	(7A9)
SSDSYM	SSD	4	(4)
SSDSYML	SSD	3	(3)
SSITEMK	EDPMI	10	(A)
*STABEND	PCOMMON	2000	(7D0)
*STARTLOC	PCOMMON	1952	(7A0)
STARTSW	PCOMMON	1922	(782)
STEP	PCOMMON	1844	(734)
STLENGTH	EVALSTCK	0	(0)
STMTNR	PCOMMON	1921	(781)
STRING	ERRENT	5	(5)
STRPTR	ERRENT	2	(2)
STYPE	PDCEDIT	4	(4)
SUBE	EPAR	7	(7)
SUBEFLAG	EPAR	5	(5)
SUBITEM	EDPMI	10	(A)
SUBEL	EPAR	6	(6)
SUBLFLAG	EPAR	0	(0)
SUBLHEAD	EPAR	0	(0)
SUBLK	EPAR	4	(4)
SUBLL	EPAR	1	(1)
SUBLN	EPAR	3	(3)
SUBSITEM	EDPMI	10	(A)
SWATRINS	EDPMI	6	(6)
SWATTR	PCOMMON	1922	(782)
SWCAFT	PCOMMON	1889	(761)

FIELD	DSECT	DISPLACEMENT DECIMAL	(HEX)
SWCAOC	PCOMMON	1889	(761)
SWCOPY	PCOMMON	1920	(780)
SWDS	PCOMMON	1922	(782)
SWEFILE	PCSR	2	(2)
SWFLUSH	PCOMMON	1920	(780)
SWGBLX	PCOMMON	1922	(782)
SWGVLST	PGVHEAD	6	(6)
SWINM	PCOMMON	1922	(782)
SWKT	PCOMMON	1922	(782)
SWLA	PCOMMON	1922	(782)
SWLASTKW	KEYTAB	6	(6)
SWMACRO	PCOMMON	1920	(780)
SWMATR	MACHEAD	6	(6)
SWMGBL	MACHEAD	6	(6)
SWMH	MACHEAD	6	(6)
SWMINM	MACHEAD	6	(6)
SWMIOC	PCOMMON	1889	(761)
SWMKEYW	MACHEAD	6	(6)
SWNOED	PCOMMON	1922	(782)
SWNOGEN	PCOMMON	1922	(782)
SWNOSTOR	PCOMMON	1922	(782)
SWOC	PCOMMON	1920	(780)
SWPMI1	EDPMI	6	(6)
SWPROTO	PCOMMON	1920	(780)
SWREPRO	PCOMMON	1920	(780)
SWSFILE	PCSR	2	(2)
SWSM	PCOMMON	1889	(761)
SWSMT	SMTENT	0	(0)
SWSMTATR	SMTENT	0	(0)
SWSMTGBL	SMTENT	0	(0)
SWSMTINM	SMTENT	0	(0)
SWSMTNED	SMTENT	0	(0)
SWSMTNGN	SMTENT	0	(0)
SWSMTNST	SMTENT	0	(0)
SWSTART	PCOMMON	1963	(7AB)
SWSTR	ERRENT	0	(0)
SWSUBST	EDPMI	6	(6)
SW2	PCOMMON	1963	(7AB)
*SYMADDR	PCOMMON	1996	(7CC)
SYMBREG	PHYR	2	(2)
SYMESDID	PHYR	5	(5)
SYMFLAGS	PHYR	2	(2)
SYMLATTR	PHYR	3	(3)
SYMLENG	PHYR	10	(A)
SYMNXT	PHYR	19	(13)
SYMSRC	PHYR	11	(B)
*SYMVALUE	PHYR	7	(7)
SYM1C	EPAR	9	(9)
SYM1FLAG	EPAR	0	(0)
SYM1I	EPAR	6	(6)
SYM1ITEM	EDPMI	10	(A)
SYM1K	EPAR	8	(8)
SYM1L	EPAR	2	(2)
SYM1S	EPAR	4	(4)
SYM1T	EPAR	1	(1)
SYM2C	EPAR	5	(5)
SYM2FLAG	EPAR	0	(0)
SYM2ITEM	EDPMI	10	(A)
SYM2K	EPAR	4	(4)
SYM2L	EPAR	2	(2)
SYM2T	EPAR	1	(1)
S1ITEMI	EDPMI	15	(F)
S1ITEMK	EDPMI	17	(11)
S1ITEML	EDPMI	11	(B)
S1ITEMS	EDPMI	13	(D)
S1ITEMST	EDPMI	18	(12)
S1ITEMT	EDPMI	10	(A)
S2ITEMK	EDPMI	13	(D)
S2ITEML	EDPMI	11	(B)
S2ITEMST	EDPMI	14	(E)
S2ITEMT	EDPMI	10	(A)
TEXT	ERRBYTES	1	(1)
THISELEM	PETFLDS	0	(0)
TYPEBC	EDPMI	11	(B)
TYPECHAR	EDPMI	11	(B)
TYPEEND	EDPMI	11	(B)
TYPEER	EDPMI	11	(B)
TYPEESD	PCOMMON	2010	(7DA)
TYPEK	EDPMI	11	(B)
TYPEKP	EDPMI	11	(B)
TYPEOM	EDPMI	11	(B)
TYPEPP	EDPMI	11	(B)
TYPESDEF	EDPMI	11	(B)
TYPESUBE	EDPMI	11	(B)
TYPESUBS	EDPMI	11	(B)
TYPES1	EDPMI	11	(B)
TYPES2	EDPMI	11	(B)
UNFLAG	CODE	0	(0)
UPDSW	PFCB	17	(11)
UPD2SW	PFCB	17	(11)
VBITLEN	PDCEDIT	5	(5)
VCMTYPE	ESDENTRY	0	(0)
VCONSW	RLDENTRY	4	(4)
VCTYPE	ESDENTRY	0	(0)
VDCEXP	PDCEDIT	0	(0)
VDCVAL	PDCEDIT	0	(0)
VDSTYPE	ESDENTRY	0	(0)
VERTYPE	ESDENTRY	0	(0)
VEXPLEN	PDCEDIT	5	(5)
VEXPON	PDCEDIT	0	(0)
VIMPLEN	PDCEDIT	5	(5)
VLDTYPE	ESDENTRY	0	(0)
VPCTYPE	ESDENTRY	0	(0)
VRDA	WORKDTF	128	(80)
VRDAT	WORKDTF	96	(60)
VSCALE	PCEDIT	0	(0)
VSADDR	PCOMMON	1982	(7BE)
VSDBLK1	PCOMMON	1993	(7C9)
VSDDIM	VSD	4	(4)
*VSDEND	PCOMMON	2016	(7E0)
VSDFLAG	VSD	3	(3)
VSDINFO	PCOMMON	1982	(7BE)
VSDNDX	VSD	1	(1)
VSDNPT	MACHEAD	31	(1F)
VSDSIZE	PCOMMON	1985	(7C1)
VSDSYM	VSD	4	(4)
VSDSYML	VSD	3	(3)
VSDTYPE	ESDENTRY	0	(0)
VSDTYPE	VSD	0	(0)
VTYPE	PDCEDIT	4	(4)
VVCTYPE	ESDENTRY	0	(0)
VWCKD	WORKDTF	96	(60)

*POINTER

FIELD	DSECT	DISPLACEMENT DECIMAL	(HEX)
VWDA	WORKDTF	128	(80)
VWDAT	WORKDTF	96	(60)
VWXTYPE	ESDENTRY	0	(0)
VXRFEOB	XRFENTRY	0	(0)
VXRFLIT	XRFENTRY	0	(0)
WADCON	WORKDTF	162	(A2)
WCC	WORKDTF	56	(38)
WCCHH	WORKDTF	56	(38)
WCCHHR	WORKDTF	56	(38)
WCCW	WORKDTF	112	(70)
WCCW2	WORKDTF	120	(78)
WCHIEXT	WORKDTF	50	(32)
WCKD	WORKDTF	96	(60)
WCLOWEXT	WORKDTF	46	(2E)
WCNTCH	WORKDTF	144	(90)
WCNTCHR	WORKDTF	144	(90)
WCNTDL	WORKDTF	150	(96)
WCNTR	WORKDTF	148	(94)
WCOUNT	WORKDTF	144	(90)
WCSW	WORKDTF	4	(4)
WCURBLK	WORKDTF	56	(38)
WDA	WORKDTF	128	(80)
WDAT	WORKDTF	96	(60)
*WDATAD	WORKDTF	121	(79)
*WDATLEN	WORKDTF	126	(7E)
WDEVTYP	WORKDTF	29	(1D)
WEFFRLEN	WORKDTF	164	(A4)
WEOF	WORKDTF	4	(4)
WEXTLIM	WORKDTF	50	(32)
WFRST	WORKDTF	61	(3D)
WH	WORKDTF	59	(3B)
WILOWEXT	WORKDTF	42	(2A)
WIOCOMPL	WORKDTF	2	(2)
WLOWHEAD	WORKDTF	38	(26)
WMCOM	WORKDTF	160	(A0)
WMODAD	WORKDTF	16	(10)
WMPCTY	WORKDTF	62	(3E)
WPCTY	WORKDTF	30	(1E)
WPNT	WORKDTF	61	(3D)
WREC	WORKDTF	60	(3C)
WRECLEN	WORKDTF	40	(28)
WTRKLM	WORKDTF	61	(3D)
WTRKPCYL	WORKDTF	167	(A7)
WUPHEAD	WORKDTF	39	(27)
WVER3FL1	WORKDTF	152	(98)
WVER3FL3	WORKDTF	153	(99)
WXTYPE	ESDENTRY	0	(0)
W1	WORKDTF	3	(3)
W2	WORKDTF	3	(3)
W3	WORKDTF	3	(3)
*POINTER			

FIELD	DSECT	DISPLACEMENT DECIMAL	(HEX)
XDEFEND	XREFREC	18	(12)
*XRAREND	PCOMMON	1982	(7BE)
*XREFADDR	PCOMMON	2016	(7E0)
XREFEND	XREFREC	11	(B)
XREFLEN	PCOMMON	2019	(7E3)
XREFPARM	PCOMMON	2013	(7DF)
*XREFPTR	PCOMMON	1979	(7BB)
XRESDID	XREFREC	13	(D)
XRFBYTE1	XRFENTRY	0	(0)
XRFEOB	XRFENTRY	0	(0)
XRFESDID	XRFENTRY	13	(D)
XRFLAG	XREFREC	8	(8)
XRFFLAG	XRFENTRY	8	(8)
XRFLATTR	XRFENTRY	11	(B)
XRFLIT	XRFENTRY	0	(0)
XRFLITLN	XRFENTRY	11	(B)
XRFLSRC	XRFENTRY	12	(C)
XRFPSEUD	XRFENTRY	0	(0)
XRFREF	XRFENTRY	0	(0)
XRFSN	XRFENTRY	9	(9)
XRFSYM	XRFENTRY	0	(0)
XRFVAL	XRFENTRY	11	(B)
XRFVALUE	XRFENTRY	15	(F)
XRFLATTR	XREFREC	11	(B)
XRSN	XREFREC	9	(9)
XRSYMBOL	XREFREC	0	(0)
*XRVALUE	XREFREC	15	(F)
XTYPE	PDCEDIT	4	(4)
YCONSW	RLDENTRY	4	(4)
YTYPE	PDCEDIT	4	(4)
ZTYPE	PDCEDIT	4	(4)

(This page intentionally left blank.)

(This page intentionally left blank.)

Diagnostic Aids

Purpose of the Section

This section contains information that may be useful in diagnosing problems within the assembler. The section includes the following information:

- Debugging aids
- I/O activity and workfile layouts
- Register usage

Debugging Aids

It is seldom possible to fix assembler faults while in the field; however, you may want to try the following "First Aid" when attempting to circumvent a problem:

First Aid

- If there has been a program check, try to rerun the job in a different-sized partition
- Find out what changes were made to the source program since it last assembled successfully and either delete them or incorporate them using another method

In general, assembler malfunctions can be divided into two classes: those in which you get the wrong output from the assembler and those in which you get a program check.

WRONG ASSEMBLER OUTPUT

An error message issued against a correct statement or an erroneous ESD, RLD, or object record, are examples of wrong assembler output. In these cases it is usually possible to trace the error to a particular statement or sequence of statements and rewrite them. At the same time, an APAR should be submitted (see Appendix J) to the assembler maintenance group.

PROGRAM CHECK

If the "First Aid" measures described above do not work, in some cases it may help to know what statement was being processed when the program check occurred. Although it is not certain that the fault was caused by the "current" statement, it may be possible to find the corresponding source statements and rewrite them in such a way that the assembly will be successful.

How to Find the Current Statement

1. Identify the object module executing when the program check occurred. General register 11 points to an address 10 bytes to the right of the identifier for the module.

2. Identify which workfiles were being read from or written on (see Figure 3 or the relevant method-of-operation diagram).

3. Find the corresponding workarea for the workfile being read from or written on -- each workfile has its own workarea in the common data area (COMMON). The offsets for these workareas in COMMON are given in the dummy control section "PCOMMON" (see the Data Area Field Cross-reference in "Data Areas") and are:

Workfile	Workarea offset	PCOMMON label
1	6A6	PWAADDR1
2	6C9	PWAADDR2
3	6EC	PWAADDR3

4. Find the beginning of COMMON by examining the contents of register 13 -- it points to COMMON.

5. Add the offset obtained in step 3 to the starting address of COMMON to find the workarea you are interested in.

Interpreting the Statement

Once you have found the current statement in its edited form, you will want to interpret it in order to identify its source-statement equivalent. You may be able to identify the statement by examining the pseudo operation code field; this field is always the fourth byte of the edited statement. The name field and the symbols used in the operand, if they are still present in the edited statement, will also help to identify it.

To fully interpret statements in their edited format, you will need the following information:

What you need to know	Where to find it
The statement formats for the phase	Overview: Appendix H Details: Relevant DSECT in "Data Areas"
Pseudo opcodes assigned by the assembler	Appendix E
The assembler internal character sets	Appendix F
Edited text flags	Appendix G

Examples of Edited Text

This section contains two statements shown at different stages in processing. The statements were made complex in order to make the examples comprehensive.

Example 1. AIF statement, object module IPKDA.

The AIF statement is part of the following macro definition:

```
         MACRO
         MAC1     &PARM1,&PARM2,&PARM3
         LCLA     &X,&Y,&Z,&A
         LCLB     &M,&N,&B
         LCLC     &Q",&C,(50)
&A       SETA     5
&B       SETB     1
         AIF      ('ABC' EQ '&C(&A)' AND &A*&B+1 LT &PARM2).SEQSYM
.SEQSYM  ANOP
         MEND
```

Following the instructions given in the previous section, we can find the record as it was written out by object module IPKDA:

```
                                        003904 2E240000 00040B0A 120F3811
14030A0B 0C0A0001 00320000 03382900   00030400 021F1810 01160C00 0534241B
12061C0E 1A1C2216 2F3A
```

We can interpret the record with the help of the record format for an AIF statement (see Appendix H, "Object Module IPKDA").

0039	record length	
04	file flag	
2E	pseudo opcode ("AIF")	DSECT pcsr
24	opcode extension (dummy)	
00	type flag	
00	flag A	
00040B0A120F3811	sequence field	
14	character string (see Appendix H)	
03	length	
0A0B0C "ABC"	(see Appendix H)	
0A	LCLC dimensioned ("&C")	
0001	&Cs index	
0032	maximum dimension of &A	
00	LCLA ("&A")	
0003	"&A"s index	
38	subscript	
29	character EQ	
00	LCLA ("&A")	
0003	"&A"s index	
04	LCLB ("&B")	
0002	"&B"s index	
1F	Boolean to arithmetic conversion	
18	* (multiplication)	
10	1-byte binary value	
01	value ("1")	
16	+ (addition)	
0C	positional parameter ("&PARM2")	
0005	"&PARM2"s index	
34	binary parameter value	
24	arithmetic LT	
1B	AND	
12	3-byte binary value	
06	length of sequence symbol	
1C0E1A1C22162F	"SEQSYM"	
3A	AIF	

Note that the expression has been translated into reverse Polish notation and is in the form

```
ABC &C &A ( ) EQ &A &B*1 + &PARM2 LT AND
```

Example 2. S-type address constant, object module IPKJA.

The constant is defined in the following coding:

```
         USING *,5
A        DC  F'0'
SCON     EQU *
XYZ      DC  (SCON-A)SL(2*(L'A-3))((SCON-A)/2(L'*+L'SCON))
```

The statement is shown below as it is read in by object module IPKJA:

```
                         003F 000B1300   00040121 2223030A 0D0C2E0E 281C0C18
172A0A2D 1C152802 2B281530 0A2A032D   2D28281C 0C18172A 0A2D2C02 2815302B
2915301C 0C18172D 2D
```

The record can be interpreted as follows

003F	record length
00	file flag
0B	pseudo opcode (DC)
13	opcode extension
00	type flag
00	flag A
04	length of name
01	column pointer (source)
212223	"XYZ"
03	length of opcode field
0A	column pointer (source)
0D0C	"DC"
2E	length of operand field
0E	column pointer (source)
281C0...172D2D	operand (in internal assembler code)

Example 3. S-type address constant, object module IPKJA.

The same DC statement as in Example 2 is shown after processing by IPKJA:

```
                  00 6A000B00 00000400   00000000 00060321 2223431C 0C18172F
2F2F2F40 0A2F2F2F 2F2F2F2F 400A2F2F   2F2F2F2F 2F031C0C 18172F2F 2F2F000A
2F2F2F2F 2F2F2F03 1C0C1817 2F2F2F2F   010D3535 2A394038 02353438 032A2B39
443B0635 352A3802 2C393634 35342939   3A
```

The record can be interpreted as follows:

006A	record length
00	file flag
0B	pseudo opcode (DC)
00	opcode extension
00	flag A
0004	statement number
0000	length attribute (not filled in until IPKKA)
00	relocation attribute (not filled in until IPKKA)
000000	location counter value (not filled in until IPKKA)
06	number of symbol buckets
03	length of name field
212223	"XYZ"

4	flag (symbol must be previously defined)	symbol bucket 1
3	length -1 of symbol	
1C0C1817	"SCON"	
2F2F2F2F	four blanks	
4	flag (symbol must be previously defined)	symbol bucket 2
0	length -1 of symbol	
0A	"A"	
2F2F2F2F2F2F2F2F	blanks	
4	flag (symbol must be previously defined	symbol bucket 3
0	length -1 of symbol	
0A	"A"	
2F2F2F2F2F2F2F2F	blanks	
0		symbol bucket 4
3	length -1 of symbol	
1C0C1817	"SCON"	
2F2F2F2F	blanks	

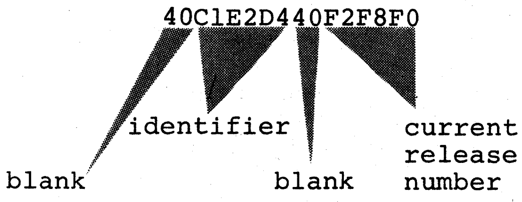
```
0
0            length -1 of symbol                                  }  symbol bucket 5
0A           "A"
2F2F2F2F2F2F2F blanks

0
3            length -1 of symbol                                  }  symbol bucket 6
1C0C1817     "SCON"
2F2F2F2F     blanks

01           number of constant
0D           type flag (S-type address constant)
35           symbol flag ("SCON")
35           symbol flag ("A" -- second bucket)
2A           - (subtraction)
39           end of expression
40           explicit length flag
38           self-defining term <256 bytes
02           "2"
35           symbol flag ("A" -- third bucket)
34           length attribute
38           self-defining term <256 bytes
03           "3"
2A           - (subtraction)
2B           * (multiplication)
39           end
44           DC exponent
3B           start of operand
06
35           symbol flag ("SCON" -- fourth bucket)
35           symbol flag ("A" -- fifth bucket)
2A           - (subtraction)
38           self-defining term <256 bytes
02           "2"
2C           / (division)
39           end of expression
36           location counter value
34           length attribute
35           symbol flag ("SCON" -- sixth bucket)
34           length attribute
29           + (addition)
39           end of expression
3A           end of operand
```

Note that the expressions have been translated into reverse Polish notation and are of the form

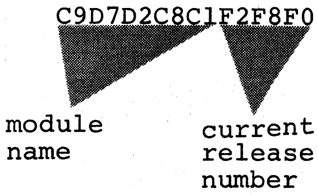
```
SCON A-
2A L'3-*
SCON A-2/
*L'SCON L'+
```

PROGRAM IDENTIFICATION

The current release number of the assembler can be found starting at the first byte of the object code in a program dump. For example:



Every assembler object module contains an identifier with the module name and current release number. The identifier is at the beginning of the module, starting at the first byte, and has the following format:



I/O Activity and Workfile Layouts

The following diagrams show the I/O activity for the phases of the assembler and the layouts of the workfiles during different operations. The following symbols are used in the diagrams:.

↗	a note/point address
⇓	an input operation
⇑	an output operation
⇕	both an input and output operation

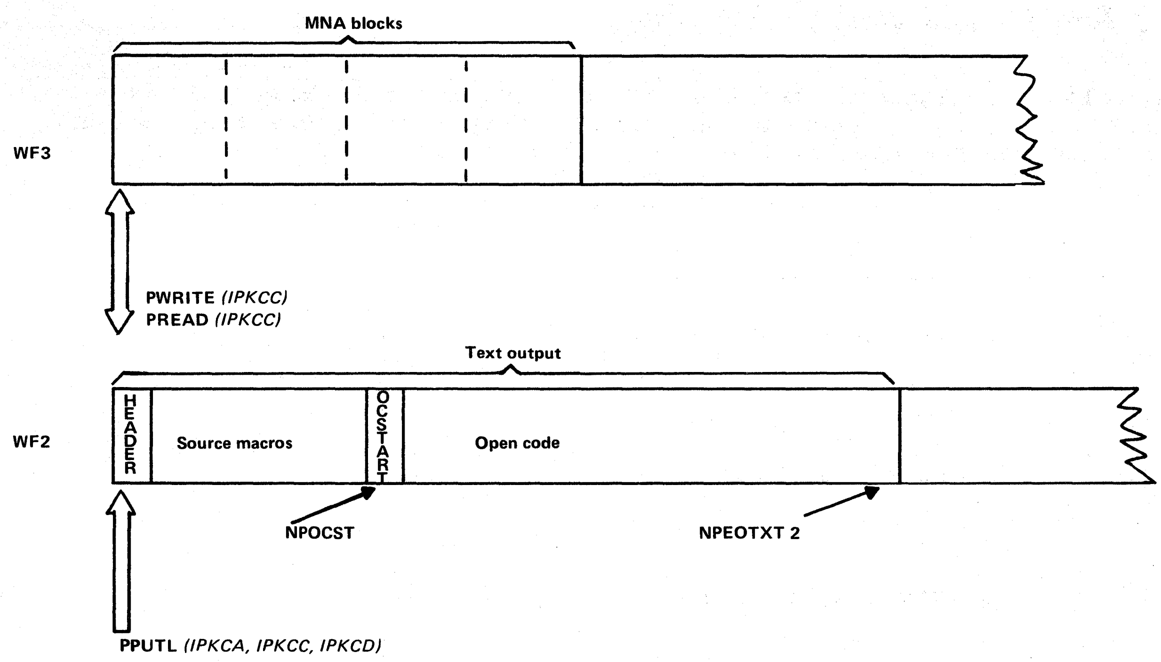
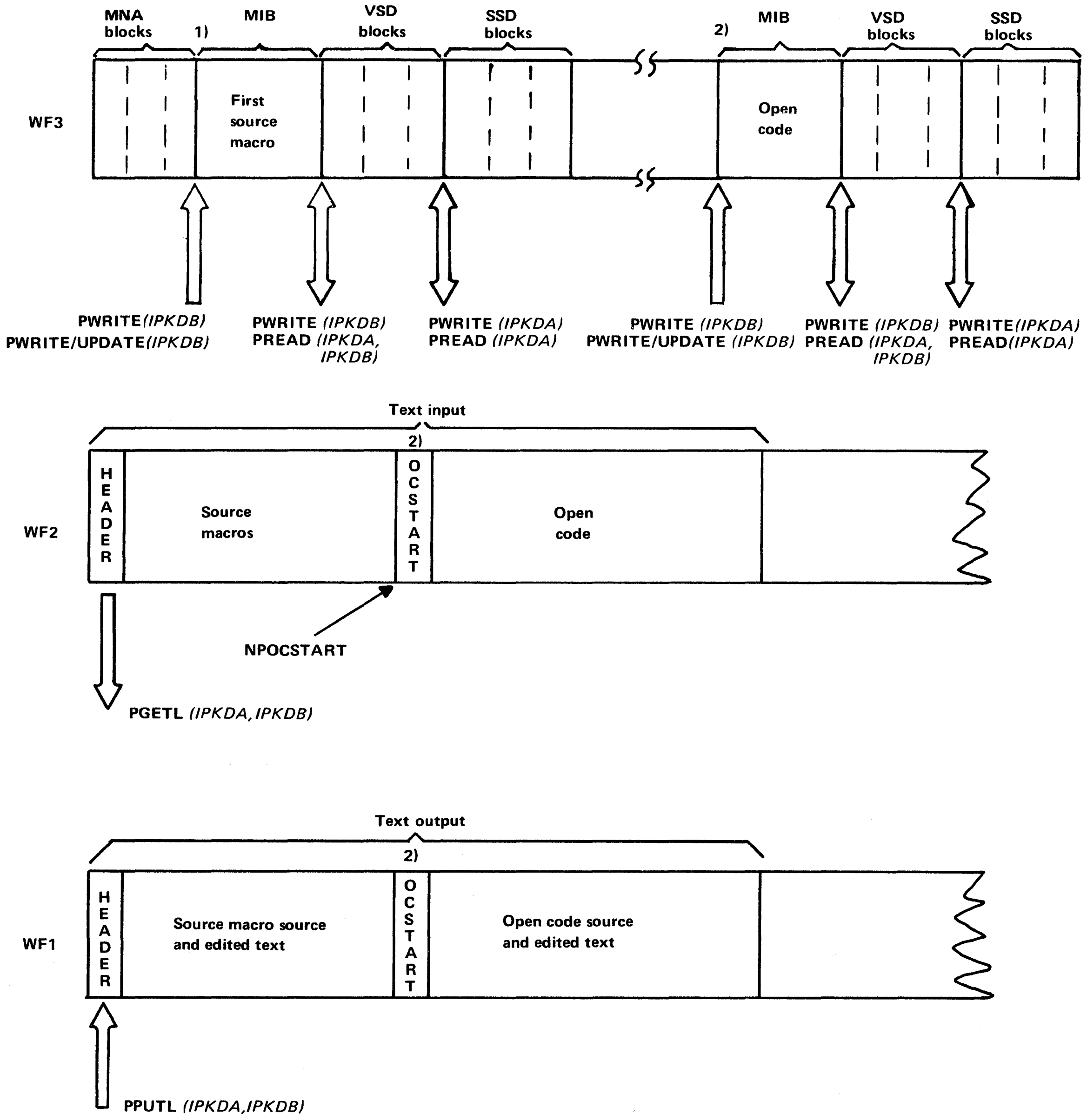


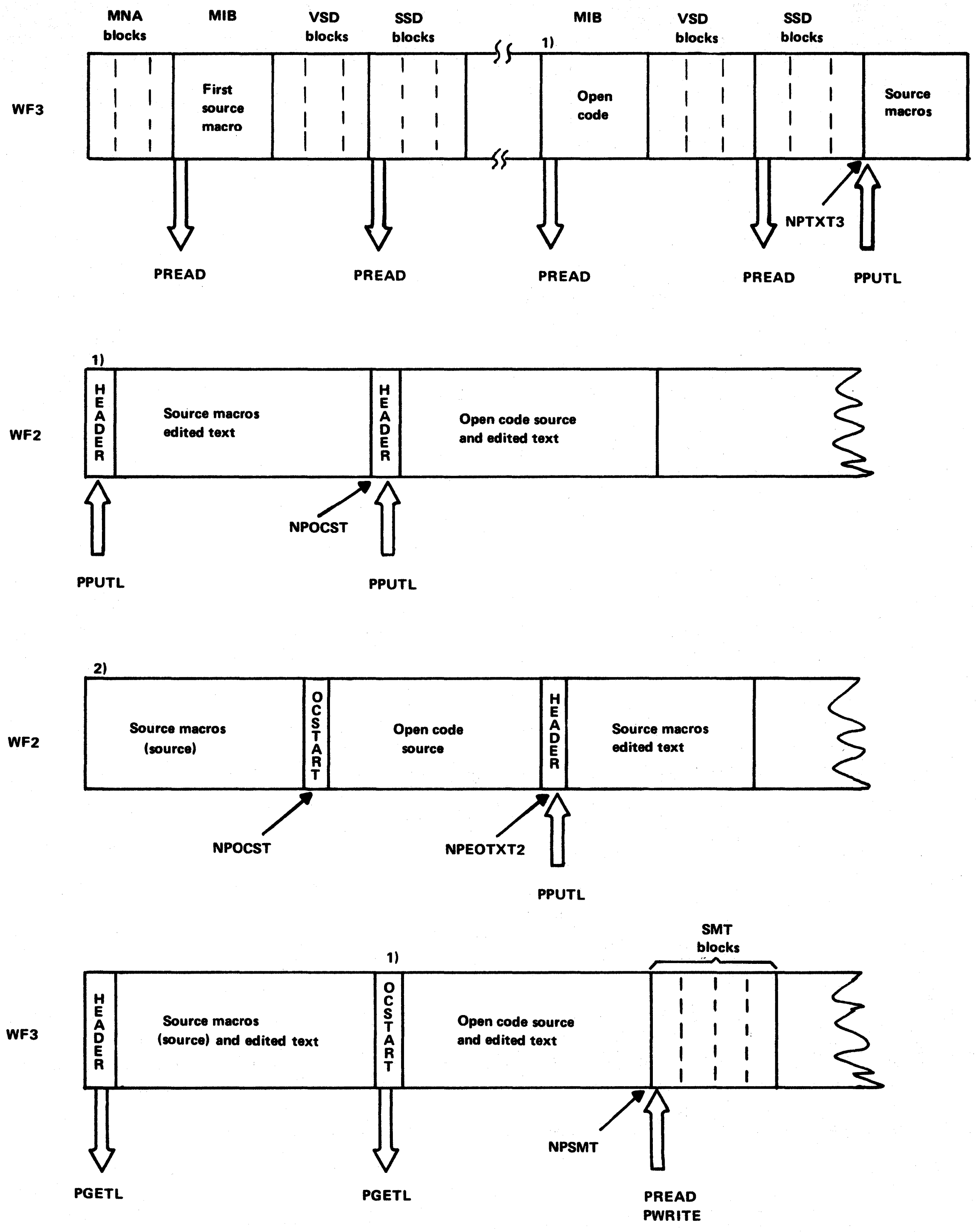
Figure 21. I/O Activity for ASSECA.



1. At the beginning of a macro, a dummy macro information block is written to reserve space. After the VSD and SSD blocks have been built, the complete macro information block is written into its reserved place with PWRITE/UPDATE.

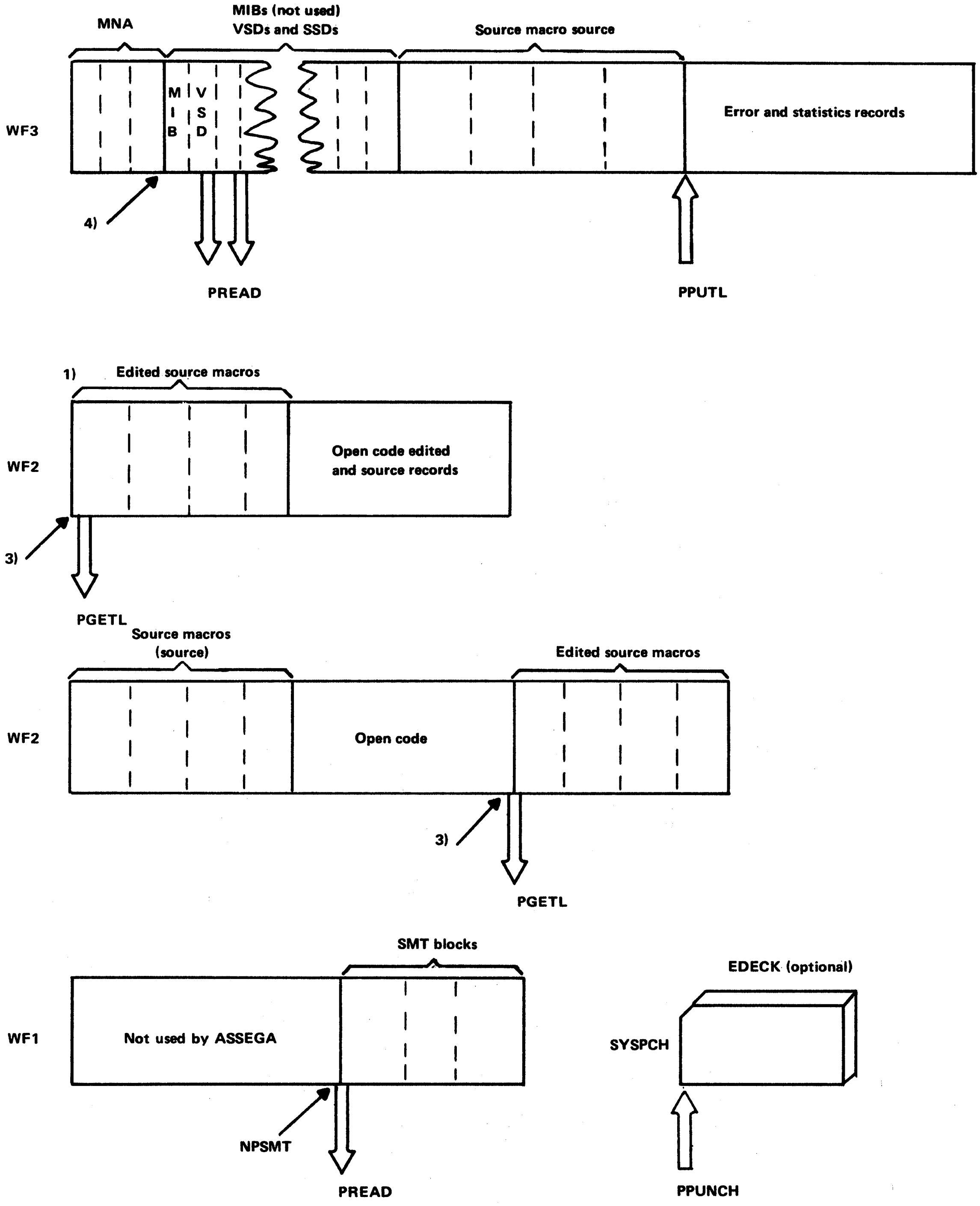
2. Open code is processed only if conditional assembly statements are detected by IPKCA.

Figure 22. I/O Activity for ASSEDA.



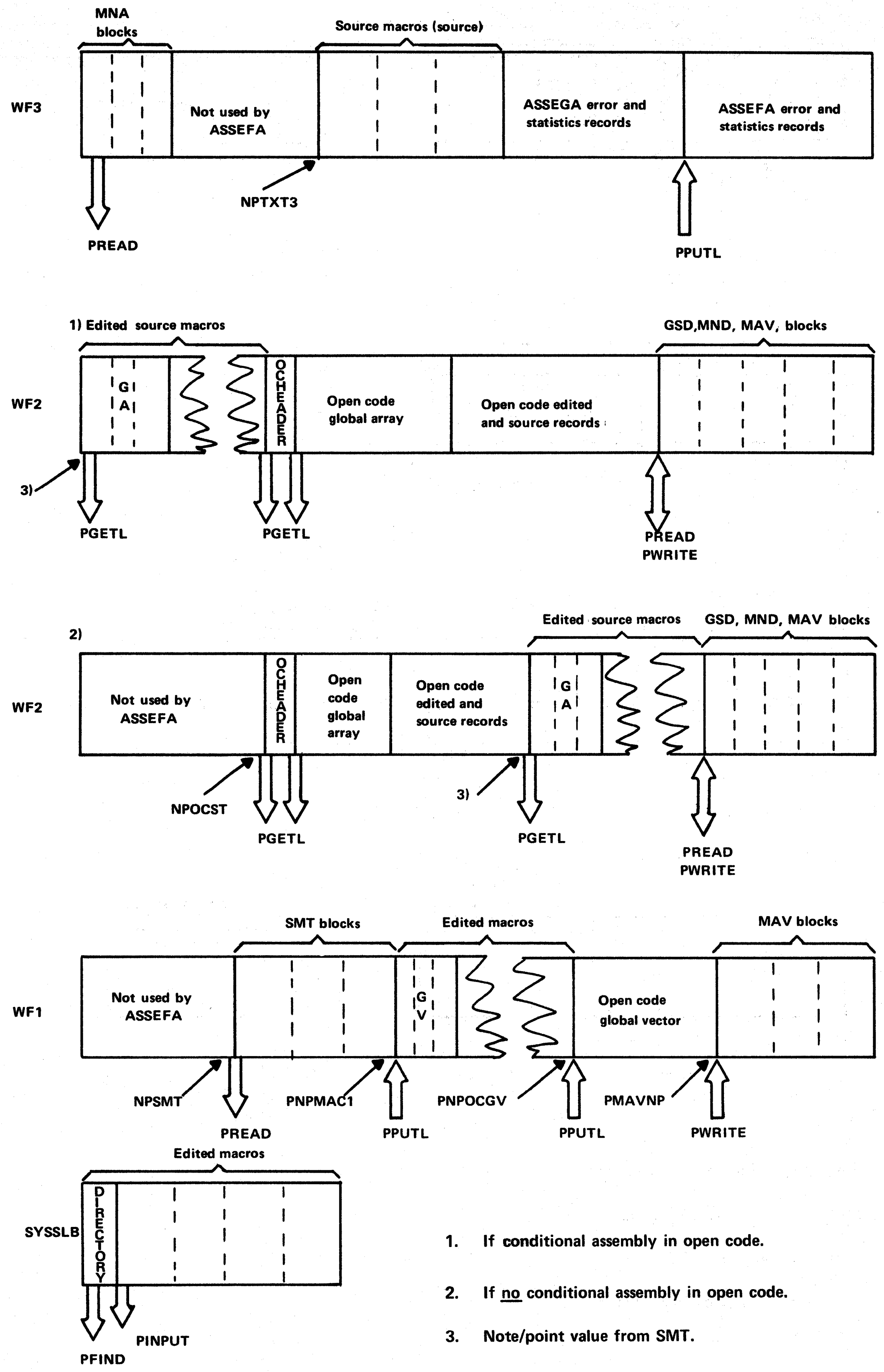
1. If conditional assembly in open code.
2. If <u>no</u> conditional assembly in open code.

Figure 23. I/O Activity for ASSEEA.



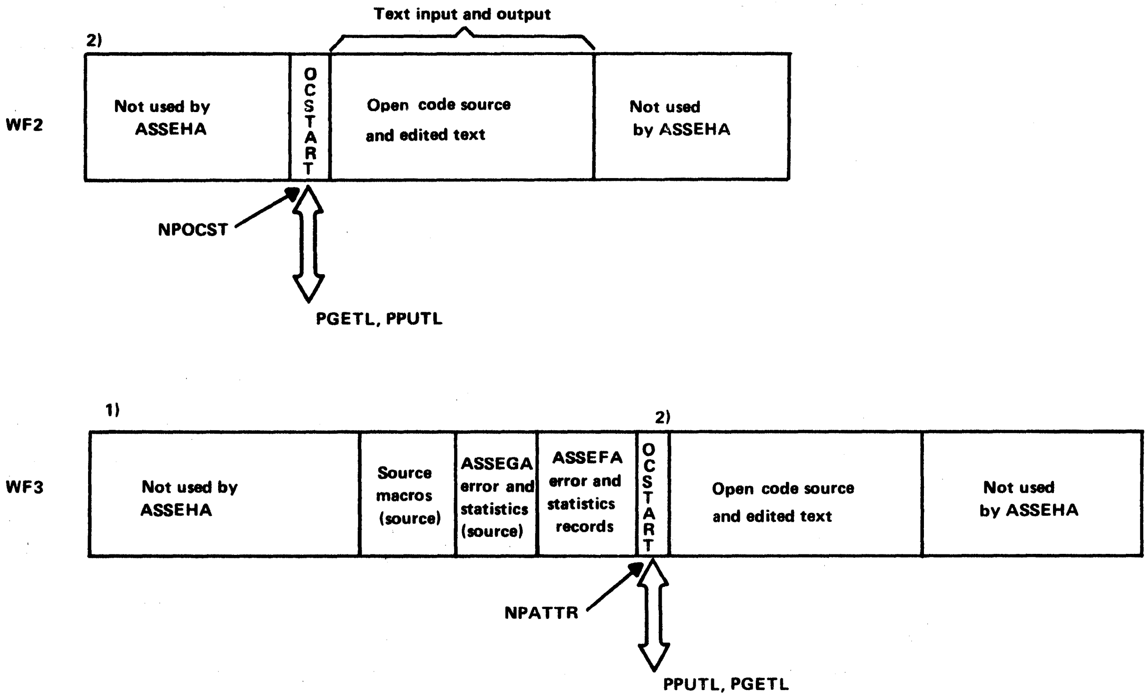
1. If conditional assembly in open code.
2. If no conditional assembly in open code.
3. Note/point value from SMT.
4. Note/point value from macro header.

Figure 24. I/O Activity for ASSEGA.



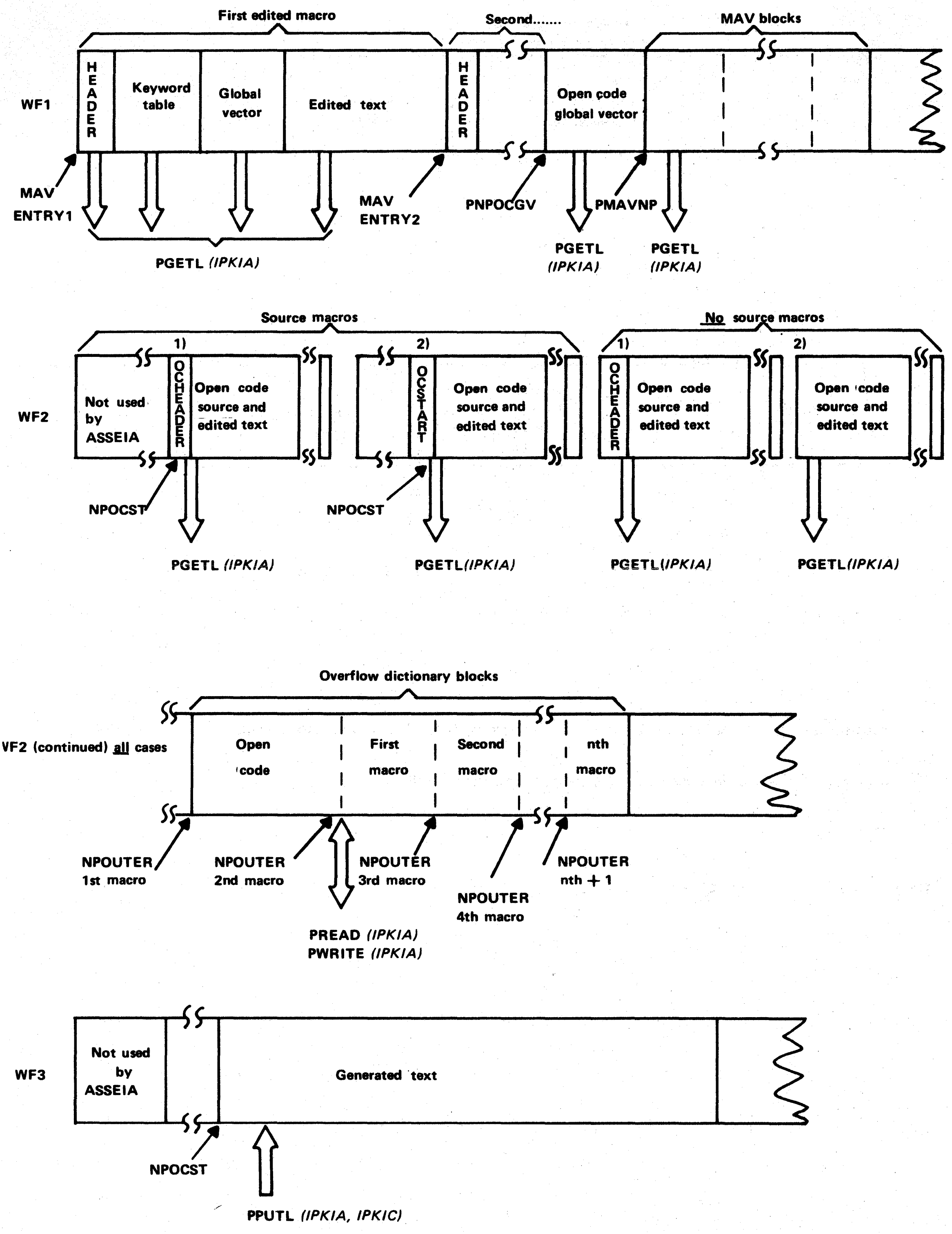
1. If conditional assembly in open code.
2. If no conditional assembly in open code.
3. Note/point value from SMT.

Figure 25. I/O Activity for ASSEFA.



1. Because of split records, ASSEHA uses WF3 to update records. The text is written back onto WF2 for ASSEFA.
2. If no source macros, the open code text begins at the start of WF2.

Figure 26. I/O Activity for ASSEHA.



1. If conditional assembly in open code.

2. If no conditional assembly in open code.

Figure 27. I/O Activity for ASSEIA.

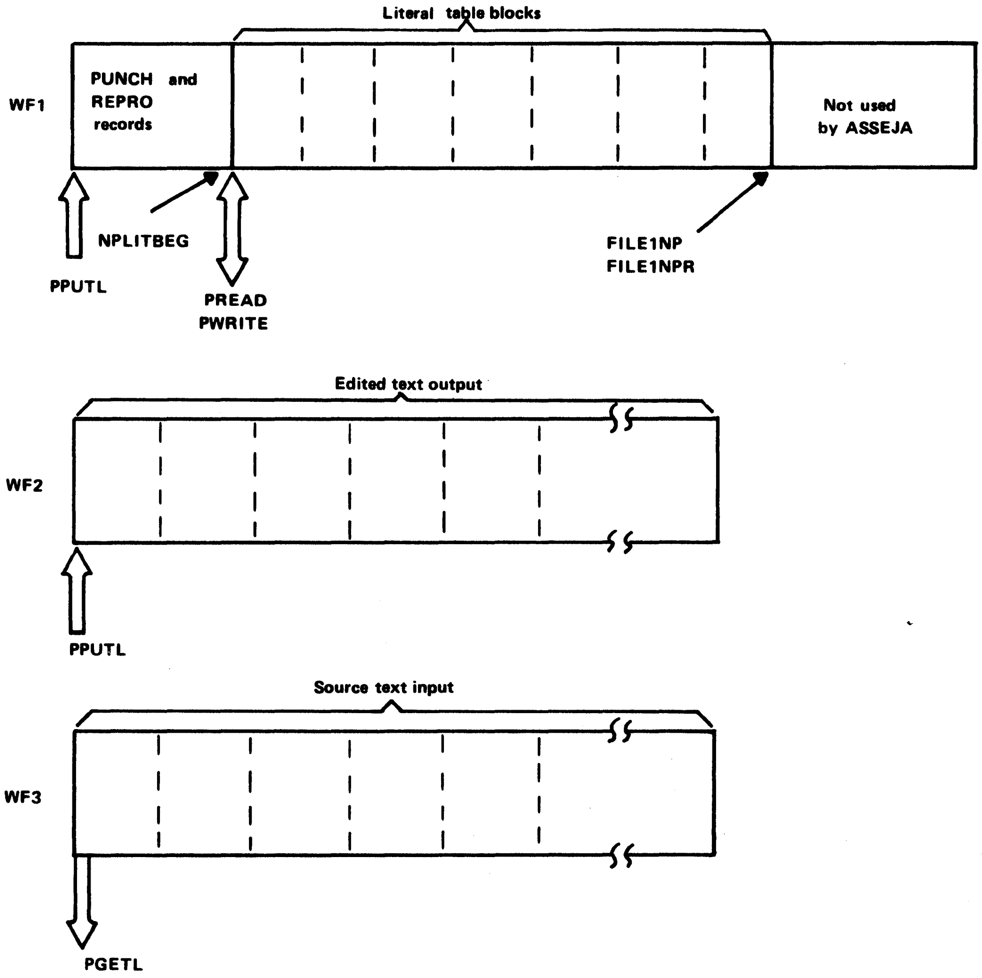
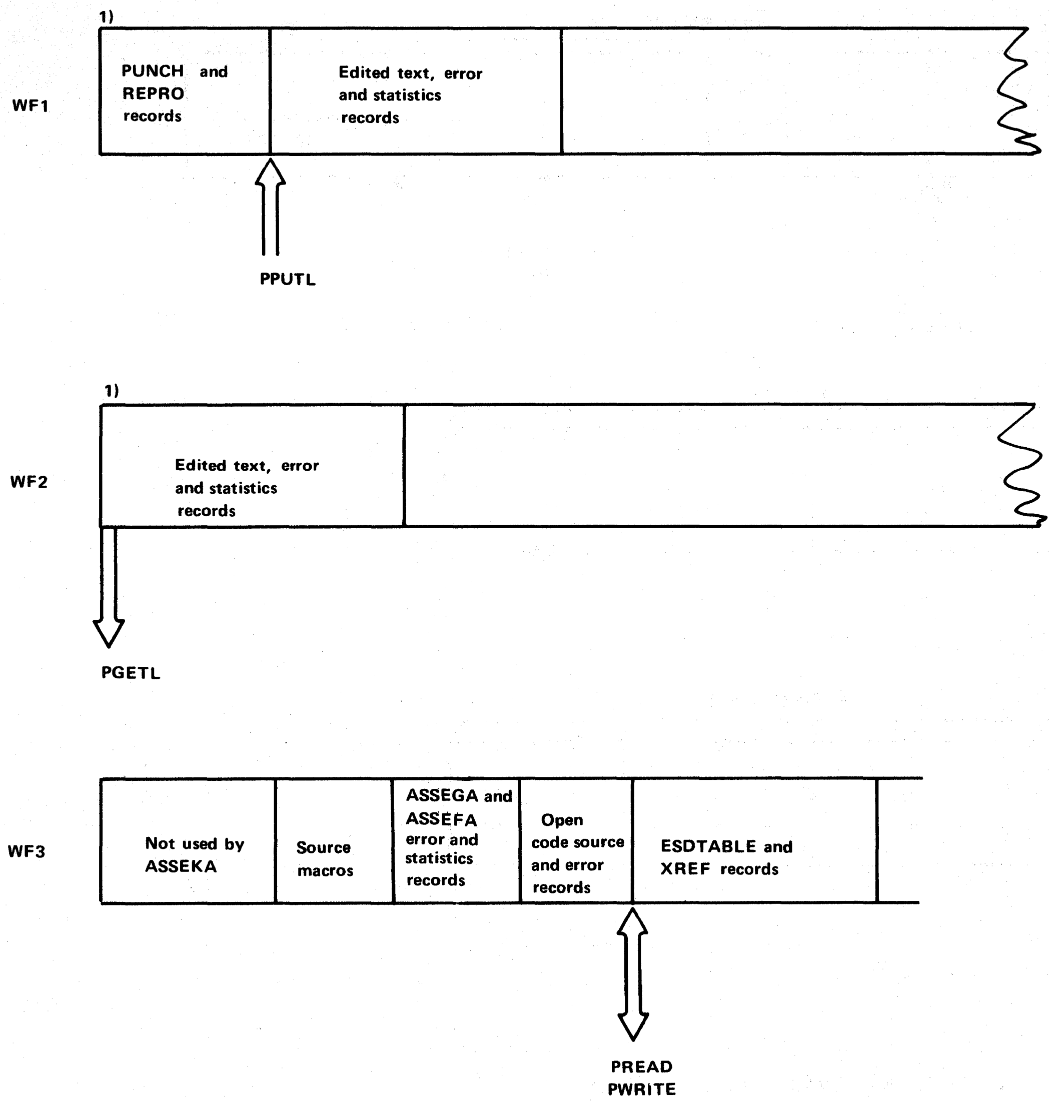
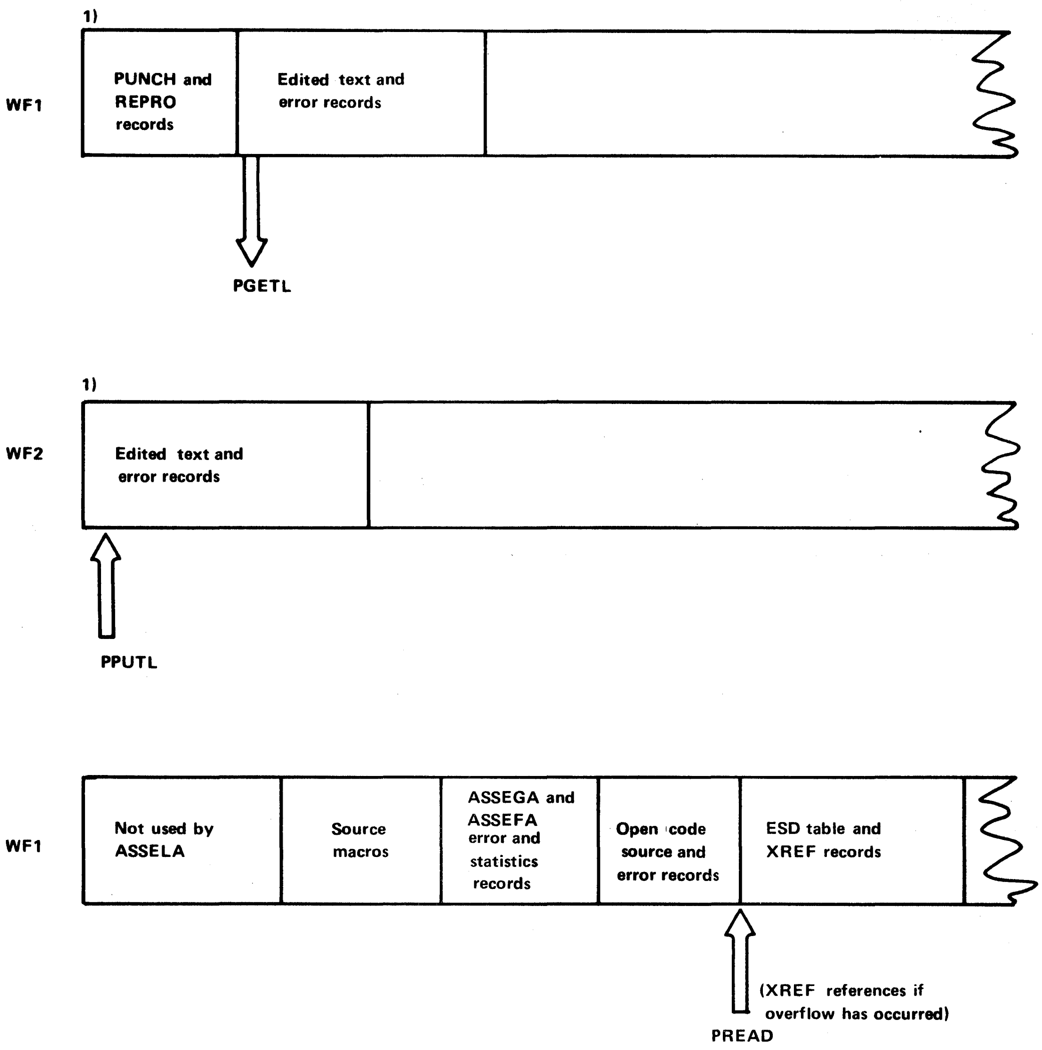


Figure 28. I/O Activity for ASSEJA.



1. If symbol table overflow occurs, workfile 1 and workfile 2 will exchange.

Figure 29. I/O Activity for ASSEKA.



1. If symbol table overflow occurred, workfile 1 and workfile 2 will be exchanged.

Figure 30. I/O Activity for ASSELA.

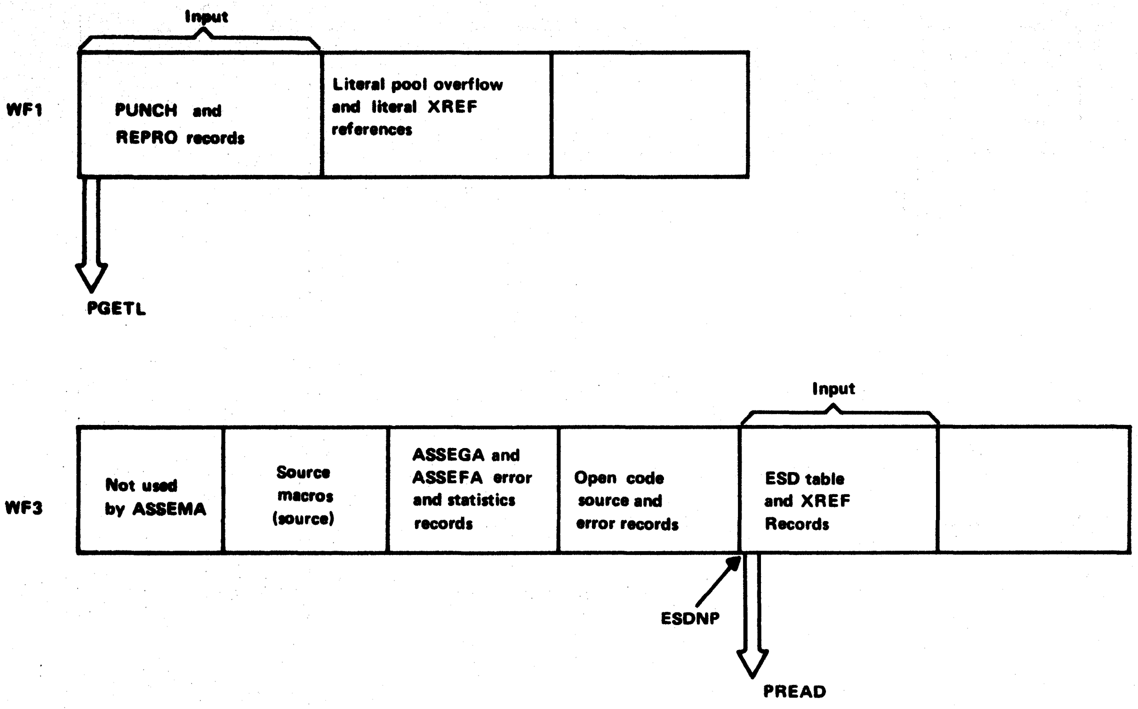
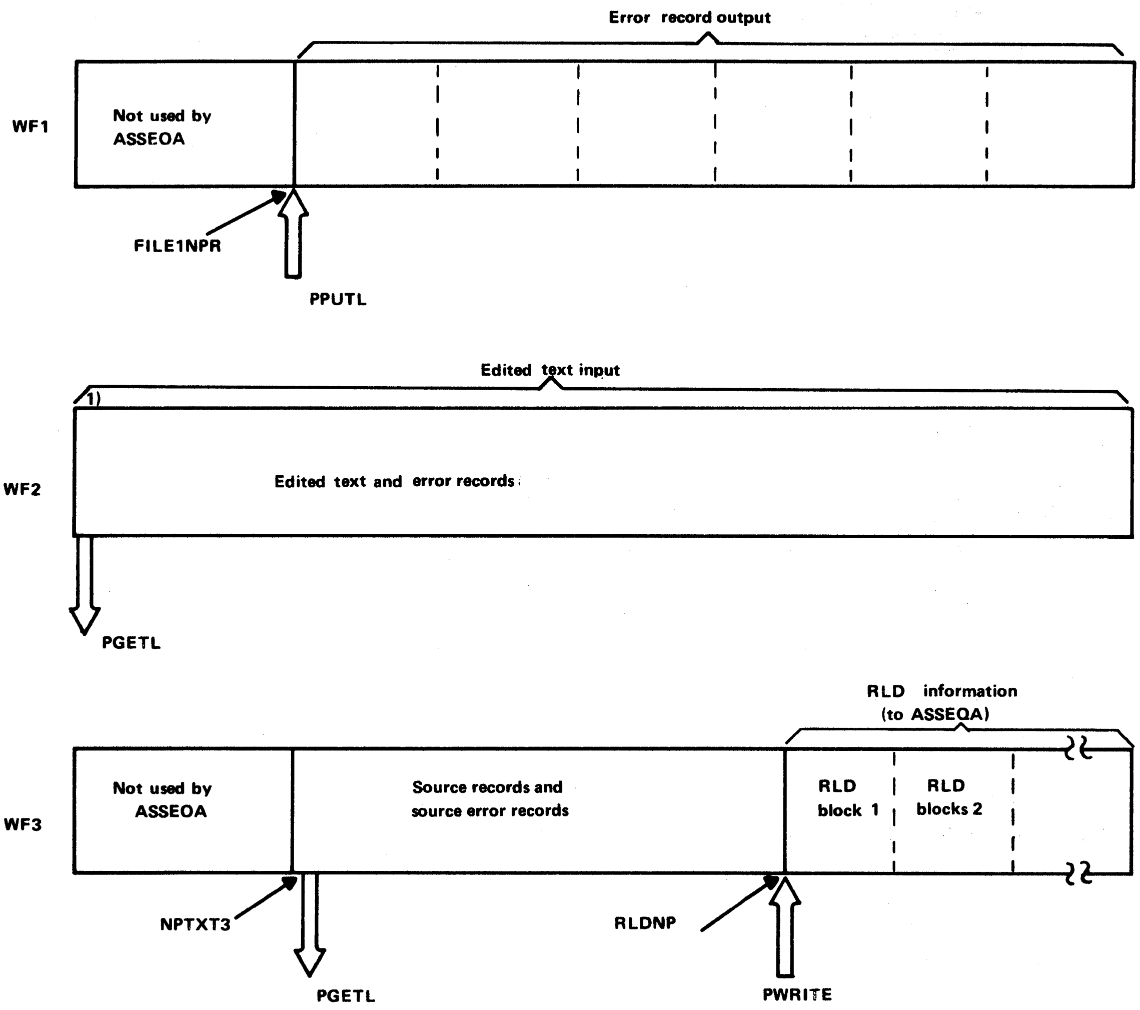


Figure 31. I/O Activity for ASSEMA.



1. The statement in error is followed by one or more error records (if present).

Figure 32. I/O Activity for ASSEOA.

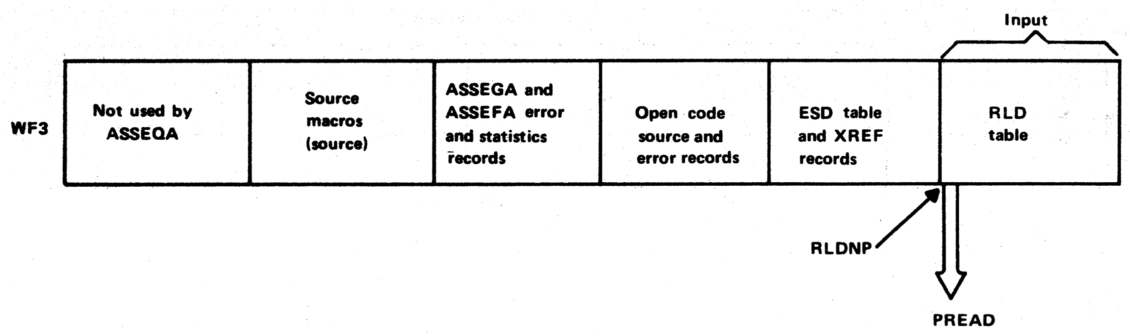


Figure 33. I/O Activity for ASSEQA.

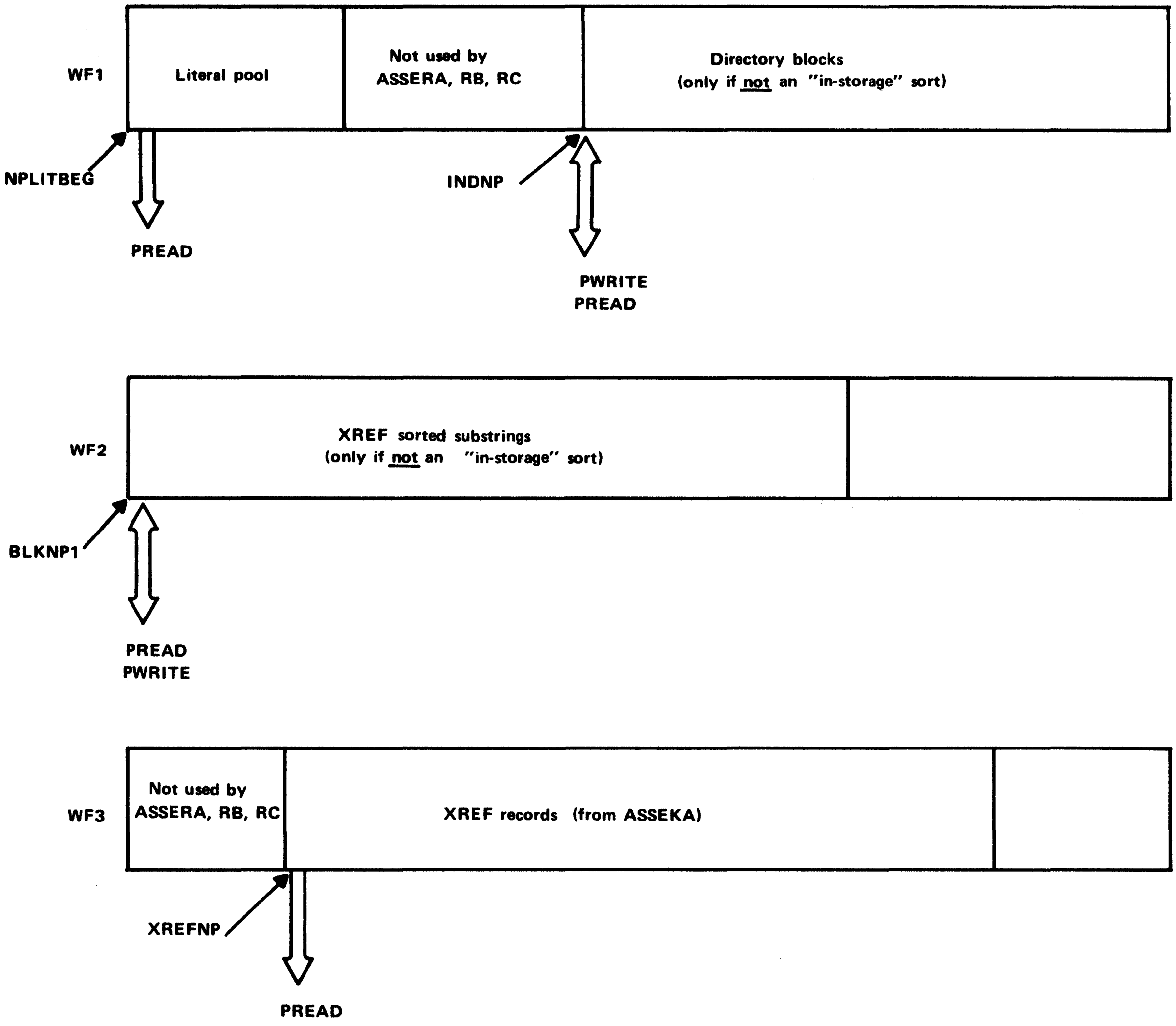


Figure 34. I/O Activity for ASSERA, ASSERB, ASSERC.

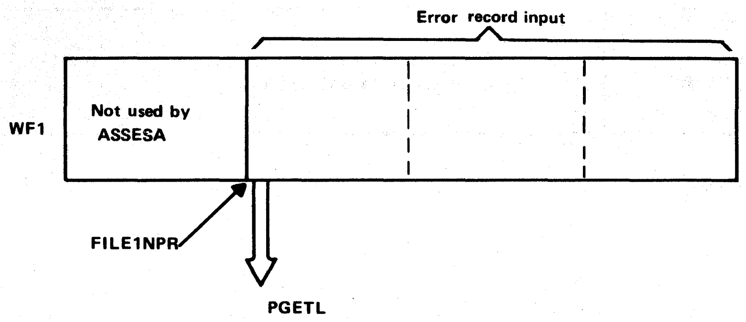


Figure 35. I/O Activity for ASSESA.

Register Usage for the Assembler

The following table shows the standard register usage for the assembler. Check the program listings for deviations from standard usage.

Register	Name	Usage
0-5		Work registers
6	RBRSAVE	Used by interface routine PSAVE as a branch register
7	RPARM	Used by some interface routines
8	RFILE	Used by interface routines, points to the file control block
9	RINPT	Input record pointer
10	ROUTPT	Output record pointer
11	RBA	First base register
12	RBB	Second base register
13	RBIF	Interface base register
14	RBR	Branch register
15		Work register

Note: Registers 0-15 are equated to R0-R15. All register names are equated to a second register name or to R0-R15; thus all references to a specific register can be found by using the cross-reference dictionary.

Figure 36. Register Usage.

Registers Routines	RBRSAVE	RPARM	RFILE[1]	RINPT	ROUTPT	RBR	R15,R0,R1
PCALL						X	
PCHECK	X		X			X	
PCLOSI			X				
PCLOSO	X	X	X		X	X	
PFETCH						X	
PFIND	X	X[3]				X	
PGETL	X		X	X		X	
PINPUT	X			X		X	
PLOAD						X	X
PNOTE	X[2]	X[2]	X			X[2]	
POPSAVE						X	
PPOINT	X	X[3]	X			X	
PPRINT	X	X				X	
PPUNCH	X	X				X	
PPUTL	X		X		X	X	
PREAD	X	X[3]	X			X	
PRETURN						X[4]	
PSAVE	X						
PWRITE	X	X[3]	X			X	

[1] RFILE is not changed if the FILE parameter is not specified
[2] Not changed if the USE=GET parameter is specified
[3] Not changed if neither the PARM nor the ADDR parameter is specified
[4] Not changed if the NOPOP parameter is specified

Figure 37. Registers Changed by Interface Routine Operation.

Appendixes

This section contains the following information:

- Appendix A. Diagnostic Message Number/Module/Diagram Cross-Reference
- Appendix B. Module/Entry Symbol/EXTRN Symbol Cross-Reference
- Appendix C. Macro and COPY Code Usage
- Appendix D. Element Formats
- Appendix E. Pseudo (Internal) Operation Codes
- Appendix F. Internal Character Set
- Appendix G. Edited Text Flags
- Appendix H. Edited Statement Formats
- Appendix I. Statements Modifying Data Areas
- Appendix J. APAR Documentation for the Assembler

Appendix A: Diagnostic Message Number/Module/ Diagram Cross-Reference

The following cross-reference list contains all the message numbers for the assembler (those marked "not used" are reserved for future use by the assembler). The list also contains the module(s) in which these messages might appear and the method-of-operation diagram(s) for the module(s). For more detailed information about the messages see <u>Guide to the DOS/VS Assembler</u>.

Message Number	Module	Method of Operation Diagram
IPK001	IPKCA	1.1.1
IPK002	IPKCA	"
IPK003	IPKCA	"
IPK004	IPKCA	"
IPK006	IPKCA,IPKDA	1.1.1, 1.1.2
IPK005	IPKCA,IPKDA,IPKDB	1.1.1, 1.1.2
IPK007	IPKCA,IPKCC	1.1.1
IPK008	IPKCA,IPKDA	1.1.1, 1.1.2
IPK009	IPKCA	1.1.1
IPK010	IPKCA	1.1.1
IPK011	IPKCA	"
IPK012	IPKCA	"
IPK013	IPKCA	"
IPK014	IPKCC	"
IPK015	IPKCC	"
IPK016	IPKCC	"
IPK017	IPKCC	"
IPK018	IPKCC	"
IPK019	IPKCC	"
IPK020	IPKCC	"
IPK021	IPKCC	"
IPK022	IPKCC	"
IPK023	IPKCC	"
IPK024	IPKCC	"
IPK025	IPKCC	"
IPK026	IPKCC	"
IPK027	IPKCC	"
IPK028	IPKCC	"
IPK029	IPKCC	"
IPK030	not used	
IPK031	IPKCD,IPKDB	1.1.1, 1.1.2
IPK032	IPKCD	1.1.1
IPK033	IPKCD	"
IPK034	IPKCD	"
IPK035	IPKCD	"
IPK036	IPKCD	"
IPK037	IPKCD	"
IPK038	IPKCD	"
IPK039	IPKCD	"
IPK040	IPKCD	"
IPK041	IPKCD	"
IPK042	IPKDA	1.1.2
IPK043	IPKDA	"
IPK044	IPKDA	"

Figure 38. Diagnostic Message Number/Module/Diagram Cross-Reference (Part 1 of 5)

Message Number	Module	Method of Operation Diagram
IPK045	IPKDA	1.1.2
IPK046	IPKDA	"
IPK047	IPKDA	"
IPK048	IPKDA	"
IPK049	IPKDA	"
IPK050	IPKDA	"
IPK051	IPKDA	"
IPK052	IPKDA	"
IPK053	IPKDA	"
IPK054	IPKDA	"
IPK055	IPKDA	"
IPK056	IPKDA	"
IPK057	IPKDA	"
IPK058	IPKDA	"
IPK059	IPKDA	"
IPK060	IPKDA	"
IPK061	IPKDA	"
IPK062	IPKDA	"
IPK063	IPKDA	"
IPK064	IPKDA	"
IPK065	IPKDA	"
IPK066	IPKDA	"
IPK067	IPKDA	"
IPK068	IPKDA	"
IPK069	IPKEA	1.1.3
IPK070	IPKEA	1.1.3
IPK071	IPKHA	1.4
IPK072	IPKHA	1.4
IPK073	IPKHA	1.4
IPK074	IPKIA	1.5
IPK075	IPKIA	"
IPK076	IPKIA	"
IPK077	IPKIA	"
IPK078	IPKIA	"
IPK079	IPKIA	"
IPK080	IPKIA	"
IPK081	IPKIA	"
IPK082	IPKIA	"
IPK083	IPKIA	"
IPK084	IPKIA	"
IPK085	IPKIA	"
IPK086	IPKIA	"
IPK087	IPKIA	"
IPK088	IPKIA	"
IPK089	IPKIA	"
IPK090	IPKIA	"
IPK091	IPKIA	"
IPK092	IPKIA	"
IPK093	IPKIA	"
IPK094	IPKIA	"
IPK095	IPKIA	"
IPK096	IPKIA	"
IPK097	IPKIA	"
IPK098	IPKIA	"
IPK099	IPKIA	"
IPK100	IPKIA	"
IPK101	IPKIA	"
IPK102	IPKIA	"

Figure 38. Diagnostic Message Number/Module/Diagram Cross-Reference **(Part 2 of 5)**

Message Number	Module	Method of Operation Diagram
IPK103	IPKIA	1.5
IPK104	IPKIC	"
IPK105	IPKIC	"
IPK106	IPKIC	"
IPK107	IPKIA	"
IPK108	IPKIA	"
IPK109	IPKJA	"
IPK110	IPKJA	"
IPK111	IPKJA	"
IPK112	IPKJA	"
IPK113	IPKJA	"
IPK114	IPKJA	"
IPK115	IPKJA	"
IPK116	IPKJA	"
IPK117	IPKJA	"
IPK118	IPKJA	"
IPK119	IPKJA	"
IPK120	IPKJA	"
IPK121	IPKJA	"
IPK122	IPKJA	"
IPK123	IPKJA	"
IPK124	IPKJA	"
IPK125	IPKJA	"
IPK126	IPKJA	"
IPK127	IPKJA	"
IPK128	IPKJA	"
IPK129	IPKJA	"
IPK130	IPKJA	"
IPK131	IPKJA	"
IPK132	IPKJA	"
IPK133	IPKJA	"
IPK134	IPKJA	"
IPK135	IPKKA,IPKLA	2.2, 2.3
IPK136	IPKKA,IPKLA	2.2, 2.3
IPK137	IPKKA	2.2
IPK138	IPKKA	2.2
IPK139	IPKKA	"
IPK140	IPKKA	"
IPK141	IPKKA	"
IPK142	IPKKA	"
IPK143	IPKKA	"
IPK144	IPKKA,IPKLA	2.2, 2.3
IPK145	IPKKA,IPKLA	2.2, 2.3
IPK146	IPKKA	2.2
IPK147	IPKKA	"
IPK148	IPKKA,IPKLA	2.2, 2.3
IPK149	IPKKA	2.2
IPK150	IPKKA	"
IPK151	IPKKA	"
IPK152	IPKKA	"
IPK153	IPKKA	"
IPK154	IPKKA	"
IPK155	IPKKA	"
IPK156	IPKLA	2.3
IPK157	IPKLA	2.3
IPK158	IPKNA	2.4
IPK159	IPKNA	"
IPK160	IPKNA	2.4

Figure 38. Diagnostic Message Number/Module/Diagram Cross-Reference (Part 3 of 5)

Message Number	Module	Method of Operation Diagram
IPK161	IPKNA	2.4
IPK162	IPKNA	"
IPK163	IPKNA	"
IPK164	IPKNA	"
IPK165	IPKNA	"
IPK166	IPKNA	"
IPK167	IPKNA	"
IPK168	IPKNA	"
IPK169	IPKNA	"
IPK170	IPKNA	"
IPK171	IPKNA	"
IPK172	IPKNA	"
IPK173	IPKNA	"
IPK174	IPKNA	"
IPK175	IPKNA	"
IPK176	IPKNA	"
IPK177	IPKNA	"
IPK178	IPKNA	"
IPK179	IPKNA	"
IPK180	IPKNA	"
IPK181	IPKNA	"
IPK182	IPKNA	"
IPK183	IPKNA	"
IPK184	IPKOA	2.6, 2.7.1
IPK185	IPKOA	2.6, 2.7.1
IPK186	IPKOA	2.6, 2.7.1
IPK187	IPKOA	2.6, 2.7.1
IPK188	IPKOA	2.6, 2.7.1, 2.4.3
IPK189	IPKOA,IPKNB	2.6, 2.7.1, 2.4.3
IPK190	IPKNA	2.4.3
IPK191	IPKNB	"
IPK192	IPKNB	"
IPK193	IPKNB	"
IPK194	IPKNB	"
IPK195	IPKNB	"
IPK196	IPKNB	"
IPK197	IPKNB	"
IPK198	IPKNB	"
IPK199	IPKNB	"
IPK200	IPKPA	2.7
IPK201	IPKPA	"
IPK202	IPKPA	"
IPK203	IPKPA	"
IPK204	IPKPA	"
IPK205	IPKPA	"
IPK206	IPKPA	"
IPK207	IPKPA	"
IPK208	IPKPA	"
IPK209	IPKPA	"
IPK210	not used	"
IPK211	IPKPA	"
IPK212	IPKPA	"
IPK213	IPKPA	"
IPK214	IPKPA	"
IPK215	IPKPA	"
IPK216	IPKPA	"
IPK217	IPKPA	"

Figure 38. Diagnostic Message Number/Module/Diagram Cross-Reference (Part 4 of 5)

Message Number	Module	Method of Operation Diagram
IPK218	IPKPA	2.7
IPK219	not used	
IPK220	not used	
IPK221	not used	
IPK222	not used	
IPK223	not used	
IPK224	not used	
IPK225	not used	
IPK226	not used	
IPK227	not used	
IPK228	not used	
IPK229	not used	
IPK230	IPKBA	3
IPK231	IPKBA	3
IPK232	IPKBA	3
IPK233	IPKBA	3
IPK234	IPKBA	3
IPK235	IPKBA	3
IPK236	IPKBA	3
IPK237	not used	
IPK238	not used	
IPK239	not used	
IPK240	IPKFA	1.3
IPK241	IPKFA	"
IPK242	IPKFA	"
IPK243	IPKFA	"
IPK244	IPKFA	"
IPK245	IPKFA	"
IPK246	IPKFA	"
IPK247	IPKFA	"
IPK248	IPKFA	"
IPK249	not used	
IPK250	IPKGA	1.2
IPK251	not used	
IPK252	not used	
IPK253	not used	
IPK254	not used	
IPK255	not used	

Figure 38. Diagnostic Message Number/Module/Diagram Cross-Reference (Part 5 of 5)

Appendix B: Module/Entry Symbol/EXTRN Symbol Cross-Reference

The following cross-reference list contains all the Entry symbol numbers and EXTRN symbol numbers used by the modules of the assembler.

Module	Entry Symbol	EXTRN Symbol
IPKAA	IPKAA101-103 IPKAA501-509 IPKAA511-12	IPKBG001 IPKAA002 IPKAI101
IPKAB	IPKAB101-103	IJJCPDV2 IPKAD100
IPKAC	IPKAC100-101	IJJCPDV
IPKAD	IPKAD101	IJJCPD3 IPKAB103
IPKAE	IPKAE101-103	IPKAD100
IPKAF	IPKAF100-102	IJJCPDV IJJCPDV1
IPKAH	IJJCPDV1	
IPKAI	IPKAI100-101	IJJCPDV1
IPKBA		IPKCD999 IPKDB999 IPKEA999 IPKFA999 IPKIC999 IPKJA999 IPKKA999 IPKNA999 IPKPA999 IPKAA000 IPKAA101-103 IPKAB100-102 IPKAD101
IPKCA	IPKCA998 IPKCA210	IPKCB001 IPKCC203 IPKCD203
IPKCB	IPKCB001 IPKCB999	
IPKCC	IPKCC203 IPKCC999	IPKCA998 IPKCA210 IPKCD203
IPKCD	IPKCD203 IPKCD999	IPKCA998 IPKCA210
IPKDA	IPKDA203 IPKDA999	IPKDB203
IPKDB	IPKDB203 IPKDB999	IPKDA203
IPKEA	IPKEA999	
IPKFA	IPKFA999	IPKAE101-103
IPKGA		IPKAC100-101
IPKIA	IPKIA201 IPKIA999	IPKIC201 IPKIC999
IPKIC	IPKIC201 IPKIC999	IPKCB001 IPKIA201
IPKJA	IPKJA001 IPKJA999	
IPKKA	IPKKA999	
IPKLA	IPKLA001 IPKLA002	IPKNA001 IPKNB100 IPKNB200

Figure 39. Module/Entry Symbol/EXTRN Symbol Cross-Reference (Part 1 of 2)

Module	Entry Symbol	EXTRN Symbol
IPKNA	IPKNA001	IPKLA002
	IPKNA999	
	IPKNB100	
	IPKNB200	
IPKMA		IPKAF100-102
		IPKAI100
IPKOA	IPKOA100	IPKPA100
	IPKOA200	IPKPA200
	IPKOA999	IPKPA250
		IPKPA300
		IPKPA350
IPKPA	IPKPA100	IPKOA100
	IPKPA200	IPKOA200
	IPKPA250	
	IPKPA300	
	IPKPA350	
	IPKPA999	
IPKQA		IPKAF101-102
IPKRA	IPKRA100	IPKRA001
	IPKRA200	IPKRA100
	IPKRA999	IPKRA200
		IPKRA999
IPKRB	IPKRB100	IPKRA001
	IPKRB200	IPKRA100
	IPKRB999	IPKRA200
IPKRC	IPKRC100	IPKRA001
	IPKRC200	IPKRA100
	IPKRC999	IPKRA200
IPKSA	IPKSA999	IPKSB100
IPKSB	IPKSB100	IPKAI100
	IPKSB999	IPKAJ000
IPKTA		IPKAA101-103
		IPKAB100
		IJJCPD1N

Figure 39. Module/Entry Symbol/EXTRN Symbol Cross-Reference (Part 2 of 2)

Appendix C: Macro and COPY Code Usage

This appendix describes the internal macro instructions and COPY code used by the assembler. The figure describing macro usage lists the object modules in which the instructions appear, gives a short functional description of the macro, and illustrates the instruction formats. The figure showing COPY code usage contains the object modules using the COPY books and a short description of the COPY book's contents.

Macro Name	Used By	Description
BACK	IPKCA IPKCC	Decrements a register by one. Format: [name] BACK opndl where opndl specifies a register
BUMPP	All modules except IPKAC-AI IPKRA-RC IPKNA IPKPA IPKOA IPKSA-SB	Adds to opndl the value specified in expn. Format: [name] BUMP opndl, expn where opndl specifies a register expn is any valid expression allowed in the operand field of an EQU statement
CHECK*	IPKAA	Waits (if necessary) for the completion of a READ or WRITE operation and detects any errors or exceptional conditions.
CLOSE*	IPKGA IPKSB IPKQA IPKTA	Deactivates any file that was previously opened in any input/output unit in the system.
COMRG*	IPKBA IPKFA IPKPA IPKSB	Allows the problem program to communicate with the supervisor by placing the address of the appropriate region in register 1.
CPMOD*	IPKAG IPKTA IPKAH	Device-independent logic module. For a description of this macro see DOS/VS LIOCS Vol. 2 SAM Logic, Order No. SY33-8560.
* system macro		

Figure 40. Macro Usage (Part 1 of 10)

Macro Name	Used By	Description
DBV	IPKKA	Symbols declared by a DBV macro may be set and interrogated with the SET and GOIF macros. DBV is used to reserve one byte of storage and to assign symbolic values with that byte. Format: [name] DBV [sym1(exp1)] [,sym2(exp2)]...... where symn is any valid symbol up to seven characters expn is any valid expression allowed in the operand field of an EQU statement
DEFSTACK	IPKJA	Defines an error stack. Format: [name] DEFSTACK opnd1,opnd2,opnd3=n3,opnd4=n4 where opnd1 defines the name of the generated stack opnd2 defines the register used as base register for the stack opnd3=n3 where n3 specifies the length of each entry in the stack opnd4=n4 where n4 specifies the number of entries in the stack
DSW	all modules except IPKAD IPKAG IPKAH	A switch must be declared with a DSW macro prior to its use in a GOIF or SET macro. If no name is specified, a name is generated internally. The resulting code is a one byte DC X'00' with equates for the declared switches. Format: [name] DSW [sw1] [,sw2] ... [,sw8] where swn is from 1 to 8 switches; omitted operands result in unused bytes
DTFCP*	IPKAB IPKAI IPKAC IPKTA IPKAF	DTF for device-independent files. For a description of this macro, see DOS/VS LIOCS Vol. 2 SAM Logic, Order No. SY33-8560.
DTFSD*	IPKAA	Defines central processing for a file contained on a DASD.
DTFSL*	IPKAD	Retrieves books (macros or COPY code) from the system and/or private source statement library (SYSSLB).
EOJ*	IPKSB IPKTA	Informs the system that a job step is finished.
EXCP*	IPKAA	Requests physical IOCS to start an I/O operation for a particular device.
FNDSL*	IPKAB IPKAE	Searches the system and private library directories for a specified book name.
GET*	IPKAB IPKAE	Reads logical records.
GETIME*	IPKPA	Obtains the time of day during program execution.
GETSL*	IPKAE	Reads a block of books and expands compressed core image.
GOEOP**	IPKDA	If the end of the input string is encountered, a branch is made in the pseudo-op table to the entry designated by opnd1. Otherwise, the next sequential pseudo-op entry is read. Generates a two-byte DC in the pseudo-op table. Format: [name] GOEOP opnd1

* system macro
** source macro

Figure 40. Macro Usage (Part 2 of 10)

Macro Name	Used By	Description
GOIF	all modules except IPKAD IPKAE IPKAG IPKAH	Generates conditional branch logic. __Format__: [name] GOIF opnd1 [,opnd2] [,cond1=loc1] [,cond2=loc2] ... MODE=$\left\{\begin{matrix}A\\ \underline{L}\end{matrix}\right\}$ where opnd1 may be a switch or a sublist of switches declared in a DSW macro; a symbolic value defined by a DBV macro; a register (in parentheses); or a storage address. opnd2 may be a value that can be used as an immediate operand in an SI instruction; a register (in parenthesis); or a storage address condn may be one or more keywords of appropriate type locn may be a register (in parenthesis); or a storage address
GOIFCAR**	IPKDA	Compares the current character in the input string (pointed to by register 9) with the character designated by opnd1. If they match, a branch is made in the pseudo-op table to the entry designated by opnd2. On a mismatch, the next sequential pseudo-op entry is read. Generates a two-byte DC in the pseudo-op table. __Format__: [name] GOIFCAR opnd1, opnd2
GOIFCAT**	IPKDA	Compares the input string category (pointed to by register 9) with the string designated by opnd1. If they match, a branch is made in the pseudo-op table to the entry designated by opnd2. On a mismatch, the next sequential pseudo-op entry is read. Generates a two-byte DC in the pseudo-op table. __Format__: [name] GOIFCAT opnd1,opnd2
GOSYNGT**	IPKDA	Compares the current operand syntax code with the syntax code designated by opnd1. If the current syntax code is the greater of the two, a branch is made in the pseudo-op table to the entry designated by opnd2. Otherwise, the next sequential pseudo-op entry is read. Generates a two-byte DC in the pseudo-op table. __Format__: [name] GOSYNGT opnd1,opnd2
LOAD*	IPKBA IPKJA IPKDB IPKKA IPKIA	Loads a phase or program segment and returns control to the caller.
NOTE*	IPKAA	Obtains the relative position of the last physical record that was read or written from a specific file.
NTSL*	IPKAB	Computes record displacement in a block and disk address of the last block read; it then stores the data in a specified register.
OPDEF**	IPKCB	Generates entries in the opcode table. __Format__: [name] OPDEF [parameter list]
OPEN*	IPKBA IPKMA IPKGA IPKTA	Activates files for processing.

*system macro
**source macro

Figure 40. Macro Usage (Part 3 of 10)

Macro Name	Used By	Description
PCALL	all modules except IPKAB-AH IPKTA	Generates a BAL or BALR with RBR as a return register. Format: [name] PCALL $\left\{\begin{matrix}\text{symbol}\\ \text{(register)}\end{matrix}\right\}$
PCHECK	all modules except IPKAB-AI IPKNA IPKBA IPKOA IPKHA IPKPA IPKKA IPKSA IPKLA IPKSB	Waits for completion of a PREAD or PWRITE operation. If the operation completed without error, PCHECK returns control to the next instruction. Format: [name] PCHECK $\left[\text{FILE=}\left\{\begin{matrix}\text{FILEn}\\ \text{(register)}\\ \underline{\text{(RFILE)}}\end{matrix}\right\}\right]$ where FILE is either FILEn (where n=1,2, or 3) or a register containing the address of PFILEn
PCLOSI	IPKFA IPKGA IPKHA IPKSB	Sets a switch which indicates that the file is closed. PCLOSI must be used when an input file is not read to end-of-file. Format: [name] PCLOSI $\left[\text{FILE=}\left\{\begin{matrix}\text{FILEn}\\ \text{(register)}\\ \underline{\text{(RFILE)}}\end{matrix}\right\}\right]$ where FILE is the same as for PCHECK
PCLOSO	all modules except IPKAA-AH IPKKA IPKBA IPKCC IPKGA IPKMA IPKRA-RC IPKIC IPKQA	Issues an end-of-file record (a field with a record length=2) and closes the file. A corresponding OPEN does not exist; the file is opened with the first PPUTL. Format: [name] PCLOSO $\left[\text{FILE=}\left\{\begin{matrix}\text{FILEn}\\ \text{(register)}\\ \underline{\text{(RFILE)}}\end{matrix}\right\}\right]$ where FILE is the same as for PCHECK
PCSECT	all modules	Defines a control section. Format: PCSECT $\left\{\begin{matrix}\text{(xxnnn=sym)}\\ \text{(xxnnn)}\end{matrix}\right\}$ where xxnnn are the five characters that will be concatenated to the component identification to form an entry point name. xx must be the same as the PHASEID specified in the PHEAD macro. nnn is a three-digit number. sym is any symbol which the user wishes to equate to the entry point name The alternate form "xxnnn" can be used to resume control sections or to define a control section without an equated name.
PENTRY	all modules except IPKAG IPKMA IPKAH IPKHA IPKGA IPKTA IPKQA	Defines an entry point. Format: PENTRY (xxnnn=sym) where xxnnn and sym are the same as for PCSECT
PEXTRN	all modules except IPKAC IPKOA IPKAF-AI IPKSA IPKHA IPKSB IPKNA	Defines an external reference. Format: PEXTRN (xxnnn=sym) where xxnnn and sym are the same as for PCSECT

Figure 40. Macro Usage (Part 4 of 10)

Macro Name	Used By	Description
PFETCH	all modules except IPKAA-AI IPKTA IPKSB	Loads the phase into core and then branches to its entry point. All registers, except RBR, are saved. Format: [name] PFETCH xy where xy is the name of the phase to be fetched
PFIND	IPKCD IPKFA	Finds a macro or COPY code in SYSSLB. If the book is not found, switch PNOPKSW is turned on. Format: [name] PFIND $\left[\text{ADDR=}\begin{Bmatrix}\text{addr}\\ \text{(register)}\\ \underline{\text{(RPARM)}}\end{Bmatrix}\right]$ where ADDR is a nine-byte field made up of a sublibrary initial or a left adjusted name padded with blanks; it is in.EBCDIC
PGETL	all modules except IPKAA-AI IPKQA IPKBA IPKKA IPKRA-RC	Returns the address of the next logical record in register RINPT. When end-of-file is encountered, PGETL branches to the address of the file control block with offset PEOF. Format: [name] PGETL $\left[\text{FILE=}\begin{Bmatrix}\text{FILEn}\\ \text{(register)}\\ \underline{\text{(RFILE)}}\end{Bmatrix}\right]$ where FILE is the same as for PCHECK
PHEAD	all modules	Generates a control section statement, a PUNCH statement, a TITLE statement, and a status MNOTE. Format: PHEAD $\begin{Bmatrix}\text{(xxnnn=sym)}\\ \text{(xxnnn)}\end{Bmatrix}$, 'heading' where xxnnn and sym are the same as for PCSECT. xx is the two-character identification assigned to the module heading is the information that is to appear in the TITLE statement, excluding the module identification.
PINPUT	IPKBA IPKCC IPKCA IPKFA IPKCD IPKTA	Reads a record from SYSIPT or SYSSLB and points a register (RINPT) to the location of that record. PINPUT keeps track of the COPY nesting depth: If the depth is zero (no nesting), PINPUT reads records from SYSIPT. When EOF is read, PINPUT sets the PINEOFSW switch. When end-of-book is read, PINPUT sets PEOBSW. If the depth is not zero, PINPUT reads from SYSSLB. When EOF is read, PINPUT sets both PLBEOFSW and PEOBSW. Format: [name] PINPUT
PMODID	all modules	Generates an embedded identifier consisting of eight characters: module name and release. Format: PMODID

Figure 40. Macro Usage (Part 5 of 10)

Macro Name	Used By	Description
PNOTE	all modules except IPKAB-AI IPKNA IPKBA IPKPA IPKSA-SB	Notes the address of a logical record. It may only be executed after a PPUTL, PGETL, PCHECK, or PCLOSO. **Format**: [name] PNOTE $\begin{bmatrix}\text{FILE=}\begin{Bmatrix}\text{FILEn}\\\text{(registers)}\\\underline{\text{(RFILE)}}\end{Bmatrix},\end{bmatrix}$ NOTEVAL=sym, $\begin{bmatrix}\text{USE=}\begin{Bmatrix}\underline{\text{GET}}\\\text{PUT}\\\text{READ}\\\text{WRITE}\\\text{READNEXT}\end{Bmatrix}\end{bmatrix}$ $\begin{bmatrix}\text{, PARM=}\begin{Bmatrix}\text{address}\\\text{(register)}\\\underline{\text{(RPARM)}}\end{Bmatrix}\end{bmatrix}$ where FILE is the same as for PCHECK NOTEVAL is the location where the assembler service routines shall position the workfile for the next read or write. USE specifies the operation to be performed. USE=READNEXT is used after a PWRITE in order to read the block that is to be written next. PARM must be used with USE=READNEXT and must specify a field containing the block length of the next block to be written. This field has the same layout as the field used by PREAD.
POINTR*	IPKAA	Repositions a file so that the next reading operation involves a record previously identified by a NOTE macro instruction.
POINTS*	IPKAA	Repositions a file to the first record.
POPP	IPKAA IPKLA	Moves the top entry from a stack into a specified area. **Format**: [name] POPP opnd1,opnd2 where opnd1 specifies the name of a stack defined by a DEFSTACK statement opnd2 specifies the address to which the top element of the stack should be moved
POPSAVE	IPKAA IPKKA IPKDA IPKNA IPKFA IPKOA IPKJA IPKRA	Removes the top element from the save stack. **Format**: [name] POPSAVE [RESET] where RESET initializes the stack
PPATCH	all modules	Generates a DS constant with a length computed with the parameters specified. If neither PERCENT nor MIN is specified, the default is PERCENT=5. **Format**: [name] CSECT name, CSECT length $\begin{bmatrix}\text{,PERCENT=}\begin{Bmatrix}\text{p}\\\text{5}\end{Bmatrix}\end{bmatrix}$ [,MIN=m] where CSECT name is the name of the CSECT that will contain the patch area CSECT length is the length of the CSECT in bytes PERCENT=p is the size of the patch area in percent of the CSECT length MIN=m is the size of the patch area in bytes

* system macro

Figure 40. Macro Usage (Part 6 of 10)

Macro Name	Used By	Description
PPOINT	all modules except IPKAB-AI	Causes an assembler service routine to position the file to the location specified on the workfile. <u>Format</u>: [name] PPOINT $\left[\text{FILE=}\begin{Bmatrix}\text{FILEn}\\ \text{(register)}\\ \underline{\text{(RFILE)}}\end{Bmatrix},\right]$ $\left[\text{ADDR=}\begin{Bmatrix}\text{address}\\ \text{(register)}\\ \underline{\text{(RPARM)}}\end{Bmatrix},\right]$ $\left[\text{OFFSET=}\begin{Bmatrix}\text{address}\\ \text{(register)}\\ \underline{\text{(ROFFS)}}\end{Bmatrix},\right]$ $\text{NEXT=}\begin{Bmatrix}\text{GET}\\ \text{PUT}\\ \text{READ}\\ \text{WRITE}\\ \text{START}\end{Bmatrix}$ where FILE is the same as for PCHECK ADDR is the address of the NOTE value OFFSET is the number of bytes from the NOTE value NEXT specifies the operation to be performed. If NEXT=START, the file is positioned to the beginning and ADDR is not necessary
PPRINT	IPKMA IPKRA IPKPA IPKSA IPKQA IPKSB	Operates in move mode and prints a line with the control number in the first byte. <u>Format</u>: [name] PPRINT $\left[\text{ADDR=}\begin{Bmatrix}\text{address}\\ \text{(register)}\\ \underline{\text{(RPARM)}}\end{Bmatrix}\right]$ where ADDR is the address of the line to be printed
PPUNCH	IPKMA IPKQA IPKPA	Operates in move mode and punches a card and/or writes on the link file. <u>Format</u>: [name] PPUNCH $\left[\text{SEQ=}\begin{Bmatrix}\underline{\text{YES}}\\ \text{NO}\end{Bmatrix},\right]$ $\left[\text{ADDR=}\begin{Bmatrix}\text{address}\\ \text{(register)}\\ \underline{\text{(RPARM)}}\end{Bmatrix}\right]$ where SEQ=YES columns 1-72 are in external format and columns 73-80 are to be sequenced SEQ=NO columns 1-80 are in external format and no sequencing is done ADDR is the address of the card image
PPUTL	all modules except IPKAA-AI IPKBA IPKKA IPKRA-RC IPKMA IPKQA IPKSA IPKSB	Returns an address (in ROUTPT) pointing to the location where the next physical record can be built. The record is considered complete when another request is made for that file (another PPUTL or PCLOSO). <u>Format</u>: [name] PPUTL $\left[\text{FILE=}\begin{Bmatrix}\text{FILEn}\\ \text{(register)}\\ \underline{\text{(RFILE)}}\end{Bmatrix}\right]$ [,NEWTRK=YES] where FILE is the same as for PCHECK NEWTRK makes the file start on a new track

Figure 40. Macro Usage (Part 7 of 10)

Macro Name	Used By	Description
PREAD	all modules except IPKAB-AI IPKNA IPKBA IPKOA IPKHA IPKPA IPKKA IPKSA IPKLA IPKSB	Causes an assembler service routine to read a physical record from a file. Format: [name] PREAD $\left[\text{FILE=}\left\{\begin{array}{l}\text{FILEn}\\ \text{(register)}\\ \underline{\text{(RFILE)}}\end{array}\right\}\text{,}\right]$ $\left[\text{PARM=}\left\{\begin{array}{l}\text{address}\\ \text{(register)}\\ \underline{\text{(RPARM)}}\end{array}\right\}\right]$ where FILE is the same as for PCHECK PARM is the address of a parameter table in which the first three bytes specify the location of the buffer and the following two bytes specify the number of bytes to be read
PRETURN	all modules except IPKAD IPKAH IPKAG IPKTA	Loads the branch register (RBR) from the push-down save area and then branches on RBR. Format: [name] PRETURN [NOPOP] where NOPOP causes PRETURN to generate a branch on RBR
PSAVE	all modules except IPKAD IPKAH IPKAG IPKTA	Saves the branch register (RBR) of the calling program in a push-down save area and returns control to the caller. Format: [name] PSAVE
PTSL*	IPKAB	Restores the disk address of the last block read, reads the block, and then computes the address of the record to be processed in the block.
PUSHP	IPKKA IPKLA	Moves information into the stack. Format: [name] PUSHP opnd1,opnd2 where opnd1 specifies the name of a stack defined by a DEFSTACK statement opnd2 points to the area containing the information which should be moved to the stack
PUT*	IPKAC IPKAI IPKAF IPKTA	Writes logical records onto a file.
PWRITE	all modules except IPKAB-AI IPKMA IPKBA IPKOA IPKGA IPKNA IPKHA IPKPA IPKKA IPKSA IPKLA IPKSB	Causes an assembler service routine to write a physical record on the file. Format: [name] PWRITE $\left[\text{FILE=}\left\{\begin{array}{l}\text{FILEn}\\ \text{(register)}\\ \underline{\text{(RFILE)}}\end{array}\right\}\text{,}\right]$ $\left[\text{PARM=}\left\{\begin{array}{l}\text{address}\\ \text{(register)}\\ \underline{\text{(RPARM)}}\end{array}\right\}\right]$ where FILE is the same as for PCHECK PARM is the address of a parameter table in which the first three bytes specify the location of the buffer and the following two bytes specify the length of the record

*system macro

Figure 40. Macro Usage (Part 8 of 10)

Macro Name	Used By	Description
READ*	IPKAA	Reads the next sequential physical record from a file.
RESTSECT	all modules except IPKAD IPKAH IPKAG	Restores the control section name. RESTSECT generates a CSECT statement. The name field is generated from a global variable set symbol by SAVESECT. Format: RESTSECT
SAVESECT	all modules except IPKAD IPKAH IPKAG	Saves the current control section name. SAVESECT stores a &SYSECT in a global SETC variable. D indicates that the control section is a DSECT. Format: SAVESECT [D]
SEGM**	IPKCB	Generates start and end of opcode table segments and creates an offset table to the various segments. Format: SEGM $\left\{\begin{array}{l}\text{a character}\\ \text{END}\\ \text{SEGTAB}\end{array}\right\}$ where SEGTAB is the offset table
SET	all modules except IPKAD IPKAI IPKAG IPKSA IPKAH	The second operand specifies that the switch(es) should be turned ON, OFF, or FLIP. Format: [name] SET opndl, $\left\{\begin{array}{l}\text{ON}\\ \text{OFF}\\ \text{FLIP}\end{array}\right\}$ where FLIP is used to change the status of the switch(es)
SETSYN**	IPKDA	Sets the current syntax code to the value designated by opndl. Generates a two-byte DC in the pseudo-op table. Format: [label] SETSYN opndl
STACKER**	IPKDA	Sends an operator and its priority to a routine which handles the stacking and unstacking of operators. Both the operator and priority are assembled into the pseudo-op table entry. Generates a two-byte DC in the pseudo-op table. Format: [label] STACKER opndl
TABGO**	IPKDA	Makes an unconditional branch in the pseudo-op table. Generates a two-byte DC in the pseudo-op table. Format: [label] TABGO opndl
TCALL**	IPKDA	Stacks the address of the next sequential pseudo-op table entry and branches to the entry designated by opndl. Generates a two-byte DC in the pseudo-op table. Format: [label] TCALL opndl
TERROR**	IPKDA	Sends the error code and string indicator to an error logging routine. Generates a two-byte DC in the pseudo-op table. Format: [label] TERROR opndl $\left[\text{,STRING=}\left\{\begin{array}{l}\text{YES}\\ \underline{\text{NO}}\end{array}\right\}\right]$
*system macro **source macro		

Figure 40. Macro Usage (Part 9 of 10)

Macro Name	Used By	Description
TEXEC**	IPKDA	Gives control to the routine designated by opndl. Generates a two-byte DC in the pseudo-op table. Format: [label] TEXEC opndl where opndl is the name of a routine written in assembler language
TRETURN**	IPKDA	Returns to the pseudo-op table at the entry following the most recent TCALL. Generates a two-byte DC in the pseudo-op table. Format: [label] TRETURN
WAIT*	IPKAA	Issued whenever a program requires that an operation (started by an EXCP instruction) be completed before the excution of the program continues.
WRITE*	IPKAA	Writes a physical record, or part of a physical record, onto a file.
* system macro ** source macro		

Figure 40. Macro Usage (Part 10 of 10)

COPY Book	Used by	Contains
EDPMI	IPKCC IPKDA IPKDB IPKFA IPKHA IPKIA	DSECT describing an edited prototype or a macro instruction
EDPMITEM	IPKCC IPKIA	DSECT describing operands in edited prototype or macro instruction
GARD	IPKDB IPKFA	DSECT describing the global array
GARENT	IPKDB IPKFA	DSECT describing a global array item
IBRTAB	all modules except: IPKAA IPKAD IPKAG IPKAH IPKAJ	Branch table in PCOMMON
KEYTAB	IPKDB IPKFA IPKIA	DSECT describing the keyword table
MACHEAD	IPKEA IPKGA IPKFA IPKIA	DSECT describing the macro header
PCOM1	all modules except: IPKAD IPKAG IPKAH IPKAJ	Basic part of PCOMMON
PCOM2	IPKAA IPKBA IPKCA IPKCB IPKCC IPKCD IPKTA	PCOMMON for IPKCA
PCOM3	all modules except: IPKAB IPKAD IPKAE IPKAF IPKAG IPKAH IPKAJ IPKTA	Part of PCOMMON
PCOM4	IPKAA IPKJA	PCOMMON for IPKJA
PCOM5	IPKAA IPKKA IPKLA IPKMA IPKNA IPKOA IPKPA IPKQA IPKRA	PCOMMON for IPKKA-IPKRC

Figure 41. COPY Code Usage
(Part 1 of 3)

COPY Book	Used by	Contains
PCOM5 (cont.)	IPKRB	
	IPKRC	
PCOM6	IPKAA	PCOMMON for IPKKA-IPKLA
	IPKKA	
	IPKLA	
PCOM7	IPKAA	PCOMMON for IPKPA-IPKSB
	IPKPA	
	IPKQA	
	IPKRA	
	IPKRB	
	IPKRC	
	IPKSA	
	IPKSB	
PCSR	IPKCA	DSECT for a compressed source record
	IPKCB	
	IPKCC	
	IPKCD	
	IPKDA	
	IPKDB	
	IPKEA	
	IPKFA	
	IPKGA	
	IPKHA	
	IPKIA	
	IPKIC	
	IPKJA	
	IPKKA	
	IPKLA	
	IPKPA	
PDCEDIT	IPKJA	DSECT for the operand part of the edited text for DC, DS, and literal DC statements
	IPKKA	
	IPKLA	
	IPKNA	
	IPKOA	
	IPKPA	
PETR	IPKCA	DSECT for an edited text record
	IPKCC	
	IPKCD	
	IPKDA	
	IPKDB	
	IPKEA	
	IPKFA	
	IPKGA	
	IPKIA	
	IPKJA	
	IPKKA	
	IPKLA	
	IPKNA	
	IPKOA	
	IPKPA	
	IPKSA	
	IPKSB	

Figure 41. COPY Code Usage
(Part 2 of 3)

COPY Book	Used by	Contains
PFCB	all modules except: IPKAB IPKAC IPKAD IPKAE IPKAF IPKAG IPKAH IPKAI IPKAJ	DSECT for a file control block and DSECT for PFETCH
PGVHEAD	IPKFA IPKIA	DSECT describing the global vector
PHYR	IPKKA IPKLA	DSECT describing a symbol table entry
PTRTAB	IPKBA IPKCA IPKCC IPKCD IPKDA IPKFA IPKGA IPKIA IPKOA IPKPA IPKRA IPKSA IPKSB	Translate table EBCDIC-internal code
RPNFLAGS	IPKIA	Flags for reverse Polish notation
SMTENT	IPKEA IPKFA IPKGA	DSECT for describing the source macro table
SSD	IPKDA IPKDB IPKEA IPKGA	DSECT for describing the sequence symbol dictionary
VSD	IPKDA IPKDB IPKGA	DSECT for describing the variable symbol dictionary
WORKDTF	IPKAA IPKBA	DSECT for describing the DTFSD

Figure 41. COPY Code Usage
(Part 3 of 3)

Appendix D: Element Formats

Reverse Polish notation expression elements, operands and operators - each starting with a type flag, are scanned from left to right by the evaluation routine CAEVAL. Operands, each with a length byte, are pushed into the stack by the IPUSH routine. Operators are one-byte elements which act upon zero to three stack elements and give a result in the stack or an exit from the evaluation routine.

The following table is divided into two parts: the first part lists the operands and their formats; the second part lists the operators and their formats.

Part 1. Operands

Type of Operand	Input Format (DSECT=IELEM)	Action Taken (Routine=IPUSH)	Stack Format (DSECT=SELEM)
Local arithmetic dimensioned variable symbol	X'02' | pointer to value area | maximum dimension 1 2 2	Element is pushed into the stack with a length byte (length of the element) added	X'02' | pointer to value area | maximum dimension | L=6 1 2 2 1
Local binary dimensioned variable symbol	X'06' | pointer to value area | maximum dimension 1 2 2	Same as above	X'06' | pointer to value area | maximum dimension | L=6 1 2 2 1
Local character dimensioned variable symbol	X'0A' | pointer to value area | maximum dimension 1 2 2	Element is pushed into the stack with a length byte and two "dummy bytes" added	X'0A' | pointer to value area | maximum dimension | dummy bytes | L=8 1 2 2 2 1
Global arithmetic dimensioned variable symbol	X'03' | pointer to GV | maximum dimension 1 2 2	The value in the number field is replaced with a value from global vector. The element is pushed into stack with length byte	X'03' | pointer to value area | maximum dimension | L=6 1 2 2 1
Global binary dimensioned variable symbol	X'07' | pointer to GV | maximum dimension 1 2 2	Same as above	X'07' | pointer to value area | maximum dimension | L=6 1 2 2 1
Global character dimensioned variable symbol	X'0B' | pointer to GV | maximum dimension 1 2 2	The value in the number field is replaced with value from GV. Element is pushed into stack with length byte and two "dummy bytes"	X'0B' | pointer to value area | maximum dimension | dummy bytes | L=8 1 2 2 2 1
Local arithmetic variable symbol	X'00' | pointer to value area 1 2	Element is pushed into the stack with a length byte (length of the element) added	X'00' | pointer to value area | L=4 1 2 1
Local binary variable symbol	X'04' | pointer to value area 1 2	Same as above	X'04' | pointer to value area | L=4 1 2 1
Local character variable symbol	X'08' | pointer to value area 1 2	Element is pushed into the stack with a length byte and four "dummy bytes" added	X'08' | pointer to value area | dummy bytes | L=8 1 2 4 1
Global arithmetic variable symbol	X'01' | pointer to GV 1 2	The value in the number field is replaced with a value from the global vector. The element is pushed into stack with length byte	X'01' | pointer to value area | L=4 1 2 1

Figure 42. Element Formats: Part 1. Operands (Part 1 of 2)

Type of Operand	Input Format (DSECT=IELEM)	Action Taken (Routine=IPUSH)	Stack Format (DSECT=SELEM)
Global binary variable symbol	X'05' pointer to GV 1 2	The value in the number field is replaced with a value from the global vector. The element is pushed into stack with length byte	X'05' pointer to value area L=4 1 2 1
Global character variable symbol	X'09' pointer to GV 1 2	The value in the number field is replaced with value from GV. Element is pushed into stack with length byte and four "dummy bytes" added	X'09' pointer to value area dummy bytes L=8 1 2 4 1
Positional Parameter	X'0C' pointer to PARPTV 1 2	The value in the number field is replaced with a value from PARPTV. Element is pushed into stack with length byte added	X'0C' address of entry in PARTBL L=5 1 3 1
Keyword Parameter	X'0D' pointer to PARPTV 1 2	Same as above	X'0D' address of entry in PARTBL L=5 1 3 1
SYSPARM	X'0E' 1	Element is pushed into stack with the format of global character variable symbol	X'0E' # 0 dummy bytes L=8 1 2 4 1
SYSLIST	X'0F' 1	Element is pushed into stack with a length byte added*	X'0F' L=2 1 1
Character string	X'14' L=0-127 string 1 1 0-127	Element is pushed into stack with a length byte (length of character string) added	X'14' string L=2-129 1 0-127 1
1 byte binary value	X'10' value 1 1	Element is pushed into stack with a length byte added	X'10' value L=3 1 1 1
2 byte binary value	X'11' value 1 2	Same as above	X'11' value L=4 1 2 1
3 byte binary value	X'12' value 1 3	Same as above	X'12' value L=5 1 3 1
4 byte binary value	X'13' value 1 4	Same as above	X'13' value L=6 1 4 1
Attribute error	X'41' error number error default value 1 1 2	Default value is pushed into stack with a length byte added	4 byte binary value

* If the input element following SYSLST is an N' attribute operator, then the stack format is that of a 4 byte binary value.

Figure 42. Element Formats: Part 1. Operands (Part 2 of 2)

Below is a list of the abbreviations used in the following table.

Local arithmetic dimensioned variable symbol	lad
Local binary dimensioned variable symbol	lbd
Local character dimensioned variable symbol	lcd
Global arithmetic dimensioned variable symbol	gad
Global binary dimensioned variable symbol	gbd
Global character dimensioned variable symbol	gcd
Local arithmetic variable symbol	la
Local binary variable symbol	lb
Local character variable symbol	lc
Global arithmetic variable symbol	ga
Global binary variable symbol	gb
Global character variable symbol	gc
Positional parameter	pp
Keyword parameter	kp
SYSLIST	sl
Character string	ch
1 byte binary value	b1
2 byte binary value	b2
3 byte binary value	b3
4 byte binary value	b4

Part 2. Operators

Operator type	Flag	Operand 1 in the stack*	Operand 2 in the stack*	Operation performed	Result in stack
+ (plus)	X'16'	lad, gad, la, ga, b1,b2, b3, or b4	lad, gad, la, ga, b1,b2, b3, or b4	Operand 2 is added to operand 1	4 byte binary value
− (minus)	X'17'	Same options as above	Same options as above	Operand 2 is subtracted from operand 1	4 byte binary value
* (multiply)	X'18'	Same options as above	Same options as above	Operand 1 is multiplied by operand 2	4 byte binary value
/ (divide)	X'19'	Same options as above	Same options as above	Operand 1 is divided by operand 2	4 byte binary value
UMIN	X'1A'	Same options as above	None	Reverse the sign of the operand (unary minus)	4 byte binary value
AND	X'1B'	lbd, gbd, lb, gb, or b4	lbd, gbd, lb, gb, or b4	Compare the logical value of operand 1 with operand 2 (logical AND)	4 byte binary value
OR	X'1C'	Same options as above	Same options as above	Compare the logical value of operand 1 with operand 2 (logical OR)	4 byte binary value
NOT	X'1D'	Same options as above	None	Reverse the logical value of the operand (logical NOT)	4 byte binary value
CBA	X'1F'	lbd, gbd, lb, or gb	None	Convert Boolean to arithmetic	4 byte binary value
CAC	X'1E'	Same options as above	None	Convert arithmetic to character	character string
CBC	X'20'	Same options as above	None	Convert Boolean to character	character string
CCA	X'21'	lcd, gcd, lc, or gc	None	Convert character to arithmetic	4 byte binary value

* Any one of the operands listed.

Figure 43. Element Formats: Part 2. Operators (Part 1 of 4)

Operator type	Flag	Operand 1 in the stack*	Operand 2 in the stack*	Operation performed	Result in stack
AGT	X'22'	lad, gad, la, ga, b1, b2, b3, or b4	lad, gad, la, ga, b1, b2, b3, or b4	Compare the character value of operand 1 with that of operand 2 (character greater than)	4 byte binary value
ALT	X'24'	Same options as above	Same options as above	Same operation as above (arithmetic less than)	4 byte binary value
ANE	X'26'	Same options as above	Same options as above	Same operation as above (arithmetic not equal)	4 byte binary value
AEQ	X'28'	Same options as above	Same options as above	Same operation as above (arithmetic equal)	4 byte binary value
AGE	X'2A'	Same options as above	Same options as above	Same operation as above (arithmetic greater than or equal)	4 byte binary value
ALE	X'2C'	lcd, gcd, lc, gc, or ch	lcd, gcd, lc, gc, or ch	Compare the arithmetic value of operand 1 with that of operand 2 (arithmetic less than or equal)	4 byte binary value
CGT	X'23'	Same options as above	Same options as above	Same operation as above (character less than)	4 byte binary value
CLT	X'25'	Same options as above	Same options as above	Same operation as above (character less than)	4 byte binary value
CNE	X'27'	Same options as above	Same options as above	Same operation as above (character not equal)	4 byte binary value
CEQ	X'29'	Same options as above	Same options as above	Same operation as above (character equal)	4 byte binary value
CGE	X'2B'	Same options as above	Same options as above	Same operation as above (character greater than or equal)	4 byte binary value
CLE	X'2D'	Same options as above	Same options as above	Same operation as above (character less than or equal)	4 byte binary value

* Any one of the operands listed.

Figure 43. Element Formats: Part 2. Operators (Part 2 of 4)

Operator type	Flag	Operand 1 in stack*	Operand 2 in stack*	Operand 3 in stack*	Operation performed	Result in stack
SST	X'2E'	lcd, gcd, lc, gc, or ch	lad, gad, la, ga, b1,b2, b3, or b4 (start character)	lad, gad, la, ga, b1,b2, b3, or b4 (length of substring)	Build a substring	character string
CAT	X'2F'	lcd, gcd, lc, gc, or ch	lcd, gcd, lc, gc, or ch		Concatenate the string of operand 1 to the string of operand 2	character string
T'	X'30'	pp or kp			Find the Type attribute in PARTBL	4 byte binary value
L'	X'31'	Same options as above			Find the Length attribute in PARTBL	4 byte binary value
S'	X'32'	Same options as above			Find the Scale attribute in PARTBL	4 byte binary value
I'	X'33'	Same options as above			Find the Integer attribute in PARTBL	4 byte binary value
N'	X'35'	pp, kp, or s1,			Find the Number attribute in PARTBL	4 byte binary value
K'	X'36'	pp or kp			Find the Count attribute in PARTBL	4 byte binary value
BP	X'34'	Same options as above			Find the binary value of the parameter in PARTBL	4 byte binary value
CP	X'37'	Same options as above			Find the character value of the parameter in PARTBL	character string

*Any one of the operands listed.

Figure 43. Element Formats: Part 2. Operators (Part 3 of 4)

Operator type	Flag	Operand 1 in the stack*	Operand 2 in the stack*	Operation performed	Result in stack
SSC	X'38'	lad, lbd, lcd, gad, gbd, or gcd	lad, gad, la, ga, b1,b2, b3, or b4	Adjust the pointer to the value area (operand 1) with the subscript (operand 2)	*lad, lbd, la, gad, gbd, or gcd
SOP	X'39'	pp, kp, or s1,	Same options as above	Adjust operand 1 (the address of the sublist operand in PARTBL) to point to the sub-operand entry defined by operand 2	Positional parameter
AIF	X'3A'	lbd, gbd, lb, gb, or b4	b3	If operand 1 is 1, go to the start of the macro + the offset (operand 2). In open code, go to START + offset (operand 2)	Exit from the evaluation routine
AGO	X'3B'	b3	None	Go to the start of the macro + the offset (operand 1). In open code, go to open code START + offset (operand 1)	Exit from the evaluation routine
SETA	X'3C'	lad, gad, la, or ga	lad, gad, la, ga, b1,b2, b3, or b4	Put the value of operand 2 in the address of operand 1	Exit from the evaluation routine
SETB	X'3D'	lbd, gbd, lb, or gb	lbd, gbd, lb, gb, b1, or b4	Same operation as above	Exit from the evaluation routine
SETC	X'3E'	lcd, gcd, lc, or gc	lcd, gcd, lc, gc, or ch	Same operation as above	Exit from the evaluation routine
ACTR	X'3F'	lad, gad, la, ga, b1,b2, b3, or b4	None	Move the value of the operand to the ACTR counter field	Exit from the evaluation routine
GEN	X'40'	lcd, gcd, lc, gc, or ch	None	Build a generated statement field	Exit from the evaluation routine
Sequence symbol error	X'43'			Output error record	Exit from the evaluation routine
NOOP	X'44'			No operation	Exit from the evaluation routine

*Any one of the operands listed

Figure 43. Element Formats: Part 2. Operators (Part 4 of 4)

Appendix E: Pseudo (Internal) Operation Codes

Hex	Mnemonic	Description
00	SUBST	Substituted op code
01	REPROED	Reproed statement
02	PUNCH	Assembler instruction
03	CNOP	Assembler instruction
04	ORG	"
05	END	"
06	ENTRY	"
07	EXTRN	"
08	WXTRN	"
09	USING	"
0A	DROP	"
0B	DC	"
0C	DS	"
0D	DCL	Literal DC assembler instruction
0E	EQU	Assembler instruction
0F	EQUL	Literal EQU assembler instruction
10	CCW	Assembler instruction
11	START	"
12	CSECT1ST	Start of the first CSECT
13	LTORG	Assembler instruction
14	CSECT	"
15	DSECT	"
16	COM	"
17	REPRO	"
18	EJECT	"
19	PRINT	"
1A	SPACE	"
1B	TITLE	"
1C	ICTL	"
1D	ISEQ	"
1E	CMENT	Comment statement
1F	COPY	Assembler instruction
20	MI	Macro instruction
21	ERROR	Error record
22	MIC	Macro instruction continuation card (source)
23	MNOTE	Macro processing instruction
24	MIED	Macro instruction edited record
25	MCMENT	Internal macro comment statement
26	MACRO	Macro processing instruction
27	MEND	"
28	MEXIT	"
29	ANOP	Conditional assembly instruction
2A	SETA	"
2B	SETB	"
2C	SETC	"
2D	ACTR	"
2E	AIF	"
	AIFB	"
2F	AGO	"
	AGOB	"
30	GBLA	"
31	GBLB	"
32	GBLC	"

Figure 44. Pseudo (Internal) Operation Codes (Part 1 of 2)

Hex	Mnemonic	Description
33	LCLA	Conditional assembly instruction
34	LCLB	"
35	LCLC	"
36	PROTO	Prototype statement (source)
37	PROTOED	Prototype edited record
38	MHR	Macro header record
39	KT	Keyword table record
3A	GAR	Global array record
3B	MNA	Macro name array record
3C	OCST	Open code start record
3D	UNDEF	Undefined operation code
3E	LITSRC	Literal source record

<u>Note</u>: The DOS/VS Assembler is dependent upon the organisation of the op-codes. Any changes made to this organisation may affect the program code.

Figure 44. Pseudo (Internal) Operation Codes (Part 2 of 2)

Appendix F: Internal Character Set

Character	Internal	External	Punch
0	00	F0	0
1	01	F1	1
2	02	F2	2
3	03	F3	3
4	04	F4	4
5	05	F5	5
6	06	F6	6
7	07	F7	7
8	08	F8	8
9	09	F9	9
A	0A	C1	12.1
B	0B	C2	12.2
C	0C	C3	12.3
D	0D	C4	12.4
E	0E	C5	12.5
F	0F	C6	12.6
G	10	C7	12.7
H	11	C8	12.8
I	12	C9	12.9
J	13	D1	11.1
K	14	D2	11.2
L	15	D3	11.3
M	16	D4	11.4
N	17	D5	11.5
O	18	D6	11.6
P	19	D7	11.7
Q	1A	D8	11.8
R	1B	D9	11.9
S	1C	E2	0.2
T	1D	E3	0.3
U	1E	E4	0.4
V	1F	E5	0.5
W	20	E6	0.6
X	21	E7	0.7
Y	22	E8	0.8
Z	23	E9	0.9
$	24	5B	11.3.8
#	25	7B	3.8
@	26	7C	4.8
=	27	7E	6.8
(	28	4D	12.5.8
+	29	4E	12.6.8
-	2A	60	11
*	2B	5C	11.4.8
/	2C	61	0.1
)	2D	5D	11.5.8
,	2E	6B	0.3.8
ƀ	2F	40	
'	30	7D	5.8
&	31	50	12
.	32	4B	12.3.8

Figure 45. Internal Character Set

Appendix G: Edited Text Flags

Phases ASSEMBLY - **ASSEIA**

Flag (hex)	**Meaning**
00	LCLA
01	GBLA
02	LCLA (dimensioned)
03	GBLA (dimensioned)
04	LCLB
05	GBLB
06	LCLB (dimensioned)
07	GBLB (dimensioned)
08	LCLC
09	GBLC
0A	LCLC (dimensioned)
0B	GBLC (dimensioned)
0C	positional parameter
0D	keyword parameter
0E	&SYSPARM
0F	&SYSLIST
10	1-byte binary value
11	2-byte binary value
12	3-byte binary value
13	4-byte binary value
14	character string
15	sequence symbol
16	+ (addition)
17	- (subtraction)
18	* (multiplication)
19	/ (division)
1A	unary minus
1B	AND
1C	OR
1D	NOT
1E	arithmetic to character conversion
1F	boolean to arithmetic conversion
20	boolean to character conversion
21	character to arithmetic conversion
22	arithmetic GT
23	character GT
24	arithmetic LT
25	character LT
26	arithmetic NE
27	character NE
28	arithmetic EQ

Phases ASSEMBLY - ASSEIA

Flag (hex)	Meaning
29	character EQ
2A	arithmetic GE
2B	character GE
2C	arithmetic LE
2D	character LE
2E	substring
2F	concatenation
30	type attribute
31	length attribute
32	scaling attribute
33	integer attribute
34	binary parameter value
35	number attribute
36	count attribute
37	character parameter value
38	subscript
39	suboperand
40	generate field
41	(,open code relevant symbol
42	)
A1	positional prototype item
A2	keyword prototype item
A3	symbol with all attributes
A4	symbol with type and length attributes
A5	character string (type attribute only)
A6	self-defining term item
A7	sublist start item
A8	sublist end item
A9	basic character expression, macro instruction item
AA	omitted operand outside sublist
AB	error record item
AC	keyword macro instruction
AD	end-of-operand item

Phases ASSEJA - ASSEQA

Flag (hex)	Meaning
00	character constant
01	hexadecimal constant
02	binary constant
03	packed decimal constant
04	zoned decimal constant
05	fixed-point fullword
06	fixed-point halfword
07	floating-point long constant
08	floating-point doubleword
09	floating point fullword
0A	A-type address constant
0B	V-type address constant
0C	Y-type address constant
0D	S-type address constant
29	+ (addition)
2A	- (subtraction)
2B	* (multiplication)
2C	/ (division)
30	explicit bit length
33	unary minus
34	length attribute
35	symbol flag
36	location counter value
37	self-defining term ≥ 256 bytes
38	self-defining term < 256 bytes
39	end of expression
3A	end of operand
3B	start of operand
3C	error flag
3D	ADDR1 field
3E	ADDR2 field
3F	object code field
40	explicit length
41	implicit length
42	scale modifier
43	exponent modifier
44	address constant
45	data constant

Appendix H: Edited Statement Formats

This section shows statement formats at different stages in assembler processing. The diagrams should be used in conjunction with the applicable dummy control section which maps the record; these DSECTs are shown on the diagrams in upper-case letters, underlined. Field lengths in bytes are shown under the field; "V" means a field of variable length.

Object Module IPKCA

COMPRESSED SOURCE RECORD (ALL RECORDS)

PCSR						PSTRINGS				
Record length	File flags	Pseudo opcode	Opcode extension	Type flags	Flag A	Name length	Col.	Name source	Opcode length	Col.
2	1	1	1	1	1	1	1	V	1	1

Opcode source	Operand length	Col.	Operand source	Remarks length	Col.	Remarks source	String length	Col.	String source
V	1	1	V	2	1	V	1	1	V

Object Modules IPKCC through IPKIC

COMPRESSED SOURCE RECORD (NON-MACRO OR CONDITIONAL ASSEMBLY)

PCSR						P STRINGS				
Record length	File flags	Pseudo opcode	Opcode extension	Type flags	Flag A	Name length	Col.	Name source	Opcode length	Col.
2	1	1	1	1	1	1	1	V	1	1

Opcode source	Operand length	Col.	Operand source	Remarks length	Col.	Remarks source	String length	Col.	String source
V	1	1	V	1	1	V	1	1	V

EDITED MACRO PROTOTYPE RECORDS

First Record

EDPMI

Record length	File flags	Pseudo opcode X'37'	Opcode extension	Type flags	Flag A	Name length	Macro name
2	1	1	1	1	1	1	8

Subsequent Records

EDPMI

Record length	File flags	Pseudo opcode X'37'	Opcode extension	Type flags	Flag A	Item(s)...[1]	
2	1	1	1	1	1	V	

EDITED MACRO INSTRUCTION RECORDS

First Record

EDPMI

Record length	File flags	Pseudo opcode X'24'	Opcode extension	Type flags	Flag A	Name length	Macro name	Index	Sequence[2]
2	1	1	1	1	1	1	8	2	8

Subsequent Records

EDPMI

Record length	File flags	Pseudo opcode X'24'	Opcode extension	Type flags	Flag A	Item(s)....[1]	
2	1	1	1	1	1	V	

[1] See item formats below

[2] Inner macros only

ERROR RECORD

PETR

Record length	Switch	X'21'	Error no.	–	String length	String
2	1	1	1	1	1	V

ITEM FORMATS FOR EDITED MACRO PROTOTYPES AND INSTRUCTIONS

										Processed by	Notes
Positional Prototype	X'A1'	Item flag	Length	String						IPKDA	Not needed after IPKDA
	1	1	1	2-8							
Keyword Prototype	X'A2'	Item flag	Length	String						IPKDA	Entered in keyword table in same format
	1	1	1	2-8							
Symbol with all attributes	X'A3'	Item flag	Length	T'	L'	S'	I'	K'	String	IPKHA IPKIA	Originates as type A5 in IPKCC; attributes added in IPKHA
	1	1	1	1	2	2	2	1	1-8		
Symbol with Type and Length Attributes	X'A4'	Item flag	Length	T'	L'	K'	String	–	–	IPKHA IPKIA	Originates as type A5 in IPKCC; attributes added in IPKHA
	1	1	1	1	2	1	1-8	2	2		
Character String (type attribute only)	X'A5'	Item flag	Length	T'	K'	String	–	–	–	(IPKHA) IPKIA	Can be transformed to type A3 or A4 in IPKHA
	1	1	1	1	1	0-255	2	2	2		
Self-Defining Term	X'A6'	Item flag	Length	T'	Binary value	K'	String			IPKIA	
	1	1	1	1	3	1	1-27				
Sublist Start	X'A7'	Item flag	Length	K'						IPKIA	
	1	1	1	1							
Sublist End	X'A8'	Item flag	Length	N'						IPKIA	
	1	1	1	1							
Basic Character Expression, Macro Instruction	X'A9'	Item flag	Length	X'FF'	String/ reverse Polish					IPKDA IPKIA	Expressions translated to reverse Polish notation in IPKDA
	1	1	1	1	1-255						
Omitted operand outside sublist	X'AA'	Item flag	Length							IPKIA	
	1	1	1								
Error Record	X'AB'	Item flag	Length	Record length	Pseudo op	Length	String			IPKDA IPKIA	
	1	1	1	2	4	1	0-8				
Keyword Macro Instruction	X'AC'	Item flag	Length	String						IPKIA	
	1	1	1	1-7							
End of Operand	X'AD'	Item flag	No positional parameters	No keyword parameters						IPKIA	
	1	1	1	1							

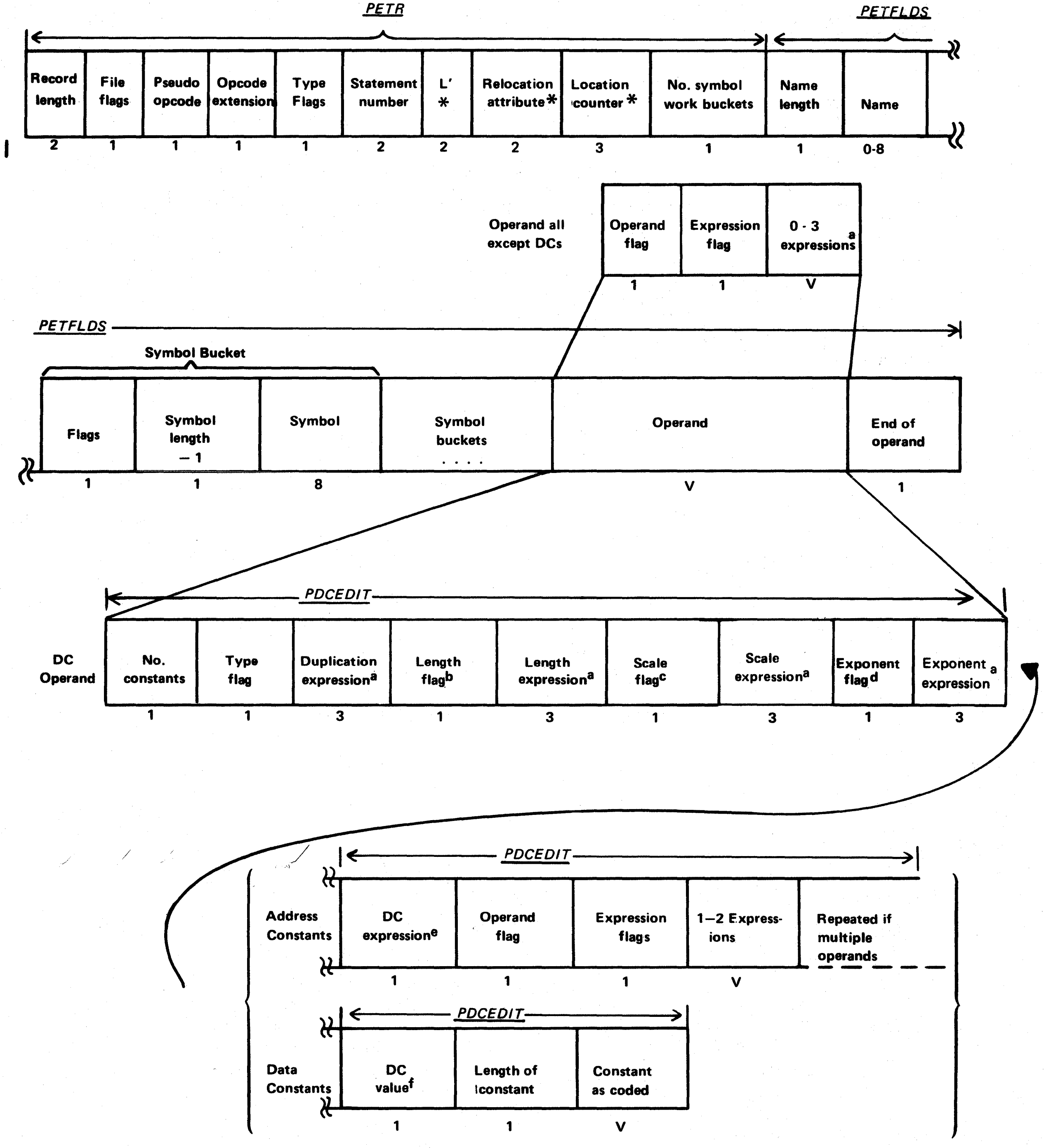
Object Module IPKDA

AIF/AGO/SETx/ACTR

Record length	File flags	Pseudo opcode	Opcode extension	Type flags	Flag A	Sequence field	Expression in reverse Polish notation
2	1	1	1	1	1	8	V

Object Module IPKJA

ALL RECORDS



* Field created but not filled until IPKKA.

** Four-byte field for ENTRY. Filled in IPKKA.

[a] In reverse Polish

[b] BITLEN(X'30') bit length
EXPLEN (X'40') explicit length; IMPLEN (X'41') implied length.

[c] SCALE (X'42')

[d] EXPON(X'43)

[e] DCEXP(X'44)

[f] DCVAL (X'45')

Object Module IPKKA

CNOP RECORDS

PETR						*PETFLDS*				
Record length	Pseudo opcode	Statement number	L'	Relocation attribute	Location counter	Code flag	X'0700'	X'0700'	X'0700'	End flag
2	4	2	2	2	3	1	2	2	2	1

EQU, ORG RECORDS

PETR						*PETFLDS*		
Record length	Pseudo opcode	Statement number	L'	Relocation attribute	Location counter	AD1 flag	Value	End flag
2	2	2	2	2	3	1	3	1

DC RECORDS

PETR							*PETFLDS*
Record length	Pseudo opcode	Statement number	L'	Relocation attribute	Location counter	No. symbol work buckets	Name length
2	4	2	2	2	3	1	1

PDCEDIT					
Symbol bucket					
Flags	Symbol length-1	Symbol		Operand	Operand end flag
1		8			1

Value of duplication factor	No. of constants	Type flag	Length flag	Length value	Scale flag	Rest of record unchanged from IPKJA
3	1	1	1	3	1	

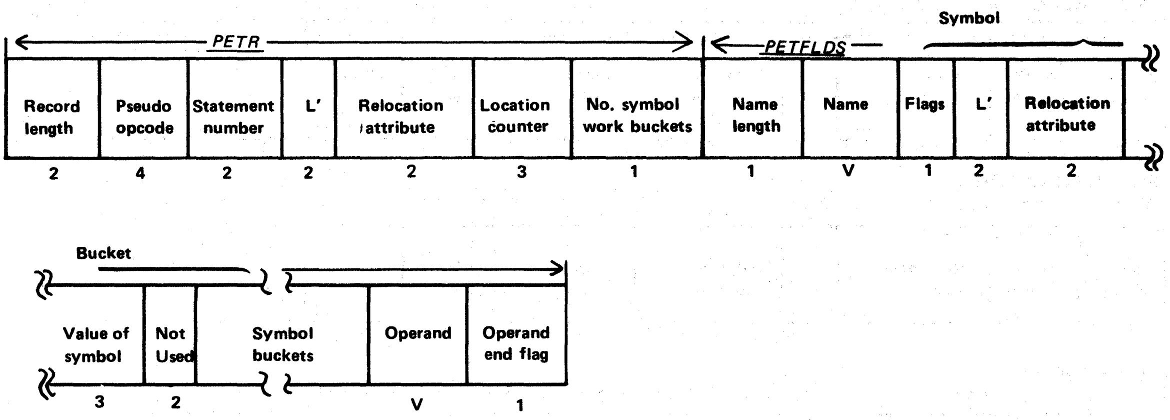
ALL OTHER RECORDS

PETR						*PETFLDS*	
Record length	Pseudo opcode	Statement number	L'	Relocation attribute	Location counter	No. symbol 'work buckets	Name length
2	4	2	2	2	3	1	1

Name	Flags	Symbol length -1	Symbol		Operand	Operand end flag
V	1		8		V	1

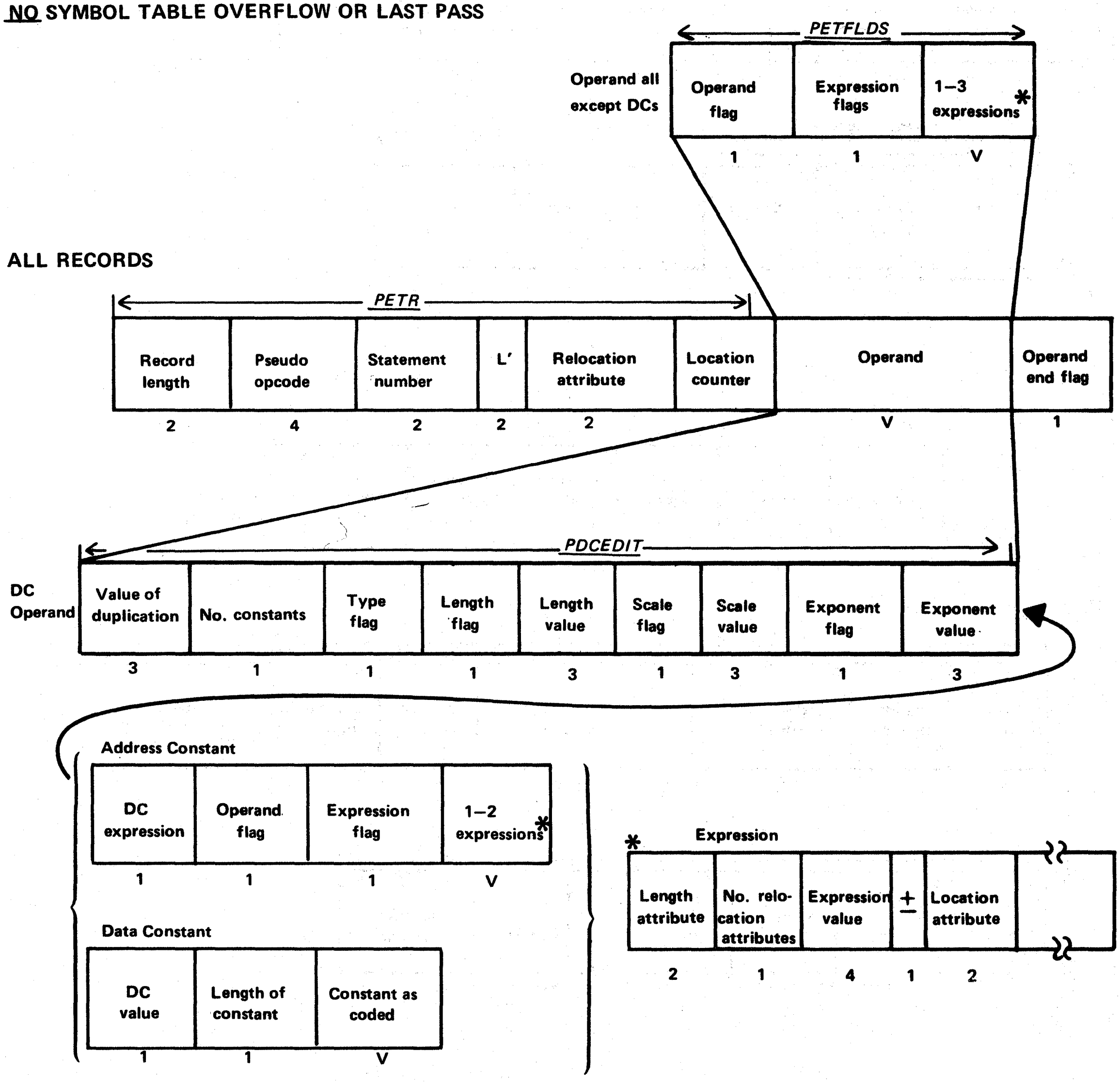
Object Module IPKLA

SYMBOL TABLE OVERFLOW, ALL RECORDS



PETR							PETFLDS									
							Symbol					Bucket				
Record length	Pseudo opcode	Statement number	L'	Relocation attribute	Location counter	No. symbol work buckets	Name length	Name	Flags	L'	Relocation attribute	Value of symbol	Not Used	Symbol buckets	Operand	Operand end flag
2	4	2	2	2	3	1	1	V	1	2	2	3	2		V	1

NO SYMBOL TABLE OVERFLOW OR LAST PASS



Operand all except DCs

PETFLDS		
Operand flag	Expression flags	1–3 expressions*
1	1	V

ALL RECORDS

PETR							
Record length	Pseudo opcode	Statement number	L'	Relocation attribute	Location counter	Operand	Operand end flag
2	4	2	2	2		V	1

DC Operand

PDCEDIT								
Value of duplication	No. constants	Type flag	Length flag	Length value	Scale flag	Scale value	Exponent flag	Exponent value
3	1	1	1	3	1	3	1	3

Address Constant

DC expression	Operand flag	Expression flag	1–2 expressions*
1	1	1	V

Data Constant

DC value	Length of constant	Constant as coded
1	1	V

* Expression

Length attribute	No. relocation attributes	Expression value	±	Location attribute
2	1	4	1	2

Object Module IPKNA

MACHINE INSTRUCTIONS

PETR									PETFLDS					
Record length	File flags	Pseudo opcode	Opcode extension	Type flags	Statement number	L'	Relocation attribute	Location counter	Flag AD1	Value	Flag AD2	Value	Code flag	Object code
2	1	1	1	1	2	2	2	3	1	3	1	3	1	2 - 6

DC INSTRUCTIONS

PETR							PDCOUT					
Record length	File flags	Pseudo opcode	Opcode extension	Type flags	Statement number	Length of object code before duplication	Duplication factor	Not used	Switches	No. bits to shift	Not used	Object code
2	1	1	1	1	2	2	3	3	1	1	3	V

Object Module IPKOA

DC INSTRUCTIONS

(See IPKNA)

Appendix I: Statements Modifying Data Areas

When checking the contents of data areas common to more than one module, it is often necessary to know how other modules modify the area. This appendix lists all symbols defined in such data areas and named in operands modified by the following operation codes: AP, CVD, MVC, MVI, MVN, MVO, MVZ, NC, NI, OC, OI, PACK, ST, STC, STCM, STH, STM, TR, TRT, UNPK, XC, and XI. In addition, the occurence of such a symbol in the operand field of an LA instruction is listed.

The following information is given:

Field name - the symbol naming the modified field.

DSECT name - the name of the dummy section in which the field is defined.

Displacement - the displacement of the field within the dummy section in decimal and hexadecimal notation.

CSECT name - the name of the control section containing the instruction with the field name in a modified operand.

Statement number - the approximate statement number of the modifying statement.

Operation - the operation code of the instruction in which the field name appears in an operand that represents a modified storage address.

Code - the following information about the modified operand:

1 - only one term in this operand.
2 - more than one expression in the operand, but the first expression contains only one term. Unless the operation is type RX, the second expression is probably a length field.
3 - one multiterm expression in the operand. The field name appears as one term but the value of the modified address will depend upon the value of the whole expression.
5 - more than one expression in this operand; the first expression contains more than one term.

Note: This list does not include EQU statements. If a symbol in a DSECT is equated to another symbol, the appearance of the symbol in a modified operand will not be entered in the list.

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
ASSGNSW	PCOMMON	1916	(77C)	IPKJA000	4886	NI	1
				IPKKA000	1418	OI	1
				IPKKA000	1448	NI	1
				IPKKA000	1452	OI	1
				IPKKA000	1612	OI	1
				IPKKA000	1748	OI	1
				IPKKA000	1873	OI	1
				IPKKA000	2223	OI	1
				IPKKA000	2229	OI	1
				IPKKA000	2235	OI	1
				IPKKA000	2253	OI	1
				IPKKA000	2458	OI	1
				IPKKA000	2545	NI	1
				IPKKA000	2566	OI	1
				IPKKA000	2594	OI	1
				IPKKA000	2990	OI	1
				IPKKA000	3195	OI	1
				IPKKA000	3287	OI	1
				IPKKA000	3349	OI	1
				IPKKA000	3417	NI	1
				IPKLA000	1371	OI	1
				IPKLA000	1373	NI	1
				IPKLA000	1554	NI	1
				IPKLA000	1623	NI	1
				IPKLA000	2371	OI	1
				IPKLA000	2572	OI	1
BCDUMMY	EDPMI	10	(A)	IPKCC000	3613	MVI	1
BCITEM	EDPMI	10	(A)	IPKHA000	1588	LA	2
BCITEMST	EDPMI	11	(B)	IPKCC000	3616	MVC	2
				IPKHA000	1590	LA	1
				IPKIA000	4865	LA	1
BEGOFIN	PCOMMON	2023	(7E7)	IPKJA000	1502	ST	1
				IPKKA000	1546	ST	1
				IPKKA000	1932	ST	1
				IPKKA000	3310	ST	1
				IPKLA000	1381	ST	1
BEGOFOUT	PCOMMON	2019	(7E3)	IPKJA000	1464	ST	1
				IPKJA000	2507	ST	1
				IPKJA000	4573	ST	1
				IPKLA000	1382	ST	1
BIT0	PCOMMON	90	(5A)	IPKNA000	1741	LA	3
				IPKNA000	1741	LA	3
BIT1	PCOMMON	90	(5A)	IPKNA000	1741	LA	3
BIT2	PCOMMON	90	(5A)	IPKNA000	1766	LA	1
				IPKNA000	1774	LA	3
BIT5	PCOMMON	90	(5A)	IPKNA000	1774	LA	3
BLANK	PCOMMON	90	(5A)	IPKCA001	2089	MVI	3
				IPKCA001	2102	MVI	3
BLKINCNO	PCOMMON	1998	(7CE)	IPKDB000	2877	STH	1
BLKNP1	DIRENTRY	5	(5)	IPKRA000	1380	MVC	2
				IPKRB000	1093	MVC	2
				IPKRB000	1177	LA	1
BLKNP2	DIRENTRY	11	(B)	IPKRB000	1190	MVC	2
BUFADDR	PFCB	6	(6)	IPKAA000	1940	LA	1
				IPKAA000	2056	LA	1
				IPKAA000	2171	LA	1
				IPKAA000	2184	LA	1
				IPKAA000	2221	LA	1
BUFADDR1	PCOMMON	1694	(69E)	IPKBA000	2227	STCM	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			CODE
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	
BUFADDR1	PCOMMON	1694	(69E)	IPKEA000	2030	STCM	1
				IPKFA000	3137	STCM	1
				IPKJA000	4794	STCM	1
				IPKJA000	4857	LA	1
				IPKKA001	4463	STCM	1
BUFADDR2	PCOMMON	1729	(6C1)	IPKBA000	2217	STCM	1
				IPKEA000	2020	STCM	1
				IPKFA000	3132	STCM	1
				IPKJA000	2536	LA	1
				IPKJA000	4260	LA	1
				IPKJA000	4783	STCM	1
				IPKJA000	4868	LA	1
				IPKKA001	4471	STCM	1
BUFADDR3	PCOMMON	1764	(6E4)	IPKBA000	2232	STCM	1
				IPKPA000	4096	STCM	1
BUFPT	PFCB	3	(3)	IPKAA000	1923	MVC	1
				IPKAA000	2022	STCM	1
				IPKAA000	2035	STH	1
				IPKAA000	2395	STCM	1
CCWS	WORKDTF	96	(60)	IPKAA002	1520	MVC	1
				IPKAA002	1562	MVC	1
CHARC	EPAR	3	(3)	IPKIA000	4925	LA	5
CHARK	EPAR	2	(2)	IPKIA000	3627	LA	1
CHITEMK	EDPMI	11	(B)	IPKCC000	3210	STC	1
				IPKHA000	1550	LA	2
CHITEMST	EDPMI	12	(C)	IPKCC000	3226	MVC	2
				IPKHA000	1552	LA	1
CHITEMT	EDPMI	10	(A)	IPKCC000	3202	MVI	1
				IPKCC000	3208	MVI	1
				IPKHA000	1547	MVI	1
				IPKHA000	1557	MVI	1
				IPKHA000	2394	MVI	1
				IPKHA000	2431	MVC	1
CORADDR	DIRENTRY	0	(0)	IPKRB000	991	STCM	1
				IPKRB000	1120	MVC	2
				IPKRB000	1337	STCM	1
CROSSNP	PCOMMON	1949	(79D)	IPKKA000	3174	LA	1
				IPKKA000	3189	MVC	2
				IPKKA000	3267	LA	1
				IPKKA000	3818	LA	1
				IPKKA000	3832	MVC	1
				IPKKA000	4343	LA	1
				IPKKA000	4360	MVC	1
				IPKLA000	2553	LA	1
				IPKLA000	2566	MVC	2
CURESD	PCOMMON	1977	(7B9)	IPKKA000	2623	ST	1
				IPKKA000	3504	ST	1
				IPKKA000	3534	ST	1
CURNP	PCOMMON	1955	(7A3)	IPKKA000	2624	MVC	1
				IPKKA000	3505	MVC	1
				IPKKA000	3535	MVC	1
				IPKKA000	3626	LA	1
				IPKKA000	3837	MVC	1
				IPKKA000	4363	MVC	1
				IPKKA001	4511	MVC	1
CURSECT	PCOMMON	1927	(787)	IPKIA000	2504	MVC	2
CURSECTL	PCOMMON	1926	(786)	IPKIA000	2503	STC	1
				IPKIA000	5748	XC	1
DBCORE	PCOMMON	1946	(79A)	IPKIA000	2332	STH	1

MODIFIED AREA				MODIFYING INSTRUCTION			CODE
FIELD NAME	DSECT NAME	DISPLACEMENT		CSECT NAME	STMNT NO. (APPROX.)	OPERATION	
		DEC	HEX				
DBCORE	PCOMMON	1946	(79A)	IPKIA000	2445	STH	1
				IPKIA000	5364	STH	1
				IPKIA000	5655	MVC	1
DBVSDADR	PCOMMON	1995	(7CB)	IPKDB000	2870	STCM	1
EAWF2END	PCOMMON	1915	(77B)	IPKFA000	3212	MVC	2
				IPKHA000	2699	MVC	2
EFLG	MNAENT	2	(2)	IPKCA001	1123	MVI	1
				IPKCC000	1741	MVI	5
				IPKCC000	1767	MVI	1
				IPKFA000	2756	MVI	1
				IPKFA000	2818	MVI	1
ENDBUF	PFCB	11	(B)	IPKMA000	1098	STCM	1
ENDID	PCOMMON	1996	(7CC)	IPKKA000	3531	MVC	1
				IPKQA000	1213	MVC	1
ENTRYCNT	PCOMMON	1993	(7C9)	IPKKA000	2537	STH	1
				IPKKA000	3972	STH	1
				IPKKA001	4513	MVC	1
EOFLDPTR	ERRENT	5	(5)	IPKDA000	2364	STCM	1
				IPKDB000	2661	MVC	1
EPARFLAG	EPAR	0	(0)	IPKIA000	5079	MVC	1
EPART	EPAR	1	(1)	IPKIA000	5093	MVC	1
ERRCONST	ERRBYTES	0	(0)	IPKKA000	3040	MVC	1
				IPKKA000	3046	MVC	1
				IPKLA000	2419	MVC	1
				IPKLA000	2425	MVC	1
ERRCOUNT	PCOMMON	2018	(7E2)	IPKJA000	3625	STC	1
				IPKJA000	3762	MVI	1
				IPKJA000	4607	MVI	1
				IPKKA000	1446	MVI	1
				IPKKA000	3059	STC	1
				IPKKA000	3126	MVI	1
				IPKKA001	4474	MVI	1
				IPKLA000	2438	STC	1
				IPKLA000	2505	MVI	1
ERRINFO	ERRENT	0	(0)	IPKDA000	2370	MVC	1
				IPKDA000	2388	MVC	1
				IPKDB000	2663	MVC	1
ERRLNG	ERRBYTES	2	(2)	IPKKA000	3050	STC	1
				IPKLA000	2429	STC	1
ERRSW	ERRBYTES	1	(1)	IPKKA000	3042	OI	1
				IPKLA000	2421	OI	1
ERRTXT	ERRBYTES	3	(3)	IPKKA000	3043	MVC	1
				IPKKA000	3051	MVC	1
				IPKLA000	2422	MVC	1
				IPKLA000	2430	MVC	1
ESDESDID	ESDENTRY	1	(1)	IPKKA000	2539	MVC	1
				IPKKA000	3521	MVC	1
				IPKKA000	3978	MVC	2
				IPKKA000	3986	MVC	1
				IPKKA000	4179	STC	1
ESDHILC	ESDENTRY	5	(5)	IPKKA000	3519	MVC	1
				IPKKA000	3642	MVC	1
ESDIDHI	PCOMMON	1947	(79B)	IPKKA000	3596	STC	1
				IPKKA000	4097	STC	1
				IPKKA000	4178	STC	1
				IPKKA001	4497	MVI	1
ESDIDLO	PCOMMON	1997	(7CD)	IPKKA000	3589	STC	1
				IPKKA001	4498	MVI	1
ESDLCTR	ESDENTRY	2	(2)	IPKKA000	2540	MVC	1

MODIFIED AREA				MODIFYING INSTRUCTION			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
ESDLCTR	ESDENTRY	2	(2)	IPKKA000	3522	MVC	1
				IPKKA000	3641	MVC	1
				IPKKA000	3987	MVC	1
				IPKKA000	4170	MVC	1
ESDNXT	ESDENTRY	16	(10)	IPKKA000	3690	LA	1
				IPKMA000	1442	LA	1
				IPKMA000	1504	LA	1
ESDPTR	PCOMMON	1973	(7B5)	IPKKA000	3706	ST	1
				IPKKA001	4508	MVC	1
ESDSYM	ESDENTRY	8	(8)	IPKKA000	2541	MVC	1
				IPKKA000	3524	MVC	1
				IPKKA000	3979	MVC	1
				IPKKA000	3988	MVC	1
				IPKKA000	4171	MVC	1
				IPKMA000	1424	TR	1
				IPKMA000	1490	TR	1
ESDTYPE	ESDENTRY	0	(0)	IPKKA000	2538	MVI	1
				IPKKA000	3523	MVC	1
				IPKKA000	3973	MVI	1
				IPKKA000	4068	MVI	1
				IPKKA000	4070	MVI	1
				IPKKA000	4169	MVI	1
				IPKMA000	1309	TRT	1
EVALUE	EVALSTCK	1	(1)	IPKKA000	2785	ST	1
				IPKKA000	2861	ST	1
				IPKKA000	2895	ST	1
				IPKKA000	2967	ST	1
				IPKLA000	2174	ST	1
				IPKLA000	2237	ST	1
				IPKLA000	2271	ST	1
				IPKLA000	2345	ST	1
EVLENGTH	EVALSTCK	5	(5)	IPKKA000	2869	LA	1
				IPKKA000	2969	MVI	1
				IPKKA000	2970	MVI	3
				IPKLA000	2246	LA	1
				IPKLA000	2347	MVC	1
EVNXT	EVALSTCK	9	(9)	IPKKA000	2790	LA	1
				IPKLA000	2179	LA	1
EVPLUS	EVALSTCK	5	(5)	IPKKA000	2787	MVI	1
				IPKKA000	2910	LA	1
				IPKKA000	2959	MVC	2
				IPKLA000	2176	MVI	1
				IPKLA000	2286	LA	1
				IPKLA000	2338	MVC	2
EVRELOC	EVALSTCK	6	(6)	IPKKA000	2788	STCM	1
				IPKLA000	2177	STCM	1
EVVARY	EVALSTCK	7	(7)	IPKKA000	2933	LA	2
				IPKKA000	2971	LA	1
				IPKLA000	2311	LA	2
				IPKLA000	2348	LA	1
FILE1NP	PCOMMON	1875	(753)	IPKJA000	2533	MVC	2
				IPKJA000	4257	MVC	2
				IPKKA001	4523	LA	1
				IPKLA000	1244	XC	2
				IPKLA000	1246	XC	2
				IPKLA000	1258	LA	1
				IPKLA000	2607	XC	2
				IPKLA000	2609	XC	2
				IPKPA000	4107	LA	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
FILE1NPR	PCOMMON	1881	(759)	IPKJA000	2538	MVC	2
				IPKJA000	4262	MVC	2
				IPKSA000	3028	LA	1
				IPKSB000	1396	LA	1
				IPKSB000	1472	LA	1
FILE2NP	PCOMMON	1887	(75F)	IPKJA000	4865	MVC	2
				IPKJA000	4873	LA	1
				IPKLA000	1245	XC	2
				IPKLA000	2608	XC	2
				IPKRA000	1590	LA	1
				IPKRA000	1599	LA	1
FILE2NPR	PCOMMON	1893	(765)	IPKJA000	4870	MVC	2
				IPKKA001	4518	LA	1
				IPKLA000	1253	LA	1
				IPKOA000	1202	LA	1
FLAG	DIRENTRY	19	(13)	IPKRA000	1399	NI	1
				IPKRB000	995	OI	1
				IPKRB000	1068	OI	1
				IPKRB000	1114	NI	1
				IPKRB000	1122	OI	1
				IPKRB000	1346	OI	1
GARDIM	GARENT	4	(4)	IPKFA000	2505	LA	2
GARLEN	GARD	0	(0)	IPKDB000	1998	LA	3
				IPKDB000	1999	STH	1
				IPKDB000	2042	LA	3
				IPKDB000	2043	STH	1
				IPKFA000	1521	MVC	1
				IPKFA000	1676	MVC	1
GARLGTH	GARENT	3	(3)	IPKFA000	2502	LA	1
GARTYPE	GARENT	0	(0)	IPKDB000	2443	MVC	2
GSDDIM	GSDENTRY	4	(4)	IPKFA000	2506	LA	2
GSDFLG	GSDENTRY	3	(3)	IPKFA000	2368	MVI	1
				IPKFA000	2445	MVI	1
				IPKFA000	3205	MVI	1
GSDLEN	GSDENTRY	3	(3)	IPKFA000	2467	MVC	2
GSDNDX	GSDENTRY	1	(1)	IPKFA000	2431	STH	1
GSDSYM	GSDENTRY	4	(4)	IPKFA000	2434	LA	2
GSDTYPE	GSDENTRY	0	(0)	IPKFA000	2426	STC	1
HASHPTR	PHYR	0	(0)	IPKKA000	2434	MVC	1
				IPKKA000	3446	MVC	1
				IPKKA000	3999	MVC	1
				IPKKA000	4049	MVC	1
				IPKKA000	4205	MVC	1
IFSAVE	PCOMMON	1672	(688)	IPKAB000	774	STM	1
				IPKAB000	880	STM	1
				IPKAC000	840	STM	1
				IPKAE000	595	STM	1
				IPKAE000	656	STM	1
				IPKAF000	768	STM	1
				IPKAI000	657	STM	1
IJJALSW	IJJCPTAB	44	(2C)	IJJCPD0	155	OI	1
				IJJCPD0	175	NI	1
				IJJCPD1N	1230	NI	1
				IJJCPD1N	1254	OI	1
				IJJCPD2	148	OI	1
IJJCPCNT	IJJCPTAB	76	(4C)	IJJCPD0	227	MVC	2
				IJJCPD1N	1273	MVC	2
IJJCPCTR	IJJCPTAB	80	(50)	IJJCPD0	230	STC	1
				IJJCPD1N	1276	STC	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
IJJCPDAT	IJJCPTAB	82	(52)	IJJCPD0	104	XC	2
				IJJCPD1N	1200	XC	2
IJJCPREC	IJJCPTAB	64	(40)	IJJCPD0	212	MVC	2
				IJJCPD0	226	MVC	2
				IJJCPD1N	1256	MVC	2
				IJJCPD1N	1272	MVC	2
				IJJCPD2	121	STC	1
				IJJCPD2	165	MVC	2
IJJCPSEK	IJJCPTAB	60	(3C)	IJJCPD0	223	ST	1
				IJJCPD1N	1268	ST	1
				IJJCPD2	164	ST	1
IJJCP2ND	IJJCPTAB	45	(2D)	IJJCPD0	139	XC	2
				IJJCPD2	135	XC	2
IJJFRSTR	IJJCPTAB	73	(49)	IJJCPD0	189	OI	1
				IJJCPD1N	1292	OI	1
INDEX	EDPMI	16	(10)	IPKCC000	1537	MVC	1
				IPKFA000	2678	STH	1
				IPKFA000	2816	STH	1
INDKEY	INDENTRY	0	(0)	IPKRA000	1424	MVC	1
				IPKRB000	968	MVI	1
				IPKRB000	1375	MVC	2
INDNP	INDENTRY	11	(B)	IPKRA000	1442	MVC	2
				IPKRB000	954	LA	1
				IPKRB000	1374	MVC	2
INDXB	EPAR	1	(1)	IPKIA000	4483	ST	1
INDXC	EPAR	6	(6)	IPKIA000	4486	UNPK	1
				IPKIA000	4487	MVZ	1
INDXCL	EPAR	5	(5)	IPKIA000	3606	LA	1
				IPKIA000	4488	MVI	1
INDXFLAG	EPAR	0	(0)	IPKIA000	4484	MVI	1
ITEM	EDPMI	10	(A)	IPKDA000	1795	LA	1
				IPKDB000	1787	LA	2
				IPKDB000	2372	LA	2
				IPKDB000	2375	LA	1
				IPKIA000	4538	LA	2
				IPKIA000	4650	LA	2
				IPKIA000	5084	LA	1
				IPKIA000	5152	LA	2
ITEMFLAG	EDPMI	8	(8)	IPKCC000	3216	XC	1
				IPKCC000	3223	OI	1
				IPKCC000	3363	XC	1
				IPKCC000	3372	OI	1
				IPKCC000	3378	OI	1
				IPKCC000	3385	OI	1
				IPKCC000	3395	OI	1
				IPKCC000	3425	OI	1
				IPKDA000	2527	NI	1
				IPKHA000	2434	OI	1
				IPKIA000	5186	NI	1
ITEML	EDPMI	9	(9)	IPKCC000	3211	STC	1
				IPKCC000	3265	STC	1
				IPKCC000	3512	STC	1
				IPKCC000	3611	STC	1
				IPKCC000	3650	STC	1
				IPKDA000	1815	LA	1
				IPKDA000	1816	LA	5
				IPKDA000	1861	LA	1
				IPKDA000	1862	LA	5
				IPKIA000	4902	LA	3

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
ITEMT	EDPMI	7	(7)	IPKCC000	1548	LA	1
				IPKCC000	3365	MVC	1
				IPKCC000	3446	MVC	2
				IPKCC000	3485	LA	1
				IPKCC000	3566	MVC	2
				IPKCC000	3578	MVC	5
				IPKCC000	3868	LA	1
				IPKCC000	4326	LA	1
				IPKDA000	1769	LA	5
				IPKDA000	1816	LA	5
				IPKDA000	1862	LA	5
				IPKDB000	1753	LA	1
				IPKHA000	2404	MVI	1
				IPKHA000	2421	MVI	1
				IPKIA000	4527	LA	1
				IPKIA000	4902	LA	3
KEY	DIRENTRY	11	(B)	IPKRA000	1374	MVC	1
				IPKRA000	1396	MVI	1
				IPKRB000	1058	MVC	2
				IPKRB000	1135	MVI	1
				IPKRB000	1215	LA	1
				IPKRB000	1293	LA	1
				IPKRB000	1330	MVC	2
KITEM	EDPMI	10	(A)	IPKDB000	1869	LA	1
LBARADDR	PCOMMON	1989	(7C5)	IPKDB000	2875	STCM	1
LCLASIZ	PCOMMON	1927	(787)	IPKDB000	2055	STCM	1
LCLASZ	MACHEAD	16	(10)	IPKIA000	5715	LA	1
LCLBSIZ	PCOMMON	1930	(78A)	IPKDB000	2061	STCM	1
LCLCSIZ	PCOMMON	1933	(78D)	IPKDB000	2065	STCM	1
LMNAME	EDPMI	7	(7)	IPKCC000	1523	MVC	2
LOCCNTHI	PCOMMON	2010	(7DA)	IPKKA000	1893	MVC	1
				IPKKA000	3503	MVC	1
				IPKKA000	3520	MVC	1
				IPKKA000	3640	MVC	1
LOCCNTR	PCOMMON	2007	(7D7)	IPKKA000	1557	STCM	1
				IPKKA000	1894	STCM	1
				IPKKA000	2210	STCM	1
				IPKKA000	2500	STCM	1
				IPKKA000	2625	MVC	1
				IPKKA000	3346	STCM	1
				IPKKA000	3414	MVC	1
				IPKKA000	3502	MVC	1
				IPKKA000	3568	MVC	1
				IPKKA000	4039	XC	1
				IPKKA000	4103	MVC	1
				IPKKA000	4130	MVC	1
				IPKKA000	4221	STCM	1
				IPKKA000	4246	STCM	1
				IPKLA000	1592	STCM	1
LOCLATR	PCOMMON	2004	(7D4)	IPKKA000	1535	STH	1
				IPKKA000	1583	STH	1
				IPKKA000	1610	MVC	1
				IPKKA000	1623	MVC	2
				IPKKA000	1647	MVC	2
				IPKKA000	1686	MVC	1
				IPKKA000	1699	MVC	1
				IPKKA000	1728	MVC	1
				IPKKA000	2055	STH	1
				IPKKA000	2579	MVC	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
LOCLATR	PCOMMON	2004	(7D4)	IPKKA000	3418	MVC	1
				IPKKA000	4036	MVC	1
				IPKLA000	1482	MVC	2
LOCRATR	PCOMMON	2006	(7D6)	IPKKA000	2626	MVC	1
				IPKKA000	3501	MVC	1
				IPKKA000	3587	STC	1
				IPKKA000	3595	STC	1
				IPKKA000	4085	MVC	1
				IPKKA000	4089	MVC	1
				IPKKA000	4099	STC	1
				IPKKA000	4104	MVC	1
				IPKKA001	4496	XC	2
LOCTYPE	PCOMMON	2013	(7DD)	IPKKA000	3371	MVI	1
				IPKKA000	3382	MVI	1
				IPKKA000	3395	MVI	1
LPNAME	EDPMI	7	(7)	IPKCC000	4314	MVC	2
MAN	MNAENT	3	(3)	IPKCC000	1712	LA	2
				IPKCC000	1740	LA	2
MFLAGS	PCOMMON	1917	(77D)	IPKDA000	1788	OI	1
				IPKDA000	2354	OI	1
				IPKDA000	3679	OI	1
				IPKDA000	3697	OI	1
				IPKDA000	3763	OI	1
				IPKDB000	1867	OI	1
				IPKDB000	1886	OI	1
				IPKDB000	1974	OI	1
				IPKDB000	2380	OI	1
				IPKDB000	2668	OI	1
				IPKEA000	1161	OI	1
				IPKEA000	1933	OI	1
MIB	PCOMMON	1917	(77D)	IPKDB000	1542	XC	2
				IPKDB000	2839	LA	1
MIBADDR	PCOMMON	1948	(79C)	IPKDB000	1486	LA	1
				IPKDB000	1546	LA	1
				IPKDB000	2840	STCM	1
				IPKEA000	1371	LA	1
MLEVEL	PCOMMON	1944	(798)	IPKIA000	2328	STH	1
				IPKIA000	2453	STH	1
				IPKIA000	5654	XC	2
MNALEN	MNAENT	2	(2)	IPKCC000	1763	MVC	2
				IPKFA000	2838	MVC	2
MNAMEL	MACHEAD	7	(7)	IPKFA000	2138	LA	1
MNANDX	MNAENT	0	(0)	IPKCC000	1746	STH	1
				IPKFA000	2813	STH	1
NEXTCODE	CODE	2	(2)	IPKSA000	3125	LA	1
				IPKSB000	1783	LA	1
NPLITBEG	PCOMMON	1908	(774)	IPKJA000	2547	MVC	1
				IPKJA000	4034	MVC	1
				IPKRA000	1493	LA	1
				IPKRA000	1512	MVC	2
NPMIB	OCSTMH	7	(7)	IPKDB000	1627	MVC	1
				IPKDB000	1658	MVC	1
				IPKEA000	1367	LA	1
NPSSDR1	PCOMMON	1958	(7A6)	IPKDA000	2767	MVC	1
NPSSDWL	PCOMMON	1965	(7AD)	IPKDA000	2733	LA	1
				IPKDA000	2750	LA	1
				IPKDA000	2762	MVC	2
				IPKDB000	1455	LA	1
				IPKDB000	1527	LA	1

MODIFIED AREA				MODIFYING INSTRUCTION			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
NPSSDWL	PCOMMON	1965	(7AD)	IPKDB000	1533	MVC	2
				IPKDB000	1536	LA	1
				IPKDB000	2104	MVC	2
NPVSD	PCOMMON	1942	(796)	IPKDB000	2109	MVC	1
NPVSDR1	PCOMMON	1982	(7BE)	IPKDB000	2108	MVC	1
				IPKDB000	2606	MVC	1
NXTENTRY	PCOMMON	1989	(7C5)	IPKKA000	2441	ST	1
				IPKKA000	3456	ST	1
				IPKKA000	4003	ST	1
				IPKKA000	4052	ST	1
				IPKKA000	4210	ST	1
				IPKKA001	4425	MVC	1
				IPKKA001	4486	ST	1
NXTRA	EVALSTCK	2	(2)	IPKKA000	2876	LA	1
				IPKKA000	2878	LA	1
				IPKKA000	2915	LA	1
				IPKKA000	2920	STH	1
				IPKKA000	2922	LA	1
				IPKKA000	2937	LA	1
				IPKKA000	2957	LA	2
				IPKLA000	2253	LA	1
				IPKLA000	2256	LA	1
				IPKLA000	2275	LA	2
				IPKLA000	2291	LA	1
				IPKLA000	2297	STH	1
				IPKLA000	2299	LA	1
				IPKLA000	2315	LA	1
				IPKLA000	2335	LA	2
OFFS	DIRENTRY	3	(3)	IPKRA000	1382	STH	1
				IPKRB000	1020	STH	1
				IPKRB000	1095	STH	1
OVFLADDR	PCOMMON	1992	(7C8)	IPKDB000	2873	STCM	1
PABENDC	PCOMMON	1669	(685)	IPKBA000	1309	MVI	1
				IPKBA000	1395	MVI	1
				IPKFA000	2237	MVI	1
				IPKFA000	2395	MVI	1
				IPKFA000	3269	MVI	1
PAFLAG2	PETFLDS	0	(0)	IPKJA000	2085	OI	1
				IPKJA000	3140	STC	1
				IPKJA000	3145	OI	1
				IPKJA000	3188	STC	1
				IPKKA000	2297	NI	1
				IPKKA000	3904	LA	5
				IPKKA000	3956	OI	1
				IPKKA000	3981	NI	1
				IPKKA000	3983	OI	1
				IPKLA000	1539	OI	1
				IPKLA000	1825	NI	1
				IPKLA000	1844	OI	1
				IPKLA000	1857	OI	1
				IPKLA000	1860	OI	1
				IPKLA000	1912	OI	1
				IPKLA000	1929	OI	1
				IPKLA000	1975	OI	1
PANXT	PETFLDS	4	(4)	IPKPA000	2529	LA	1
PASSGNSW	PCOMMON	2017	(7E1)	IPKKA000	3289	OI	1
				IPKKA000	3699	NI	1
				IPKKA000	3834	OI	1
				IPKKA000	4352	OI	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			CODE
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	
PASSGNSW	PCOMMON	2017	(7E1)	IPKKA001	4505	MVI	1
				IPKKA001	4510	NI	1
				IPKLA000	1235	OI	1
				IPKLA000	2616	NI	1
PBITALGN	PDCEDIT	7	(7)	IPKNA000	1531	XC	1
				IPKPA000	2230	STC	1
				IPKPA000	2238	MVI	1
PBLKPTRK	WORKDTF	70	(46)	IPKBA000	1912	STC	1
				IPKBA000	2085	STC	1
PBUFLEN1	PCOMMON	1697	(6A1)	IPKBA000	1983	STH	1
				IPKFA000	3139	MVC	1
				IPKJA000	4792	STH	1
				IPKKA001	4457	MVC	1
PBUFLEN2	PCOMMON	1732	(6C4)	IPKBA000	1897	STH	1
				IPKJA000	4781	MVC	1
PBUFLEN3	PCOMMON	1767	(6E7)	IPKBA000	1807	STH	1
PB1FISIZ	PCOMMON	1885	(75D)	IPKBA000	2072	STH	1
PB12SIZ	PCOMMON	1887	(75F)	IPKBA000	2151	STH	1
PCBTYPE	PETR	5	(5)	IPKNA000	1738	LA	1
PCNXT	PETFLDS	3	(3)	IPKPA000	2557	LA	5
PCODE	PETFLDS	1	(1)	IPKPA000	2546	LA	1
				IPKPA000	2557	LA	5
PCSRFLGA	PCSR	6	(6)	IPKCA001	1421	OI	1
				IPKCA001	1448	OI	1
				IPKCA001	1538	OI	1
				IPKCA001	1612	OI	1
				IPKCA001	1627	NI	1
				IPKCD001	1398	OI	1
				IPKCD001	1669	OI	1
				IPKDA000	1674	NI	1
				IPKDA000	2922	OI	1
				IPKDA000	2946	NI	1
				IPKDB000	1934	NI	1
				IPKDB000	2010	NI	1
				IPKEA000	1144	OI	1
				IPKGA000	1525	OI	1
				IPKIA000	2094	OI	1
				IPKIA000	2249	OI	1
				IPKIA000	2665	OI	1
				IPKJA000	4550	MVC	1
PCSRHEAD	PCSR	0	(0)	IPKCC000	1499	XC	2
				IPKCC000	1518	XC	2
				IPKCC000	4268	XC	2
				IPKCC000	4310	XC	2
				IPKDB000	1962	XC	2
				IPKDB000	1995	XC	2
				IPKDB000	2039	XC	2
PCSRIOP	PCSR	2	(2)	IPKJA000	4549	MVC	1
PCSRLEN	PCSR	0	(0)	IPKCA001	1194	MVC	1
				IPKCA001	1597	STCM	1
				IPKCA001	1900	MVC	1
				IPKCC000	1530	MVC	1
				IPKCC000	1535	MVC	1
				IPKCC000	1632	MVC	1
				IPKCC000	1863	STCM	1
				IPKCC000	3476	STH	1
				IPKCC000	4312	MVI	3
				IPKCD001	1641	MVC	1
				IPKDA000	2834	STH	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			CODE
FIELD NAME	DSECT NAME	DISPLACEMENT		CSECT NAME	STMNT NO. (APPROX.)	OPERATION	
		DEC	HEX				
PCSRLEN	PCSR	0	(0)	IPKDB000	1623	MVC	1
				IPKDB000	1656	MVC	1
				IPKDB000	1817	STH	1
				IPKDB000	1978	STH	1
				IPKEA000	1383	MVC	1
				IPKFA000	2063	MVC	1
				IPKGA000	1389	STCM	1
				IPKIA000	2085	STH	1
				IPKPA000	3783	MVC	1
PCSROP	PCSR	3	(3)	IPKCA001	1193	MVI	1
				IPKCA001	1393	MVI	1
				IPKCA001	1401	MVI	1
				IPKCA001	1446	MVI	1
				IPKCA001	1457	MVI	1
				IPKCA001	1901	MVI	1
				IPKCB000	1067	MVC	2
				IPKCC000	1500	MVI	1
				IPKCC000	1519	MVI	1
				IPKCC000	1633	MVI	1
				IPKCC000	3840	MVI	1
				IPKCC000	3843	MVI	1
				IPKCC000	3863	MVI	1
				IPKCC000	3866	MVI	1
				IPKCC000	4269	MVI	1
				IPKCC000	4311	MVI	1
				IPKCD001	1642	MVI	1
				IPKDB000	1655	MVI	1
				IPKDB000	1802	MVI	1
				IPKDB000	1963	MVI	1
				IPKDB000	1996	MVI	1
				IPKDB000	2040	MVI	1
				IPKEA000	1384	MVC	1
				IPKFA000	2026	STCM	1
				IPKPA000	3175	MVI	1
				IPKPA000	3520	MVI	1
PCSROPX	PCSR	4	(4)	IPKFA000	2065	XC	2
				IPKJA000	1613	MVI	1
PCSROP0	PCSR	2	(2)	IPKCA001	1352	OI	1
				IPKCB000	1053	MVZ	1
				IPKCC000	3835	OI	1
				IPKDA000	2284	OI	1
				IPKDA000	2899	NI	1
				IPKDA000	2901	OI	1
				IPKDA000	2925	NI	1
				IPKDA000	2927	OI	1
				IPKDB000	2771	NI	1
				IPKDB000	2773	OI	1
				IPKDB000	2778	NI	1
				IPKDB000	2780	OI	1
				IPKEA000	1573	NI	1
				IPKFA000	2064	XC	1
				IPKJA000	1509	NI	1
				IPKPA000	3806	OI	1
PCSROP3	PCSR	5	(5)	IPKCA001	1350	OI	1
				IPKCA001	1729	OI	1
				IPKCA001	1760	OI	1
				IPKCA001	1805	OI	1
				IPKCA001	1822	OI	1
				IPKCA001	1868	OI	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
PCSROP3	PCSR	5	(5)	IPKCA001	2167	OI	1
				IPKCB000	1054	MVN	1
				IPKCB000	1064	OC	1
				IPKDA000	1872	OI	1
				IPKDA000	2244	OI	1
				IPKGA000	1390	MVC	2
				IPKIC000	1033	OI	1
PCSRSTR1	PCSR	7	(7)	IPKCC000	1574	LA	1
				IPKCC000	1962	LA	1
				IPKDA000	1856	LA	1
				IPKDA000	1873	LA	1
				IPKDA000	1880	LA	1
				IPKEA000	1269	LA	1
				IPKHA000	1293	LA	1
				IPKHA000	1652	LA	1
				IPKIA000	1999	LA	3
				IPKIA000	2495	LA	1
				IPKIC000	938	LA	1
				IPKJA000	2394	LA	1
				IPKJA000	2568	LA	1
				IPKJA000	2609	LA	1
				IPKJA000	2912	LA	1
				IPKJA000	3755	LA	5
				IPKPA000	1729	LA	1
				IPKPA000	2726	LA	1
				IPKPA000	3560	LA	1
PDCCODE	PDCOUT	11	(B)	IPKNA000	1535	LA	5
				IPKNB000	3041	LA	1
				IPKNB000	3047	LA	5
				IPKNB000	3117	ST	1
				IPKNB000	3186	ST	1
				IPKNB000	3227	STCM	1
				IPKNB000	3231	STCM	1
				IPKNB000	3233	STCM	1
				IPKNB000	3235	STCM	1
				IPKNB000	3237	STCM	1
				IPKNB000	3268	ST	1
				IPKNB000	3311	ST	1
				IPKOA000	1445	LA	1
				IPKOA000	1480	LA	1
				IPKOA000	1522	STC	5
				IPKOA000	1524	LA	5
				IPKOA000	1555	STC	5
				IPKOA000	1559	STC	5
				IPKOA000	1564	XC	2
				IPKOA000	1585	LA	2
				IPKOA000	1685	MVC	2
				IPKOA000	1758	MVC	1
PDCFL	PDCOUT	6	(6)	IPKOA000	1489	OI	1
PDCFLAG	PDCEDIT	6	(6)	IPKLA000	1669	NI	1
				IPKLA000	1677	OI	1
				IPKNA000	1528	XC	2
				IPKNA000	1530	NI	1
PDCFLD	PDCEDIT	2	(2)	IPKLA000	1654	LA	2
				IPKOA000	1481	LA	1
				IPKOA000	1500	TR	2
				IPKOA000	1622	LA	1
				IPKOA000	1724	LA	1
PDCLEN	PDCEDIT	1	(1)	IPKLA000	1657	MVC	1

MODIFIED AREA				MODIFYING INSTRUCTION			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
PDCLEN	PDCEDIT	1	(1)	IPKNB000	3105	LA	1
				IPKNB000	3299	LA	1
PDTFADR1	PCOMMON	1688	(698)	IPKBA000	1449	MVC	1
PDTFADR2	PCOMMON	1723	(6BB)	IPKBA000	1450	MVC	1
PDTFADR3	PCOMMON	1758	(6DE)	IPKBA000	1451	MVC	1
PDUPEXP	PDCEDIT	0	(0)	IPKLA000	1627	MVC	3
				IPKLA000	1629	LA	5
				IPKLA000	1679	MVC	2
PDUPLFAC	PDCOUT	0	(0)	IPKNA000	1535	LA	5
				IPKNB000	3047	LA	5
PENDBUF1	PCOMMON	1699	(6A3)	IPKBA000	2241	STCM	1
				IPKEA000	2026	STCM	1
				IPKFA000	3152	STCM	1
				IPKJA000	4789	STCM	1
				IPKKA001	4459	STCM	1
PENDBUF2	PCOMMON	1734	(6C6)	IPKBA000	2245	STCM	1
				IPKEA000	2016	STCM	1
				IPKFA000	3149	STCM	1
				IPKJA000	4780	STCM	1
				IPKKA001	4469	STCM	1
PENDBUF3	PCOMMON	1769	(6E9)	IPKBA000	2249	STCM	1
				IPKPA000	4094	STCM	1
PENTVAL	PETFLDS	9	(9)	IPKLA000	1911	MVC	1
PEOF	PFCB	18	(12)	IPKDA000	1387	STCM	3
				IPKEA000	2098	MVC	5
				IPKJA000	4839	MVC	1
				IPKMA000	1088	MVC	1
				IPKOA000	1208	STCM	1
				IPKSA000	3034	STCM	1
				IPKSB000	1401	STCM	1
				IPKSB000	1508	STCM	1
PEOFADR1	PCOMMON	1706	(6AA)	IPKPA000	1551	STCM	1
PEOFADR2	PCOMMON	1741	(6CD)	IPKHA000	1219	MVC	1
				IPKIA000	5609	MVC	1
				IPKKA001	4424	STCM	1
				IPKLA000	1239	STCM	1
				IPKLA000	1249	STCM	1
PEOFADR3	PCOMMON	1776	(6F0)	IPKHA000	1220	MVC	1
				IPKPA000	1549	STCM	1
PERLNG	PETR	6	(6)	IPKDA000	2436	STC	1
				IPKDB000	2712	STC	1
				IPKFA000	2911	MVC	2
				IPKFA000	2912	MVC	2
				IPKGA000	1743	MVC	2
				IPKGA000	1744	MVC	2
				IPKIA000	2308	MVC	2
				IPKIA000	4408	MVI	1
				IPKIA000	4703	MVC	2
				IPKIA000	5226	STC	1
				IPKIA000	5321	MVC	2
				IPKJA000	3754	STC	1
				IPKKA000	3088	MVI	1
				IPKKA000	3117	STC	1
				IPKLA000	2467	MVI	1
				IPKLA000	2496	STC	1
				IPKNA000	1916	MVI	1
				IPKNA000	1927	STC	1
				IPKNA000	1968	STC	1
				IPKNB000	2888	STC	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			CODE
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	
PERLNG	PETR	6	(6)	IPKNB000	2907	STC	1
				IPKNB000	2913	LA	5
				IPKOA000	1303	STC	1
				IPKOA000	1322	STC	1
				IPKOA000	1328	LA	5
PERREXC	PETR	9	(9)	IPKSA000	3064	LA	1
				IPKSB000	1717	LA	1
PERRHD	PERR	0	(0)	IPKPA000	4033	MVC	1
				IPKPA000	4045	MVC	5
PERRLEN	PERR	0	(0)	IPKPA000	4035	STH	1
PERRSTNR	PERR	6	(6)	IPKPA000	4032	STH	1
PERRSTR	PETR	8	(8)	IPKSA000	3066	MVC	2
				IPKSB000	1719	MVC	2
PERSRC	PETR	7	(7)	IPKDA000	2454	MVC	2
				IPKDB000	2730	MVC	2
				IPKIA000	4417	LA	5
				IPKIA000	5238	MVC	2
				IPKJA000	3739	MVC	1
				IPKKA000	3092	MVC	2
				IPKKA000	3098	MVC	2
				IPKKA000	3101	LA	3
				IPKKA000	3118	LA	5
				IPKLA000	2471	MVC	2
				IPKLA000	2477	MVC	2
				IPKLA000	2480	LA	3
				IPKLA000	2497	LA	5
				IPKNA000	1912	LA	1
				IPKNA000	1920	MVC	2
				IPKNA000	1925	MVC	2
				IPKNB000	2881	MVC	2
				IPKNB000	2885	MVC	2
				IPKNB000	2896	MVC	2
				IPKNB000	2900	MVC	5
				IPKNB000	2904	MVC	5
				IPKOA000	1296	MVC	2
				IPKOA000	1300	MVC	2
				IPKOA000	1311	MVC	2
				IPKOA000	1315	MVC	5
				IPKOA000	1319	MVC	5
PETEPX	PETR	4	(4)	IPKCA001	2376	MVC	1
				IPKCC000	4359	MVC	1
				IPKCD001	1717	MVC	1
				IPKDA000	2419	MVC	1
				IPKDB000	2695	MVC	1
				IPKEA000	1961	MVC	1
				IPKFA000	2855	MVC	1
				IPKGA000	1765	MVC	1
				IPKIA000	2306	MVI	1
				IPKIA000	2317	MVI	1
				IPKIA000	2635	MVI	1
				IPKIA000	2745	STC	1
				IPKIA000	4698	MVI	1
				IPKIA000	4701	MVI	1
				IPKIA000	5225	MVI	1
				IPKIA000	5319	MVI	1
				IPKIA000	5469	STC	1
				IPKJA000	3730	MVC	1
				IPKKA000	3081	MVC	1
				IPKLA000	2460	MVC	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			CODE
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	
PETEPX	PETR	4	(4)	IPKNA000	1949	MVC	2
				IPKNB000	2917	STC	1
				IPKOA000	1332	STC	1
PETHEAD	PETR	0	(0)	IPKCA001	2370	XC	2
				IPKCD001	1711	XC	2
				IPKDA000	2416	XC	2
				IPKDB000	2692	XC	2
				IPKEA000	1958	XC	2
				IPKGA000	1706	MVC	1
				IPKGA000	1764	MVC	1
				IPKJA000	1570	LA	5
				IPKKA000	1820	LA	5
				IPKKA000	3118	LA	5
				IPKLA000	2497	LA	5
				IPKNB000	2913	LA	5
				IPKOA000	1328	LA	5
PETIOP	PETR	2	(2)	IPKFA000	2854	MVC	1
				IPKJA000	2604	MVC	1
				IPKJA000	2908	MVC	1
				IPKJA000	3694	MVC	1
				IPKJA000	4487	MVC	1
PETLEN	PETR	0	(0)	IPKCA001	2372	MVC	1
				IPKCC000	4360	MVC	1
				IPKCD001	1713	MVC	1
				IPKDA000	2444	STH	1
				IPKDB000	2720	STH	1
				IPKEA000	1960	MVC	1
				IPKFA000	2901	STH	1
				IPKGA000	1713	STH	1
				IPKGA000	1731	STH	1
				IPKGA000	1776	STH	1
				IPKIA000	4418	STH	1
				IPKJA000	2228	MVC	1
				IPKJA000	2746	STH	1
				IPKJA000	3696	MVC	1
				IPKJA000	3756	STH	1
				IPKKA000	1648	MVC	1
				IPKKA000	1670	MVC	1
				IPKKA000	1821	STH	1
				IPKKA000	1897	MVC	1
				IPKKA000	1919	MVC	1
				IPKKA000	2155	STH	1
				IPKKA000	2242	MVC	1
				IPKKA000	2590	MVC	1
				IPKKA000	2596	MVC	1
				IPKKA000	2646	MVC	1
				IPKKA000	3089	MVC	1
				IPKKA000	3119	STH	1
				IPKKA000	3514	MVC	1
				IPKKA000	3542	MVC	1
				IPKLA000	1611	STH	1
				IPKLA000	2468	MVC	1
				IPKLA000	2498	STH	1
				IPKNA000	1891	STH	1
				IPKNB000	2914	STH	1
				IPKNB000	3048	STH	1
				IPKNB000	3053	STH	1
				IPKNB000	3462	STH	1
				IPKOA000	1329	STH	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
PETLEN	PETR	0	(0)	IPKOA000	1452	STH	1
				IPKOA000	1459	STH	1
PETOP	PETR	3	(3)	IPKCA001	2371	MVI	1
				IPKCC000	4358	MVI	1
				IPKCD001	1712	MVI	1
				IPKDA000	2418	MVI	1
				IPKDB000	2694	MVI	1
				IPKEA000	1959	MVI	1
				IPKIA000	4412	MVI	1
				IPKJA000	3729	MVI	1
				IPKKA000	3080	MVI	1
				IPKLA000	2459	MVI	1
				IPKNA000	1948	MVI	1
PETOP0	PETR	2	(2)	IPKFA000	2870	OI	1
				IPKIA000	4413	XC	1
				IPKJA000	1562	NI	1
				IPKJA000	1783	OI	1
				IPKJA000	2047	OI	1
				IPKJA000	3728	MVI	1
				IPKJA000	4629	MVI	1
				IPKKA000	1453	NI	1
				IPKKA000	1541	NI	1
				IPKKA000	1542	OC	1
				IPKKA000	1706	NI	1
				IPKKA000	1707	OC	1
				IPKKA000	1738	NI	1
				IPKKA000	1739	OC	1
				IPKKA000	1977	OC	1
				IPKKA000	3079	MVI	1
				IPKKA000	3100	OI	1
				IPKLA000	1238	OI	1
				IPKLA000	1369	NI	1
				IPKLA000	1667	MVC	2
				IPKLA000	1675	OI	1
				IPKLA000	1746	OI	1
				IPKLA000	1757	OI	1
				IPKLA000	1854	NI	1
				IPKLA000	2458	MVI	1
				IPKLA000	2479	OI	1
				IPKNA000	1521	OI	1
				IPKNA000	1907	NI	1
				IPKNA000	1909	OI	1
				IPKNB000	3016	OI	1
				IPKSB000	1725	OI	1
PETOP3	PETR	5	(5)	IPKIA000	5559	OI	1
				IPKJA000	2227	OI	1
				IPKKA000	1569	OI	1
				IPKKA000	1669	OI	1
				IPKKA000	1845	OI	1
				IPKKA000	1918	OI	1
				IPKKA000	2241	OI	1
				IPKKA000	3567	OI	1
				IPKNA000	1659	OI	1
				IPKNA000	1664	NI	1
PETSTNO	PETR	6	(6)	IPKJA000	2729	STH	1
				IPKJA000	4591	MVC	1
PEXPFLAG	PETFLDS	1	(1)	IPKNA000	2182	OI	1
PFCBSW	PFCB	17	(11)	IPKAA000	1925	OI	1
				IPKAA000	2004	NI	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			CODE
FIELD NAME	DSECT NAME	DISPLACEMENT		CSECT NAME	STMNT NO. (APPROX.)	OPERATION	
		DEC	HEX				
PFCBSW	PFCB	17	(11)	IPKAA000	2174	OI	1
				IPKAA000	2227	NI	1
				IPKAA000	2239	OI	1
				IPKAA000	2249	NI	1
				IPKAA000	2296	NI	1
				IPKAA000	2385	NI	1
				IPKAA000	2397	OI	1
				IPKDB000	1488	OI	1
				IPKDB000	1643	NI	1
				IPKFA000	2345	OI	1
				IPKFA000	2349	NI	1
				IPKFA000	2721	OI	1
				IPKFA000	2725	NI	1
				IPKFA000	2951	NI	1
				IPKHA000	2636	NI	1
				IPKIA000	2522	NI	1
				IPKIA000	2528	NI	1
				IPKJA000	4287	OI	1
				IPKKA000	3653	OI	1
				IPKKA000	3768	OI	1
				IPKKA000	4079	OI	1
				IPKKA000	4329	OI	1
				IPKRB000	1383	OI	1
				IPKSB000	1468	NI	1
PFILE2	PCOMMON	1723	(6BB)	IPKBA000	1585	LA	1
				IPKCA001	1131	LA	1
				IPKCA001	1904	LA	1
				IPKCA001	2379	LA	1
				IPKCC000	1414	LA	1
				IPKCC000	1541	LA	1
				IPKCC000	1636	LA	1
				IPKCC000	1640	LA	1
				IPKCC000	1866	LA	1
				IPKCC000	3479	LA	1
				IPKCC000	4320	LA	1
				IPKCC000	4369	LA	1
				IPKCD001	1072	LA	1
				IPKCD001	1325	LA	1
				IPKCD001	1373	LA	1
				IPKCD001	1546	LA	1
				IPKCD001	1550	LA	1
				IPKCD001	1645	LA	1
				IPKCD001	1672	LA	1
				IPKCD001	1720	LA	1
				IPKDA000	1387	STCM	3
				IPKDA000	1540	LA	1
				IPKDB000	1594	LA	1
				IPKDB000	1641	LA	1
				IPKDB000	1707	LA	1
				IPKDB000	2863	LA	1
				IPKEA000	1390	LA	1
				IPKEA000	1490	LA	1
				IPKEA000	1615	LA	1
				IPKEA000	2079	LA	1
				IPKEA000	2085	LA	1
				IPKEA000	2090	LA	1
				IPKFA000	1456	LA	1
				IPKFA000	1639	LA	1
				IPKFA000	1667	LA	1

MODIFIED AREA				MODIFYING INSTRUCTION			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
PFILE2	PCOMMON	1723	(6BB)	IPKFA000	1765	LA	1
				IPKFA000	1961	LA	1
				IPKFA000	2210	LA	1
				IPKFA000	2302	LA	1
				IPKFA000	2949	LA	1
				IPKFA000	2982	LA	1
				IPKFA000	3028	LA	1
				IPKFA000	3210	LA	1
				IPKFA000	3253	LA	1
				IPKFA000	3277	LA	1
				IPKFA000	3355	LA	1
				IPKGA000	1226	LA	1
				IPKGA000	1509	LA	1
				IPKGA000	1806	LA	1
				IPKGA000	1881	LA	1
				IPKHA000	1210	LA	1
				IPKHA000	1216	LA	1
				IPKHA000	1237	LA	1
				IPKHA000	2628	XC	1
				IPKHA000	2630	XC	1
				IPKHA000	2640	LA	1
				IPKHA000	2697	LA	1
				IPKIA000	2450	LA	1
				IPKIA000	2472	LA	1
				IPKIA000	2526	LA	1
				IPKIA000	5340	LA	1
				IPKIA000	5478	LA	1
				IPKIA000	5500	LA	1
				IPKIA000	5683	LA	1
				IPKIA000	5712	LA	1
				IPKIA000	5751	LA	1
				IPKJA000	1462	LA	1
				IPKJA000	3668	LA	1
				IPKJA000	3699	LA	1
				IPKJA000	3759	LA	1
				IPKJA000	4571	LA	1
				IPKJA000	4594	LA	1
				IPKJA000	4624	LA	1
				IPKJA000	4852	LA	1
				IPKJA000	4856	LA	1
				IPKKA000	1436	LA	1
				IPKKA001	4517	LA	1
				IPKLA000	1242	XC	1
				IPKLA000	1252	LA	1
				IPKLA000	1362	LA	1
				IPKLA000	2605	XC	1
				IPKMA000	1087	LA	1
				IPKOA000	1201	LA	1
				IPKOA000	1206	LA	1
				IPKOA000	1221	LA	1
				IPKRA000	1586	LA	1
				IPKRA000	1595	LA	1
				IPKRA000	1668	LA	1
PFILE3	PCOMMON	1758	(6DE)	IPKBA000	1589	LA	1
				IPKCC000	1689	LA	1
				IPKCC000	1777	LA	1
				IPKCD001	1621	LA	1
				IPKDA000	2732	LA	1
				IPKDA000	2749	LA	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			CODE
FIELD NAME	DSECT NAME	DISPLACEMENT		CSECT NAME	STMNT NO. (APPROX.)	OPERATION	
		DEC	HEX				
PFILE3	PCOMMON	1758	(6DE)	IPKDA000	3008	LA	1
				IPKDA000	3013	LA	1
				IPKDB000	1454	LA	1
				IPKDB000	1480	LA	1
				IPKDB000	1485	LA	1
				IPKDB000	1520	LA	1
				IPKDB000	1526	LA	1
				IPKDB000	1545	LA	1
				IPKDB000	1553	LA	1
				IPKDB000	2090	LA	1
				IPKDB000	2502	LA	1
				IPKDB000	2517	LA	1
				IPKDB000	2588	LA	1
				IPKEA000	1357	LA	1
				IPKEA000	1366	LA	1
				IPKEA000	1405	LA	1
				IPKEA000	1434	LA	1
				IPKEA000	1475	LA	1
				IPKEA000	1581	LA	1
				IPKEA000	1860	LA	1
				IPKEA000	1865	LA	1
				IPKEA000	1876	LA	1
				IPKEA000	1912	LA	1
				IPKEA000	2095	LA	1
				IPKEA000	2101	LA	1
				IPKFA000	2905	LA	1
				IPKFA000	3062	LA	1
				IPKFA000	3223	LA	1
				IPKFA000	3238	LA	1
				IPKFA000	3366	LA	1
				IPKGA000	1534	LA	1
				IPKGA000	1623	LA	1
				IPKGA000	1739	LA	1
				IPKGA000	1779	LA	1
				IPKHA000	1189	LA	1
				IPKHA000	1233	LA	1
				IPKHA000	2629	XC	1
				IPKHA000	2634	LA	1
				IPKHA000	2652	LA	1
				IPKHA000	2676	LA	1
				IPKHA000	2690	LA	1
				IPKIA000	2117	LA	1
				IPKIA000	2516	LA	1
				IPKIA000	2671	LA	1
				IPKIA000	4422	LA	1
				IPKIA000	4985	LA	1
				IPKIA000	5261	LA	1
				IPKIA000	5492	LA	1
				IPKIA000	5563	LA	1
				IPKIA000	5699	LA	1
				IPKIC000	1006	LA	1
				IPKIC000	1036	LA	1
				IPKJA000	1500	LA	1
				IPKJA000	4838	LA	1
				IPKJA000	4877	LA	1
				IPKKA000	3173	LA	1
				IPKKA000	3178	LA	1
				IPKKA000	3183	LA	1
				IPKKA000	3187	LA	1

MODIFIED AREA				MODIFYING INSTRUCTION			CODE
FIELD NAME	DSECT NAME	DISPLACEMENT		CSECT NAME	STMNT NO. (APPROX.)	OPERATION	
		DEC	HEX				
PFILE3	PCOMMON	1758	(6DE)	IPKKA000	3266	LA	1
				IPKKA000	3271	LA	1
				IPKKA000	3276	LA	1
				IPKKA000	3283	LA	1
				IPKKA000	3625	LA	1
				IPKKA000	3761	LA	1
				IPKKA000	3780	LA	1
				IPKKA000	3817	LA	1
				IPKKA000	3822	LA	1
				IPKKA000	4280	LA	1
				IPKKA000	4322	LA	1
				IPKKA000	4342	LA	1
				IPKKA000	4348	LA	1
				IPKKA001	4441	LA	1
				IPKLA000	2552	LA	1
				IPKLA000	2557	LA	1
				IPKMA000	1078	LA	1
				IPKMA000	1236	LA	1
				IPKNB000	2776	LA	1
				IPKNB000	3393	LA	1
				IPKNB000	3589	LA	1
				IPKPA000	1666	LA	1
				IPKPA000	2467	LA	1
				IPKPA000	4101	LA	1
				IPKQA000	1088	LA	1
				IPKRA000	1075	LA	1
				IPKRA000	1604	LA	1
PFINDPT	PCOMMON	1891	(763)	IPKBA000	2169	MVC	1
				IPKFA000	3112	MVC	1
PFLDCOL	PSTRINGS	1	(1)	IPKCA001	1197	MVI	1
				IPKCA001	2018	STC	1
				IPKCC000	1936	STC	1
				IPKCC000	1974	LA	2
				IPKCC000	2000	LA	1
				IPKEA000	1279	LA	2
				IPKIA000	2181	LA	2
				IPKJA000	4562	MVI	1
PFLDLEN	PSTRINGS	0	(0)	IPKCA001	1196	MVI	1
				IPKCA001	1998	STC	1
				IPKCC000	1610	MVI	1
				IPKCC000	1919	STC	1
				IPKIA000	2202	STC	1
				IPKIA000	2237	MVI	1
				IPKJA000	4561	STC	1
PFLDSRC	PSTRINGS	2	(2)	IPKCA001	1198	MVC	2
				IPKCA001	2021	LA	5
				IPKCA001	2024	MVC	2
				IPKCC000	1939	LA	5
				IPKCC000	1942	MVC	2
				IPKCD001	1236	LA	1
				IPKDA000	2318	LA	1
				IPKDA000	2485	LA	5
				IPKIA000	2185	LA	1
				IPKIA000	2203	LA	1
				IPKIA000	2265	LA	1
				IPKJA000	4576	MVC	2
PGBLASIZ	PCOMMON	1940	(794)	IPKFA000	2928	STCM	1
				IPKIA000	5647	LA	1
PGBLBSIZ	PCOMMON	1943	(797)	IPKFA000	2937	STCM	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
PGBLCSIZ	PCOMMON	1946	(79A)	IPKFA000	2941	STCM	1
PGBLSIZ	PCOMMON	1940	(794)	IPKFA000	3344	XC	1
PGENSW	PCOMMON	1921	(781)	IPKFA000	2959	NI	1
				IPKIA000	1918	NI	1
				IPKIA000	1942	NI	1
				IPKIA000	2375	OI	1
				IPKIA000	2416	OI	1
				IPKIA000	2452	NI	1
				IPKIA000	4461	NI	1
				IPKIA000	4707	OI	1
				IPKIA000	4977	OI	1
				IPKIA000	5256	OI	1
				IPKIA000	5326	OI	1
				IPKIC000	1009	OI	1
PGVENT1	PGVHEAD	7	(7)	IPKIA000	2615	LA	1
				IPKIA000	2617	LA	3
PGVHLEN	PGVHEAD	0	(0)	IPKFA000	2569	STH	1
PHICORE	PCOMMON	1666	(682)	IPKBA000	1299	MVC	1
PIERCNT	PCOMMON	1969	(7B1)	IPKIA000	2000	XC	1
				IPKIA000	2028	XC	1
				IPKIA000	2755	XC	1
				IPKIA000	4393	STH	1
				IPKIA000	5750	XC	1
PIERSTK	PCOMMON	1971	(7B3)	IPKIA000	4390	STC	2
PINPUTPT	PCOMMON	1894	(766)	IPKBA000	2165	MVC	1
				IPKFA000	3108	MVC	1
				IPKTA000	1013	MVC	1
PITEM	EDPMI	10	(A)	IPKCC000	3676	MVC	2
				IPKDB000	1853	LA	1
PLA	PETFLDS	2	(2)	IPKLA000	1579	LA	1
				IPKNB000	3252	LA	5
				IPKNB000	3256	LA	5
PLENATTR	PETR	8	(8)	IPKKA000	1536	STH	1
				IPKKA000	1582	STH	1
				IPKKA000	1644	MVC	2
				IPKKA000	1687	MVC	2
				IPKKA000	1700	MVC	2
				IPKKA000	1818	STH	1
				IPKKA000	1861	MVC	2
				IPKKA000	2056	MVC	2
				IPKKA000	3513	MVC	2
				IPKKA000	3541	MVC	2
				IPKLA000	1457	MVC	1
				IPKNA000	1527	STH	1
				IPKNB000	3038	STH	1
				IPKNB000	3094	STCM	1
				IPKNB000	3289	STCM	1
				IPKOA000	1441	STH	1
				IPKOA000	1681	STH	1
				IPKOA000	1716	STCM	1
				IPKPA000	2077	STH	1
				IPKPA000	2226	STH	1
				IPKPA000	2247	STH	1
				IPKPA000	2345	STH	1
PLINECNT	PCOMMON	1843	(733)	IPKBA000	1339	MVC	5
PLINENUM	PCOMMON	1917	(77D)	IPKQA000	1098	MVC	1
				IPKSA000	3143	MVC	1
PLITBLK	PCOMMON	1904	(770)	IPKJA000	2493	STH	1
				IPKJA000	4030	STH	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
PLITBLK	PCOMMON	1904	(770)	IPKJA000	4882	STH	1
				IPKRA000	1497	STH	1
PLITLEN	PCOMMON	1906	(772)	IPKJA000	4772	MVC	1
				IPKJA000	4891	STH	1
PLOCCNTR	PETR	11	(B)	IPKKA000	1540	STCM	1
				IPKKA000	1708	STCM	1
				IPKKA000	1737	STCM	1
				IPKLA000	1456	MVC	1
				IPKPA000	1806	STCM	1
				IPKPA000	1814	STCM	1
				IPKPA000	2040	STCM	1
				IPKPA000	2112	STCM	1
PLORMIN	EVALSTCK	0	(0)	IPKKA000	2873	MVI	1
				IPKKA000	2875	MVI	1
				IPKKA000	2921	MVC	2
				IPKKA000	2936	MVC	2
				IPKLA000	2250	MVI	1
				IPKLA000	2252	MVI	1
				IPKLA000	2298	MVC	2
				IPKLA000	2314	MVC	2
PLRELATR	PETR	10	(A)	IPKKA000	1543	MVC	1
				IPKKA000	1740	MVC	1
				IPKKA000	2587	MVC	1
PMAVBSIZ	PCOMMON	1936	(790)	IPKFA000	3342	MVC	1
PMAVNO	PCOMMON	1938	(792)	IPKFA000	2187	STH	1
				IPKFA000	3343	XC	1
PMAVNP	PCOMMON	1930	(78A)	IPKFA000	2974	MVC	2
				IPKIA000	5634	LA	1
PMIBLEN	PCOMMON	1951	(79F)	IPKDB000	2842	STH	1
PMODEXP	PDCEDIT	1	(1)	IPKLA000	1644	LA	1
PMODFLAG	PDCEDIT	0	(0)	IPKLA000	1627	MVC	3
				IPKLA000	1634	LA	5
PMODIFS	PDCEDIT	9	(9)	IPKLA000	1628	LA	1
				IPKLA000	1629	LA	5
				IPKNA000	1532	LA	3
				IPKNB000	3018	LA	1
				IPKOA000	1413	LA	1
PNAMCOL	PSTRINGS	1	(1)	IPKCC000	1578	LA	2
				IPKHA000	1658	LA	2
				IPKIC000	942	LA	2
				IPKJA000	2398	LA	2
				IPKJA000	3569	LA	2
PNAME	PETFLDS	1	(1)	IPKJA000	3558	MVC	2
				IPKJA000	3561	LA	2
				IPKKA000	2279	LA	2
				IPKKA000	2351	LA	2
				IPKKA000	3322	LA	2
				IPKLA000	1766	OI	1
				IPKLA000	1770	LA	2
PNAMFLD	PETR	15	(F)	IPKJA000	1568	MVI	1
				IPKJA000	1570	LA	5
				IPKJA000	2608	LA	1
				IPKJA000	2834	LA	3
				IPKJA000	2911	LA	1
				IPKJA000	3695	MVI	1
				IPKJA000	4490	LA	1
				IPKKA000	1554	LA	1
				IPKKA000	1646	MVC	2
				IPKKA000	1709	LA	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
PNAMFLD	PETR	15	(F)	IPKKA000	1820	LA	5
				IPKKA000	1822	LA	1
				IPKKA000	1895	STCM	1
				IPKKA000	2188	LA	1
				IPKKA000	2273	LA	1
				IPKKA000	2346	LA	1
				IPKKA000	2589	MVC	2
				IPKKA000	3318	LA	1
				IPKKA000	3369	MVC	2
				IPKKA000	3380	MVC	2
				IPKKA000	3393	MVC	2
				IPKKA000	3401	LA	1
				IPKKA000	3873	LA	5
				IPKKA000	4189	LA	1
				IPKLA000	1489	LA	1
				IPKLA000	1782	LA	1
				IPKLA000	1813	LA	1
				IPKLA000	1880	LA	3
				IPKLA000	1954	LA	3
PNAMLEN	PSTRINGS	0	(0)	IPKJA000	4554	MVI	1
				IPKPA000	2775	LA	5
				IPKPA000	3093	LA	5
PNAMLNG	PETFLDS	0	(0)	IPKJA000	3560	STC	1
				IPKKA000	3556	MVC	2
				IPKKA000	3557	MVC	3
PNAMSRC	PSTRINGS	2	(2)	IPKCD001	1097	LA	1
				IPKJA000	2579	LA	1
				IPKJA000	3527	LA	1
				IPKPA000	2744	LA	1
				IPKPA000	3070	LA	3
PNEXTNP	PFCB	29	(1D)	IPKAA000	1946	MVC	2
				IPKAA000	2052	LA	1
				IPKAA000	2068	MVC	2
				IPKAA000	2167	LA	1
				IPKAA000	2186	MVC	2
				IPKAA000	2214	LA	1
PNEXTNP3	PCOMMON	1787	(6FB)	IPKHA000	2693	MVC	1
PNPMAC1	PCOMMON	1907	(773)	IPKFA000	3352	MVC	2
				IPKIA000	4592	LA	1
PNPOCGV	PCOMMON	1922	(782)	IPKFA000	1658	MVC	2
				IPKIA000	5736	LA	1
PNPOFFS	PFCB	27	(1B)	IPKAA000	2024	STH	1
				IPKAA000	2393	STH	1
PNPRW	PFCB	21	(15)	IPKAA000	1942	MVC	2
				IPKAA000	2064	MVC	2
				IPKAA000	2181	MVC	2
				IPKEA000	1345	LA	1
				IPKEA000	1709	LA	1
PNXTBKT	PETFLDS	9	(9)	IPKJA000	3161	LA	1
				IPKJA000	3191	LA	1
				IPKKA000	2184	LA	1
				IPKKA000	2326	LA	1
				IPKKA000	2360	LA	1
				IPKKA000	2783	LA	1
				IPKKA000	2793	LA	1
				IPKKA000	2993	LA	1
				IPKKA000	3904	LA	5
				IPKLA000	1540	LA	1
				IPKLA000	1861	LA	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			CODE
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	
PNXTBKT	PETFLDS	9	(9)	IPKLA000	1930	LA	3
				IPKLA000	1979	LA	1
				IPKLA000	2171	LA	1
				IPKLA000	2183	LA	1
				IPKLA000	2375	LA	1
PNXTMOD	PDCEDIT	4	(4)	IPKLA000	1633	LA	1
				IPKLA000	1634	LA	5
				IPKOA000	1704	LA	1
				IPKOA000	1710	LA	1
POLSTR	PETFLDS	2	(2)	IPKKA000	1609	LA	1
				IPKKA000	1758	LA	1
				IPKKA000	1780	LA	1
				IPKKA000	1870	LA	1
				IPKKA000	2578	LA	1
				IPKKA000	3329	LA	1
POPCOL	PSTRINGS	1	(1)	IPKHA000	1667	LA	2
POPDCOL	PSTRINGS	1	(1)	IPKHA000	1320	LA	2
				IPKHA000	1937	LA	2
				IPKJA000	2402	LA	2
POPDSRC	PSTRINGS	2	(2)	IPKCD001	1159	LA	1
				IPKCD001	1412	LA	1
				IPKHA000	1322	LA	1
				IPKJA000	2403	LA	1
				IPKPA000	2786	LA	1
				IPKPA000	2887	LA	1
				IPKPA000	3115	LA	1
				IPKPA000	3245	LA	1
				IPKPA000	3396	LA	1
POPFLAG	PETFLDS	0	(0)	IPKJA000	2741	MVI	1
				IPKLA000	1577	MVC	2
				IPKLA000	1599	MVI	1
				IPKLA000	1606	MVI	1
POPLEN	PSTRINGS	0	(0)	IPKDA000	1980	MVC	2
				IPKPA000	2777	LA	5
				IPKPA000	3095	LA	5
POPNUMB	PETR	4	(4)	IPKJA000	1567	STC	1
				IPKJA000	1786	MVI	1
				IPKJA000	2050	MVI	1
POPSRC	PSTRINGS	2	(2)	IPKIC000	952	LA	1
				IPKJA000	2399	LA	3
PPAGENO	PCOMMON	1919	(77F)	IPKMA000	1531	AP	1
				IPKMA000	1612	AP	1
				IPKPA000	3938	AP	1
				IPKQA000	1369	AP	1
				IPKRA001	2123	AP	1
				IPKSA000	3186	AP	1
				IPKSB000	1835	AP	1
PRECLEN	PFCB	9	(9)	IPKMA000	1094	MVC	1
				IPKMA000	1193	MVC	2
PREFCNT	PCOMMON	1928	(788)	IPKKA000	3165	ST	1
				IPKKA000	3260	ST	1
				IPKKA001	4503	XC	1
				IPKLA000	2544	ST	1
PROGID	PCOMMON	1845	(735)	IPKCD001	1088	MVC	1
				IPKCD001	1119	MVC	2
				IPKJA000	2570	MVC	1
				IPKJA000	2578	LA	1
				IPKJA000	2589	MVC	1
				IPKMA000	1171	TR	1

MODIFIED AREA				MODIFYING INSTRUCTION			CODE
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	
PSAVPT	PCOMMON	1833	(729)	IPKCC000	1461	XC	1
				IPKDA000	1531	XC	1
				IPKFA000	2164	XC	1
				IPKIA000	4710	XC	1
				IPKJA000	1478	XC	1
				IPKNA000	1474	XC	1
				IPKTA000	1015	XC	1
PSFLAG2	PETFLDS	0	(0)	IPKKA000	2315	OI	1
PSIGN	PETFLDS	9	(9)	IPKNB000	3196	LA	1
				IPKNB000	3252	LA	5
				IPKNB000	3256	LA	5
				IPKNB000	3328	LA	1
				IPKNB000	3526	LA	1
				IPKNB000	3572	LA	1
PSLENATR	PETFLDS	1	(1)	IPKKA000	2314	MVC	2
				IPKKA000	4146	MVC	5
				IPKLA000	1538	MVC	2
				IPKLA000	1842	MVC	2
PSPILLA	PCOMMON	1963	(7AB)	IPKIA000	5341	LA	1
				IPKIA000	5359	MVC	2
				IPKIA000	5610	MVC	1
PSRELATR	PETFLDS	3	(3)	IPKKA000	2183	MVI	1
				IPKKA000	4153	MVC	5
				IPKKA000	4180	STC	5
PSTMCSEQ	PCOMMON	1910	(776)	IPKAC000	855	AP	1
				IPKGA000	1717	AP	1
PSTRCOL	PSTRINGS	1	(1)	IPKCA001	1585	LA	2
				IPKCA001	1964	STC	1
				IPKCC000	1607	LA	2
				IPKCC000	1853	LA	2
				IPKCC000	3911	STC	1
				IPKHA000	1719	LA	2
PSTRLEN	PSTRINGS	0	(0)	IPKCA001	1949	STC	1
				IPKCC000	3906	STC	1
PSTRSRC	PSTRINGS	2	(2)	IPKCA001	1965	LA	1
				IPKCC000	3912	LA	1
				IPKPA000	3695	LA	5
				IPKPA000	3830	LA	5
PSVALUE	PETFLDS	4	(4)	IPKKA000	2182	XC	1
PSYMBOL	PETFLDS	1	(1)	IPKJA000	3138	MVC	1
				IPKJA000	3166	MVC	2
				IPKJA000	3189	MVC	1
				IPKJA000	3194	MVC	2
				IPKLA000	1974	OI	1
PSYMNO	PETR	14	(E)	IPKJA000	1517	MVI	1
				IPKJA000	2607	LA	1
				IPKJA000	2835	MVI	1
				IPKJA000	2840	STC	1
				IPKKA000	1645	MVI	1
				IPKKA000	1819	MVI	1
				IPKKA000	1862	MVI	1
				IPKKA000	2141	STC	1
				IPKKA000	2588	MVI	1
				IPKKA000	2595	MVI	1
				IPKKA000	2645	MVI	1
				IPKLA000	1455	STCM	1
				IPKLA000	1547	LA	1
				IPKLA000	1664	LA	1
				IPKLA000	1694	LA	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
PSYMNO	PETR	14	(E)	IPKNA000	1476	LA	1
				IPKNA000	1883	LA	1
				IPKNB000	2795	LA	1
				IPKNB000	2798	LA	1
				IPKOA000	1225	LA	1
				IPKOA000	1228	LA	1
				IPKOA000	1361	LA	1
				IPKPA000	1828	LA	1
				IPKPA000	1953	LA	1
				IPKPA000	2054	LA	1
				IPKPA000	2206	LA	1
PSYMTABL	PCOMMON	2014	(7DE)	IPKKA001	4502	STCM	1
PSYSNDX	PCOMMON	1922	(782)	IPKIA000	4482	ST	1
				IPKIA000	5749	XC	1
PSYSPSTR	PCOMMON	1876	(754)	IPKBA000	1375	TR	1
				IPKSB000	1619	LA	1
PVSDSIZE	PCOMMON	1899	(76B)	IPKBA000	2016	STH	1
PWAADDR	PFCB	14	(E)	IPKDA000	1796	STCM	3
				IPKDA000	1800	STCM	3
				IPKFA000	2953	MVC	5
PWAADDR1	PCOMMON	1702	(6A6)	IPKBA000	2229	STCM	1
				IPKEA000	2033	STCM	1
				IPKFA000	3135	STCM	1
				IPKJA000	4796	STCM	1
				IPKKA000	3295	MVC	1
				IPKKA001	4465	STCM	1
				IPKPA000	3988	STCM	1
				IPKPA000	4041	STCM	1
PWAADDR2	PCOMMON	1737	(6C9)	IPKBA000	2219	STCM	1
				IPKEA000	2023	STCM	1
				IPKFA000	3134	STCM	1
				IPKJA000	4786	STCM	1
				IPKKA000	3293	MVC	1
				IPKKA001	4466	STCM	1
PWAADDR3	PCOMMON	1772	(6EC)	IPKBA000	2234	STCM	1
				IPKHA000	1185	MVC	1
				IPKHA000	2694	MVC	1
				IPKPA000	4098	STCM	1
P0	PCOMMON	90	(5A)	IPKNA000	1583	STH	2
				IPKNA000	1583	STH	2
				IPKNA000	1666	ST	2
				IPKNA000	1666	ST	2
				IPKNA000	1667	MVI	2
				IPKNA000	1667	MVI	2
				IPKNA000	1944	STC	2
				IPKNA000	1944	STC	2
				IPKNA000	2250	ST	2
				IPKNA000	2250	ST	2
				IPKNA000	2251	MVC	2
				IPKNA000	2251	MVC	2
				IPKNA000	2515	MVC	2
				IPKNA000	2515	MVC	2
P1	PCOMMON	90	(5A)	IPKNA000	1886	MVC	2
				IPKNA000	1888	LA	2
				IPKNA000	1933	UNPK	2
				IPKNA000	1934	NI	2
				IPKNA000	1938	MVC	2
				IPKNA000	1946	LA	2
				IPKNA000	1969	LA	2

MODIFIED AREA				MODIFYING INSTRUCTION			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
P1	PCOMMON	90	(5A)	IPKNA000	1980	LA	2
				IPKNA000	2048	LA	1
				IPKNA000	2064	LA	2
				IPKNA000	2381	LA	2
P2	PCOMMON	90	(5A)	IPKNA000	1584	LA	2
				IPKNA000	1929	LA	1
				IPKNA000	1935	NI	2
				IPKNA000	1967	LA	2
				IPKNA000	1971	MVC	2
				IPKNA000	1973	LA	1
				IPKNA000	2329	LA	2
				IPKNA000	2372	LA	2
				IPKNA000	2508	LA	2
P4	PCOMMON	90	(5A)	IPKNA000	1668	LA	2
				IPKNA000	2252	LA	2
				IPKNA000	2509	LA	3
P7	PCOMMON	90	(5A)	IPKJA000	1922	LA	2
				IPKNA000	2360	LA	2
P8	PCOMMON	90	(5A)	IPKNA000	1919	LA	1
				IPKNA000	1976	LA	1
				IPKNA000	2347	LA	1
P9	PCOMMON	90	(5A)	IPKKA001	4429	LA	2
				IPKNA000	1972	LA	2
RANR	EVALSTCK	0	(0)	IPKKA000	2786	MVI	1
				IPKKA000	2927	STC	1
				IPKKA000	2946	STC	1
				IPKKA000	2952	STC	1
				IPKKA000	2968	MVI	1
				IPKLA000	2175	MVI	1
				IPKLA000	2304	STC	1
				IPKLA000	2325	STC	1
				IPKLA000	2330	STC	1
				IPKLA000	2346	MVI	1
RDATAD	WORKDTF	129	(81)	IPKAA002	1560	STCM	1
RDATLEN	WORKDTF	134	(86)	IPKAA002	1561	MVC	1
RELLEN	EVALSTCK	7	(7)	IPKKA000	2789	MVC	1
				IPKLA000	2178	MVC	1
RLADDR	RLDENTRY	3	(3)	IPKNB000	3386	STCM	1
RLADID	RLDENTRY	0	(0)	IPKNB000	3378	MVC	1
RLDNXT	RLDENTRY	6	(6)	IPKNB000	3388	LA	1
				IPKQA000	1152	LA	1
				IPKQA000	1189	LA	1
				IPKQA000	1422	LA	1
RLFLAG	RLDENTRY	2	(2)	IPKNB000	3380	MVC	1
				IPKNB000	3384	OI	1
RLREFID	RLDENTRY	1	(1)	IPKNB000	3379	MVC	1
SAVADDR	PCOMMON	2027	(7EB)	IPKKA001	4473	STCM	1
SAVAR	PCOMMON	2030	(7EE)	IPKKA000	1622	MVC	1
				IPKKA000	1770	MVC	2
SAVESDNP	PCOMMON	1961	(7A9)	IPKKA000	3670	MVC	1
				IPKKA000	3701	MVC	1
				IPKKA000	3762	LA	1
				IPKKA000	3777	MVC	1
				IPKKA000	3806	MVC	1
				IPKKA000	4291	MVC	1
				IPKKA001	4451	MVC	1
				IPKKA001	4512	MVC	1
SAVREG1	PCOMMON	2036	(7F4)	IPKJA000	1679	ST	1
				IPKKA000	1939	ST	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
SAVREG1	PCOMMON	2036	(7F4)	IPKKA000	2274	ST	1
				IPKLA000	1578	ST	1
SAVREG2	PCOMMON	2040	(7F8)	IPKJA000	1794	ST	1
				IPKJA000	2098	ST	1
				IPKKA000	2280	ST	1
				IPKLA000	1640	ST	1
SDEFC	EPAR	6	(6)	IPKIA000	4930	LA	5
SDEFITEM	EDPMI	10	(A)	IPKCC000	3260	LA	5
SDEFK	EPAR	5	(5)	IPKIA000	3644	LA	1
SDITEMB	EDPMI	11	(B)	IPKCC000	3309	STCM	1
SDITEMK	EDPMI	14	(E)	IPKCC000	3267	STC	1
SDITEMST	EDPMI	15	(F)	IPKCC000	3260	LA	5
				IPKCC000	3315	MVC	2
SDITEMT	EDPMI	10	(A)	IPKCC000	3266	MVI	1
SECTCL	EPAR	11	(B)	IPKIA000	3619	LA	1
				IPKIA000	4493	MVC	2
SECTFLAG	EPAR	10	(A)	IPKIA000	4494	MVI	1
SEQFLD	EDPMI	18	(12)	IPKCC000	1528	MVC	1
SMTEFLG	SMTENT	9	(9)	IPKEA000	1730	MVI	1
				IPKEA000	1766	MVI	1
				IPKEA000	2042	MVI	1
				IPKFA000	3178	MVI	1
SMTLEN	SMTENT	9	(9)	IPKEA000	1793	MVC	2
SMTNAME	SMTENT	10	(A)	IPKEA000	1675	LA	5
				IPKEA000	1765	LA	5
SMTNP	SMTENT	1	(1)	IPKEA000	1761	MVC	1
				IPKFA000	1837	LA	1
				IPKGA000	1227	LA	1
SMTSIZE	PCOMMON	1897	(769)	IPKBA000	2043	STH	1
SSDADDR	PCOMMON	1953	(7A1)	IPKDA000	2737	LA	1
				IPKDA000	2754	LA	1
				IPKDB000	1460	LA	1
				IPKDB000	1531	LA	1
				IPKDB000	2859	STCM	1
				IPKEA000	2057	STCM	1
SSDEND	PCOMMON	2014	(7DE)	IPKDA000	2726	STCM	1
				IPKDB000	2084	STCM	1
SSDFLAG	SSD	3	(3)	IPKDA000	2725	MVI	1
				IPKDA000	2746	MVI	1
				IPKDB000	1449	LA	5
				IPKDB000	2083	MVI	1
				IPKGA000	1680	MVI	5
				IPKGA000	1681	LA	5
SSDINFO	PCOMMON	1953	(7A1)	IPKDA000	2661	LA	1
SSDNP	PCOMMON	1936	(790)	IPKDB000	1469	MVC	2
				IPKDB000	1472	MVC	1
				IPKDB000	1521	LA	1
				IPKEA000	1406	LA	1
SSDOFFS	SSD	0	(0)	IPKDA000	2723	MVC	1
SSDSIZE	PCOMMON	1956	(7A4)	IPKDB000	1451	STH	1
				IPKDB000	1465	MVC	1
				IPKDB000	2857	STH	1
SSDSYM	SSD	4	(4)	IPKEA000	1546	LA	2
				IPKEA000	1892	LA	5
SSDSYML	SSD	3	(3)	IPKDA000	2771	MVC	2
SSITEMK	EDPMI	10	(A)	IPKCC000	3510	STC	1
STABEND	PCOMMON	1985	(7C1)	IPKKA001	4487	MVC	1
				IPKKA001	4493	ST	1
STARTLOC	PCOMMON	1944	(798)	IPKKA000	3347	STCM	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
STARTLOC	PCOMMON	1944	(798)	IPKKA001	4506	MVC	1
STLENGTH	EVALSTCK	0	(0)	IPKKA000	2942	STH	1
				IPKKA000	2956	STH	2
				IPKLA000	2321	STH	1
				IPKLA000	2334	STH	2
STMTNR	PCOMMON	1914	(77A)	IPKJA000	2728	STH	1
				IPKJA000	4884	MVC	1
				IPKKA000	1445	MVC	1
STRPTR	ERRENT	2	(2)	IPKDA000	2363	STCM	1
				IPKDB000	2660	STCM	1
SUBE	EPAR	7	(7)	IPKIA000	3685	LA	1
				IPKIA000	3721	LA	1
				IPKIA000	3874	LA	1
SUBEFLAG	EPAR	5	(5)	IPKIA000	3711	LA	1
				IPKIA000	3858	LA	1
				IPKIA000	4750	MVI	1
				IPKIA000	4799	LA	1
SUBEL	EPAR	6	(6)	IPKIA000	4749	STC	1
SUBLFLAG	EPAR	0	(0)	IPKIA000	4797	MVI	1
SUBLK	EPAR	4	(4)	IPKIA000	4798	MVC	1
SUBLL	EPAR	1	(1)	IPKIA000	4830	STH	1
SUBLN	EPAR	3	(3)	IPKIA000	4831	MVC	1
SWMH	MACHEAD	6	(6)	IPKEA000	1381	MVC	2
SWSMT	SMTENT	0	(0)	IPKEA000	1760	MVC	1
SYMADDR	PCOMMON	1981	(7BD)	IPKKA001	4485	ST	1
SYMESDID	PHYR	5	(5)	IPKKA000	2436	MVC	1
				IPKKA000	3598	MVC	1
				IPKKA000	4074	MVC	1
				IPKKA000	4098	STC	1
SYMFLAGS	PHYR	2	(2)	IPKKA000	1639	OI	1
				IPKKA000	2437	MVC	1
				IPKKA000	2522	MVI	1
				IPKKA000	2527	MVI	1
				IPKKA000	3476	OI	1
				IPKKA000	3485	OI	1
				IPKKA000	3599	MVC	1
				IPKKA000	3601	OI	1
				IPKKA000	3990	OI	1
				IPKKA000	4002	MVI	1
				IPKKA000	4008	OI	1
				IPKKA000	4053	MVI	1
				IPKKA000	4057	OI	1
				IPKKA000	4110	MVI	1
				IPKKA000	4206	MVC	1
				IPKKA000	4235	MVC	1
				IPKKA000	4236	OI	1
				IPKKA000	4238	MVI	1
				IPKLA000	1748	NI	1
				IPKLA000	1852	OI	1
				IPKLA000	1914	OI	1
				IPKLA000	1978	MVI	1
SYMLATTR	PHYR	3	(3)	IPKKA000	2435	MVC	1
				IPKKA000	2459	MVC	2
				IPKKA000	3447	MVC	1
				IPKKA000	3474	MVC	2
				IPKKA000	4054	MVC	1
				IPKKA000	4207	MVC	2
				IPKKA000	4231	MVC	2
SYMLENG	PHYR	9	(9)	IPKKA000	2440	STC	1

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
SYMLENG	PHYR	9	(9)	IPKKA000	3455	STC	1
				IPKKA000	4001	STC	1
				IPKKA000	4051	STC	1
				IPKKA000	4209	STC	1
SYMSRC	PHYR	10	(A)	IPKKA000	2452	MVC	2
				IPKKA000	4121	MVI	1
SYMVALUE	PHYR	6	(6)	IPKKA000	2438	MVC	1
				IPKKA000	3453	MVC	1
				IPKKA000	4055	MVC	1
SYM1C	EPAR	9	(9)	IPKIA000	4915	LA	5
SYM1K	EPAR	8	(8)	IPKIA000	3653	LA	1
SYM2C	EPAR	5	(5)	IPKIA000	4920	LA	5
SYM2K	EPAR	4	(4)	IPKIA000	3680	LA	1
S1ITEMI	EDPMI	15	(F)	IPKHA000	2411	STH	1
S1ITEMK	EDPMI	17	(11)	IPKHA000	2406	STC	1
S1ITEMST	EDPMI	18	(12)	IPKCC000	3198	LA	5
				IPKHA000	2438	MVC	2
				IPKIA000	4880	LA	5
S1ITEMT	EDPMI	10	(A)	IPKCC000	3198	LA	5
				IPKHA000	2405	MVC	2
				IPKIA000	4880	LA	5
S2ITEMK	EDPMI	13	(D)	IPKHA000	2423	STC	1
S2ITEMST	EDPMI	14	(E)	IPKHA000	2439	MVC	2
S2ITEMT	EDPMI	10	(A)	IPKHA000	2422	MVC	2
THISELEM	PETFLDS	0	(0)	IPKJA000	2381	MVC	2
				IPKJA000	2388	MVC	2
				IPKJA000	2411	LA	3
				IPKJA000	2437	MVC	1
				IPKJA000	2447	MVI	1
TYPEESD	PCOMMON	1995	(7CB)	IPKKA000	3370	MVI	1
				IPKKA000	3381	MVI	1
				IPKKA000	3389	MVI	1
				IPKKA000	3394	MVI	1
				IPKKA000	3560	MVI	1
VERTYPE	ESDENTRY	0	(0)	IPKMA000	1349	MVI	1
VRDA	WORKDTF	128	(80)	IPKAA002	1557	MVI	1
VRDAT	WORKDTF	96	(60)	IPKAA000	1930	MVI	1
				IPKAA002	1700	MVI	1
VSDADDR	PCOMMON	1977	(7B9)	IPKDB000	2095	LA	1
				IPKDB000	2852	STCM	1
VSDDIM	VSD	4	(4)	IPKDB000	2410	STH	2
VSDEND	PCOMMON	2011	(7DB)	IPKDB000	1580	STCM	1
				IPKDB000	2423	STCM	1
VSDFLAG	VSD	3	(3)	IPKDB000	1579	MVI	1
				IPKDB000	2076	LA	5
				IPKDB000	2422	MVI	1
				IPKDB000	2479	LA	5
				IPKDB000	2584	MVI	1
				IPKDB000	2636	LA	5
				IPKGA000	1619	MVI	5
				IPKGA000	1620	LA	5
VSDINFO	PCOMMON	1977	(7B9)	IPKDA000	4133	LA	1
VSDNDX	VSD	1	(1)	IPKDB000	2406	STH	1
VSDSIZE	PCOMMON	1980	(7BC)	IPKDB000	2078	STH	1
				IPKDB000	2100	MVC	1
				IPKDB000	2853	MVC	1
VSDSYM	VSD	4	(4)	IPKDA000	4153	LA	2
				IPKDA000	4155	LA	5
				IPKDB000	2458	LA	2

---------MODIFIED AREA----------				----MODIFYING INSTRUCTION-----			
FIELD NAME	DSECT NAME	DISPLACEMENT DEC	HEX	CSECT NAME	STMNT NO. (APPROX.)	OPERATION	CODE
VSDSYM	VSD	4	(4)	IPKDB000	2461	LA	5
VSDSYML	VSD	3	(3)	IPKDB000	2387	MVC	2
VSDTYPE	VSD	0	(0)	IPKDB000	2399	MVC	1
VWDA	WORKDTF	128	(80)	IPKAA002	1592	MVI	1
WCC	WORKDTF	56	(38)	IPKAA002	1530	STCM	1
				IPKAA002	1706	MVC	1
WCCHH	WORKDTF	56	(38)	IPKAA002	1724	MVC	1
WCCHHR	WORKDTF	56	(38)	IPKAA002	1576	MVC	1
WCCW	WORKDTF	112	(70)	IPKBA000	1524	MVC	1
WCCW2	WORKDTF	120	(78)	IPKBA000	1534	MVI	1
WCNTCH	WORKDTF	144	(90)	IPKAA002	1541	MVC	1
				IPKAA002	1571	STCM	1
WCNTCHR	WORKDTF	144	(90)	IPKAA002	1716	MVC	1
WCNTDL	WORKDTF	150	(96)	IPKAA002	1519	STH	1
WCNTR	WORKDTF	148	(94)	IPKAA002	1540	STC	1
WDATAD	WORKDTF	121	(79)	IPKAA002	1516	STCM	1
WDATLEN	WORKDTF	126	(7E)	IPKAA002	1518	STH	1
WDEVTYP	WORKDTF	29	(1D)	IPKBA000	1517	MVC	2
WEFFRLEN	WORKDTF	68	(44)	IPKBA000	1904	STH	1
				IPKBA000	2077	STH	1
WH	WORKDTF	59	(3B)	IPKAA002	1531	STCM	1
				IPKAA002	1707	MVC	2
WMPCTY	WORKDTF	62	(3E)	IPKBA000	1519	MVC	2
WPCTY	WORKDTF	30	(1E)	IPKAA000	1934	XC	1
				IPKAA002	1545	STH	1
				IPKAA002	1573	STH	1
				IPKAA002	1714	MVC	1
				IPKAA002	1725	MVC	1
WREC	WORKDTF	60	(3C)	IPKAA002	1536	STC	1
				IPKAA002	1579	MVI	1
				IPKAA002	1726	MVI	1
WRECLEN	WORKDTF	40	(28)	IPKAA000	2270	MVC	1
				IPKAA000	2435	MVC	1
WTRKPCYL	WORKDTF	71	(47)	IPKBA000	1917	STC	1
				IPKBA000	2090	STC	1
XDEFEND	XREFREC	17	(11)	IPKKA000	3209	LA	1
				IPKLA000	2586	LA	1
XRAREND	PCOMMON	1970	(7B2)	IPKKA001	4478	STCM	1
XREFADDR	PCOMMON	1999	(7CF)	IPKKA000	3179	LA	1
				IPKLA000	2558	LA	1
XREFEND	XREFREC	11	(B)	IPKKA000	3214	LA	1
				IPKLA000	2590	LA	1
XREFLEN	PCOMMON	2002	(7D2)	IPKKA000	3163	STH	1
				IPKKA000	3255	STH	1
				IPKLA000	2542	STH	1
XREFPARM	PCOMMON	1998	(7CE)	IPKKA000	3272	LA	3
				IPKKA001	4482	ST	1
XREFPTR	PCOMMON	1967	(7AF)	IPKKA000	3210	STCM	1
				IPKKA001	4480	STCM	1
				IPKLA000	2587	STCM	1
XRFLAG	XREFREC	8	(8)	IPKKA000	3204	MVC	1
				IPKLA000	2581	MVC	1
XRFLSRC	XRFENTRY	12	(C)	IPKRA001	2017	LA	1
XRFPSEUD	XRFENTRY	0	(0)	IPKRA000	1102	TR	1
XRFSYM	XRFENTRY	0	(0)	IPKRA000	1100	TR	1
XRLATTR	XREFREC	11	(B)	IPKKA000	3208	MVC	2
				IPKLA000	2585	MVC	2
XRSN	XREFREC	9	(9)	IPKKA000	3199	MVC	1
				IPKLA000	2576	MVC	1
XRSYMBOL	XREFREC	0	(0)	IPKKA000	3200	MVC	1
				IPKLA000	2577	MVC	1

Appendix J: APAR Documentation for the Assembler

Certain material should be sent to the assembler maintenance group along with the submission of an APAR. The material is required in order to successfully recreate the problem. The following information should accompany an APAR:

- Information on the CPU model, main storage size, and devices used
- DOS release level
- Partition size
- Source cards or tape with the failing program
- User-written macros and COPY code
- Main storage dump if applicable
- PTFs applied
- Output showing the error
- Any other material necessary for this particular problem
- Cataloged procedures used

Note: APARs concerning errors in system macro definitions should not be sent to the assembler maintenance group. Only assembler language errors, including those in which the assembler has failed to expand the macro instruction correctly according to the language manual, are valid APARs against the assembler.

Index

Part 2 — ESERV Logic

Organization of Part 2

- Introduction
- Method of Operation
- Program Organization
- Data Areas
- Diagnostic Aids

Contents

Figures

Introduction

Three object modules of the ESERV program are written in assembler language: IPKAD, IPKVA, and IPKVB. All other modules are written in Programming Language System (PL/S) II. For information on reading and interpreting PL/S II program listings see Guide to PL/S Generated Listings, Order No. GC28-6786.

Size of the ESERV Program

The minimum virtual partition size required by the ESERV program is 24K.

Purpose and Function of ESERV

The ESERV program generates a complete source macro definition from an edited macro. Several macros can be "de-edited" in one run of ESERV and macros can be updated in combination with the de-editing. For a complete description of the ESERV program and how to use it see Guide to the DOS/VSE Assembler.

Environmental Characteristics

SYSTEM CONFIGURATION

The configuration required is the same as that required by the DOS/VSE Assembler.

SYSTEM INTERFACES

System-dependent functions and operations of ESERV are centralized in interface modules to allow relative ease of modification for new features of the Disk Operating System. The names and functions of these modules are listed below.

IPKVA	Interface logic, I/O logic, common data area (COMMON) and initialization code.
IPKAD	SYSSLB logic module (DTFSL)

Interface macros used by ESERV to provide service functions and to call for functions from the interface modules, are described in Part 1, Appendix C, 'Macro Usage'.

Physical Considerations

The ESERV program is made up of 7 phases residing on a core image library. See "Program Organization" for a table showing the phases, control sections, and object modules of ESERV.

Operational Considerations

INPUT

Input to ESERV is as follows:

Control statements (cards, disk, or tape)	SYSIPT
Edited macro definitions (sublibrary on a source statement library)	SYSRES/SYSSLB

For a complete description of the input see Guide to the DOS/VSE Assembler.

OUTPUT

Output for ESERV is as follows:

Source format macro definition (updated if requested)	SYSLST/SYSPCH

If ESERV is run with the UPDATE option, an update survey listing, each update control statement, and the affected source record(s) are printed on SYSLST. A statement number is attached to each statement to enable a list of error references to be printed following the macro definition. For a complete description of the output see Guide to the DOS/VSE Assembler.

CONTROL INFORMATION

The user specifies options of the ESERV program through special control statements. These control statements are fully described in Guide to the DOS/VSE Assembler.

Method of Operation

Purpose of the Section

The purpose of this section is to give a functional description of the ESERV program and to provide a cross-reference from this description to other parts of the manual and the program listings.

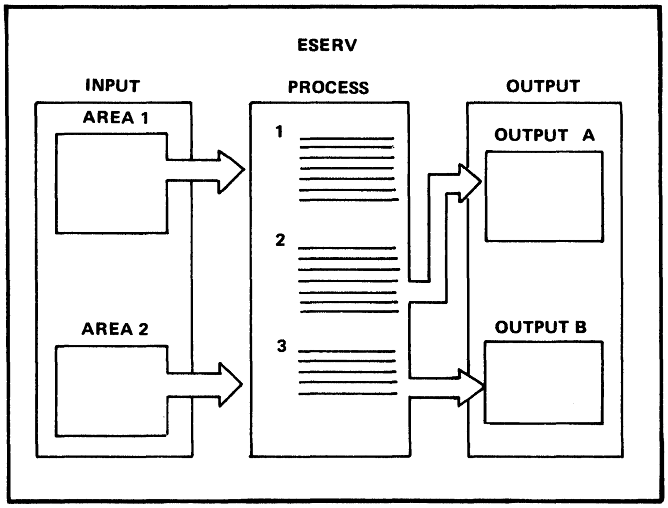
HOW TO READ THE DIAGRAM AND DESCRIPTION

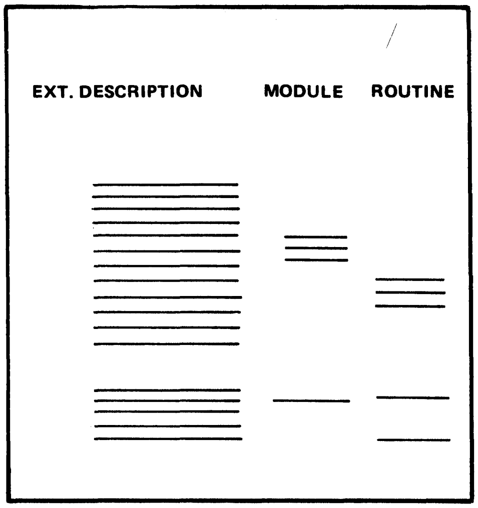
The following diagram illustrates:

- Input - showing what the data is and where it is from
- Process - describing how the data is processed by ESERV
- Output - showing where the data goes

Data areas are identified on the diagram in two ways: main storage address (upper case, parentheses), and by PL/S II structure name (upper case, underlined).

The extended description is related to the diagram by numbered process steps. In addition, the description supplies the names of the modules and routines which perform the function. Start reading the process block and refer to the input and output as you proceed through the diagram. Use the extended description if you require more detailed information.

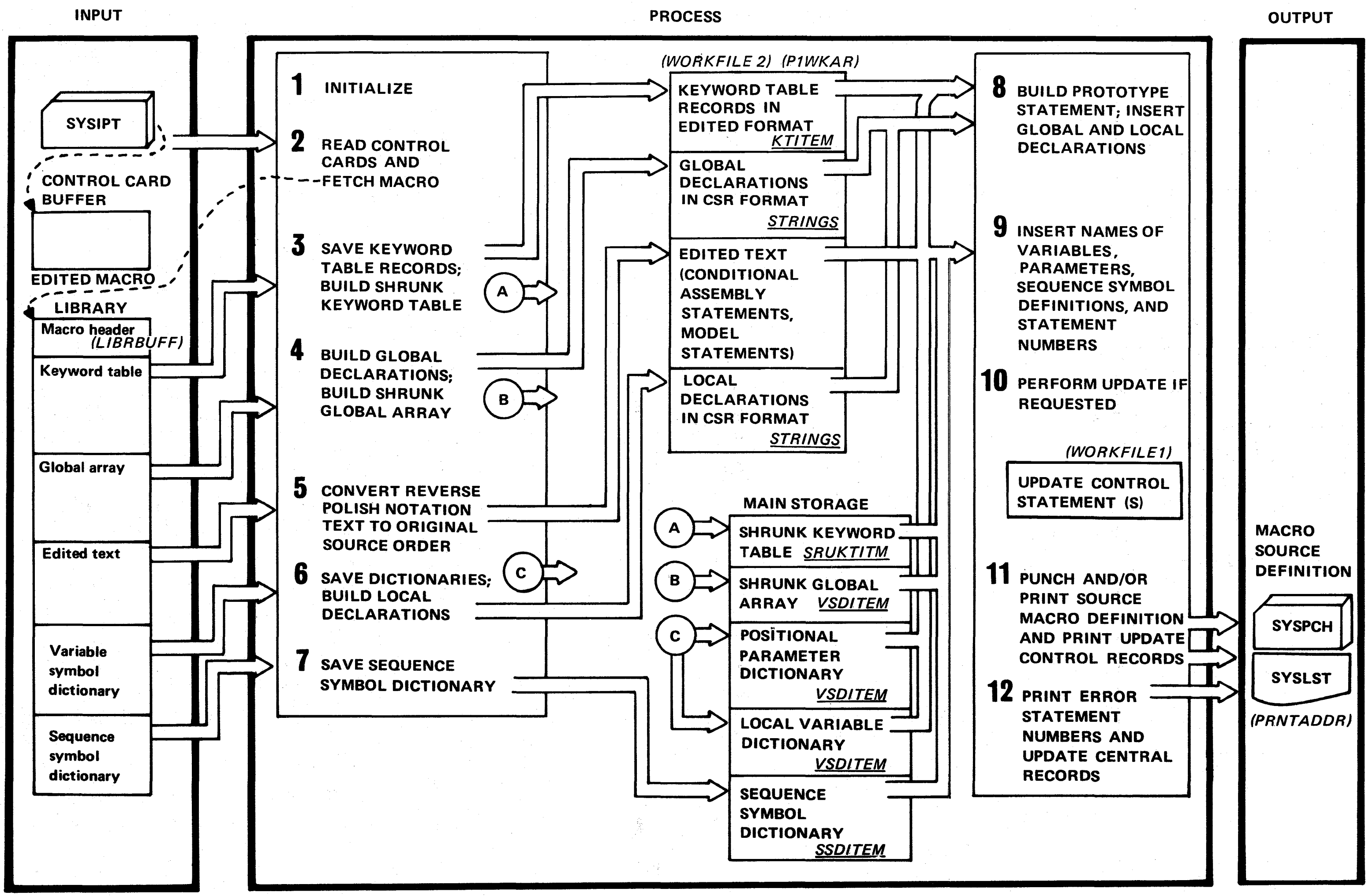




The following symbols are used in the diagram.

⇒	Data flow	- - - ->	Data reference
→	Pointer	*(LIBRBUFF)*	Main storage address
➡	Control flow	***STRINGS***	Structure name

ESERV


INPUT
PROCESS
OUTPUT
SYSIPT
CONTROL CARD BUFFER
EDITED MACRO
LIBRARY
Macro header
(LIBRBUFF)
Keyword table
Global array
Edited text
Variable symbol dictionary
Sequence symbol dictionary
1 INITIALIZE
2 READ CONTROL CARDS AND FETCH MACRO
3 SAVE KEYWORD TABLE RECORDS; BUILD SHRUNK KEYWORD TABLE
A
4 BUILD GLOBAL DECLARATIONS; BUILD SHRUNK GLOBAL ARRAY
B
5 CONVERT REVERSE POLISH NOTATION TEXT TO ORIGINAL SOURCE ORDER
6 SAVE DICTIONARIES; BUILD LOCAL DECLARATIONS
C
7 SAVE SEQUENCE SYMBOL DICTIONARY
(WORKFILE 2) (P1WKAR)
KEYWORD TABLE RECORDS IN EDITED FORMAT
KTITEM
GLOBAL DECLARATIONS IN CSR FORMAT
STRINGS
EDITED TEXT (CONDITIONAL ASSEMBLY STATEMENTS, MODEL STATEMENTS)
LOCAL DECLARATIONS IN CSR FORMAT
STRINGS
MAIN STORAGE
A
SHRUNK KEYWORD TABLE SRUKTITM
B
SHRUNK GLOBAL ARRAY VSDITEM
C
POSITIONAL PARAMETER DICTIONARY
VSDITEM
LOCAL VARIABLE DICTIONARY
VSDITEM
SEQUENCE SYMBOL DICTIONARY
SSDITEM
8 BUILD PROTOTYPE STATEMENT; INSERT GLOBAL AND LOCAL DECLARATIONS
9 INSERT NAMES OF VARIABLES, PARAMETERS, SEQUENCE SYMBOL DEFINITIONS, AND STATEMENT NUMBERS
10 PERFORM UPDATE IF REQUESTED
(WORKFILE1)
UPDATE CONTROL STATEMENT (S)
11 PUNCH AND/OR PRINT SOURCE MACRO DEFINITION AND PRINT UPDATE CONTROL RECORDS
12 PRINT ERROR STATEMENT NUMBERS AND UPDATE CENTRAL RECORDS
MACRO SOURCE DEFINITION
SYSPCH
SYSLST
(PRNTADDR)

EXTENDED DESCRIPTION	MODULE	ROUTINE
2. ESERV control records are read in from SYSIPT and the specified macro is fetched from the macro library. The edited macro records are read in and processed one at a time. Update control records are kept in the control card buffer for later processing (see step 10).	IPKVD	MAIN1
3. Keyword table records are saved in their edited format on workfile 2. A shrunk keyword table -- containing the length and name of the symbol -- is built and kept in main storage to resolve any keyword parameter references in the edited text.	IPKVE	KTPROC1 (BLKBLD)
4. Global declarations are built in compressed source record format and saved on workfile 2. A shrunk global array -- containing the type, index, length, and name (and dimension if present) of the symbol -- is built and kept in main storage to resolve any global variable references in the edited text.	IPKVE	GAPROC (BLKBLD)
5. Text written in reverse Polish notation is converted back to the original source order among operators and operands. References to symbolic variables remain in index number format. An offset value is placed following the common header of each record to enable the insertion of sequence symbol definitions (see step 9).	IPKVF	EDTXT (DEPO)
6. The variable symbol dictionary is split into two separate dictionaries: the positional parameter dictionary and the local variable dictionary. Local declarations are built in compressed source record format from the variable symbol dictionary, immediately before its division, and they are saved on workfile 2. The positional parameter and local variable dictionaries are then saved in main storage for later processing (see steps 8 and 9).	IPKVG	VSDPROC LVDPROC
7. The sequence symbol dictionary is read from the library and saved in main storage for later processing (see steps 8 and 9).	IPKVG	SSDPROC
8. Keyword parameters are read from the keyword table and positional parameters are read from the positional parameter dictionary in main storage. The parameters are combined to build the macro prototype statement. Global and local declarations are read in from workfile 2 and decompressed.	IPKVI	PROTYP KTPROC2 DECL

EXTENDED DESCRIPTION (continued)	MODULE	ROUTINE
9. All index references to variables and parameters are replaced by the actual variable names. Sequence symbol definitions are inserted into their proper locations by means of ANOPs. A statement number is attached to each statement and any error references are written on workfile 2.	IPKVI	NAMEINS
10. The sequence number of the record is compared with the sequence number of the current update control statement in the control card buffer. If the numbers match, the updating is performed. The update control statement is then saved on workfile 1 for later printing (see step 11) and the next update control statement is read.	IPKVK	
11. The reconstructed source macro definition is printed and/or punched on SYSLST/SYSPCH. A complete list of all update changes is printed on SYSLST following the MEND statement.	IPKVM	
12. A list of error statement numbers from workfile 2 is printed on SYSLST following the macro definition and update changes.	IPKVI	ERRPROC

Overflow and search technique

Each overflow block starts with an entry containing the index number (the offset value in the case of sequence symbol dictionary entries) of the last variable in the block. This entry is used to help speed up the searching process. The entry is three bytes long in all dictionaries except the local variable dictionary which has three separate three-byte entries: one for LCLA, one for LCLB, and one for LCLC variables. Each of the three types of local variables has its own index series.
If overflow occurs, the dictionary -- shrunk keyword table, shrunk global array, positional parameter dictionary, local variable dictionary, or sequence symbol dictionary -- overflows onto workfile 1. After the last entry is made, the dictionary block is written onto workfile 1 in order to free the dictionary area. Each index or offset searched for (see steps 8 and 9) is compared with the entry numbers in the dictionary blocks.

Program Organization

Purpose of the Section

The purpose of this section is to describe the structure of the ESERV program: how it is divided into phases, how the phases are loaded into main storage, and how control and data are passed within the program.

This section contains:

- Phase/control section/object module directory
- Summary of ESERV phases and functions
- ESERV control and data flow
- ESERV main storage allocation
- Main storage work area layouts
- ESERV common

Phase/Control Section/Object Module Directory

Phase	Control section	Object module	Description of the object module
ESERV	IPKVM000	IPKVM	ESERV identifier, text output with statement numbers
	IPKAD000 IPKAD100	IPKAD	SYSLB logic module (DTFSL)
	IPKVA003	IPKVA	Initializer
	IJJCPD1	IPKVA	SYSIPT/SYSPCH/SYSLST logic module (CPMOD)
	IPKVA000	IPKVA	Basic interface routines, common data area
	IPKVA002	IPKVA	Workfile logic module
ESERVD	IPKVD000	IPKVD	Control record and dictionary overflow
ESERVE	IPKVE000	IPKVE	Process keyword table and global array
ESERVF	IPKVF000	IPKVF	Convert edited text to source
ESERVG	IPKVG000	IPKVG	Process variable and sequence symbol dictionary
ESERVI	IPKVI000	IPKVI	Process prototype, symbolic variable references, and sequence symbol definitions
	IPKVK000	IPKVK	Update processing (if necessary)
ESERVB	IPKVB000	IPKVB	ABEND routine

Figure 1. Phase/Control Section/Object Module Directory. **This figure shows how the phases of the ESERV program are divided into control sections and object modules.**

Summary of ESERV Phases and Functions

The following figure describes the functions and subfunctions accomplished in each phase of the ESERV program. For information on the various control sections and object modules of the phases see **Figure 1.**

Phase	Function
ESERV	• Open workfiles, SYSIPT, and SYSLST • Compute buffer sizes and table addresses • Perform I/O • Check file assignments • Check edited deck for sequence error • Output record and insert statement number
ESERVD	• Read and process control records from SYSIPT • Read edited macro from source statement library • Branch to appropriate subroutines
ESERVE	• Read and save keyword table • Read global array and build global declarations • Build shrunk keyword table and global array
ESERVF	• Determine type of record • Determine field to be de-edited • Reconstruct original source order of operands and operators
ESERVG	• Build positional parameter and local variable dictionaries • Build local declarations • Build sequence symbol dictionary blocks
ESERVI	• Build prototype statement • Decompress and insert global and local declarations • Decompress and insert model statements • Regenerate conditional assembly and inner macro calls • Perform update operation (if necessary) • Output source record • Output update survey (if update performed)

Figure 2. Summary of ESERV Phases and Functions

ESERV Control and Data Flow

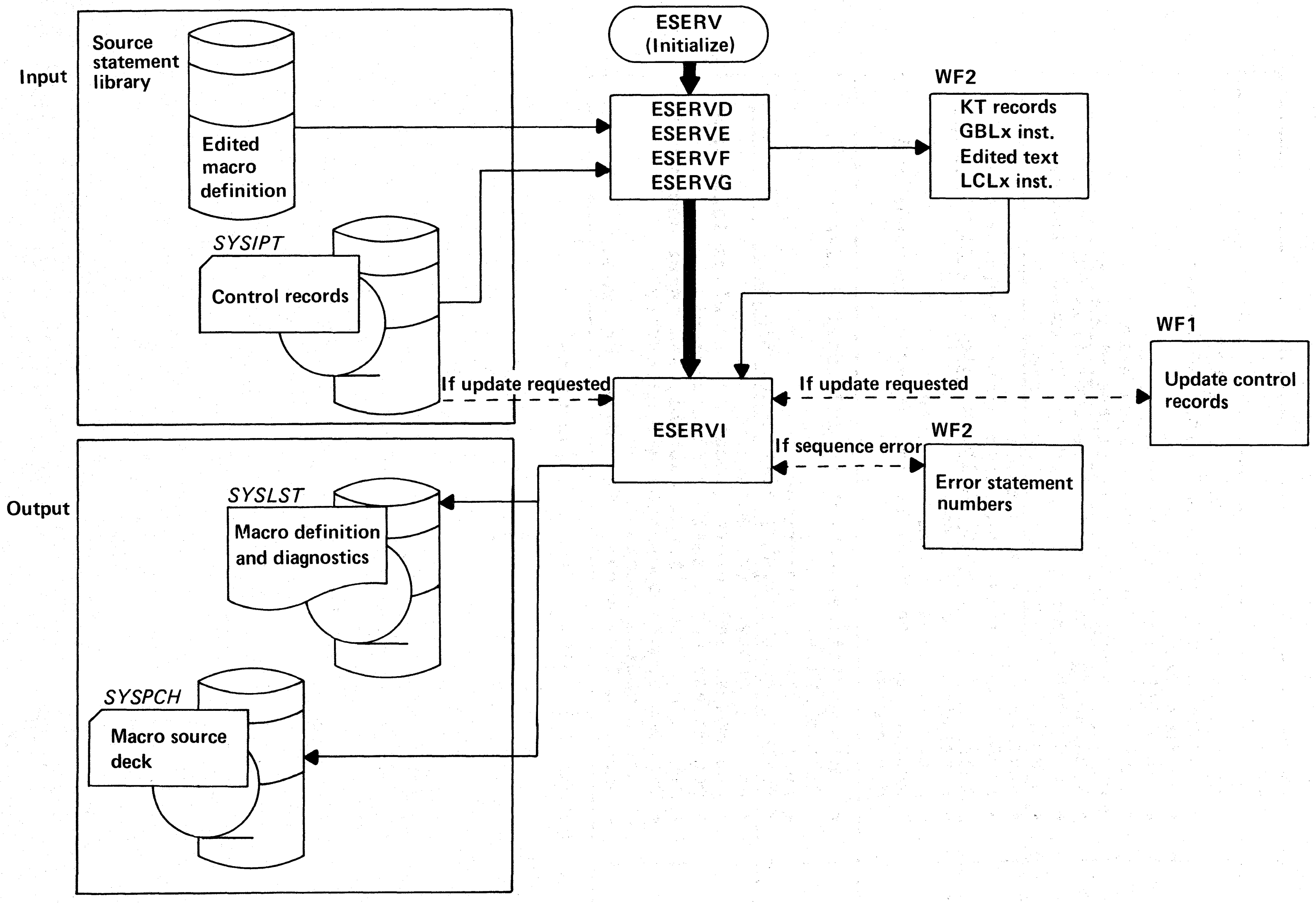


Figure 3. ESERV Control and Data Flow

ESERV Main Storage Allocation

The vertical axis of the diagram below represents the amount of main storage available to the partition. The horizontal axis represents time, the order in which the phases are loaded and executed in main storage. Most of the ESERV phase is in main storage throughout execution. ESERVD is in main storage during the execution of phases ESERVE, ESERVF, and ESERVG. The shaded portion of the diagram represents the work areas, buffers, dictionaries, tables, etc., of the phases. These work areas are illustrated in Figures 5 and 6.

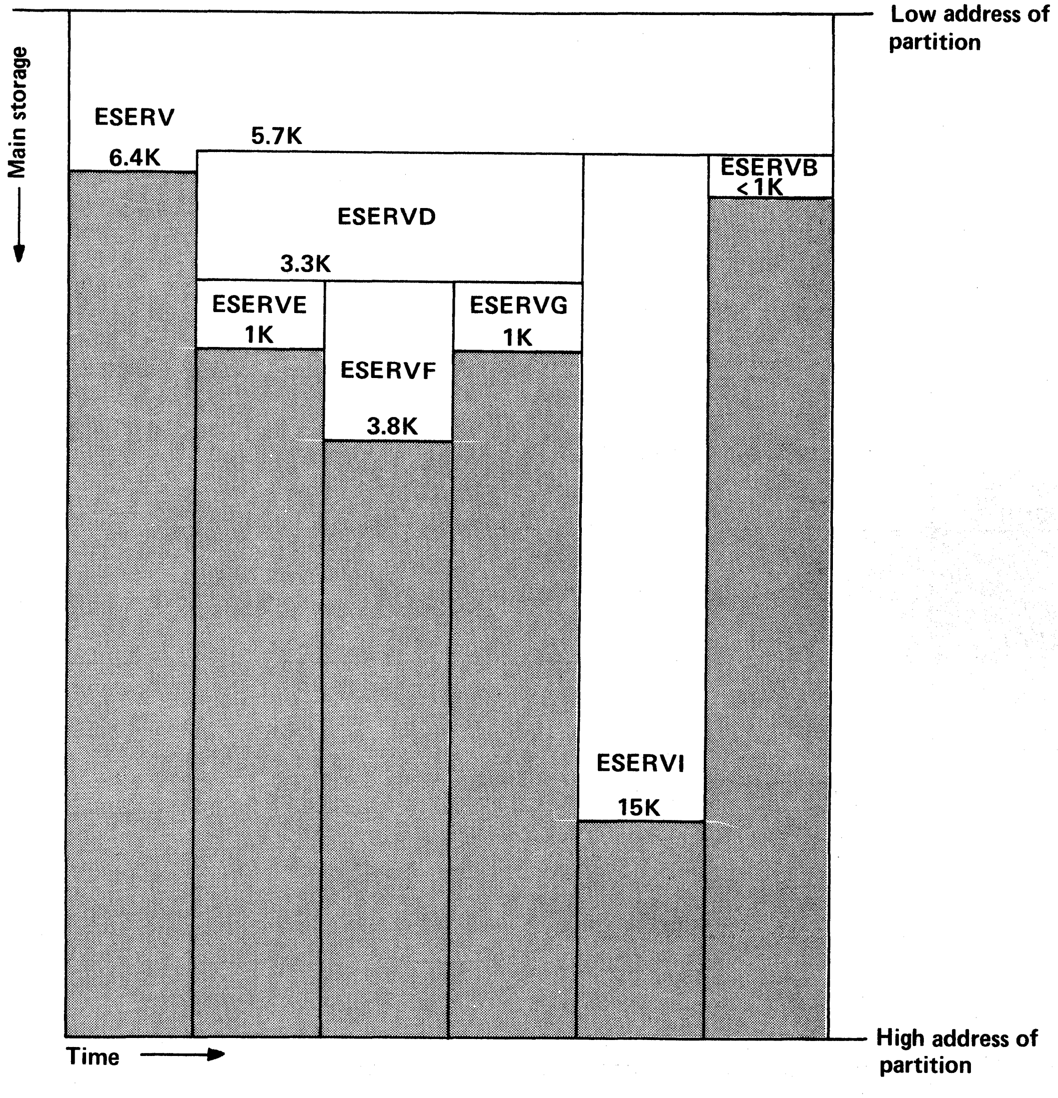


Figure 4. ESERV Main Storage Allocation

Main Storage Work Area Layouts

The following figures illustrate the work areas, buffers, dictionaries, etc., used by the phases of ESERV while they are in main storage. Figure 5 describes the contents of the work areas used jointly by phases ESERVE, ESERVF, and ESERVG. Work areas, etc., generally begin at the high storage address and work downwards using only as much of the available storage as they require. The information shown on Figures 5 and 6 corresponds with the shaded portion of Figure 4.

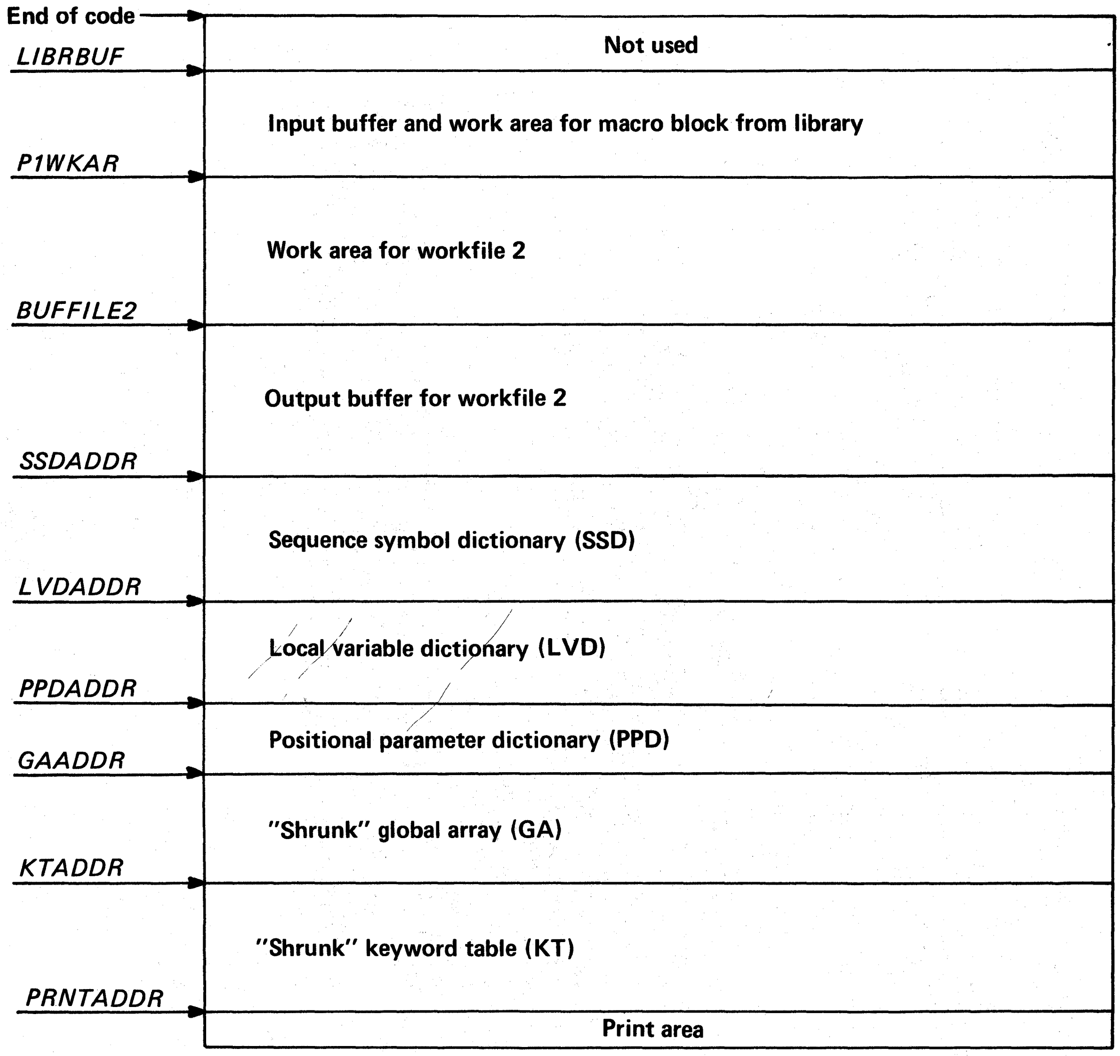


Figure 5. Work Area Layout for Phases ESERVE, ESERVF, and ESERVG

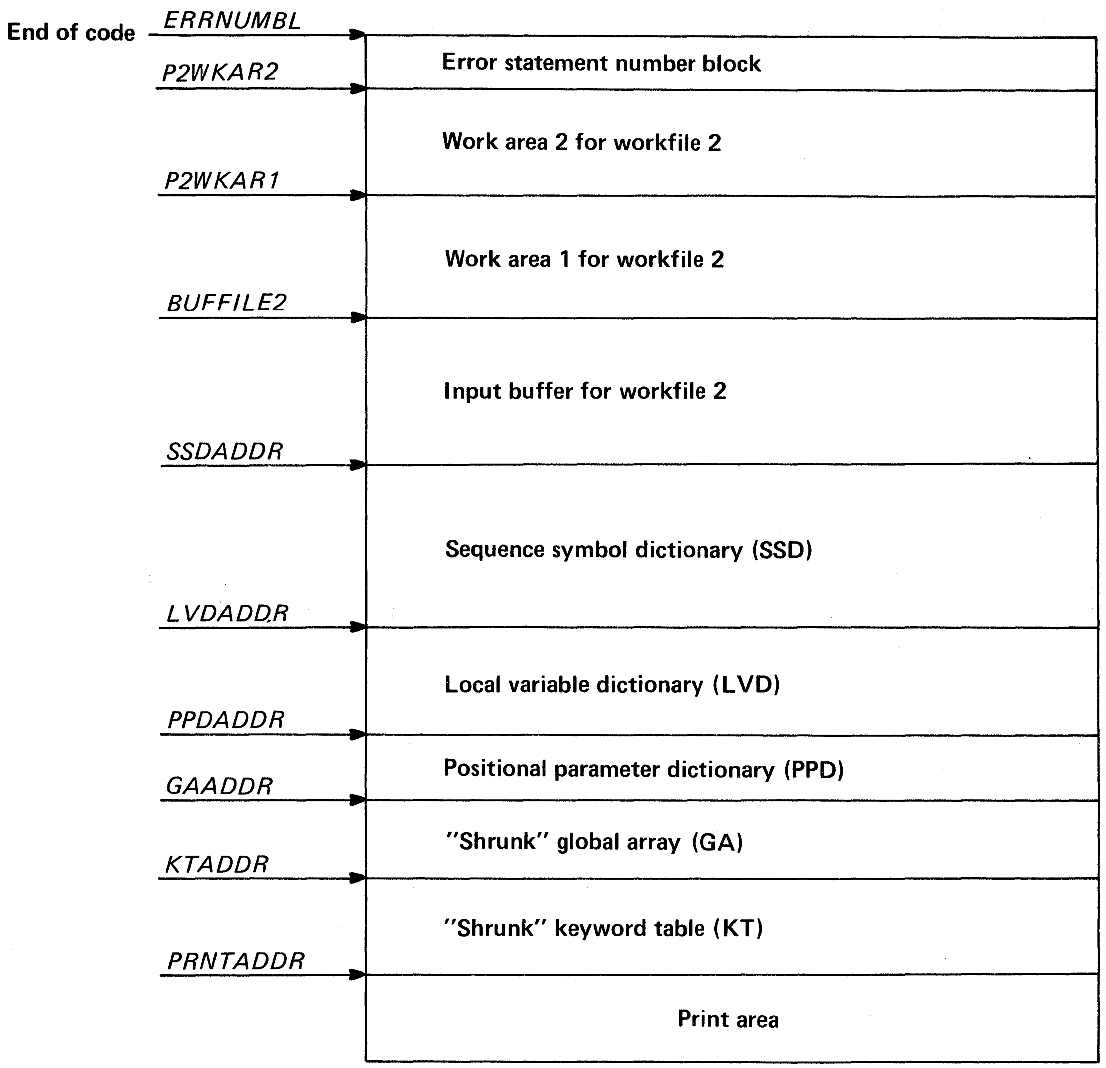


Figure 6. Work Area Layout for Phase ESERVI

ESERV Common

Information used in common by various phases of ESERV is divided up as follows:

- COMMONEQ
- COMNDATA
- COMSTRUC
- INTFBRTB

COMMONEQ

All information in COMMONEQ (basically the same as the assembler EQU function) is in the form of CONSTANT declarations in PL/S II code. COMMONEQ is compiled as INCLUDE code in each module of the ESERV program. No storage space is allocated for COMMONEQ. The following information is contained in COMMONEQ:

- Internal code equates
- Pseudo (Internal) operation code equates
- Operand and operator equates
- Register equates
- Miscellaneous equates (print control character,
  END flag for dictionaries and continuations cards etc.)

COMNDATA

All the information in COMNDATA is coded as one based structure. This is done in order to achieve the same effect as a "DSECT-version" of a common data area written in assembler language. COMNDATA is complied as INCLUDE code in each module of the ESERV program. The "CSECT-version" of COMNDATA is contained in the interface module IPKVA and is written in assembler language.

The following information is contained in COMNDATA:

- Option switches
- Buffer sizes
- NOTE/POINT values
- Parameter tables
- Miscellaneous pointers (addresses of various buffers and workareas, some special pointers to certain based structures, etc.)

COMSTRUC

All the information in COMSTRUC is in the form of based structures. COMSTRUC is compiled as INCLUDE code in each module of the ESERV program.

The following information is contained in COMSTRUC:

- Declarations of various structures used by more than one phase of the ESERV program (COMHEAD, KTITEM, STRINGS, etc.)

INTFBRTB

The INTFBRTB data area is the same as the DOS/VS Assembler branch table in COMMON except that some symbols have been added (FINDBOK and GETREC) and some have been deleted (PFETCH, PRETURN, PSAVE, and PPOINTGN). The DSECT PFCB is also included in the INTFBRTB data area. INTFBRTB is included in each module as COPY code during assembly time.

Data Areas

Purpose of the Section

This section shows the contents of the common data area located in the beginning of the interface module IPKVA. It contains information such as parameter tables and NOTE/POINT value tables for dictionaries, addresses of workareas, end-of-file addresses, switches, etc., used by various modules of ESERV. It is in storage throughout the execution of ESERV. Workarea addresses referred to in the method-of-operation diagram and the workarea layout diagrams, are defined in this data area. This data area is accessable through the structure COMNDATA in all modules except IPKVB.

DATA AREA: **IPKVA**

```
DISPLMNT                       FIELD        DESCRIPTION:  CONTENTS
DEC    (HEX)      SIZE         NAME          MEANING/USE

0       (0)        8                         EMBEDDED IDENTIFIER

                       **************************************************************
                       *                                                            *
                       *        BRANCH TABLE FOR THE INTERFACE MACROS               *
                       *                                                            *
                       **************************************************************


                       **************************************************************
                       *                                                            *
                       *        REGISTER EQUATES                                    *
                       *                                                            *
                       **************************************************************

                .... ....      R0
                .... ...1      R1
                .... ..1.      R2
                .... ..11      R3
                .... .1..      R4
                .... .1.1      R5
                .... .11.      R6
                .... .111      R7
                .... 1...      R8
                .... 1..1      R9
                .... 1.1.      R10
                .... 1.11      R11
                .... 11..      R12
                .... 11.1      R13
                .... 111.      R14
                .... 1111      R15
                .... .11.      ROFFS          PARAMETER REGS
                .... .111      RPARM          *FOR
                .... 1...      RFILE          *INTERFACE MACROS
                .... 1..1      RINPT          PGETL RECORD POINTER
                .... 1.1.      ROUTPT         PPUTL RECORD POINTER
                .... 1.11      RBIF           INTERFACE BASE REGISTER
                .... 11..      RBB            *
                .... 11.1      RBA            *
                .... 111.      RBR            STANDARD BRANCH REGISTER
                .... ..1.      RBRSAVE        BRANCH REGISTER FOR PSAVE


                       **************************************************************
                       *                                                            *
                       *        BIT EQUATES FOR BIT HANDLING MACROS                 *
                       *                                                            *
                       **************************************************************

                1... ....      BIT0
                .1.. ....      BIT1
                ..1. ....      BIT2
                ...1 ....      BIT3
                .... 1...      BIT4
                .... .1..      BIT5
                .... ..1.      BIT6
```

DISPLNMT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS MEANING/USE
		1	BIT7	
		1111 1111	BITFF	

```
*******************************************************
*                                                     *
*        MASK EQUATES                                 *
*                                                     *
*******************************************************
```

DISPLNMT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS MEANING/USE
		.... .111	ADDR	'ICM MASK' FOR ADDRESS
		.... 1...	HIGHBYTE	
		1	LOWBYTE	
86	(56)	1	JSW0005	PROGRAM SWITCH
		1...	POPSW	TELLS PRETURN TO POP STACK
		.1..	PLIST	OPTION LIST
		..1.	PINEOFSW	END OF FILE ON SYSIPT
87	(57)	6	NPTEMP	TEMPORARY STORAGE FOR READNEXT NOTE VALUE
87	(57)	2	NPTEMPCC	CYLINDER
89	(59)	1	NPTEMPH	HEAD
90	(5A)	1	NPTEMPR	RECORD
91	(5B)	2	NPTEMPTB	REMAINING TRACK CAPACITY
		.1.1 .111	PABENDC	CODE FOR ABEND REASON

```
*******************************************************
*                                                     *
*        FILE CONTROL BLOCKS                          *
*                                                     *
*******************************************************
```

DISPLNMT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS MEANING/USE
93	(5D)	35	PFILE2	FILE CONTROL BLOCK FOR FILE 2
93	(5D)	3	PDTFADR2	ADDRESS OF DTFSD
96	(60)	3	BUFPT2	POINTER TO NEXT RCD IN BUFFER
99	(63)	3	BUFADDR2	ADDRESS OF BUFFER
102	(66)	2	PBUFLEN2	BUFFER LENGTH
104	(68)	3	PENDBUF2	ADDRESS OF LAST BYTE OF BUFFER
107	(6B)	3	PWAADDR2	ADDRESS OF WORKAREA
110	(6E)	1		SWITCHES (SEE DSECT PFCB)
111	(6F)	3	PEOFADR2	ADDRESS OF END-OF-FILE ROUTINE
114	(72)	8	PNPOINT2	NOTE/POINT VALUE
122	(7A)	6	PNEXTNP2	N/P VALUE FOR NEXT BLOCK
128	(80)	3	PFILE1	FILE CONTROL BLOCK FOR FILE 1
128	(80)	3	PDTFADR1	ADDRESS OF DTFSD
131	(83)	14		NOT USED FOR FILE1
145	(91)	1		SWITCHES

```
*******************************************************
*                                                     *
* PFCB:     FILE CONTROL BLOCK DSECT                  *
*                                                     *
*******************************************************
```

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS MEANING/USE
0	(0)	3	PDTFADDR	ADDRESS OF DTF
3	(3)	3	BUFPT	POINTER TO RECORD IN BUFFER
6	(6)	3	BUFADDR	ADDRESS OF BUFFER
9	(9)	2	PRECLEN	MAX RECORD LENGTH
11	(B)	3	ENDBUF	ADDRESS OF LAST BYTE OF BUFFER
14	(E)	3	PWAADDR	ADDRESS OF WORKAREA
17	(11)	1	PFCBSW	PROGRAM SWITCH
		1...	OPENSW	IF 1, FILE OPEN
		.1..	READSW	IF 1, READ, IF 0, WRITE
		..1.	UPDSW	IF 1, WRITE UPDATE
		...1	BUF2SW	IF 1, TWO BUFFERS
		.... 1...	UPD2SW	IF 1, UPDATE MODE
		.... .1..	FIRSTSW	IF 1, FIRST I/O OPERATION
		.... ..1.	PFCBSW1	*
		1	PFCBSW2	*
18	(12)	3	PEOF	EOF ADDRESS
21	(15)	8	PNOTEPNT	NOTE POINT VALUES
21	(15)	6	PNPRW	
21	(15)	4	PCCHR	CYLINDER, HEAD, RECORD
25	(19)	2	PTRKBAL	TRACK BALANCE
27	(1B)	2	PNPOFFS	RECORD OFFSET FROM BUFFER START
29	(1D)	1	PNEXTNP	N/P VALUE FOR NEXT BLOCK (LNG=PNPRW)

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS MEANING/USE
146	(92)	1	SAVMHFLG	SAVEAREA FOR MAC HEAD FLAG
146	(93)	1	JSW0009	PROGRAM SWITCH
		..1.	GLBVARB	TWO UNUSED BITS
		.... ..1.	KWPARB	GLOBAL ARRAY PRESENT SW
147	(93)	2	OFSINCR	KEYWORD TABLE PRESENT
149	(95)	1		NOT USED YET
150	(96)	1	MACLGTH	LENGTH OF MACRO NAME
151	(97)	1	MACLIB	MACRO LIBRARY
152	(98)	8	MACNAME	NAME OF MACRO
160	(A0)	6	ERNPTWR	N/P VAL TO WRITE ERRNUM BLOCK
166	(A6)	6	ERNPTRD	N/P VAL TO READ ERRNUM BLOCKS
172	(AC)	8	TXTNPT	N/P VAL FOR EDITED TEXT
180	(B4)	8	LCLXNPT	N/P VAL FOR LCLX RECORDS
188	(BC)	6	UPDATNPT	N/P VAL FOR UPDATE CONTROL STATEMENTS
194	(C2)	0	NOTVALTB	TABLE OF N/P VALUES (LNG=6*5)
194	(C2)	6		N/P VAL FOR PPC
200	(C8)	6		N/P VAL FOR LVD
206	(CE)	6		N/P VAL FOR GA
212	(D4)	6		N/P VAL FOR SSD
218	(DA)	6		N/P VAL FOR KT
224	(E0)	0	PRMTABLS	TABLE OF PARAMETER TABLES (LNG=6*5) FOR PREAD/PWRITE
224	(E0)	5		PARAMETER TABLE FOR PPD
224	(E0)	3	PPDADDR	PPD BUFFER ADDRESS
227	(E3)	2	PPDL	PPD BUFFER LENGTH
229	(E5)	5		PARAMETER TABLE FOR LVD
229	(E5)	3	LVDADDR	LVD BUFFER ADDRESS
231	(E8)	2	LVDL	LVD BUFFER LENGTH

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
234	(EA)	5		PARAMETER TABLE FOR GA
234	(EA)	3	GAADDR	GA BUFFER ADDRESS
237	(ED)	2	GAL	GA BUFFER LENGTH
239	(EF)	5		PARAMETER TABLE FOR SSD
239	(EF)	3	SSDADDR	SSD BUFFER ADDRESS
242	(F2)	2	SSDL	SSD BUFFER LENGTH
244	(F4)	5		PARAMETER TABLE FOR KT
244	(F4)	3	KTADDR	KT BUFFER ADDRESS
247	(F7)	2	KTL	KT BUFFER LENGTH
249	(F9)	5		
249	(F9)	3	ERRNUMBL	ERRNUMBRS
252	(FC)	2	ERRL	ERRNUMBL BUFFER LENGTH
254	(FE)	1	JSW0010	PROGRAM SWITCH
		1...	SWCRDMIS	ON IF GETREC FOUND SEQ ERR
		.1..	SWREC	ON IF CONTROL RECORD IN PROCESS
		..1.	SWLIST	LIST OPTION REQUESTED
		...1	SWPUNCH	PUNCH OPTION REQUESTED
		.... 1...	SWUPD	ON IF UPDATE MODE
		.... .1..	SWCOL1	ON IF FIRST RCD
		.... ..1.	SWPCHOPN	ON IF SYSPUNCH OPENED
		1	SWPASGND	ON IF SYSPUNCH ASSIGNED
255	(FF)	1	JSW0011	PROGRAM SWITCH
		1...	SWSURVEY	ON IF UPDATE SURVEY
		.1..	SWGENEND	ON IF END TO BE GENERATED
		..1.	SWLCLX	ON IF LCLX PRESENT
		...1	SWEOMAC	ON WHEN MEND PROCESSED
		.... 1...	SWLSTNAM	ON IF LAST NAME MET
		.... .1..	SWMACHED	ON WHEN MAC HEAD RCD EXPECTED
		.... ..1.	SWCANCEL	ON IF JOB TO BE CANCELLED
		1	SWCOLERR	ON IF ERROR IN) COL CARD
256	(100)	1	JSW0012	PROGRAM SWITCH
		1...	SWCOLCRD	ON IF) COL CARD PRESENT
		.1..	PDF	ON IF SUBLIB OPTION
257	(101)	4	DAQOFSET	ACCUMULATED OFFSET VALUE
261	(105)	3	NPT	PTR TO N/P TABLE NOTVALTB
264	(108)	3	PRMTBPTR	PTR TO PARAMETER TABLE
267	(10B)	3	CTRLPTR	POINTS TO NEXT MAC NAME
270	(10E)	3	SEQPTR	PTS TO FIRST COLUMN USED BY *SEQ. NUMB. WITHIN RANGE 73-80
273	(111)	4	EOFADDR	ADDR FOR END OF BOOK
276	(114)	3	EOFIPT	ADDR FOR EOF ON SYSIPT
279	(117)	3	PRNTADDR	ADDR OF PRINT AREA
282	(11A)	3	P1WKAR	ADDR OF WK AR FOR FILE2 IN PS1
285	(11D)	3	P2WKAR2	ADDR OF WK AR 2 FOR FILE2 PS2
288	(120)	3	INPTR	PTS TO FIRST NON-CTRL BYTE IN CONTROL RECORD BUFFER
		1111 .1..	P2WKAR1L	LENGTH OF WORK AREA 1
		1111 .1..	P2WKAR2L	LENGTH OF WORK AREA 2
		1111 .1..	BUF2LEN	LENGTH OF BUFFER 2
		...1 .1.1	P2WKAR1	
		...1 .11.	LIBRBUF	ADDR OF BUFFER FOR MACRO BLOCK

DISPLMNT DEC	(HEX)	SIZE	FIELD NAME	DESCRIPTION: CONTENTS, MEANING/USE
291	(123)	2	LINCOUNT	NO. OF LINES PER PAGE
293	(125)		DATEFLD	DATEFIELD (YY/MM/DD)
293	(125)		SHEADYR	YEAR
295	(127)			/
296	(128)		SHEADMON	MONTH
298	(12A)			/
299	(12B)		SHEDDAY	DAY
301	(12D)	2	SEQLEN	LENGTH OF SEQUENCE NUMBER FLD
303	(12F)	8	PROTMCNM	NAME FOR PROTOTYPE STMNT
321	(137)	81	PINPBUF1	SYSIPT BUFFER
392	(188)	3	PHICORE	ADDRESS OF HIGHEST BYTE, THAT
395	(18B)	2	PLRECLN	LENGTH OF RECORD LENGTH
397	(18D)	16	IFSAVE	INTERFACE ROUTINE SAVE AREA
			PUSH-DOWN SAVE-AREA DEFINITION	
		.... 1.1.	PSAVELVL	MAXIMUM NUMBER OF LEVELS
		.... .1..	PSAVESZ	SIZE OF EACH LEVEL
413	(19D)	0	PSAVETBL	SAVE AREA
	(1C5)	1.11 .11.	PSAVEND	END OF SAVE AREA
453	(1C5)	2	PSAVPT	CURRENT SAVE AREA INDEX
455	(1C7)	2	PRESCNT	SYSIPT RESIDUAL COUNT
457	(1C9)	4	PSAVTEMP	SAVES RWAA FOR PSAVE & PRETURN
	(18B)	.... 1...	LPHNAME	LENGTH OF PHASE NAME
		11	HW	'ICM MASK' FOR HALFWORDS
		....	REVEN1	EVEN/ODD REGISTER
		1	RODD1	*PAIR FOR MVCL
		.... 111.	RBUFPT	POINTS AT RECORD *EVEN/
		 1111	RRECLEN	CONTAINS RECORD LENGTH *ODD
		 1111	RWAA	INTERFACE WORK REGISTER
		....	RWAB	INTERFACE WORK REGISTER
		1	RWAC	INTERFACE WORK REGISTER
		.... ..1.	RWAD	INTERFACE WORK REGISTER
		.... .11.	RWAE	INTERFACE WORK REGISTER
461	(1CD)	16	PHSAVE	PREAD/PWRITE/PCHECK SAVE AREA
477	(1DD)	1	JSW0013	PROGRAM SWITCH
		1...	PUTSW	1 MEANS PPUTL, 0 MEANS PGETL
478	(1DE)	4	SRPARM	RPARM SAVE AREA
	(1E2)	2	PPROGCHK	EMERGENCY EXIT

ESERV DATA AREA FIELD CROSS-REFERENCE

The following is a directory of field entries in the data areas illustrated in this section. The list includes the field name, DSECT/CSECT name, and the field displacement in decimal and hexadecimal.

FIELD	DSECT/ CSECT	DISPLACEMENT DECIMAL	(HEX)
ADDR	IPKVA000	84	(54)
ADDR	IPKVA000	84	(54)
BITFF	IPKVA000	84	(54)
BIT0	IPKVA000	84	(54)
BIT0	IPKVA000	84	(54)
BIT1	IPKVA000	84	(54)
BIT2	IPKVA000	84	(54)
BIT3	IPKVA000	84	(54)
BIT4	IPKVA000	84	(54)
BIT5	IPKVA000	84	(54)
BIT6	IPKVA000	84	(54)
BIT7	IPKVA000	84	(54)
*BUFADDR	PFCB	6	(6)
*BUFADDR2	IPKVA000	99	(63)
*BUFPT	PFCB	3	(3)
*BUFPT2	IPKVA000	96	(60)
BUF2SW	PFCB	17	(11)
*CTRLPTR	IPKVA000	267	(10B)
DAQOFSET	IPKVA000	257	(101)
DATEFLD	IPKVA000	293	(125)
*ENDBUF	PFCB	11	(B)
*EOFADDR	IPKVA000	273	(111)
*EOFIPT	IPKVA000	276	(114)
ERNPTRD	IPKVA000	166	(A6)
ERNPTWR	IPKVA000	160	(A0)
*ERRL	IPKVA000	252	(FC)
ERRNUMBL	IPKVA000	249	(F9)
FIRSTSW	PFCB	17	(11)
GAADDR	IPKVA000	234	(EA)
*GAL	IPKVA000	237	(ED)
GLBVARB	IPKVA000	146	(92)
HIGHBYTE	IPKVA000	84	(54)
HW	IPKVA000	457	(1C9)
IFSAVE	IPKVA000	397	(78D)
*INPTR	IPKVA000	288	(120)
KTADDR	IPKVA000	244	(F4)
*KTL	IPKVA000	247	(F7)
KWPARB	IPKVA000	146	(92)
KWPARB	IPKVA000	146	(92)
LCLXNPT	IPKVA000	180	(B4)
LINCOUNT	IPKVA000	291	(123)
LOWBYTE	IPKVA000	84	(54)
LVDADDR	IPKVA000	229	(E5)
*LVDL	IPKVA000	232	(E8)
MACLGTH	IPKVA000	150	(96)
MACLIB	IPKVA000	151	(97)
MACNAME	IPKVA000	152	(98)
NOTVALTB	IPKVA000	194	(C2)
*NPT	IPKVA000	261	(105)
NPTEMP	IPKVA000	87	(57)
NPTEMPCC	IPKVA000	87	(57)
NPTEMPH	IPKVA000	89	(59)
NPTEMPR	IPKVA000	90	(5A)
NPTEMPTB	IPKVA000	91	(5B)
OPENSW	PFCB	17	(11)
*PBUFLEN2	IPKVA000	102	(66)
PCCHR	PFCB	21	(15)
*PDTFADDR	PFCB	0	(0)
*PDTFADR1	IPKVA000	128	(80)
*POINTER.			

FIELD	DSECT/ CSECT	DISPLACEMENT DECIMAL	(HEX)
PDF	IPKVA000	256	(100)
*PDTFADR2	IPKVA000	93	(5D)
*PENDBUF2	IPKVA000	104	(68)
*PEOF	PFCB	18	(12)
*PEOFADR2	IPKVA000	111	(6F)
PFCBSW	PFCB	17	(11)
PFCBSW1	PFCB	17	(11)
PFCBSW2	PFCB	17	(11)
PFILE1	IPKVA000	128	(80)
*PHICORE	IPKVA000	392	(188)
PHSAVE	IPKVA000	461	(1CD)
PINEOFSW	IPKVA000	86	(56)
PINPBUF1	IPKVA000	303	(137)
PLIST	IPKVA000	86	(56)
PLRECLN	IPKVA000	395	(18B)
PNEXTNP	PFCB	29	(1D)
PNEXTNP2	IPKVA000	122	(7A)
PNOTEPNT	PFCB	21	(15)
PNPOFFS	PFCB	27	(1B)
PNPOINT2	IPKVA000	114	(72)
PNPRW	PFCB	21	(15)
POPSW	IPKVA000	86	(56)
PPDADDR	IPKVA000	224	(E0)
*PPDL	IPKVA000	227	(E3)
PRECLEN	PFCB	9	(9)
PRESCNT	IPKVA000	455	(1C7)
PRMTABLS	IPKVA000	224	(E0)
*PRMTBPTR	IPKVA000	264	(108)
*PRNTADDR	IPKVA000	279	(117)
PROTMCNM	IPKVA000	295	(12F)
PSAVETBL	IPKVA000	413	(19D)
PSAVPT	IPKVA000	453	(1C5)
PSAVTEMP	IPKVA000	457	(1C9)
PTRKBAL	PFCB	25	(19)
PUTSW	IPKVA000	477	(100)
*PWAADDR	PFCB	14	(E)
*PWAADDR2	IPKVA000	107	(6B)
*P1WKAR	IPKVA000	282	(11A)
*P2WKAR2	IPKVA000	285	(11D)
READSW	PFCB	17	(11)
SAVMHFLG	IPKVA000	146	(92)
SEQLEN	IPKVA000	293	(120)
*SEQPTR	IPKVA000	270	(10E)
SHEADDAY	IPKVA000	299	(12B)
SHEADMON	IPKVA000	296	(128)
SHEAD YR	IPKVA000	293	(125)
*SRPARM	IPKVA000	478	(1DE)
SSDADDR	IPKVA000	239	(EF)
*SSDL	IPKVA000	242	(F2)
SWCANCEL	IPKVA000	255	(FF)
SWCOLCRD	IPKVA000	256	(100)
SWCOLERR	IPKVA000	255	(FF)
SWCOL1	IPKVA000	254	(FE)
SWCRDMIS	IPKVA000	254	(FE)
SWEOMAC	IPKVA000	255	(FF)
SWGENEND	IPKVA000	255	(FF)
SWLCLX	IPKVA000	255	(FF)
SWLIST	IPKVA000	254	(FE)

FIELD	DSECT/ CSECT	DISPLACEMENT DECIMAL	(HEX)
SWLSTNAM	IPKVA000	255	(FF)
SWMACHED	IPKVA000	255	(FF)
SWPASGND	IPKVA000	254	(FE)
SWPCHOPN	IPKVA000	254	(FE)
SWPUNCH	IPKVA000	254	(FE)
SWREC	IPKVA000	254	(FE)
SWSURVEY	IPKVA000	255	(FF)
SWUPD	IPKVA000	254	(FE)
TXTNPT	IPKVA000	172	(AC)
UPDATNPT	IPKVA000	188	(BC)
UPDSW	PFCB	17	(11)
UPD2SW	PFCB	17	(11)

Diagnostic Aids

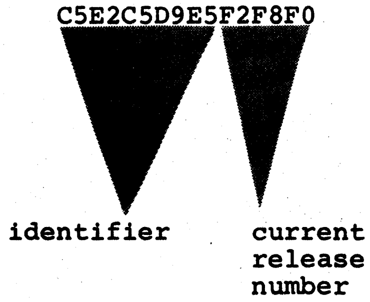
Purpose of the Section

This section contains information that may be useful in diagnosing problems within ESERV. The section is comprised of the following:

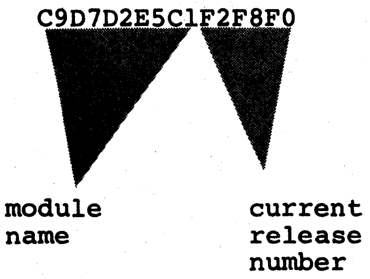
- Program identification
- I/O activity and workfile layouts
- ESERV register usage

Program Identification

The current release number of the ESERV program can be found starting at the first byte of the object code in a program dump. For example:



Every ESERV object module contains an identifier with the module name and current release number. The identifier is at the beginning of the module, starting at the first byte, and has the following format:



I/O Activity and Workfile Layouts

The following diagrams show the I/O activity for the phases of ESERV and the layouts of the workfiles during processing. The following symbols are used in the diagram:

↗	a NOTE/POINT address
⇓	an input operation
⇑	an output operation
⇕	both and input and output operation

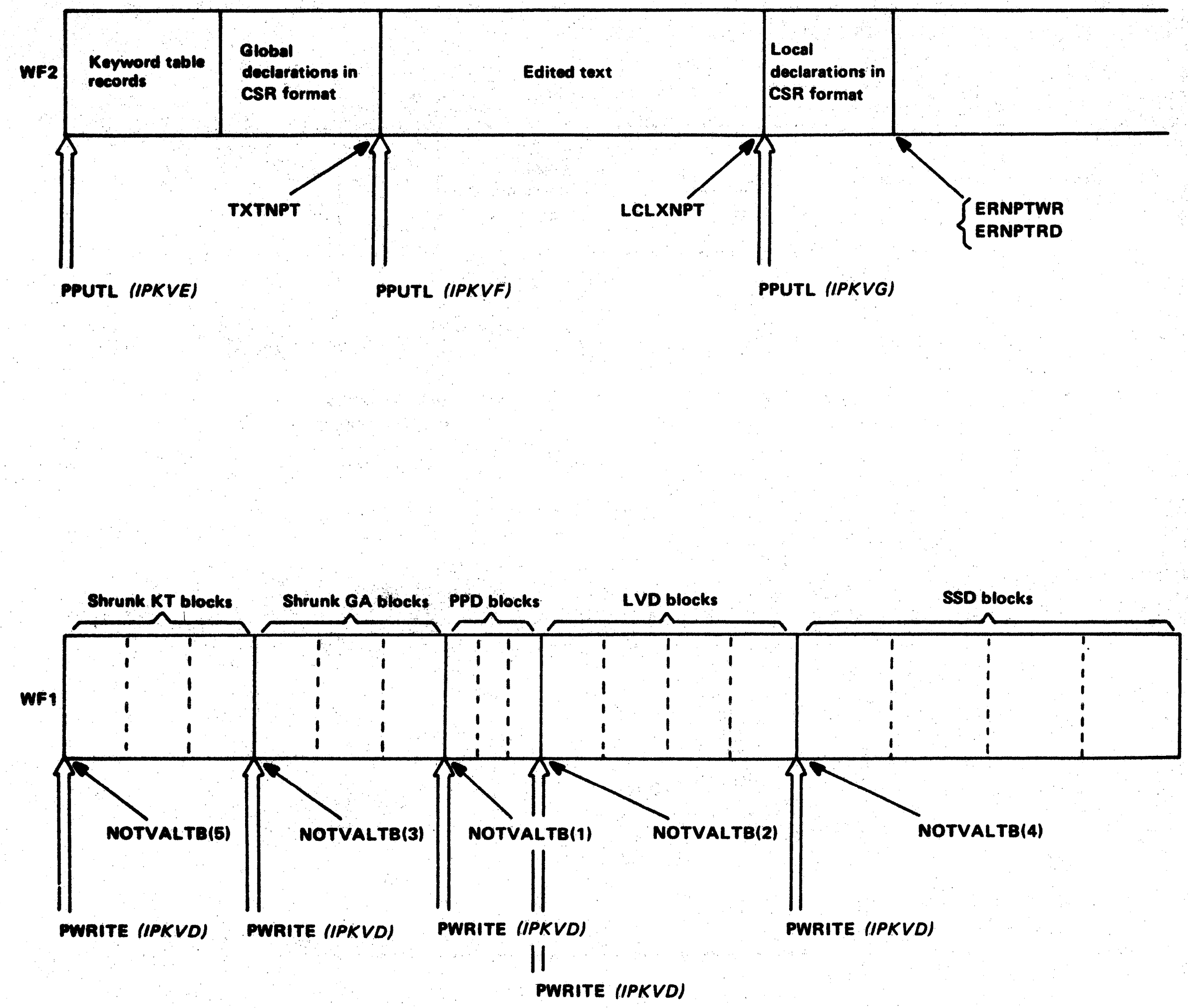


Figure 7. I/O Activity for ESERVD/ESERVE/ESERVF/ESERVG.

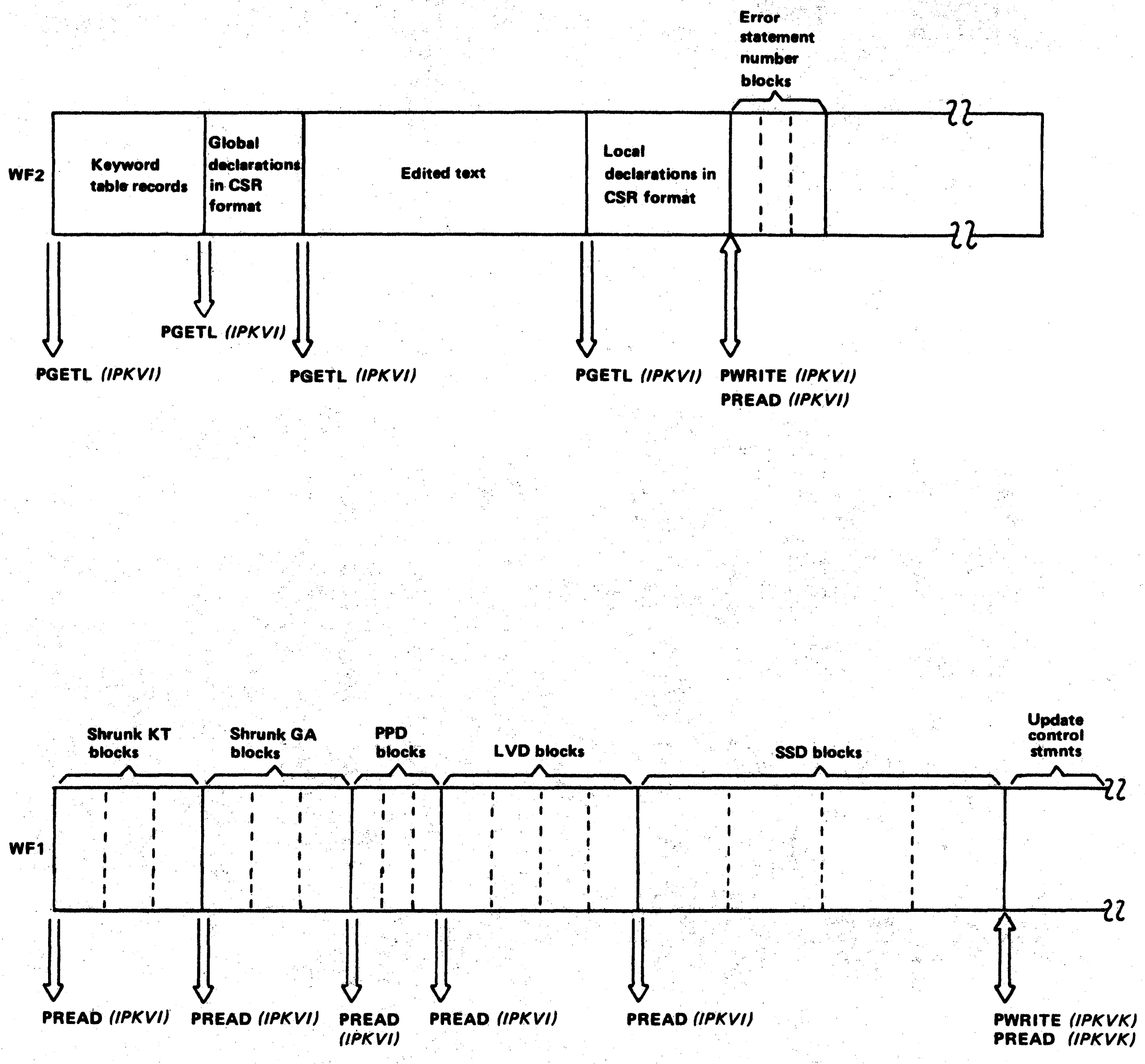


Figure 8. I/O Activity for ESERVI.

Register Usage for ESERV

The PL/S II compiler assigns work registers for the ESERV program as the need arises. For more information on PL/S II register assignment see Guide to PL/S-Generated Listings. The following registers have restricted usage for ESERV:

Register	Name	Usage
5	RECPTR	Pointer for VSDITEM; used by IPKVE, IPKVG, and IPKVI.
6	RGET	Input register for the macro block read by the GETREC routine.
7-10	Same as for the DOS/VS Assembler.	
11	RBIF	Interface base register.
14	Same as for the DOS/VS Assembler.	

Figure 9. ESERV Register Usage

Appendixes

This section contains the following information:

- Appendix A. ESERV Edited Statement Formats
- Appendix B. Pseudo (Internal) Operation Codes

Appendix A. Edited Statement Formats

This appendix shows statement formats at different stages in ESERV processing. Field lengths, in bytes, are shown under the field; "V" means a field of variable length.

Object Module IPKVA

PHYSICAL RECORDS IN AN EDITED MACRO BLOCK

CRDSTRUC

Column	Edited text	Blanks	Sequence number
1	2 - 70	71 - 76	77 - 80

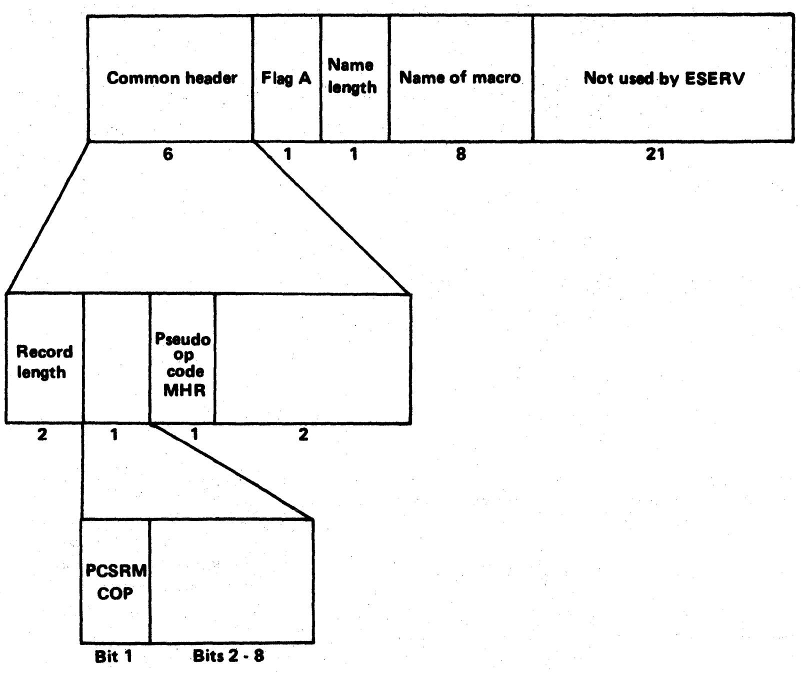
NOTES

"Column" points to the first column in the header of the first logical record in this physical record.

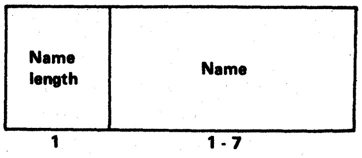
Object Module IPKVD

NOTES

MACRO BLOCK HEADER

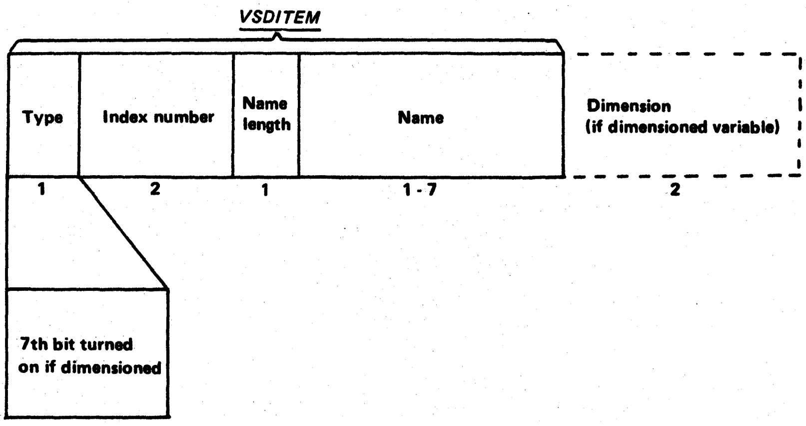


ITEM FORMAT OF SHRUNK KEYWORD TABLE



Taken from keyword table records in IPKVE

ITEM FORMAT OF SHRUNK GLOBAL ARRAY



Taken from global array records in IPKVE

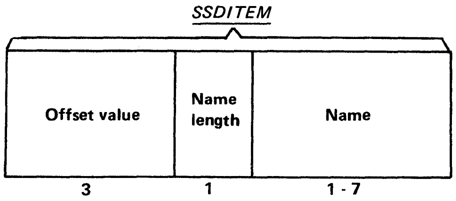
ITEM FORMAT OF POSITIONAL PARAMETER DICTIONARY

Same as shrunk global array; no dimension bytes allowed.

ITEM FORMAT OF LOCAL VARIABLE DICTIONARY

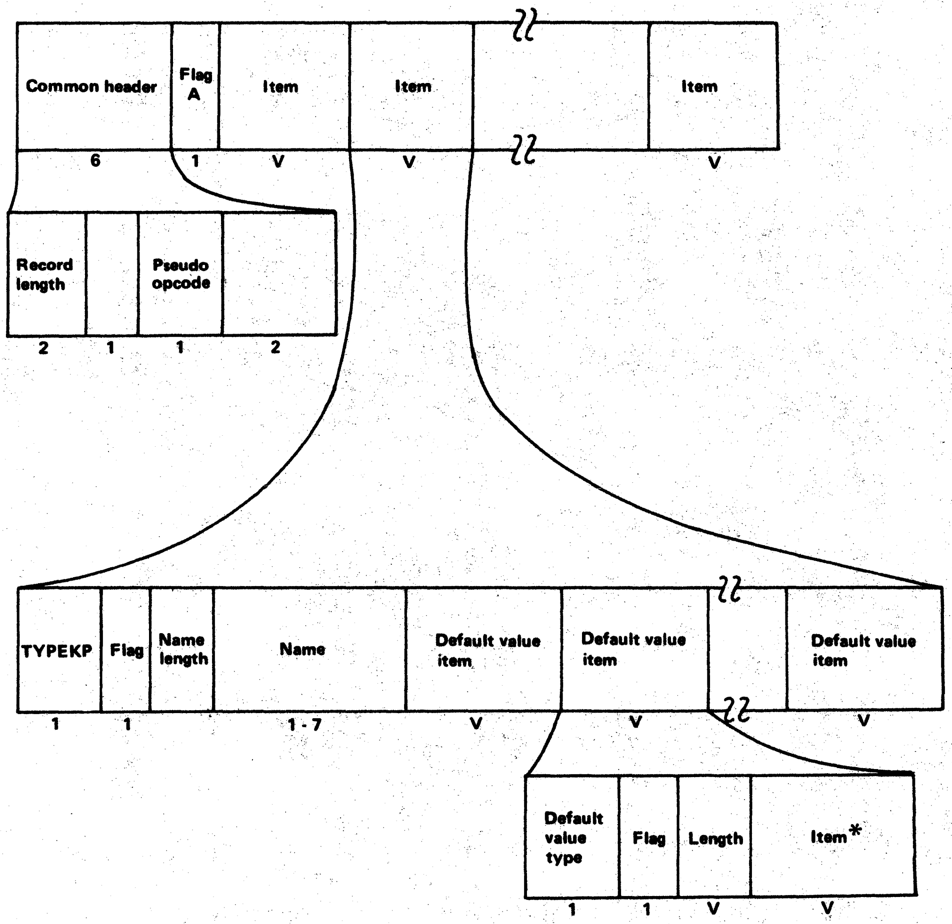
Same as shrunk global array.

ITEM FORMAT OF SEQUENCE SYMBOL DICTIONARY



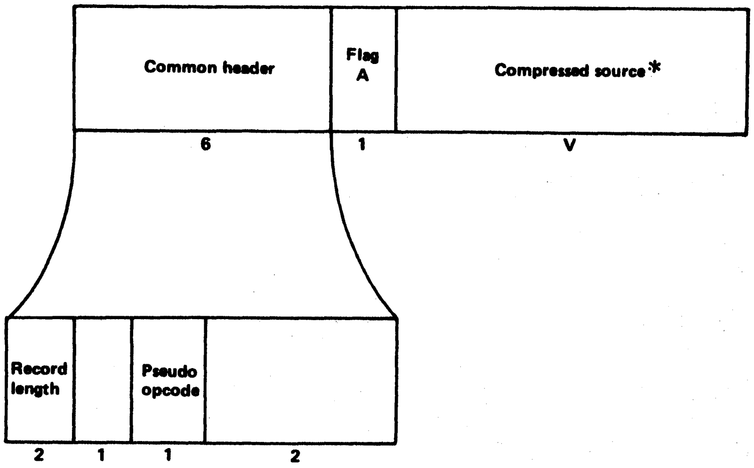
Offset value	Name length	Name
3	1	1 - 7

KEYWORD TABLE RECORDS



*These items are the same as those described in this section under IPKVI edited macro instructions item formats

COMPRESSED SOURCE RECORDS FOR GLOBAL DECLARATIONS



* As described in the Appendix H of DOS/VS Assembler Logic.

Object Module IPKVF

REVERSE POLISH NOTATION

The original order among operands and operators is retrieved but some operators require special treatment as follows:

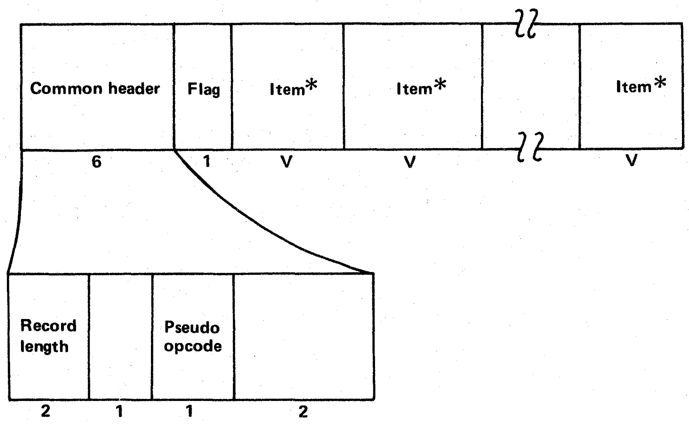
TCAC, TCBC	The operand is enclosed within apostrophes
TSST	A comma is inserted and the operand is enclosed within parentheses
TCAT	A concatenation character (period) is inserted where necessary
TCARPAR	The operand is enclosed within apostrophes
TIND	The subscript operand is enclosed within parentheses
TSUP	A comma is inserted and the operand is enclosed within parentheses
TAIF, TSETB	The operand is enclosed within parentheses

RECORD FORMATS

Record formats for IPKVF are the same as those described in this section under IPKVI.

Object Module IPKVG

INPUT: VARIABLE SYMBOL DICTIONARY RECORDS



* See items described earlier in this section under IPKVD item formats

SEQUENCE SYMBOL DICTIONARY RECORDS

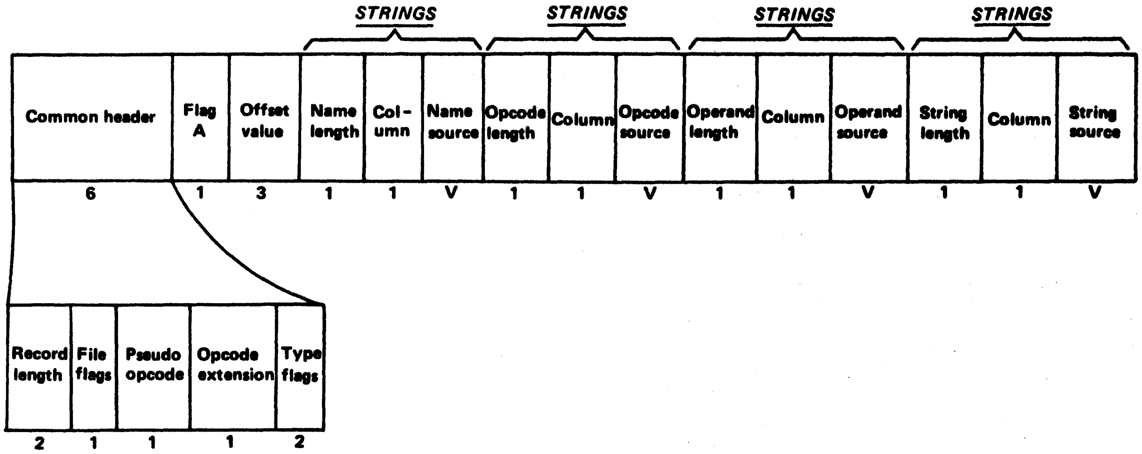
Same as variable symbol dictionary.

OUTPUT: COMPRESSED SOURCE RECORDS FOR LOCAL DECLARATIONS

Same as shrunk global array items described earlier in this section under IPKVD, Item formats.

Object Module IPKVI

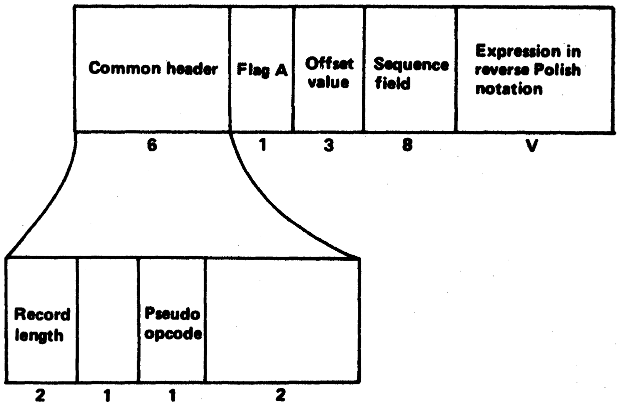
COMPRESSED SOURCE RECORDS



			STRINGS			*STRINGS*			*STRINGS*			*STRINGS*		
Common header	Flag A	Offset value	Name length	Col-umn	Name source	Opcode length	Column	Opcode source	Operand length	Column	Operand source	String length	Column	String source
6	1	3	1	1	V	1	1	V	1	1	V	1	1	V

Record length	File flags	Pseudo opcode	Opcode extension	Type flags
2	1	1	1	2

AGO/SETX/ACTR RECORDS

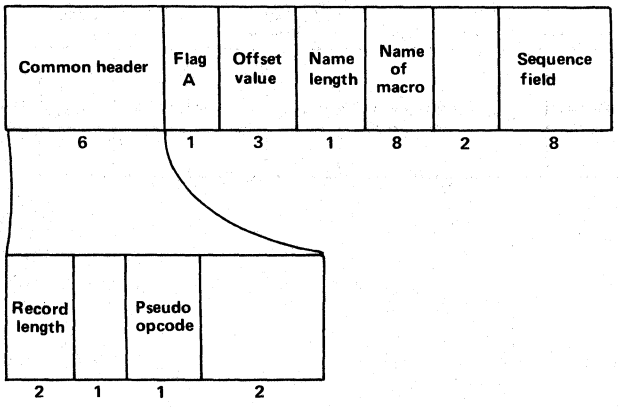


Common header	Flag A	Offset value	Sequence field	Expression in reverse Polish notation
6	1	3	8	V

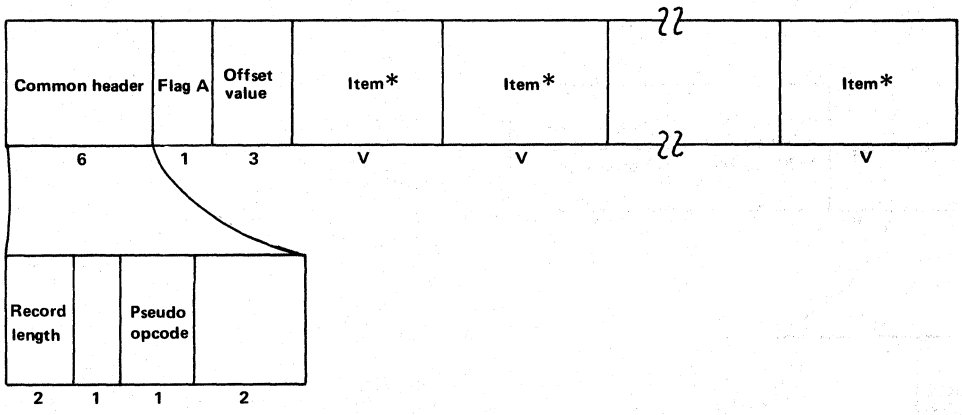
Record length		Pseudo opcode	
2	1	1	2

EDITED MACRO INSTRUCTION RECORDS

First record



Subsequent Records



*See the following item formats

ITEM FORMATS

Some of the following item formats are valid also for items in keyword table records (see IPKVE).

Item format of keyword prototype

X'A2'	Item* flag	Length	Name
1	1	1	1 - 7

NOTES

This item in the keyword table only.

Item format of character string

X'A5'	Item* flag	Length		K'	String	
1	1	1	1	1	0 - 255	6

This item in both the keyword table and macro instructions.

Item format of self-defining term

X'A6'	Item* flag	Length		K'	String
1	1	1	4	1	1 - 27

This item in both the keyword table and macro instructions.

Item format of sublist start

X'A7'	Item* flag	Length	
1	1	1	1

This item in both the keyword table and macro instruction.

* Eight-bit flag

NALTSRC	ITEM1ST	ITEMLISW		ITEMLONG			

Item format of sublist

X'A8'	Item* flag	Length	
1	1	1	1

Item format of basic character expression

X'A9'	Item* flag	Length	String in reverse Polish notation
1	1	1	1 - 255

Item format of omitted operand outside sublist

X'AA'	Item* flag	Length
1	1	1

Item format of keyword in macro instruction

X'AC'	Item* flag	Length	Name
1	1	1	1 - 7

Item format of end-of-operand

X'AD'	Item* flag	
1	1	2

* Eight-bit flag

NALTSRC	ITEM1ST	ITEMLISW		ITEMLONG			

NOTES

This item in both the keyword table and macro instructions.

This item in macro instructions only.

This item in both the keyword table and macro instructions.

This item in macro instructions only.

This item in both the keyword table and macro instructions.

Appendix B. Pseudo (Internal) Operation Codes

Hexadecimal	Mnemonic	Description
00	SUBST	Substituted opcode
01	REPROED	Reproed statement
1E	CMENT	Comment statement
24	MIED	Macro instruction edited record
27	MEND	Macro processing instruction
28	MEXIT	"
29	ANOP	Conditional assembly instruction
2A	SETA	"
2B	SETB	"
2C	SETC	"
2D	ACTR	"
2E	AIF	"
2E	AIFB	"
2F	AGO	"
2F	AGOB	"
30	GBLA	"
31	GBLB	"
32	GBLC	"
33	LCLA	"
34	LCLB	"
35	LCLC	"
37	PROTOED	Prototype edited record
38	MHR	Macro header record
39	KT	Keyword table record
3A	GAR	Global array record
3B	VSDR	Positional parameter dictionary or local variable dictionary record
3C	SSDR	Sequence symbol dictionary record
40	GX	GBLx compressed source record
41	LX	LCLx compressed source record

Note: The ESERV code is dependent on the organization of the opcodes. Any changes made to this organization may effect the program code.

Index

SY33-8567-1

DOS/VSE Assembler Logic (File No. S370-21 (DOS/VSE)) Printed in U.S.A. SY33-8567-1

IBM

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

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

SY33-8567-1

DOS/VSE Assembler Logic

READER'S COMMENT FORM

Your views about this publication may help improve its usefulness; this form will be sent to the author's department for appropriate action. Using this form to request system assistance or additional publications will delay response, however. *For more direct handling of such request, please contact your IBM representative or the IBM Branch Office serving your locality.*

Reply requested:
Yes ☐
No ☐

Name: ____________________
Job Title: ____________________
Address: ____________________
____________ Zip ____________

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

CUT ALONG DOTTED LINE

SY33-8567-1

DOS/VSE Assembler Logic

Your comments, please . . .

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. Your comments on the other side of this form will be carefully reviewed by the persons responsible for writing and publishing this material.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Fold Fold

First Class
Permit 40
Armonk
New York

Business Reply Mail

No postage stamp necessary if mailed in the U.S.A.

Postage will be paid by:

International Business Machines Corporation
Department 813 L
1133 Westchester Avenue
White Plains, New York 10604

Fold Fold

CUT OR FOLD ALONG LINE

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

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

DOS/VSE Assembler Logic (File No. S370-21 (DOS/VSE)) Printed in U.S.A. SY33-8567-1

SY33-8567-1

DOS/VSE Assembler Logic

READER'S COMMENT FORM

CUT ALONG DOTTED LINE

Your views about this publication may help improve its usefulness; this form will be sent to the author's department for appropriate action. Using this form to request system assistance or additional publications will delay response, however. *For more direct handling of such request, please contact your IBM representative or the IBM Branch Office serving your locality.*

Reply requested:
Yes ☐
No ☐

Name: ______________________________
Job Title: ______________________________
Address: ______________________________
______________ Zip ______________

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

SY33-8567-1

DOS/VSE Assembler Logic

Your comments, please . . .

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. Your comments on the other side of this form will be carefully reviewed by the persons responsible for writing and publishing this material.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Fold Fold

First Class
Permit 40
Armonk
New York

Business Reply Mail

No postage stamp necessary if mailed in the U.S.A.

Postage will be paid by:

International Business Machines Corporation
Department 813 L
1133 Westchester Avenue
White Plains, New York 10604

Fold Fold

IBM

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

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

CUT OR FOLD ALONG LINE

SY33-8567-1

DOS/VSE Assembler Logic

READER'S COMMENT FORM

Your views about this publication may help improve its usefulness; this form will be sent to the author's department for appropriate action. Using this form to request system assistance or additional publications will delay response, however. *For more direct handling of such request, please contact your IBM representative or the IBM Branch Office serving your locality.*

Reply requested:
Yes ☐
No ☐

Name: ______________________________
Job Title: ______________________________
Address: ______________________________
______________________________ Zip ______________

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

CUT ALONG DOTTED LINE

SY33-8567-1

DOS/VSE Assembler Logic

Your comments, please . . .

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. Your comments on the other side of this form will be carefully reviewed by the persons responsible for writing and publishing this material.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Fold Fold

First Class
Permit 40
Armonk
New York

Business Reply Mail

No postage stamp necessary if mailed in the U.S.A.

Postage will be paid by:

International Business Machines Corporation
Department 813 L
1133 Westchester Avenue
White Plains, New York 10604

Fold Fold

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

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

CUT OR FOLD ALONG LINE

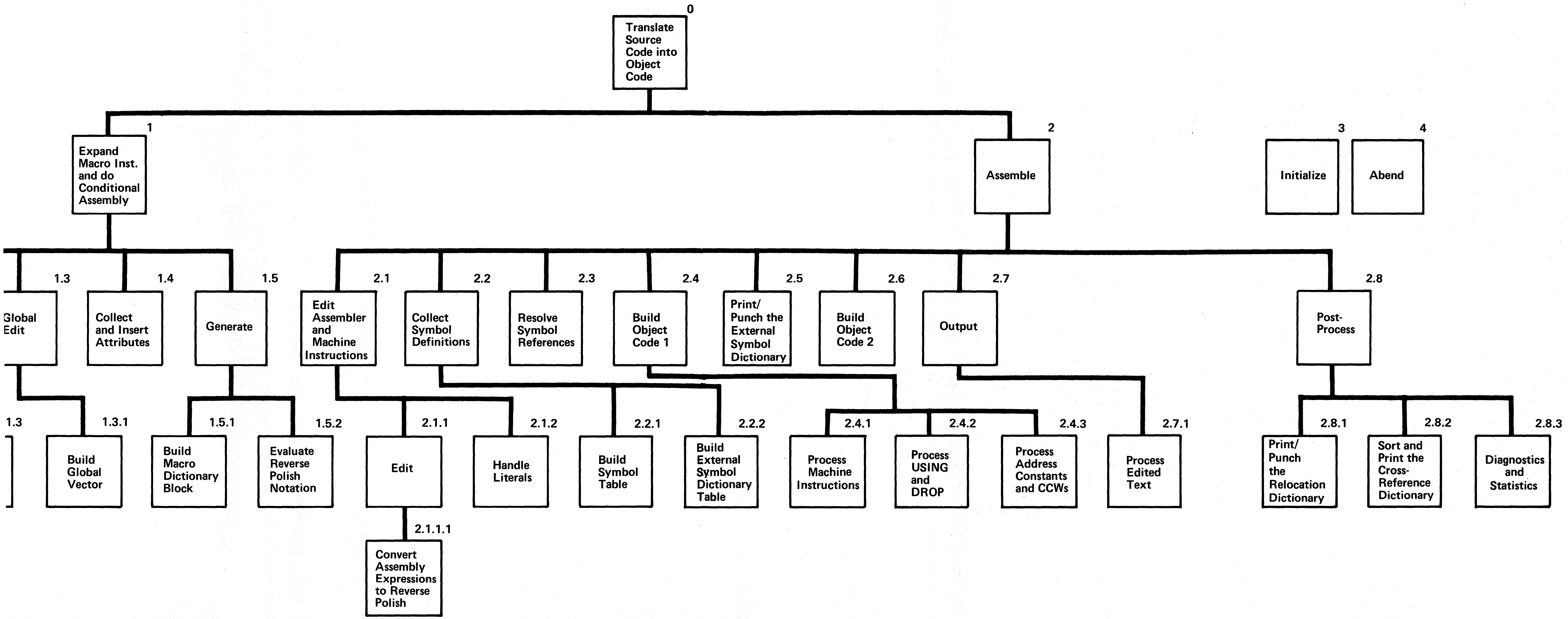


Figure 46. Table of Contents for Method of Operation Diagrams

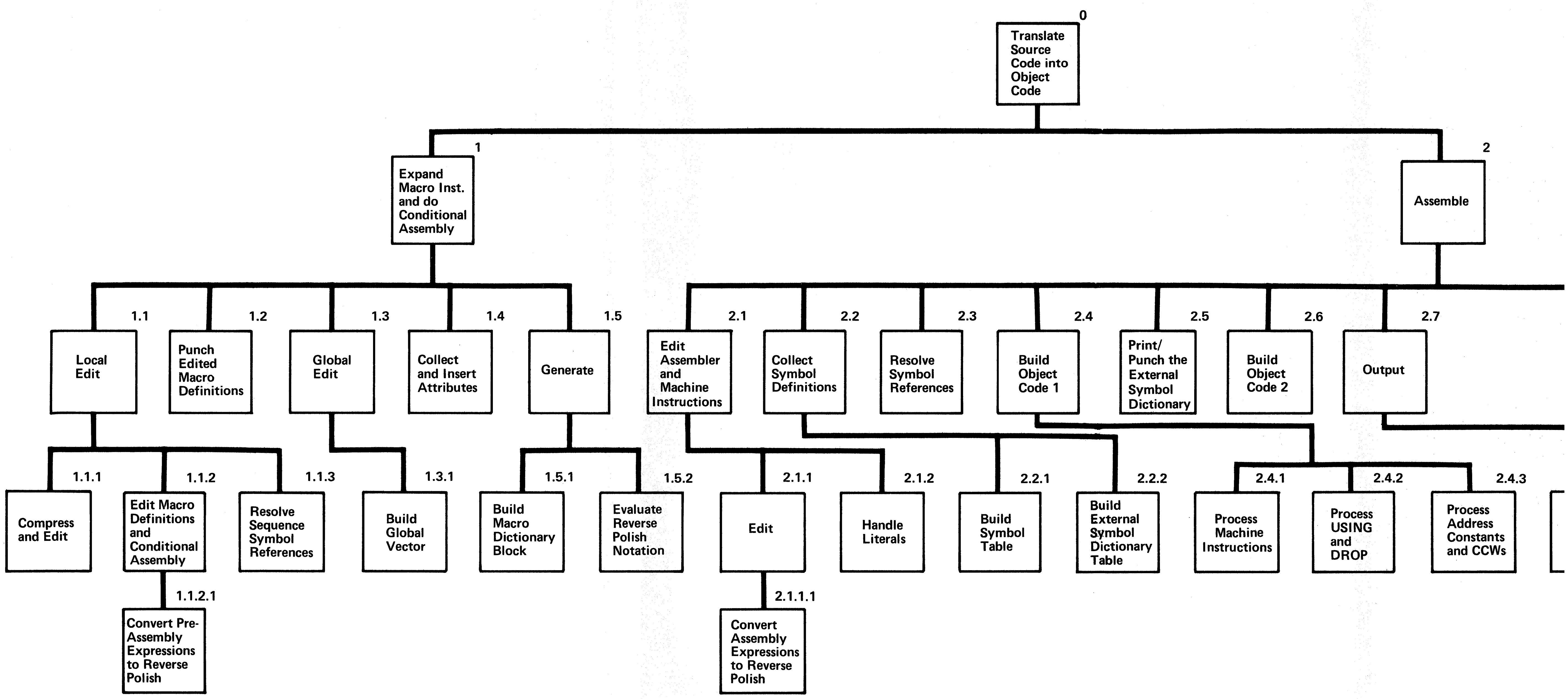

0
Translate Source Code into Object Code
1
Expand Macro Inst. and do Conditional Assembly
2
Assemble
1.1
Local Edit
1.2
Punch Edited Macro Definitions
1.3
Global Edit
1.4
Collect and Insert Attributes
1.5
Generate
2.1
Edit Assembler and Machine Instructions
2.2
Collect Symbol Definitions
2.3
Resolve Symbol References
2.4
Build Object Code 1
2.5
Print/ Punch the External Symbol Dictionary
2.6
Build Object Code 2
2.7
Output
1.1.1
Compress and Edit
1.1.2
Edit Macro Definitions and Conditional Assembly
1.1.3
Resolve Sequence Symbol References
1.3.1
Build Global Vector
1.5.1
Build Macro Dictionary Block
1.5.2
Evaluate Reverse Polish Notation
2.1.1
Edit
2.1.2
Handle Literals
2.2.1
Build Symbol Table
2.2.2
Build External Symbol Dictionary Table
2.4.1
Process Machine Instructions
2.4.2
Process USING and DROP
2.4.3
Process Address Constants and CCWs
1.1.2.1
Convert Pre-Assembly Expressions to Reverse Polish
2.1.1.1
Convert Assembly Expressions to Reverse Polish

Figure 46. Ta