LY24-5215-0
File No. S370/4300-50

Program Product

Data Language/I Disk Operating System/ Virtual Storage (DL/I DOS/VS) Logic Manual, Volume 2

Program Number 5746-XX1

IBM

Preface

This manual is to be used with the *Data Language/I Disk Operating System/Virtual Storage (DL/I DOS/VS) Logic Manual, Volume 1,* LY12-5016, and program listings for DL/I DOS/VS. It contains the HIPO diagrams that illustrate the program logic described in Volume 1. It is intended for use by persons involved in program maintenance and by system programmers who are altering the program design.

DL/I DOS/VS is a data management control system that assists the user in creating, accessing, and maintaining large common data bases. In conjunction with the Customer Information Control System (CICS/VS), DL/I DOS/VS can be used in an online teleprocessing environment.

Because DL/I DOS/VS is a functional subset of the IBM Information Management System/Virtual Storage (IMS/VS), some specific IMS or OS terms are used in this manual. These terms are used to allow easy reference to the documentation of the related systems.

This manual contains only "Section 2: Method of Operation" which consists of HIPO diagrams that describe the DL/I modules. The diagrams include cross reference to labels in the program listings.

Because Section 2 was formerly a part of Volume 1, considerable cross reference exists between other sections of Volume 1 and Section 2. The figure numbering system has been retained for Section 2 to ensure credibility of cross references found in Volume 1.

Note: *In this publication, the system and component name DOS/VS should be read as DOS/VSE unless the name explicitly refers to DOS/VS release 34 or an earlier DOS/VS release.*

Related Publications

DL/I DOS/VS General Information Manual, GH20-1246
DL/I DOS/VS Application Program Reference Manual, SH12-5411
DL/I DOS/VS Data Base Administration, SH24-5011
DL/I DOS/VS Resource Definition and Utilities, SH24-5021
DL/I DOS/VS Messages and Codes, SH12-5414
DL/I DOS/VS Guide for New Users, SH24-5001
DL/I DOS/VS Diagnostic Guide, SH24-5002
DL/I DOS/VS Logic Manual, Volume 1, LY12-5016.

For DOS/VS messages and return codes:

DOS/VSE Messages, GC33-5379
DOS/VSE Macro User's Guide, GC24-5139
DOS/VSE Macro Reference, GC24-5140
Using VSE/VSAM Commands and Macros, SC24-5144
VSE/VSAM Messages and Codes, SC24-5146.

Users employing DL/I DOS/VS in an online environment should have access to the following CICS/VS publications:

CICS/VS System Programmer's Reference Manual, SC33-0069
CICS/VS Application Programmer's Reference Manual, SC33-0079
CICS/VS System Application Design Guide, SC33-0068
CICS/VS System Programmer's Guide (DOS/VS), SC33-0070.

First Edition (June 1981)

This edition, LY24-5215-0, applies to Version 1, Release 6 (Version 1.6) of IBM System/370 Data Language/I Disk Operating System/Virtual Storage (DL/I DOS/VS), Program Number 5746-XX1. This manual was formerly Section 2 of *Data Language/I Disk Operating System/Virtual Storage Logic Manual,* LY12-5016. This edition applies to all subsequent releases and modifications unless otherwise indicated in new editions or Technical Newsletters.

Changes are made periodically to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest *IBM System/370 and 4300 Processors Bibliography,* GC20-0001.

It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming or services in your country.

Publications are not stocked at the address given below; requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for reader's comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Programming Publications, Dept. G60, P.O. Box 6, Endicott, NY, U.S.A. 13760. 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.

Contents

Figures

Section 2: Method of Operation

This section contains HIPO (Hierarchy, plus Input, Process, Output) diagrams.

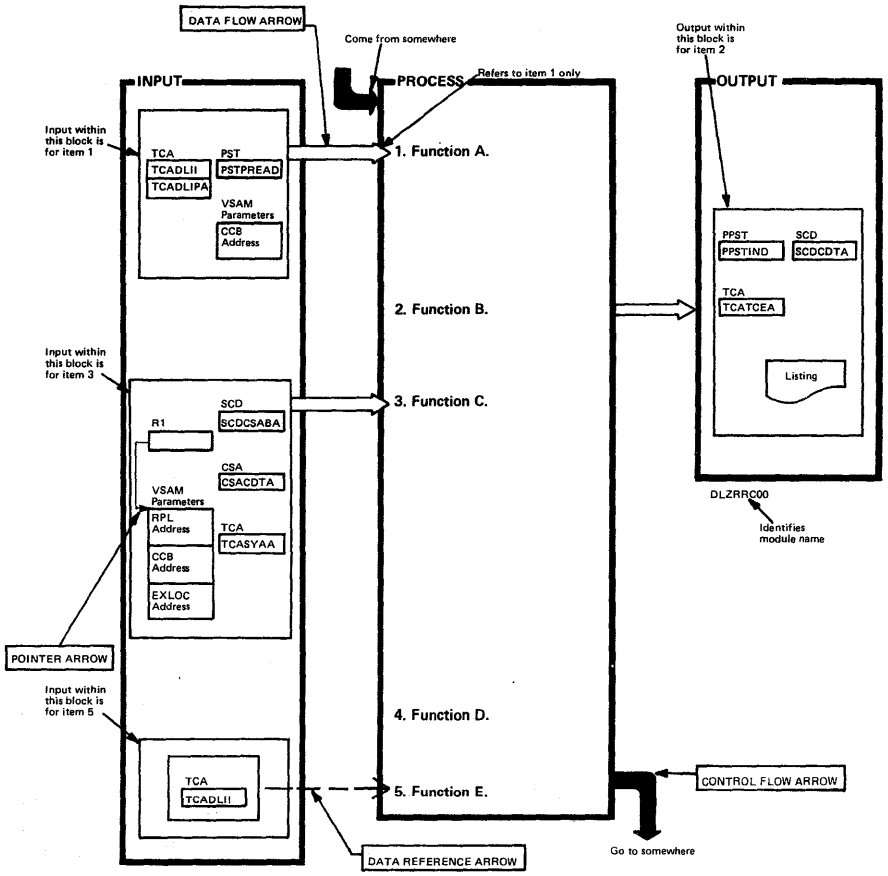
The three areas of each HIPO diagram are, from left to right, the input area, process area, and output area. Read the diagrams beginning with the process area. This describes a function that is performed. Arrows leading from the input area show what, if any, input is used to perform that function. Arrows leading to the output area show what output, if any, is produced.

At the bottom of each HIPO diagram is an area called "extended descriptions." This area contains comments not included in the process area of the diagram. For most items in the process area, extended description items with the same numbers give details that cannot be easily shown in diagram form or in the space allowed.

Various forms of arrows represent different usage conventions. Also, items are often boxed in to show that they are related to the same function. Figure 2-1 shows the conventions used in the HIPO diagrams.

Figure 2-2 is a visual table of contents with figure numbers. The figure numbers refer to the HIPO diagrams.

Figure 2-1. Guide to Reading Method of Operation Diagrams



Extended Description	Module	Label
1. More about function A. 2. More about function B.		

Extended Description	Module	Label

Figure 2-2. Visual Table of Contents for DL/I DOS/VS HIPO Charts

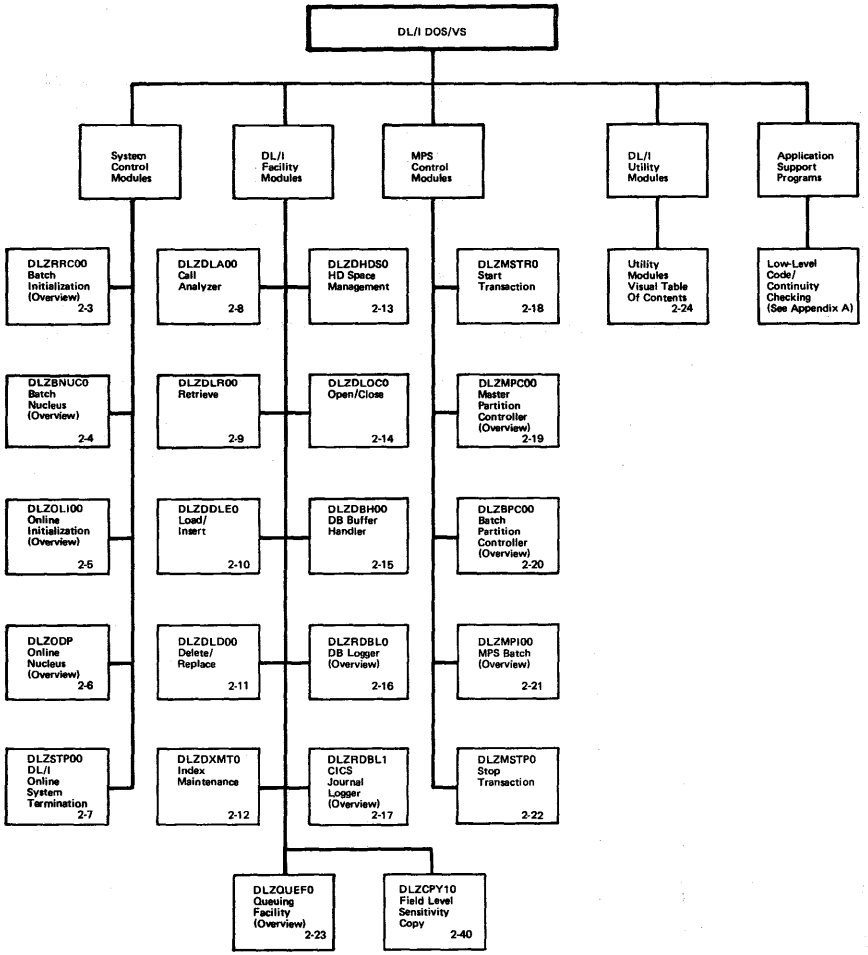
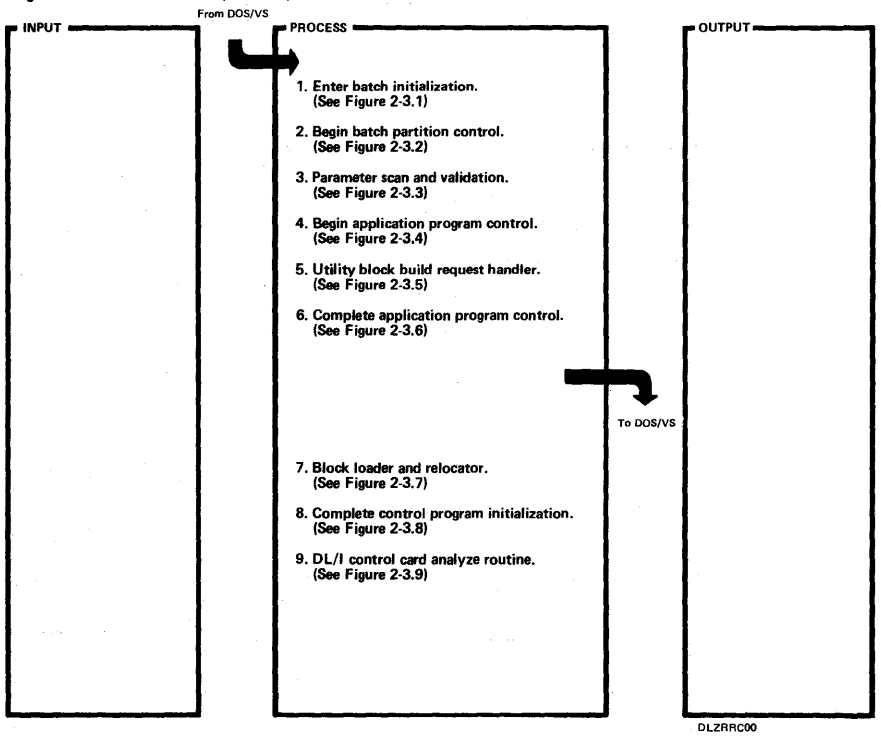


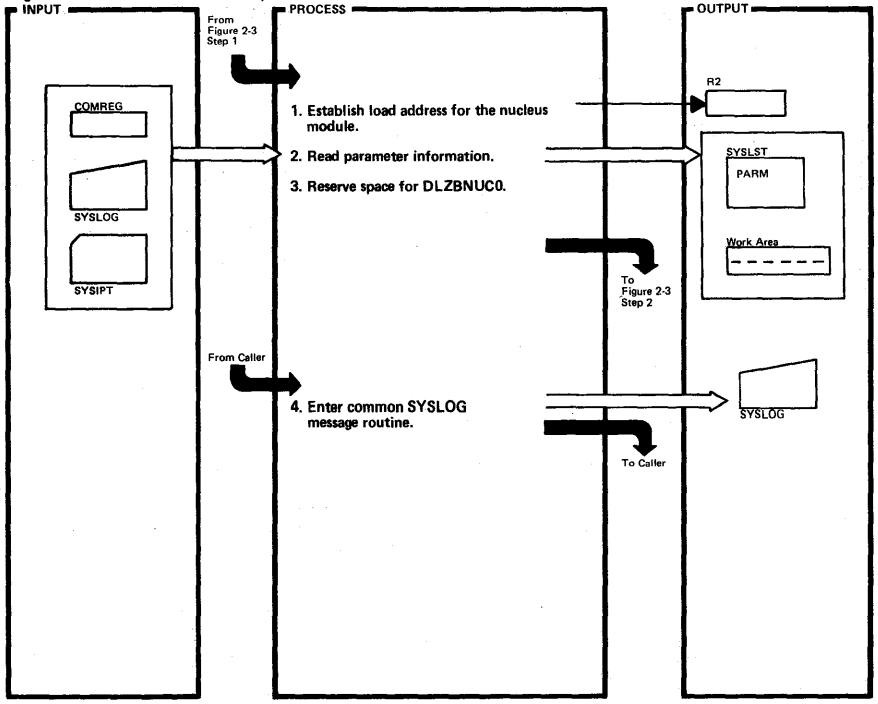
Figure 2-3. Batch Initialization (Overview)



Extended Description	Routine	Label
1.	DLZRRC00	
2.	DLZRRC10	
3.	DLZRRA00	
4.	DLZPCC00 DLZPINIT	
5.	ULUPRHEP	
6.	DLZPCC00	
7.	DLZPINIT	
8.	DLZCPI00	
9.	NXTPORT	

Extended Description	Routine	Label

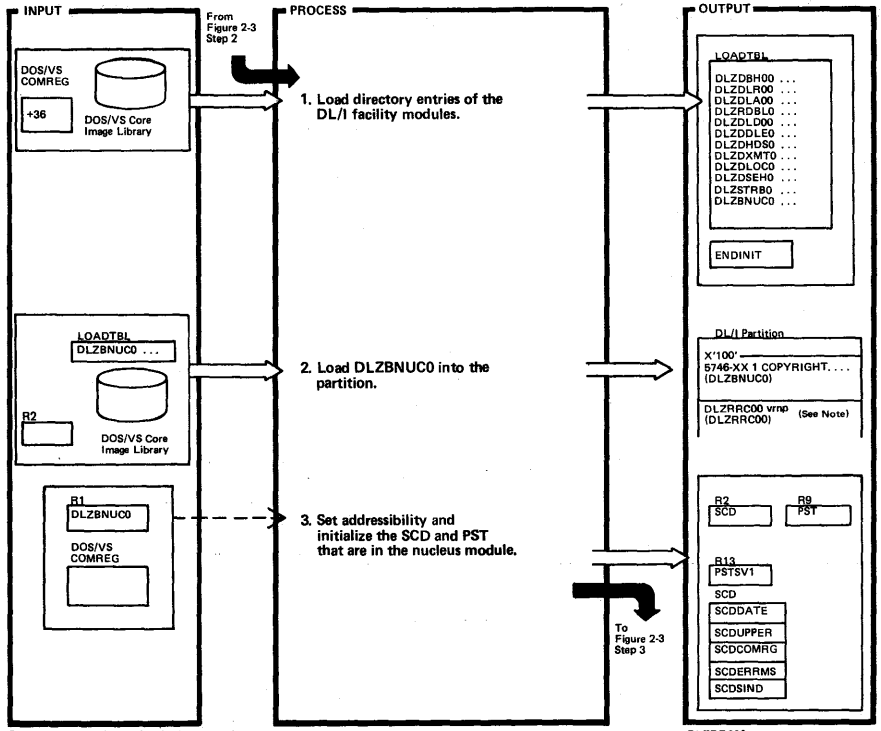
Figure 2-3.1. Batch Initialization Entry



Extended Description	Routine	Label
1. DLZBNUC0 load address is set at DLZRRC00 start + X'100'.	DLZRRC00	DLZRRC00
2.		PARMGET
3. The reserved space allows loading of DLZBNUC0 without overlaying critical code in this module.		
4.		ERRORMSG

Extended Description	Routine	Label

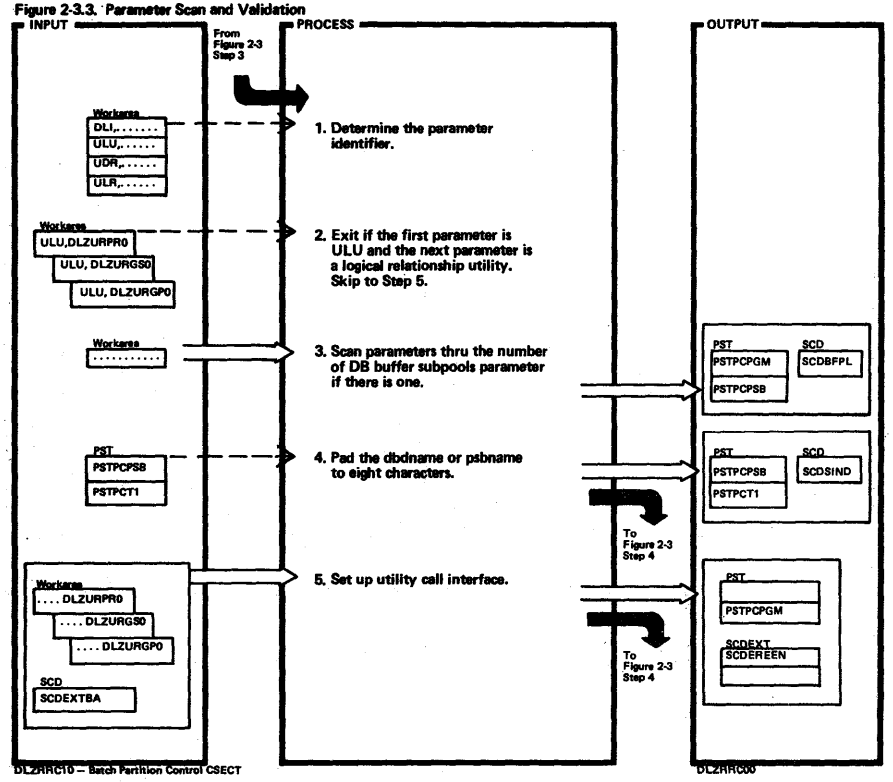
Figure 2-3.2. Batch Partition Control



Extended Description	Routine	Label
1. The end address of phase DLZRRC00 is obtained from the DOS/VS COMREG and saved at ENDINIT. Write message DLZ011I if a required module is not found in the DOS/VS core image library.	DLZRRC10	DLZRRC10 LOADNUC
2.		LOAD2
Note: DLZRRC00vrnp is the module identifier. Each DL/I module is identified with its full eight-byte module identifier in character format followed with a four-byte field identifying the module level. The level format is vrnp; where 'v' is the version, 'r' is the release, 'n' is an additional identification digit, and 'p' is the latest PTF number that has been applied.		

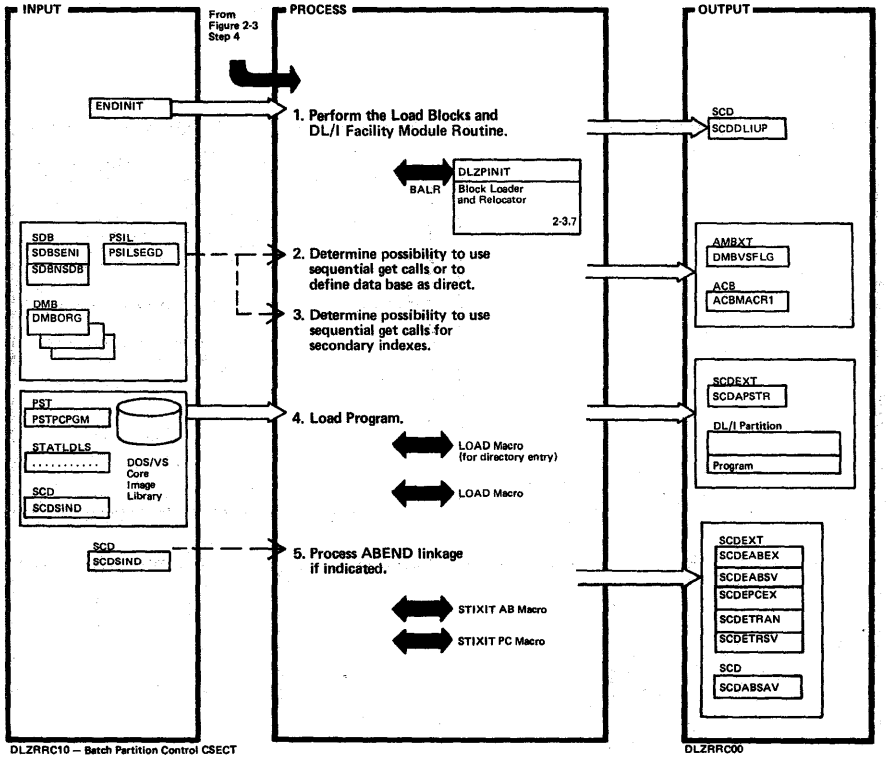
Extended Description	Routine	Label

Figure 2-3.3. Parameter Scan and Validation



Extended Description	Routine	Label
1. Write message DLZ015I if the first parameter is not DLI, ULU, UDR, ULR, or PLU. Also write message if PLU and program is not DLZURGU0 or if ULR and program is not DLZURGL0. Except for padding in Step 4, PLU is treated as ULU in all other places.	DLZRRA00	DLZRRA00
2. Although the DB prefix resolution utility is a logical relationship utility, it is not processed with the others because it executes directly, not as an application to DL/I.		CHK1ST
3. Write message DLZ015I if syntax error occurs.		SCANPARM
4. PSTPCPSB now contains the dbdname from the ULU, UDR, or ULR parameter card or the psbname from the DLI or PLU parameter card. Insert a utility DBD suffix (U) or insert a PSB suffix (P).		PARMPAD
5. No control blocks are loaded for DLZURPR0, DLZURGS0 or DLZURGP0 during batch initialization. These three utilities issue the DLZBLKLD macro specifying the utility PSB and the BLDB call for each data set used. The ACB utility builds the utility PSBs they use.	DLZRRC10	ULUSTART

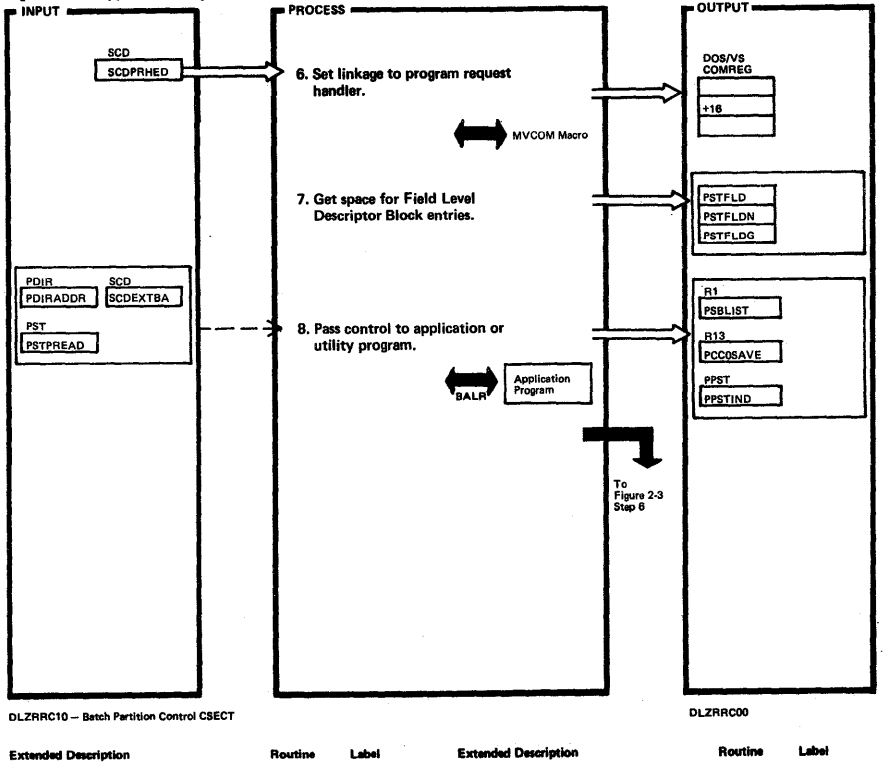
Figure 2-3.4. Application Program Control (Part 1 of 2)



Extended Description	Routine	Label
1. This module's end address is used to initialize the beginning of storage available for control block building.	DLZPCC00	DLZPCC00
2.		LOOKDMBS
3.		FINDISS
4. Write message DLZ012I if program is not found.		CONTPCC LOAD5
5. UPSI card information has been moved to the SCD.		STXITAB

Extended Description	Routine	Label

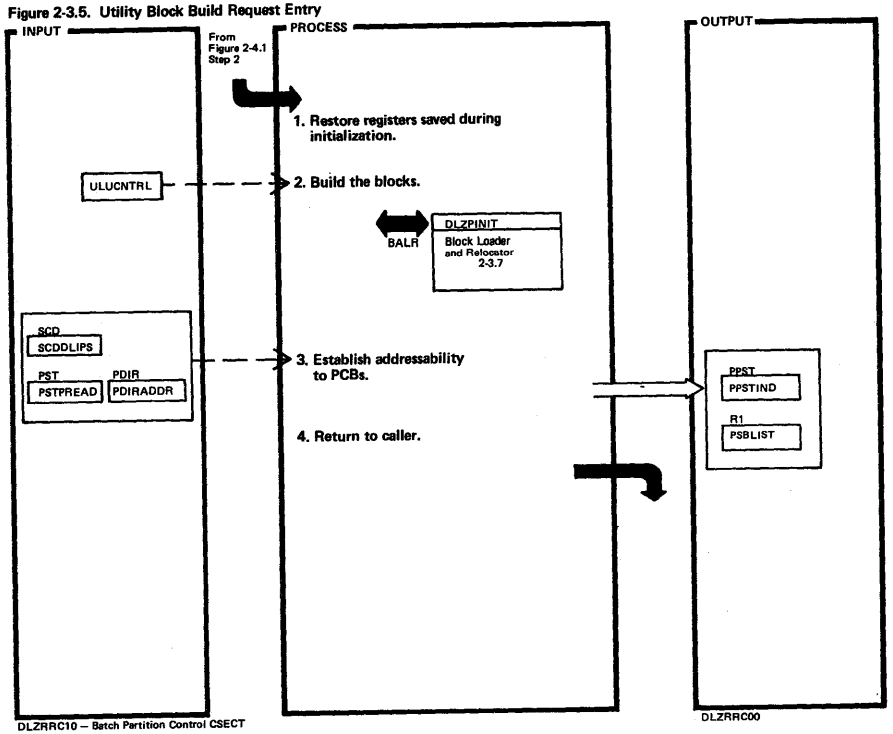
Figure 2-3.4. Application Program Control (Part 2 of 2)



Extended Description	Routine	Label
6. Linkage to DLZPRHB0 is done via MVCOM macro.		MVCOM
7. Use GETVIS macro. Issue message DLZ038I if GETVIS error.		GETVIS
8. If utility program is a logical relationship utility, set R1 to point to the PST before passing control to the utility. Set R1 to point to the user PCB list for all other programs.		BALRUSER

Extended Description	Routine	Label

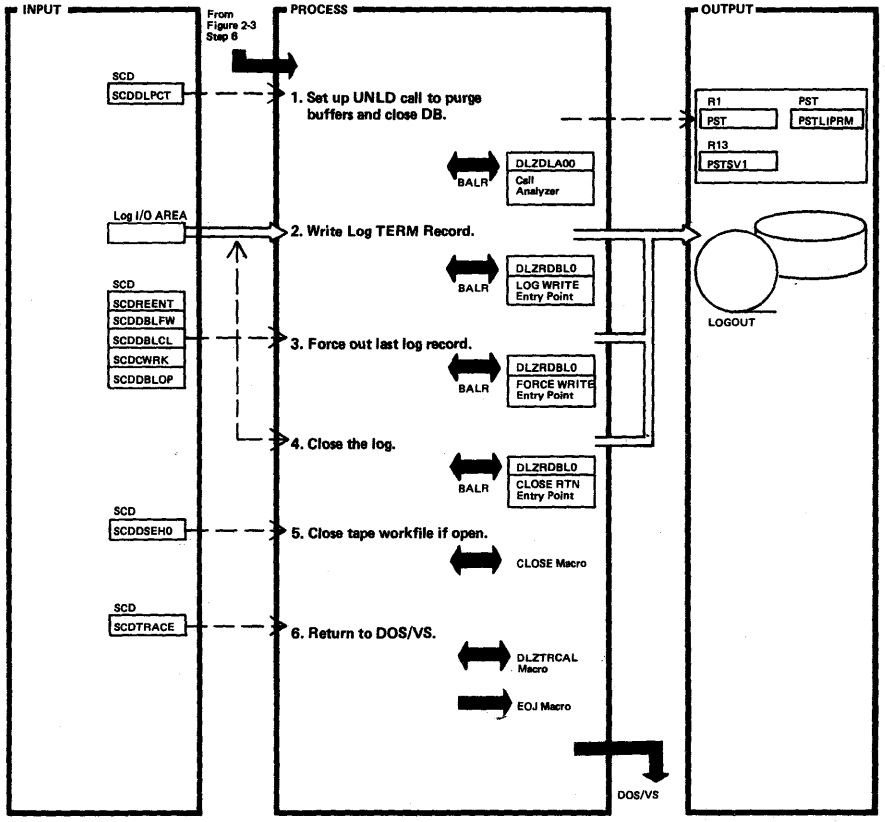
Figure 2-3.5. Utility Block Build Request Entry



Extended Description	Routine	Label
1. Control comes from the batch program request handler (DLZBNUC0) when a utility block build request (BLDB) is detected.	ULUPRHEP	ULUPRHEP
2. If a block build error is indicated in ULUCNTRL, X'0C' is set in register 15 and control returns to the utility program.		ULUGOOD
4.		ULUEXTZ

Extended Description	Routine	Label

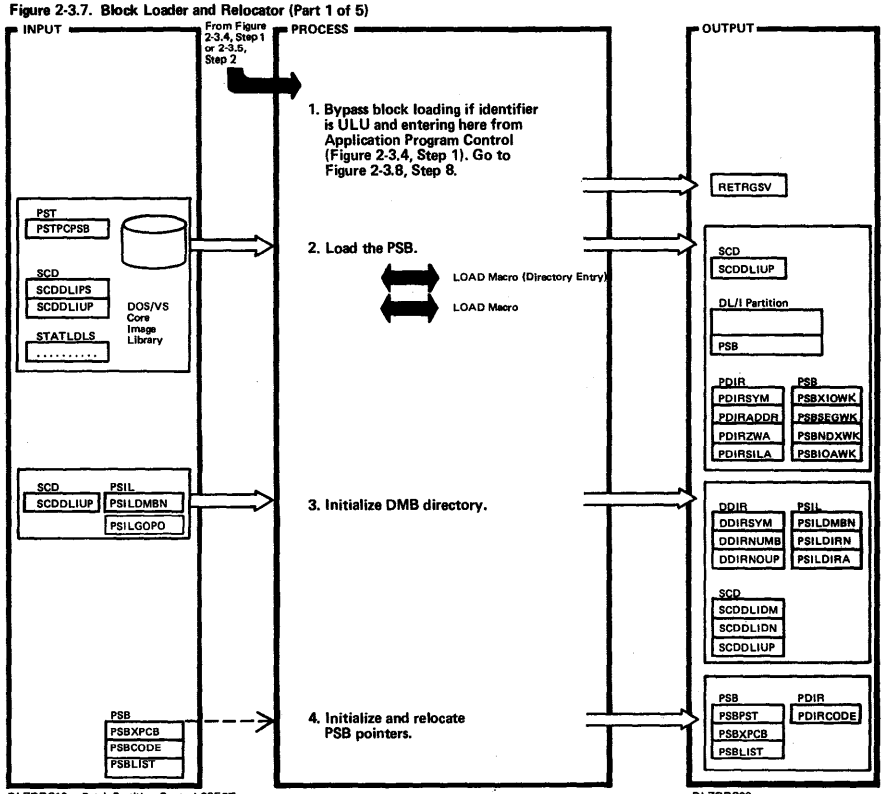
Figure 2-3.6. Application Program Control Completion



Extended Description	Routine	Label
1. TERM record ID=X'07'.	DLZPCC00	BALRUSER
4. The UNLD call is bypassed if the ULU return code in register 15 is not zero.		BYULUEND
6. Issue macro DLZTRCAL TYPE=STOP. Trace ID=X'FC'.		DLZEOJ

Extended Description	Routine	Label

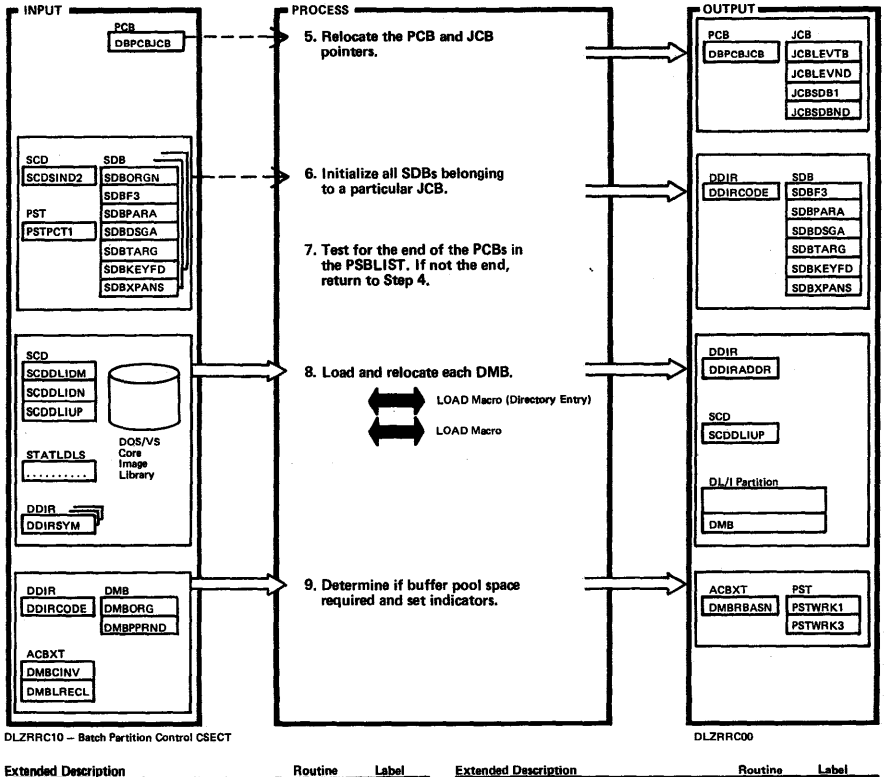
Figure 2-3.7. Block Loader and Relocator (Part 1 of 5)



Extended Description	Routine	Label
1. There are no blocks to load for the logical relationship utilities if this is the first call. The return address is saved in RETRGSV.	DLZPINIT	DLZPINIT
2. If the PCB is not found and the parameter identifier is ULU for a logical relationship utility, set a block load error indicator and return to Figure 2-3.5, Step 3. Write message DLZ012I if the PSB is not found. Write message DLZ017I if the PSB version/modification level is incorrect.		DLZDBLM0

Extended Description	Routine	Label
3. The PSILs are scanned for DMB names and a DDIR is created for each unique DMB encountered. The address of the DDIR replaces the respective DMBNAME in each PSIL.		DDIRBILD
4.		PCBRLUIP

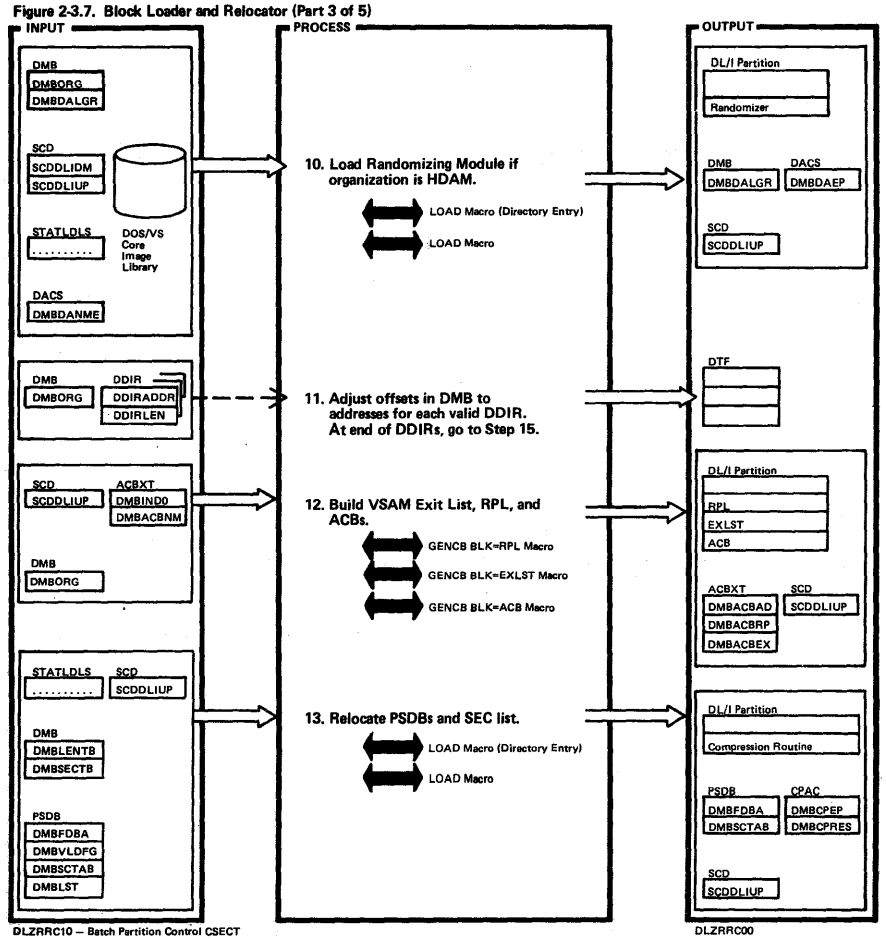
Figure 2-3.7. Block Loader and Relocator (Part 2 of 5)



Extended Description	Routine	Label
5.	DLZPINIT	PCBPLIB
6. The call sensitivity and data base organization of the SDB is checked to see if buffer pool space is required. An indicator in the DDIR is turned on if space is required.		SDBRELO
7. The pointer to the PSBLIST is bumped to the next PCB pointer entry and processing returns to Step 4 if we are not at the last PCB. If the index PCB exists, return to Step 5 and relocate it.		NPCBCK
8. Write message DLZ012I if the DMB is not found. Write message DLZ018I if the DMB version/modification level is incorrect.		LOADDMBS DMBLOADR

Extended Description	Routine	Label
If the DMB is not found and the parameter identifier is ULU for a logical relationship utility, set a block load error indicator and return to Figure 2-3.5, Step 3.		
9. If buffer pool space is required, the size of each control interval (rounded to the next multiple of 512) is indicated in PSTWRK1 and PSTWRK2 for later allocation of the buffer pool.		GETBUFRS

Figure 2-3.7. Block Loader and Relocator (Part 3 of 5)



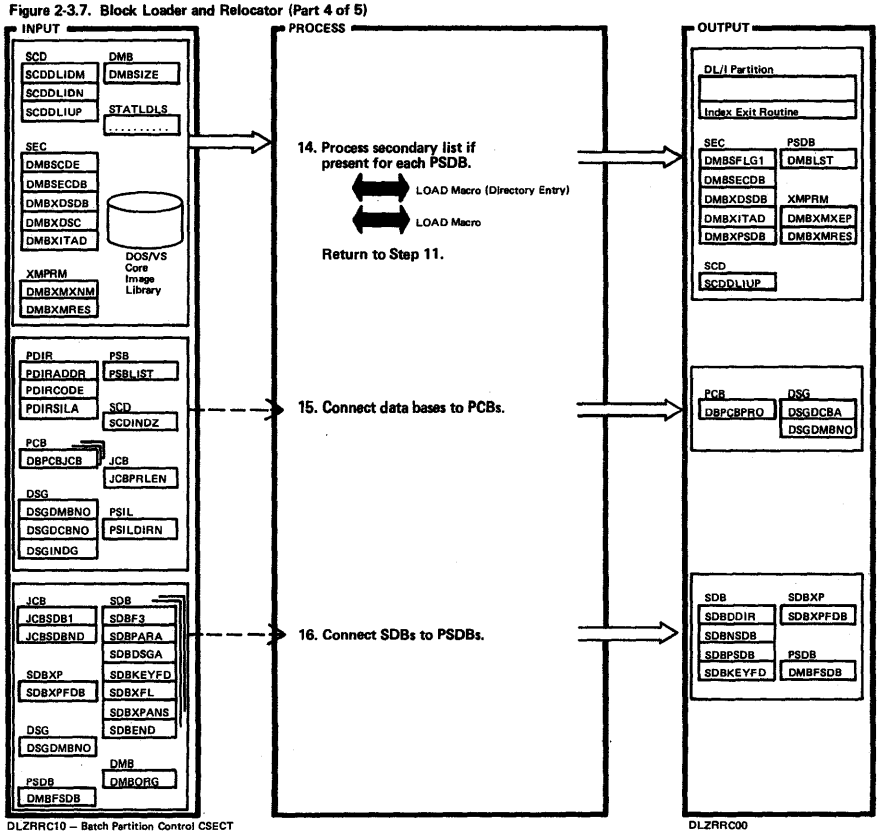
DLZRRC10 – Batch Partition Control CSECT

DLZRRC00

Extended Description	Routine	Label
10. Before loading the randomizer a check is made with all currently loaded randomizers. If one with the same name as the one we are loading is found, the entry point is resolved and the actual load is bypassed. Return to Step 8 until there are no more DDIRs. Write message DLZ012I if the randomizing module is not found.	DLZPINIT	RANCKLUP

Extended Description	Routine	Label
11. The DTF address constants are adjusted if access is HSAM or simple HSAM.		DMBOFFAJ
12. If HISAM, two sets of control blocks will be built.		DLZBVBLD
13. The segment compression routine for each PSDB is loaded (if it hasn't already been loaded for a previous PSDB). Write message DLZ012I if the compression routine is not found.		PSDBROUT

Figure 2-3.7. Block Loader and Relocator (Part 4 of 5)



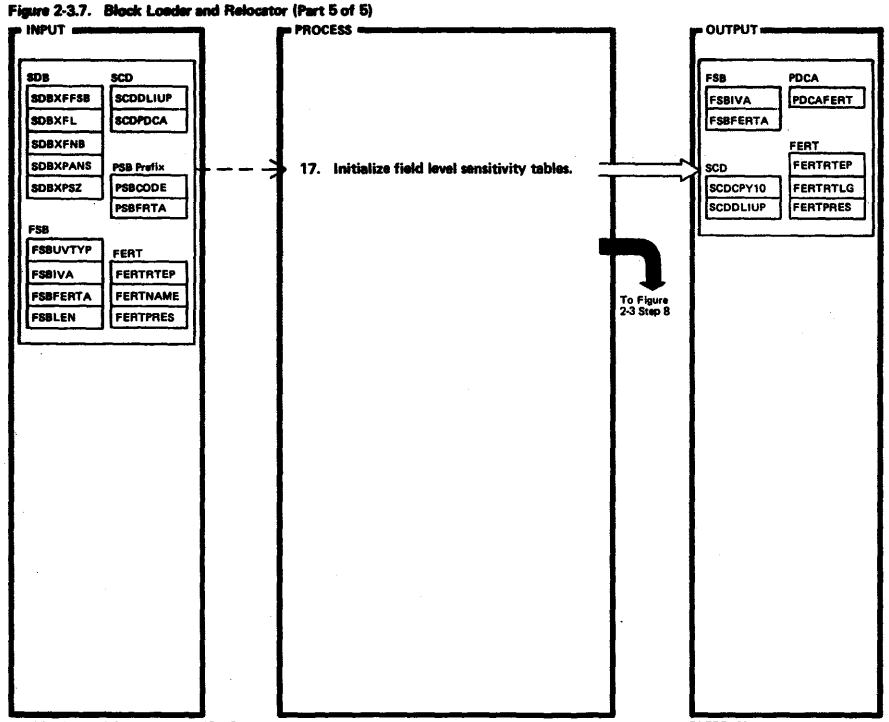
DLZRRC10 – Batch Partition Control CSECT

DLZRRC00

Extended Description	Routine	Label
14. If a secondary list is present, its code is tested and referenced DMBs are resolved to DDIR pointers and placed in the list. If an index user exit routine is present it is loaded if it hasn't already been loaded for a previous SEC. Write message DLZ266I if there is an invalid secondary list code. Write message DLZ012I if the user exit routine is not found. Write message DLZ263I if the SEC makes an invalid DMB reference.	DLZPINIT	PROCSEC

Extended Description	Routine	Label
15. If this is Reload Restart, the parameter ID is ULR. Bypass checking whether the processing option should be changed to load or not.		PCBROUT
16.		CONSDBS

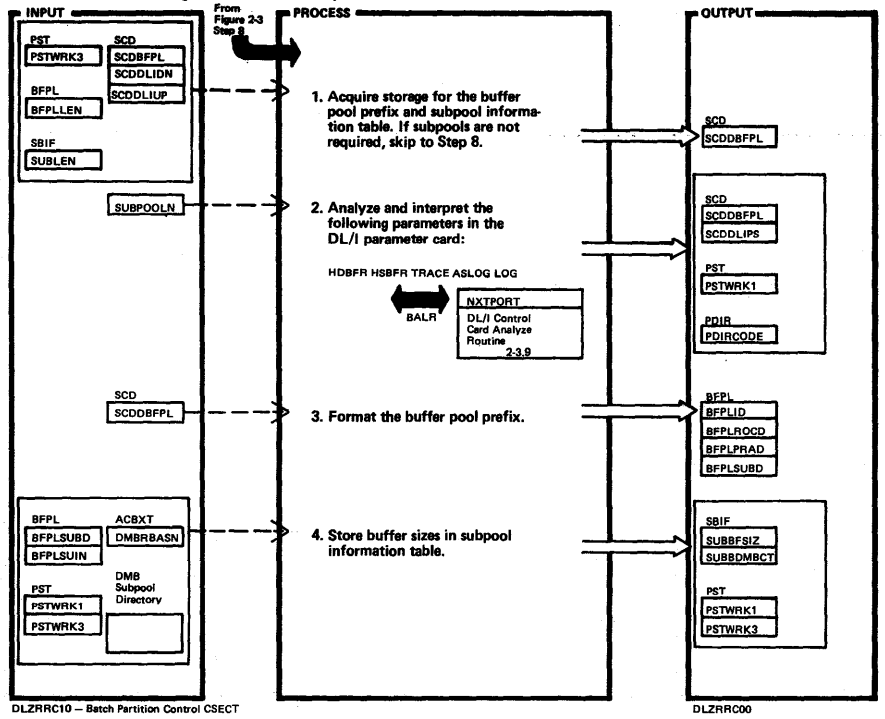
Figure 2-3.7. Block Loader and Relocator (Part 5 of 5)



Extended Description	Routine	Label
17. Includes FERT, FSB, and loading user field exit routines.		

Extended Description	Routine	Label

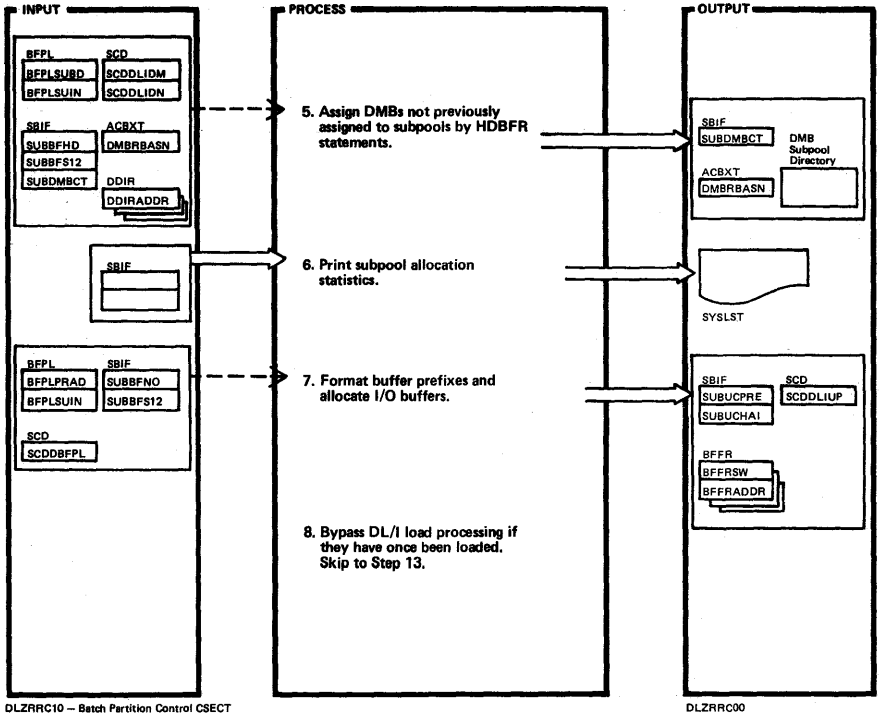
Figure 2-3.8. Control Program Initialization Completion (Part 1 of 3)



Extended Description	Routine	Label
1.	DLZCPI00	DLZCPI00 BFRPRNT
2. Write message DLZ009I if number of subpools specified in the parameter statement are not equal to the number of HDBFR statements.		BFPNDCLR PRMSRET
3.		BFPREADY
4. This step determines the size of the subpools. They are allocated, largest first, until the specified number is exhausted. Remaining DMBs requiring subpools are assigned evenly across all existing subpools. If you specified more subpools than necessary, an additional pool of 512 buffer size is allocated for delete workspace. The subpool sizes are sorted so that the largest subpool appears first in the information table.		NODMMOV

Extended Description	Routine	Label

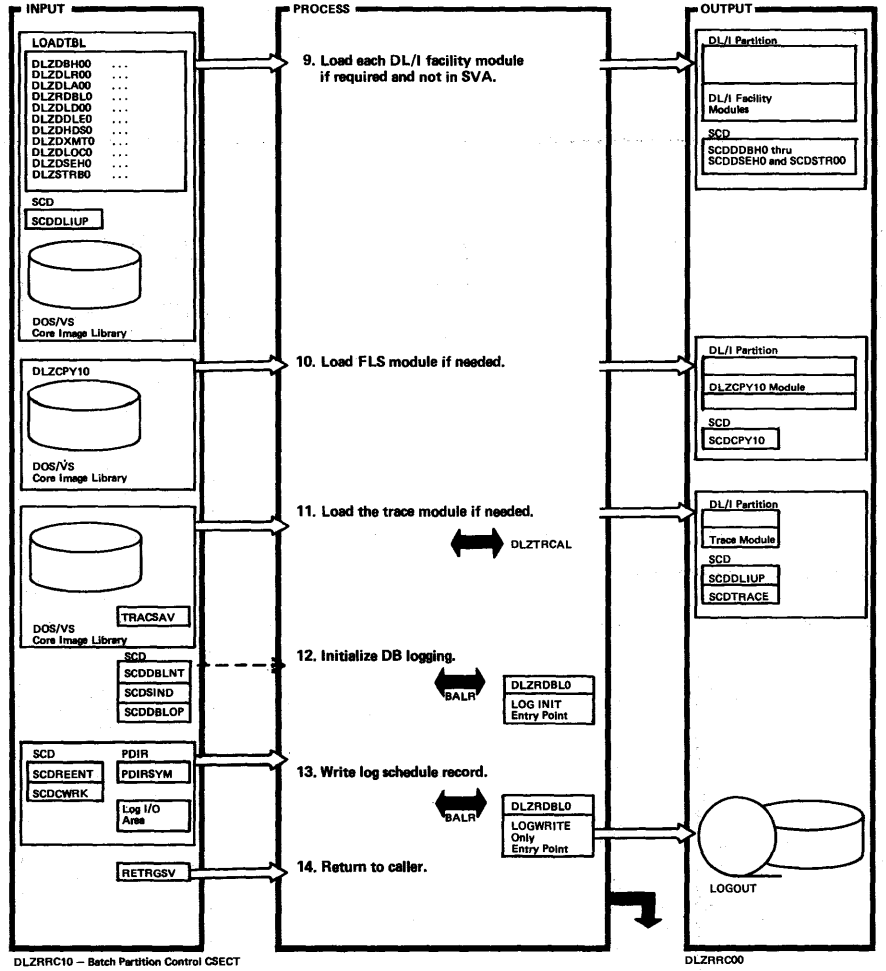
Figure 2-3.8. Control Program Initialization Completion (Part 2 of 3)



Extended Description	Routine	Label
5. Write message DLZ262I if buffer allocation error occurs.		GREATPRO
6.		SPSTAT
7.		BFRINIT
8.	DLZCPI00	DLILOAD

Extended Description	Routine	Label

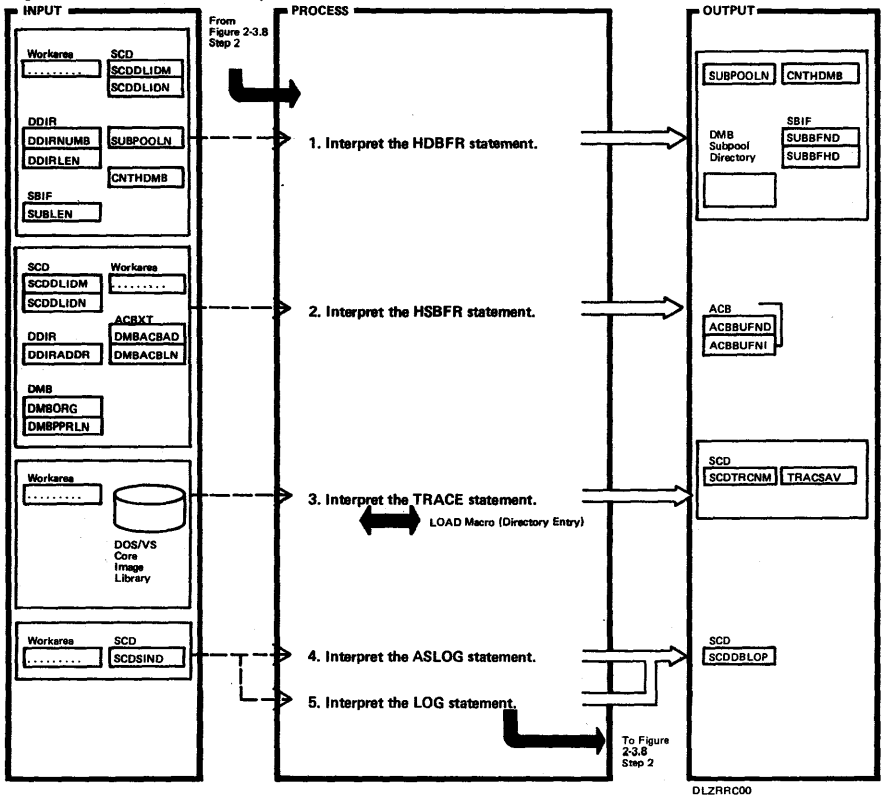
Figure 2-3.8. Control Program Initialization Completion (Part 3 of 3)



Extended Description	Routine	Label
9.		NUCLODUC
11. Issue macro DLZTRCAL TYPE=START following the load. Trace ID=X'FE'. Write message DLZ026I if initialization fails.		LOAD9

Extended Description	Routine	Label
12. Cancel if open error returned. Upon return, the entry points to DLZRDBL0 in the 'Data Base Change Log Section' of the SCD (beginning with the SCDREENT) are initialized.		NOLOMOD
13. The scheduled record ID='08'.		
14. Return is made to the instruction following the BALR to DLZPINIT.		PCCORET

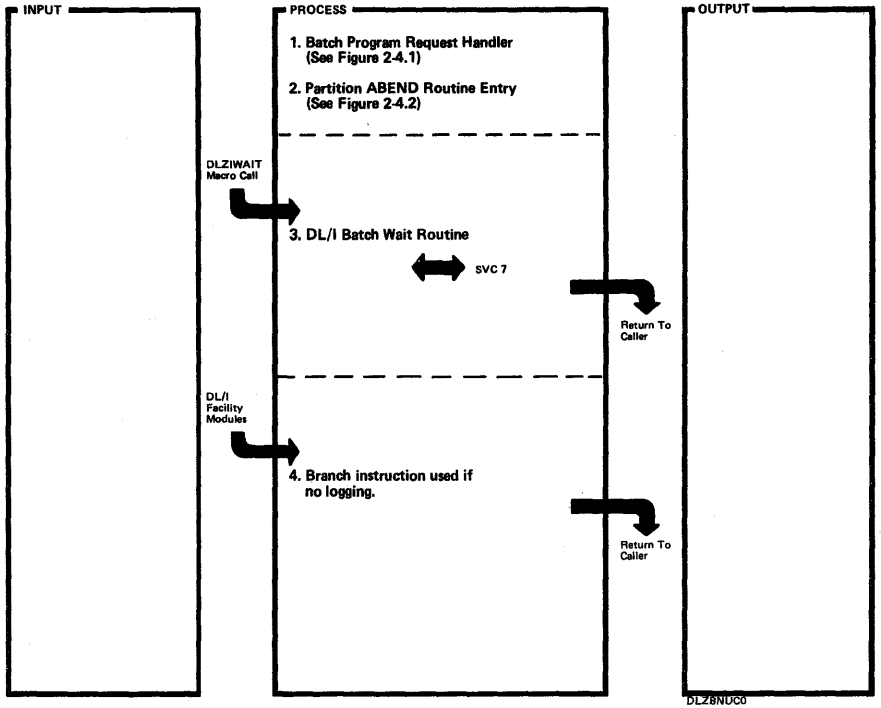
Figure 2-3.9. DL/I Control Card Analyze Routine



Extended Description	Routine	Label
1. The number of buffers/subpools specified in the HDBFR statement is set in the SBIF. Write message DLZ019I if the number is greater than 32 or less than 2. Default is 2. The SUBPOOLN is incremented 1 for every HDBFR statement. Each DMB is assigned by placing the relative subpool number (SUBPOOLN) it is being assigned to into a byte of the DMB SUBP DIR which corresponds to that DMB. The length in bytes of the DMB SUBP DIR equals the total number of DMBs. Write message DLZ008I if this DMB has already been assigned a subpool. CNTHDMB is a count of all the data bases assigned by the user in the HDBFR statements.	NXTPORT	NXTPORT HDBFR

Extended Description	Routine	Label
Write message DLZ008I if a DMB name is invalid.		
2. The user specified VSAM buffer allocations are set in the ACB for HISAM and INDEX DBDs. Write message DLZ008I for an invalid DMB reference. Write message DLZ019I if valid values were not specified.		HSBFR
3. Write message DLZ012I if module is not found.		TRACE
4. Write message DLZ015I if there is a syntax error.		ASLOG
5. Write message DLZ078I if UPSI card said no log. Write message DLZ075I if invalid parameters.		LOG

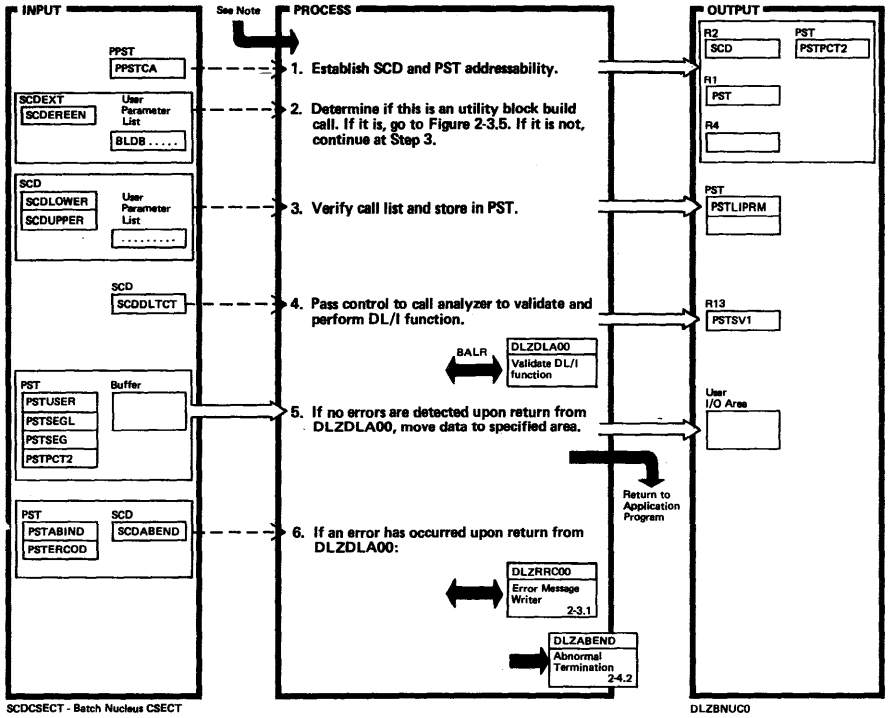
Figure 2-4. Batch Nucleus (Overview)



Extended Description	Routine	Label
3. The DLZIWAIT macro is used by DLZRDBH00, DLZDBH02 and DLZRDBL0.	DLZIWAIT	DLZIWAIT
4. After the DLZBNUC0 module is loaded, SCDDBLNT contains the entry point of this routine. If, however, batch initialization (DLZRRC00) determines that the DB logger is required, the entry point of the log initialization routine in DLZRDBL0 is stored in SCDDBLNT. The log initialization routine changes SCDDBLNT once more to point to the log writer entry point. With this routine, the DL/I facility modules need not know if logging is required or not.	DLZBR14	DLZBR14

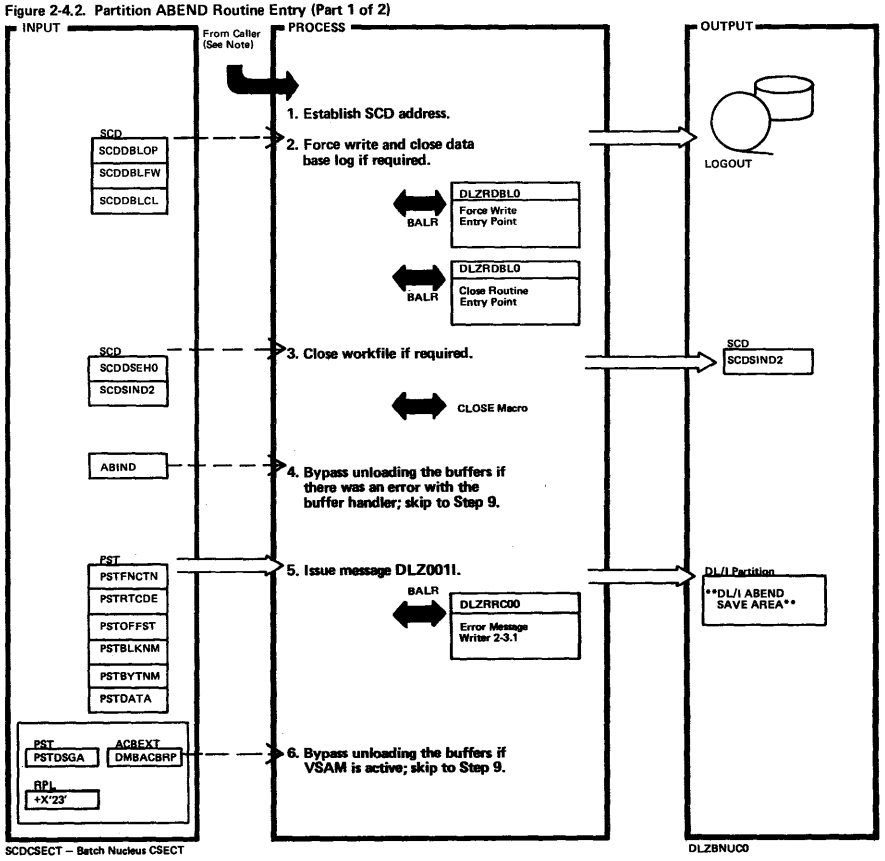
Extended Description	Routine	Label

Figure 2-4.1. Batch Program Request Handler



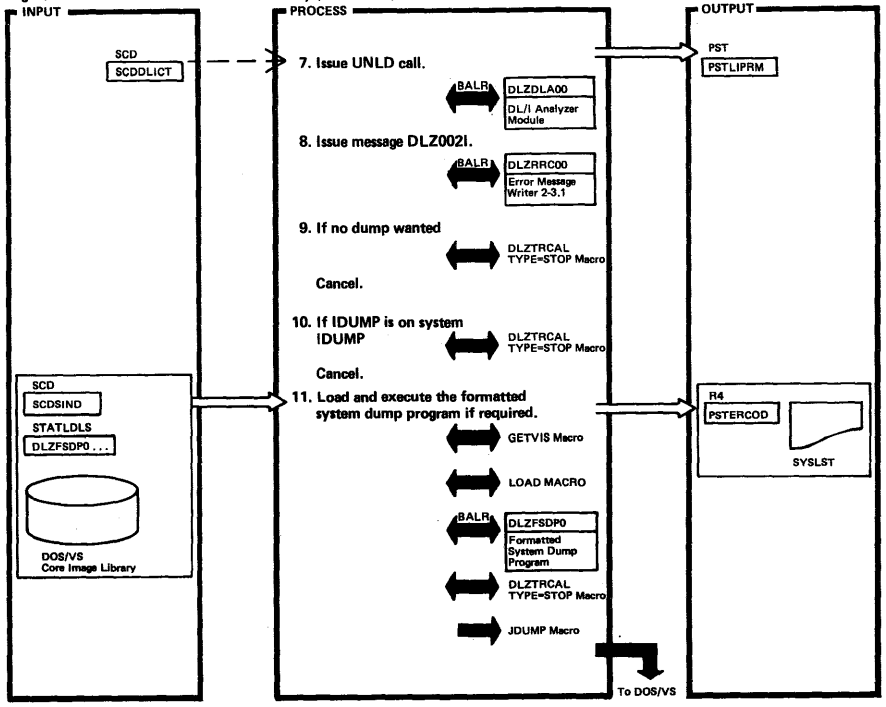
Extended Description	Routine	Label
Note: This routine receives control from the language interface module (DLZLI000) linked with the application program.		
1. When control is passed to the program request handler, register 1 must point to the user parameter list and register 13 to the user save area. During the first entry to DLZPRHB0, the PL/I STXIT routine and savearea addresses from the PC option table are saved if the application program is written in PL/I. DLZPRHB0 also sets/resets a switch (SCDLIPL1 flag in SCD) on exit/entry to indicate whether current execution is in DL/I code or PL/I code. This is done to enable high level language debugging for PL/I to give diagnostic information if a program check occurs in PL/I code. Reset PC exits if this is a PL/I application.	DLZPRHB0	DLZPRHB0
2.		BYPLSTXT
3. Write message DLZ260I if invalid list count. Write message DLZ261I if invalid parameter address. Then exit to DLZABEND.		CNTLUP
4.		MOVLUPBP
5. Write message DLZ105I if a checkpoint was taken.		
6. If a DL/I routine determined that DL/I should be terminated, go to the common error message routine to write an error message using the message number stored in PSTERCOD by the DL/I routine.		PRHABEND

Figure 2-4.2. Partition ABEND Routine Entry (Part 1 of 2)



Extended Description	Routine	Label
Note: The ABEND routine is invoked by the DOS/VS supervisor if (1) there is a program check or other ABEND situation found by DOS/VS, (2) if the job is being abnormally ended by a DL/I routine that determines DL/I should be abnormally ended, or (3) specifically by the buffer handler when there is an error concerning buffers.		
1. If there is a program check, DLZABEND checks the switch (SCDLIPLI flag in SCD) set by DLZPRHB0 to determine if program check occurred in PL/I code. If error occurred while in PL/I code (SCDLIPLI=1), a return address is modified and a branch is made to PL/I STXIT PC routine. (The address of the PL/I STXIT PC routine was saved during the first entry to DLZPRHB0 - see Figure 2-4.4, Step 1.) After PL/I completes diagnostic information, processing returns to the modified address in DLZABEND.	DLZABEND	DLZABEND
3. If the HD reorganization reload module (DLZURGL0) is running for either a standard reload or a reload restart, close the workfile generator file if it is open.		ABLOGCBP
4.		RELODCBP

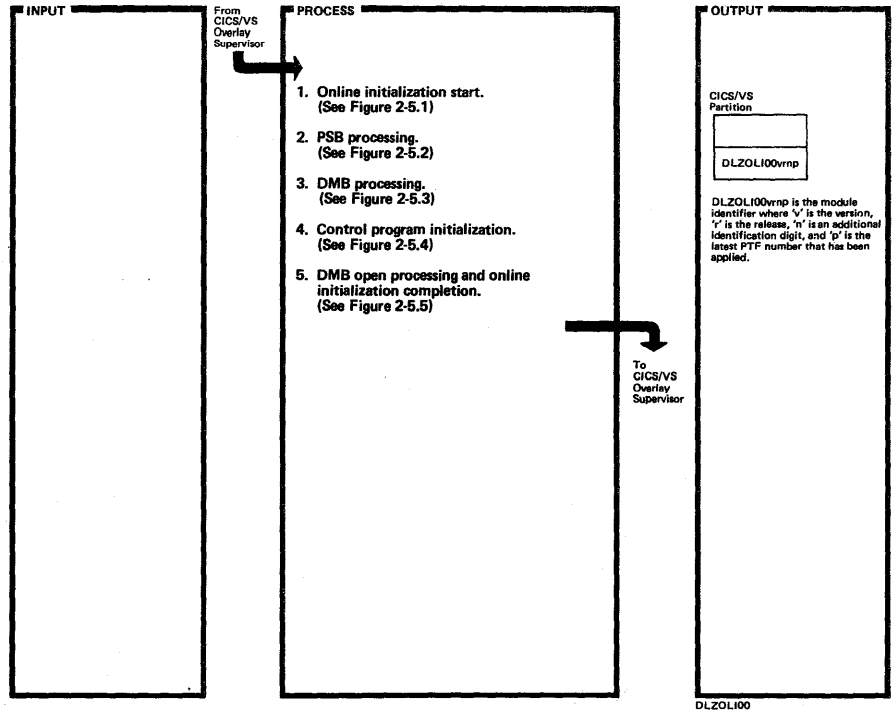
Figure 2-4.2. Partition ABEND Routine Entry (Part 2 of 2)



Extended Description	Routine	Label
7.		ABUNLD
9.		ABBYMSG
10. DLZIDUMP macro determines if IDUMP is available.		DLZIDUMP
11. The GETVIS macro is used to acquire storage for DLZFSDP0. If there is not enough storage available to DLZFSDP0, only JDUMP output is put to SYSLST.		ABBYMSG

Extended Description	Routine	Label

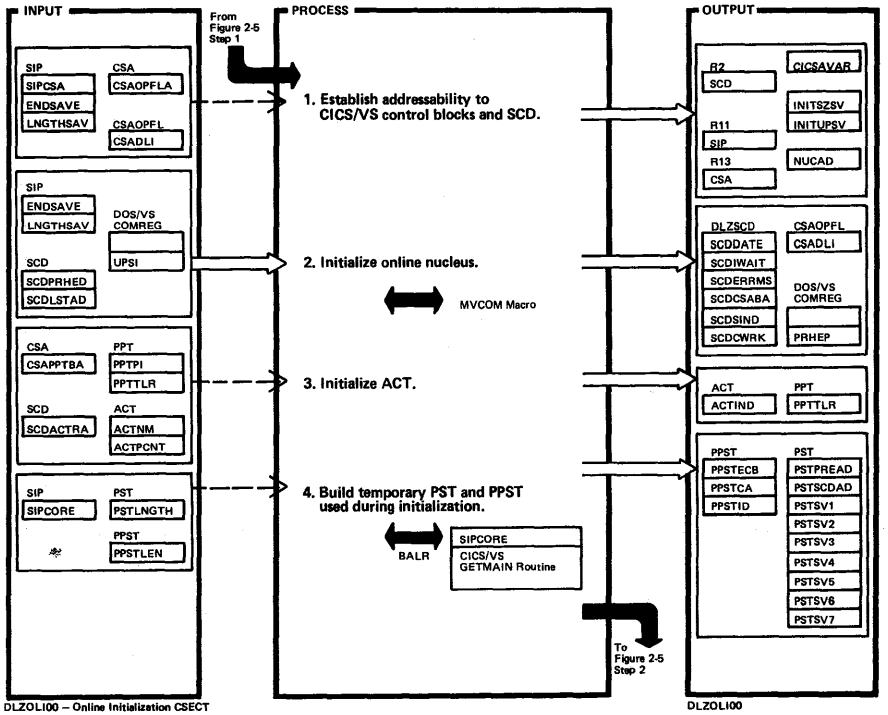
Figure 2-5. Online Initialization (Overview)



Extended Description	Routine	Label
1.	DLZOLI00	
2.	PSBLOADL	
3.	DDIRINIT	
4.	DLZCPI00	
5.	DMBOPENA	

Routine	Label

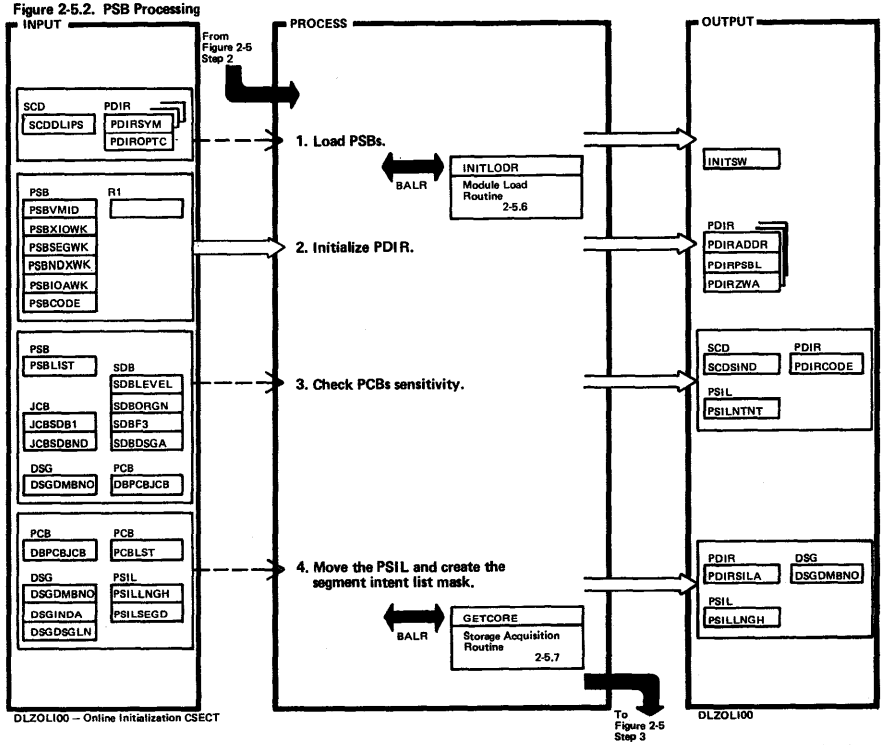
Figure 2-5.1. Online Initialization Start



Extended Description	Routine	Label
1. Module identifier (DLZOLI00vrmp) is defined here. Upon entry from the CICS/VS Overlay Supervisor, SIPBAR2 contains the overlay entry point, and SIPBAR1 contains the SIP common communications area. The current storage allocation information is saved in order to release storage if DL/I initialization fails. The DL/I systems contents directory is located from the CSADLI field in the CSA optional features list (CSAOPFL). Write message DLZ031I if program isolation is being used and either CICS/VS journalling is not being used or program name DFHDBP (dynamic transaction backout) is not in the PPT. Write message DLZ050I if the nucleus is not found. Write message DLZ064I if the nucleus is an invalid version.	DLZOLI00	DLZOLI00
2. CSADLI is modified to point to the DL/I interface module address list, (DFHDLIAL DSECT), which is a table of entry points for the task initiation module and the task and system termination routines.		NUCFOUND

Extended Description	Routine	Label
SCDSIND is initialized with bits 6 and 7 of the UPSI switch from the COMREG. The program request handler entry point is moved to byte 16 of the COMREG and temporary entry points are established for the error message routine and the DL/I wait routine. Also, SCDCWRK is initialized at this time to point to the beginning of the low end of free storage.		
3. Indicators are set in the CICS/VS PPT marking the program eligible for DL/I services. They are set in the DL/I ACT entry indicating the program was located in the PPT. Write message DLZ034I for each ACT program not in the PPT. Write message DLZ051I if any ACT program is not in the PPT.		ACTLOOP
4. The PST and PPST are built directly after the initialization overlay high storage address. The save areas are chained and SCDCWRK is updated to indicate the new upward core allocation starting address. A dummy task ID of '01' is set in the PPST. Since a DL/I task with its associated PST is not yet involved, a temporary PST is needed to provide work space and save areas during the execution of this module.		PSTPPST

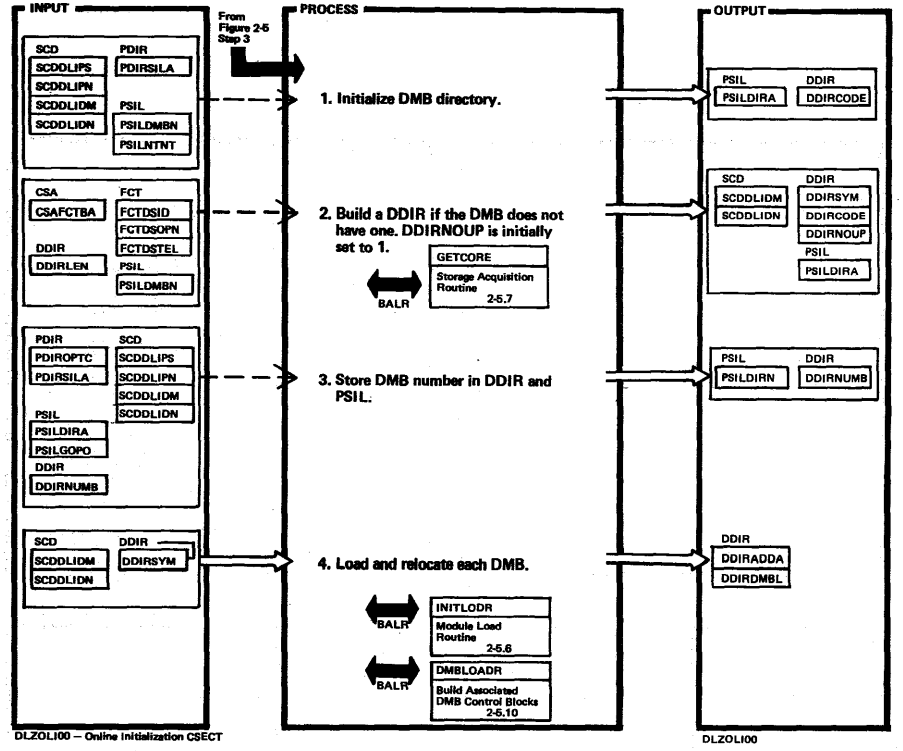
Figure 2-5.2. PSB Processing



Extended Description	Routine	Label
1. The PDIR address is located in the SCD and each local PSB is loaded temporarily, directly behind the dummy PST. The PSIL entries that indicate the DMBs which may be used by this PSB are loaded along with the PSB. They are appended in front of the PSB. If PSB initialization is successful, it will be moved up prior to completion of initialization. Write message DLZ044I if the PSB is not found.	PSBLOADL	PSBLOADL
2. Write message DLZ071I if the PSB is not version/modification 1.1 or later.		
3. All the SDBs for each PCB in a PSBLIST are tested for correct processings options. Indicators are set in the corresponding PDIR entry, PSIL entry, and the SCD to indicate intent. The corresponding SIL entry is found by using the offset value found in DSGDMBNO. Write message DLZ042I if a PSB accesses a HSAM DBD online.		PDIRNPCB

Extended Description	Routine	Label
Write message DLZ043I if load sensitivity is detected. A PSB contains a PCB with PROCOPT=L which is invalid online.		
4. The PSILs are moved from the temporary position and as they are moved, the size of each entry is increased by the size of PSILSEGD to allow for the segment intent bits mask copy. For program isolation, non-exclusive intent bits are translated to read-only to allow simultaneous scheduling of update-sensitive segments. All the DSGDMBNOs are adjusted to show the new offset. Also, each PSILLNGH is adjusted to show each new PSIL entry length as each entry is moved.		PDIRSMUV
Return is made to Step 1 to repeat this routine for PSB until there are no more.		PSBNXT

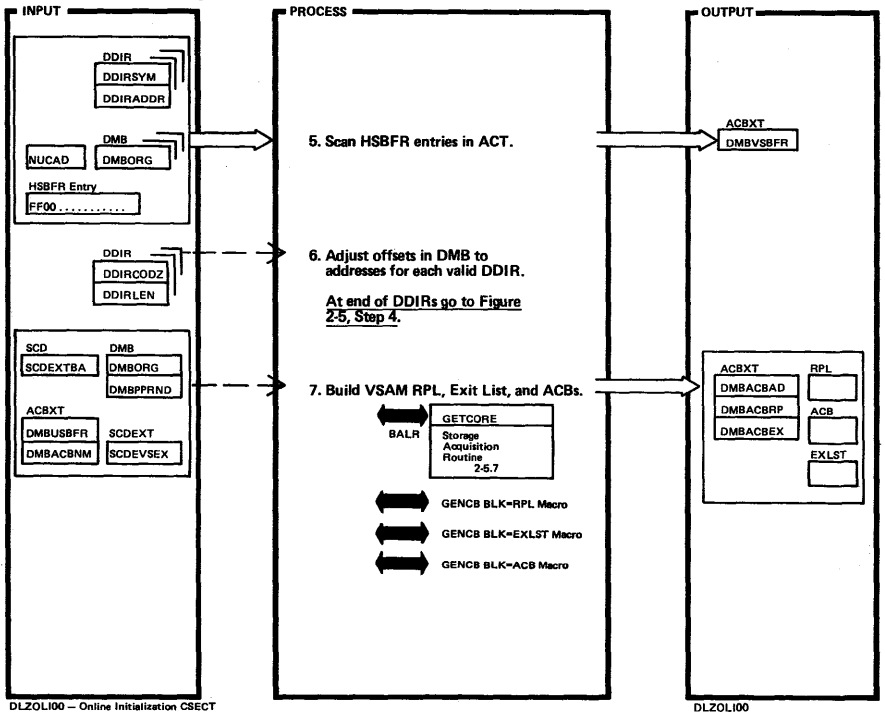
Figure 2-5.3. DMB Processing (Part 1 of 3)



Extended Description	Routine	Label
1. The PSILs are scanned for DMB names and a DDIR is created for each unique DMB encountered. The address of the DDIR replaces the respective dmbname in each PSIL. If PSILGOPO = 0, set DDIRNOUP = 0 for corresponding DMB.	DDIRINIT	DDIRINIT DDIRFOND
2. Write message DLZ045I if no CICS/VS FCT. Write message DLZ046I if no FCT entry existed matching the dbdname.		DDIRBLD
3. Write message DLZ049I if no valid DMBs are found.		DDIRNUML
4. Write message DLZ047I if DMB not in library. Write message DLZ072I if the DMB is not version 1.1 or later. Write message DLZ048I if the randomizing module is not found.		DMBLLUP

Extended Description	Routine	Label
Write message DLZ049I if no valid DMBs are found.		

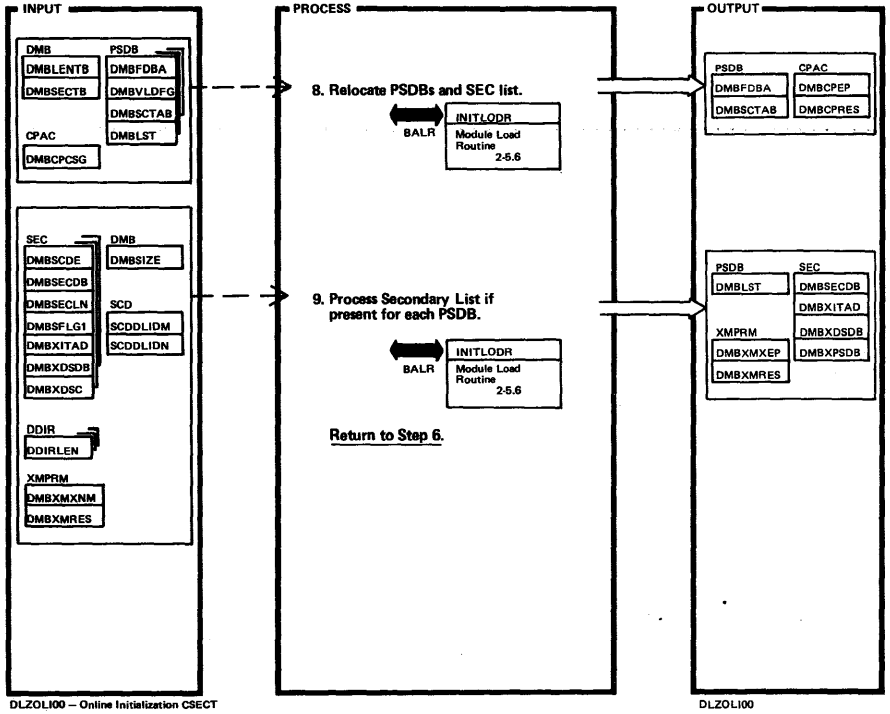
Figure 2-5.3. DMB Processing (Part 2 of 3)



Extended Description	Routine	Label
5. Write message DLZ029I if invalid DBDNAME in HSBFR statement. The number of index buffers and KSDS buffers in the HSBFR entry is moved to the ACB extension. If the organization is HISAM the number of ESDS buffers is moved to the second ACBXT. These values are used in building the VSAM ACBs (in Step 7).	DDIRINIT	CHKHSB
6.		DMBRLUP
7. If HISAM, two sets of control blocks will be built. Information obtained from HSBFR statements is used for the GENCB BLK=ACB BUFND=parameter and BUFNI parameter. If none was specified, the default of 3 index buffers and 2 data buffers is used.		DMBOFFAJ ACBADLUP

Extended Description	Routine	Label

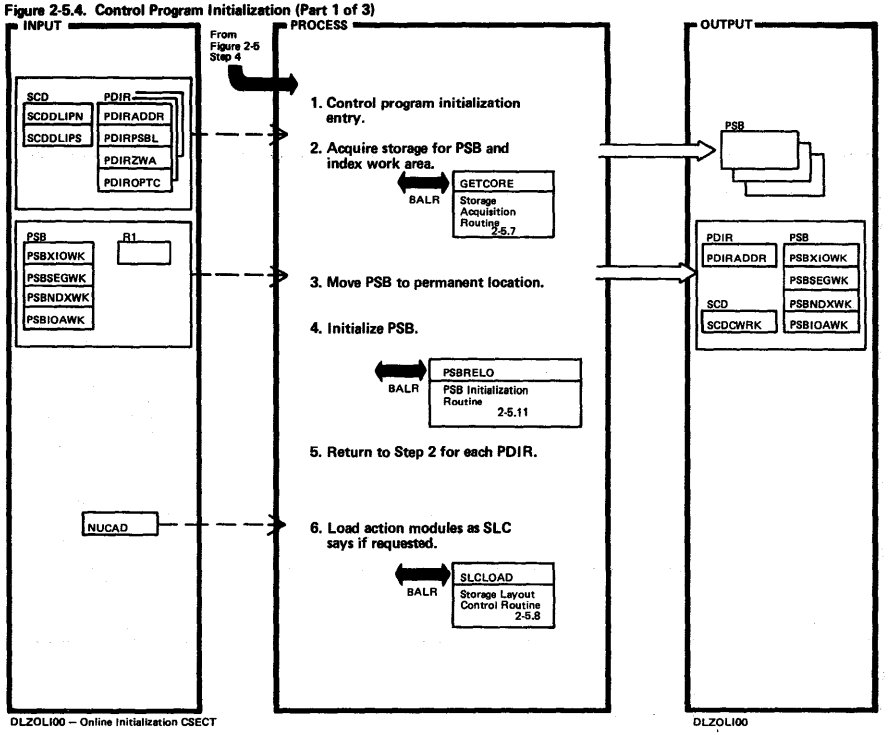
Figure 2-5.3. DMB Processing (Part 3 of 3)



Extended Description	Routine	Label
8. The segment compression modules for each PSDB is loaded if present. Write message DLZ073I if the compression module is not found.	DDIRINIT	PSDBROUT
9. If a secondary list is present, its code is tested, and referenced DMBs are resolved to DDIR pointers and placed in the list. If an index user exit routine is present it is loaded. Write message DLZ013I if the SEC makes an invalid DMB reference. Write message DLZ266I if there is an invalid secondary code. Write message DLZ074I if the indexing module is not found.		PROSEC

Extended Description	Routine	Label

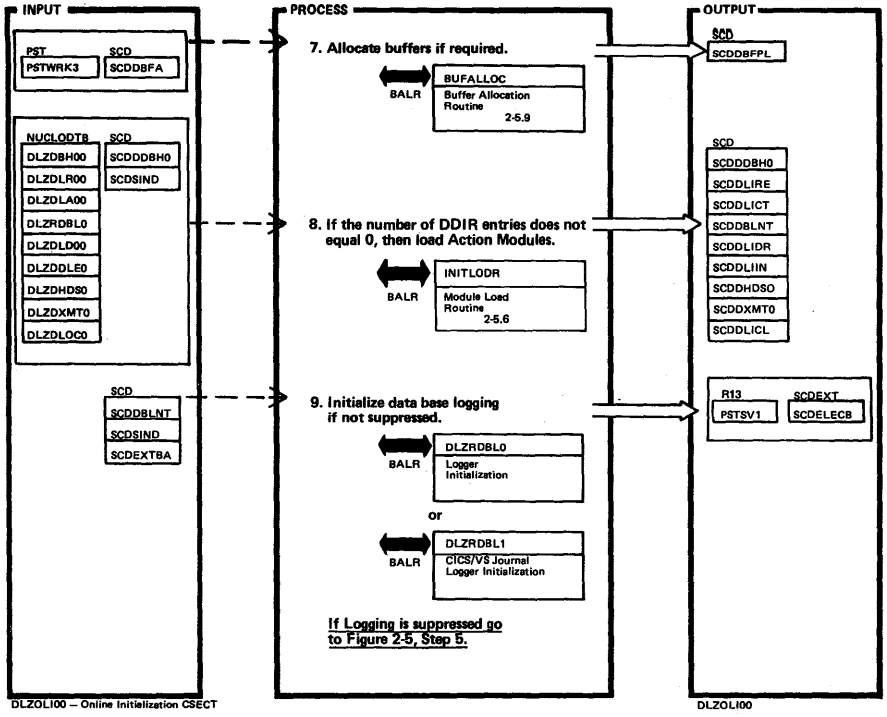
Figure 2-5.4. Control Program Initialization (Part 1 of 3)



Extended Description	Routine	Label
1.	DLZCPI00	DLZCPI00
2. The address of the storage acquired is used as the 'move-to' address in Step 3.		PCCORET PSBMLUP
3. Once the PSB is moved, its new address is stored in the PDIR and the old PSB address becomes the new code upper boundary address (upward core allocation starting address), SCDCWRK.		
4. Write message DLZ056I if PSB fails to initialize.		PSBNWA4
6. If there is an SLC name at the nucleus address +8, the SLCLOAD routine loads the action modules. Write message DLZ032A and terminate DL/I initialization if CICS/VS journalling is to be used but there is no system log entry (X'01') in the JCT.		DLILOAD NUCLDNJL

Extended Description	Routine	Label

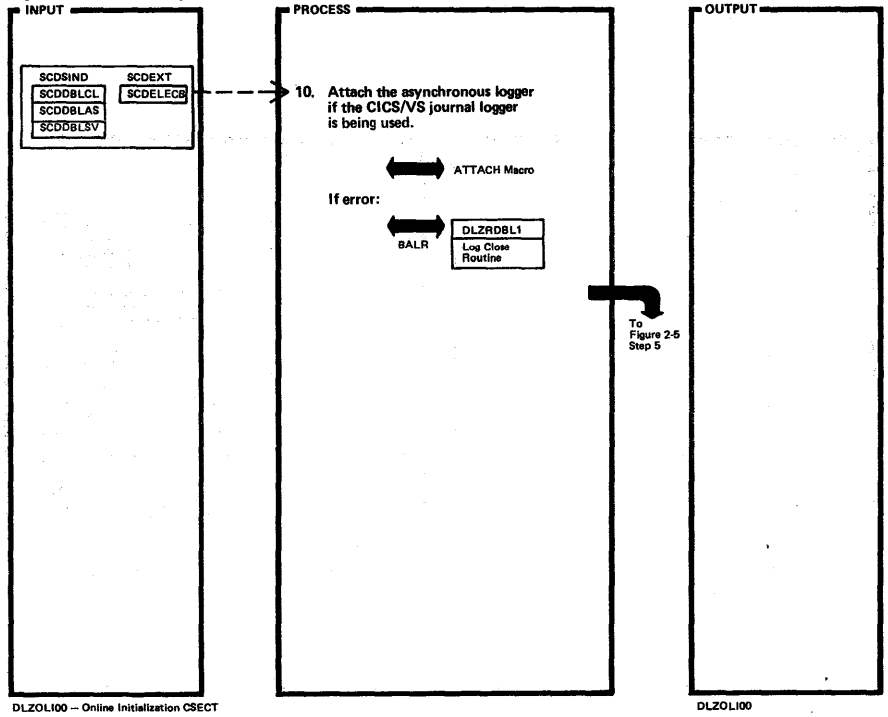
Figure 2-5.4. Control Program Initialization (Part 2 of 3)



Extended Description	Routine	Label
7. If there is not a buffer pool prefix address in the SCD indicating the buffers have already been allocated and the number of subpools required is not zero, go allocate the buffers.	DLZCPI00	BFRALOC
8. If CICS/VS journaling is supported, DLZRDBL0 in the NUCLODTB is replaced with DLZRDBL1. Any module already loaded by SLCLOAD will not be reloaded. Write message DLZ055I if a DL/I facility module is not found.		DLILOAD
9. SCDDBLNT is loaded with the entry point at the proper logger initialization routine (DLZRDBL0 or DLZRDBL1). If logging was suppressed SCDDBLNT will continue to point to a branch 14 for immediate return to caller.		NUCLODNX

Extended Description	Routine	Label

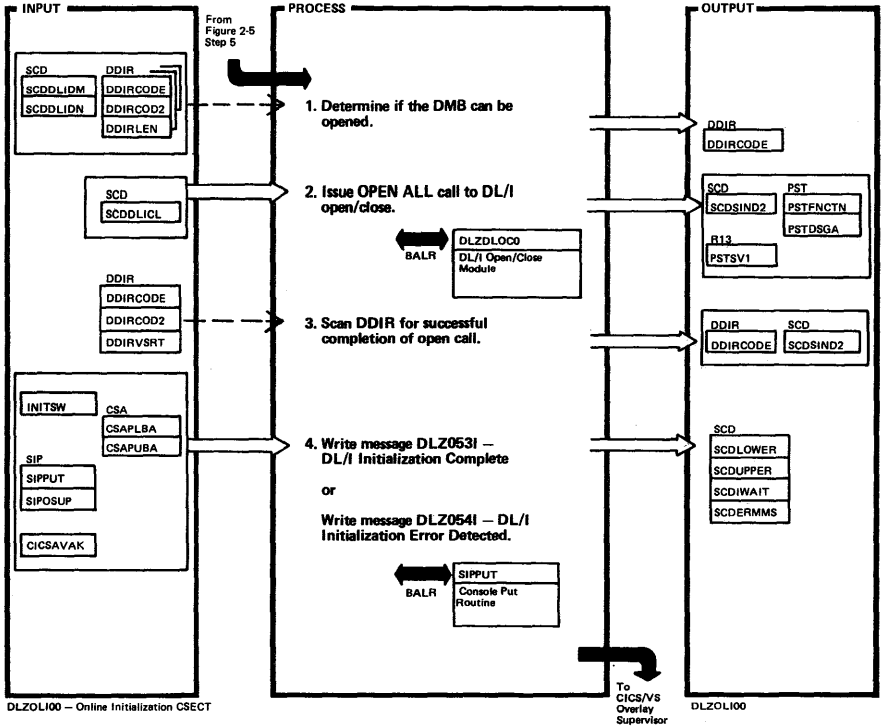
Figure 2-5.4. Control Program Initialization (Part 3 of 3)



Extended Description	Routine	Label
10. Write message DLZ0061 if the asynchronous logger did not successfully attach and go close the log. The address list for the asynchronous portion of the database logger and its save area address are located in the database log load module just prior to the entry point. If the attach fails, the database log is closed and the system continues without log support	DLZCPI00	NUCLODNX

Extended Description	Routine	Label

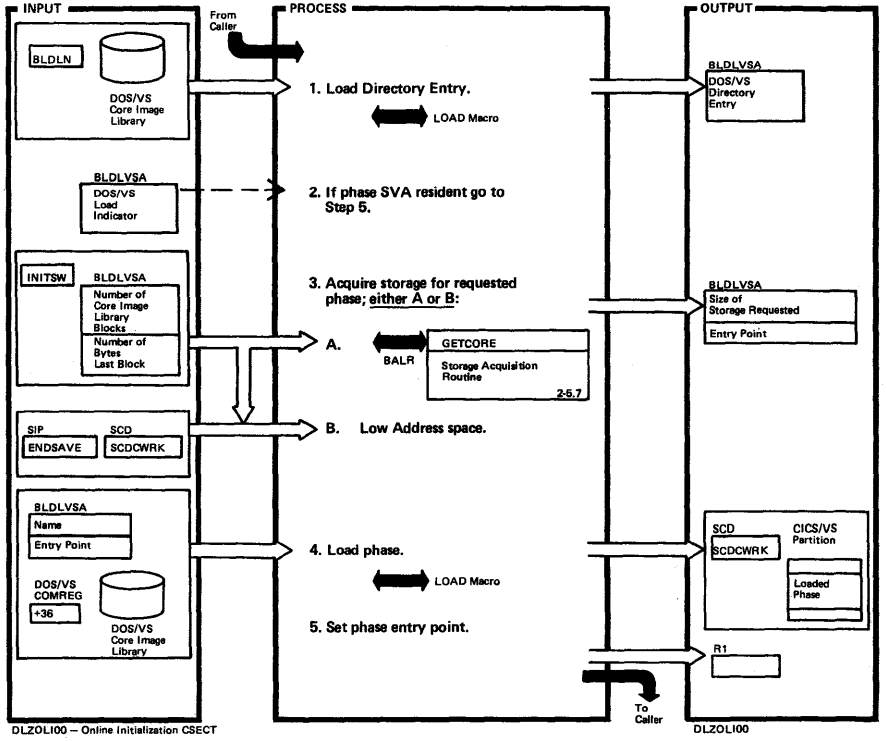
Figure 2-5.5. DMB Open Processing and Online Initialization Completion



Extended Description	Routine	Label
1. If all PSBs are remote, do not attempt OPEN.	DMBOPENA	DMBOPENA
2.		DMBSCNX
3. Write message DLZ057I if an open error occurred attempting to open a DMB.		DMBSCLP2
4. During the course of initialization an error can also cause a direct return to CICS/VS with message DLZ052I - Initialization Failed.		EXITOVL

Extended Description	Routine	Label

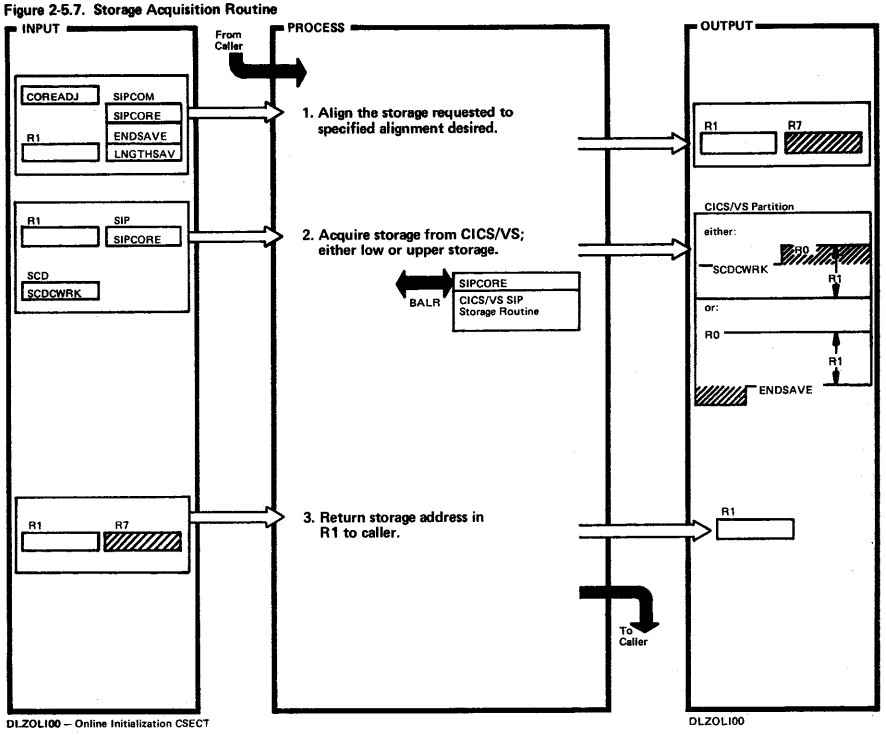
Figure 2-5.6. Module Load Routine.



Extended Description	Routine	Label
1. Caller passes requested phase name in a work field BLDLN. The output of the load call is a DOS/VS directory entry at BLDLVSA.	INITLODR	INITLODR
2.		BLDLFND
3. Amount of storage is determined by: Number of Library Block x 1024 + Number of Bytes in Last Block. INITSW indicates whether caller wants low address space or CICS/VS to acquire the space. B. Write message DLZ058I if insufficient core to initialize DL/I.		

Extended Description	Routine	Label
4. If low address space is requested by the caller SCDCWRK is used as the load address. After the load macro the end address of the module will be in the DOS/VS COMREG. SCDCWRK is updated with this end address to show the new low end of free storage.		LODRLOD
5. The phase entry point is passed back to the caller in register 1.		LODROK

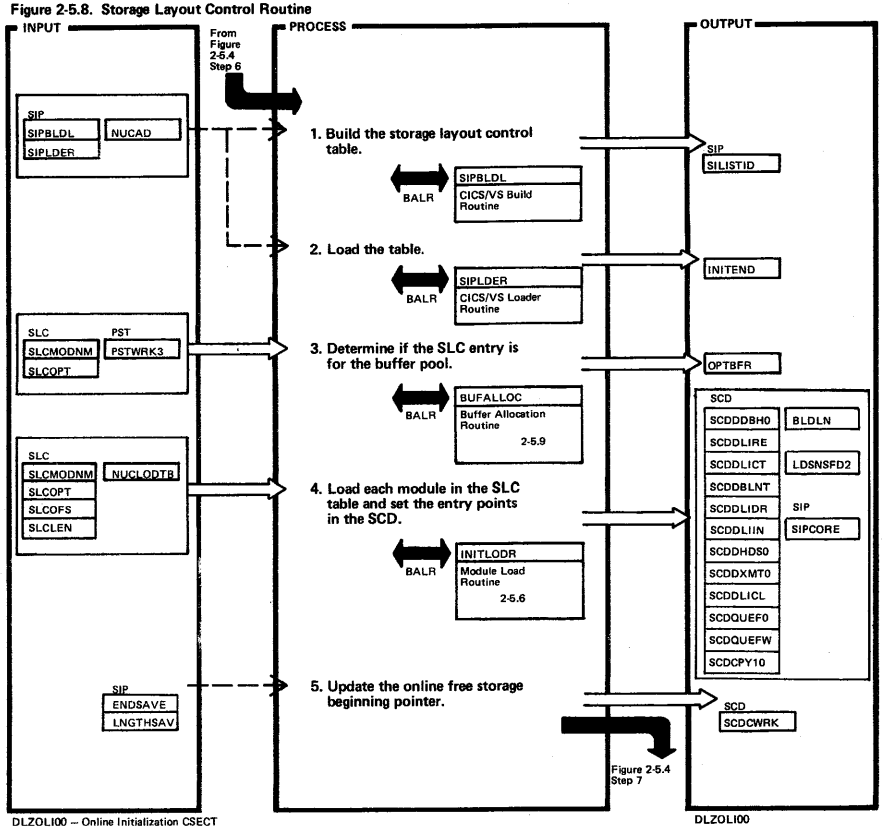
Figure 2-5.7. Storage Acquisition Routine



Extended Description	Routine	Label
1. If alignment is desired the number of additional bytes needed to align is calculated and put in R7. Register 1 is updated to show the new total number of bytes required.	GETCORE	GETCOREA GETCORE
2. Write message DLZ058I if insufficient storage to initialize DL/I. The SLCOPT flag byte (moved to SIPCORE from a storage layout control table entry) is used by CICS/VS GETMAIN routine. If we are doing our own alignment SLCOPT is examined to determine whether low or high storage was desired by the user.		BYCRALGN
3. The load point returned in R0 by Step 2 is adjusted by the additional bytes in R7 to get the requested alignment. This needs to be done only if low storage was acquired.		

Extended Description	Routine	Label

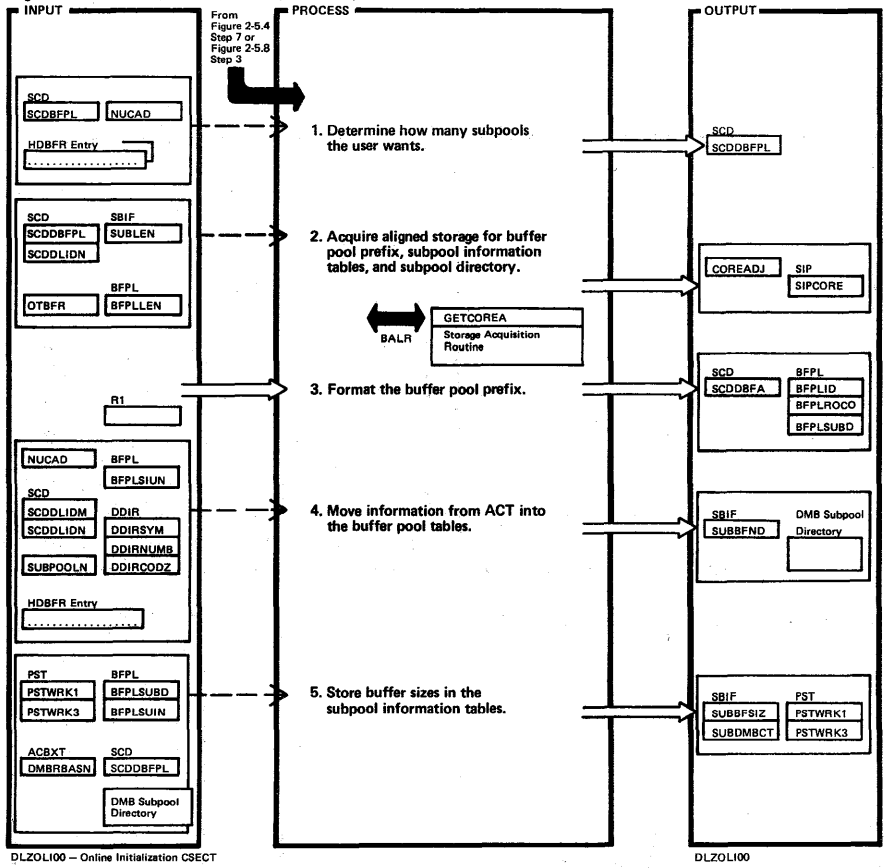
Figure 2-5.8. Storage Layout Control Routine



Extended Description	Routine	Label
1.	SLCLOAD	SLCLOAD
2. Write message DLZ030I if the loaded SLC table does not begin with *DLZSLC*. The table is loaded directly after module DLZOLI00.		
3. The user would have specified the DLZSLC statement with MODULE=BUFFER.		SLCLUP SLCBUF
4. As each module in the SLC table is loaded, the need to load flag is turned off so that upon return to Figure 2-5.4 these modules will not be reloaded. The SLCOPT for each module, as it is loaded, is moved to SIPCORE for use by the GETCORE routine.		SLCLUP

Extended Description	Routine	Label
5.		SLCXIT

Figure 2-5.9. Buffer Allocation Routine (Part 1 of 2)



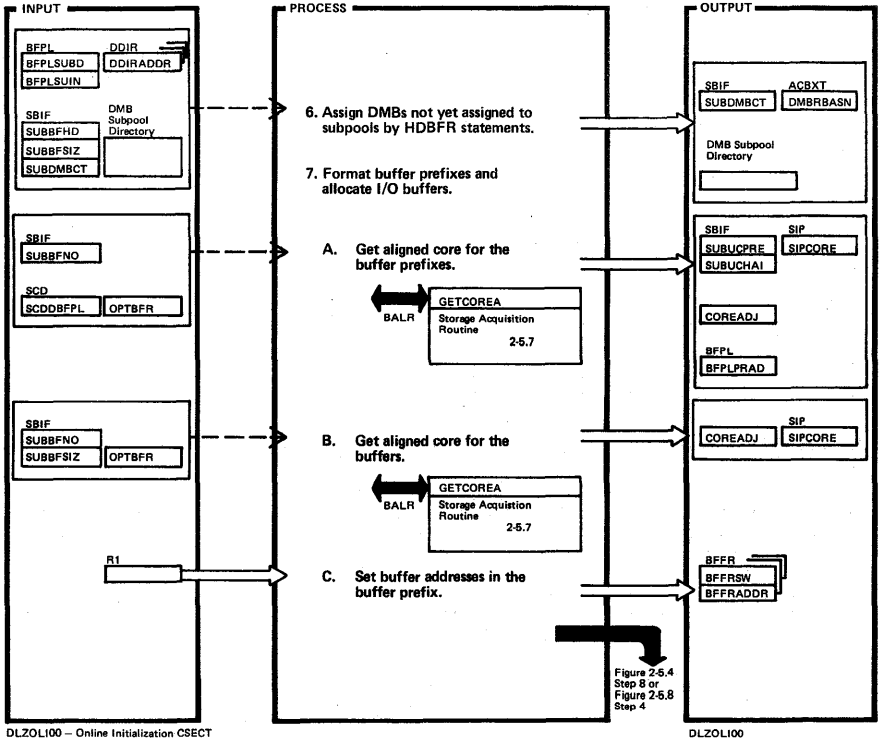
DLZOLI00 – Online Initialization CSECT

DLZOLI00

Extended Description	Routine	Label
1. Buffer allocation is done by this subroutine. The required number is set to the user specified total in the DLI parameter if the user number is smaller than required. Write message DLZ060I followed by DLZ061A if buffer pool allocation is missing or invalid.	BUFALLOC	BUFALLOC
2.		BFPREADY
4. Write message DLZ029I if there is an invalid DBDNAME in HDBFR entry.		SCANHD

Extended Description	Routine	Label
5. At this point the size of the subpools are determined. They are allocated, largest first, until the specified number is exhausted. Remaining DMBs requiring subpools are assigned evenly across all existing subpools. If the user specified more subpools than necessary, an additional pool of 512 buffer size is allocated for delete workspace. The subpool sizes are sorted so that the largest subpool appears first in the subpool information table.		PRPEND
		SUBTSHFL

Figure 2-5.9. Buffer Allocation Routine (Part 2 of 2)



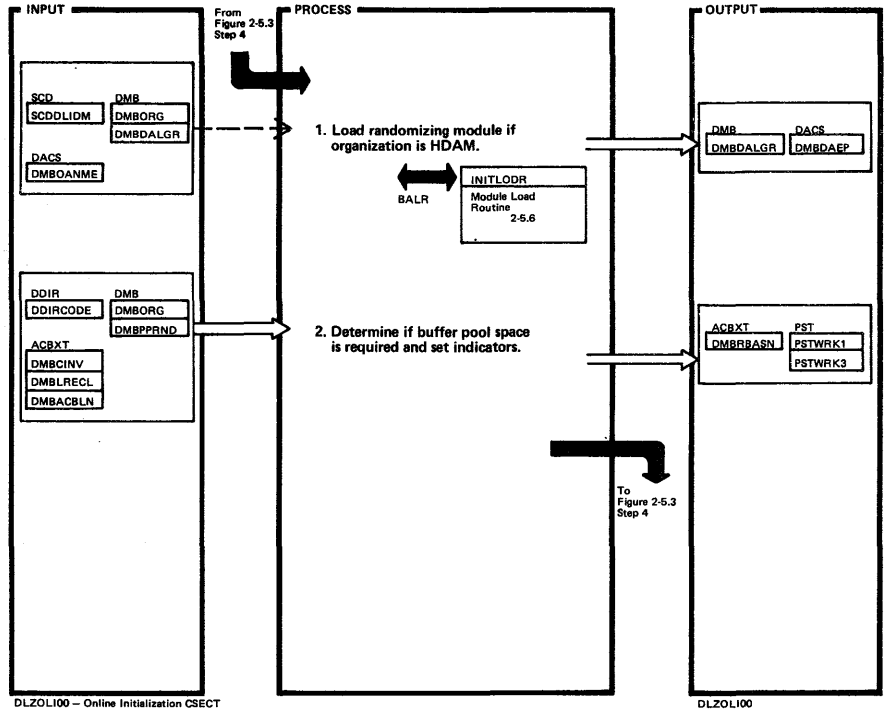
DLZOLI00 – Online Initialization CSECT

DLZOLI00

Extended Description	Routine	Label
6. Assign DMBs by corresponding control interval sizes. Each DMB is assigned by placing its DDIR position pointer into the subpool directory. It is possible to get message DLZ262I if there is a buffer allocation logic error.	BUFALLOC	GREATPRO
7. The user specified number of buffers is allocated per pool; default is 32.		BFRINIT
A.		ACCLOOP
B.		BFFRFRMT
C.		BFRSPLUP

Extended Description	Routine	Label

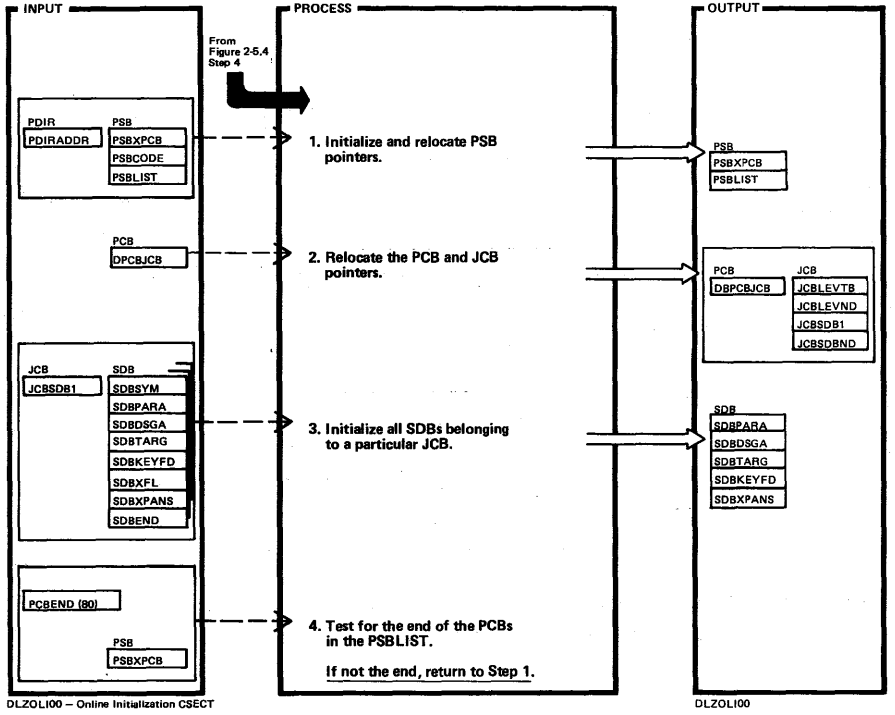
2-5.10. Build Associated DMB Control Blocks



Extended Description	Routine	Label
1. Before loading the randomizer a check is made with all currently loaded randomizers. If one with the same name as the one we are loading is found, the entry point is resolved and the actual load is bypassed. If the randomizing module is not found, the DDIR is updated to show DMB initialization failed.	DMBLOADR	DMBLOADR
2. If buffer pool space is required, the size of each control interval rounded to the next multiple of 512 is indicated in PSTWRK1 for later allocation of the buffer pool.		GETBUFRS

Extended Description	Routine	Label

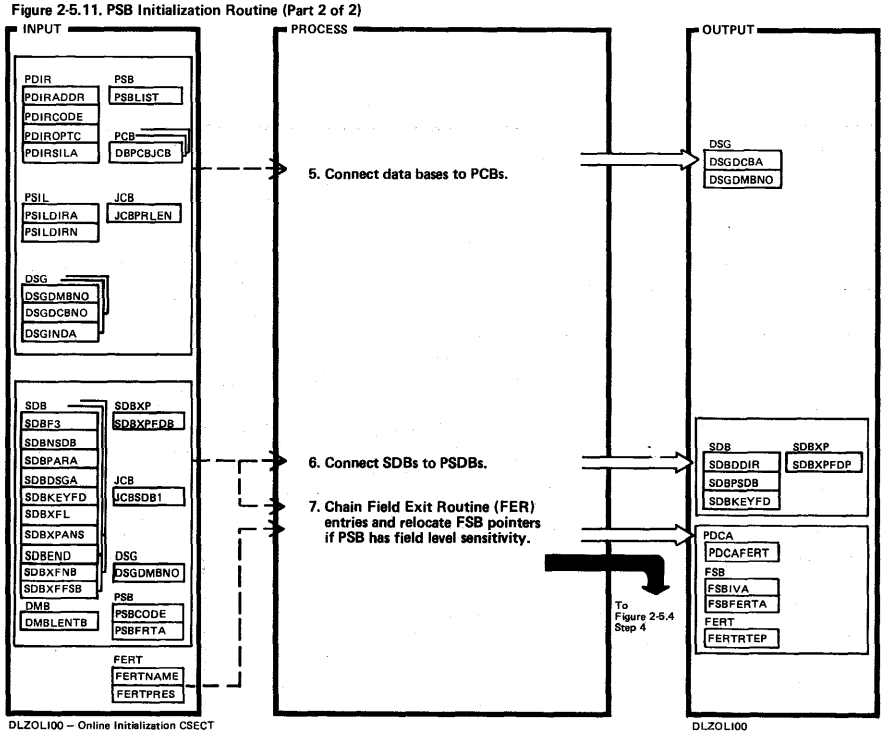
Figure 2-5.11. PSB Initialization Routine (Part 1 of 2)



Extended Description	Routine	Label
1.	PSBRELO	PSBRELO PCBRLLUP
2.		PCBPLIP
3.		SDBRELO
4. The pointer to the PSBLIST is bumped to the next PCB pointer entry and processing returns to Step 1 if we are not at the last PCB. If the index PCB exists and has not been relocated return to Step 2.		NPCBCK

Extended Description	Routine	Label

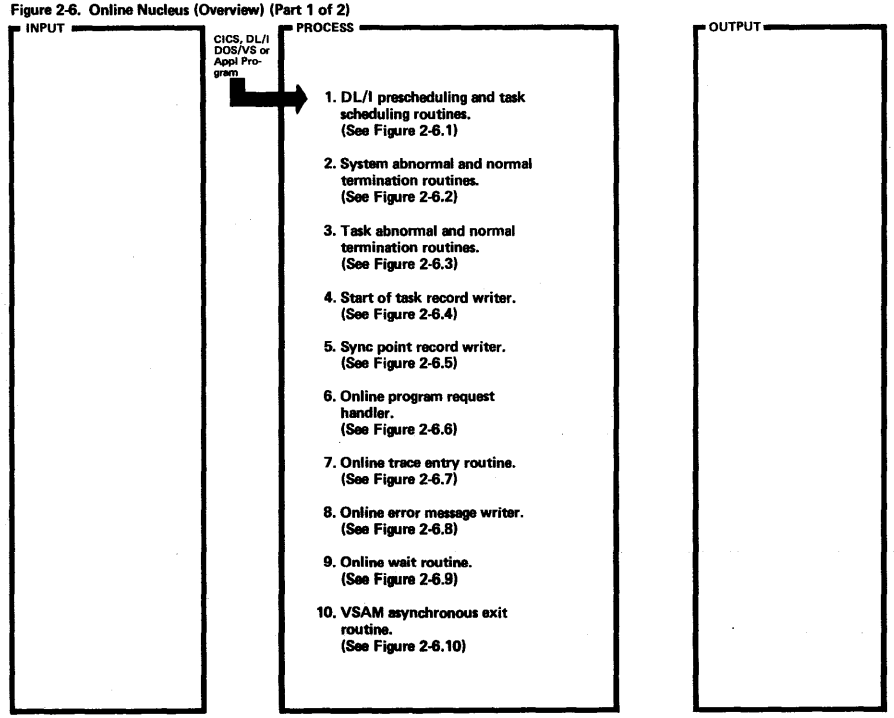
Figure 2-5.11. PSB Initialization Routine (Part 2 of 2)



Extended Description	Routine	Label
5. For each DSG belonging to a PCB, the offset to the corresponding PSIL is found by the offset in DSGDMBNO. The DMB number at the corresponding DMB to this PSIL is then moved to DSGDMBNO. Thus, data bases specifically used for this PCB are connected.	PSBRELO	PCBROUT
7. Return to Step 4 to process the next PCB in the PSBLIST when there are no more SDBs.		CONSDBS

Extended Description	Routine	Label

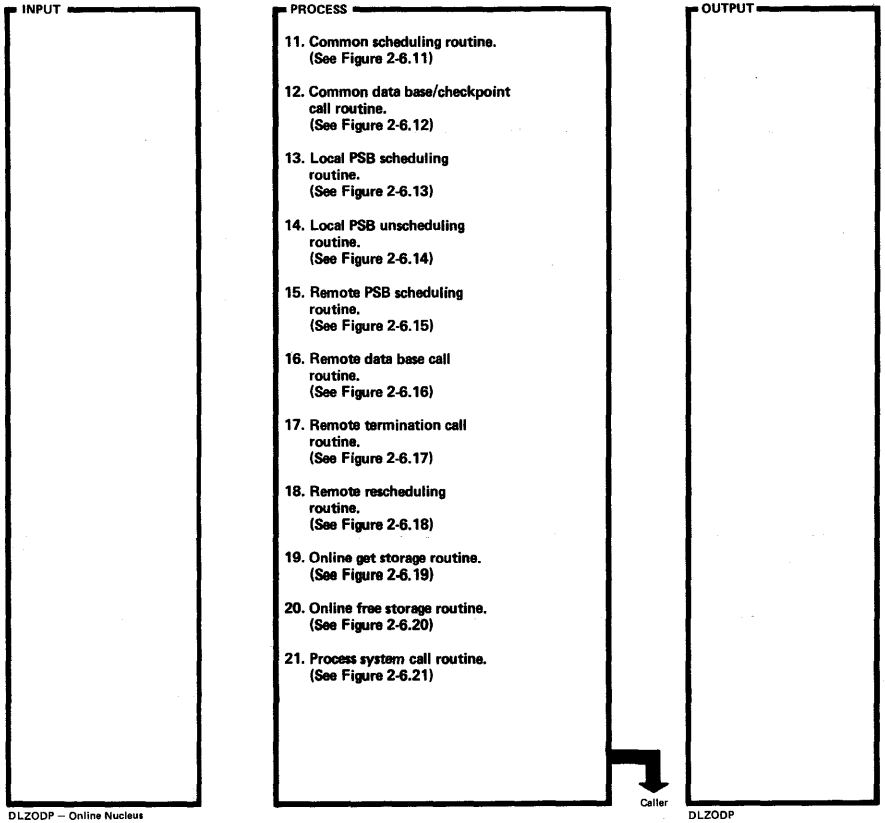
Figure 2-6. Online Nucleus (Overview) (Part 1 of 2)



Extended Description	Routine	Label
1.	DLZODP00	
2.	DLZODP03 DLZODP02	
3.	DLZODP07 DLZODP06 DLZODP01	
4.	DLZODP04	
5.	DLZODP05	
6.	DLZPRH00	
7.	DLZOLT00 DLZOLT01 DLZOLT02	
8.	DLZERMSG	
9.	DLZOWAIT	
10.	DLZOVSEX	

Extended Description	Routine	Label

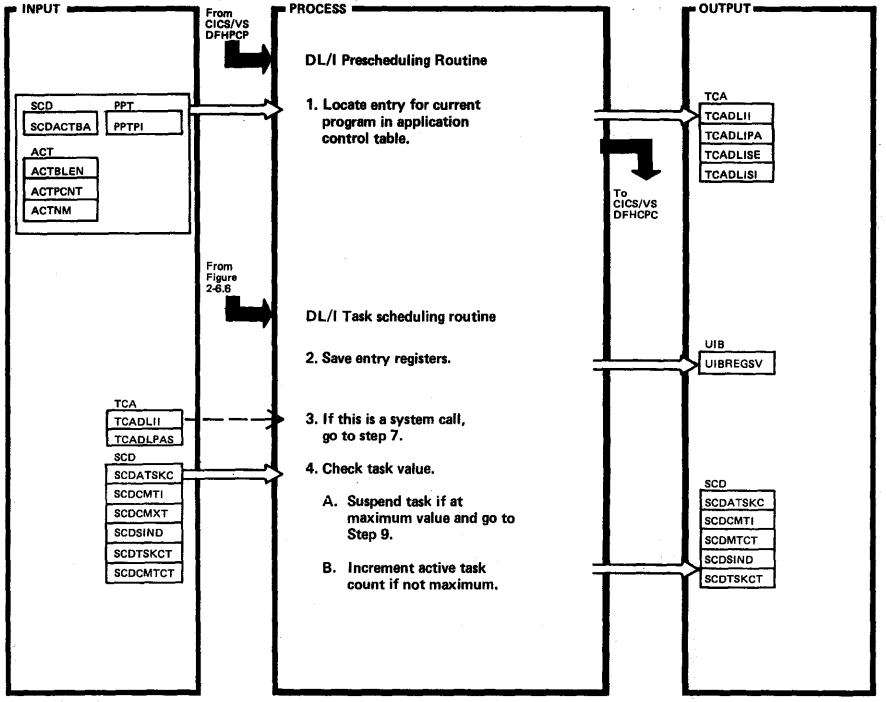
Figure 2-6. Online Nucleus (Overview) (Part 2 of 2)



Extended Description	Routine	Label
11.	DLZCOM00	
12.	DLZCOM01	
13.	DLZLOC00	
14.	DLZLOC01	
15.	DLZISC00	
16.	DLZISC01	
17.	DLZISC02	
18.	DLZISC03	
19.	DLZODP10	
20.	DLZODP11	

Extended Description	Routine	Label
21.	PROCSYS	

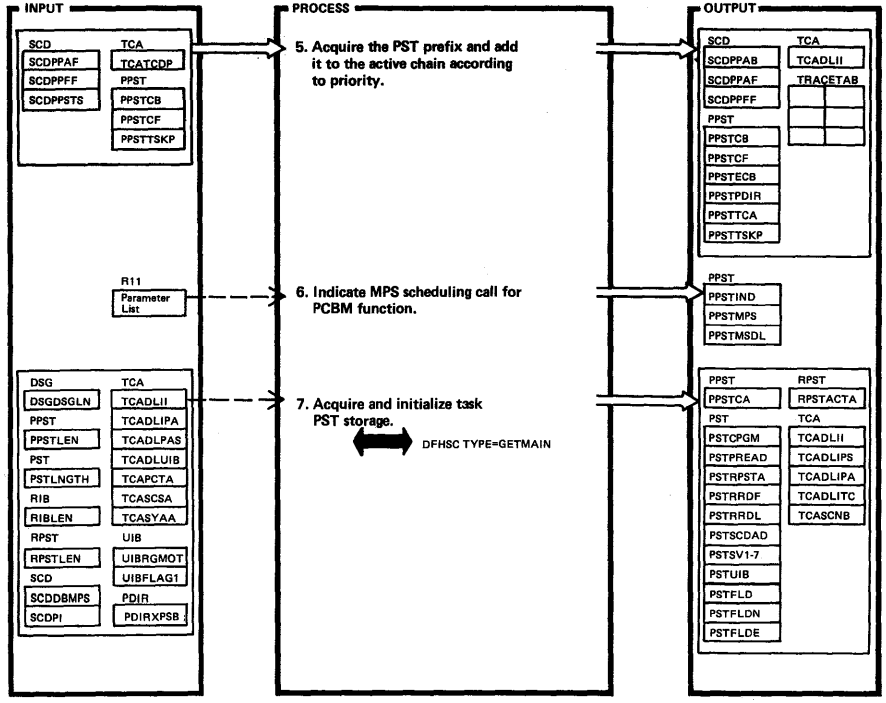
Figure 2-6.1. DL/I Prescheduling and Task Scheduling Routines (Part 1 of 3)



Extended Description	Routine	Label
1. Nucleus identifier (*DLZNUCXX*0160) and module identifier (DLZODPvrnp) are defined here. The level format is vrnp; where 'v' is the version, 'r' is the release, 'n' is an additional identification number, and 'p' is the latest PTF that has been applied. This step checks the authorization of the CICS application program to use DL/I. If the program named is not located in the ACT an error indicator is turned on in the TCA (TCADLISE). If the ACT search is successful the ACT entry address is placed in the system TCA and DL/I scheduling initiated indicator (TCADLISI) is turned on.	DLZODP	DLZODP DLZODP00
2.	DLZODP	DLZSCHDL

Extended Description	Routine	Label

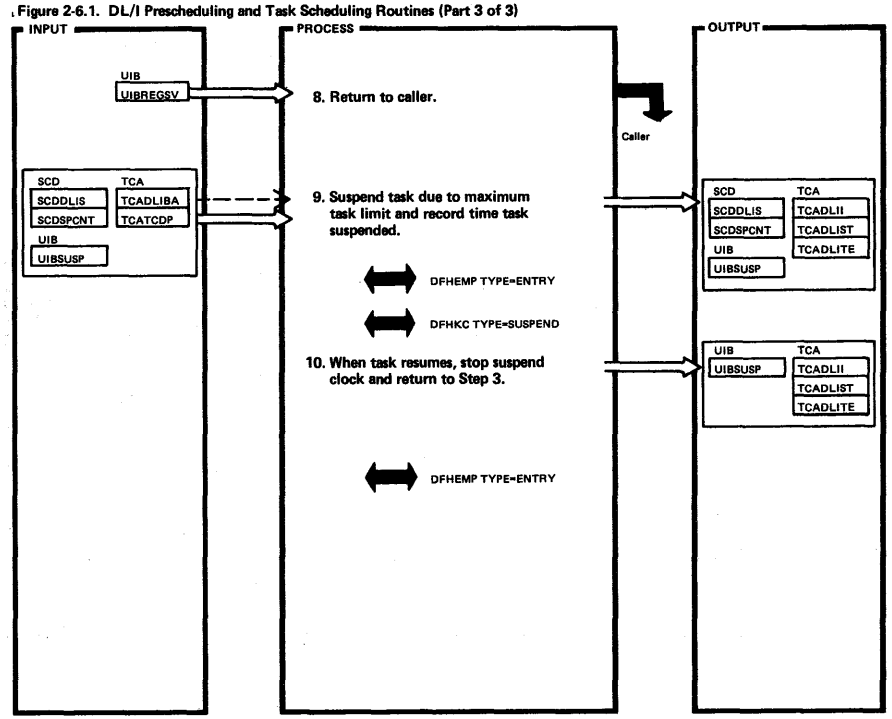
Figure 2-6.1. DL/I Prescheduling and Task Scheduling Routines (Part 2 of 3)



Extended Description	Routine	Label
5.		TASKPPST
7. Storage acquired for PST and FLD control blocks. Acquires storage for RIB and RPST if UIBREMOT=1.		TASKPSTT

Extended Description	Routine	Label

Figure 2-6.1. DL/I Prescheduling and Task Scheduling Routines (Part 3 of 3)



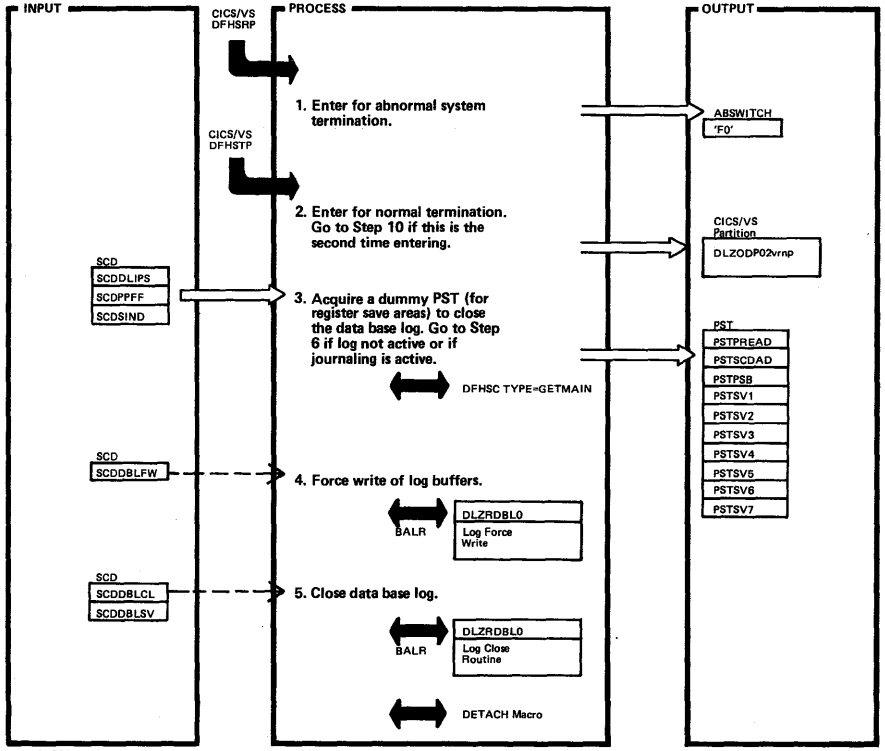
DLZODP00 – DL/I Prescheduling and Task Scheduling Routines

DLZODP00

Extended Description	Routine	Label
9.		TASKSUSP

Extended Description	Routine	Label

Figure 2-6.2. System Abnormal and Normal Termination (Part 1 of 2)



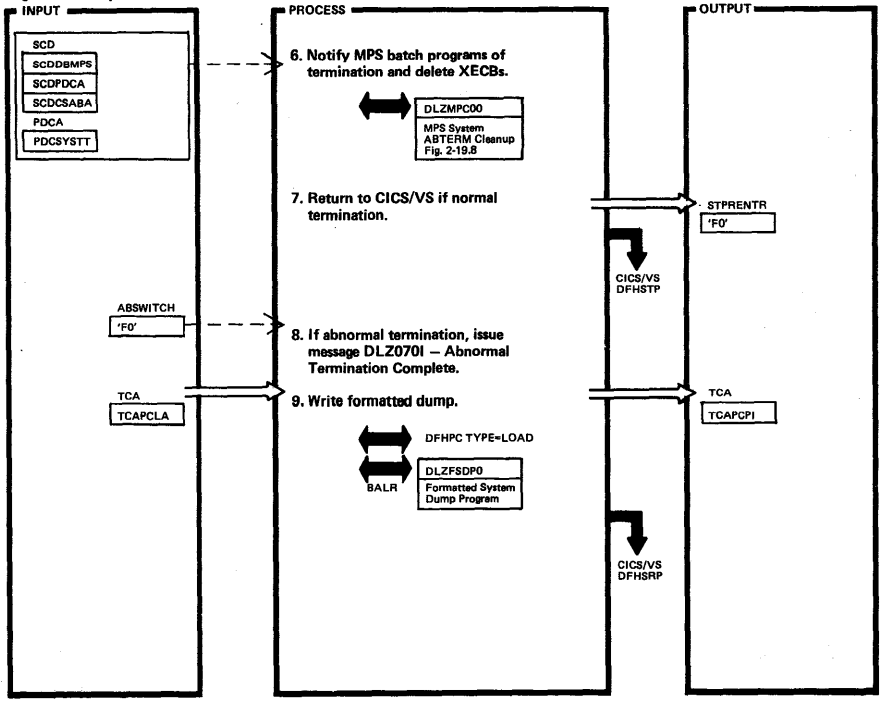
DLZODP02 – System Abnormal and Normal Termination CSECT

DLZODP

Extended Description	Routine	Label
1. Entry is made from CICS/VS System Recovery Program on abnormal termination. Return is back to DFHSRP.	DLZODP03	DLZODP03
2. Routine identifier (DLZODP02vrnp) is defined here. For normal termination DFHSTP enters this routine twice.	DLZODP02	DLZODP02
3. Issue message DLZ067I if there is insufficient storage to terminate DL/I.		STPRENTR
4.		STPSVCH1

Extended Description	Routine	Label

Figure 2-6.2. System Abnormal and Normal Termination (Part 2 of 2)



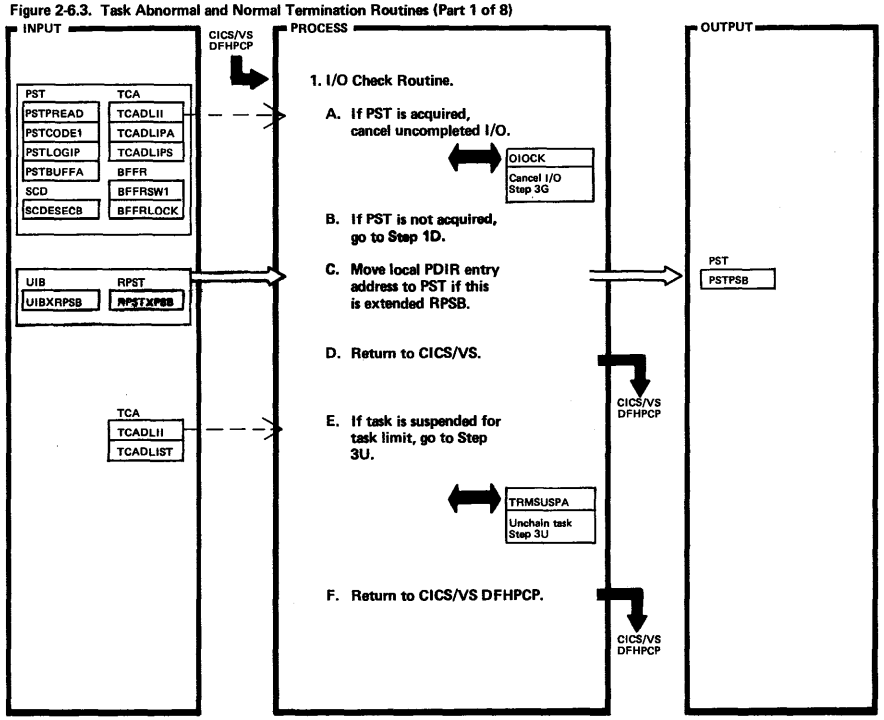
DLZODP02 – System Abnormal and Normal Termination CSECT

DLZODP

Extended Description	Routine	Label
6. MPS batch programs may be active or waiting for online MPS processing and are therefore notified if CICS/VS terminates. Online XECBs defined for MPS are also deleted.	DLZODP02	STPEXLOG
7.		STPNOXCB
8.		ABTERM
9.		STPEXIT3

Extended Description	Routine	Label

Figure 2-6.3. Task Abnormal and Normal Termination Routines (Part 1 of 8)



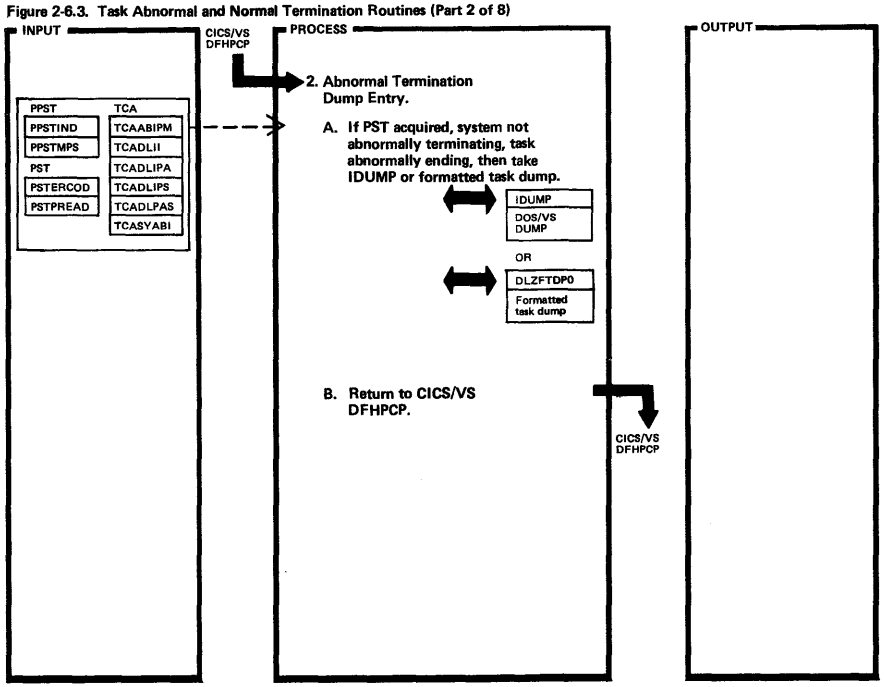
DLZODPnn – Task Abnormal and Normal Termination Routines

DLZODPnn

Extended Description	Routine	Label
1. Post logger ECB if logger was doing I/O for this task when ABEND occurred.	DLZODP07	
2.		DODP07A

Extended Description	Routine	Label

Figure 2-6.3. Task Abnormal and Normal Termination Routines (Part 2 of 8)



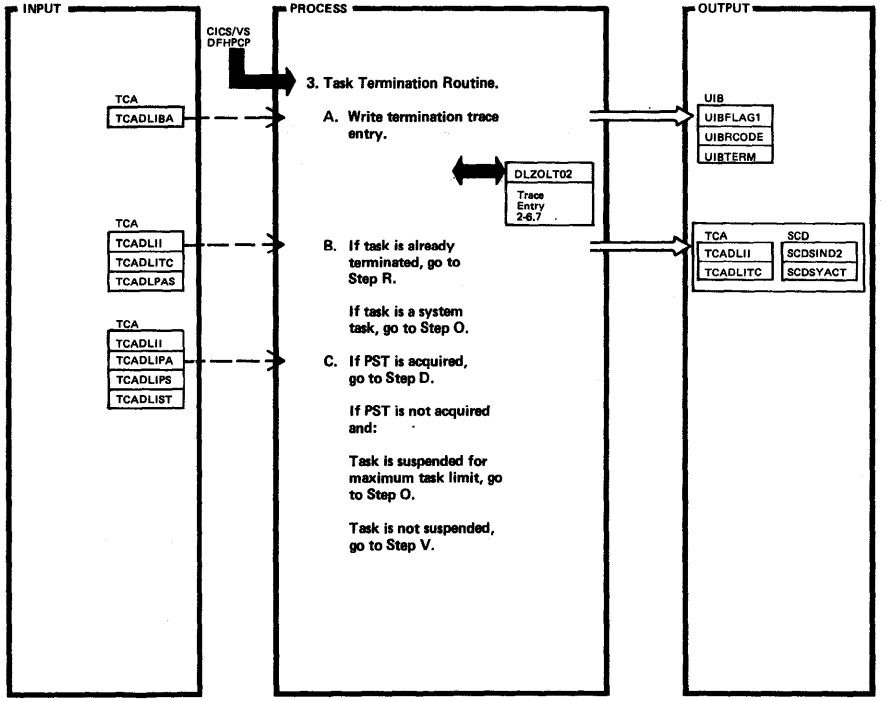
DLZODPnn – Task Abnormal and Normal Termination Routines

DLZODPnn

Extended Description	Routine	Label
2. No formatted dump is produced in case of missing PST or insufficient storage available. If SYSDUMP=YES was specified for the DOS/VS partition, an IDUMP is taken instead of the formatted dump. DL/I system ABEND is reduced to task ABENDs. In case of DL/I system ABEND, all DL/I tasks are abended by DL/I. For each task, DLZFTDP0 is called. DLZFTDP0 uses the CICS dump macro DFHDC, that dumps DL/I blocks on the CICS dump data set. To get the dump on printer, use offline CICS program DFHDUP.	DLZODP06	

Extended Description	Routine	Label

Figure 2-6.3. Task abnormal and Normal Termination Routines (Part 3 of 8)



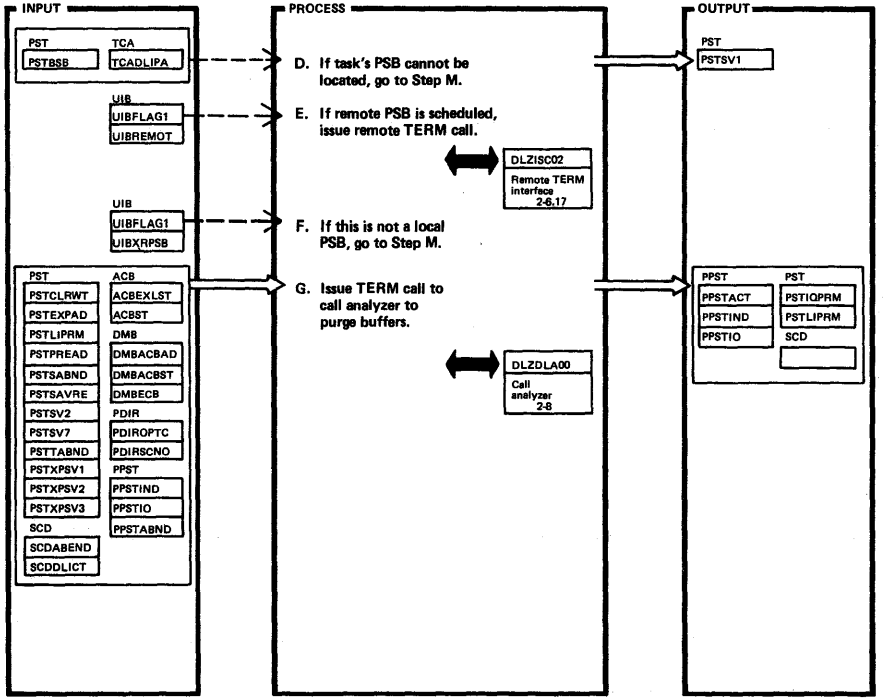
DLZODPnn – Task Abnormal and Normal Termination Routines

DLZODPnn

Extended Description	Routine	Label
3A. Routine identifier (DLZODP01vrmp) is defined here. This is the entry point for CICS/VS if the PCP termination exit indicator (TCADLITE) is on and the task is about to be detached. This is the entry point for the Program Request Handler for a TERM or a T call. If trace is enabled, a task termination entry with an ID=X'F8' is made showing why termination was requested, and the DL/I status.	DLZODP01	DLZODP01 DLZTKTRM DLZOLT02
3C.		NOSYSTSK

Extended Description	Routine	Label

Figure 2-6.3. Task Abnormal and Normal Termination Routines (Part 4 of 8)



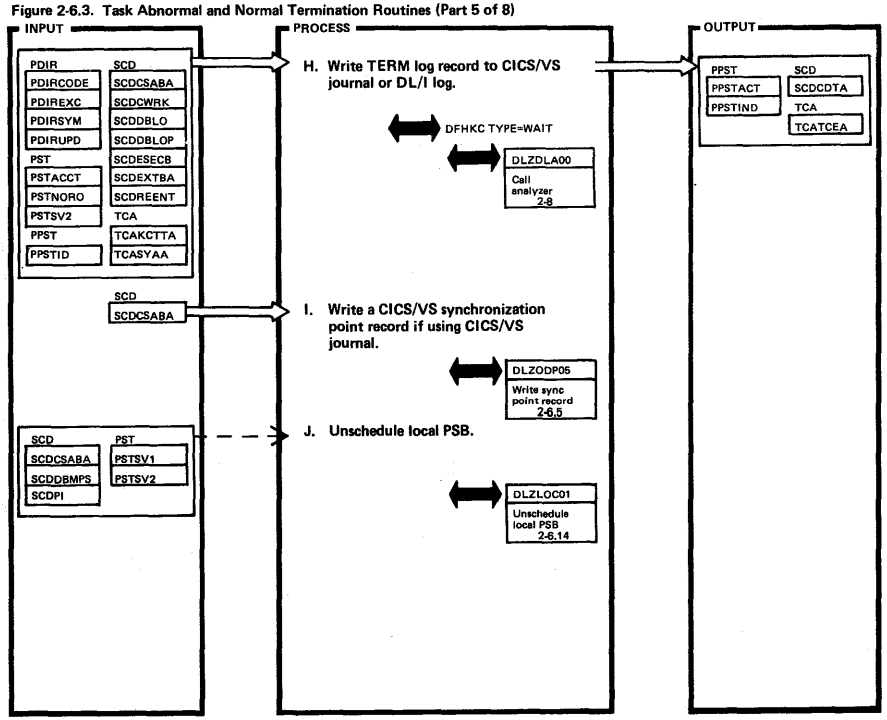
DLZODP01 – Task Abnormal and Normal Termination Routine

DLZODP01

Extended Description	Routine	Label
3D.		TERMWPST
3G.		PURGBUF OIOCK

Extended Description	Routine	Label

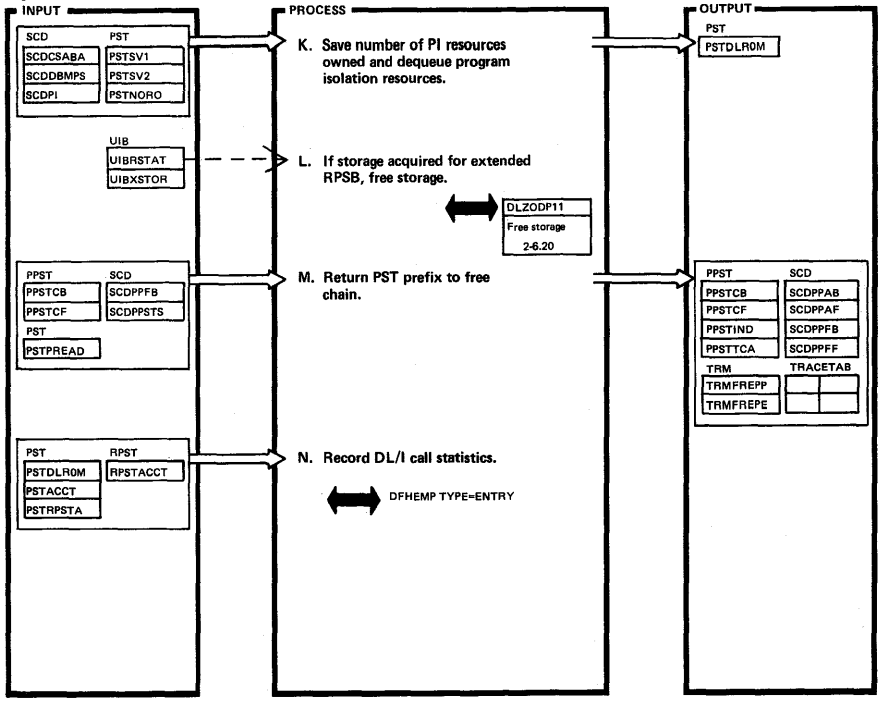
Figure 2-6.3. Task Abnormal and Normal Termination Routines (Part 5 of 8)



Extended Description	Routine	Label
3H. A wait may be done at this time for the logger if it is not available.		
3I.		TRMLGBY

Extended Description	Routine	Label

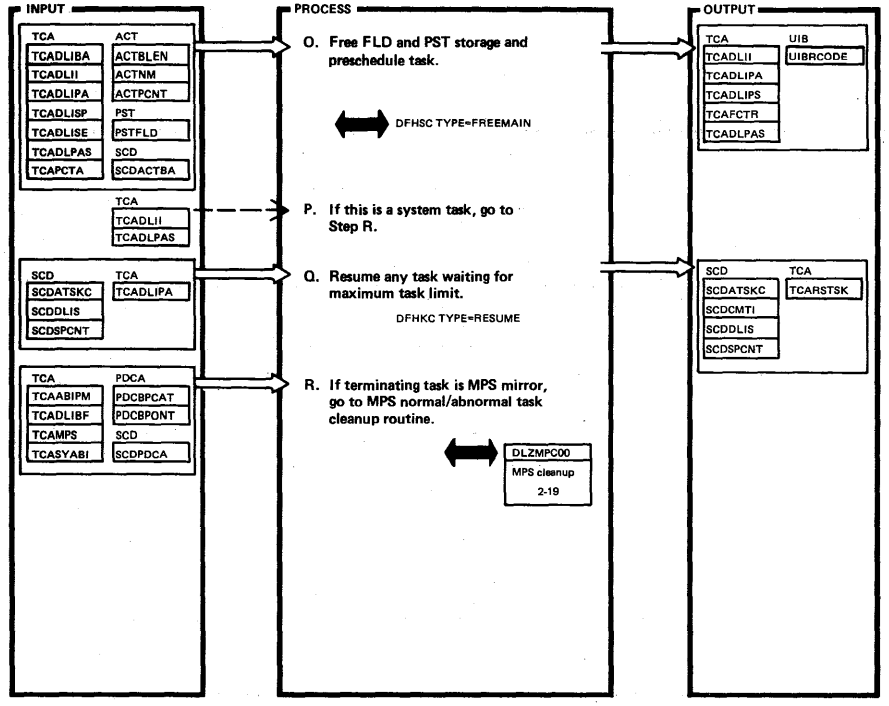
Figure 2-6.3. Task Abnormal and Normal Termination Routines (Part 6 of 8)



Extended Description	Routine	Label
3K. Number of PI resources owned are saved for CMF Hook statistics.		TRMNOTBP
3L.		NOPIDEQ

Extended Description	Routine	Label

Figure 2-6.3. Task Abnormal and Normal Termination Routines (Part 7 of 8)



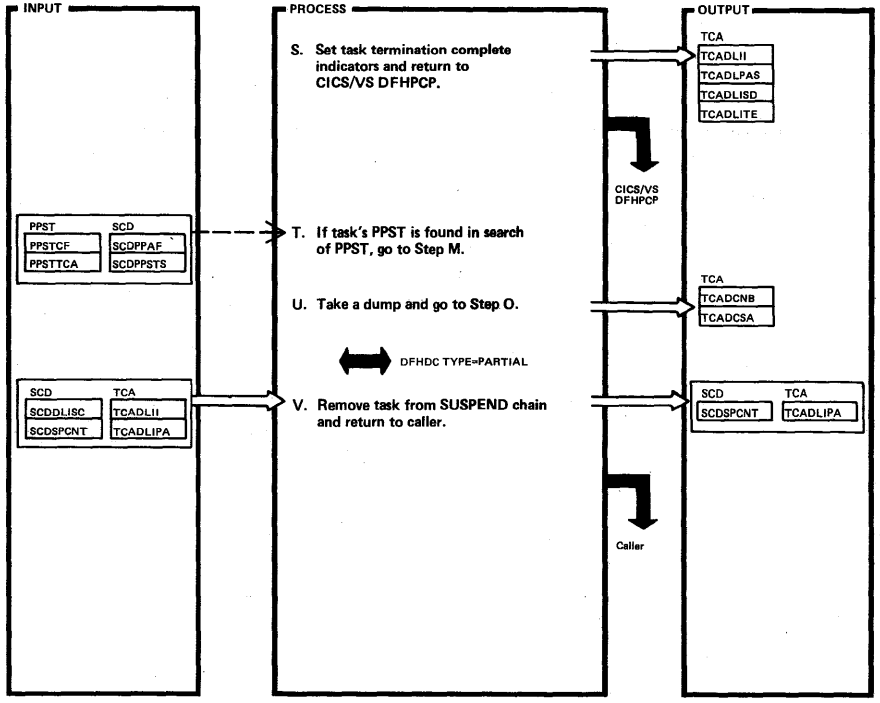
DLZODP01 – Task Abnormal and Normal Termination Routines

DLZODP01

Extended Description	Routine	Label
3O.		FREEPST TRMSCHD
3R.		TRMEXIT

Extended Description	Routine	Label

Figure 2-6.3. Task Abnormal and Normal Termination Routines (Part 8 of 8)



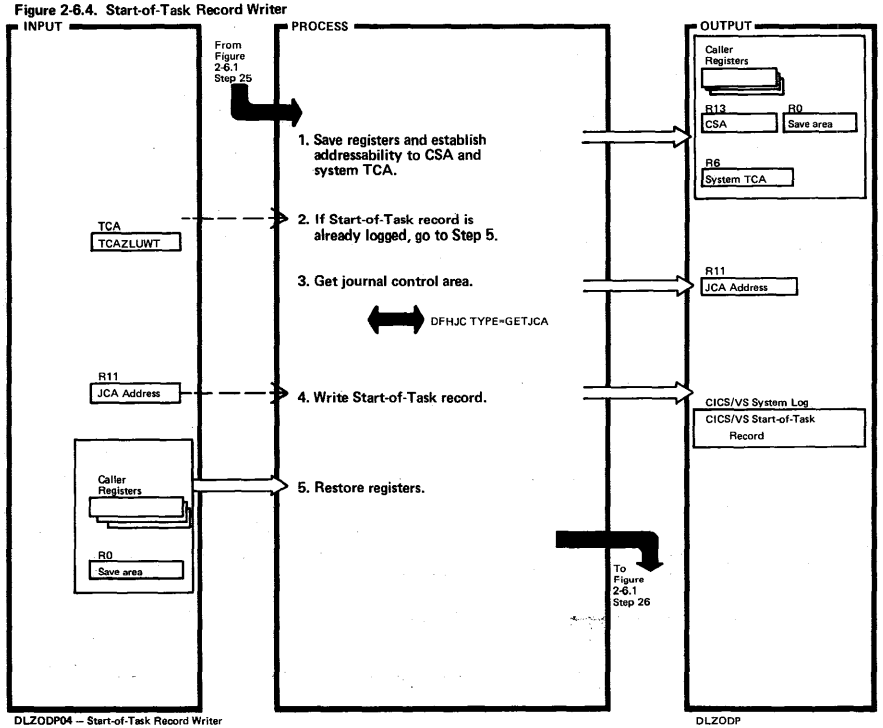
DLZODP01 – Task Abnormal and Normal Termination Routines

DLZODP01

Extended Description	Routine	Label
3T.		TRMFREPO
3V.		TRMSUSPA

Extended Description	Routine	Label

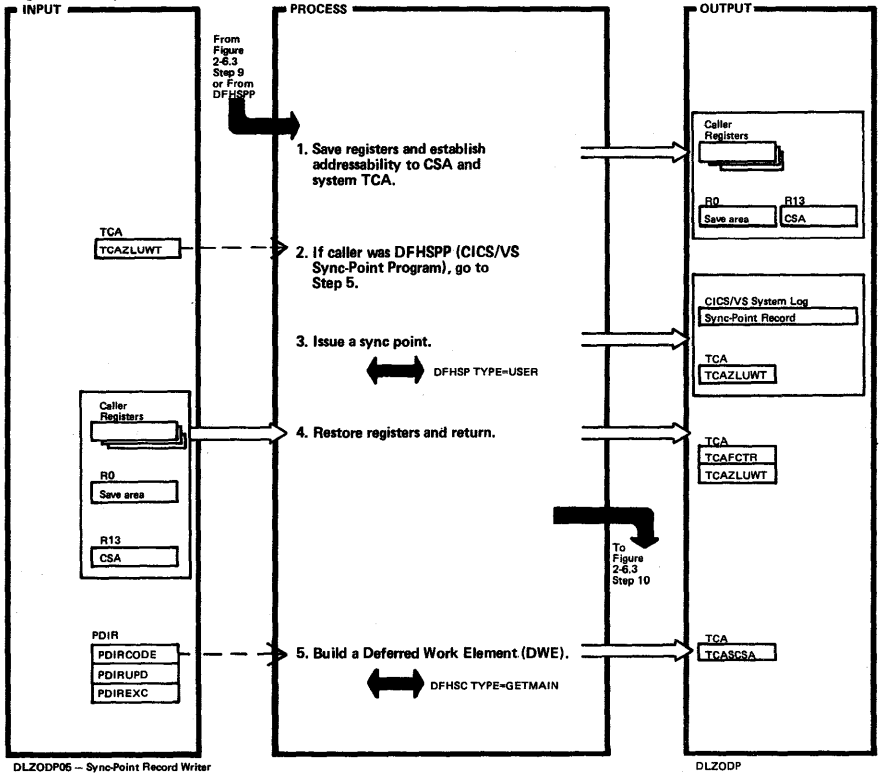
Figure 2-6.4. Start-of-Task Record Writer



Extended Description	Routine	Label
1. Routine identifier DLZODP04vrmp is defined here.	DLZODP04	DLZODP04
3. Acquire task's JCA and establish JCA addressability.		
4. Dummy record is written to system log and CICS Start-of-Task indicator is set in it to mark start-of-task on system log.		
5. Restore save area address in register 13 and reload registers 14 through 12 from the save area.		DLXSCRET

Extended Description	Routine	Label

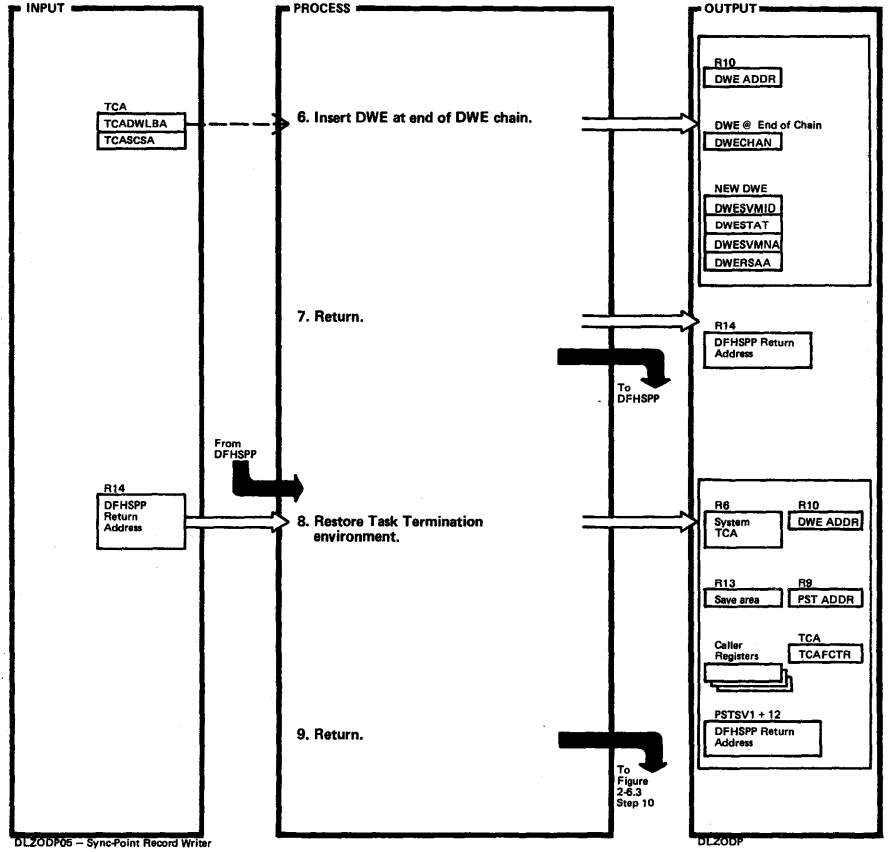
Figure 2-6.5. Sync-Point Record Writer (Part 1 of 2)



Extended Description	Routine	Label
1. Routine Identifier DLZODP05vrnp is defined here.	DLZODP05	DLZODP05
5. DWE is not built for read-only PSB.		DLXACDWE

Extended Description	Routine	Label

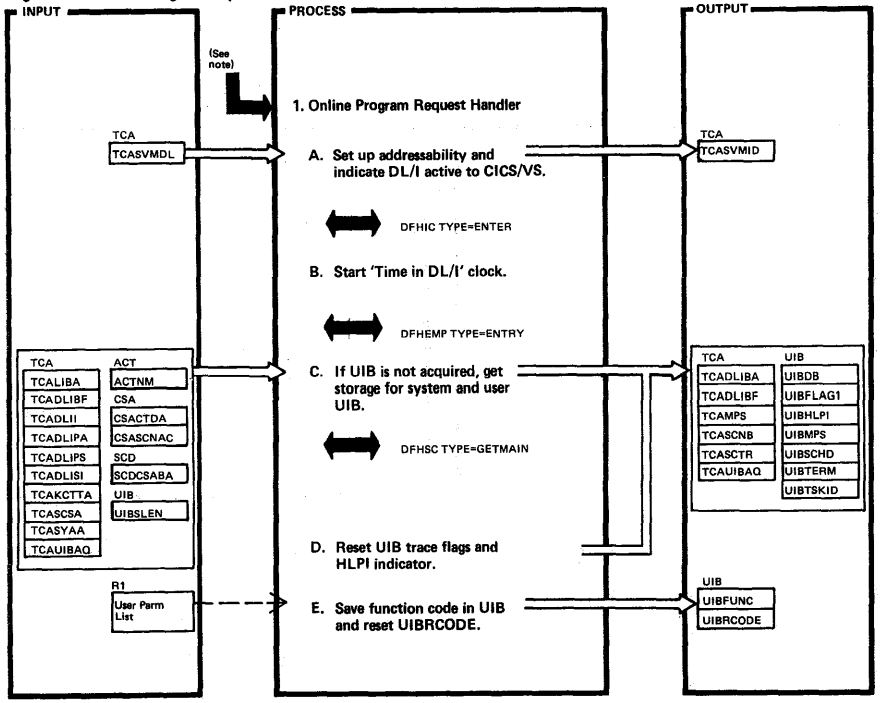
Figure 2-6.5. Sync-Point Record Writer (Part 2 of 2)



Extended Description	Routine	Label
6. Store address of DWE entry point of DLZODP05 in DWE for deferred processing by CICS/VS.	DLZODP05	DLXDWECN
8. Set up registers and PST save area to complete task termination and then return to DFHSPP.		DLXTDWEN

Extended Description	Routine	Label

Figure 2-6.6. Online Program Request Handler (Part 1 of 7)



NOTE: DLZPRHO0 is entered from the Language Interface Module, DLZEIPO0, DLZBPCO0, or DFHMIR.

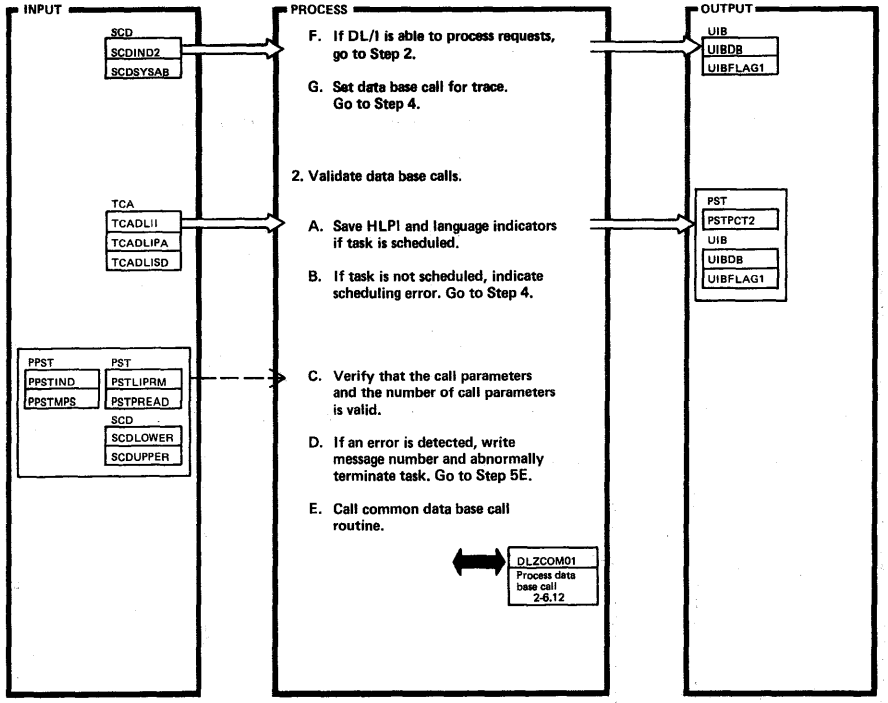
DLZPRHO0 — Online Program Request Handler

DLZPRHO0

Extended Description	Routine	Label
1. Module identifier (DLZPRHO0vrnp) is defined here. The level format is vrnp, where 'v' is the version, 'r' is the release, 'n' is an additional identification number, and 'p' is the latest PTF number that has been applied.	DLZPRHO0	DLZPRHO0

Extended Description	Routine	Label

Figure 2-6.6. Online Program Request Handler (Part 2 of 7)



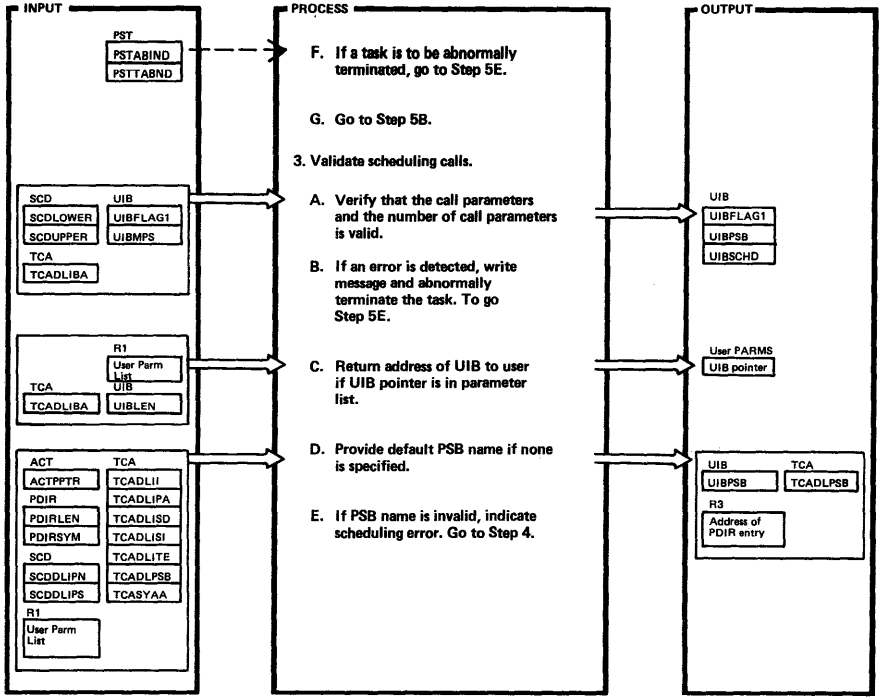
DLZPRHO0 — Online Program Request Handler

DLZPRHO0

Extended Description	Routine	Label
1F.		TESTFUNC
2A.		PRHDB0
2B. Error returned is X'0808' – DB call when not scheduled.		
2D. Write message DLZ2601 for invalid number of parameters or message DLZ2611 for invalid call parameter.		

Extended Description	Routine	Label

Figure 2-6.6. Online Program Request Handler (Part 3 of 7)



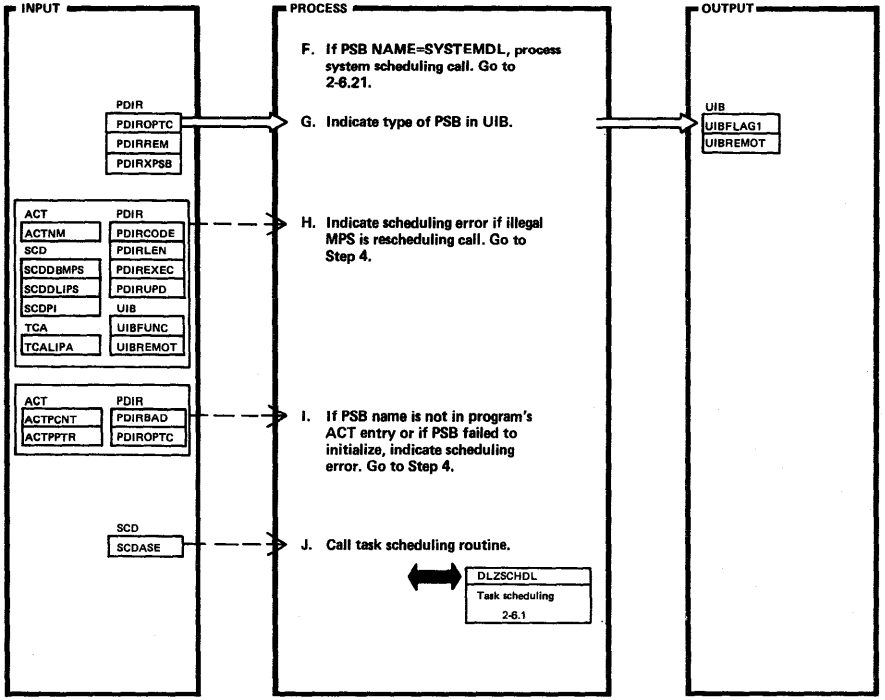
DLZPRHO0 – Online Program Request Handler

DLZPRHO0

Extended Description	Routine	Label
3A.		PRHSCHD0
3B. Write message DLZ260I for invalid number of parameters. Write message DLZ261I for invalid call parameters.		
3E. Error returned is: X'0801' – PSB not in PSB directory X'0802' – Task not prescheduled X'0803' – Task already scheduled X'0806' – PSB name too long or too short		

Extended Description	Routine	Label

Figure 2-6.6. Online Program Request Handler (Part 4 of 7)



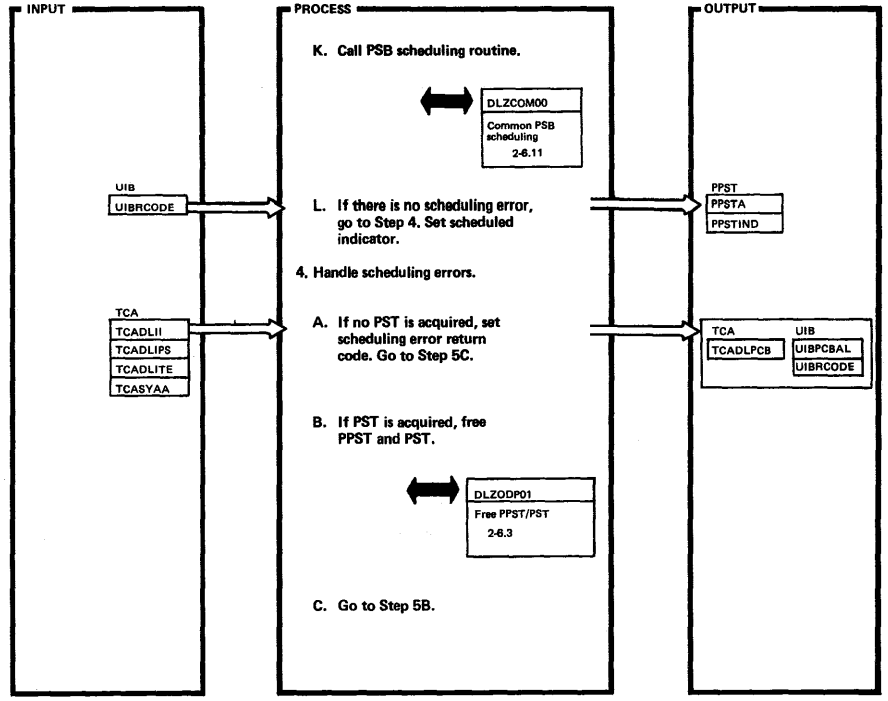
DLZPRHO0 – Online Program Request Handler

DLZPRHO0

Extended Description	Routine	Label
3H. Error returned is X'0809' – Illegal MPS scheduling call. 3I. Error returned is: X'0805' – PSB initialization failed, or X'0806' – PSB not authorized for program.		

Extended Description	Routine	Label

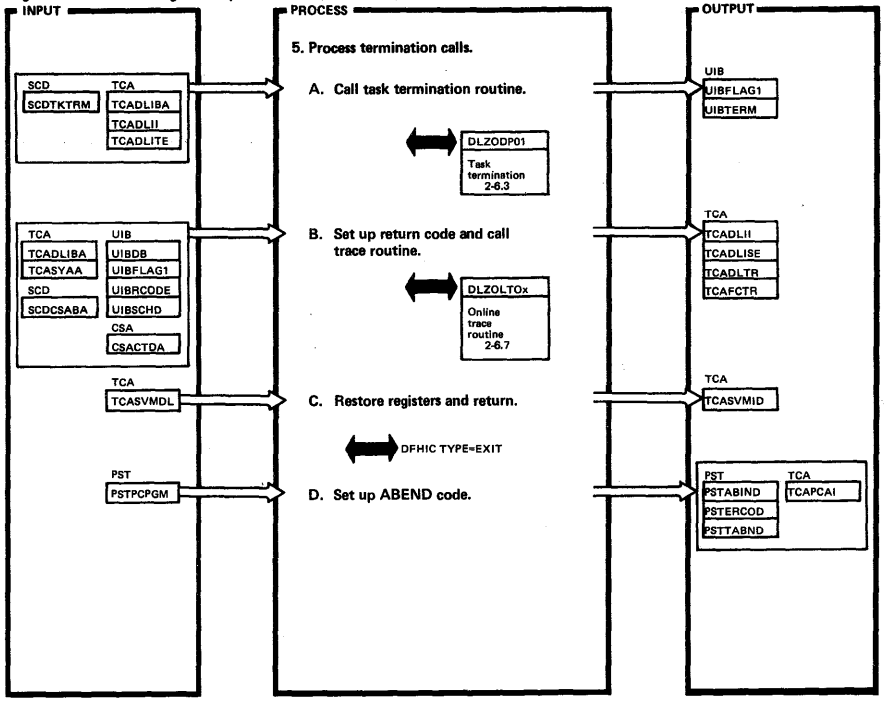
Figure 2-6.6. Online Program Request Handler (Part 5 of 7)



Extended Description	Routine	Label
4A.		DFRSERR8 DFRSERRS DFRSERRB

Extended Description	Routine	Label

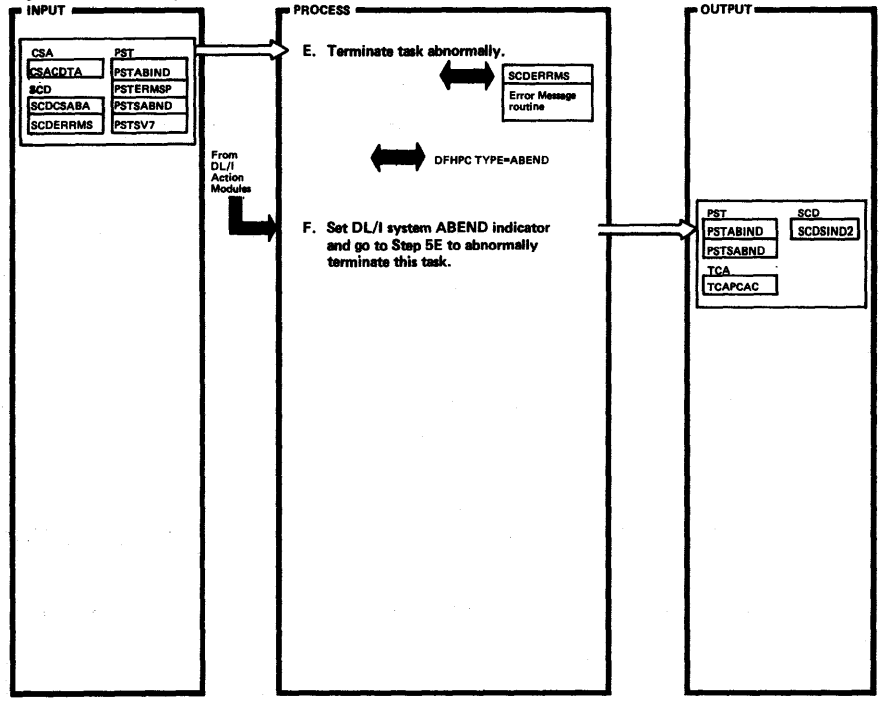
Figure 2-6.6. Online Program Request Handler (Part 6 of 7)



Extended Description	Routine	Label
5C.		RETEXIT
5D.		ECICNTER

Extended Description	Routine	Label

Figure 2-6.6. Online Program Request Handler (Part 7 of 7)



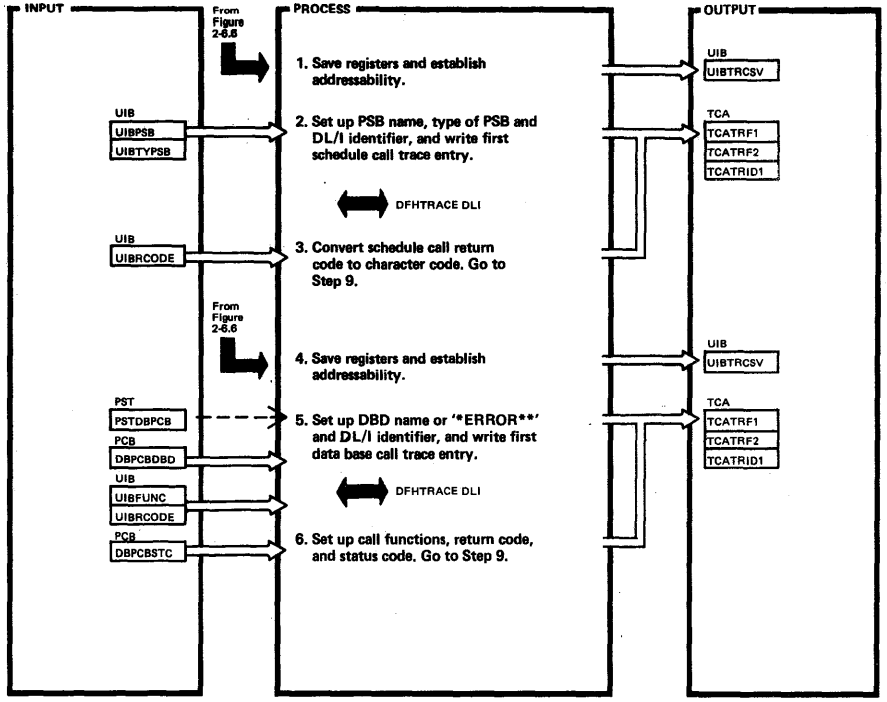
DLZPRH00 — Online Program Request Handler

DLZPRH00

Extended Description	Routine	Label
5E.		PRHABEND
5F.		PRHSYSAB

Extended Description	Routine	Label

Figure 2-6.7. Online Trace Entry Routine (Part 1 of 2)



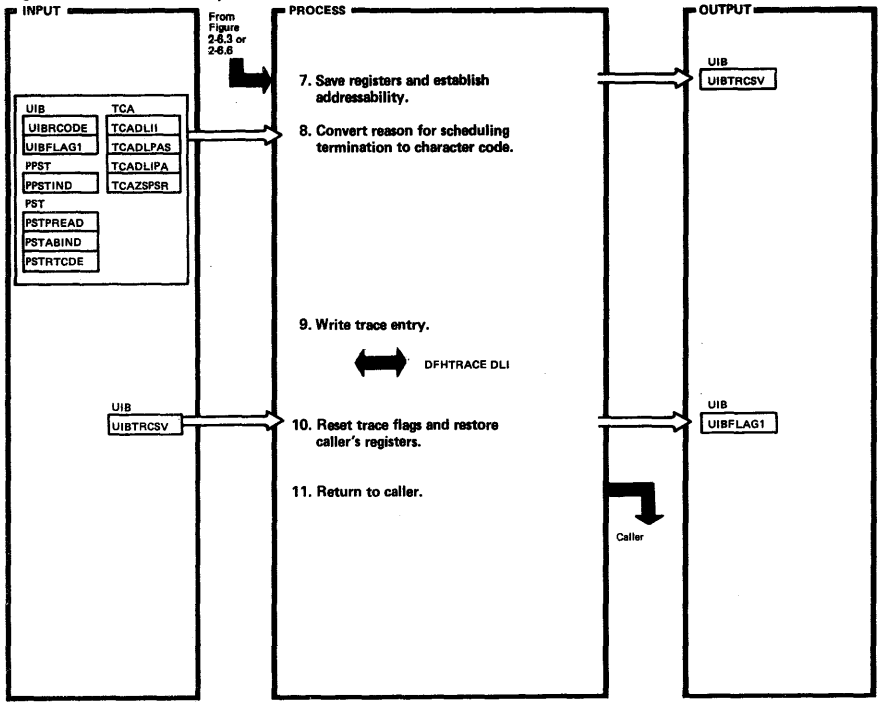
DLZOLT00 – Online Trace Entry Routine

DLZOLT00

Extended Description	Routine	Label
1. Trace entries are written to the CICS/VS Trace Table or auxiliary trace data set.	DLZOLT00	
4.	DLZOLT01	

Extended Description	Routine	Label

Figure 2-6.7. Online Trace Entry Routine (Part 2 of 2)



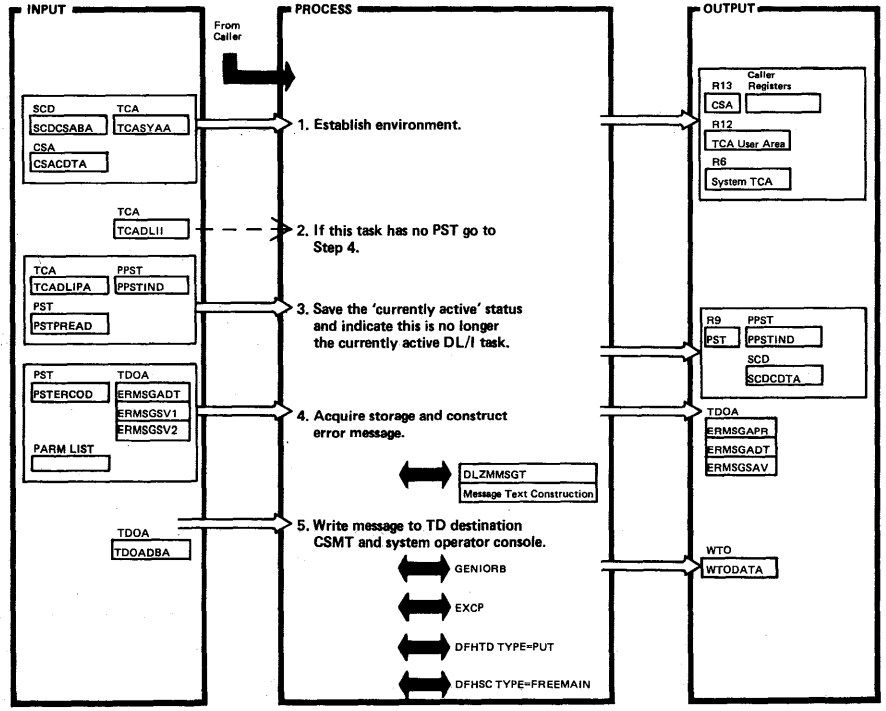
DLZOLT00 – Online Trace Entry Routine

DLZOLT00

Extended Description	Routine	Label
7.		DLZOLT02

Extended Description	Routine	Label

Figure 2-6.8. Online Error Message Routine (Part 1 of 2)



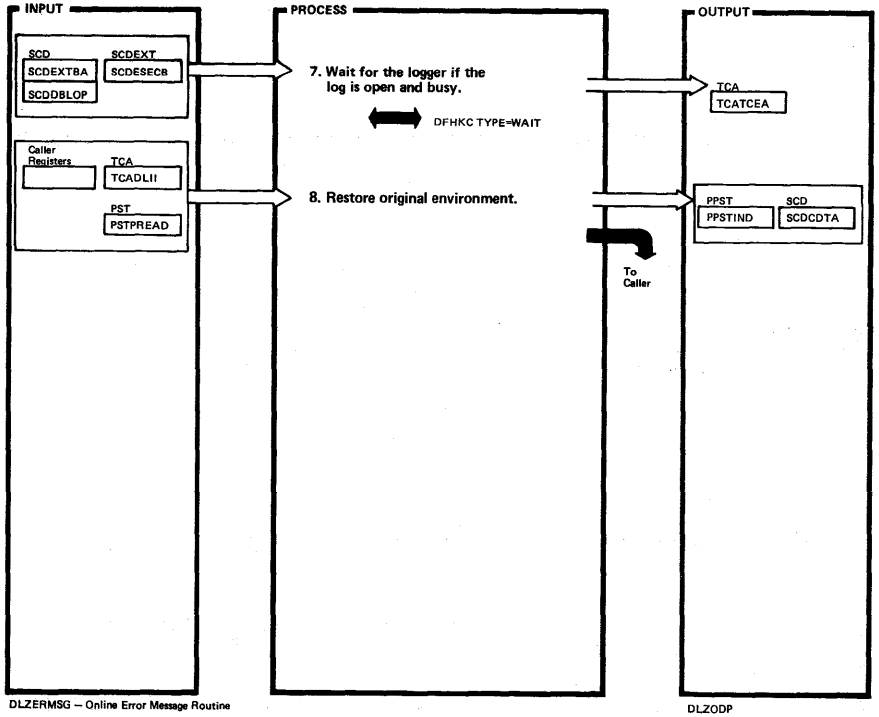
DLZERMSG – Online Error Message Routine

DLZODP

Extended Description	Routine	Label
1. Routine identifier (DLZERMSGvrmp) is defined here. The level format is vrmp, where 'v' is the version, 'r' is the release 'n' is an additional identification number, and 'p' is the latest PTF number that has been applied.	DLZERMSG	DLZERMSG
2. If there is no PST the message number will be in the parameter list which is pointed to by R1.		
3. 'Currently active' has a special meaning. There may be many DL/I tasks active at this time. Therefore, DL/I uses a bit (PPSTACT) in the PPST to make it easy to spot (in a dump) the single DL/I task that is currently processing non-scheduling DL/I calls (Non-scheduling calls being calls handled by the call analyzer and other DL/I action modules). Tasks like the MPS Batch Partition Controller can have a PST and can call DLZERMSG while not being the currently active task.		

Extended Description	Routine	Label
Since CICS/VS can schedule another DL/I task during a CICS/VS operation, (for example, I/O) PPSTACT is reset by DLZERMSG for the duration of the write because there can only be one task marked as 'currently active' by definition.		
4. The GETMAIN output buffer is needed by CICS/VS transient data services and IORB required by DOS/VS EXCP. DLZMMSGT is used to construct text for messages with message numbers from 1 – 255.		ERMSGETM
5.		ERMSGPUT

Figure 2-6.8. Online Error Message Routine (Part 2 of 2)



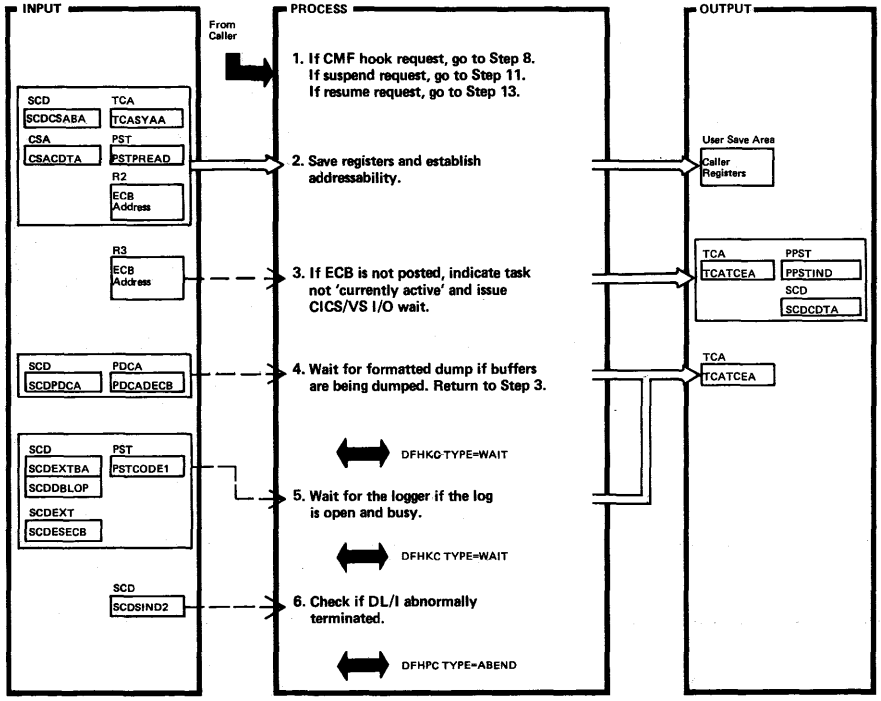
DLZERMSG – Online Error Message Routine

DLZODP

Extended Description	Routine	Label
6. Since control was lost during the I/O another job may have activated the logger. To assure serialization, this routine must wait until the logger is done.	DLZERMSG	ERMSGERT
7. Reset 'currently active' status to the value it was on entry.		

Extended Description	Routine	Label

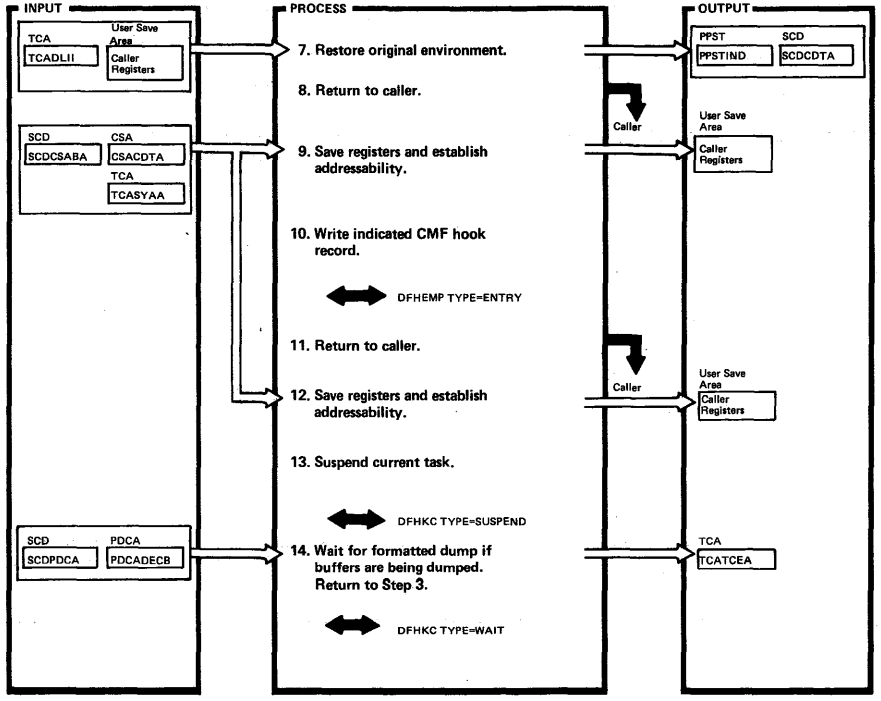
Figure 2-6.9. Online Wait Routine (Part 1 of 3)



Extended Description	Routine	Label
2. Routine identifier (DLZOWAITvrnp) is defined here. The level format is vrnp; where 'v' is the version, 'r' is the release, 'n' is an additional identification number, and 'p' is the latest PTF number that has been applied.	DLZOWAIT	DLZOWAIT
3. A non-scheduling task is a task that does not issue the special scheduling call (PCB,SYSTEMDL,password) to schedule itself so that it may issue system calls: CMXT, STRT, STOP, TSTR, and TSTP. 'Currently active' has a special meaning. There may be many DL/I tasks active at this time. Therefore, DL/I uses a bit (PPSTACT) in the PPST to make it easy to spot (in a dump) the single DL/I task that is currently processing non-scheduling DL/I calls (non-scheduling calls being calls processed by the call analyzer and other DL/I action modules). Since CICS/VS can schedule another DL/I task during a CICS/VS operation		OWATRECK

Extended Description	Routine	Label
(for example, I/O) PPSTACT is turned off until return is made to the caller of DLZOWAIT because there can only be one task marked as 'currently active' by definition.		
5. Serialization of the logger resource is another job of the online nucleus. Since control was lost during the wait, another job may have activated the logger. To assure serialization, this routine must wait until the logger is done. Return is then made to Step 2 to recheck that this task's ECB is posted and to reissue the wait again if it isn't.		
6.		OWAITRET

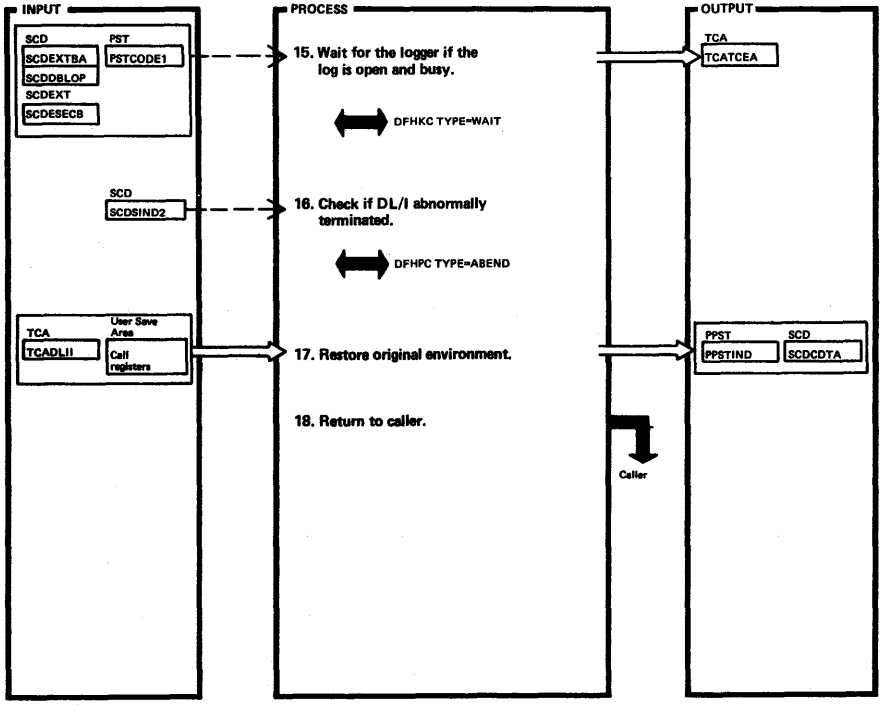
Figure 2-6.9. Online Wait Routine (Part 2 of 3)



Extended Description	Routine	Label
7. Reset non-scheduling task to currently active status as it was on entry.		
9.		DLZCMFHK
12.		DLZSUSP
14.		DLZRESUM

Extended Description	Routine	Label

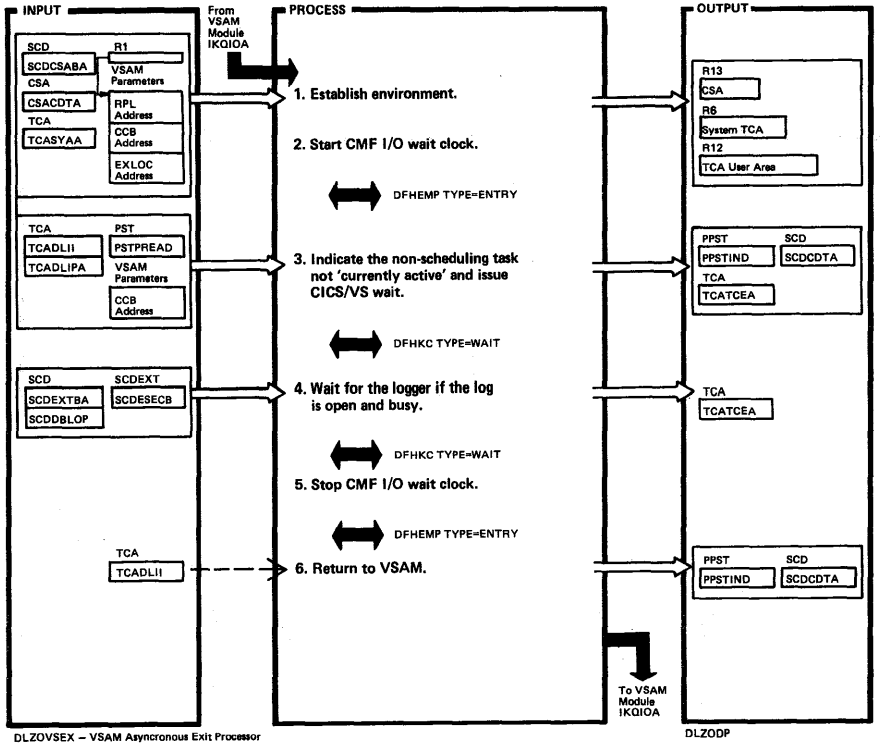
Figure 2-6.9. Online Wait Routine (Part 3 of 3)



Extended Description	Routine	Label
15. See "Extended Description" comment for Step 5.		
16.		OWAITRET

Extended Description	Routine	Label

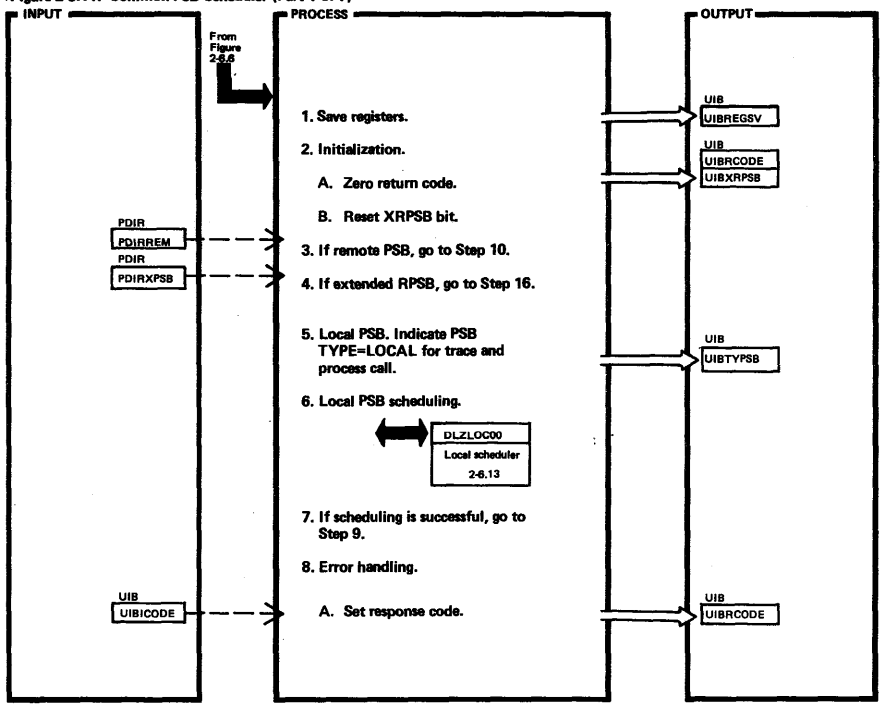
Figure 2-6.10. VSAM Asyncronous Exit Processor



Extended Description	Routine	Label
1. Routine identifier (DLZOVSEXvrnp) is defined here.	DLZOVSEX	DLZOVSEX
3. A non-scheduling task is a task that does not issue the special scheduling call (PCB,SYSTEMDL,password) to schedule itself so it may issue system calls: CMXT, STRT, STOP, TSTR, and TSTP. 'Currently active' has a special meaning. There may be many DL/I tasks active at this time. Therefore, DL/I uses a bit (PPSTACT) in the PPST to make it easy to spot (in a dump) the single DL/I task that is currently using the call analyzer and other DL/I action modules. Since CICS/VS can schedule another DL/I task during a CICS/VS operation (for example, I/O) PPSTACT is turned off until return is made to the caller of DLZOVSEX because there can only be one task marked as 'currently active' by definition.		OWATSYS

Extended Description	Routine	Label
4. Since control was lost during the wait another job may have activated the logger. To assure serialization, this routine must wait until the logger is done.		
6. Reset non-scheduling task to 'currently active' status as it was on entry.		OWATNLOG

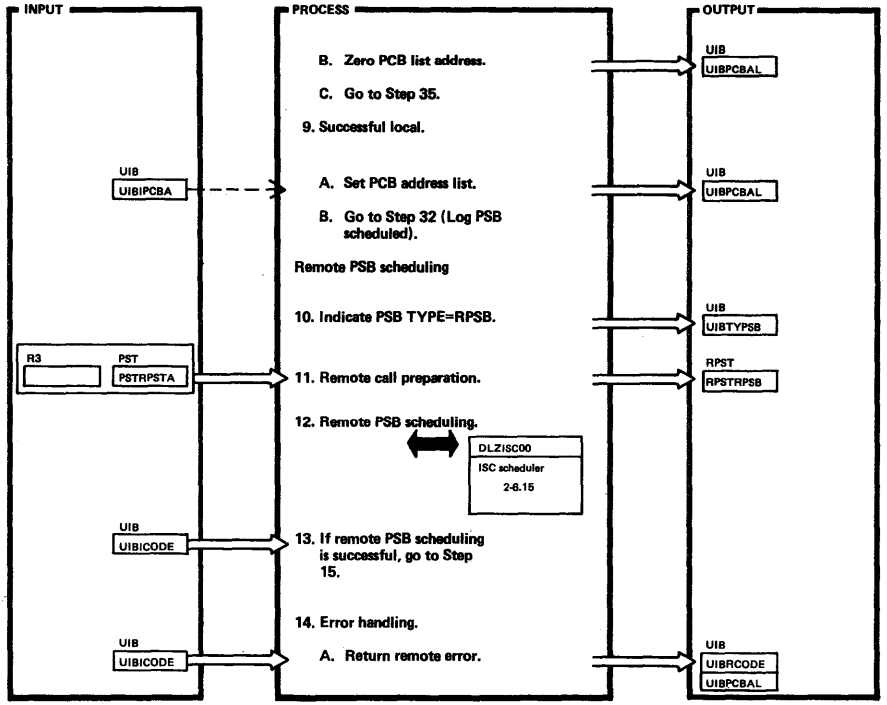
Figure 2-6.11. Common PSB Scheduler (Part 1 of 7)



Extended Description	Routine	Label

Extended Description	Routine	Label

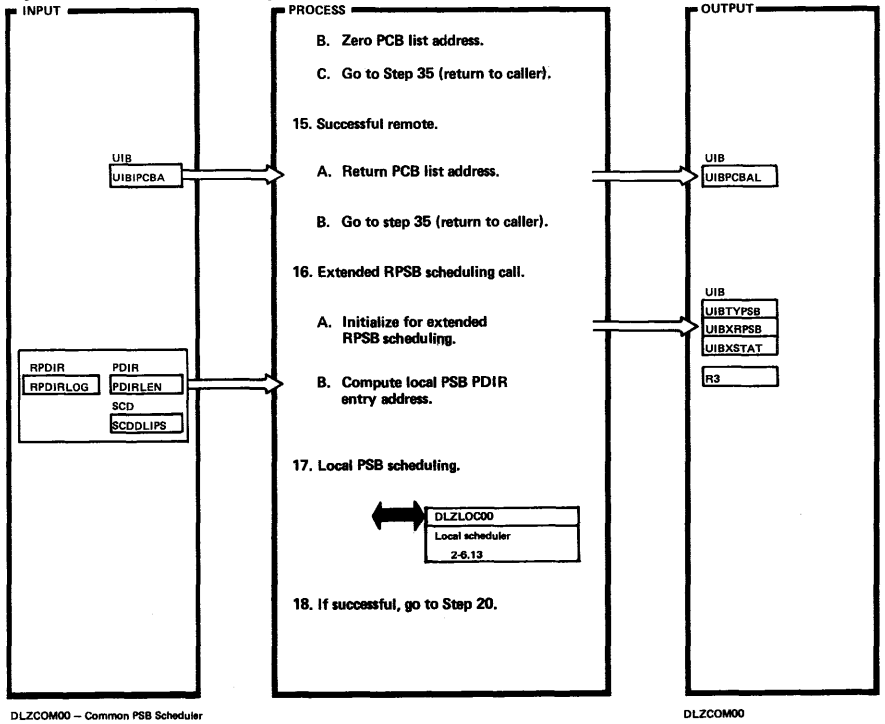
Figure 2-6.11. Common PSB Scheduler (Part 2 of 7)



Extended Description	Routine	Label

Extended Description	Routine	Label

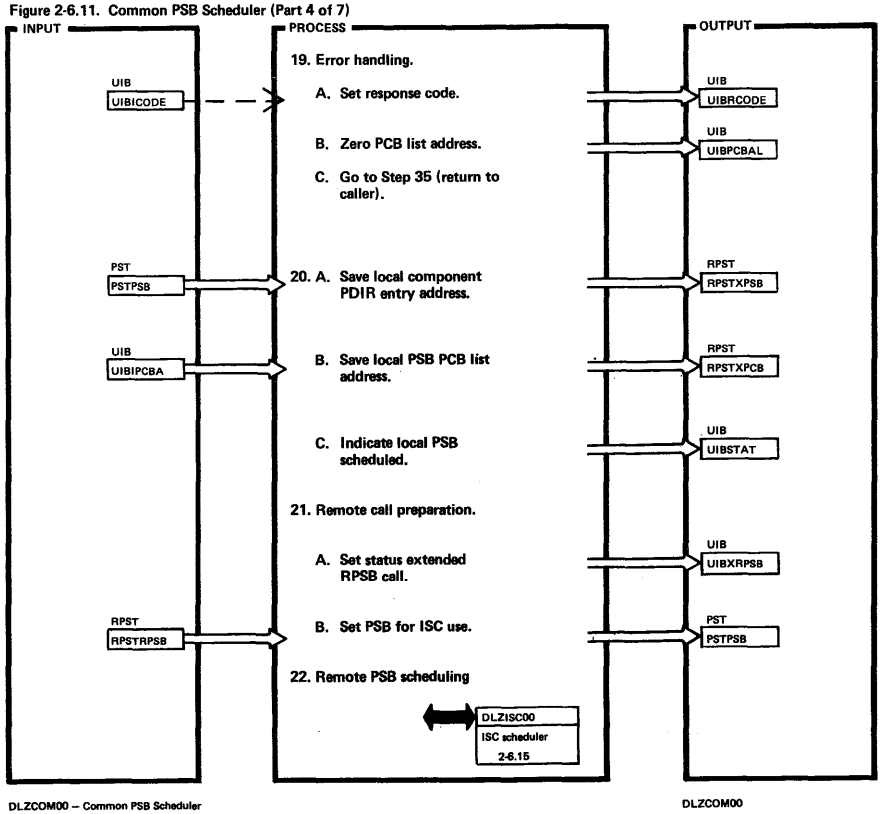
Figure 2-6.11. Common PSB Scheduling (Part 3 of 7)



Extended Description	Routine	Label

Extended Description	Routine	Label

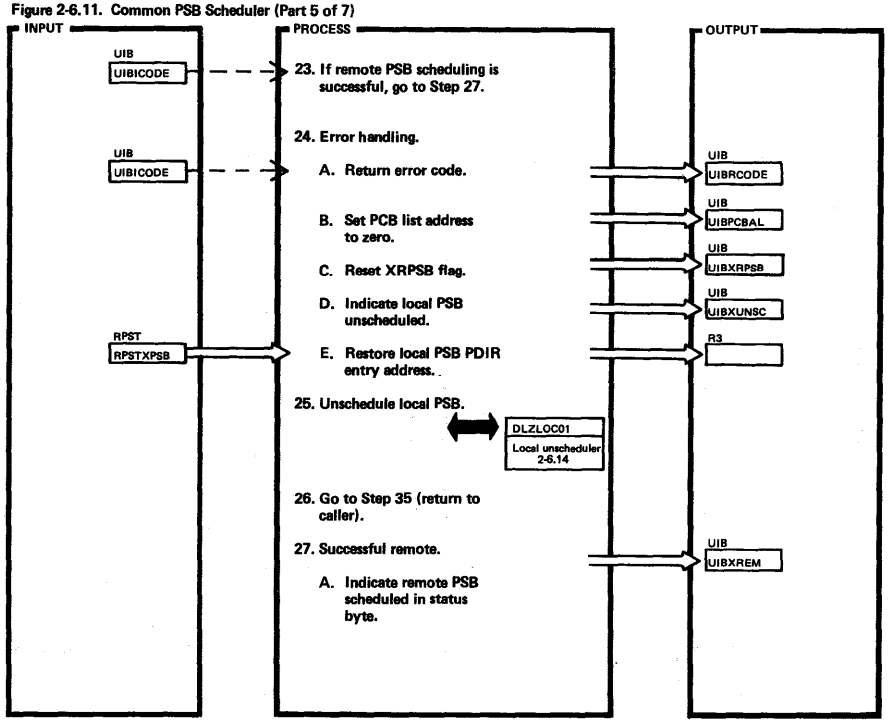
Figure 2-6.11. Common PSB Scheduler (Part 4 of 7)



Extended Description	Routine	Label

Extended Description	Routine	Label

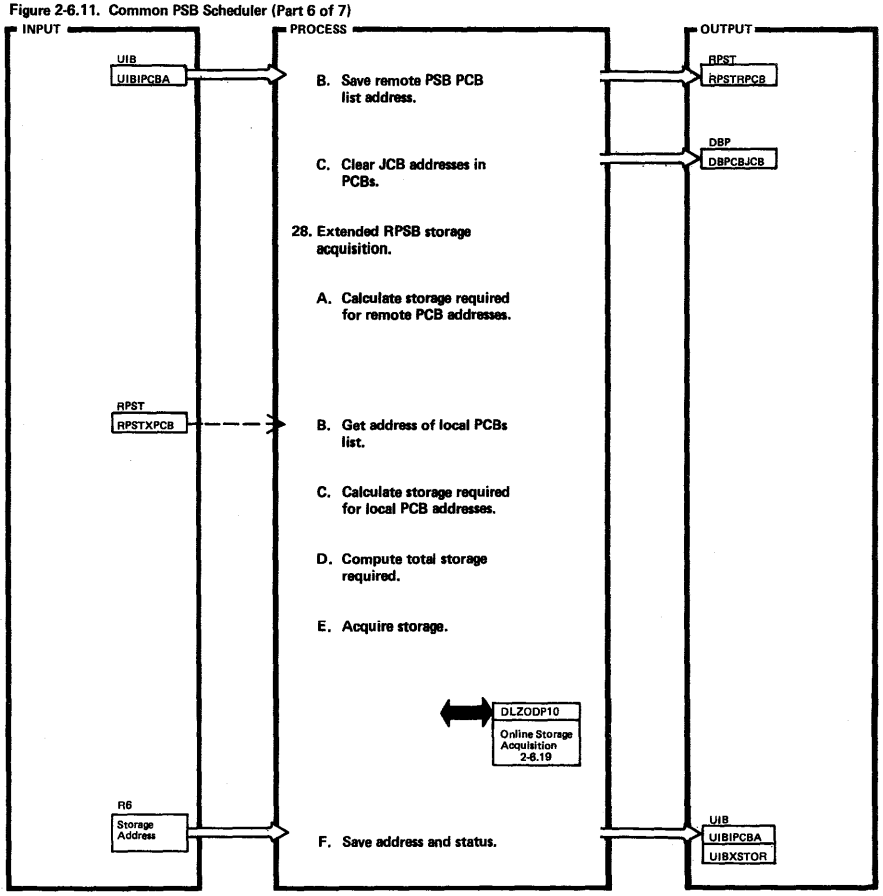
Figure 2-6.11. Common PSB Scheduler (Part 5 of 7)



Extended Description	Routine	Label

Extended Description	Routine	Label

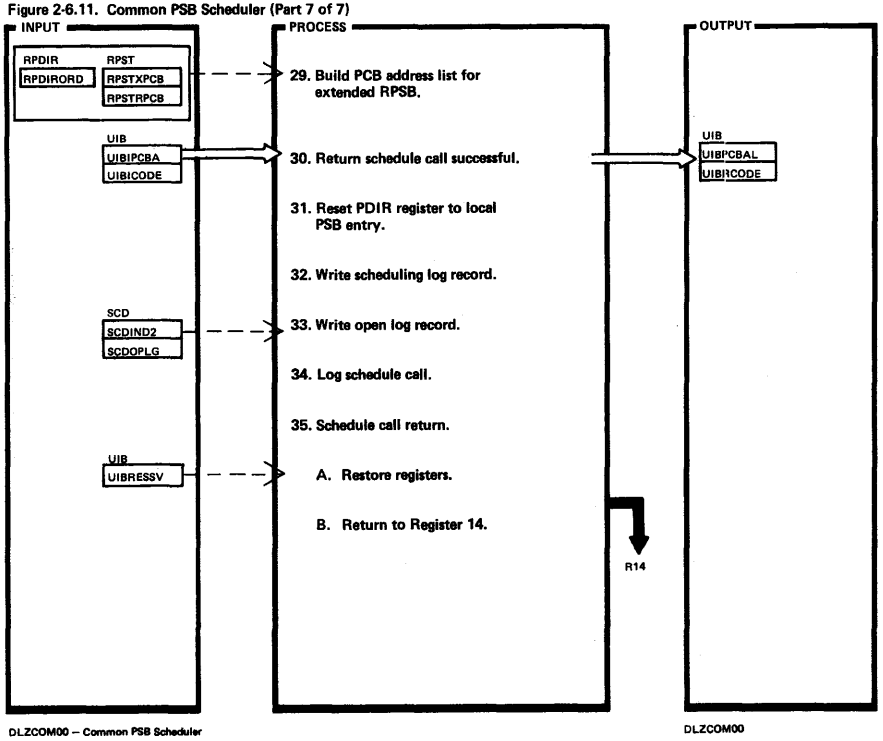
Figure 2-6.11. Common PSB Scheduler (Part 6 of 7)



Extended Description	Routine	Label
27C. JCB addresses in local copies of remote PCBs are used to determine if a data base call is to a local or remote PCB.		DBPCBJCB

Extended Description	Routine	Label

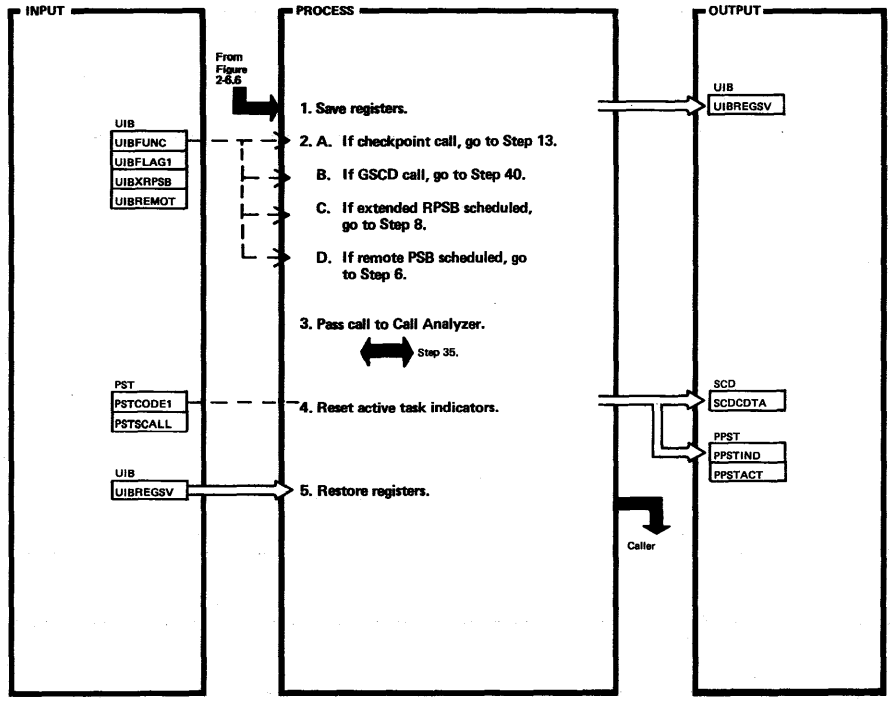
Figure 2-6.11. Common PSB Scheduler (Part 7 of 7)



Extended Description	Routine	Label

Extended Description	Routine	Label

Figure 2-6.12. Data Base Call Handler (Part 1 of 6)



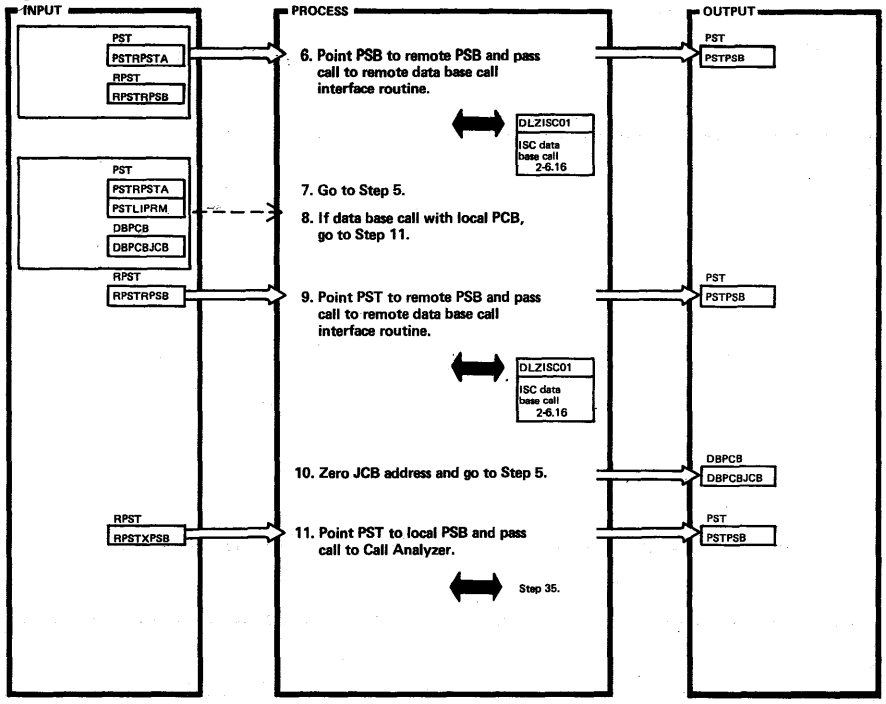
DLZCOM01 – Data Base Call Handler

DLZCOM01

Extended Description	Routine	Label
2. UIBREMOT is set during task scheduling. UIBREMOT=0 if not scheduled to a remote PSB. UIBXRPSB set by scheduling an extended PSB. DLZPRH00 sets UIBFUNC. Checkpoint call to RPSB is an implicit syncpoint call.	DLZCOM01	
5.		DBRETN

Extended Description	Routine	Label

Figure 2-6.12. Data Base Call Handler (Part 2 of 6)



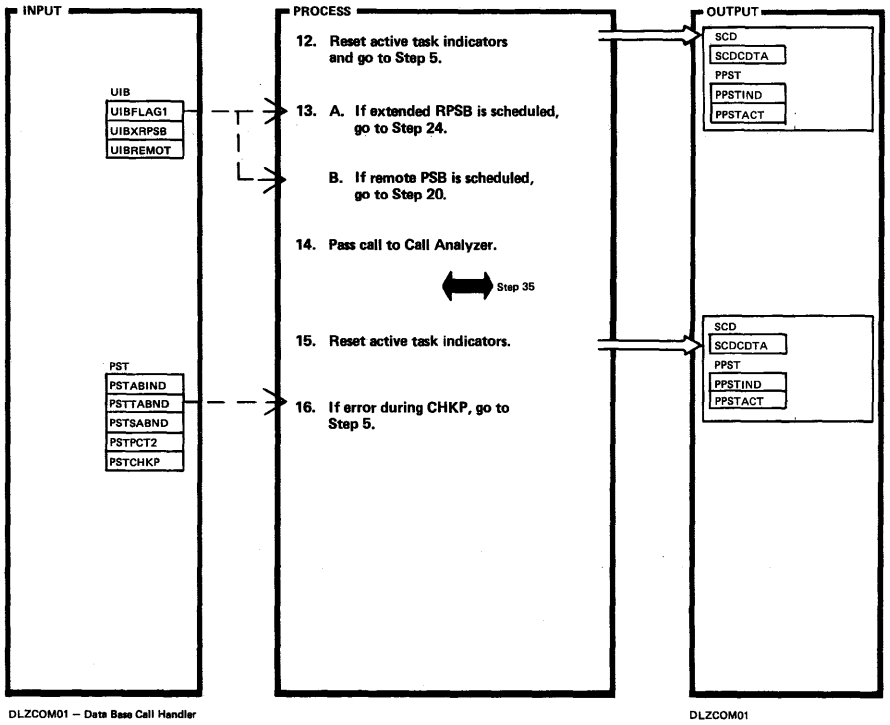
DLZCOM01 – Data Base Call Handler

DLZCOM01

Extended Description	Routine	Label
6.		DBREM
8. If JCB address is not zero, PCB is assumed to be a local PCB.		DBEXT

Extended Description	Routine	Label

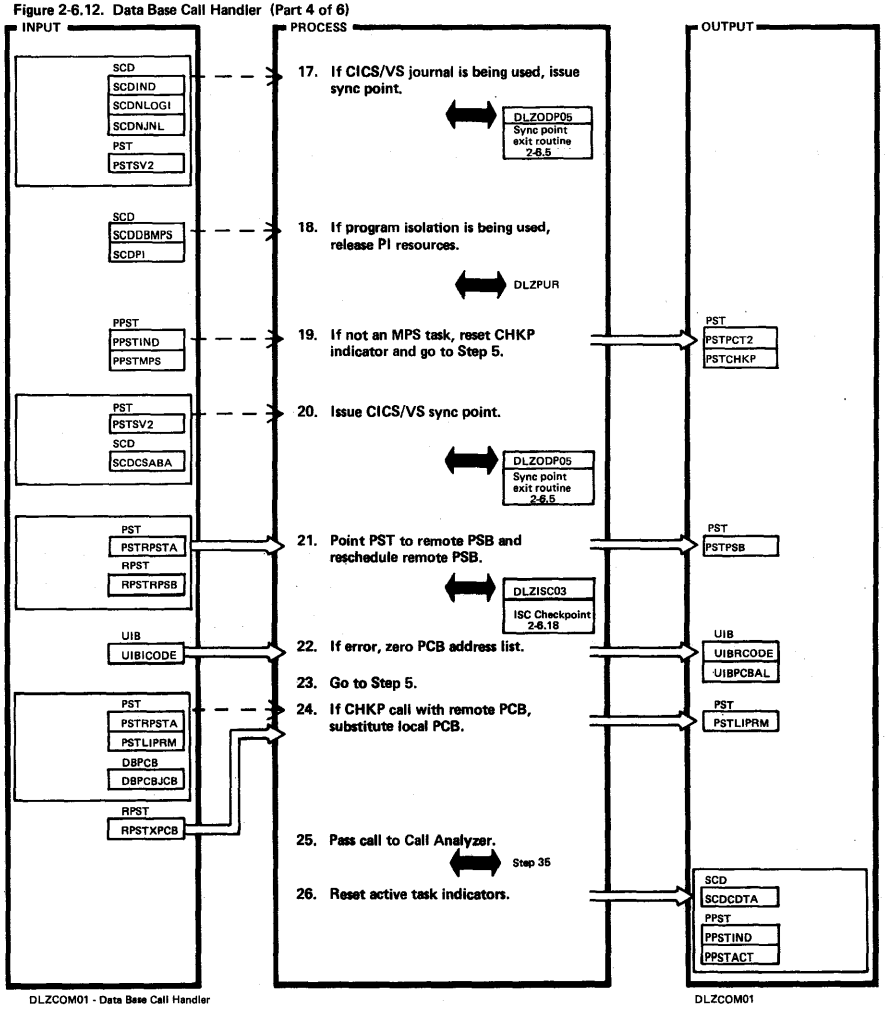
Figure 2-6.12. Data Base Call Handler (Part 3 of 6)



Extended Description	Routine	Label
13.		DBCHKP
14. CHKP call with local PSB scheduled		

Extended Description	Routine	Label

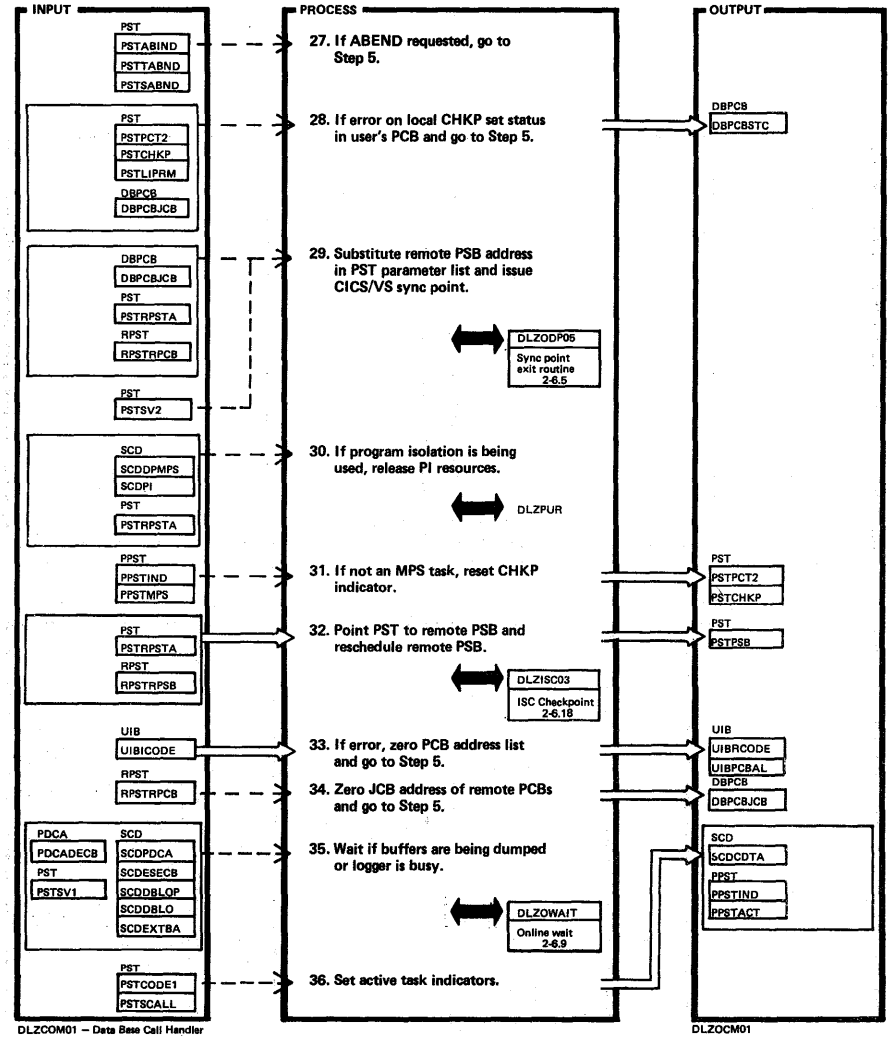
Figure 2-6.12. Data Base Call Handler (Part 4 of 6)



Extended Description	Routine	Label
20.		CHKPREM
24.		CHKPEXT

Extended Description	Routine	Label

Figure 2-6.12. Data Base Call Handler (Part 5 of 6)



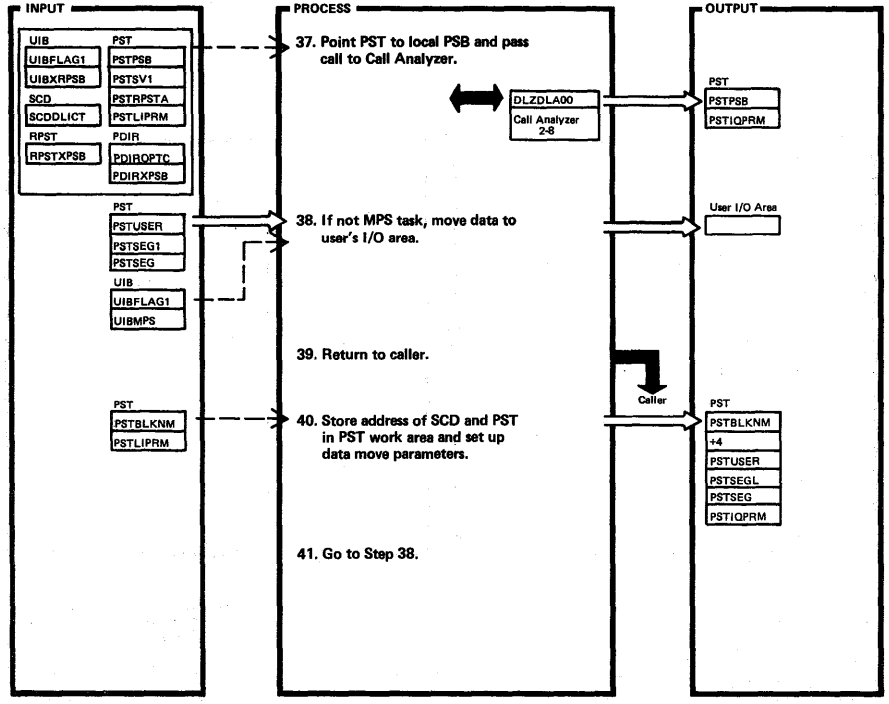
DLZCOM01 – Data Base Call Handler

DLZOCM01

Extended Description	Routine	Label
35.		DBANLYZ

Extended Description	Routine	Label

Figure 2-6.12. Data Base Call Handler (Part 6 of 6)



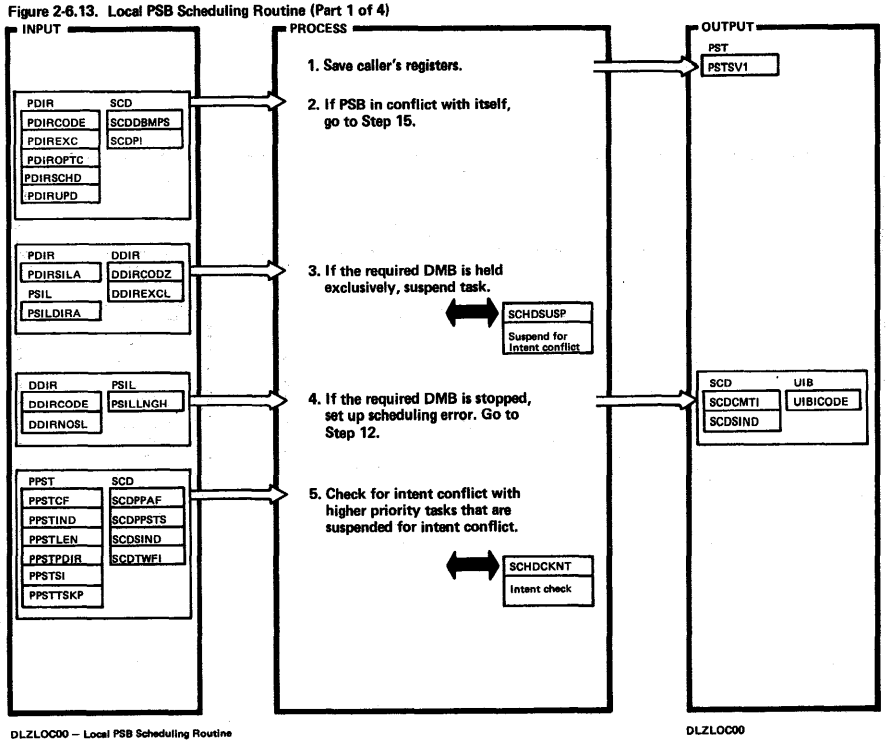
DLZCOM01 – Data Base Call Handler

DLZCOM01

Extended Description	Routine	Label
38.		DBANMOVE
40.		DBGSCD

Extended Description	Routine	Label

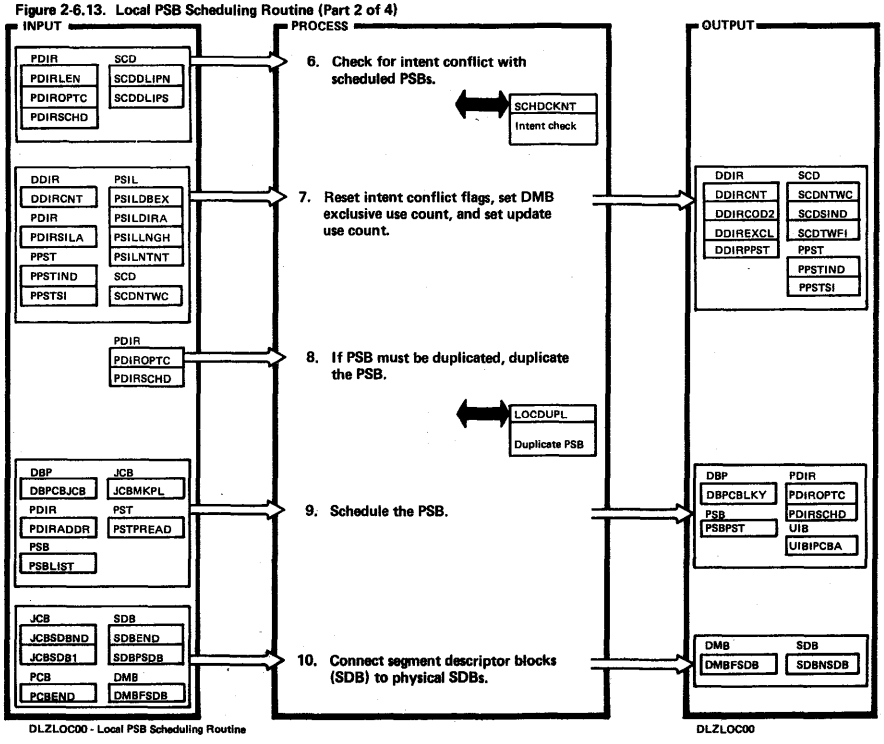
Figure 2-6.13. Local PSB Scheduling Routine (Part 1 of 4)



Extended Description	Routine	Label
3.		SCHDCKSI
4. Error retruned is X'0C01' (DB stopped).		

Extended Description	Routine	Label

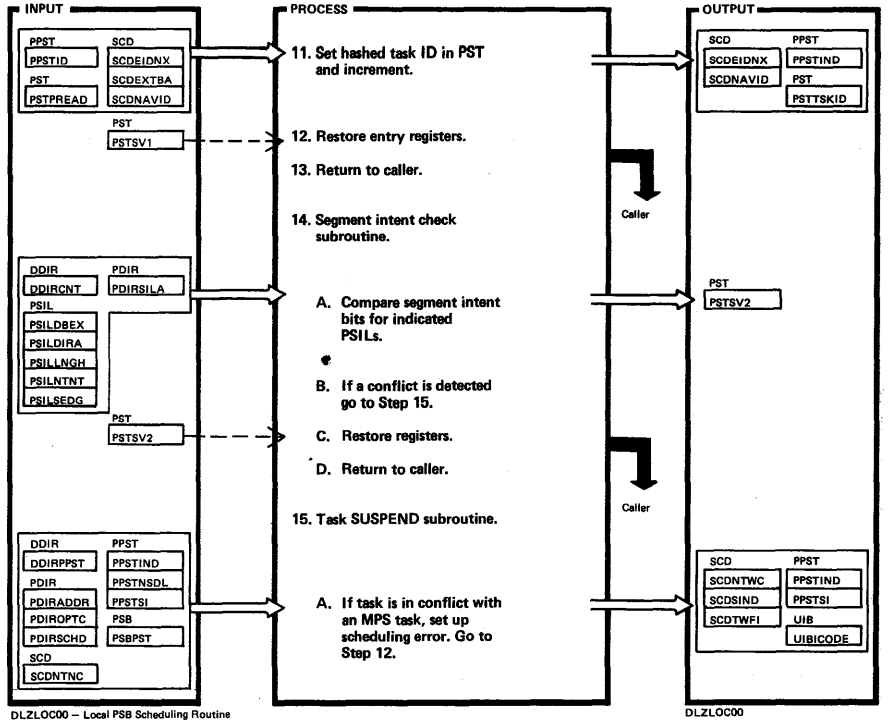
Figure 2-6.13. Local PSB Scheduling Routine (Part 2 of 4)



Extended Description	Routine	Label
8.		SCHDCKSI

Extended Description	Routine	Label

Figure 2-6.13. Local PSB Scheduling Routine (Part 3 of 4)



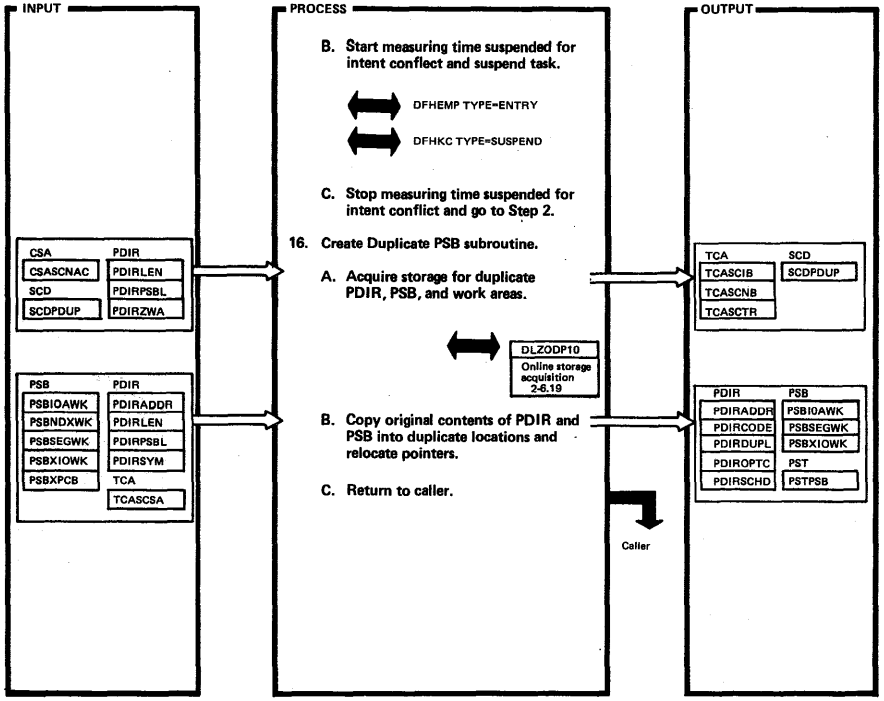
DLZLOC00 – Local PSB Scheduling Routine

DLZLOC00

Extended Description	Routine	Label
12.		LOCEXIT
14A.		SCHDCKNT
15A.		SCHDSUSE

Extended Description	Routine	Label

Figure 2-6.13. Local PSB Scheduling Routine (Part 4 of 4)



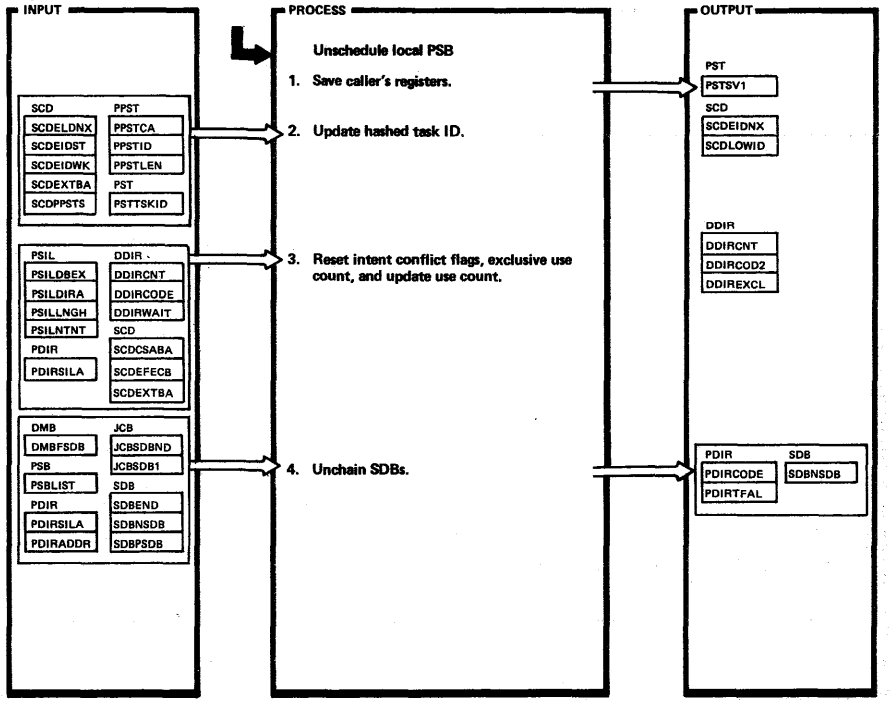
DLZLOC00 – Local PSB Scheduling Routine

DLZLOC00

Extended Description	Routine	Label
16.		LOCDUPL
16B. For read only or update with program isolation. Duplicate PSBs are identified by the PDIR indicator PDIRDUPL.		

Extended Description	Routine	Label

Figure 2-6.14. Unschedule Local PSB Routine (Part 1 of 2)



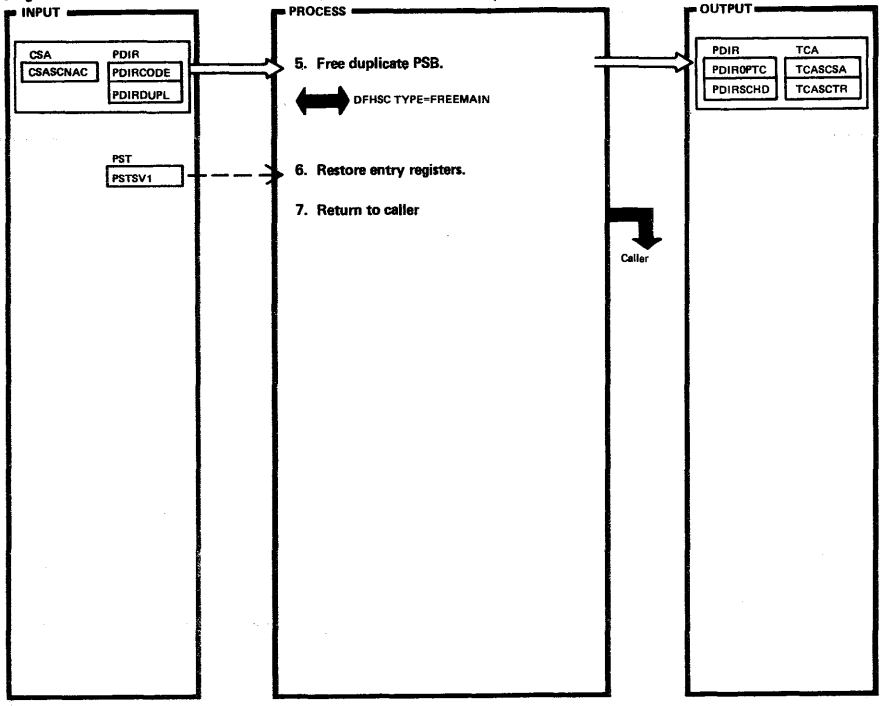
DLZLOC01 - Unschedule Local PSB Routine

DLZLOC01

Extended Description	Routine	Label
2. The lowest active identifier is maintained in the SCD. DL/I Space Management uses the low and high identifiers to exclude free space belonging to active tasks from reuse.		OVERID1
3.		TRIDEXIT
4. DB stopped. PDIRCODE=X '0C01'.		

Extended Description	Routine	Label

Figure 2-6.14. Unschedule Local PSB Routine (Part 2 of 2)



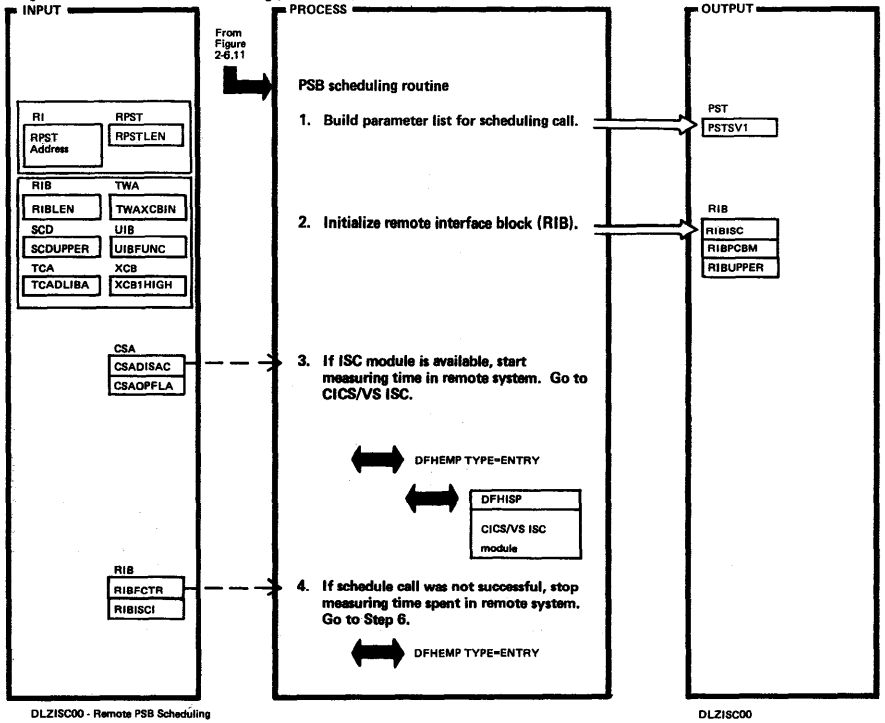
DLZLOC01 - Unschedule Local PSB Routine

DLZLOC01

Extended Description	Routine	Label
5.		TRMDUPCK

Extended Description	Routine	Label

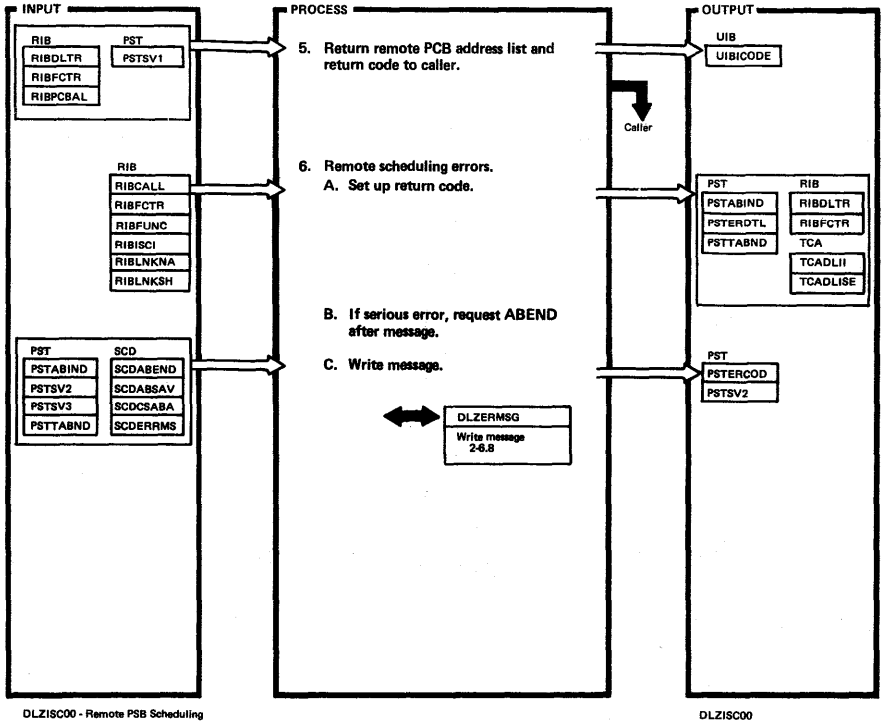
Figure 2-6.15. Remote PSB Scheduling (Part 1 of 2)



Extended Description	Routine	Label
1. RPSTISC1= (X'00') RPSTISC2=A (RIB) RPSTISC3=A (User PARM list) RPSTISC4=A (PDIR entry)		
3.		ISCBALR

Extended Description	Routine	Label

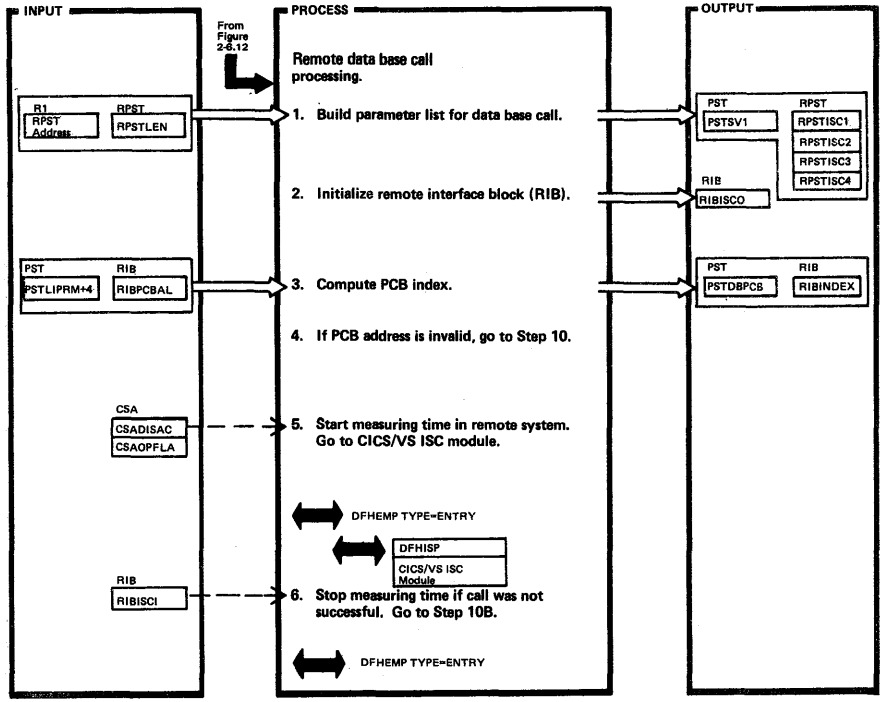
Figure 2-6.15. Remote PSB Scheduling (Part 2 of 2)



Extended Description	Routine	Label
6A. ABEND if PARM list is invalid, function is invalid, XECB cannot be found, PSB cannot be rescheduled following a checkpoint, or internal error.		ISCNOMOD ISCRIBER ISCTCAER
6C. Write message DLZ033I.		ISC033I

Extended Description	Routine	Label

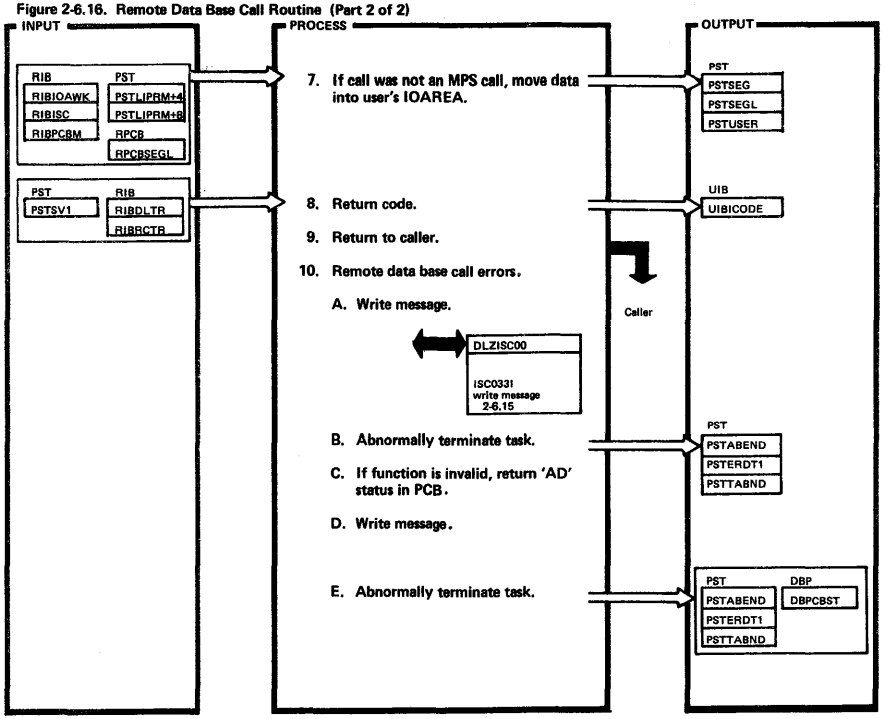
Figure 2-6.16. Remote Data Base Call Routine (Part 1 of 2)



Extended Description	Routine	Label
1. RPSTISC1=A (X'04') RPSTISC2=A (RIB) RPSTISC3=A (User PARM list) RPSTISC4=A (PDIR entry)	OVERID1	
3.	ISCINDEX	
5.	ISCBALR1	

Extended Description	Routine	Label

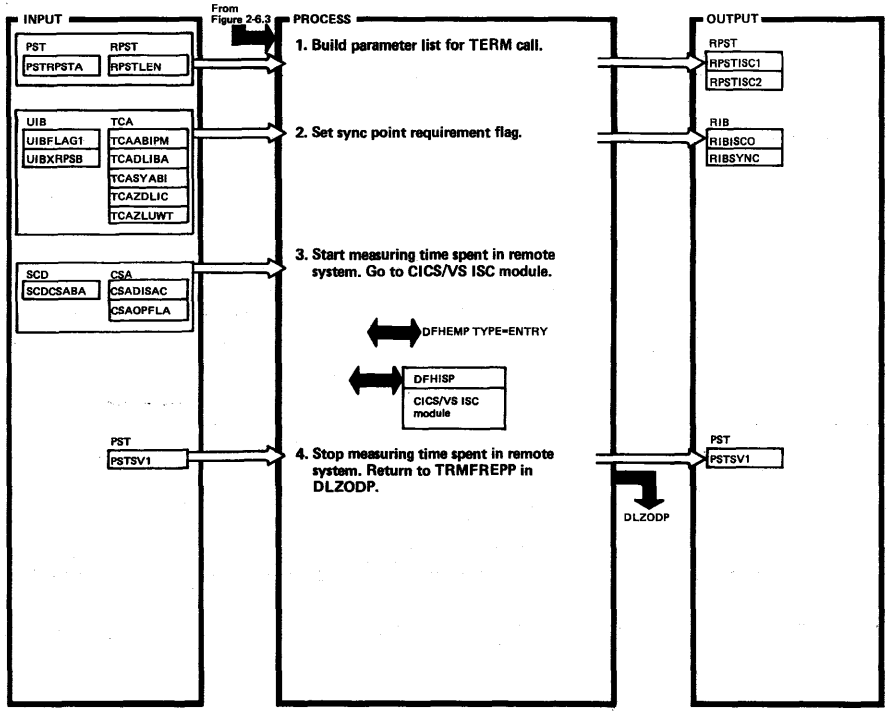
Figure 2-6.16. Remote Data Base Call Routine (Part 2 of 2)



Extended Description	Routine	Label
8.		ISCRET01
10A. Write message DLZ476I.		ISCNONDX
10D. Write message DLZ033I.		ISCRIBAD

Extended Description	Routine	Label

Figure 2-6.17. Remote Termination Call Routine



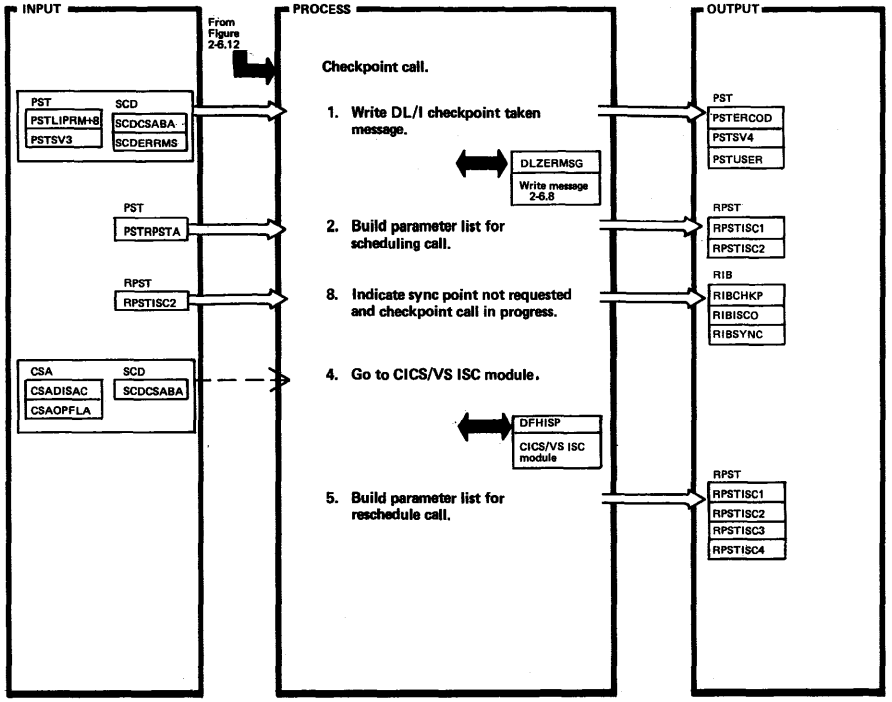
DLZISC02 – Remote Termination Call Routine

DLZISC02

Extended Description	Routine	Label
1. RPSTISC1=A(X'08') RPSTISC2=A(RIB)		

Extended Description	Routine	Label

Figure 2-6.18. Remote Rescheduling Routine (Part 1 of 2)



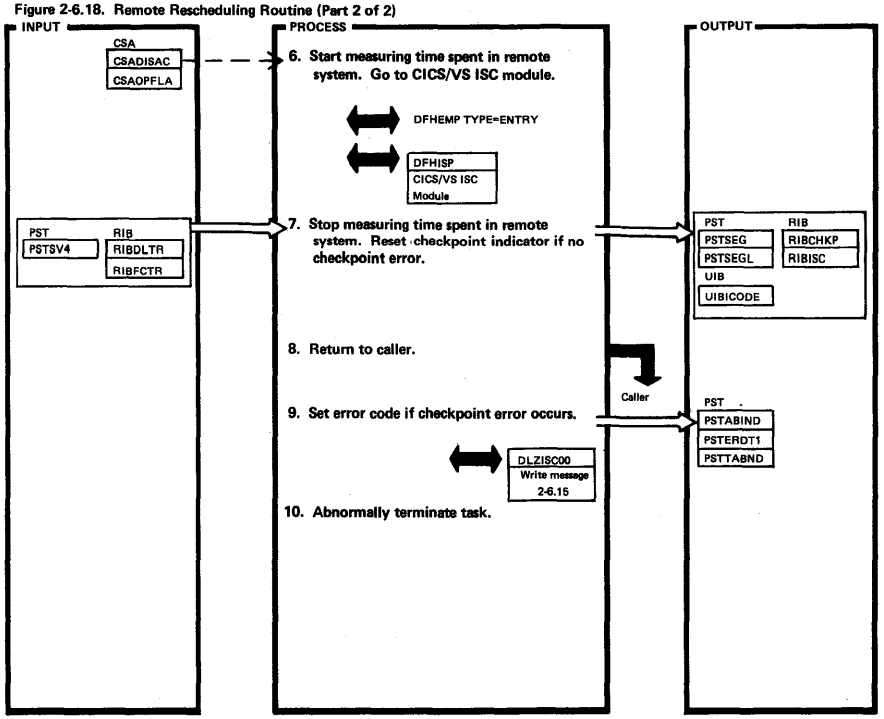
DLZISC03 - Remote Rescheduling Routine

DLZISC03

Extended Description	Routine	Label
1. Write message DLZ1051. 2. RPSTISC1=A (X' 08') RPSTISC2=A (RIB) 5. RPSTISC1=A (X'00') RPSTISC2=A (RIB) RPSTISC3=A (CHKPSCHD) RPSTISC4=A (PDIR entry) CHKPSCHD is: DC X'80' DC AL3 ('PCB ')		

Extended Description	Routine	Label

Figure 2-6.18. Remote Rescheduling Routine (Part 2 of 2)



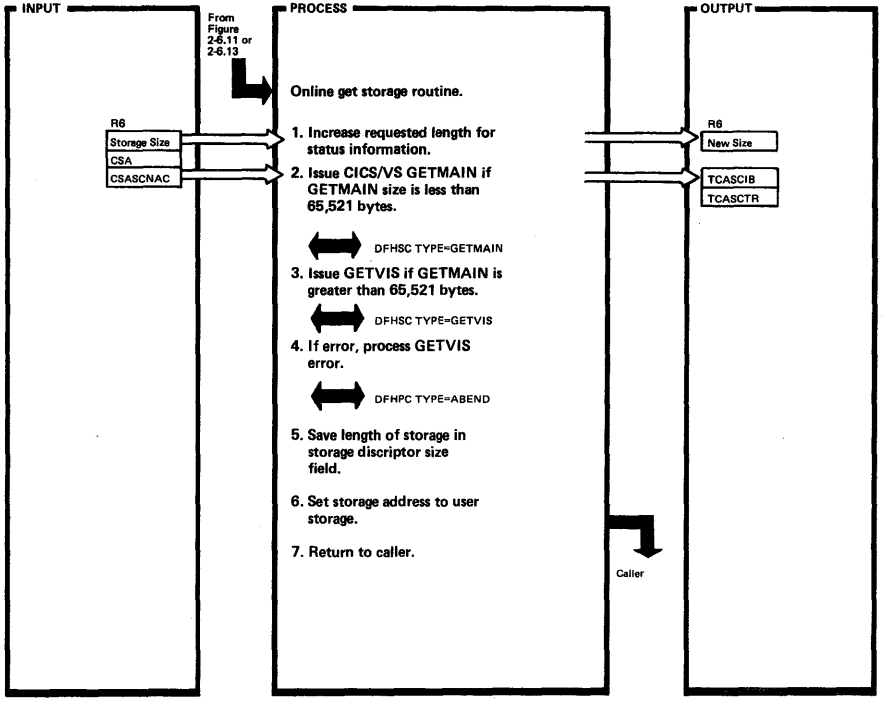
DLZISC03 - Remote Rescheduling Routine

DLZISC03

Extended Description	Routine	Label
9.		ISCKPERR

Extended Description	Routine	Label

Figure 2-6.19. Online Get Storage Routine



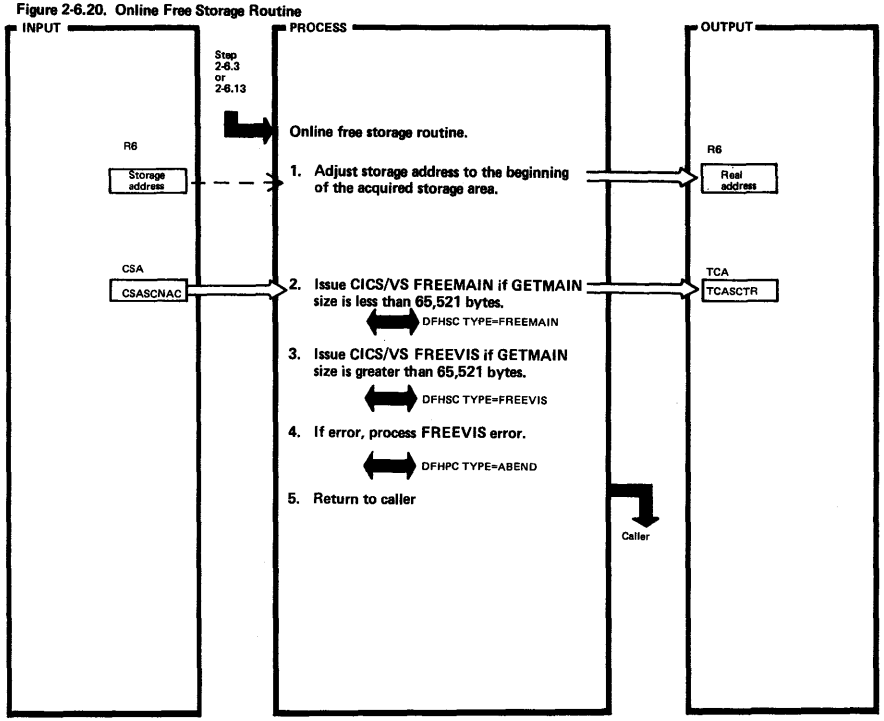
DLZODP10 – Online Get Storage Routine

DLZODP10

Extended Description	Routine	Label
4. Convert return code to printable form. Write DLZ038I message. Abnormally terminate task.		

Extended Description	Routine	Label

Figure 2-6.20. Online Free Storage Routine



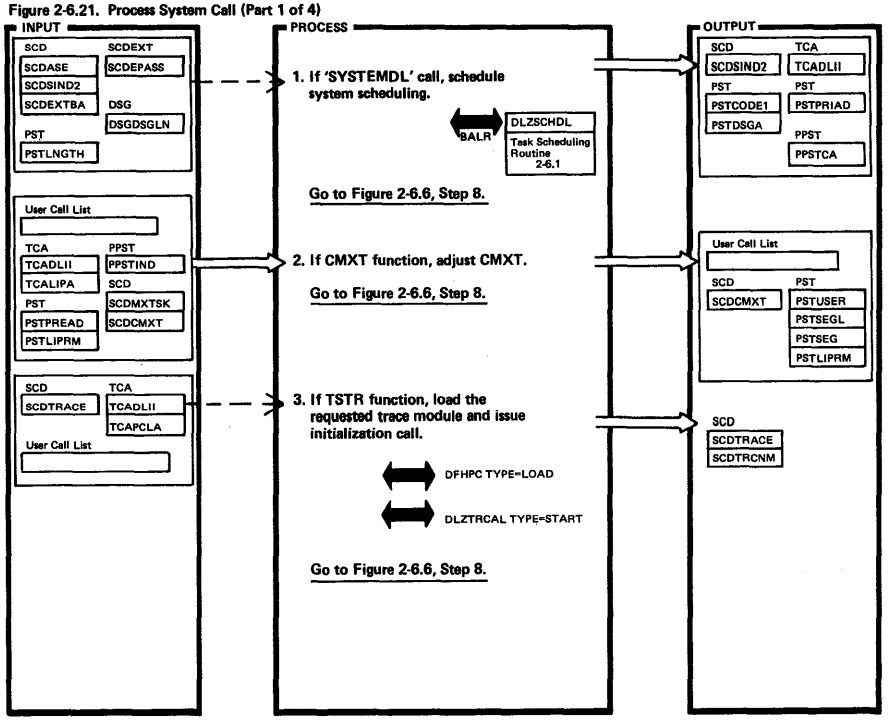
DLZODP11 - Online Free Storage Routine

DLZODP11

Extended Description	Routine	Label
4. Covert return code to printable form. Write DLZ038I message. Abnormally terminate task.		

Extended Description	Routine	Label

Figure 2-6.21. Process System Call (Part 1 of 4)



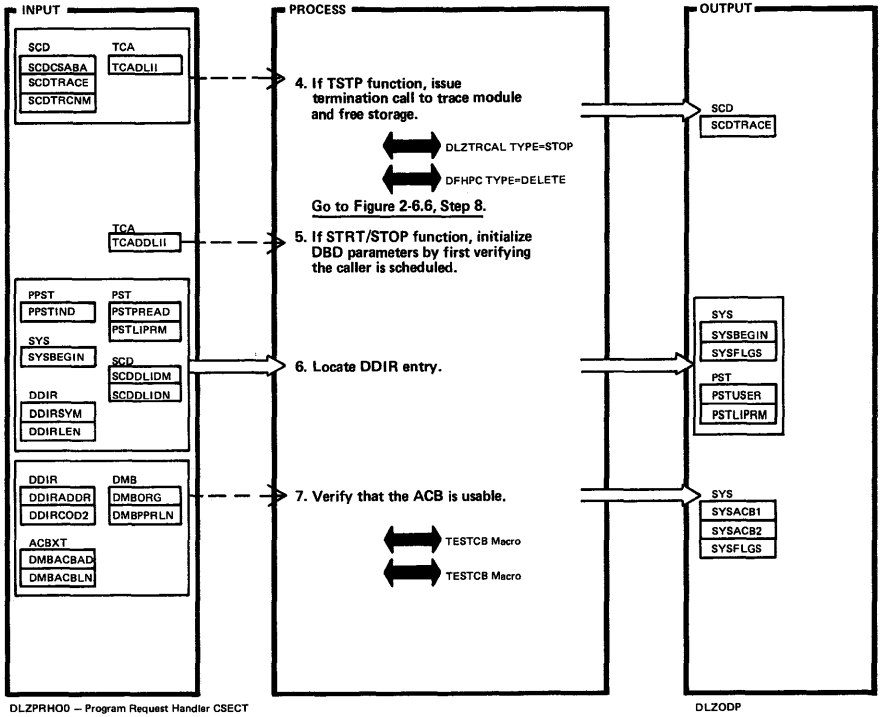
DLZPRHOO – Program Request Handler CSECT

DLZODP

Extended Description	Routine	Label
1. A task requesting services through the system calls must have been previously scheduled by password with this special schedule PCB SYSTEMDL call. If the password does not match, the caller abends via DFHPC with code DLPV. Important indicators set are: SCDSYACT – system interface active TCADLITE – termination required TCADLPAS – system task scheduled TCADLISD – task scheduled Exit is taken to scheduling routine to get a PST and initialized upon return. PSTSCALL (system call in progress) is set and return is made to caller.		PROCSYS
2. The value passed by the user validated and moved to the SCD.		PROCMXT

Extended Description	Routine	Label
Without MPS, data is moved to the user call list area for work space. With MPS this would cause a storage protection exception. To avoid this, an area in the PST (PSTLIPRM) (in the CICS/VS DL/I partition) is used as a work area. The MPS batch program request handler (DLZMPI00) then moves the data from the PST into the user call list. If the new request is zero, negative, or exceeds MAXTASK indicate an invalid request error (set 08 in TCAFCTR).		
3. If task not scheduled for system calls (TCADLPAS not on) abend via DFHPC. If tracing is already active set X'01' in TCAFCTR. If the load fails, set X'02' in TCAFCTR. If GETMAIN fails during initialization, set X'04' in TCAFCTR.		PROCTSTR

Figure 2-6.21. Process System Call (Part 2 of 4)



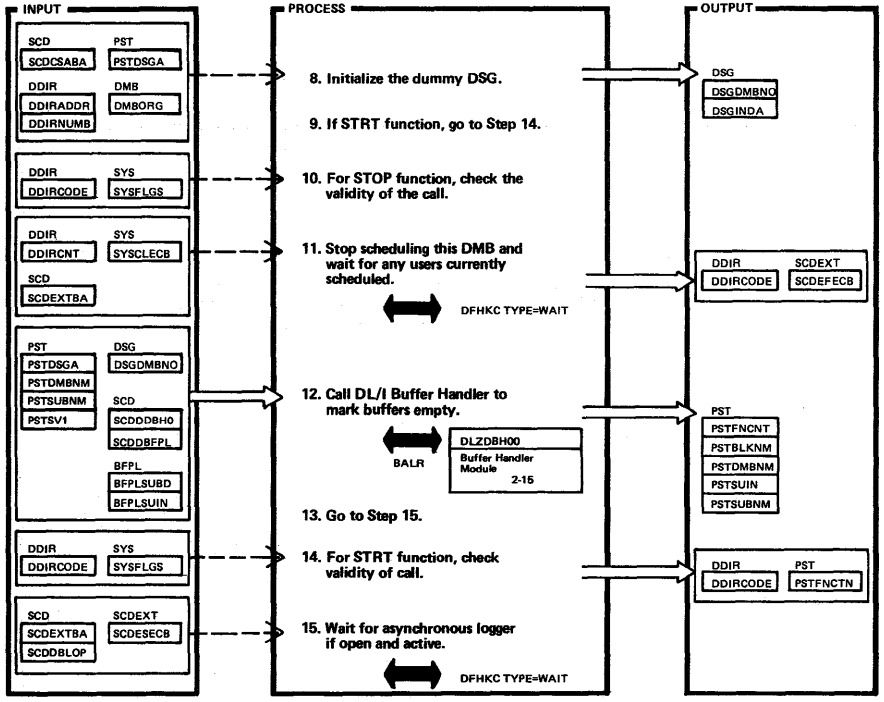
DLZPRHOO – Program Request Handler CSECT

DLZODP

Extended Description	Routine	Label
4. If task not scheduled for system calls (TCADLPAS not on) abend via DFHPC. If tracing not active, set X'01' in TCAFCTR for invalid request.		PROCTSTP
5. If task not scheduled for system calls (TCADLPAS not on) abend via DFHPC.		PROCINIT
6. The DMB name passed by the caller is used to scan the DDIR. If this is not an MPS task, data (VSAM return code and ACB address) is moved into the user call list for the caller. With MPS, a work area is used in the PST (PSTLIPRM) to build the data from the PST to the user call list. If the DDIR is not found, set X'08' in TCAFCTR to indicate an invalid request.		PROICON

Extended Description	Routine	Label
7. The ACBs are checked for open/close status. The ACB address (2 if HISAM organization) and whether the ACB is open or not is put into the user call list (or if this is an MPS task, into the PST). Reference to fields within either of these two areas is by the system call parameter list DSECT (DLZSYSDS). If the DDIR failed to initialize, set X'02' in TCAFCTR. If TESTCB request fails, set X'03' in TCAFCTR.		PROCACB

Figure 2-6.21. Process System Call (Part 3 of 4)



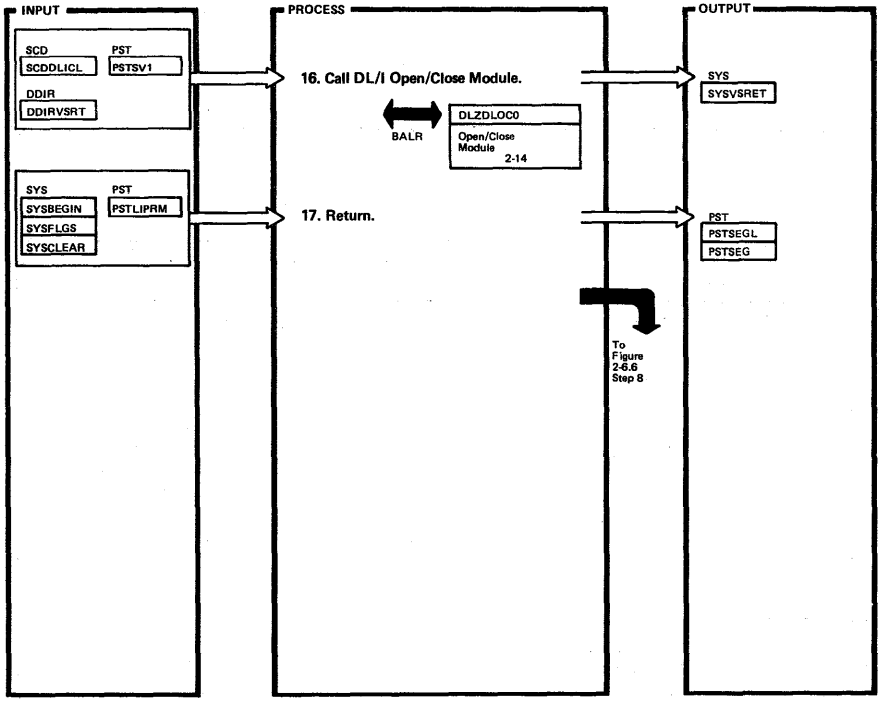
DLZPRH00 – Program Request Handler CSECT

DLZODP

Extended Description	Routine	Label
8.		PROCDSG
10. If the ACB is not open, set X'01' to indicate the STOP call is invalid.		PROCSTOP
12. Set X'04' in PSTFNCNT to indicate 'mark buffers empty request to buffer handler'. The DMB number is used to index into the subpool directory to get the subpool number for this DMB. If there is no subpool number, bypass calling the buffer handler. Set X'01' in PSTFNCNT to indicate 'close DMB request' to open/close module upon return from buffer handler.		GOTOBUFF
14. If this data base is not stopped or if the ACB is open, X'01' in TCAFCTR to indicate the STRT call is invalid. Set X'09' in PSTFNCTN to indicate 'open DMB request' to the open/close module.		PROCSTRT
15.		PROCOCR

Extended Description	Routine	Label

Figure 2-6.21. Process System Call (Part 4 of 4)



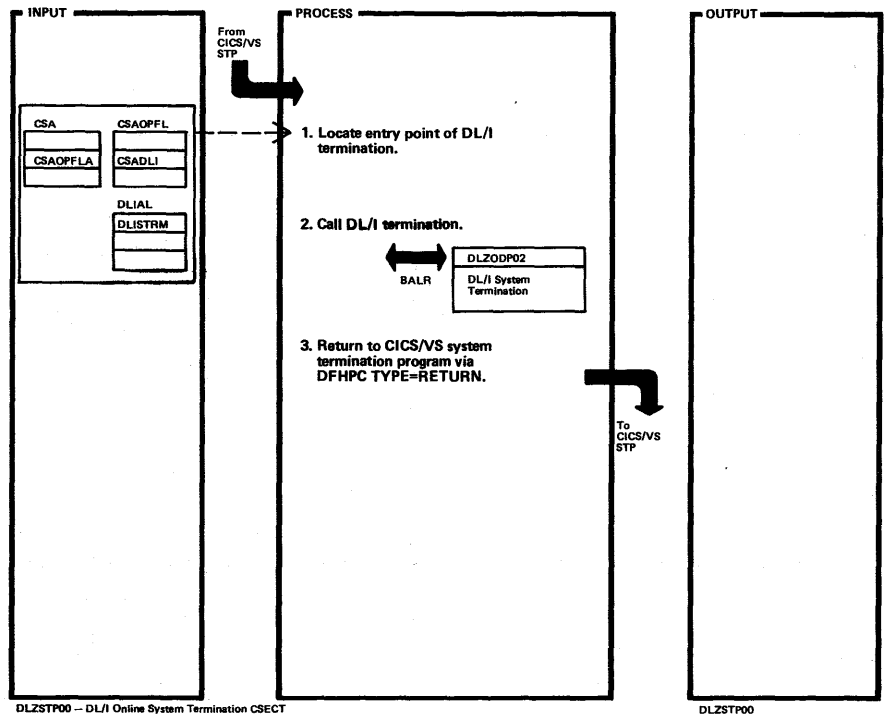
DLZPRH00 – Program Request Handler CSECT

DLZODP

Extended Description	Routine	Label
16. The open/close module issues and SVC2 and CICS/VS loses control for the duration of the call. The VSAM return code in the DDIR upon return from the buffer handler, is moved to the user call list area (or if this is an MPS task, into the PST) which is mapped by DLZSYSDS – the system call parameter list DSECT.		BYSYSWAT
17. If this is an MPS task, PSTSEGL and PSTSEG must be set up for MPS batch PRH (DLZMPI00) to move data from PSTLIPRM to the user call list area. PSTUSER already contains the address of user call list area.		RETURNB

Extended Description	Routine	Label

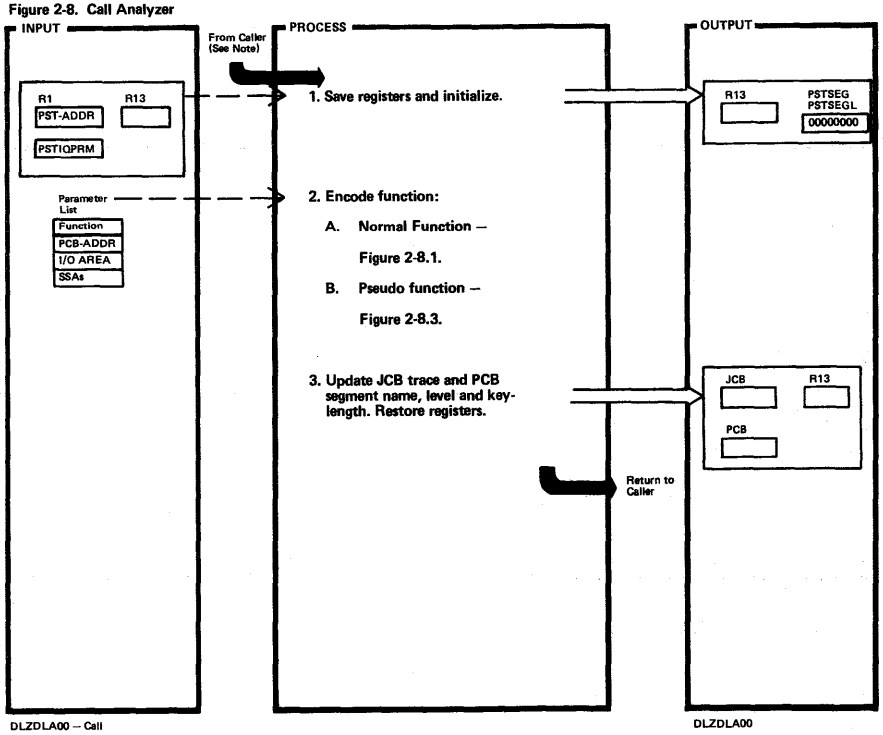
Figure 2-7. DL/I Online System Termination



Extended Description	Routine	Label
1. Control is gained from CICS/VS System Termination Program (STP) because of DLZSTP00s presence in the program list table (DFHPLT).	DLZSTP00	DLZSTP00

Extended Description	Routine	Label

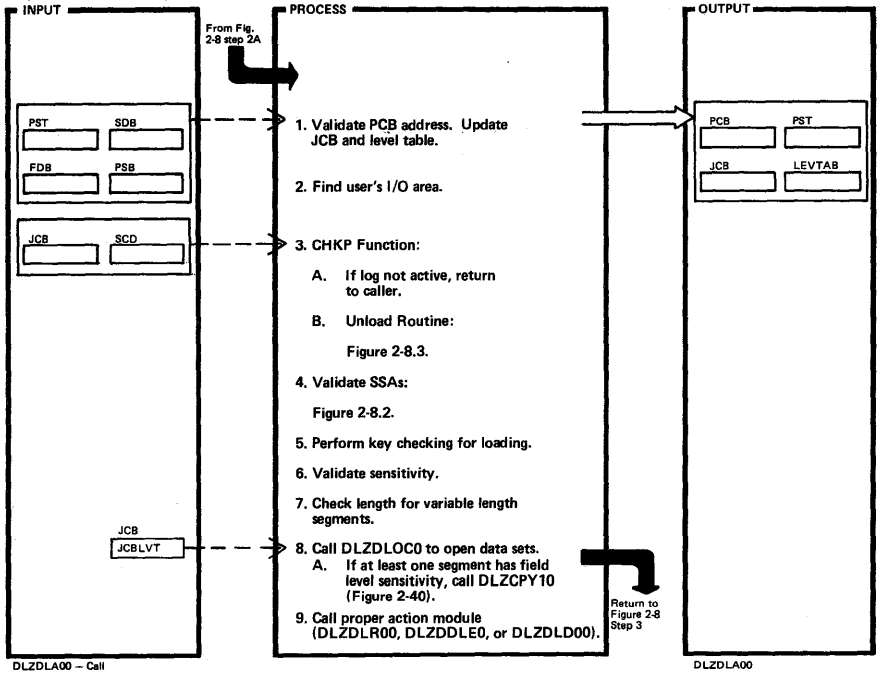
Figure 2-8. Call Analyzer



Extended Description	Routine	Label
Note: DLZDLA00 is called from the program request handler (DLZBNUC0-DLZPRHB0) in a batch system, from (DLZODP-DLZPRH00) in an online system, or if at termination, it is called from either the application program control (DLZRRC00-DLZPCC00) or from online task termination (DLZODP-DLZODP01). It is also called from DLZDXMT0. 2. The function (first parameter in list) is encoded. If no valid function is found, 'AD' status is returned. Normal functions are GU, GN, GHN, GHU, GNP, GHNP, DLET, REPL, ISRT, ASRT, and CHKP. Pseudo functions are GSCD, UNLD, and TERM.		

Extended Description	Routine	Label

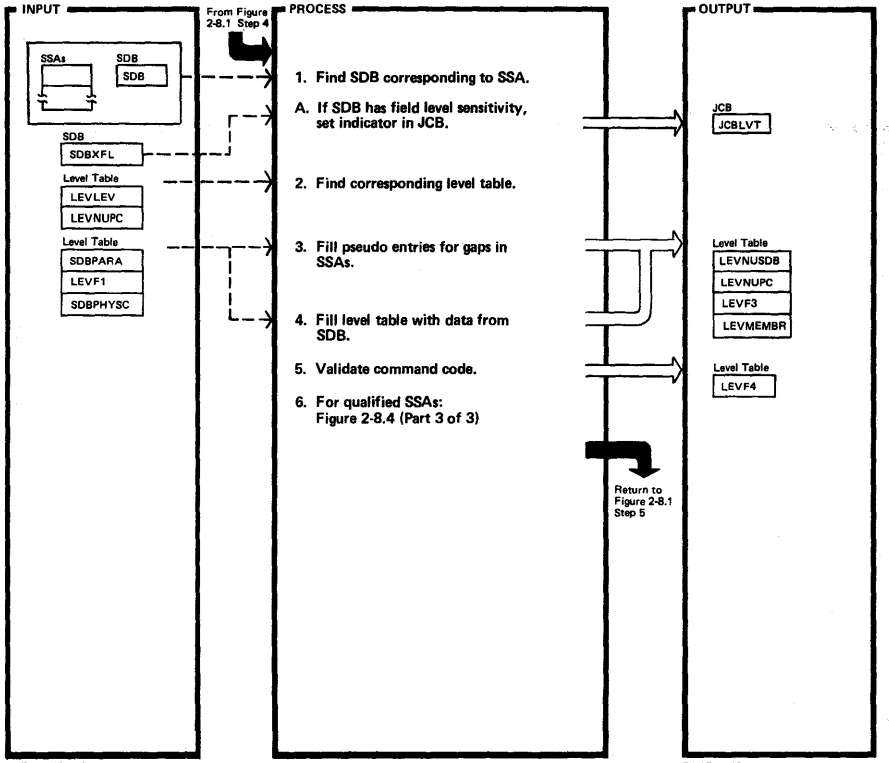
Figure 2-8.1. Call Analyzer – Normal Function



Extended Description	Routine	Label
1. If no valid PCB address is provided, abend code '476' is returned. The JCB and PCB are updated and the second part of the level tables cleared.		TESTPCB VALIDCK2 DBPCBFND GETJCB
2. If no I/O area is provided, 'AB' status code is returned.		
3A. If log is not active, return to caller with 'XH' status code in the PCB. The function call is ignored.		
3B. Purge all buffers.	DLZDLA01	DLBUNLD
4. All SSAs in the call are checked.		SDBLOOP SDBLOOP1
5. Key checking is done for load mode and the last SSA of an ISRT call. For PROCOPT=LS and for HISAM, the root key is compared to the previously loaded root. Status code 'LB' indicates invalid sequence.		LDCHCK
6. Sensitivity checking is done for ISRT, DLET, and REPL calls. Violations return 'AM'. Extra checking is done for DLET and REPL calls, if successful GH call was executed before 'DJ' status.		NOTLOAD7 FSTDATAL ISREPL TSTISRTS

Extended Description	Routine	Label
7. For variable length segments, 2-byte field in the user I/O area is compared to the maximum length and to the key+ keyoffset. If it is greater or smaller, 'V1' status is returned.		DOVLTST
8. When the data base that the PCB references is not open, DLZDLOC0 is called to open all data bases related to this PCB.		ANYSEN
A. If field level sensitivity indicator is set, exit is made to DLZCPY10 to map the user view to the physical view. Only done if ISRT, REPL, or Retrieve (called on behalf of ISRT) action modules will be executed.		
9. For GET calls, DLZDLR00 is called. For DLET/REPL calls, DLZDLD00 is called. For ISRT/ASRT calls in load mode, DLZDDLE0 is called for all segments except for HIDAM root, where DLZDLR00 is called. For ISRT not load mode, DLZDLR00 is called for all segments except HISAM root, where DLZDDLE0 is called.		ACTION

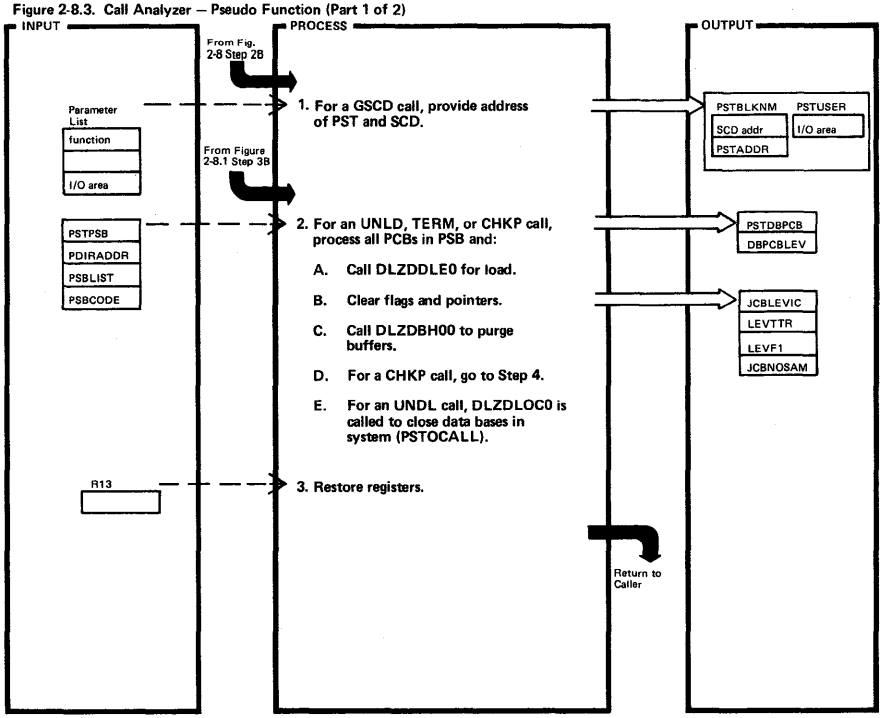
Figure 2-8.2. Call Analyzer – Validate SSAs



Extended Description	Routine	Label
1. When the segment named specified in the SSA cannot be found in the SDB, 'AC' status is returned.		SDBLOOP
A. Flag SDBFLS is on in field SDBXFL if SDB has field level sensitivity. Flag JCBFLS is set on in field JCBLVT to indicate at least one segment in call has field level sensitivity.		SSASDBEQ
2. When a hierarchy error is detected, an 'AC' status is returned.		GETLEV RIGHTLEV
3. The levels corresponding to gaps in the SSAs are filled with data from the previous call. For loading, no parent level may be empty. 'LD' status is returned.		
4. Extra checks are made for DLET and REPL calls. When no GH call was previously made for this SDB, a 'DJ' status is returned.		

Extended Description	Routine	Label
5. Valid command codes are C, D, F, L, N, Q, T, U, V, and X. The status code for invalid command code is 'AJ'. For D call and no path sensitivity, the status code is 'AM'.		NOTDORR

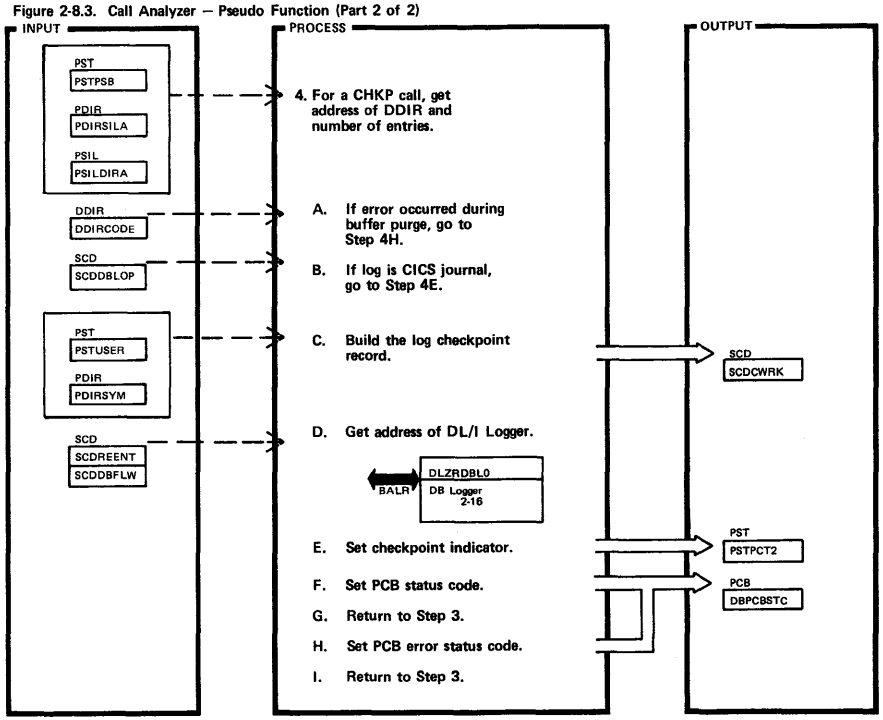
Figure 2-8.3. Call Analyzer – Pseudo Function (Part 1 of 2)



Extended Description	Routine	Label
1. Input to the GSCD call is function and I/O area address. DLZDLA01 puts the SCD and PST addresses in PSTBLKNM.Program request handler moves it to the I/O area.		PSEUDOCA
2. The TERM call is issued in online to end a task. The UNLD call is issued in batch to end the batch program.		DLBUNLD UNLDLOOP
A. If the UNLD call is made for load mode, DLZDDLE0 is called to write the last records for HSAM and HISAM. For HISAM and index data bases, a record is written with FF keys.		
B. Flags and pointers are cleared so that the PSB can be used by another task. If program isolation is active, clear all enqueue indicators in all level table entries.		
C. All user buffers are written to the data base now. RSTBLKNM, DMBNM, and ACBNM are cleared. PSTPGUSR flag of PSTFNCTN is set.		

Extended Description	Routine	Label
3. If an error occurs during the purge of the buffers, an 'XD' status code is returned in the PCB.		

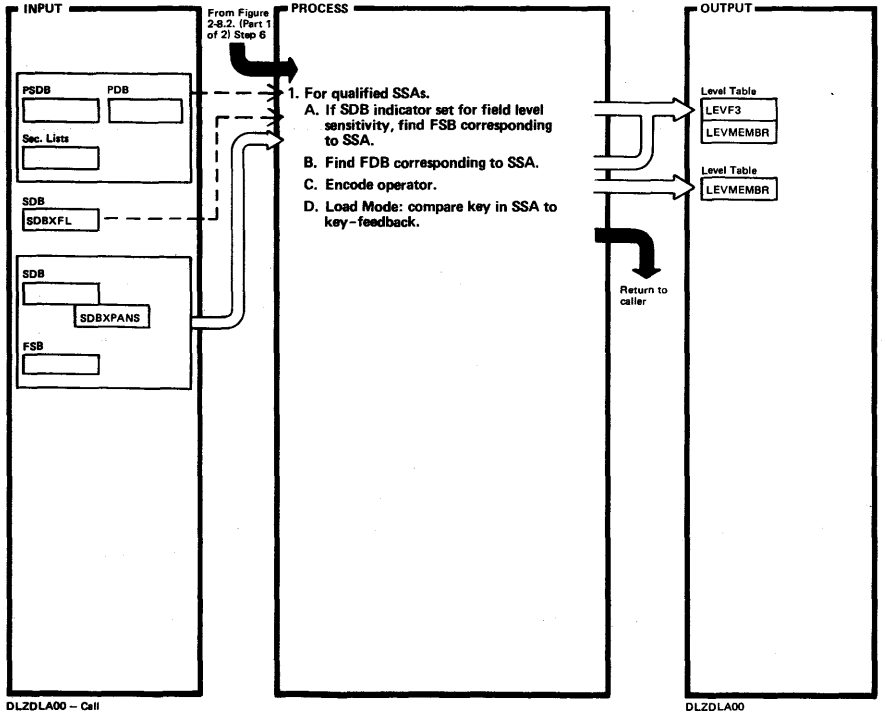
Figure 2-8.3. Call Analyzer – Pseudo Function (Part 2 of 2)



Extended Description	Routine	Label
4. A. All DDIR entries are scanned to see if an error occurred (bit DDIRNOSE X'04' in byte DDIRCODE set on) during purge of the buffers.		DDIRCHK
B. If the log is the CICS journal, checkpoint record is not written, but a CICS synch point is.		DDIRCHK1
D. The DL/I Logger is entered twice; 1st to move the checkpoint record to its buffer, and 2nd to force-write the checkpoint record.		
E. On return from the logger, the checkpoint indicator (bit PSTCHKP X'04' in byte PSTPCT2 is set on) to notify the program request handler to issue the checkpoint message.		BYPASSCK

Extended Description	Routine	Label
F. Status code of 'blanks' is set in the PCB indicating successful completion of the CHKP call.		
H. Status code of 'XD' is returned in the PCB indicating an error occurred during checkpoint processing.		DDIRER

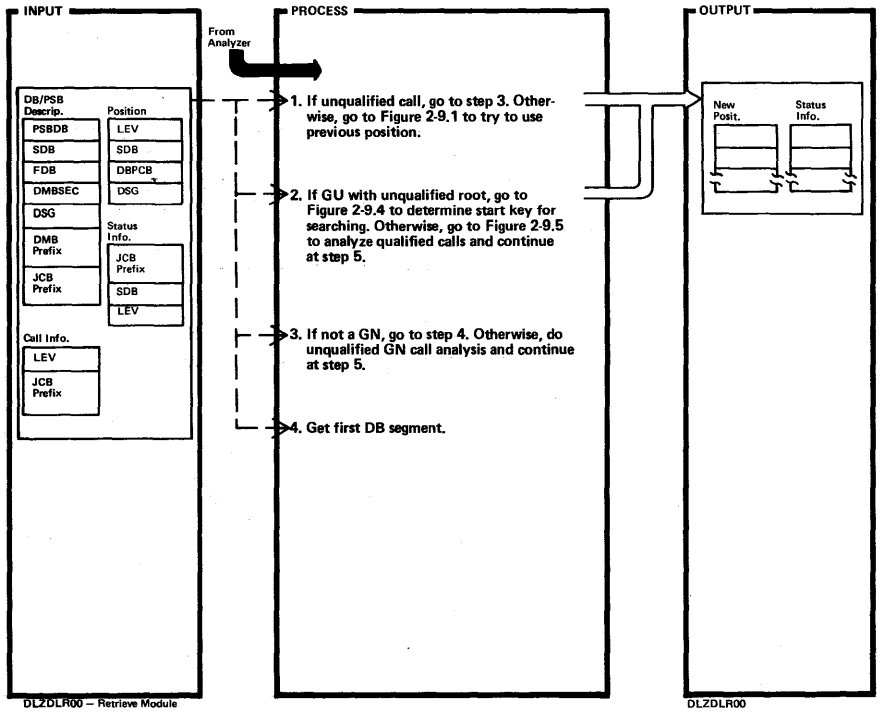
Figure 2-8.4. Call Analyzer – Validate Qualified SSAs



Extended Description	Routine	Label
1. For errors in qualification, format 'AJ' status is returned.		
A. Flag SD3FLS is on in field SDBXFL if SDB has field level sensitivity. If an FSB is not found, or if the FSB is not marked as an allowable field, status code 'AK' is returned in PCB.		
B. Valid field names are any normal field of the segment, the XDFLD name (if the secondary processing sequence is used). For a concatenated segment field, names of the logical child and the destination parent are valid. 'AK' status for invalid field name. 'AC' status if /CK or /SX is used.		NXTBOOL PDBEQUAL
C. Invalid operator returns status code 'AJ'.		CODES ROHIT
D. If qualified SSAs are specified for loading, the key has to correspond to the key-feedback area, otherwise 'LD' status code is returned.		

Extended Description	Routine	Label

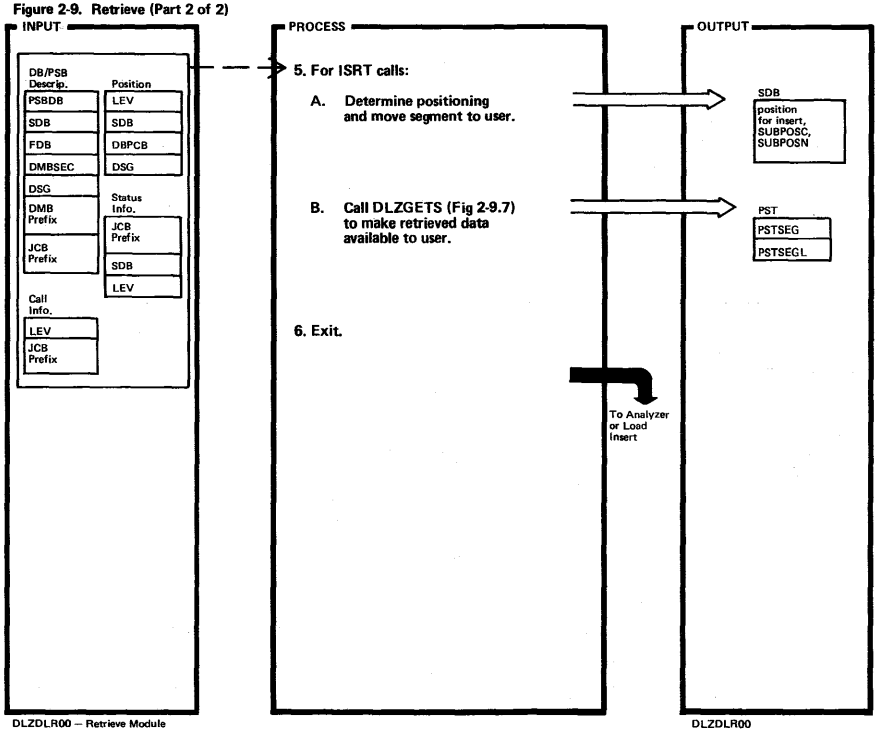
Figure 2-9. Retrieve (part 1 of 2)



Extended Description	Routine	Label
1. I/O information: • The Position block includes RBA of segment (HD) or lrec (HS), RBA of previous and next positions (HD), offset to segment from begin lrec (HS), concatenated key, level, block number (HSAM), and block number and RAP number of current RAP (HDAM). RAP = root anchor point. • The DB/PSB Description block includes segment and data set descriptions, data base specifications, sensitivity, and HDAM randomizing facility. • The Status Information block includes prior status codes, segment status, and (for output) pseudo abends (801 and 800). • The Call Information block includes SSA and call type. • Processing starts with initialization. Level of previous call stored in LASTLEV.		LTWSSA

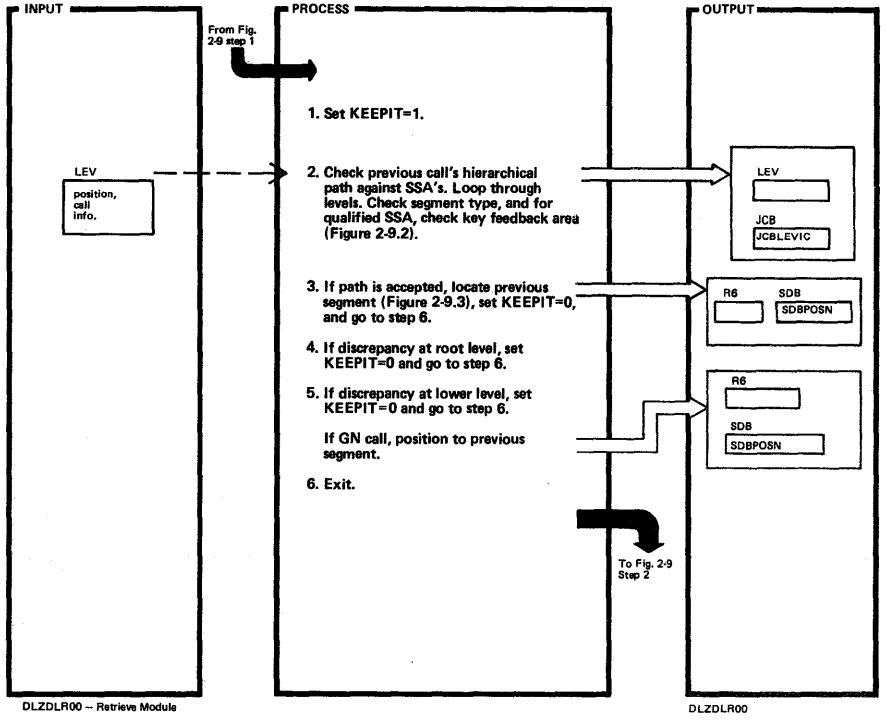
Extended Description	Routine	Label
2.		XLTFINDR
3.		MTNOSSA NOSSA

Figure 2-9. Retrieve (Part 2 of 2)



Extended Description	Routine	Label
5. DLZSSA is called, if necessary, to find insert position for key. Control is then passed to DLZISRT to prepare position information in the SDB for INSERT. Return is to DLZGETS (Figure 2-9.6). PSTSEG is address of data, PSTSEGL gives its length. For IRST calls, DLZGETS does only housekeeping (no data moving). DLZGETS will pass control to DLZRETN and DLZDLR1 to exit. For a segment with logical relationship, DLZGETS will call DLZLOGR for data move/insert positioning. 6. If call type is GET, go to Analyzer. If it is ISRT, go to Load/Insert.		RETURNIE NOTEOD ARETURNA

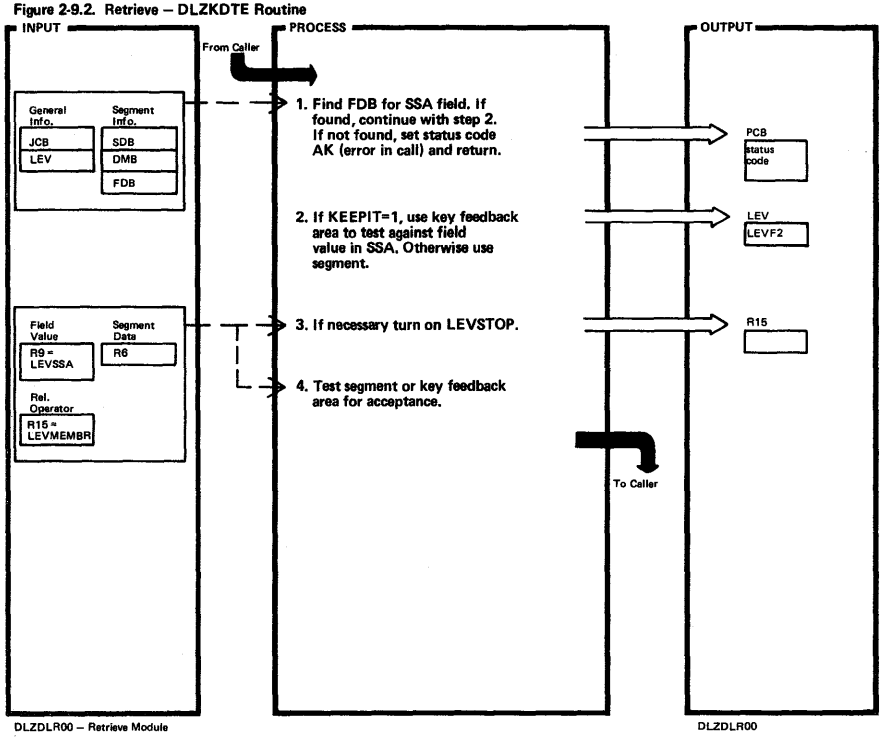
Figure 2-9.1. Retrieve – DLZLTW Routine



Extended Description	Routine	Label
1. KEEPIT=1 indicates: try to use previous position. KEEPIT=0 indicates: DLZLTW has been left. Other values have special meanings. (Entry point when R15 = 0.)		LTWSSA
2. DLZKDTE is invoked via DLZSSA which is called by return to DLZDLR0 and back to DLZLTW. Logically, this is part of DLZLTW as indicated by KEEPIT=1.		
Qualified SSA test: After entering several routines, return to DLZLTW entry LTWSSACA, LTWSSAF, or LTWSSAG.		LTWSSAQ
Lowest level found valid is stored in JCBLEVIC.		LTWSSACA
3. Set code for exit: Entry UNQLA in DLZSSA for GU or ISRT, entry SSAEVALH for GN.		
DLZPCHK loads buffer location of previous segment into register 6 (except for HD or GN calls) and, for HD, loads available SUBPOSN positions.		

Extended Description	Routine	Label
5. Set exit code for entry SSAEVALL in DLZSSA.		

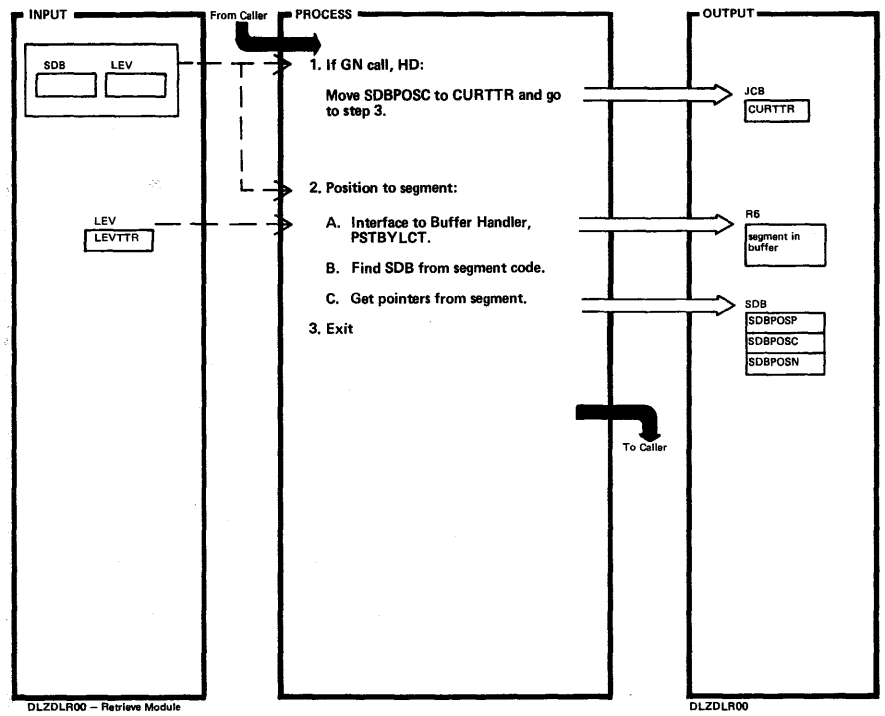
Figure 2-9.2. Retrieve – DLZKDTE Routine



Extended Description	Routine	Label
1.		KDTESTI KDTESTK
2. If logical relationship, build concatenated segment. If variable length, build data.	DLZKDTL DLZVLRT	KDTESTER
3. If qualification is on key, relational operator is greater than or equal to, and key is less than or equal to, SSA.		KDTESTHA
4. If accepted, R15=0, otherwise, R15=4.		KDTESTE

Extended Description	Routine	Label

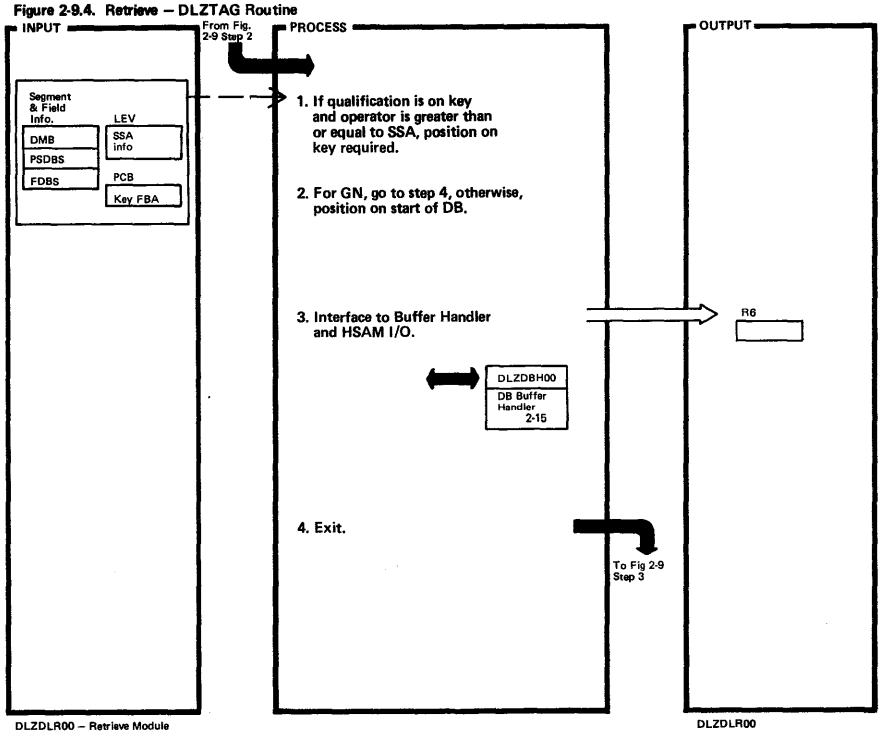
Figure 2-9.3. Retrieve – DLZPCHK Routine



Extended Description	Routine	Label
2. For HSAM, more than 1 PCB: restore position. For HISAM: take care of control interval splits.		POSCHKA POSCHKA2
B. If not found (segment not sensitive), turn on LEVDLET and go to step 3.		
C. For HS, relational record number and offset to SDBPOSC. SDBPOSN already posted by DLZSETL.		
For HD, post twin pointers.	DLZPSTN	
Clear dependent positions (SUBPOSP, SDBPOSC, and SDBPOSN). For HD, post child pointers.	DLZPSTA	
For HD logical relation with inverted structure, post child pointers. Subroutine called by DLZPSTA.	DLZAPST	
Clear SDBPOSP, SDBPOSC, and SDBPOSN in preceding sibling SDBs unless multi-processing.	DLZPOSA	

Extended Description	Routine	Label

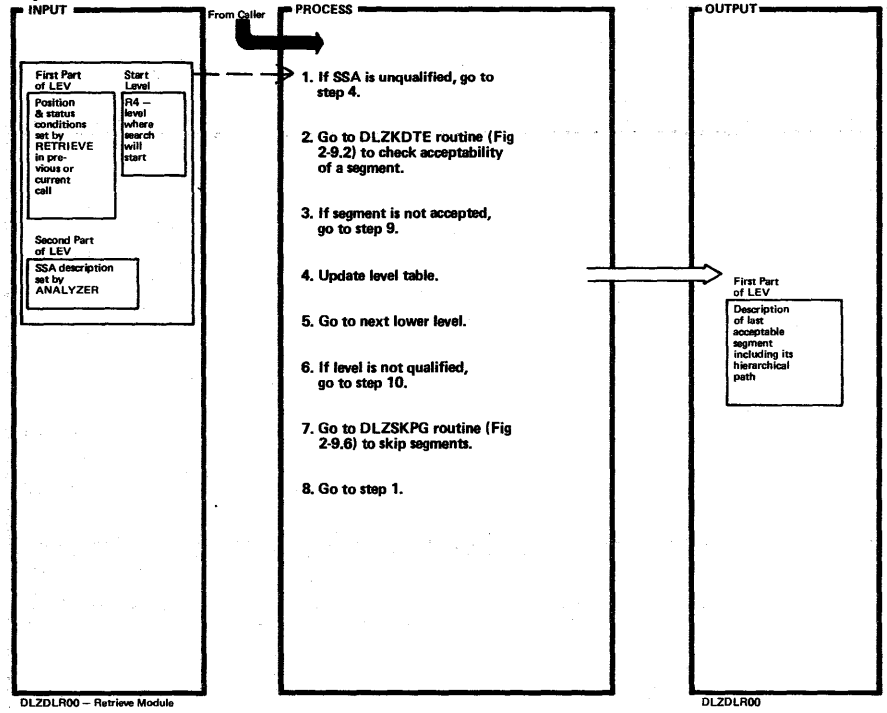
Figure 2-9.4. Retrieve – DLZTAG Routine



Extended Description	Routine	Label
1. Set code PSTSTLEQ for DLZSETL. Set exit code for entry SSAEVAL in DLZSSA.		MTWISSA
2. Set code PSTSTLBG for DLZSETL. Set exit code for entry SSAEVAL or SSAEVALM in DLZSSA.		NOLL
For GN, set exit code for entry SSAEVALM in DLZSSA.		KPURTC
3. DLZSETL branches to subroutines according to DB organization. R6 points to segment in buffer pool.		

Extended Description	Routine	Label

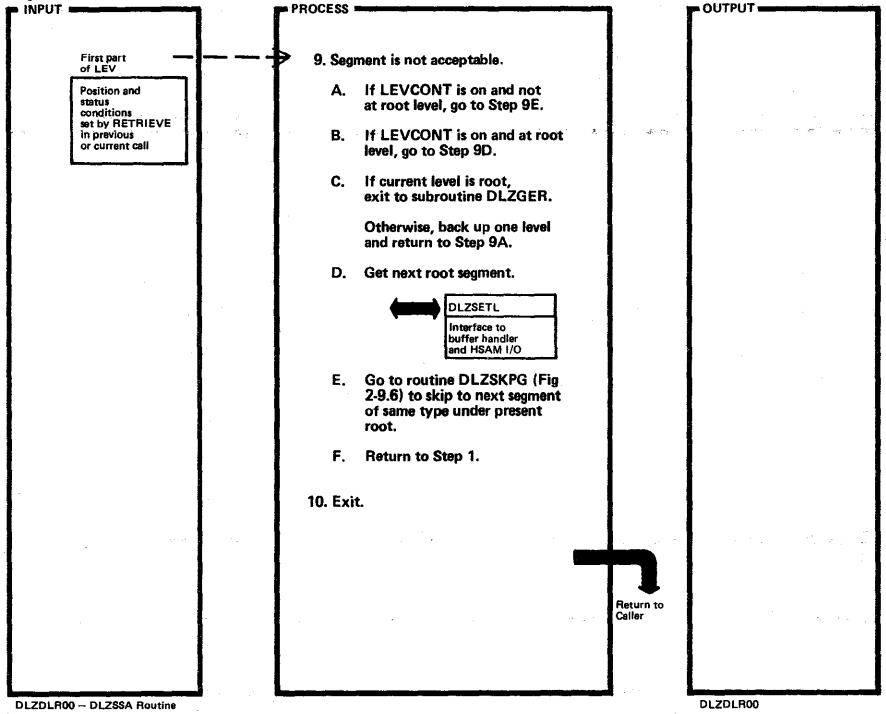
Figure 2-9.5. Retrieve – DLZSSA Routine (Part 1 of 2)



Extended Description	Routine	Label
2. With a Boolean SSA all Boolean statements connected by AND operators are considered a 'set' of qualification statements. An OR operator between two qualification statements begins a new set of qualification statements. A set can consist of one or more qualification statements. To satisfy any set, the segment must satisfy all statements within the set.		
6. Prepare input (segment type, etc.) before entering the central DLZSKPG routine.	DLZSKPG	SKIPGENS

Extended Description	Routine	Label

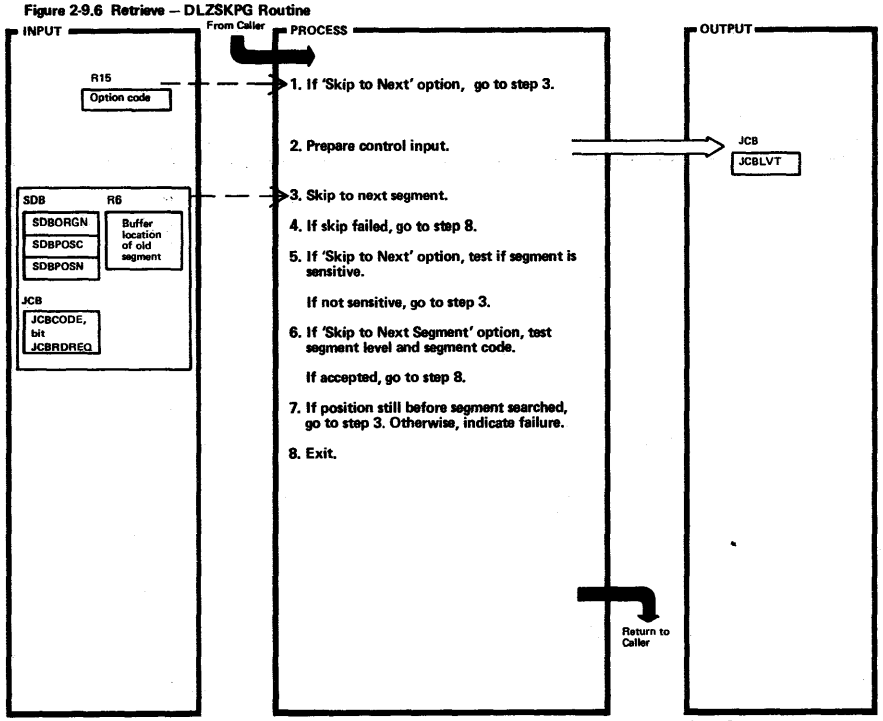
Figure 2-9.5. Retrieve – DLZSSA Routine (Part 2 of 2)



Extended Description	Routine	Label

Extended Description	Routine	Label

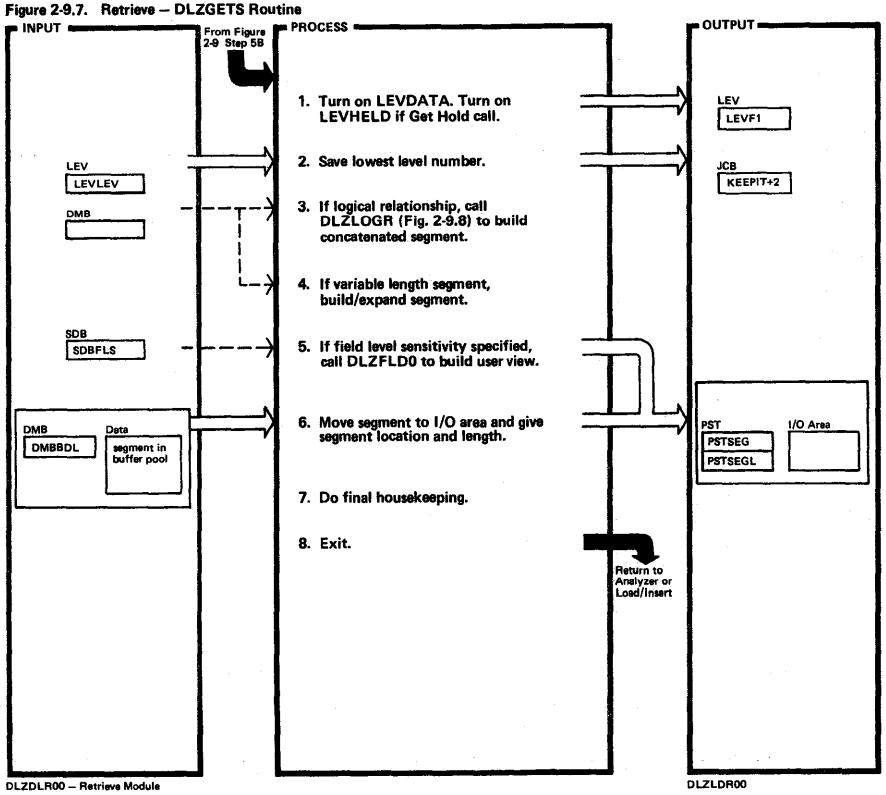
Figure 2-9.6 Retrieve – DLZSKPG Routine



Extended Description	Routine	Label
1. Option is 'Skip to Next Segment' if R15 is positive. If R15 is negative, the option is 'Skip to Specified Segment'.		
2. If JCBLVT = X'02', require segment code, segment level in physical DB, and parentage level.		SKIPGENS
3. For JCBRDREQ off, current segment is examined first. DLZSKPS calls general skip routine DLZSKPE, which calls specific skip routines: For HS, DLZSKPD. For HD using SUBPOSN, DLZSTLA. In some cases (HS, skip to first child of current segment), DLZSKPD is called directly from DLZSKPS.		SKIPGEN
4. End of ESDS chain reached for HISAM.		

Extended Description	Routine	Label
7. If segment code of segment found is not larger than that required.		

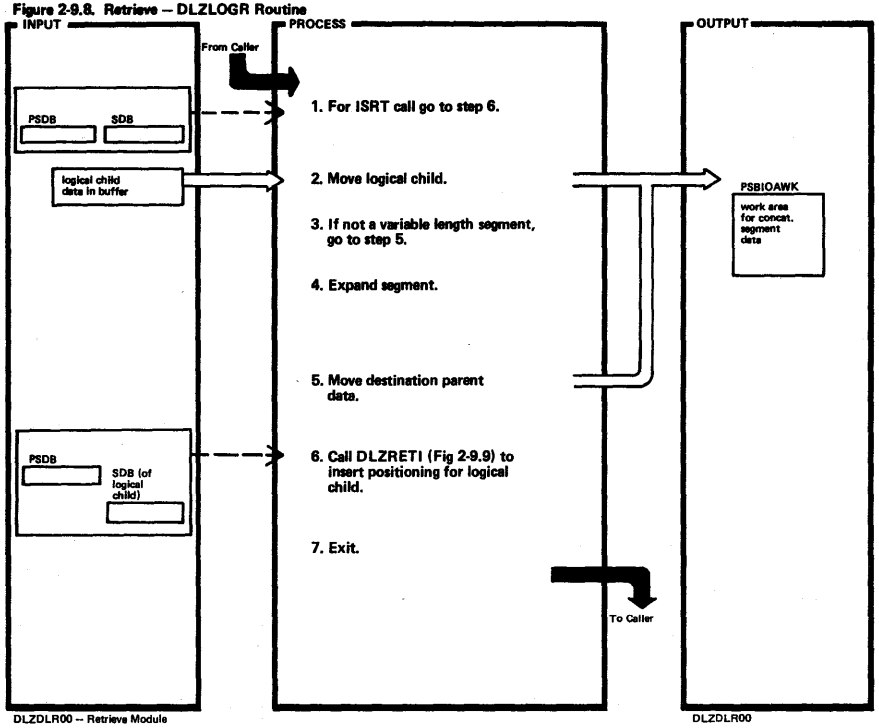
Figure 2-9.7. Retrieve – DLZGETS Routine



Extended Description	Routine	Label
6. If batch, only one task active, or no field level sensitivity specified, and if segment is fixed length and not involved in logical relationship, segment data is not moved (left in buffer pool). The same is true for Insert calls. For a path call (*D command), data has already been moved in DLZUPDT and is not moved here. Address of I/O area is PSBIOAWK. 7. For Insert calls, return is to Load/Insert.		

Extended Description	Routine	Label

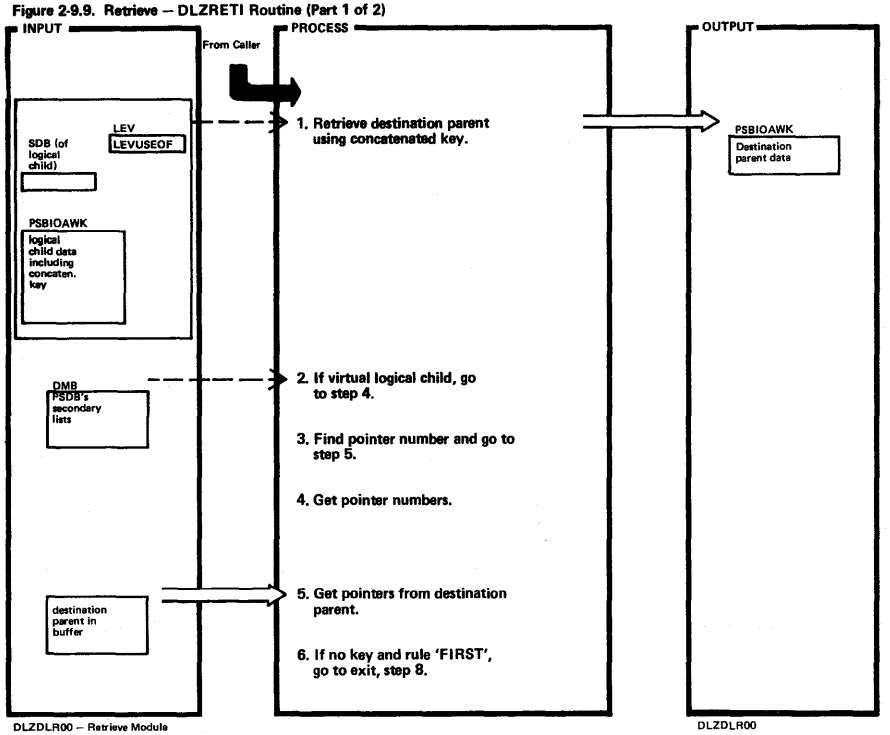
Figure 2-9.8. Retrieve – DLZLOGR Routine



Extended Description	Routine	Label
1. Destination parent concatenated key and logical child data. 6. Destination parent exists. Position segment on alternate twin chain.	DLZYENT	

Extended Description	Routine	Label

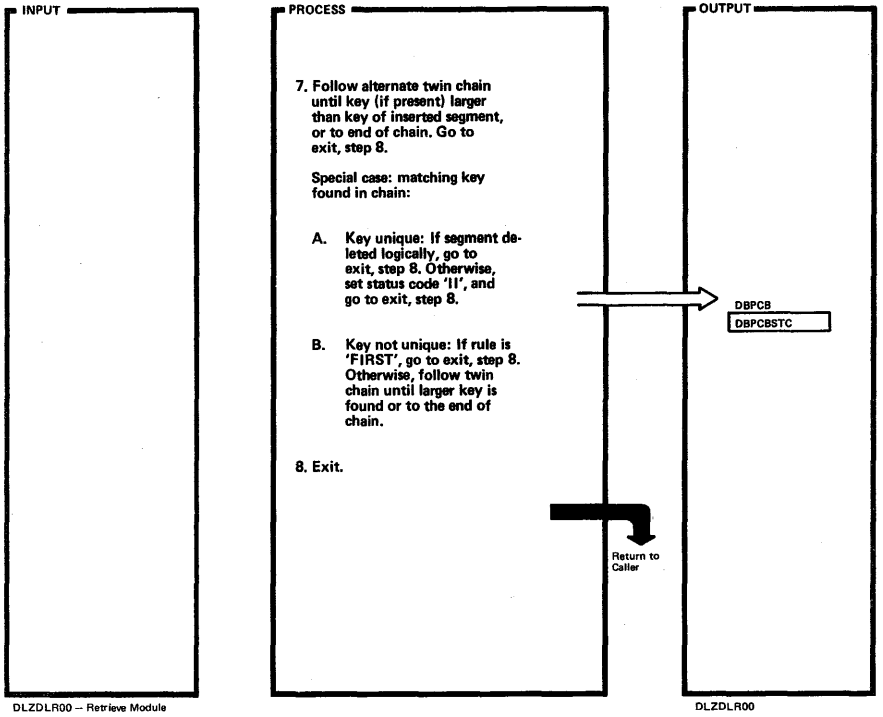
Figure 2-9.9. Retrieve – DLZRETI Routine (Part 1 of 2)



Extended Description	Routine	Label
1. LEVUSEOF indicates offset of key for this level in concatenated key.		
Destination parent data is stored behind concatenated key and logical child.	DLZRETK	
2. For virtual logical child (insert through logical path), positioning on physical twin chain is required.		
3. Find logical twin pointer number. Find logical child first and last pointers in logical parent. Find FDB for key of logical child, if present.		RETISRTF
4. Find physical twin pointer number. Find physical child first and last pointers in parent. Find FDB for key of virtual logical child, if present.		RETISRTR
Logical twin key is moved to key feedback area.	DLZUPDL	

Extended Description	Routine	Label

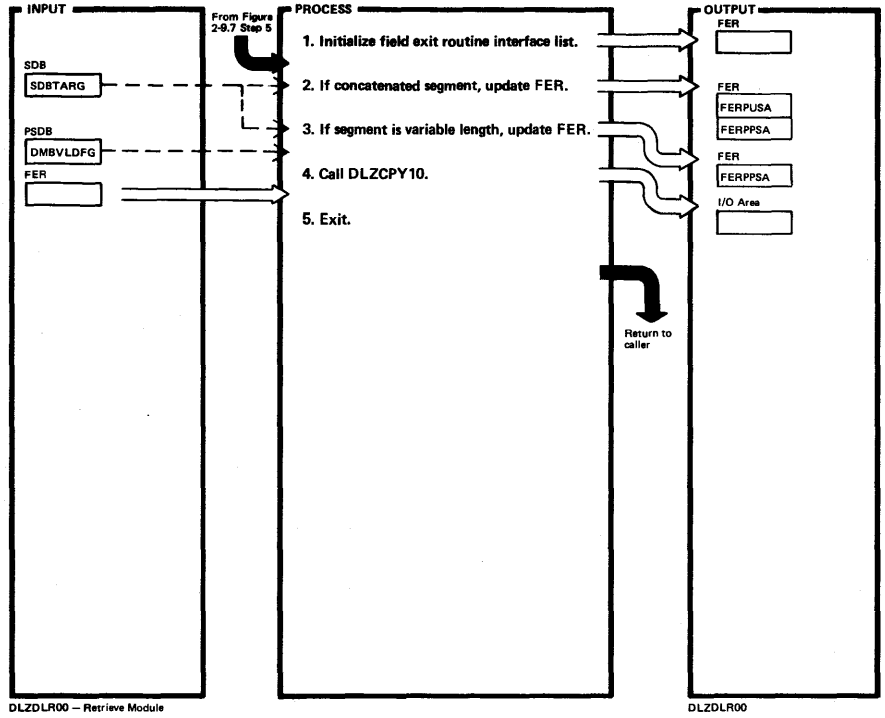
Figure 2-9.9. Retrieve – DLZRETI Routine (Part 2 of 2)



Extended Description	Routine	Label
7. Alternate means: Logical twin chain if entering from physical path, physical if entering from logical path.		RETISRTL
If sequence field is in destination parent concatenated key (possible only for virtual logical child), the virtual area (physical parent, concatenated key, and logical child data) is built in PSBIOAWK, calling routines DLZYSTC and DLZMOVA. As an indication, the first byte of PSTWRKT5 is set to X'FF'.		RETIVK
A. For logically deleted segment, turn on bit JCBDEFDL in field JCBCODE.		

Extended Description	Routine	Label

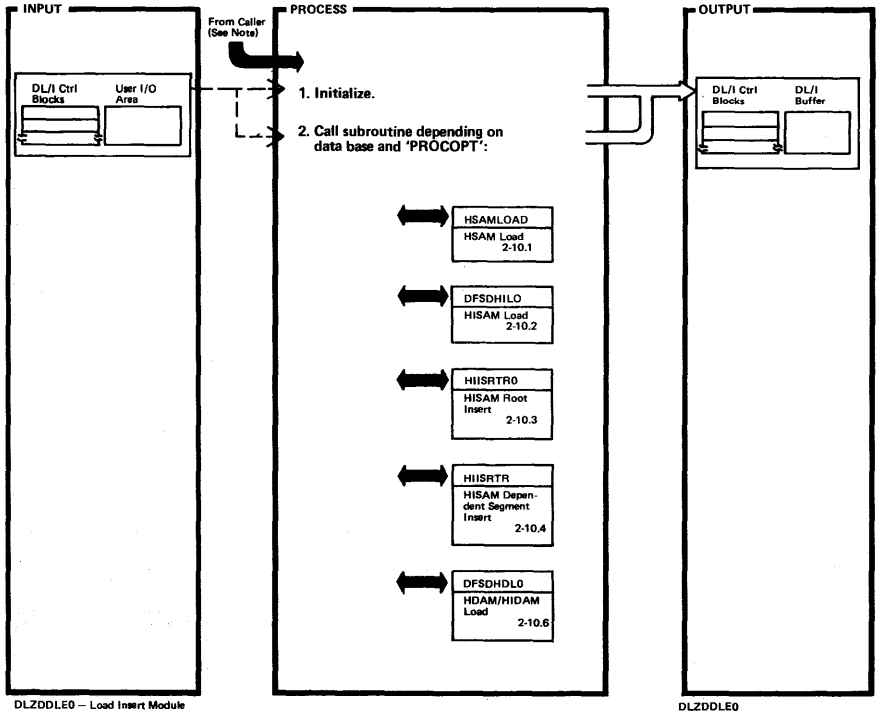
Figure 2-9.10. Retrieve – DLZFLD0 Subroutine



Extended Description	Routine	Label
1. FER is located at address in PSBNDXWK.		
2. The concatenated segment has been built in PSBIOAWK and the user's view must be constructed in another area (PSBXIOWK). For path calls (*D command), the user's view will be moved back to PSBIOAWK after conversion to the user's view.		FLDCSEG
3. Fields may be defined that are outside the physical segment, so they must be defaulted so conversion errors do not occur. If such fields do exist, the segment is moved to PSBXIOWK and the defaults provided.		FLDVAR
4. If a conversion error is detected, an immediate exit to the Call Analyzer is taken.		FLDERR

Extended Description	Routine	Label

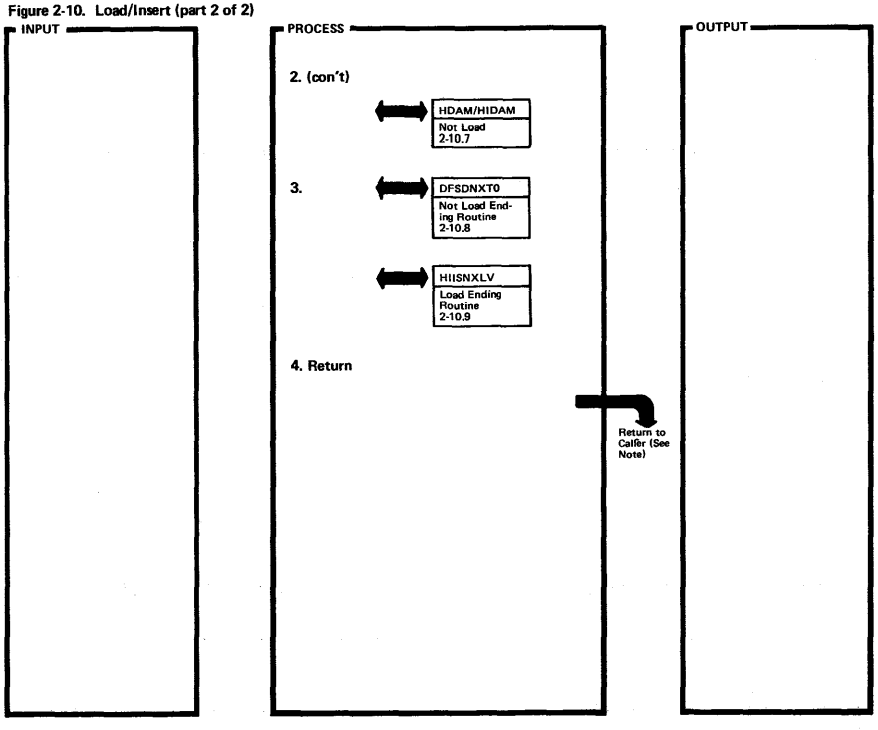
Figure 2-10. Load/Insert (Part 1 of 2)



Extended Description	Routine	Label
Note: DLZDDLE0 is called from DLZDLA00 (Call Analyzer) or from DLZDLR00 (Retrieve Module).		

Extended Description	Routine	Label

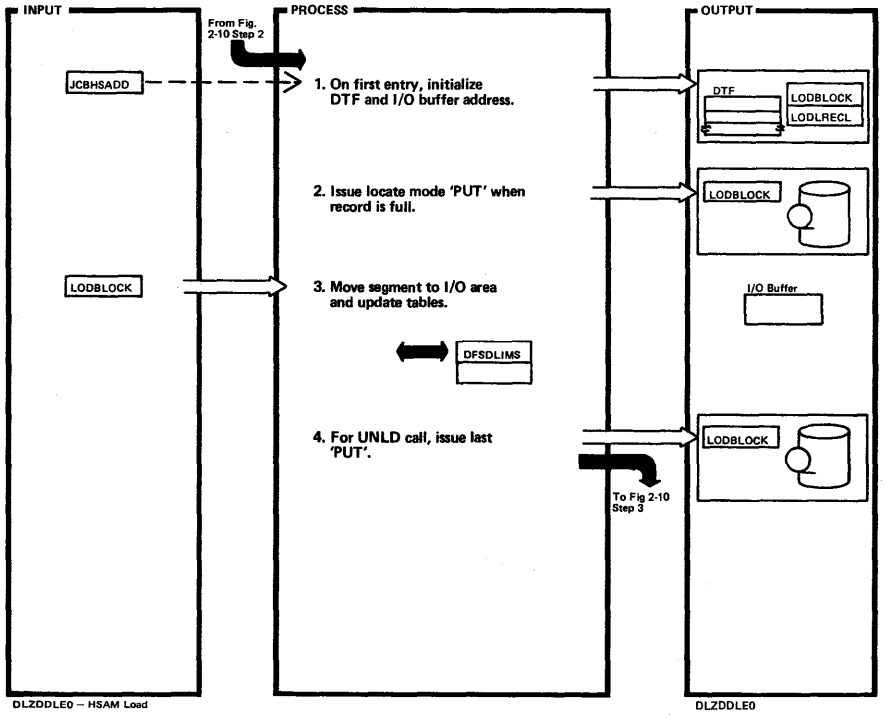
Figure 2-10. Load/Insert (part 2 of 2)



Extended Description	Routine	Label
Note: DLZDDLE0 is called from DLZDLA00 (Call Analyzer) or from DLZDLR00 (Retrieve Module).		

Extended Description	Routine	Label

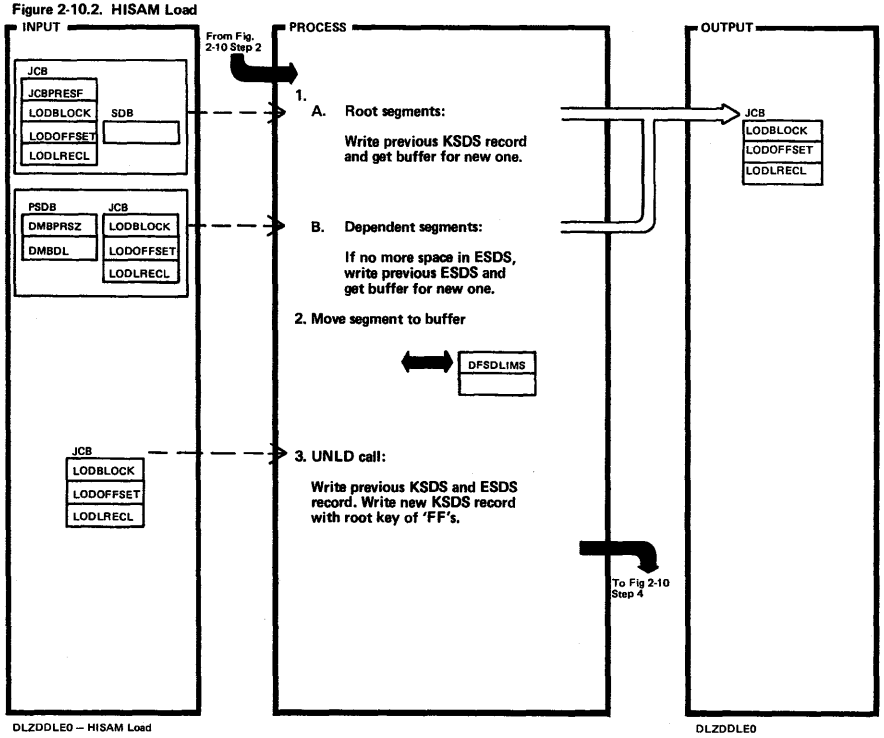
Figure 2-10.1. HSAM Load



Extended Description	Routine	Label
1. DLZDLOC0 stores the I/O area address in the JCB. It is updated with every 'PUT'. The record size is taken from the DTF and the error exit address in the DTF is updated.		HSAMFRST

Extended Description	Routine	Label

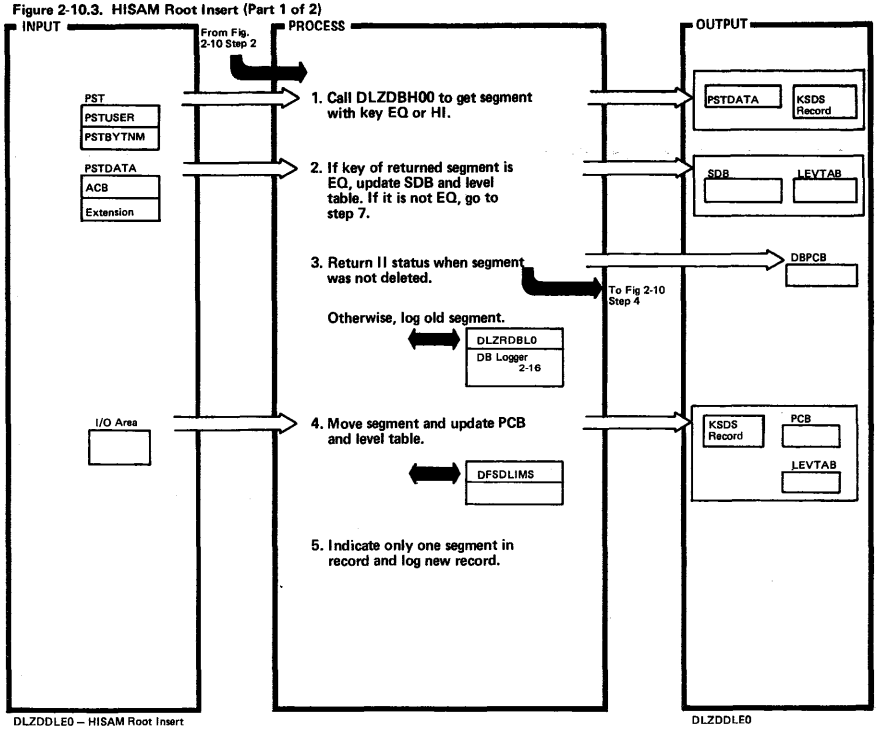
Figure 2-10.2. HISAM Load



Extended Description	Routine	Label
1. A. Record length, buffer address, and offset into buffer is stored in the JCB and passed from call to call. When a call for a new root segment is made, the buffer handler (DLZDBH00) is called to write the previous KSDS record and to get buffer space for the new one.	WRITEOLD	HISIMPLS
B. If there is space left in the ESDS records, continue with step 2. Otherwise, the RBA of the next ESDS record is calculated, the pointer of the current ESDS record updated, and the buffer handler called to write the ESDS. Another call to DLZDBH00 is made to get buffer space for a new ESDS record.	NEWRBA	NEEDOSAM
ABEND 855 is given if VSAM returns an RBA different from the calculated one.		CATERROR

Extended Description	Routine	Label
2. The segment is moved, the PCB key fed back, and the level table updated.		

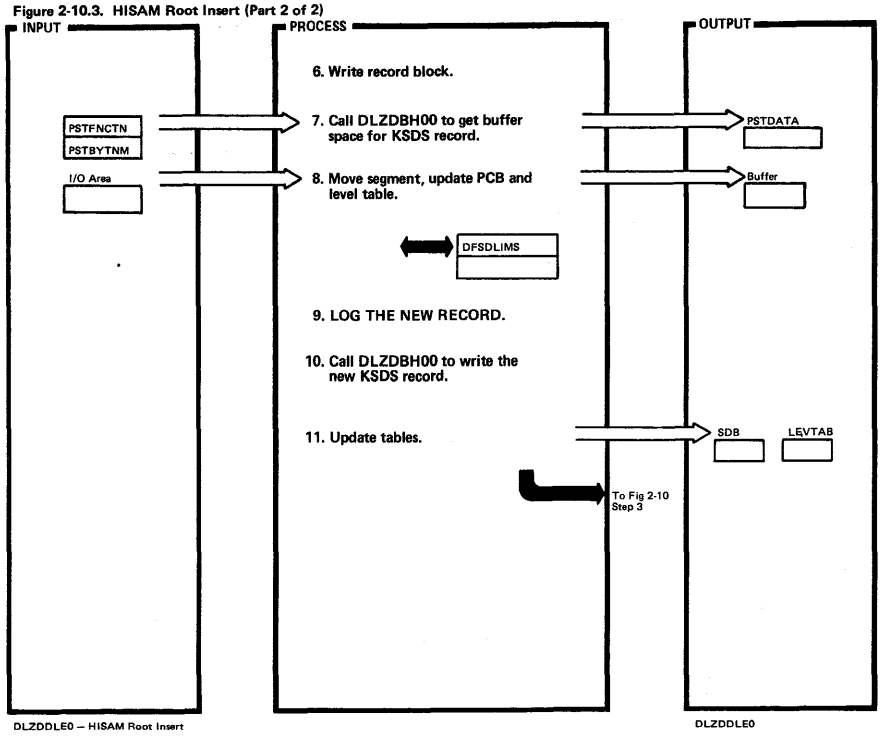
Figure 2-10.3. HISAM Root Insert (Part 1 of 2)



Extended Description	Routine	Label
1. The buffer handler is called with 'PSTSTLEQ' to get a segment with key equal or higher than the one to be inserted.	GOTOFUNC	HIISRTR0
2. If the key returned is higher, processing continues with step 7.		ISIS015
3. When the delete flag is not on in the segment returned, status code 'II' is returned to the caller.		
The data base log module is called to log the old KSDS record.		ISSDELET
4. The new root segment is moved to the KSDS record. The PCB key feedback area and level table are updated.		
5. The pointer to the ESDS record is cleared and '00' moved to the KSDS record behind the root segment. The data base log module is called to log the new KSDS record.		

Extended Description	Routine	Label

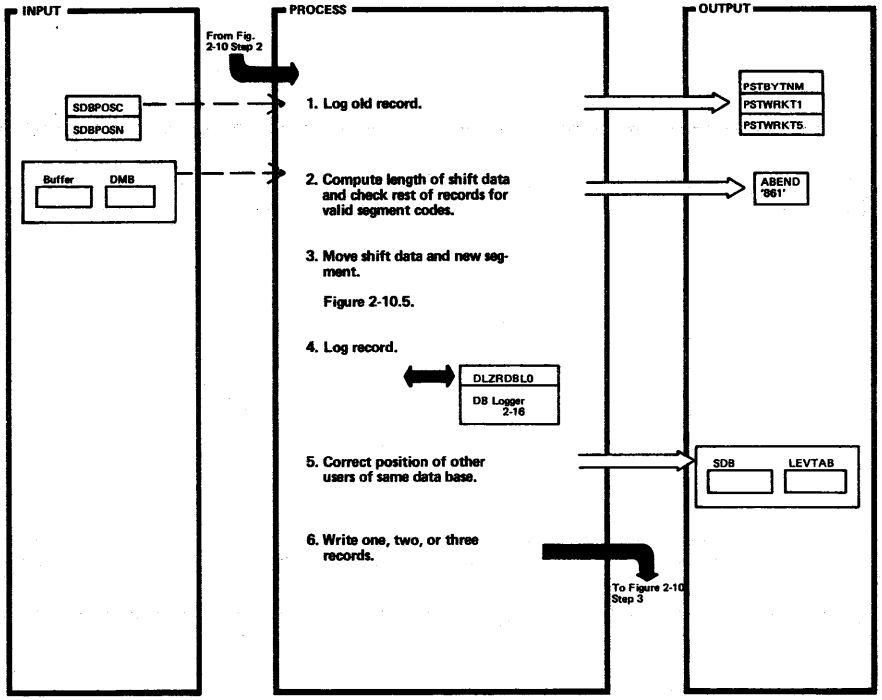
Figure 2-10.3. HISAM Root Insert (Part 2 of 2)



Extended Description	Routine	Label
6. The buffer handler is called to write the KSDS record back (PSTBFALT).	GOTOFUNC	
7. The buffer handler is called (PSTGBSPC) to get buffer space for one KSDS record.	GOTOFUNC	ISNOTEQ
10. PSTPUTKY is used to write the new KSDS record.	GOTOFUNC	ISSIMPL1

Extended Description	Routine	Label

Figure 2-10.4. HISAM Dependent Segment Insert



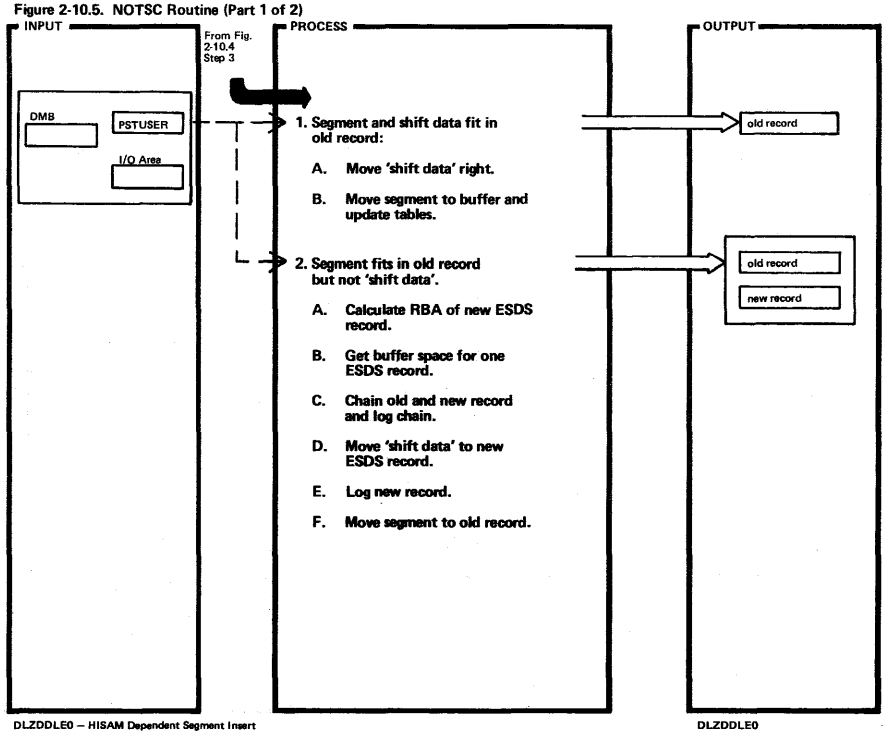
DLZDDLE0 – HISAM Dependent Segment Insert

DLZDDLE0

Extended Description	Routine	Label
1. DLZDLR00 has located within a KSDS or ESDS record, where the new segment has to be inserted. The old record is logged from insert point on to the right.		HIISRTR
2. The record is inspected from the insert point to the right. The segment code is checked and the length of the remaining segments is added to give the 'shift data'.		HAVELREC COMPSHFT ABEND861
4. Log the old record from insert point to the right.	DLZDLBL0	LOGLEVCO
5. SDBs and level tables of other PCBs that are positioned in the same record are updated to show the shifted position of the segments.		INSADJUS
6. DLZDBH00 is called to write back the old record and to write one or two new ESDS records.	GOTOFUNC	KNNDONEX

Extended Description	Routine	Label

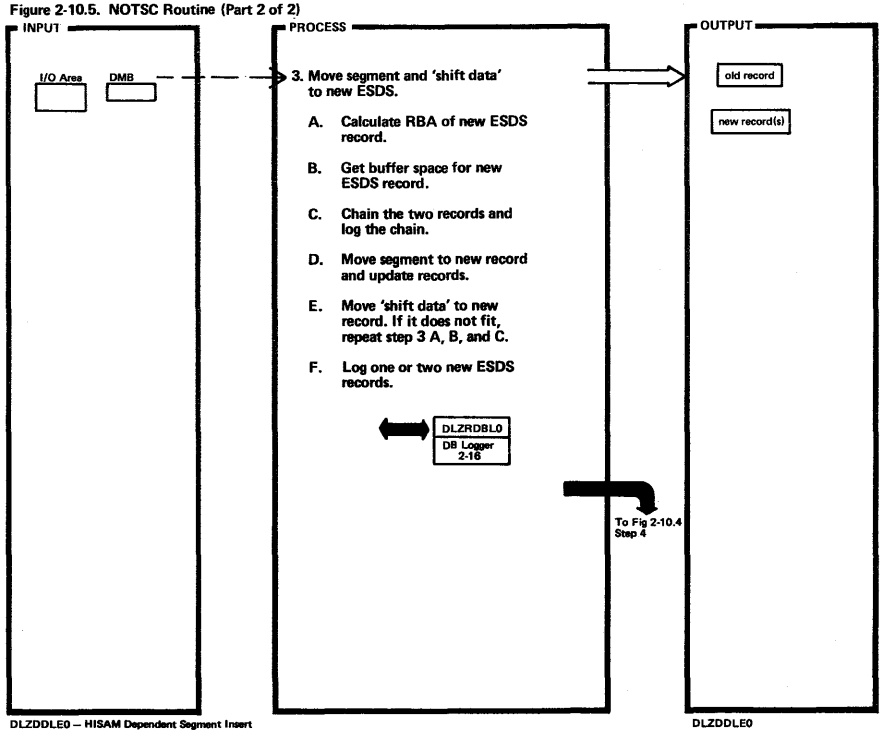
Figure 2-10.5. NOTSC Routine (Part 1 of 2)



Extended Description	Routine	Label
1. When both the new segment and the shift data fit in the old record, the shift data is moved right by segment length. The segment is moved to the record and the PCB and level table are updated.	DFSDLIMS	OVERLAPL
2. A new ESDS record has to be built.	GETNESDS LOGCHAIN COMMOVE LOGNEWOS DFSDLIMS	SEGTOOLD

Extended Description	Routine	Label

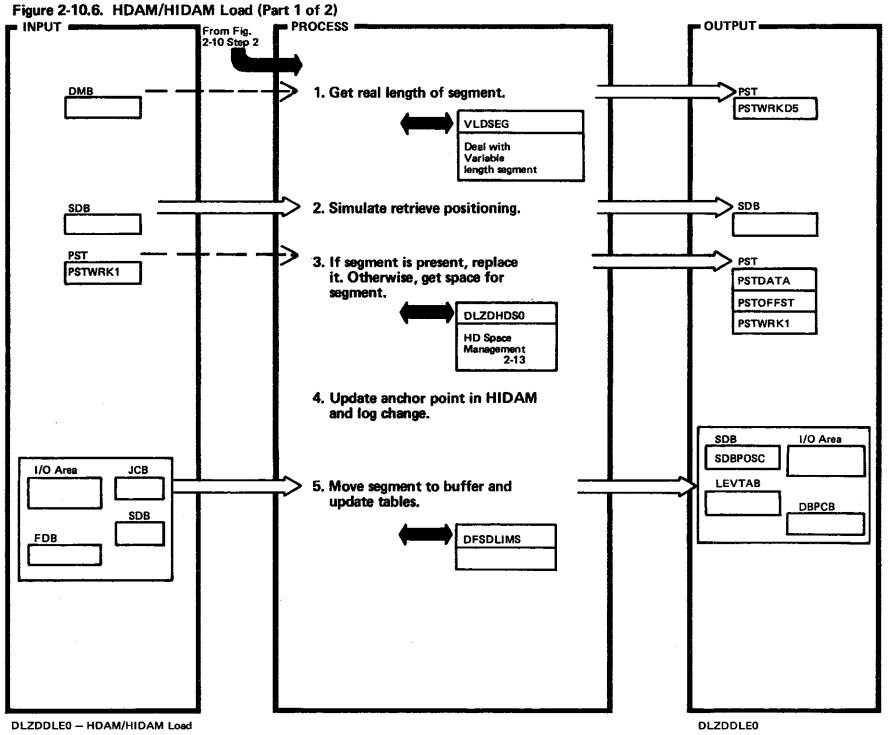
Figure 2-10.5. NOTSC Routine (Part 2 of 2)



Extended Description	Routine	Label
3. Neither segment or 'shift data' fit in the old record. A new record has to be built. If it does not have room for the segment and 'shift data', another new ESDS record has to be built. The records are chained and logged.	GETNESDS LOGCHAIN DFSDLIMS COMMOVE LOGNEWOS NEWRBA GOTOFUNC	SEGTONEW SHIFT00 SHIFTOS2 LOGLEVCO

Extended Description	Routine	Label

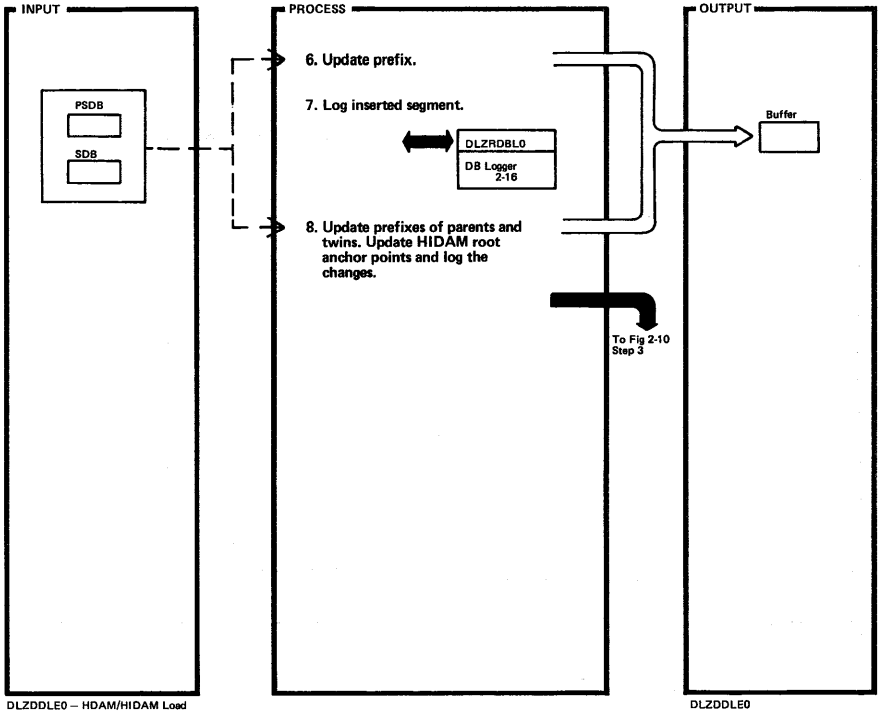
Figure 2-10.6. HDAM/HIDAM Load (Part 1 of 2)



Extended Description	Routine	Label
1. The subroutine VLDSEG takes the length from the PSDB for fixed length segments and from the user's I/O area for variable length segments. The compaction exit routine is called, if it exists.	VLDSEG	DFSDHDLO
ABEND '863' is given when the compaction routine changes the sequence field.		ABEND863
2. For HIDAM root segments, DLZDLR00 did the positioning. For other segments, it is done here.		
3. Space management is called to get space for the segment. If the segment was deleted in one path only, i.e. it was not removed by DLZDLD00, the segment is replaced with the new data.	TOSPACE	GETSPACE SPACEOUT
4. HIDAM root segments without PTB pointers are chained off the anchor point in chronological sequence.		SPACEOK
5. Move segment to buffer, update PCB key feedback, and update level table.	DFSDLIMS	ANCHOROK

Extended Description	Routine	Label

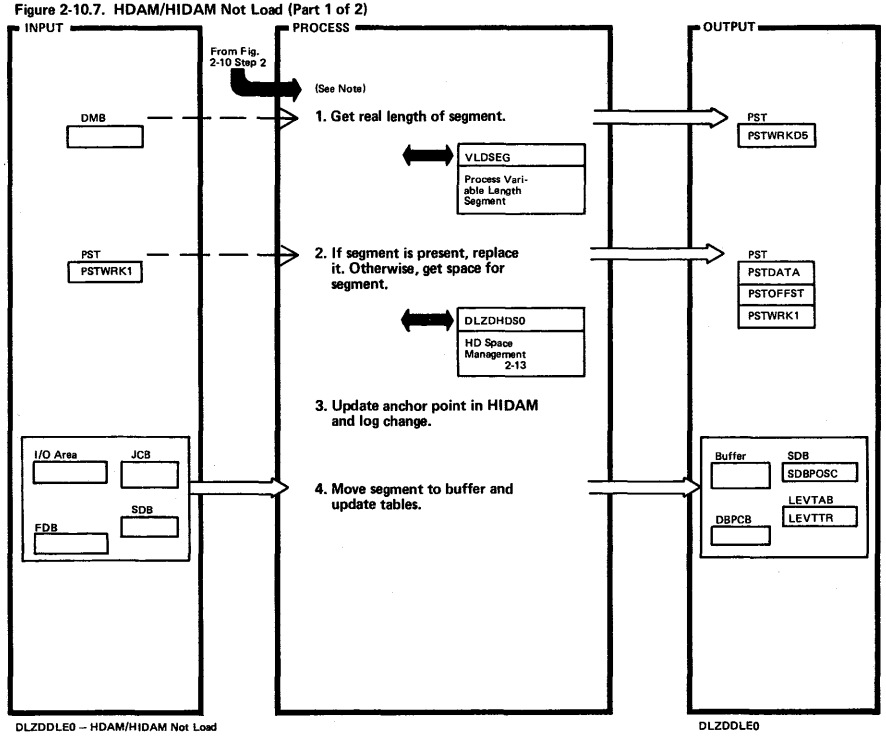
Figure 2-10.6. HDAM/HIDAM Load (Part 2 of 2)



Extended Description	Routine	Label
6. The prefix of the segment is updated: physical twin pointers, physical parent pointer, logical parent pointer, and logical twin pointers.		
7. The data base log module is called to log the inserted segment.		MYPREOK
8. Call space management (DLZDHDS0) to update the prefix of physical twins, logical twins, physical parents, and logical parents. Update anchor point for HDAM root segments and call the data base log module to log all changes.	TOSPACE UPPARENT UPPREFIX	UPBITMAP BITMAPOK HDDANCOR

Extended Description	Routine	Label

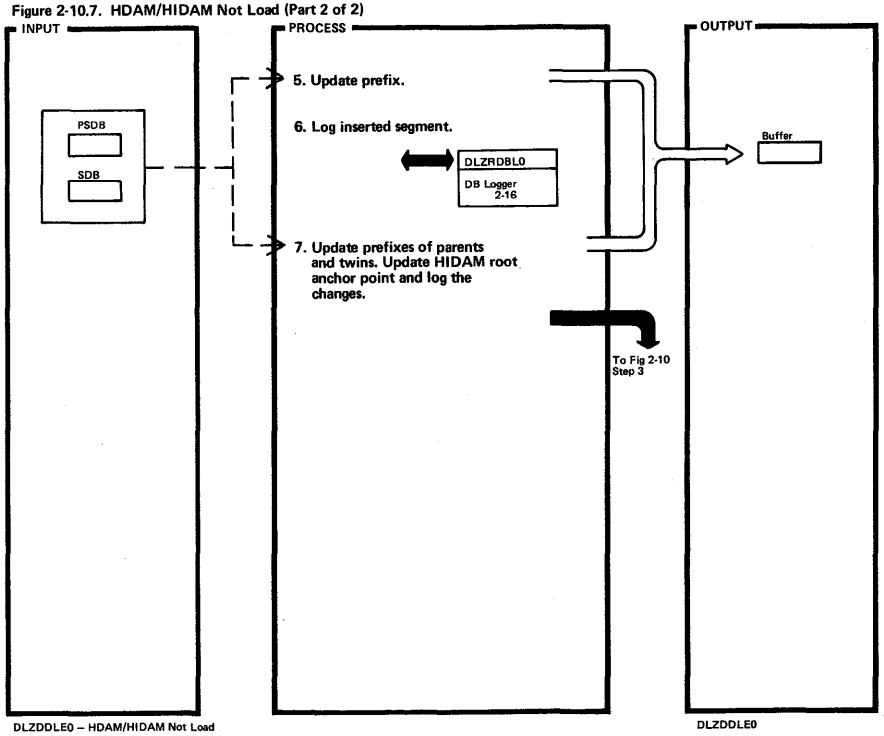
Figure 2-10.7. HDAM/HIDAM Not Load (Part 1 of 2)



Extended Description	Routine	Label
Note: When this entry is used, DLZDLR00 had done the positioning.		DFSDHDIO
2. Space management (DLZDHDS0) is called to get space for the segment. If the segment was deleted in one path only, i.e. it was not removed by DLZDLD00, the segment is replaced with the new data.	TBSPACE	GETSPACE SPACEOUT POSTPST SPACEOK
3. HIDAM root segments without PTB pointers are chained off the anchor point in chronological sequence.		SPACEOK
4. Move segment to buffer, update PCB feedback and the level table.	DFSDLIMS	ANCHOROK

Extended Description	Routine	Label

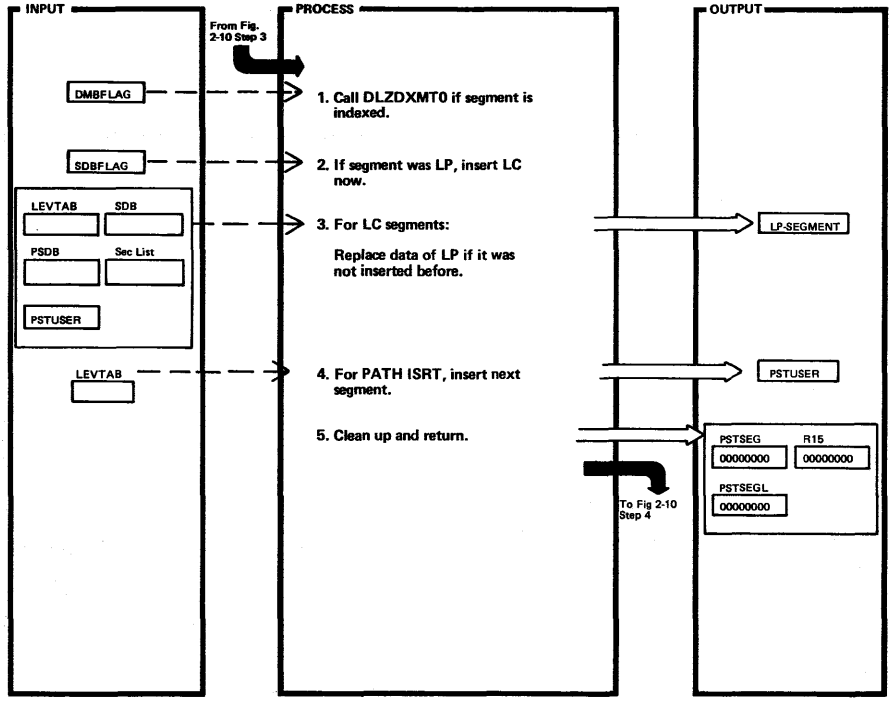
Figure 2-10.7. HDAM/HIDAM Not Load (Part 2 of 2)



Extended Description	Routine	Label
5. The prefix of the segment is updated: physical twin pointers, physical parent pointer, logical parent pointer, and logical twin pointers.		
6. The data base log module is called to log the inserted segment.		MYPREOK
7. Call space management (DLZDHDS0) to update the bitmap if required: update prefix of physical twins, logical twins, physical parents, and logical parents. Update anchor point for HDAM root segments, call the data base log module to log all changes.	TOSPACE UPPARENT UPPREFIX	UPBITMAP BITMAPOK HDDANCOR

Extended Description	Routine	Label

Figure 2-10.8. Not Load Ending Routine



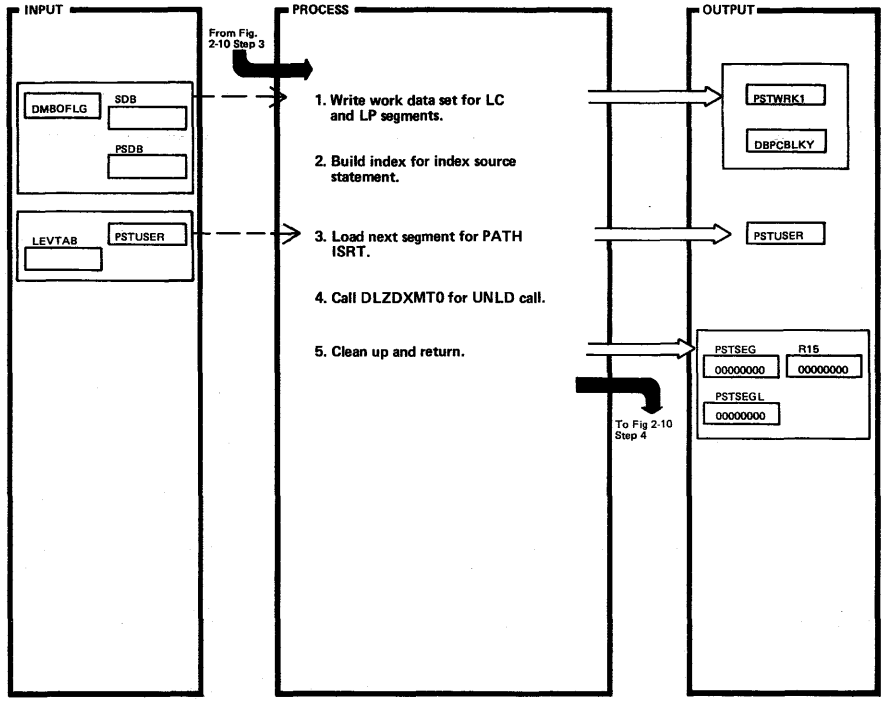
DLZDDLE0 – DFSDXNT0 Ending Routine for Not Load

DLZDDLE0

Extended Description	Routine	Label
1. Index Maintenance is called to build the primary or secondary index for an index source segment.		
2. If the ISRT call was for a concatenated segment, the destination parent was inserted first (if it did not exist before the ISRT call). The next step is to insert the logical child segment. The insert process is repeated from Figure 2-10 step 2.		NXTLEVIS
3. If the ISRT rule of the destination parent is virtual and this segment existed already, then the data of the destination parent is replaced. DLZDXMT0 is called to replace the index if the destination parent is an index source segment.		
4. If there are more segments to be inserted in a PATH, then point to the next segment in the I/O area and continue with Figure 2-10 step 2.		NOLPAREN

Extended Description	Routine	Label

Figure 2-10.9. Load Ending Routine



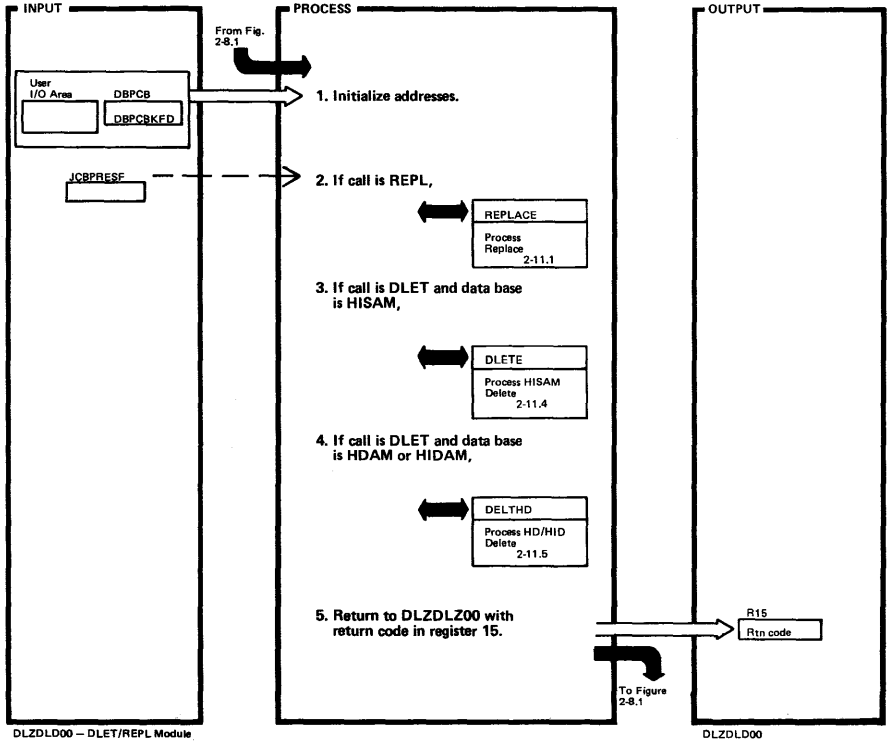
DLZDDLE0 – HIISNXLV Ending Routine Load Mode

DLZDDLE0

Extended Description	Routine	Label
1. If the segment just loaded was a logical child or a logical parent segment, DLZDSEH0 is called to write the work data set. If opening of the work data set fails due to 'ASSGN SYS013,IGN' and the segment was an LP, processing continues. On any other open failure, 'ABEND 864 is given.		CALLERN CALLWORK
2. If the segment is an index source segment, DLZDXMT0 is called. It writes the work data set or writes the index pointer segment directly.		NOLOAD NCALLNDX
3. For PATH ISRT, the pointer to the I/O area is updated and processing continues with Figure 2-10 step 2.		NOINDEX2
4. DLZDXMT0 is called to inspect all PSDBs of the DMB for index source segments and builds an FF key index pointer record for it.		

Extended Description	Routine	Label

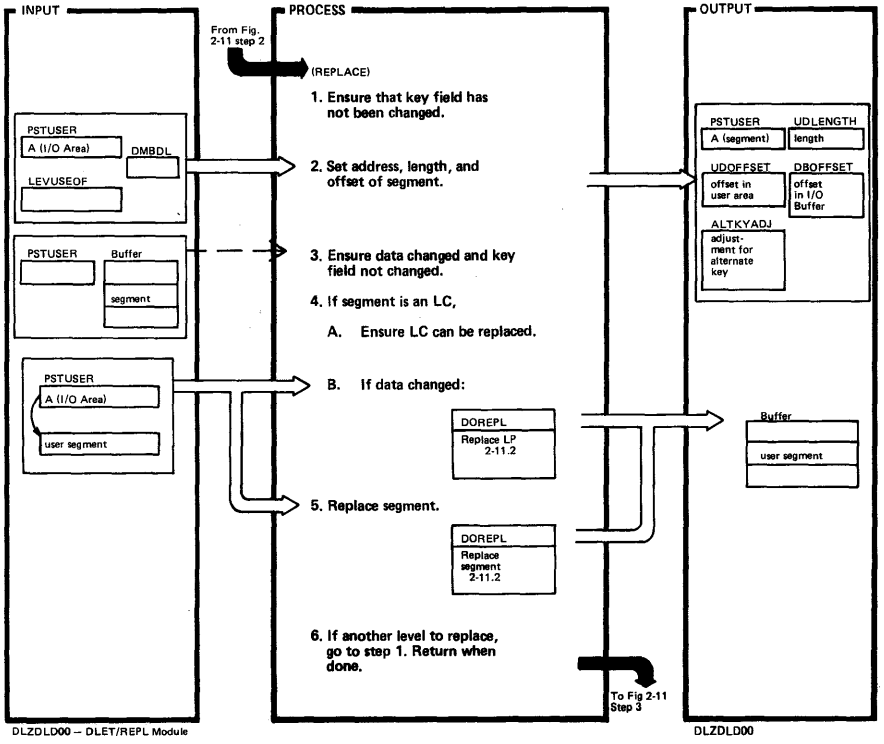
Figure 2-11. Delete/Replace



Extended Description	Routine	Label
1. The segment to be deleted or replaced is identified by the contents of JCBLEVIC. Position is established by DLZDLR00 in the previous call.	DLZDLD00	DELREPEP
2.		REPLACE
3.		DELETE
5. If a user error occurred, DBPCBSTC has return code. If abend, PSTERCD1 has abend code and registers are saved at SCDABSAV + 8.		RETURN

Extended Description	Routine	Label

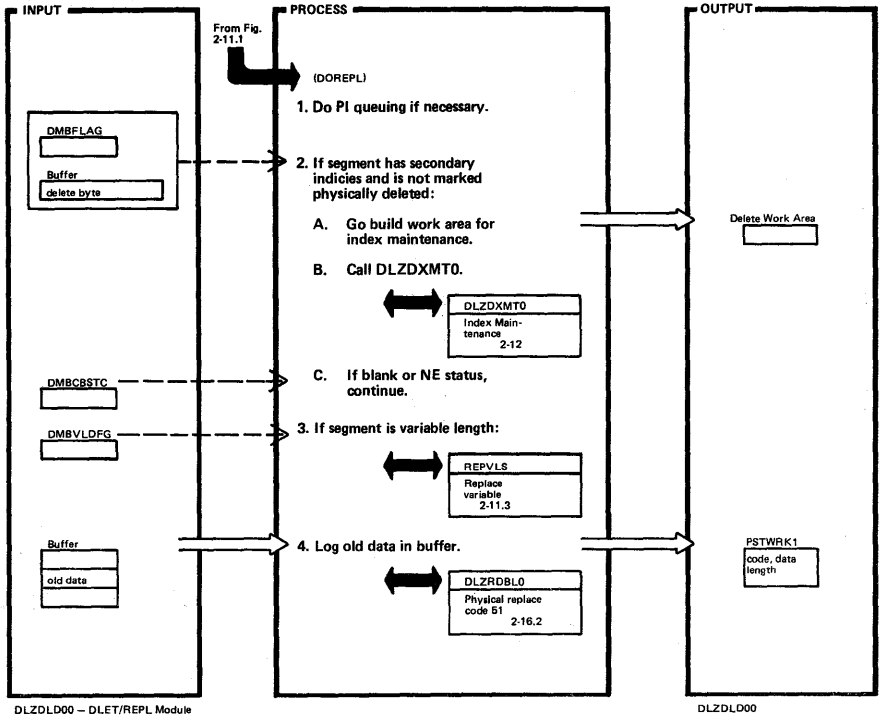
Figure 2-11.1. Replace



Extended Description	Routine	Label
1.		REPCK01
2. PSTUSER will have new value if path call had been made. The length is taken from the first two bytes of the I/O area if segment is variable length.		
3. Additional logic is needed if segment is variable length or if PROCSEQ is specified.		CHKRLP CHKREPL1
4. A. The following check is made for the LC:		CHKREPFF
Neither the physical nor logical key fields can be changed (DA status).		
The following checks are made for the destination parent:		CHKRLP01
a) If data didn't change, no replace.		

Extended Description	Routine	Label
4. (con't)		
b) If replace rule is physical, RX status. If logical, no change and blank status. If virtual, the key of the LP cannot be changed (DA status). The segment can be replaced.		
B.		REPPAR01
5. This replaces normal segment or LC.		REPFINAL
6. If path call, see if another segment in hierarchy can be replaced.		LEVDONE

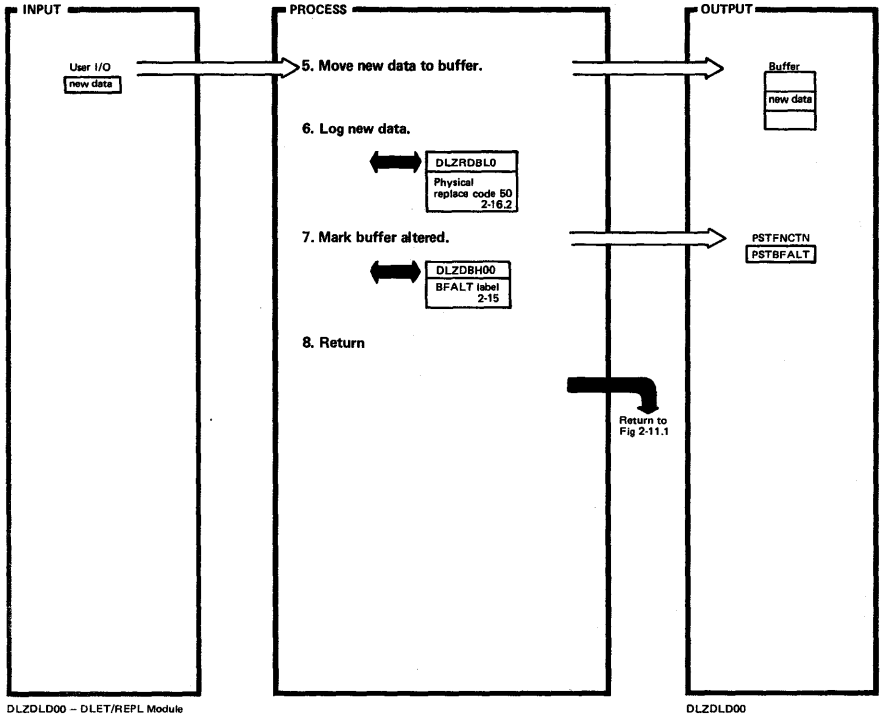
Figure 2-11.2. Replace Data (Part 1 of 2)



Extended Description	Routine	Label
2. Index Maintenance needs the actual concatenated key of this segment. If return code is NE, we still continue processing because index is now set as per new data. Work area is freed.		DOREPL09
3.		DOREPL10
4. DBLPHYR+DBLPHYR0 is set in first byte of PSTWRK1.		

Extended Description	Routine	Label

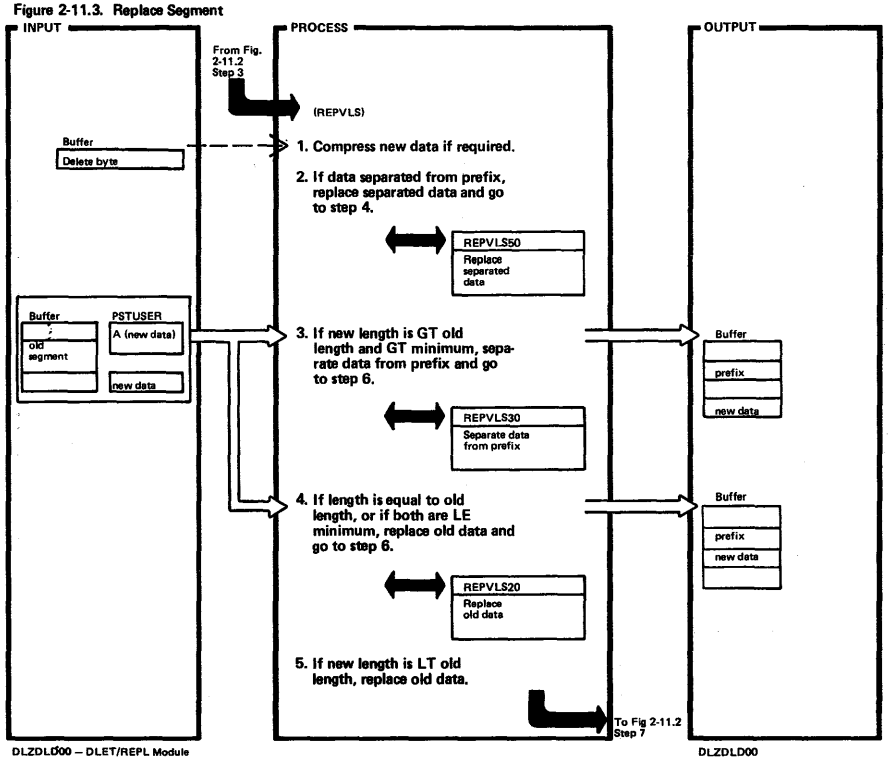
Figure 2-11.2. Replace Data (part 2 of 2)



Extended Description	Routine	Label
5. The address of the user's I/O area is in PSTUSER.		
6. DBLPHYR is set in PSTWRK1 with the length of the segment.		DOREPL92 REPL18

Extended Description	Routine	Label

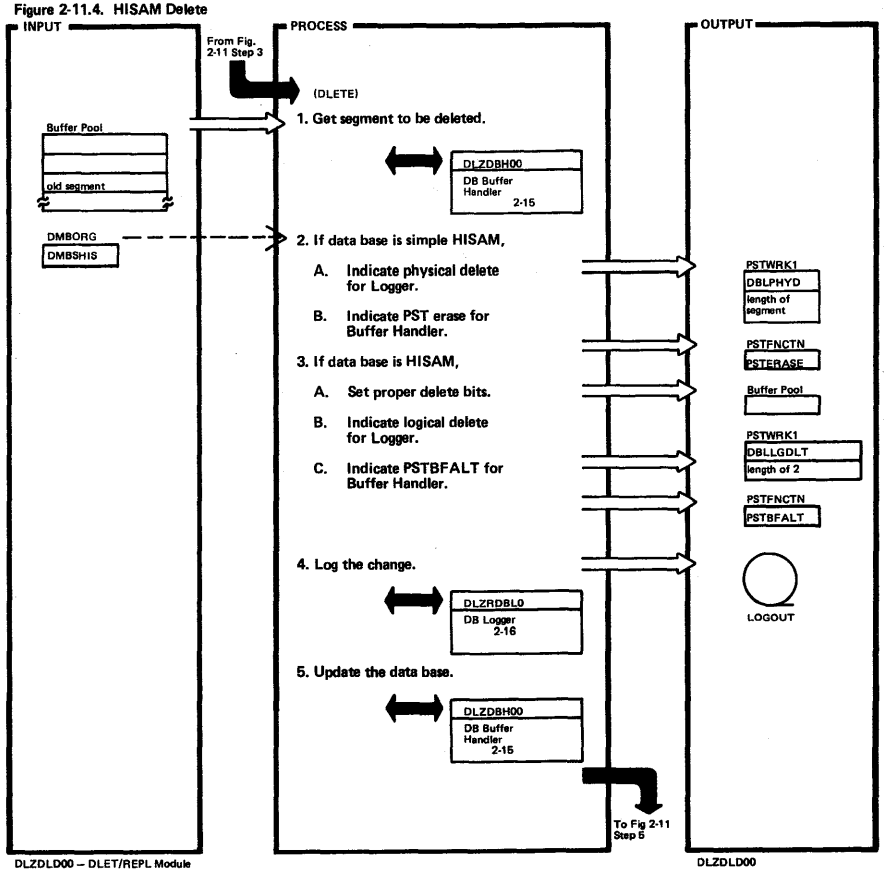
Figure 2-11.3. Replace Segment



Extended Description	Routine	Label
2. When the data is previously separated and the new data length is less than the old length, an attempt is made to relocate the new data adjacent to the prefix.	DLZDLDR0	REPVLS01
3. When the old segment size is not large enough for the new segment, the data is separated from the prefix. A pointer overlays the first four bytes of the old data and will be used to find the new.		REPVLS03
5. When the new data will fit in the old location, it is moved over the old data with any excess bytes being freed.		REPVLS10
All changes to the data base have been logged.		REPVLS38

Extended Description	Routine	Label

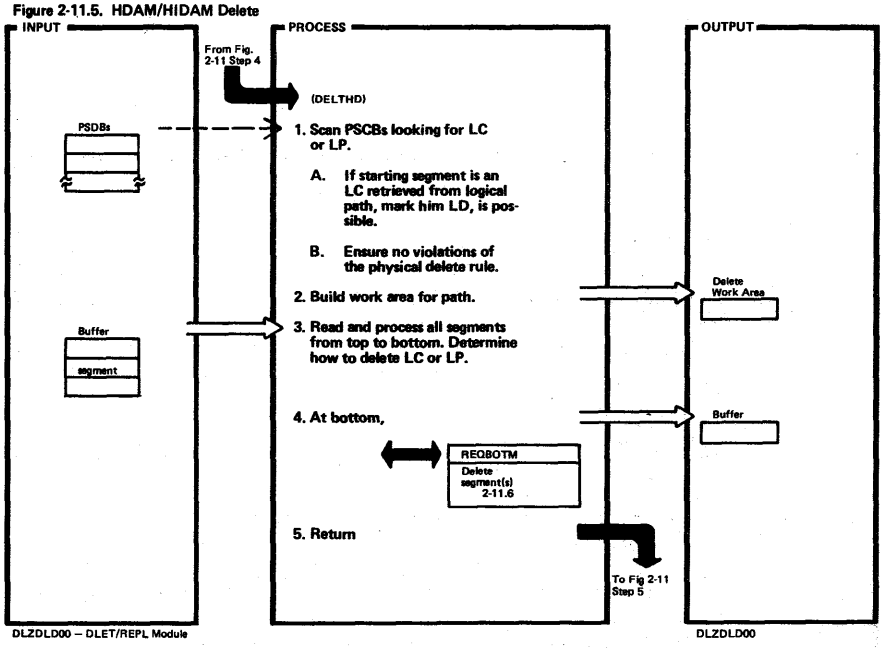
Figure 2-11.4. HISAM Delete



Extended Description	Routine	Label
1.		DELT01
2. The entire segment to be erased is logged.		SHISAM
3. Only the segment code and delete byte are logged.		DELT41 LOGDLT

Extended Description	Routine	Label

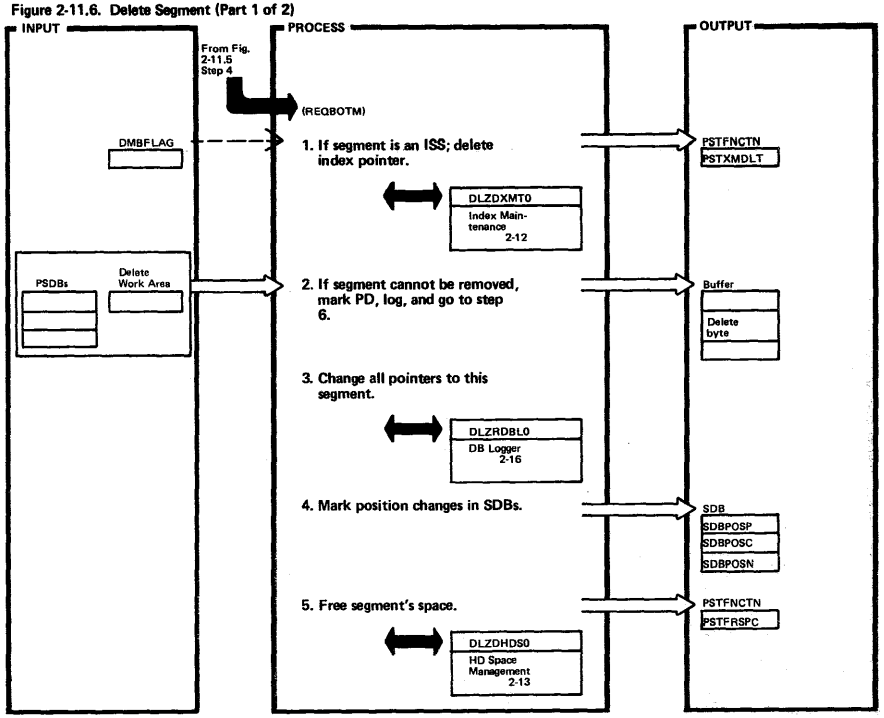
Figure 2-11.5. HDAM/HIDAM Delete



Extended Description	Routine	Label
1. A. LC will be marked logically deleted (LD) if delete rule = physical or logical and segment not PD (physically deleted).	DLZDLD00	DELTHD ILCDLT
B. A logical parent can have no active logical children. An LC must not be accessable by his logical path.		DELT09 PHYSCAN
2. This is needed to remember where we are during scan of data base and to build concatenated keys.		DELTHA NEWOMB
3. LCF and LCL pointers in logical parents, and LTF and LTB pointers in logical children, will be updated now.		REQSCAN2 SCANDMB REQDOWN
4. Segments may be marked deleted or physically removed.		REQBOTM
5. All work sets are freed.		ENDLTSCN

Extended Description	Routine	Label

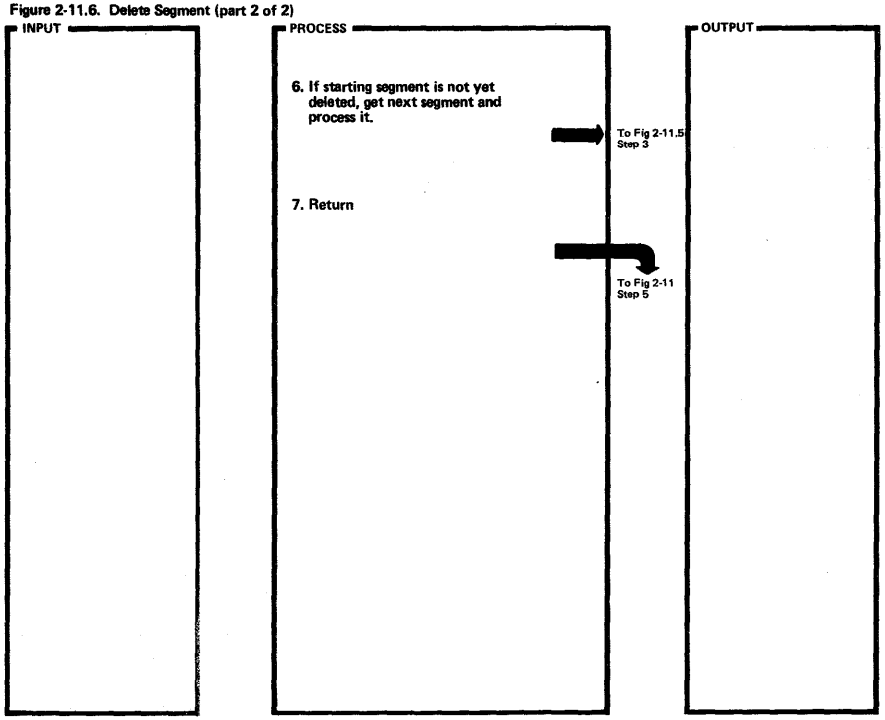
Figure 2-11.6. Delete Segment (Part 1 of 2)



Extended Description	Routine	Label
1. If the index source segment (ISS) has been marked physically deleted (PD), no index maintenance is performed. Delete processing continues with blank or 'NE' status from DLZDXMT0.	DLZDLDD0	REQB01
2. A segment will not be physically removed if still required because of a logical relationship. Note that the delete work area and DL/I blocks (primary PSDBs) are used as input to every step.		REQB02
3. If segment is an LC or LP, the logical relationship pointers (LC, LP, and LT) have already been changed.	DLZDLDD0 DLZDLDA0	FREESPCE FRSPC00
4. The current position (SDBPOSC) is marked 'lost' in this caller's PCB. If any other PCB has position on this segment, the position should be changed to bypass this segment.	DLZDLDA0	PRSPC05 MARKSDB
5. DLZDHDS0 makes the log calls for the physical delete.	DLZDLDA0	FRSPC05G

Extended Description	Routine	Label

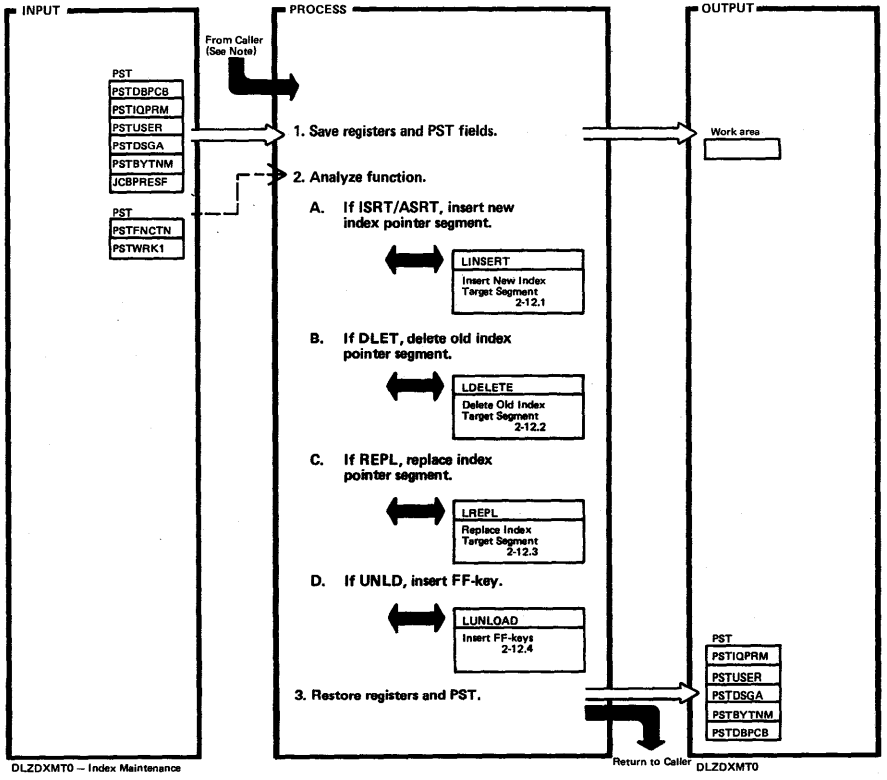
Figure 2-11.6. Delete Segment (part 2 of 2)



Extended Description	Routine	Label
6. Next segment is physical twins, sibling, or parent.	DLZDLD00	BOTM1B
7. At end, a final log call is made to DLZRDBL0 which signifies delete is finally accomplished.	DLZDLD00	ENDLTSCN

Extended Description	Routine	Label

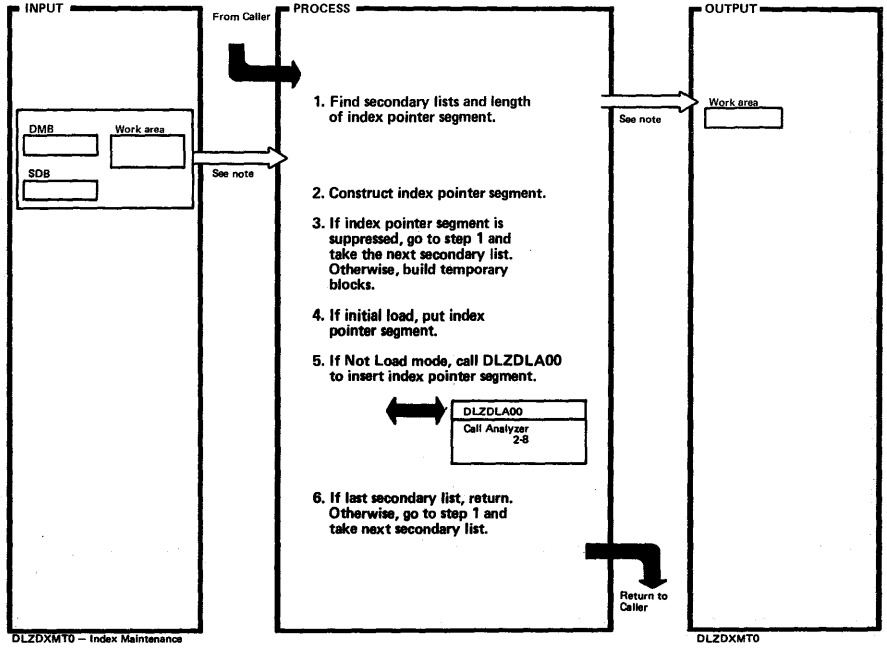
Figure 2-12. Index Maintenance



Extended Description	Routine	Label
Note: DLZDXMT0 is called from DLZDDLE0 or DLZDLD00.		
2. When called from DLZDDLE0, the function is ISRT, ASRT, UNLD, or REPL. When called from DLZDLD00, the function is REPL or DLET. PSTWRK1 contains the PSDB address of the index source segment for DLET or the LSDB address of the index source segment.		
A. Construct and insert all index pointer segments for this index source segment that should not be suppressed.	LINSERT	
B. Construct and delete all old index pointer segments existing for this index source segment.	LDELETE	

Extended Description	Routine	Label
2. (con't)		
C. Construct all old and new index pointer segments that can be constructed from that index source segment. Depending on the data changed and the status of suppression, delete old index pointer segment, or insert new index pointer segment, or delete old and insert new index pointer segment, or replace data of index pointer segment.	LREPL	
D. If DLBL card is provided, write index pointer segment with all FF-keys for all index data bases to belonging to this PCB.	LUNLOAD	

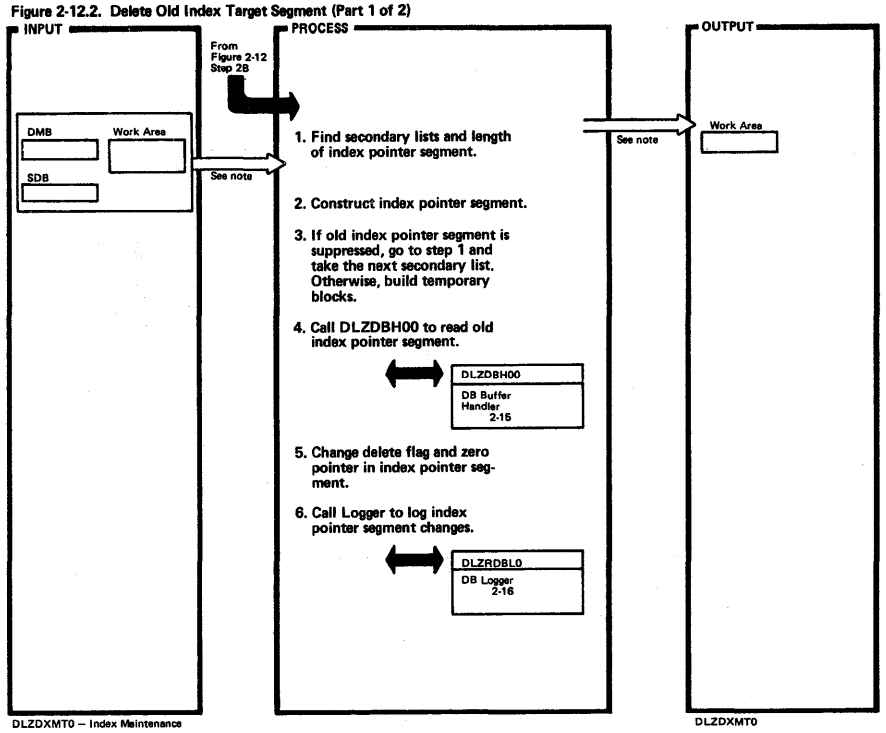
Figure 2-12.1. Insert New Index Target Segment



Extended Description	Routine	Label
Note: The input control blocks are used in all process steps. The output work area is modified in all process steps.		
1. Find SECLISTs and PSDBs of index source segment, index target segment, and index pointer segment and save their address in work area. Decide if primary or secondary has to be built. Find length of index pointer segment, sequence field, segment length, and protected data length.	LBLDWKA	
2. For primary indexes, move HIDAM root sequence field from user I/O area to work area. For secondary indexes, construct SRCH, SUBSEQ, and DDATA fields.	LBLDXNS LGRBACK LNULSUP LCALLBH	
3. When the index entry has to be suppressed due to SRCH equal to NULLVALUE or due to exit routine return code, the index pointer segment is not inserted.		

Extended Description	Routine	Label
3. (con't)		
Build temporary blocks: • SDB • Segment name=sequence field name of index pointer segment • Update Index Maintenance JCB and DSG.	LBLDCTLB	
4. If DLBL cards are provided, write index pointer segment to index data base and call DLZDLOC0 to open index data base if not open yet. Otherwise, write index pointer segment to workfile and call DLZDSEH0 to open the workfile.	LLOAD LWORKDS LCALLBH DLZDLOC0	
5. Prepare DL/I call list to call DLZDLA00 with an *X call.	LINXNS DLZDLA00	
6. When the last secondary list is reached, exit is to LRETURN. On error in secondary lists, exit is to LABND772 (abend code 772).		

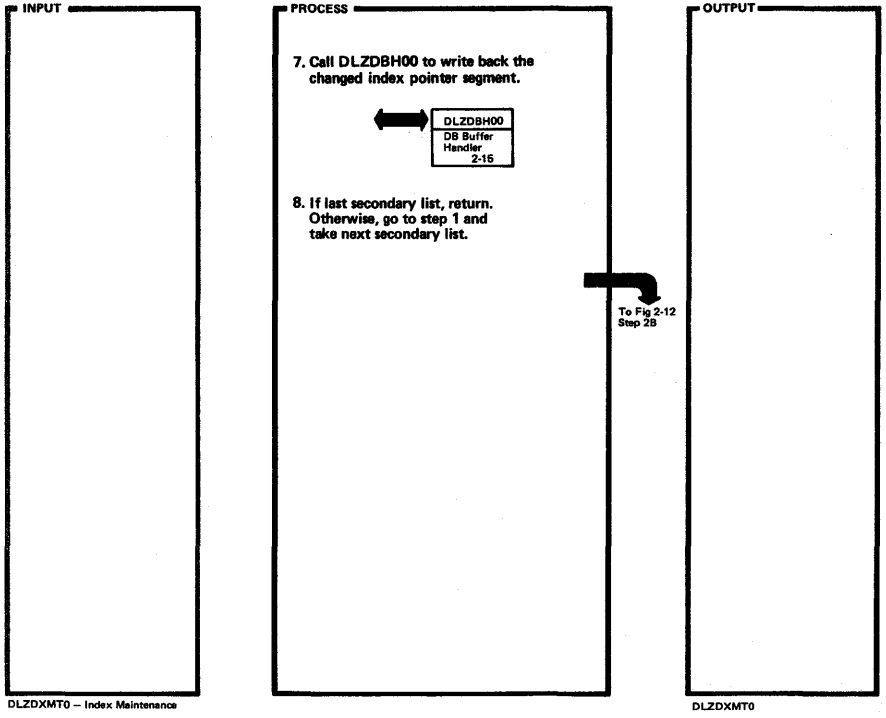
Figure 2-12.2. Delete Old Index Target Segment (Part 1 of 2)



Extended Description	Routine	Label
Note: The input control blocks are used in all process steps. The output work area is modified in all process steps.		
1. Find SECLISTs and PSDBs of index source segment, index target segment, and index pointer segment and save their address in work area. Decide if primary or secondary has to be built. Find length of index pointer segment, sequence field, segment length, and protected data length.	LBLDWKA	
2. For primary indexes, move HIDAM root sequence field from user I/O area to work area. For secondary indexes, construct SRCH, SUBSEQ, and DDATA fields.	LBLDXNS LGRBACK LNULSUP LCALLBH	
3. When the old index entry has to be suppressed due to SRCH equal to NULLVALUE or due to exit routine return code, the index pointer segment is not inserted.		

Extended Description	Routine	Label
3. (con't) Build temporary blocks: • SDB • Segment name=sequence field name of index pointer segment • Update Index Maintenance JCB and DSG.	LBLDCTLB	
4. The Buffer Handler is called (PSTSTLEQ) to find the old index pointer segment. If it is not found, or it is already deleted, or the pointer or key are not correct, an NE status code is returned to the caller.	LGOXNS LDOXNS	
5. Delete flag is set to C0.		
6. Chain maintenance and logical delete calls are made to data base module.	DLZRDBL0	

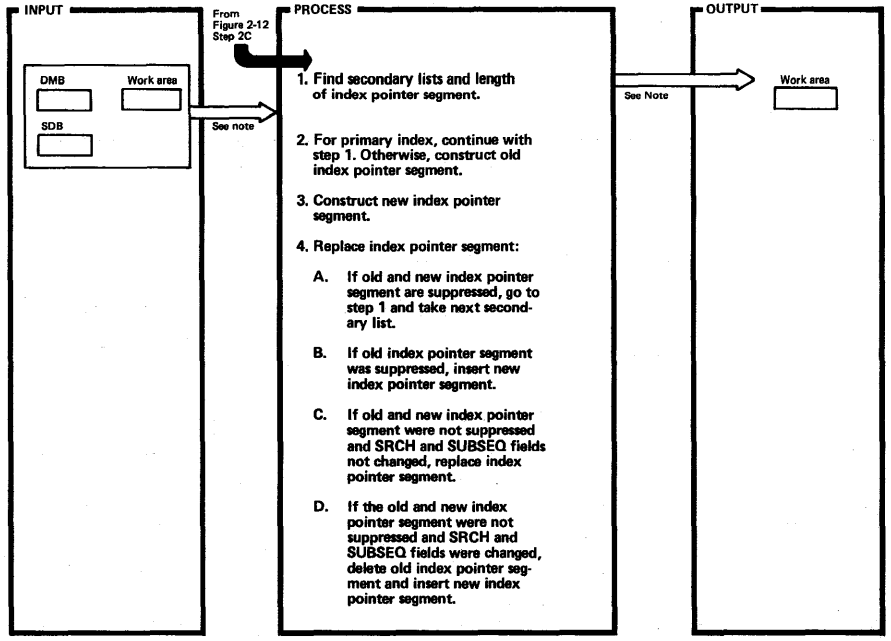
Figure 2-12.2. Delete Old Index Target Segment (Part 2 of 2)



Extended Description	Routine	Label
7. The Buffer Handler is called (PSTBFALT) to write the changed index pointer segment back.	DLZRDBL0	
8. When the last secondary list is reached, exit is to LRETURN. On error in secondary lists, exit is to LABND772 (abend code 772).		

Extended Description	Routine	Label

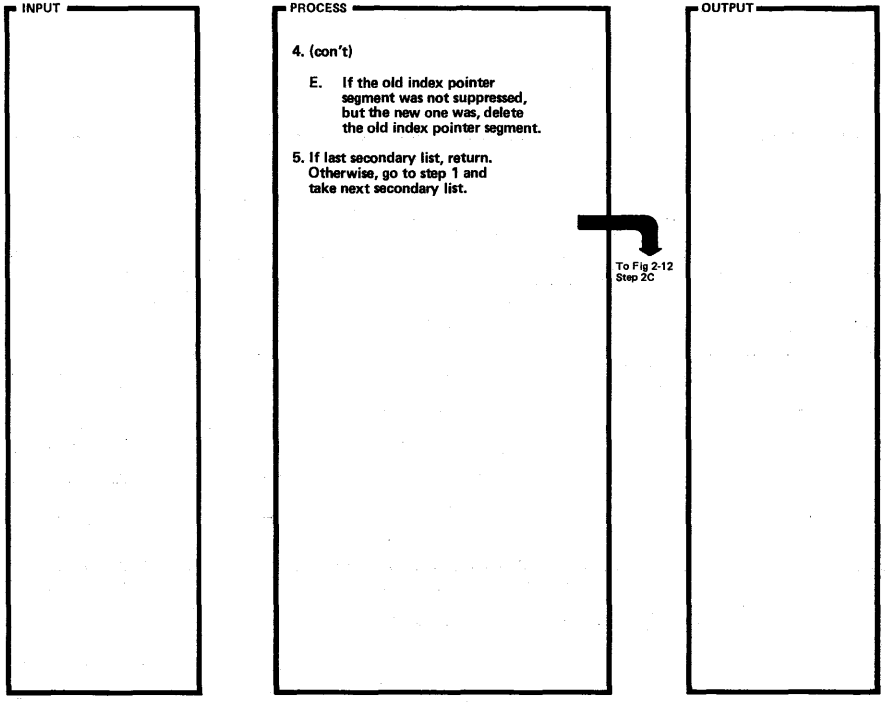
Figure 2-12.3. Replace Index Target Segment (Part 1 of 2)



Extended Description	Routine	Label
Note: The input control blocks are used in all process steps. The output work area is modified in all process steps.		
1. Find SECLISTs and PSDBs of index source segment, index target segment, and index pointer segment and save their address in work area. Decide if primary or secondary has to be built. Find length of index pointer segment, sequence field, segment length, and protected data length.	LBLDWKA	
2. Construct old index pointer segment from SRCH, SUBSEQ, and DDATA fields.	LBLDXNS LGRBACK LNULSUP LCALLBH	
3. Construct new index pointer segment from SRCH, SUBSEQ, and DDATA fields.	LBLDXNS LGRBACK LNULSUP LCALLBH	

Extended Description	Routine	Label
4. Replacing of the index pointer is done in different ways, depending on suppression of old and new index pointer segment.		
A. When both old and new index pointer segments are suppressed, no action takes place.		
B. Continue with insert subroutine	LINXNS	
C. DLZDBH00 is called to read the old index pointer segment. On errors, NE is returned. The data base log module is called to log the old index pointer segment, and after the change of the DDATA fields, the new index pointer segment. DLZDBH00 is called again to write the index pointer segment back (PSTBFALT).	LGOXNS LROXNS	
D.	LGOXNS LDOXNS LINXNS	

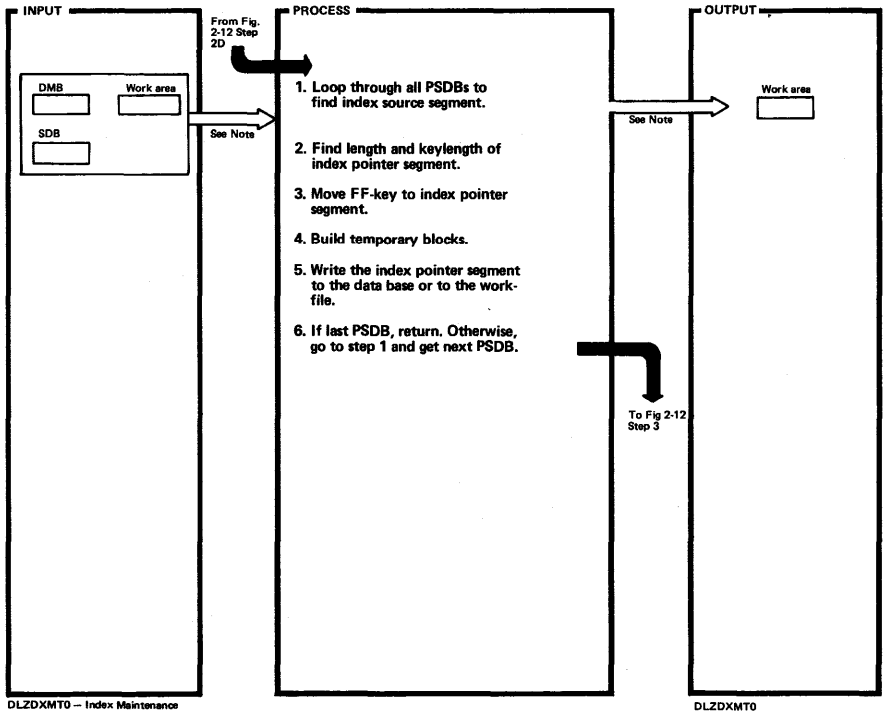
Figure 2-12.3. Replace Index Target Segment (part 2 of 2)



Extended Description	Routine	Label
5. When the last secondary list is reached, exit is to LRETURN. On error in secondary lists, exit is to LABND772 (abend code 772).		

Extended Description	Routine	Label

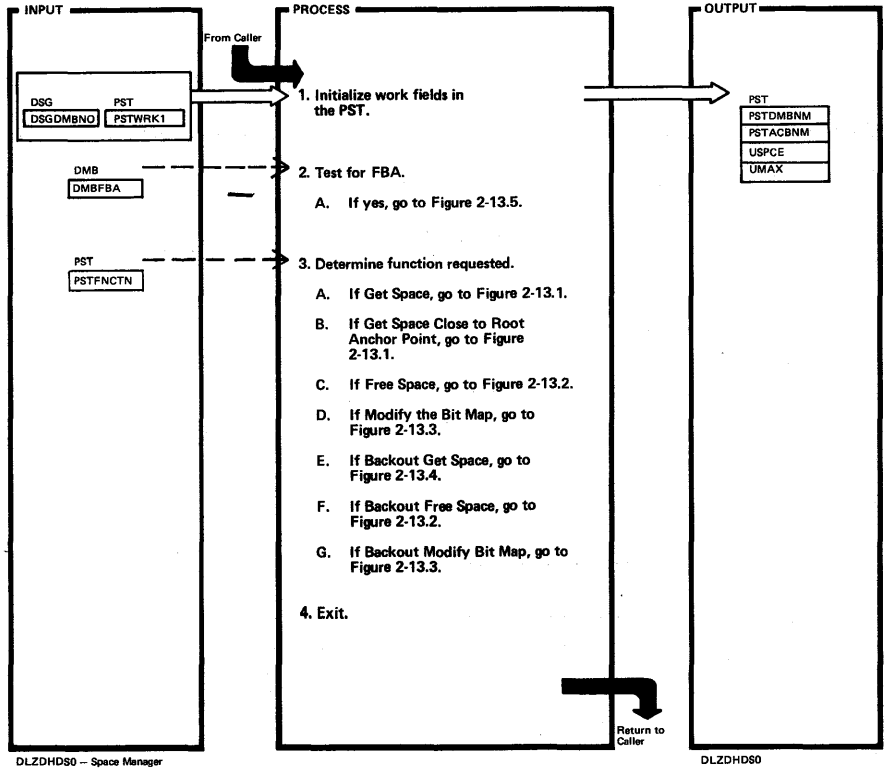
Figure 2-12.4. Insert FF-Keys



Extended Description	Routine	Label
Note: The input control blocks are used in all process steps. The output work area is modified in all process steps.		
1. DLZDDLE0 passes the LSDB address of the root segment with an UNLD call. The DDIR address is used and all PSDBs in that DMB are inspected if an index exists.		
2. Find length of index pointer segment and its key length. Decide if primary or secondary index has to be built.	LBLDWKA	
3. Move FFs in the length of the index pointer segment sequence field to the index pointer segment.	LBLDXNS	

Extended Description	Routine	Label
4. Build temporary blocks: • SDB • segment name = sequence field name of index pointer segment • update index maintenance JCB and DSG.	LBLDCTLB	
5. Write index pointer segment to index data base if DLBL cards are provided. Call DLZDLOC0 to open index data base if not yet open.	LLOAD LWORKD5 LCALLBH DLZLOC0	

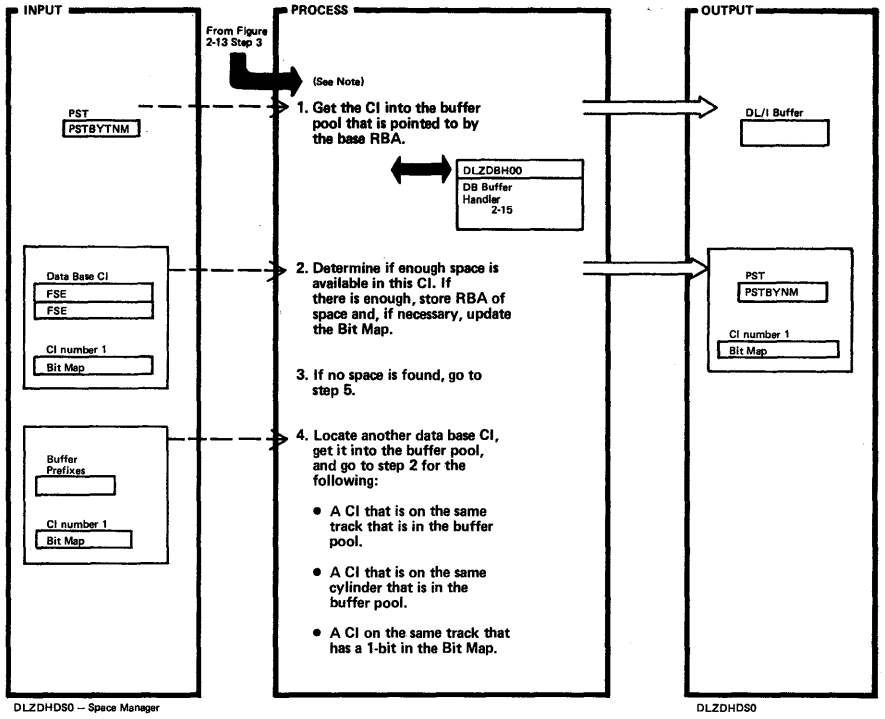
Figure 2-13. HD Space Management



Extended Description	Routine	Label
1. PSTWRK1 contains the length of the space to be obtained or freed.		
2. A. If the device is FBA, the device characteristics must be obtained and the number of CIs per track and CIs per cylinder calculated.	DLZDCI00	
3. A. Get space in a data base CI for the specified segment as close as possible to a specified base RBA. The caller passes the address of the involved segment's PSDB in R5 and the base RBA in PSTBYTNM.	DLZGGSPC	
B. Get space in a data base CI for the specified segment as close as possible to a root anchor point. The caller passes the address of the involved segment's PSDB in R5 and the CI number/RAP number (in the format BBBR) of the involved root anchor point in PSTBYTNM.	DLZGGSPC	

Extended Description	Routine	Label
3. (con't)		
C. Free space that has been allocated for the specified segment in a data base CI. The caller passes the address of the involved segment's PSDB in R5.	DLZFRSPC	
D. Turn on or off the bit in the Bit Map representing the specified CI of a data base. The caller specifies the CI number in PSTBLKNM.	DLZDHDS0	FIXBTMP
E. Backs out a previously processed 'Get Space' call.	DLZDHDS0	
F. Backs out a previously processed 'Free Space' call.	DLZFRSPC	
G. Backs out a previously processed 'Modify Bit Map' call.	DLZDHDS0	FIXBTMP

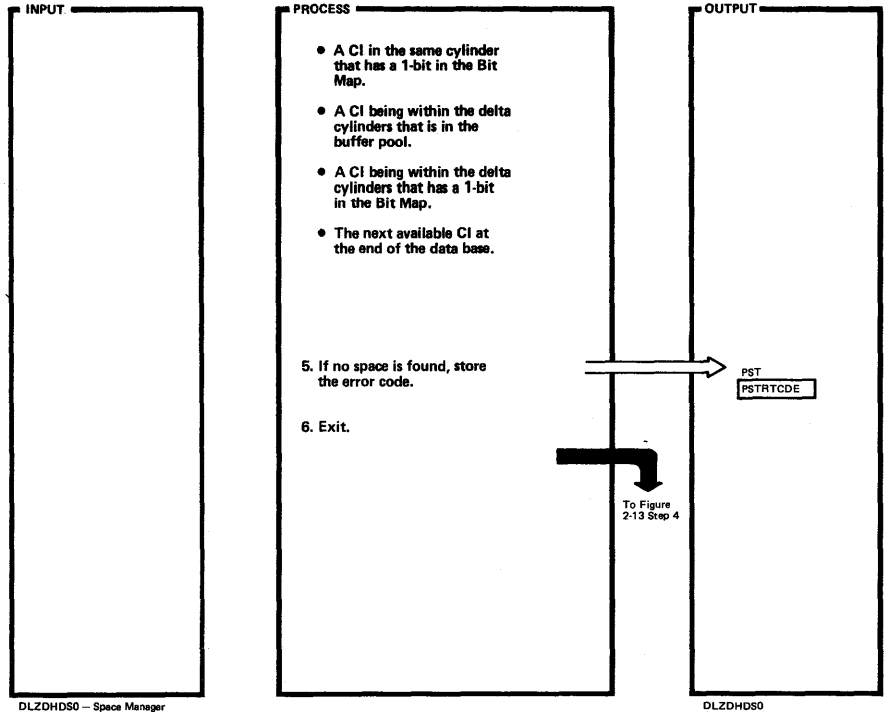
Figure 2-13.1. Get Space (Part 1 of 2)



Extended Description	Routine	Label
Note: For the functions 'Get Space' and 'Get Space Close to RAP', the following csects are used: Main routine: DLZDHDS0 DLZDHDS0 calls DLZGGSPC. DLZGGSPC calls DLZRCHBK, DLZLLCLC, DLZRRHPL, DLZRRHMP, DLZMNLCT, and DLZMMUDT. DLZRRHPL calls DLZRCHBK. DLZRRHMP calls DLZRCHBK. 2. If distributed free space has been specified, a check is made if this block is to be left free. If not, a check is made to see if a percentage of this block is to be left free. If so, this percentage is added to the space requested. To determine if enough space is available in a CI, the FSE's in this CI are checked. If there is more than one FSE in a CI, the free space with the largest of the following values that will not cause a Bit Map change is taken:		

Extended Description	Routine	Label
2. (con't) • the size itself • the size+minimum segment length • the size+2. A Bit Map Change is necessary if the data base CI cannot accomodate the maximum size segment because the available space has been used. The Bit Map update is performed by DLZMMUDT. 4. The calculation of the CI numbers for a given range is done by routine DLZLLCLC. Searching through the buffer prefixes is done by routine DLZRRHPL. Searching through the Bit Map is done by routine DLZRRHMP.		

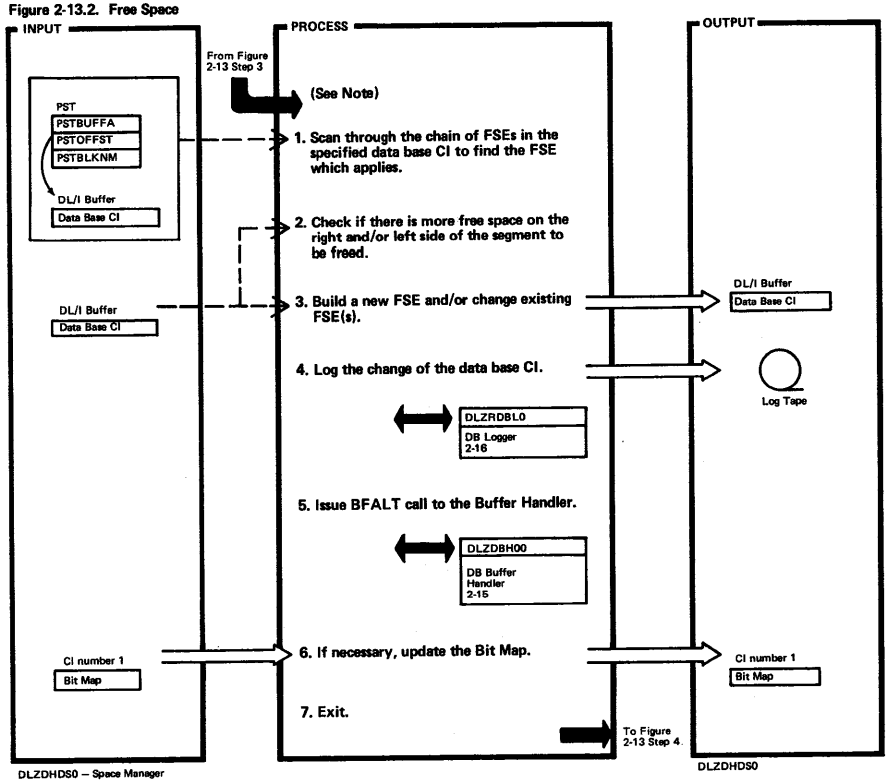
Figure 2-13.1. Get Space (Part 2 of 2)



Extended Description	Routine	Label
5. A return code of X'0C' will be returned to the caller.		

Extended Description	Routine	Label

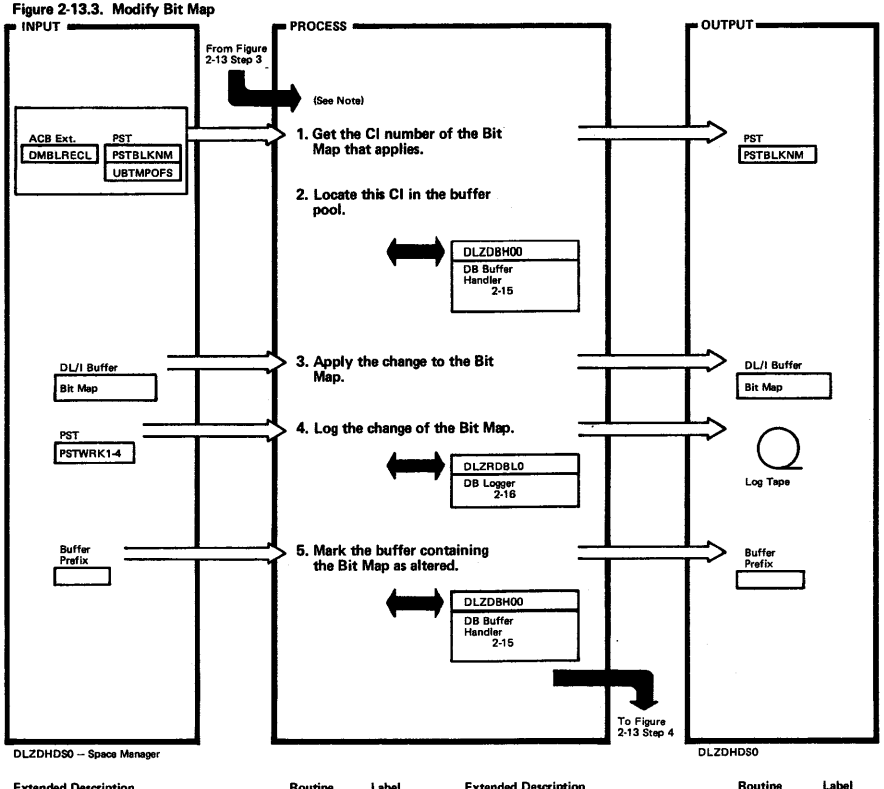
Figure 2-13.2. Free Space



Extended Description	Routine	Label
Note: For the functions 'Free Space' and 'Backout Free Space', the following csects are used: Main routine: DLZDHDS0 DLZDHDS0 calls DLZFRSP0 DLZFRSP0 calls DLZMMLCT and DLZMMUDT. 1. The scan will be finished when a FSE with a higher offset than the one in PSTOFFST is reached, or, when the end of the FSE chain is reached. 2. The purpose of this check is to find out whether there will be a contiguous piece of free space after processing the current Free Space call. 6. A Bit Map change is necessary if the data base CI can accomodate the maximum size segment after process-		

Extended Description	Routine	Label
6. (con't) ing the Free Space call. In this case, the appropriate bit in the Bit Map has to be turned on. The Bit Map update is performed by routine DLZMMUDT.		

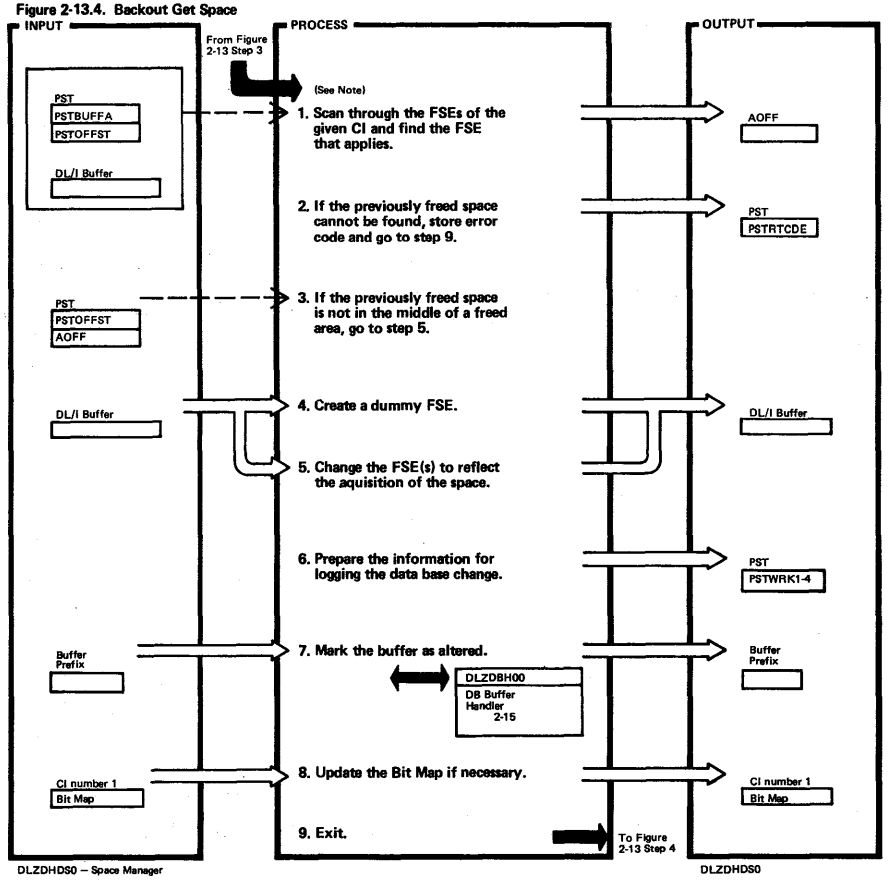
Figure 2-13.3. Modify Bit Map



Extended Description	Routine	Label
Note: For functions 'Fix Bit Map' and 'Backout Fix Bit Map', the following csects are used: Main Routine: DLZDHDS0 DLZDHDS0 calls DLZMMLCT and DLZMMUDT. 1. This step is performed by routine DLZMMLCT. 2. A 'Byte Locate' call is issued.	DLZDBH00	
3. The update of the Bit Map is performed by routine DLZMMUDT. 5. A 'Buffer Alter' call is issued.	DLZDBH00	

Extended Description	Routine	Label

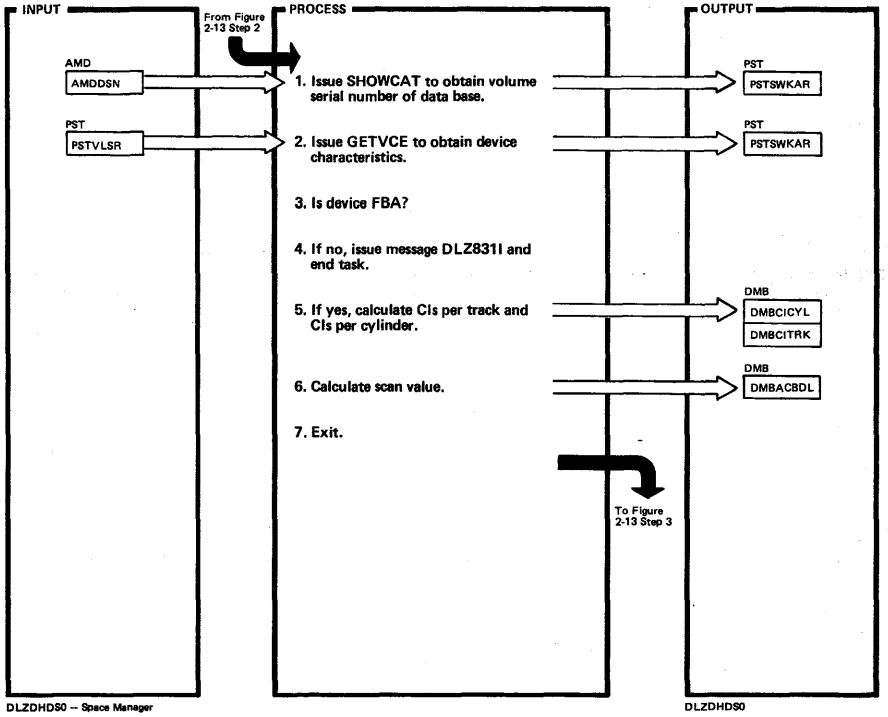
Figure 2-13.4. Backout Get Space



Extended Description	Routine	Label
Note: This function backs out a Free Space call processed previously. The following csects are used: Main routine: DLZDHDS0 DLZDHDS0 calls DLZRCHBK. 2. PSTOFFST contains the offset to the part of the data base CI which was freed during the Free Space call to be backed out. 3. A return code of X'0C' is stored in PSTRTCDE. 7. A 'Buffer Alter' call is issued to the Buffer Handler.	DLZDBH00	

Extended Description	Routine	Label
8. A Bit Map update is necessary if the data base CI cannot accomodate the maximum length segment after backing out the previously processed Free Space call. The Bit Map update is performed by routine DLZMMUDT.		

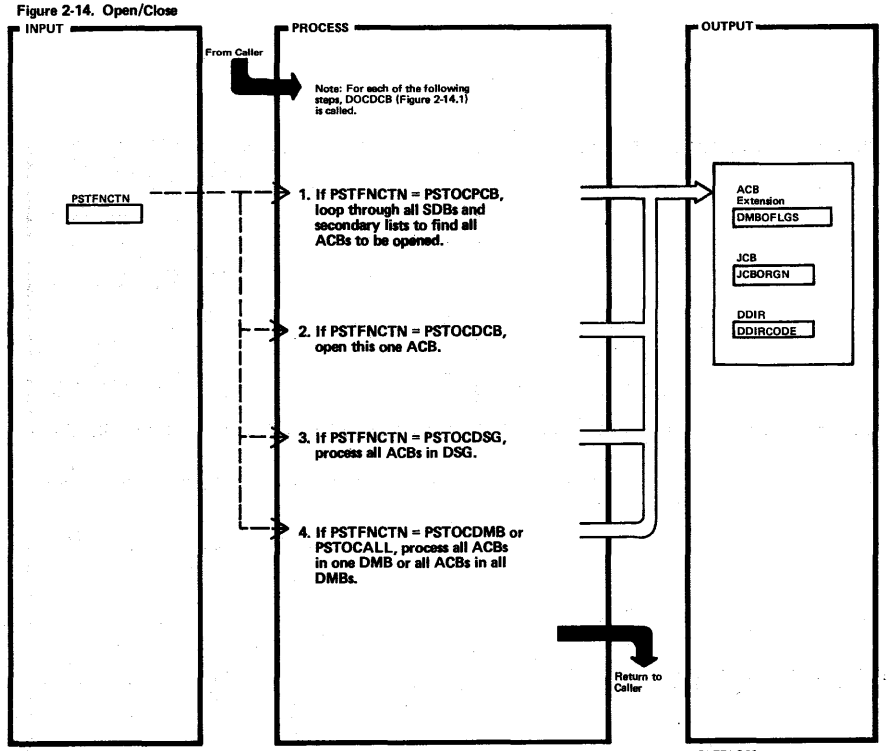
Figure 2-13.5. FBA Support Device Characteristics Routine



Extended Description	Routine	Label

Extended Description	Routine	Label

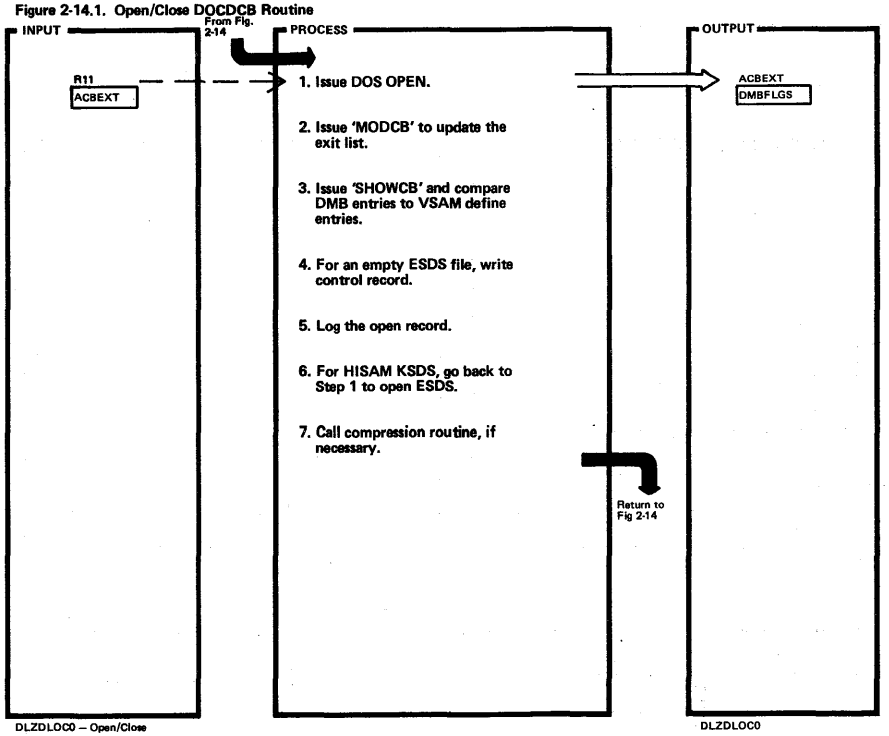
Figure 2-14. Open/Close



Extended Description	Routine	Label
1. This function is used by the utilities DLZRDBC0 and DLZURGP0. It is also used by DLZDLA00 when the first data base call to a not open data base is issued (batch only). For PROCOPT=L, only one data base is opened. For all other processing options, all related data bases are opened as well (index data bases and logically related data bases).	PCENTRY PSROUT DOCDCB	AROUND
2. DLZDLR00 uses this function for positioning a HSAM data base at the start point. It is also used by DLZURDB0. It opens only one ACB, i.e. for HISAM only KSDS or ESDS.	ACBENTRY DOCDCB	
3. DLZDLD00 uses this function when it finds a logically related data base that is not opened (this can happen because of delete sensitivity propagation).	DGENTRY DOCDCB	

Extended Description	Routine	Label
4. PSTOCALL + PSTOCOPN : DLZOLI00 uses this call to open all data bases in the system eligible for initial opening (online only). PSTOCALL + PSTOCCLS : This call is used to close all ACBs in the DL/I system (e.g. DLZDLA00). PSTOCDMB : This call is used by DLZOLI00 for deferred opening (online). It is also used by DLZDXMT0 and by data base utilities. It opens/closes one ACB (two ACBs for HISAM).	DENTRY DROUTINE DOCDCB	

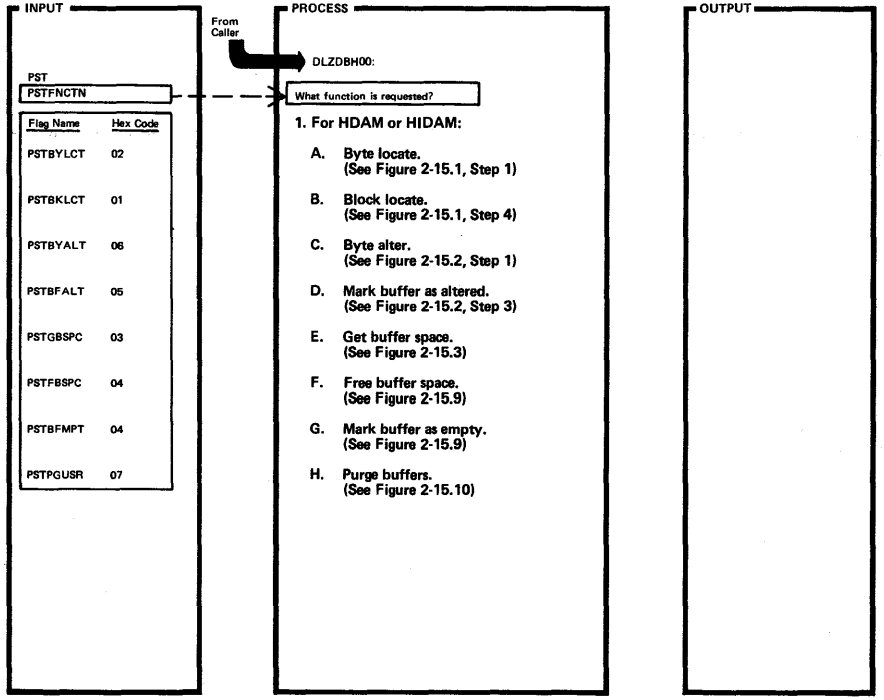
Figure 2-14.1. Open/Close DOCDCB Routine



Extended Description	Routine	Label
1. This part is called from all steps of Figure 2-14 for opening. If the data base is open, return immediately. Immediate return is also done when the call is PSTOCALL and initial opening is not planned. Unsuccessful opens have return error code in PST and flag in JCB. 'DLZ020I' is issued.	DOCDCB	DOCOPEN
2. The exit list is updated with the address of the error handling routines of DLZDBH00.	DOCMOD	
3. Control interval size, relative key position, and key length of DMB is compared to VSAM catalog entries. MISMATCH: DLZ025I, DLZ027I, and DLZ028I. For HISAM, the number of logical records in VSAM catalog has to be zero for PROCOPT=L. For HD, the high used RBA is inspected. Message 'DLZ023I' is issued for conflicts.	DOCSHOW	

Extended Description	Routine	Label
4. The first control interval is written (for HISAM, as many records as fill one CI). It contains DL/I control information. For HD, the ACB is closed and opened again to simulate 'NOT LOAD' to VSAM.	DOCFIRST	NOTHIDAM
7. All PSDBs are inspected to determine if a compaction routine with 'INIT' specified exists.	DOCVARI	

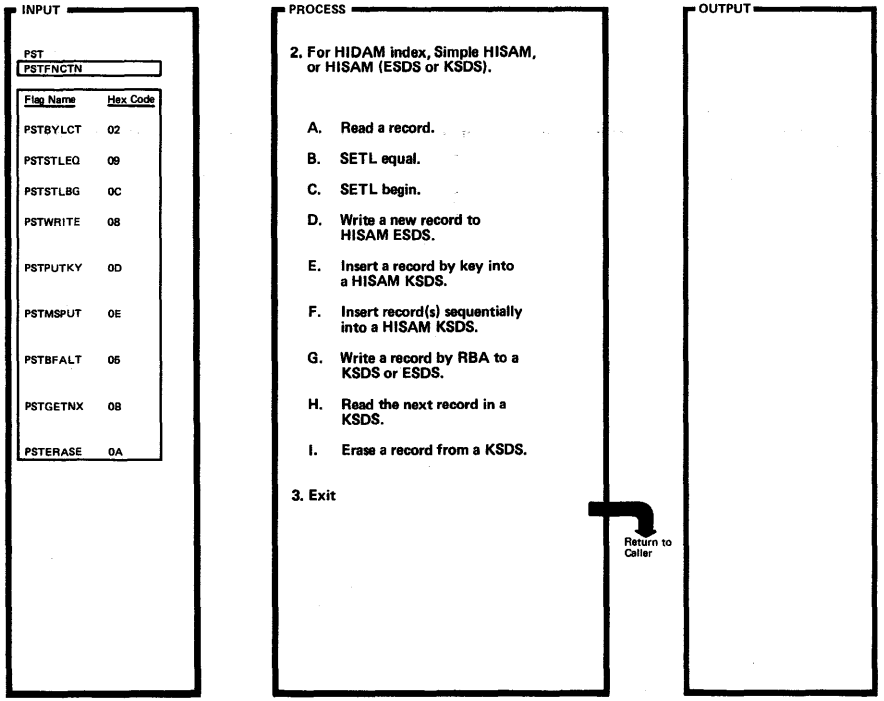
Figure 2-15. DB Buffer Handler (Part 1 of 2)



Extended Description	Routine	Label
1.		
A. Locate relative byte number.	DLZDBH00	BYLCT
B. Locate relative block number.	DLZDBH00	BKLCT
C. Locate a relative byte number and mark buffer altered.	DLZDBH00	BYALT
D. Mark a buffer containing data as altered.	DLZDBH00	BFALT
E.	DLZDBH00	GBSPC
F.	DLZDBH03	MRKEMPT
G.	DLZDBH03	MRKEMPT
H. Purge all buffers altered by a task.	DLZDBH03	PGUSR1

Extended Description	Routine	Label

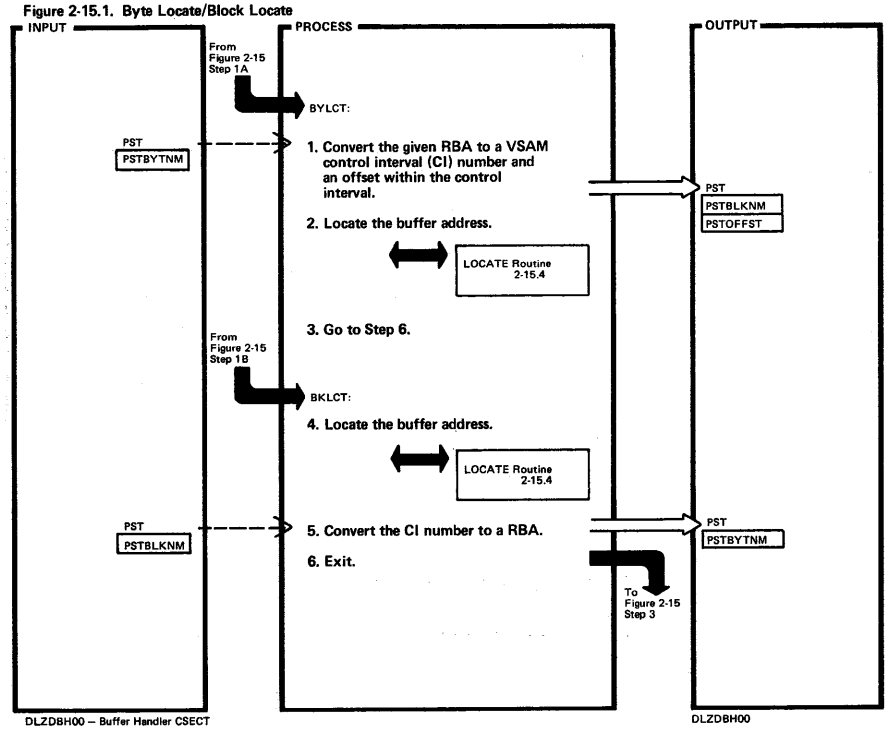
Figure 2-15. DB Buffer Handler (Part 2 of 2)



Extended Description	Routine	Label
2.		
A. Read a record by RBA from a KSDS or ESDS.	DLZDBH02	HSREAD
B. Get a record by root key from a KSDS.	DLZDBH02	STLEQ
C. Read the record containing the first root segment in a KSDS.	DLZDBH02	STLBG
D.	DLZDBH02	LOWRITE
E.	DLZDBH02	PUTKY
F.	DLZDBH02	MSPUT
G.	DLZDBH02	HSWRITE
H.	DLZDBH02	GETNX
I.	DLZDBH02	HSWRITE

Extended Description	Routine	Label

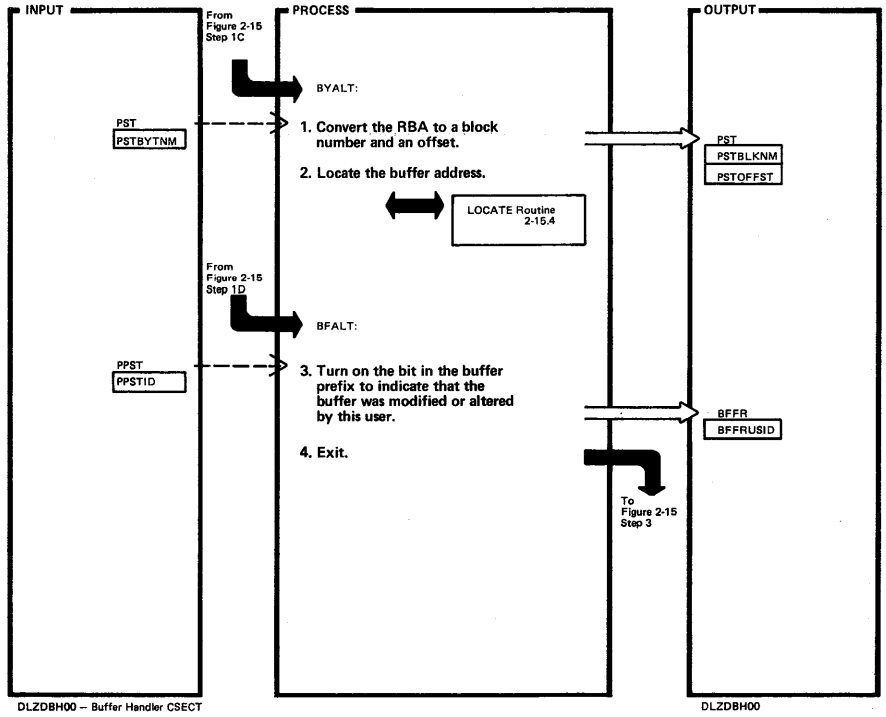
Figure 2-15.1. Byte Locate/Block Locate



Extended Description	Routine	Label
1. The relative byte number for the control interval is retrieved from PSTBYTNM. 5. The same as in Step 1 except that a control interval number is passed in PSTBLKNM.	DLZDBH00	CONVER

Extended Description	Routine	Label

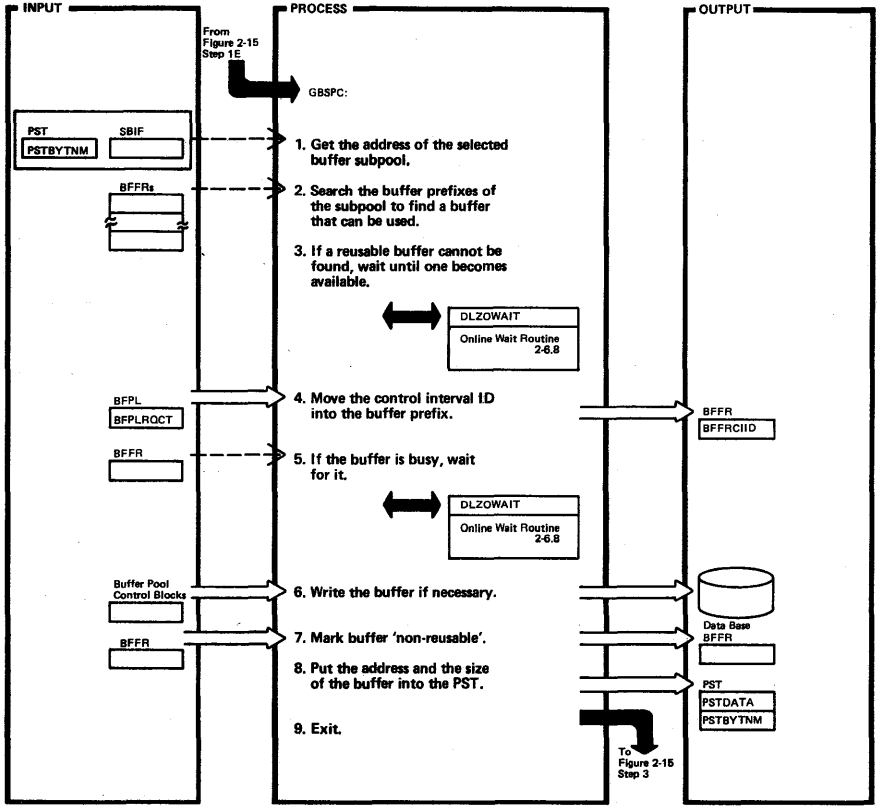
Figure 2-15.2. Byte Alter/Buffer Alter



Extended Description	Routine	Label
1. Byte alter is a combination of byte locate (Figure 2-15.1, Steps 1-3) and buffer alter (Figure 2-15.2, Steps 3 and 4). 3. The bit that is turned on is in the 2-byte field BFFRUSID. The 16 bits correspond from right to left to the user ID indicated in the PPST. If a user ID higher than 16 is assigned, two or more users share the same bit.	DLZDBH00	MARKALT

Extended Description	Routine	Label

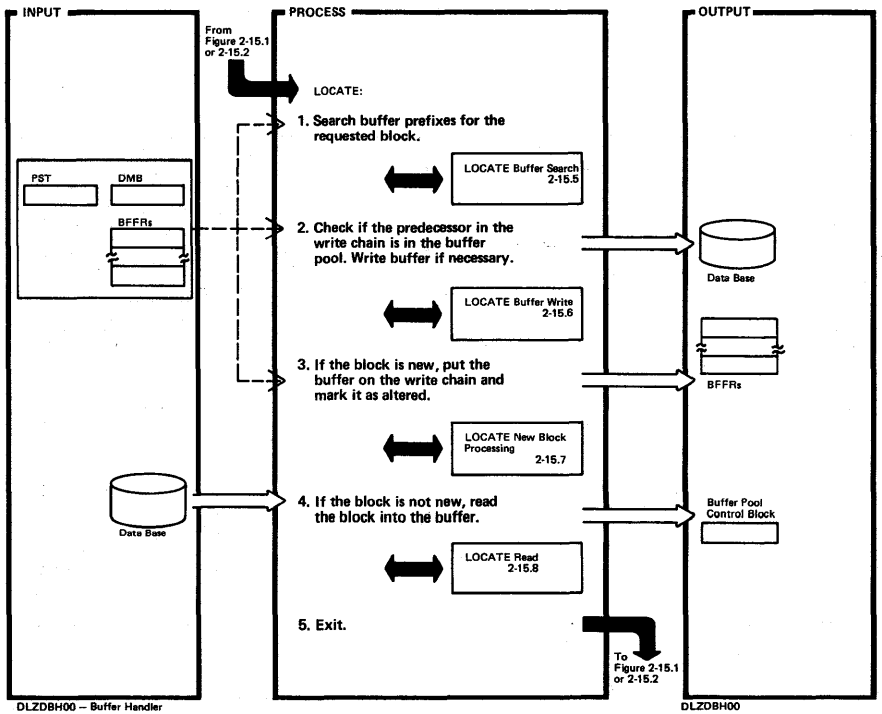
Figure 2-15.3. Get Buffer Space



Extended Description	Routine	Label
1. The subpool information table (SBIF) is used to find a buffer subpool with buffers that contain the least number of bytes needed for this space request. 2. Buffers that are marked non-reusable or are permanent write error buffers cannot be used.		

Extended Description	Routine	Label

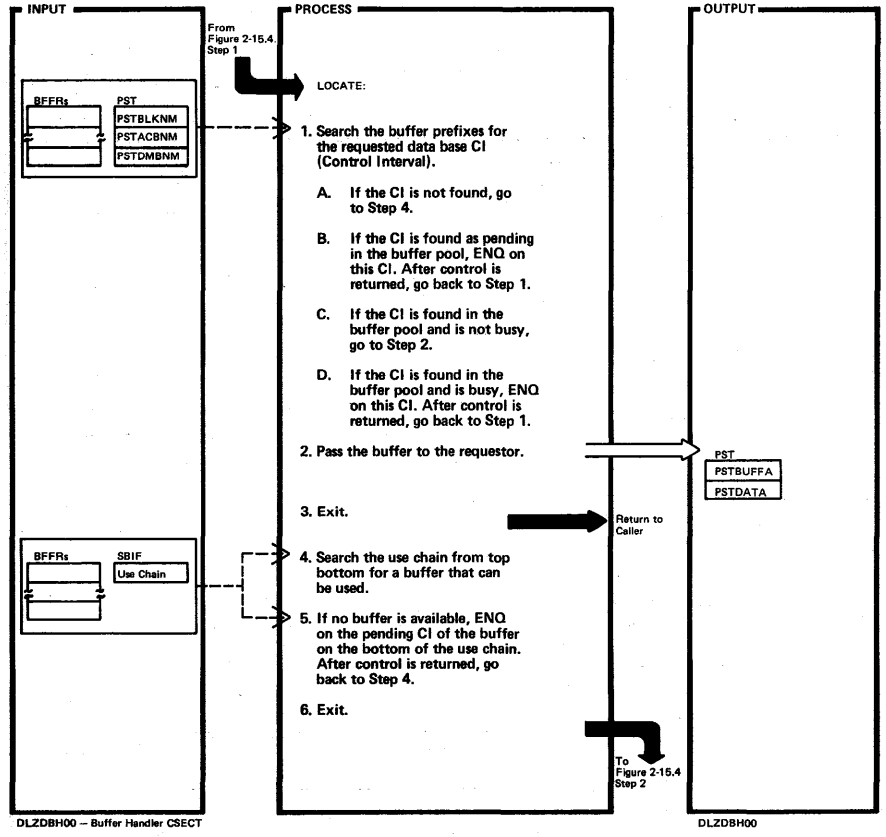
Figure 2-15.4. LOCATE Routine



Extended Description	Routine	Label

Extended Description	Routine	Label

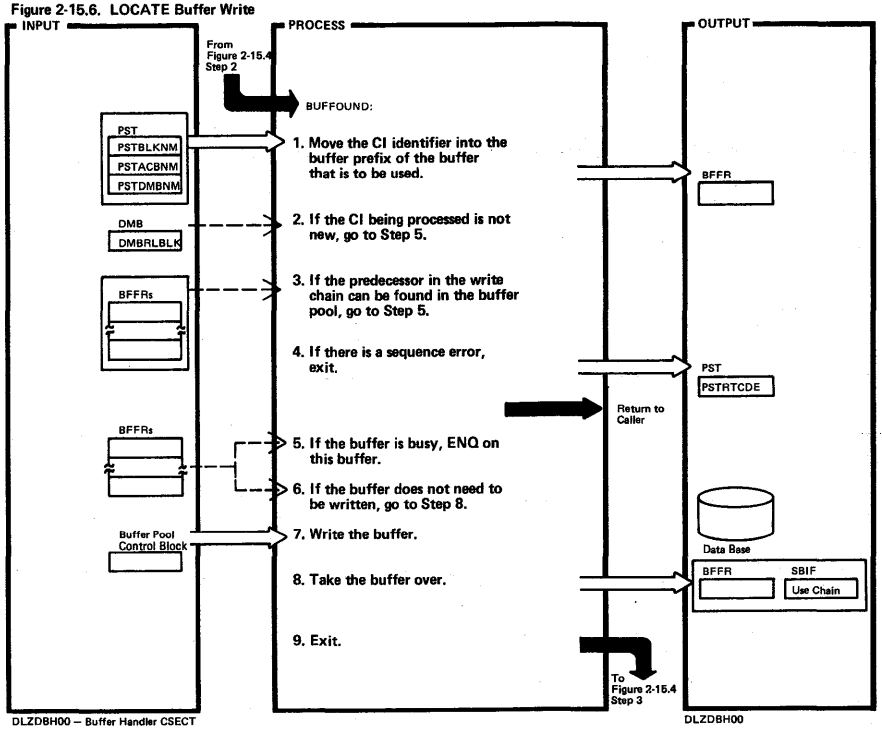
Figure 2-15.5. LOCATE Buffer Search



Extended Description	Routine	Label
2. Put the buffer prefix address into PSTBUFFA and the buffer address into PSTDATA. 4. A buffer can be used if: • It is not marked non-reusable. • It is not a permanent write error buffer. • It is not currently enqueued for a pending CI.		

Extended Description	Routine	Label

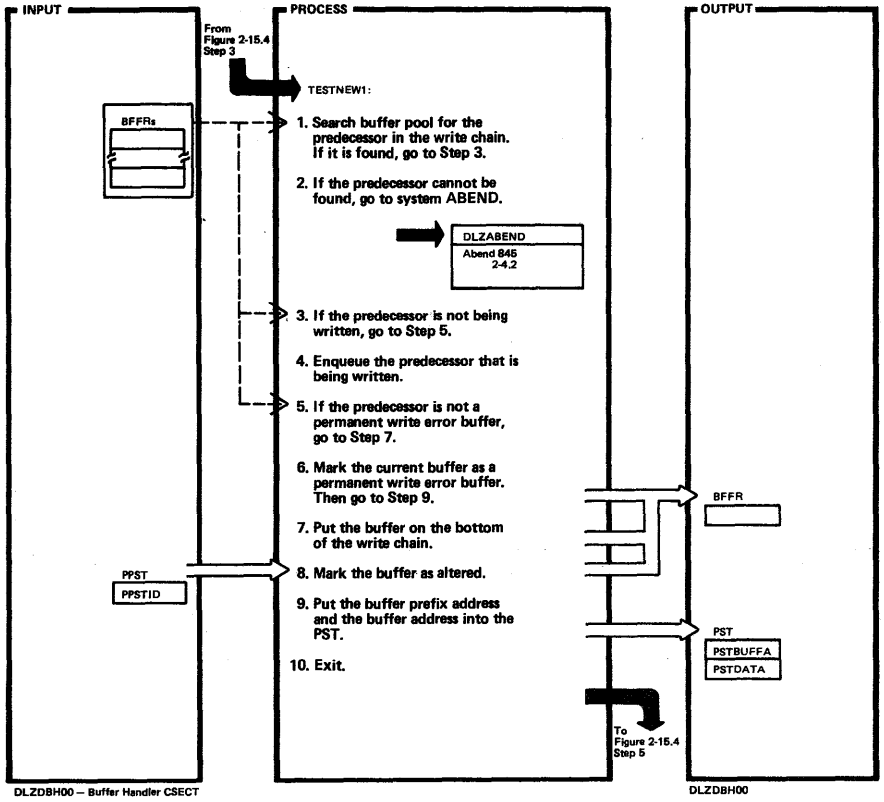
Figure 2-15.6. LOCATE Buffer Write



Extended Description	Routine	Label
1. Moving the CI identifier means enqueuing on the pending CI. 2. This check is made to ensure that the CIs of the data base get initialized in sequence. 4. X'04' is stored in PSTRTCDE. 5. A buffer is busy if: • It is being read into. • It is being written. • It is waiting for its predecessor in the write chain to be written. 8. 'Taking over' a buffer consists of: • Moving the CI identifier from BFFRNPST to BFFRPST. • Turning off BFFRPNNQ and turning on BFFREXNQ in BFFRSW. • Putting the buffer at the top of the use chain. • Clearing the buffer (with zeros).		

Extended Description	Routine	Label

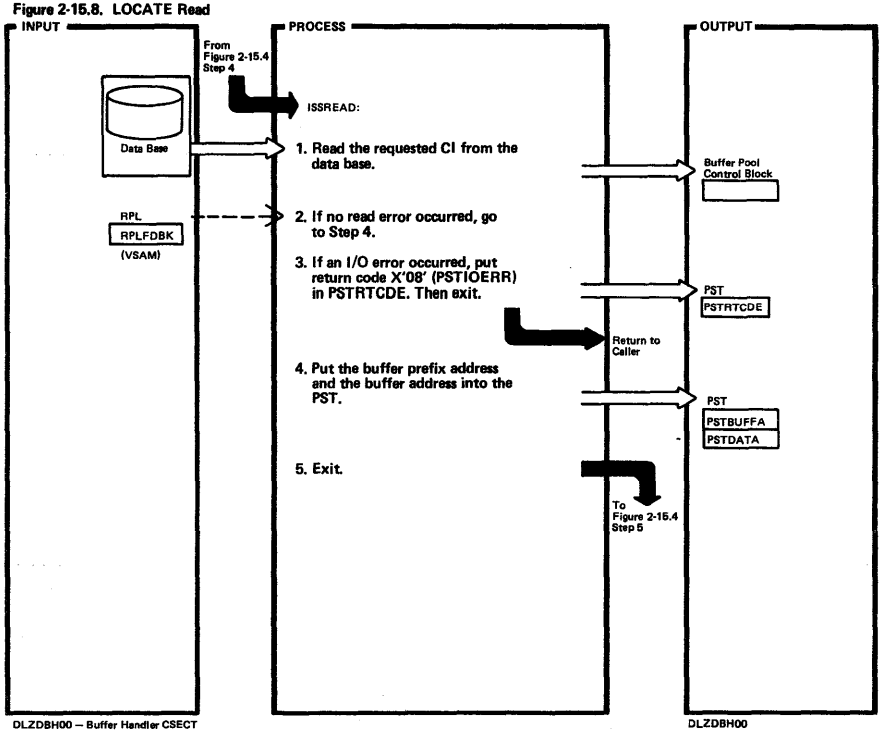
Figure 2-15.7. LOCATE New Block Processing



Extended Description	Routine	Label
4. The purpose for enqueuing the predecessor is to wait for completion of the writing. This is necessary to find out if the buffer is a permanent error buffer.		

Extended Description	Routine	Label

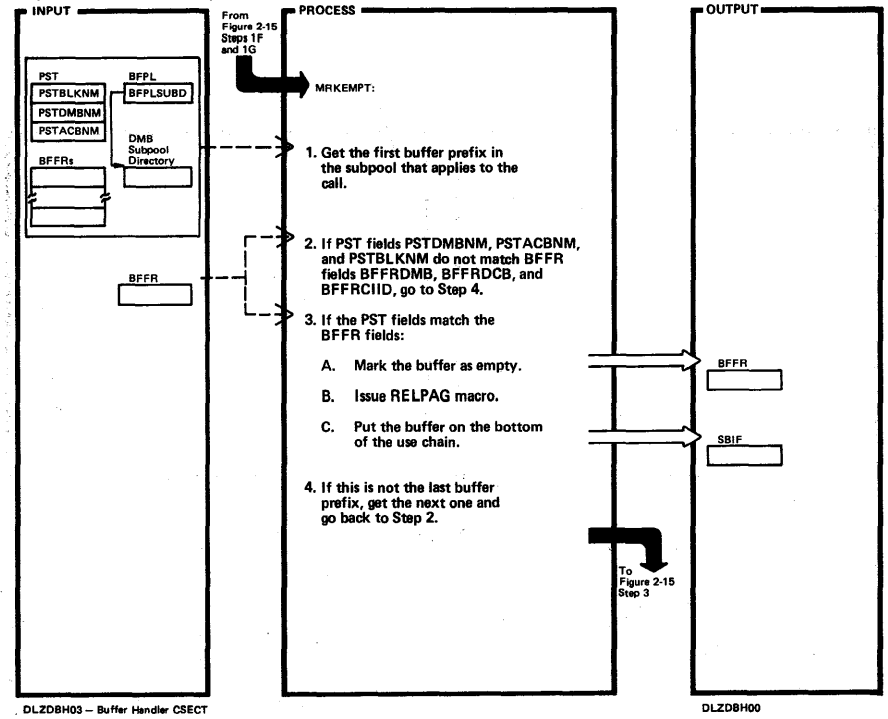
Figure 2-15.8. LOCATE Read



Extended Description	Routine	Label

Extended Description	Routine	Label

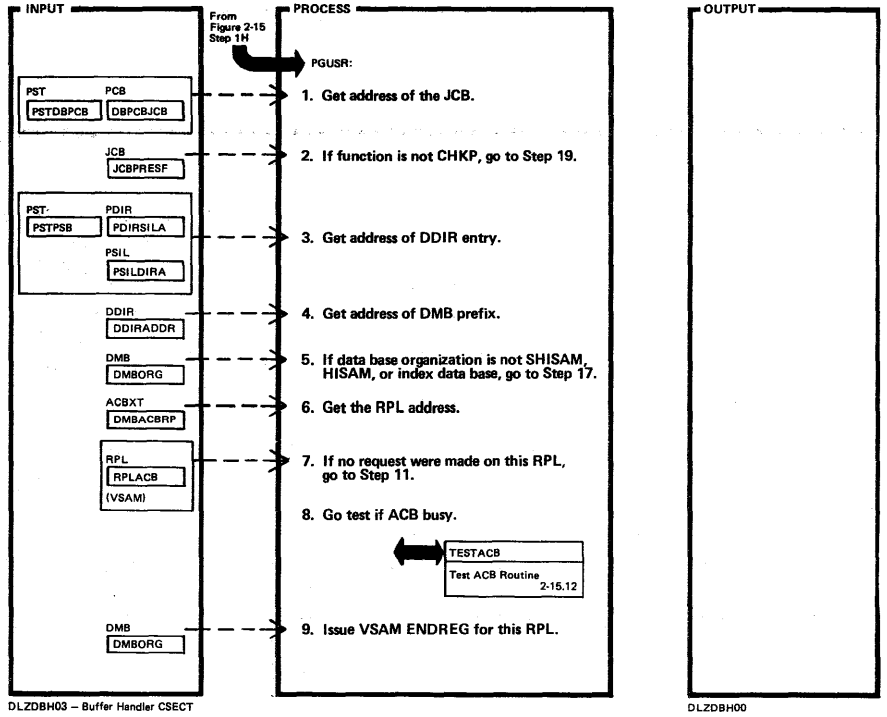
Figure 2-15.9. Free Buffer Space



Extended Description	Routine	Label
1. The DMB subpool directory and the DMB number in PSTDMBNM are used to find the buffer subpool that applies to the call. 2. The caller can have the buffer handler free: • Only one buffer. (PSTDMBNM, PSTACBNM, and PSTBLKNM ≠0). • All buffers of a data set. (PSTDMBNM and PSTACBNM ≠0; PSTBLKNM = 0). • All buffers of a data base. (PSTDMBNM = DMB number of the data base; PSTACBNM and PSTBLKNM = 0).		

Extended Description	Routine	Label

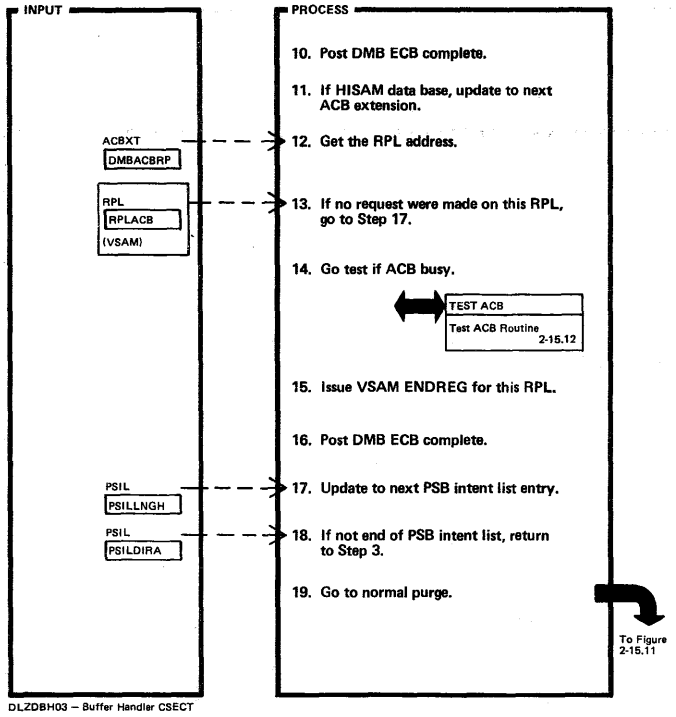
Figure 2-15.10. Purge Buffers (CHKP Function) (Part 1 of 2)

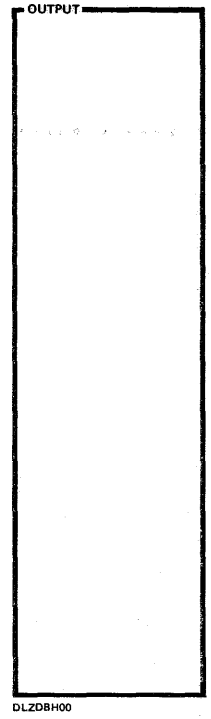


Extended Description	Routine	Label
2. Field JCBPRESF in the JCB is checked for encoded checkpoint function (FUNCCHKP X'85'). If function is not CHKP, go purge the DL/I buffers. 5. If data base organization is not SHISAM (field DMBORG; bitDMBSHIS X'01' not set on), or HISAM (bit DMBISAM1 X'02' not set on), or an index data base (bit DMBNDEX X'08' not set on) go update to next PSIL entry. 9. VSAM ENDREQs are issued for every SHISAM, HISAM, and index data base to ensure that the VSAM buffers are written to the data base.		

Extended Description	Routine	Label

Figure 2-15.10. Purge Buffers (CHKP Function) (Part 2 of 2)

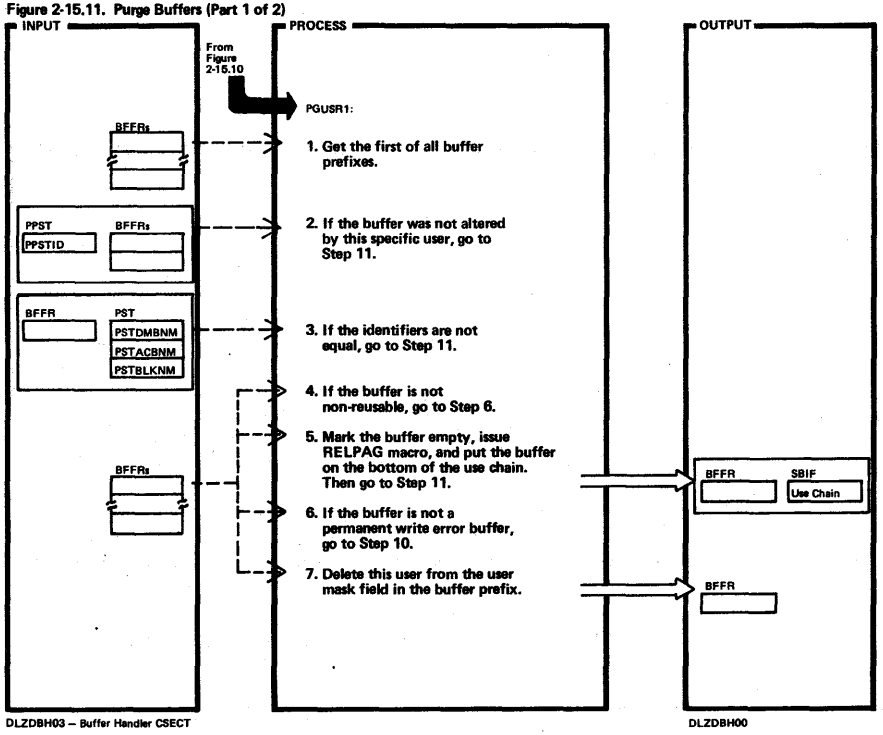




Extended Description	Routine	Label
11. If the entry is for HISAM, update to the next ACB extension. For HISAM, ENDREQs must be issued for both the KSDS and ESDS. 18. Continue scan of the PSB intent list until all have been processed. When processing is completed, go purge the DL/I buffers.		

Extended Description	Routine	Label

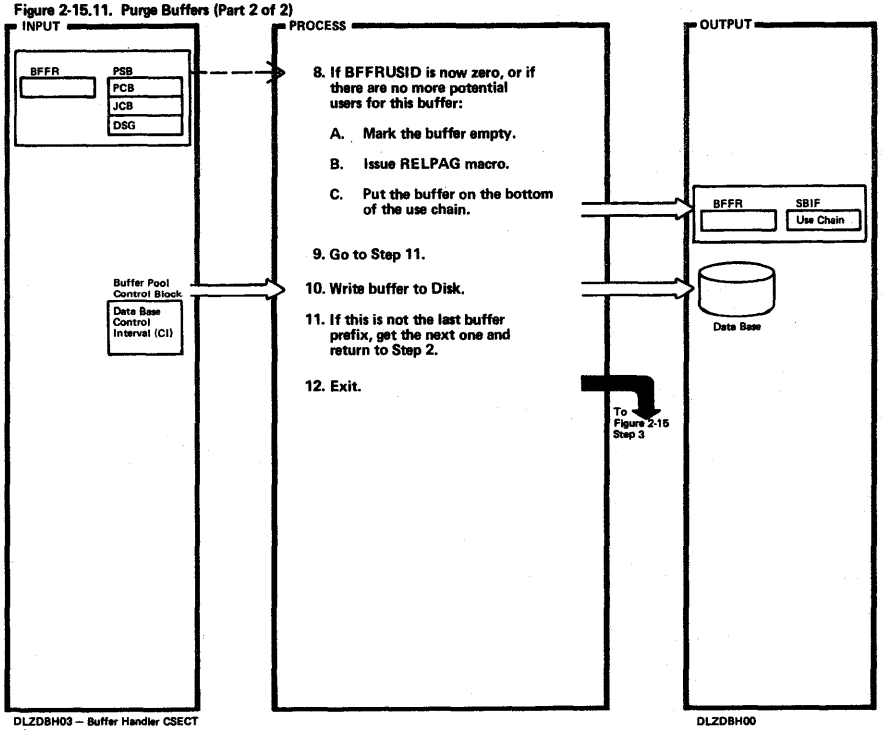
Figure 2-15.11. Purge Buffers (Part 1 of 2)



Extended Description	Routine	Label
1. This routine scans all buffer prefixes. 3. The caller may select a certain data base, a certain data set, or certain buffers to be purged. The choice is indicated by putting the number of the desired item into PSTDMBNM, PSTACBNM, or PSTBLKNM. Zeroes in these fields indicate that purging of all components of the item on the next higher level is desired. This module checks the contents of the above mentioned PST fields against the contents of fields BFFRDMB, BFFRACB, and BFFRCIID in the buffer prefix. 4. Buffers that are non-reusable are freed during a purge call. 6. Permanent write error buffers are not freed until all tasks, which either altered the buffer, or might be interested in it because they use the data base, have terminated.		

Extended Description	Routine	Label
7. Before the bit in BFFRUSID, which corresponds to the user identifier (in the PPST) of the current task, is turned off, a check is made whether any tasks are active that would share the bit with the current task. (Refer to the notes in the Figure for routine BFALT.)		

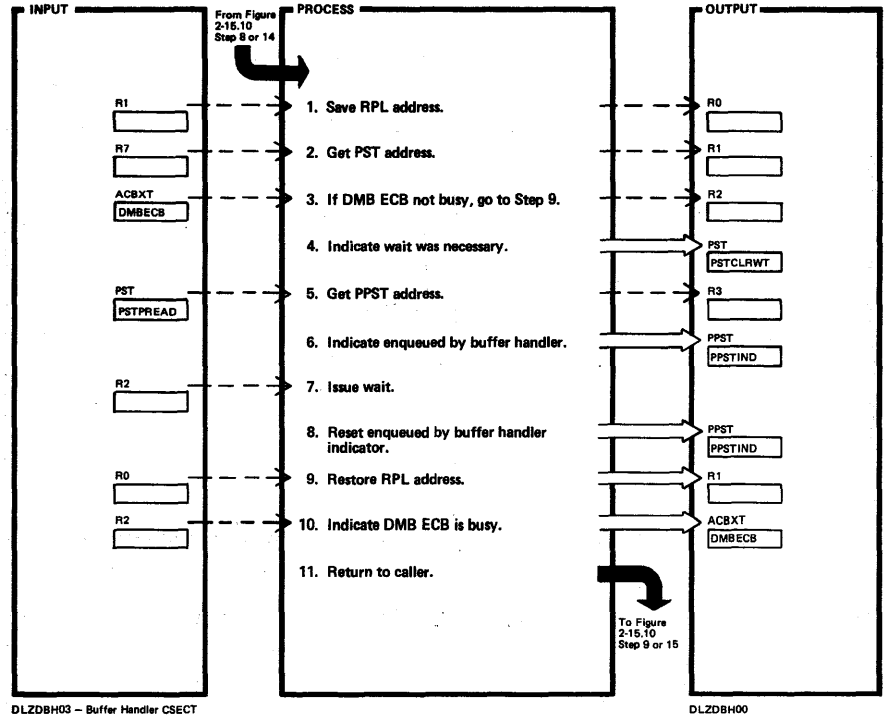
Figure 2-15.11. Purge Buffers (Part 2 of 2)



Extended Description	Routine	Label
8. A task is a potential user of a buffer if at least one of the DSGs in the PSB has the same DMB and ACB as in fields BFFRDMB and BFFRACB of the buffer prefix.		

Extended Description	Routine	Label

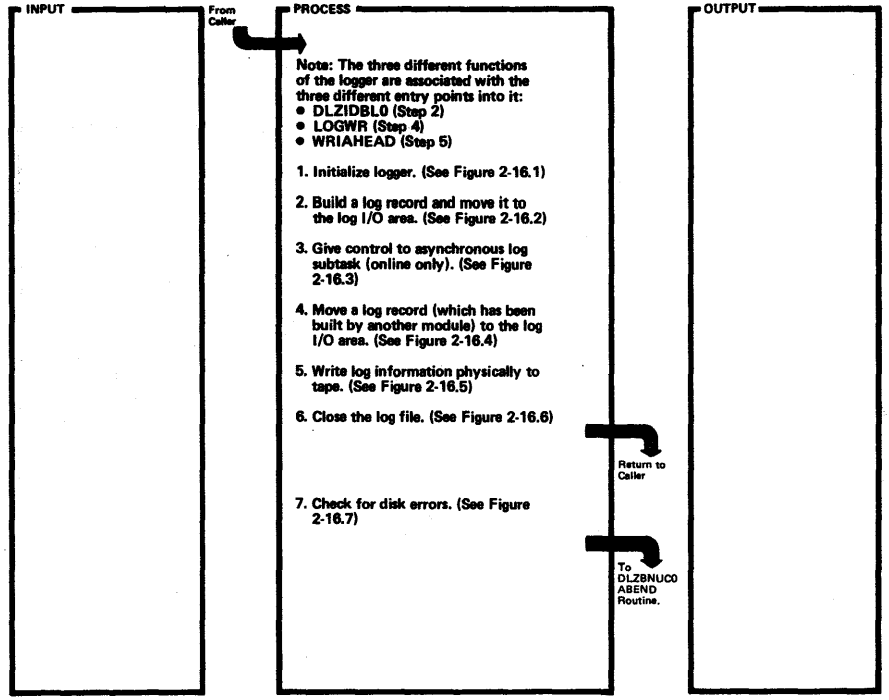
Figure 2-15.12. Test ACB Routine



Extended Description	Routine	Label
3. Byte 2 of DMB ECB set to X'80'.	TESTACB	
4. Bit PSTIWAIT in field PSTCLRWT in PST set on.		
6. Bit PPSTBF in field PPSTIND in PPST set on.		
7. DLZIWAIT macro issued.		
8. Bit PPSTBF in field PPSTIND in PPST set off.		
9.		NOWAIT
10. X'80' in byte 2 of DMBECB turned off.		

Extended Description	Routine	Label

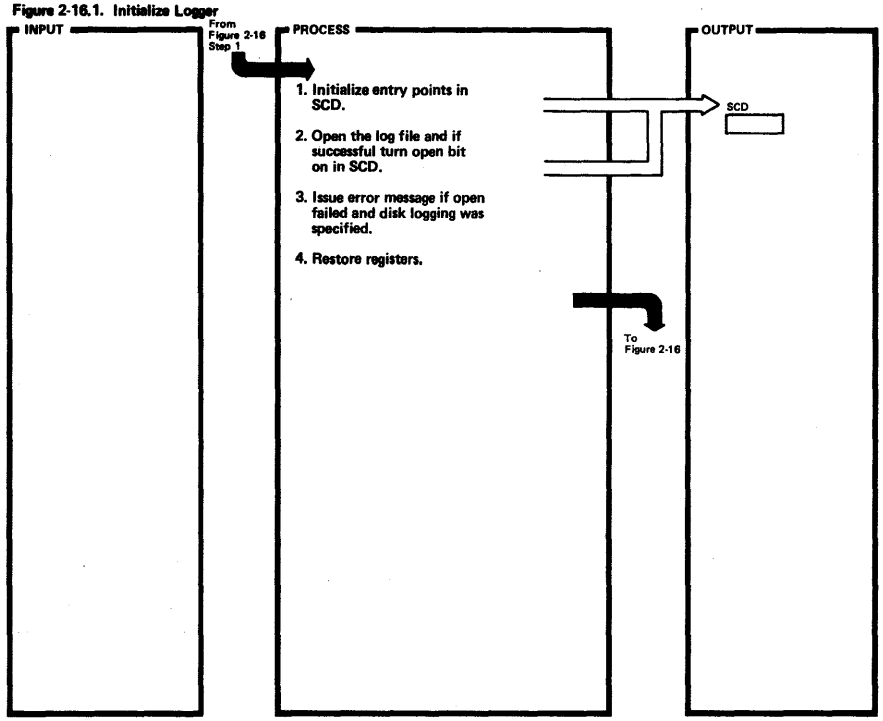
Figure 2-16. DB Logger (Overview)



Extended Description	Routine	Label
1.	DLZRDBL0	DLZRDBL0
2.		DLZIDBL0
3.		ONLINT ONLLOGWR
4.		LOGWR
5.		WRIAHEAD
6.		LOGCLOSE
7.		PUTERROR

Extended Description	Routine	Label

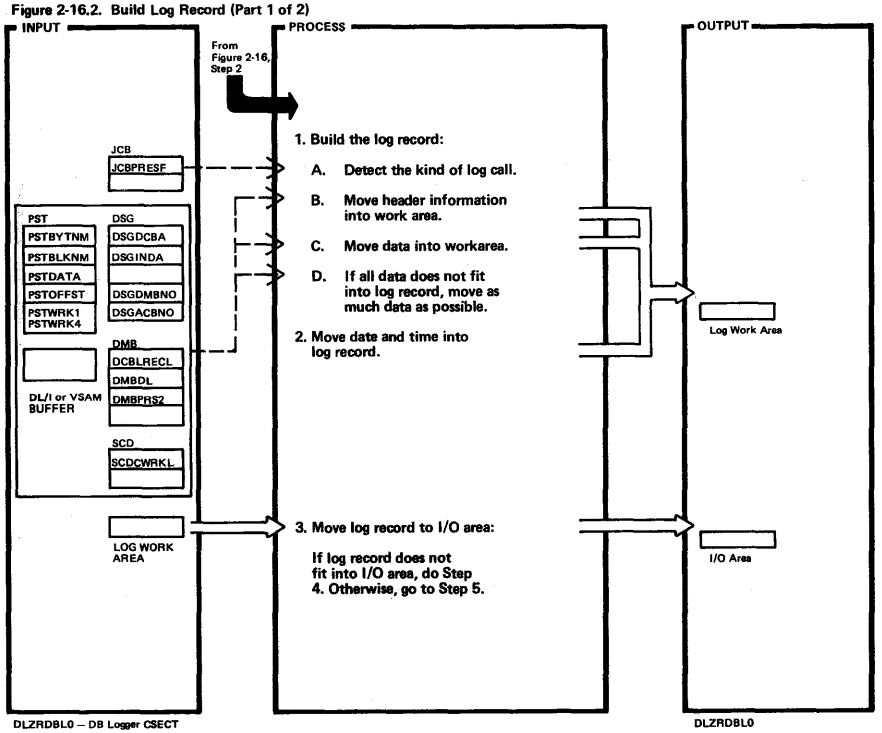
Figure 2-16.1. Initialize Logger



Extended Description	Routine	Label
1. The entry point to the logger module initially points to the initialization routine. After initialization it contains the entry point of DLZIDBL0 All of the entry points to the various logger routines are in the SCD after initialization. 2. If tape logging is specified, the DTF is opened. If disk logging is specified, the ACB is opened and tested if successful. 3. Message DLZ020I is issued if an open error occurred with disk logging. Message DLZ077I is issued if the log was opened successfully with disk logging.	DLZRDBL0	

Extended Description	Routine	Label

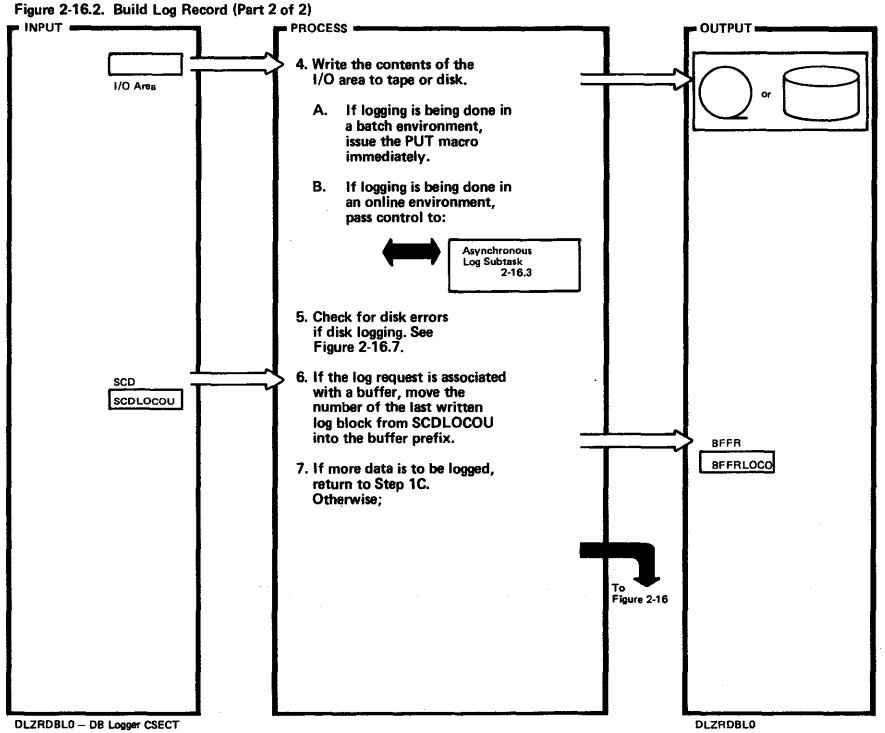
Figure 2-16.2. Build Log Record (Part 1 of 2)



Extended Description	Routine	Label
1. Depending on the kind of log record being processed, the logger will build one of the following type of log record: • Physical insert record • Physical replace record • Physical delete record • Logical delete record • Pointer maintenance record. The maximum logical record size for a log record is 512 bytes. The blocks are undefined with a maximum of 1024 bytes.	DLZRDBL0	DLZIDBL0

Extended Description	Routine	Label

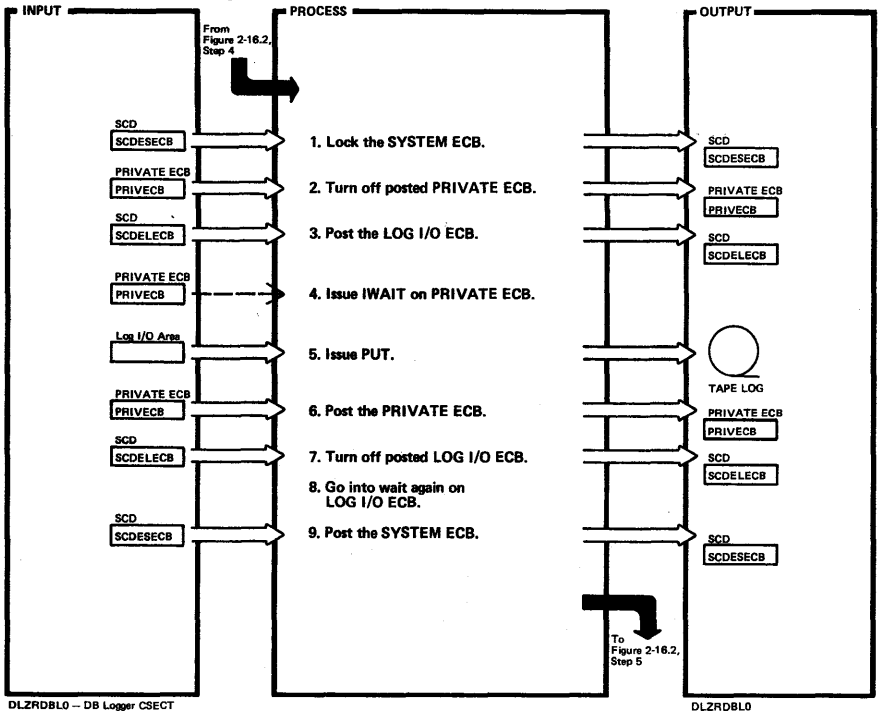
Figure 2-16.2. Build Log Record (Part 2 of 2)



Extended Description	Routine	Label
4. B. In an online environment, the PUT macro is issued from the Asynchronous Log Subtask in order to avoid losing tasks when EOV is encountered on the log tape. 6. The purpose for keeping the number of the last written log block in the SCD and in the BFFR is to enable DLZDBH00 to determine if a log record has to be written out before an update is applied to a data base. 7. There will be more data to be logged if all data did not fit into the log record. See Step 1D.		

Extended Description	Routine	Label

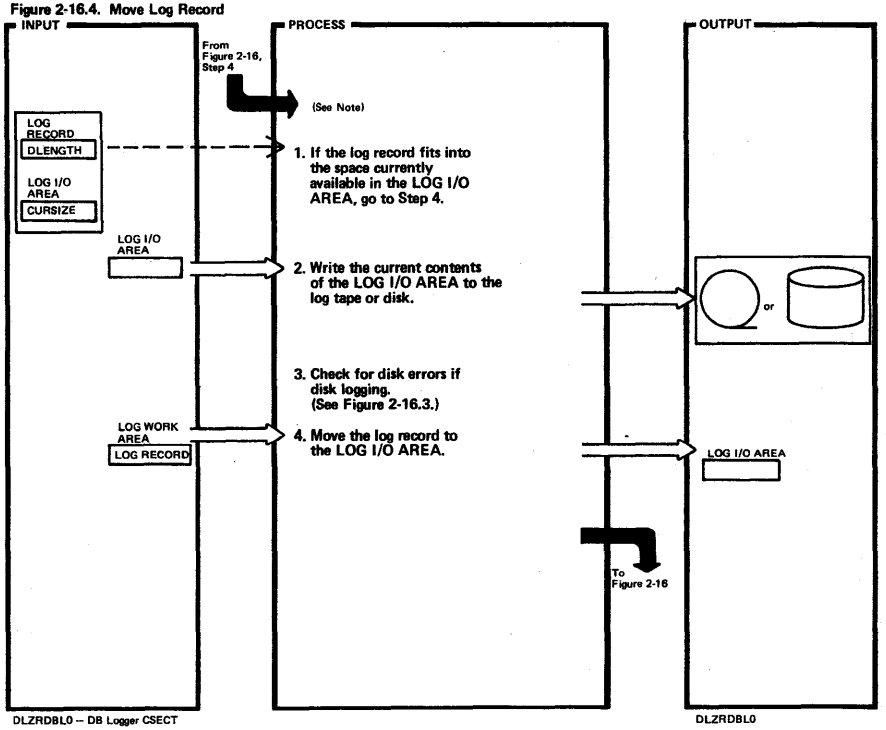
Figure 2-16.3. Asynchronous Log Subtask



Extended Description	Routine	Label
Steps 1, 2, 3, 4, and 9 are performed within CSECT DLZRDBL0.	DLZRDBL0	ONLINT
Steps 5, 6, 7, and 8 are performed within the Asynchronous Log Writer Subtask.	DLZRDBL0	ONLLOGWR
1. The SYSTEM ECB is used for communication between DLZRDBL0 and DLZODP. It is locked in order to prevent any other task from entering the logger while the I/O is going on.		
2. The PRIVATE ECB is used for communication about the completion of I/O between the Asynchronous Log Subtask a and DLZRDBL0.		
3. The LOG I/O ECB is used for communication about the need to issue a PUT macro between DLZRDBL0 and the Asynchronous Log Subtask.		

Extended Description	Routine	Label
The Asynchronous Log Subtask is waiting on this ECB and when it gets posted, DOS/VS will mark this subtask as dispatchable.		
4. IWAIT will have the effect that the DL/I 'maintask' will be put into wait. The Asynchronous Log Subtask can then be started by DOS/VS.		

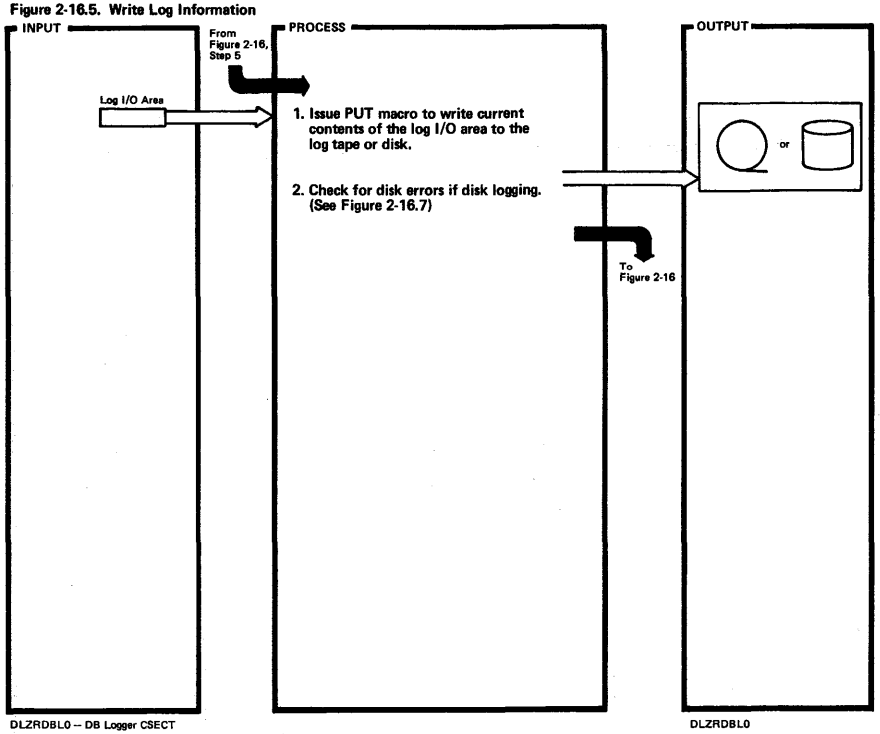
Figure 2-16.4. Move Log Record



Extended Description	Routine	Label
Note: This function is used • Open log records (ID X'2F') • Scheduling records (ID X'08') • Termination records (ID X'07') • Checkpoint records (ID X'41')	DLZRDBL0	LOGWR

Extended Description	Routine	Label

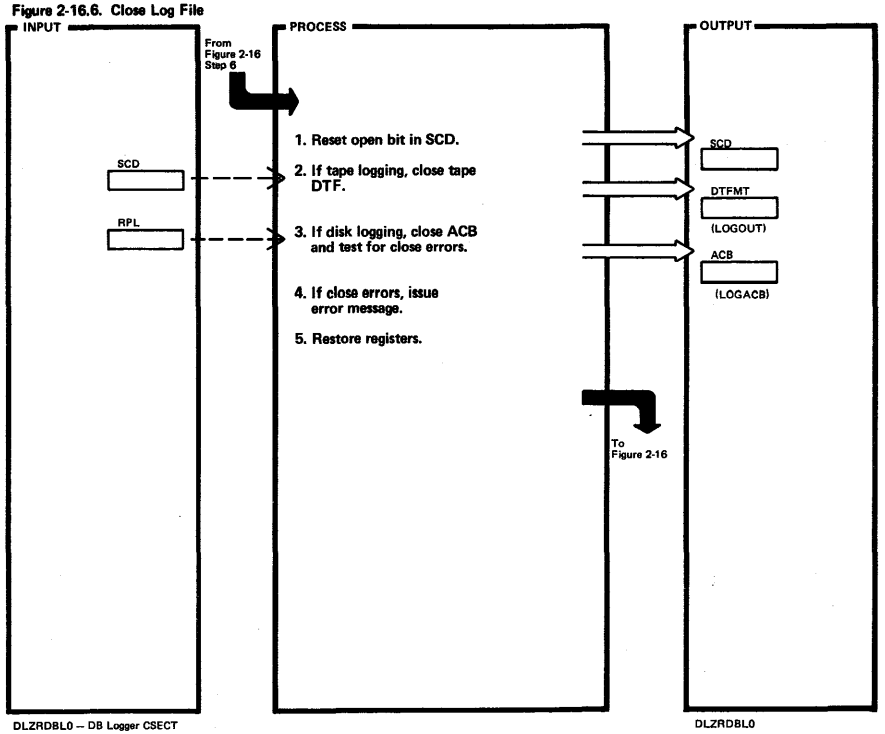
Figure 2-16.5. Write Log Information



Extended Description	Routine	Label
1. This function is used by DLZDBH00, when log information associated with a data base update has not been written to tape at the time the data base update was being done.	DLZRDBL0	WRIAHEAD

Extended Description	Routine	Label

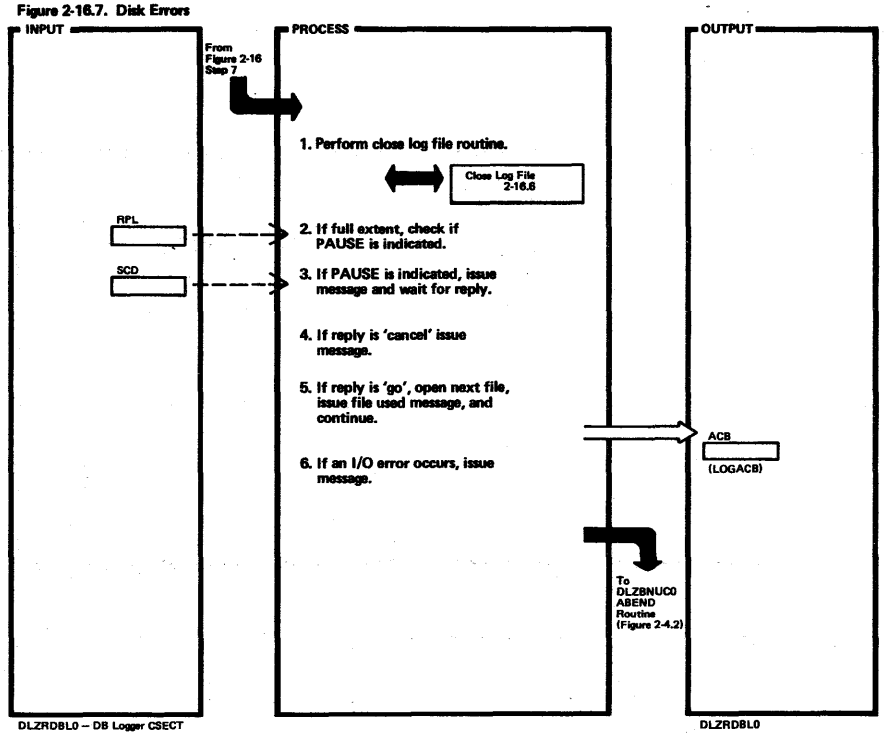
Figure 2-16.6. Close Log File



Extended Description	Routine	Label
1. Either the tape or disk log file is closed in this subroutine. This subroutine is used by DLZODP, DLZRDBL0, DLZRRC00, and DLZBNUC0. 4. Message DLZ021I is issued.	DLZIDBL0	LOGCLOSE

Extended Description	Routine	Label

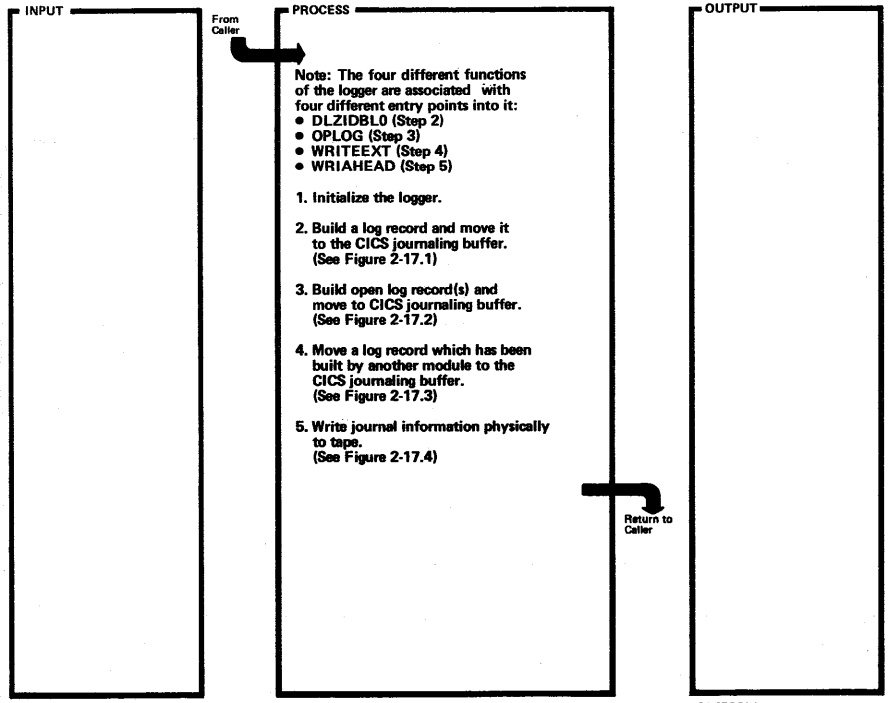
Figure 2-16.7. Disk Errors



Extended Description	Routine	Label
1. The log file is closed so that the operator could dump the file (optional) before continuing.	DLZIDBL0	LOGCLOSE
2. Checks to see if the user specified PAUSE on the DL/I control parameter.	DLZRDBL0	PUTERROR
3. Message DLZ076I is issued.		
4. Message DLZ077I is issued.		
5. Message DLZ079I is issued.		
6. If the reply is 'GO', a check is made to determine if 1 or 2 disk files are being used for logging. If there are 2 files, the second file is opened and control is returned to the PUT routine.		
Message DLZ004I is issued.		

Extended Description	Routine	Label

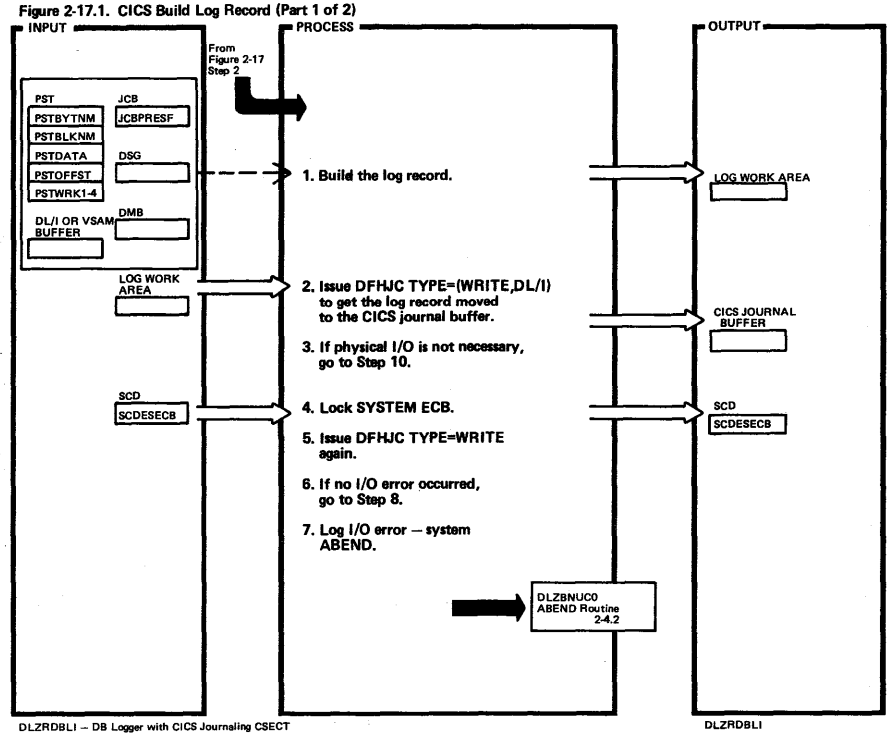
Figure 2-17. CICS Journal Logger (Overview)



Extended Description	Routine	Label
1. Move all of the entry points to the logger into the SCD.	DLZRDBL1	DLZRDBL0
2.		DLZRDBL0
3.		OPLOG
4.		WRITEEXT
5.		WRIAHEAD

Extended Description	Routine	Label

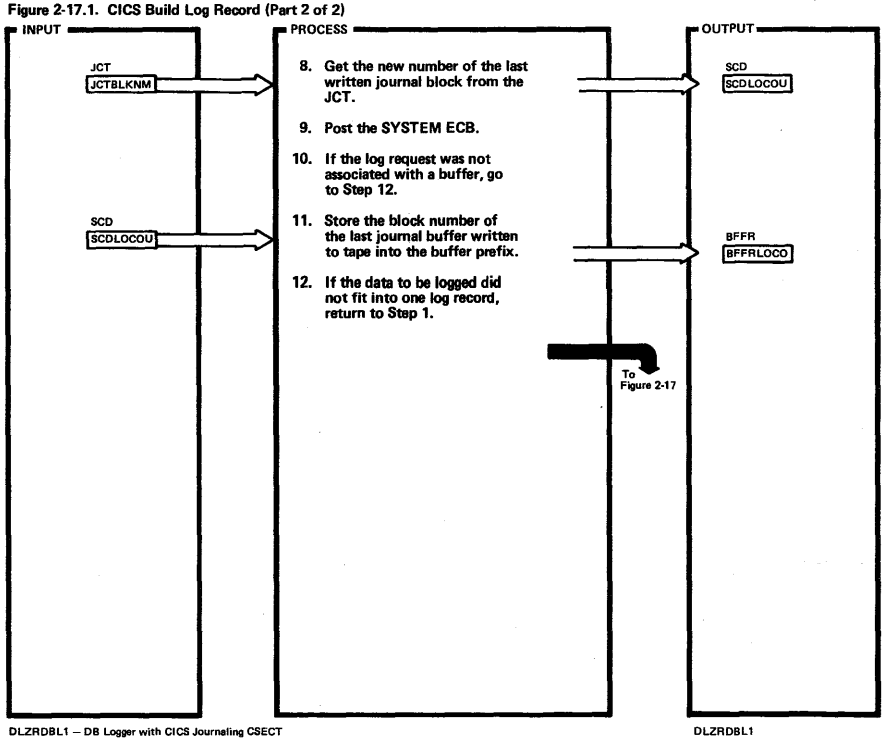
Figure 2-17.1. CICS Build Log Record (Part 1 of 2)



Extended Description	Routine	Label
1.	DLZRDBLI	DLZRDBL0
4. The SYSTEM ECB is locked in order to prevent any other task from entering the logger while the I/O is going on.		IONEC1

Extended Description	Routine	Label

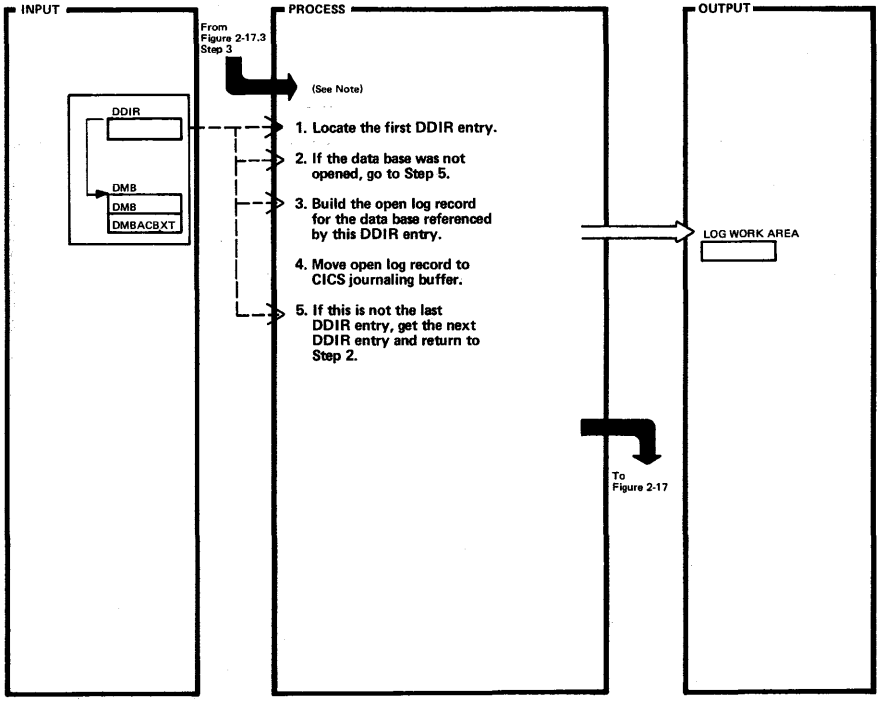
Figure 2-17.1. CICS Build Log Record (Part 2 of 2)



Extended Description	Routine	Label
8. The purpose for keeping the CICS event control number is to enable DLZRBH00 to determine if a log buffer has to be written before an update is applied to a data base.	DLZRDBL1	GETECN

Extended Description	Routine	Label

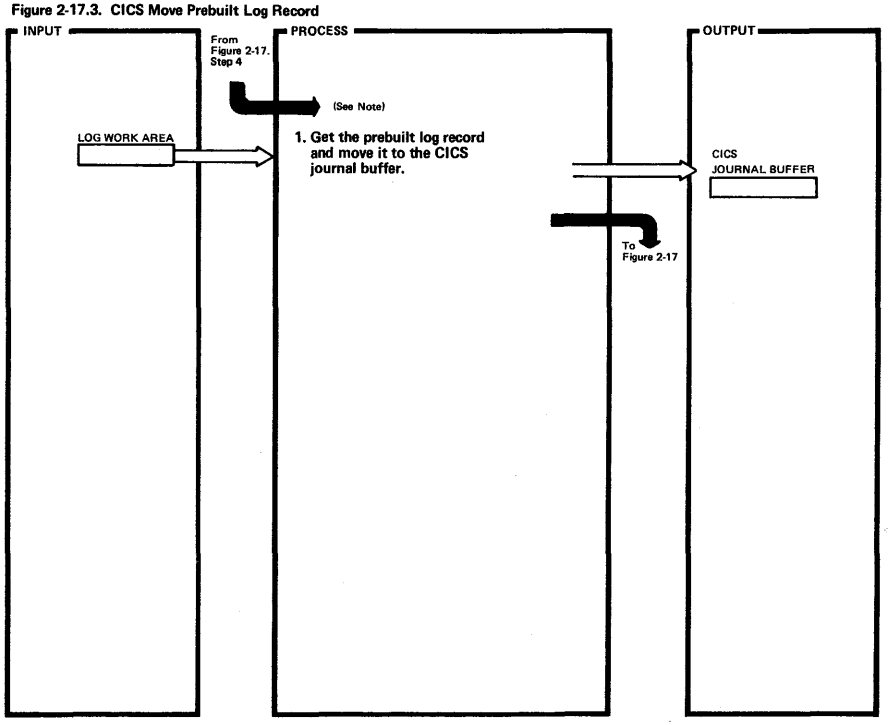
Figure 2-17.2. CICS Move Log Record



Extended Description	Routine	Label
Note: Since the CICS journal tape is not yet open at DL/I initialization, the open log record(s) are built and moved before the first scheduling call is logged. 2. A data base might not have been opened because of the OPEN= DEFERRED option or because of an open error. 4. See Figure 2-17.1, Steps 4-9.	DLZRDBL1	OPLOG

Extended Description	Routine	Label

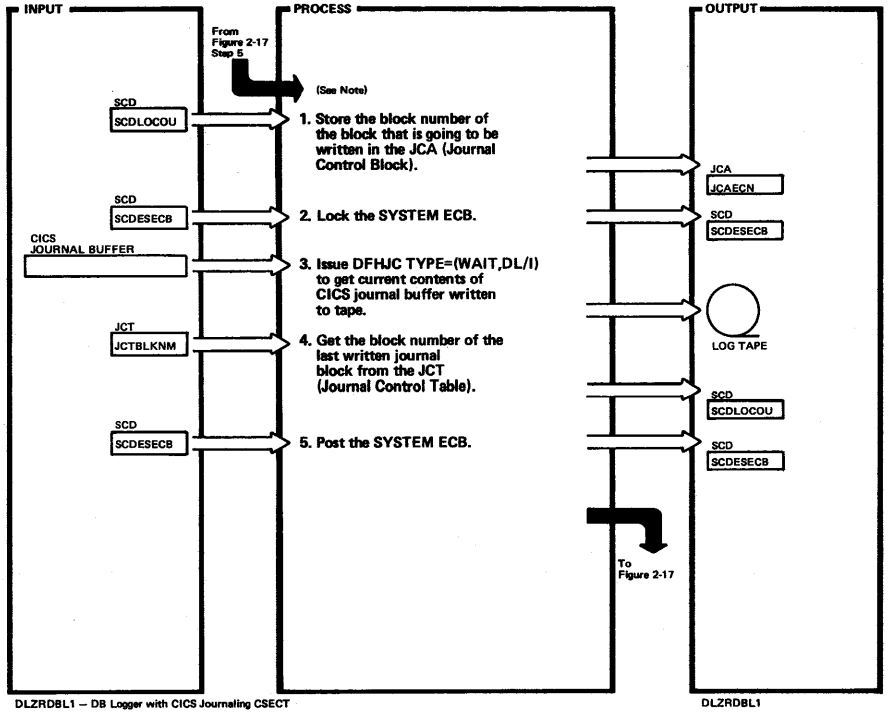
Figure 2-17.3. CICS Move Prebuilt Log Record



Extended Description	Routine	Label
Note: This function applies to scheduling and termination records built by the scheduling termination routine. 1. See Figure 2-17.1, Steps 4-9.	DLZRDBL1	WRITEEXT

Extended Description	Routine	Label

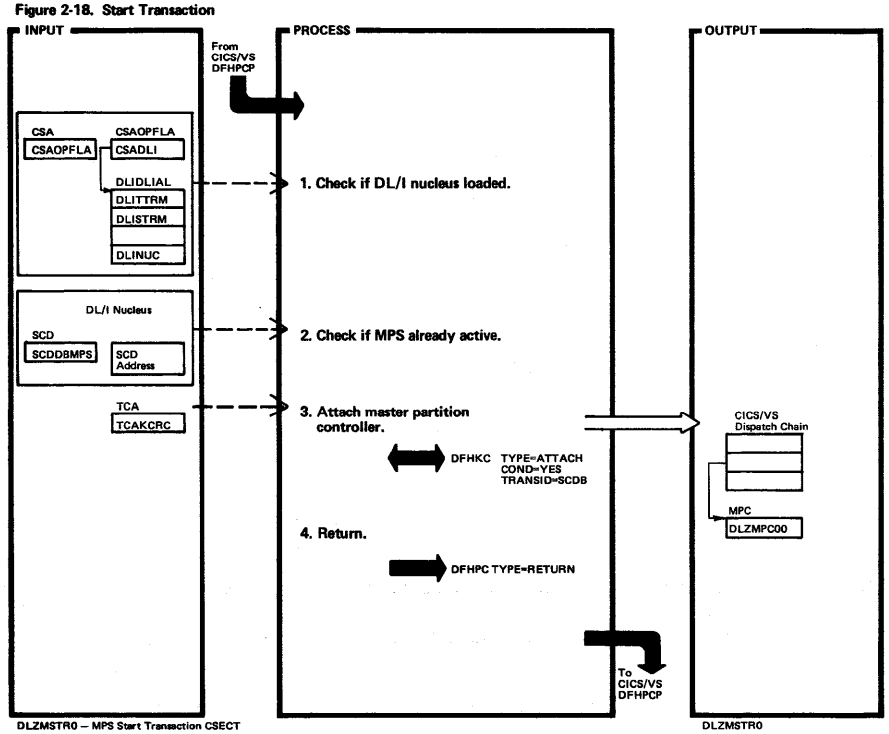
Figure 2-17.4. CICS Log Writing



Extended Description	Routine	Label
Note: This function is used by DLZDBH00 when log information associated with a data base update was not written to tape when the data base update was being done. 1. Refer to note for Step 8 of Figure 2-17.1. 2. Refer to note for Step 4 of Figure 2-17.1.	DLZRDBL1	WRIAHEAD

Extended Description	Routine	Label

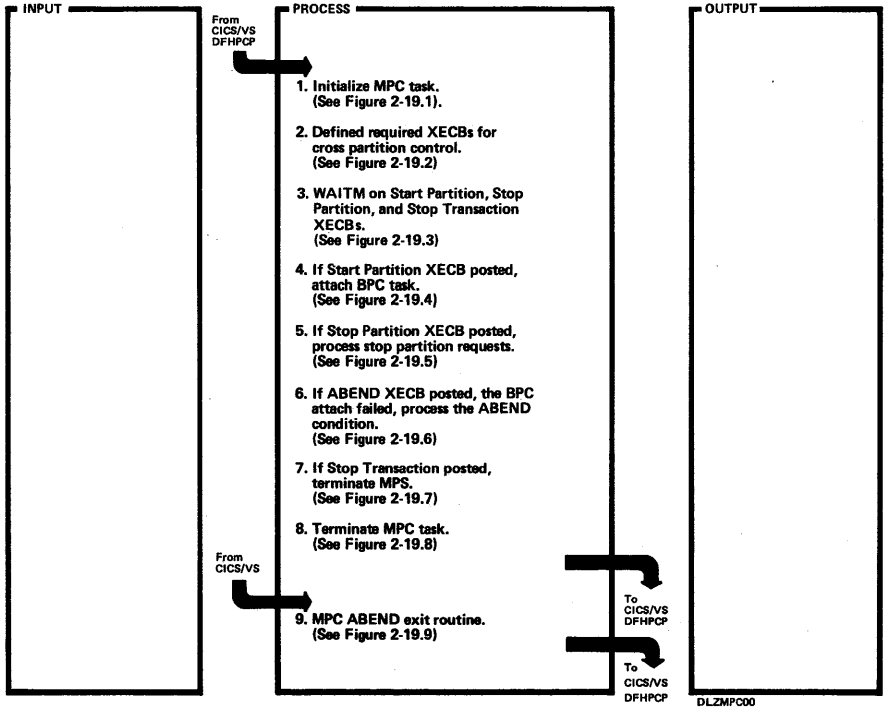
Figure 2-18. Start Transaction



Extended Description	Routine	Label
1. Module identifier (DLZMSTR0) is defined here. Write message DLZ097I if nucleus not loaded or not active and go to Step 4. 2. Write message DLZ101I if flag SCDXECB indicates MPS XECBs already defined and go to Step 4. 3. Write message DLZ083I if attach fails with a return code in TCAKCRC and go to Step 4. 4.	DLZMSTR0	DLZMSTR0 RETURN

Extended Description	Routine	Label

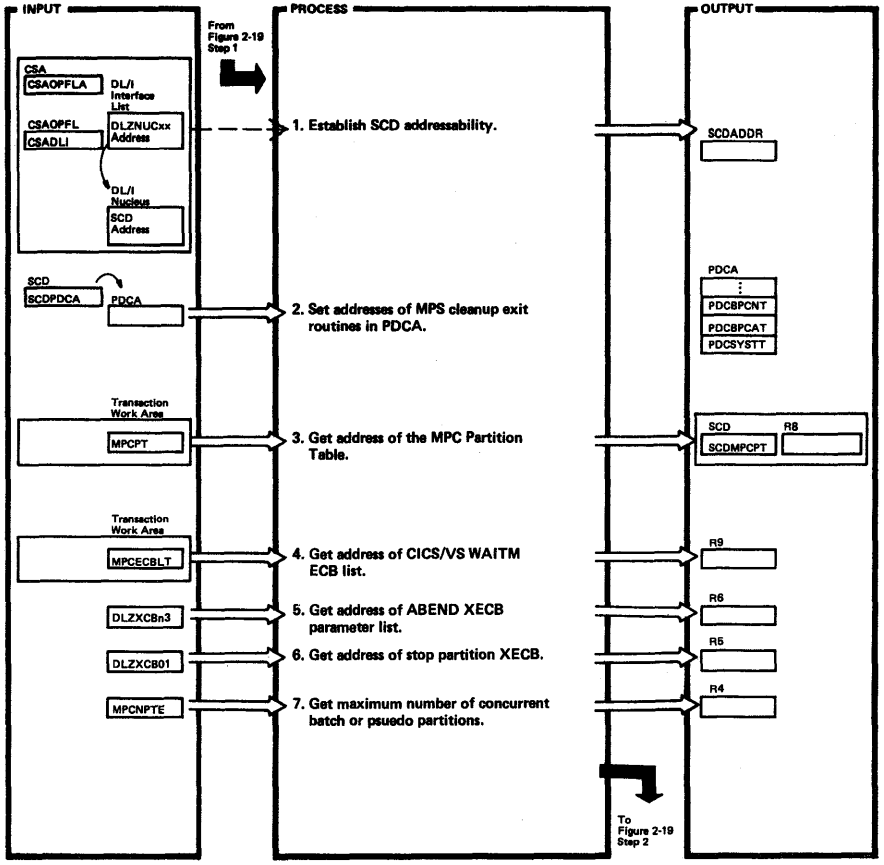
Figure 2-19. Master Partition Controller (Overview)



Extended Description	Routine	Label
1.		DLZMPC00 MPCSTART
2.		MPCDEFIN
3.		MPCWAIT
4.		MPCSTRP
5.		MPCSTOP
6.		MPCABNP
7.		MPCSTRN
8.		MPCEXIT
9.		MPCABEXT

Extended Description	Routine	Label

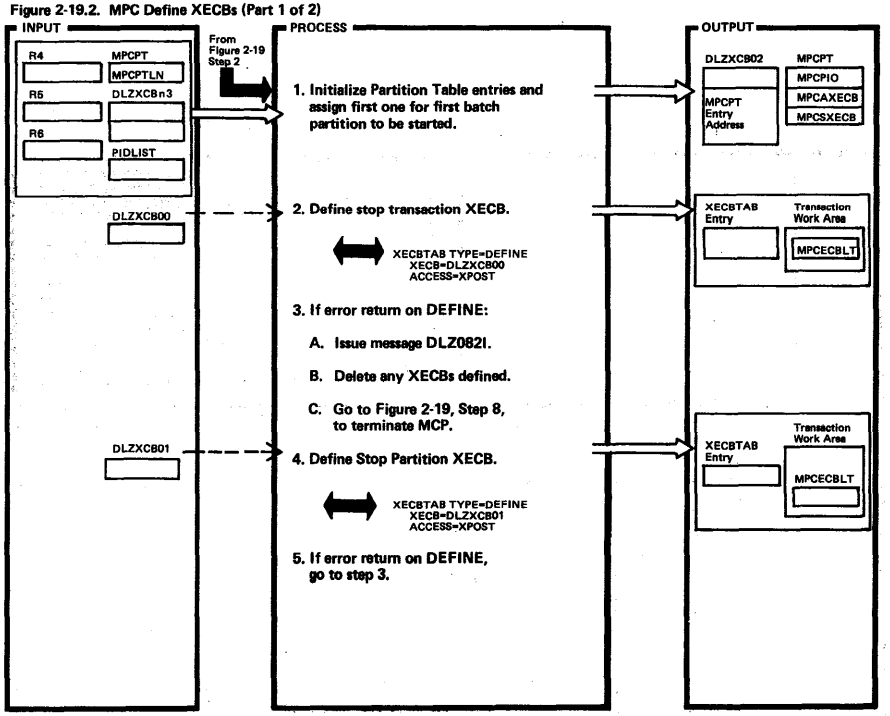
Figure 2-19.1. MPC Task Initialization



Extended Description	Routine	Label
1. Module identifier (DLZMPC00 vmp) is defined here. MPC is attached by the MPS start transaction (DLZMSTR0) via CICS/VS. Ignore request if DL/I is not defined to CICS/VS or the nucleus is not loaded. 3. The transaction work area is a logical extension of the TCA. 7. This controls the number of partition table entries that will be initialized later. This is an equated value in the partition table DSECT (DLZMPCPT).	DLZMPC00	DLZMPC00 MPCSTART

Extended Description	Routine	Label

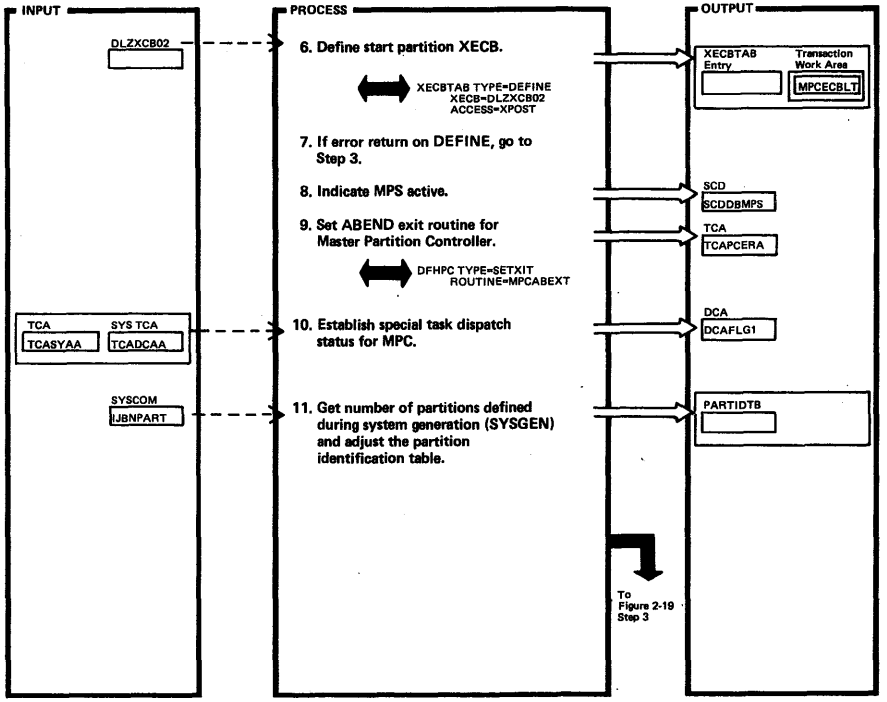
Figure 2-19.2. MPC Define XECBs (Part 1 of 2)



Extended Description	Routine	Label
1. The XECB identifier is an alphabetic character assigned by this routine. There is no relation between this identifier and any particular batch partition. The addresses of an ABEND XECB and the stop partition XECB are placed in each partition table entry even though no XECBs have been defined at this point. The currently defined values of 'n' are the alphabetic characters L-A. 2. DLZXCB00 is the XECB name to stop the MPS transaction.	DLZMAC00	MPCDEFIN

Extended Description	Routine	Label
3. In all steps where a message is issued: • R1 is set up with the applicable message parameter list. • Control is passed to the MPC message writer at MPCMSGRT. • BALR to the DL/I online message module, DLZERMSG, to write the message. 4. DLZXCB01 is the XECB name to stop a partition.		XECBDFN1

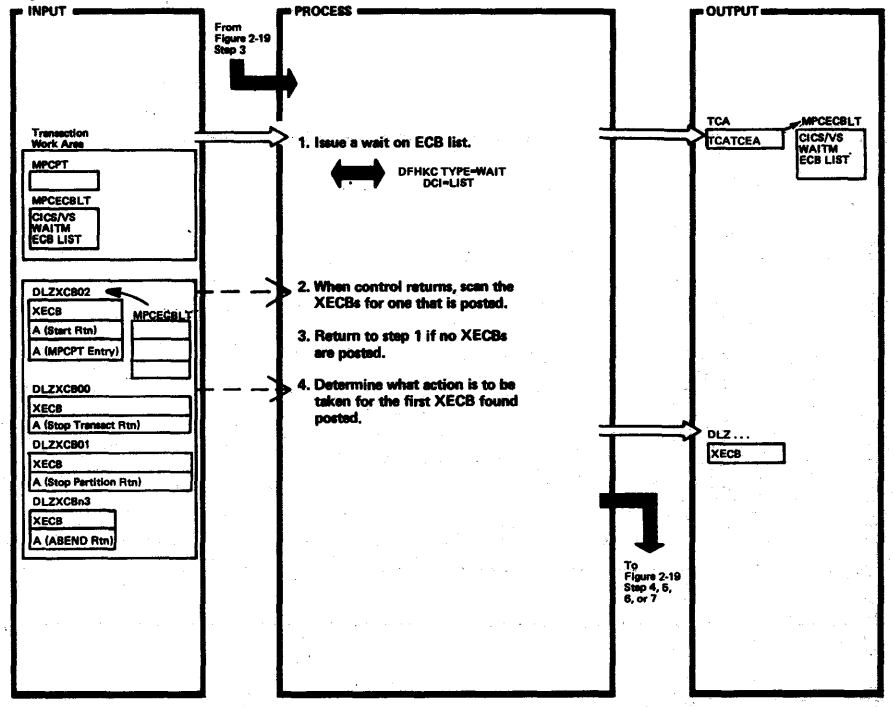
Figure 2-19.2. MPC Define XECBs (Part 2 of 2)



Extended Description	Routine	Label
6. DLZXCB02 is the XECB name to start a batch partition. 8. Turn on SCDXECB at SCDDBMPS and issue message DLZ093I to indicate MPS started. 9. MPCABEXT routine is within this module (see Figure 2-19.9). 10. Turn on the DCAAPURG flag in the DCAFLG1 byte of the DCA for this task. When a CICS/VS task control will not count this task as part of AMXT nor will it take the short wait interval if this is the only waiting task in the CICS/VS system.		XECBDFN2

Extended Description	Routine	Label
11. Adjusts the partition identifier table to the form BG, Fn-1, "F_1 where n is the number of DOS/VSE partitions defined during SYSGEN. This is done so that a printable partition identifier may later be computed from the PIK, which is passed from the batch partition on a start partition request.		

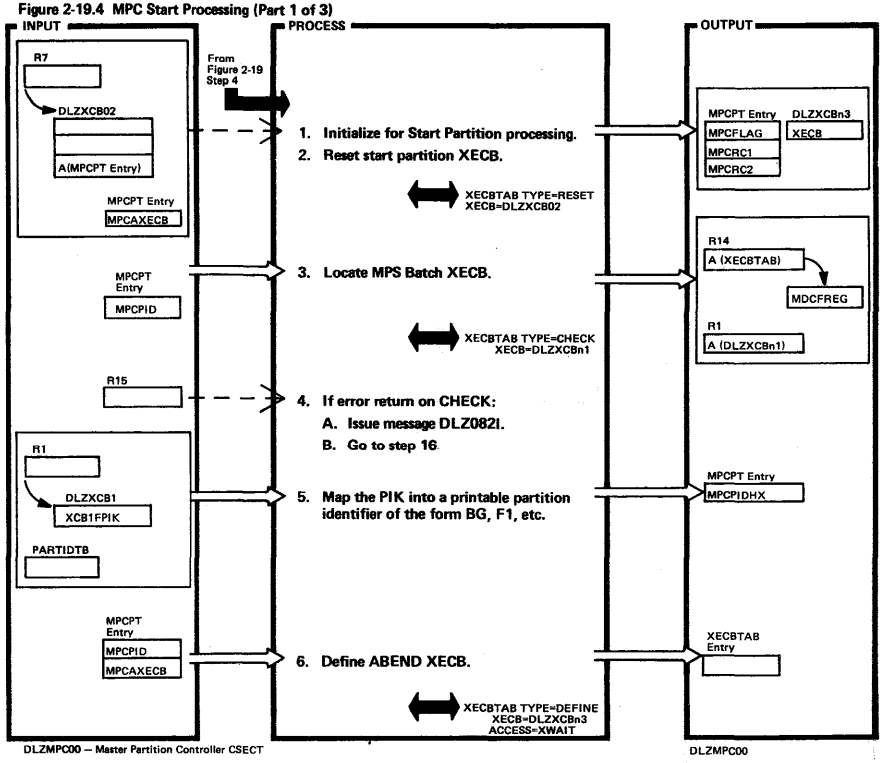
Figure 2-19.3. MPC Wait



Extended Description	Routine	Label
1. Note that the ABEND XECB (DLZXCBn3) pointer is placed in the ECB list only when the BPC attach is unsuccessful.	DLZMPC00	MPCWAIT
2. The XECBs are posted on the following conditions: DLZXCB02 • DLZMPI00 – activate BPC for a specific partition. DLZXCB00 • DLZMSTP0 – terminate MPS. DLZXCB01 • DLZBPC00 – normal batch EOJ; error conditions in BPC or batch partitions. • DLZODP01 – ABEND. DLZXCBn3 • DLZMPI00 – BPC attach failure.		MPCECBCK
4. Before going to the appropriate routine, the post bit in the XECB is turned off.		MPCECBOK

Extended Description	Routine	Label

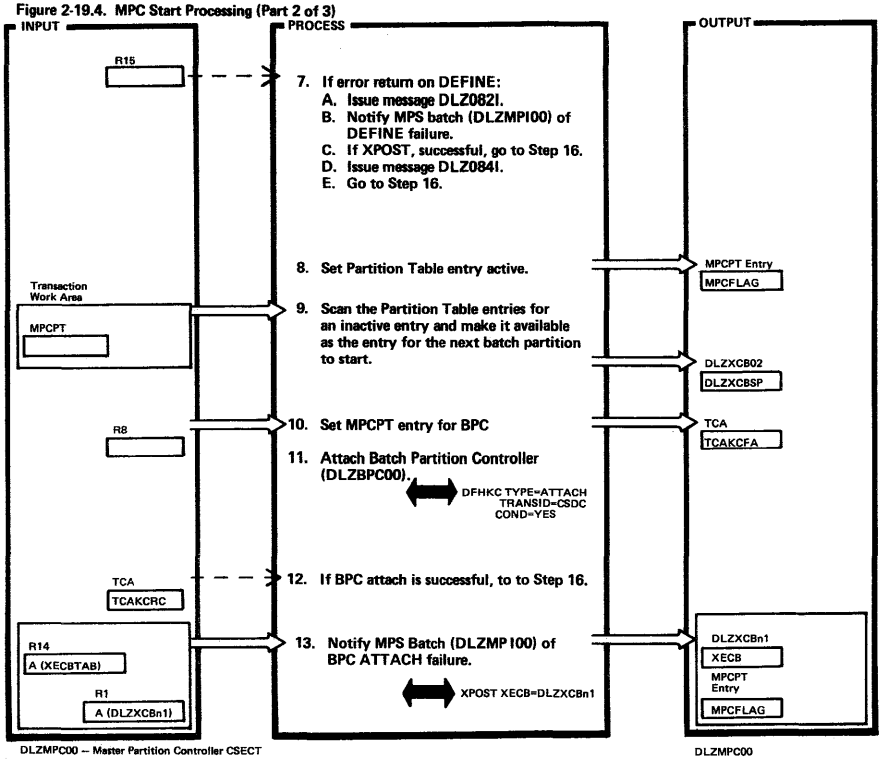
Figure 2-19.4 MPC Start Processing (Part 1 of 3)



Extended Description	Routine	Label
1. This routine is entered from the MPC Wait Routine when a Start Partition XECB (DLZXCB02) is posted (XPOST) by DL/I MPS Batch Module (DLZMPI00). Register 7 contains the address of the XECB posted.	DLZMPC00	MPCSTRP
3. The XECBTAB/CHECK macro is issued to obtain the address (in register 14) of Batch Initialization's XECBTAB entry for the specific partition.		XECBN1CK
5. The PIK is in printable form.		MPCCHKOK MPCGETPI
6. DLZXCBn3 is the XECB name for handling an ABEND situation for a specific partition.		MPCSTRPC XECBABN

Extended Description	Routine	Label

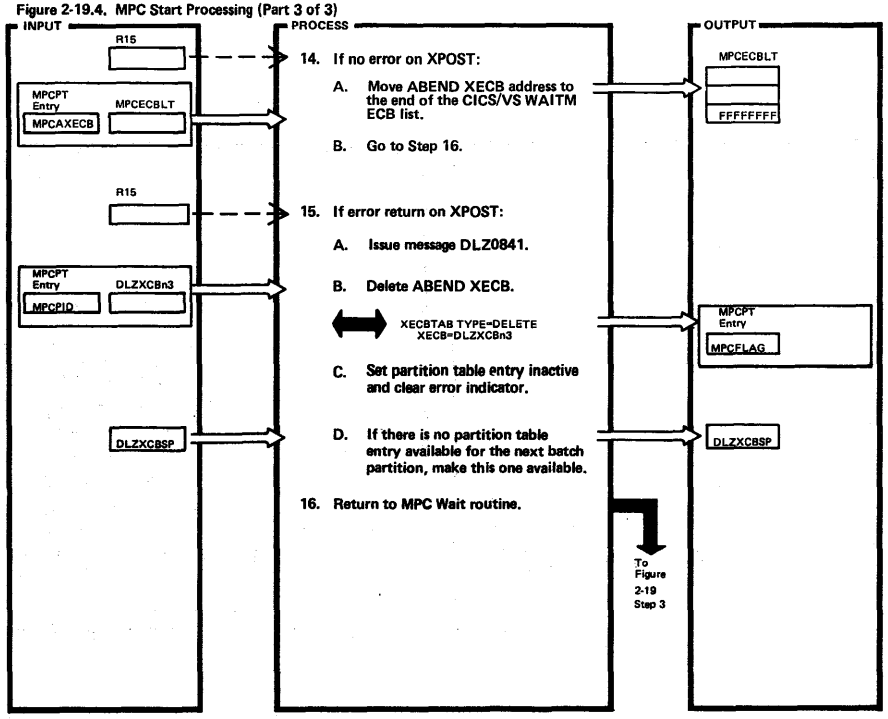
Figure 2-19.4. MPC Start Processing (Part 2 of 3)



Extended Description	Routine	Label
9. If there are no inactive entries set DLZXCBSP to zero.		MPCPTLP MPCCKPT MPCBPCA
12. A 'X31' in TCAKCRC indicates an ATTACH failure.		
13. The BPC ATTACH failed flag. MPCERR, is turned on.		MPCBPCFL

Extended Description	Routine	Label

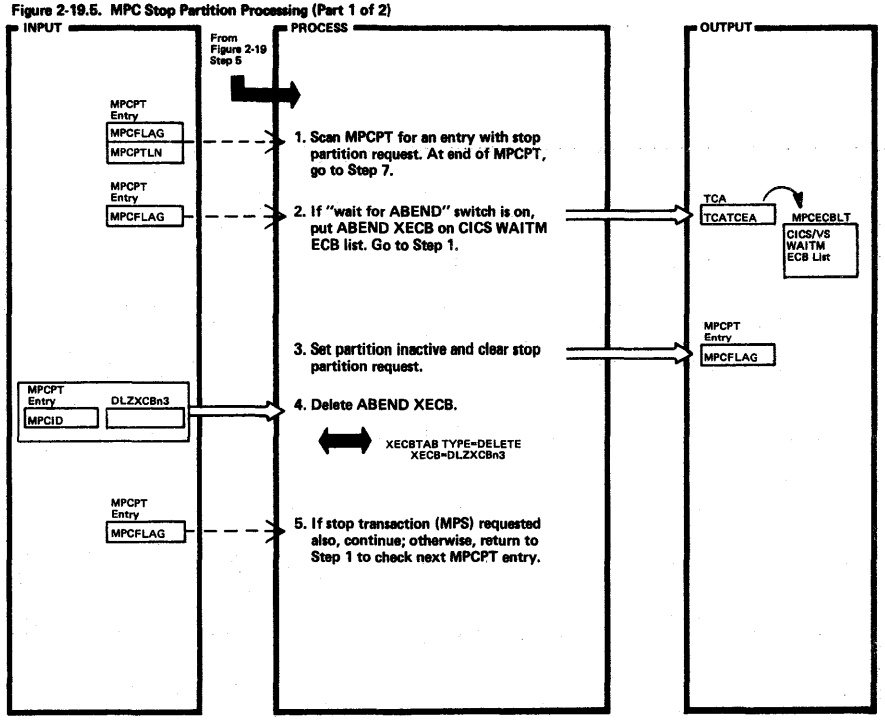
Figure 2-19.4. MPC Start Processing (Part 3 of 3)



Extended Description	Routine	Label
14.A The CICS/VS WAITM ECB list is updated to include a pointer to the ABEND XECB pointer to provide recover.		MPCXPOST
15.B If error return on DELETE, issue message DLZ082I.		XECBDLN3
C. The flags MPCPACT and MPCERR are turned off.		MPCABDOK
D. If DLZXCBSP is zero, there is no inactive entry other than the one just made inactive.		

Extended Description	Routine	Label

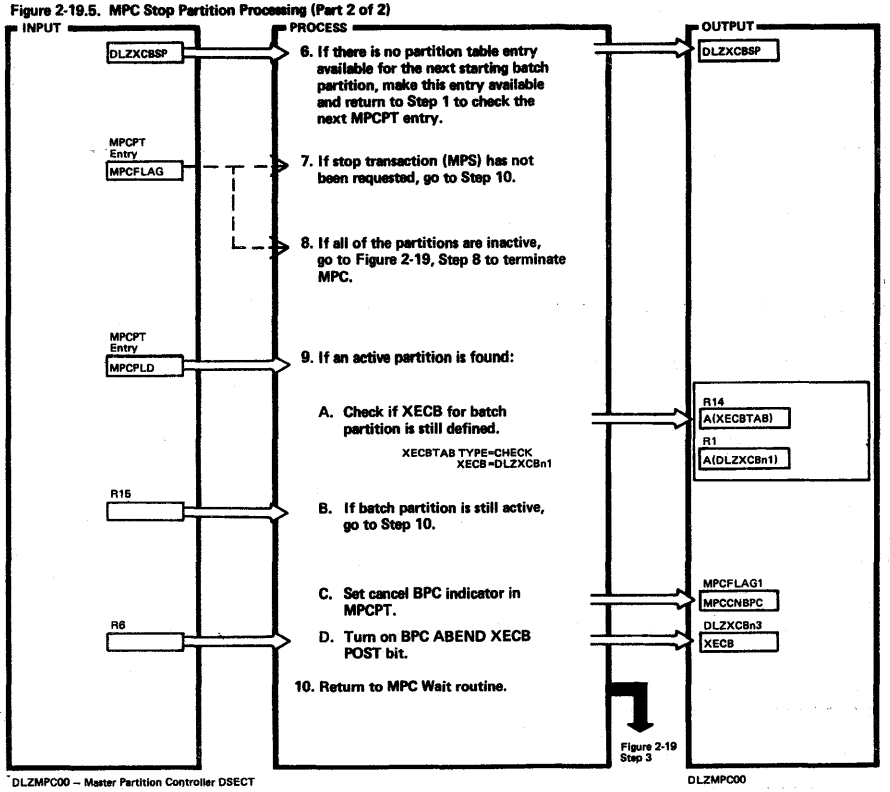
Figure 2-19.5. MPC Stop Partition Processing (Part 1 of 2)



Extended Description	Routine	Label
1. This routine is entered from the MPC wait routine when a stop partition XECB (DLZXCB01) is posted by DL/I MPS Batch Partition Controller (DLZBPC00) or Task Termination (DLZODP01). A scan is done on every entry in the partition table to avoid losing a stop partition request on a double post.		MPCSTOP
2. Flag MPCABWT in MPCFLAG indicates that the ABEND XECB (DLZXCBn3) should be waited on.		
4. If error return on DELETE, issue message DLZ082I.		XECBDELA

Extended Description	Routine	Label

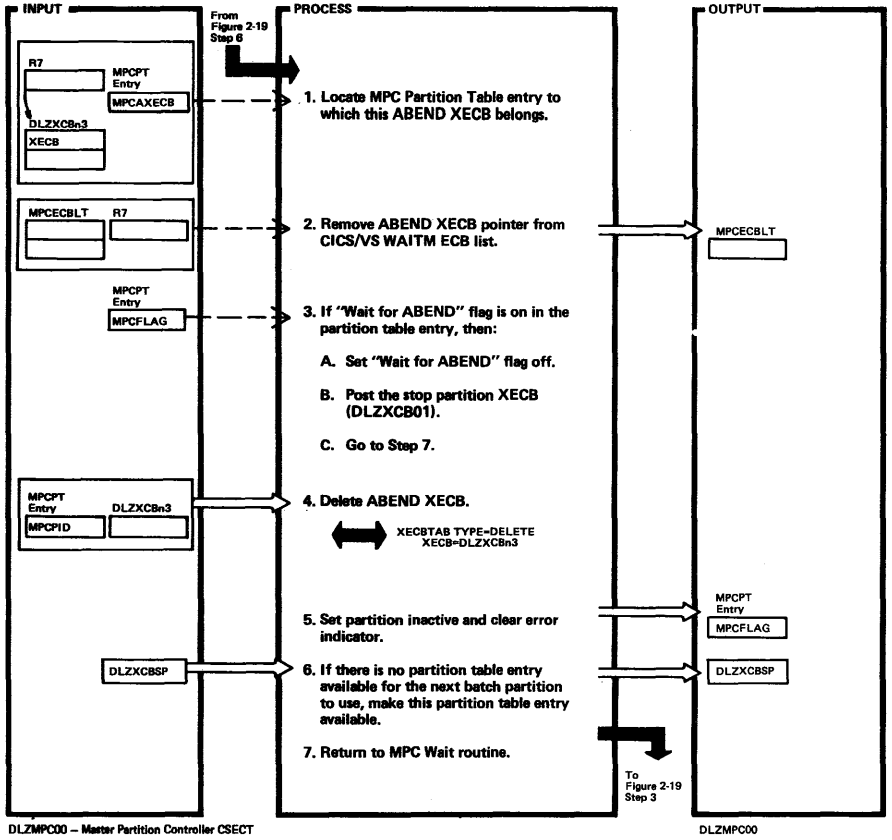
Figure 2-19.5. MPC Stop Partition Processing (Part 2 of 2)



Extended Description	Routine	Label
8. Flag MPCPACT in MPCFLAG indicates whether the partition is active or inactive.		MPCCKACT
9. A partition is active if MPCPACT is on.		MPCCKN
A. The XECBTAB/CHECK macro is issued to determine if the batch partition is still defined.		XECBTCH
C. Bit MPCCNBPC in field MPCFLAG1 in the MPCPT is set on.		
D. R6 contains pointer to the BPC ABEND XECB (DLZXCBn3). The XPOST macro is not needed to turn on the POST bit because the ABEND XECB is defined by MPC.		

Extended Description	Routine	Label

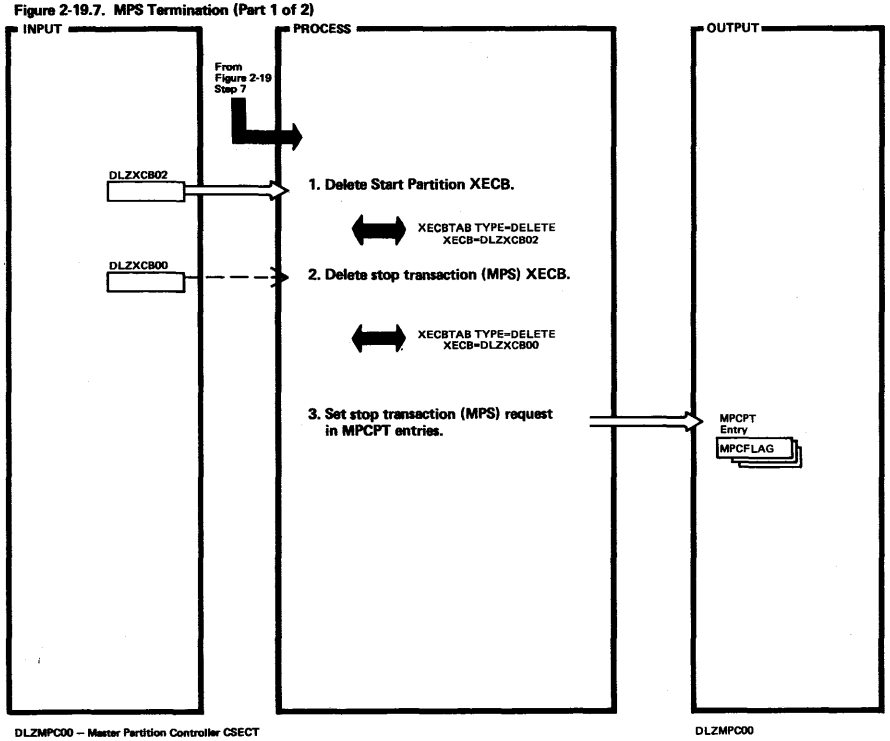
Figure 2-19.6. MPC ABEND Processing



Extended Description	Routine	Label
1. This routine is entered from the MPC Wait routine when an ABEND XECB (DLZXCBn3) is posted (XPOST) by DL/I MPS Batch Initialization Module (DLZMPI00) on a BPC ATTACH Failure. Register 7 contains the address of the XECB posted.		MPCABNP
3. Flag MPCABWT is the "wait for ABEND" flag.		
4. If error return on DELETE, issue message DLZ0821.		XECBDEL3
5. Flags MPCPACT and MPCERR are turned off.		MPCABDEC

Extended Description	Routine	Label

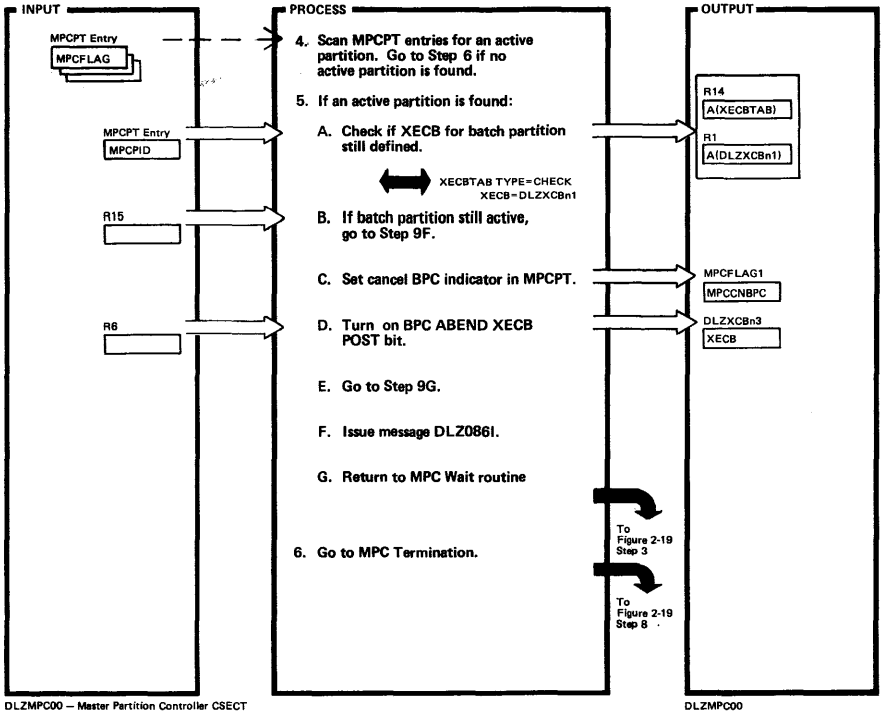
Figure 2-19.7. MPS Termination (Part 1 of 2)



Extended Description	Routine	Label
1. This routine is entered from the MPC Wait routine when the Stop Transaction (MPS) XECB (DLZXCB00) is posted by DL/I Stop Transaction Task (DLZMSTP0).		MPCSTRN
If error return on DELETE, issue message DLZ082I.		
2. If error return on DELETE, issue message DLZ082I.		MPCSTRN2
3. Flag MPCTSTP is turned on.		MPCSXCK

Extended Description	Routine	Label

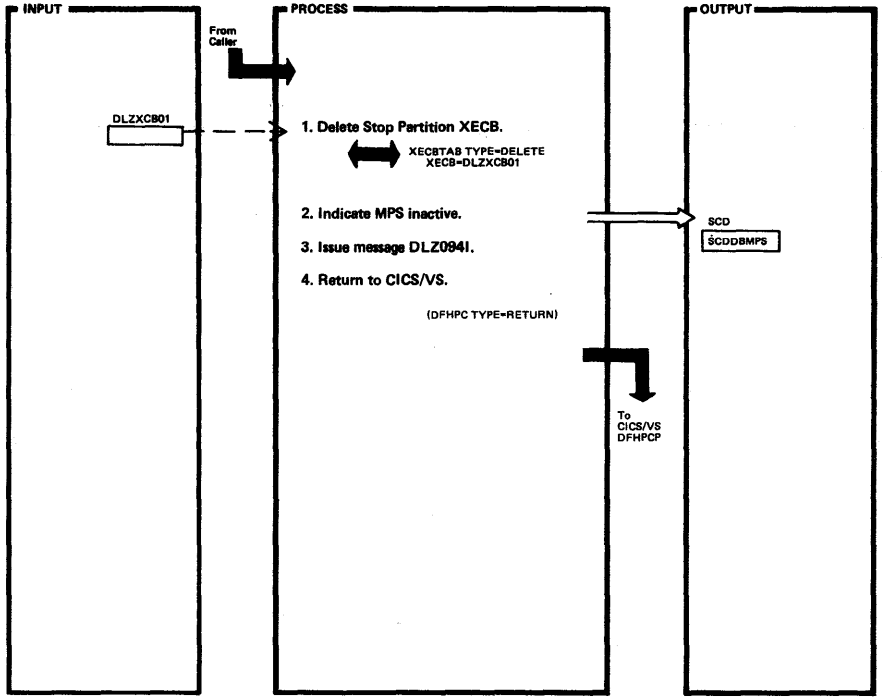
Figure 2-19.7. MPS Termination (Part 2 of 2)



Extended Description	Routine	Label
4. A partition is active if MPCPACT is on.		MPCDELCP MPCSXEND
5. A. The XECBTAB/CHECK macro is is issued to determine if the batch partition is still defined.		XECBATCH
C. Bit MPCCNBPC in field MPCFLAG1 in the MPCPT is set on.		
D. R6 contains pointer to the BPC ABEND XECB (DLZXCBn3). The XPOST macro is not needed to turn on the POST bit because the ABEND XECB is defined by MPC.		

Extended Description	Routine	Label

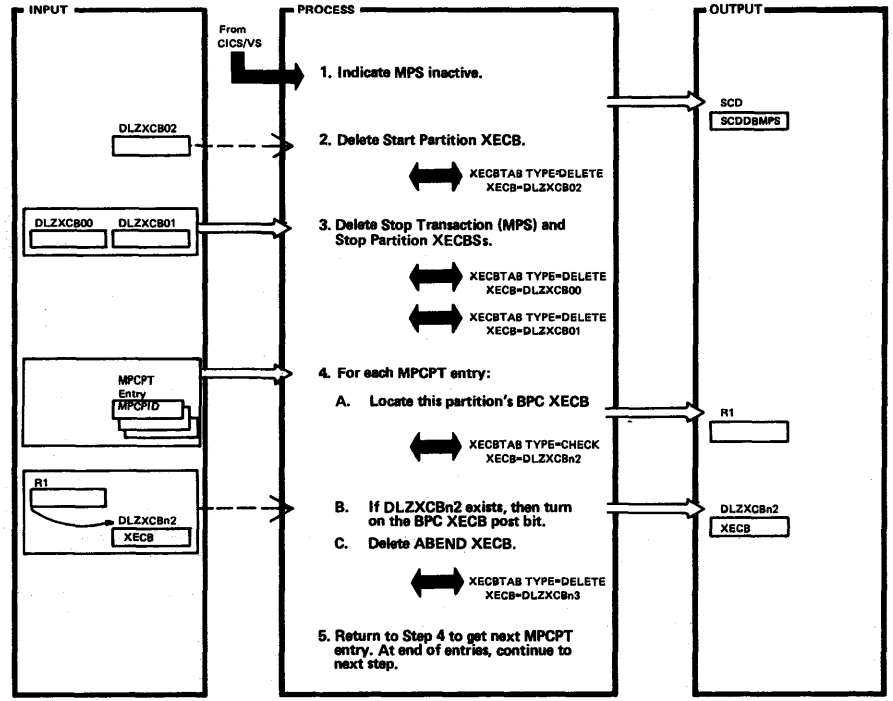
Figure 2-19.8. MPC Stop Transaction Processing



Extended Description	Routine	Label
1. This routine is entered when MPS is to be terminated normally or abnormally. If error return on DELETE, issue message DLZ082I. 2. Flag SCDXECB at SCDDBMPS is turned off and message DLZ094I is issued to indicate MPS stopped. 4.		MPCEXIT CICSRTN

Extended Description	Routine	Label

Figure 2-19.9 MPC ABEND Exit Routine (Part 1 of 2)



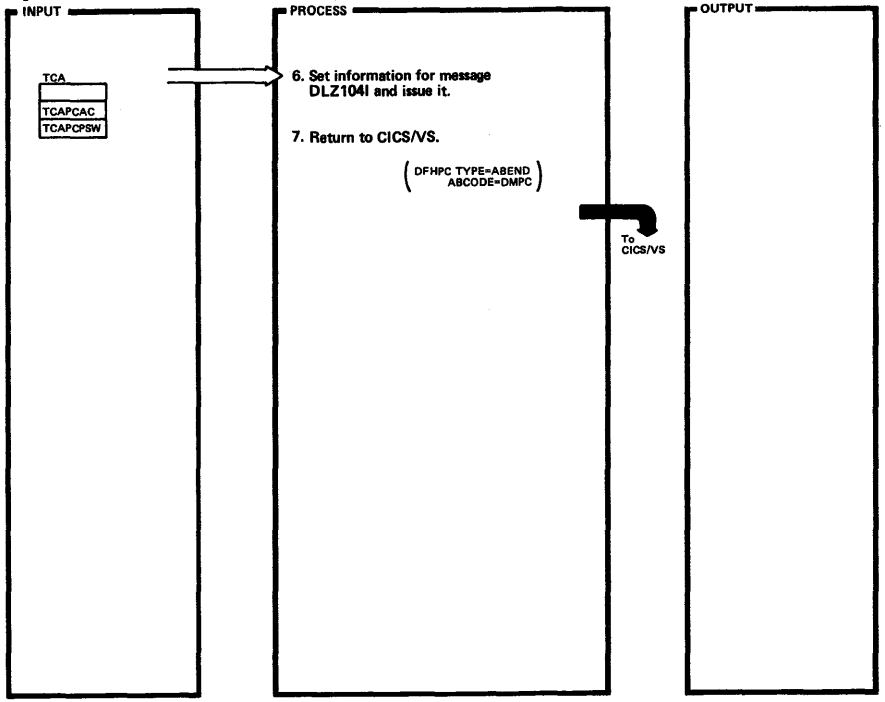
DLZMPC00 — Master Partition Controller CSECT

DLZMPC00

Extended Description	Routine	Label
1. This routine is entered from CICS/VS if an ABEND occurs in MPC. Linkage was established through DFHPC TYPE=SETXIT in Figure 2-19.2. Flag SCDXECB at SCDDBMPS is turned off to show MIPS as inactive. 4B. Note that the XPOST macro is not needed to turn on the POST bit because BPC XECB (DLZXCBn2) is defined in the same partition as this module. DLZXCBn2 is defined by DLZBPC00.		MPCABEXT

Extended Description	Routine	Label

Figure 2-19.9. MPC ABEND Exit Routine (Part 2 of 2)



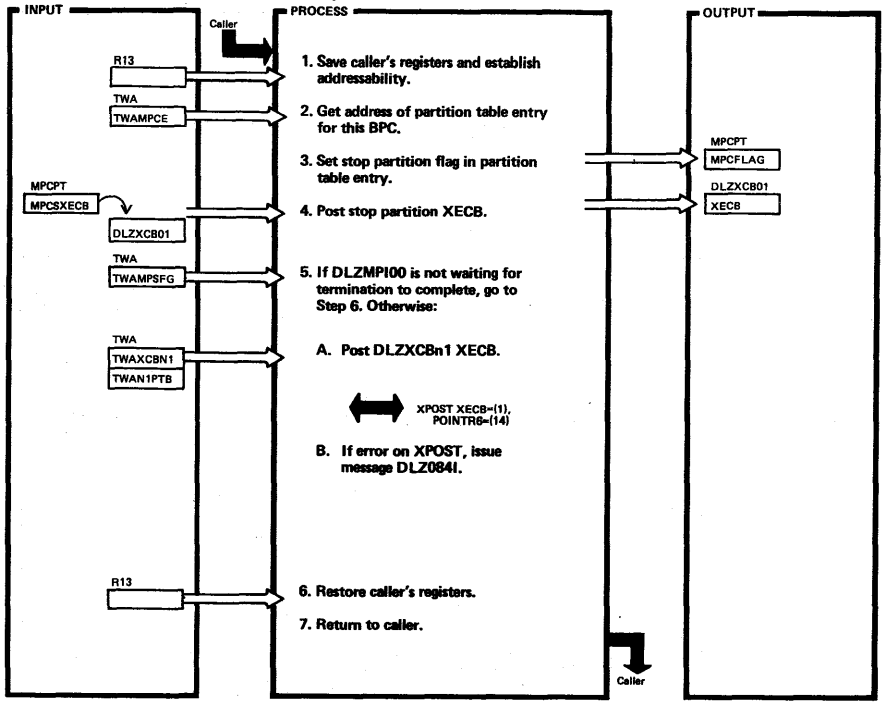
DLZMPC00 — Master Partition Controller CSECT

DLZMPC00

Extended Description	Routine	Label
6. 7. DMPC ABEND code defines MPC failure for CICS/VS dump ID.		MSG104

Extended Description	Routine	Label

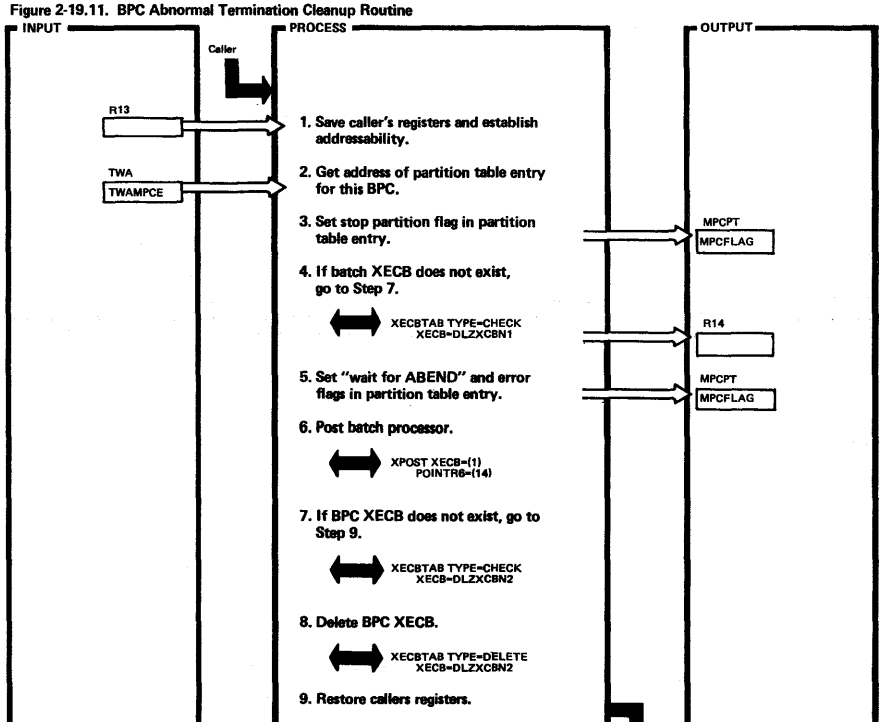
Figure 2-19.10. BPC Normal Termination Cleanup Routine



Extended Description	Routine	Label
1. This routine is entered from DLZODP when BPC normal termination occurs. 3. Flag MPCPSTP indicates stop partition. 5. Flag TWAEOJSW indicates that DLZMPI00 is waiting for termination processing to complete.		MPCBPCNT

Extended Description	Routine	Label

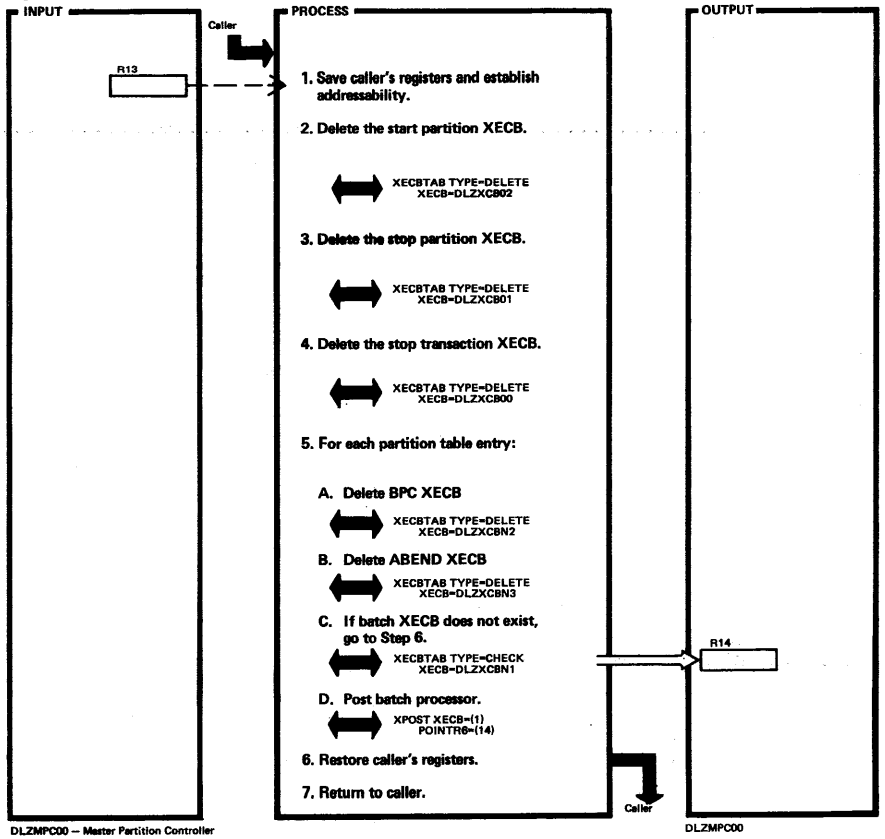
Figure 2-19.11. BPC Abnormal Termination Cleanup Routine



Extended Description	Routine	Label
1. This routine is entered from DLZODP when BPC abnormal termination occurs. 3. Flag MPCPSTP indicates stop partition. 5. Flag MPCABWT indicates wait on ABEND XECB and flag MPCERR is the error flag. 9.		MPCBPCAT MPCBPCA2

Extended Description	Routine	Label

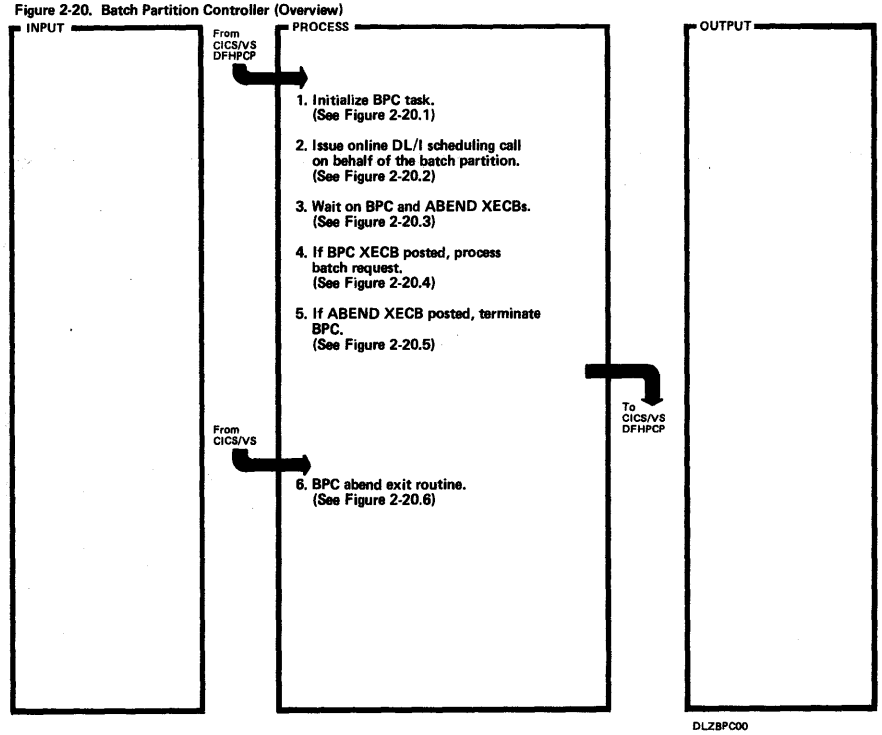
Figure 2-19.12. MPS Abnormal System Termination Cleanup Routine



Extended Description	Routine	Label
1. This routine is entered from DLZODP when the system abnormally terminates.		MPCSYSTM

Extended Description	Routine	Label

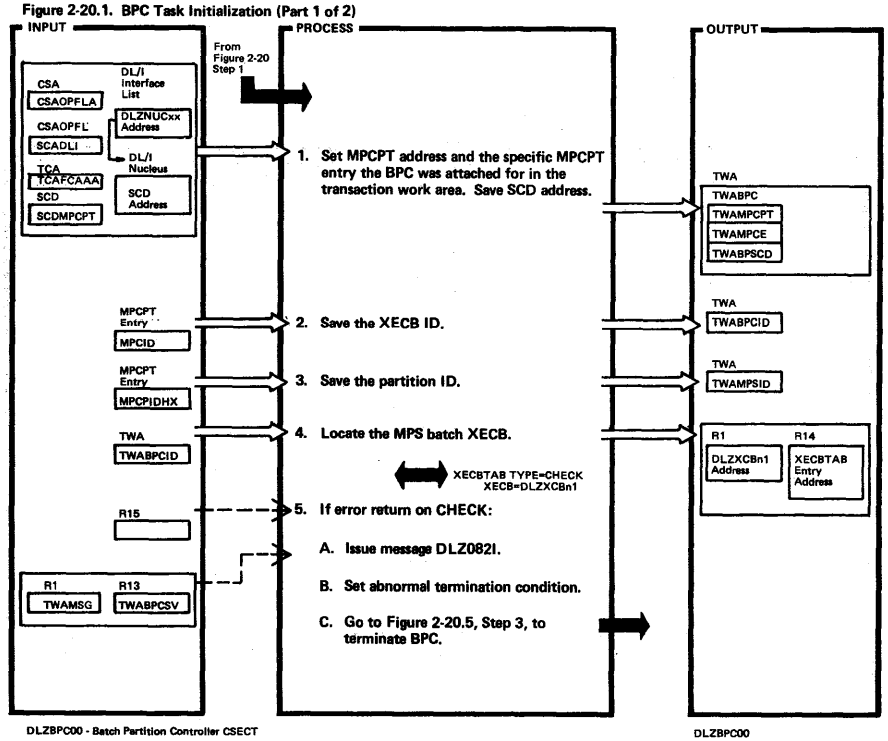
Figure 2-20. Batch Partition Controller (Overview)



Extended Description	Routine	Label
1.	DLZBPC00	DLZBPC00 BPCSTART
2.		BPCSCHCK
3.		BPCWAIT
4.		BPCCALL
5.		BPCEXIT
6.		BPCABND

Extended Description	Routine	Label

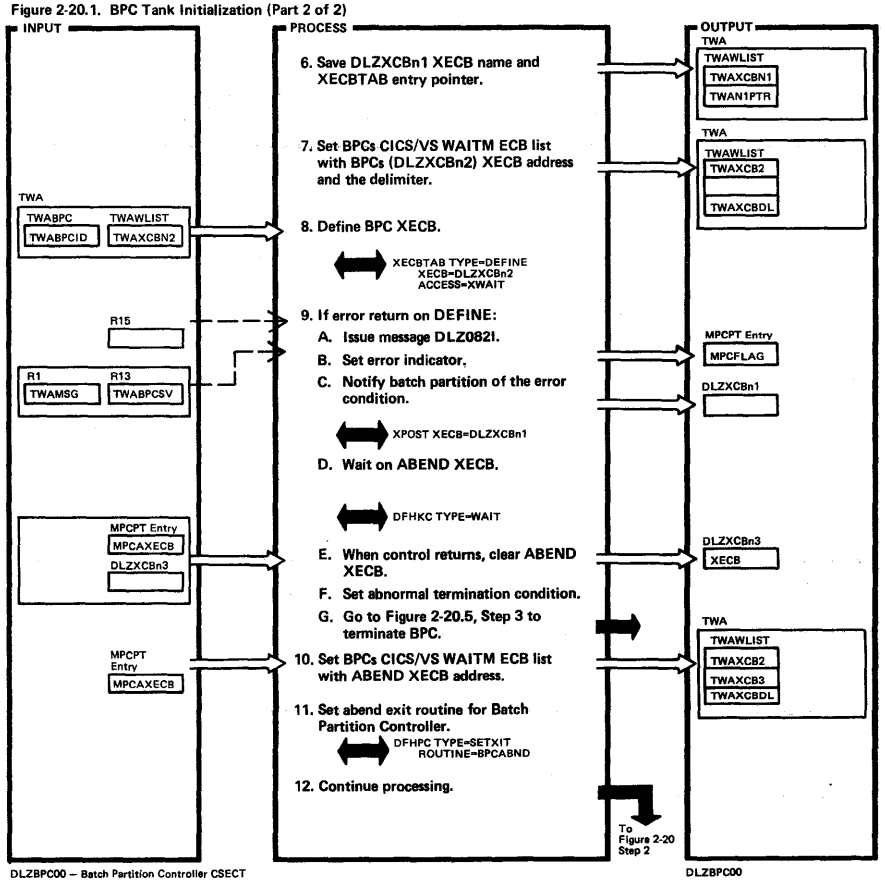
Figure 2-20.1. BPC Task Initialization (Part 1 of 2)



Extended Description	Routine	Label
1. Module identifier (DLZBPC00 vmp) is defined here. The Batch Partition Controller (BPC) is attached by DLZMPC00 when a start request has been made by a partition. On entry, R12 contains address of TCA and R13 contains address of CSA. DLZXCBn1 = XECB name for MPS batch partition, where n is the partition ID, which is assumed by DLZMPC00. DLZXCBn2 = XECB name for a BPC for a specific partition. DLZXCBn3 = XECB name for handling an abend condition for a specific partition.	DLZBPC00	DLZBPC00 BPCSTART
4. XECBTAB TYPE=CHECK macro is used to obtain the address of MPS batch XECB.		XECBCHK

Extended Description	Routine	Label

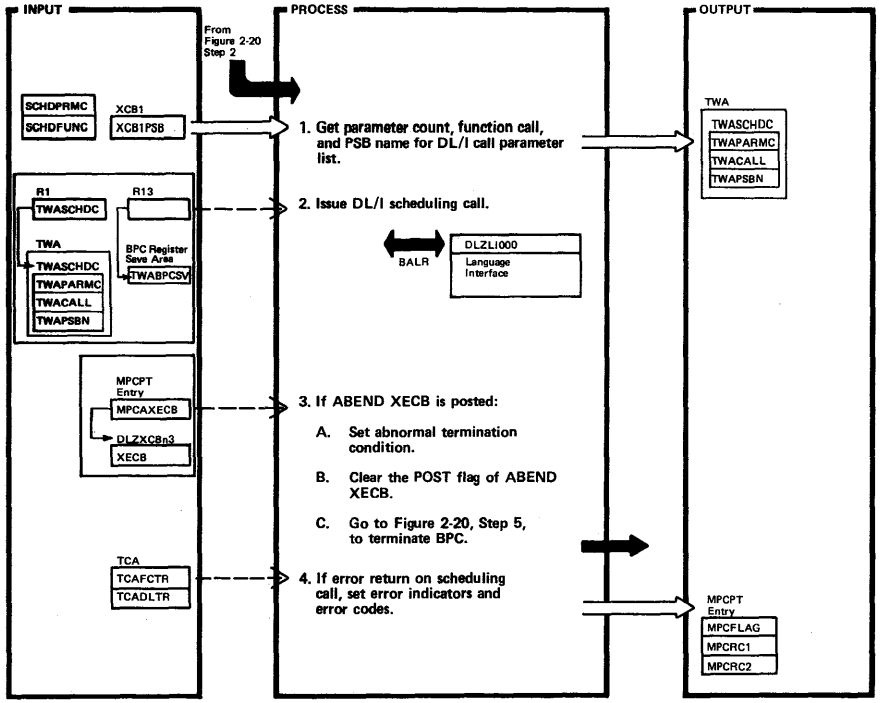
Figure 2-20.1. BPC Tank Initialization (Part 2 of 2)



Extended Description	Routine	Label
8. BPC XECB (DLZXCBn2) is defined for cross partition communication with MPS Batch Initialization (DLZMINIT), MPS Batch Program Request Handler (DLZMPRH), and (DLZMTERM).		XECBDFN
9. B. Flag MPCERR indicates an error condition. E. The POST bit X'80' in the XECB is turned off.		BPCDFERR
11. The BPCABND routine (see Figure 2-20.6) is within this module.		

Extended Description	Routine	Label

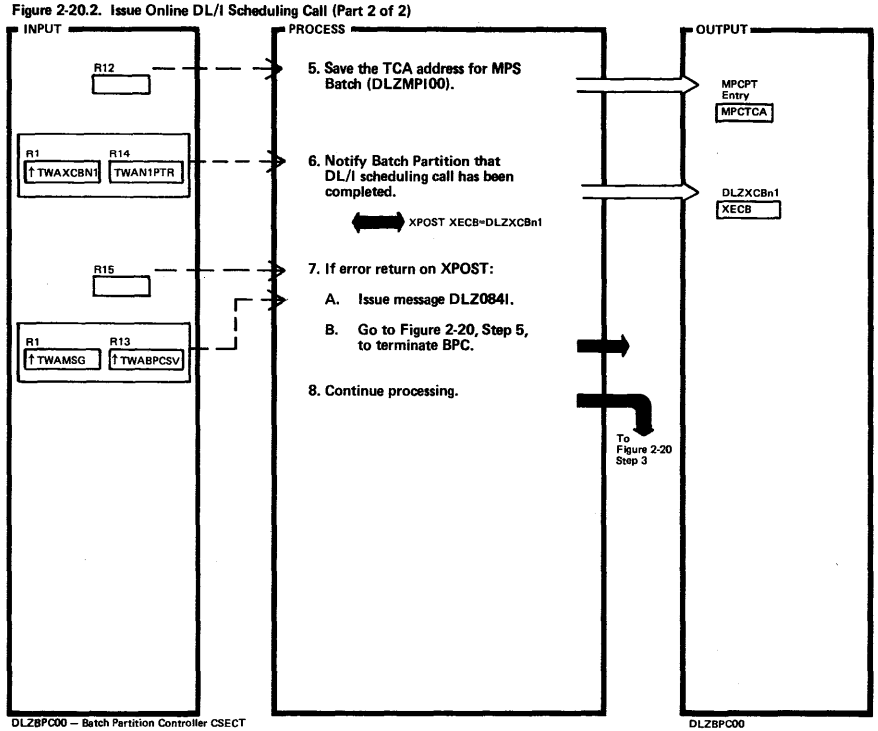
Figure 2-20.2. Issue Online DL/I Scheduling Call (Part 1 of 2)



Extended Description	Routine	Label
1. Macro DLZXCB1 defines a DSECT that represents the format of the fields after the MPS Batch XECB (DLZXCBn1) used here as a parameter list by BPC. Addressability to DLZXCBn1 was obtained by the XECBTAB TYPE=CHECK macro in Figure 2-20.1, Step 5.		
4. Flag MPCERR at MPCFLAG is turned on.		BPCSCHCK

Extended Description	Routine	Label

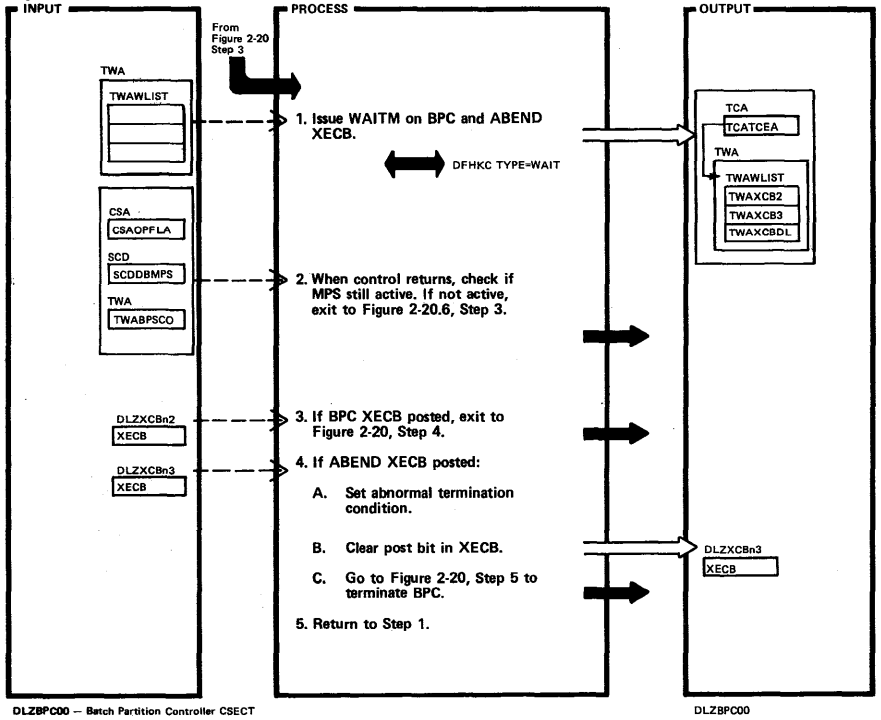
Figure 2-20.2. Issue Online DL/I Scheduling Call (Part 2 of 2)



Extended Description	Routine	Label
6.		BPCSCHDK

Extended Description	Routine	Label

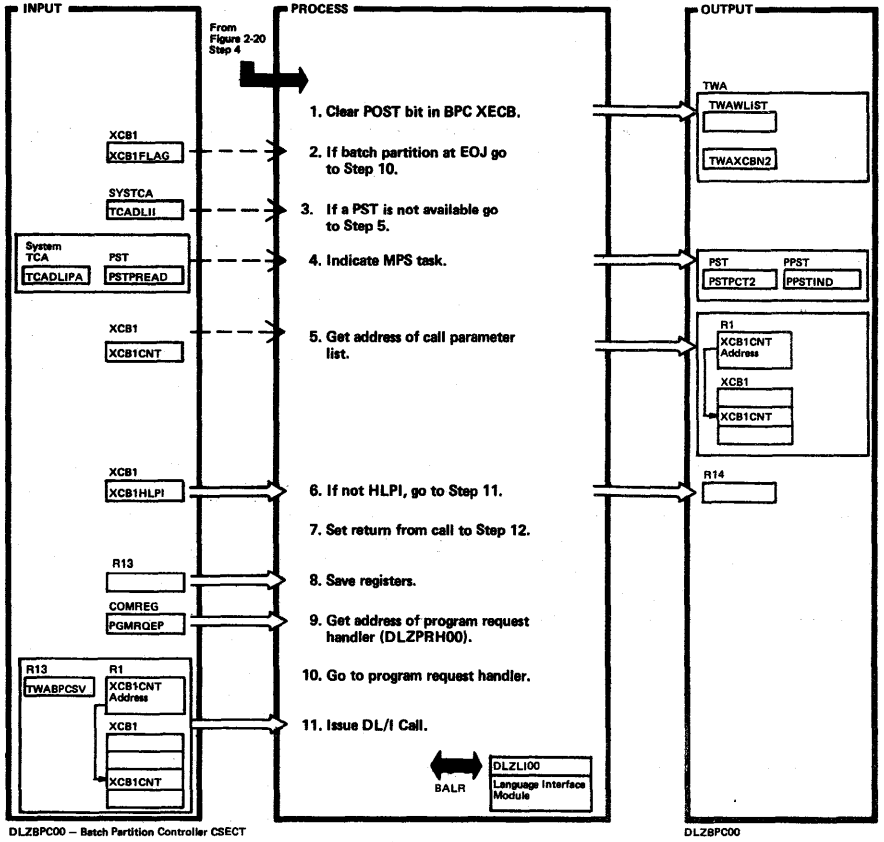
Figure 2-20.3. Wait on BPC and ABEND XECBs



Extended Description	Routine	Label
1. The XECBs are posted for the following conditions: DLZXCBn2 • Process call on behalf of batch partition. • EOJ has been encountered in batch partition. DLZXCBn3 • An ABEND condition has been encountered in the batch partition. 2. Flag SCDXECB is tested in SCDDBMPS.		BPCWAIT

Extended Description	Routine	Label

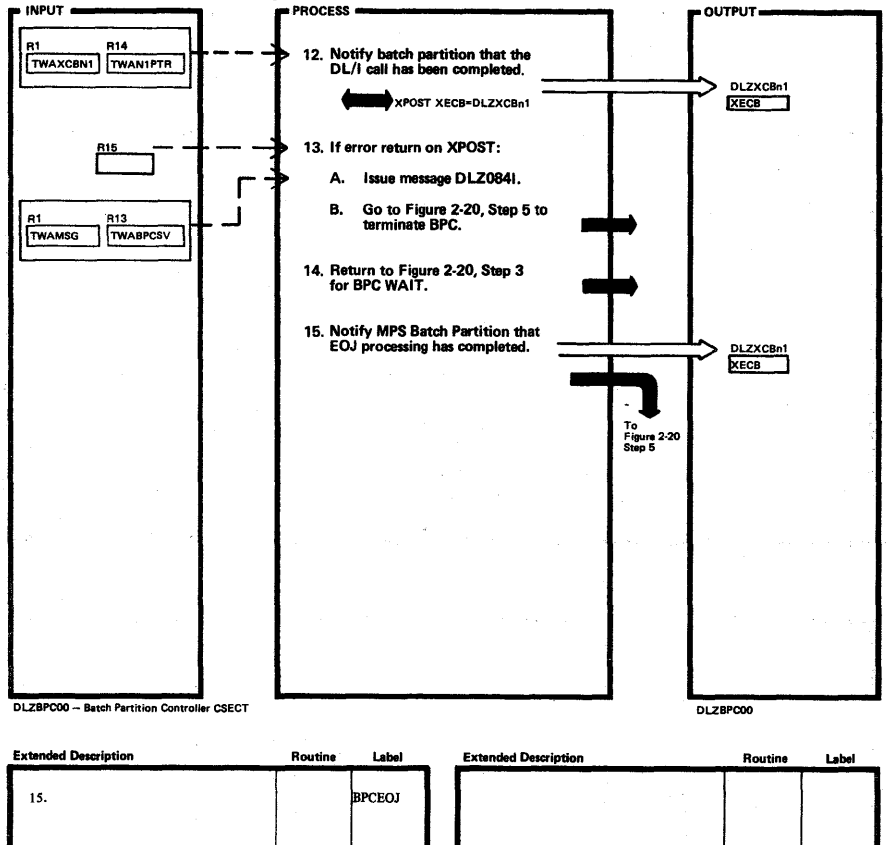
Figure 2-20.4. Batch Request Processing (Part 1 of 2)



Extended Description	Routine	Label
1. This routine is entered from BPC WAIT routine when the BPC XECB (DLZXCBn2) is posted by the batch program request handler. R10 points to the call parameter list in the MPS Batch partition. 2. Macro DLZXCB1 defines a DSECT representing the format of the MPS batch XECB and following fields used for communication between the batch and online partitions. The End-of-Job flag is set by DLZMPI00 termination routine in the field following the XECB.		

Extended Description	Routine	Label
3. Flag TCADLIPS indicates a PST is available. 4. Flag PPSTMPS in PPSTIND indicates this is an MPS task. 10. 11. Entry point in the language interface module will be ASMTDLI or PLITDLI, depending on whether the user program is assembler or PL/I.		BPCNOPST BPCDLICD

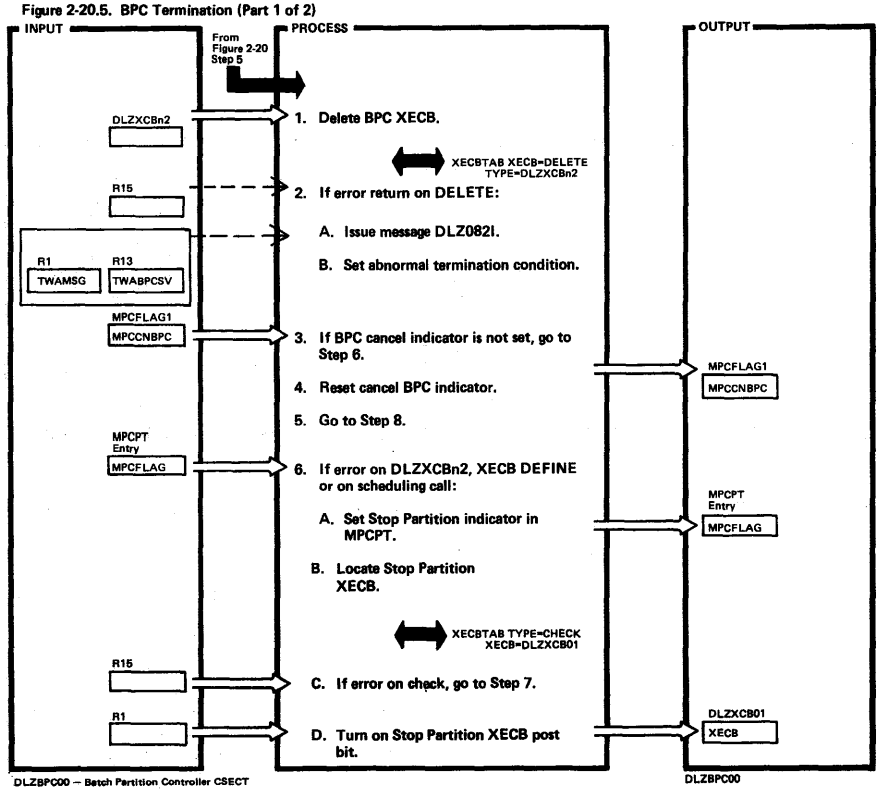
Figure 2-20.4. Batch Request Processing (Part 2 of 2)



Extended Description	Routine	Label
15.		BPCEOJ

Extended Description	Routine	Label

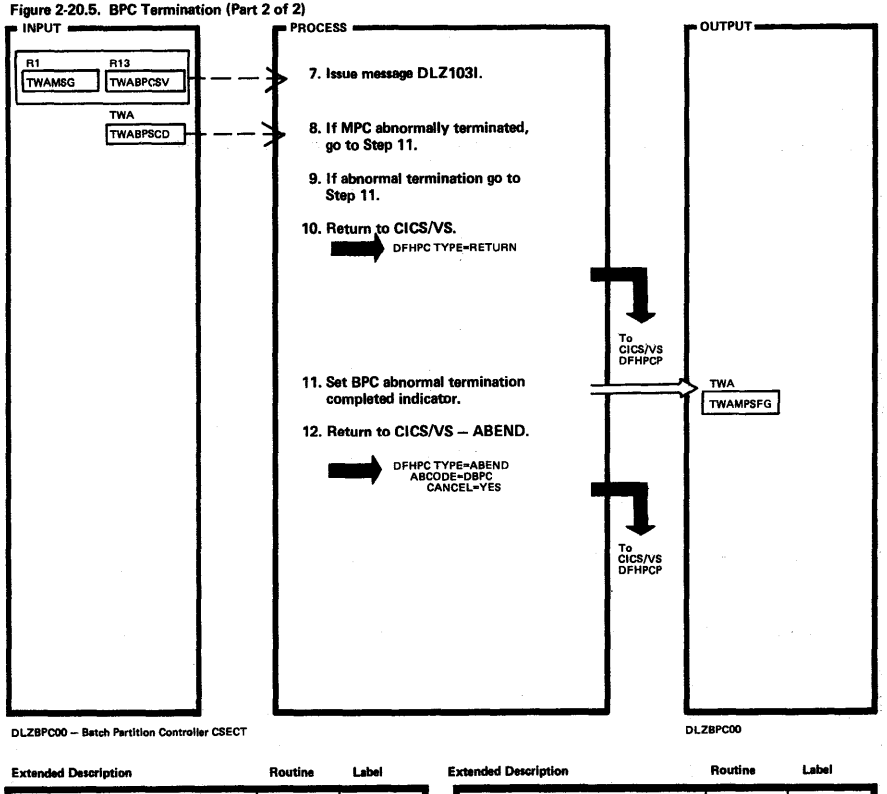
Figure 2-20.5. BPC Termination (Part 1 of 2)



Extended Description	Routine	Label
1. This routine is entered on normal or abnormal termination of BPC.		BPCEXIT
3. Bit MPCCNBPC is set by MPC when it determines that the batch partition no longer is active.		
6. Flag MPCERR indicates an error condition.		BPCEXIT2
A. Flag MPCPSTP indicates stop partition.		
B. Address of XECB is returned in R1.		

Extended Description	Routine	Label

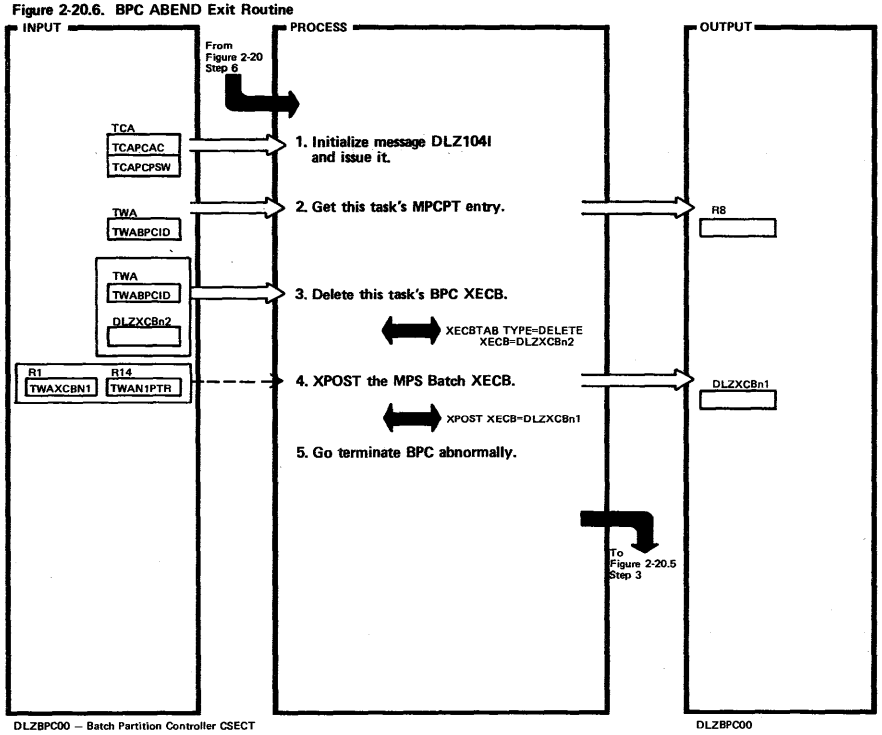
Figure 2-20.5. BPC Termination (Part 2 of 2)



Extended Description	Routine	Label
8. Flag SCDXECB at SCDDBMPS indicates if MPS is active or not.		
11. Flag TWABPCOK indicates BPC ABEND processing was successful.		BPCABEND
12. DBPC ABEND code defines BPC failure for CICS/VS dump ID.		

Extended Description	Routine	Label

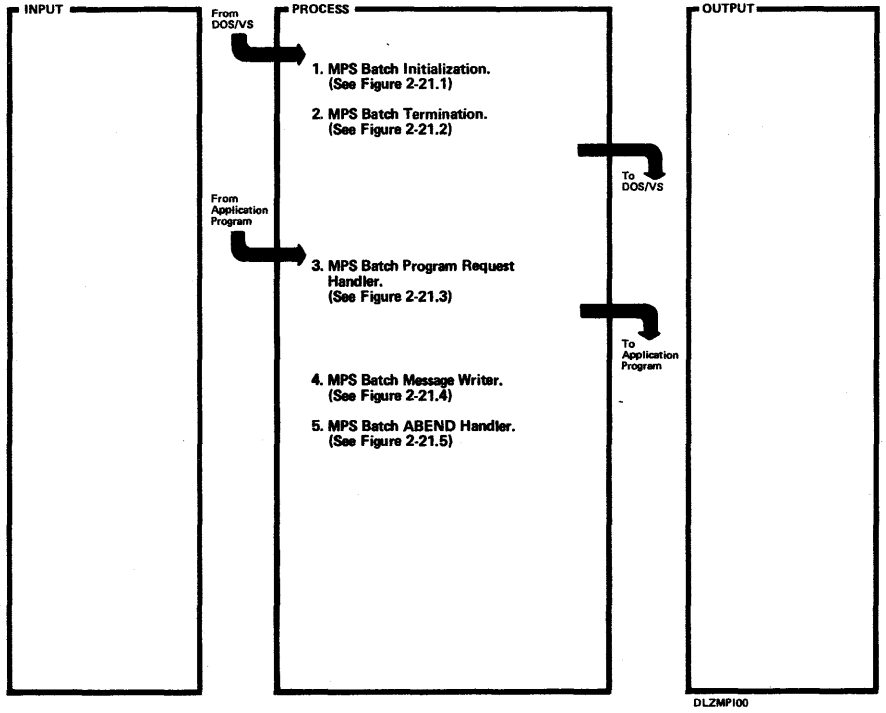
Figure 2-20.6. BPC ABEND Exit Routine



Extended Description	Routine	Label
1. This routine is entered from CICS/VS if an abend occurs in the Batch Partition Controller Module (DLZBPC00).		BPCABND
3.		MPCABEND

Extended Description	Routine	Label

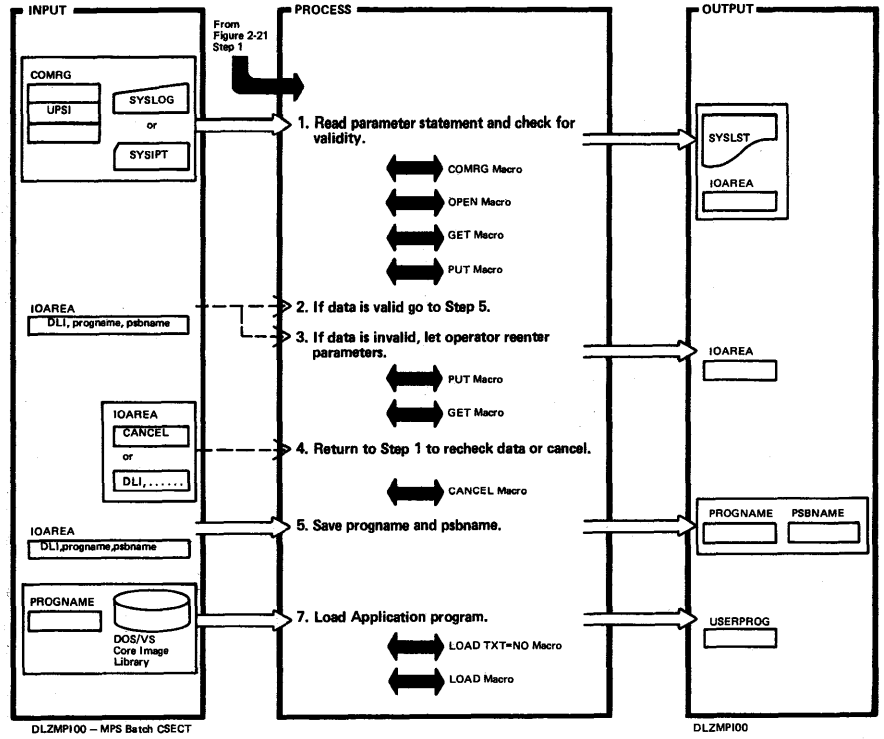
Figure 2-21. MPS Batch (Overview)



Extended Description	Routine	Label
1.	DLZMPI00	DLZMINIT
2.		DLZMTERM
3.		DLZMPRH
4.		DLZMMSG
5.		DLZMABND

Extended Description	Routine	Label

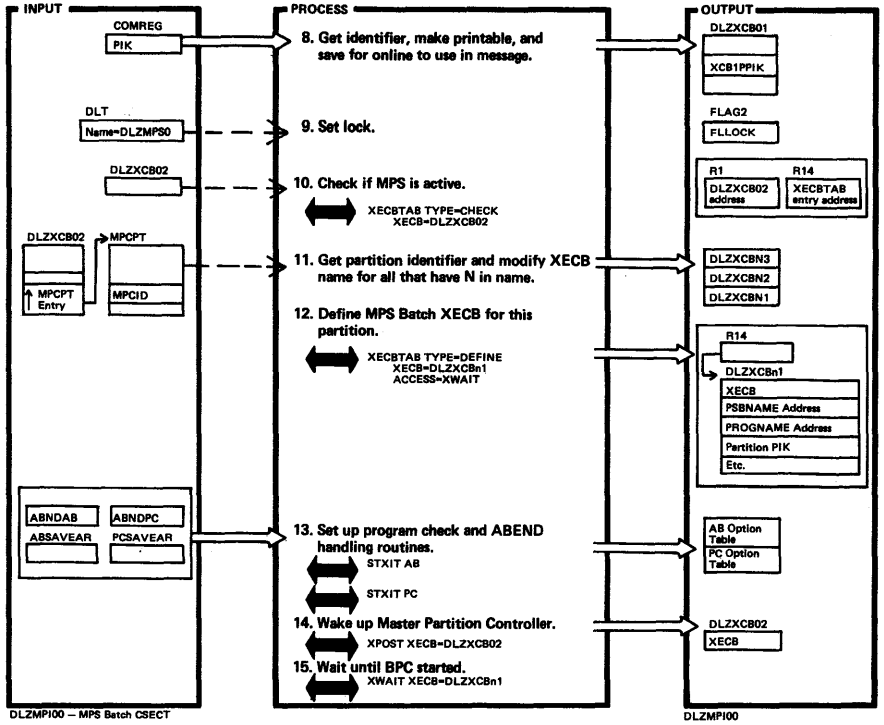
Figure 2-21.1. MPS Batch Initialization (Part 1 of 3)



Extended Description	Routine	Label
1. If UPSI byte bit 0 is on, input is from SYSLOG. Send message DLZ010A to have operator enter information. If UPSI byte bit 0 is off, input is from SYSIPT. Write message DLZ014A if end of file on SYSIPT.	DLZMPI00	DLZMINIT CHECKIN
3. Write message DLZ087A if data invalid.		PARMERR
4. Read operator reply to decide if more parameters provided or if job cancelled.		GETCONS3
5.		CHECKOK
7. If program not found, write message DLZ012I and cancel job.		LOADAP

Extended Description	Routine	Label

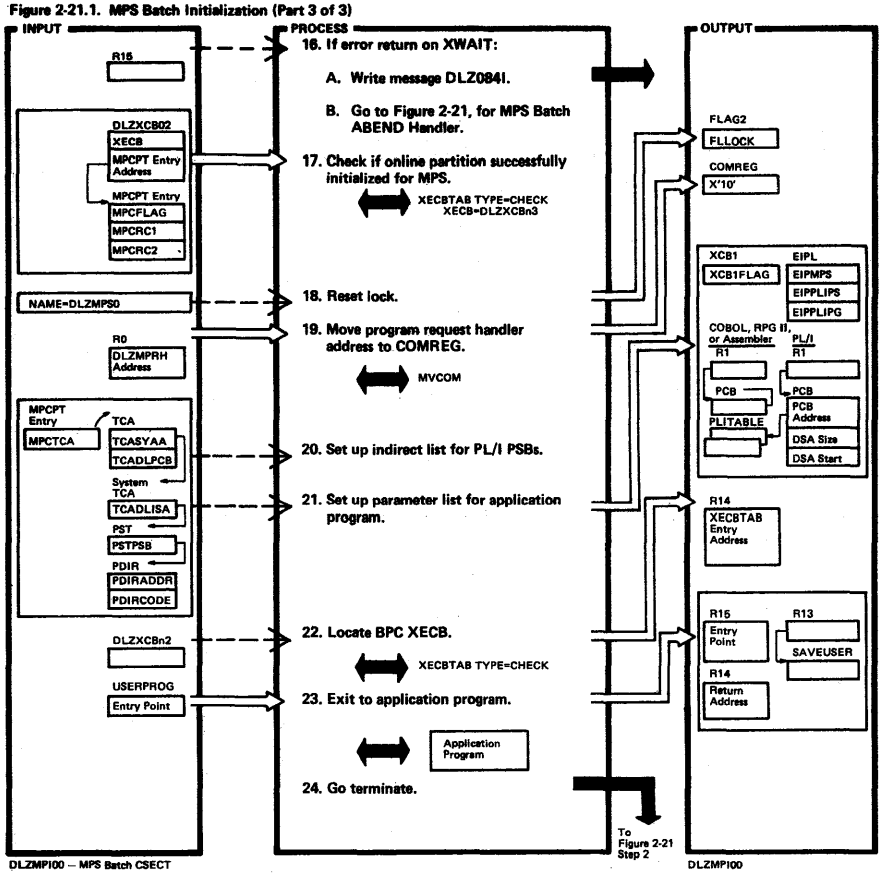
Figure 2-21.1. MPS Batch Initialization (Part 2 of 3)



Extended Description	Routine	Label
9. This prevents another MPS batch task from starting or stopping while this one is initializing. The value for N in Step 10 is updated during this process and must complete before another batch task can use it. FLLOCK in FLAG2 indicates lock is set if 1, is not set if 0.		
10. If the MPC XECB (DLZXCB02) is not found, write message DLZ089I. If there is no ID available in the partition table, write message DLZ088I. Both messages are followed by cancel.		
12. Write message DLZ082I if DEFINE is not successful. Following MPS batch XECB (DLZXCB (DLZXCBn1) are parameter fields used in communicating with the online partition. Macro DLZXCB1 contains the DSECT which describes this XECB and following fields.		XECBDEF1

Extended Description	Routine	Label
14. Notify the online partition (DLZMPC00 specifically) that an MPS batch job is ready to execute and write batch started message DLZ081I if XPOST successful. If XPOST unsuccessful, delete MPS batch XECB (DLZXCBn1), write message DLZ084I, and cancel.		XPOST1
15. Wait is made for DLZBPC00 to post the MPS batch XECB (DLZXCBn1) to notify us it is initialized and has completed a DL/I scheduling call for us.		XWAIT1

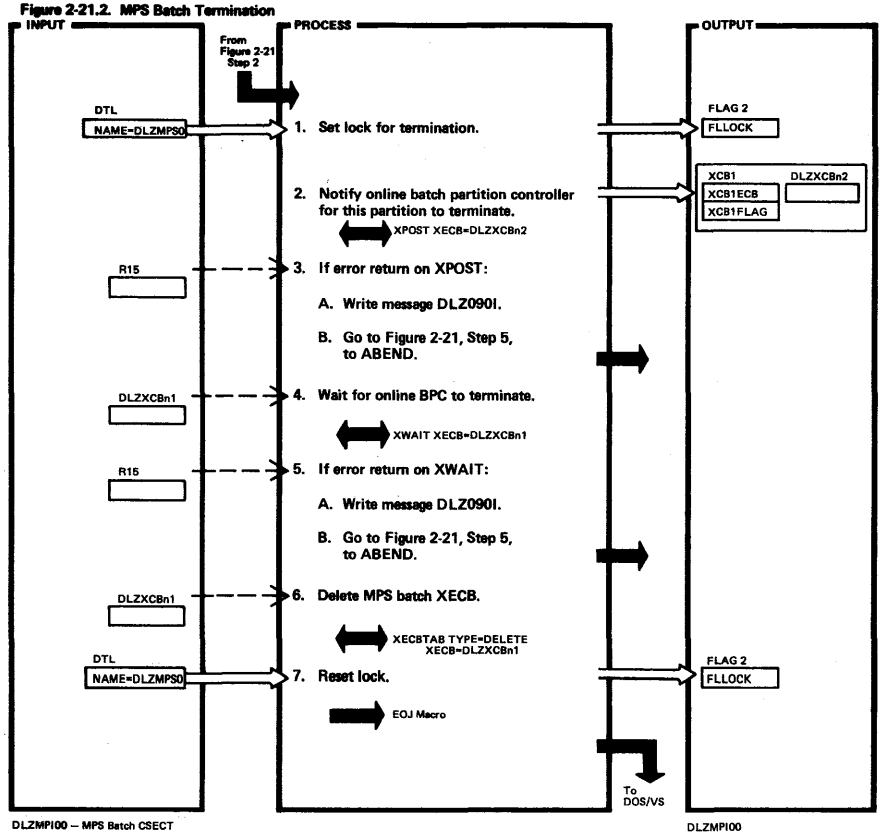
Figure 2-21.1. MPS Batch Initialization (Part 3 of 3)



Extended Description	Routine	Label
17. If online side successfully initializes, XECB DLZXCBN3 is created. Write message DLZ099I if DLZXCBN3 did not exist. Write message DLZ095I and the return code if a scheduling error indicated. Write message DLZ085I if BPC could not be attached.		XECBCHK3
18. No more need to serialize.		
19.		MVCOM
20.		PLILPI

Extended Description	Routine	Label
21. If PL/I – a three-word list is set up with pointers to PCBs in PLITABLE, amount of dynamic storage, and start of dynamic storage area for PL/I. If COBOL, RPG II. or Assembler – R1 points to first PCBADDR. The language indicator is set in the parameter field following the MPS batch XECB (See DLZXCB1 Macro).		
23. An application program runs as a subroutine of DLZMPI00.		

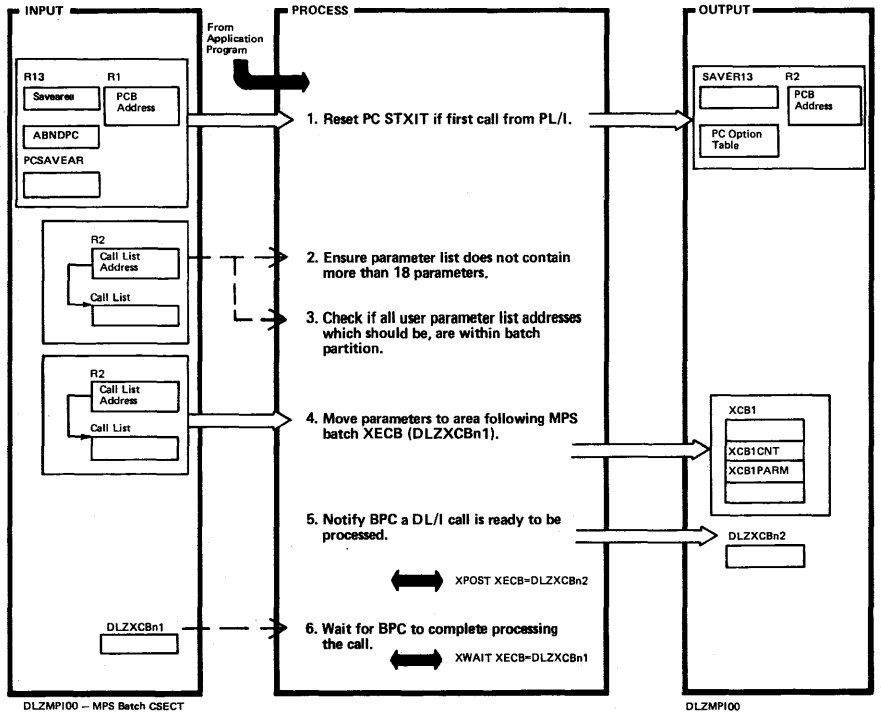
Figure 2-21.2. MPS Batch Termination



Extended Description	Routine	Label
1. This prevents another MPS batch task from starting or stopping while this one is terminating. The value of N is being set up for reuse during this locked interval and must complete before another MPS batch task can use it. 2. This routine is entered when the application program completes. If the application program is written in PL/I code, the SCDLIPLI flag in the SCD is reset to 0. Macro DLZXCB1 defines a DSECT representing the format of the MPS batch		

Extended Description	Routine	Label
XECB and following fields used for communicating with the online partition. The end-of-job flag is set to tell DLZBPC00 the batch partition is at EOJ and the BPC XECB is posted to tell DLZBPC00 to stop. 4. 7. No more need to serialize.		XWAIT2

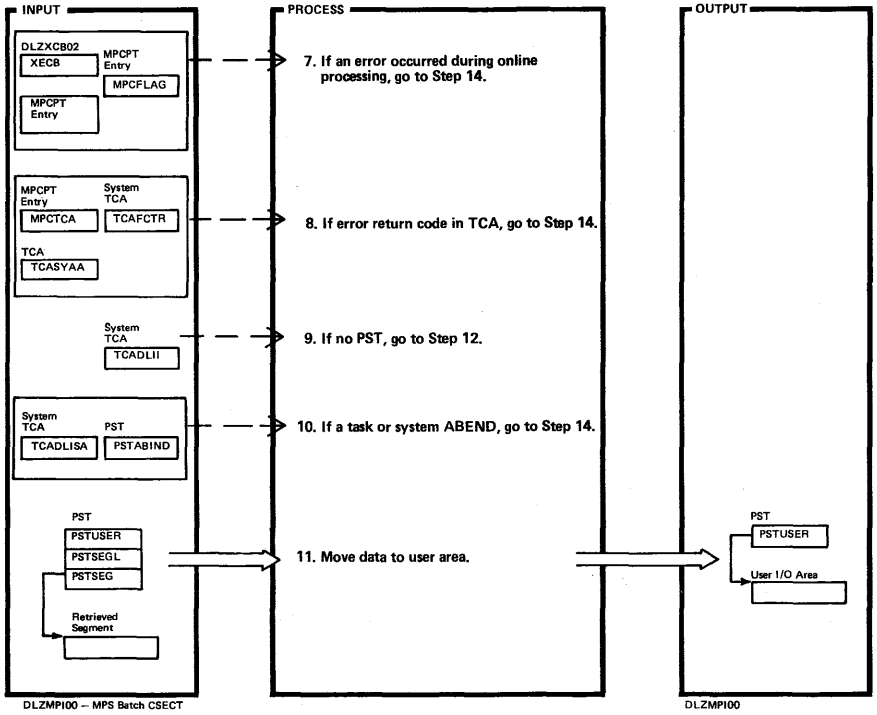
Figure 2-21.3. MPS Batch Program Request Handler (Part 1 of 3)



Extended Description	Routine	Label
1. This routine is entered on each call to DL/I made by the application program. During the first entry to DLZMPRH, the PL/I STXIT routine and savearea addresses from the PC option table are saved if the application program is written in PL/I. DLZMPRH also sets/resets a switch (SCDLIPLI flag in SCD) on exit/entry to indicate whether current execution is in DL/I code or PL/I code. This is done to enable high level language debugging for PL/I to give diagnostic information if a program check occurs in PL/I code. PL/I reissues STXIT PC when application program starts. Therefore, DL/I must reissue STXIT to get control after PL/I issues its STXIT PC.		DLZMPRH
2. Write message DLZ091I if more than 18 parameters.		COUNTLP
3. Ensure call list and addresses it points to are within batch partition (except for PCB). If PL/I, ensure that pointers pointed to by pointers, are within the batch partition.		CHKMOVE1

Extended Description	Routine	Label
3. (cont'd) Write message DLZ092I if there is a bad address and ABEND.		
4. Macro DLZXCB1 defines the DSECT describing the DLZXCBn1 XECB used for communicating with the online batch partition controller (DLZBPC00).		
5. If error return on XPOST, write message DLZ084I, then ABEND.		XPOST0
6. If error return on XWAIT, write message DLZ084I, then ABEND.		XWAIT0

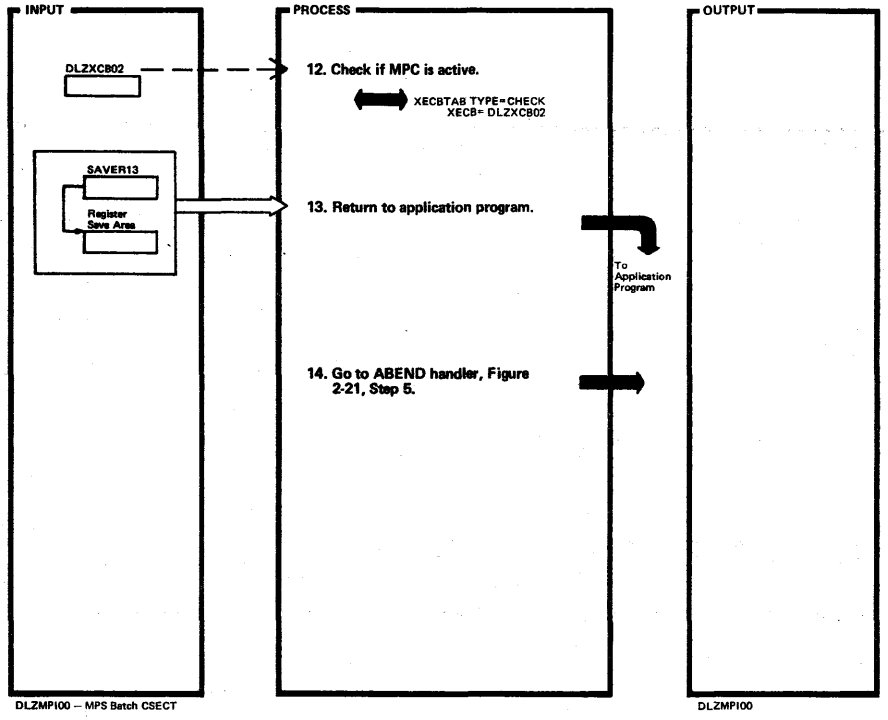
Figure 2-21.3. MPS Batch Program Request Handler (Part 2 of 3)



Extended Description	Routine	Label
7. MPCERR flag indicates an error condition occurred during BPC processing. Write message DLZ100I if on.		
8. Write message DLZ102I including the return code if present.		
9. The storage acquired indicator TCADLIPS is turned on by online nucleus DLZODP when the PST is acquired. If no PST, it just did a TERM call.		
10. Write message DLZ098I if PST contains an error.		
11. Write message DLZ100I if the data addresses are invalid.		

Extended Description	Routine	Label

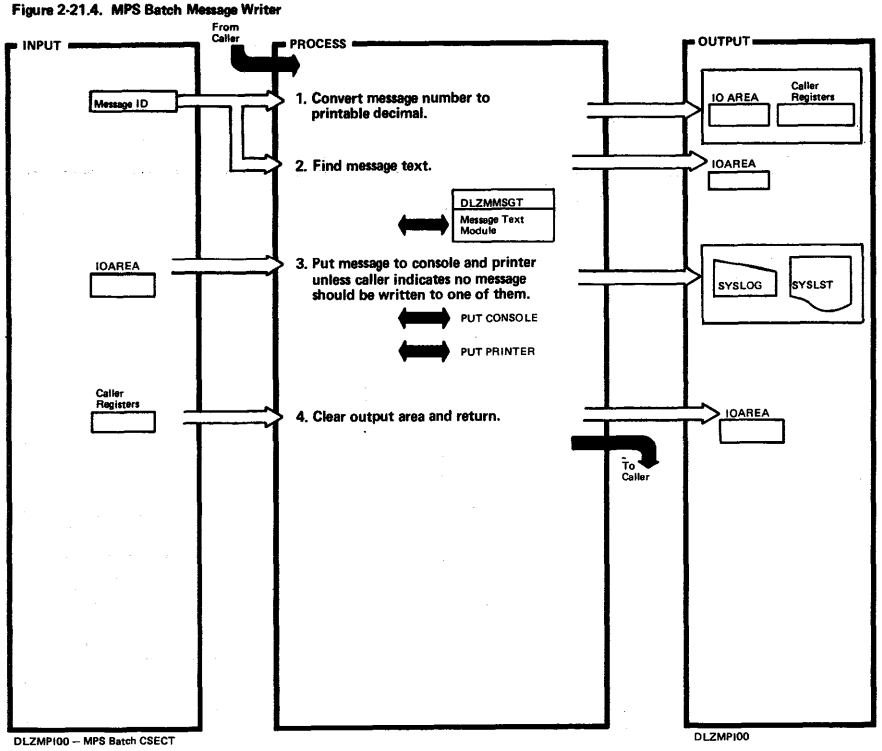
Figure 2-21.3. MPS Batch Program Request Handler (Part 3 of 3)



Extended Description	Routine	Label
12. If the START PARTITION XECB is not at the same address as when the batch job started indicating there was a deletion and new define, or if it no longer exists, write message DLZ082I and go to Step 14.		NODATA

Extended Description	Routine	Label

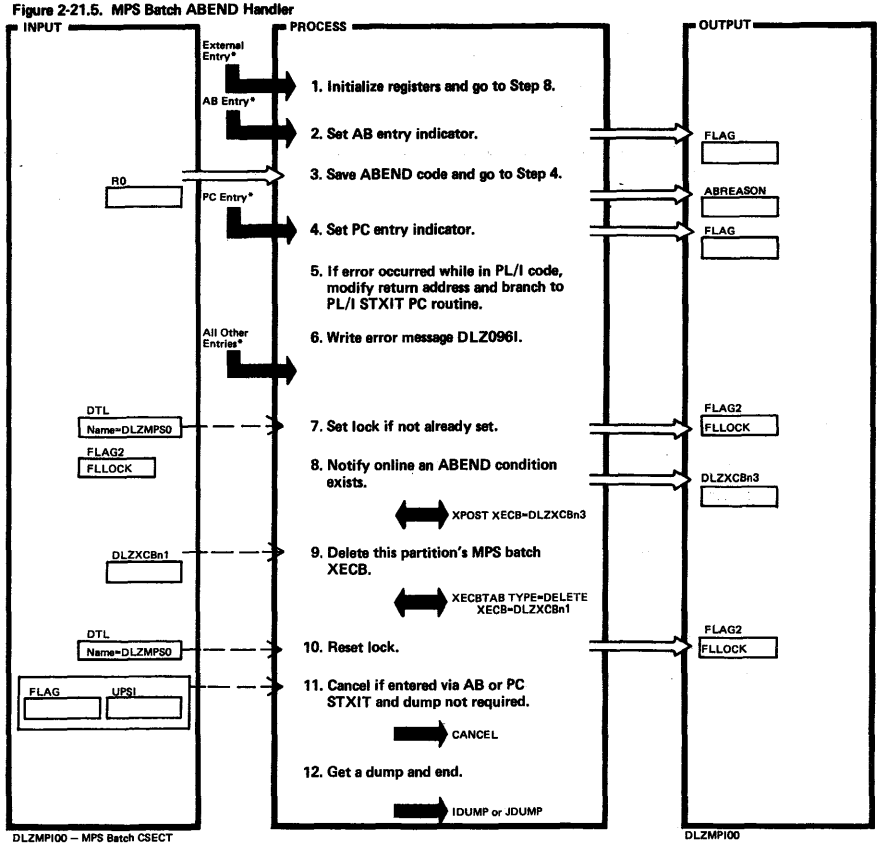
Figure 2-21.4. MPS Batch Message Writer



Extended Description	Routine	Label
This routine is entered at one of two labels: DLZMMSG – Main entry DLZMMSGX – Entry if branching to this routine from an external module. In either case, initialize registers and then do the listed steps.		
1. This routine is entered when a message is to be written to SYSLOG and/or SYSLST.		DLZMMSG
2. The message module DLZMMSGT includes all messages that can be issued by MPS and is used in both the batch and online partitions.		
3.		PUTCONS2

Extended Description	Routine	Label

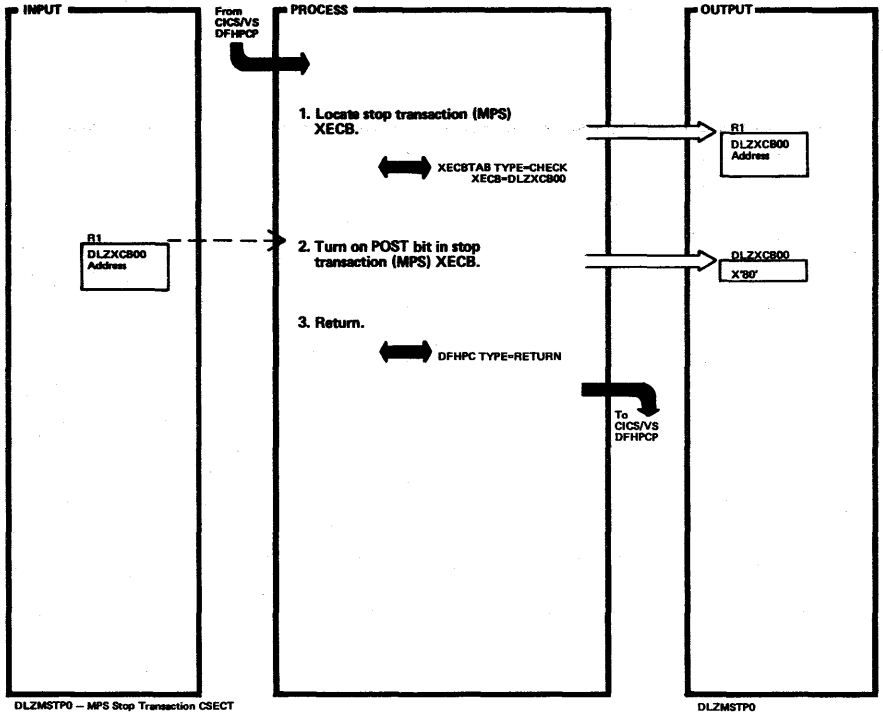
Figure 2-21.5. MPS Batch ABEND Handler



Extended Description	Routine	Label
*There are four entries to this routine: 1. External. 2. AB STXIT 3. PC STXIT 4. The MPS Batch Initialization, MPS Batch Termination, and MPS Program Request Handler routines (whenever XPOST is needed to tell online that batch completed unsuccessfully).		
2. The AB output area is located in a dump following the DC C'AB SAVE' characters.		ABNDAB
3. The AB reason code is located in a dump following the DC C 'AB ABEND CODE'.		
4. The PC output area is located in a dump following the DC C 'PC SAVE' characters.		ABNDPC

Extended Description	Routine	Label
5. The address of the PL/I STXIT PC routine was saved during the first entry to DLZMPRH (see Figure 2-21.3, Step 1). After PL/I completes diagnostic information, processing returns to modified address in DLZMABND.		
6.		ABPC
7. Prevents another MPS task from starting or stopping concurrently.		
8.		DLZMABND
9.		XPOST3
10. No longer need to serialize.		
12. IDUMP issued if support exists; otherwise JDUMP.		DLZIDUMP

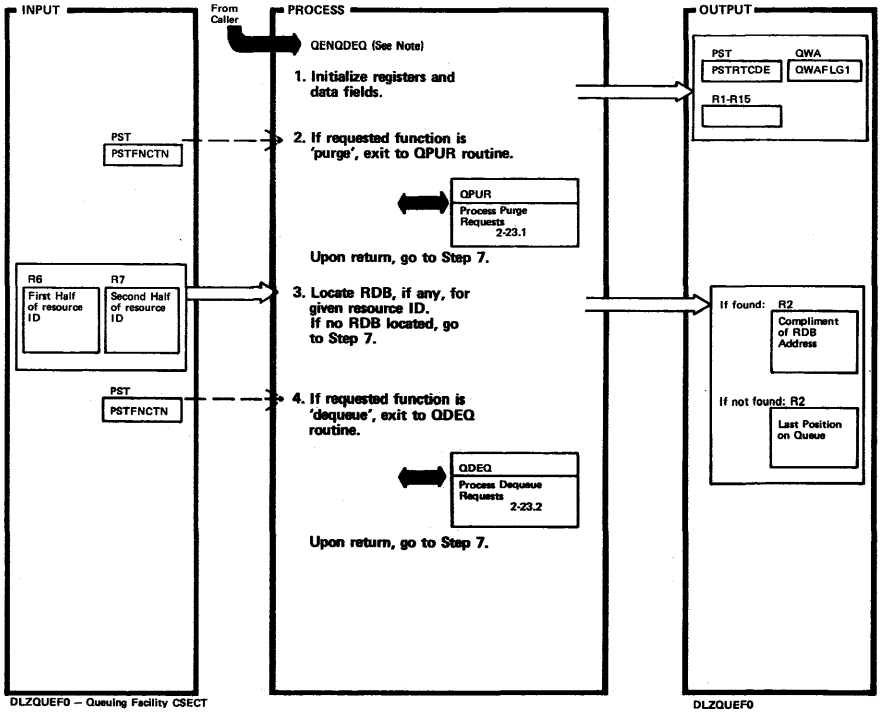
Figure 2-22. Stop Transaction



Extended Description	Routine	Label
1. Module identifier (DLZMSTP0) is defined here. Write message DLZ080I if DLZXCB00 does not exist -- MPS not active -- and go to Step 3.	DLZMSTP0	DLZMSTP0
2. Note that the XPOST macro is not needed to turn on the POST bit because stop transaction XECB (DLZXCB00) is defined in the same partition as this module. DLZXCB00 is defined by DLZMPC00.		
3.		RETURN

Extended Description	Routine	Label

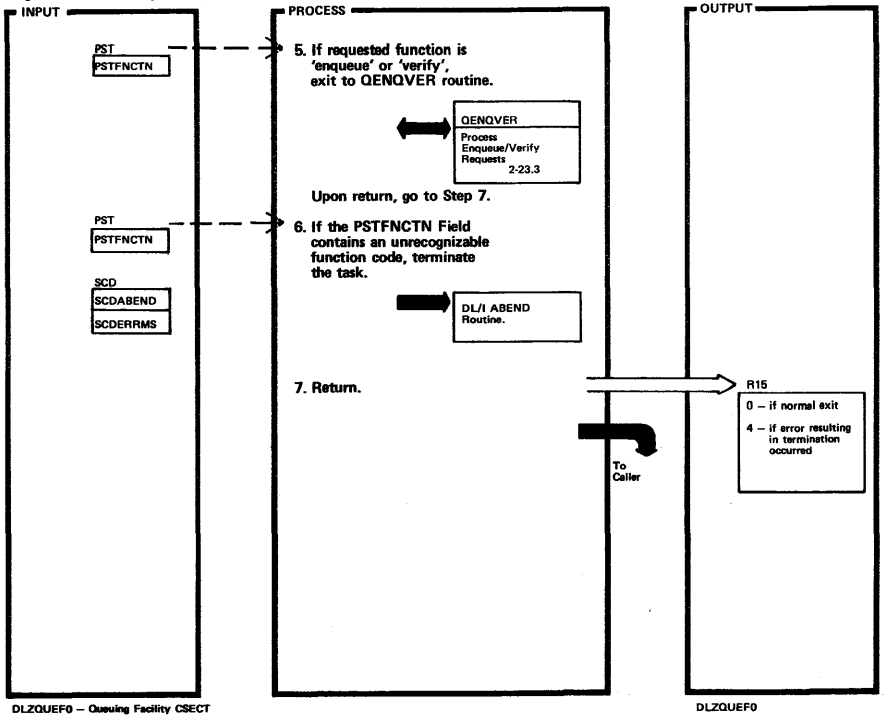
Figure 2-23. Queuing Facility (Overview) (Part 1 of 2)



Extended Description	Routine	Label
Note: QENQDEQ is the general purpose entry point for requests to 'enqueue; 'dequeue', or 'verify' a resource, or to 'purge' all enqueues for a task.		
1. Module identifier (DLZQUEF0vrnp) is defined here. The level format is vrnp; where 'v' is the version, 'r' is the release, 'n' is an additional identification number, and 'p' is the latest PTF number that has been applied.	QENQDEQ	
3. The proper queue head is first located by hashing the resource ID. That queue is then searched for a RDB with the same resource ID. If found, its address is passed back compli-mented. Otherwise, the address of the last position on the queue is returned.	QLOCRDB	

Extended Description	Routine	Label

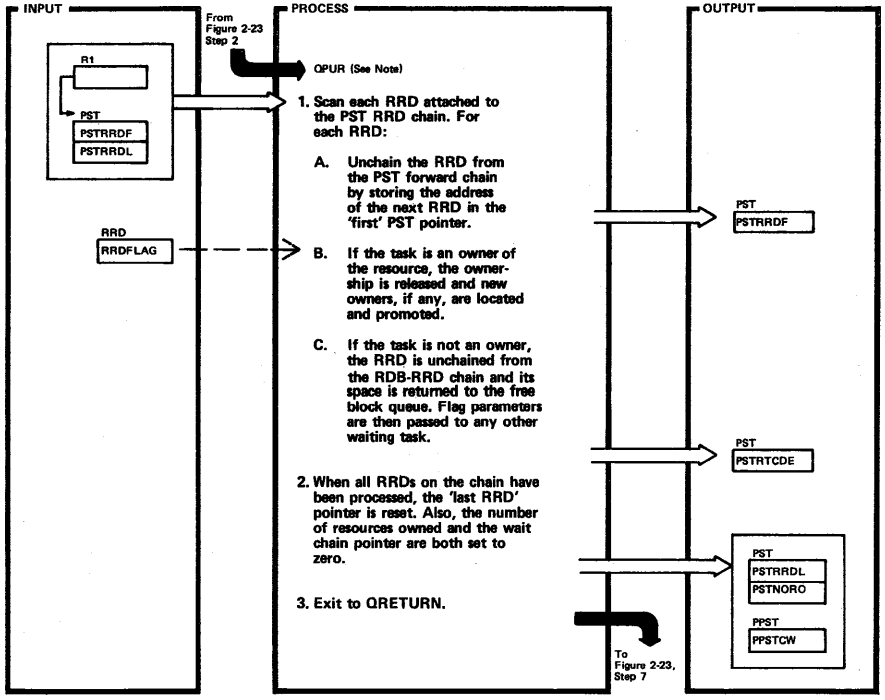
Figure 2-23. Queuing Facility (Overview) (Part 2 of 2)



Extended Description	Routine	Label
6. Message DLZ2671 is issued.		
7. This routine is used for common processing during exit for enqueue, dequeue, verify, and purge request. If processing is successful, a 0 return code is set in R15. Otherwise, the PST return code field (PSTRTCDE) is checked for error return codes. If present, the R15 return code is set to 4 and the registers are saved in the ABEND save area.	QRETURN	

Extended Description	Routine	Label

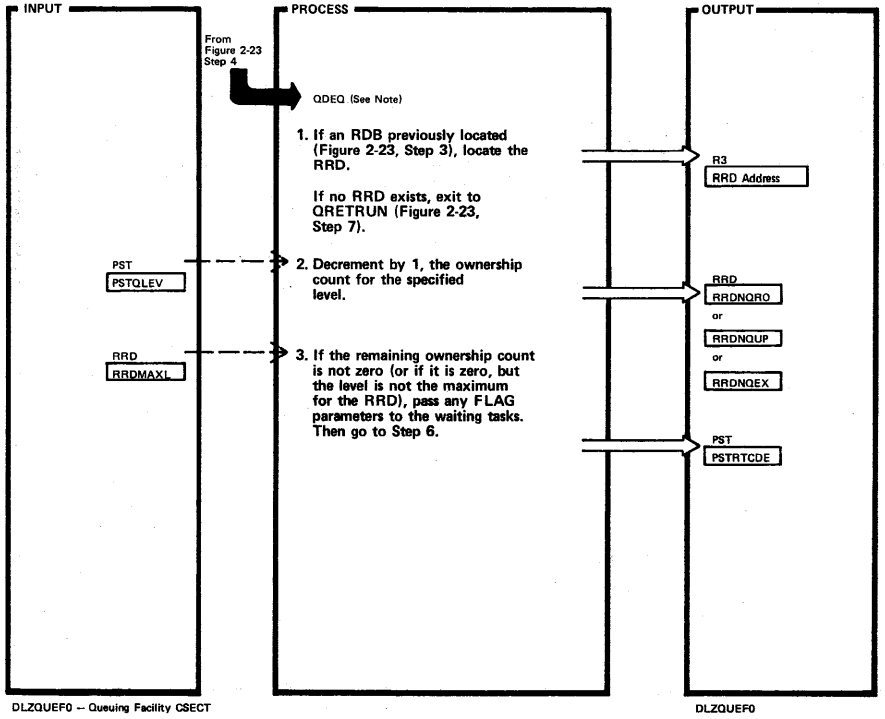
Figure 2-23.1. Process Purge Requests



Extended Description	Routine	Label
Note: This routine is used to purge all complete or outstanding enqueue requests for a given task.		
1. A. The address of the next RRD is saved because it would be destroyed when the current RRD is returned to the free space.	QPUR	
B. This routine is called to relinquish ownership of a resource by dequeuing the RRD for the task. It is entered by purge at entry point QDEQPUR to avoid the unnecessary overhead of unchaining the RRD from the PST.	QRELRSC	
C.	QPUR1 QRETBLK QPFLAGP	
2.	QPUR2	
3.	QPURX	

Extended Description	Routine	Label

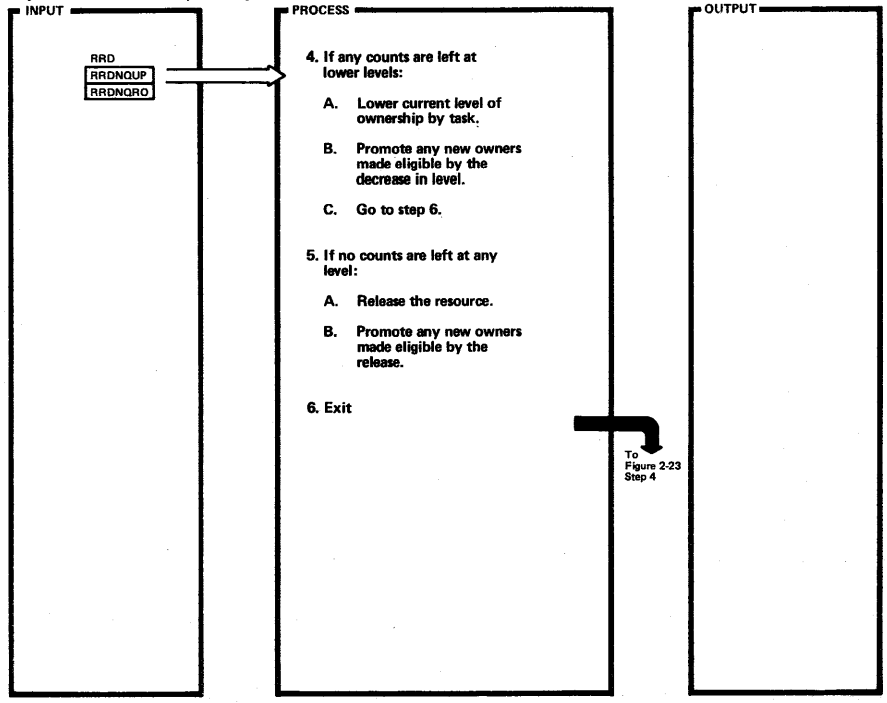
Figure 2-23.2. Process Dequeue Requests (Part 1 of 2)



Extended Description	Routine	Label
Note: This routine is used to process a request to dequeue a resource.		
1.		QLOCRRD
2. This routine is also used by 'verify' processing to dequeue a resource that was enqueued due to a required wait.	QDEQVER	
3.	QDEQVER2 QPFLAGP	

Extended Description	Routine	Label

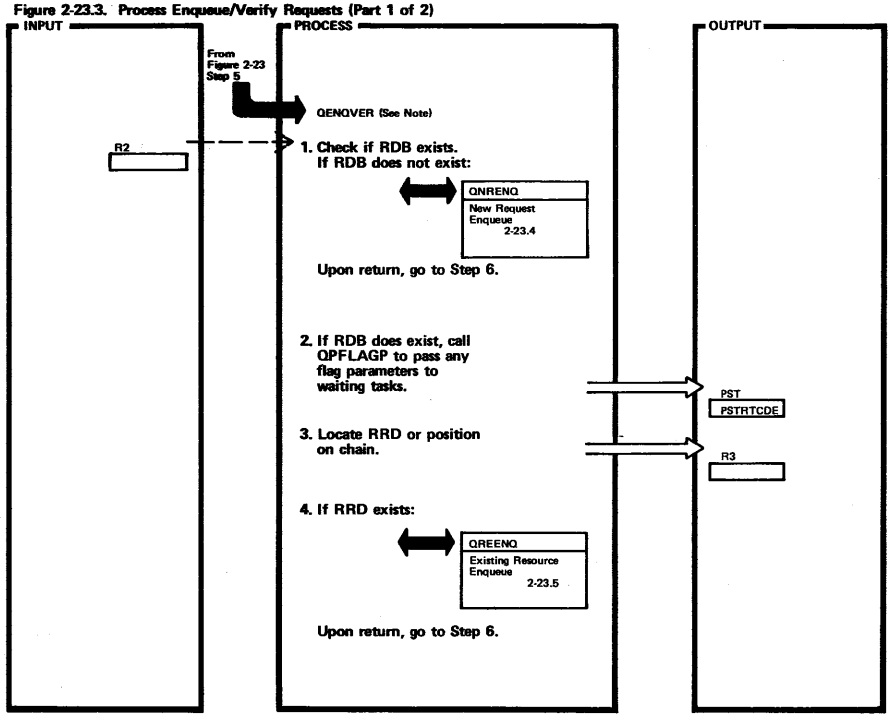
Figure 2-23.2. Process Dequeue Requests (Part 2 of 2)



Extended Description	Routine	Label
4.	QDEQVER3 QDEQVER4 QDEQVER5 QPNDWCM	
5. Entry point QDEQPUR in this routine is used by 'purge' processing to release a resource already unchained from the PST/RRD chain.	QRELRSC	
6.	QDEQVERX QRELRSCX QDEQPURX	

Extended Description	Routine	Label

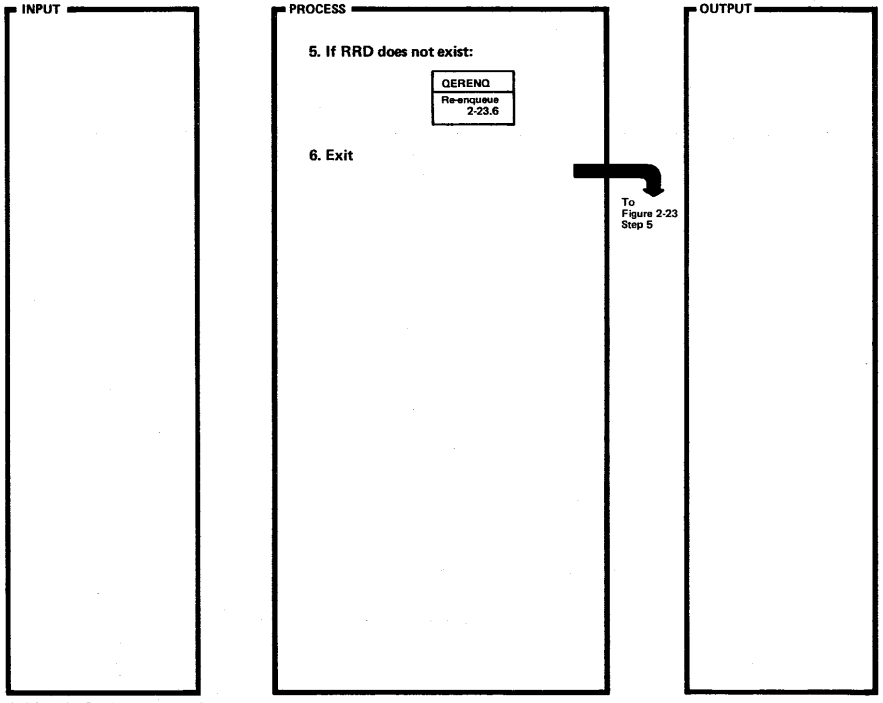
Figure 2-23.3. Process Enqueue/Verify Requests (Part 1 of 2)



Extended Description	Routine	Label
Note: This routine is used to process enqueue and verify requests. It determines the type of enqueue (or verify) and enters the proper routine for the type. The possible types are: • New resource enqueue – The enqueue of a resource not currently enqueued. • Existing resource enqueue – The enqueue of a resource currently enqueued, but not by this task. • Re-enqueue – The enqueue of a resource currently enqueued by this task. Processing of enqueue and verify requests is essentially the same. The difference is that on the return from verify processing, the user is not the owner of the resource.		
1.	QENQVER	
2.	QPFLAGP	
3.	QLOCRRD	

Extended Description	Routine	Label

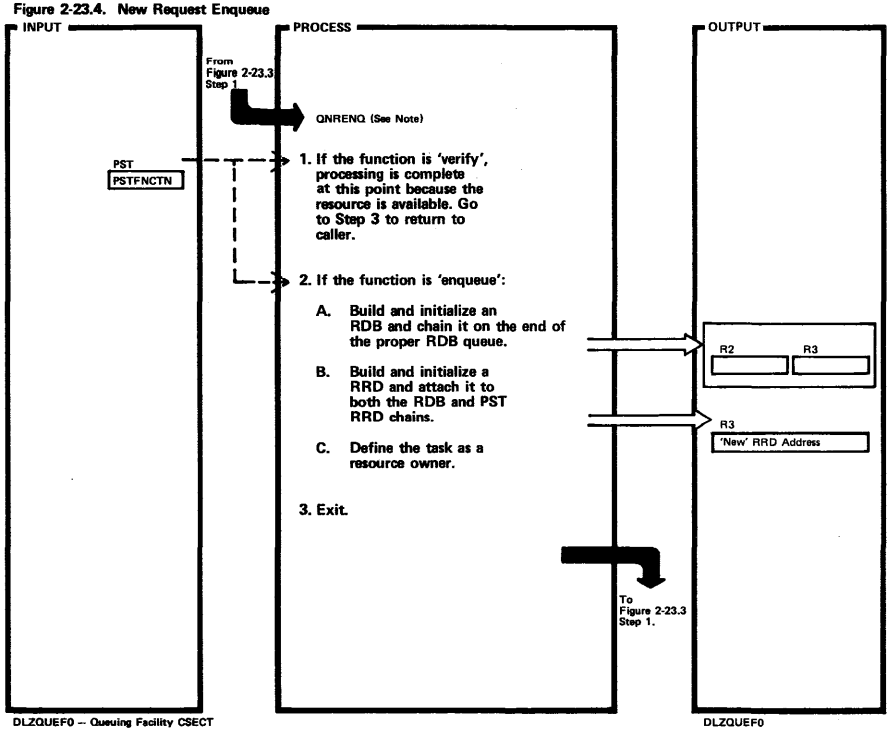
Figure 2-23.3. Process Enqueue/Verify Requests (Part 2 of 2)



Extended Description	Routine	Label
6.	QENQVERX	

Extended Description	Routine	Label

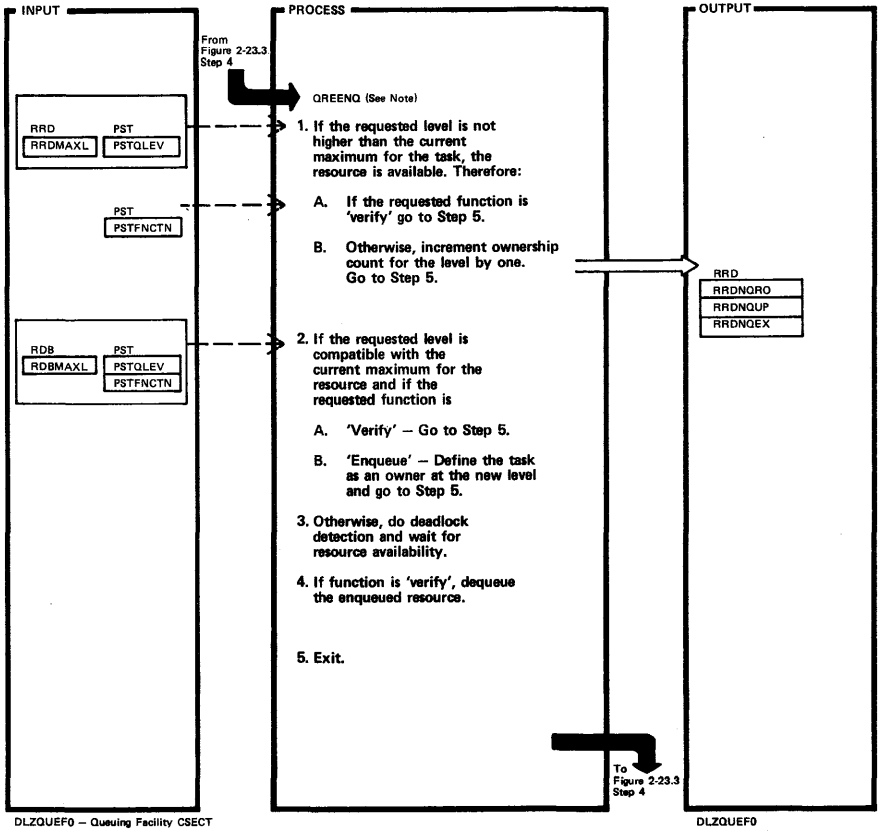
Figure 2-23.4. New Request Enqueue



Extended Description	Routine	Label
Note: This routine is used to process an enqueue (or verify) request for a resource that has no current enqueues outstanding.		
2. A. R2 points to RDB chain location. R3 points to RRD chain head in RDB.	QBLDRDB	
B.	QBLDRRD	
C.	QDASOWN	
3.	QNRENQX	

Extended Description	Routine	Label

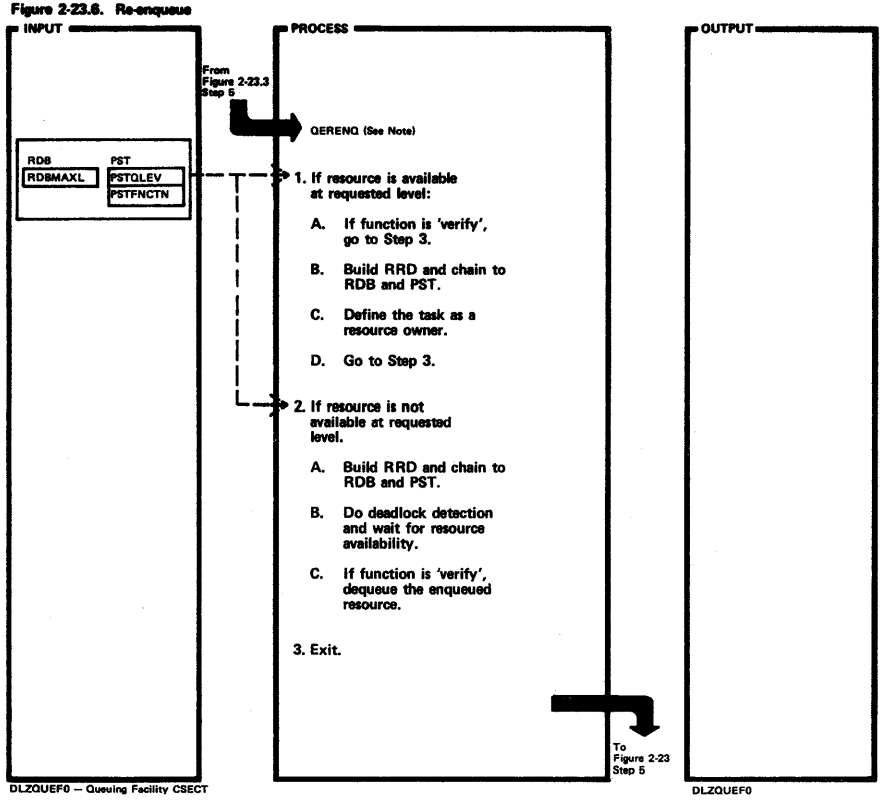
Figure 2-23.5. Existing Resource Enqueue



Extended Description	Routine	Label
Note: This routine is used to process an 'enqueue' or 'verify' for a resource that is currently enqueued by the requesting task.		
2.	QREENQ1 QREENQ4	
B.	QREENQ1 QREENQ5	
3. The task will be defined as an owner of the resource during 'dequeue' processing for other task.	QREENQ3 QREENQ4 QWAIT	
The QDLKDTN routine detects a deadlock condition and resolves the deadlock by picking and terminating one of the tasks involved. The task terminated is selected as follows:	QDLKDTN	

Extended Description	Routine	Label
1. Online tasks are picked before MPS tasks.		
2. Within a class, the task with the fewest resources currently enqueued is chosen.		
3. In the event of a tie, the choice is arbitrary.		
4.	QREENQ3 QREENQ4 QDEQVER	
5.	QREENQX	

Figure 2-23.6. Re-enqueue



Extended Description	Routine	Label
Note: This routine is used to process an 'enqueue' or 'verify' request for a resource that is currently enqueued, but not by the requesting task.		
1.	QERENQ	
B. To build and chain RRD, call QBLDRRD.	QBLDRRD	
C. To make task a resource owner, call QDASOWN.	QDASOWN	
2.	QERENQ2	
A. To build and chain RRD, call QBLDRRD.	QBLDRRD	
B. The task will be defined as an owner of the resource, during 'dequeue' processing for other tasks.	QWAIT QDLKDTN	

Extended Description	Routine	Label
The QDLKDTN routine detects a deadlock condition and resolves the deadlock by picking and terminating one of the tasks involved. The task terminated is selected as follows: 1. Online tasks are picked before MPS tasks. 2. Within a class, the task with the fewest resources currently enqueued is chosen. 3. In the event of a tie, the choice is arbitrary.		
C.	QERENQ3 QDEQVER	
3.	QERENQX	

Figure 2-24. Visual Table of Contents for DL/I Utility Modules HIPO Charts

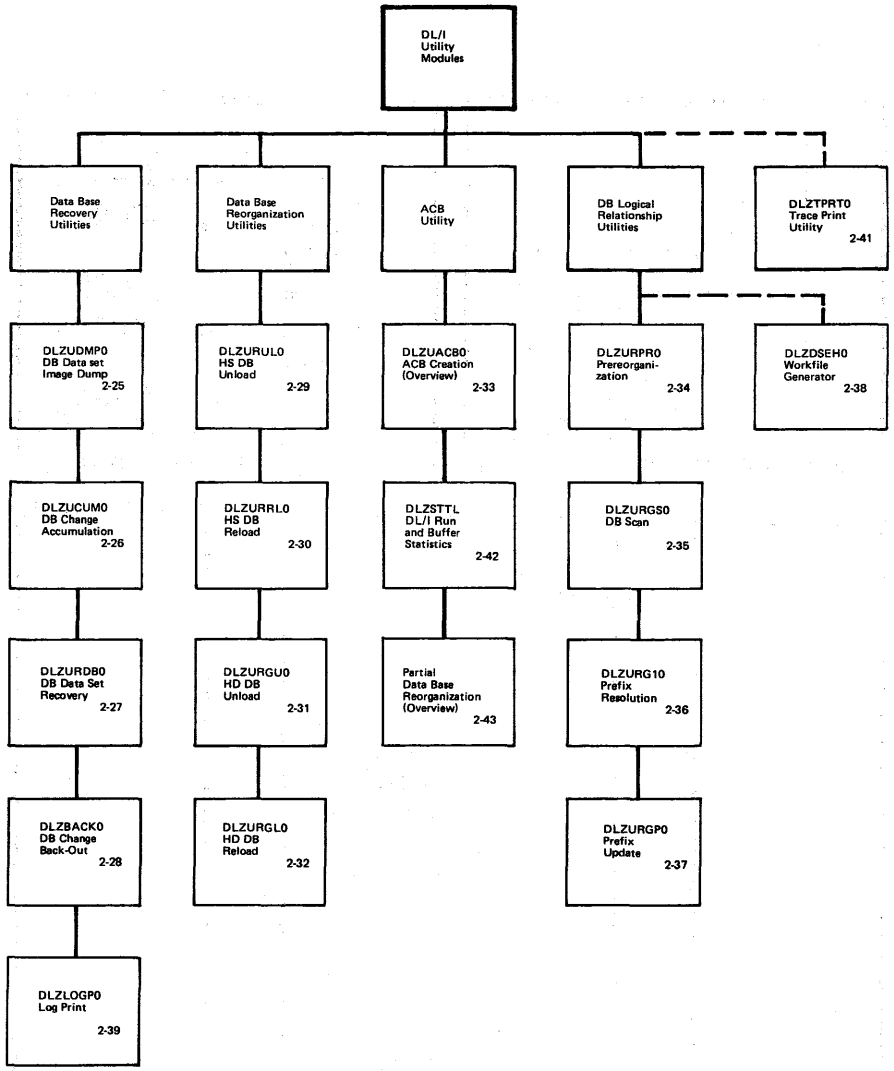
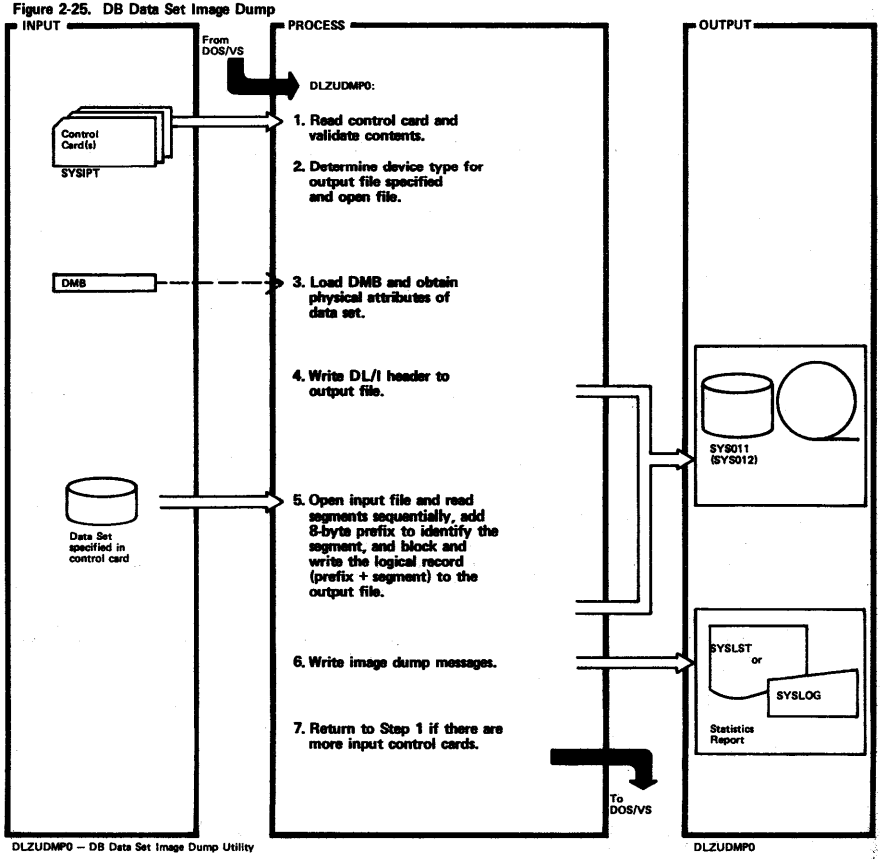


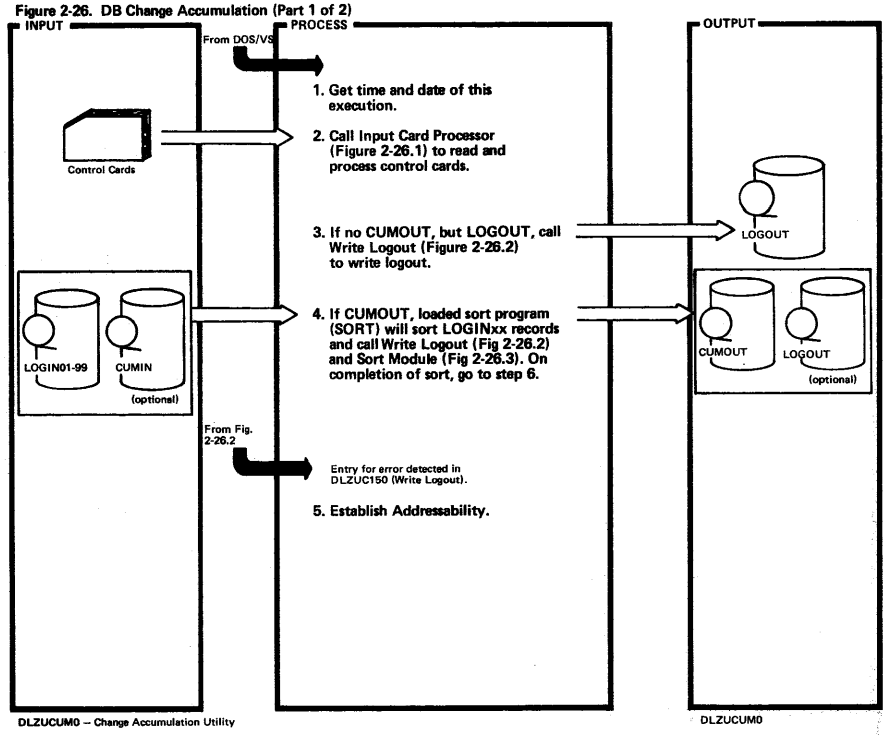
Figure 2-25. DB Data Set Image Dump



Extended Description	Routine	Label
1. Read and validate control statement. Write the following messages as needed: DLZ302I - Column 1 not D DLZ303I - Column 2 not 1 or 2 DLZ304I - DBD name field not specified DLZ307I - Input filename not specified DLZ308I - Output filename not specified DLZ309I - Error(s) found in control statement DLZ310I - Image of erroneous control statement 2. DLZDVCE macro obtains data from PUB.Device type may be tape or DASD. 4. The header record contains information that allows the use of the image dump file by the DB data set recovery utility.		

Extended Description	Routine	Label

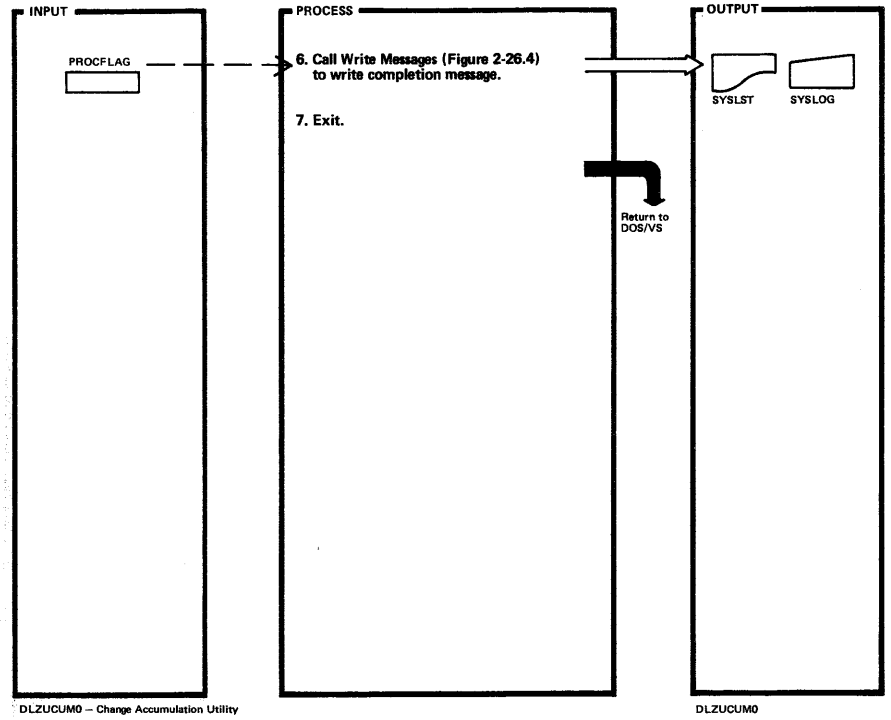
Figure 2-26. DB Change Accumulation (Part 1 of 2)



Extended Description	Routine	Label
1. Header line is printed on SYSLST.		TIMEDEC
2. Three returns as follows:		READCD
• Error – issue error message.		BADEND
• No accumulation output, call Write Logout (Fig 2-26.2). Then issue successful run message.		GOODEND
• Accumulation output, call SORT.		SORT
4. SORT is invoked by LOAD and BALR. At exit 35, Sort Module (Fig 2-26.3) is called.		SORT
5. This entry point is necessary because Write Logout, not knowing who called (DB Change Accumulation or SORT), must return to this module if an error was detected.		DLZERRTN

Extended Description	Routine	Label

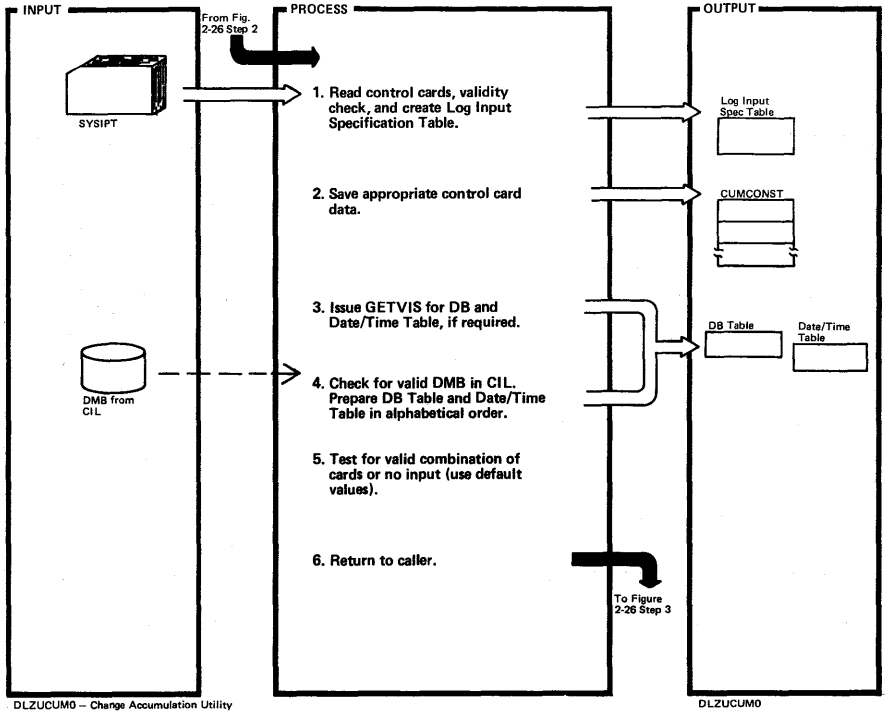
Figure 2-26. DB Change Accumulation (Part 2 of 2)



Extended Description	Routine	Label
6. May be OK message or error message from SORT, Write Logout, or Sort Module. If PROCTERM X'01' bit is on in PROCFLAG, an error occured.		CLOSE

Extended Description	Routine	Label

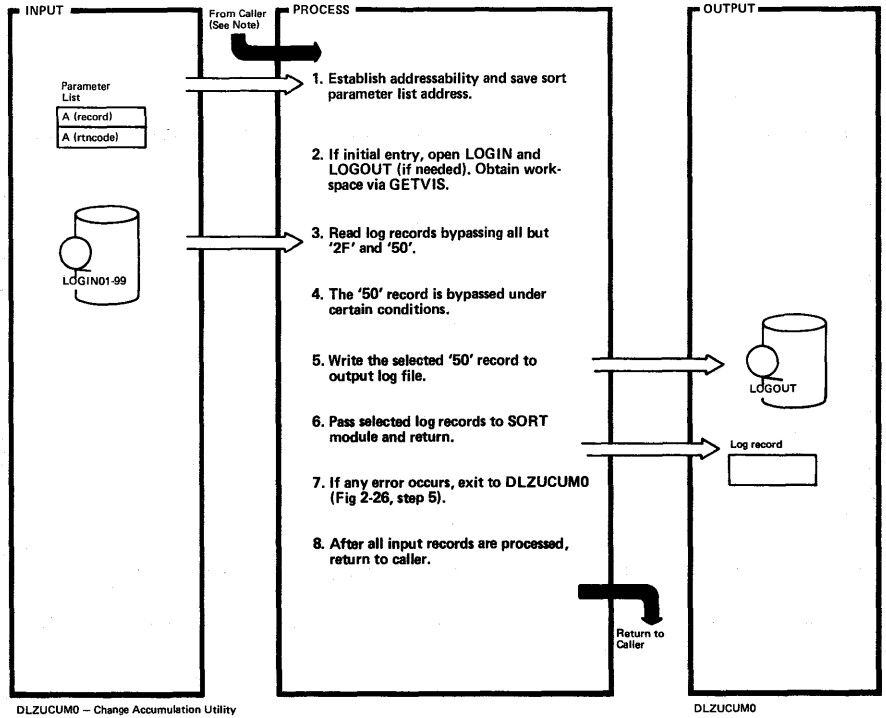
Figure 2-26.1. Input Card Processor (DLZUCCT0)



Extended Description	Routine	Label
1. Possible card types are: • 'ID' specifies db number, max key length, number of sort, work, and log files. • 'DB0' describes records to be accumulated from input and written to CUMOUT. • 'DB1' describes records to be written to new log file. • 'LI' describes a log input file. • Error card – call Write Message (Fig 2-26.4) to write appropriate error message.		GETCARD ERROR
2. Data from control card(s) is saved in a dsect residing in DLZUCUM0, addressable by all modules in this utility. The dsect name is DLZUCUMC.		

Extended Description	Routine	Label
3. Tables are not required if *ALL was specified. The number of entries in each table is equal to the number of data bases as specified on the ID control card or default of 16.		GETMAIN
4. This information is filled from the DB0 and/or DB1 card(s) if present.		DDNUMCHK
5. If any errors occur during steps 3, 4, of 5, call Write Messages (Fig 2-26.4) to write error messages and exit.		

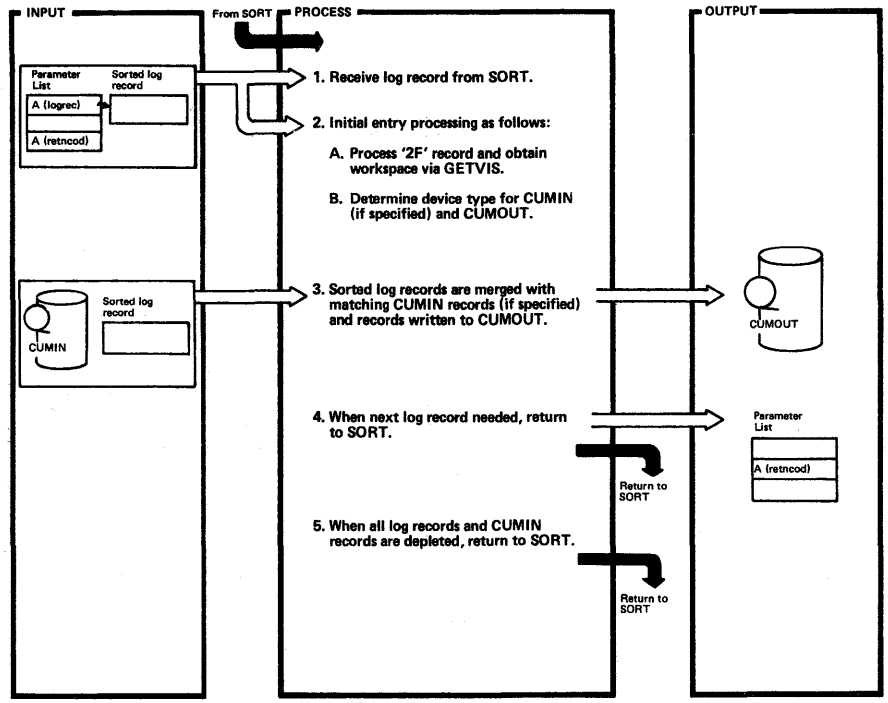
Figure 2-26.2. Write Logout (DLZUC150)



Extended Description	Routine	Label
Note: This program has two entry points: • DLZUC150 – from SORT. Entered when SORT wants another input record. • DLZUEX15 – from Figure 2-26, step 3 (DLZUCUM0). 3. On EOF, the file is closed. If more input specified, xx (LOGINxx) in the DTF or ACB is incremented by 1 and the next log file is opened. 4. Bypass '50' record for the following: • *ALL and log date/time less than purge date/time. • dbname match and log date/time less than purge date/time. • No dbname match and *OTHER not specified.		

Extended Description	Routine	Label
4. (con't) • No dbname match and log date/time less than purge date/time. 5. Write log record for the following '50': • *ALL on DB1 card. • Dbname match and dbname on DB1 card. • No dbname match and *OTHER on DB1 card.		

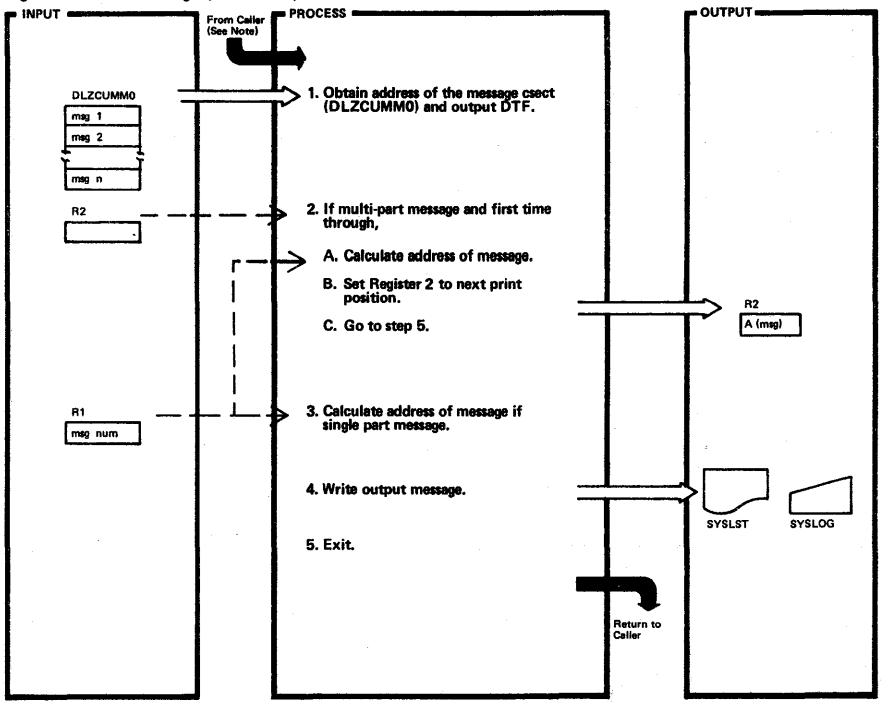
Figure 2-26.3. Sort Module (DLZUC350)



Extended Description	Routine	Label
1. SORT returns at EOF with an indication that no more records exist.		DLZUEX35
2. DLZDVCE macro obtains data from PUB. Device type may be TAPE or DASD.		TSTEODDB
3. The following merging logic is used for comparison of LOGIN and CUMIN to create CUMOUT. • For every new DMB name (data set ID), an accumulation header record is written either from the CUMIN record or created from the '2F' record. • Every CUMIN record is purge checked by date/time as specified by the user. The DB table as modified by DLZUC150 is used for a specific DMB or the *ALL/ *OTHER purge date is used as applicable. • If a matching log record is found, all log records with the same data ID will be merged with the CUMIN		

Extended Description	Routine	Label
3. (con't) record and written to CUMOUT. • If there is no matching log record, the CUMIN record is written to CUMOUT unchanged. • If log records exist but no CUMIN, the log records are accumulated by data ID and written to CUMOUT.		
4. A 'delete' return code is given to SORT so that SORT does not further process the current record. SORT will prepare the next input record and enter this program at step 1.		
5. Free all work areas and close CUMIN and CUMOUT. Indicate 'no return' to SORT.		ENDJOB ENDSORT

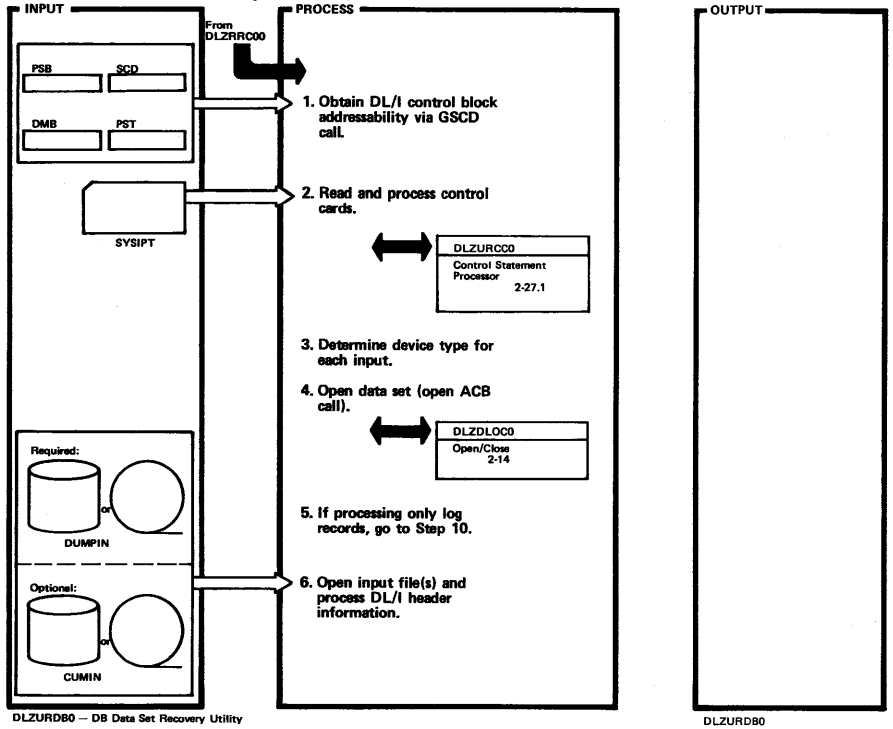
Figure 2-26.4. Write Messages (DLZUCER0)



Extended Description	Routine	Label
Note: This module can be called by DLZUCUM0, DLZUCCT0, DLZUC150, or DLZUC350.		
1. The address of the output DTF which has already been opened, is found in the CUMCONST table.		INITSV
2. Multi-part message is indicated by negative R2.		TESTR2
3. R1 contains message number.		MSGCOMM
4. Output can be to SYSLST or SYSLOG.		MSGWRT
5.		RETURN

Extended Description	Routine	Label

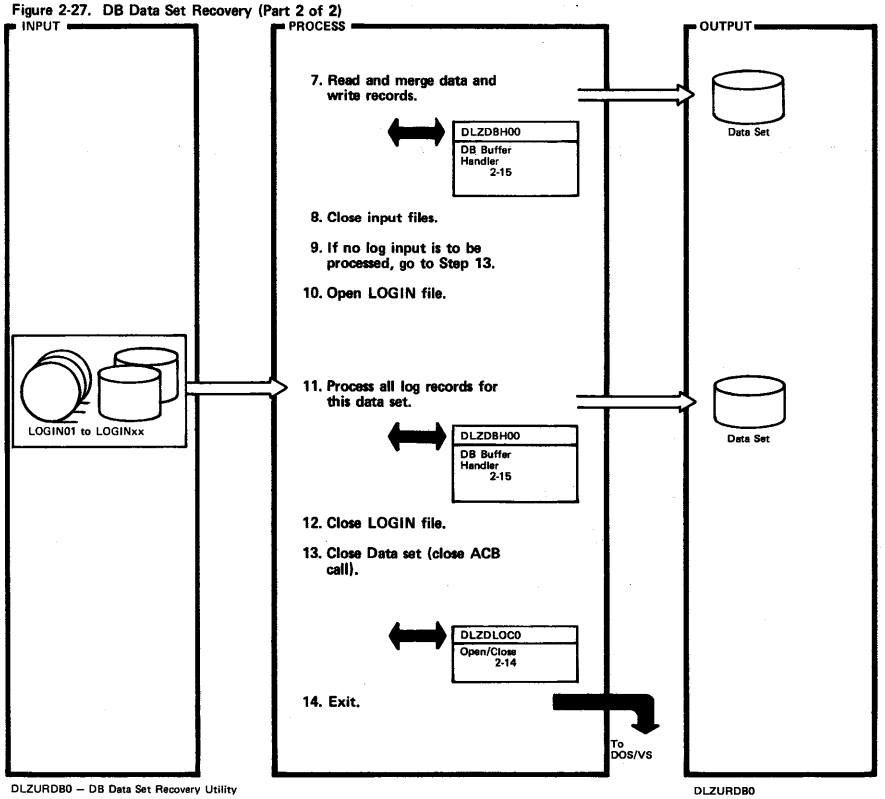
Figure 2-27. DB Data Set Recovery (Part 1 of 2)



Extended Description	Routine	Label
2. There are three returns: • No errors – continue processing. • No input statements – terminate processing. • Error – issue error message and terminate processing 3. DLZDVCE macro obtains data from PUB. Device type may be tape or DASD. 6. DUMPIN file is mandatory and may be output from DLZUDMP0 or DLZURUL0. CUMIN file is optional and is output from DLZUCUM0.		GETDMB CLEANUP BADRUN

Extended Description	Routine	Label

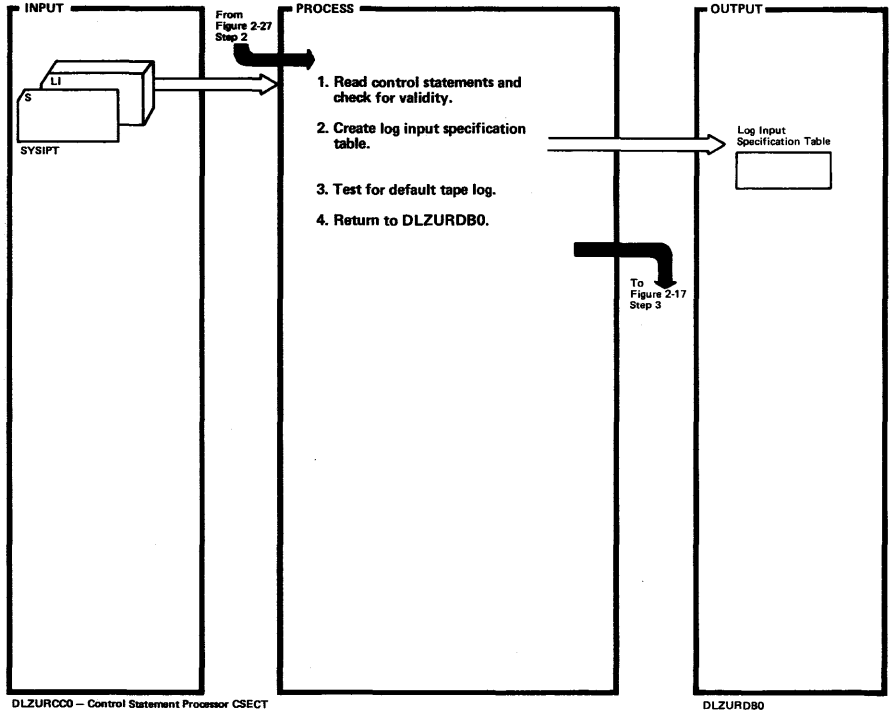
Figure 2-27. DB Data Set Recovery (Part 2 of 2)



Extended Description	Routine	Label
7. Records are read from DUMPIN and CUMIN via GET calls and are written in ascending order (compare by key if KSDS, and by RBA if ESDS). The proper PSTFNCTN is supplied for call of buffer handler. 9. LOGIN file is optional. 11. LOGIN01 to LOGINxx files are processed sequentially.		SETFLOW PROCLOGS

Extended Description	Routine	Label

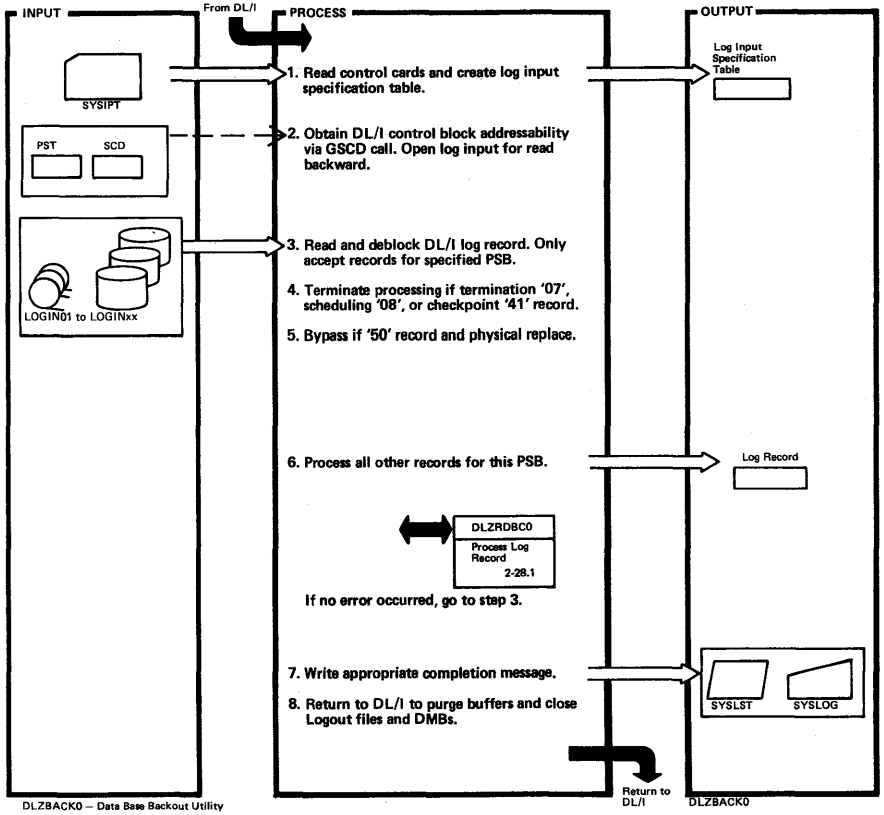
Figure 2-27.1. Control Statement Processor



Extended Description	Routine	Label
1. Possible card types are: 'S' – identifies data set to be recovered. 'LI' – describes log input file(s). Write the following messages as needed: DLZ302I - Column 1 not S DLZ304I - DBD name not specified DLZ307I - Input filename not specified DLZ310I - Image of erroneous control statement DLZ342I - Invalid number of log files DLZ372I - Invalid log buffer size	DLZURCC0	GETCARD
2. One entry in table describing file type, logical unit, and buffer size for each log file.		
3. If no log file is specified, issue macro DLZDVCE to see if SYS013 assigned to tape.		CLEANUP

Extended Description	Routine	Label

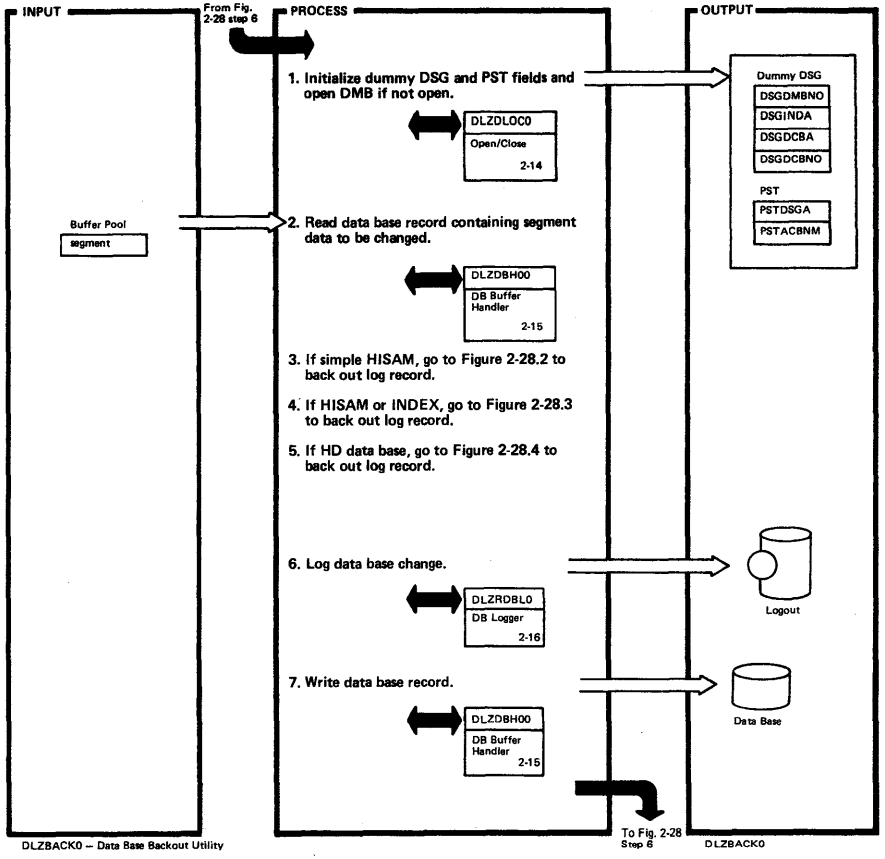
Figure 2-28. DB Change Backout



Extended Description	Routine	Label
1. 'LI' control cards describe one input log file each.		
2. Initialize PSTDBPCB, PSTDGU, and PSTDGN.		INIT
3. At end of file, go to step 7.		READ NXTLREC
4.		CHKLOGT
5.		CHKDPHYR
6. The log record is placed in a work area (READAREA) whose address DLZRDBC0 obtains via a V-con.		OK CALLBO
7. The input log file is closed. If another log file exists, it is opened and processing continues with step 2. The message texts are found in DLZBACM0 csect.		EOF MSGGEN

Extended Description	Routine	Label

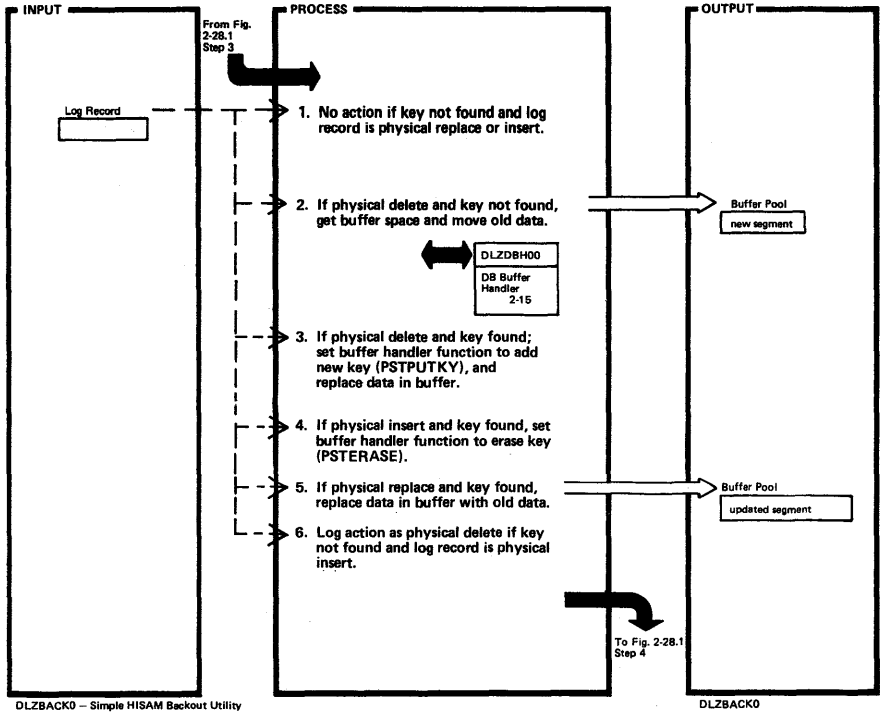
Figure 2-28.1. Process Log Record (DLZRDBC0)



Extended Description	Routine	Label
1.		INIT
2. The following calls were made to the buffer handler:		READREC
A. If HISAM KSDS, issue PSTSTLEQ call.		LOCKEY
B. If HISAM ESDS, issue PSTBYLCT call.		LOCBYTE
C. If HD ESDS, issue PSTBKLCT call.		LOCKBLK
6. Output log records contain the 'opposite' function to which was on the input log.		LOGNDWRT LOG
7. The return code is checked and appropriate action is taken depending on the call and return code.		WRITEBFR

Extended Description	Routine	Label

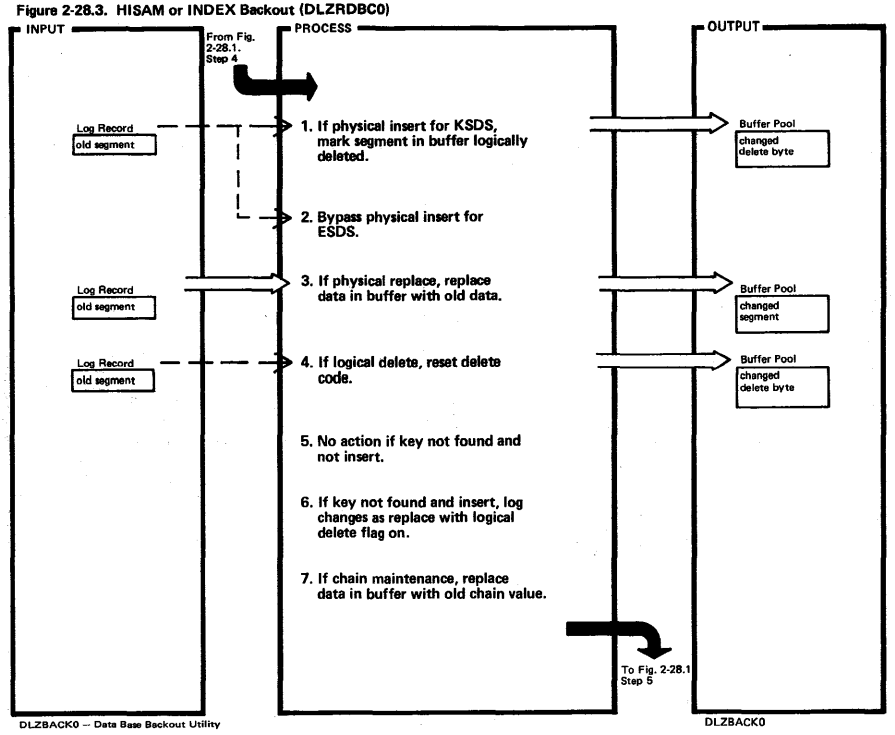
Figure 2-28.2. Simple HISAM Backout (DLZRDBC0)



Extended Description	Routine	Label
1. The address of the log record is input to this routine.		KEYNOTFD SHISNREC
2.		KEYNOTFD
4.		SHISREC
5.		CALLREP
6.		SHISNREC

Extended Description	Routine	Label

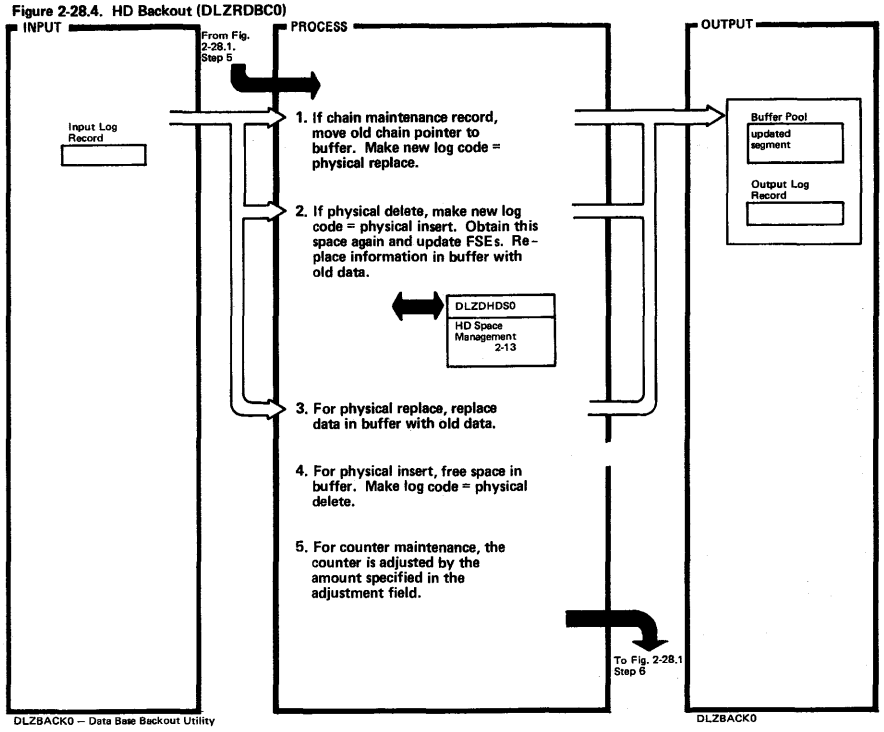
Figure 2-28.3. HISAM or INDEX Backout (DLZRDBC0)



Extended Description	Routine	Label
1. If segment is an INDEX data base (primary or secondary), the pointer to the index target segment is also zeroed.		HISREC HISISRT
2. Chain maintenance log records for KSDS effectively back out physical insert to ESDS.		HISREC
3.		HISREC
4.		LGDLET
5.		CHKISRTH
6. If segment is an INDEX data base (primary or secondary), the pointer to the index target segment is also zeroed in the log record.		CHKISRTH
7.		CHNMAIN

Extended Description	Routine	Label

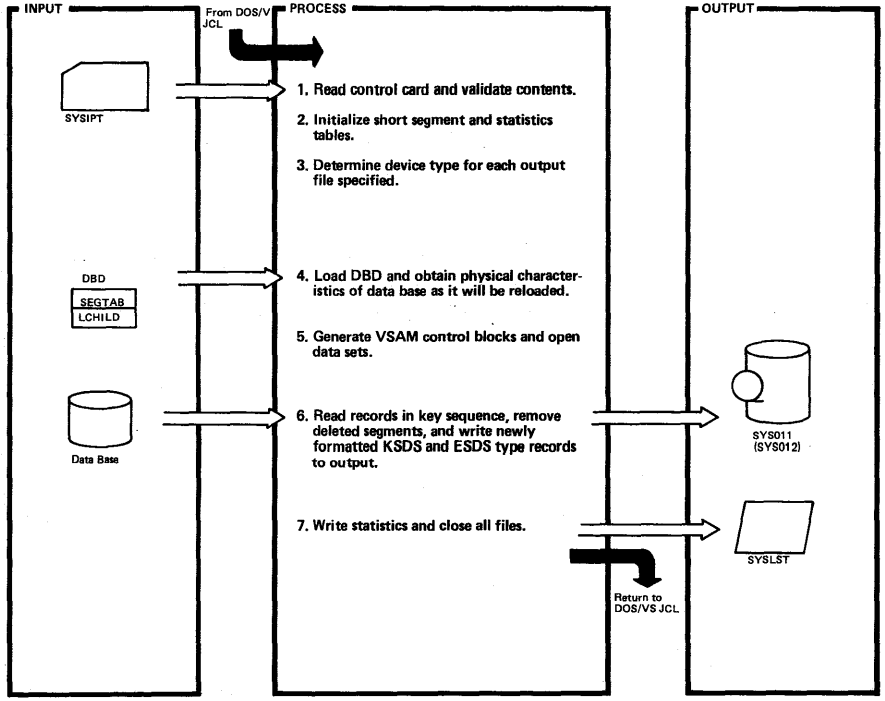
Figure 2-28.4. HD Backout (DLZRDBC0)



Extended Description	Routine	Label
1.		CHNMAIN
2.		HDFDLET HDNDLET HDUDLET
3.		HDUREPL
4.		HDFISRT HDNISRT HDUISRT
5.		CTRMAIN

Extended Description	Routine	Label

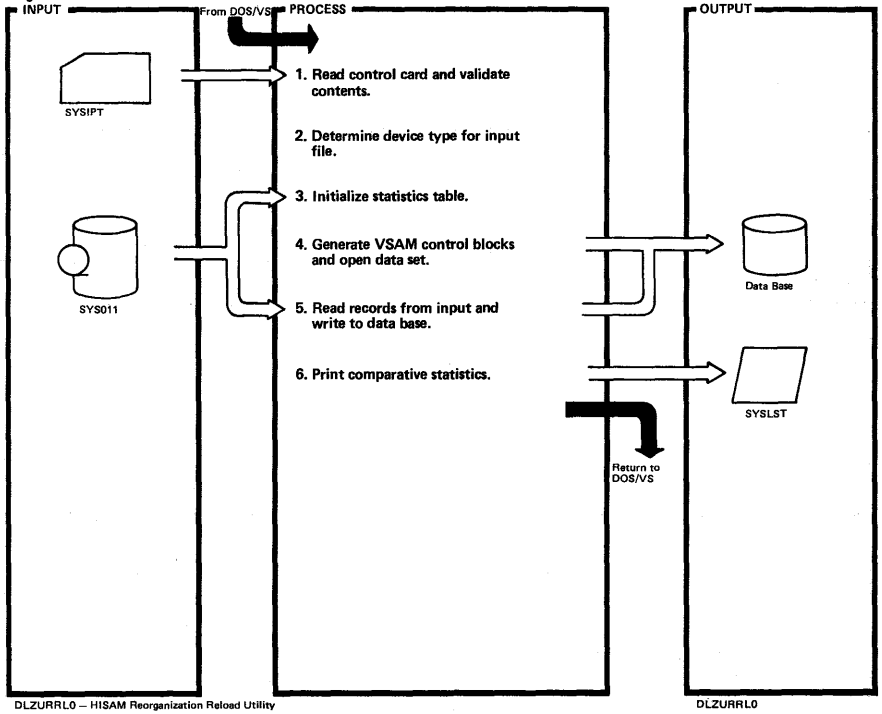
Figure 2-29. HS DB Unload



Extended Description	Routine	Label
1. Read and validate control statement. Write the following messages as needed: DLZ302I - Column 1 not R DLZ303I - Column 2 not 1 or 2 DLZ304I - DBD name not specified DLZ307I - Input filename not specified DLZ308I - Output filename not specified DLZ309I - Error(s) found in control statement DLZ310I - Image of erroneous control statement DLZ334I - Column 3 not N, R, U, or blank		
3. DLZDVCE macro obtains data from PUB. Device type may be TAPE or DASD.		
5. Issue GENCB for ACB, RPL, and EXLST. Open KSDS and ESDS unless ACCESS = SHISAM (KSDS only).		
6. Processing as follows: A. Read KSDS records in key sequence, bypass if deleted, ESDS records containing overflow dependent segments are read by RBA.		
B. Format work area like KSDS record with new attributes.		

Extended Description	Routine	Label
6. (con't) C. Move as many segments as will fit into KSDS work area, bypassing deleted segments. Calculate overflow RBA. Write image of KSDS to output.		
D. Format work area like ESDS record with new attributes.		
E. Move any dependent segments as will fit into ESDS work area, bypassing deleted segments. Calculate RBA for next record, if required. Write image of ESDS to output.		
7. Statistics also written to SYS011 to be used for comparative purposes during reload. Processing will continue if additional input cards.		

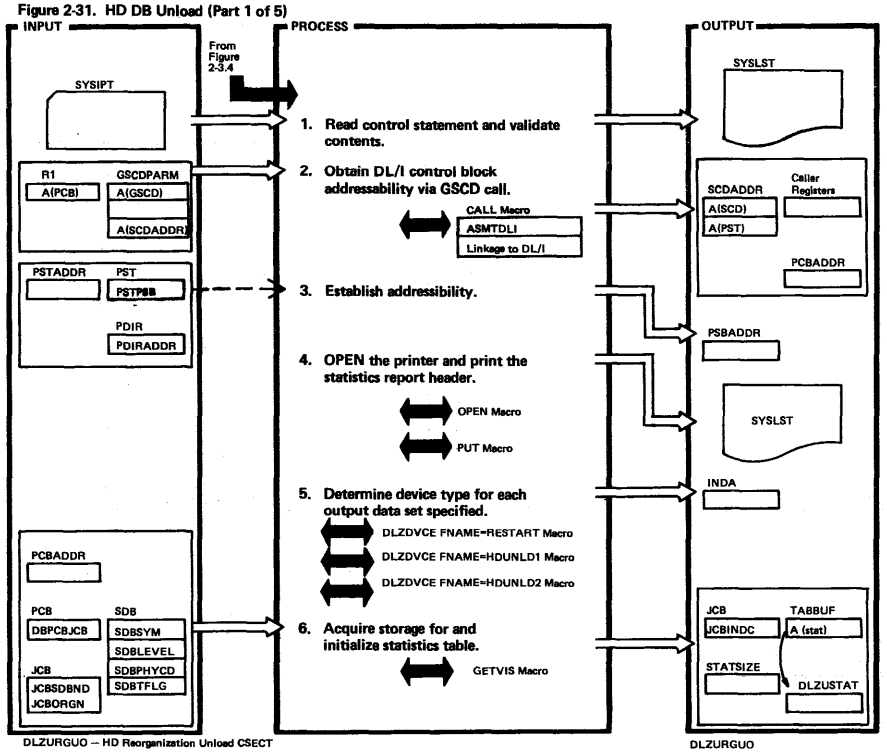
Figure 2-30. HS DB Reload



Extended Description	Routine	Label
1. Read and validate control statement. Write the following messages as needed: DLZ302I - Column 1 not L DLZ307I - Input filename not specified DLZ309I - Error(s) found in control statement DLZ310I - Image of erroneous control statement DLZ344I - Column 3 not N, R, U, or blank		
2. DLZDVCE macro obtains data from PUB. Device may be TAPE or DASD.		
3. The first record on the input file contains a statistics table initialized to zero. Included is the segment code and length for all segment types in the data base.		
4. Issue GENCB for ACB, RPL, and EXLST. Open KSDS and ESDS unless ACCESS = SHISAM (KSDS only).		
5. KSDS image records written to KSDS as key sequence records. ESDS image records written to ESDS as address sequence records.		

Extended Description	Routine	Label

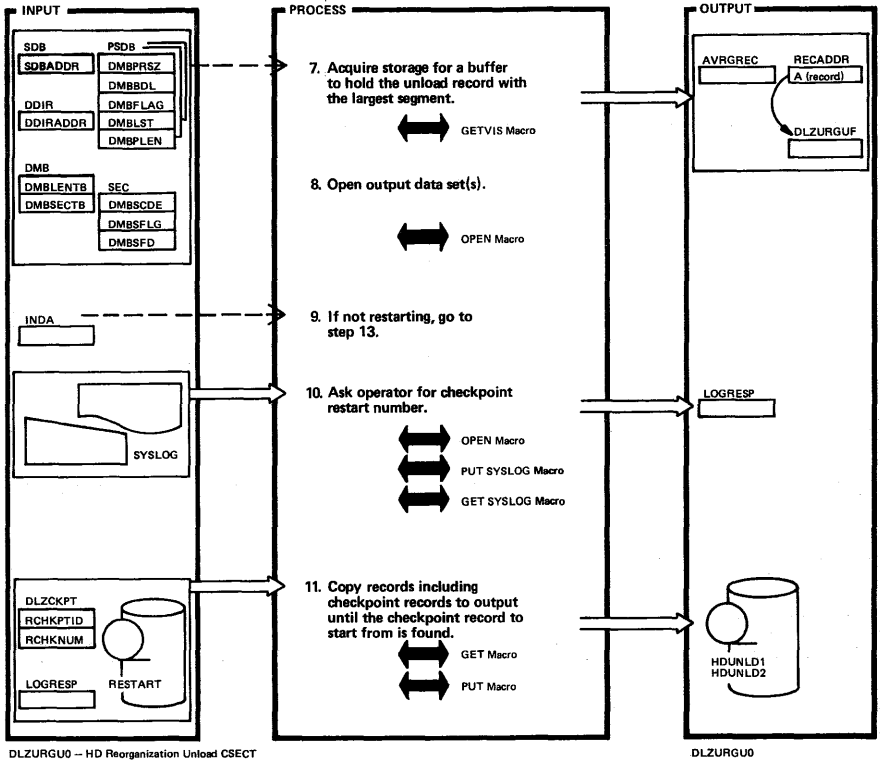
Figure 2-31. HD DB Unload (Part 1 of 5)



Extended Description	Routine	Label
1. Write the following messages, as needed: DLZ329I CHKPT parameter invalid DLZ359I Error - Not REW= or CHKP= DLZ368I Error - Not an N, R, or U DLZ386I Error - Duplicate control card		
2. Module identifier (DLZURGU0vrmp) is defined here. The GSCD call returns the SCD address +X'60' and the PST address in the call parameter I/O area (SCDADDR). The PCB address is passed to this module in R1 and stored at PCBADDR for later use.	DLZURGU0	DLZURGU0 BEGIN
4.		PUTHEAU
5. If restarting, set the restart in process indicator on at INDA.		PUBCHK1

Extended Description	Routine	Label
If HDUNLD1 is IGN or UA, write DLZ311I followed by DLZ347I, and then terminate. If HDUNLD2 is IGN or UA, write DLZ345I and continue processing.		
6.. Indicate to DL/I Retrieve that HD UNLOAD is running by setting the indicator (JCBHDULD) at JCBINDC. Macro DLZUSTAT contains the DSECT defining the format of a statistics table entry. The table contains the segment code and length for all segment types in the data base.		MAIN1

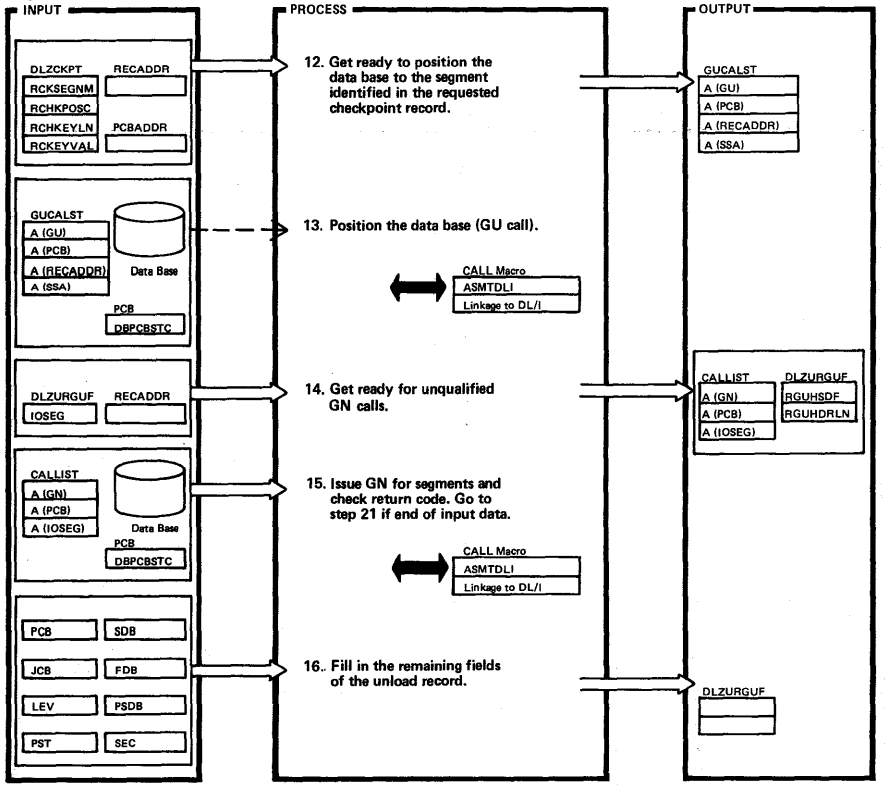
Figure 2-31. HD DB Unload (Part 2 of 5)



Extended Description	Routine	Label
7. Macro DLZURGUF contains the DSECT defining the format of an unload record.		STATEND
8.		TESTSECD
9.		SETOUT
10. Open the RESTART file and write message DLZ318A to SYSLOG requesting restart number and read response.		RESTRT0
11. Macro DLZCKPT contains the DSECT defining the format of a checkpoint record. Write message DLZ315I if end-of-file is reached on the restart data set without finding the requested checkpoint record.		RESTRT2A

Extended Description	Routine	Label

Figure 2-31. HD DB Unload (Part 3 of 5)



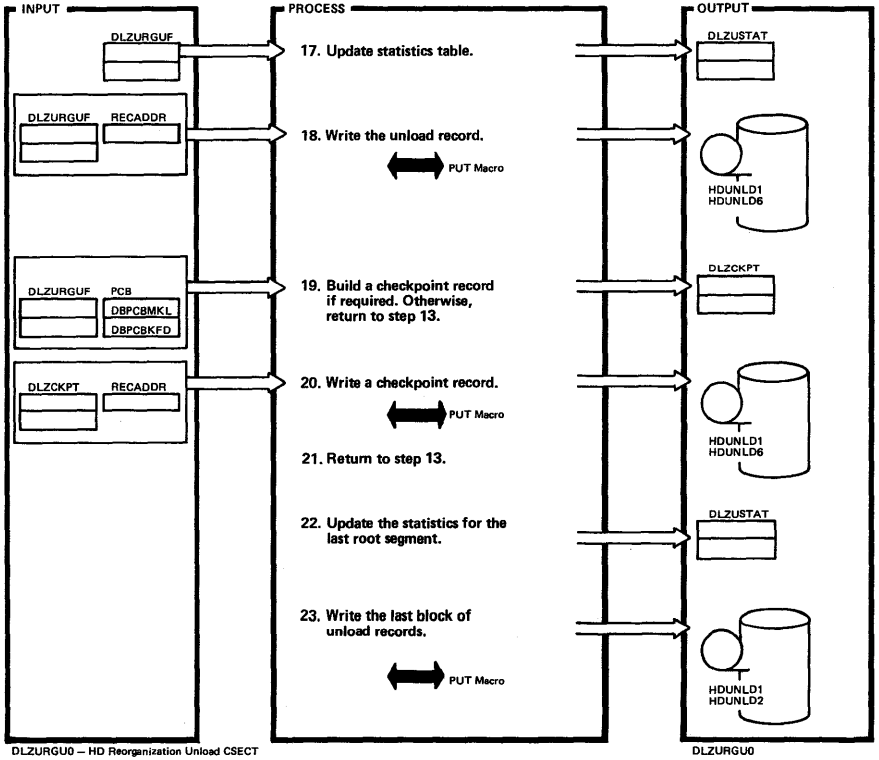
DLZURGU0 – HD Reorganization Unload CSECT

DLZURGU0

Extended Description	Routine	Label
12. If the RBN is available in the checkpoint record, the SSA will be "segname*T (rbn)" (HDAM or HIDAM). If the RBN is not available, a qualified key call is required. The GETVIS macro is used to get a work area to build the SSA for the call. The SSA will be "segname*C (key)". Following the call, the work area is freed.		RESTRT4
13. Write the following messages as needed: DLZ3011 – Open failure DLZ3481 – Unexpected status from return code DLZ3491 – Input I/O error DLZ3781 – Restart successful DLZ3791 – Restart failed DLZ3801 – Segment not found		

Extended Description	Routine	Label
14. IOSEG is the beginning of the variable length data field following the DL/I prefix information of the unload record.		RECREATE
15. Write the following messages as needed: DLZ3011 – Open error DLZ3481 – Unexpected return code DLZ3491 – I/O error		
16. Write message DLZ4001 for a sequence error.		HDRFILL

Figure 2-31. HD DB Unload (Part 4 of 5)



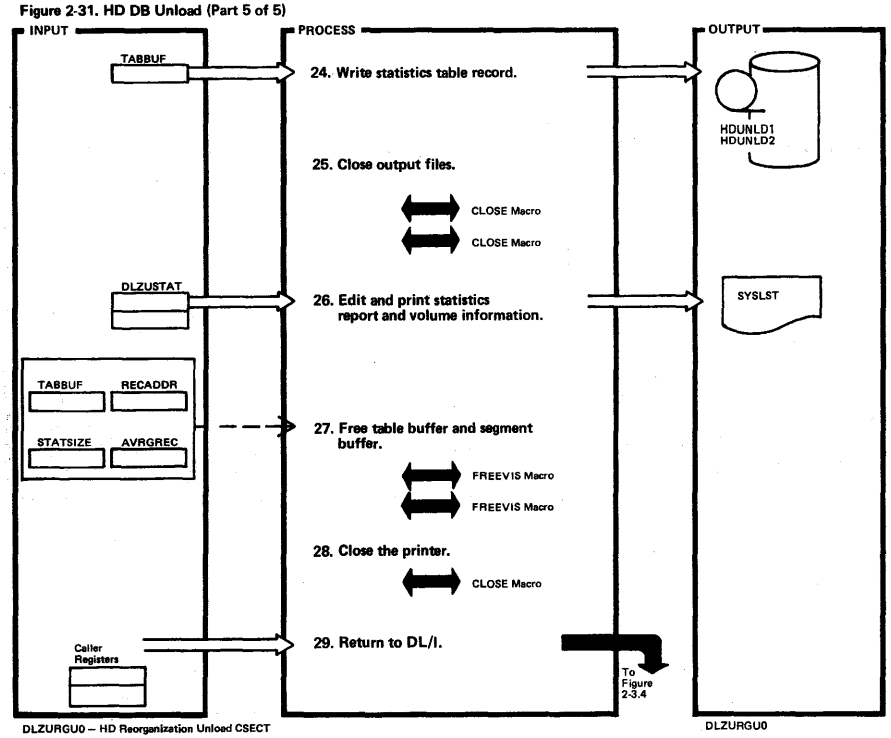
DLZURGU0 – HD Reorganization Unload CSECT

DLZURGU0

Extended Description	Routine	Label
17.		SETDLEN UPSTATS
18. The records are moved to the output block. When the block of records is full, the block is written.		WRITE3
19. Checkpoint records are written at the first root segment after every 5000 segments.		TSTCHK CHKPNT
20. Write message DLZ3811 every time a checkpoint is taken.		CHKPNT2
22.		EODINPUT LASTROOT

Extended Description	Routine	Label

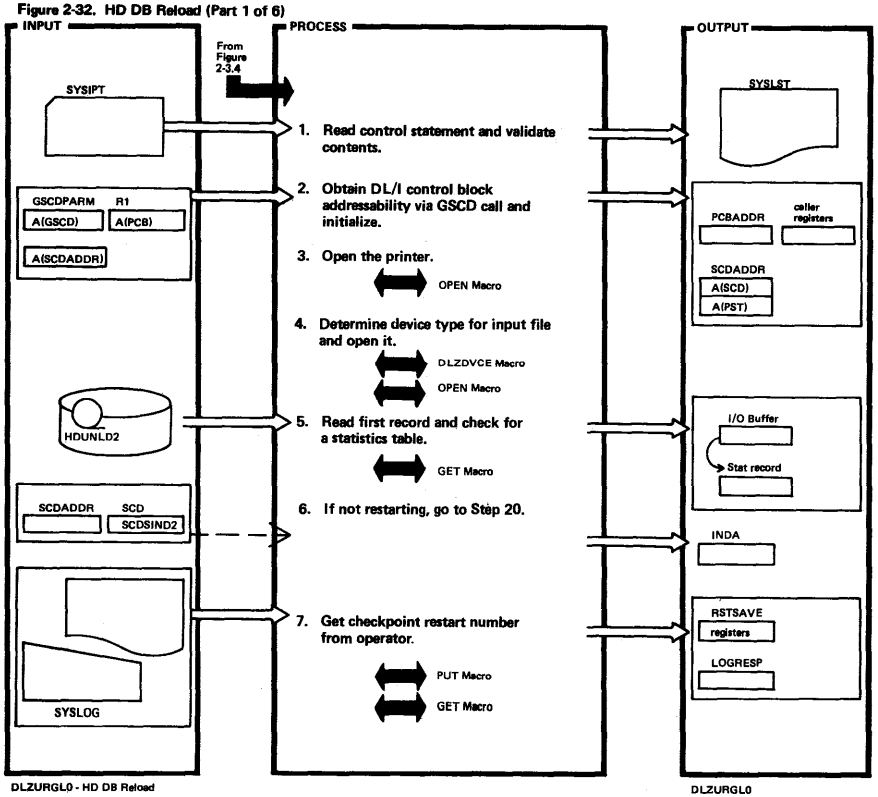
Figure 2-31. HD DB Unload (Part 5 of 5)



Extended Description	Routine	Label
24.		WRTTLST
25.		CLOSE2
26. Write message DLZ352I (no errors detected).		EDITSTAT
27. Write message DLZ392I for FREEVIS error.		STOPRUN
28.		STOPRUN2
29.		NODUMP

Extended Description	Routine	Label

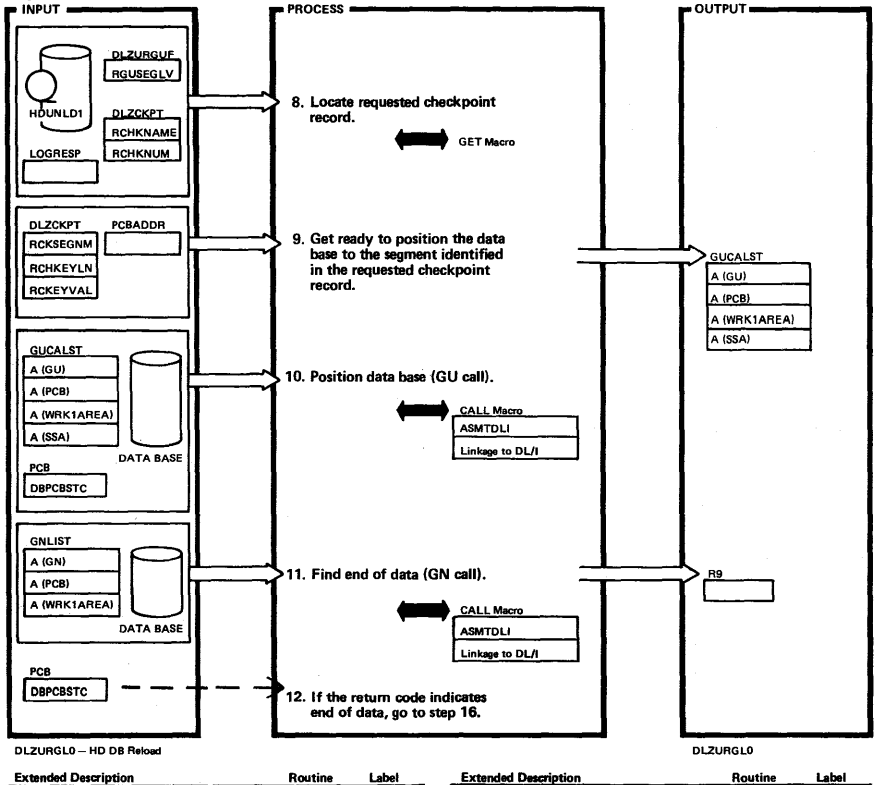
Figure 2-32. HD DB Reload (Part 1 of 6)



Extended Description	Routine	Label
1. Write the following messages, as needed: DLZ351I Error - Not REW DLZ368I Error - Not an N, R, or U DLZ386I Error - Duplicate control card		
2. Module identifier (DLZURGL0vrmp) is defined here The PCB address is passed to this module in R1 by DLZRRC00 and stored at PCBADDR for later use. The GSCD call returns to the SCD address +X'60' and the PST address in the call parameter I/O area (SCDADDR).	DLZURGL0	DLZURGL0 BEGIN
4. Write DLZ311I if HDUNLD1 is not assigned.		

Extended Description	Routine	Label
5. Issue DLZ389I if abnormal statistic table record.		STATINIT
6. If the HD DB Reload Utility Program fails, the reload restart capability allows you to restart from a checkpoint record. Before submitting the job for a reload restart, change the parameter card from ULU to ULR.		
7. Write DLZ318A message to SYSLOG requesting restart number and read response. The number of the last valid checkpoint record on the unloaded file is found in console message DLZ381I. Valid checkpoint numbers are decimal values between 1 and 9999.		RSTMESSG

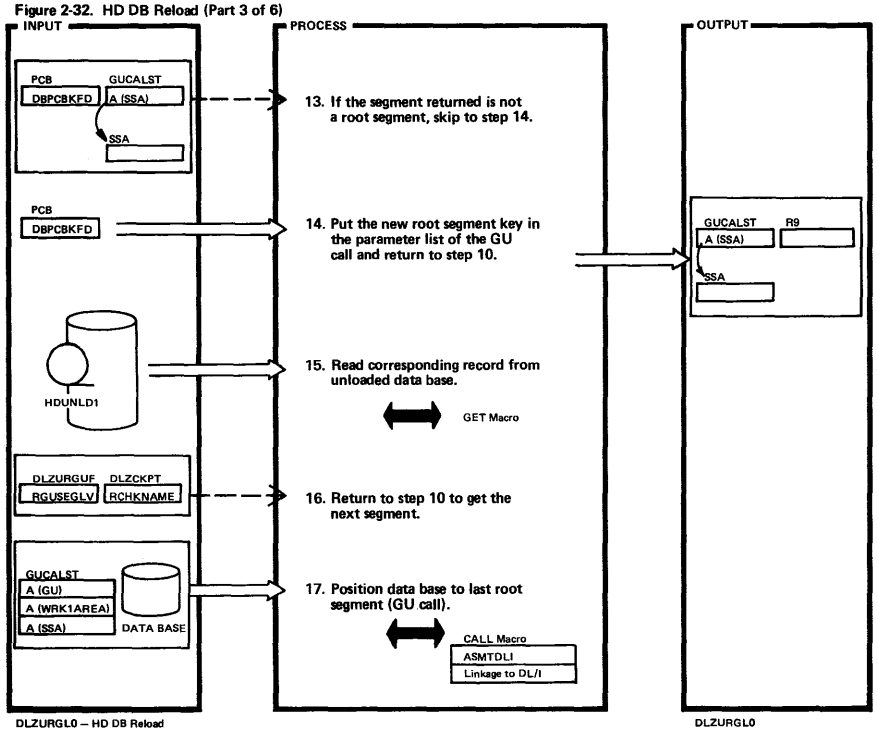
Figure 2-32. HD DB Reload (Part 2 of 6)



Extended Description	Routine	Label
8. Write DLZ370I if the checkpoint requested is less than the first checkpoint record encountered. Write DLZ315I if checkpoint record not found. Write DLZ381I checkpoint information message.		RSTGETLP
9. The SSA for the GU call is 'segname*C (key)', a qualified key call.		RSTPOSIT
10. The return code is checked. Write DLZ380I unable to position DB, checkpoint record not found.		CALLIT

Extended Description	Routine	Label
11. Now that the data base is positioned at the segment identified by the checkpoint record, issue GN calls to the end of the data base at the same time reading corresponding records from the unloaded data base in order to keep the two synchronized. A counter for GN calls (R9) is incremented by one for every GN call following a root segment. Write DLZ380I unable to position DB if invalid return code.		GNCALL

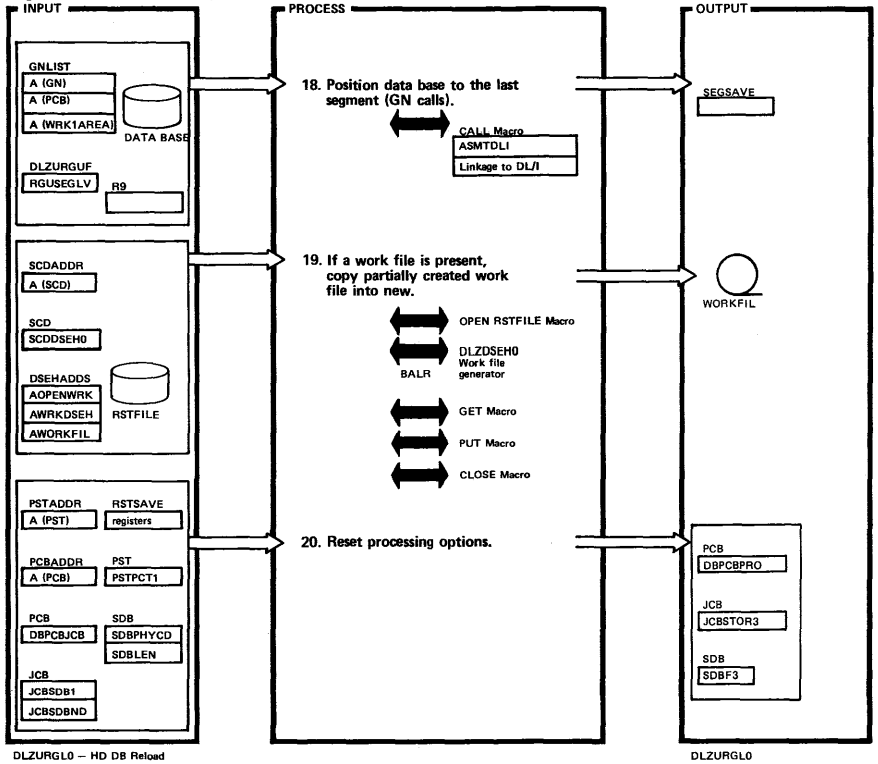
Figure 2-32. HD DB Reload (Part 3 of 6)



Extended Description	Routine	Label
13.		CHKKEY
14. Reset the counter for GN calls (R9) to zero.		
15.		UNLDGET
16. If the record returned from the unloaded data base is a checkpoint record, return to step 14 to get the next record in order to keep the partially reloaded data base and the input unloaded data base synchronized. Issue DLZ381I checkpoint information message.		
17.		REPOSN

Extended Description	Routine	Label

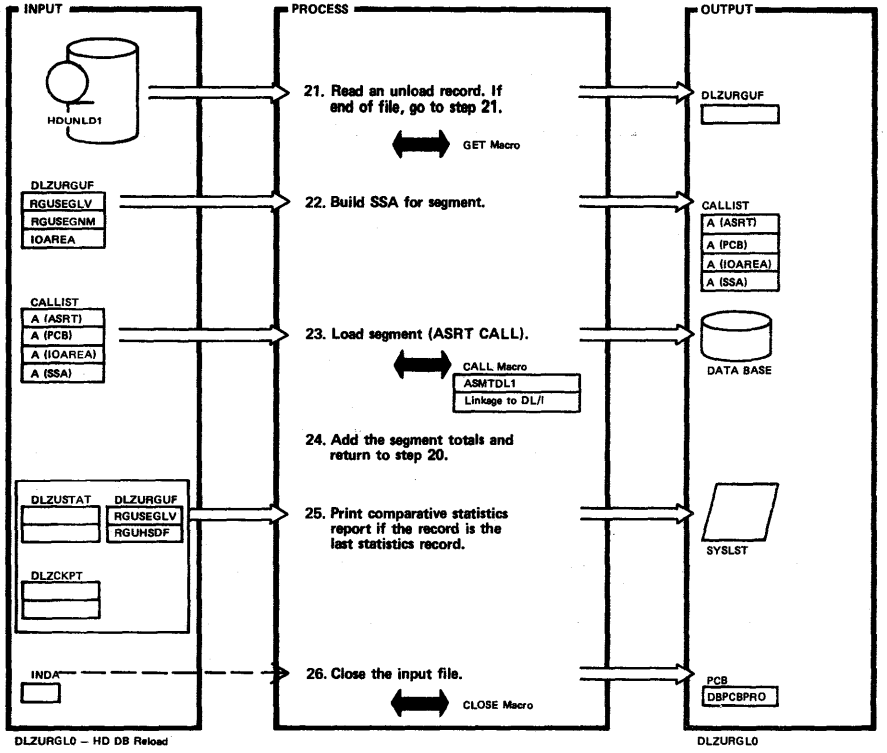
Figure 2-32. HD DB Reload (Part 4 of 6)



Extended Description	Routine	Label
18. Save the segment code of the last segment. The number of GN calls to make to get to the last segment is in R9.		GNLOOP
19. If a work file (for logical relationships or secondary indices) was being created during the reload, the partially created work file should be submitted as input to the restarted job assigned as SYS010 with a file name of RSTFILE. Write DLZ376I invalid device assignment.		SETWKFIL RSTOPEN
20. Now that Reload Restart processing is complete, set the processing options to indicate that a load is in process. Then resume processing as usual. If HISAM, set LS in the PCB. Otherwise, set L.		SETPROPT

Extended Description	Routine	Label
Save the address of the SDB for insert processing if it is for the last segment found in JCBSTOR3.		

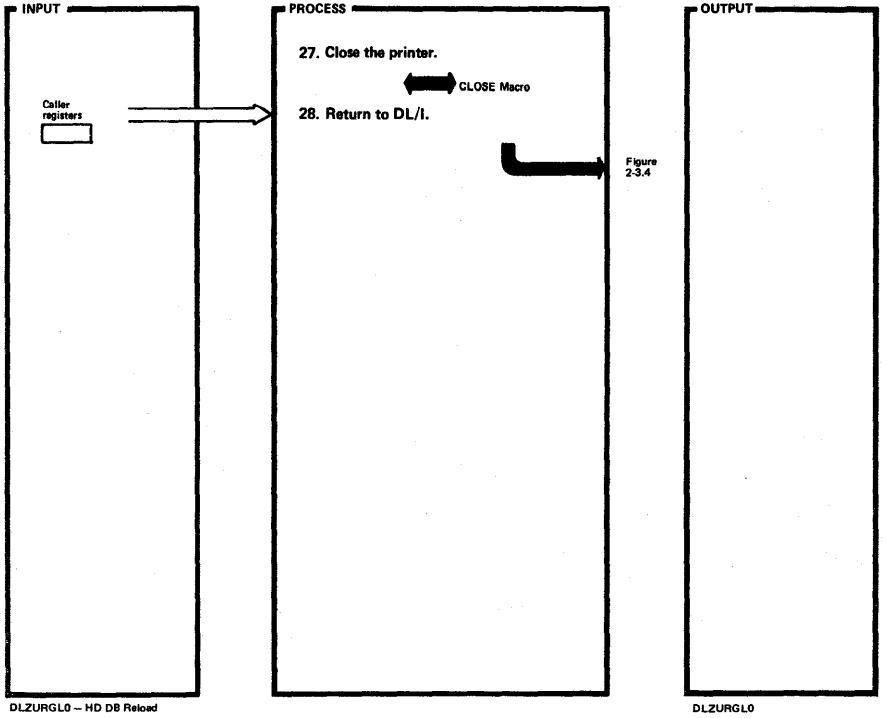
Figure 2-32. HD DB Reload (Part 5 of 6)



Extended Description	Routine	Label
21.		GETLOOP
22. IOAREA is the address of the data portion of the unload record DLZURGUF.		NOSTAT
23. Write DLZ301I OPEN error.		
Write DLZ319I IO error.		
Write DLZ348I invalid return code.		
24.		STATCOMP
25.		LASTCOMP
26. Write DLZ355I if reload is not okay, or DLZ379I if checkpoint restart is not okay.		EOD
Write DLZ354I if reload is okay, or DLZ378I if checkpoint restart is okay.		GOODRUN
If this was RESTART (ULR), reset the processing option in the PCB back to A.		

Extended Description	Routine	Label

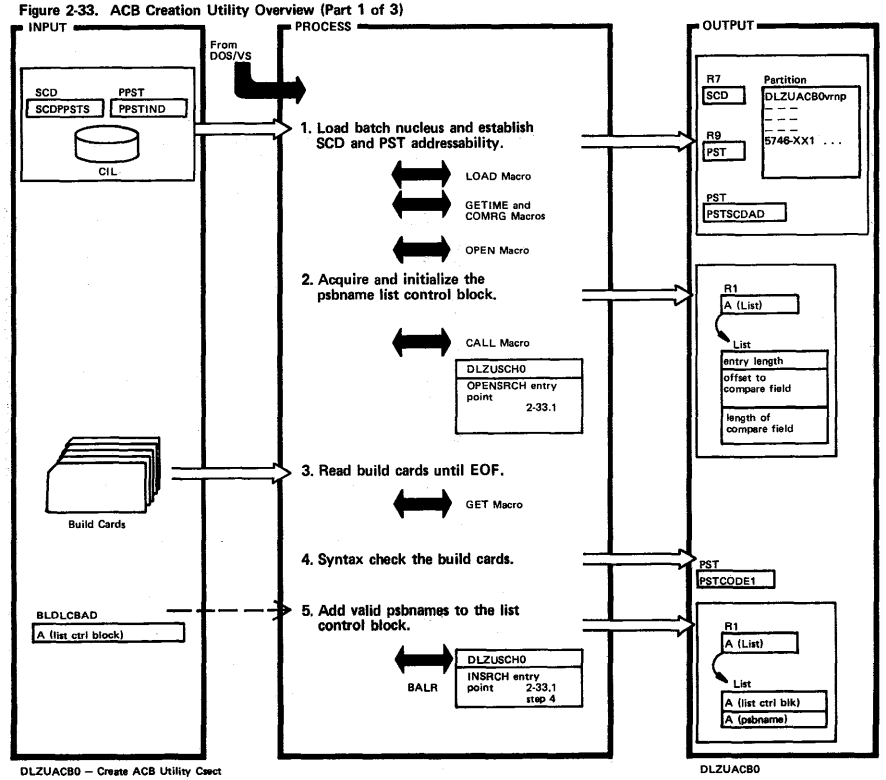
Figure 2-32. HD DB Reload (Part 6 of 6)



Extended Description	Routine	Label
27.		STOPRUN
28.		NODUMP

Extended Description	Routine	Label

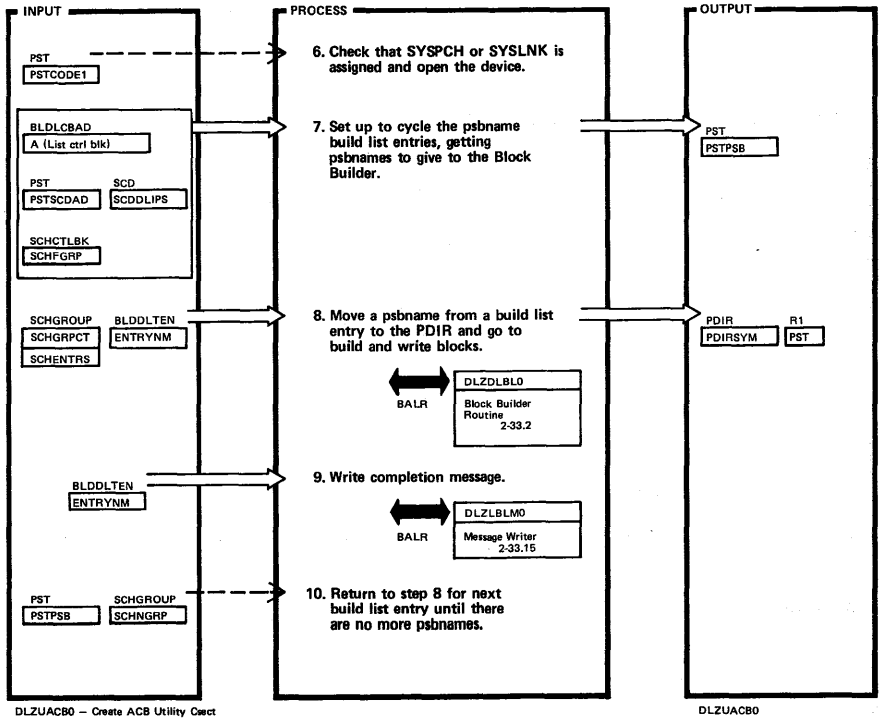
Figure 2-33. ACB Creation Utility Overview (Part 1 of 3)



Extended Description	Routine	Label
1. Module identifier (DLZUACB0vrnp) is defined here. The level format is vrnp; where 'v' is the version, 'r' is the release, 'n' is an additional identification number, and 'p' is the latest PTF number that has been applied. The time and date are required for the report heading to DLZUACB0 messages and control statements that are printed as this utility executes.	DLZUACB0	DLZUACB0
2. The parameter list for OPENSRCH is passed in R1. OPENSRCH returns the address of the list control block in R1, which is then put in the parameter list for INSRCH. Write DLZ905I if GETVIS error returned from OPENSRCH.		INITSORT
3.		READCARD GETT

Extended Description	Routine	Label
4. Write DLZ588I for an invalid delimeter, label, opcode, block type, operand, continuation, or invalid format. The output device is set at this time (SYSLINK or SYSPCH). If OUT=LINK was specified in the build card, SYSLNK is indicated in the PST.		
5. The psbnames from the PSB= operand are passed one at a time to INSRCH. The parameter list for INSRCH is passed in R1. Write DLZ905I if GETVIS error returned from INSRCH or DLZ571I warning message if a duplicate psbname is found.		MOVEIT

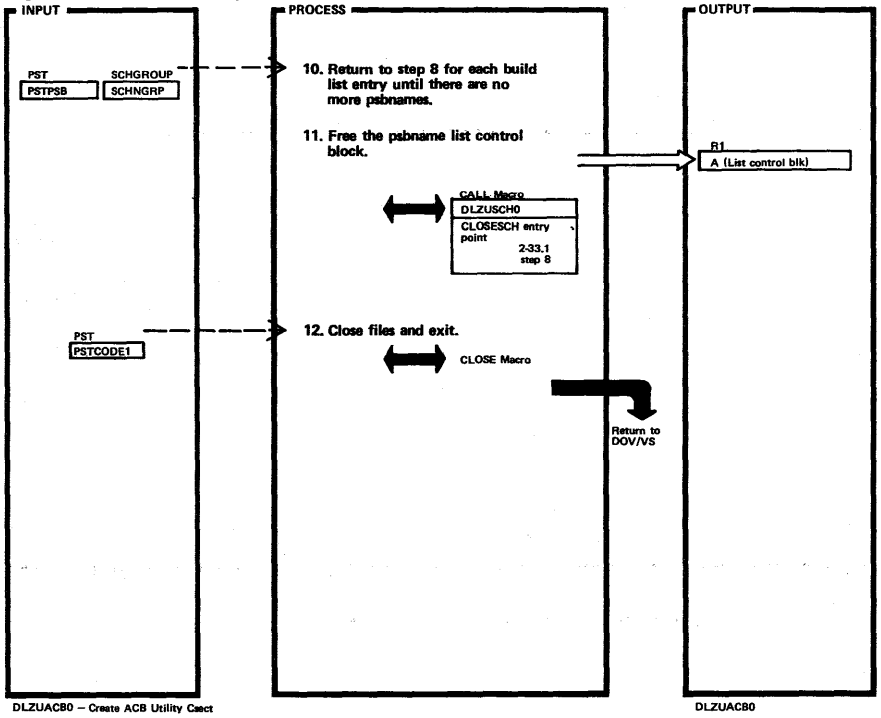
Figure 2-33. ACB Creation Utility Overview (Part 2 of 3)



Extended Description	Routine	Label
6.		CARDEOF OPEN
8. PSTERCOD is examined to see if any errors were posted by the Block Builder. If not zero, issue message DLZ587I to indicate the PSB will not be built and go to step 10 to attempt to build remaining blocks.		BLDGROUP CALLBB
9. The completion message is normal unless a non-zero return code is found from DLZDLBL0. Write DLZ589I to indicate processing has been completed for the PSB specified.		
10.		NXTPSB

Extended Description	Routine	Label

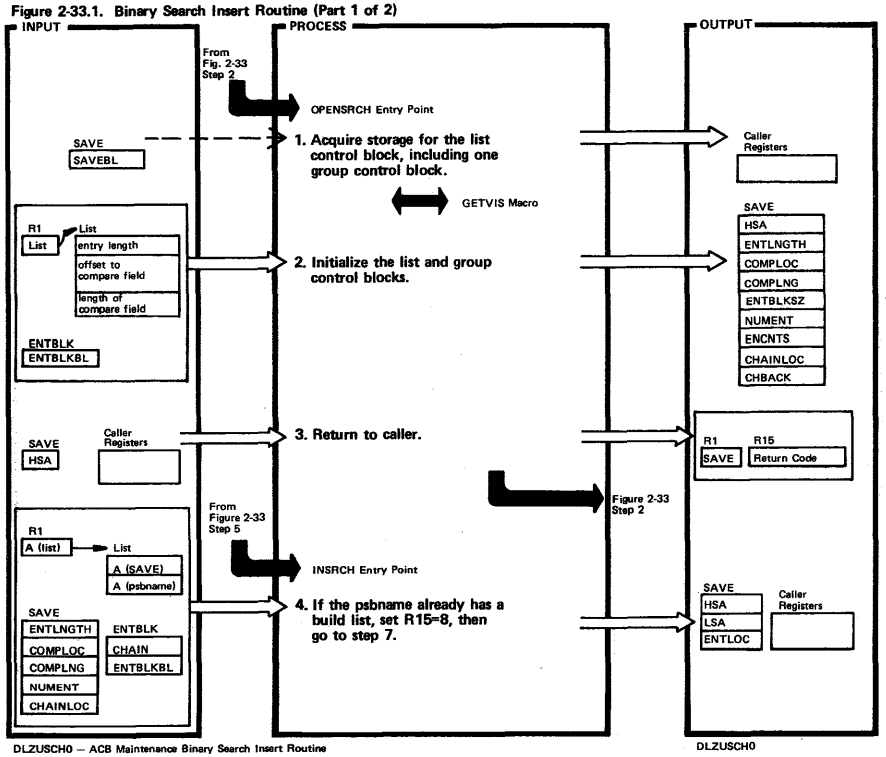
Figure 2-33. ACB Creation Utility Overview (Part 3 of 3)



Extended Description	Routine	Label
10.		NXTPSB
11.		BLDDONE
12.		CLOSE

Extended Description	Routine	Label

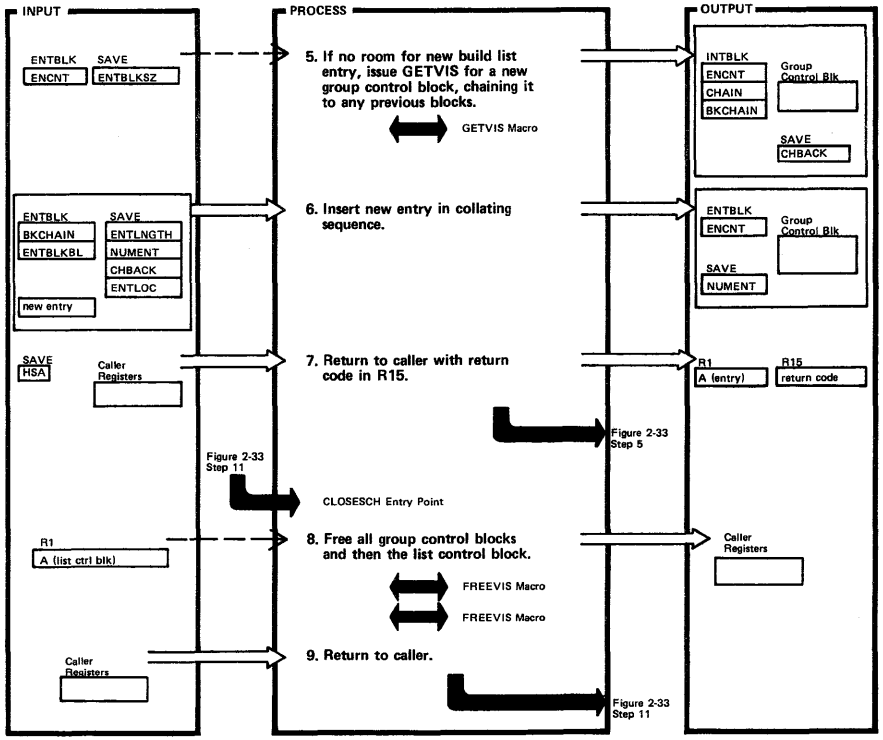
Figure 2-33.1. Binary Search Insert Routine (Part 1 of 2)



Extended Description	Routine	Label
1. Module identifier is defined here. Length acquired is X'80' bytes. One group control block will hold 16 build list entries.	DLZUSCH0	DLZUSCH0 OPENSRCH
2. Block now contains information needed to build a group control block. The first (or only) block is obtained before the first actual insert.		
3. The address of the created block is returned to the caller. Note: This routine is very generalized and could be used for other purposes, but it is only used by DLZUACB0 to build the psbname build list control block.		
4. Routine identifier (INSRCH. .) is defined here.	INSRCH	INSRCH INSRT1

Extended Description	Routine	Label

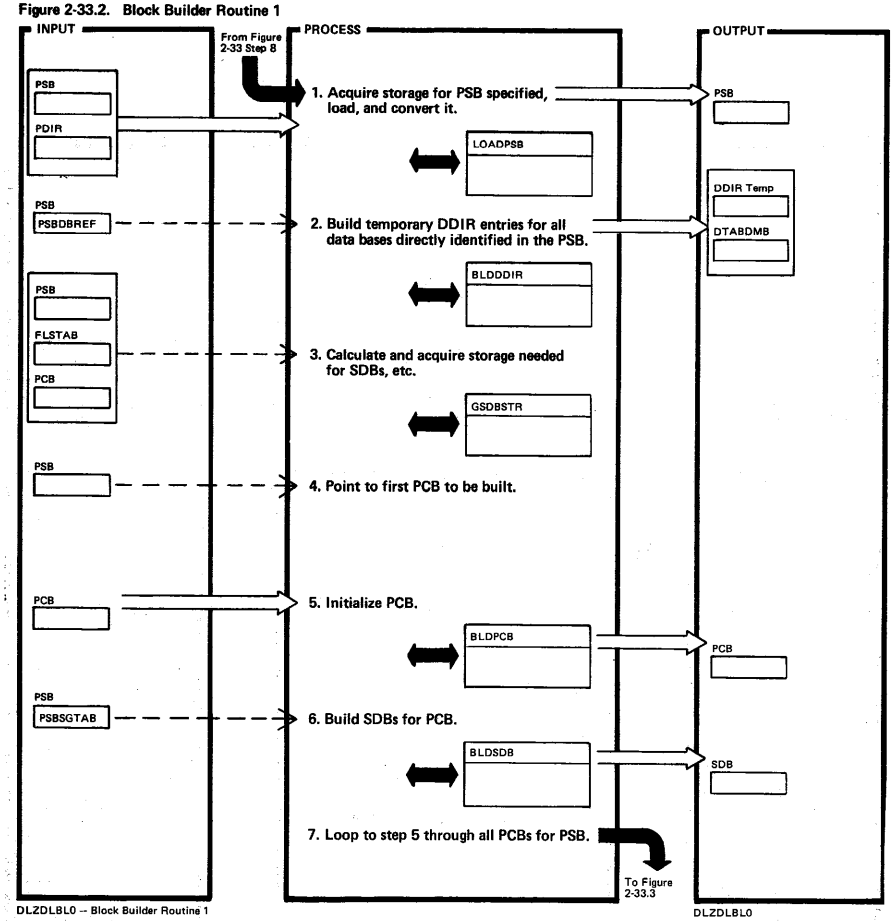
Figure 2-33.1. Binary Search Insert Routine (Part 2 of 2)



Extended Description	Routine	Label
5. There is enough room for 16 entries in a group control block.	INSRCH	NOTFND
6.		MVDK
7.		NOGO
8. Routine identifier (CLOSESCH) is defined here. All group control blocks can be found beginning from the list control block.	CLOSESCH	CLOSESCH

Extended Description	Routine	Label

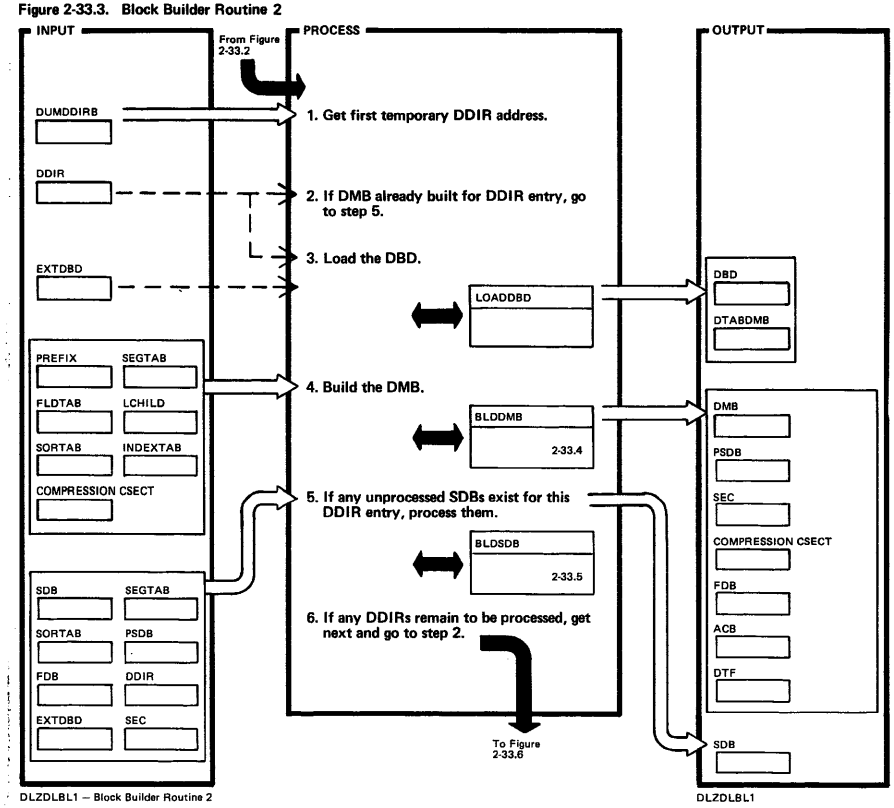
Figure 2-33.2. Block Builder Routine 1



Extended Description	Routine	Label
1.		PSBPASS
5.		PSBPASS1 BLDPCB
6. Only those SDBs for segments directly referenced by SENFLDs are built at this time. Generated SDBs will be built in DLZDLBL1.		PSBPASS3

Extended Description	Routine	Label

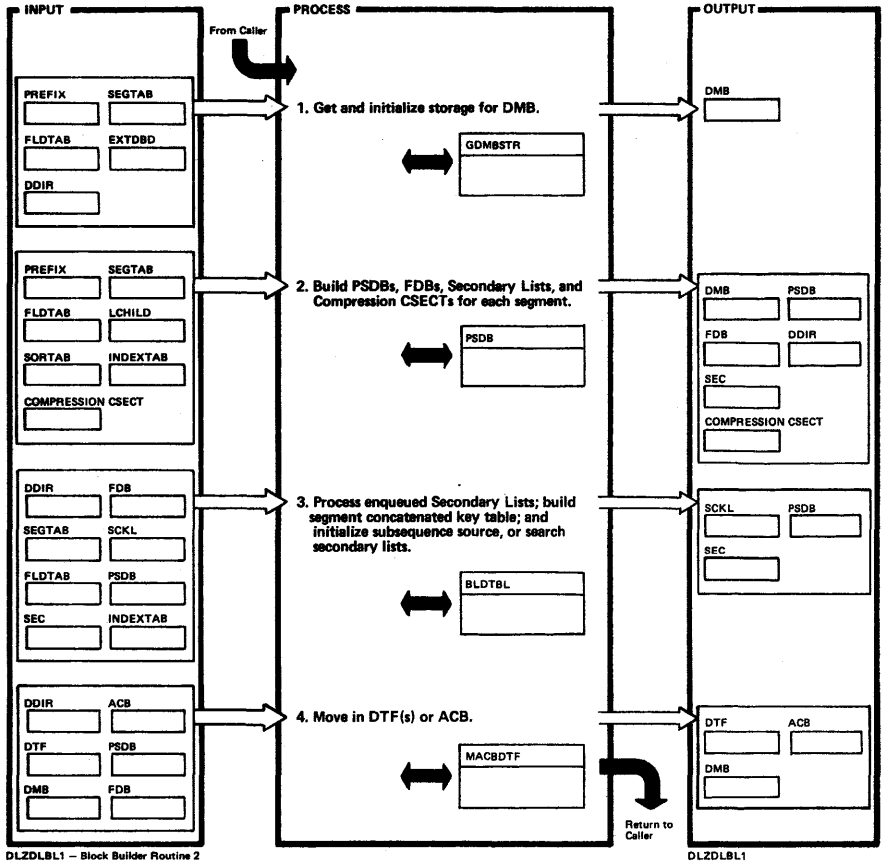
Figure 2-33.3. Block Builder Routine 2



Extended Description	Routine	Label
2. The DBD address is reset from the DMB Name Table.		DMBPASS1
3. Any DBDs referenced by this DBD are added to the dummy DDIR list.		DMBPASS2
4. A. If the DBD is for an index data base, the DDIR entry for the target data base is located and processed first.		DMBPASS3
B. No DMB is built for logical DBDs.		
5. A. SDBs pointing to VLC or logical segments are reset to point to the physical segment PSDB.		DMBPASS8
B. All generated SDBs are built here.		
6. If additional SDBs were chained to a prior DDIR during step 5, return to that DDIR, and go to step 2.		DMBPASS9

Extended Description	Routine	Label

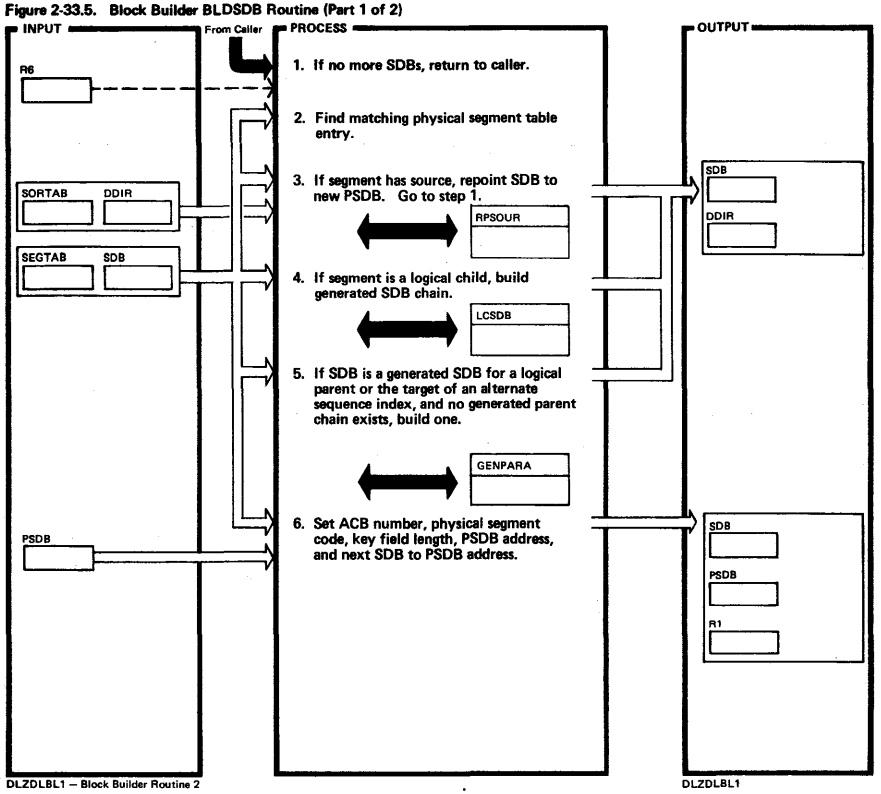
Figure 2-33.4. Block Builder BLDDMB Routine



Extended Description	Routine	Label
1.		BLDDMB
2. Until the concatenated key table is built, all secondary lists are enqueued on the DDIR and will be built during step 3.		BLDDMB1
3.		BLDDMB4

Extended Description	Routine	Label

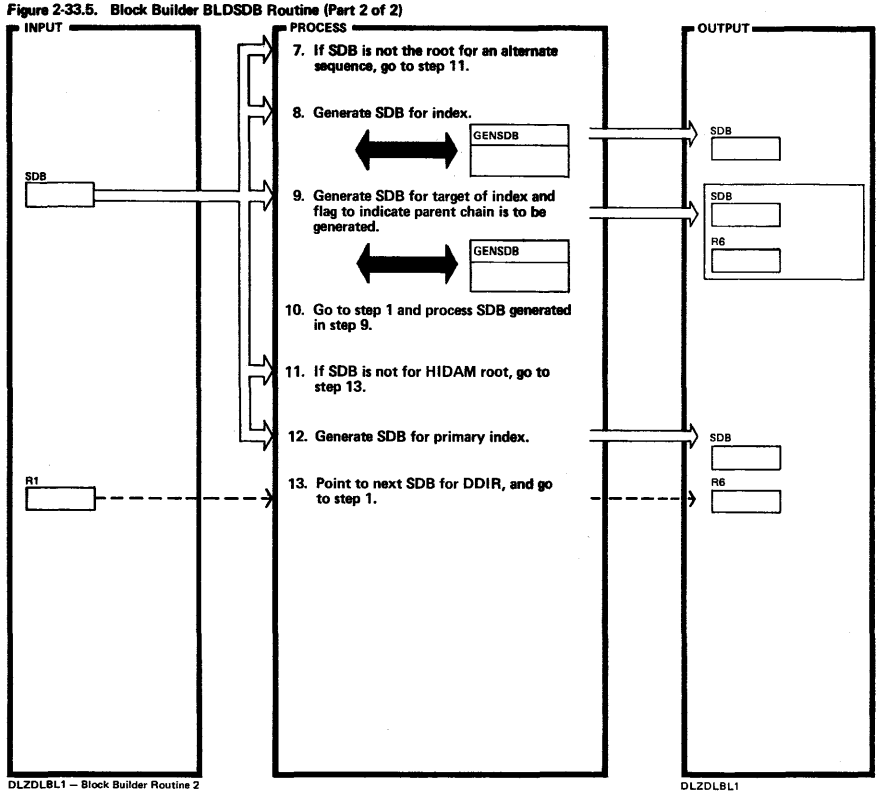
Figure 2-33.5. Block Builder BLDSDB Routine (Part 1 of 2)



Extended Description	Routine	Label
2. Output message DLZ912I if not found.		BLDSDB1
3. If source is in another DBD, the SDB to be processed is the next one for this DBD. Otherwise, process the original with the new segment name.		BLDSDB3
4. If segment is a normal logical child, the logical parent SDB is generated and flagged to cause generation of the parent chain when the LP is processed. All generated SDBs are chained to the DDIRs for the related data bases.		BLDSDB7
5.		BLDSDB8
6.		BLDSDB9

Extended Description	Routine	Label

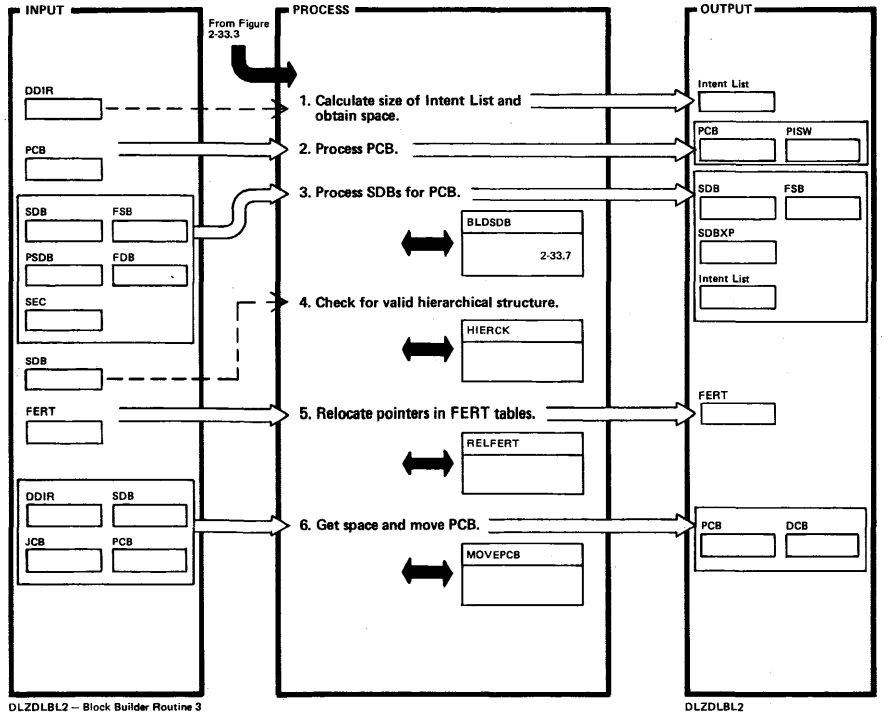
Figure 2-33.5. Block Builder BLDSDB Routine (Part 2 of 2)



Extended Description	Routine	Label
7.		BLDSDBF
11.		BLDSDBG
13.		BLDSDBI

Extended Description	Routine	Label

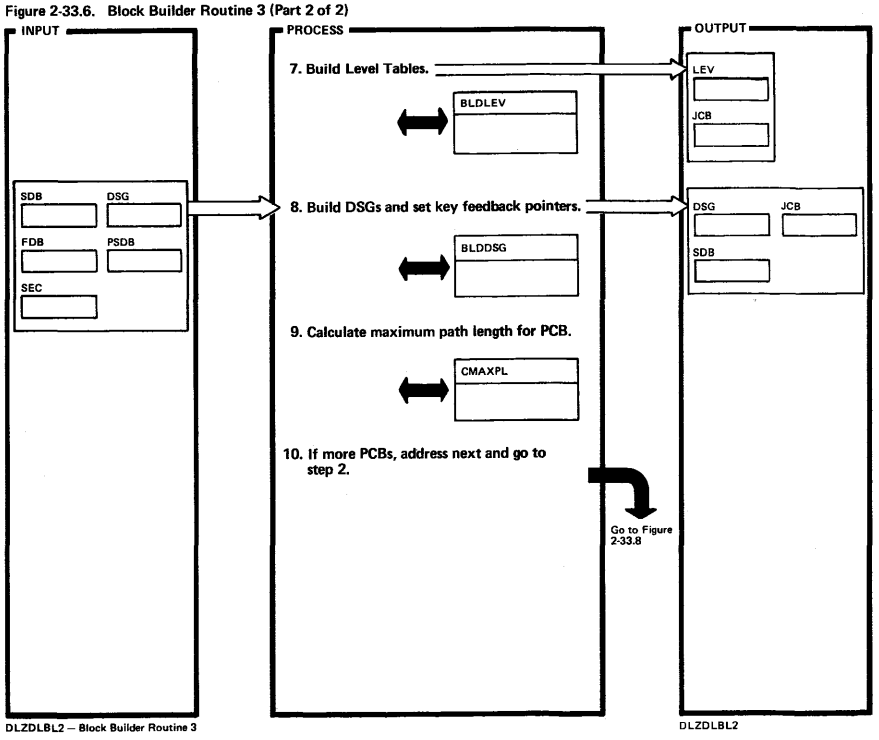
Figure 2-33.6. Block Builder Routine 3 (Part 1 of 2)



Extended Description	Routine	Label
1.		GTILIST
2.		BLDPCB
3. The SDB expansion and FSBs for any field level sensitivity are built here. The parentage flags in the SDB are set, the Intent List entry is built, and any propagation is done. The index SDB, if any, is initialized.		BLDPCB4
6. Space is allocated for the PCB, JCB, key feedback area, DSG, and Level Table.		BLDPCB5

Extended Description	Routine	Label

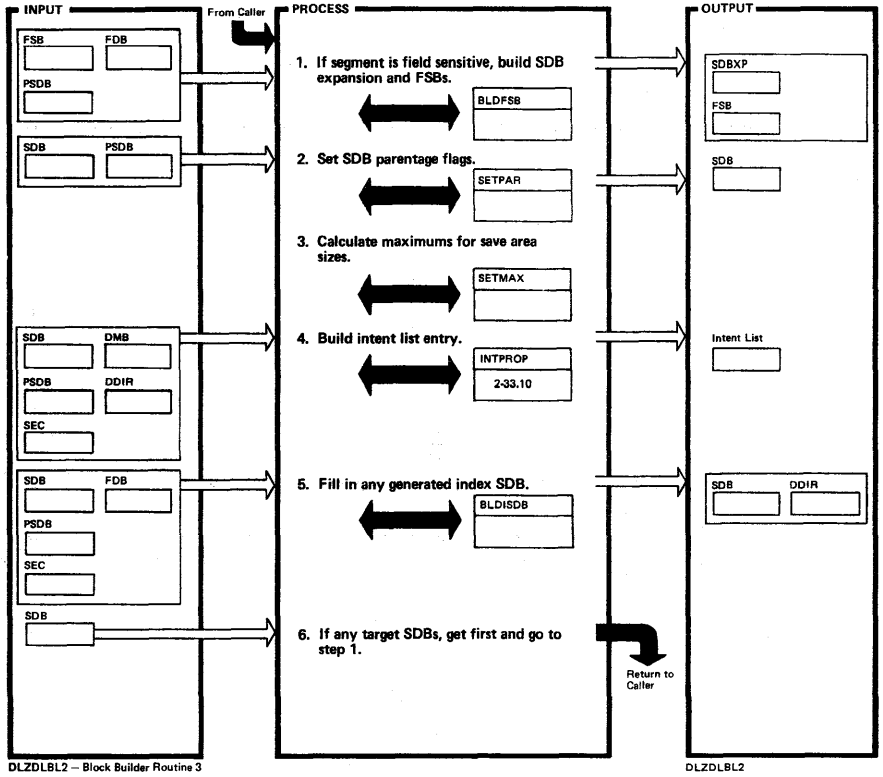
Figure 2-33.6. Block Builder Routine 3 (Part 2 of 2)



Extended Description	Routine	Label

Extended Description	Routine	Label

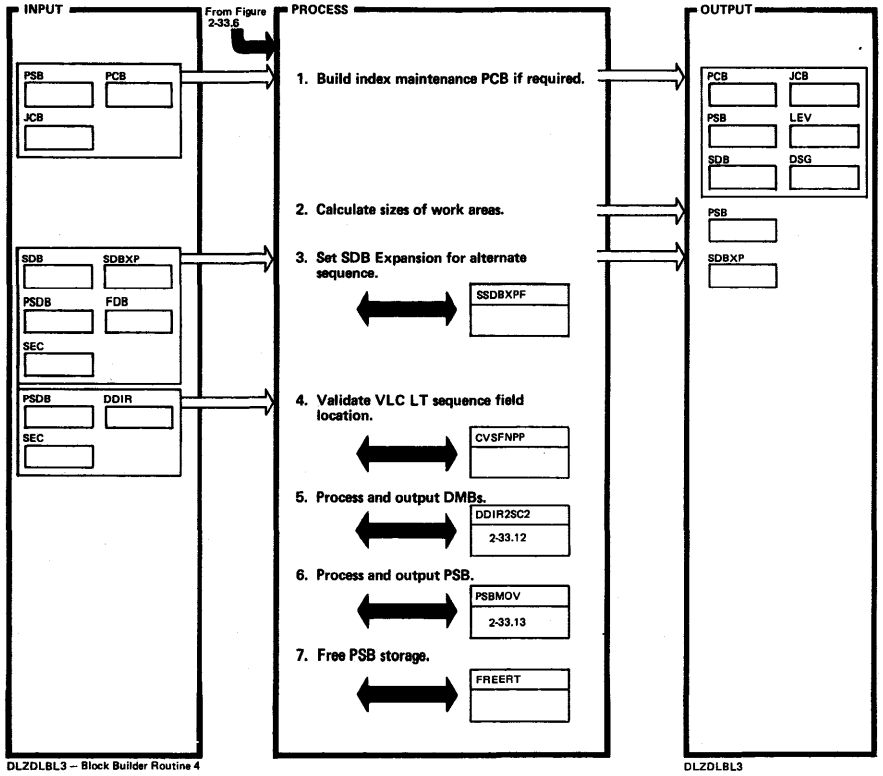
Figure 2-33.7. **Block Builder BLDSDB Routine**



Extended Description	Routine	Label
1.		BLDSDB2
2.		BLDSDB3
3. Maximums set are: a. Maximum segment length in either physical or user's view. b. Maximum concatenated key length. c. Maximum concatenated segment length. d. Longest segment at this level and path sensitive.		
4. a. Output message DLZ9091 if PROCOPT changed. b. All intent propagation is done here.		INTENTL
5. Fill in the generated index SDB for HIDAM primary indexes or alternate sequence.		BLDSDB4

Extended Description	Routine	Label

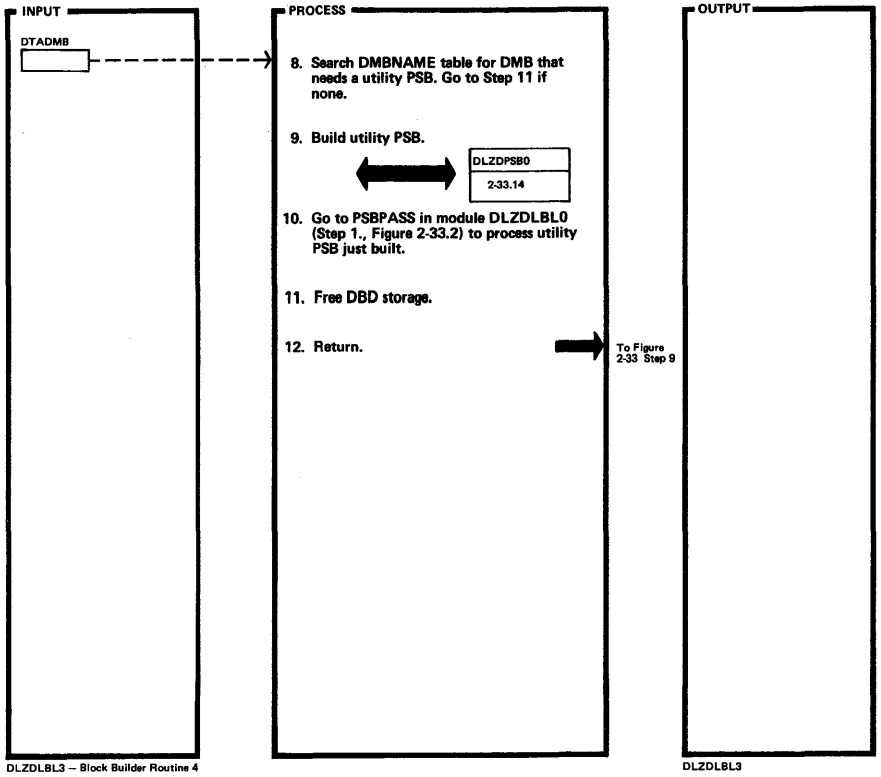
Figure 2-33.8. Block Builder Routine 4 (Part 1 of 2)



Extended Description	Routine	Label
1.		BLDEND
2. Index work area, Index I/O area, segment compression work area, and I/O work area		CALWAS
5.		DDIR2SC2
6.		PSBMOV
7.		FREEPSB

Extended Description	Routine	Label

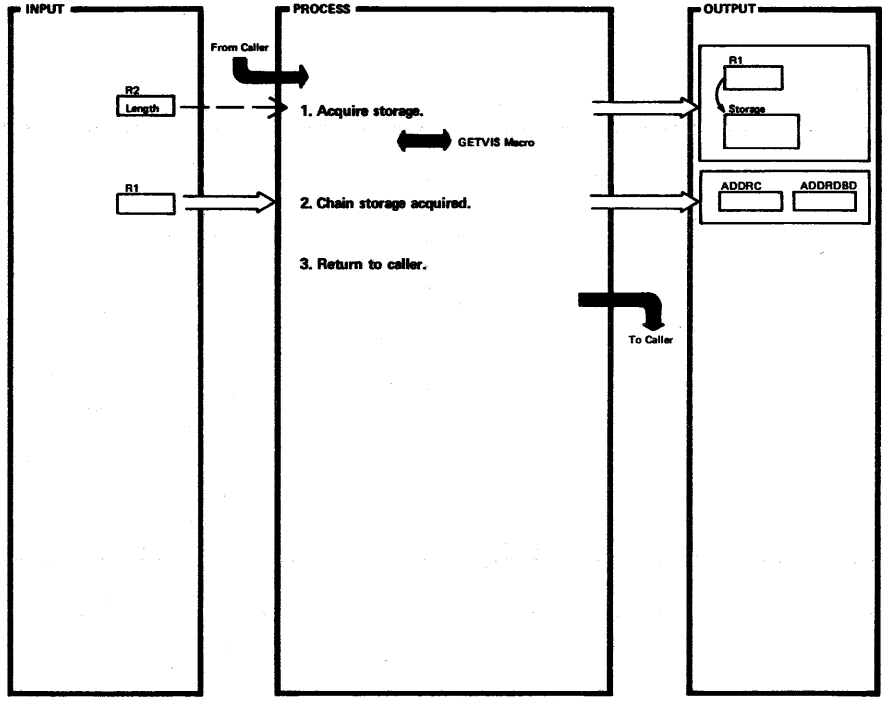
Figure 2-33.8. Block Builder Routine 4 (Part 2 of 2)



Extended Description	Routine	Label
8. Utility PSB is built for every HISAM, HIDAM, HDAM, and secondary index DMB just outputted.		UTILPSB
9. The dmbname is moved to the psbname location and a suffix 'u' added. The no utility PSB required indicator is turned on at DTABFLAG so we don't try to build another utility PSB for this DBD the next time around. Return to Step 9 to build the PSB. The output of DLZDPSB0 is like PSBGEN output.		
11.		FREEDBD
12. PSTERLOD = 0 if OK and non-zero if not.		RETURN

Extended Description	Routine	Label

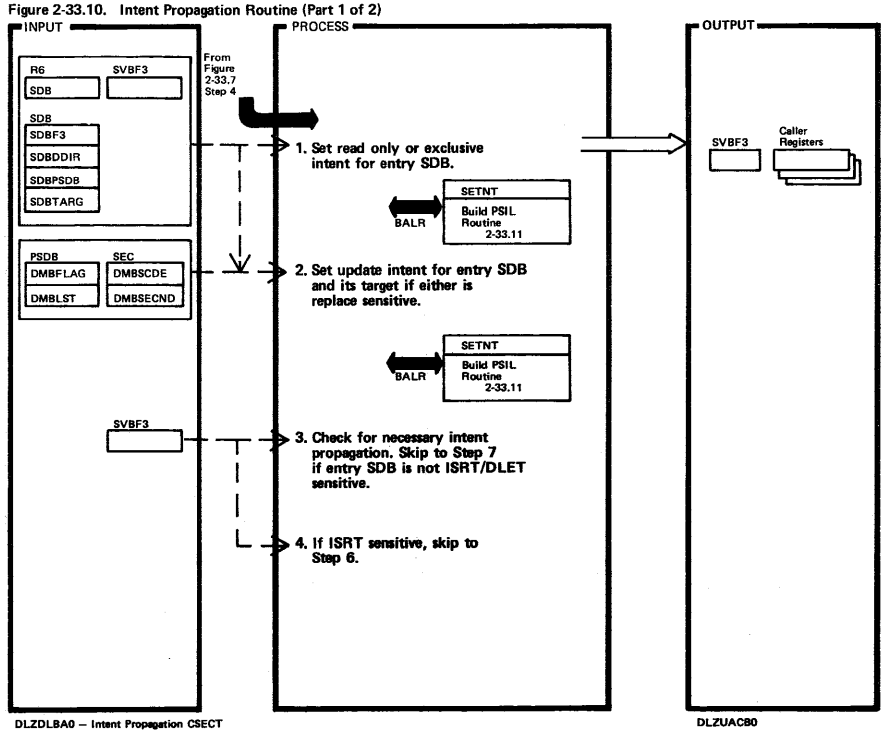
Figure 2-33.9. Acquire Storage Routine



Extended Description	Routine	Label
1. Write message DLZ905I if GETVIS space is not available.		GETSTOR
2. If this is a storage request for a DBD, the storage is chained off at ADDRDBD rather than ADDRC.		GETSTOR1

Extended Description	Routine	Label

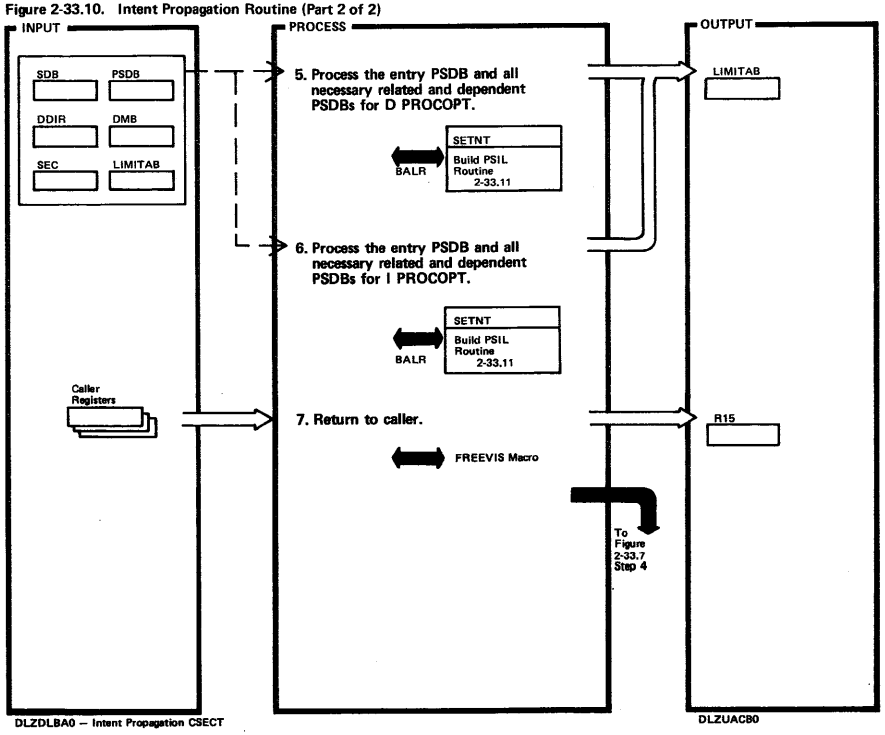
Figure 2-33.10. Intent Propagation Routine (Part 1 of 2)



Extended Description	Routine	Label
1. This routine propagates intent to all PSDBs related or dependent on the PSDB for the entry SDB. Only SDBs that are built directly from SENSEG statements in the associated PSB are passed to this routine. For a process option (PROCOPT) of G, the entry SDB and all targets are set to exclusive. For E PROCOPT, the entry SDB and its immediate target are checked for key sensitivity. If the SDB is data sensitive, the intent is set to exclusive. Otherwise, no intent is set.		INTPROP
2.		SETREPL
3.		PROPTYPE
4.		UPTYPE

Extended Description	Routine	Label

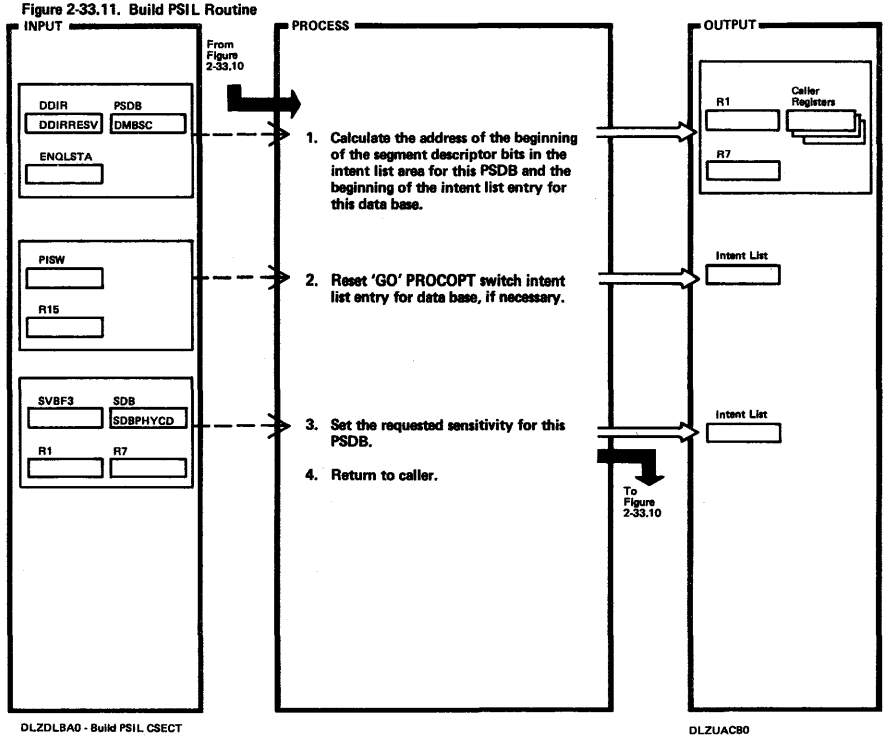
Figure 2-33.10. Intent Propagation Routine (Part 2 of 2)



Extended Description	Routine	Label
5. Any intent set will be update type. Storage is acquired for a limit table and is constructed with PSDB addresses. These addresses show children of the entry PSDB, any necessary higher related PSDBs, any index relationships, any logical children, the logical parent, physical parent, and physical pair of the entry PSDB. After the table is constructed, each entry is passed to SETNT to set update intent and the table area is freed. Exclusive intent is propagated along with delete.		DLETPA
6. Any intent set will be update type. A limit table is constructed with addresses of those PSDBs that will be passed to SETNT. It is also determined if the entry SDB can insert its logical parent or physical parent as a result of a concatenated segment definition. If it can, the logical parent or physical parent is processed in the same manner as the entry PSDB. The physical pair is also processed if one exists.		UPISRT UPCHKPP

Extended Description	Routine	Label
After processing the LIMITAB entries built, the area is freed. 7. Set the return code and make sure all the limit tables are freed.		INTRETO

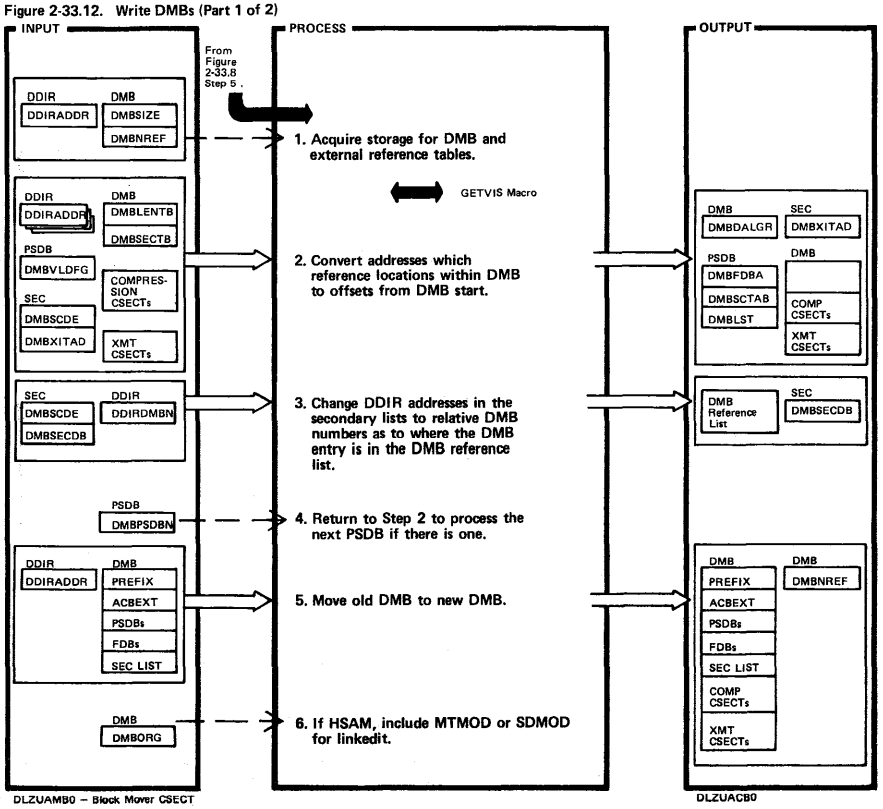
Figure 2-33.11. Build PSIL Routine



Extended Description	Routine	Label
1.	DLZDLBA0	SETNT
3.		ENQI

Extended Description	Routine	Label

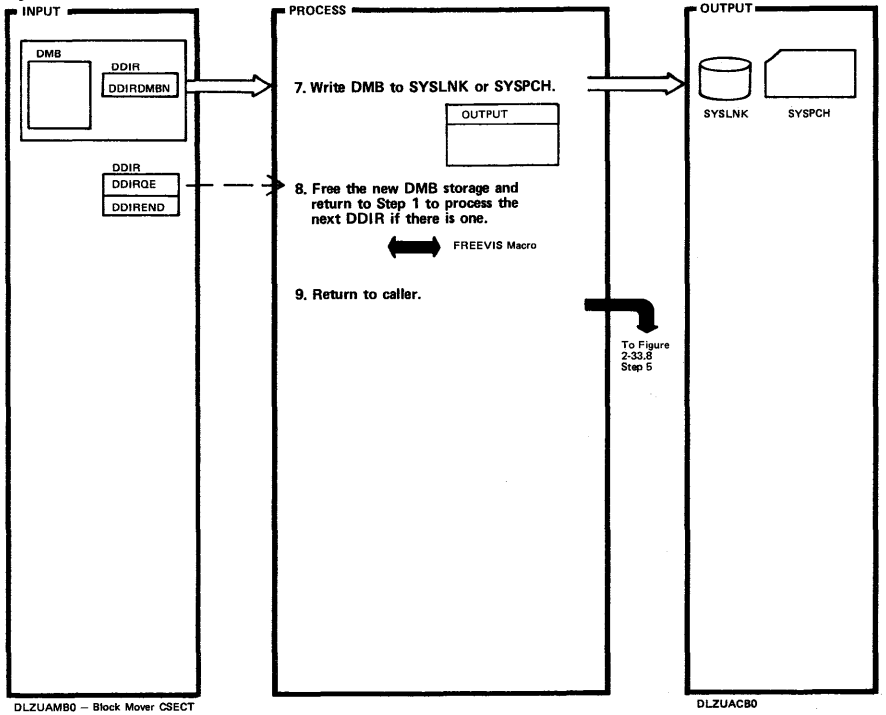
Figure 2-33.12. Write DMBs (Part 1 of 2)



Extended Description	Routine	Label
1. Process all DMBs referenced in DDIR entries unless already built or LOGICAL. Write message DLZ905I for GETVIS error.		DDIR2SC2
2. The compression CSECTs and index maintenance CSECTs are moved to the new DMB. Write message DLZ570I for an invalid SEC list code found in a DMB.		DMBREL
3. The reference list is the last part of the DMB. If the DMB is not in the list, it is added. Repeat this step for each SEC.		SECREL
4.		PSDBREL2
5. Any addresses which do not fall within the DMB are set to zero.		DMBOUT

Extended Description	Routine	Label
6. The names in the relocatable library of the required modules are DLZTAPE or DLZDISKI and DLZDISKO.		DMBOUT1

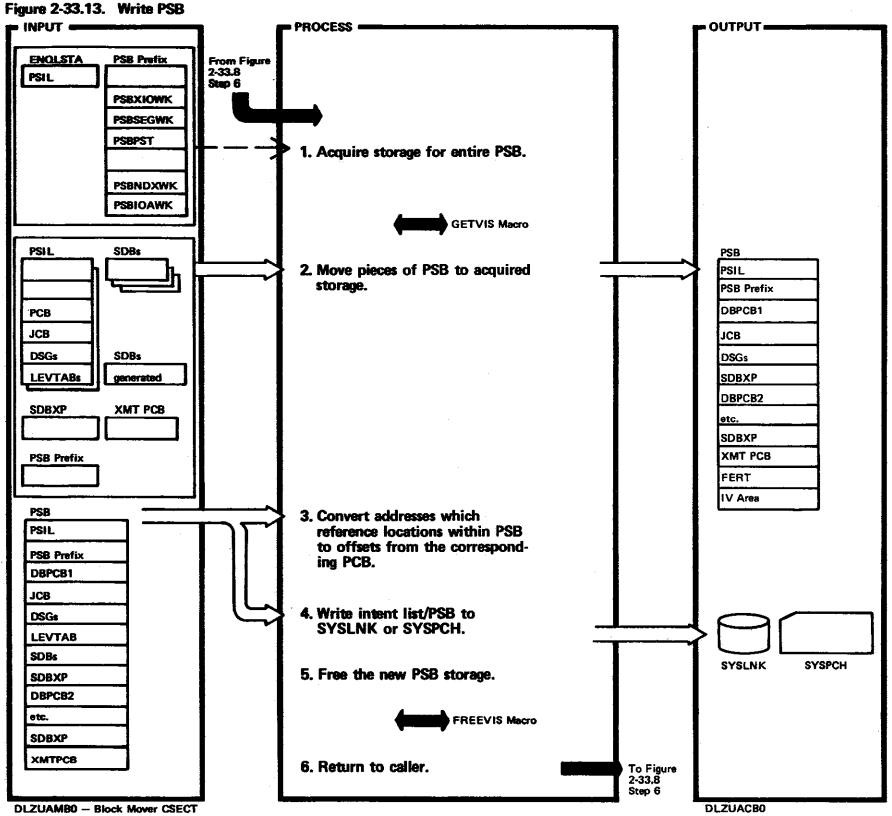
Figure 2-33.12. Write DMBs (Part 2 of 2)



Extended Description	Routine	Label
7. The same subroutine is used for the PSB also. 8. Write message DLZ926I for a FREEVIS error.		DMBOUT3

Extended Description	Routine	Label

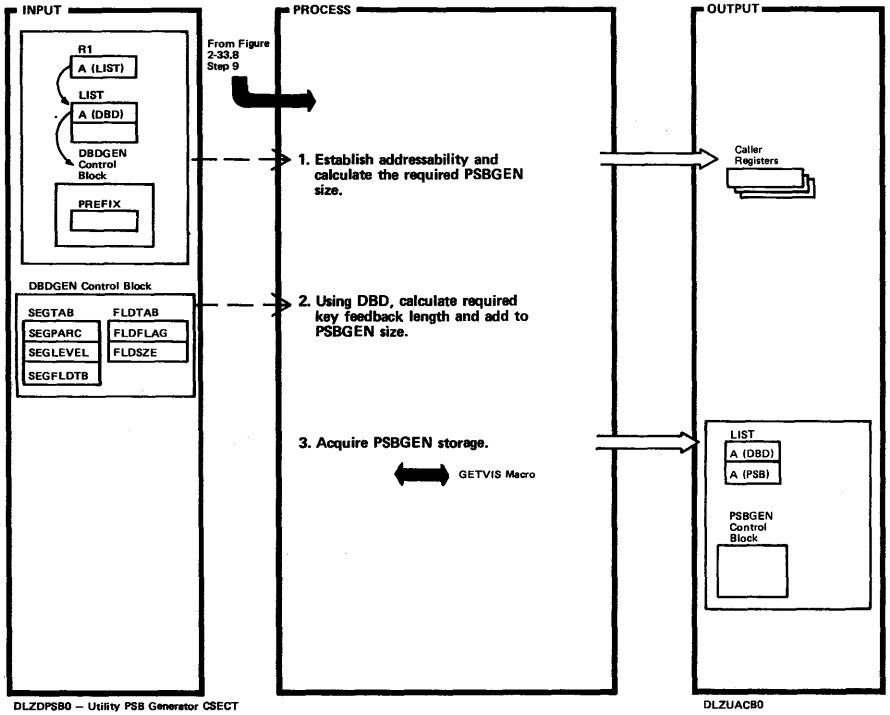
Figure 2-33.13. Write PSB



Extended Description	Routine	Label
1. The size calculation formula is PSBPST – PSBXIOWK – PSBSEGWK – PSBNDXWK – PSBIOAWK + length of PSIL. Write message DLZ905I for GETVIS error.		PSBMOV
2.		PCBMOV SDBMOV FSBMOV FRIVMOV
3.		FERTREL PCBREL DSGREL SDBREL FSBREL
4.		PSBOUT
5. Write message DLZ926I for FREEVIS error.		

Extended Description	Routine	Label

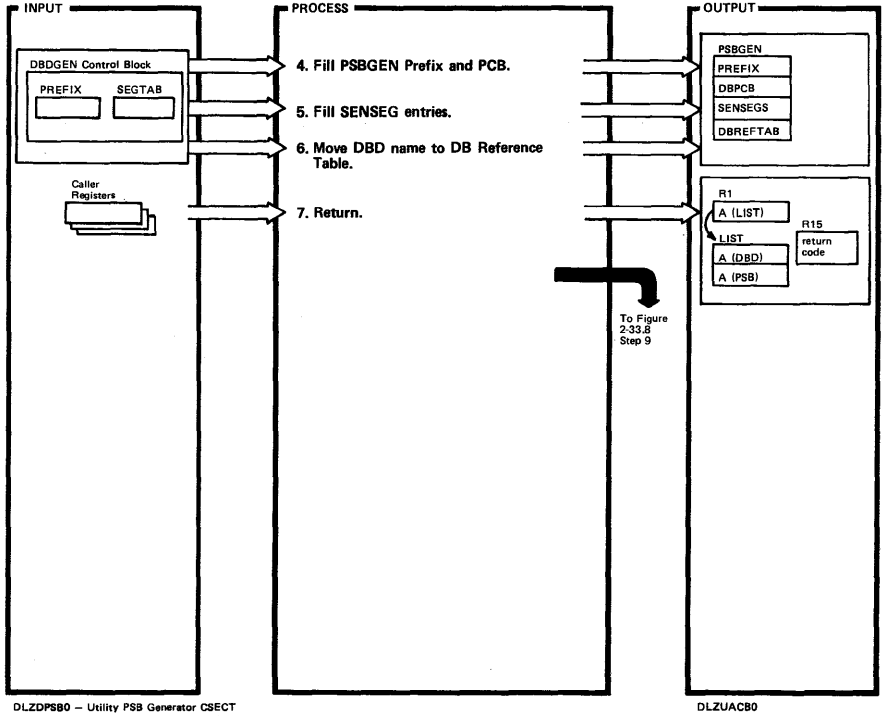
Figure 2-33.14. Build PSB (Part 1 of 2)



Extended Description	Routine	Label
1. Routine identifier DLZDPSB0 vrnp is defined here. A parameter list containing DBD address is passed in Register 1. The contents of this DBD are used to create the utility PSB. The PSBGEN size will be the fixed size plus the number of segments times the length of SENSEG entry. It is possible to have an invalid access method error to pass back to DLZDLBL0.	DLZDPSB0	DLZDPSB0 INIT
2. The result will be stored in PSB Prefix.		SEGLOOP GETKEYSZ
3. The area is also cleared to zeros. Write message DLZ9051 for GETVIS error.		USECURR1

Extended Description	Routine	Label

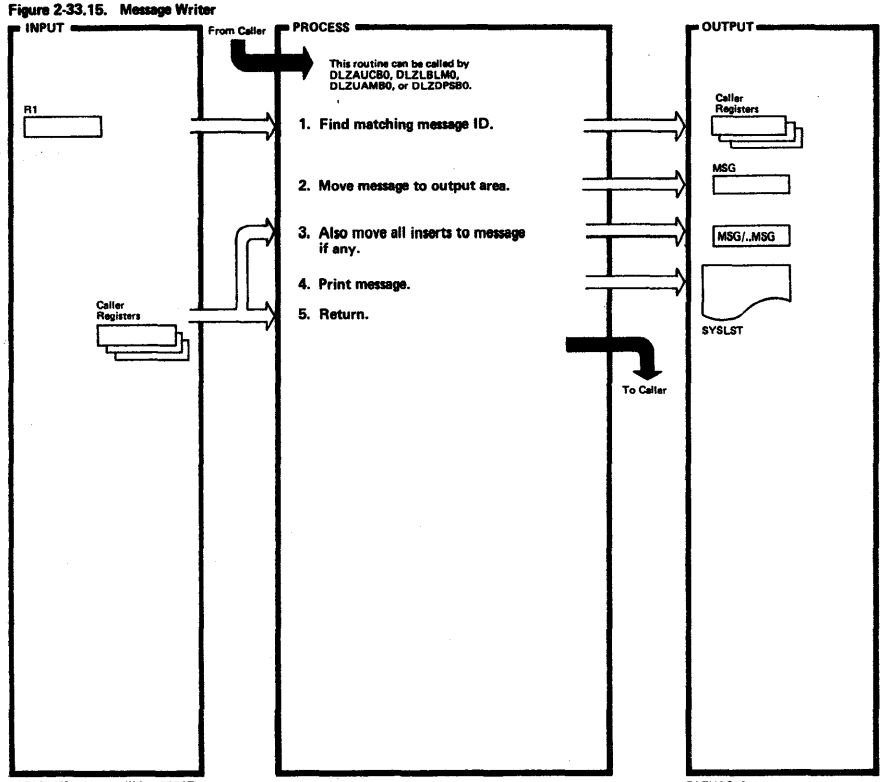
Figure 2-33.14. Build PSB (Part 2 of 2)



Extended Description	Routine	Label
4. PROCOPT of 'A' is set in PCB for all DBDs except secondary index where 'LS' is set. The 'A' is changed to the proper 'load' by batch initialization if necessary.	DLZDPSB0	CLRDONE
5. Same PROCOPT as in Note 4.		PSEGLOOP
6. In addition, no SORTAB is indicated.		SETDBREF
7. The address of the built utility PSB is returned to the caller in the parameter list.		RETURN

Extended Description	Routine	Label

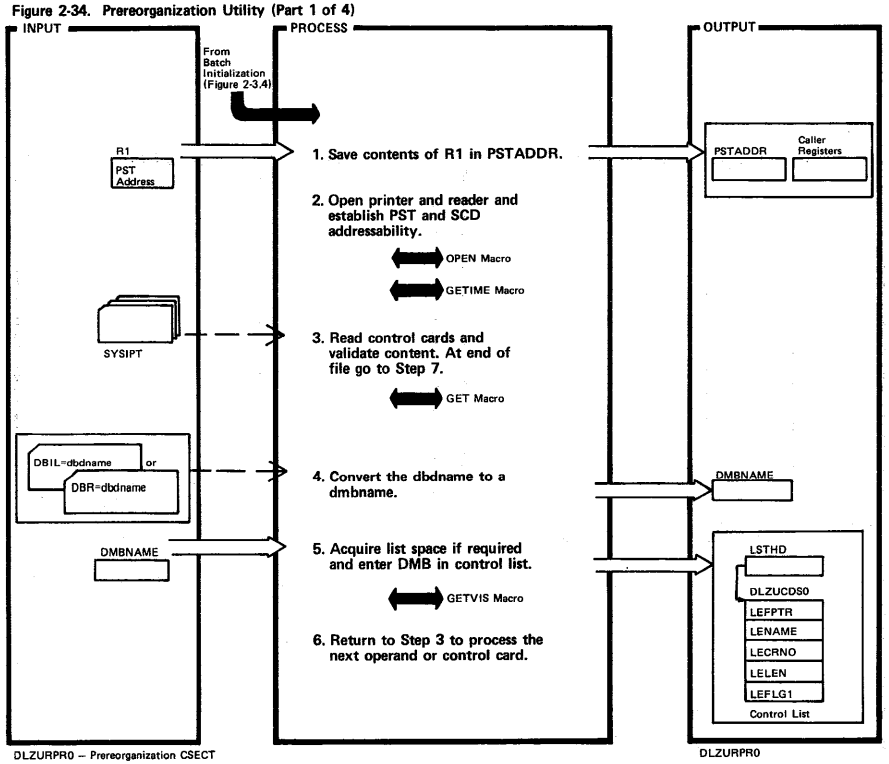
Figure 2-33.15. Message Writer



Extended Description	Routine	Label
4. A subroutine in DLZUACB0 CSECT is called to do the actual printing.		

Extended Description	Routine	Label

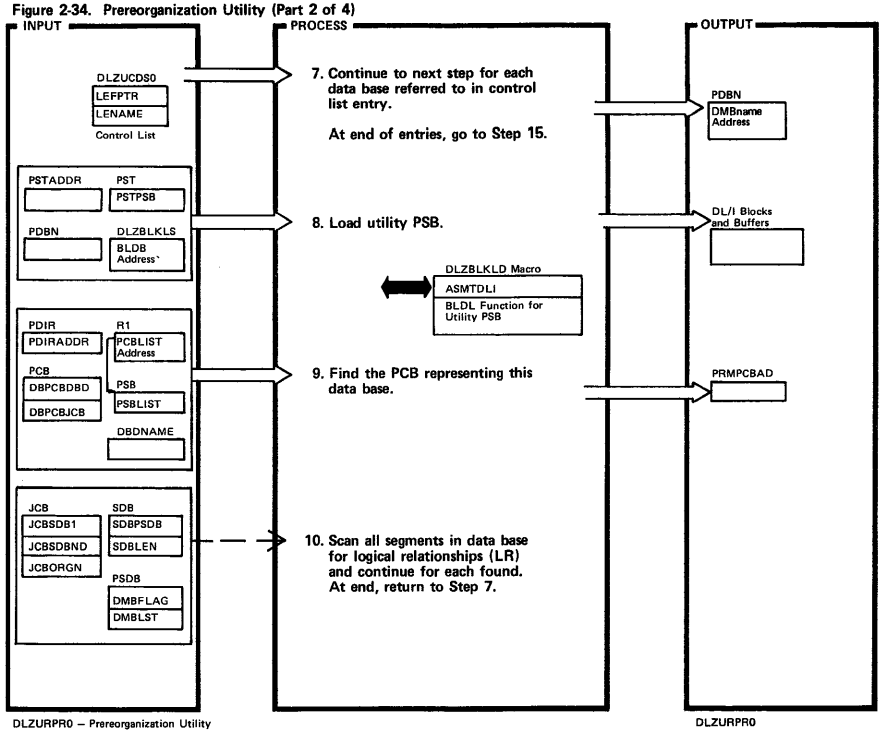
Figure 2-34. Prereorganization Utility (Part 1 of 4)



Extended Description	Routine	Label
1. Module identifier (DLZURPR0vrnp) is defined here. This utility executes as 'ULU' under DL/I control. No blocks or buffers have been loaded yet. Only the nucleus exists. Batch initialization passes the PST address to the logical relationship utilities rather than a PCB address.	DLZURPR0	DLZURPR0 HERE
2.		OPEN1
3. Control card contains identifier as DBIL= (initial load), DBR= (reorganize), OPTIONS=, and dbdnames. In case of an input control card format error, message DLZ954I is printed and job terminates.		NXTCR
4.		DBIREC DBRREC

Extended Description	Routine	Label
5. Control list entries contain DMB names of data base and user options specified in control cards. Macro DLZUCDS0 contains the DSECT defining the format of a control list entry. Write message DLZ963I if an entry already exists for a dmbname. Write message DLZ965I if the number of control list entries exceeds the maximum of 20. Write message DLZ391I if GETVIS fails.		LSTINS

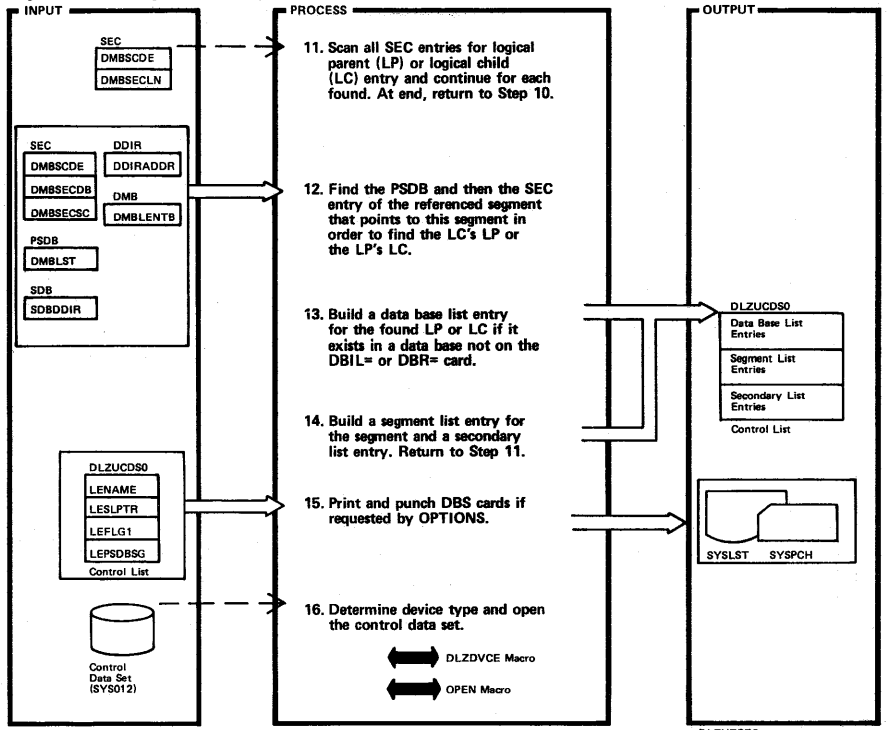
Figure 2-34. Prereorganization Utility (Part 2 of 4)



Extended Description	Routine	Label
7. Write message DLZ964I for no DBIL or DBR control cards. Write message DLZ976I if the dbdname specified is an index DBD.	DLZURPR0	SCAN SCAN10
8. The DLZBLKLD macro moves the dmbname to PST at PSTPCPSB and sets utility suffix 'U' and calls DL/I with the BLDB call function. Write message DLZ956I for a data base control block build failure or if there is no PCB. R1 is returned with the address of the PSBLIST.		BLDBLKS
9.		NEWDBLP
10. If a LP/LC exists for a segment, continue to the next step to look at the secondary list entries for the segment. After each SDB with a LR has been processed return to Step 7 to process the next data base control list entry.		SCAN20

Extended Description	Routine	Label

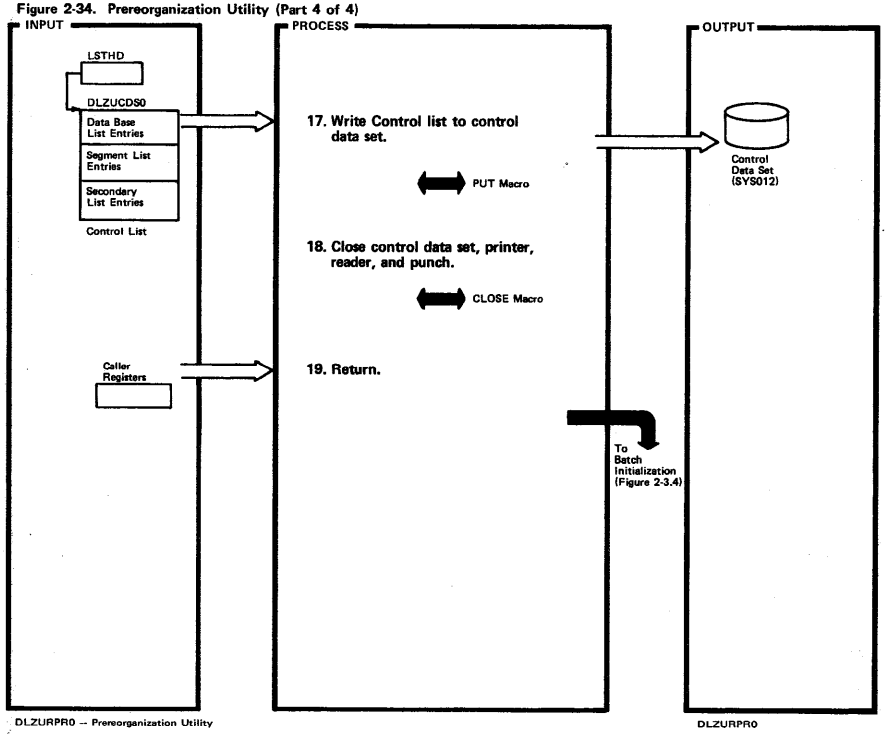
Figure 2-34. Prereorganization Utility (Part 3 of 4)



Extended Description	Routine	Label
11.	DLZURPR0	SCAN50
12. The correct SEC is found by comparing the dbdname in the SEC to the dbdname in the SDB. Each SEC for the segment is examined to see if it is a LP or LC entry. Write message DLZ965I if a SEC is not found with a matching dbdname.		SCAN60
13. Write message DLZ985I for a limit check failure.		SCAN110 SCAN140
14. Segment entries contain segment names involved in logical relationships (LR). Secondary list entries contain DMB names which refer to logically related data bases. As control data set list entries are built, each record is calculated to determine a maximum record length. The largest size is saved and put into field LESRTSZE when the control data set is written (Step 17).		

Extended Description	Routine	Label
15. DBS indicates data base must be scanned using SCAN utility (DLZURGS0). Write message DLZ962I to list segments scanned.		SCNLST NXTDB
16. Write message DLZ984I if there is an invalid device assignment for SYS012.		SCAN500 OPENCTL

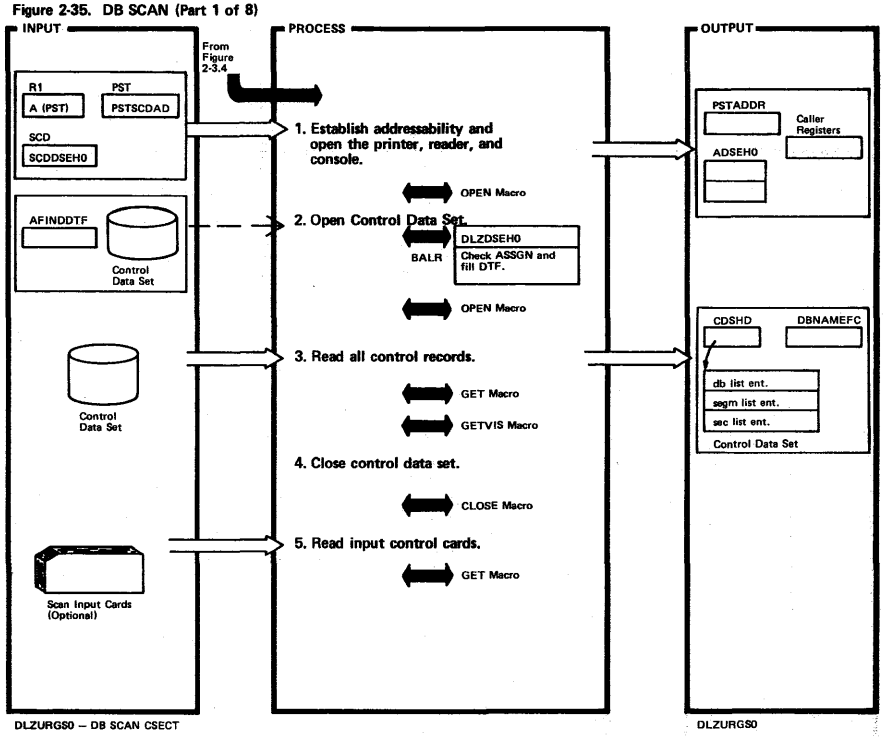
Figure 2-34. Prereorganization Utility (Part 4 of 4)



Extended Description	Routine	Label
17.	DLZURPR0	SCLPS
18. Write message DLZ966I for a normal termination.		SCAN700 TERM
19.		GOODRET

Extended Description	Routine	Label

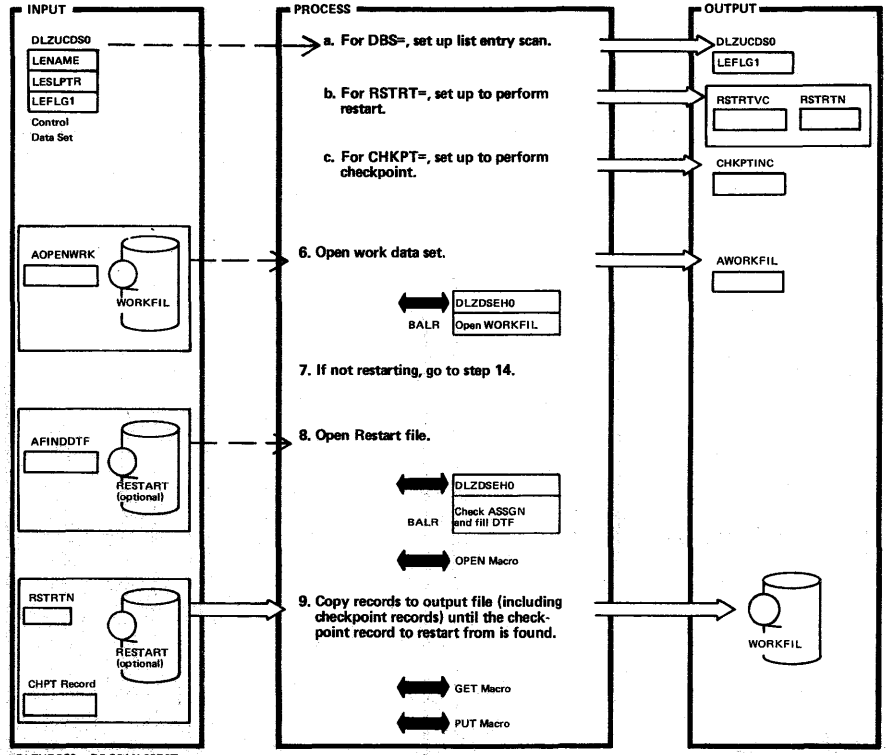
Figure 2-35. DB SCAN (Part 1 of 8)



Extended Description	Routine	Label
1. Module identifier is defined here. This utility executes as 'ULU' under DL/I control. No blocks or buffers have been loaded yet. Only the nucleus exists. Batch Initialization passes the **PST** address to the logical relationship **utilities** rather than a PCB address. The DLZDSEH0 prefix contains addresses to routines within DLZDSEH0. The routines are used by this utility. The addresses are moved to a constant area beginning at ADSEH0.	DLZURGS0	DLZURGS0 OPEN1
2. The 'FINDDTF' subroutine of DLZDSEH0 is used to check that SYS012 is properly assigned to a disk and to fill the correct device type in DTF. Write DLZ984I if the control file is not assigned to a disk.		PROCCTL OPEN
3. The Control Data Set is moved from the I/O area to storage acquired by GETVIS for further processing.		

Extended Description	Routine	Label
3. (con't) Write DLZ391I for a GETVIS failure. Write DLZ957I for no control data set or, if the ID is not 'CONTROL DATA SET'.		
4.		CDSEOFB
5. Input on 'DBS=' card is used to modify Control Data Set in core. 'RSTRT=' and/or 'CHKPT=' specify checkpoint/restart capabilities. 'ABEND' card used for testing. Write DLZ954I for a control card format error.		NXTCR

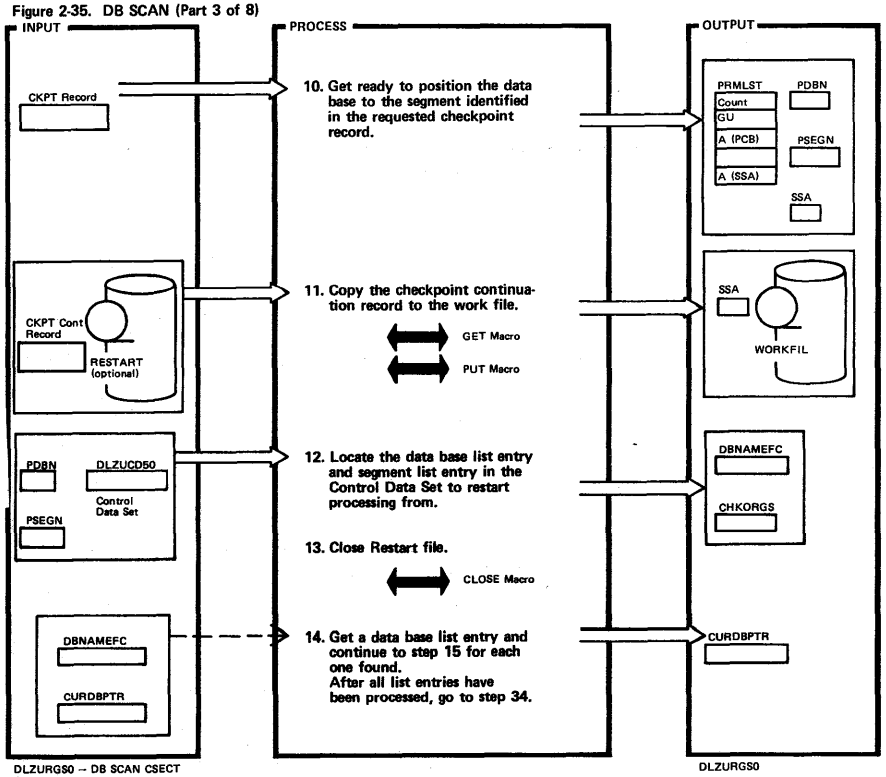
Figure 2-35. DB SCAN (Part 2 of 8)



Extended Description	Routine	Label
a.		DBSREC
b.		RSTRTREC
c.		CHKPTRC
6. Write DLZ984I for unknown or ignored device type.		SCAN
7.		SCAN 10
9. Restart records are copied from the previous WORKFIL until the specified checkpoint record is found. An SSA is set to do qualified GU on last segment to reestablish position.		RSTRT40

Extended Description	Routine	Label

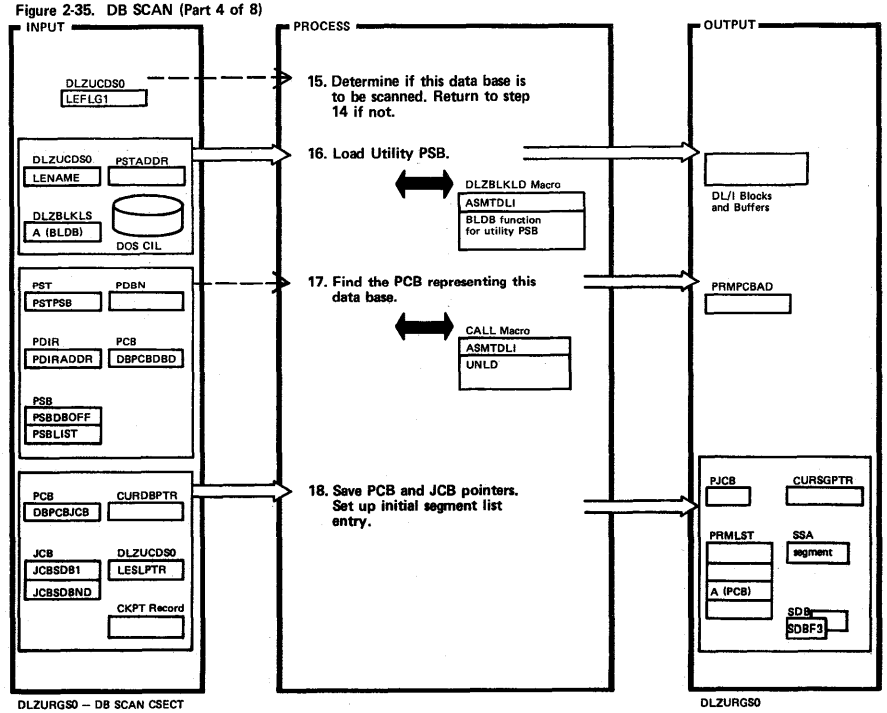
Figure 2-35. DB SCAN (Part 3 of 8)



Extended Description	Routine	Label
10. The SSA built is a qualified key call -- 'segname*C (key)' to be used in the call at step 21.		RSTRT50
11. A continuation checkpoint record contains the key value which is moved to the SSA for a qualified key call.		
12.		RSTRT62
13. Write DLZ975I Restart complete.		RSTRT70
14. Write DLZ966I for normal program termination if there are no more DB list entries to process.		NXTDBP NXTDB

Extended Description	Routine	Label

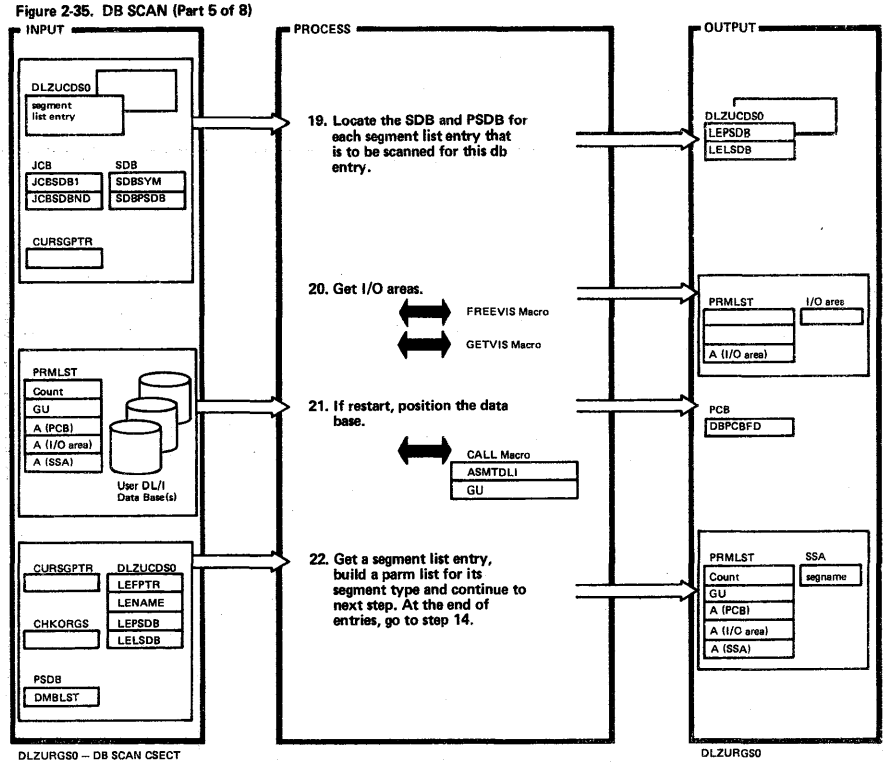
Figure 2-35. DB SCAN (Part 4 of 8)



Extended Description	Routine	Label
15. If DBS control cards were present, only those data bases in the control list that were on DBS cards are scanned. Write DLZ970I scan processing started.		NXTDBA
16. The DLZBLKLD macro moves the dmbname to PST at PSTPCPSB and sets the utility suffix '4'. The 'BLDB' call loads all blocks for the PSB specified and allocates buffers. Write DLZ956I for a data base control block build failure.		BLDBLKS
17. If a PCB is not found, an UNLD is done to release the buffers before the next BLDB call (return to step 16).		NEWDB

Extended Description	Routine	Label

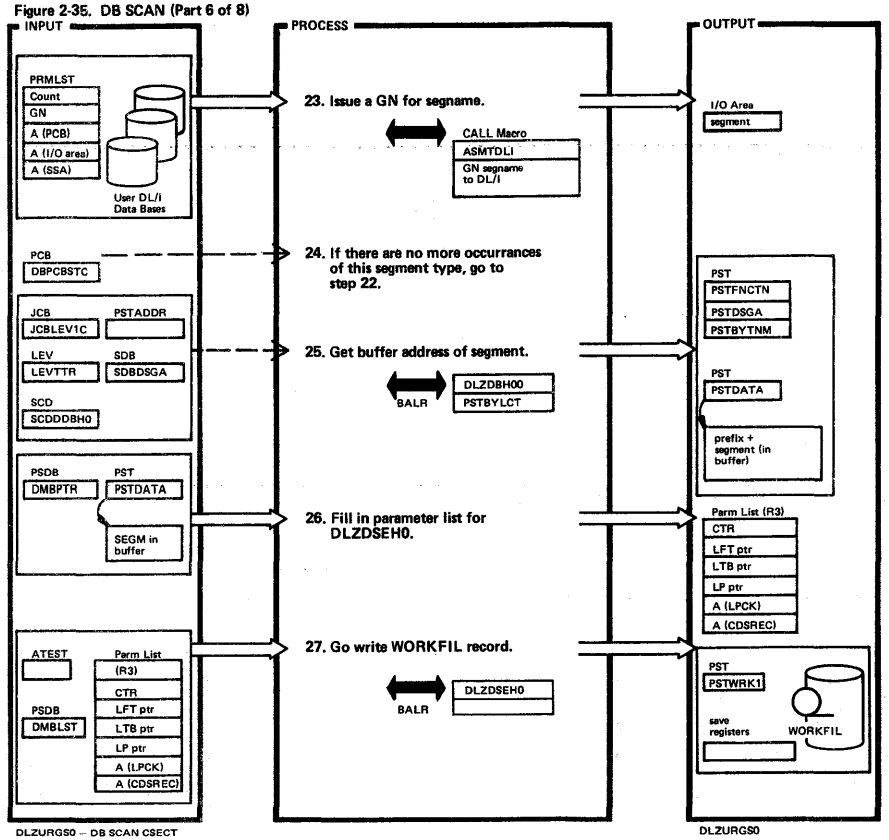
Figure 2-35. DB SCAN (Part 5 of 8)



Extended Description	Routine	Label
19. The Control Data Set entries are modified to save SDB and FSDB addresses. Write DLZ9691 if an SDB is not found for the segment in the segment list entry.		NXTSEG
20. The size is the longest needed for this data base. Any previous I/O area is freed.		SETUP LENSEG
21. Position is by a qualified key call.		CRST
22. The first segment list entry will either be the initial entry for the db or the segment list entry to restart from. Write DLZ9711 Scan processing completed for this DB if no more segment entries.		PROC

Extended Description	Routine	Label

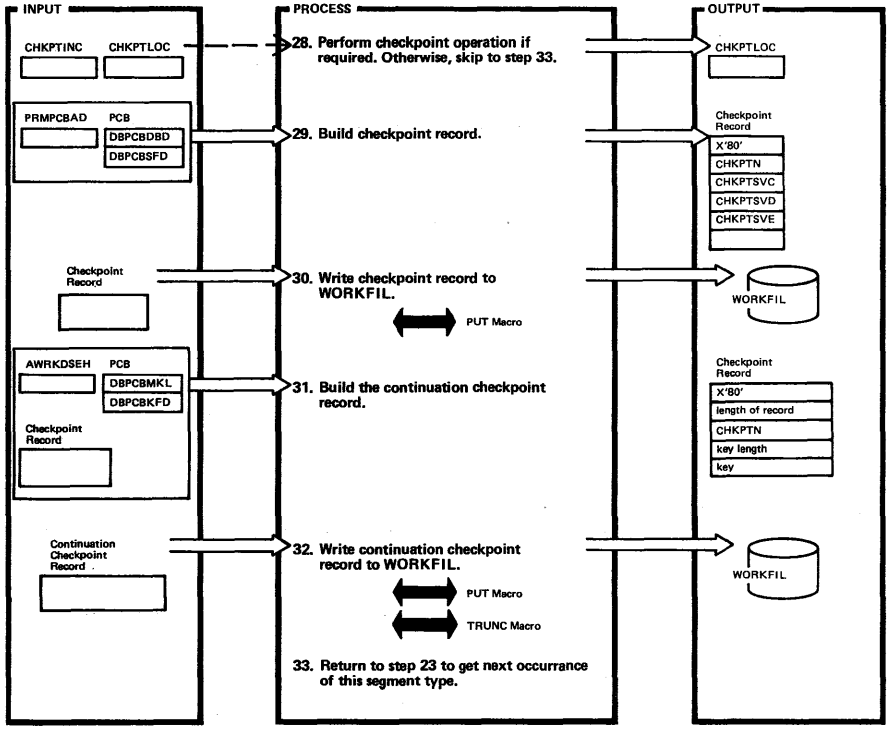
Figure 2-35. DB SCAN (Part 6 of 8)



Extended Description	Routine	Label
23. Every occurrance in the data base of the LC or LP segment type to be scanned is read and a workfile record created for it.		PROC20
24. If the return code is 'GB', indicating EOF for this segment type, return to process next segment list entry.		
25. Scan must have the prefix information to give to DLZDSEH0. Write DLZ9581 for a buffer handler error return.		TEST TESTRSTA
26. In addition, R11 has the address of WORKFIL DTF, R1 has the PST address, and PSTWRK1 has the 'FUNCIHPS' and SDB address.		LPOFFR

Extended Description	Routine	Label
26. (con't) Test routine in DLZDSEH0 will determine the output records required by scanning the SEC list.		
27. Registers are saved and then restored upon return. Write DLZ9521 for an error return code from DLZDSEH0.		TESTRT

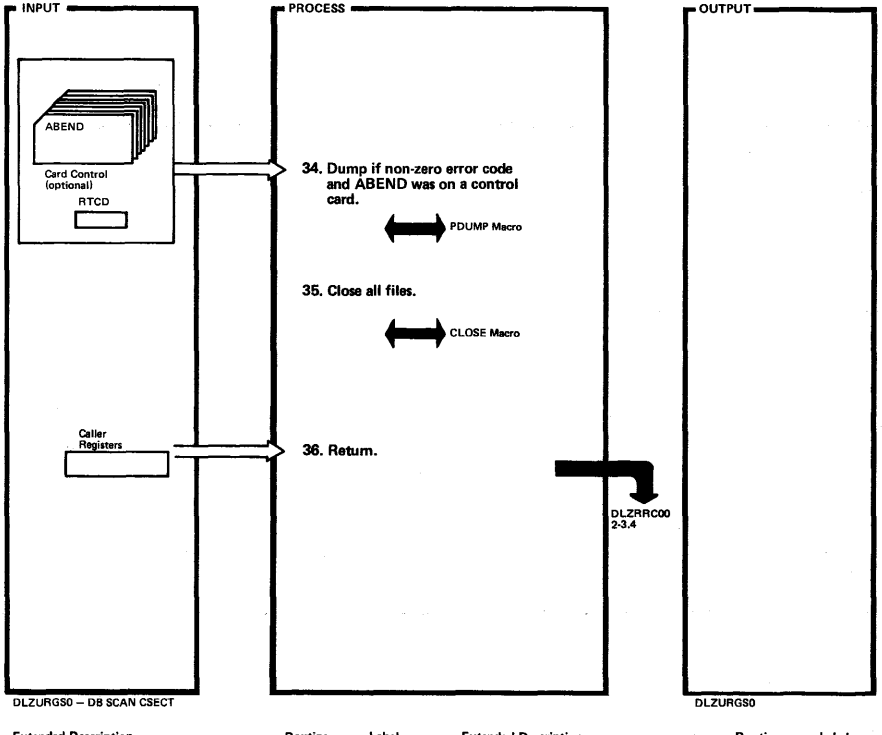
Figure 2-35. DB SCAN (Part 7 of 8)



Extended Description	Routine	Label
28. A checkpoint record is written after every 'n' work file records. 'n' is specified on the CHKPT input card.		CHKPT
29. Note that the checkpoint record is built with zeros where the RBN number would be (CHKPTSUB). This forces an *C call (qualification by concatenated key) instead of *T (retrieve by direct address) during restart processing. Retrieve by direct address would be used if working with HISAM data sets.		CHKPT18
30.		CHKPT19
31.		CHKPT22
32. Message DLZ967I is written to the console giving the current checkpoint record number for later reference.		CHKPT24
33.		TLISTD

Extended Description	Routine	Label

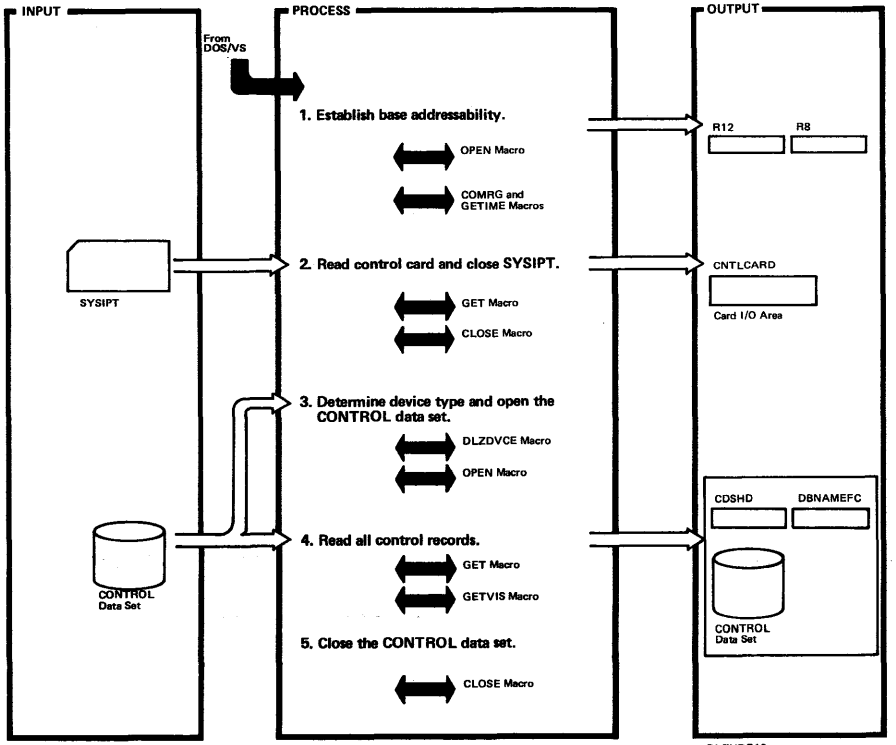
Figure 2-35. DB SCAN (Part 8 of 8)



Extended Description	Routine	Label
34.		TERM
36. Indicate to batch initialization that UNLD is not necessary (R15 = 4) if this utility is terminating before any blocks were loaded or if there was a control block build failure.		Return

Extended Description	Routine	Label

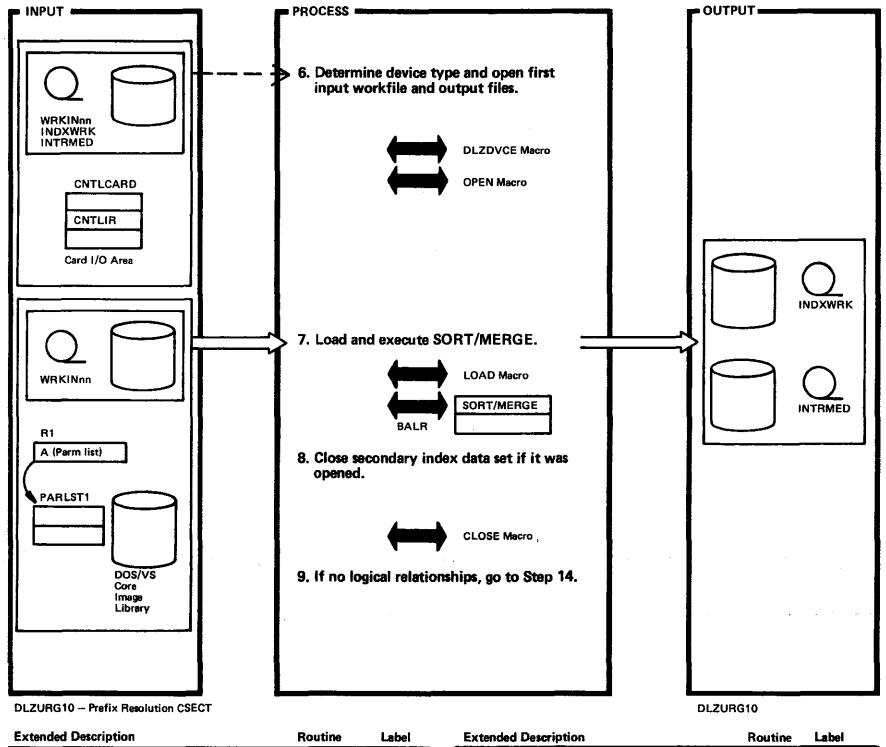
Figure 2-36. Prefix Resolution (Part 1 of 4)



Extended Description	Routine	Label
1. Module identifier (DLZURG10vrnp) is defined here. The time and date are acquired and message DLZ968I is printed at this time to indicate the beginning of execution for DLZURG10.	DLZURG10	DLZURG10
2. Write message DLZ954I for an input control card format error.		
3. Write message DLZ984I for an invalid device assignment for the file.		CDSIN OPENCTL
4. Write message DLZ957I if there is no control data set or if the ID is not "CONTROL DATA SET". Write message DLZ391I for a GETVIS failure. The maximum record length calculated by the prereorganization utility is obtained from field LESRTSZE and passed to SORT.		

Extended Description	Routine	Label
5.		CDSEOF CDSEOFA

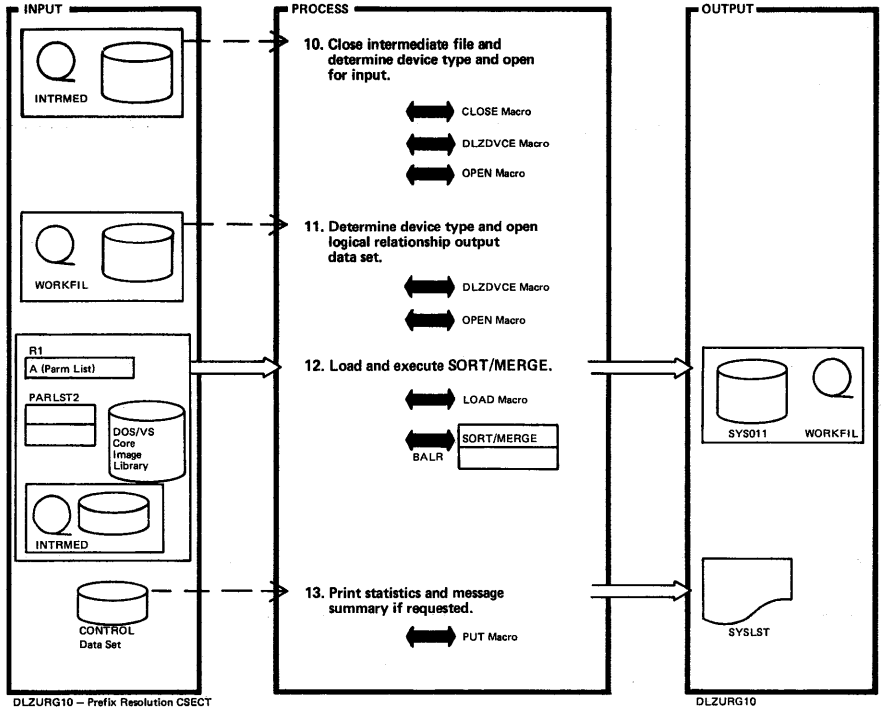
Figure 2-36. Prefix Resolution (Part 2 of 4)



Extended Description	Routine	Label
6. The secondary or logical data sets may or may not be opened depending on the user option on the input control card.		OPENRT1 OPIND OPENLR
7. Write message DLZ982I if the return code from SORT is not zero and go to Step 15. Sort is by (13, 255, A, 5, 1, A). Exits E15 and E35 are described in Figures 2-36.1 and 2-36.2.		SORT1
8. If there was no data put to the secondary index data set, put a dummy record before closing.		SORT11B
9.		SORT11F

Extended Description	Routine	Label

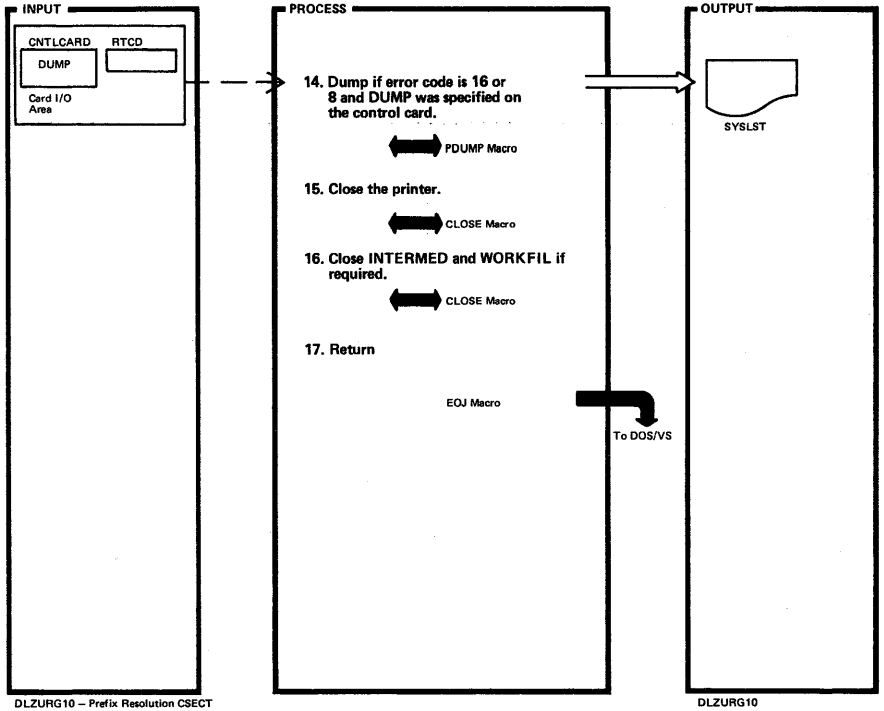
Figure 2-36. Prefix Resolution (Part 3 of 4)



Extended Description	Routine	Label
10.		SORT11D OPENRT2
11.		OPENWLR
12. Sort is by (29, 16, A, 5, 1, A). Exits E15 and E35 are described in Figures 2-36.3 and 2-36.4.		
Write message DLZ982I if the return code from SORT is not zero and go to Step 15.		SORT2
13. Control data set contains options as specified in DLZURPR0.		SUMM STATFLG

Extended Description	Routine	Label

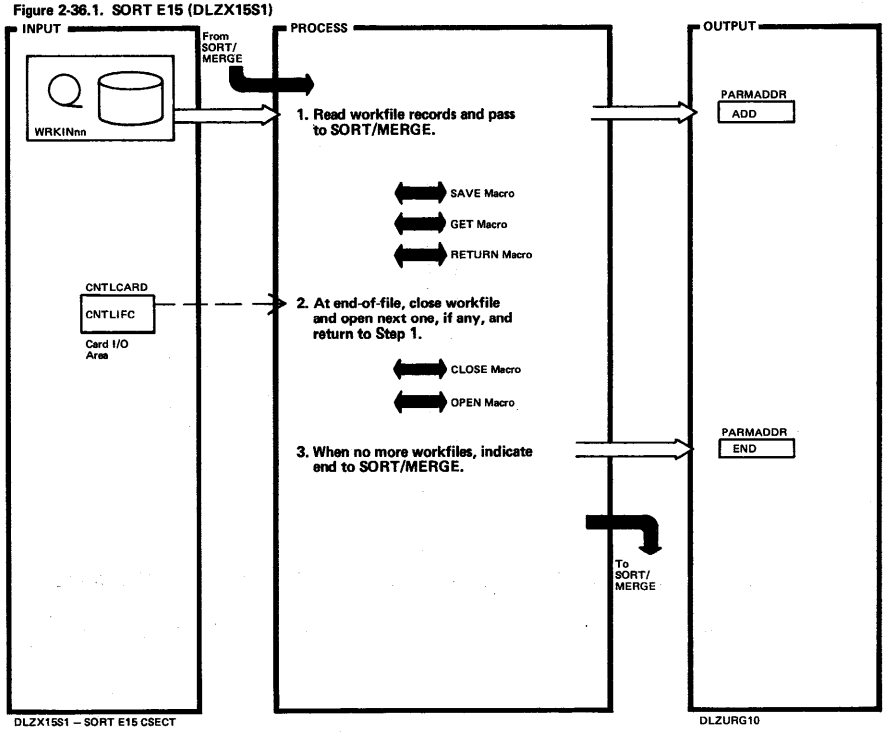
Figure 2-36. Prefix Resolution (Part 4 of 4)



Extended Description	Routine	Label
14. Write message DLZ966I for normal program termination.		STATENO CLOSRT2A
15.		CLOSRT2B
17.		CLOSRT2D

Extended Description	Routine	Label

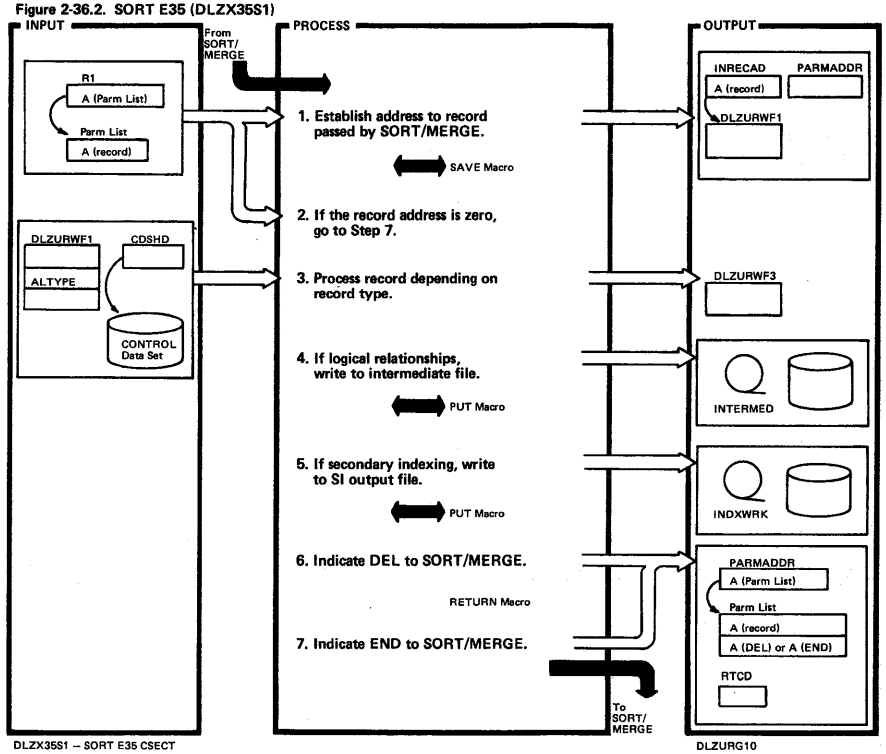
Figure 2-36.1. SORT E15 (DLZX15S1)



Extended Description	Routine	Label
1. Record length is changed to the maximum record length calculated by the prereorganization utility (DLZURPR0) and passed to SORT by the prefix resolution utility (DLZURG10). Original record length is saved in last 2 bytes of LRECL field. Indicate ADD to SORT/MERGE after each GET.	DLZX15S1	DLZX15S1 EXIT15S1
2. CNTLIFC is the number of input workfiles specified on the utility control card.		WRKEOFI NXTFILE

Extended Description	Routine	Label

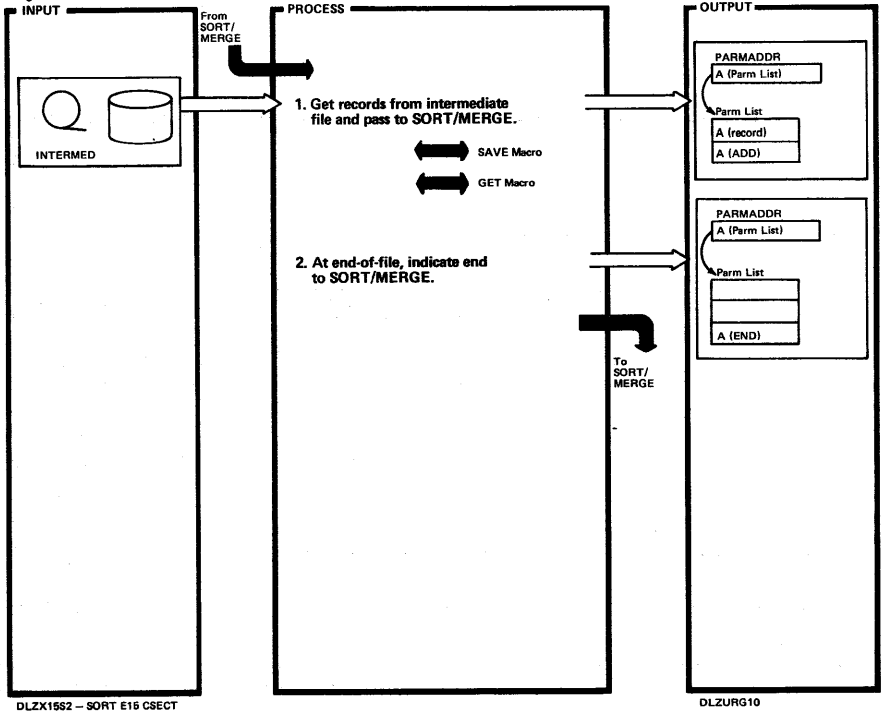
Figure 2-36.2. SORT E35 (DLZX35S1)



Extended Description	Routine	Label
1. SORT/MERGE passes one record at a time to this exit. The record is represented by the macro DLZURWF1 which contains the DSECT defining the format. The original record length is restored before processing.	DLZX35S1	DLZX35S1 EXIT35S1
3. Macro DLZURWF3 contains the DSECT defining the format of the output logical record and later used as input for DLZURGP0. Possible errors are: DLZ955I – Invalid input record. DLZ977I – Duplicate record for LP. DLZ978I – Caution – no LC for LP. DLZ979I – No LP found for LC. DLZ980I – No LC found for LT. DLZ989I – Multiple LC/LP with no LT pointer specified.		ESTTYPE
4. This file used as input for second SORT/MERGE.		STATRIZ OUTPRV1A
5. This is final output for secondary index relationships.		TYPE04RT

Extended Description	Routine	Label
6. SORT/MERGE gets another record and reenters this exit at Step 1.		RETSORT1
7.		ENDSORT1

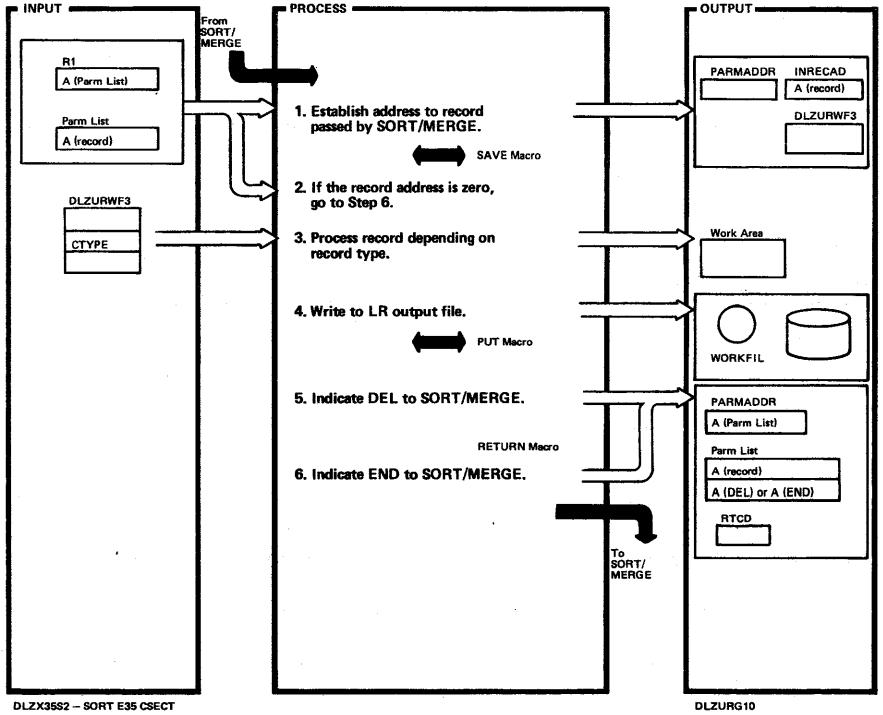
Figure 2-36.3. SORT E15 (DLZX15S2)



Extended Description	Routine	Label
1. This file was written during first sort.	DLZX15S2	DLZX15S2 EXIT15S2
2.		MEDEOFI

Extended Description	Routine	Label

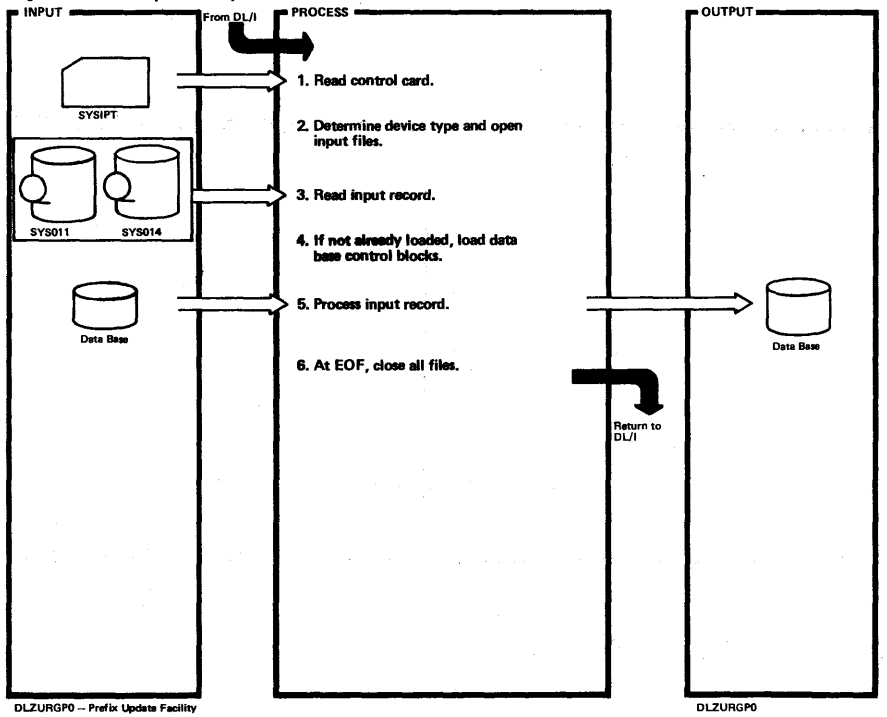
Figure 2-36.4. SORT E35 (DLZX35S2)



Extended Description	Routine	Label
1. SORT/MERGE passes one record at a time to this exit.	DLZX35S2	DLZX35S2 EXIT35S2
3. Possible errors are:		TP30RT
DLZ955I – Invalid input record. DLZ980I – No LC found for LT. DLZ981I – Duplicate record for LT.		
4. This file used as input for the prefix update utility (DLZURGP0).		
5. SORT/MERGE gets another record and reenters this exit at Step 1.		RETSORT2
6.		ENDSORT2

Extended Description	Routine	Label

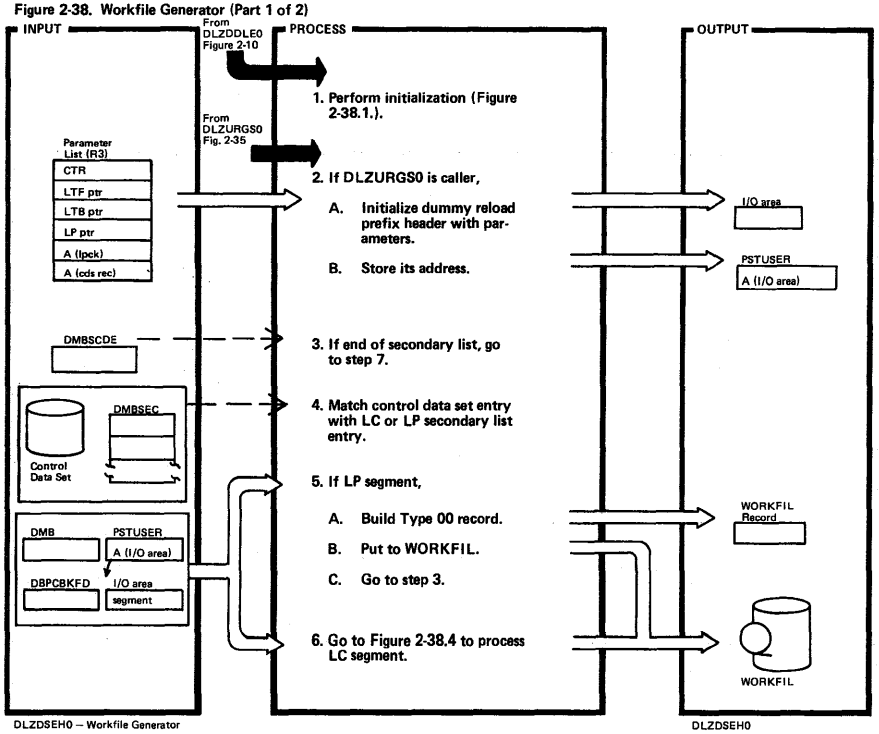
Figure 2-37. Prefix Update Utility



Extended Description	Routine	Label
1.		OPEN1
2. DLZDVCE macro obtains data from PUB. Device type may be TAPE or DASD.		OPENINP
4. DLZBLKLD macro is used to load DB blocks dynamically.		BLDBLKS
5. TYPE 0 and TYPE 1 records (LC/LP) are processed by buffer handler calls. TYPE 4 records (SD) are processed by DL/I INSERT/UPDATE calls.		TYPE0 TYPE1

Extended Description	Routine	Label

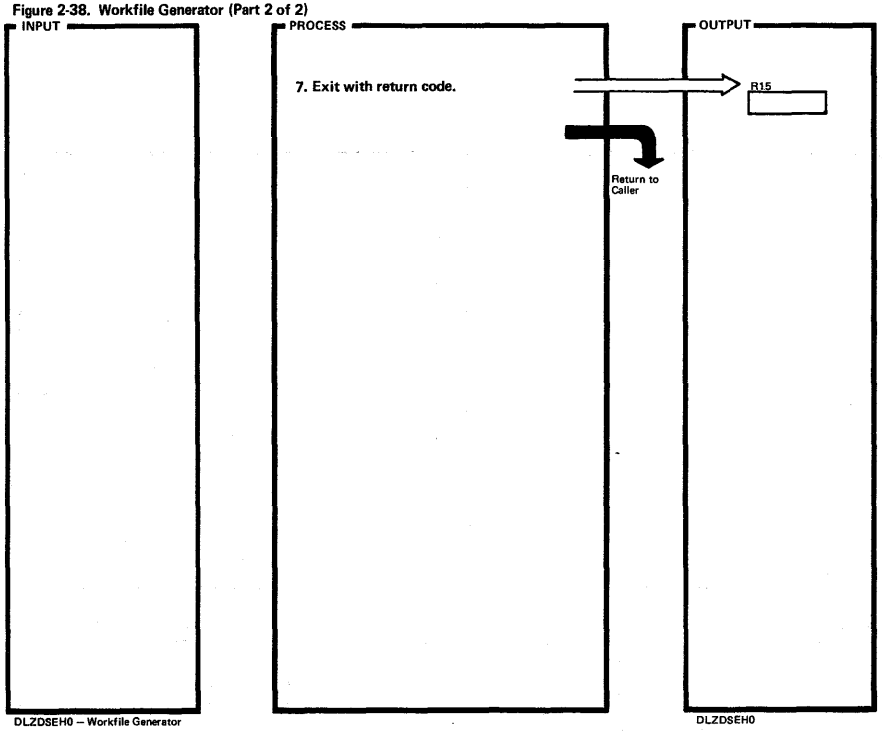
Figure 2-38. Workfile Generator (Part 1 of 2)



Extended Description	Routine	Label
1. This primary entry point is used by Load/Insert when a data base is being initially loaded or reloaded. There are 7 fullwords of addresses immediately preceding this entry point used by modules that interface with DLZDSEH0. A logical parent or logical child record is input to this module.	DLZDSEH0	INIT
2. This is the primary entry point for the scan utility.		TEST
3. This routine must be re-entered when the input segment is an LP because it could have more than one LC type.		TLISTEND
4.		TESTC
5. Description of WORKFIL record can be found in DLZURWF1 dsect.	DLZDSEH0	LP1
6.		CHILD

Extended Description	Routine	Label

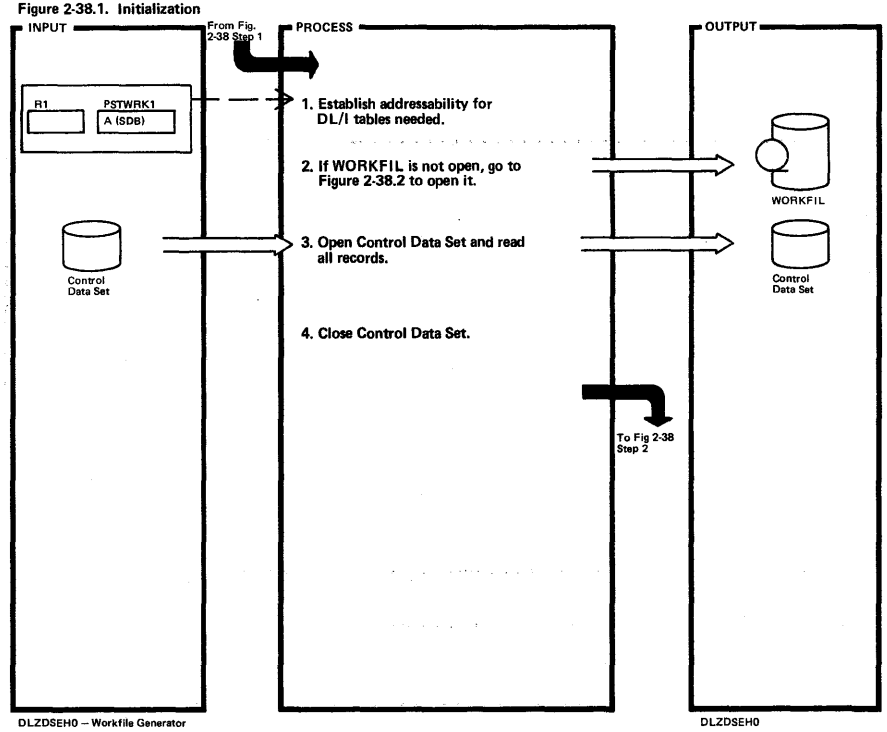
Figure 2-38. Workfile Generator (Part 2 of 2)



Extended Description	Routine	Label
7. If any error occured, call DL/I error message module to write DLZ007I message on console with return code.		RETURN

Extended Description	Routine	Label

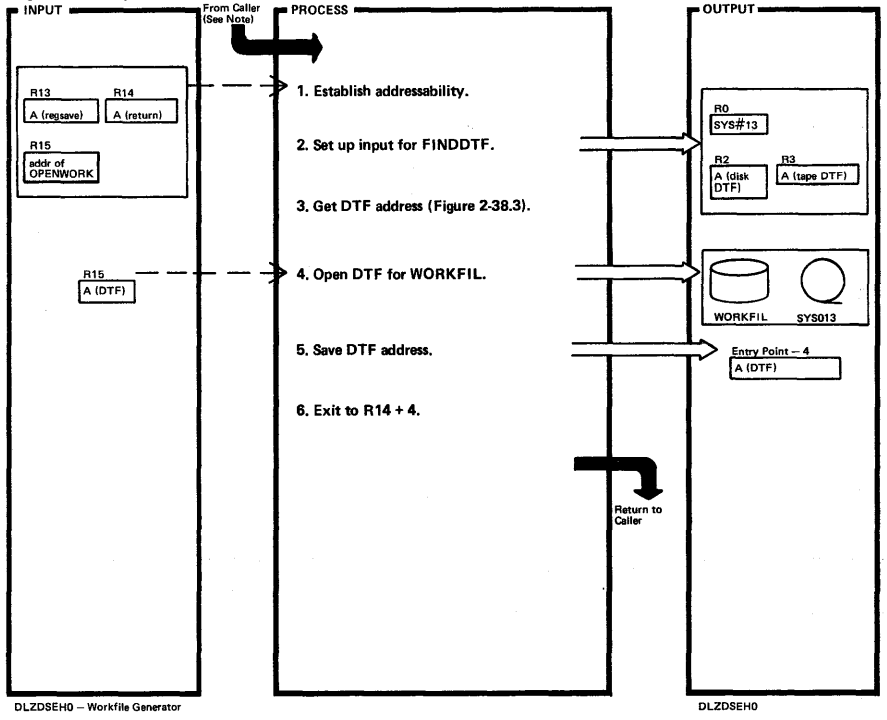
Figure 2-38.1. Initialization



Extended Description	Routine	Label
1. The secondary list entries for the input segment are the primary source of information from the DL/I blocks.	DLZDSEH0	INIT
2. The address of the DTF is found in the address list at the beginning of DLZDSEH0. If it is 0, this workfile must be opened.		
3. This open is done only once. The 'FINDDTF' routine is used to determine the correct DTF. If more than one record exists on the CDS, a GETVIS is done to hold the entire file in storage at one time.		LPLCA

Extended Description	Routine	Label

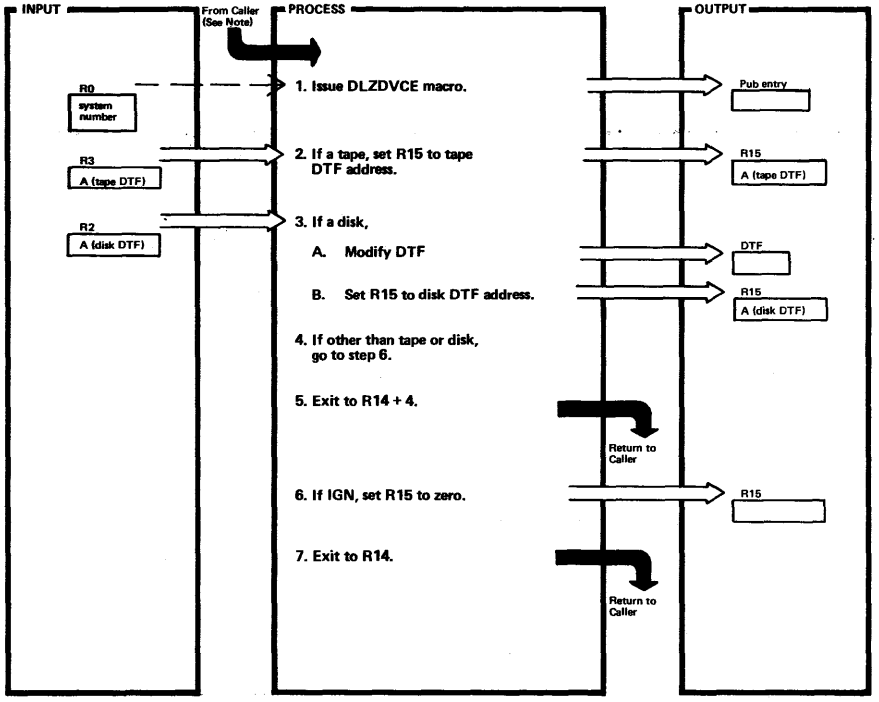
Figure 2-38.2. Open Workfile



Extended Description	Routine	Label
Note: This routine is called by DLZDSEH0, DLZDXMT0, and DLZURGS0.		
1.	OPENWORK	OPENWORK
3. If control is returned to address in R14, an error occurred. R14 + 4 is the normal return.		
4. R15 has address of correct DTF as returned by FINDDTF.		
5. When the WORKFIL is open, the address is saved in the address list in the beginning of DLZDSEH0 csect.		
6. If an error was detected, control is returned to the address in R14. Normal return is R14 + 4.		OPENEXIT

Extended Description	Routine	Label

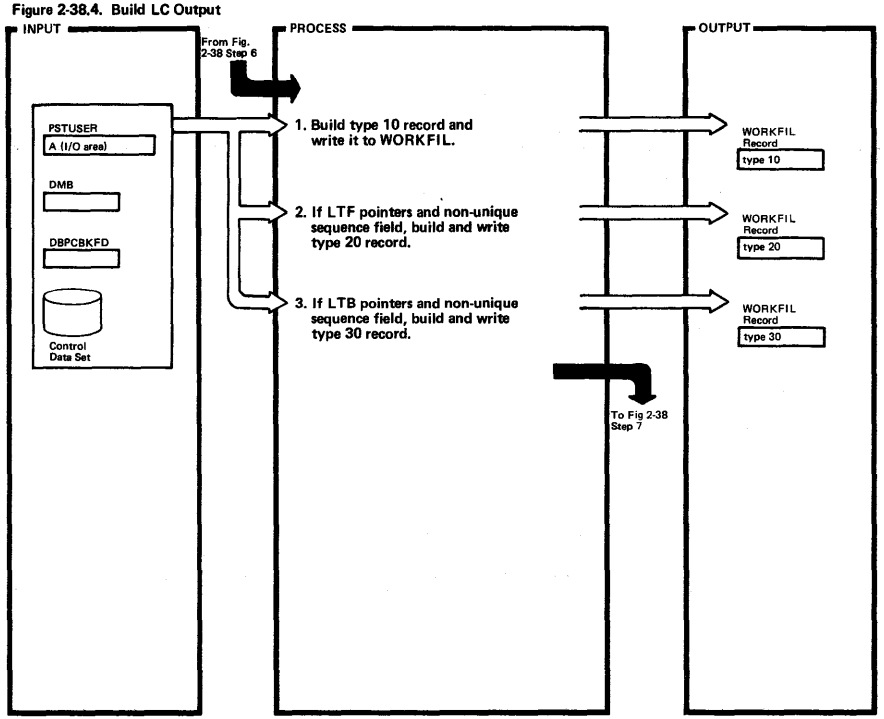
Figure 2-38.3. Find DTF



Extended Description	Routine	Label
Note: This subroutine is called by OPENWORK, DLZDSEH0, and DLZURGS0. DLZDVCE macro finds PUB entry for given programmer logical unit and the device type byte is used to determine further processing.		
1.	OPENWORK	FINDDTF
2. 2400, 3410, and 3420 are supported.		FINDTF0
3. 2314, 3330, 3333, 3340A & B are supported.		FINDTF1 FINDTF2
5. Normal return.		FINDEXIT
6. This allows DLZDXMT0 to build secondary entries.		FINDERRX
7. This is the error exit.		FINDERRU

Extended Description	Routine	Label

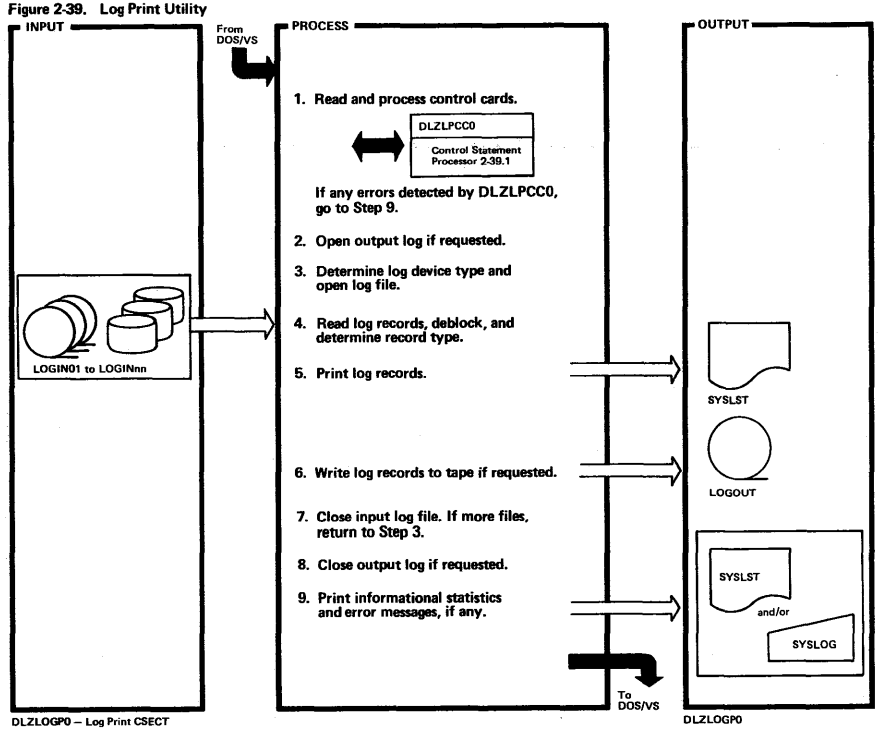
Figure 2-38.4. Build LC Output



Extended Description	Routine	Label
1.		CHILD LC1
2.		LC120A
3.		LC130B

Extended Description	Routine	Label

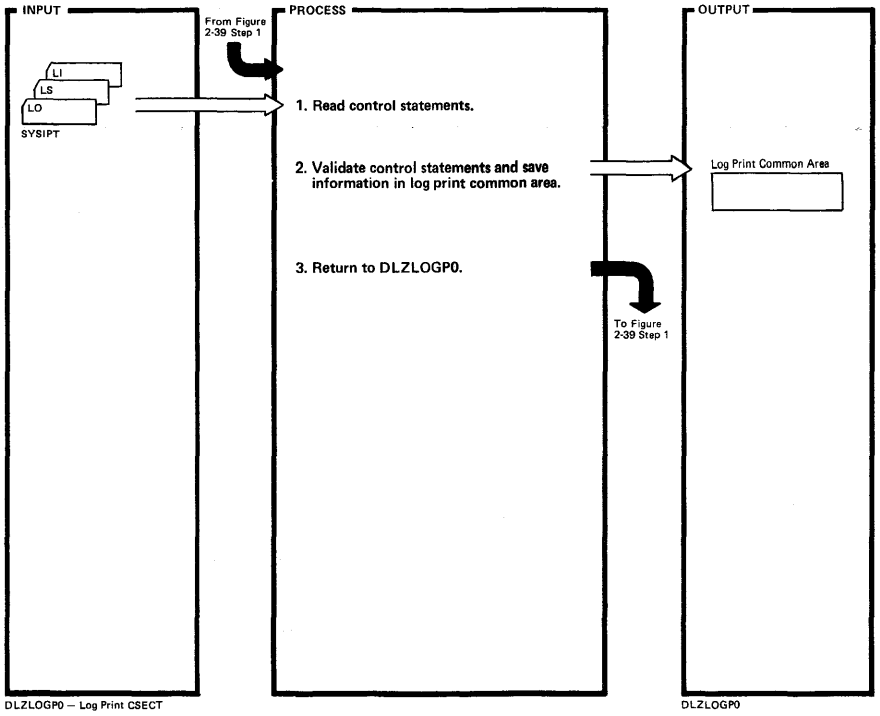
Figure 2-39. Log Print Utility



Extended Description	Routine	Label
1. Register 1 points to log print common area.	DLZLOGP0	GETCARD
2. Output log requested by 'COPY' on 'LO' statement.		CARDEOF
3. DLZDVCE macro obtains data from PUB (physical unit block) and modifies DTF. If VSAM log, ACB is modified manually.		LOGOPEN
4. Valid DL/I record types are: o Data base record (X'50' and X'51') o Open record (X'2F') o Scheduling record (X'08') o Termination record (X'07') o Checkpoint record (X'41')		GETLOG
5. Records are printed in either keyword or dump format.		PRINT
6. Log records are written to tape as read.		GETREC
7. nn of LOGINnn is incremented by 1 if more files.		LOGEOF
8. Output log is closed when log record in error is encountered.		

Extended Description	Routine	Label

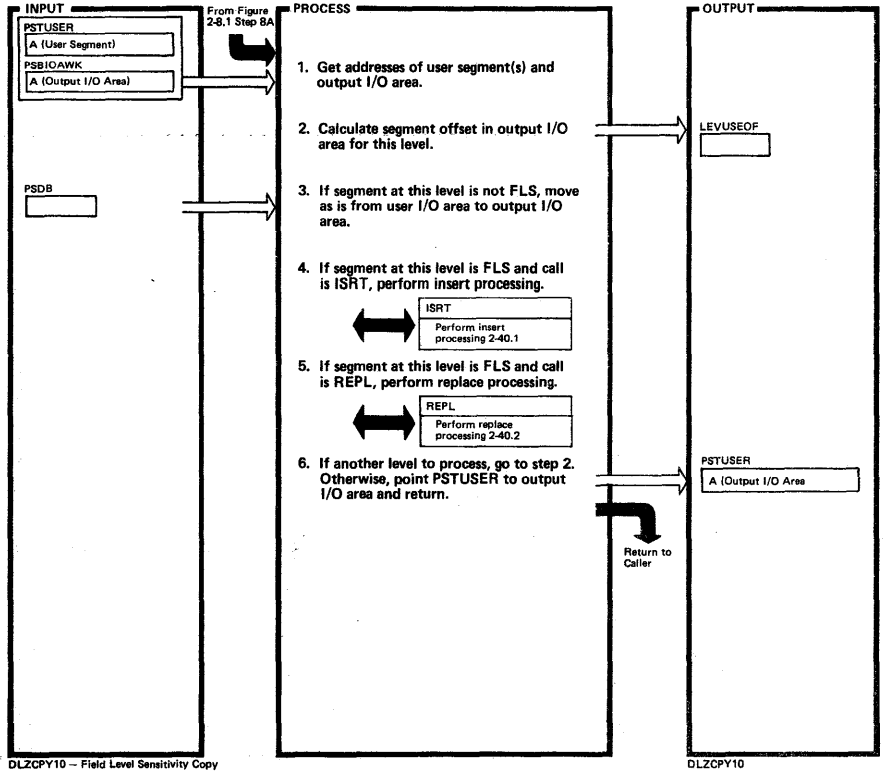
Figure 2-39.1. Control Statement Processor (DLZLPCC0)



Extended Description	Routine	Label
1. Possible card types are: 'LO' – describes print options, 'LS' – describes additional selective print options, 'LI' – describes input log files.	DLZLPCC0	GETCARD
2. Flag ERROROCC in LOGPFLG1 is set if any errors are detected.		LO000 LI000 LS000
3. If no input statements received, print default message DLZ4161.		CARDEOF

Extended Description	Routine	Label

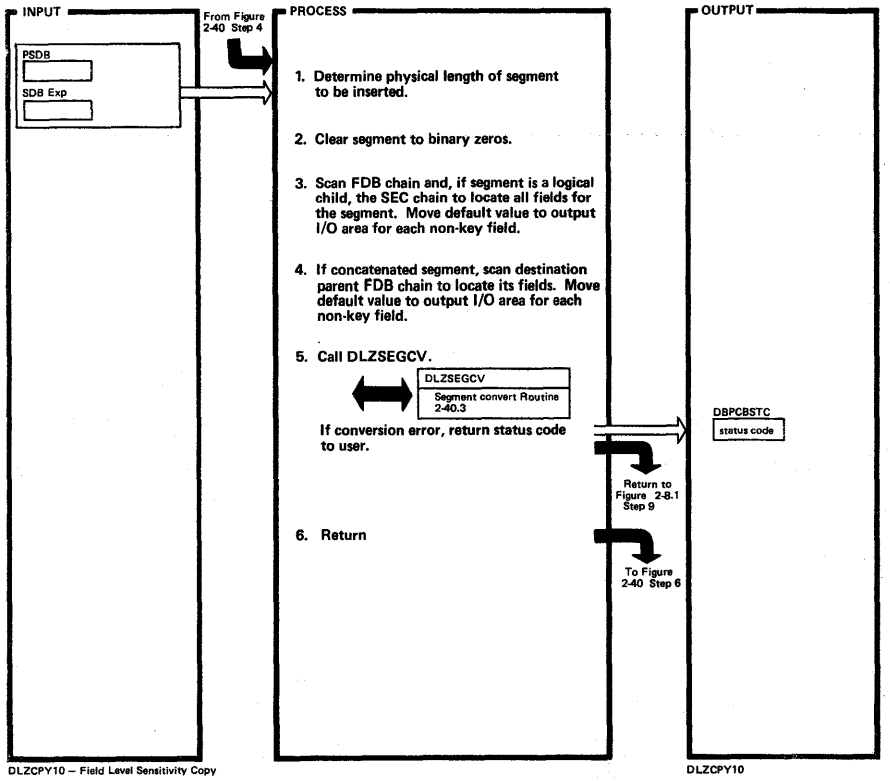
Figure 2-40. Field Level Sensitivity Copy



Extended Description	Routine	Label
1. PSTUSER points to user's view of segment(s). PSBIOAWK points to I/O area to contain physical view of segment(s).	DLZCPY10	DLZCPY10
2. LEVUSEOF will be changed from offset in user I/O area to offset in output I/O area.		LEVLOOP
3. Length of segment to be moved must be determined. If variable length segment, length is in first two bytes of segment, otherwise in PSDB. If concatenated segment, logical child and destination parent lengths must be added.		LEV000
6.		NEXTLEV

Extended Description	Routine	Label

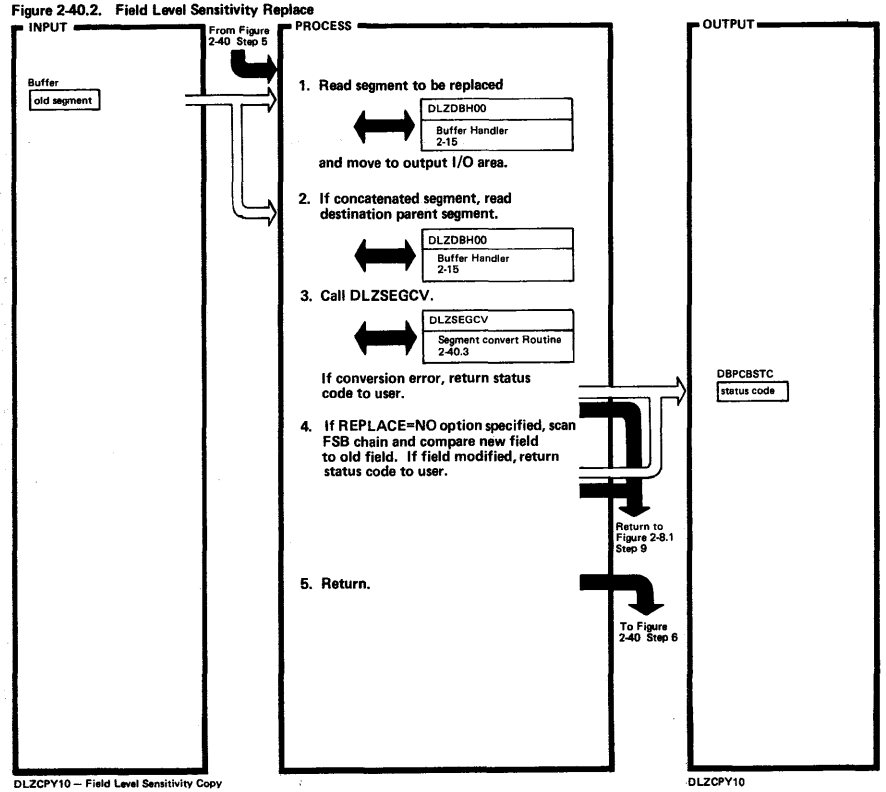
Figure 2-40.1. Field Level Sensitivity Insert



Extended Description	Routine	Label
1. If variable length segment, use insert length in SDB expansion block. Otherwise use length in PSDB. If concatenated segment, logical child and destination parent lengths must be added.	DLZCPY10	ISRT
2.		ISRT100
3.		ISRT105
4.		ISRT180
5. Possible status codes are KA, KB, KC, and KD.		SEGCV

Extended Description	Routine	Label

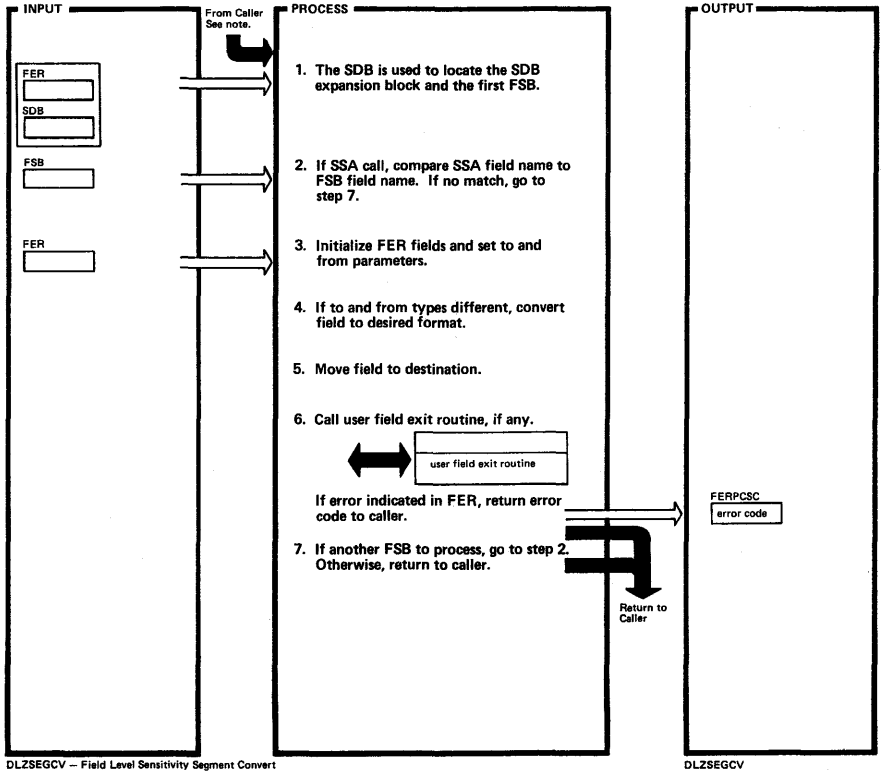
Figure 2-40.2. Field Level Sensitivity Replace



Extended Description	Routine	Label
1.	DLZCPY10	REPL
3. Possible status codes are KA, KB, KC, and KD.		SEGCV
4. Status code KE is returned.		REPL135

Extended Description	Routine	Label

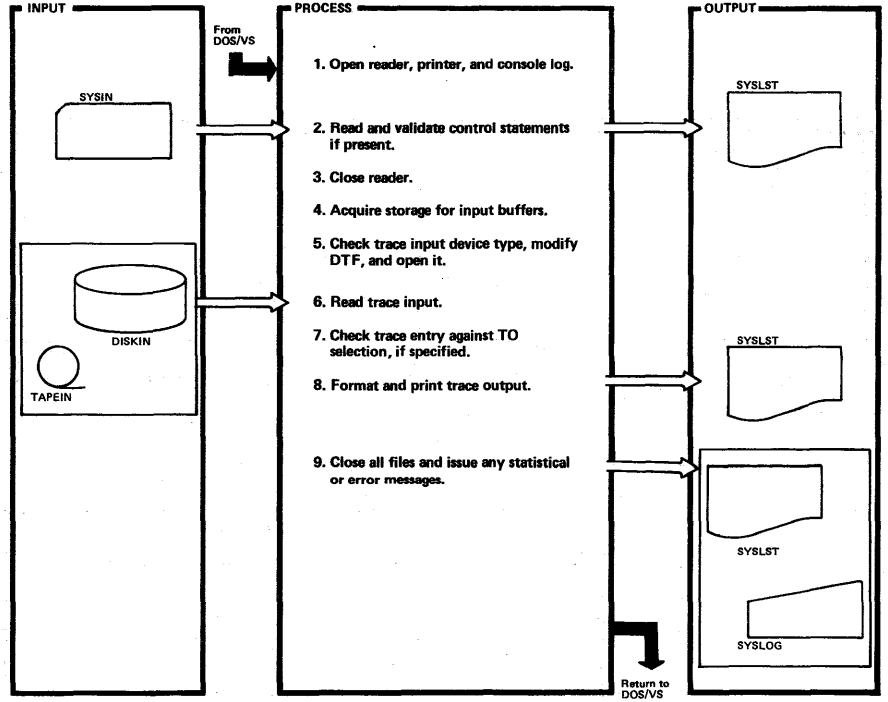
Figure 2-40.3. Field Level Sensitivity Segment Convert



Extended Description	Routine	Label
Note: DLZSEGCV is called by DLZCPY10 and DLZDLR00.		
1.	DLZSEGCV	DLZSEGCV
2. DLZDLR00 makes SSA call to convert SSA user field to physical view. Only this field and its subfields will be converted.		FSBLOOP
3.		FSB010
4.		CONVERT
5.		MOVE
6. Possible error codes are A, B, C, and D.		USEREXIT
7.		NEXTFSB

Extended Description	Routine	Label

Figure 2-41. Trace Print Utility



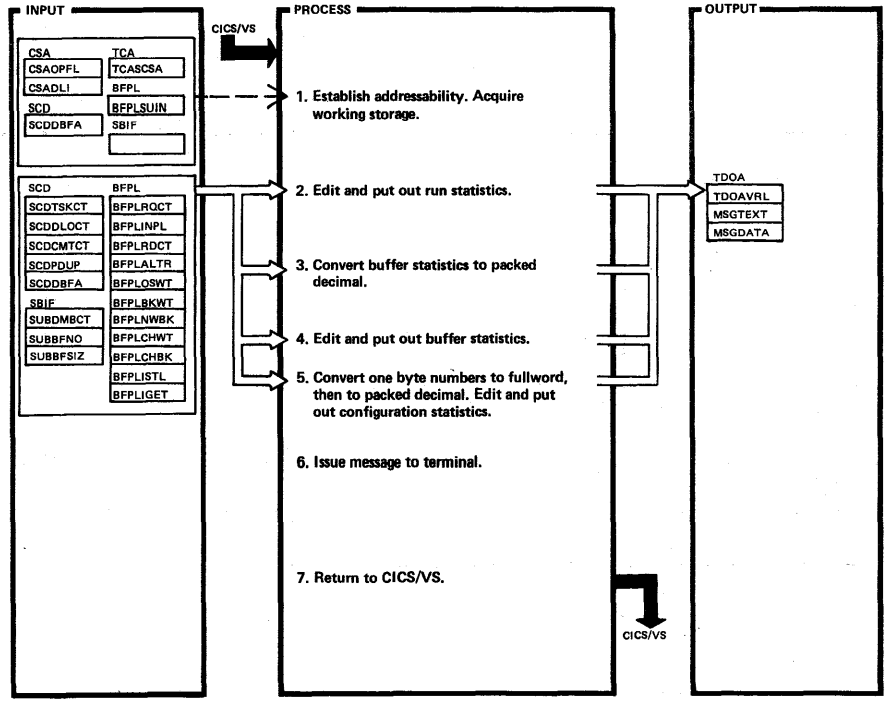
DLZTPRT0 – Trace Print Utility

DLZTPRT0

Extended Description	Routine	Label
2. Reads TI statement if present and prints card image on SYSLST. If any other type statement is present, it is also printed, but no further processing of it takes place.	DLZTPRT0	READCARD
4. Storage is acquired for two input buffers.		GETSTOR
5. The DLZDVCE macro is used to validate the trace input device, and modify the tape or disk DTF.		CKDEV
6. The unformatted trace records are read from the trace input file until EOF is returned.		GETENTRY
8. The trace entries are formatted and printed one at a time until all entries in the record are processed. Control is then passed to step 6 for the next record.		PRTRACE

Extended Description	Routine	Label

Figure 2-42. DL/I Run and Buffer Statistics



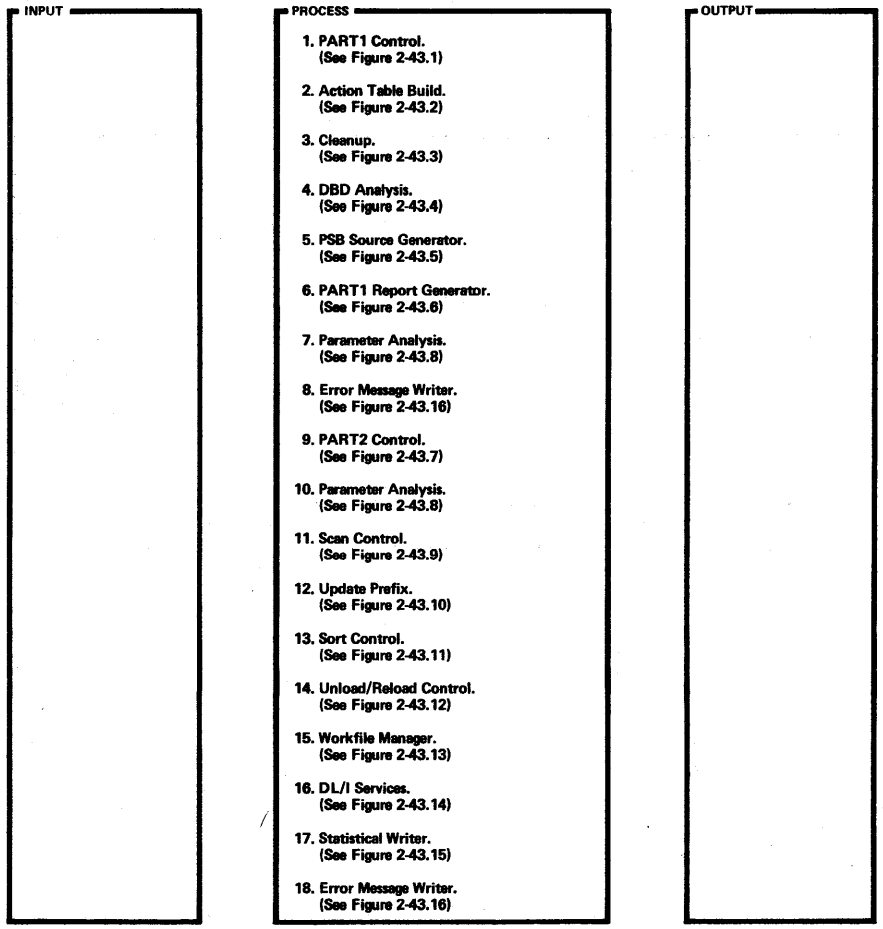
DLZSTTL – DL/I Run and Buffer Statistics

DLZSTTL

Extended Description	Routine	Label
1. Entry is from CICS/VS when the user enters the transaction code 'CSDE' or when the CICS/VS master terminal operator issues the 'CSMT SHUTDOWN' command. R12 contains the TCA address, R13 contains the CSA address, and R14 contains the entry point address.		
2. The CICS/VS macro DFHSC is used to acquire storage for the transient data output area (TDOA).		
3,4,5 Internal subroutine STTLPUT writes the statistics to the CSSL extra-partition transient data set, using the DFHTD macro.		
6. If DL/I is not active, issue message DLZ280I using the DFHWTO macro.		
7. Use DFHPC macro to return to CICS/VS.		

Extended Description	Routine	Label

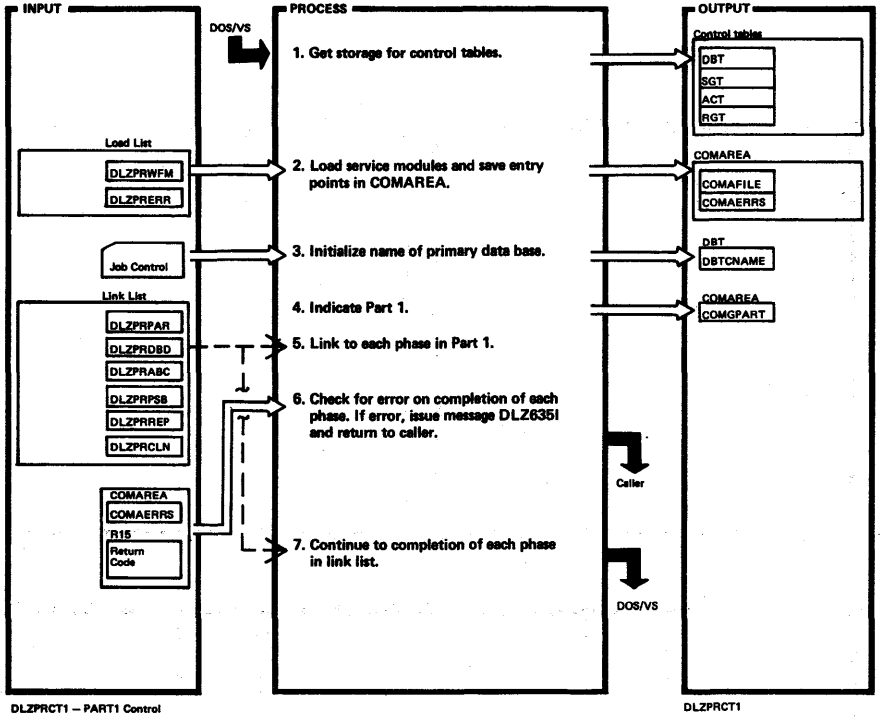
Figure 2-43. Partial Data Base Reorganization (Overview)



Extended Description	Routine	Label
1.	DLZPRCT1	
2.	DLZPRABC	
3.	DLZPRCLN	
4.	DLZPRDBD	
5.	DLZPRPSB	
6.	DLZPRREP	
7.	DLZPRPAR	
8.	DLZPRERR	
9.	DLZPRCT2	

Extended Description	Routine	Label
10.	DLZPRPAR	
11.	DLZPRSCC	
12.	DLZPRUPD	
13.	DLZPRSTC	
14.	DLZPRURC	
15.	DLZPRWFM	
16.	DLZPRDLI	
17.	DLZPRSTW	
18.	DLZPRERR	

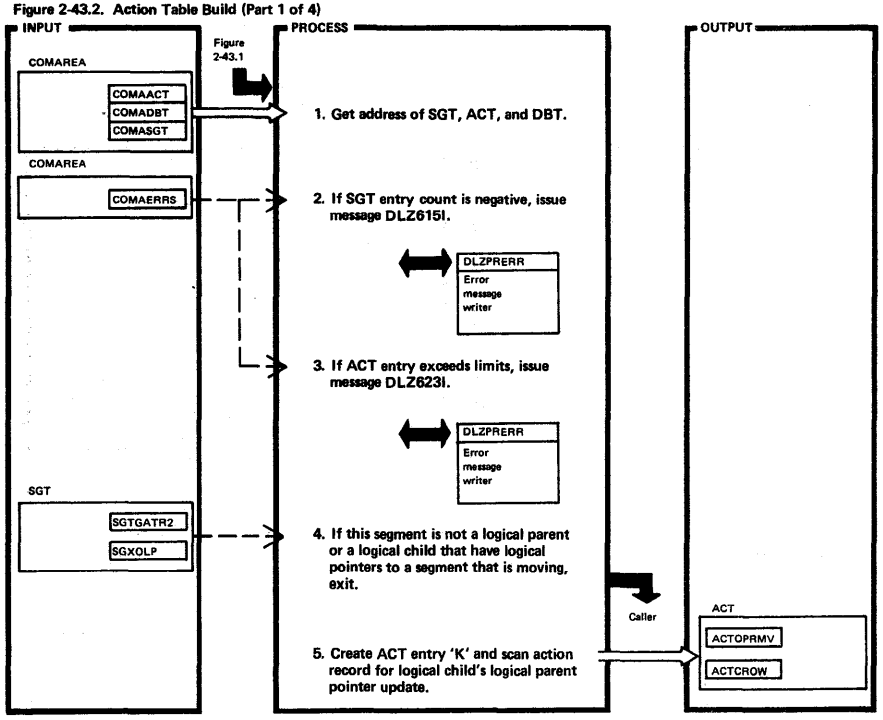
Figure 2-43.1. PART1 Control.



Extended Description	Routine	Label
1. DBT – data base table. SGT – Segment table. ACT – action table. RGT – range table DOS/VS GETVIS issued. 2. The common area (COMAREA) is part (CSECT) of this module and is not dynamically acquired.		

Extended Description	Routine	Label

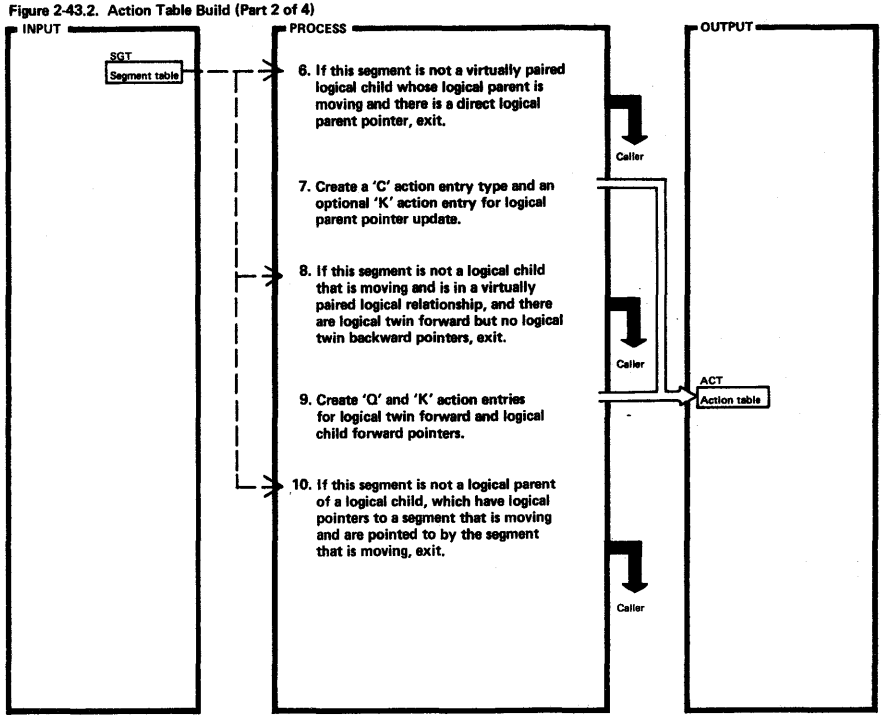
Figure 2-43.2. Action Table Build (Part 1 of 4)



Extended Description	Routine	Label
4. 'K' action records.	DLZROW01	

Extended Description	Routine	Label

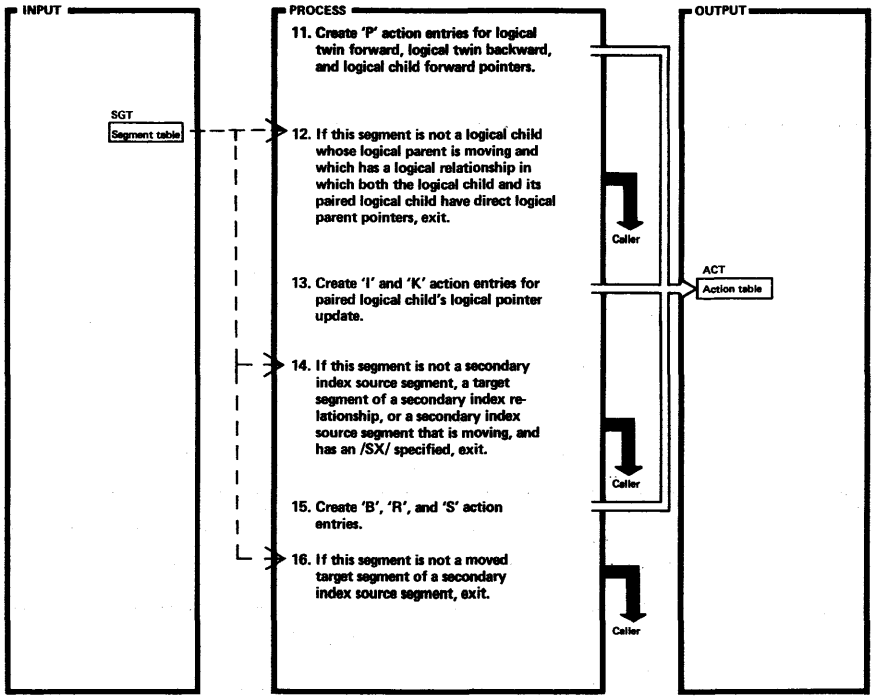
Figure 2-43.2. Action Table Build (Part 2 of 4)



Extended Description	Routine	Label
7.	DLZROW02	
8.	DLZROW03 DLZROW04	

Extended Description	Routine	Label

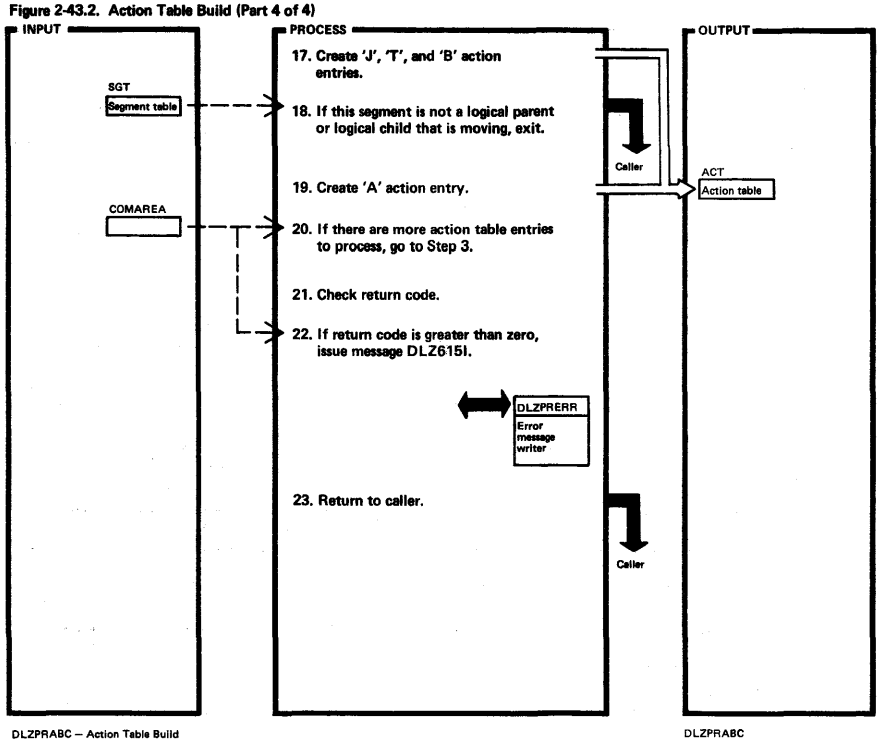
Figure 2-43.2. Action Table Build (Part 3 of 4)



Extended Description	Routine	Label
11.	DLZROW05 DLZROW06	
12.	DLZROW13	
14.	DLZROW16	
16.	DLZROW19 DLZROW20	

Extended Description	Routine	Label

Figure 2-43.2. Action Table Build (Part 4 of 4)



Extended Description	Routine	Label
17.	DLZROW19 DLZROW20	

Extended Description	Routine	Label

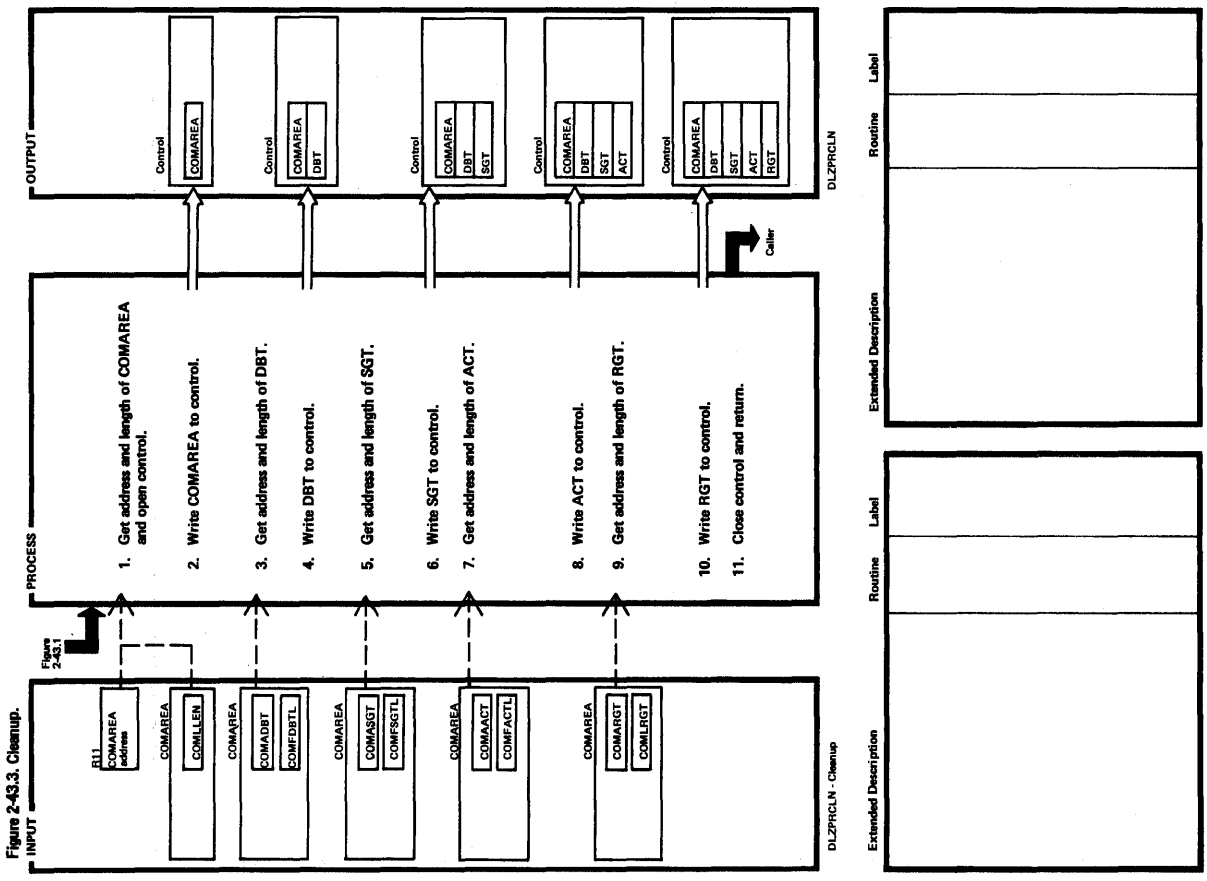
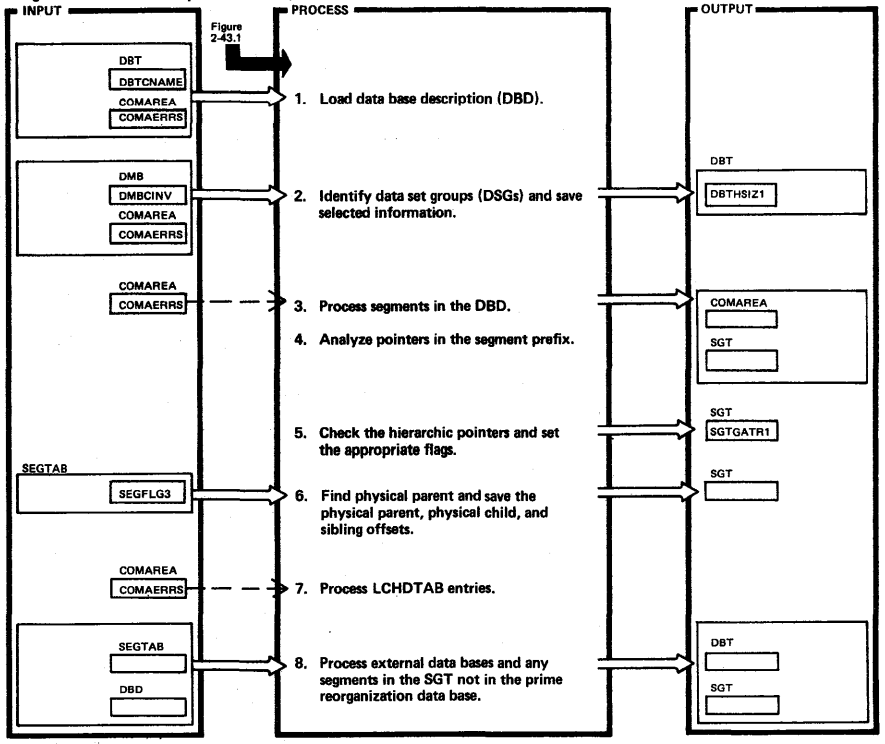


Figure 2-43.3. Cleanup.

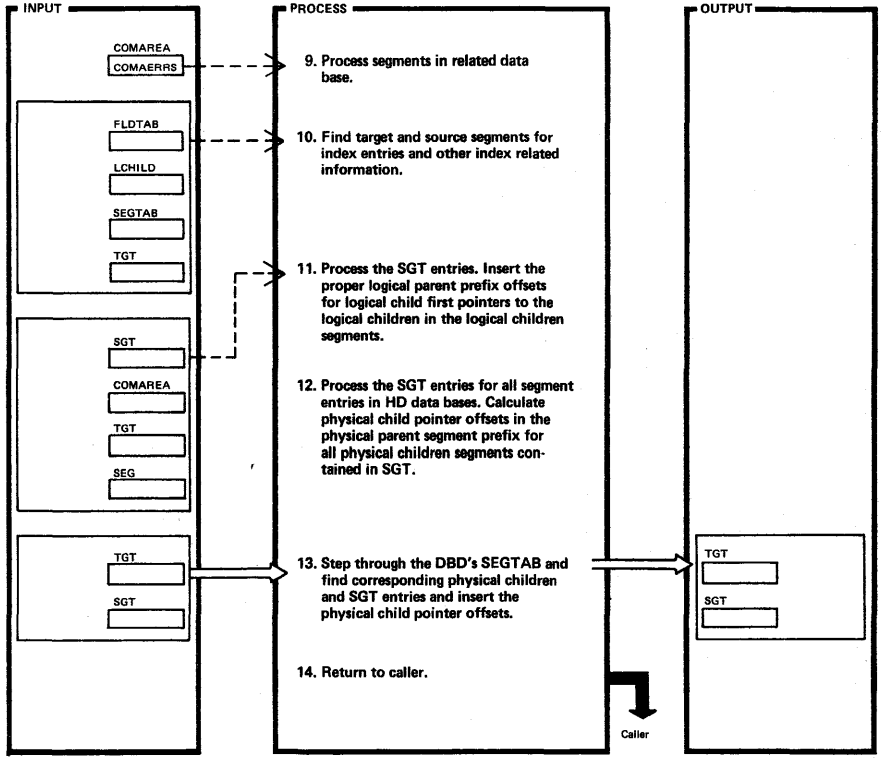
Figure 2-43.4. DBD Analysis (Part 1 of 2)



Extended Description	Routine	Label
1. If DBD is not in core image library, issue message DLZ612I. If DBD is not HD, issue message DLZ613I. If FROMAREA is for HIDAM or KEYRANGE is for HDAM, issue message DLZ604I. If HI/LO block number is not in RAA, issue message DLZ645I.	DLZPRERR	
2. If error, issue message DLZ614I.	DLZPRERR	
3. If error, issue message DLZ618I.	DLZPRERR	
7. If error, issue message DLZ616I, DLZ617I, or DLZ618I.	DLZPRERR	

Extended Description	Routine	Label

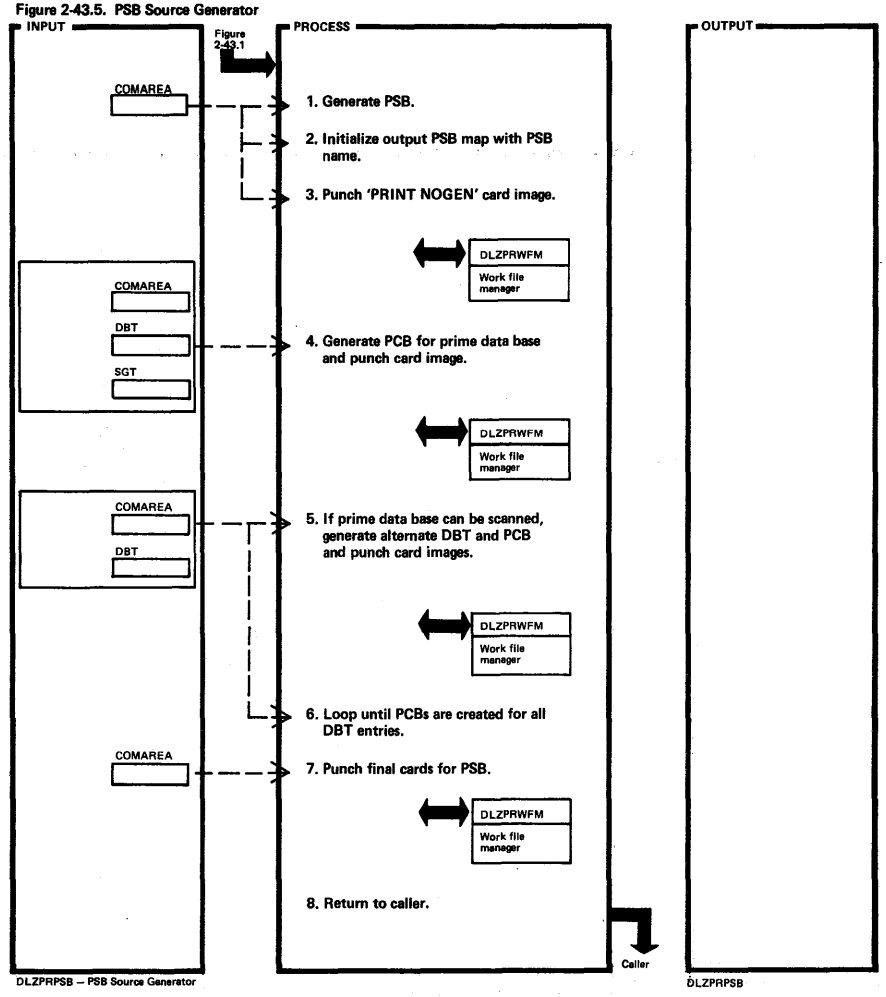
Figure 2-43.4. DBD Analysis (Part 2 of 2)



Extended Description	Routine	Label
9. If error, issue message DLZ615I.	DLZPRERR	
11. If error, issue message DLZ616I.	DLZPRERR	
14. R15 contains the return code. If an error message has been issued, R15 does not equal zero; if an error message has not been issued, R15 equals zero.		

Extended Description	Routine	Label

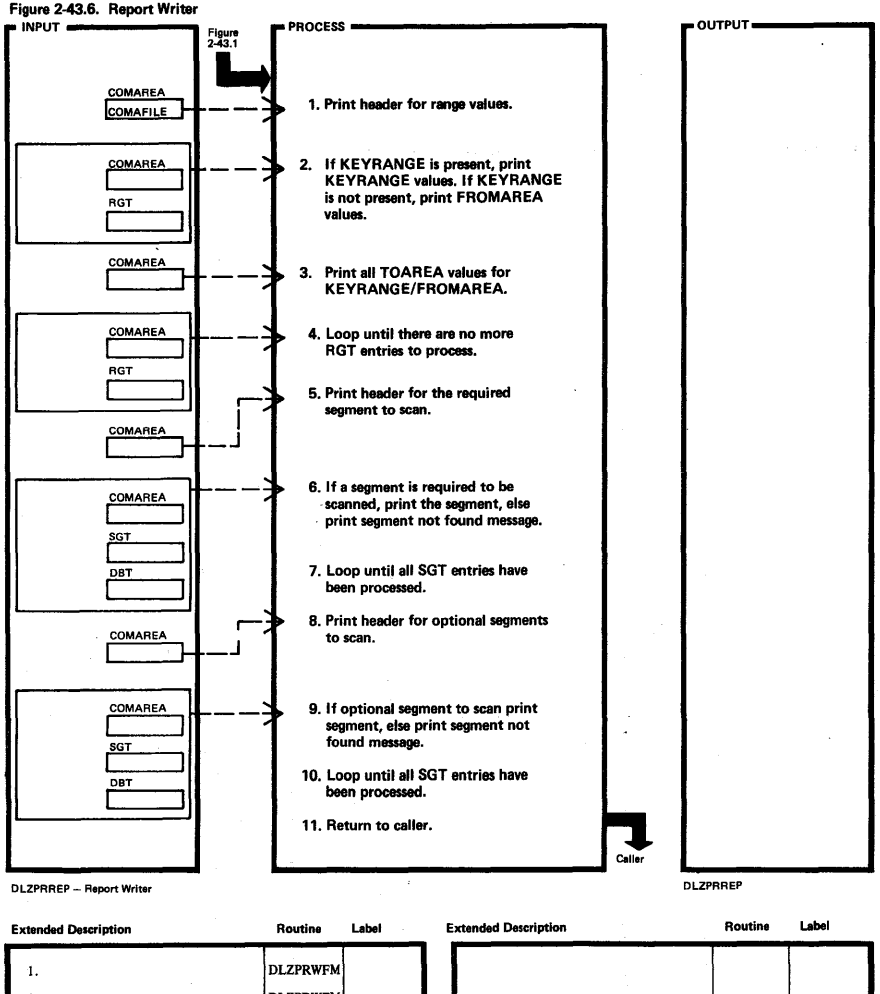
Figure 2-43.5. PSB Source Generator



Extended Description		Label
1. If PSB is not generated, issue message DLZ6271 and return to caller.	DLZPRERR	

Extended Description	Routine	Label

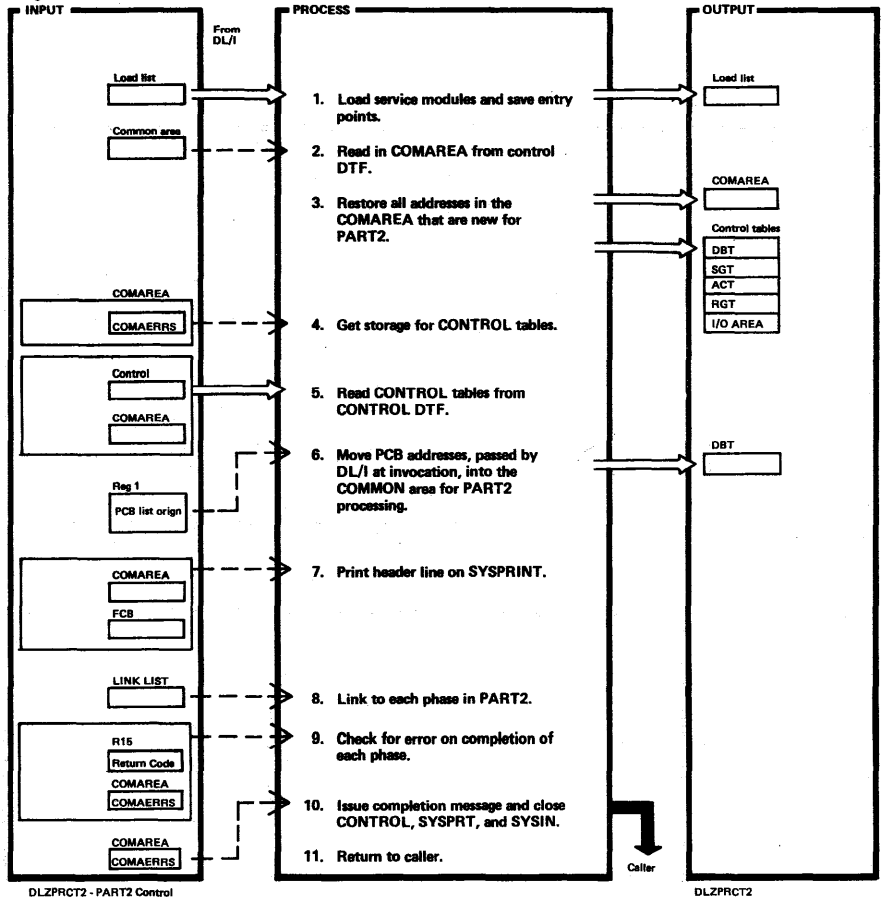
Figure 2-43.6. Report Writer



Extended Description	Routine	Label
1.	DLZPRWFM	
2.	DLZPRWFM	
3.	DLZPRWFM	
5.	DLZPRWFM	
6.	DLZPRWFM	
9.	DLZPRWFM	

Extended Description	Routine	Label

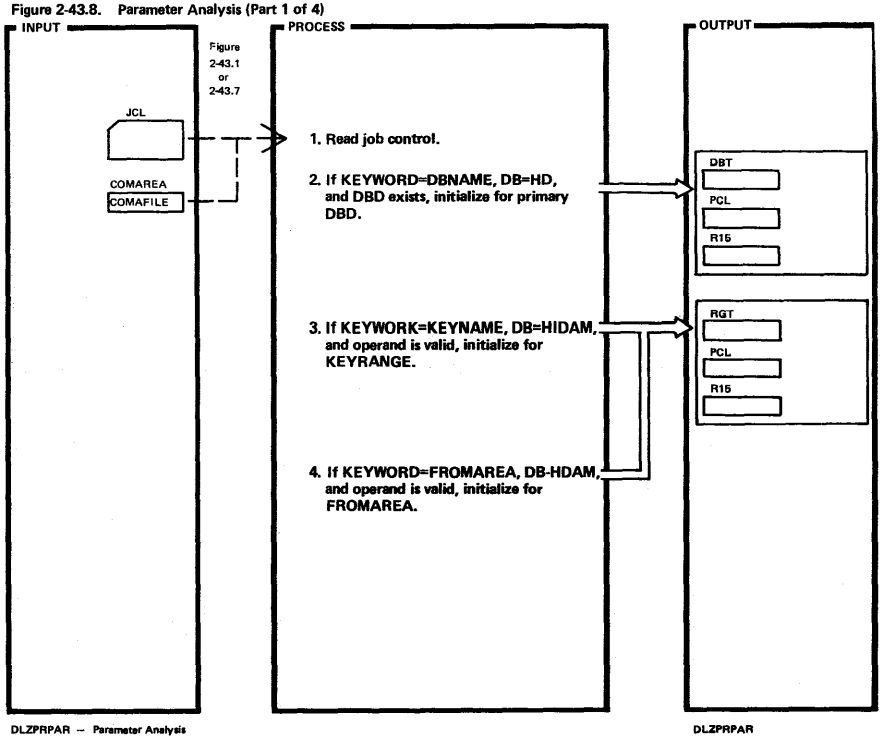
Figure 2-43.7. PART2 Control



Extended Description	Routine	Label
1. Load list includes: DLZPRWFM DLZPRERR DLZPRDLI DLZPRSTW		
4. DOS/VS GETVIS is issued. If an error occurs, issue message DLZ639I and return.	DLZPRERR	
5. If error, issue message DLZ634I and return.	DLZPRERR	

Extended Description	Routine	Label
8. Link list includes: DLZPRPAR DLZPRURC DLZPRSCC DLZPRSTC DLZPRUPD		
9. If error, issue message DLZ636I, close data sets, and return.	DLZPRERR	
10. Issue message DLZ636I.	DLZPRERR	

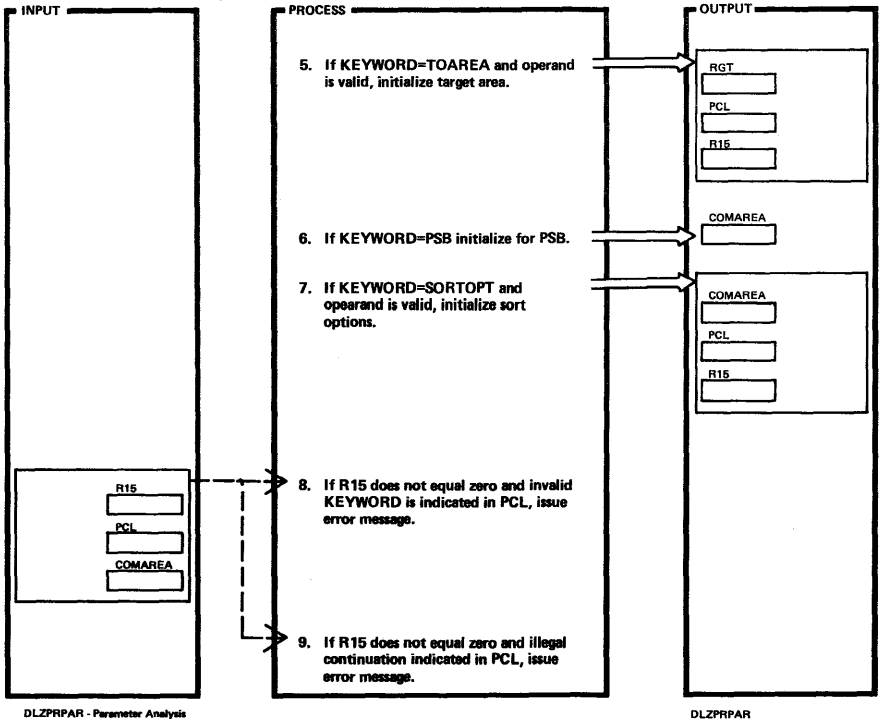
Figure 2-43.8. Parameter Analysis (Part 1 of 4)



Extended Description	Routine	Label
1.	DLSPRWFM	
2, 3, 4 If error occurs, set R15 not equal to zero and PCL (PARM Control Table) equal to type of error.		

Extended Description	Routine	Label

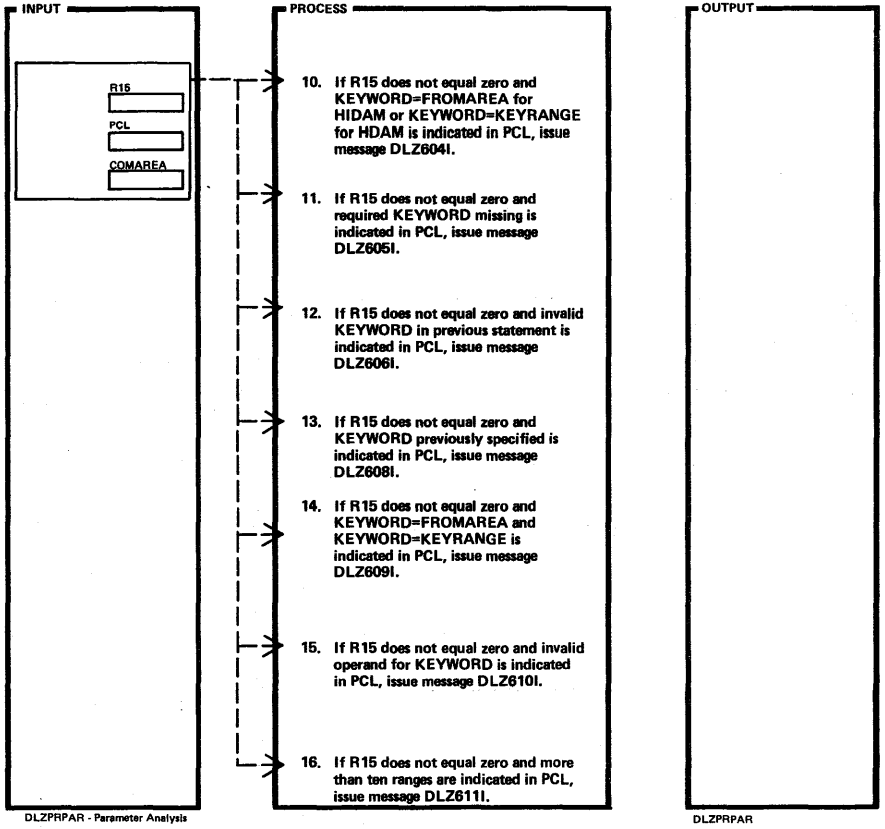
Figure 2-43.8. Parameter Analysis (Part 2 of 4)



Extended Description	Routine	Label
5, 7 If error occurs, set R15 not equal to zero and PCL equal to type of error.		
8. If error, issue error message DLZ602I.	DLZPRERR	
9. If error, issue error message DLZ603I.	DLZPRERR	

Extended Description	Routine	Label

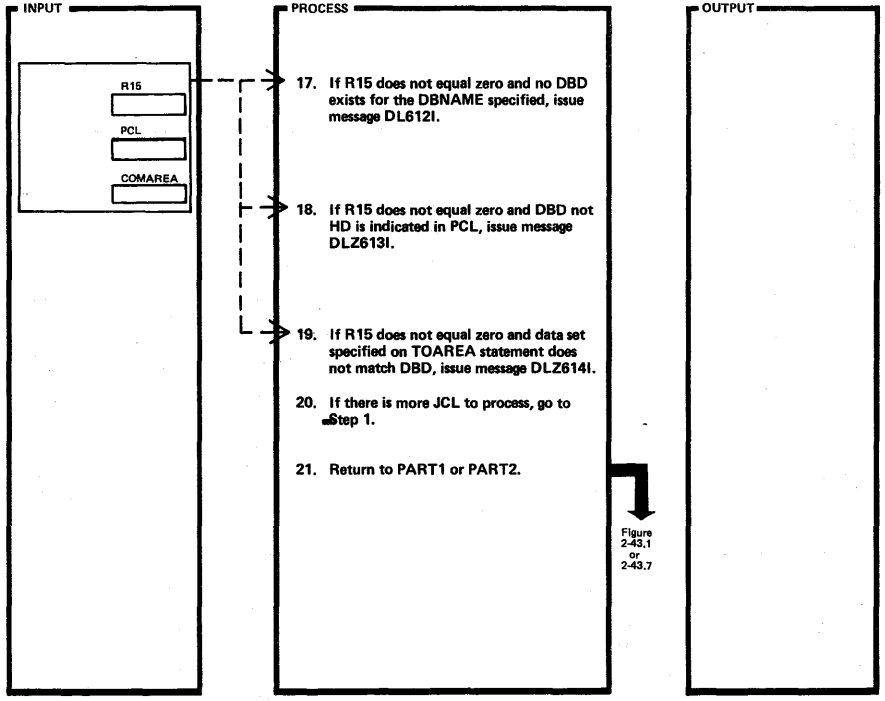
Figure 2-43.8. Parameter Analysis (Part 3 of 4)



Extended Description	Routine	Label
10.	DLZPRERR	
11.	DLZPRERR	
12.	DLZPRERR	
13.	DLZPRERR	
14.	DLZPRERR	
15.	DLZPRERR	
16.	DLZPRERR	

Extended Description	Routine	Label

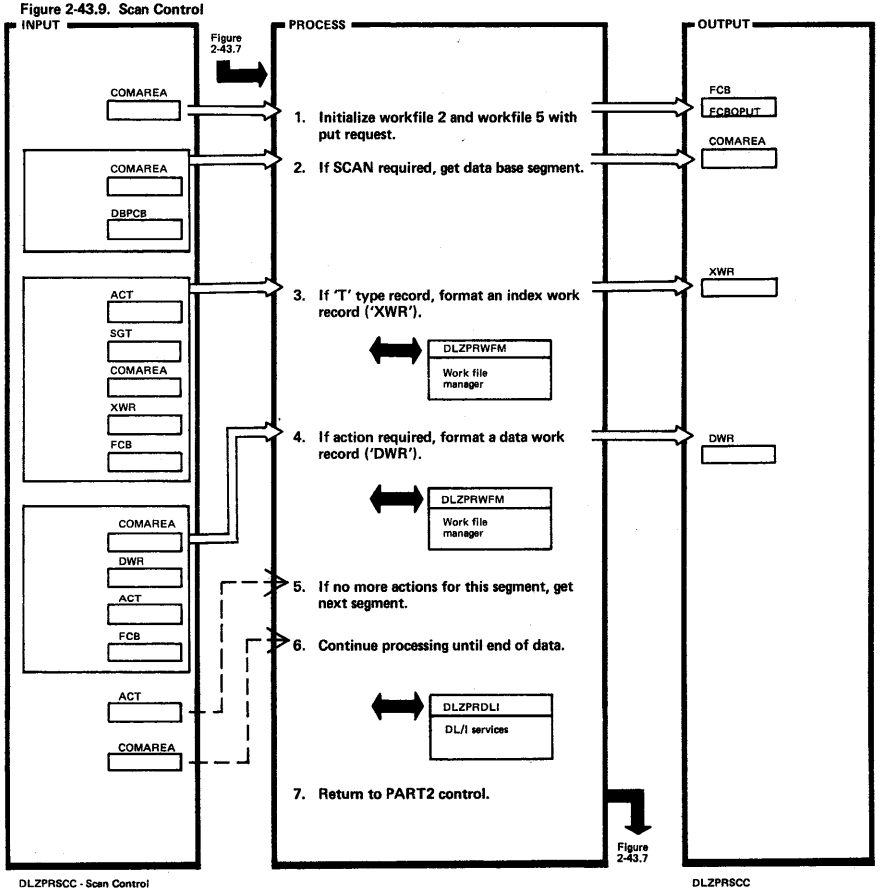
Figure 2-43.8. Parameter Analysis (Part 4 of 4)



Extended Description	Routine	Label
17.	DLZPRERR	
18.	DLZPRERR	
19.	DLZPRERR	

Extended Description	Routine	Label

Figure 2-43.9. Scan Control



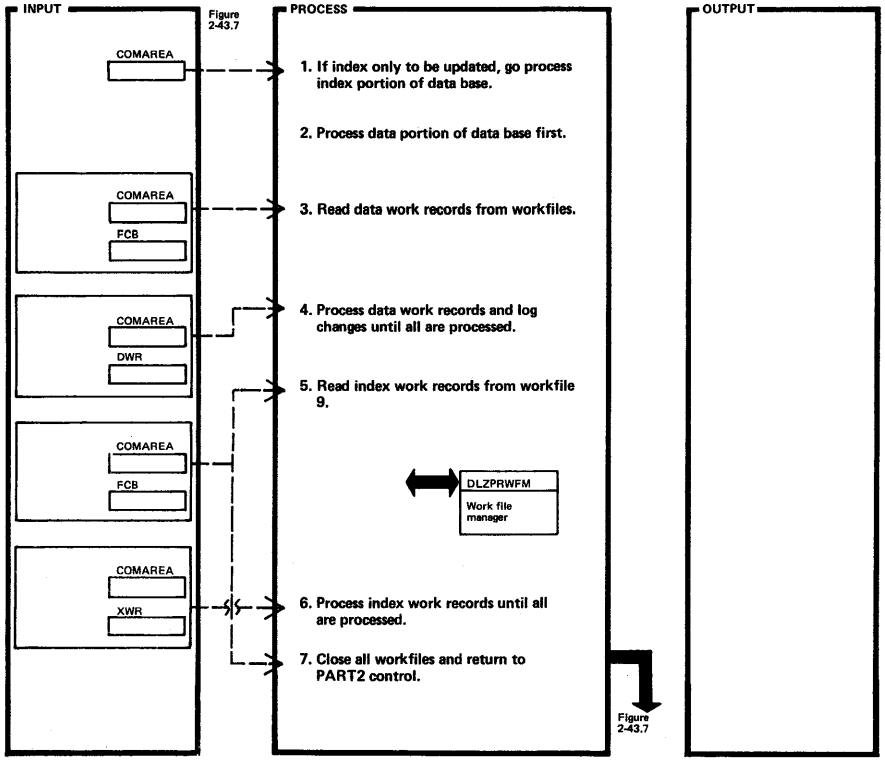
DLZPRSCC - Scan Control

DLZPRSCC

Extended Description	Routine	Label
1. If bad return from DL/I, issue message DLZ653I.	ASMTDLI DLZPRERR	
6. If error, issue message DLZPR653I.	DLZPRERR	

Extended Description	Routine	Label

Figure 2-43.10. Update Prefix



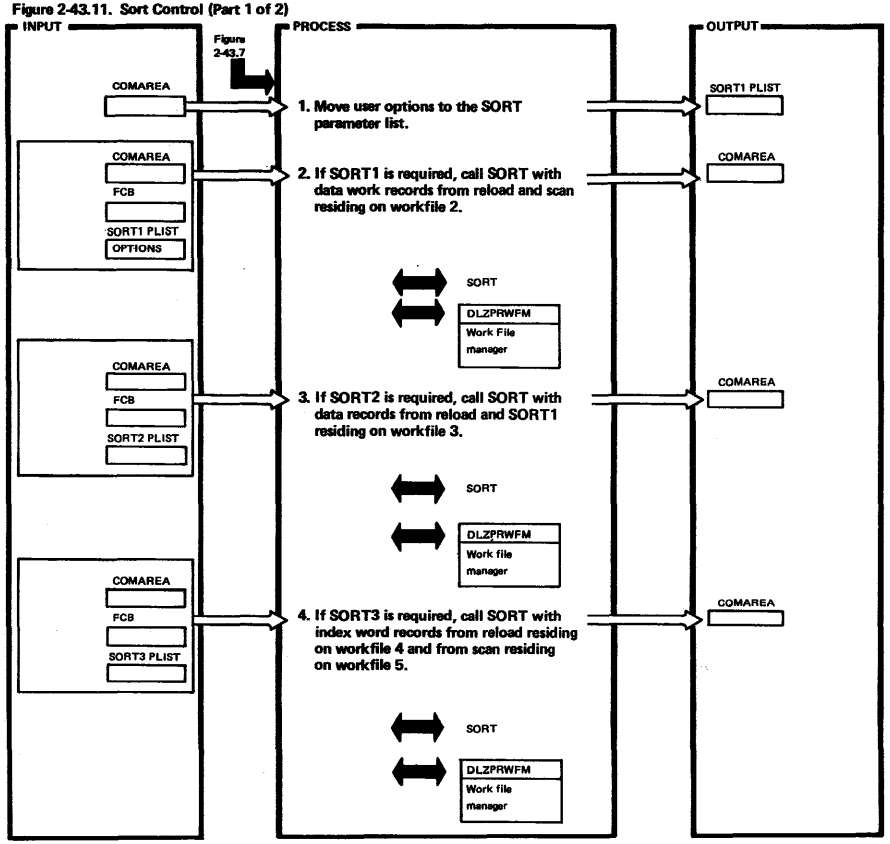
DLZPRUPD – Update Prefix.

DLZPRUPD

Extended Description	Routine	Label
1.	PROCINDX	
2.	PROCDATA	
3. Workfile 7, Workfile 3.	DLZPRWFM	
4. If error, issue message DLZ650I or DLZ653I.	ASMTDLI DLZPRDLI DLZPRERR	
6. If error, issue message DLZ659I or DLZ653I.	ASMTDLI DLZPRDLI DLZPRERR	
7.	DLZPRERR	

Extended Description	Routine	Label

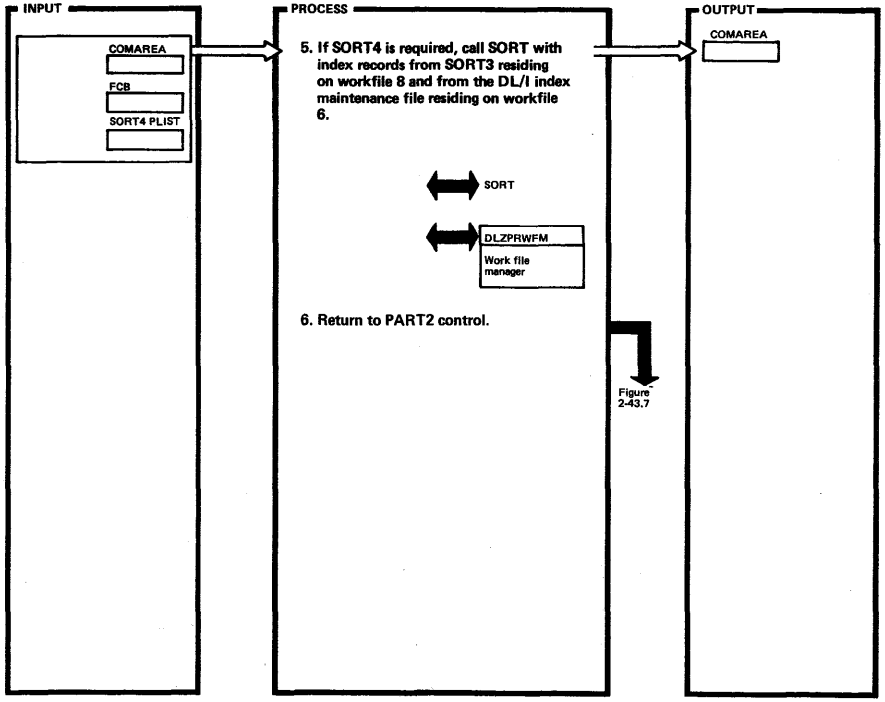
Figure 2-43.11. Sort Control (Part 1 of 2)



Extended Description	Routine	Label
2, 3, 4 If error, issue message DLZ647I.	DLZPRERR	
2.	SORT1REQ	
3. SORT1 and SORT2 process data work records exclusively. Input to SORT1 is from reload and scan, and input for SORT2 is from reload and SORT1. Together these routines save the new RBA of the segment moved in the associated work records and arrange them in physical sequence as they exist in the data bases.		

Extended Description	Routine	Label

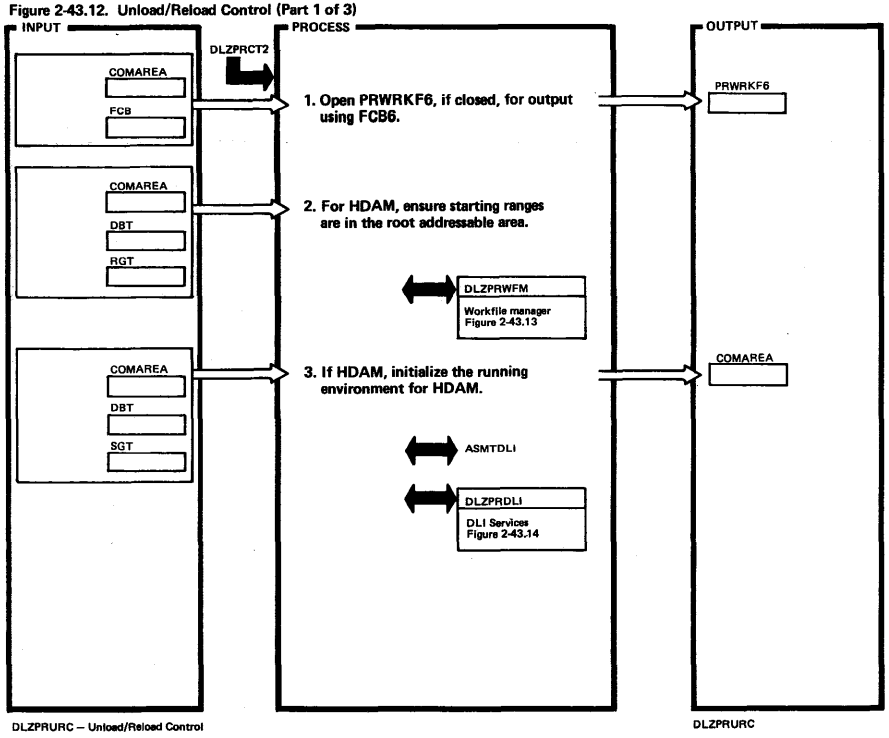
Figure 2-43.11. Sort Control (Part 2 of 2)



Extended Description	Routine	Label
4.	SORT3REQ	
5. If error, issue message DLZ647I, DLZ648I, or DLZ649I. SORT3 and SORT4 process index work records exclusively. Input to SORT3 is from reload and scan, and input to SORT4 is from the DL/I index maintenance file and SORT3. Together these routines eliminate index work records that are not involved in update, convert the DL/I index maintenance records into partial reorganization format, and arrange the index work records in physical sequence.	DLZPRERR SORT4REQ	

Extended Description	Routine	Label

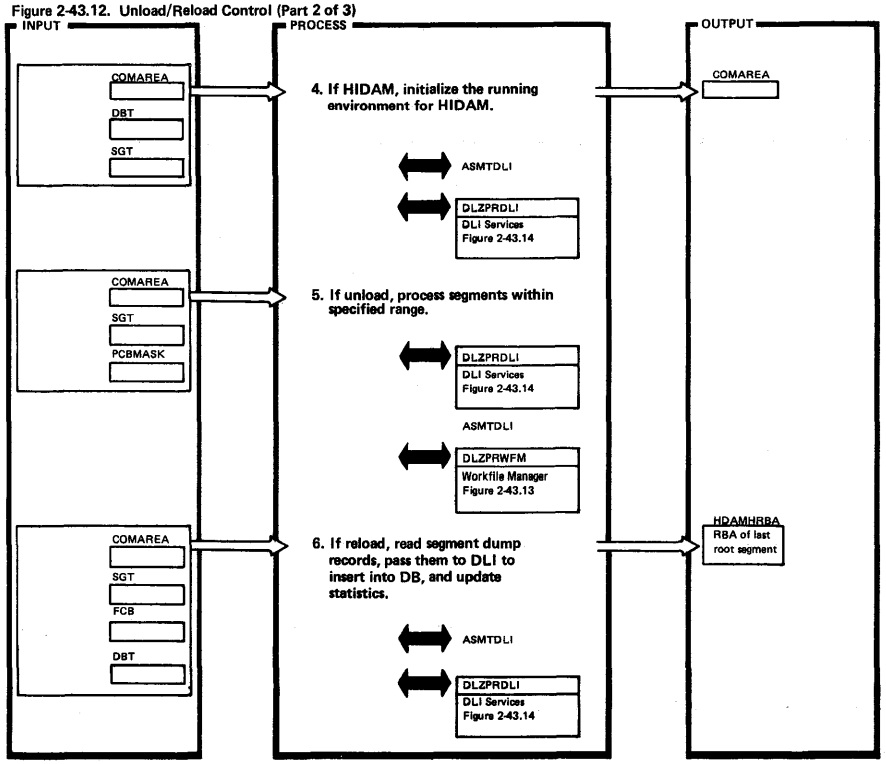
Figure 2-43.12. Unload/Reload Control (Part 1 of 3)



Extended Description	Routine	Label
1. If error, return to caller with return code greater than zero.		
2. DPRWFMIF is the interface routine to Workfile Manager (DLZPRWFM). When I/O required, checks return code and saves the highest code.	DPRWFMIF	
3. DPRDLISV is the interface routine to DLI Services (DLZPRDLI). On return, checks the return code and saves the highest code.	DPRDLISV	
Open data base, set end process values in common area, check the RGT, and position DB at the beginning of the range.	DPRHDAMI	
If error, issue message DLZ646I.	DLZPRERR	

Extended Description	Routine	Label

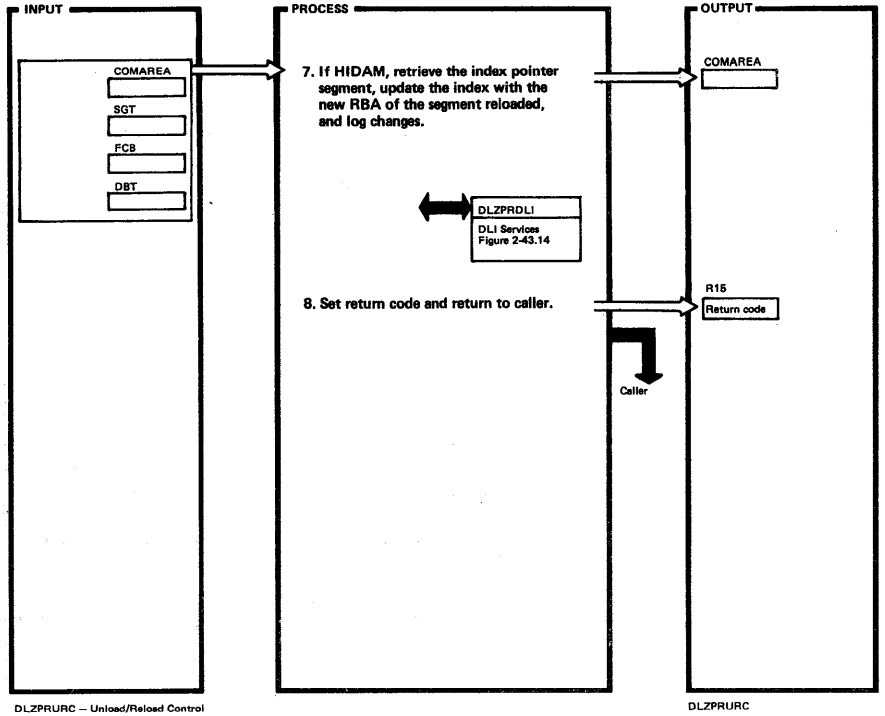
Figure 2-43.12. Unload/Reload Control (Part 2 of 3)



Extended Description	Routine	Label
4. DPRDLISV is the interface routine to DLI Services (DLZPRDLI). On return, checks the return code and saves the highest code.	DPRDLISV	
Opens data base, sets end process test values in common area, and positions DB at beginning of range.	DPRHIDMI	
If error, issue message DLZ646I.	DLZPRERR	
5. Unload segment to dump file, free space occupied by each unloaded segment, and update the statistics information for the unload processing.		
DPRDLERR is the interface routine to error message writer (DLZPRERR).	DPRDLERR	
If error, issue message DLZ653I.	DLZPRERR	

Extended Description	Routine	Label
6. DPRDLISV is the interface routine to DLI Services (DLZPRDLI). On return checks the return code and saves the highest code.	DPRDLISV	
DPRDLERR is the interface routine to Error message writer (DLZPRERR).	DPRDLERR	
If error, issue message DLZ653I.	DLZPRERR	

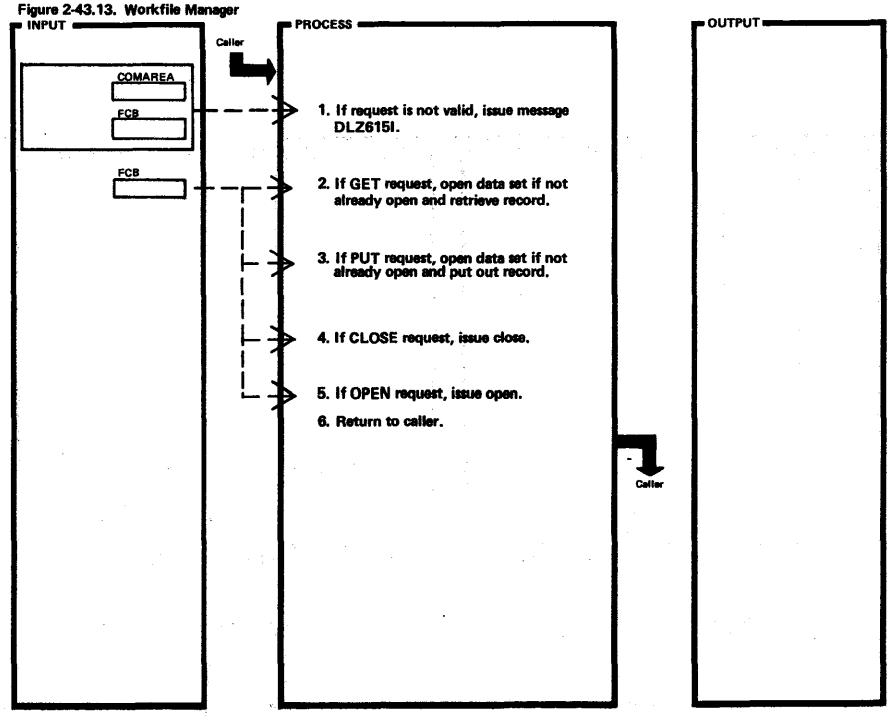
Figure 2-43.12. Unload/Reload Control (Part 3 of 3)



Extended Description	Routine	Label
7. DPRDLISV is the interface module to DLI Services (DLZPRDLI). On return, checks the return code and saves the highest code. If error, issue message DLZ643I or DLZ644I.	DPRDLISV DPRPRXUP DLZPRERR	

Extended Description	Routine	Label

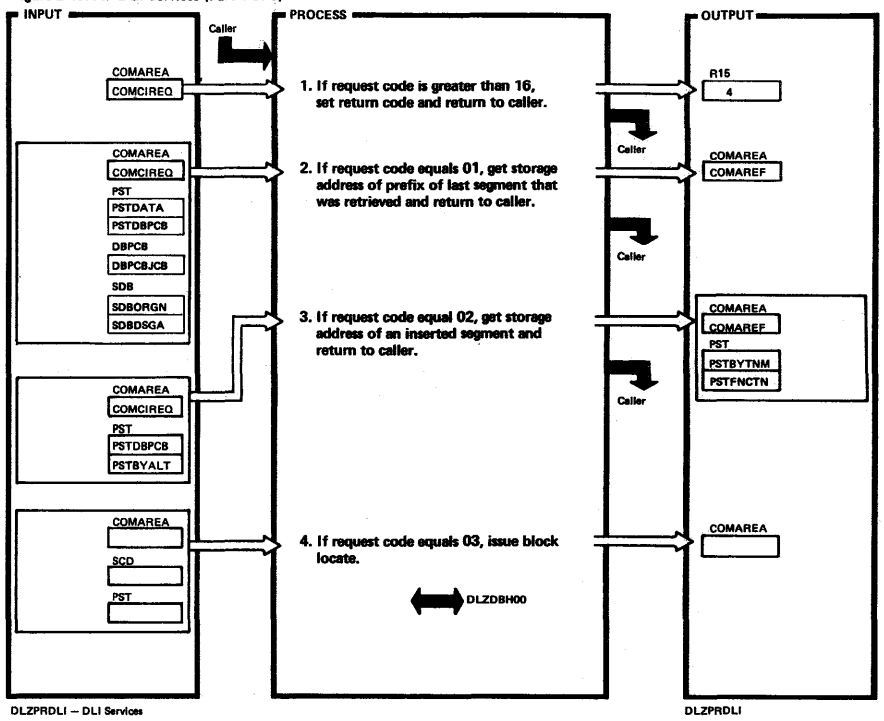
Figure 2-43.13. Workfile Manager



Extended Description	Routine	Label
1. 2, 3, 4, 5: R7 contains the address of the DTF.	DLZPRERR	

Extended Description	Routine	Label

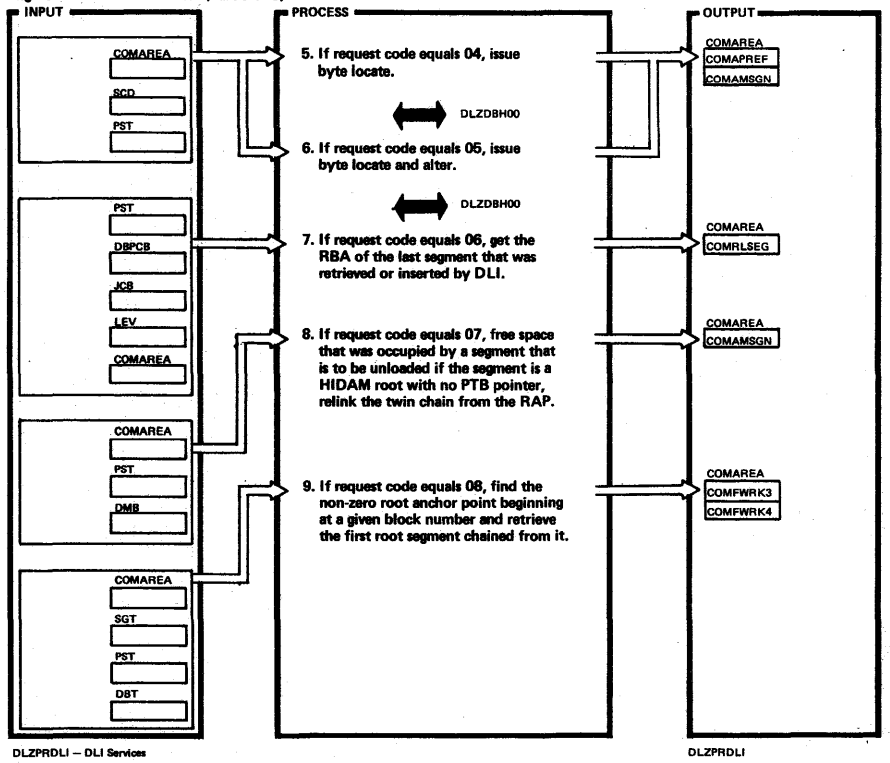
Figure 2-43.14. DLI Services (Part 1 of 5)



Extended Description	Routine	Label
2. A 'GU' or 'GNP' call must be the last regular call to DLI before making this request.	DLI01	
3. The call to DLI immediately preceding this call must have been an 'ISRT' call to the primary DB.	DLI02	
4. If error, issue message DLZ636I or DLZ615I and branch to SCD ABEND routine.	DLI03 DLZPRERR	

Extended Description	Routine	Label

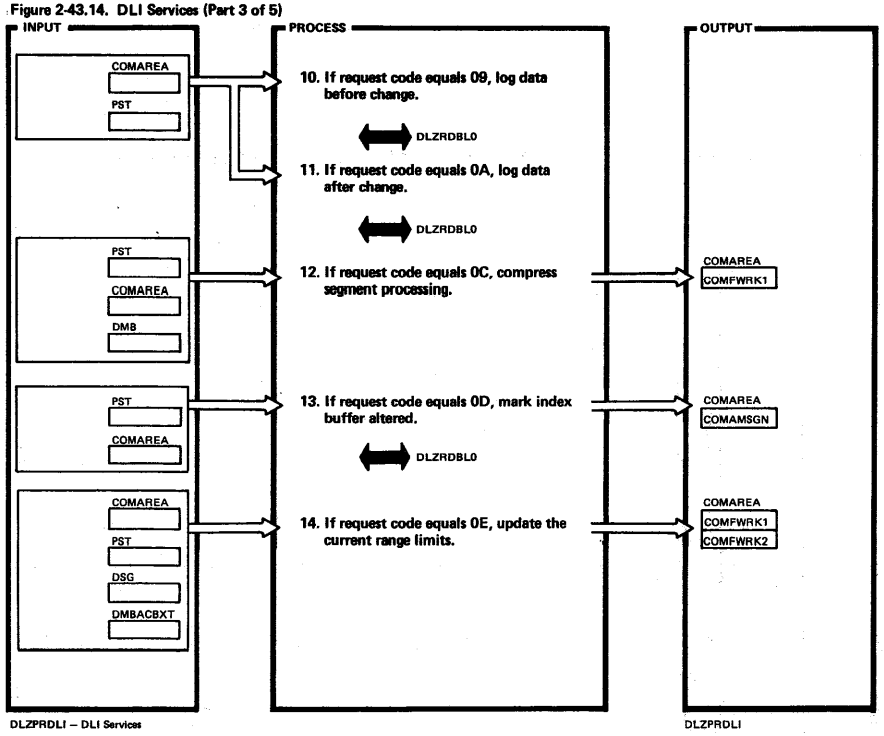
Figure 2-43.14. DLI Services (Part 2 of 5)



Extended Description	Routine	Label
5. If error, issue message DLZ636I or DLZ615I and branch to SCD ABEND routine.	DLZPRERR	
6. If error, issue message DLZ636I or DLZ615I and branch to SCD ABEND routine.	DLZPRERR	
8. Last call to DLI must have been a 'GU' or 'GN' for the prime DB. If error, issue message DLZ655I.	DLI05 DLZFRSP0 DLZRDBL0 DLZPRERR	
9.	DLI03 DLI05	

Extended Description	Routine	Label

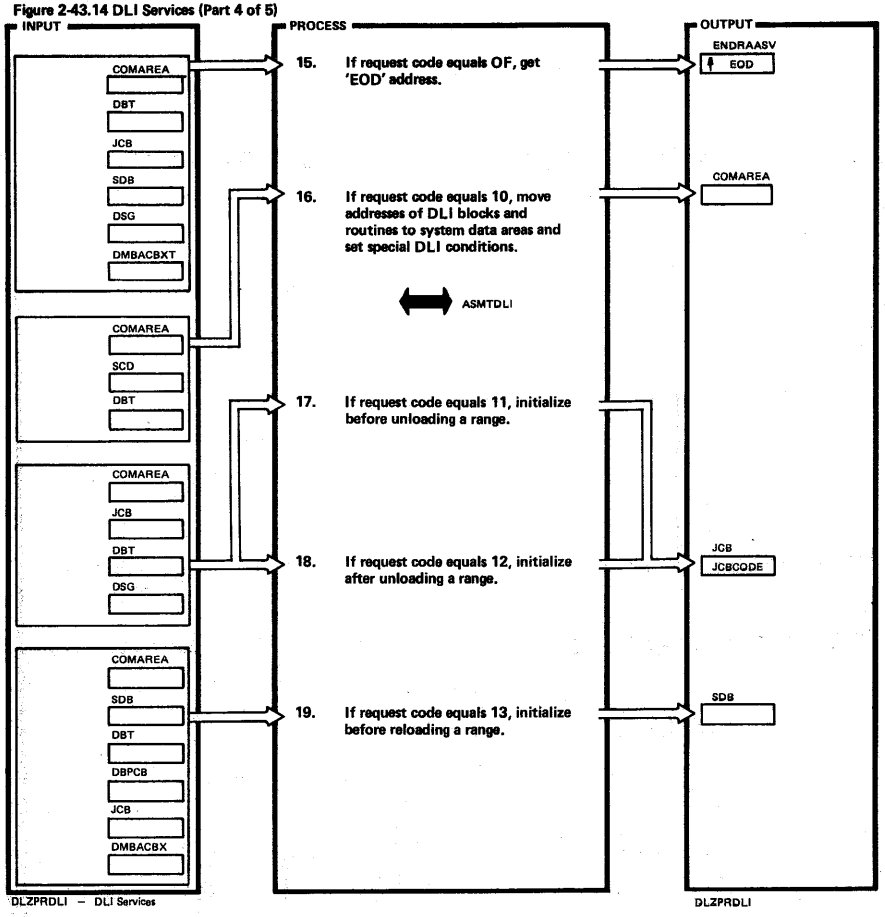
Figure 2-43.14. DLI Services (Part 3 of 5)



Extended Description	Routine	Label
10. Request must follow request type 01 (get prefix address) or request type 05 (byte locate and mark buffer altered) or request type 16 (retrieve index by key).	DLI09	
11. Request must follow request type 09. No intervening calls to DLI via the language interface or to DLZPRDLI are permitted.		
12. Processing is done by the compression routine.		
13. If error, issue message DLZ636I or DLZ615I and branch to SCD ABEND routine.	DLI0D DLZPRERR	
14. For each segment to be unloaded, determine if it extends the range of blocks being unloaded. If it does, perform this processing step.	DLI0E	

Extended Description	Routine	Label

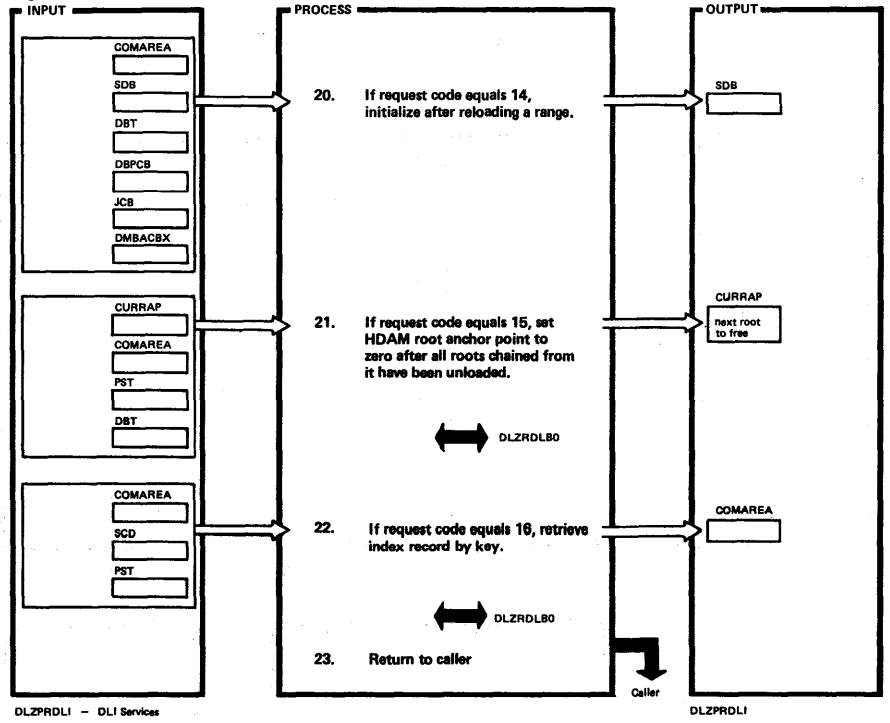
Figure 2-43.14 DLI Services (Part 4 of 5)



Extended Description	Routine	Label
15.	DLI0F	
16. DLI control block addresses are acquired by a GSCD call.	DLI10	
If error, issue message DLZ651I	DLZPRERR	
17.	DLI11	
18.	DLI12	

Extended Description	Routine	Label

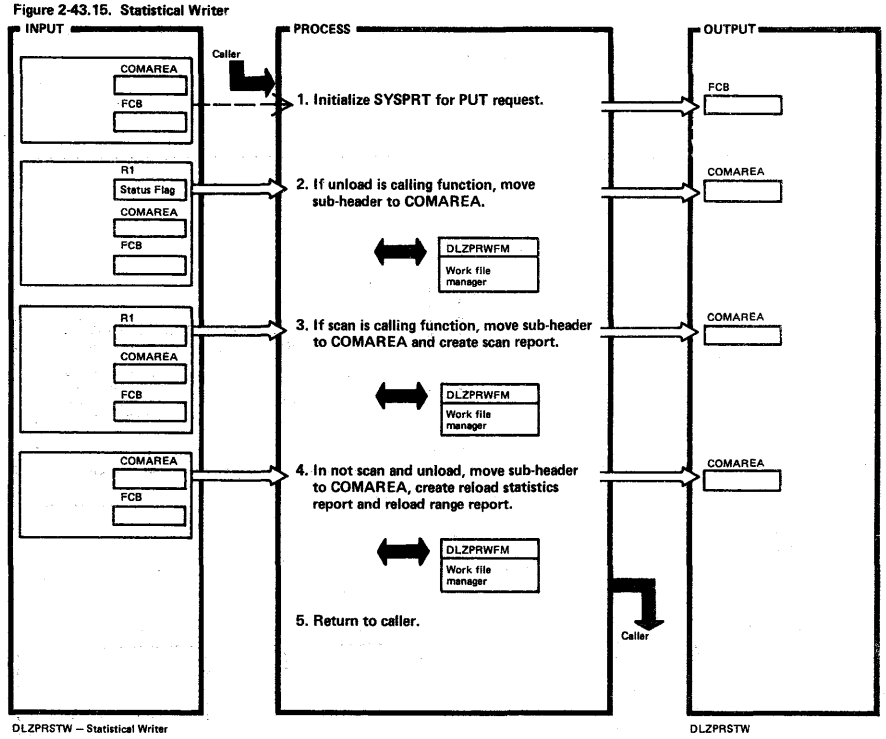
Figure 2-43.14. DLI Services (Part 5 of 5)



Extended Description	Routine	Label
20.	DLI14	
21. First call for each range must be preceded by request 08 for first block and anchor point.	DLI05 DLI03 DLI15	
22. If error, issue message DLZ636I or DLZ615I and branch to SCD ABEND routine.	DLI16 DLZPRERR	

Extended Description	Routine	Label

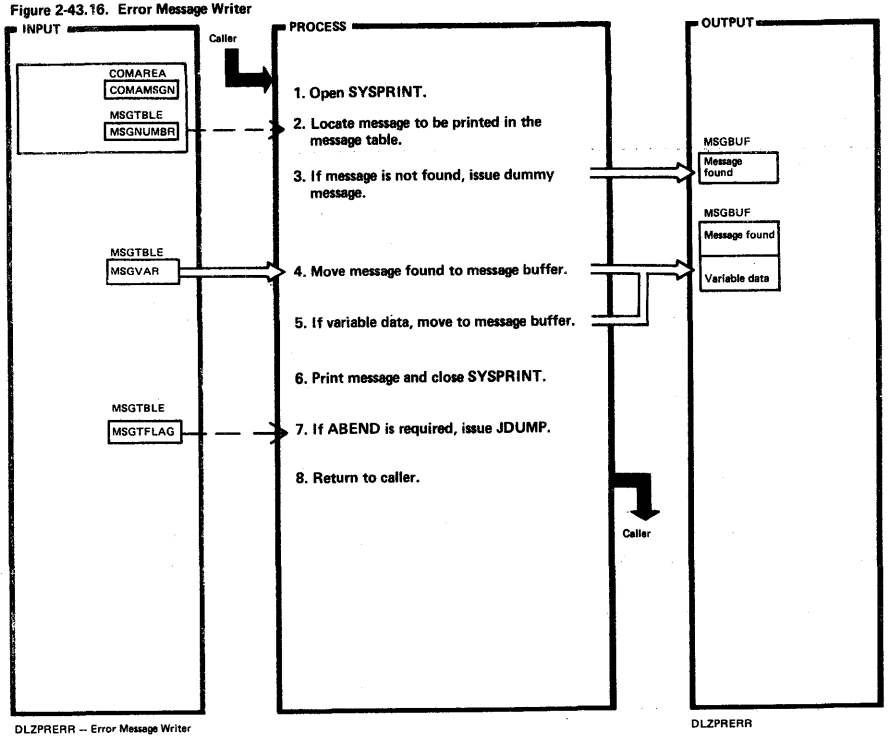
Figure 2-43.15. Statistical Writer



Extended Description	Routine	Label
2. Create unload statistics report, unload range report, unload distribution report and return to mainline processing.	DLZPRWFM	
4. If error, issue message DLZ642I.	DLZPRERR	

Extended Description	Routine	Label

Figure 2-43.16. Error Message Writer



Extended Description	Routine	Label

Extended Description	Routine	Label

Figure 2-44.1. HLPI (PL/I Online Control Flow-CICS/VS)

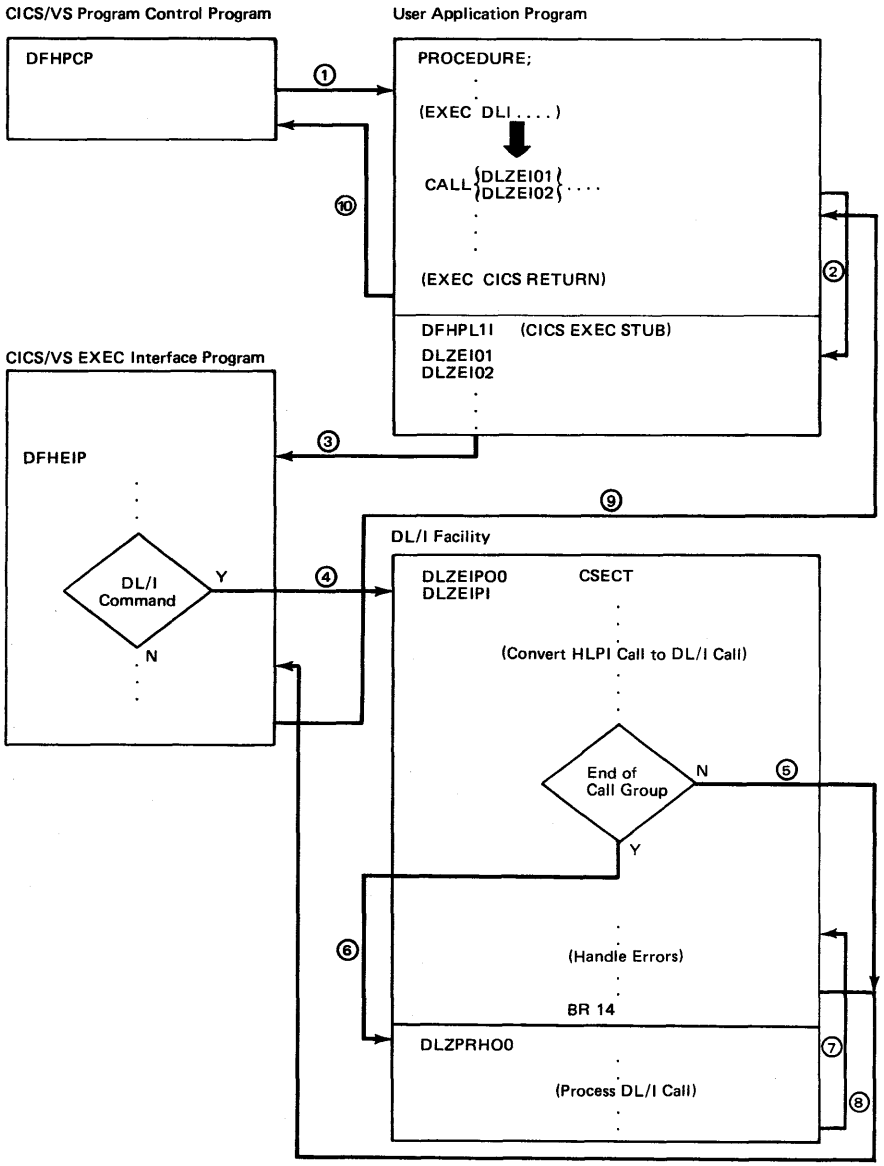


Figure 2-44.2. HLPI (COBOL Online Control Flow-CICS/VS)

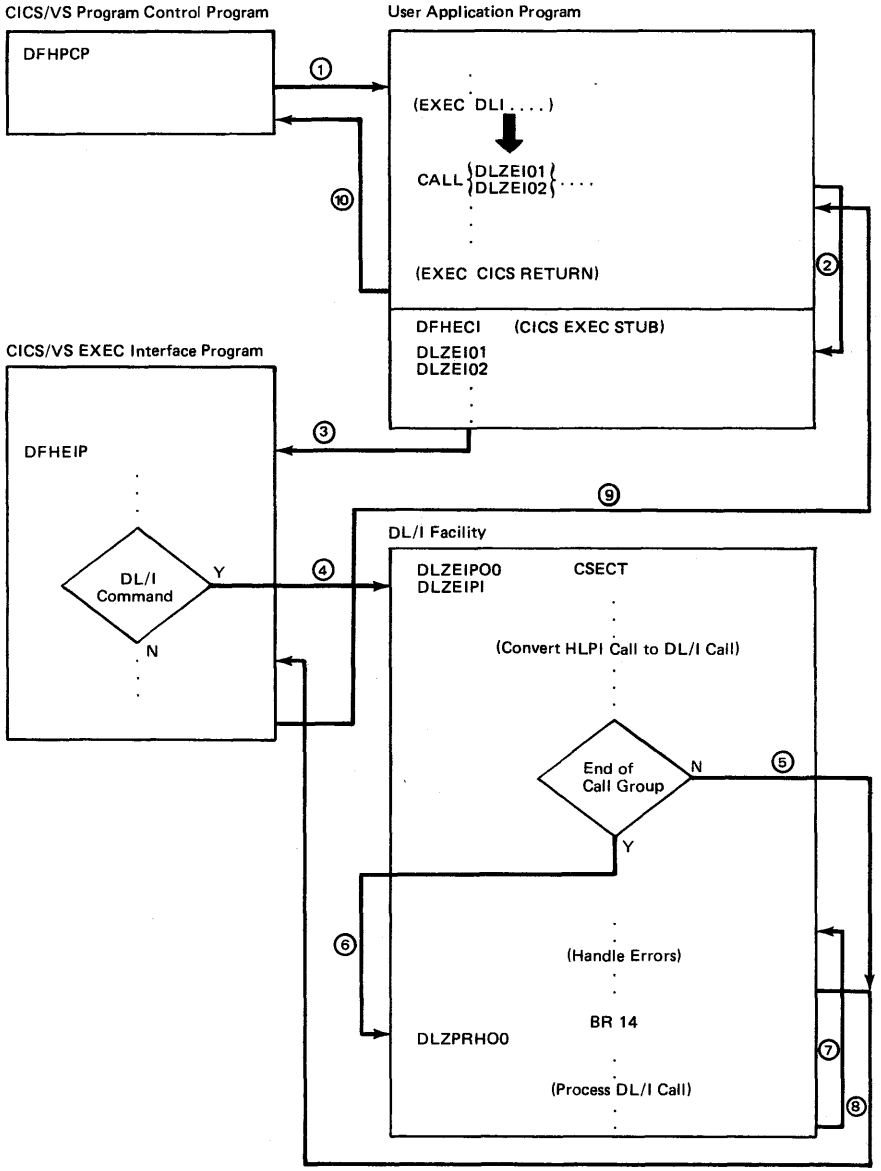


Figure 2-44.3. HLPI (PL/I MPS Batch Control Flow)

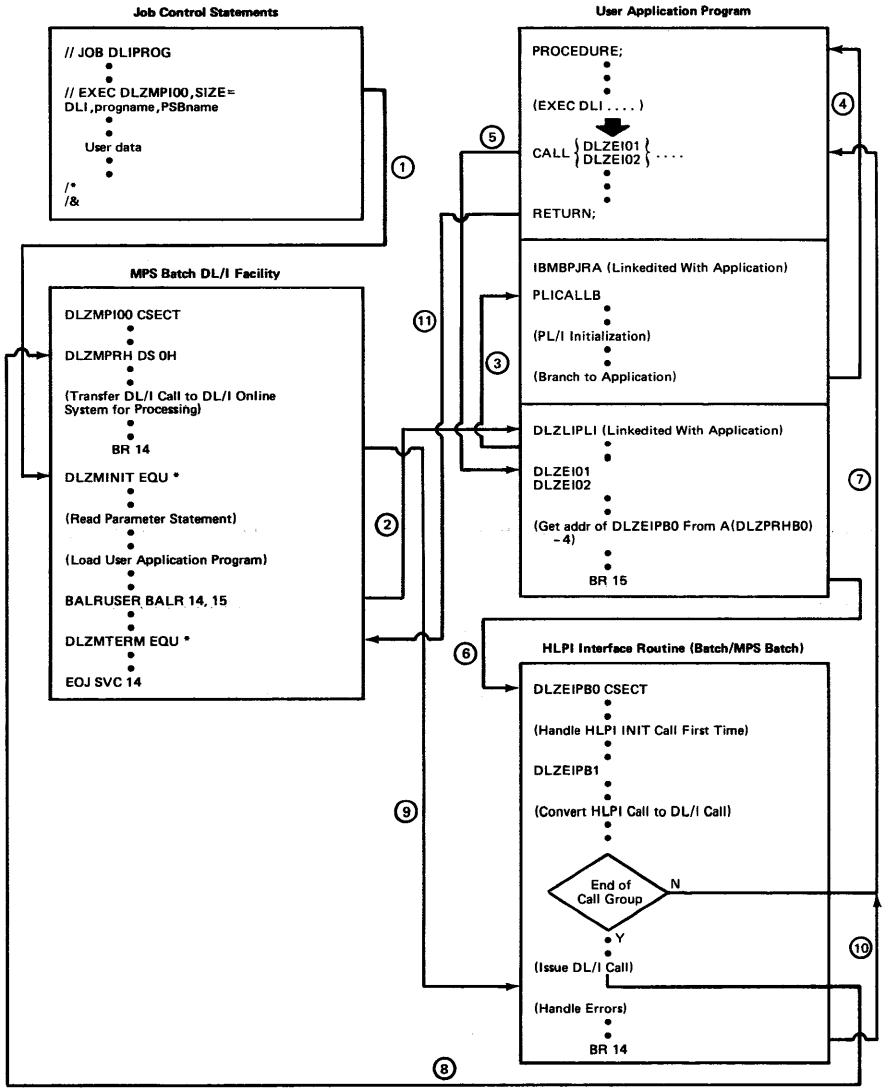


Figure 2-44.4. HLPI (COBOL MPS Batch Control Flow)

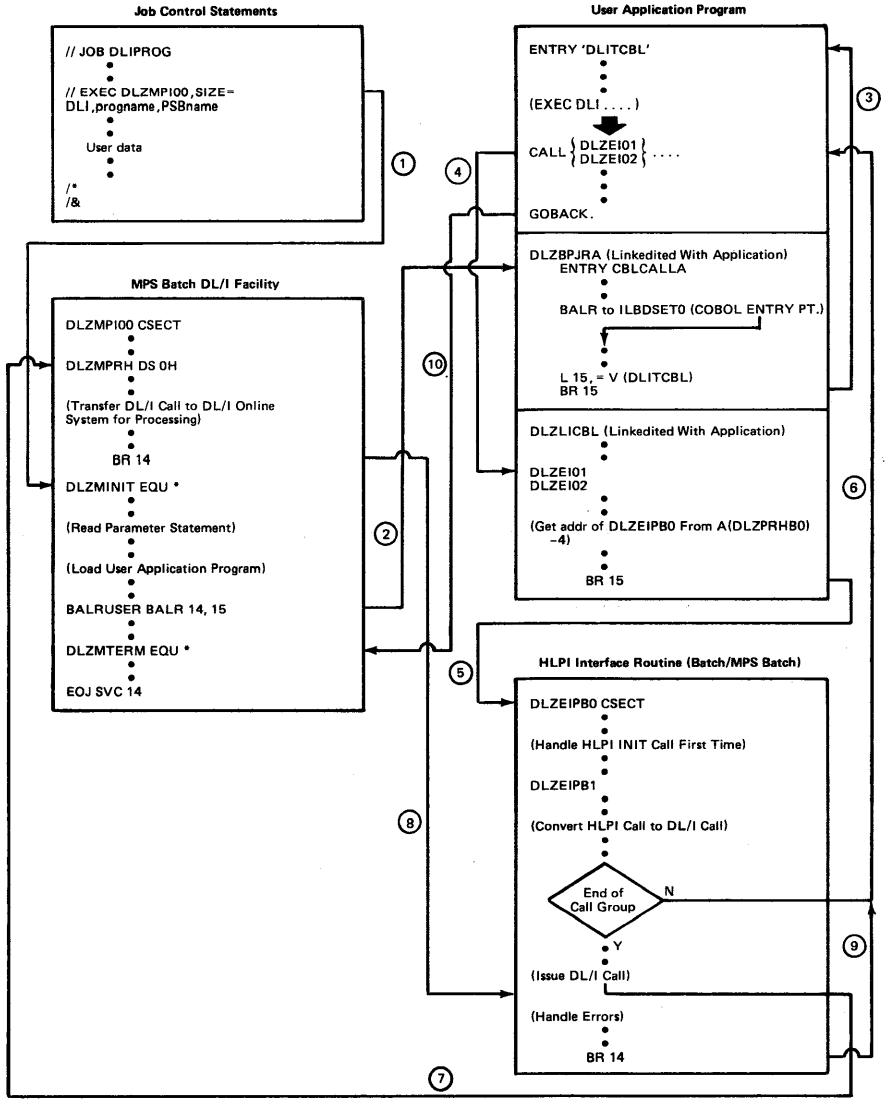


Figure 2-44.5. HLPI (PL/I Batch Control Flow)

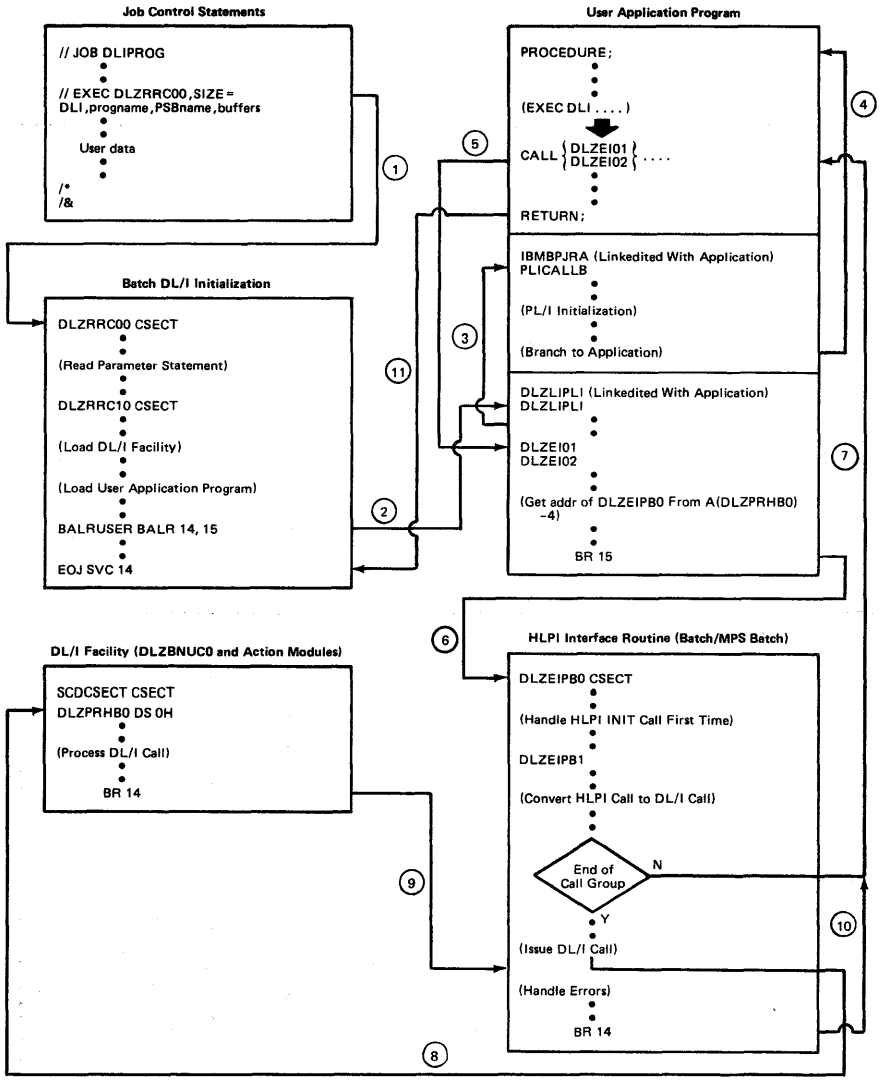


Figure 2-44.6. HLPI (COBOL Batch Control Flow)

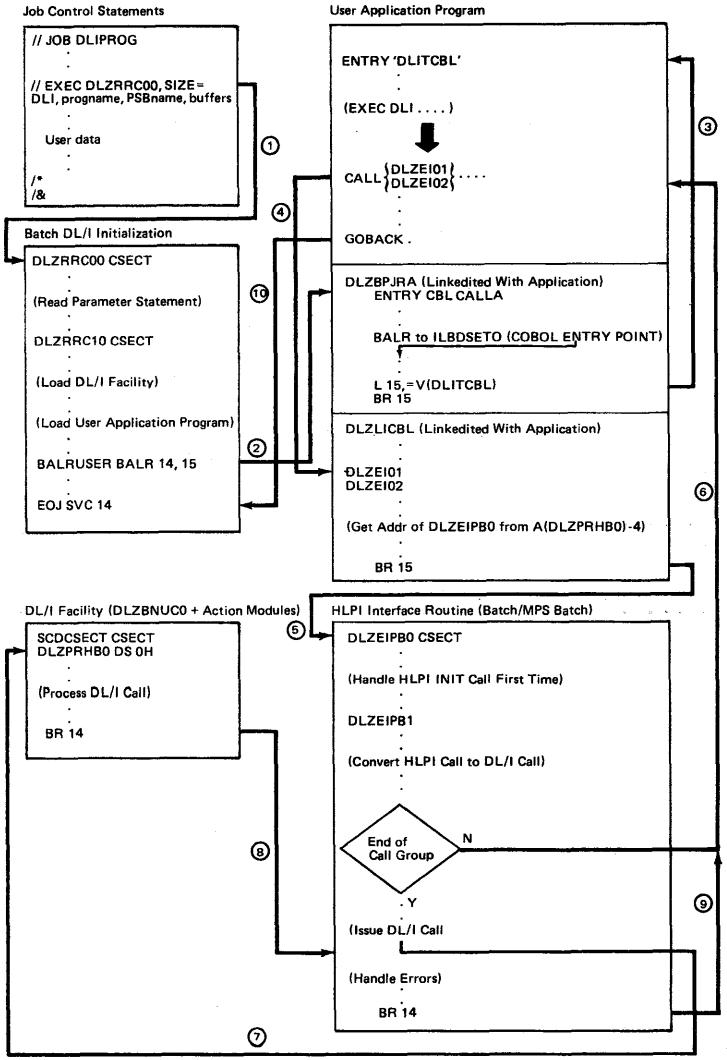
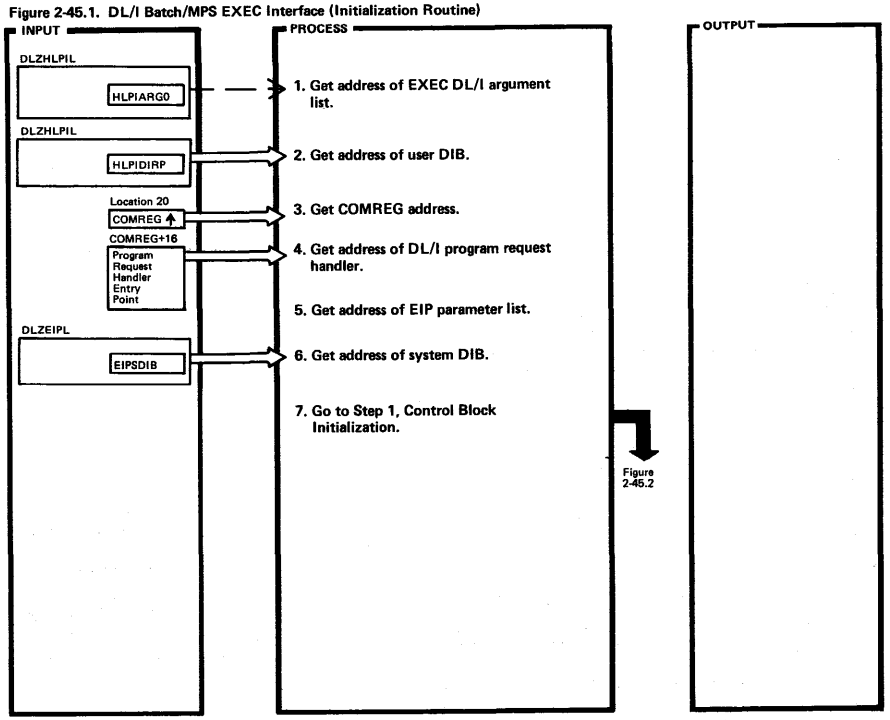


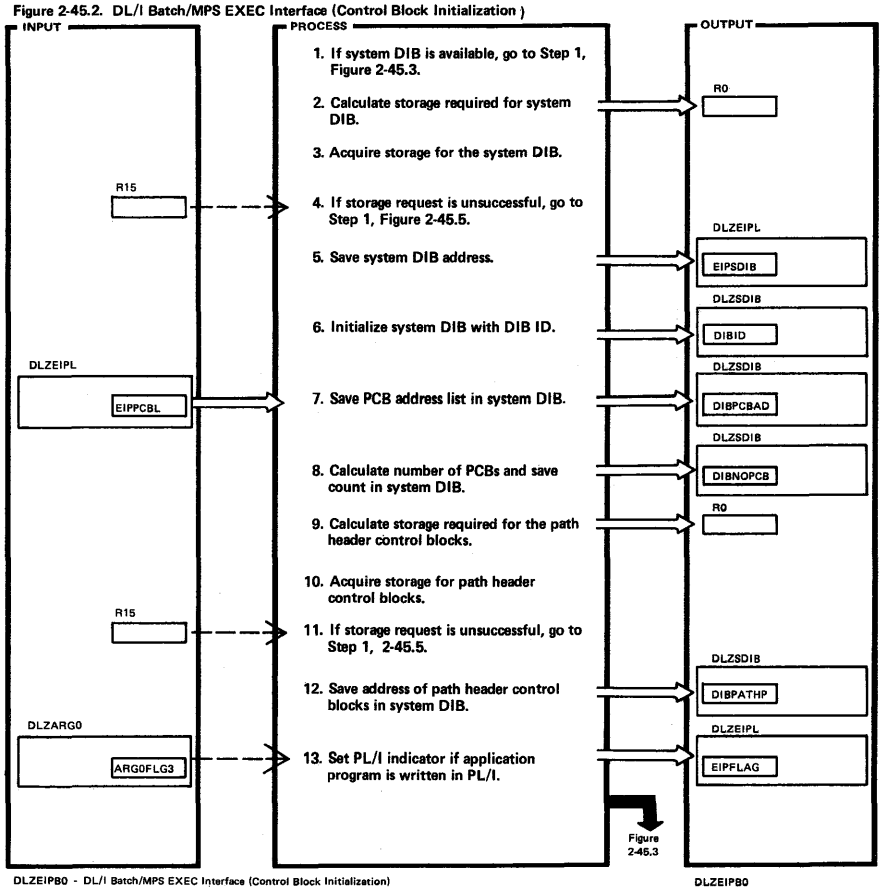
Figure 2-45.1. DL/I Batch/MPS EXEC Interface (Initialization Routine)



Extended Description	Routine	Label
1. Address of the HLPI parameter list is passed to this program in register 1. 5. Address of the EIP parameter list is located 4 bytes in front of entry point for the DL/I program request handler.		DLZEIPI

Extended Description	Routine	Label

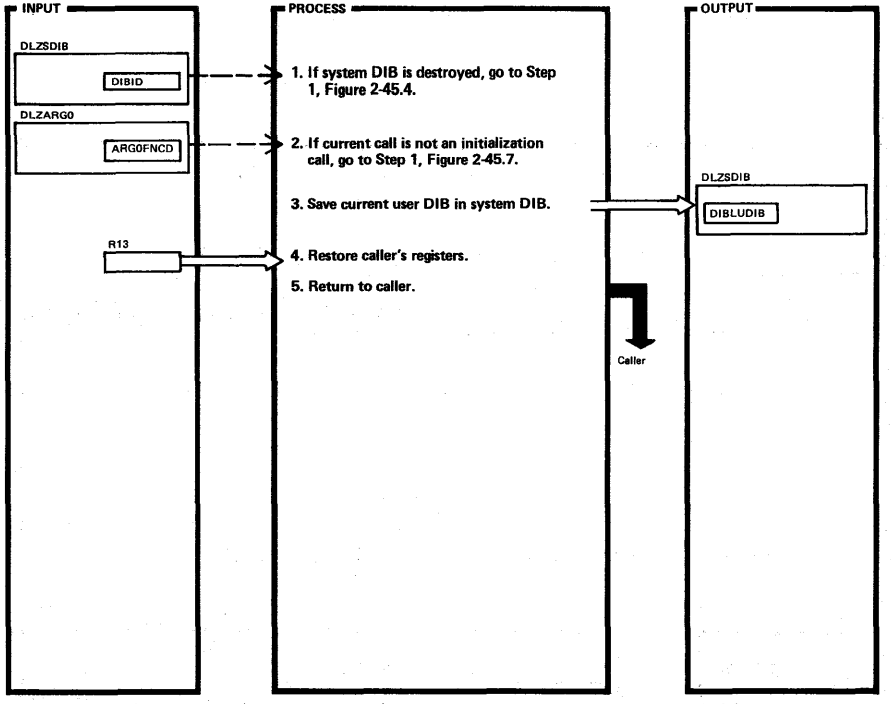
Figure 2-45.2. DL/I Batch/MPS EXEC Interface (Control Block Initialization)



Extended Description	Routine	Label
3. DOS/VS GETVIS issued. 4. Return code is in register 15. 5. Acquired storage address is returned in register 1. 8. Scan is made down the PCB list. 9. Number of PCBs X length of one path header control block.		EIPSTART PCBLOOP PCBLEND

Extended Description	Routine	Label
10. DOS/VS GETVIS issued 11. Return code is in register 15. 12. Required storage address is returned in register 1. 13. Bit APPLPLI is set in field ARG0FLG3.		

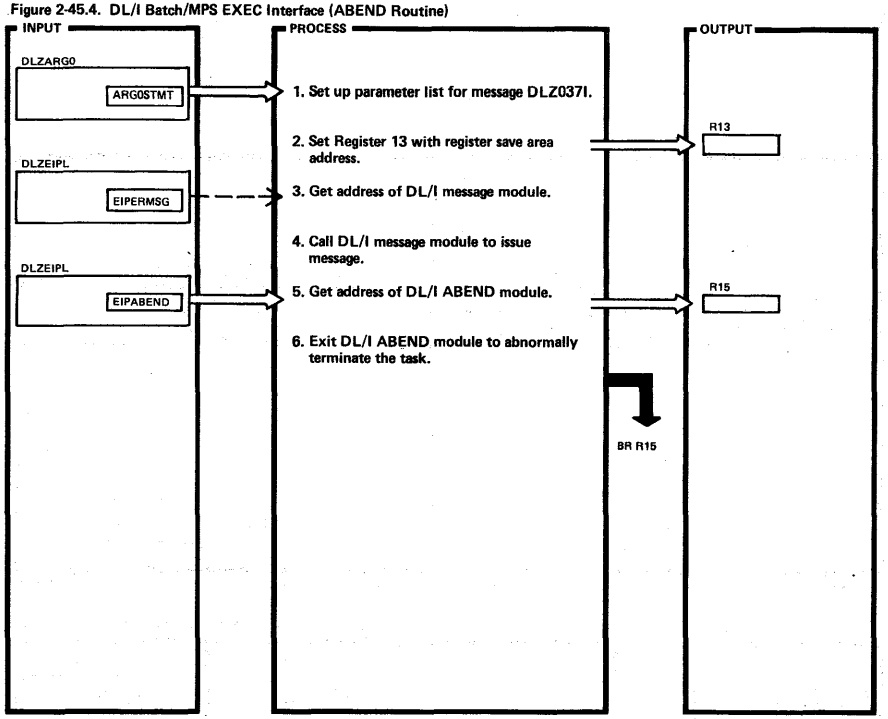
Figure 2-45.3. DL/I Batch/MPS EXEC Interface (Call Determination Routine)



Extended Description	Routine	Label
1. The first eight bytes of the system DIB are checked for 'DLZSDIB'.		

Extended Description	Routine	Label

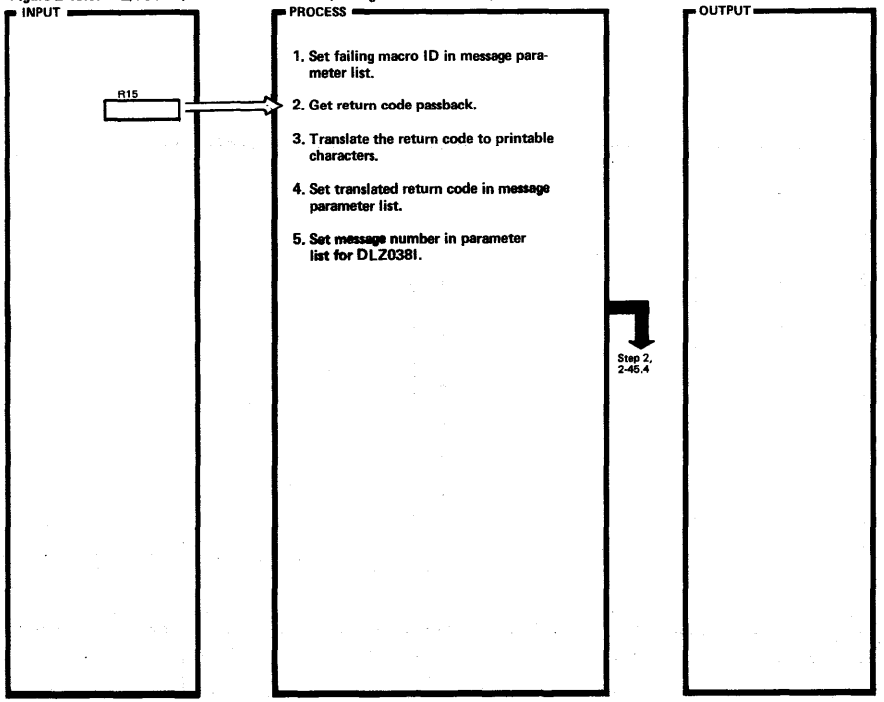
Figure 2-45.4. DL/I Batch/MPS EXEC Interface (ABEND Routine)



Extended Description	Routine	Label
1. Message parameter list is composed of the message number, status code 'TN', and failing statement number.		DIBABEND
2.		ABEXIT
4. Batch = ERRORMSG MPS = DLZMMSGX		
5. Batch = DLZABEND MPS = DLZMABNX		

Extended Description	Routine	Label

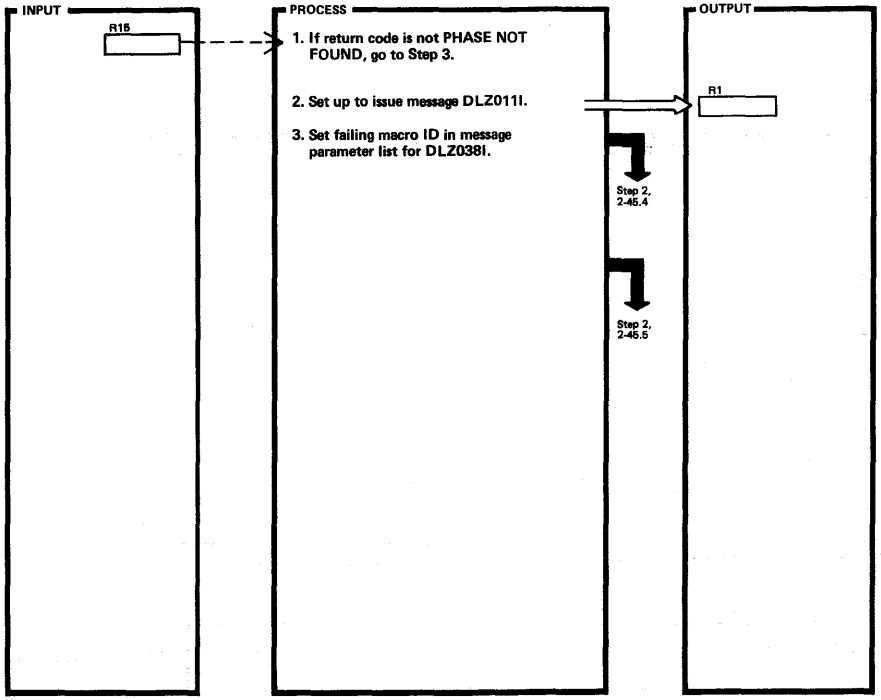
Figure 2-45.5. DL/I Batch/MPS EXEC Interface (Storage Failure Routine)



Extended Description	Routine	Label
1. Failure on GETVIS.		STRABEND
2.		ABENDFX

Extended Description	Routine	Label

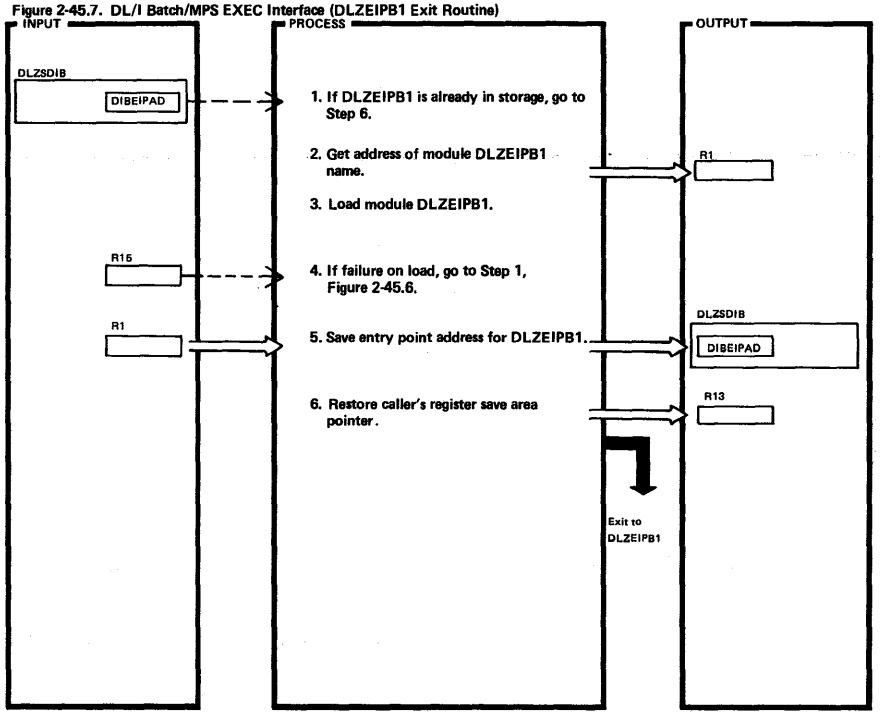
Figure 2-45.6. DL/I Batch/MPS EXEC Interface (Load Failure Routine)



Extended Description	Routine	Label
1. PHASE NOT FOUND return code = X'14'		LOADFAIL
3. Failure on CDLOAD.		GETFAIL

Extended Description	Routine	Label

Figure 2-45.7. DL/I Batch/MPS EXEC Interface (DLZEIPB1 Exit Routine)



DLZEIPB0 - DL/I Batch/MPS EXEC Interface (DLZEIPB1 Exit Routine) DLZEIPB0

Extended Description	Routine	Label
1.		GETFNCAL
2.		LOADEIP
3. DOS/VS CDLOAD issued.		
5. Entry point address of DLZEIPB1 returned in register 1.		
6. Control will be returned directly to the caller by DLZEIPB1.		

Extended Description	Routine	Label

Figure 2-46. DL/I Batch/MPS EXEC Interface (Overview) (Part 1 of 2)

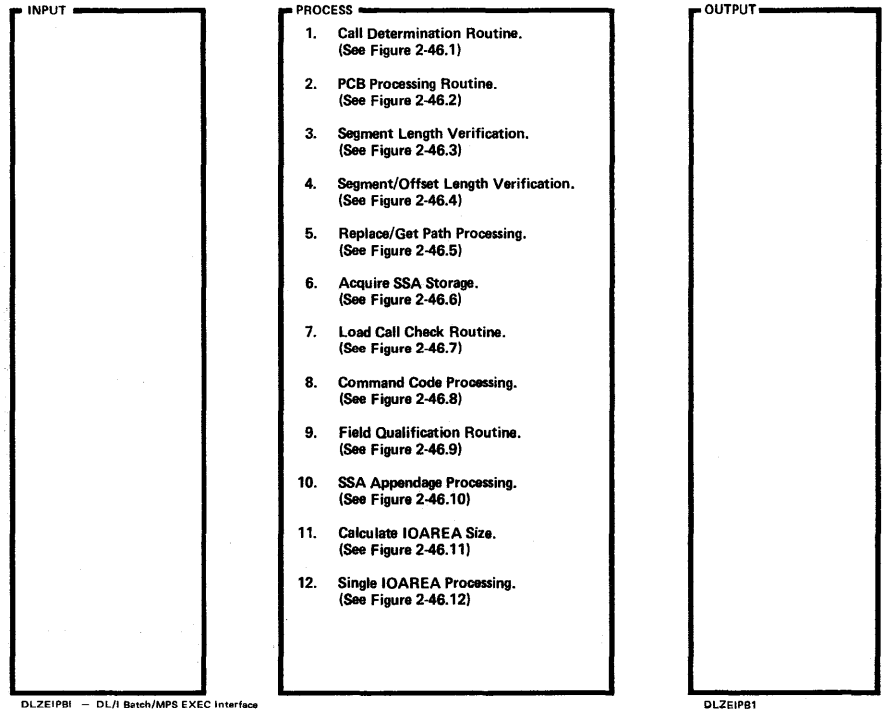


Figure 2-46. DL/I Batch/MPS EXEC Interface (Overview) (Part 2 of 2)

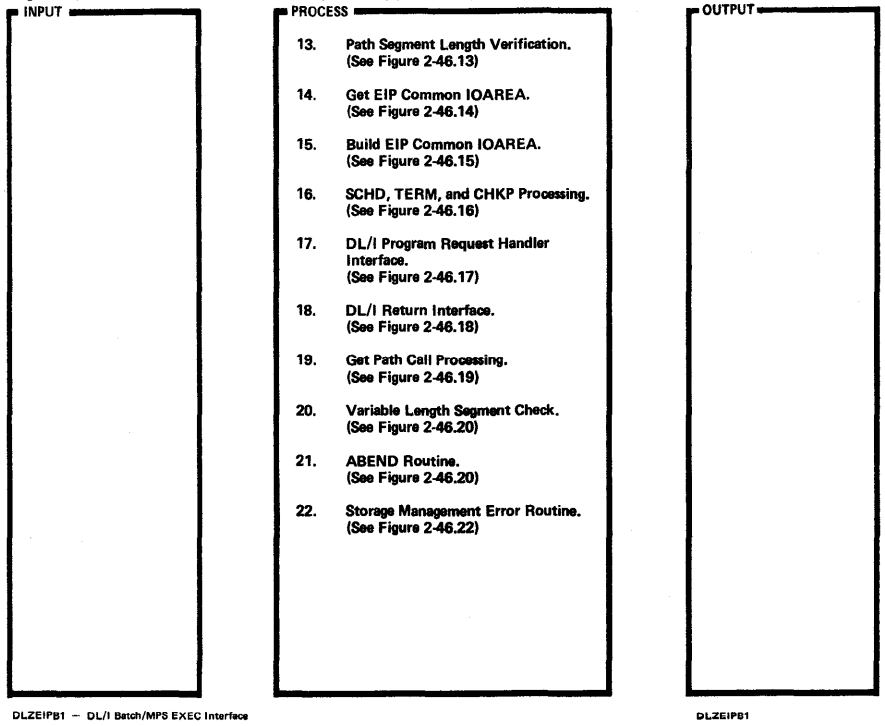
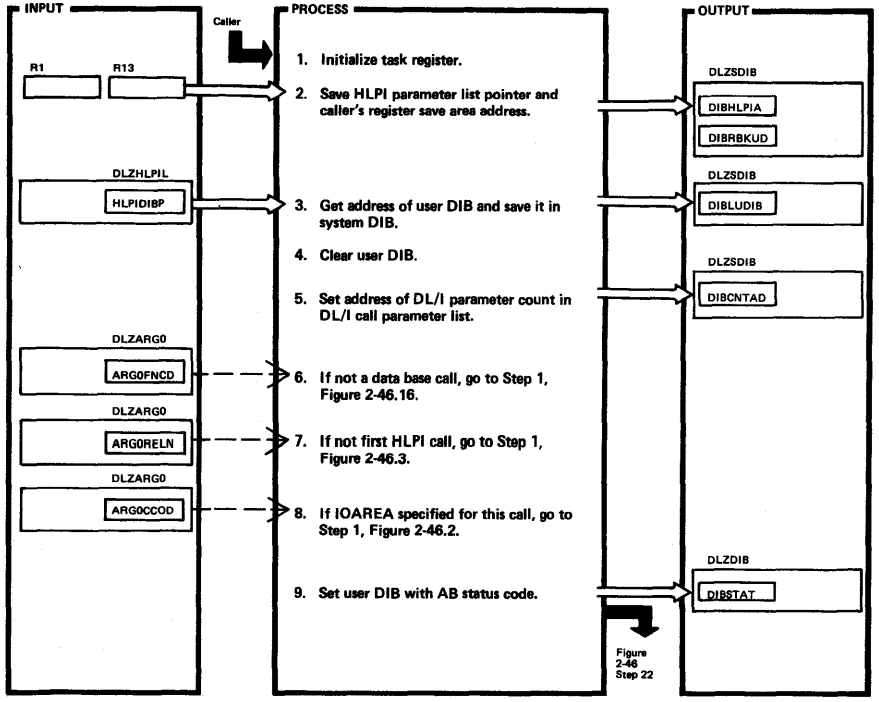


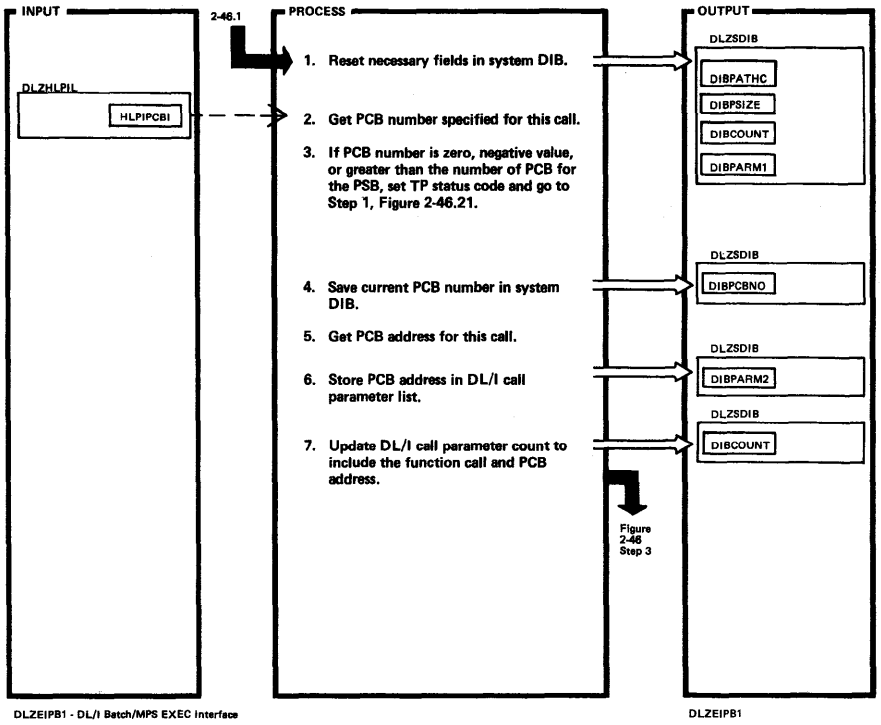
Figure 2-46.1. Call Determination Routine



Extended Description	Routine	Label
1.		DLZEIP0
2.		GETFNCAL
4. User DIB set to binary zeroes except for verison.		
6. Data base calls as defined here are get, insert, replace, delete, and load.		
7. The first call for an EXEC DLI statement passed is actually the call for the object statement.		
8. Bit CCINFROM on in byte ARG0CCOD.		
9. AB status code = statement IOAREA required for object segment.		

Extended Description	Routine	Label

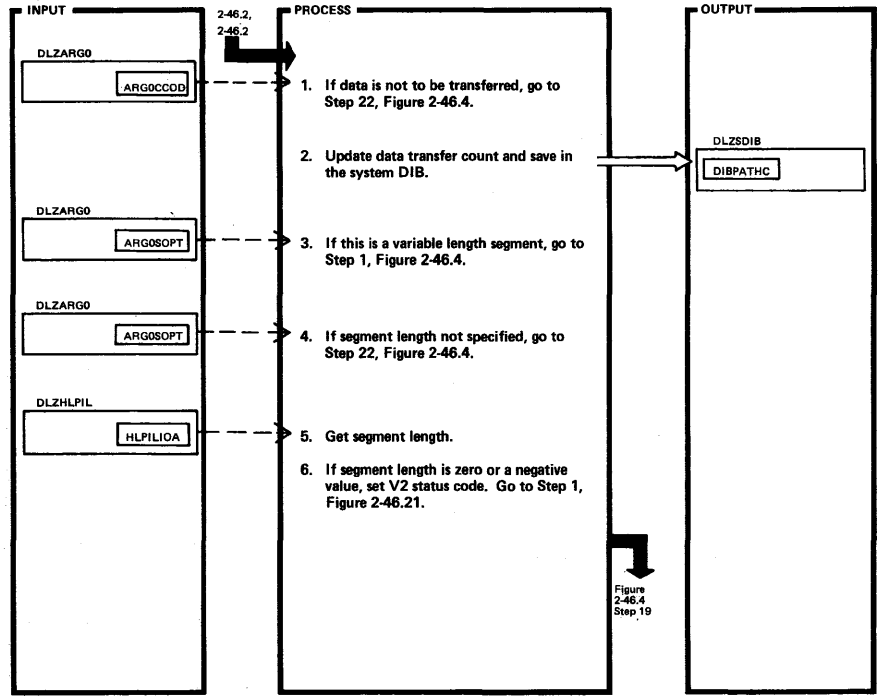
Figure 2-46.2. PCB Processing Routine



Extended Description	Routine	Label
1. On the first HLPI call for each EXEC DLI statement the path count, current required IOAREA size, the DL/I parameter count are set to binary zeroes and the DL/I function-call is set in the DL/I call parameter list.		FRSTIOOK
5. Using the PCB number, index into the PCB list for the current PCB address.		SETPCBPR

Extended Description	Routine	Label

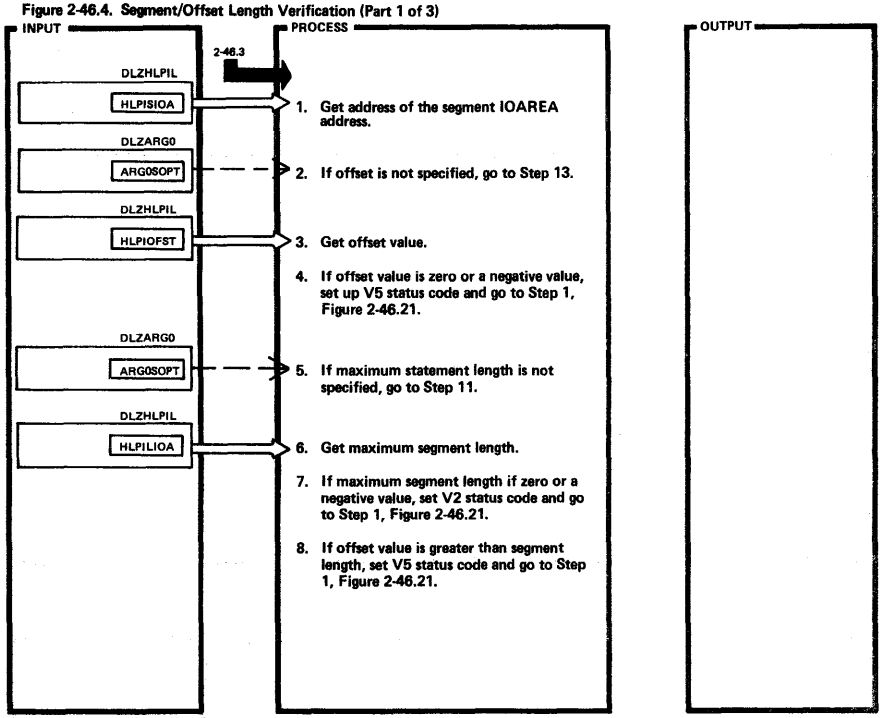
Figure 2-46.3. Segment Length Verification



Extended Description	Routine	Label
1. Bit CCINFROM on in byte ARG0CCOD.		CHKTRANS
2. A count is kept for each call that has requested data transfer for the EXEC DLI statement.		
3. Bit OPTVAR on in byte ARG0SOPT.		
4. Bit OPTSEGL on in byte ARG0SOPT.		FXSGSIZE
5. Field HLPILIOA contains a pointer to the segment length.		

Extended Description	Routine	Label

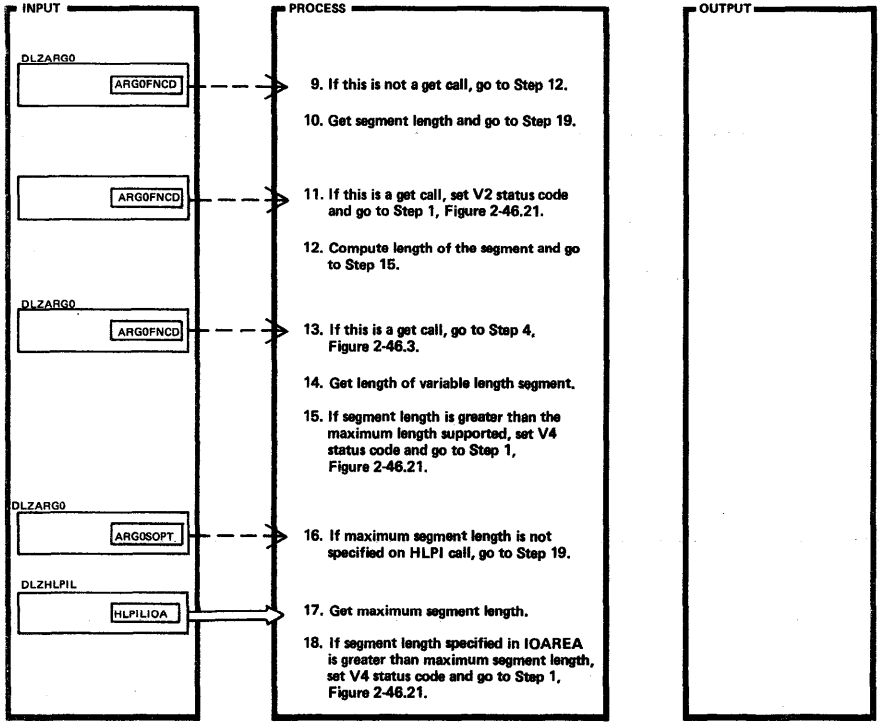
Figure 2-46.4. Segment/Offset Length Verification (Part 1 of 3)



Extended Description	Routine	Label
1.		VRSGSIZE
2. Bit OPTOFF not on in byte ARG0SOPT.		
3. Field HLPIOFST contains a pointer to the OFFSET value. This value is the length of the concatenated key plus the length of the intersection data (If any).		
6. Field HLPILIOA contains a pointer to the segment length.		

Extended	Routine	Label

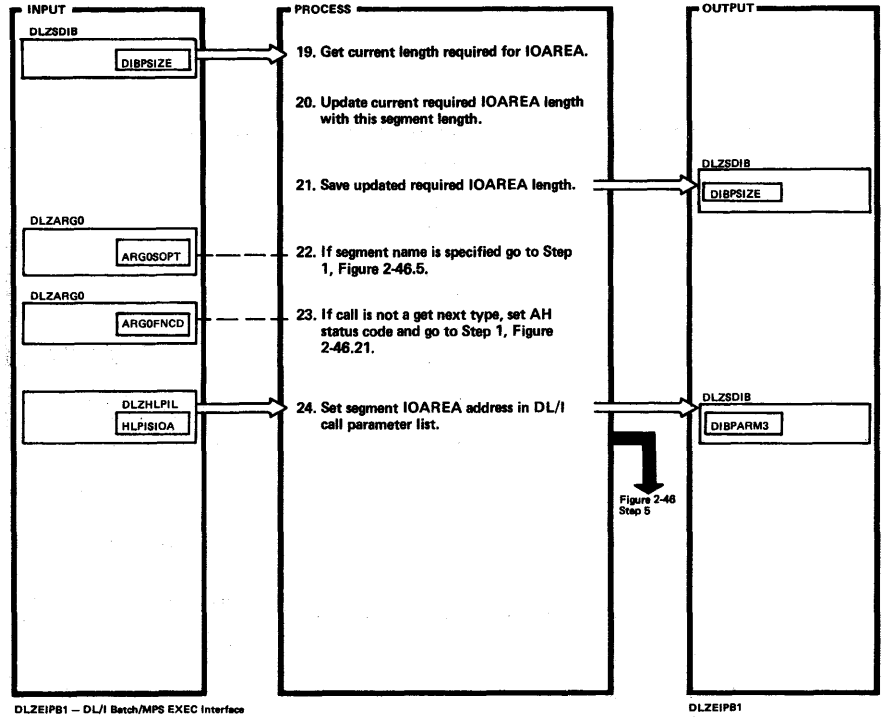
Figure 2-46.4. Segment/Offset Length Verification (Part 2 of 3)



Extended Description	Routine	Label
9. This check is made for all get type call GN, GU, and GNP.		
11.		VGETCALL
12. This length includes the concatenated key, intersection data, and the segment.		VRSGCON
13.		NOFFSET
14. This length is in the first two bytes of the segment IOAREA.		
15.		MAXCHECK
16. Bit OPTSEGL not on in field ARG0SOPT.		
17. Field HLPILIOA contains a pointer to maximum segment length.		

Extended Description	Routine	Label

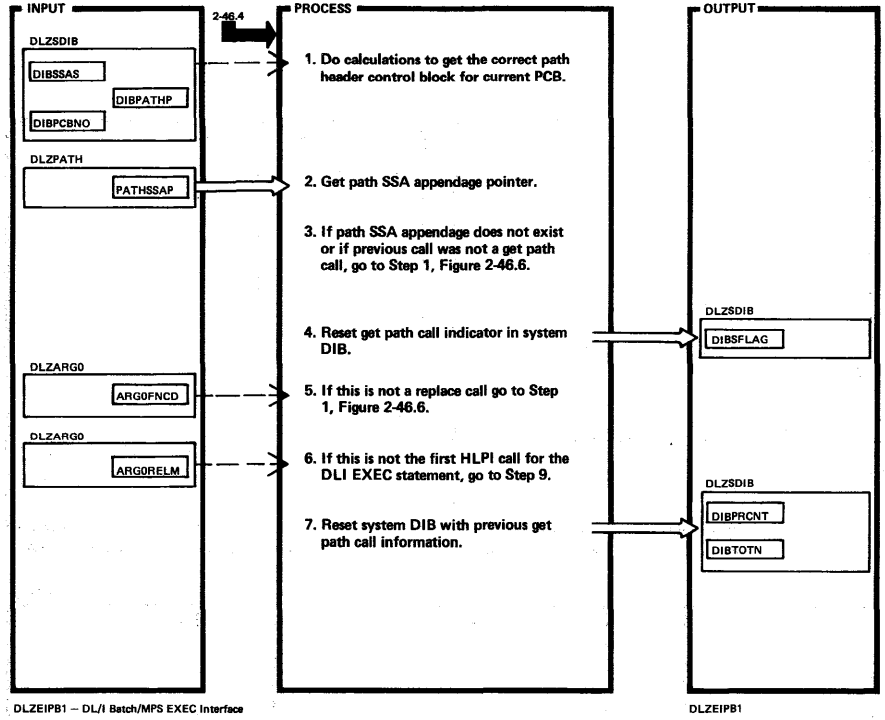
Figure 2-46.4. Segment/Offset Length Verification (Part 3 of 3)



Extended Description	Routine	Label
19.		SETSGSIZ
22. Bit OPTSEGM on in field ARG0SOPT.		NOTRMNSF

Extended Description	Routine	Label

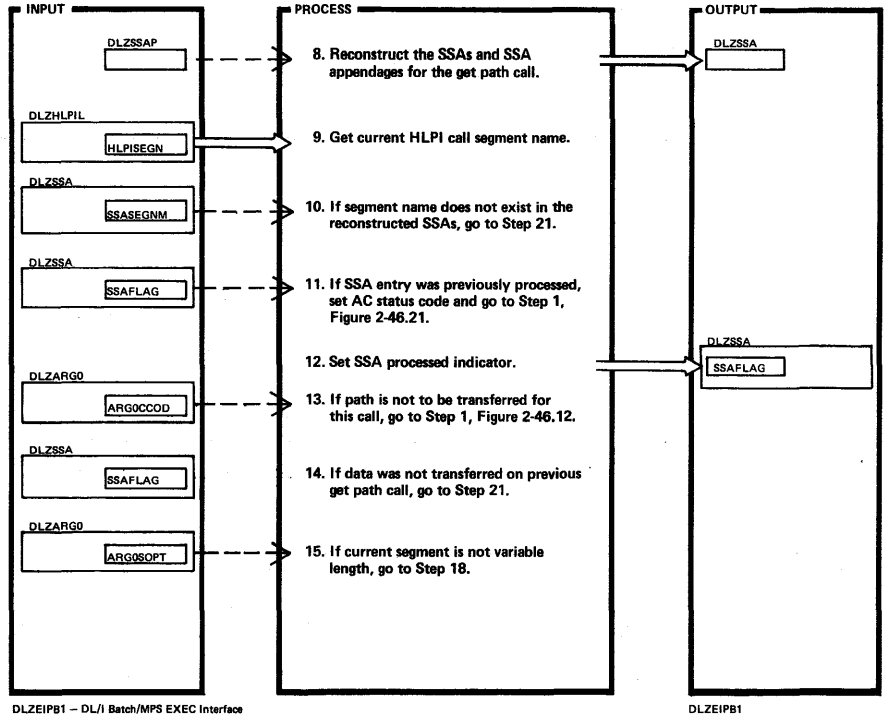
Figure 2-46.5. Replace/Get Path Processing (Part 1 of 3)



Extended Description	Routine	Label
1. Current PCB number-1 x length of the path header control block. 4. Bit DIBGPATH set ON in field DIBSFLAG. 6. If the previous call was a get path call and current call is a replace call and this is the first HLPI call for the DLI EXEC statement, the SSAs and SSA appendages are reconstructed based on the get path call.		STARTSSA

Extended Description	Routine	Label

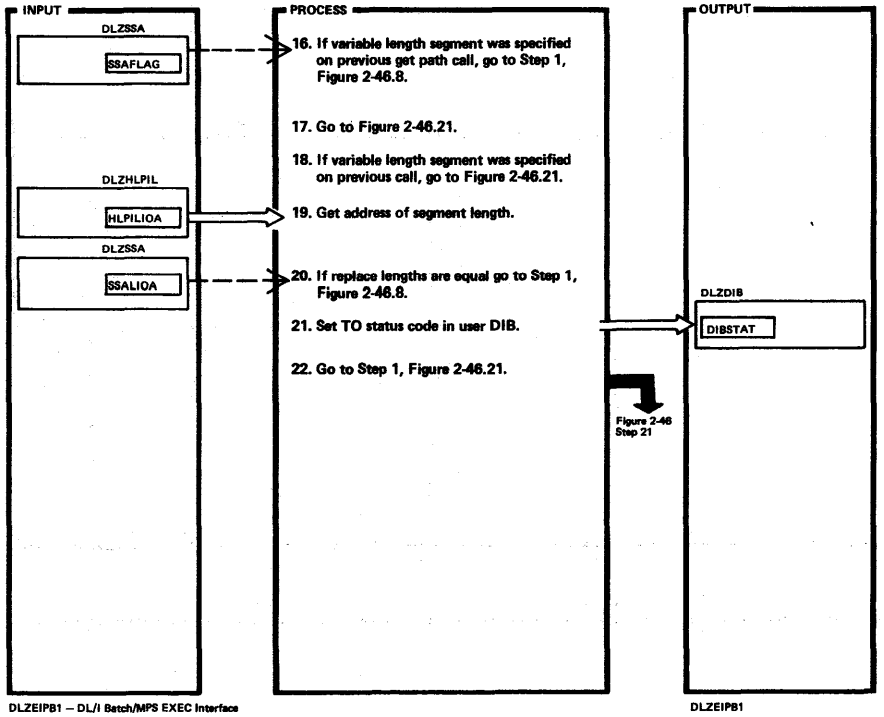
Figure 2-46.5. Replace/Get Path Processing (Part 2 of 3)



Extended Description	Routine	Label
8. SSAs are reconstructed with segment name, n command code, and blank delimiter. 9. HLPISEGN contains a pointer to the segment name. 10. 11. Bit SSAPROC not set in field SSAFLAG. 12. Bit SSAPROC set ON in field SSAFLAG. This is done to ensure duplicate segment names are not specified in current call. 13. Bit CCINFROM not on in field ARG0CCON. 14. Bit SSADATAT not on in field SSAFLAG. 15. Bit OPTVAR not on in field ARG0SOPT.		RELOOP SEGNCHK SEGLOOP SSAFOUND SSAOK

Extended Description	Routine	Label

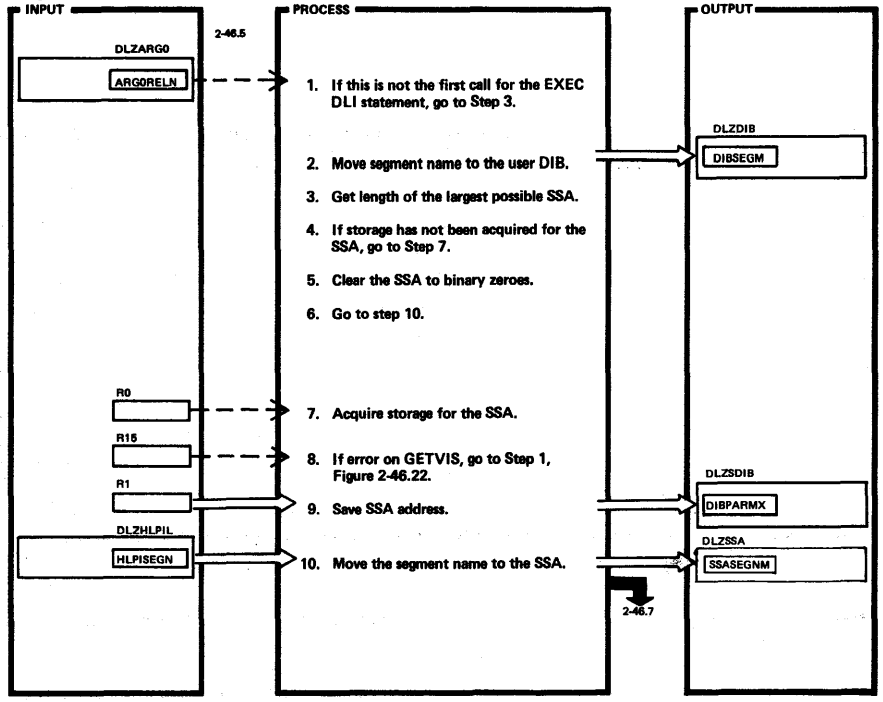
Figure 2-46.5. Replace/Get Path Processing (Part 3 of 3)



Extended Description	Routine	Label
16. Bit SSAVARL on in byte SSAFLAG.		
18.		SSANOTV
19. HLPILIOA contains a pointer to the segment length.		
21. Status code 'TO' indicates replace/get path calls inconsistent.		ERRORTO

Extended Description	Routine	Label

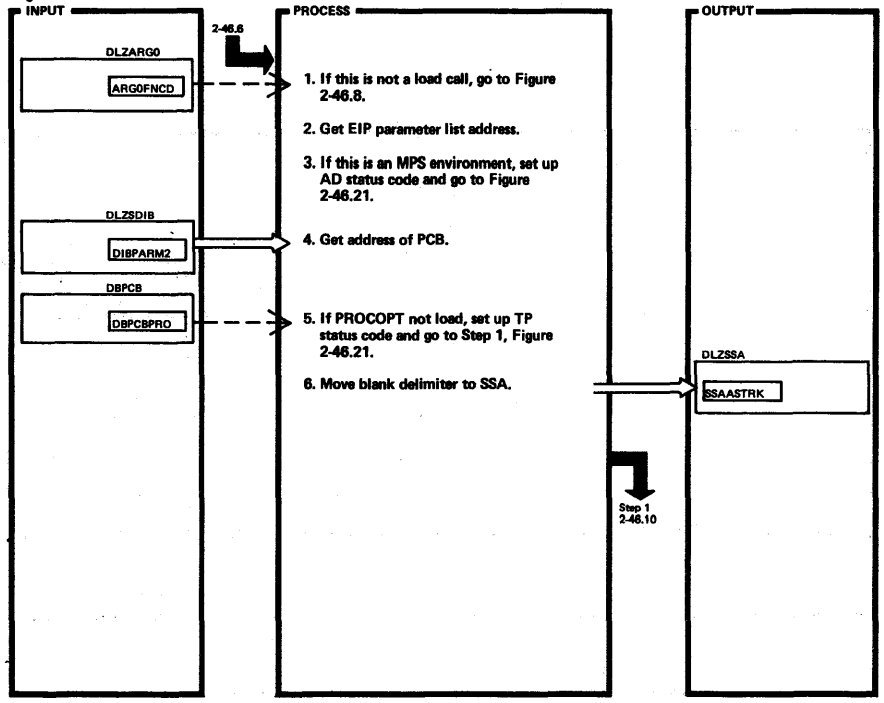
Figure 2-46.6. Acquire SSA Storage



Extended Description	Routine	Label
1. Relative call number in field ARG0RELN is one.		SSANPATH
2. The first HLPI call for the EXEC DLI statement is for the object segment.		SETSEGM
3.		SUBTWO
7. DOS/VS GETVIS issued.		SCAAQ
9. The SSA pointer area in the DLI PARM-list in the system DIB where the SSA pointer stored is dependent on the relative number of the HLPI call.		
10. Field HLPISGM contains a pointer to the segment name.		

Extended Description	Routine	Label

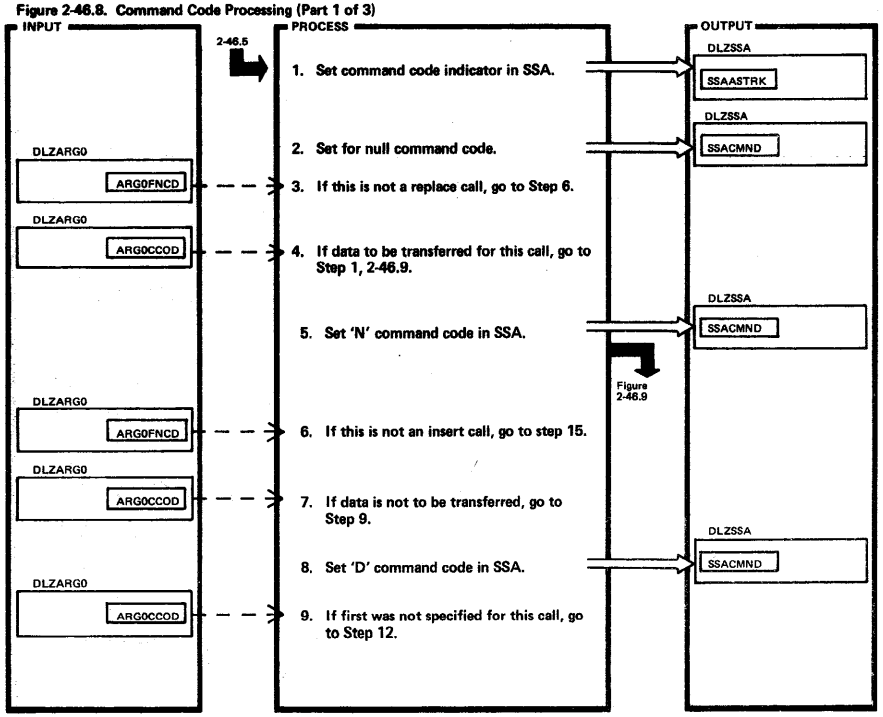
Figure 2-46.7. Load Call Check Routine



Extended Description	Routine	Label
1. Function code not equal to X'18'.		
2. The EIP parameter list address is stored four bytes in front of the entry point of the DLI program request handler.		
4.		PCBCHECK
5. PCB processing option must be 'L' or 'LS'.		
6.		PROCOK

Extended Description	Routine	Label

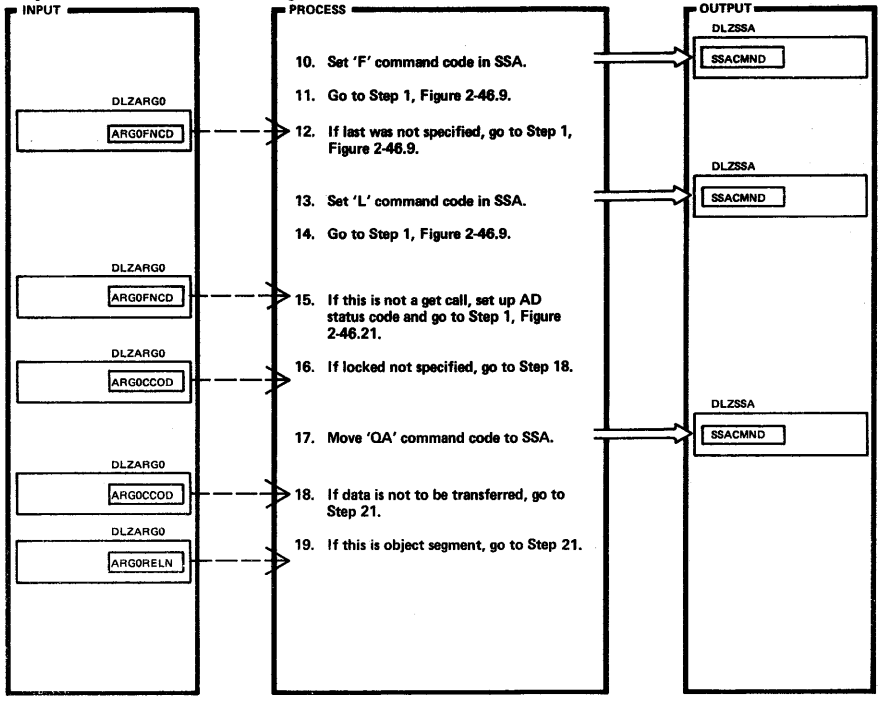
Figure 2-46.8. Command Code Processing (Part 1 of 3)



	Extended Description	Routine	Label
1.	Command code indicator is an astrisk (*).		CONTSSA
2.	Null command codes are dashes (-).		
3.	Replace call code equals X'14'.		
4.	Bit CCINFROM on in byte ARG0CCOD.		
5.	'N' command code indicates that this segment is not to be replaced.		
6.	Function code is not equal to X'12'.		CKCCISRT
7.	Bit CCINFROM not on in byte ARG0CCOD.		
8.	'D' indicates multiple insert path call.		
9.	Bit CCFIRST not on in byte ARG0CCOD.		

Extended Description	Routine	Label

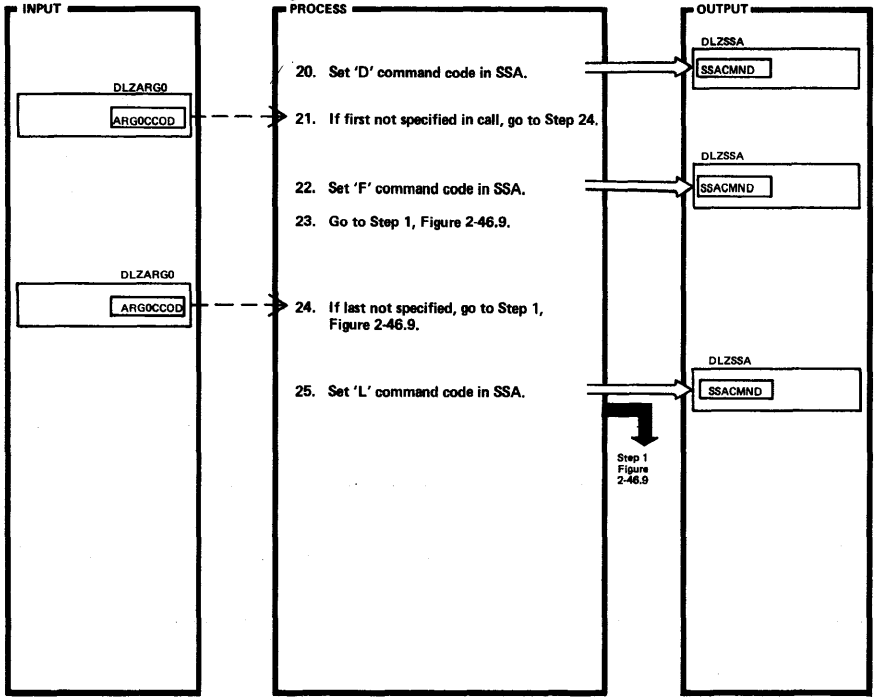
Figure 2-46.8. Command Code Processing (Part 2 of 3)



	Extended Description	Routine	Label
10.	'F' indicates start with the first occurrance of this segment type to satisfy this level of call.		
12.	Bit CCLAST not on is field ARG0CCOD.		CKCCLSTI
13.	'L' indicates use last occurance of segment type to satisfy this level of call.		
15.	Function code is not in range of X'10' and X'0A'.		CKCCG0
16.	Bit CCLOCKED not on in byte ARG0CCOD.		
17.	'QA' indicates to lock segment(s) returned to prevent modification by another task.		

Extended Description	Routine	Label

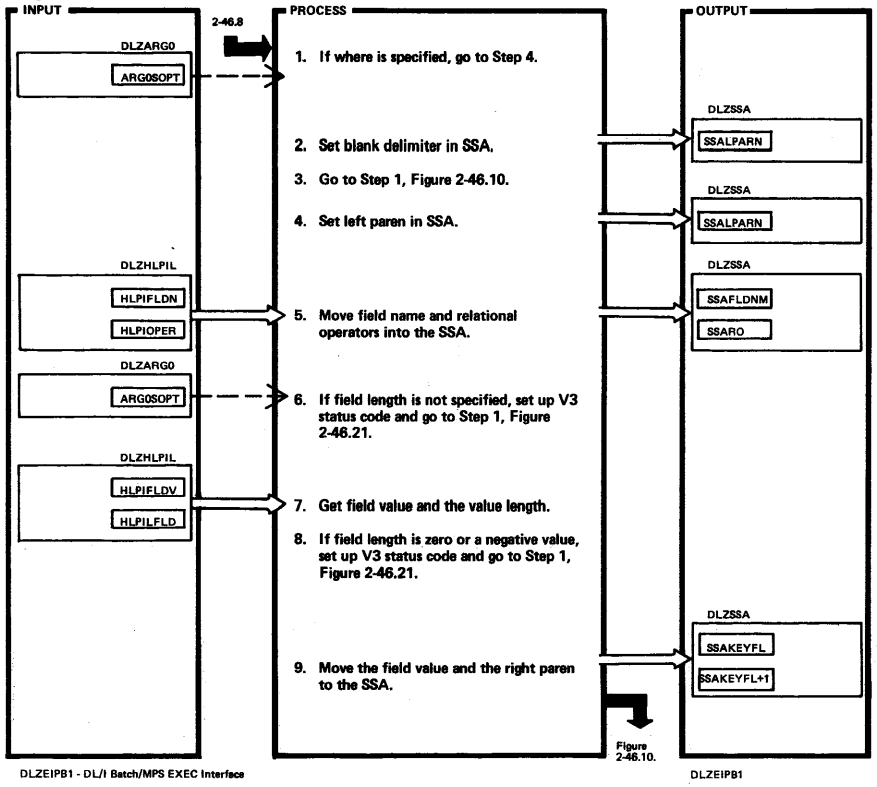
Figure 2-46.8. Command Code Processing (Part 3 of 3)



Extended Description	Routine	Label
21.		CKCCISTG
24.		CKCCLSTG

Extended Description	Routine	Label

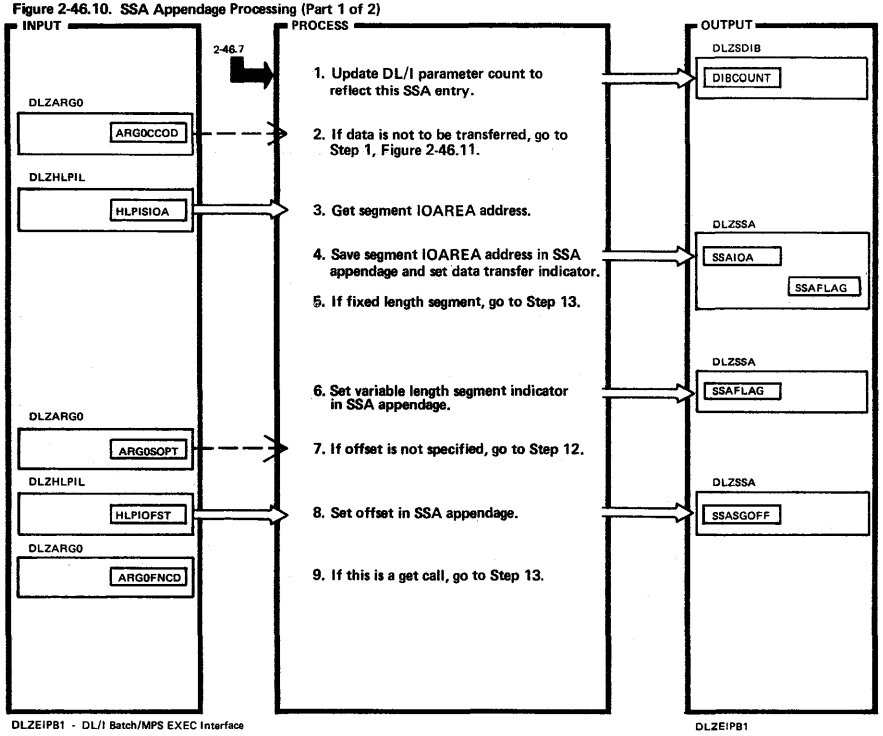
Figure 2-46.9. Field Qualification Routine



Extended Description	Routine	Label
1. Bit OPTWHERE set on in byte ARG0SOPT.		FIELDCHK
2. Blank indicates unqualified SSA.		
4. Left paren indicates qualified SSA.		QUALSSA
5. Field HLPIFLDN has a pointer to the field name and HLPIOPER has a pointer to the relational operators.		
6. Bit OPTFLDL not on in byte ARG0SOPT.		
7. Field HLPIFLDV is a pointer to the field value and HLPILFLD is a pointer to the field value length.		FLDLNOK
9. Right paren is delimiter for qualification of SSA.		

Extended Description	Routine	Label

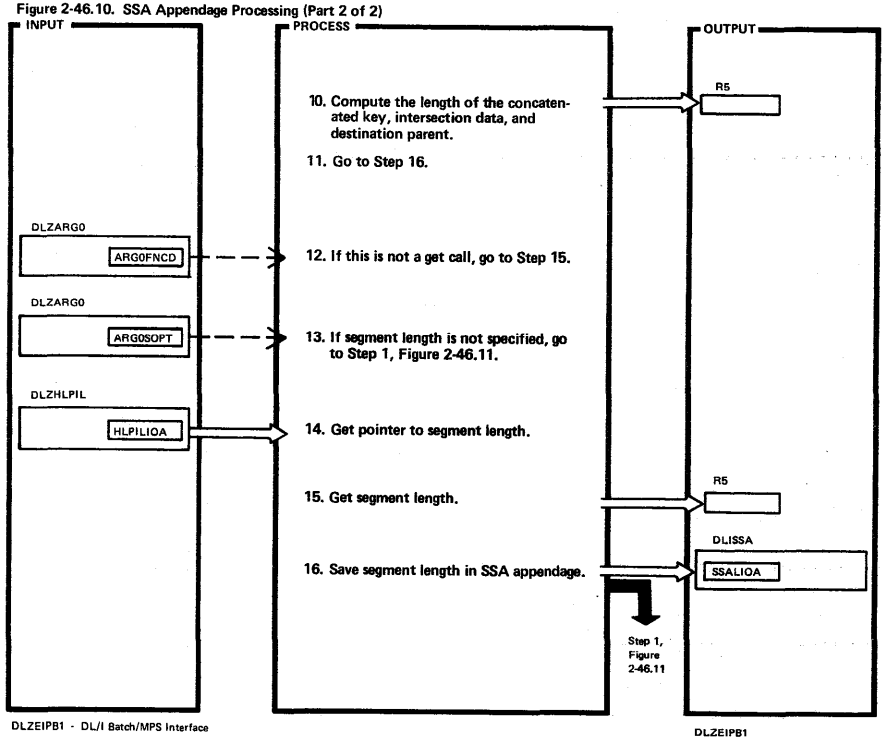
Figure 2-46.10. SSA Appendage Processing (Part 1 of 2)



Extended Description	Routine	Label
1.		BMPSSA
2. Bit CCINFROM not on in byte ARG0CCOD.		
3. Field HLPISIOA contains the address of the segment I/O area.		
4. Bit SSADATAT indicates data transfer in the SSA appendage.		
6. Bit SSAVARL in the SSA appendage indicates variable length segment.		VARSEGL
7. Bit OPTOFF not on in byte ARG0SOPT.		
8. HLPIOFST contains a pointer to the offset.		

Extended Description	Routine	Label

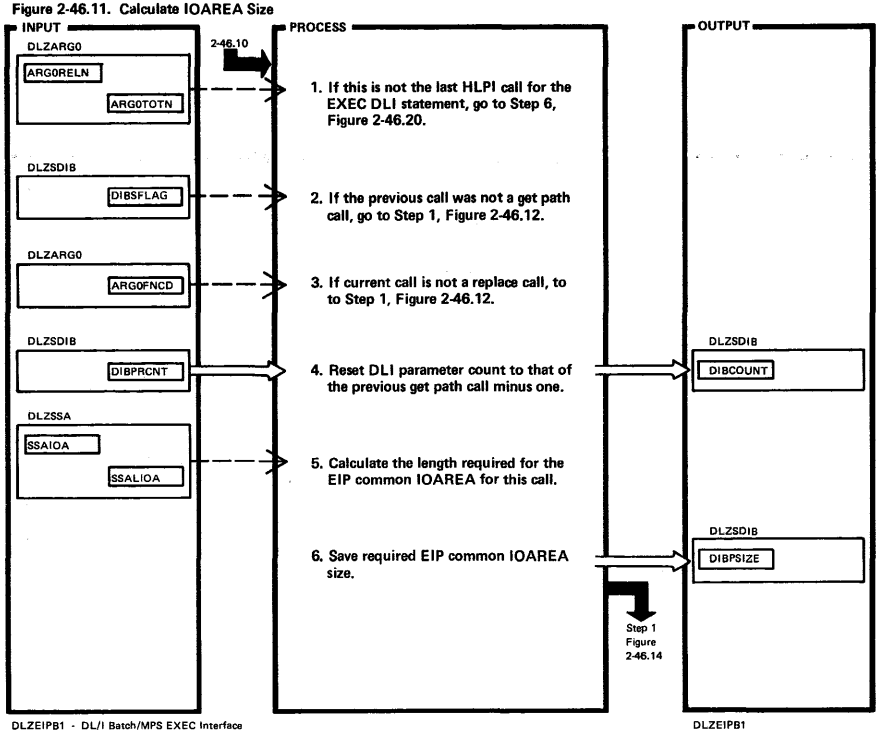
Figure 2-46.10. SSA Appendage Processing (Part 2 of 2)



Extended Description	Routine	Label
12.		FIXSEGL
13. Bit OPTSEGL not on in byte ARG0SOPT.		FIXSEG
14. HLPILIOA contains a pointer to the segment length.		
15.		SETSEGLN
16.		SETLNCIS

Extended Description	Routine	Label

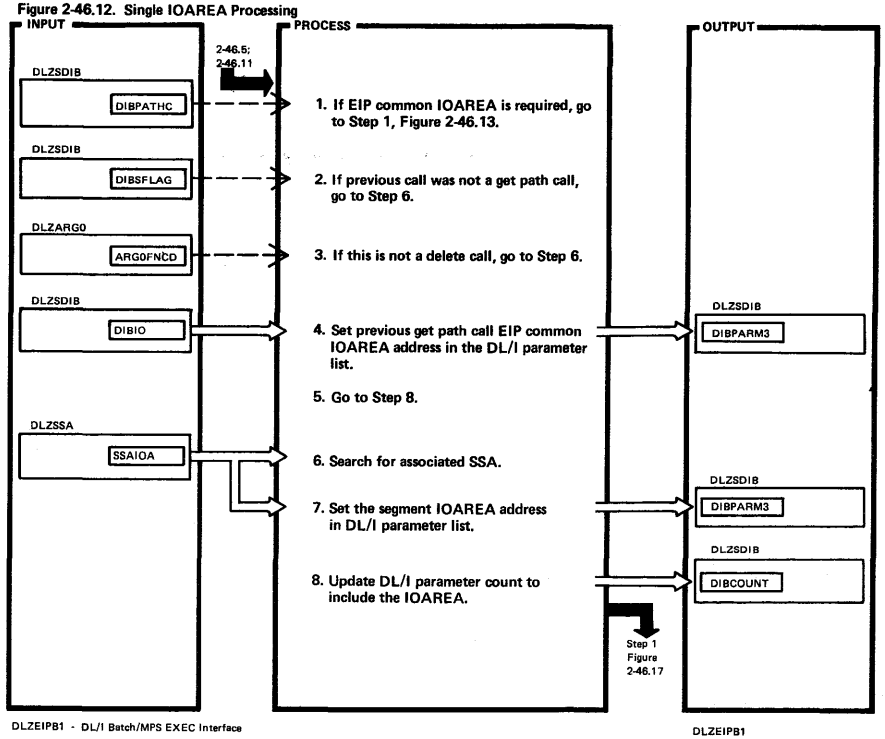
Figure 2-46.11. Calculate IOAREA Size



Extended Description	Routine	Label
1. ARGORELN contains the relative number of this call and ARG0TOTN contains the total number of calls for the EXEC DLI statement.		ANYMORE
2. Bit DIBGPATH call not on in byte DIBSFLAG.		
5. A scan is made of all SSAs associated with this call, adding the length of each segment that has data transfer required in the SSA appendage.		PROCNEXT

Extended Description	Routine	Label

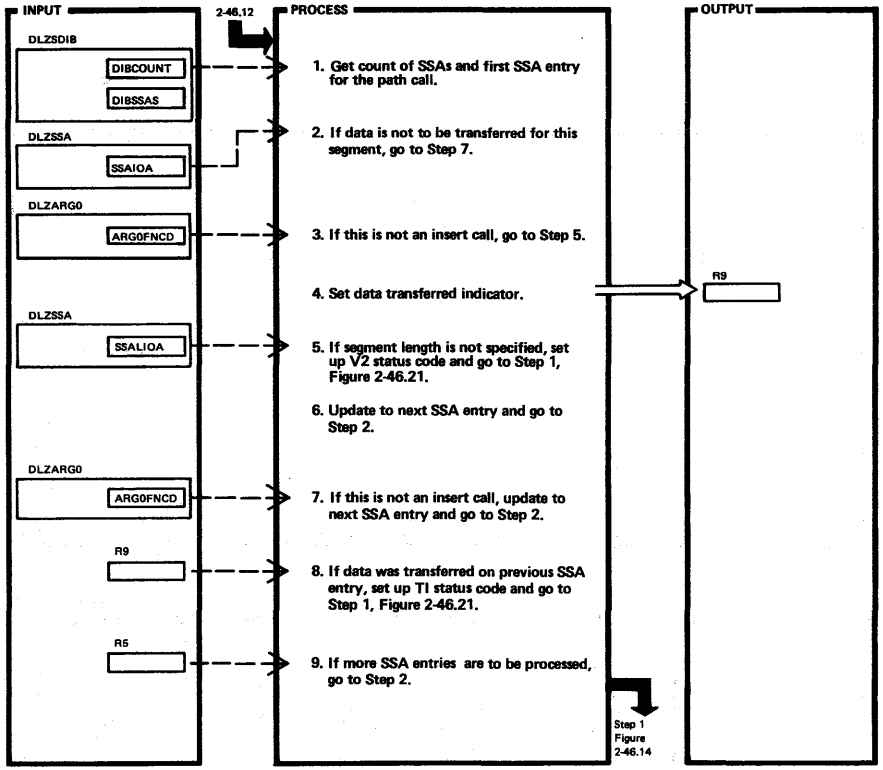
Figure 2-46.12. Single IOAREA Processing



Extended Description	Routine	Label
1. DIBPATHC contains the number of IOAREAs specified for this call.		GETIOARE
2. Bit DIBGPATH not on in byte DIBSFLAG.		
3. If delete call follows a get path call, the IOAREA must reflect the way it was after the get path call.		
6.		SEG1SRCH
7.		SEG1FND
8.		NOSSA

Extended Description	Routine	Label

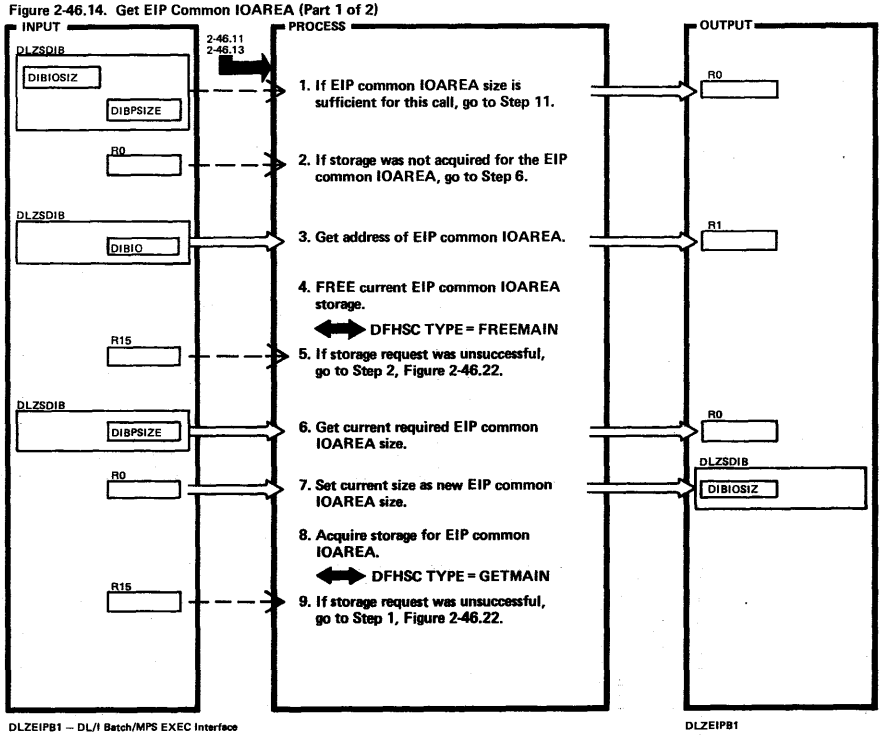
Figure 2-46.13. Path Segment Length Verification



Extended Description	Routine	Label
1. Get set to scan SSAs.		LENVERY
2.		VERFYLEN
5. Segment length must be specified for every segment that has data transfer in a path call.		CHKSEGLN
7.		CHKISRT
8. For insert calls, data transfer must be specified for every segment from the first one encountered to the object segment.		

Extended Description	Routine	Label

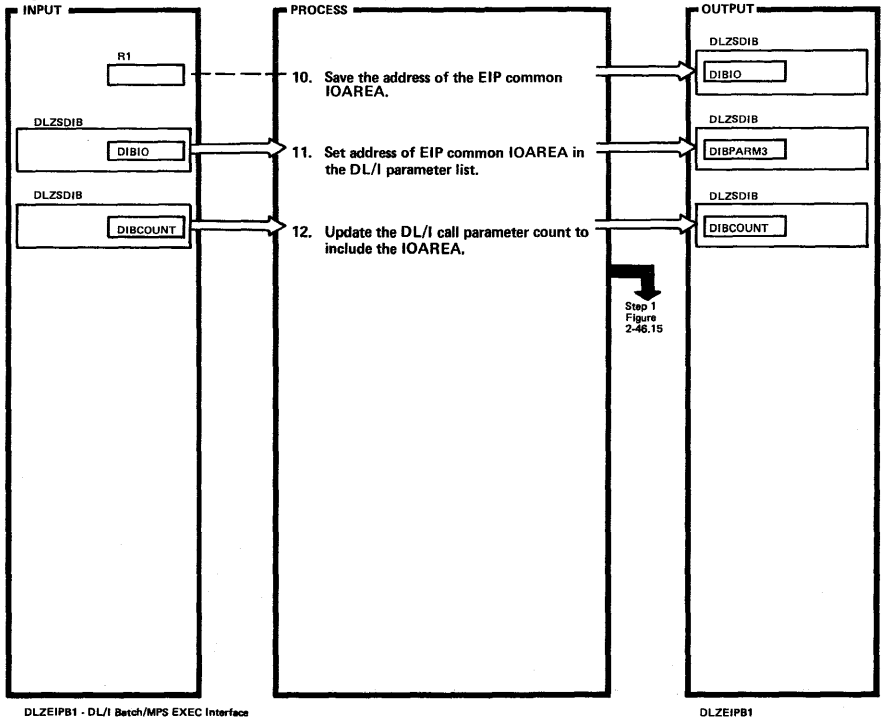
Figure 2-46.14. Get EIP Common IOAREA (Part 1 of 2)



Extended Description	Routine	Label
1. If the storage size (DIBPSIZE) calculated for this call is less than or equal to the size of the existing EIP common IOAREA, the existing one is used.		PROVDIOA
4. DOS/VS FREEVIS issued.		
5. Unsuccessful return code is in register 15.		
6.		GETEIPIO
8. DOS/VS GETVIS issued.		
9. Unsuccessful return code is in register 15.		

Extended Description	Routine	Label

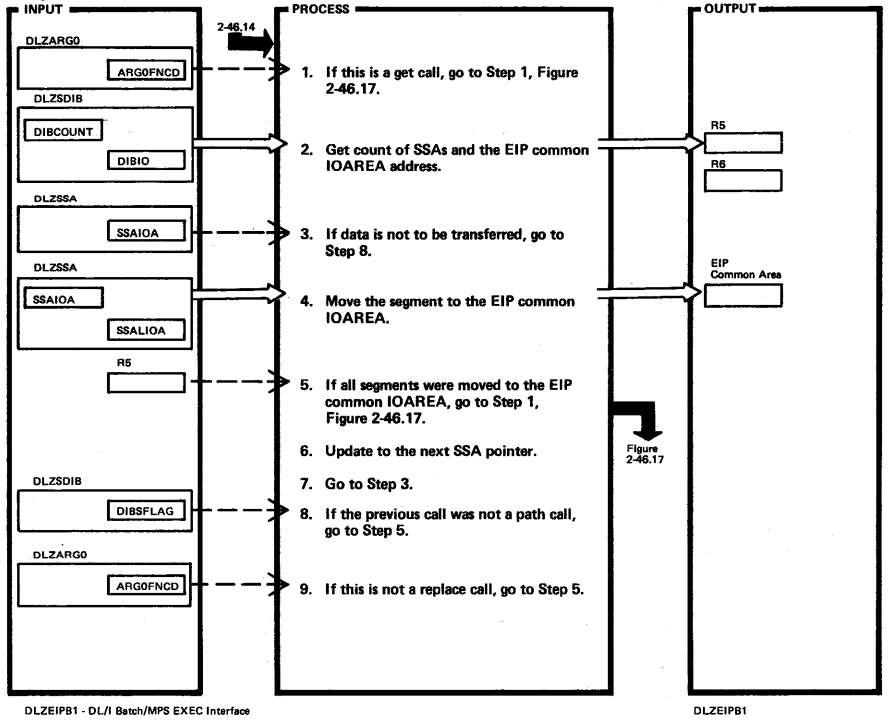
Figure 2-46.14. Get EIP Common IOAREA (Part 2 of 2)



Extended Description	Routine	Label
10. Address of the area processed by GETMAIN is returned in register 1.		
11.		EIPIOAOK

Extended Description	Routine	Label

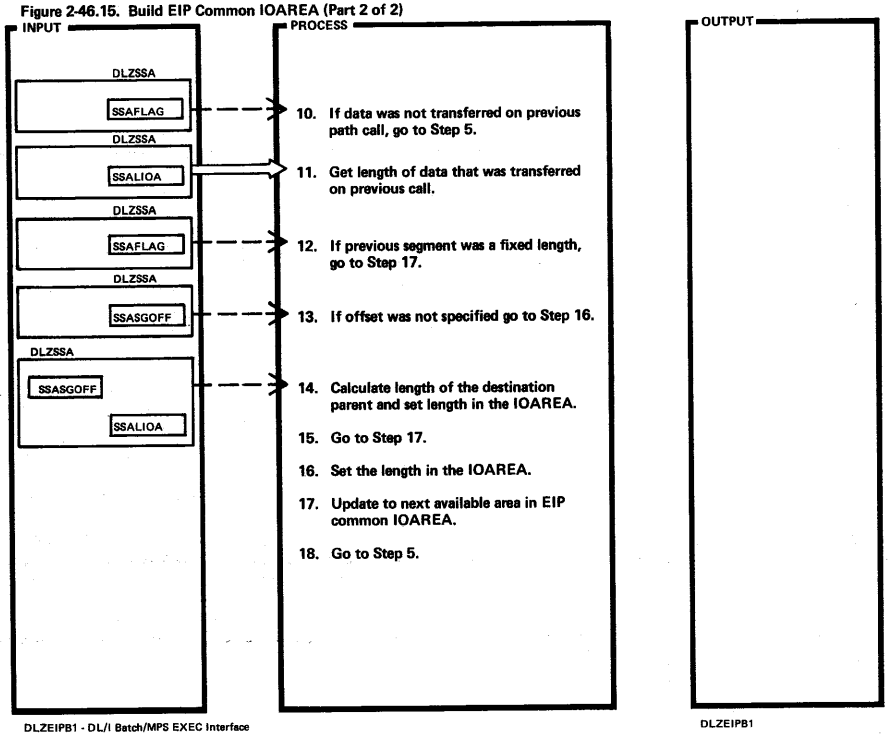
Figure 2-46.15. Build EIP Common IOAREA (Part 1 of 2)



Extended Description	Routine	Label
2. SSA count is set in register 5 and EIP common IOAREA address is set in register 6.		
3.		MOVELOOP
5.		MOVECHK
6.		MOVENEXT
8. Bit DIBGPATH not on in byte DIBSFLAG.		NEXTMOVE

Extended Description	Routine	Label

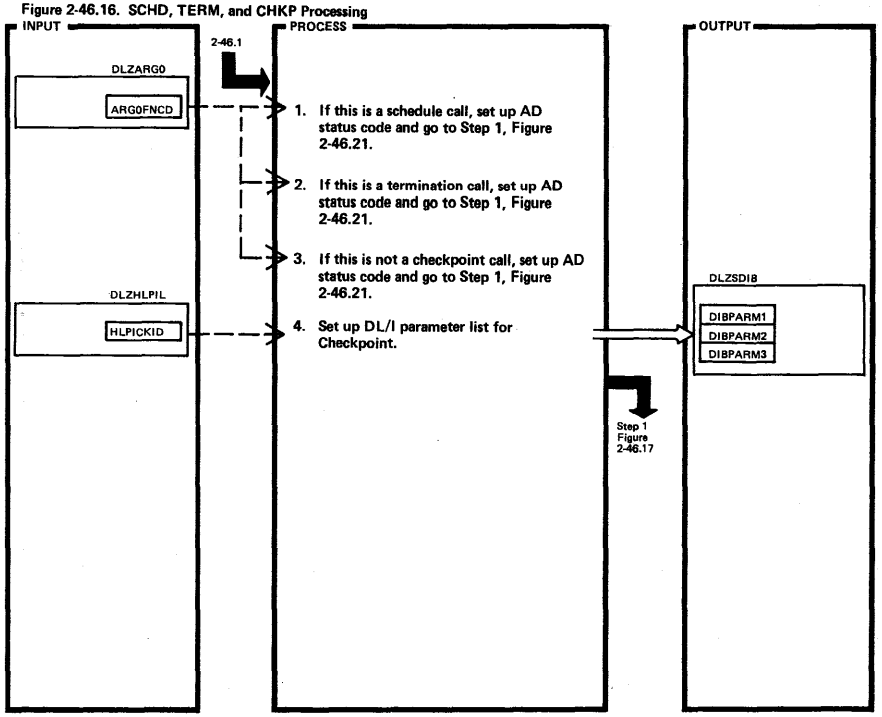
Figure 2-46.15. Build EIP Common IOAREA (Part 2 of 2)



Extended Description	Routine	Label
12. Bit SSAVARL not on in byte SSAFLAG.		
14. The length of a variable destination parent must be in the first two bytes of the segment that is after the concatenated key and intersection data.		
16. For variable length segments the length must appear in the first two bytes of the segment.		MOVEVLEN
17.		MOVEFIX

Extended Description	Routine	Label

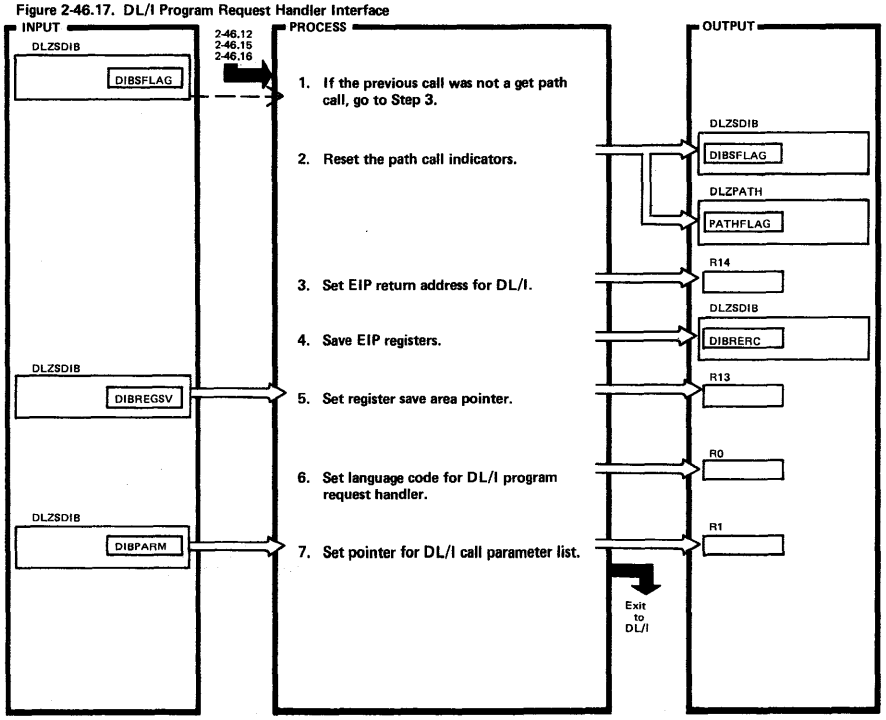
Figure 2-46.16. SCHD, TERM, and CHKP Processing



Extended Description	Routine	Label
1.		CALLSCHD
2.		CALLTERM
3.		CALLCHKP
4. HLPICKID contains a pointer to the checkpoint ID.		

Extended Description	Routine	Label

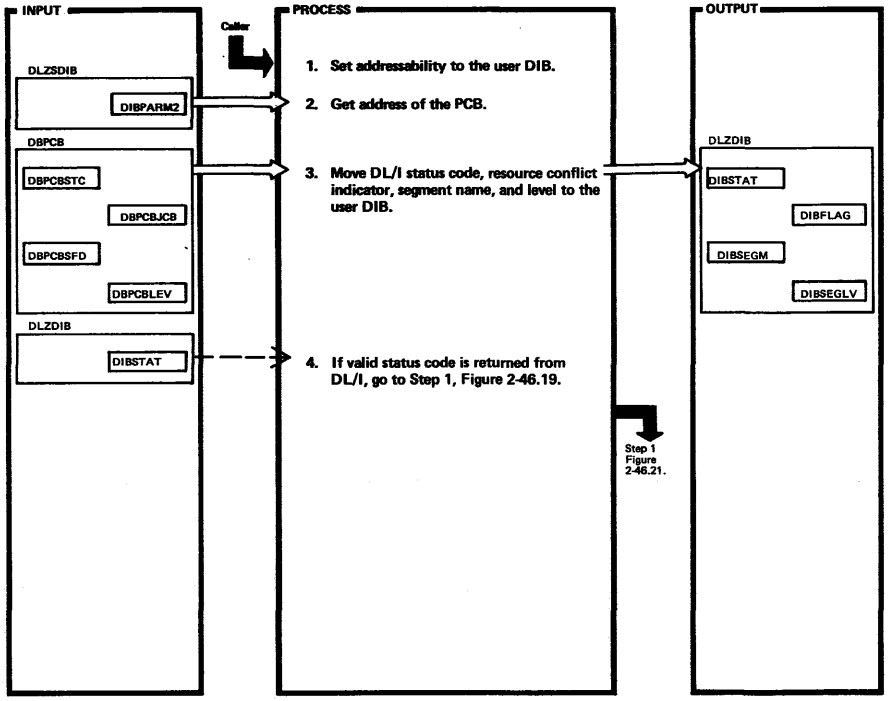
Figure 2-46.17. DL/I Program Request Handler Interface



Extended Description	Routine	Label
1. Bit DIBGPATH not on in byte DIBSFLAG.		DLIPRHEX
2. Bits DIBGPATH and PATHCALL turned off.		
3. Return address at label DLIRETRN		DLIEXPRH
6. X'02' = non-PL/I HLPI program X'03' = PL/I HLPI program		
7.		SETPARM
NOTE: Batch exit to DLZPRHB0 MPS exit to DLZMPRH		

Extended Description	Routine	Label

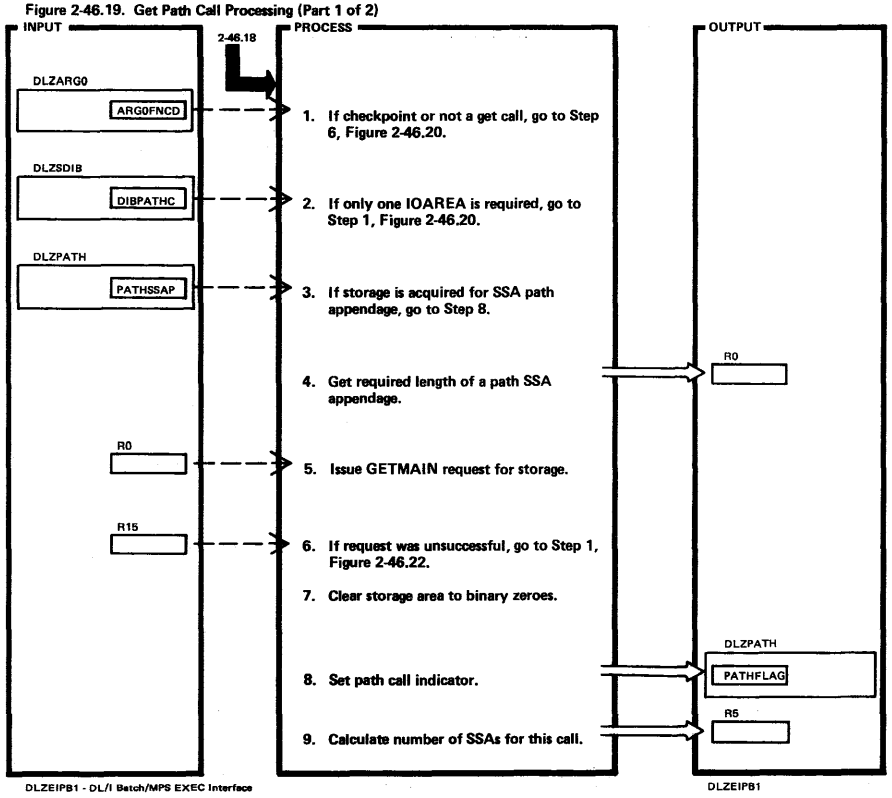
Figure 2-46.18. DL/I Return Interface



Extended Description	Routine	Label
1.		DLIRETRN
3.		COMPDIB
4. Valid DL/I status codes for HLPI: 'bb', 'GA', 'GB', 'GE', 'GK', 'II', 'LB', and 'NE'.		STCLOOP

Extended Description	Routine	Label

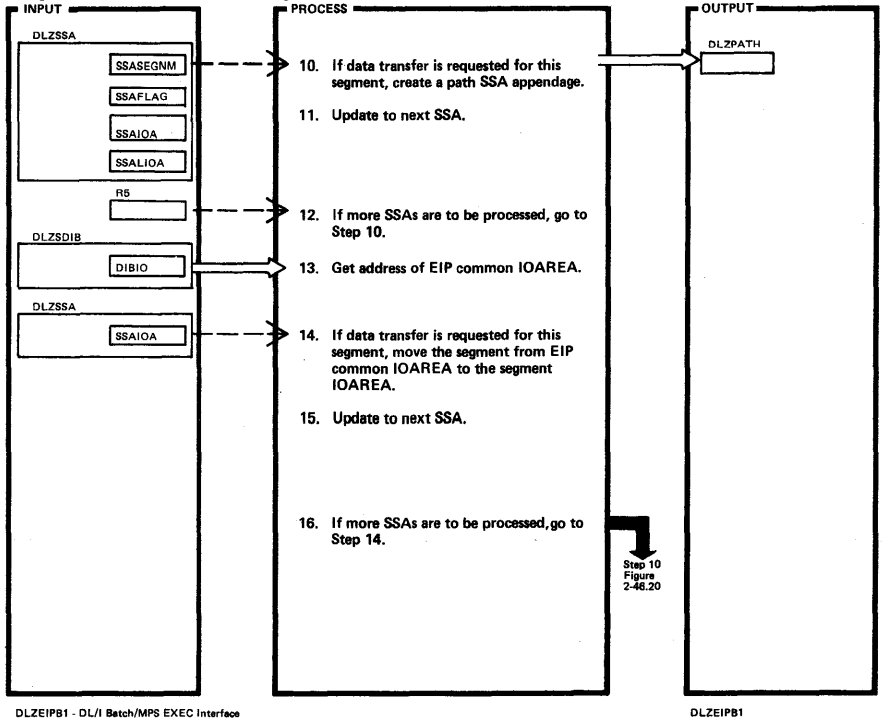
Figure 2-46.19. Get Path Call Processing (Part 1 of 2)



Extended Description	Routine	Label
1.		DBCALLOK
5. DOS/VS GETVIS request issued.		
8. Bit PATHCALL set on in field PATHFLAG.		GOTSSAP

Extended Description	Routine	Label

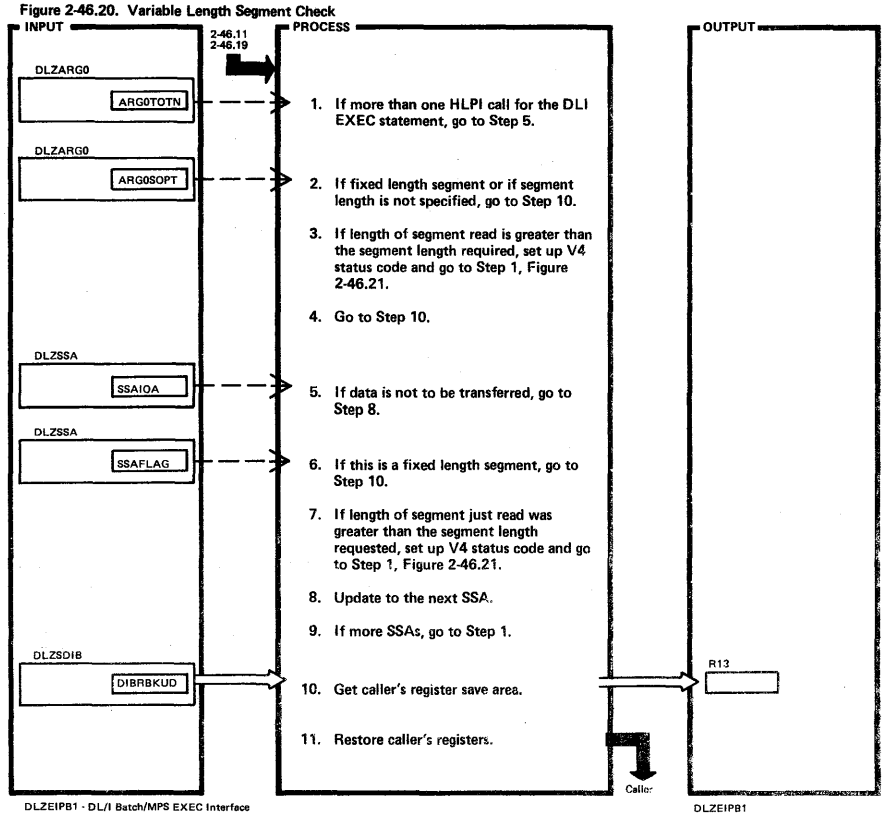
Figure 2-46.19. Get Path Call Processing (Part 2 of 2)



Extended Description	Routine	Label
10.		MOVESEG
12.		CHCKPATH
14.		GETLOOP
16.		NEXTGET

Extended Description	Routine	Label

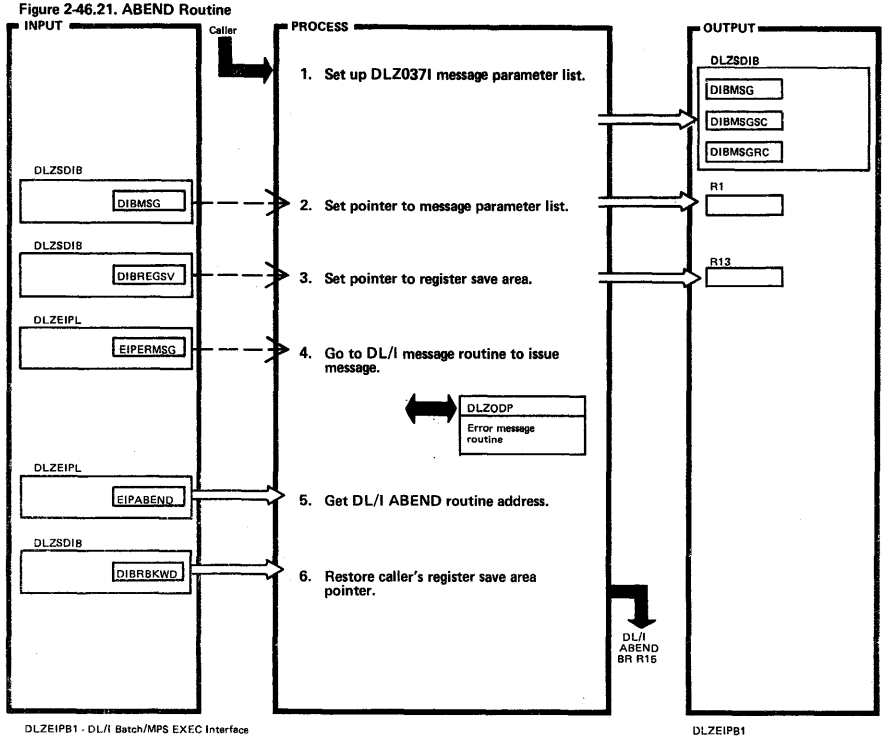
Figure 2-46.20. Variable Length Segment Check



Extended Description	Routine	Label
1.		GET1CHK
3.		NOFFSPEC
5.		IOACHK
7.		CHECKLEN
8.		UPSSA
10.		CALLDONE EIPEXIT

Extended Description	Routine	Label

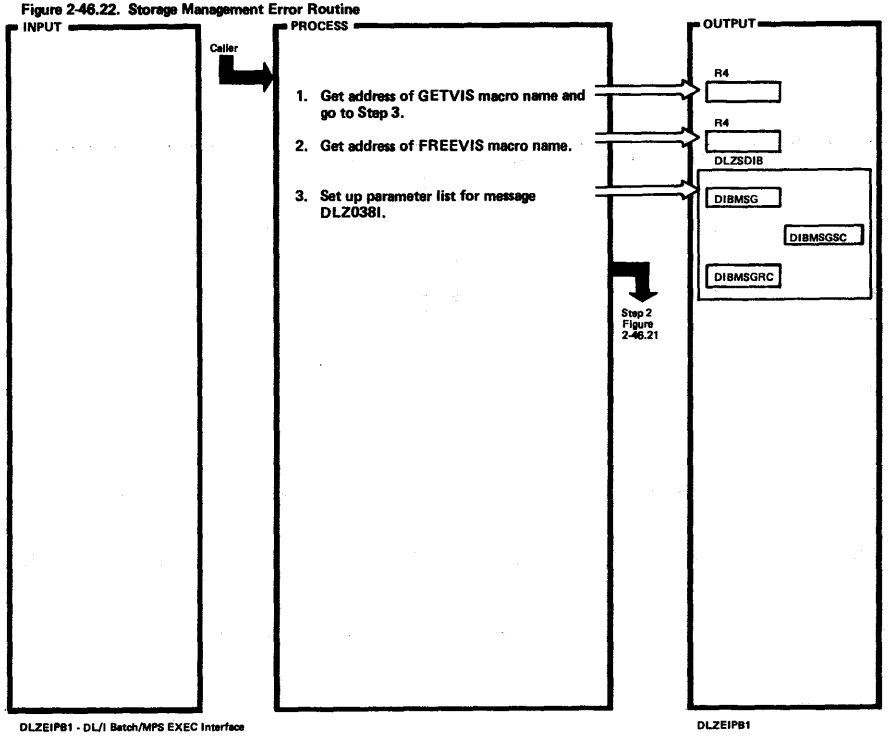
Figure 2-46.21. ABEND Routine



Extended Description	Routine	Label
4. BATCH=ERRORMSG MPS=DLZMMSGX		
5. BATCH=DLZABEND MPS=DLZMABNX		

Extended Description	Routine	Label

Figure 2-46.22. Storage Management Error Routine



Extended Description	Routine	Label
1. Name at label GETID.		GETABEND
2. Name at label FREEID.		FREABEND

Extended Description	Routine	Label

Figure 2-47. DL/I Online EXEC Interface (OVERVIEW) (Part 1 of 3)

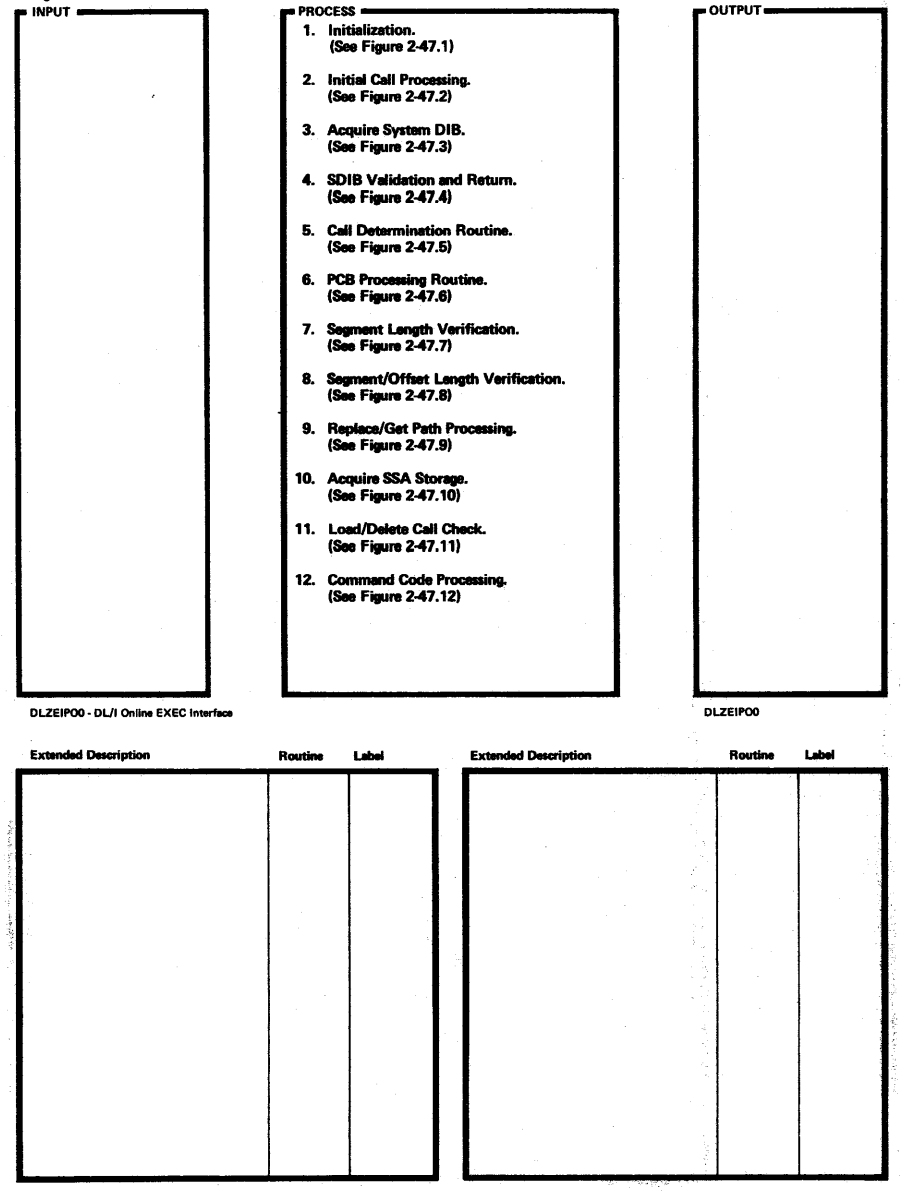


Figure 2-47. DL/I Online EXEC Interface (OVERVIEW) (Part 2 of 3)

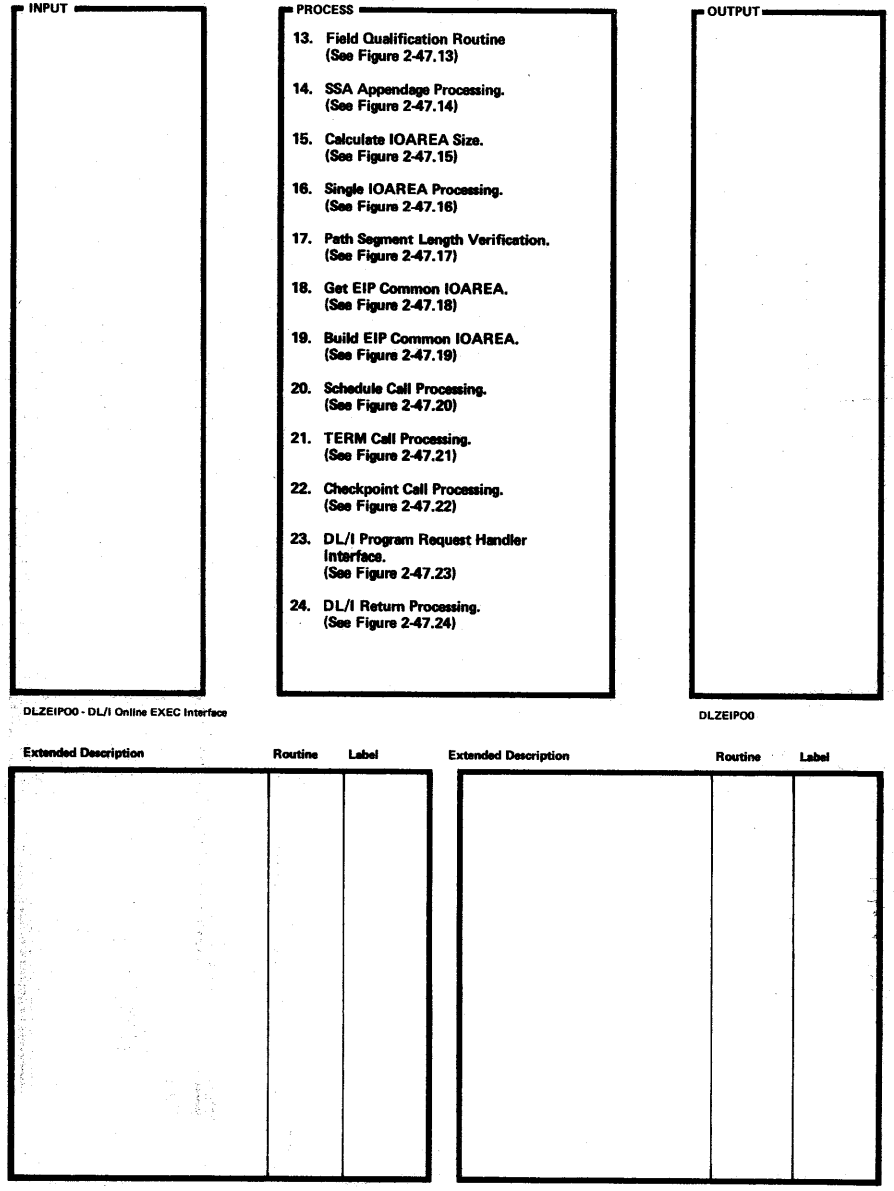


Figure 2-47. DL/I Online EXEC Interface (OVERVIEW) (Part 3 of 3)

INPUT

PROCESS

25. DL/I Psuedo ABEND Processing. (See Figure 2-47.25)

26. DIB Initialization. (See Figure 2-47.26)

27. Get Path Call Processing. (See Figure 2-47.27)

28. Variable Length Segment Check. (See Figure 2-47.28)

29. Invalid DIB Processing. (See Figure 2-47.29)

OUTPUT

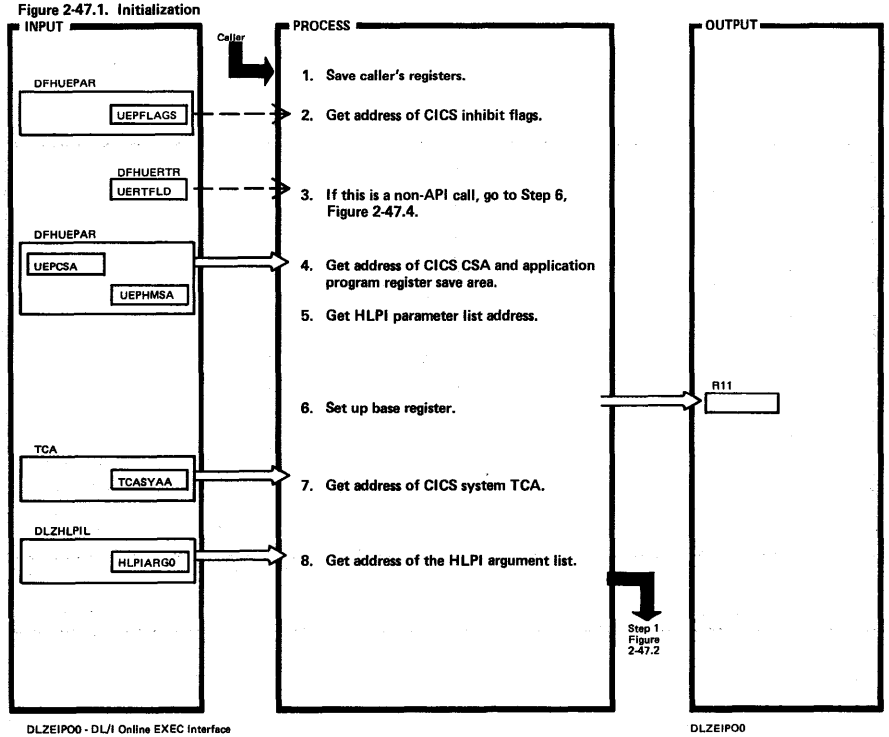
DLZEIP00 - DL/I Online EXEC Interface

DLZEIP00

Extended Description	Routine	Label

Extended Description	Routine	Label

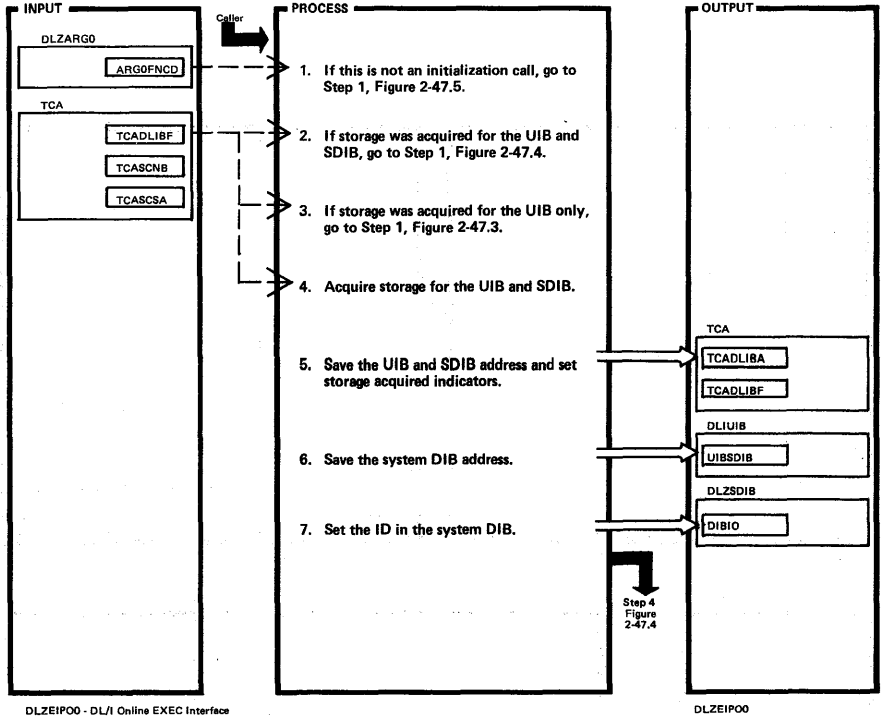
Figure 2-47.1. Initialization



Extended Description	Routine	Label
1. DFHUEPAR is the CICS user interface block whose address is passed in register 1. 5. The HLPI parameter list address is in the sixth fullword of the user's register save area.		DLZEIPI

Extended Description	Routine	Label

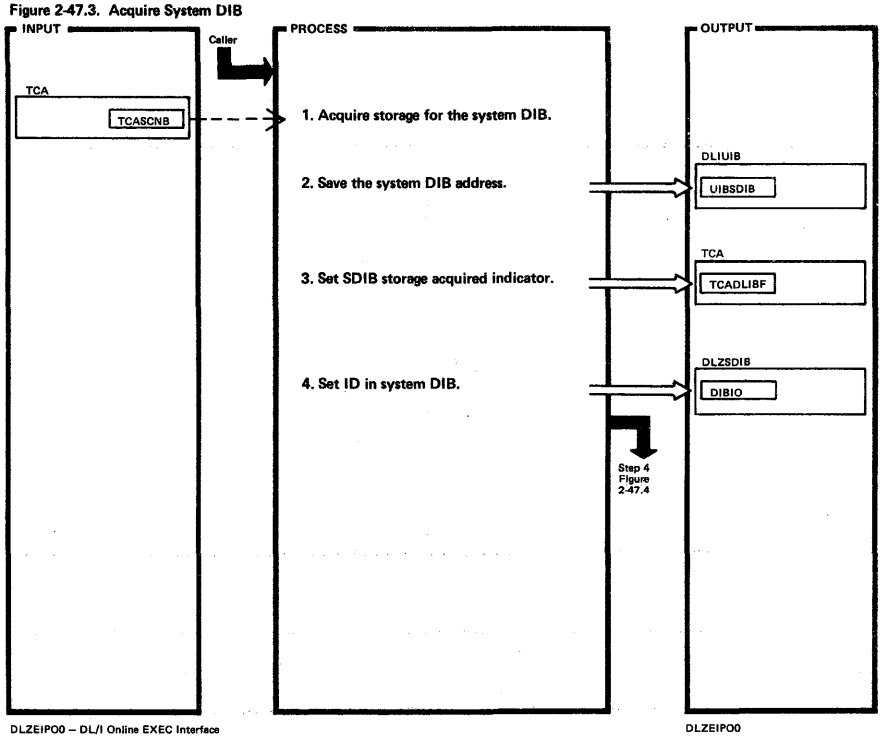
Figure 2-47.2. Initial Call Processing



Extended Description	Routine	Label
1. 2. Bits TCAUIBAQ and TCADIBAQ on. 3. Bit TCAUIBAQ only is on. 4. CICS GETMAIN issued. If unsuccessful, the task is terminated. 7. ID of 'DLZSDIB' initialized in the first seven bytes of the control block.		EIPSTART

Extended Description	Routine	Label

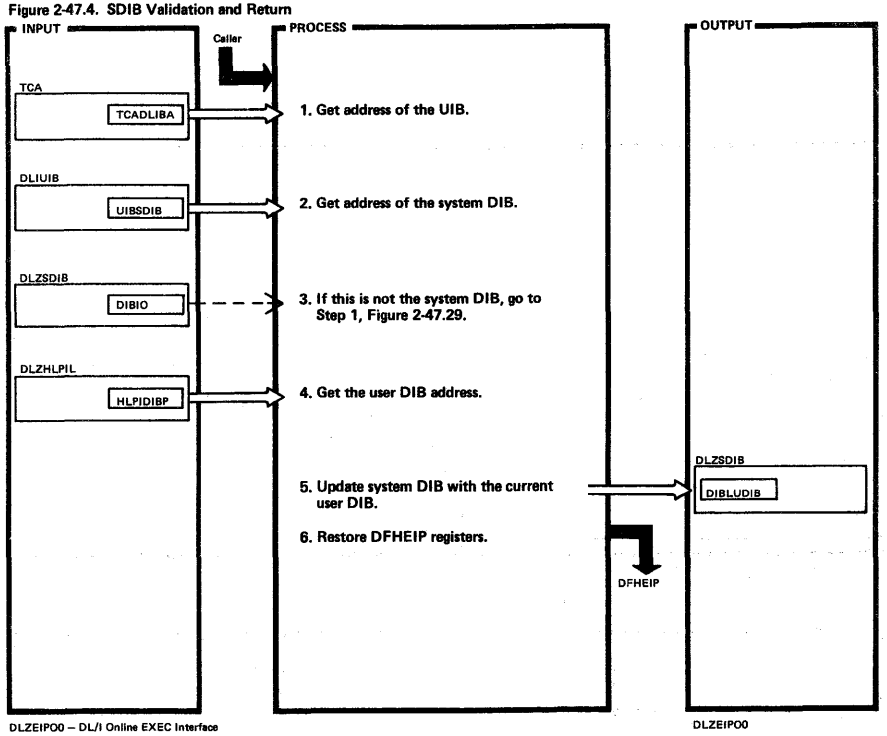
Figure 2-47.3. Acquire System DIB



Extended Description	Routine	Label
1. CICS GETMAIN issued. If unsuccessful, CICS terminates the task. 3. Bit TCADIBAQ set on. 4. ID of 'DLZSDIB' initialized in the first seven bytes of the control block.		AQSDIB

Extended Description	Routine	Label

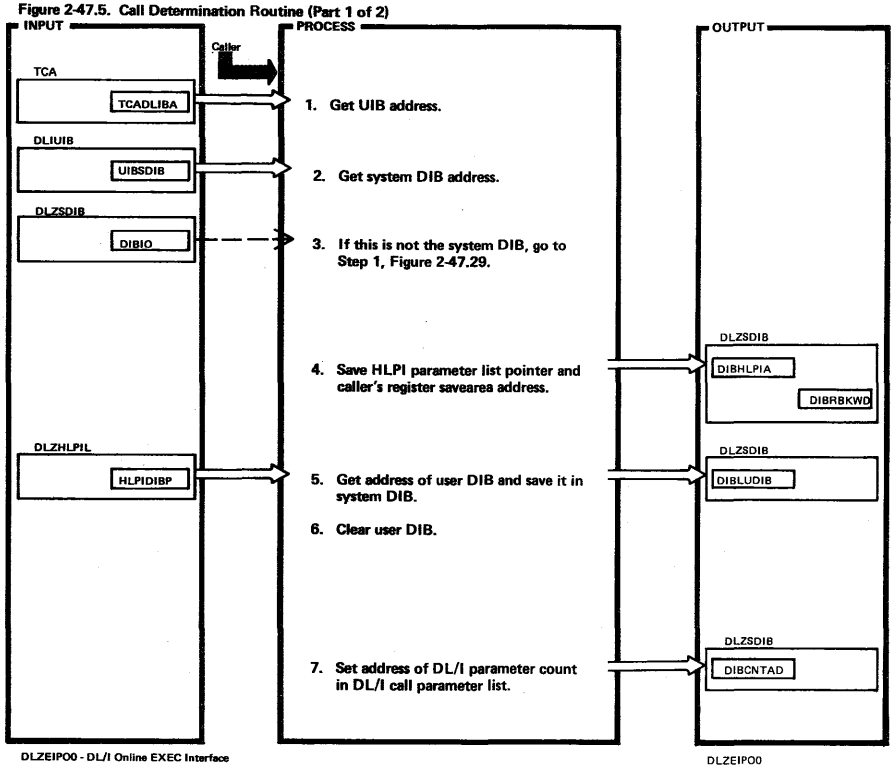
Figure 2-47.4. SDIB Validation and Return



Extended Description	Routine	Label
1. 3. ID of 'DLZSDIB' must be in the first seven bytes of the control block. 4. 6. Returns to DFHEIP.		DIBAVAIL INITEXIT EIPEXIT

Extended Description	Routine	Label

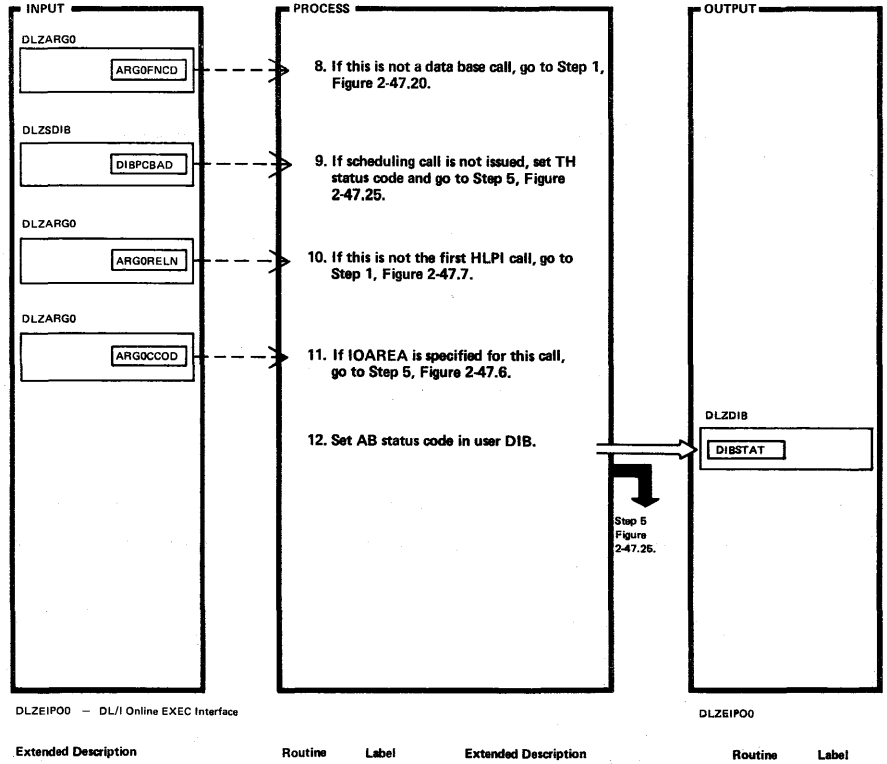
Figure 2-47.5. Call Determination Routine (Part 1 of 2)



Extended Description	Routine	Label
1.		DLZEIPO
4.		GETFNCAL
6. User DIB set to binary zeroes except for version.		

Extended Description	Routine	Label

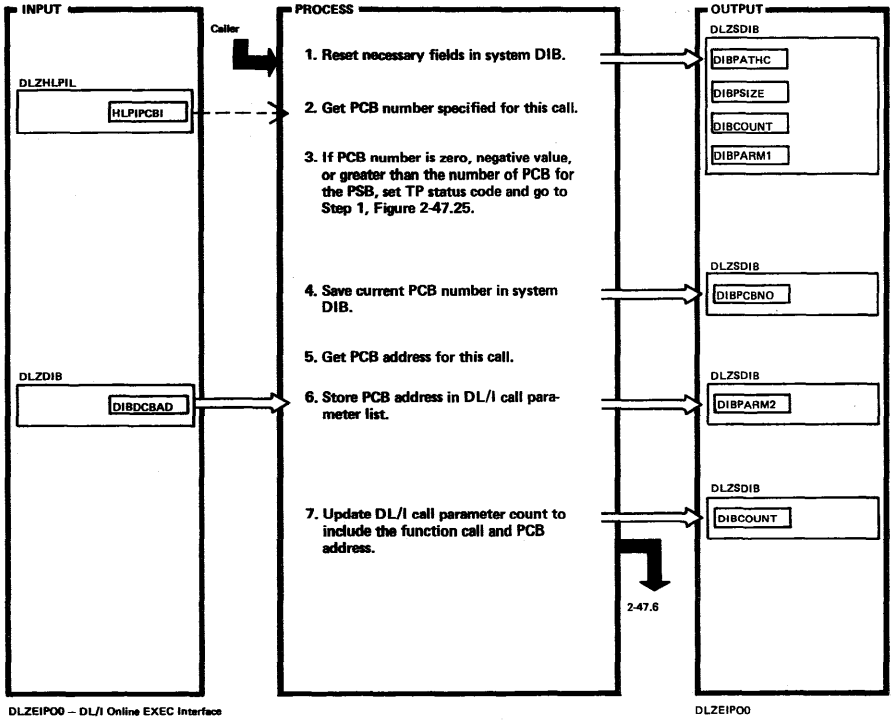
Figure 2-47.5. Call Determination Routine (Part 2 of 2)



Extended Description	Routine	Label
8. Data base calls, as defined here, are get, insert, replace, and delete.		
9. Scheduling call must be first DL/I call request.		
10. The first call for an EXEC DL/I statement passed is actually the call for the object segment.		
12. AB status code equals segment IOAREA required for object segment.		

Extended Description	Routine	Label

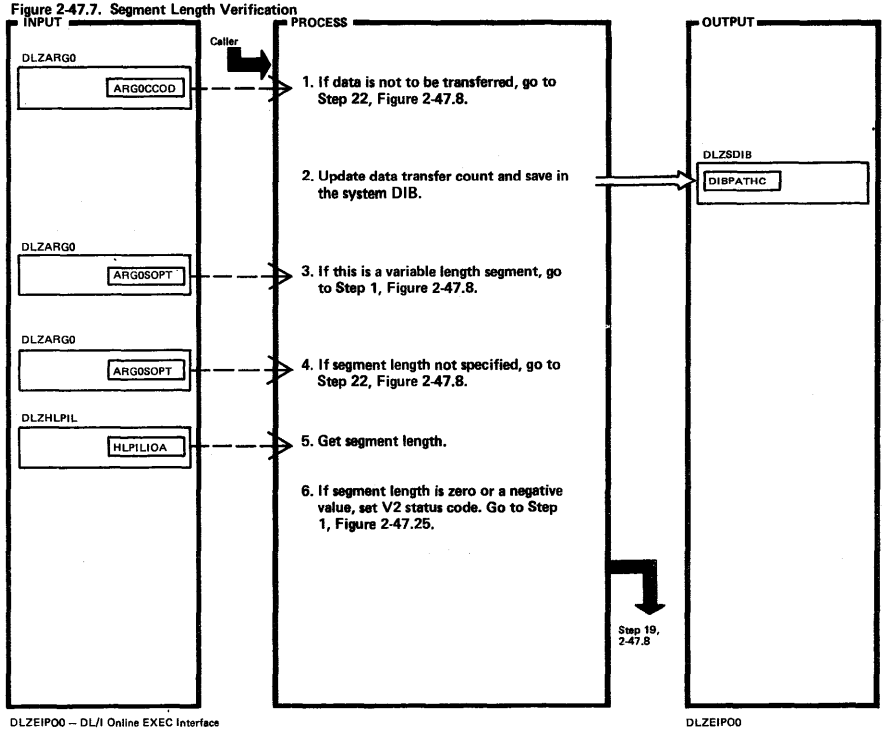
Figure 2-47.6. PCB Processing Routine



Extended Description	Routine	Label
1. On the first HLPI call for each EXEC DLI statement the path count, current required IOAREA size, the DL/I parameter count are set to binary zeroes and the DL/I function-call is set in the DL/I call parameter list.		FRSTIOOK
5. Using the PCB number, index into the PCB list for the current PCB address.		SETPCBPR

Extended Description	Routine	Label

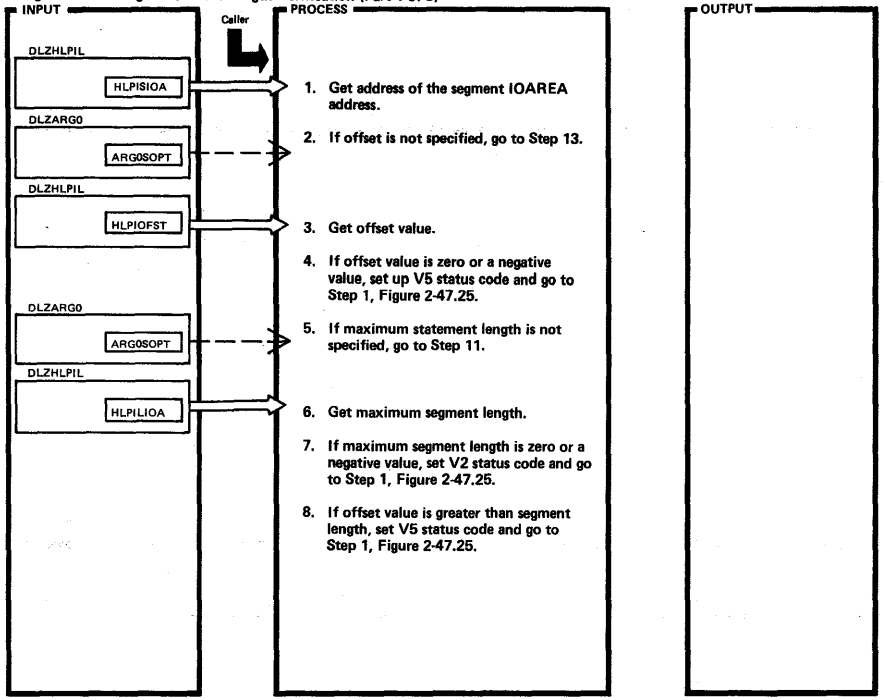
Figure 2-47.7. Segment Length Verification



Extended Description	Routine	Label
1. Bit CCINFROM on in byte ARG0CCOD.		CHKTRANS
2. A count is kept for each call that has requested data transfer for the EXEC DLI statement.		
3. Bit OPTVAR on in byte ARG0SOPT.		
4. Bit OPTSEGL on in byte ARG0SOPT.		FXSGSIZE
5. Field HLPILIOA contains a pointer to the segment length.		

Extended Description	Routine	Label

Figure 2-47.8. Segment/Offset Length Verification (Part 1 of 3)

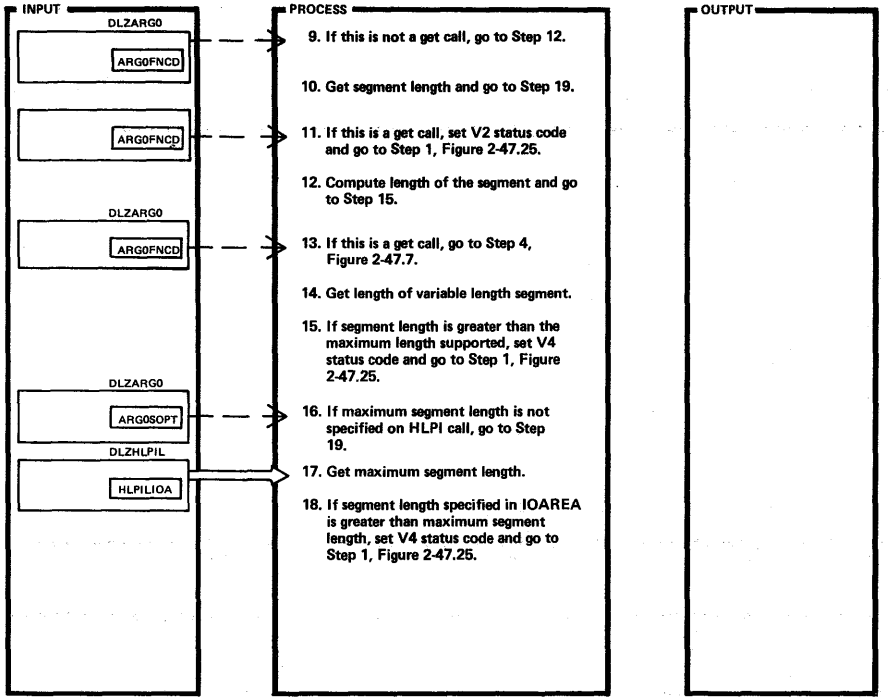


DLZEIP00 - DL/I Online EXEC Interface

Extended Description	Routine	Label
1.		VRSGSIZE
2. Bit OPTOFF not on in byte ARG0SOPT.		
3. Field HLPIOFST contains a pointer to the OFFSR value. This value is the length of the concatenated key plus the length of the intersection data (if any).		
6. Field HLPILTOA contains a pointer to the segment length.		

Extended Description	Routine	Label

Figure 2-47.8. Segment/Offset Length Verification (Part 2 of 3)

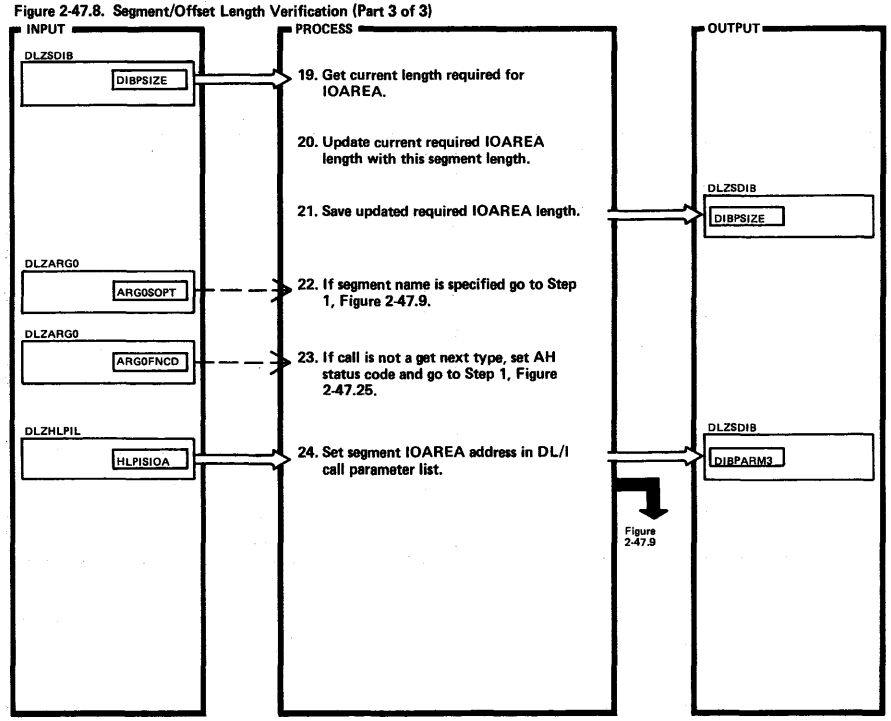


DLZEIP00 — DL/I Online EXEC Interface

Extended Description	Routine	Label
9. This check is made for all get type call GN, GU, and GNP.		
11.		VGETCALL
12. This length includes the concatenated key, intersection data, and the segment.		VRSGCON
13.		NOFFSET
14. This length is in the first two bytes of the segment IOAREA.		
15.		MAXCHECK
16. Bit OPTSFGL not on in field ARG0SOPT.		
17. Field HLPILIOA contains a pointer to maximum segment length.		

Extended Description	Routine	Label

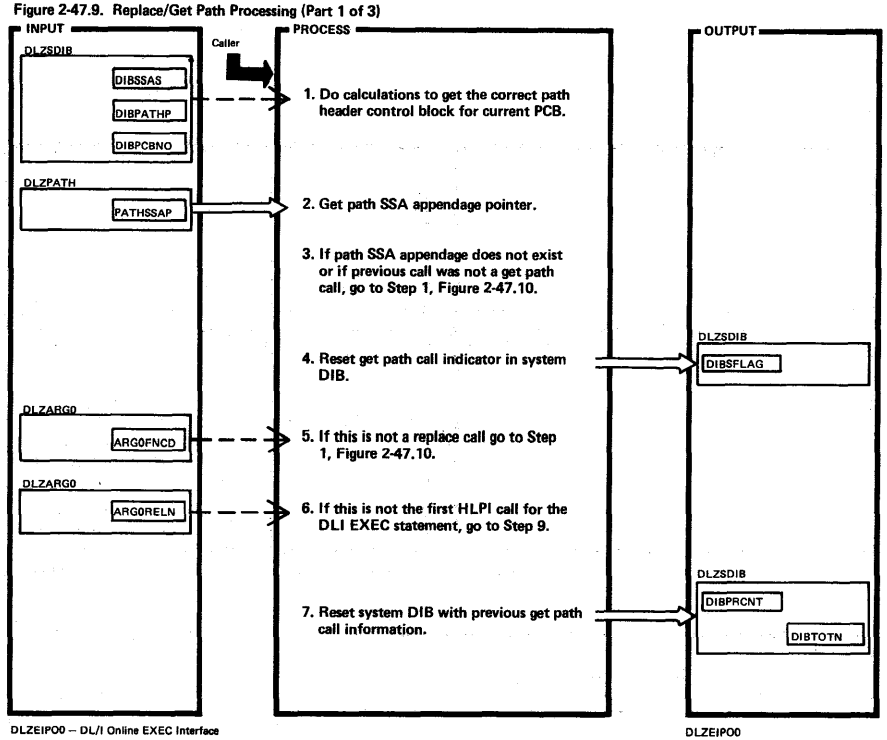
Figure 2-47.8. Segment/Offset Length Verification (Part 3 of 3)



Extended Description	Routine	Label
19.		SETSGSIZ
22. Bit OPTSEGM on in field ARG0SOPT.		NOTRMNSF

Extended Description	Routine	Label

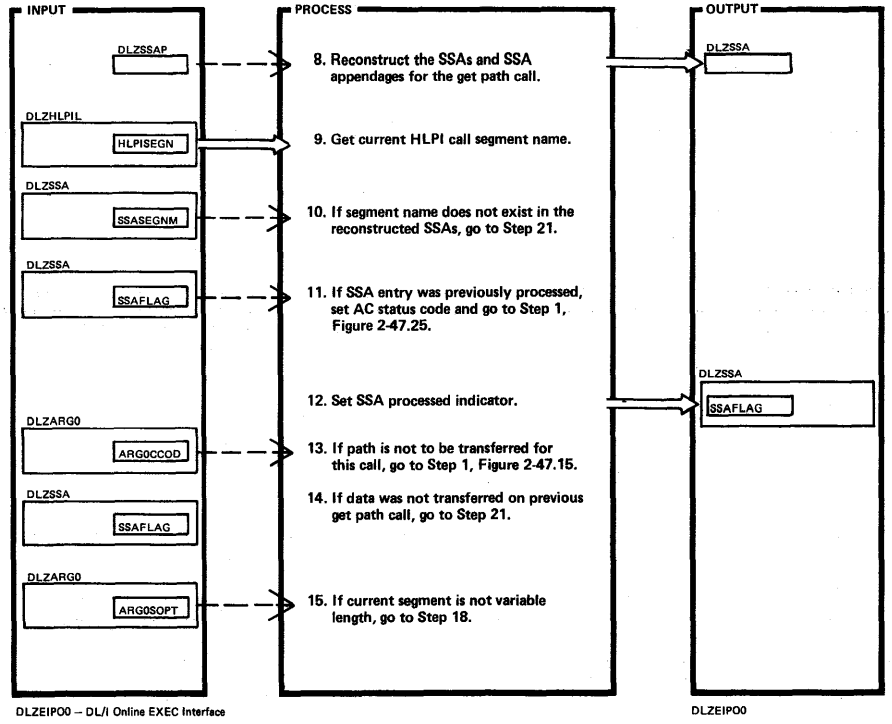
Figure 2-47.9. Replace/Get Path Processing (Part 1 of 3)



Extended Description	Routine	Label
1. Current PCB number-1 x length of the path header control block. 4. Bit DIBGPATH set ON in field DIBSFLAG. 6. If the previous call was a get path call and current call is a replace call and this is the first HLPI call for the DLI EXEC statement, the SSAs and SSA appendages are reconstructed based on the get path call.		STARTSSA

Extended Description	Routine	Label

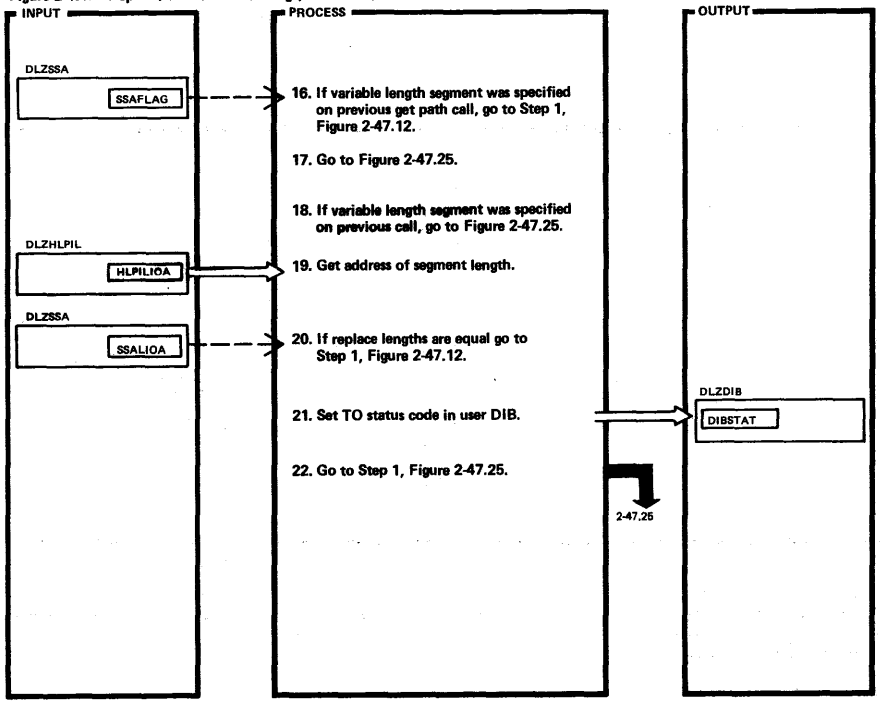
Figure 2-47.9. Replace/Get Path Processing (Part 2 of 3)



Extended Description	Routine	Label
8. SSAs are reconstructed with segment name, n command code, and blank delimiter. 9. HLPISEGN contains a pointer to the segment name. 10. 11. Bit SSAPROC not set in field SSAFLAG. 12. Bit SSAPROC set ON in field SSAFLAG. This is done to ensure duplicate segment names are not specified in current call. 13. Bit CCINFROM not on in field ARG0CCON. 14. Bit SSADATAT not on in field SSAFLAG. 15. Bit OPTVAR not on in field ARG0SOPT.		RELOOP SEGNCHK SEGLOOP SSAFOUND SSAOK

Extended Description	Routine	Label

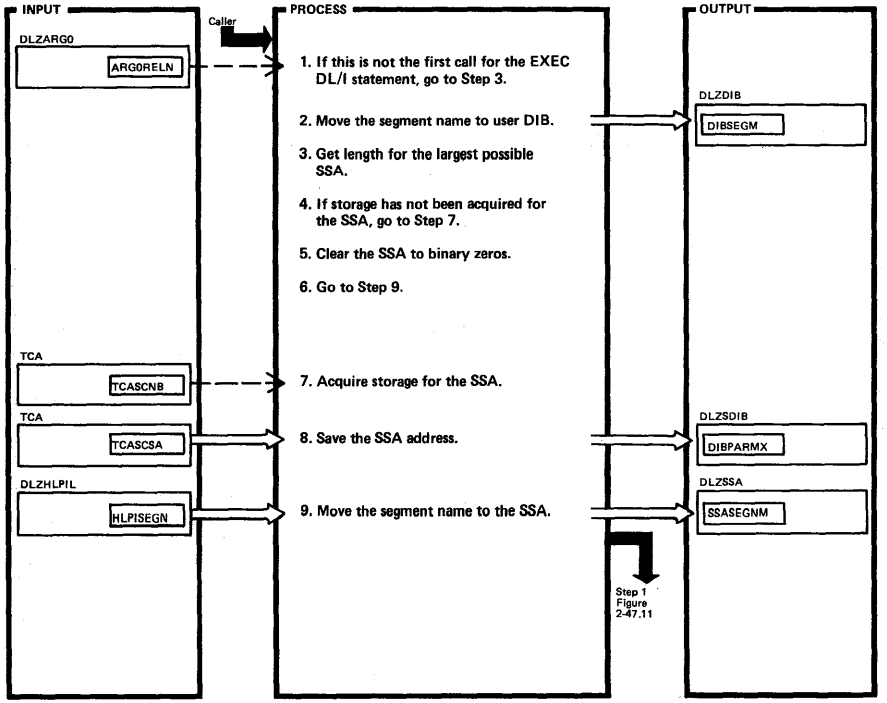
Figure 2-47.9. Replace/Get Path Processing (Part 3 of 3)



Extended Description	Routine	Label
16. Bit SSAVARL on in byte SSAFLAG.		
18.		SSANOTV
19. HLPILIOA contains a pointer to the segment length.		
21. Status code 'TO' indicates replace/get path calls inconsistent.		ERRORTO

Extended Description	Routine	Label

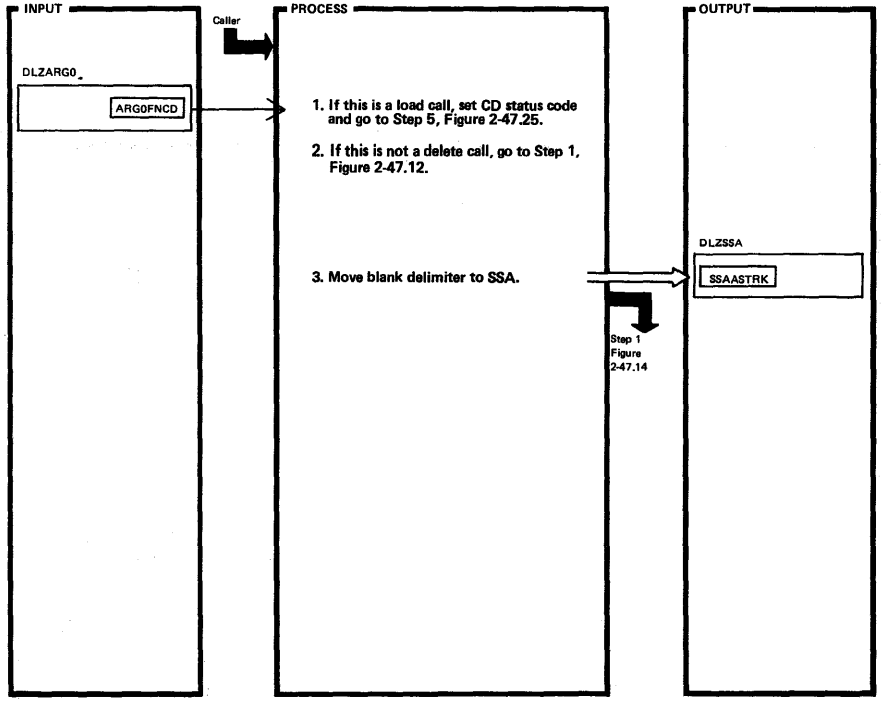
Figure 2-47.10. Acquire SSA Storage



Extended Description	Routine	Label
1. Relative number in field ARG0RELN is one.		SSANPATH
2. The first HLPI call for the EXEC DLI statement is for the object segment.		
3.		SUBTWO
7. CICS GETMAIN issued. If failure occurs, CICS terminates the task.		
8. The SSA pointer area in the DLI parameter list in the system DIB where the SSA pointer is stored is dependent on the number of the HLPI call.		
9. Field HLPISEGM contains a pointer to the segment name.		SSAAQOK

Extended Description	Routine	Label

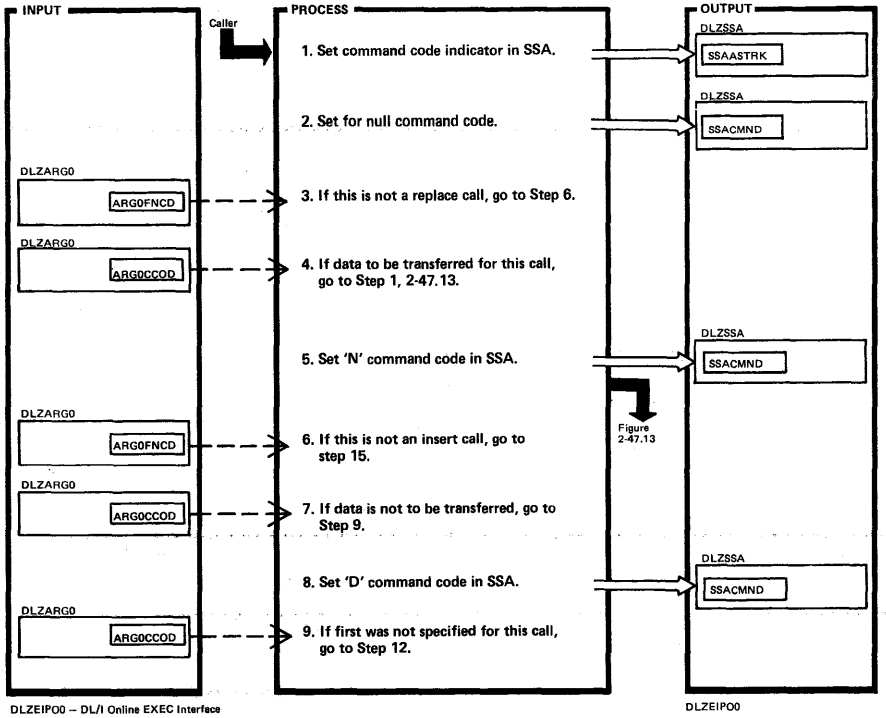
Figure 2-47.11. Load/Delete Call Check



Extended Description	Routine	Label
1. Load call is invalid in online environment.		
2. Delete call is unqualified.		

Extended Description	Routine	Label

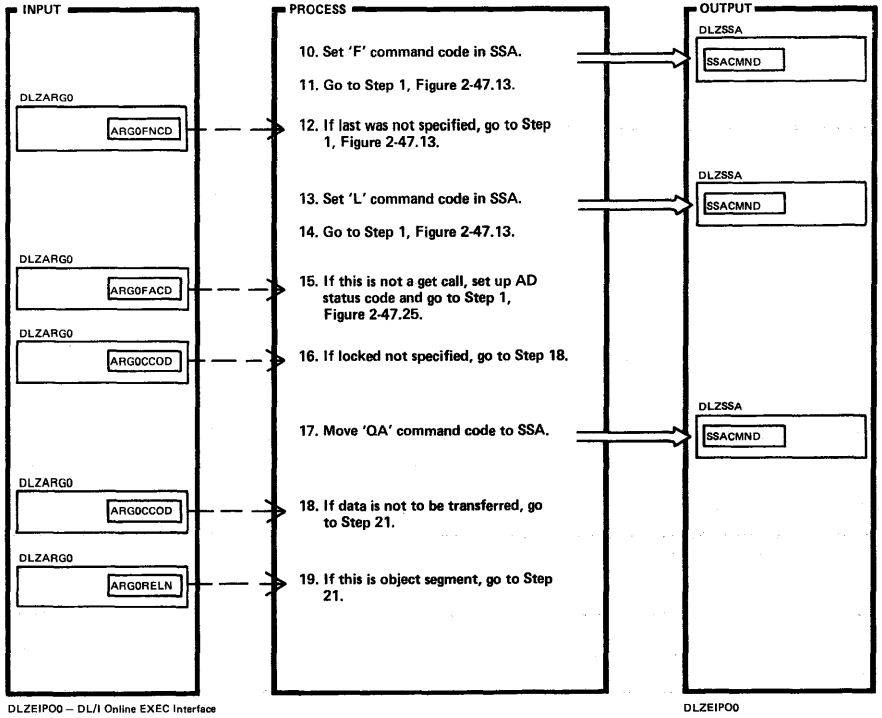
Figure 2-47.12. Command Code Processing (Part 1 of 3)



Extended Description	Routine	Label
1. Command code indicator is an astrisk (*).		CONTSSA
2. Null command codes are dashes (-).		
3. Replace call code equals X'14'.		
4. Bit CCINFROM on in byte ARG0CCOD.		
5. 'N' command code indicates that this segment is not to be replaced.		
6. Function code is not equal to X'12'.		CKCCISRT
7. Bit CCINFROM not on in byte ARG0CCOD.		
8. 'D' indicates multiple insert path call.		
9. Bit CCFIRST not on in byte ARG0CCOD.		

Extended Description	Routine	Label

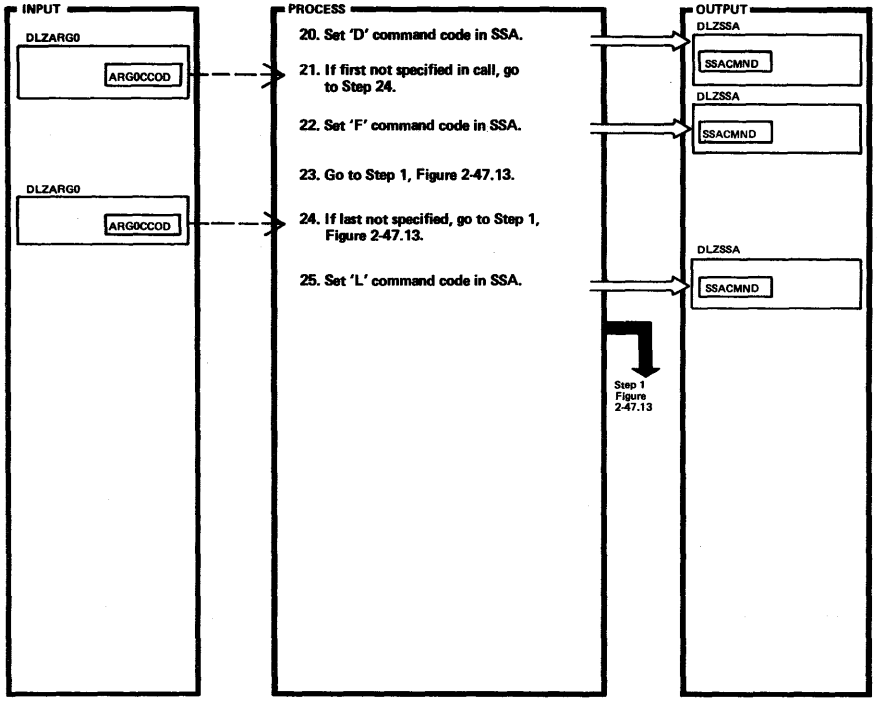
Figure 2-47.12. Command Code Processing (Part 2 of 3)



Extended Description	Routine	Label
10. 'F' indicates start with the first occurrance of this segment type to satisfy this level of call.		
12. Bit CCLAST not on is field ARG0CCOD.		CKCCLSTI
13. 'L' indicates use last occurance of segment type to satisfy this level of call.		
15. Function code is not in range of X'10' and X'0A'.		CKCCG0
16. Bit CCLOCKED not on in byte ARG0CCOD.		
17. 'QA' indicates to lock segment(s) returned to prevent modification by another task.		

Extended Description	Routine	Label

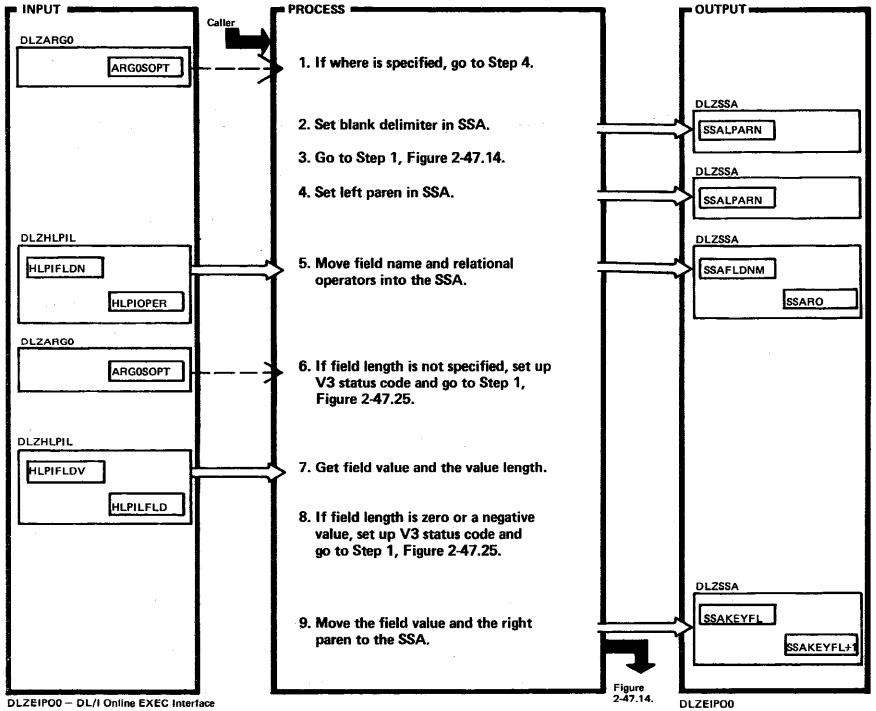
Figure 2-47.12. Command Code Processing (Part 3 of 3)



Extended Description	Routine	Label
21.		CKCCISTG
24.		CKCCLSTG

Extended Description	Routine	Label

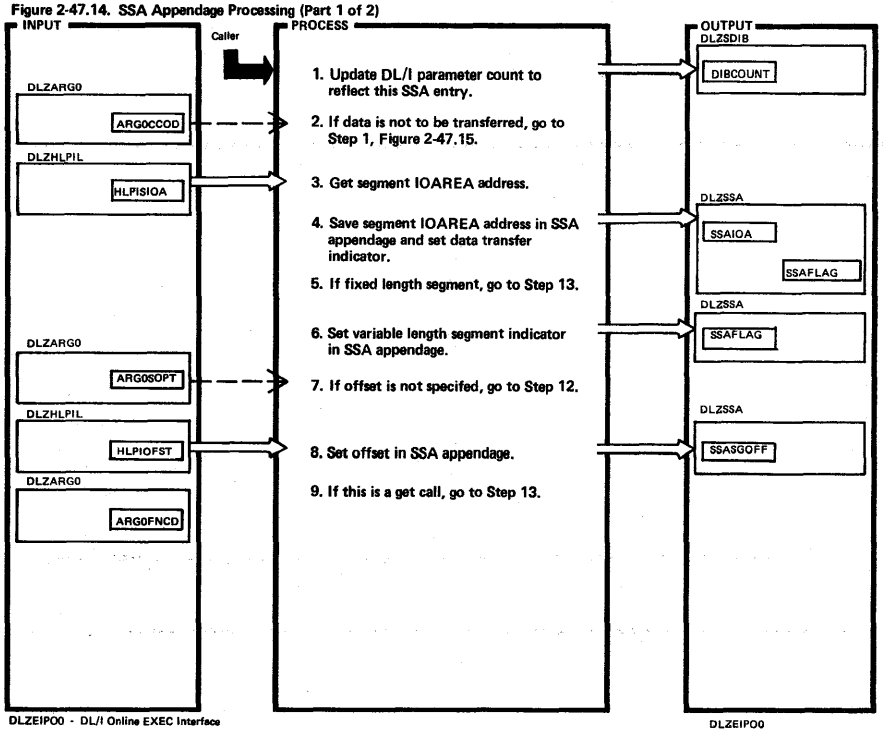
Figure 2-47.13. Field Qualification Routine



Extended Description	Routine	Label
1. Bit OPTWHERE set on in byte ARGOSOPT.		FIELDCHK
2. Blank indicates unqualified SSA.		
4. Left paren indicates qualified SSA.		QUALSSA
5. Field HLPIFLDN has a pointer to the field name and HLPIOPER has a pointer to the relational operators.		
6. Bit OPTFLDL not on in byte ARGOSOPT.		
7. Field HLPIFLDV is a pointer to the field value and HLPILFLD is a pointer to the field value length.		FLDLNOK
9. Right paren is delimiter for qualification of SSA.		

Extended Description	Routine	Label

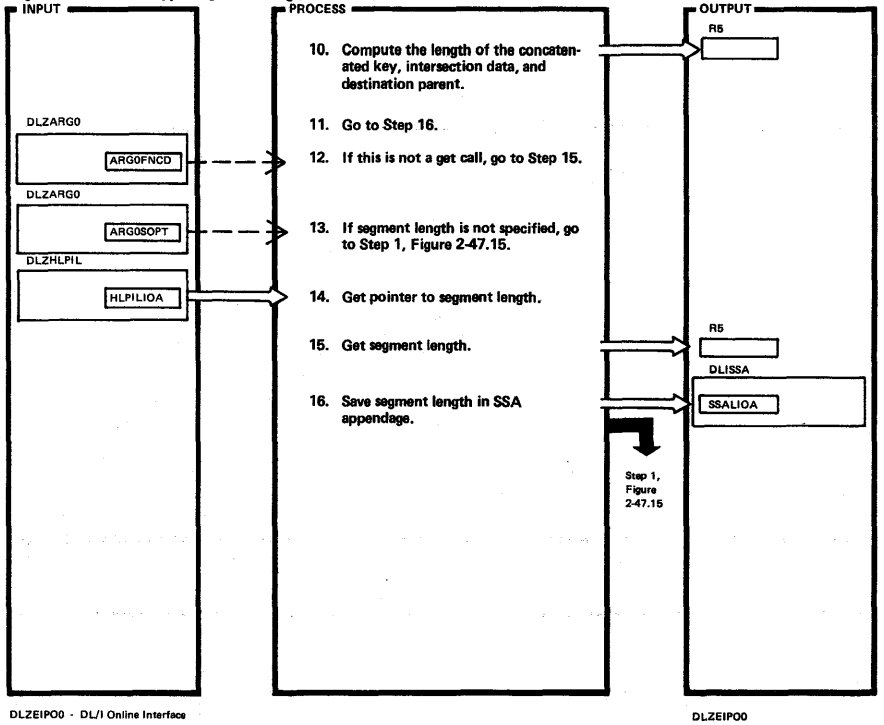
Figure 2-47.14. SSA Appendage Processing (Part 1 of 2)



Extended Description	Routine	Label
1.		BMPSSA
2. Bit CCINFROM not on in byte ARG0CCOD.		
3. Field HLPISIOA contains the address of the segment I/O area.		
4. Bit SSADATAT indicates data transfer in the SSA appendage.		
6. Bit SSAVARL in the SSA appendage indicates variable length segment.		VARSEGL
7. Bit OPTOFF not on in byte ARG0SOPT.		
8. HLPIOFST contains a pointer to the offset.		

Extended Description	Routine	Label

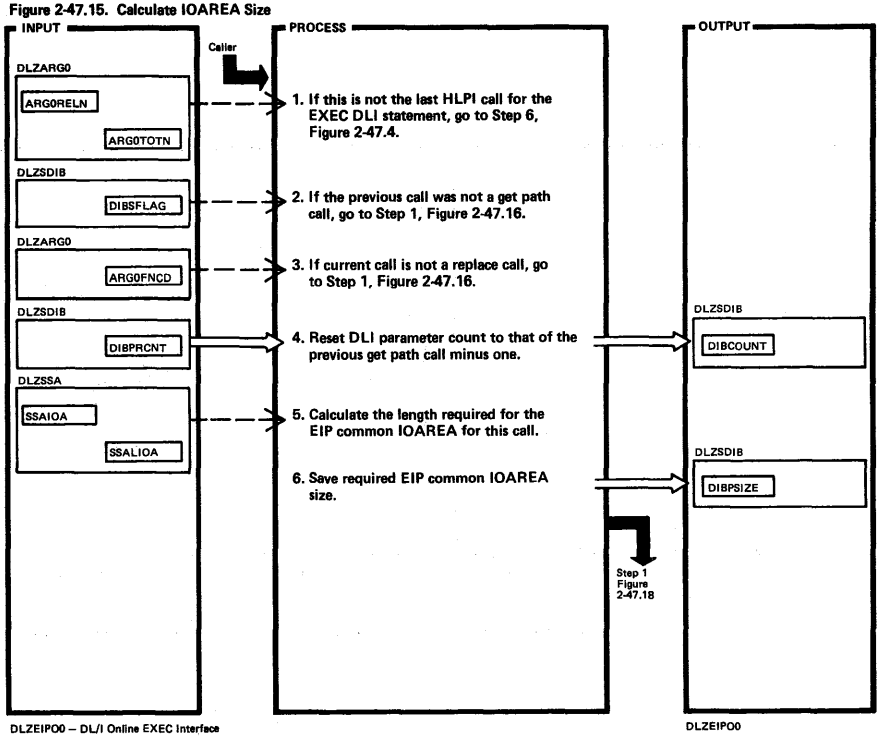
Figure 2-47.14. SSA Appendage Processing (Part 2 of 2)



Extended Description	Routine	Label
12.		FIXSEGL
13. Bit OPTSEGL not on in byte ARG0SOPT.		FIXSEG
14. HLPILIOA contains a pointer to the segment length.		
15.		SETSEGLN
16.		SETLNCIS

Extended Description	Routine	Label

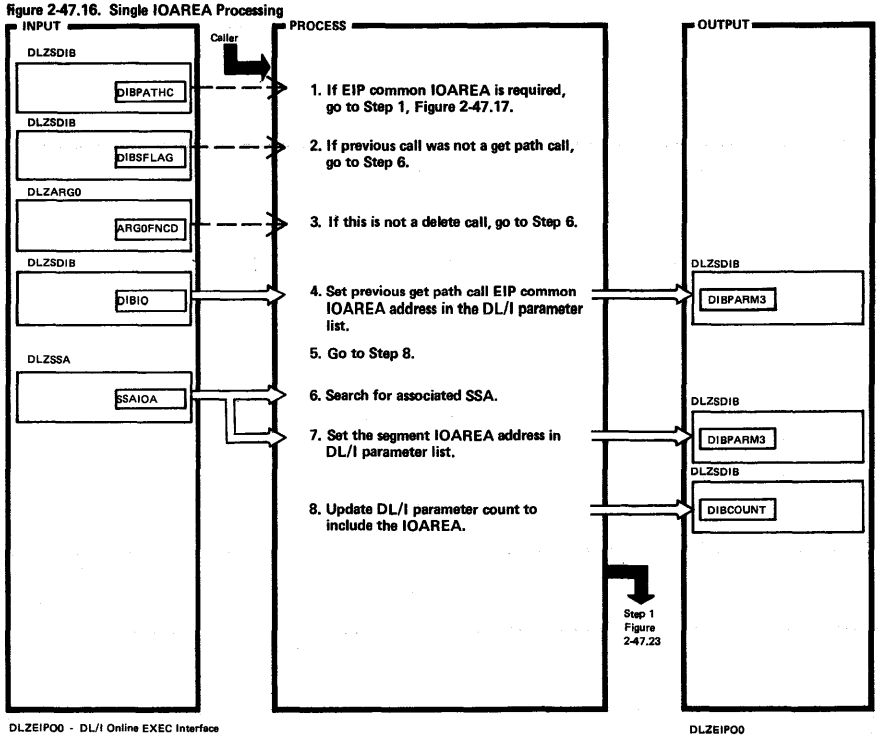
Figure 2-47.15. Calculate IOAREA Size



Extended Description	Routine	Label
1. ARG0RELN contains the relative number of this call and ARG0TOTN contains the total number of calls for the EXEC DLI statement.		ANYMORE
2. Bit DIBGPATH call not on in byte DIBSFLAG.		
5. A scan is made of all SSAs associated with this call, adding the length of each segment that has data transfer required in the SSA appendage.		PROCNEXT

Extended Description	Routine	Label

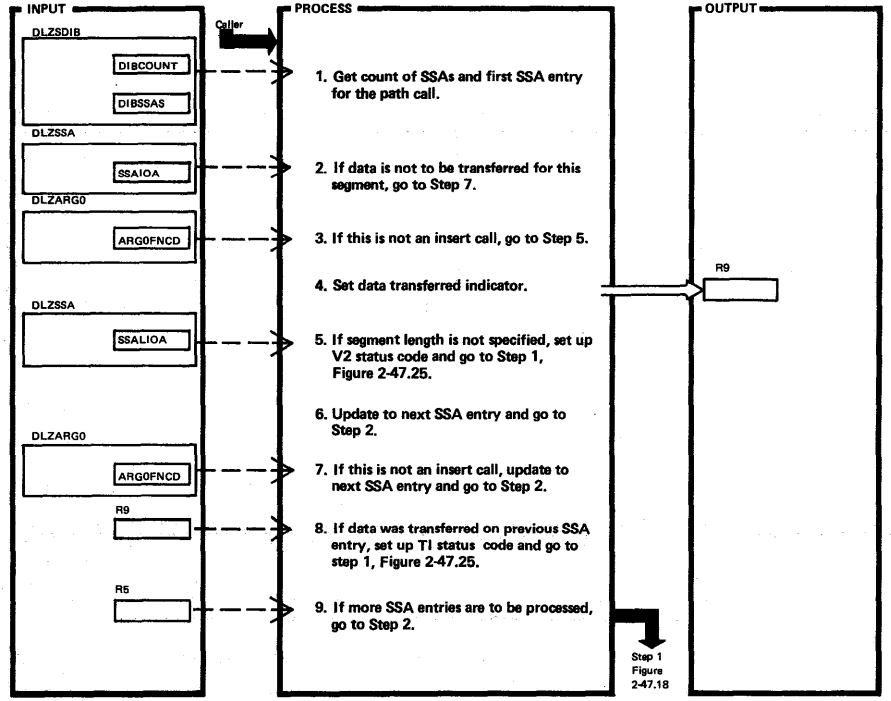
Figure 2-47.16. Single IOAREA Processing



Extended Description	Routine	Label
1. DIBPATHC contains the number of IOAREAs specified for this call.		GETIOARE
2. Bit DIBGPATH not on in byte DIBSFLAG.		
3. If delete call follows a get path call, the IOAREA must reflect the way it was after the get path call.		
6.		SEG1SRCH
7.		SEG1FND
8.		NOSSA

Extended Description	Routine	Label

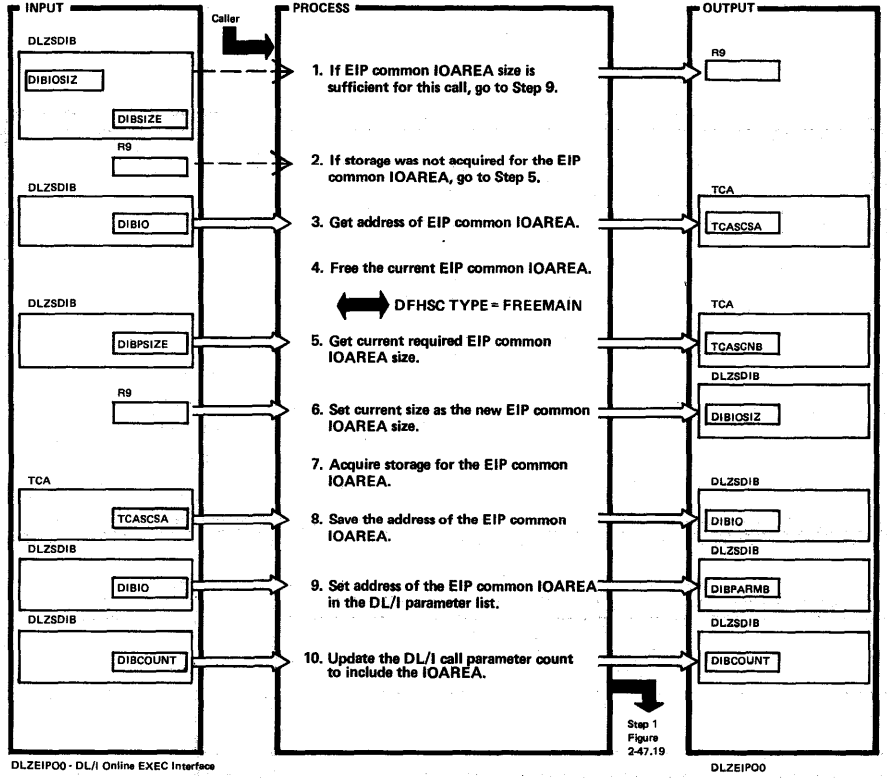
Figure 2-47.17. Path Segment Length Verification



Extended Description	Routine	Label
1. Get set to scan SSAs.		LENVERY
2.		VERFYLEN
5. Segment length must be specified for every segment that has data transfer in a path call.		CHKSEGLN
7.		CHKISRT
8. For insert calls, data transfer must be specified for every segment from the first one encountered to the object segment.		

Extended Description	Routine	Label

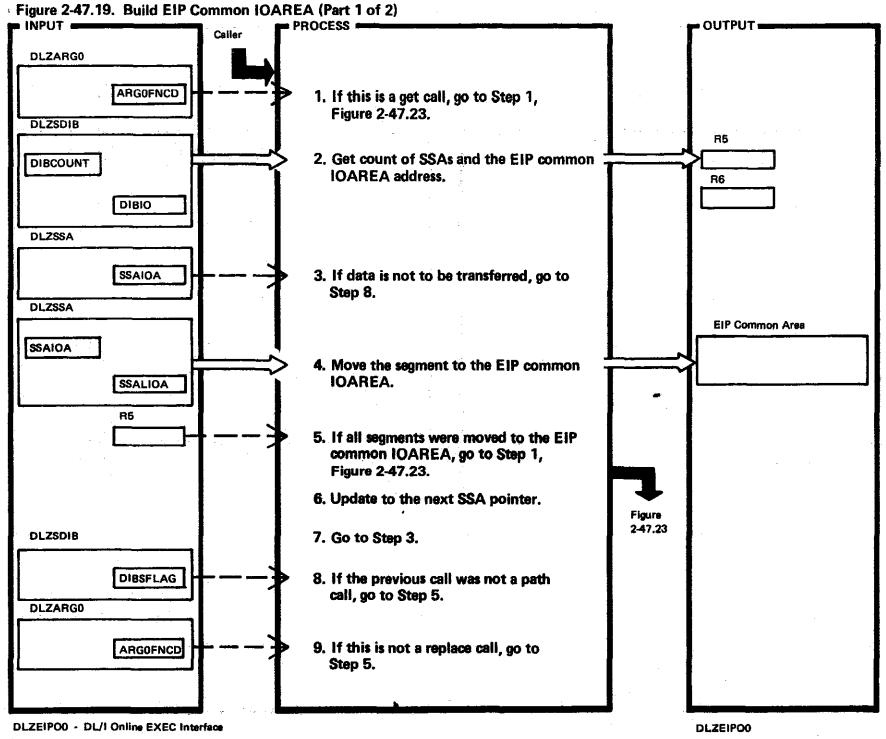
Figure 2-47.18. Get EIP Common IOAREA



Extended Description	Routine	Label
1. If the storage size (DIBPSIZE) calculated for this call is less than or equal to the size of the existing EIP common IOAREA, use the existing size.		PROVDIOA
4. CICS FREEMAIN issued. If unsuccessful, CICS terminates the task.		
5.		GETEIPIO
7. CICS GETMAIN issued. If unsuccessful, CICS terminates the task.		
8. Address of the area processed by GETMAIN is returned in the TCA.		
9.		EIPIOAOK

Extended Description	Routine	Label

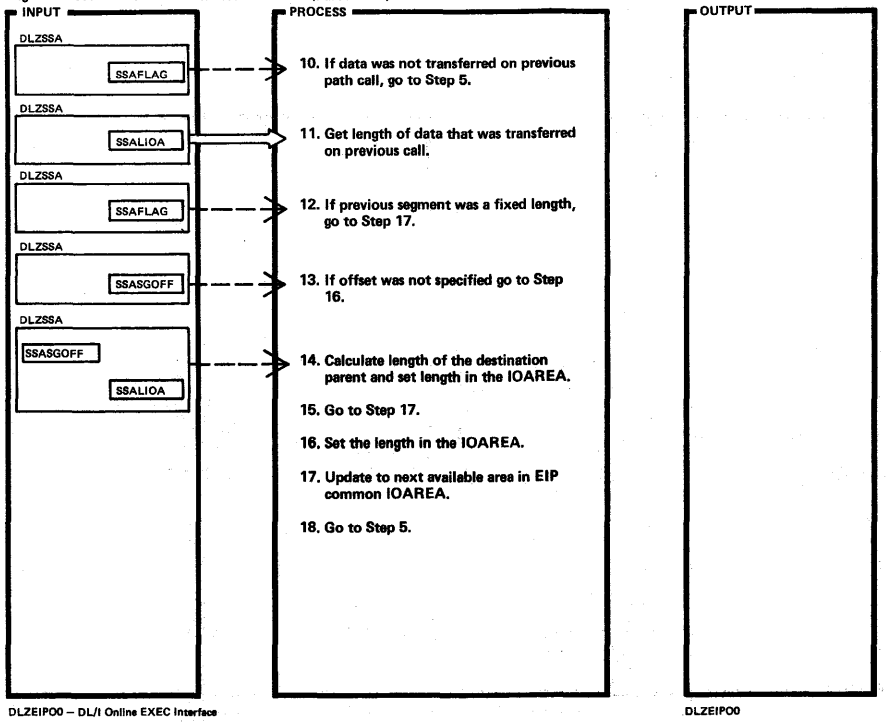
Figure 2-47.19. Build EIP Common IOAREA (Part 1 of 2)



Extended Description	Routine	Label
2. SSA count is set in register 5 and EIP common IOAREA address is set in register 6.		
3.		MOVELOOP
5.		MOVECHK
6.		MOVENEXT
8. Bit DIBGPATH not on in byte DIBSFLAG.		NEXTMOVE

Extended Description	Routine	Label

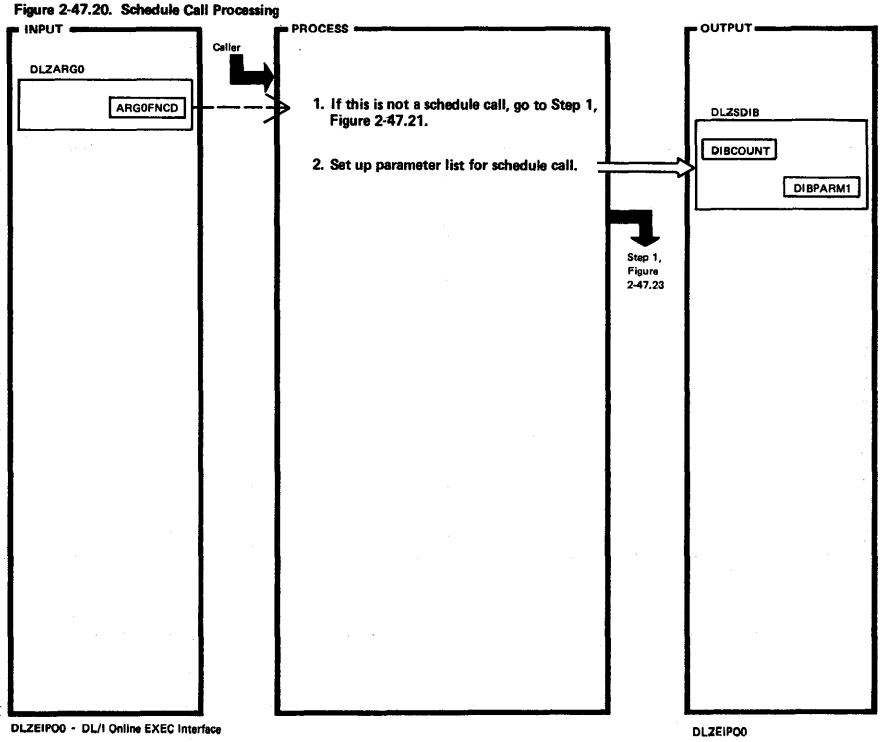
Figure 2-47.19. Build EIP Common IOAREA (Part 2 of 2)



Extended Description	Routine	Label
12. Bit SSAVARL not on in byte SSAFLAG.		
14. The length of a variable destination parent must be in the first two bytes of the segment that is after the concatenated key and intersection data.		
16. For variable length segments the length must appear in the first two bytes of the segment.		MOVEVLEN
17.		MOVEFIX

Extended Description	Routine	Label

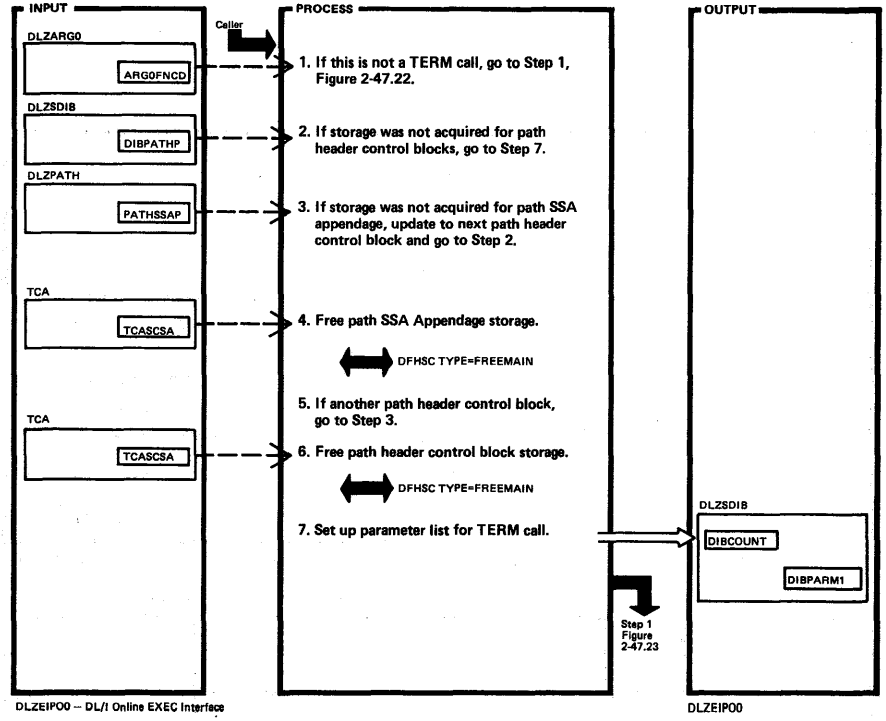
Figure 2-47.20. Schedule Call Processing



Extended Description	Routine	Label
1. Scheduling function code is X'04'.		CALLSCHD

Extended Description	Routine	Label

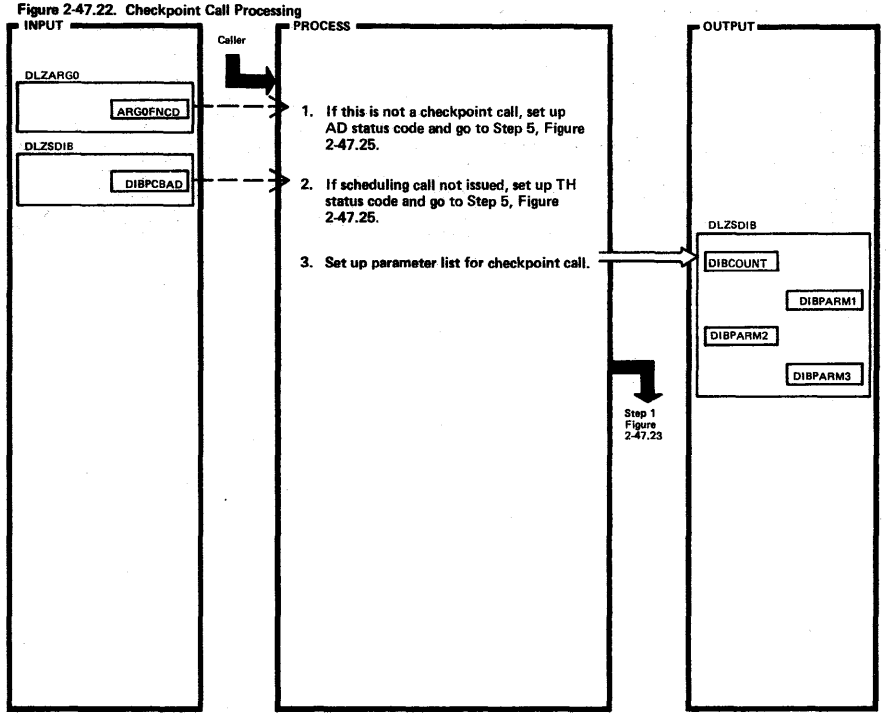
Figure 2-47.21. TERM Call Processing



Extended Description	Routine	Label
1. TERM function code is X'06'.		CALLTERM
3.		FREELOOP
7.		CONTTERM

Extended Description	Routine	Label

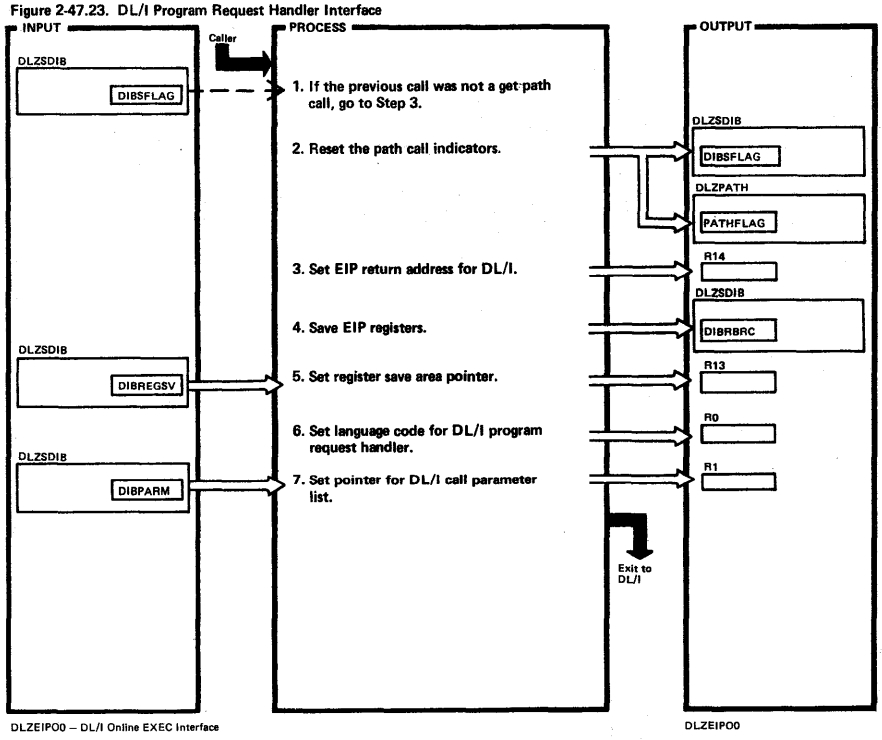
Figure 2-47.22. Checkpoint Call Processing



Extended Description	Routine	Label
1. Checkpoint function code is X'08'.		CALLCHKP

Extended Description	Routine	Label

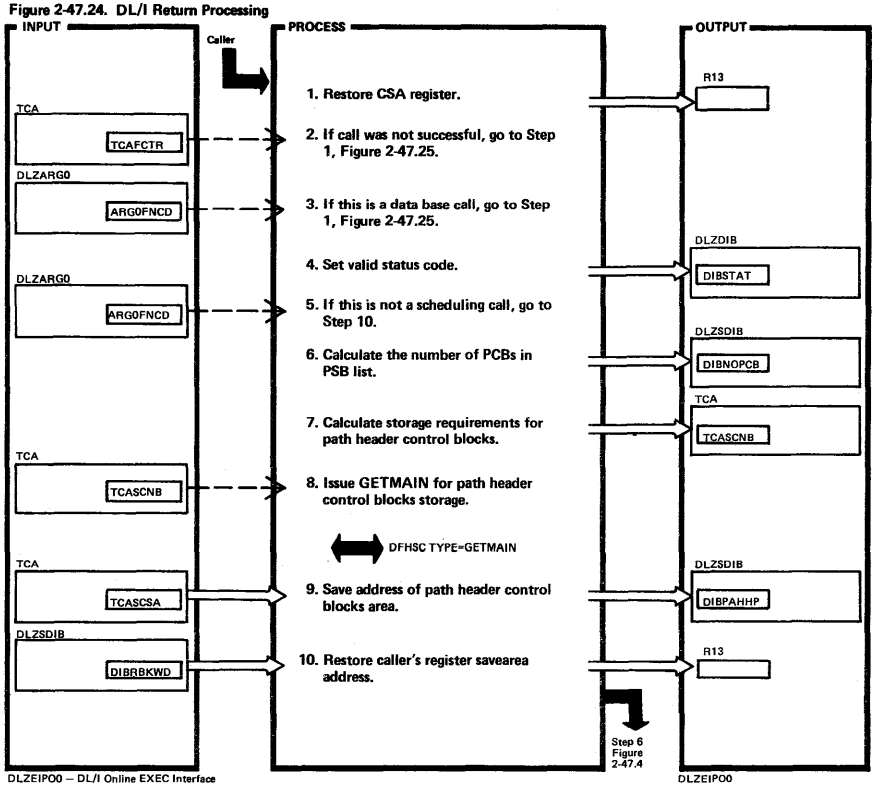
Figure 2-47.23. DL/I Program Request Handler Interface



Extended Description	Routine	Label
1. Bit DIBGPATH not on in byte DIBSFLAG.		DLIPRHEX
2. Bits DIBGPATH and PATHCALL turned off.		
3. Return address at label DLIRETRN.		DLIEXPRH
6. X'02' = non-PL/I-HLPI program X'03' = PL/I HLPI program		
7.		SETPARM
NOTE: Batch exit to DLZPRHB0 MPS exit to DLZMPRH		

Extended Description	Routine	Label

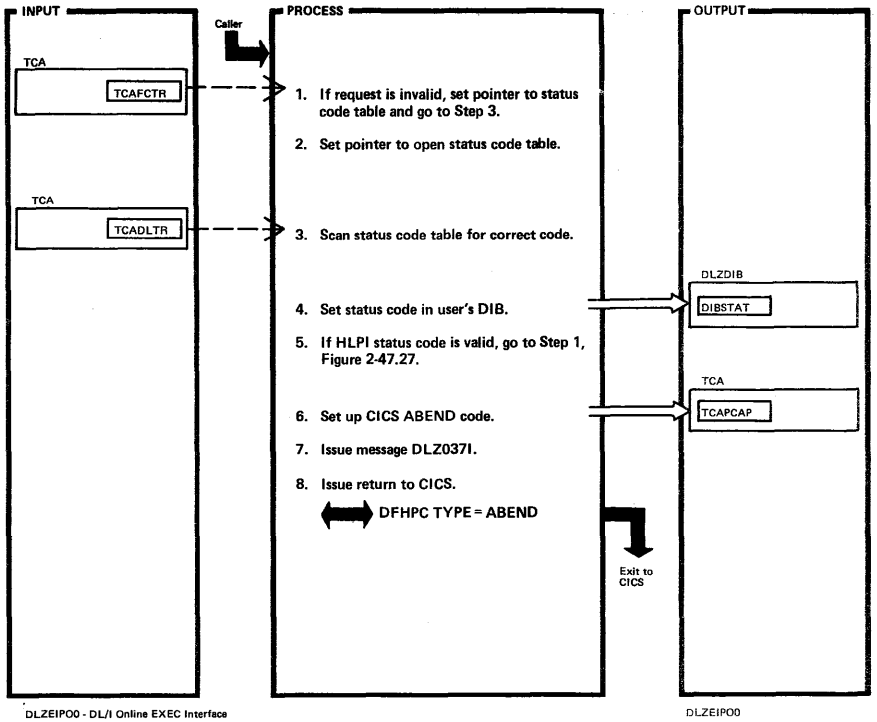
Figure 2-47.24. DL/I Return Processing



Extended Description	Routine	Label
1.		DLIRETRN
6.		PCBLOOP
7.		PCBLEND
10.		NOTSCHD

Extended Description	Routine	Label

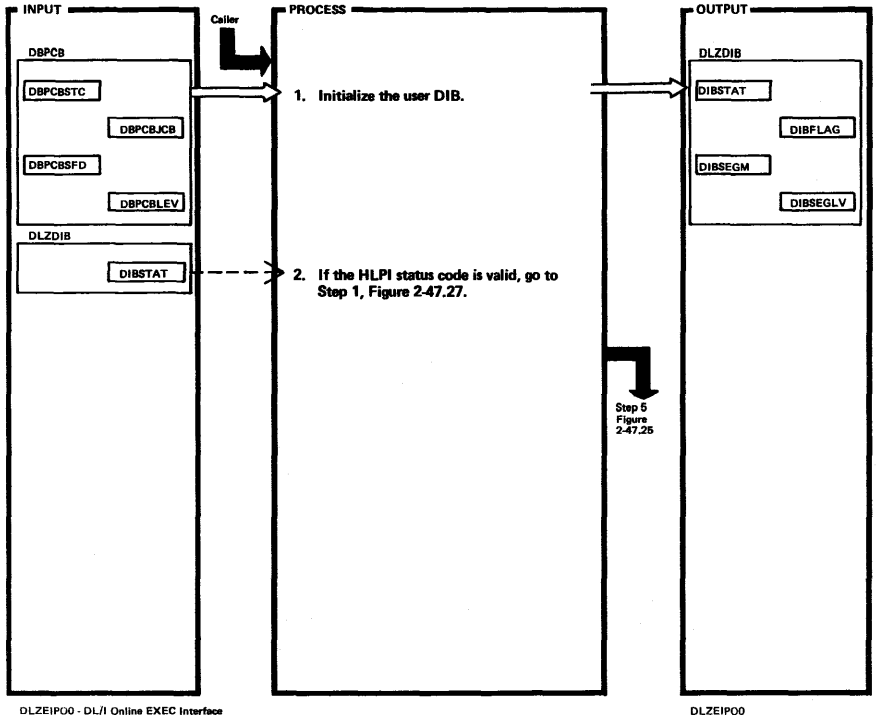
Figure 2-47.25. DL/I Psuedo ABEND Processing



Extended Description	Routine	Label
1.		CHKSTAT
2.		CHKTAB8
3.		LOCSTATC
5. Valid HLPI status codes to be returned to the user task are: '00', 'GA', 'GB', 'GK', 'II', 'NE', and 'TG'.		

Extended Description	Routine	Label

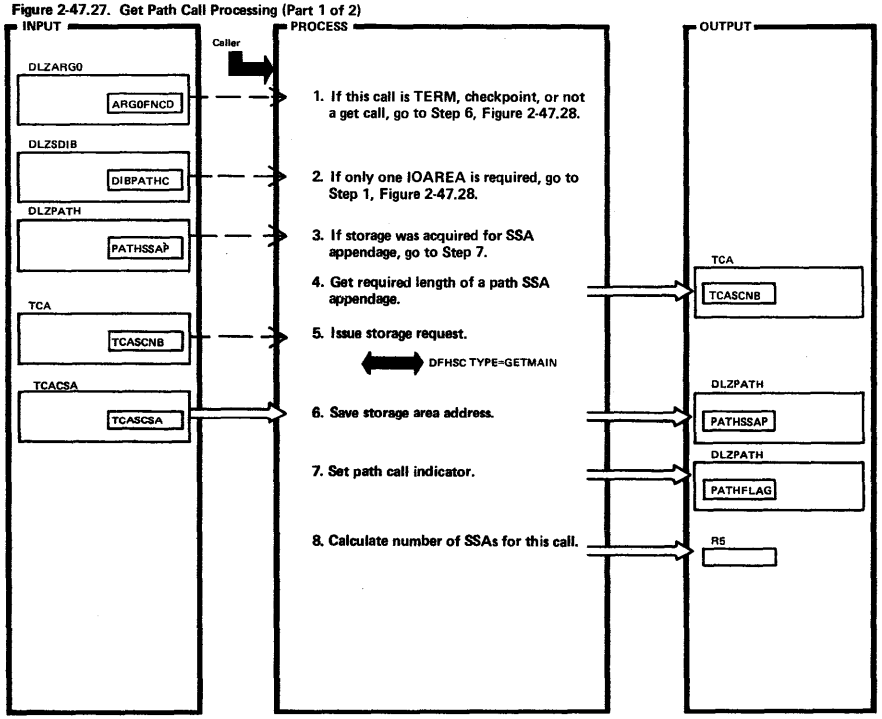
Figure 2-47.26. DIB Initialization



Extended Description	Routine	Label
1.		CKDBCALL
2. Valid HLPI status codes to be returned to the user task are: 'bb', 'GA', 'GB', 'GE', 'GK', 'II', 'NE', and 'TG'..		

Extended Description	Routine	Label

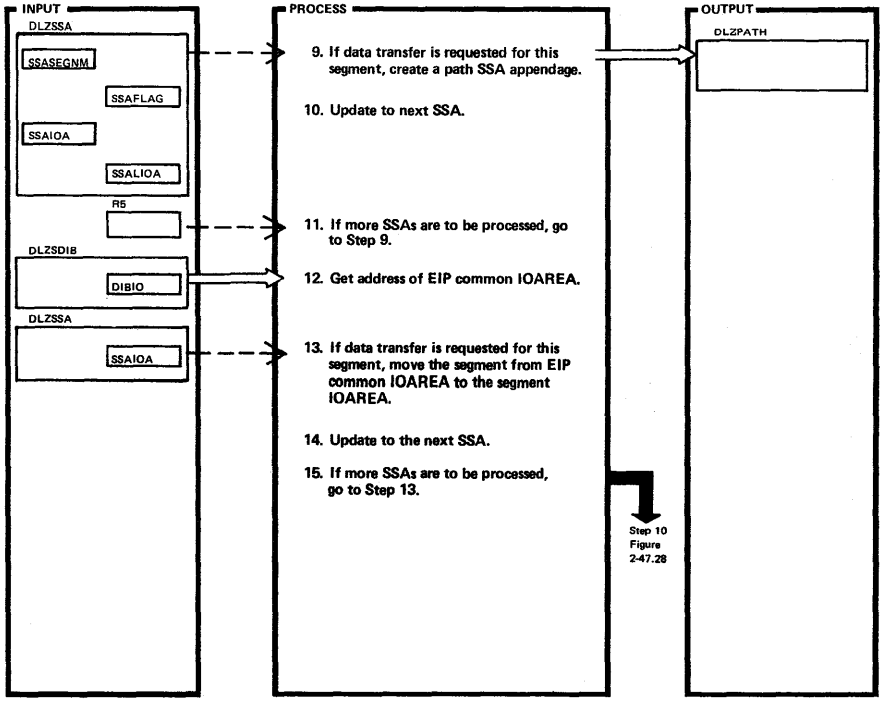
Figure 2-47.27. Get Path Call Processing (Part 1 of 2)



Extended Description	Routine	Label
1.		DBCALLOK
7.		GOTSSAP

Extended Description	Routine	Label

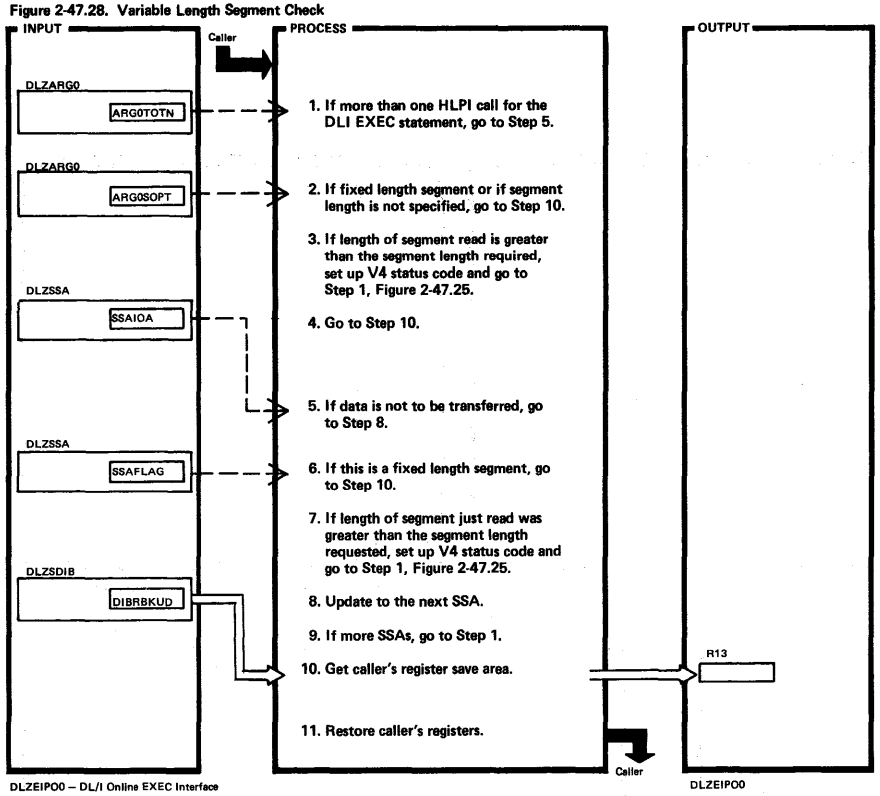
Figure 2-47.27. Get Path Call Processing (Part 2 of 2)



Extended Description	Routine	Label
9.		MOVESEG
11.		CHCKPATH
13.		GETLOOP
15.		NEXTGET

Extended Description	Routine	Label

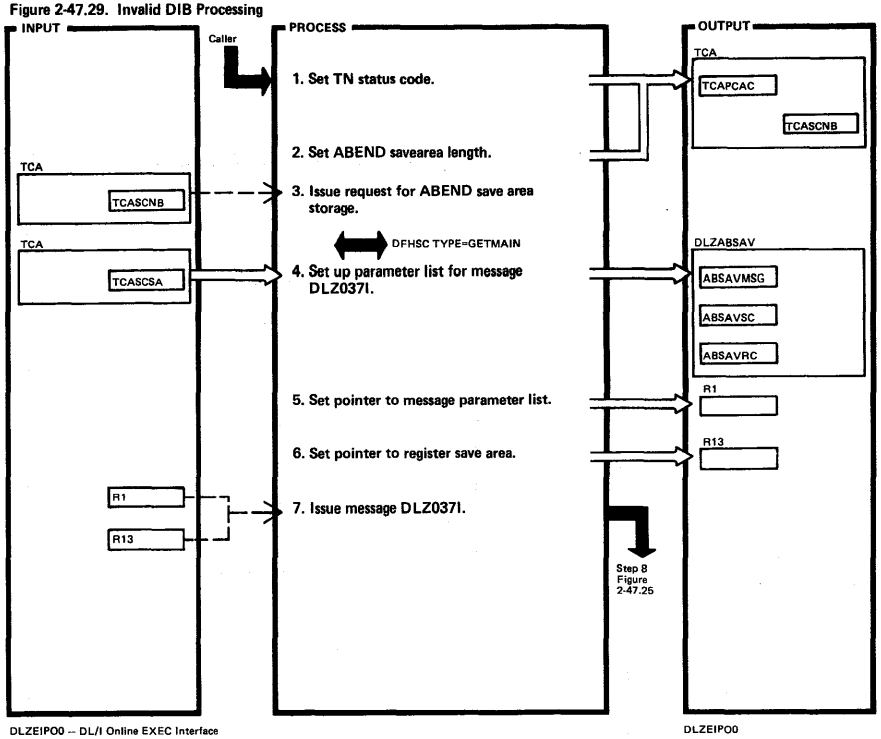
Figure 2-47.28. Variable Length Segment Check



Extended Description	Routine	Label
1.		GET1CHK
3.		NOFFSPEC
5.		IOACHK
7.		CHECKLEN
8.		UPSSA
10.		CALLDONE EIPEXIT

Extended Description	Routine	Label

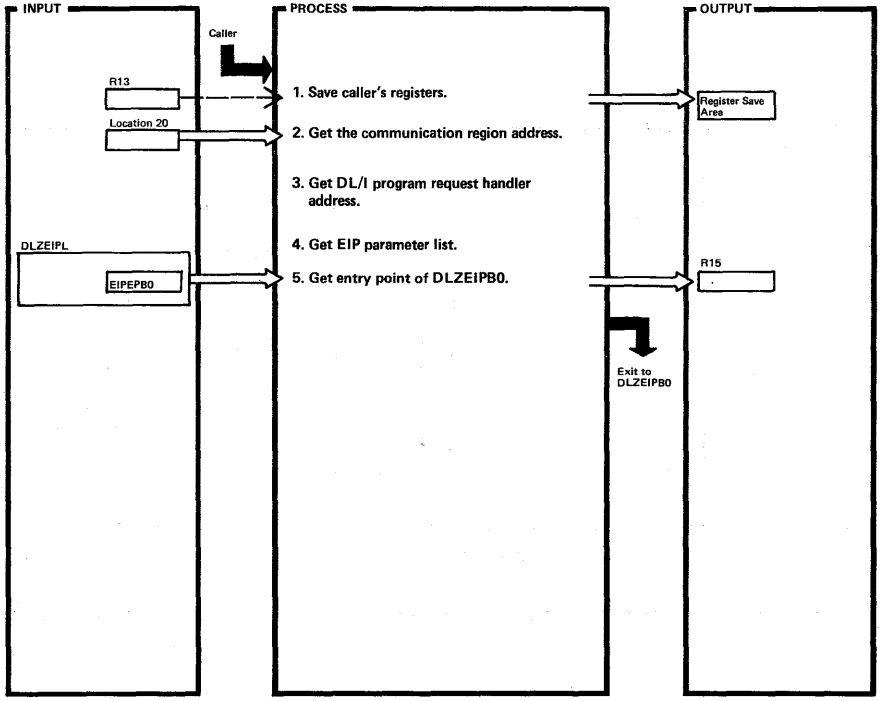
Figure 2-47.29. Invalid DIB Processing



Extended Description	Routine	Label
1.		DINABND1

Extended Description	Routine	Label

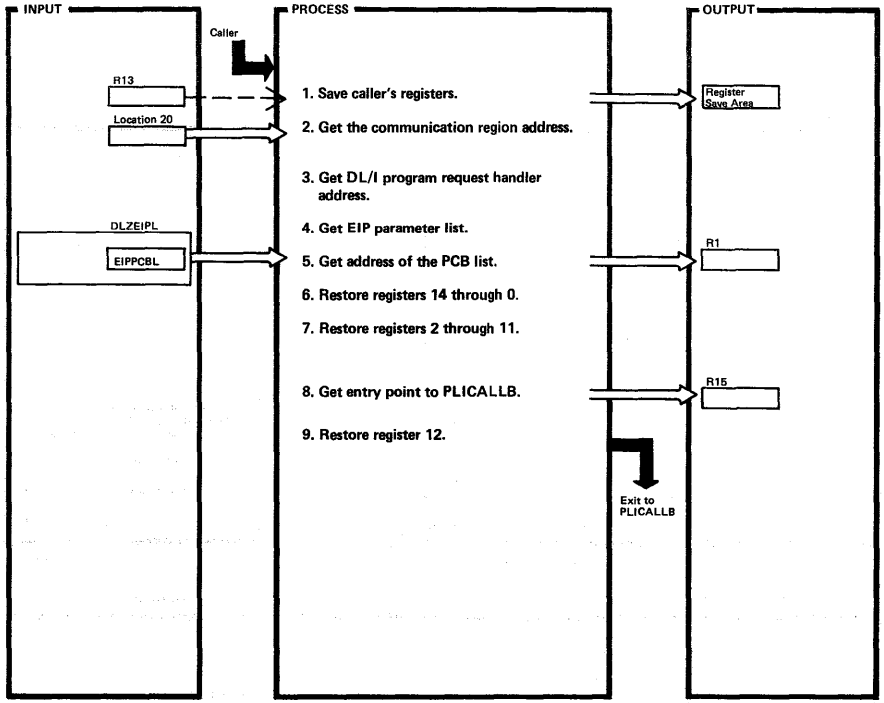
Figure 2-48. HLPI COBOL Language Interface



Extended Description	Routine	Label
1. Entry point for each HLPI call for an EXEC DLI statement. 3. Address of DL/I program request handler is 16 bytes into the COMREG. 4. Four bytes in front of the DL/I program request handler entry point is the address of the EIP parameter list.		DLZEL01 DLZEI02 DLZEI03 DLZEI04

Extended Description	Routine	Label

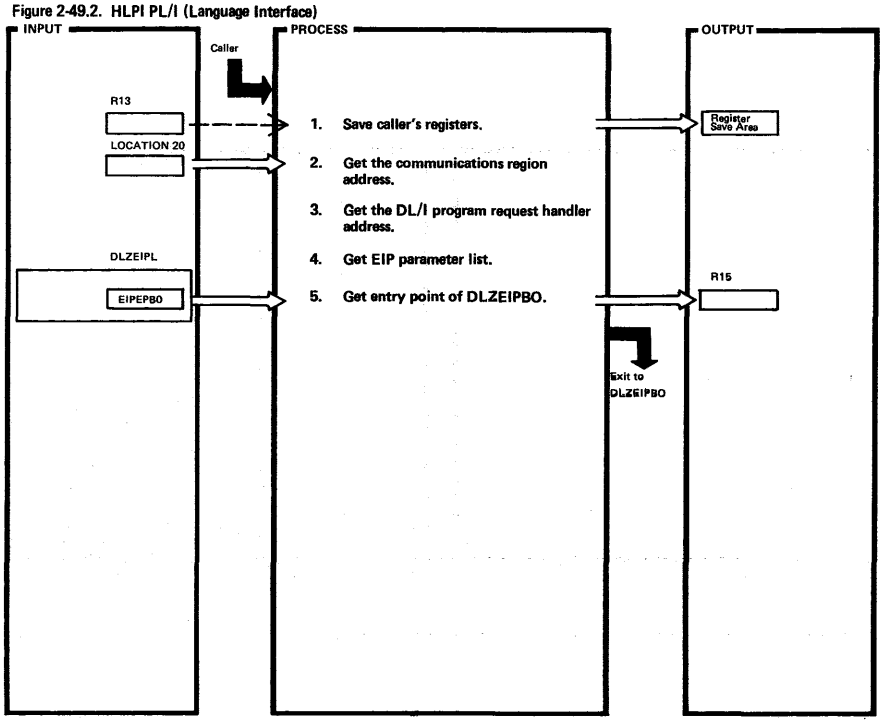
Figure 2-49.1. HLPI PL/I (PLICALLB Interface)



Extended Description	Routine	Label
1. This entry point is used only when the PL/I application program is using a non-PLI PSB. It sets up the parameter list for PL/I initialization. 3. Address of DL/I program request handler is the address of the EIP parameter list.		DLZLIPLI

Extended Description	Routine	Label

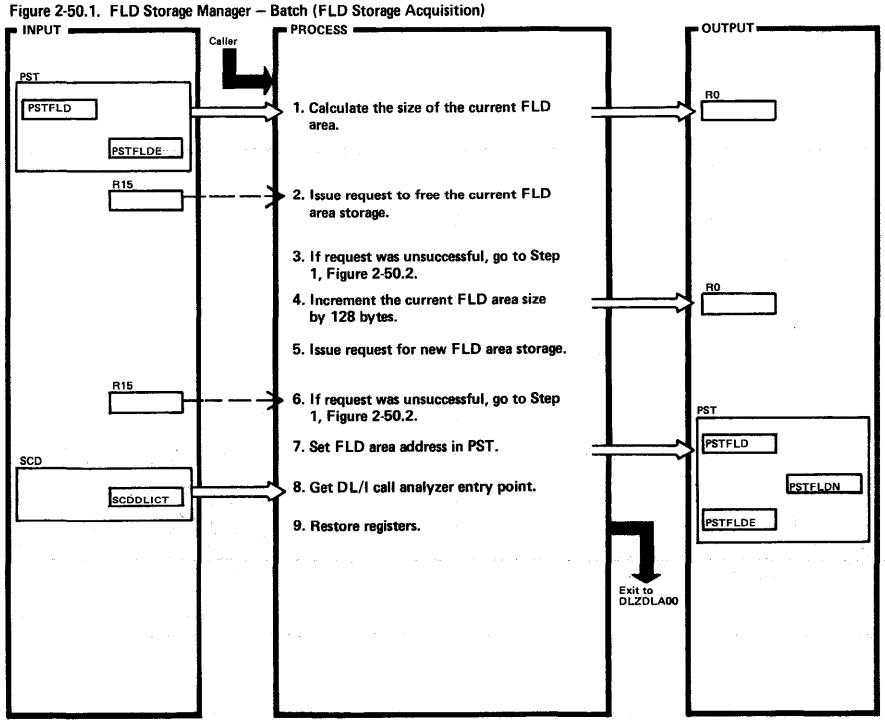
Figure 2-49.2. HLPI PL/I (Language Interface)



Extended Description	Routine	Label
1. Entry point for each HLPI call for and EXEC DLI statement. 3. Address of DL/I program request handler is 16 bytes into the COMREG. 4. Four bytes in front of the DL/I program request handler is the address of the EIP parameter list.		DLZEI01 DLZEI02 DLZEI03 DLZEI04

Extended Description	Routine	Label

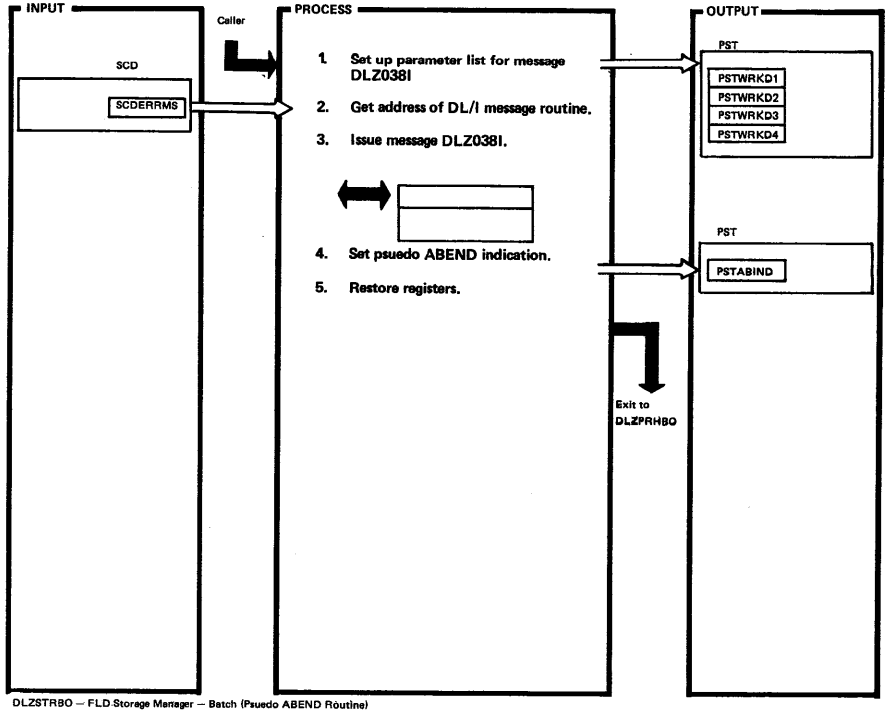
Figure 2-50.1. FLD Storage Manager — Batch (FLD Storage Acquisition)



Extended Description	Routine	Label
1. 2. DOS/VS FREEVIS issued. 5. DOS/VS GETVIS issued. 9. Registers are restored as they initially were when DL/I program request handler first called the call analyzer.		START

Extended Description	Routine	Label

Figure 2-50.2. FLD Storage Manager - Batch (Psuedo ABEND Routine)

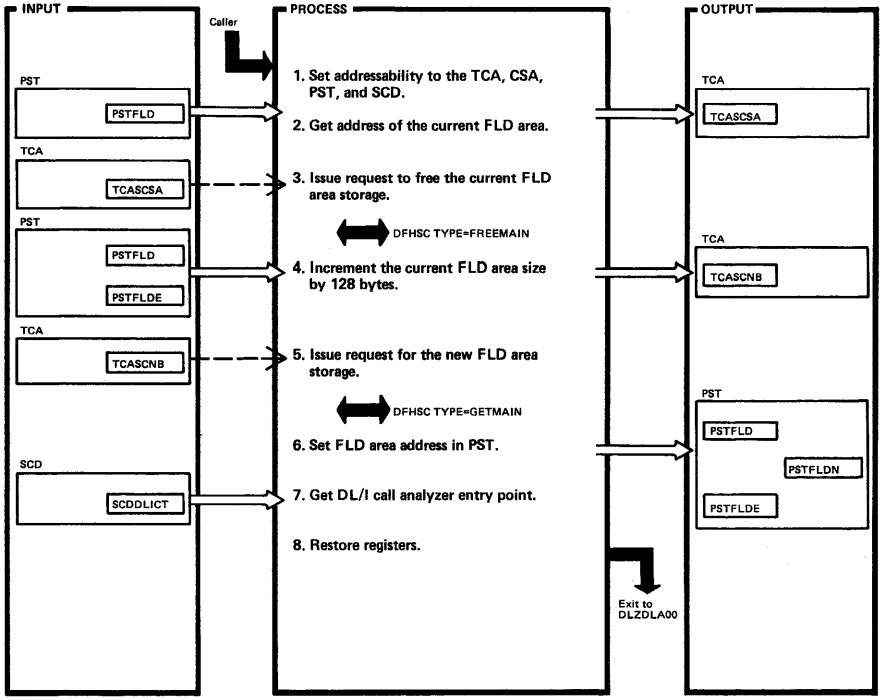


DLZSTRBO – FLD Storage Manager – Batch (Psuedo ABEND Routine)

Extended Description	Routine	Label
1.	FREABEND GETABEND	

Extended Description	Routine	Label

Figure 2-51. Online FLD Storage Manager



DLZSTR00 – Online FLD Storage Manager

DLZSTR00

Extended Description	Routine	Label
1.		DLZSTR00
2.		START
8. Registers are restored as they initially were when DL/I program request handler first called the call analyzer.		

Extended Description	Routine	Label

DL/I DOS/VS Logic Manual,
Volume 2
Order No. LY24-5215-0

READER'S COMMENT FORM

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate. Comments may be written in your own language; English is not required.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

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

• What is your occupation? _____

• How do you use this publication:

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

Your comments:

Note: Staples can cause problems with automated mail sorting equipment. Please use pressure sensitive or other gummed tape to seal this form.

If you would like a reply, please supply your name and address on the reverse side of this form.

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.)

Reader's Comment Form

Cut or Fold Along Line

Fold and Tape | Please Do Not Staple | Fold and Tape

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department G60
P. O. Box 6
Endicott, New York 13760

Fold | Fold

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

Your Name _____

Company Name _____ *Department* _____

Street Address _____

City _____

State _____ *Zip Code* _____

IBM Branch Office serving you _____

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

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

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

DL/I DOS/VS Logic Manual, Volume 2 (File No. S370/4300-50) Printed in U.S.A. LY24-5215-0

LY24-5215-0

DL/I DOS/VS Logic Manual, Volume 2 (File No. S370/4300-50) Printed in U.S.A. LY24-5215-0

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

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

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