SY26-3860-0
File No. S370-30

Systems

OS/VS2 CVOL Processor Logic

VS2.03.808

IBM

The minor revision incorporates the OS/VS2 MVS Data Management Selectable Unit **VS2.03.808.**

First Edition (August 1976)

This is a reprint of SY35-0011-1 incorporating changes released in Selectable Unit Newsletter SN26-0860.

This edition applies to Release 3.7 of OS/VS2 and to any subsequent releases of that system unless otherwise indicated in new editions or technical newsletters. (Information on the Mass Storage System is only for planning purposes until the availability of that product.)

Information in this publication is subject to significant change. Any such changes will be published in new editions or technical newsletters. Before using the publication, consult the latest *Virtual Storage Supplement (to IBM System/360 and System/370 Bibliography),* GC20-0001, and the technical newsletters that amend the bibliography, to learn which editions and technical newsletters are applicable and current.

Requests for copies of IBM publications should be made to the IBM branch off that serves you.

Forms for readers' comments are provided at the back of the publication. If the forms have been removed, comments may be addressed to IBM Corporation, P.O. Box 50020, Programming Publishing, San Jose, California 95150. All comments and suggestions become the property of IBM.

Preface

This book describes the internal logic of the CVOL Processor and provides diagnostic information. This information is directed to maintenance personnel and development programmers who require in-depth knowledge of the program's design, organization, and data areas. It is not required for effective use of the CVOL Processor.

You should be familiar with general programming techniques, OS/VS2 concepts and use, the general concepts of catalog management, and System/370 before reading this book.

The information in the following manuals, as they apply to the CVOL Processor, should be understood before you read this publication:

- *OS/VS2 MVS CVOL Processor,* GC26-3864, for an introduction to the CVOL Processor.
- *OS/VS1 Catalog Management Logic,* SY35-0003, for the internal logic of OS/VS catalog management.
- *OS/VS2 Catalog Management Logic,* SY26-3826, for information on Controller III and VSAM Catalog Management.

Other publications that this book references and that you may find helpful are:

- *OS/VS2 Planning Guide for Release 2,* GC28-0667, for information on the OS/VS2 system configuration, as well as a list of devices supported.
- *OS/VS Message Library: VS2 System Messages,* GC38-1002, for VSAM Catalog Management return codes. (See the chapter "Access Method Services Messages (IDC)" for these messages.)
- *Guide to PL/S II,* GC28-6794, for an explanation of PL/S and its listings.
- *OS/VS2 TSO Command Processor Logic Volume IV,* SY28-0652-0, for information on the VS2 Release 1 TSO LISTCAT command.
- *OS/VS-DOS/VS-VM/370 Assembler Language,* GC33-4010, for an explanation of Assembler language and its listings.
- *OS/VS-VM/370 Assembler Programmer's Guide,* GC33-4021, for an explanation of Assembler language and its listings.
- *OS/VS1 System Data Areas,* SY28-0605, which shows the content of most of the operating system control blocks and tables for OS/VS1.
- *OS/VS2 Data Areas,* SYB8-0606, which shows the content of most of the operating system control blocks and tables for OS/VS2.
- *OS/VS Data Management Services Guide,* GC26-3783, for a general introduction to Catalog Management, as well as information on generation data groups.
- *OS/VS2 Access Method Services,* GC26-3841, which describes the general syntax of the Access Method Services language, the commands of this processor, and how they are used.
- *OS/VS1 Debugging Guide,* GC24-5093, which describes how to analyze a main storage dump from OS/VS1.

- *OS/VS2 System Programming Library: Debugging Handbook, Volume 1,* GC28-0708, and *Volume 2,* GC28-0709, which describes how to analyze a main storage dump from OS/VS2.
- *OS/VS2 System Programming Library: Service Aids,* GC28-0674, which describes several service aids and programs available under the VS2 operating system.
- *OS/VS2 System Programming Library: Supervisor,* GC28-0628, for information on the ESTAE macro.
- *Data Processing Glossary,* GC20-1699, for other data processing definitions not found in the glossary of this publication.

This book is divided into six chapters and a glossary:

- "Introduction" describes the CVOL Processor and defines the terms used throughout the book.
- "Method of Operation" provides the design overview. Emphasis is on the flow of data and the concepts of the CVOL Processor, rather than on the organization of the CSECTs.
- "Program Organization" describes each CSECT of the CVOL Processor and identifies the specific function that each CSECT performs to achieve the CVOL Processor objectives. This chapter shows the logical flow from CSECT to CSECT and contains the flowcharts of the CSECTs.
- "Microfiche Directory" relates information in this book to the listings on microfiche.
- "Data Areas" describes the work areas that are used by the CVOL Processor.
- "Diagnostic Aids" shows you how to determine what CSECTs and subroutines are used for a particular request. It also shows how to dump and analyze the CVOL Catalog.
- "Glossary" lists terms and acronyms used in this publication.

In this manual, any references made to an IBM program product are not intended to state or imply that only IBM's program product may be used; any functionally equivalent program may be used instead. This manual has references to the following IBM program products:

RACF - Resource Access Control Facility, Program Number 5740-XXH

Figures

Diagrams

SUMMARY OF AMENDMENTS

Data Management (VS2.03.808)

The CVOL Processor has been enhanced to provide CVOL support that is equivalent to VS2 Release 1 and OS/MVT.

Master Catalog support is unchanged from VS2 Release 3.

The VSAM master catalog is still the only system master catalog.

Release 3

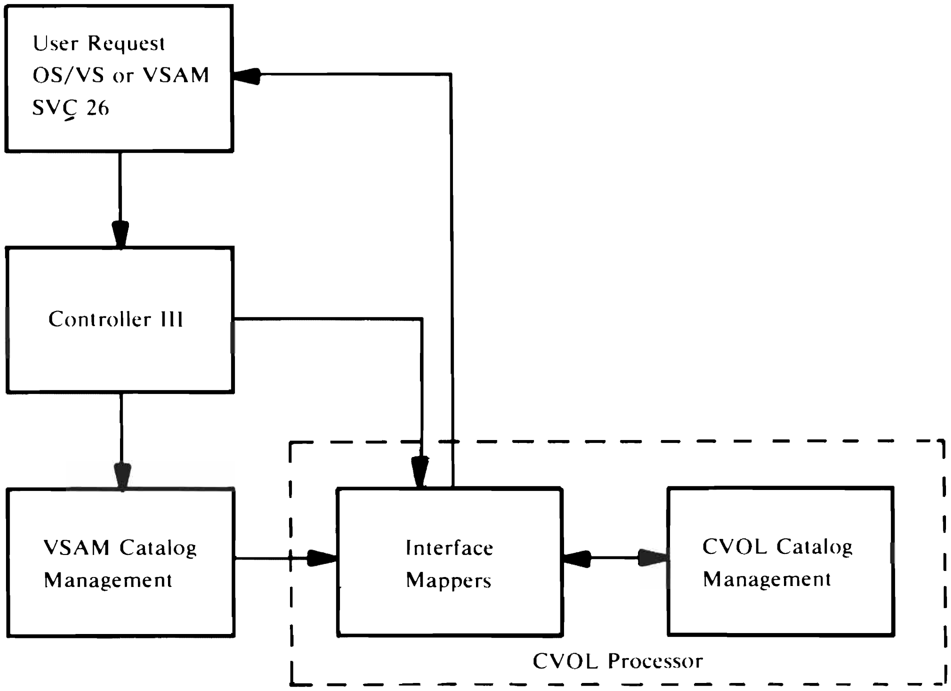
OS Catalog Management in the CVOL Processor has been repackaged in four control sections (CSECTS). A fourth CSECT, IGG0CLCF, has been added. This addition has changed several program organization figures.

Staging of data between mass storage and direct-access storage has been added for the IBM 3850 Mass Storage System (MSS). Several return codes and a program organization figure have been changed for MSS.

This book describes the program logic of the OS/VS2 CVOL Processor, hereafter called the CVOL Processor. The program is based on the OS Catalog Management function, which is included in OS, OS/VS1, or OS/VS2 Release 1. The program gets data from and puts data into CVOLs (control volumes), which can be created under OS, OS/VS1, or OS/VS2.

Overview

Figure 1-1 shows the flow of control to the CVOL Processor.



1. When an SVC 26 instruction is issued, Controller III (IGC0002F) receives control. SVC 26 passes a parameter list to Controller III. The parameter list has two possible formats, VSAM or OS, depending upon the type of request.
2. Controller III tests the parameter list. If it is an OS parameter list which specifies a CVOL volume serial, Controller III simply passes this request to the CVOL Processor. If it is an OS parameter list without a CVOL volume serial, Controller III creates a VSAM parameter list and passes the OS parameter list and the newly created VSAM parameter list to VSAM Catalog Management. If it is a VSAM parameter list, Controller III simply passes it on to VSAM Catalog Management.
3. VSAM Catalog Management searches the VSAM Catalog for the data set requested in the VSAM parameter list. If VSAM finds an alias to a SYSCTLG.*x* data set in the VSAM Master Catalog, it gives control to the CVOL Processor (IGG0CLCA) via an XCTL. (Where *x* is one or more characters that make this name unique from any other entry in the VSAM Master Catalog.) Along with control, VSAM passes the parameter list(s) from Controller III on to the CVOL Processor. For more information on VSAM Catalog Management and Controller III, refer to *OS/VS2 Catalog Management Logic.*
4. After the CVOL Processor has processed the SVC request, it gives control directly to the program that issued the SVC 26 instruction.

Figure 1-1. Flow of Control to the CVOL Processor

Requirements of the CVOL Processor

Purpose and Function

The CVOL Processor's objective is to provide support for CVOLs within the single (VSAM) master catalog environment of MVS. The CVOL Processor permits the use of existing CVOLs in a multiple CPU environment when running OS, OS/VS1, or any release of OS/VS2 without converting back and forth between the types of catalog structures supported by each operating system.

If a request is made for a catalog VSAM function against a CVOL Catalog, the CVOL Processor maps the request into an OS request and performs the catalog function. For more information on how the CVOL Processor operates, see Chapter 2 of this publication, "Method of Operation." For a list of requests and what the CVOL Processor maps them into, as well as a list of requests that the CVOL Processor does not accept, read the "Introduction" of the IBM publication *OS/VS2 MVS CVOL Processor.*

Physical Characteristics

The CVOL Processor occupies 20,000 bytes of storage and consists of one load module named IGGOCLCA. It resides in SYS1.LPALIB and can be paged into real storage. The IGGOCLCA load module contains six CSECTs: IGGOCLCA, IGGOCLCB, IGGOCLCC, IGGOCLCD, IGGOCLCE, and IGGOCLCF.

The program organization of the CVOL Processor can be thought of as two sections: the Interface Mappers and CVOL Catalog Management. The Interface Mappers consist of CSECTs IGGOCLCA and IGGOCLCB. CVOL Catalog Mangement consists of CSECTs IGGOCLCC, IGGOCLCD, IGGOCLCE, and IGGOCLCF (repackaged OS/VS2 Release 1 Catalog Management). For more information on the subroutines and their use within each CSECT, see Chapter 4, "Microfiche Directory," of this publication.

When the CVOL Processor gains control, register 12 points to the work area, WORKCLCA, that is passed by VSAM Catalog Management. Controller III created WORKCLCA and passed it to VSAM Catalog Management. See Figure 3-1 and Chapter 5, "Data Areas," for a description of WORKCLCA.

If the request is successful, the data is returned as expected by the original OS or VSAM request. Register 15 contains zero. If the request is not successful, the CVOL Processor passes a return code in register 15 to the issuer of the SVC 26. For a list of return codes and their meanings, refer to the heading "Processor Exit and Output" in Chapter 3 of this publication. For a list of control information required and any restrictions on the use of the CVOL Processor, refer to the IBM publication *OS/VS2 MVS CVOL Processor.*

For examples of which subroutine within the CVOL Processor is involved in any given situation, see Figures 6-1 and 6-2 in this publication. For more information on diagnostic aids for the CVOL Processor, see Chapter 6, "Diagnostic Aids," of this publication.

Note: Because all CVOL Catalogs are named 'SYSCTLG,' the terms CVOL Catalog and SYSCTLG are used interchangeably in this documentation.

Method of Operation

This chapter contains method of operation diagrams of the main elements of the CVOL Processor. A table is included on each diagram which lists each step of the diagram, the CSECTs name, and the subroutines used. Using these names, you can go either to the chapter "Program Organization" or to the chapter "Microfiche Directory" (or the microfiche itself) for more information.

The following legend explains the symbols used throughout this chapter:

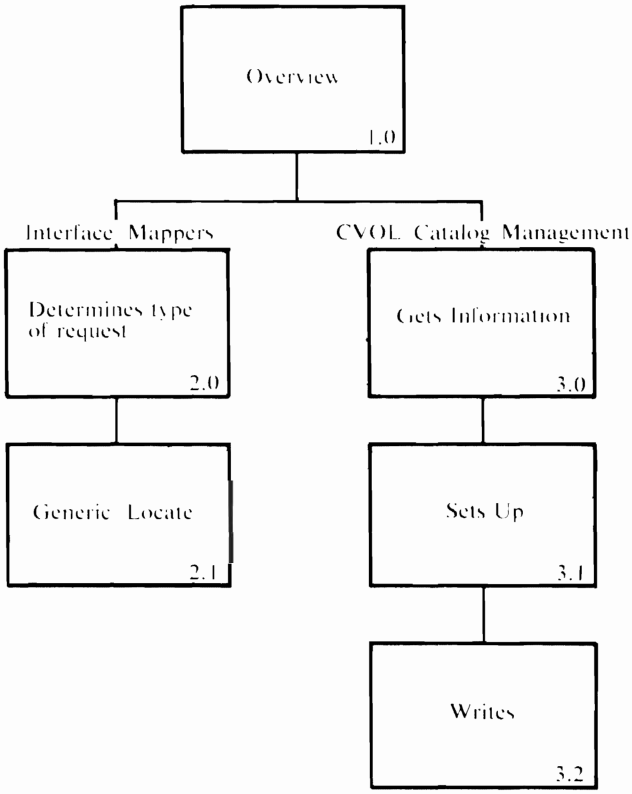
⇨	Data flow
➡	Flow of control, entry and exit points
▨⇨	Data flow when existing data has been changed
Ⓐ ↗ Ⓐ	On-page connector
⊐	Off-page connector
→	Pointer to more information

Figure 2-1 lists the abbreviations used in the diagrams.

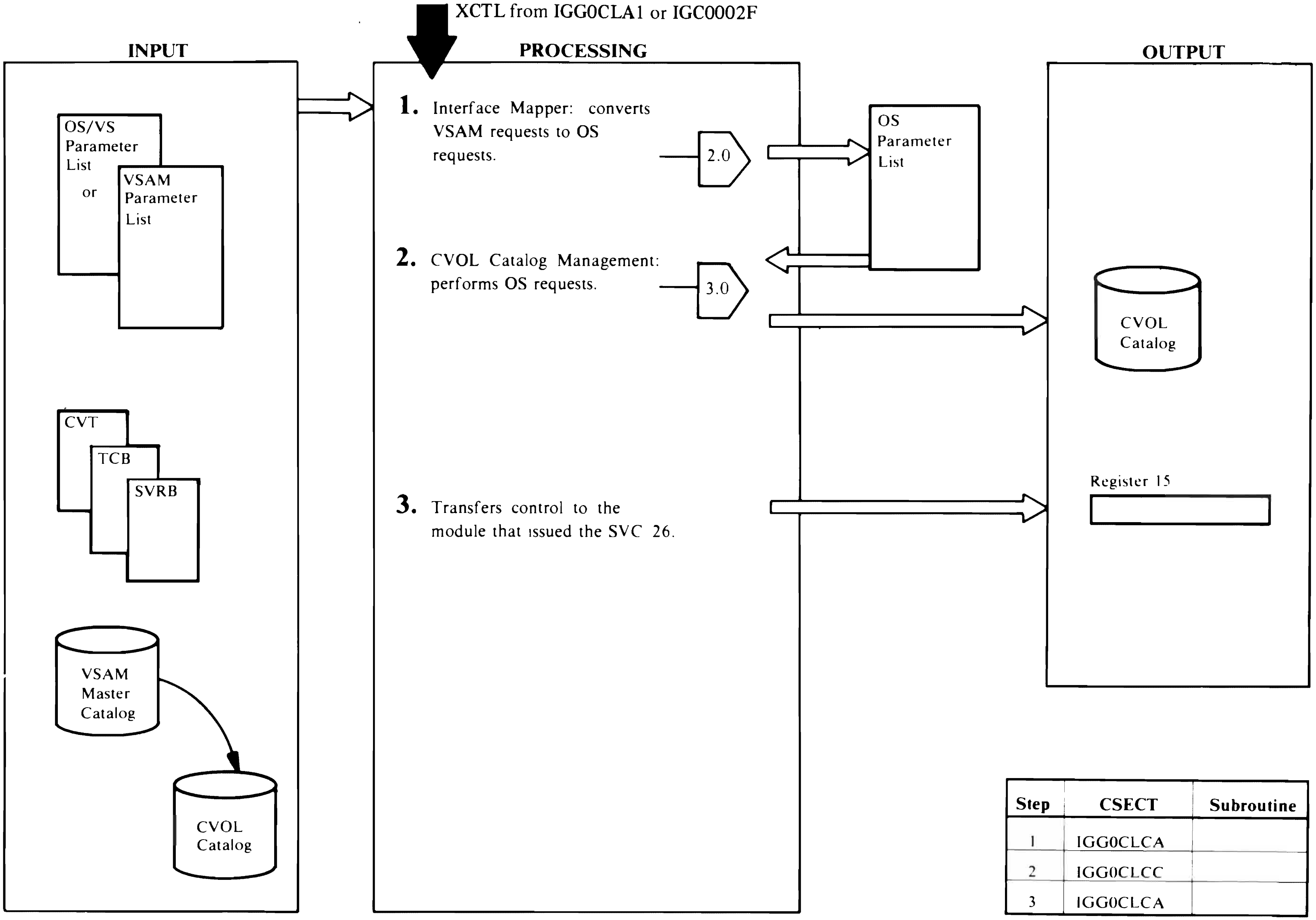
Abbreviation	Name	Description
CVT	Communication vector table	An operating system control block that provides the address of information in the nucleus of the non-resident routines.
DSPE	Data set pointer entry	Contains the simple name of a data set and provides the location of this data set.
GIPE	Generation index pointer entry	Points to the lowest index for a generation data group.
ILE	Index link entry	Links this block to the next block in a chain of blocks for one index.
IPE	Index pointer entry	Points to a lower-level index of this name.
SVRB	Supervisor request block	An operating system control block containing program status information and general register contents.
TCB	Task control block	An operating system control block that contains information and pointers associated with the task in progress.

Figure 2-1. Abbreviations Used in the Diagrams

CVOL Processor



CVOL Processor Visual Table of Contents



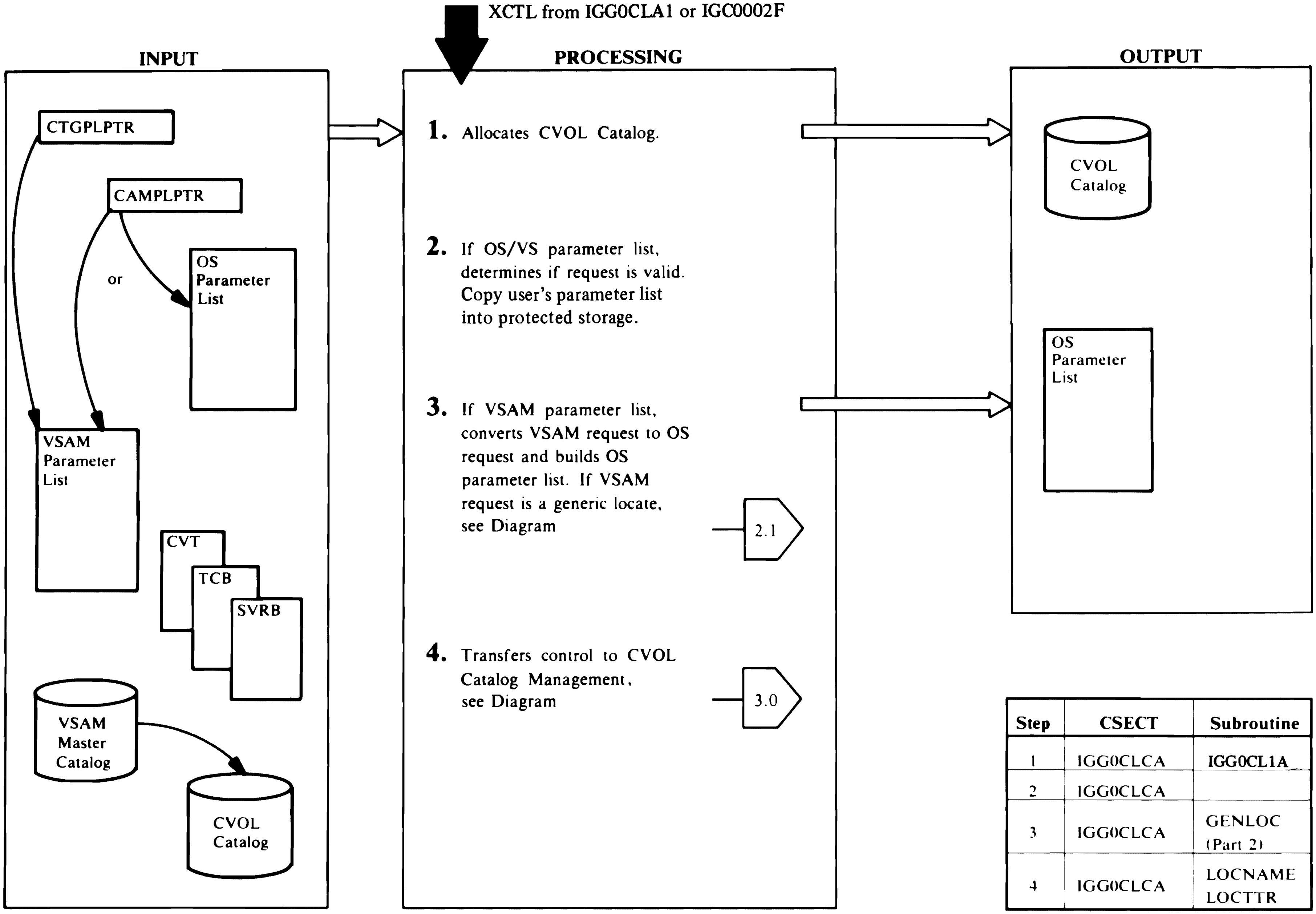
1. Interface Mapper: converts VSAM requests to OS requests.

2. CVOL Catalog Management: performs OS requests.

3. Transfers control to the module that issued the SVC 26.

Step	CSECT	Subroutine
1	IGG0CLCA	
2	IGG0CLCC	
3	IGG0CLCA	

Diagram 1.0 CVOL Processor (Overview)



1. Allocates CVOL Catalog.

2. If OS/VS parameter list, determines if request is valid. Copy user's parameter list into protected storage.

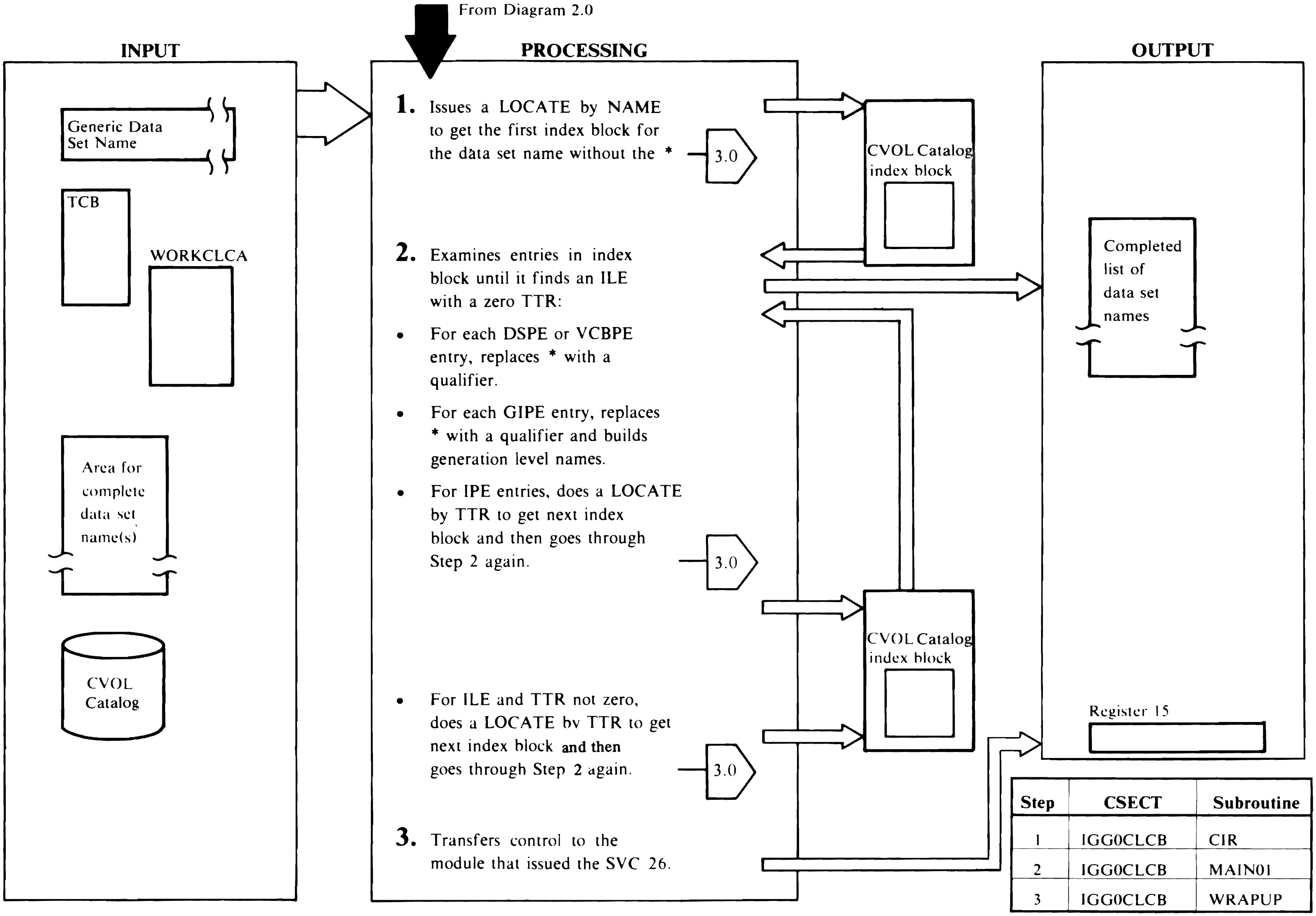
3. If VSAM parameter list, converts VSAM request to OS request and builds OS parameter list. If VSAM request is a generic locate, see Diagram 2.1

4. Transfers control to CVOL Catalog Management, see Diagram 3.0

Step	CSECT	Subroutine
1	IGG0CLCA	IGG0CL1A
2	IGG0CLCA	
3	IGG0CLCA	GENLOC (Part 2)
4	IGG0CLCA	LOCNAME LOCTTR

VS2.03.808

Diagram 2.0 Interface Mappers (Determines Type of Request)



From Diagram 2.0

INPUT

Generic Data Set Name

TCB

WORKCLCA

Area for complete data set name(s)

CVOL Catalog

PROCESSING

1. Issues a LOCATE by NAME to get the first index block for the data set name without the * → 3.0

2. Examines entries in index block until it finds an ILE with a zero TTR:

- For each DSPE or VCBPE entry, replaces * with a qualifier.
- For each GIPE entry, replaces * with a qualifier and builds generation level names.
- For IPE entries, does a LOCATE by TTR to get next index block and then goes through Step 2 again. → 3.0
- For ILE and TTR not zero, does a LOCATE bv TTR to get next index block and then goes through Step 2 again. → 3.0

3. Transfers control to the module that issued the SVC 26.

CVOL Catalog index block

CVOL Catalog index block

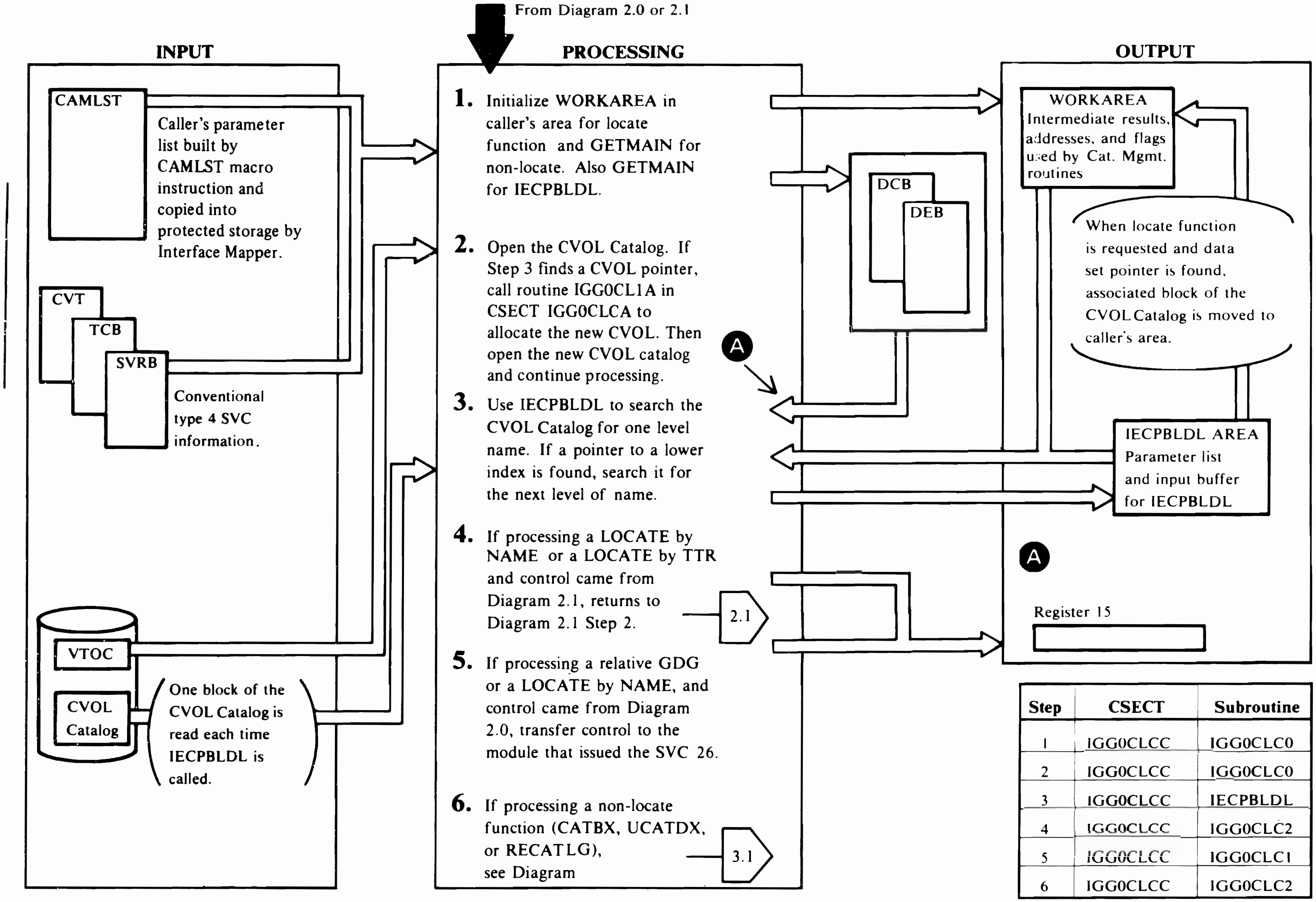
OUTPUT

Completed list of data set names

Register 15

Step	CSECT	Subroutine
1	IGG0CLCB	CIR
2	IGG0CLCB	MAIN01
3	IGG0CLCB	WRAPUP

Diagram 2.1 Interface Mappers (Generic Locate)


From Diagram 2.0 or 2.1
INPUT
PROCESSING
OUTPUT
CAMLST
Caller's parameter list built by CAMLST macro instruction and copied into protected storage by Interface Mapper.
CVT
TCB
SVRB
Conventional type 4 SVC information.
VTOC
CVOL Catalog
One block of the CVOL Catalog is read each time IECPBLDL is called.
WORKAREA
Intermediate results, addresses, and flags used by Cat. Mgmt. routines
DCB
DEB
When locate function is requested and data set pointer is found, associated block of the CVOL Catalog is moved to caller's area.
A
IECPBLDL AREA
Parameter list and input buffer for IECPBLDL
A
Register 15
2.1
3.1

1. Initialize WORKAREA in caller's area for locate function and GETMAIN for non-locate. Also GETMAIN for IECPBLDL.

2. Open the CVOL Catalog. If Step 3 finds a CVOL pointer, call routine IGG0CL1A in CSECT IGG0CLCA to allocate the new CVOL. Then open the new CVOL catalog and continue processing.

3. Use IECPBLDL to search the CVOL Catalog for one level name. If a pointer to a lower index is found, search it for the next level of name.

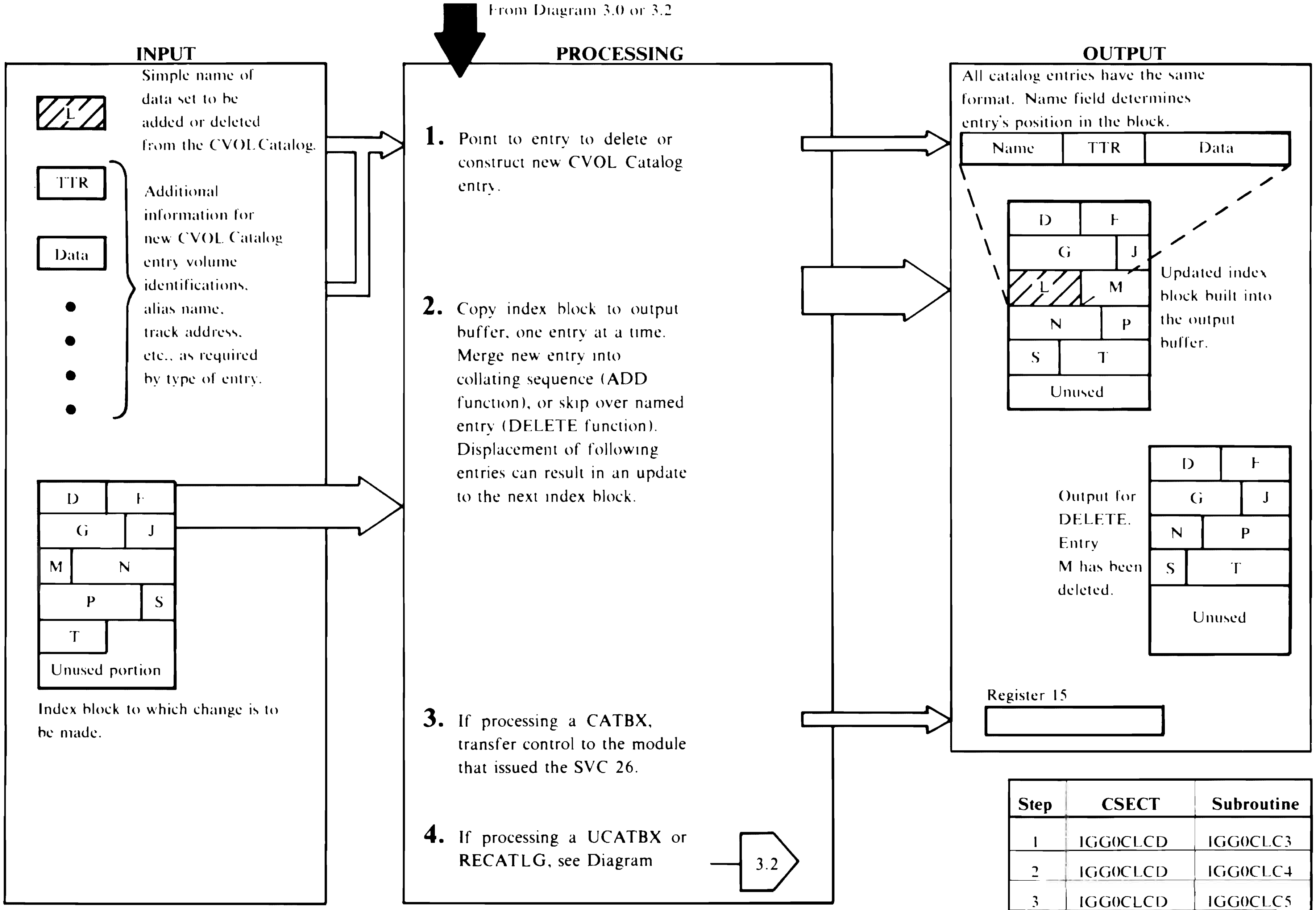
4. If processing a LOCATE by NAME or a LOCATE by TTR and control came from Diagram 2.1, returns to Diagram 2.1 Step 2.

5. If processing a relative GDG or a LOCATE by NAME, and control came from Diagram 2.0, transfer control to the module that issued the SVC 26.

6. If processing a non-locate function (CATBX, UCATDX, or RECATLG), see Diagram

Step	CSECT	Subroutine
1	IGG0CLCC	IGG0CLC0
2	IGG0CLCC	IGG0CLC0
3	IGG0CLCC	IECPBLDL
4	IGG0CLCC	IGG0CLC2
5	IGG0CLCC	IGG0CLC1
6	IGG0CLCC	IGG0CLC2

Diagram 3.0 CVOL Catalog Management (Gets Information)



Simple name of data set to be added or deleted from the CVOL Catalog.

Additional information for new CVOL Catalog entry volume identifications, alias name, track address, etc., as required by type of entry.

Index block to which change is to be made.

1. Point to entry to delete or construct new CVOL Catalog entry.

2. Copy index block to output buffer, one entry at a time. Merge new entry into collating sequence (ADD function), or skip over named entry (DELETE function). Displacement of following entries can result in an update to the next index block.

3. If processing a CATBX, transfer control to the module that issued the SVC 26.

4. If processing a UCATBX or RECATLG, see Diagram

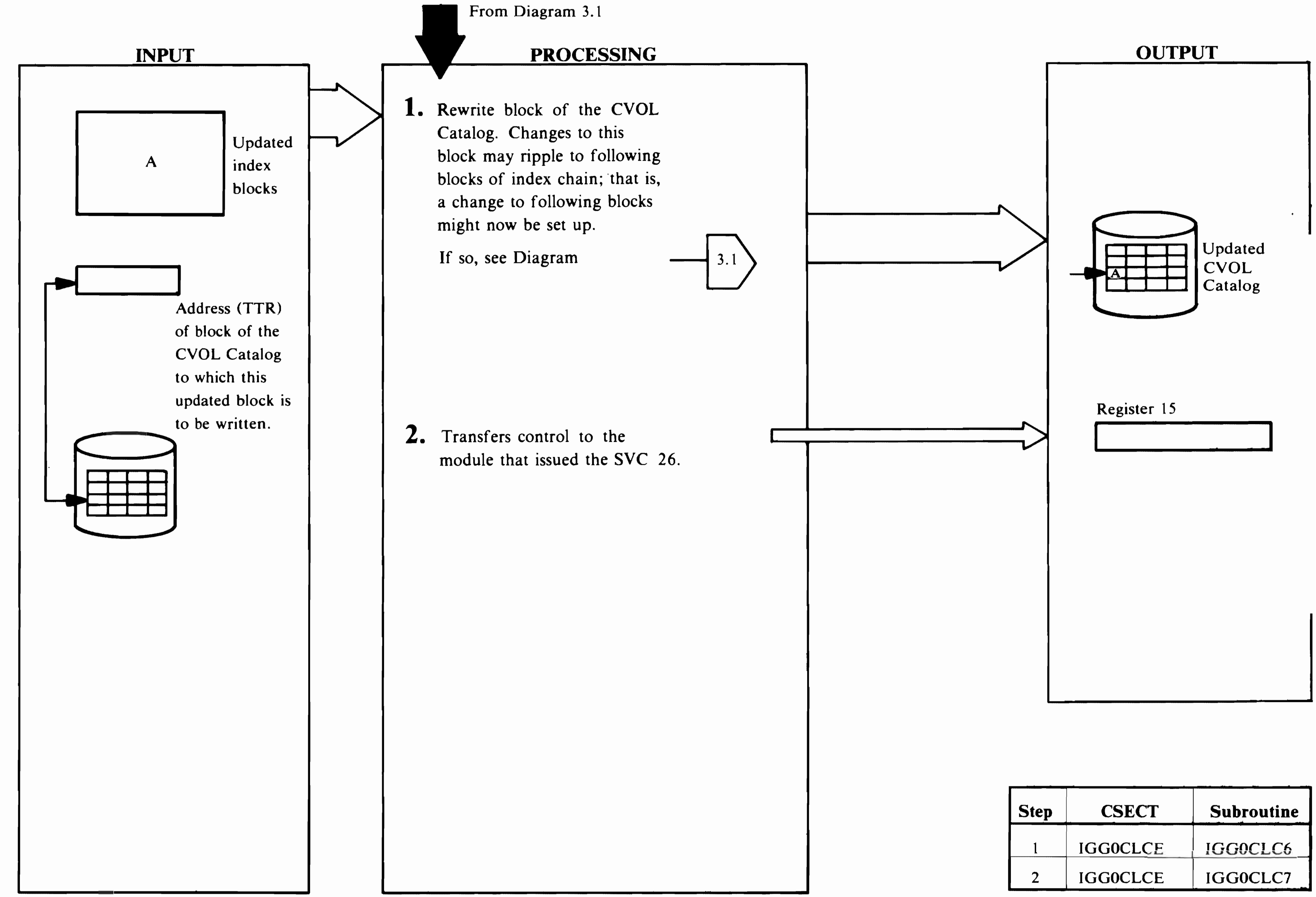
All catalog entries have the same format. Name field determines entry's position in the block.

Updated index block built into the output buffer.

Output for DELETE. Entry M has been deleted.

Step	CSECT	Subroutine
1	IGG0CLCD	IGG0CLC3
2	IGG0CLCD	IGG0CLC4
3	IGG0CLCD	IGG0CLC5
4	IGG0CLCD	IGG0CLC5

Diagram 3.1 CVOL Catalog Management (Sets Up)



1. Rewrite block of the CVOL Catalog. Changes to this block may ripple to following blocks of index chain; that is, a change to following blocks might now be set up.

   If so, see Diagram 3.1

2. Transfers control to the module that issued the SVC 26.

Step	CSECT	Subroutine
1	IGG0CLCE	IGG0CLC6
2	IGG0CLCE	IGG0CLC7

Diagram 3.2 CVOL Catalog Management (Writes)

Program Organization

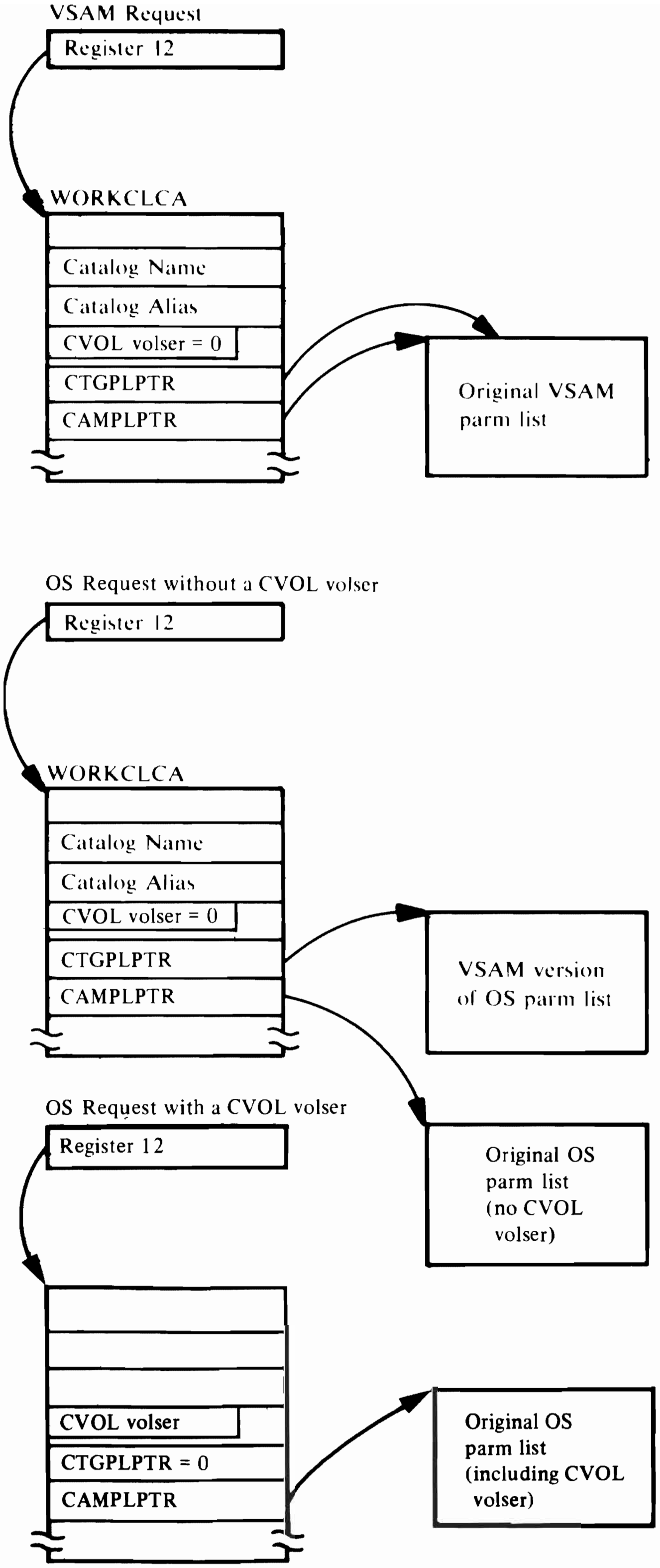
The CVOL Processor consists of one load module named IGGOCLCA that resides in SYS1.LPALIB. IGGOCLCA contains six CSECTs: IGGOCLCA, IGGOCLCB, IGGOCLCC, IGGOCLCD, IGGOCLCE, and IGGOCLCF. The program organization of the CVOL Processor can be thought of as two sections: the Interface Mappers and CVOL Catalog Management. The Interface Mappers consist of CSECTs IGGOCLCA and IGGOCLCB. CVOL Catalog Management consists of CSECTs IGGOCLCC, IGGOCLCD, IGGOCLCE, and IGGOCLCF (repackaged OS/VS2 Release 1 Catalog Management functions). See Figure 1-1 in this publication for an overview of the flow of control to the CVOL Processor.

Processor Invocation and Input

The CVOL Processor, module IGGOCLCA, gains control via an XCTL from Controller III, module IGC0002F, when an OS/VS style catalog request is issued which specifies a CVOL volume serial in the parameter list.

The CVOL Processor, module IGGOCLCA, also gains control via an XCTL from VSAM Catalog Management, module IGGOCLA1, when VSAM finds an alias to a SYSCTLG.*x* data set in the VSAM Master Catalog. (Where *x* is one or more characters that make this name unique from any other entry in the VSAM Master Catalog.) This alias entry indicates that the data set requested by SVC 26 resides on a CVOL Catalog.

Standard linkage is not used with the CVOL Processor. Register 1 points to a parameter list that is not needed by the CVOL Processor. Register 12 points to the work area named WORKCLCA that was created by Controller III. When the CVOL Processor gets control, it ignores the contents of register 13. The CVOL Processor puts the address of its own save area in register 13 and saves registers in that save area. Register 15 contains the entry point address of IGGOCLCA. Register 14 is not used. Figure 3-1 illustrates the key fields within WORKCLCA that the CVOL Processor depends upon.



If the parameter list passed to SVC 26 indicates a VSAM request, CTGPLPTR and CAMPLPTR point to the VSAM parameter list.

If the parameter list passed to SVC 26 indicates an OS request, CAMPLPTR points to the OS parameter list.

If the OS parameter list specifies a CVOL volser, then CTGPLPTR is zero, the CVOL volume serial field has been filled in by Controller III, and the catalog name and alias fields remain uninitialized. If the OS parameter list specifies no CVOL volser, then CTGPLPTR points to the VSAM parameter list created by Controller III, the CVOL volume serial field is set to binary zeros, and the catalog name and alias fields have been filled in by VSAM Catalog Management.

Figure 3-1. WORKCLCA at Processor Invocation

Processor Exit and Outout

CVOL Processor gives control to the issuer of the SVC 26. If no errors were encountered, register 15 contains zero. If an error has occurred, register 15 contains a return code indicating the type of error. When the contents are significant, the meaning is noted below. In some cases registers 0 and 1 provide further information concerning the error. The meaning of the return code varies according to the type of catalog request. Refer to the following lists for return code meanings.

If the request is a VSAM request, register 15 contains a return code defined by VSAM Catalog Management. These return codes are explained in *OS/VS Message Library: VS2 System Messages,* in the chapter called "Access Method Services Messages (IDC)." If the request is an OS request, register 15 contains one of the return codes described in the following lists. Register 0 contains the VSAM Catalog Management return code if the OS request was satisfied in a VSAM Catalog and if register 15 does not contain a 0. Refer to the following lists for return code meanings.

OS LOCATE Macro Return Codes

If processing an OS locate request, register 15 may contain:

- 0—Successful.
- 4—Either the required CVOL Catalog does not exist, could not be allocated, or an MSS (Mass Storage System) acquire failed.
- 8—One of the following:
  1. Entry not found. R0 contains number of index levels.
  2. Protection violation. R0=56.
  3. GDG alias found. R0 contains number of index levels.
- 12—Non-data set found at last qualifier. R0 contains number of index levels.
- 16—Data set exists at an earlier level of qualification. R0 contains number of index levels where data set was encountered.
- 20—Syntax error in data set name.
- 24—One of the following:
  1. Permanent I/O error. R0=VSAM return code or 0 if error in CVOL.
  2. Unrecoverable error (including 'Do not allocate'). R0=0
  3. Non-zero ESTAE return code. R0=0.
  4. Error in CAMLST. R0=0.
- 28—TTR is out of range.

OS INDEX Macro Return Codes

When processing an OS BLDX, DLTX, LNKX, BLDG, BLDA, DLTA, or DRPX request, register 15 may contain:

- 0—Successful.
- 4—CVOL not available.

- 8—Catalog structure inconsistent with specified operation. R0 same as R0 on a LOCATE on this name. R1 same as R15 on a LOCATE on this name.
- 12—Can't delete a non-empty index.
- 16—Necessary index structure does not exist.
- 20—Space unavailable in catalog.
- 28—One of the following:
  1. Permanent I/O error.
  2. Non-zero ESTAE return code.

OS CATALOG, UNCATALOG, or RECATALOG Return Codes

When processing an OS CATALOG, UNCATALOG, or RECATLOG request, register 15 may contain:

- 0–Successful.
- 4–Either the required CVOL Catalog does not exist, or the CVOL Catalog cannot be allocated or acquired.
- 8–One of the following:

  1. Catalog structure inconsistent with the operation requested (including alias for GDG found). R0 same as R0 on a LOCATE on this name. R1 same as R15 on a LOCATE on this name.
  2. Protection violation. R0=56. R1=0.

- 20–Insufficient space on a CVOL Catalog data set. Register 0 contains zero.
- 24–Improperly named generation data group not cataloged.
- 28–One of the following:

  1. A permanent I/O error or an unrecoverable error occurred.
  2. An error was found in the OS parameter list.
  3. An I/O error occurred in a CVOL Catalog.
  4. An ESTAE return code was non-zero.

VSAM SUPERLOCATE Return Codes When Accessing CVOL Catalogs

When processing a VSAM SUPERLOCATE request, register 15 may contain:

- 0–Successful.
- 4–Allocation error occurred or unable to open a CVOL Catalog.
- 8–Data set not found or the structure of the CVOL Catalog was inconsistent.
- 24–I/O error or unrecoverable error.
- 44–Insufficient space available to CVOL Processor.
- 68–The CVOL Catalog cannot be allocated.
- 164–ESTAE return code was non-zero.

Other VSAM Request Return Codes When Accessing CVOL Catalogs

When processing VSAM requests other than SUPERLOCATE, register 15 may contain:

- 0–Successful.
- 4–Allocation error or unable to open a CVOL Catalog.
- 8–Data set not found or the structure of the CVOL Catalog was inconsistent.
- 24–I/O error or unrecoverable error trying to locate information.
- 28–I/O error or unrecoverable error on any request action except trying to locate information.
- 40–Insufficient space.
- 48–Invalid function, not consistent with a CVOL Catalog.
- 164–ESTAE return code was non-zero.

Overall Program Organization of the CVOL Processor.

Figure 3-2 gives the overall program organization of the CVOL Processor. The figure is followed by a description of each of the CSECTs that the CVOL Processor contains.

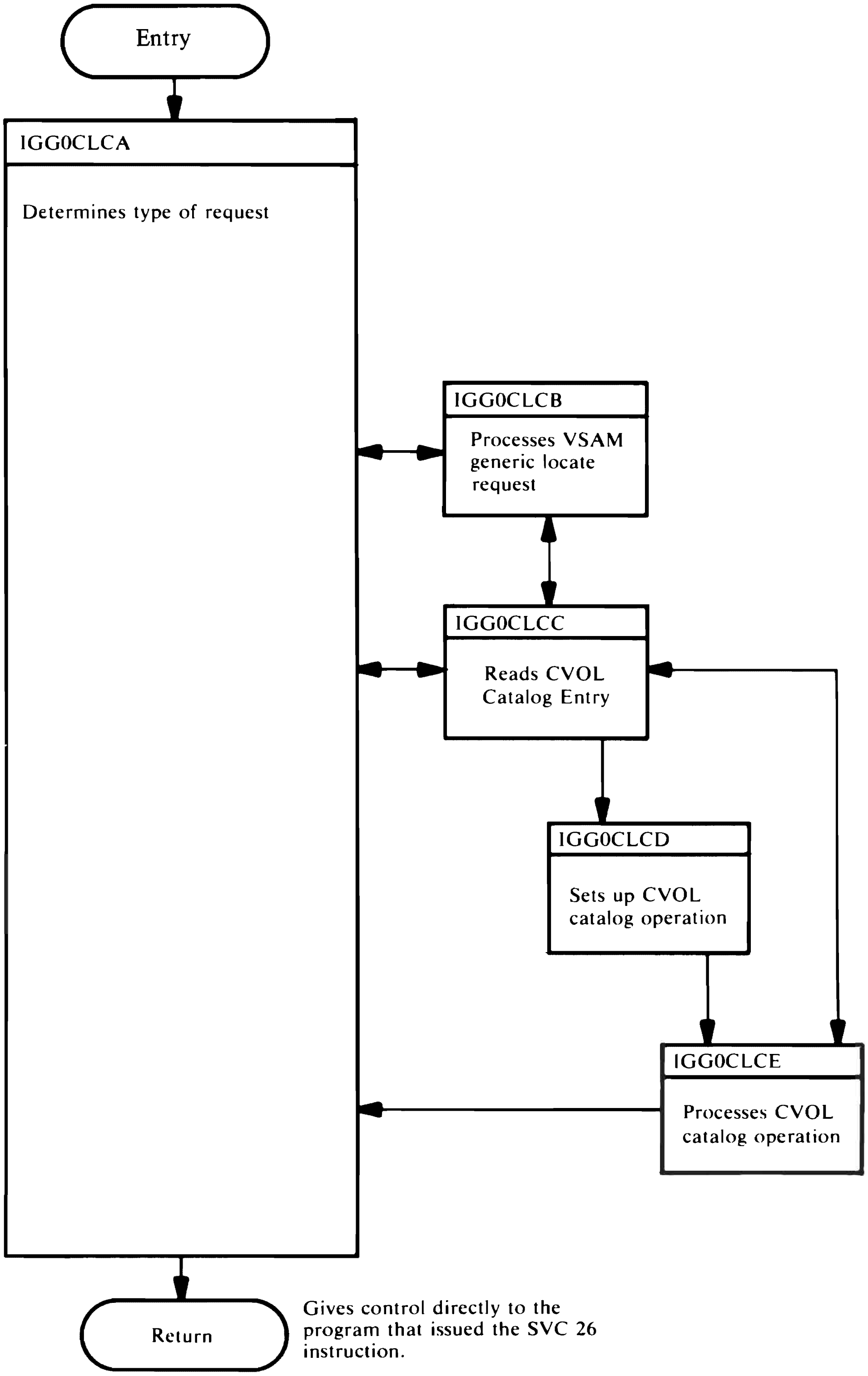

Entry
IGG0CLCA
Determines type of request
IGG0CLCB
Processes VSAM generic locate request
IGG0CLCC
Reads CVOL Catalog Entry
IGG0CLCD
Sets up CVOL catalog operation
IGG0CLCE
Processes CVOL catalog operation
Return
Gives control directly to the program that issued the SVC 26 instruction.

Figure 3-2. Overall Program Organization of the CVOL Processor

Interface Mappers

CSECTs IGG0CLCA and IGG0CLCB are called the First and Second Interface Mappers because they map VSAM requests into OS requests.

CSECT IGG0CLCA

CSECT IGG0CLCA, First Interface Mapper, is the entry and exit point for the CVOL Processor. After ensuring that the PCCB (Private Catalog Control Block) is valid, IGG0CLCA determines what type of request has been sent to the CVOL Processor and calls the appropriate subroutine. Figure 3-3 lists the types of requests IGG0CLCA honors, the subroutine that receives control, the action performed, and any other CSECTs called.

Type of Request	Subroutine	Action Performed	Other CSECTs Called
OS CAMLST format	OSREQ	Sets up and executes an original OS CAMLST format request.	IGG0CLCC
SUPERLOCATE without generic locate specified	SUPERLOCATE	Determines type of superlocate and calls the appropriate procedure: SLGDG, base generation number supplied; SLGDGB, locate GDG base only supplied; or SLNAME, normal superlocate.	IGG0CLCC
VSAM locate	VLOC	Processes a VSAM locate.	IGG0CLCC
VSAM delete	DELETE	Processes a VSAM delete request by issuing an OS UCATDX request and optionally a SCRATCH.	IGG0CLCC
SUPERLOCATE with generic locate specified	GENLOC	Processes a VSAM generic locate.	IGG0CLCB
Access Method Services LISTCAT without GET NEXT option	VLOC	Processes an Access Method Services LISTCAT (not GET NEXT) request	IGG0CLCC

Figure 3-3. Requests to IGG0CLCA

All other VSAM requests not listed in Figure 3-3 are rejected with a return code of 48 in register 15, and control is returned to the issuer of the SVC 26 instruction. CSECT IGG0CLCA is written in PL/S-2, a high-level, proprietary system language. Listings produced for microfiche consist of the PL/S-2 source code, a cross-reference and attribute table, and the assembly code. See the IBM publication *Guide to PL/S – Generated Listings,* for a more detailed explanation of PL/S and its listings.

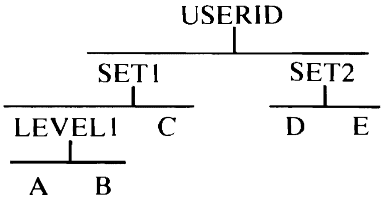
Note: *Guide to PL/S – Generated Listings* describes PL/S-1, but IGG0CLCA uses PL/S-2. If you can read PL/S-1, you can read PL/S-2.

CSECT IGG0CLCB

CSECT IGG0CLCB, Second Interface Mapper, is a modification of TSO LISTCAT from VS2 Release 1. All TSO options have been removed for IGG0CLCB leaving the basic function of LISTCAT. The basic function produces a list of data set names found cataloged under the requested high-level qualifiers. Listings produced for microfiche consist of the Assembler source code, a cross-reference and attribute table, and the assembly code. For more information on Assembler language, see the IBM publications *OS Assembler Language* and *OS Assembler (F) Programmer's Guide.*

Figure 3-4 shows how the Segment (CIRBLOCK) entries are processed after the first segment block information is returned by CIR. This example assumes the '01' and '02' option codes (data set names and index names) have been requested, and that the USERID is used as a node point for the catalog search.

The catalog structure for this example is:



where SET1, SET2, and LEVEL1 are index names and A-E represent the lowest level, fully qualified, data set names.

1. Four segment blocks are initialized. CURNTBLK and FRSTBLK are made to point to the first segment block. The current entry pointer is zeroed. IGG0CLCB then uses routine OBTBLK to find the first segment block containing a zeroed current entry field.

2. Then IGG0CLCB calls CIR, which reads the first index block and formats the entries.

3. Control returns to IGG0CLCB, which then gets the CURNTBLK value, establishes a pointer to, and makes the current-entry field reflect the first entry (see MAIN00).

4. IGG0CLCB analyzes the list entry (see label MAIN01) and finds it to be an index name. Control is then passed to routine INDEXRT, which sets up a parameter list for CIR and uses subroutine OBTBLK to get a new block for the next lower level of qualifiers. (OBTBLK checks the chain, sees that the current-entry pointer is not zeroed, gets the address of the next block in the chain and puts it in CURNTBLK.)

5. OBTBLK returns control to INDEXRT, which calls CIR and reads the next block from the catalog. The current-entry pointer of the second block is updated to point at the first entry in that block. A check is then made to see if the entry is a link entry (in this case, no).

6. Control returns to IGG0CLCB at MAIN01, which continues processing as in step 4.

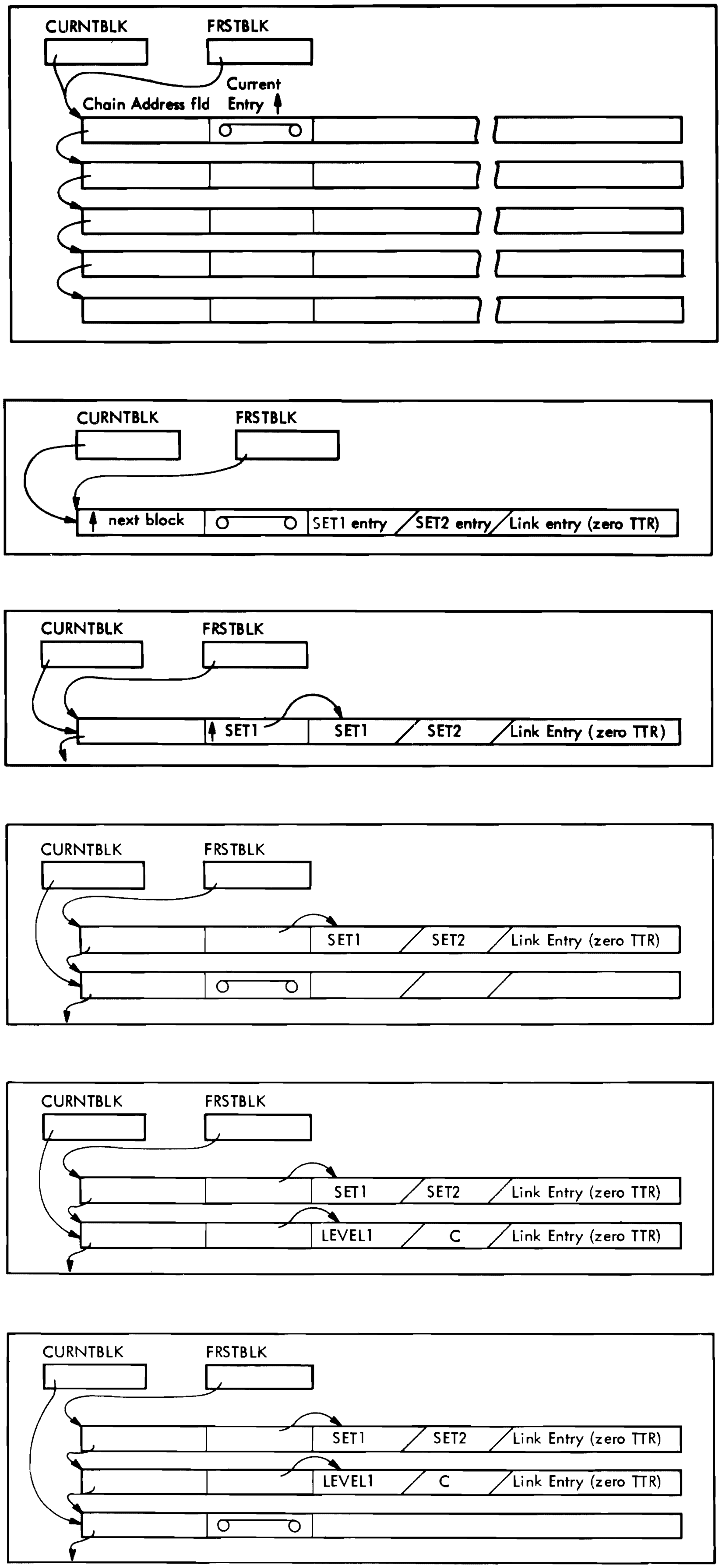
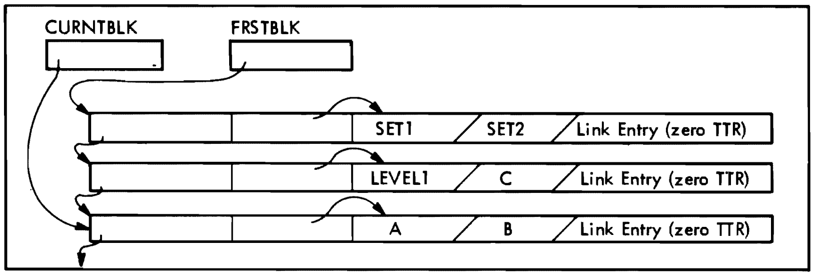


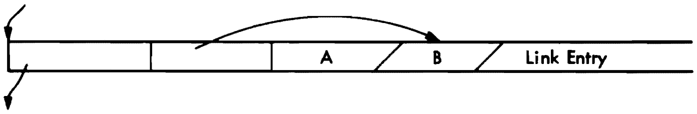
Figure 3-4. IGG0CLCB Example of Catalog Segment Block Handling (1 of 2)

7. Control passes from INDEXRT to CIR, which reads in the block upon return to INDEXRT, the current-entry pointer is updated to point at the first entry of the third block. A check is made for a link entry in this position (in this case, no).

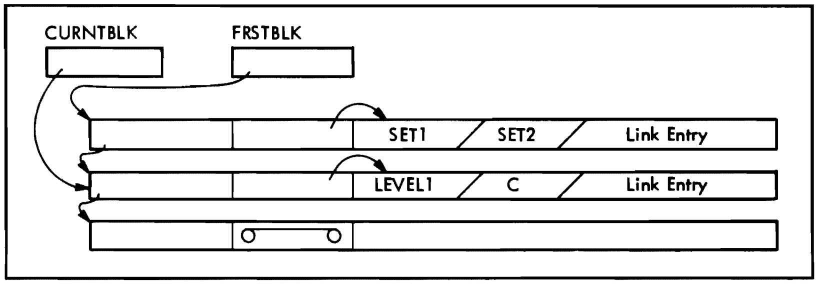


8. Control is returned to IGG0CLCB, through label MAIN01, which tests for entry type and finds the data set name .SET1.LEVEL1.A.

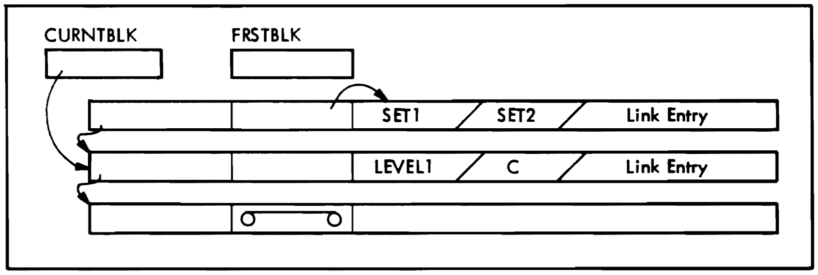
9. After the current entry is processed, control is returned to the POINTER subroutine, which updates the current-entry pointer of the third segment block to point to the B entry. A check is made to see if it is a link-entry (in this case, no).



10. Processing for .SET1.LEVEL1.B continues (as in steps 8 and 9). This time, when the current-entry pointer is updated, the POINTER subroutine finds a zeroed link-entry. The current-entry pointer of the third segment block is cleared, releasing the block for possible future use. CURNTLBK is updated to point to the second block.



11. Control is returned to IGG0CLCB, through MAIN01, which updates the current-entry pointer of the current block to point to the next entry. The next entry is a data set name entry and is processed.



12. The remainder of the operation is summarized as follows:

- When the zero-TTR in the second segment block is encountered, the block is released, and the CURNTBLK is updated to point to the first block.
- The current-entry pointer is updated to point to SET2. SET2 is an index name, which means that CIR is entered to read into segment block 2. The new second-level information (D, E, and a zero-TTR link entry) overlays the old.
- When no more entries remain to be processed (that is, when a zero-TTR link entry is encountered in the first segment block), the POINTER routine passes control to WRAPUP in IGG0CLCB, which cleans up and returns control to IGG0CLCA.

Figure 3-4. IGG0CLCB Example of Catalog Segment Block Handling (2 of 2)

Character Dependency for CSECTs IGG0CLCA and IGG0CLCB

The CSECTs IGG0CLCA and IGG0CLCB require that the character set used at execution time be equivalent to that used at assembly time. The IBM-supplied version of the interface mappers assumes EBCDIC character representations. If a different character set is to be used during execution, the CSECTs must be re-assembled.

System Macros Used by CSECTs IGG0CLCA and IGG0CLCB

Figure 3-5 lists all system macros used by CSECTs IGG0CLCA and IGG0CLCB and the label closest to each point of issue.

Macro	CSECT	Label
DEQ	IGG0CLCA	SRCHPCCB
ENQ	IGG0CLCA	SRCHPCCB
	IGG0CLCA	IGG0CL1A
ESTAE	IGG0CLCA	IGG0CLCA ESTAEXIT
	IGG0CLCB	ESTAEDK WRAPUP
FREEMAIN	IGG0CLCA	IGG0CLCA
	IGG0CLCB	WRAPUP00 WRAPUP02 FREEMMDL FREEML
GETMAIN	IGG0CLCB	IGG0CLCB OUTBLK02 GETMLMDL GETML
LINK	IGG0CLCA	IGG0CLCA
MODESET	IGG0CLCA	GETUSERK GETSVCK
	IGG0CLCB	BUILDNAM OUTBLK07
RETURN	IGG0CLCB	ERREXIT NORMEXIT
SAVE	IGG0CLCB	IGG0CLCB CIR
SCRATCH	IGG0CLCA	DELETE

Figure 3-5. System Macros Used by CSECTs IGG0CLCA and IGG0CLCB

Resource Enqueuing for CSECTs IGG0CLCA and IGG0CLCB

During catalog allocation, CSECT IGG0CLCA enqueues on a chain of Private Catalog Control Blocks (PCCBs). The major name for enqueuing is always SYSZPCCB, and the minor name for enqueuing is always PCCB. CSECT IGG0CLCB does not use resource enqueuing.

During catalog allocation, IGG0CLCA also issues two ENQs to preserve data integrity. For both ENQs the minor name is SYSCTLG.Vxxxxxx, where xxxxxx is the volume serial of the CVOL. The major names used are (1) SYSZOPEN and (2) SYSDSN.

The SYSDSN ENQ prevents the CVOL from being scratched during SVC 26 processing. The SYSZOPEN ENQ is issued to prevent an unallocation that could dequeue the SYSDSN ENQ.

Register Usage for the Interface Mappers

Both Interface Mappers use registers in an identical manner, except as noted.

Registers

10	Second base register for CSECT–IGG0CLCA only
11	Base register for CSECT
12	Base register for WORKCLCA structure

Repackaged Catalog Management

OS Catalog Management in the CVOL Processor has been repackaged into four CSECTs: IGG0CLCC, IGG0CLCD, IGG0CLCE, and IGG0CLCF. The first three CSECTs contain the three OS Catalog Management phases referred to in *OS/VS1 Catalog Mangement Logic*. The three OS/VS phases contain eleven separate modules, while the four CVOL Processor CSECTs contain eleven subroutines. Figure 3-6 gives a comparison of the four CVOL Processor CSECTs versus the three OS Catalog Mangement phases.

Old Phase	Modules Contained	New CSECT	Subroutines Contained	Changes Made
Phase I	IGG0CLC0 IGG0CLC1 IGG0CLC2	IGG0CLCC	IGG0CLC0 IGG0CLC1 IGG0CLC2 IECPBLDL	IGG0CLC1 and IGG0CLC2 return to IGG0CLCA or IGG0CLCB, whichever called IGG0CLCC. IECPBLDL was previously a separate service routine and is now included in IGG0CLCC.
Phase II	IGG0CLC3 IGG0CLC4	IGG0CLCD	IGG0CLC3 IGG0CLC4 IGG0CLC5	IGG0CLC5 was previously included in Phase III.
Phase III	IGG0CLC5 IGG0CLC6 IGG0CLC7	IGG0CLCE	IGG0CLC6 IGG0CLC7	IGG0CLC7 returns to IGG0CLCA or IGG0CLCB, whichever called IGG0CLCC.
	IGC0002H IGG0CLF2	IGG0CLCF	IGC0002H IGG0CLF2	IGC0002H and IGG0CLF2 were not previously considered part of Phase III but as auxiliary service routines. IGC0002H calls IGG0553A for new extents. IGC0002H returns to caller, as does IGG0CLF2. IGG0CLF2 is now only the SYSCTLG Formatter; it no longer performs BPAM directory format.

Figure 3-6. OS Catalog Management Compared to the New CVOL Catalog Management

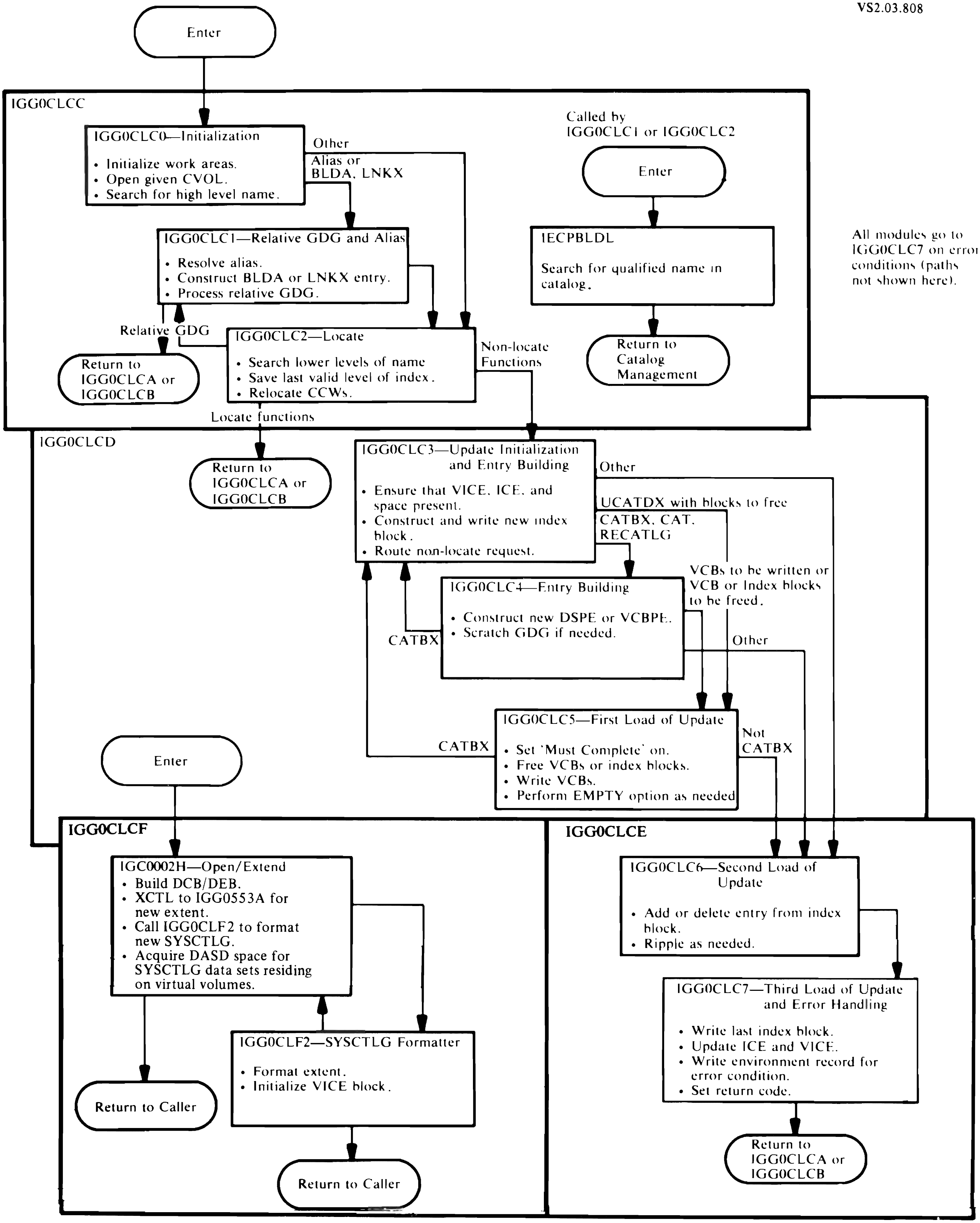
Program Organization of CSECTs IGG0CLCC, IGG0CLCD, IGG0CLCE, and IGG0CLCF

CSECT IGG0CLCC, the entry point for CVOL Catalog Management, is called from CSECT IGG0CLCA or CSECT IGG0CLCB. IGG0CLCC passes control to CSECTs IGG0CLCD and IGG0CLCE via branch instructions. IGC0002H, one of the service subroutines, is invoked via a branch instruction; it passes control to IGG0CLF2 via a branch instruction. The path that occurs through the remaining subroutines of the three CVOL Catalog Management CSECTs depends on both the particular function requested and the entries that are found in the CVOL Catalog.

All of the CVOL Catalog Management CSECTs are re-entrant. They use a common work space, WORKAREA, that is initialized by IGG0CLC0. (See Chapter 5, "Data Areas," for a description of WORKAREA.)

Each block in Figure 3-7 represents a subroutine of the CVOL Catalog Management routines and contains a brief description of the functions it performs. Each path is identified by the function/condition it represents.

Figure 3-7 gives the overall program organization of CSECTs IGG0CLCC, IGG0CLCD, IGG0CLCE, and IGG0CLCF.


Enter
IGG0CLCC
IGG0CLC0—Initialization
• Initialize work areas.
• Open given CVOL.
• Search for high level name.
Other
Alias or
BLDA, LNKX
IGG0CLC1—Relative GDG and Alias
• Resolve alias.
• Construct BLDA or LNKX entry.
• Process relative GDG.
Relative GDG
Return to
IGG0CLCA or
IGG0CLCB
IGG0CLC2—Locate
• Search lower levels of name
• Save last valid level of index.
• Relocate CCWs.
Non-locate
Functions
Locate functions
Called by
IGG0CLC1 or IGG0CLC2
Enter
IECPBLDL
Search for qualified name in catalog.
Return to
Catalog
Management
IGG0CLCD
Return to
IGG0CLCA or
IGG0CLCB
IGG0CLC3—Update Initialization
and Entry Building
• Ensure that VICE, ICE, and space present.
• Construct and write new index block.
• Route non-locate request.
Other
UCATDX with blocks to free
CATBX, CAT,
RECATLG
VCBs to be written or
VCB or Index blocks
to be freed.
IGG0CLC4—Entry Building
• Construct new DSPE or VCBPE.
• Scratch GDG if needed.
CATBX
Other
IGG0CLC5—First Load of Update
• Set 'Must Complete' on.
• Free VCBs or index blocks.
• Write VCBs.
• Perform EMPTY option as needed
CATBX
Not
CATBX
Enter
IGG0CLCF
IGC0002H—Open/Extend
• Build DCB/DEB.
• XCTL to IGG0553A for new extent.
• Call IGG0CLF2 to format new SYSCTLG.
• Acquire DASD space for SYSCTLG data sets residing on virtual volumes.
IGG0CLF2—SYSCTLG Formatter
• Format extent.
• Initialize VICE block.
Return to Caller
Return to Caller
IGG0CLCE
IGG0CLC6—Second Load of
Update
• Add or delete entry from index block.
• Ripple as needed.
IGG0CLC7—Third Load of Update
and Error Handling
• Write last index block.
• Update ICE and VICE.
• Write environment record for error condition.
• Set return code.
Return to
IGG0CLCA or
IGG0CLCB

All modules go to IGG0CLC7 on error conditions (paths not shown here).

Figure 3-7. Overall Program Organization of CSECTs IGG0CLCC, IGG0CLCD, IGG0CLCE, and IGG0CLCF.

CSECTs IGG0CLCC, IGG0CLCD, IGG0CLCE, and IGG0CLCF use Assember language. Listings produced for microfiche consist of the Assembler source code, a cross-reference and attribute table, and the assembly code. For more information on Assembler language, see the IBM publications *OS/VS-DOS/VS-VM/370 Assembler Language* and *OS/VS-VM/370 Assembler Programmer's Guide.*

CSECT IGG0CLCC

CSECT IGG0CLCC performs the read operation. IGG0CLCC performs locate functions and the locate part of non-locate functions. A locate function is a LOCATE by NAME or LOCATE by TTR, that is, a read-only function. A non-locate function is CATBX,UCATDX, BLDA, BLDG, BLDX, DLTA, DLTX, LNKX, DRPX, or RECATLG, that is, an update function.

- IGG0CLC0 (Initialization) initializes work areas and opens the CVOL Catalog.
- IGG0CLC1 (Relative GDG and Alias) resolves aliases and relative GDG numbers.
- IGG0CLC2 (Locate) searches the lower levels of the index structure.
- IECPBLDL (Search) searches for the qualified name in the CVOL Catalog.

CSECT IGG0CLCD

CSECT IGG0CLCD performs the setup operation for adding or deleting entries in the CVOL Catalog. IGG0CLCD checks the validity of the requests against the existing entries in the CVOL Catalog and builds new entries to be added or names entries to be deleted. IGG0CLCD consists of the following subroutines:

- IGG0CLC3 (Update Initialization and Entry Building) begins the update process by building new index blocks and routing the request as needed.
- IGG0CLC4 (Entry Building) builds data set pointer entries to add to the last valid level of the index.
- IGG0CLC5 (First Load of Update) frees index blocks, frees volume control blocks (VCBs), and writes new VCBs.

CSECT IGG0CLCE

CSECT IGG0CLCE performs the write operation. It merges entries into CVOL Catalog blocks, deletes entries from the blocks, and does most of the writing that is needed. IGG0CLCE consists of the following subroutines:

- IGG0CLC6 (Second Load of Update) updates blocks, writes updated blocks to the CVOL Catalog, and ripples the changes as needed to the last block of the updated chain.
- IGG0CLC7 (Third Load of Update and Error Handling) writes the last updated block, updates the control entries, returns control to IGG0CLCA or IGG0CLCB (whichever called IGG0CLCC), and handles error conditions.

Note: Subroutines IGG0CLC5, IGG0CLC6, and IGG0CLC7 are entitled First, Second, and Third Loads respectively. These subroutines are not load modules, and the use of "Load" in their titles is a part of their name in the CVOL Processor.

CSECT IGG0CLCF

The two service subroutines included in IGG0CLCF are:

- IGC0002H (SYSCTLG Open/Extend) opens the CVOL Catalog data set or gets the next extent of that data set when needed.
- IGG0CLF2 (SYSCTLG Formatter) formats a new CVOL Catalog.

For example, follow the path for a CATBX function (request to add a data set name to the CVOL Catalog and create any missing index levels) on a CVOL Catalog. Assume that part of the index structure already exists; that is, this request extends an existing index structure before adding the data set name to the catalog. Refer to Figure 3-7 to coordinate the labels mentioned in the example.

Specifically, this is what each subroutine does to accomplish the CATBX request:

- Entry to CVOL Catalog Management is at IGG0CLCC. IGG0CLCC routes the request to IGG0CLC0.
- IGG0CLC0 initializes work areas, opens the CVOL Catalog, and locates the high-level name of the index structure. The arrow labeled "Other" is the exit path for this example.
- IGG0CLC2 locates the remaining levels of the existing index structure to find the last valid level. The new index is added to that level.
- CATBX is a non-locate function, so control passes to IGG0CLC3. This subroutine reads the control entries, index control entry (ICE) and volume index control entry (VICE), and routes the request (via the arrow labeled "CATBX, CAT, RECAT"). IGG0CLC4 constructs the DSPE. Control returns to IGG0CLC3 (via the arrow labeled 'CATBX'), where the required index levels are built and written into the CVOL Catalog. When an existing level is reached, control passes to IGG0CLC6.

- IGG0CLC6 inserts the new IPE into the index block left by IGG0CLC2. IGG0CLC6 writes the updated block to the CVOL Catalog, and ripples the effect of the change down the index chain, if necessary. The last block of the chain is left in the input/output buffers, but it is not written to the CVOL Catalog.

- IGG0CLC7 writes the last block of the updated index chain, then reads, updates, and rewrites blocks containing the ICE and VICE. Resources are released and control passes back to the caller of IGG0CLCA or IGG0CLCB.

Traces, such as the one just described, are illustrated in the chapter "Diagnostic Aids" as an aid in identifying the CSECTs involved in any particular situation.

Services Used by CSECTs IGG0CLCC, IGG0CLCD, IGG0CLCE, and IGG0CLCF

Two services are used throughout the CVOL Catalog Management subroutines. The prologue commentary for each CSECT lists the specific services used in that CSECT; the services are:

- IECPCNVT is a routine used to convert relative track addresses to absolute addresses. It is accessed through entry point IECPCNVT whose address is found in field CVTPCNVT of the Communication Vector Table (CVT). In the CVOL Catalog Management routines, this routine is used in the closed subroutine labeled "TOABSL".

- IECPRLTV is a routine used to convert absolute track addresses to relative addresses. It is accessed through entry point IECPRLTV, whose address is found in field CVTPRLTV of the CVT. In the CVOL Catalog Management routines, this routine is used in a closed subroutine labeled "TORLTV".

Character Dependency for CSECTs IGG0CLCC, IGG0CLCD, IGG0CLCE, and IGG0CLCF

The CSECTs of CVOL Catalog Management require that the character set used at execution time be equivalent to that used at assembly time. The IBM-supplied version of CVOL Catalog Management assumes EBCDIC character representations. If a different character set is to be used during execution, the CSECTs must be re-assembled. The instructions involved in this dependency are identified by label in the prologue commentary of each CSECT.

System Macros Used by CSECTs IGG0CLCC, IGG0CLCD, IGG0CLCE, and IGG0CLCF

Figure 3-8 lists all of the executable system macros used by CSECTs IGG0CLCC, IGG0CLCD, IGG0CLCE, and IGG0CLCF and the label closest to each point of issue.

Macro	CSECT	Label
DEQ	IGG0CLCC	DEQUE DEQVI ERR00 EXCLUSIV
	IGG0CLCE	FREERES
ENQ	IGG0CLCC	EXCLUSIV
	IGG0CLCD	ENQVI IGG0CLC5
ESTAE	IGG0CLCC	ESTAESET IGG0CLCA
	IGG0CLCE	FREWA2
EXCP	IGG0CLCC	B1
	IGG0CLCD	EXCP3
	IGG0CLCE	EXCP1 EXCP2
	IGG0CLCF	IO3 IO
FREEMAIN	IGG0CLCC	DEQVI
	IGG0CLCD	FRVCBEND
	IGG0CLCE	SKIP5 RPSTST FREEWA2 RB2 RETURN CONTINUE
	IGG0CLCF	RNVIRT4 RBT CONTINUE
GETMAIN	IGG0CLCC	OPENGTMN RELOC
	IGG0CLCD	ENQVI SCRATCH FRVCBTN
	IGG0CLCE	RTTRP
	IGG0CLCF	GETMAINB NOFMT FORMAT
ICBACREL	**IGG0CLCF**	**RTTCTA**
MODESET	IGG0CLCF	EXTENDC EXTENDAA EXTENDB
RACHECK	IGG0CLCC	RACSETUP
WAIT	IGG0CLCC	B1
	IGG0CLCD	EXCP3
	IGG0CLCE	EXCP1 EXCP2
	IGG0CLCF	IO3 IO
WTO	IGG0CLCF	RVIRT8
XCTL	IGG0CLCE	RXP4
	IGG0CLCF	EXTENDAA

Figure 3-8. System Macros Used by CSECTs IGG0CLCC, IGG0CLCD, IGG0CLCE, and IGG0CLCF

Resource Enqueuing for CSECTs IGG0CLCC, IGG0CLCD, IGG0CLCE, and IGG0CLCF

Three resources are used: high-level name, volume index, and volume index control entry (VICE). To prevent an interlock between two callers, the high-level name is *always* enqueued first, the volume index is enqueued second, and the VICE is enqueued last.

The conditions of enqueuing are determined from the request. If the volume index is to be modified, then the volume index must be enqueued exclusively. Otherwise, it can be shared. If a locate function is requested, then the high-level name can be shared. If a non-locate function is requested, the high-level name is enqueued exclusively to protect all lower-level indexes under it.

The major name for enqueuing is always 'SYSCTLG'. The minor name is the high-level name with the UCB (unit control block) address appended to it, 'SYSCTLG' with the UCB address appended to it, or zeros with the UCB address appended to it.

Register Usage for CVOL Catalog Management

With the exception of IGC0002H and IGG0CLF2, the CVOL Catalog Management CSECTs use a common set of registers. Subroutine IGG0CLC0 initializes these registers, and their contents remain throughout. Contents of registers not described are considered destroyed.

Registers

4	Base register for the CSECT
6	Base register for WORKAREA DSECT
8	Base register for CAMLSTD DSECT
12	Linkage register for BAL instructions
14	Linkage register for BAL instructions

CSECT/Subroutine Descriptions

Each of the CSECTs of the CVOL Processor and the subroutines of CVOL Catalog Management are described in this section. The flowcharts are organized into two parts. Supporting text for the subroutine appears beside each part.

Error-condition tests are not shown on the flowcharts. An error condition in CVOL Catalog Management results in a branch to label ERR*xx*, where *xx* is the appropriate error code. There, the error exception code is set, and a branch to IGG0CLC7 occurs. The labels on the flowchart are those used in the assembly listing.

CSECT IGG0CLCA

IGG0CLCA: First Interface Mapper

IGG0CLCA is the entry point. Control comes from IGG0CLA1 or IGG0002F via an XCTL.

Registers

10 Second base register for CSECT
11 First base register for CSECT
12 Base register for WORKCLCA data area

Functions

This CSECT is the entry and exit point for the CVOL Processor. After ensuring that the PCCB is valid, IGG0CLCA determines what type of request has been sent to the CVOL Processor and calls the appropriate subroutine.

Internal Subroutines

For a list of internal subroutines used by IGG0CLCA, please see Figure 3-3 in this chapter.

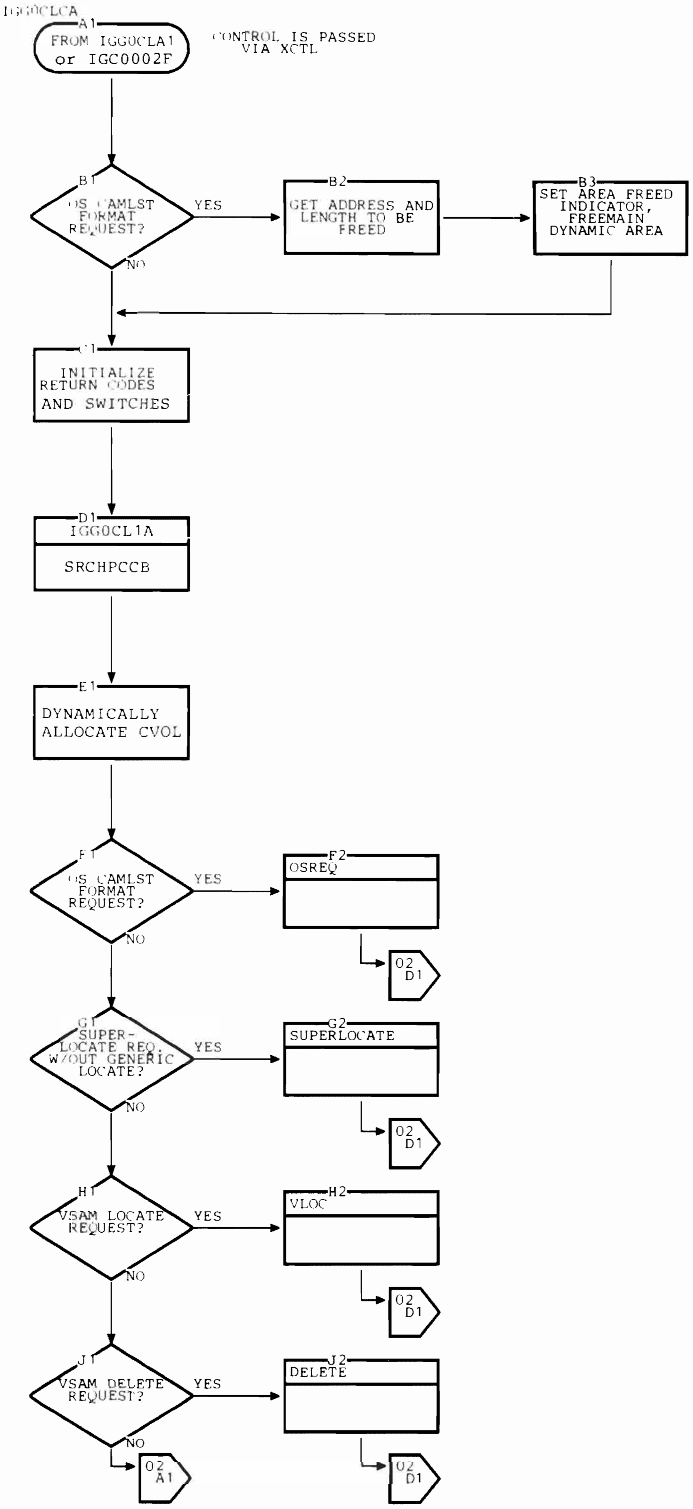
Exits

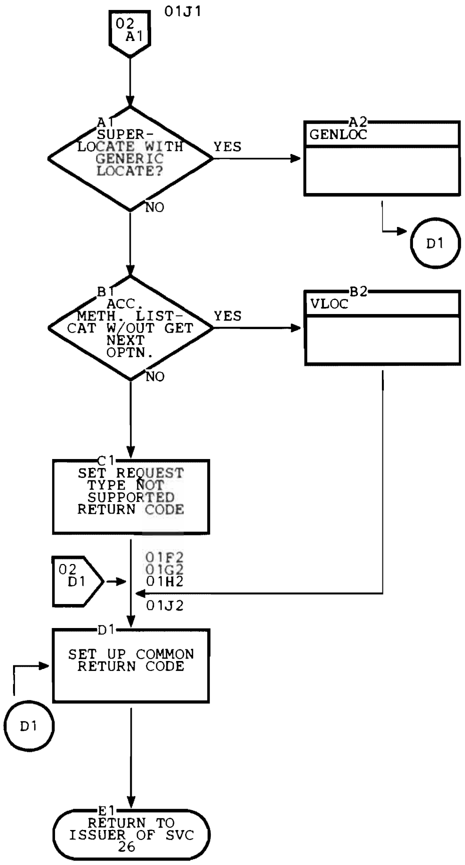
Control passes via a branch instruction to:

- IGG0CLCB from subroutine GENLOC to process a VSAM generic locate.
- IGG0CLCC for all other valid requests.

Error Conditions

For a list of error conditions, please see the lists of return codes under the heading "Processor Exit and Output" at the beginning of this chapter.


IGG0CLCA
A1
FROM IGG0CLA1 or IGC0002F
CONTROL IS PASSED VIA XCTL
B1
OS CAMLST FORMAT REQUEST?
YES
NO
B2
GET ADDRESS AND LENGTH TO BE FREED
B3
SET AREA FREED INDICATOR, FREEMAIN DYNAMIC AREA
C1
INITIALIZE RETURN CODES AND SWITCHES
D1
IGG0CL1A
SRCHPCCB
E1
DYNAMICALLY ALLOCATE CVOL
F1
OS CAMLST FORMAT REQUEST?
YES
NO
F2
OSREQ
02 D1
G1
SUPER-LOCATE REQ. W/OUT GENERIC LOCATE?
YES
NO
G2
SUPERLOCATE
02 D1
H1
VSAM LOCATE REQUEST?
YES
NO
H2
VLOC
02 D1
J1
VSAM DELETE REQUEST?
YES
NO
J2
DELETE
02 A1
02 D1


02
A1
01J1
A1
SUPER-LOCATE WITH GENERIC LOCATE?
YES
NO
A2
GENLOC
D1
B1
ACC. METH. LIST-CAT W/OUT GET NEXT OPTN.
YES
NO
B2
VLOC
C1
SET REQUEST TYPE NOT SUPPORTED RETURN CODE
02
D1
01F2
01G2
01H2
01J2
D1
SET UP COMMON RETURN CODE
D1
E1
RETURN TO ISSUER OF SVC 26

IGG0CLCB: Second Interface Mapper

IGG0CLCB is the entry point. Control comes from subroutine GENLOC in the IGG0CLCA CSECT.

Registers

11 Base register for CSECT
12 Base Register for WORKCLCA data area

Functions

CSECT IGG0CLCB produces a list of data set names found cataloged under the requested high-level qualifiers.

Internal Subroutines

CIR provides an interface between IGG0CLCB and CVOL Catalog Management.

POINTER updates the current entry pointer in the current block.

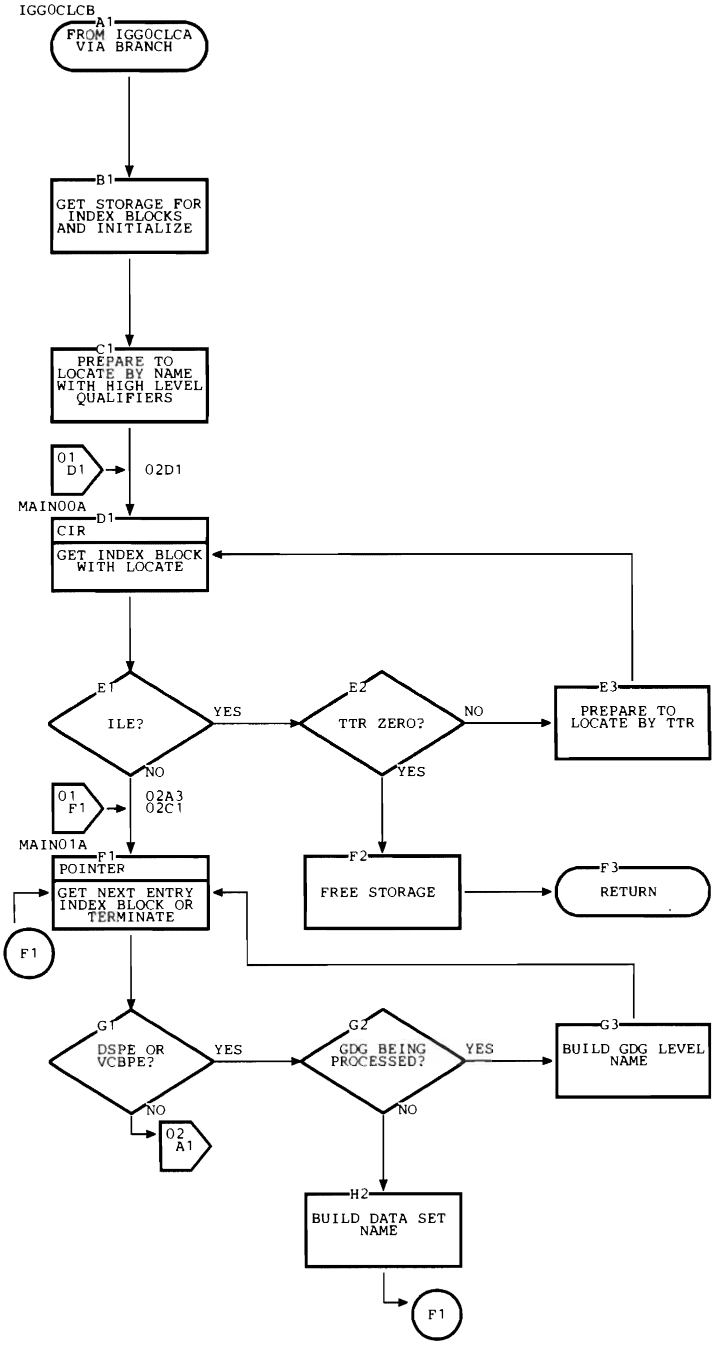
Exits

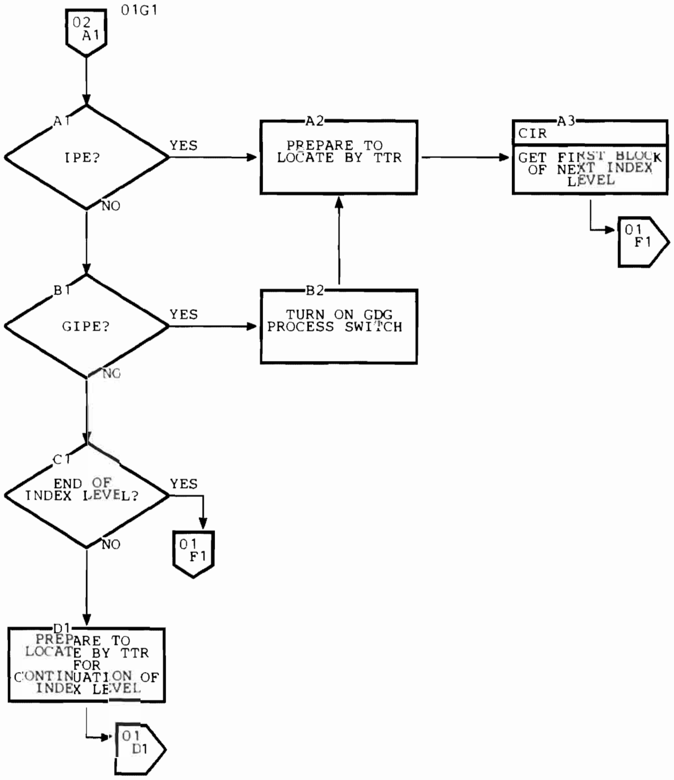
Control passes to IGG0CLCA via a branch instruction with a return code of zero in register 15.

Error Conditions

Control passes to IGG0CLCA via a branch instruction with one of the following return codes in register 15:

Code	*Reason*
4	Data set(s) not found
8	Insufficient storage or ESTAE macro failed
12	User's work area too small




01G1
02 A1
A1
IPE?
YES
NO
A2
PREPARE TO LOCATE BY TTR
A3
CIR
GET FIRST BLOCK OF NEXT INDEX LEVEL
01 F1
B1
GIPE?
YES
NO
B2
TURN ON GDG PROCESS SWITCH
C1
END OF INDEX LEVEL?
YES
NO
01 F1
D1
PREPARE TO LOCATE BY TTR FOR CONTINUATION OF INDEX LEVEL
01 D1

CSECT IGG0CLCC

IGG0CLC0: Initialization

IGG0CLC0 is the entry point. Control comes from IGG0CLCA or ICC0CLCB.

Registers

On Entry:

1	Address of caller's parameter list (CAMLST)
12	Address of Controller III work area
13	Address of register save area within the Controller III work area

On Exit:

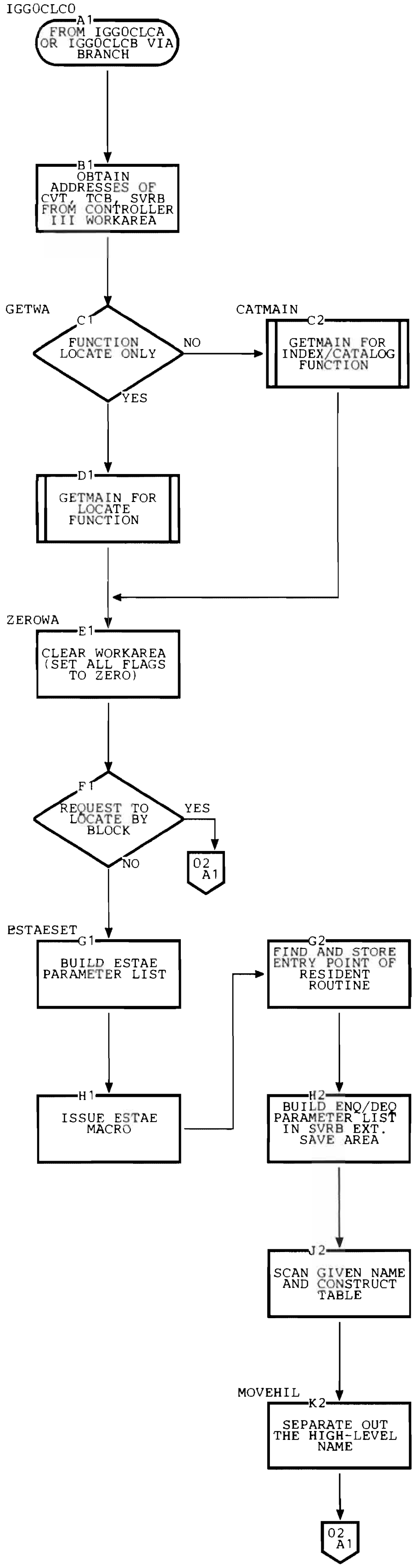
2	Address of UCB
4	Base register for this subroutine
5	Pointer to SVRB extension
6	Base register for WORKAREA DSECT
8	Base register for CAMLSTD DSECT
9	Address of CVT
12	Linkage register for BAL instructions
13	Base register for BLDLAREA
14	Linkage register for BAL instructions

Functions

WORKAREA is the common workspace and communications area for all CVOL Catalog Management subroutines. (WORKAREA has been modified slightly from the OS/VS2 Release 1 version. Refer to Chapter 5, "Data Areas," for a description of WORKAREA.) When a locate function is requested, WORKAREA is built over the caller's 265-byte area, and a second area (called BLDLAREA) is obtained by GETMAIN. BLDLAREA is used with the routine IECPBLDL.

When a non-locate function is requested, a larger area is obtained by GETMAIN for WORKAREA. Part of this area is used for BLDLAREA during execution of subroutines IGG0CLC0, IGG0CLC1, and IGG0CLC2. The BLDLAREA portion of WORKAREA is redefined for use as input/output buffers thereafter.

The first 256 bytes of WORKAREA are set to zero, which initializes all switches and flags. Supervisor addresses and the data set name go into WORKAREA, and the data set name is separated into its components.



IGG0CLC0
A1
FROM IGG0CLCA OR IGG0CLCB VIA BRANCH
B1
OBTAIN ADDRESSES OF CVT, TCB, SVRB FROM CONTROLLER III WORKAREA
GETWA
C1
FUNCTION LOCATE ONLY
NO
YES
CATMAIN
C2
GETMAIN FOR INDEX/CATALOG FUNCTION
D1
GETMAIN FOR LOCATE FUNCTION
ZEROWA
E1
CLEAR WORKAREA (SET ALL FLAGS TO ZERO)
F1
REQUEST TO LOCATE BY BLOCK
YES
NO
02 A1
ESTAESET
G1
BUILD ESTAE PARAMETER LIST
G2
FIND AND STORE ENTRY POINT OF RESIDENT ROUTINE
H1
ISSUE ESTAE MACRO
H2
BUILD ENQ/DEQ PARAMETER LIST IN SVRB EXT. SAVE AREA
J2
SCAN GIVEN NAME AND CONSTRUCT TABLE
MOVEHIL
K2
SEPARATE OUT THE HIGH-LEVEL NAME
02 A1

BLDLAREA is initialized for use as input/output buffers.

The UCB table is searched for device information about the given CVOL Catalog. GETMAIN allocates space for a DCB and a DEB, and IGC0002H opens the CVOL Catalog.

Note: The OPEN macro instruction is not used to open a CVOL Catalog. IGC0002H constructs a modified DCB/DEB for use by CVOL Catalog Management. No CLOSE macro is issued to close a CVOL Catalog. FREEMAIN simply releases the main storage that is used for the modified DCB/DEB.

The first component of the data set name is used as the search parameter for BLDL. Searching begins with the first block of the CVOL Catalog. If BLDL returns a CVOL pointer entry, an error return code is returned to the user.

Internal Subroutines

None.

Exits

Control passes via a branch instruction to:

- IGG0CLC1 if the requested function is BLDA or LNKX, or if the high-level name is an alias.
- IGG0CLC7 for an error condition.
- IGG0CLC2 for all other functions or conditions.

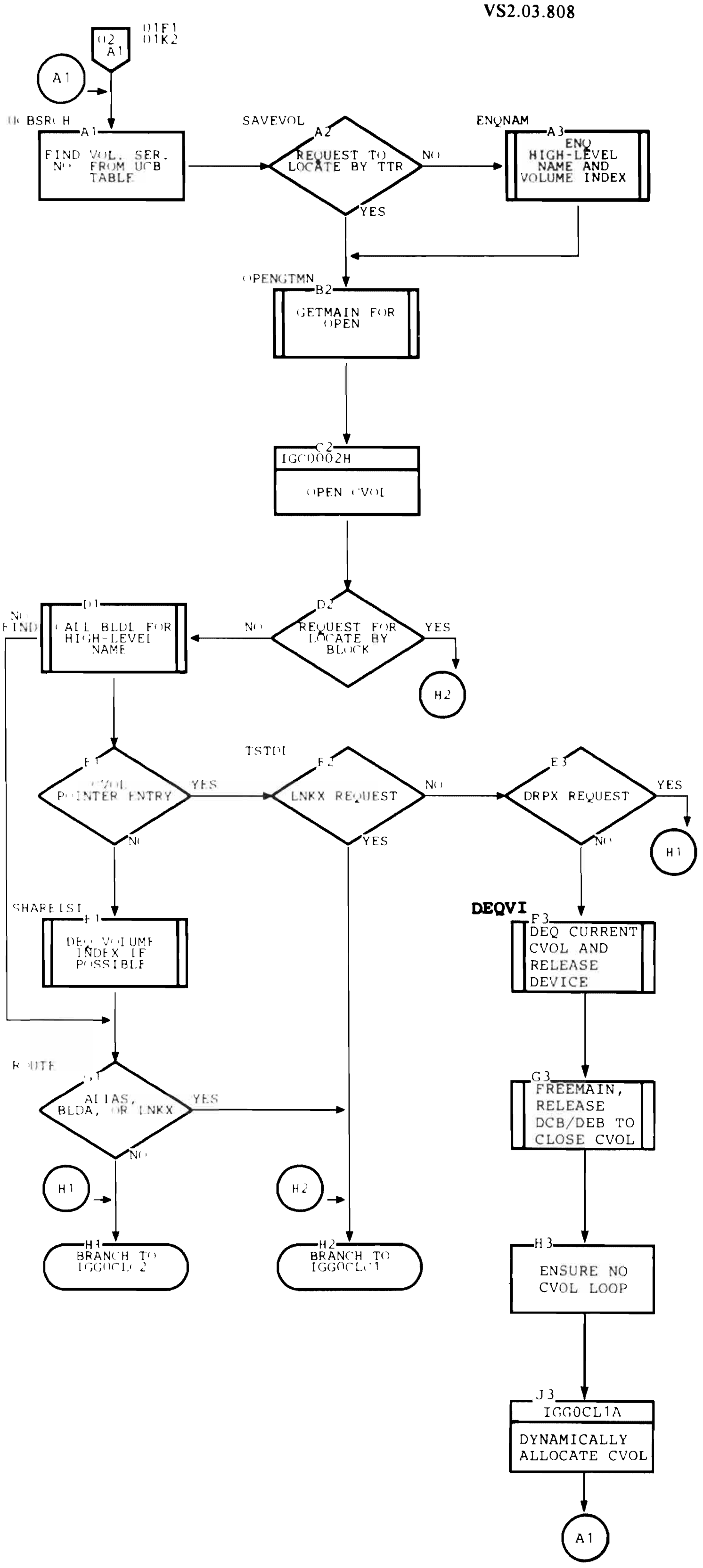
Control passes via a branch to IGC0002H to open the CVOL Catalog and returns to this subroutine.

Error Conditions

Code	*Reason*
4	Volume not mounted or does not contain the CVOL Catalog.
20	Syntax error in data set name.
24	Permanent input/output error.
28	Bad relative track address for the CVOL Catalog.
32	Bad address for caller's area.

References

CVT, TCB, SVRB, DCB, DEB, and UCB are described in *OS/VS2 Data Areas.*



IGG0CLC1: Relative GDG and Alias

IGG0CLC1 is the entry point. Control comes from:

- IGG0CLC0 when the requested function is either BLDA or LNKX, locate-by-block, or when an alias is found (except with a DLTA request).

- IGG0CLC2 when a relative GDG number is found in the data set name.

Registers

4 Base register for this subroutine
6 Base register for WORKAREA DSECT
8 Base register for CAMLSTD DSECT
12 Linkage register for BAL instructions
13 Base register for BLDLAREA
14 Linkage register for BAL instructions

Functions

When locate-by-block is requested, the block is read and returned to the caller.

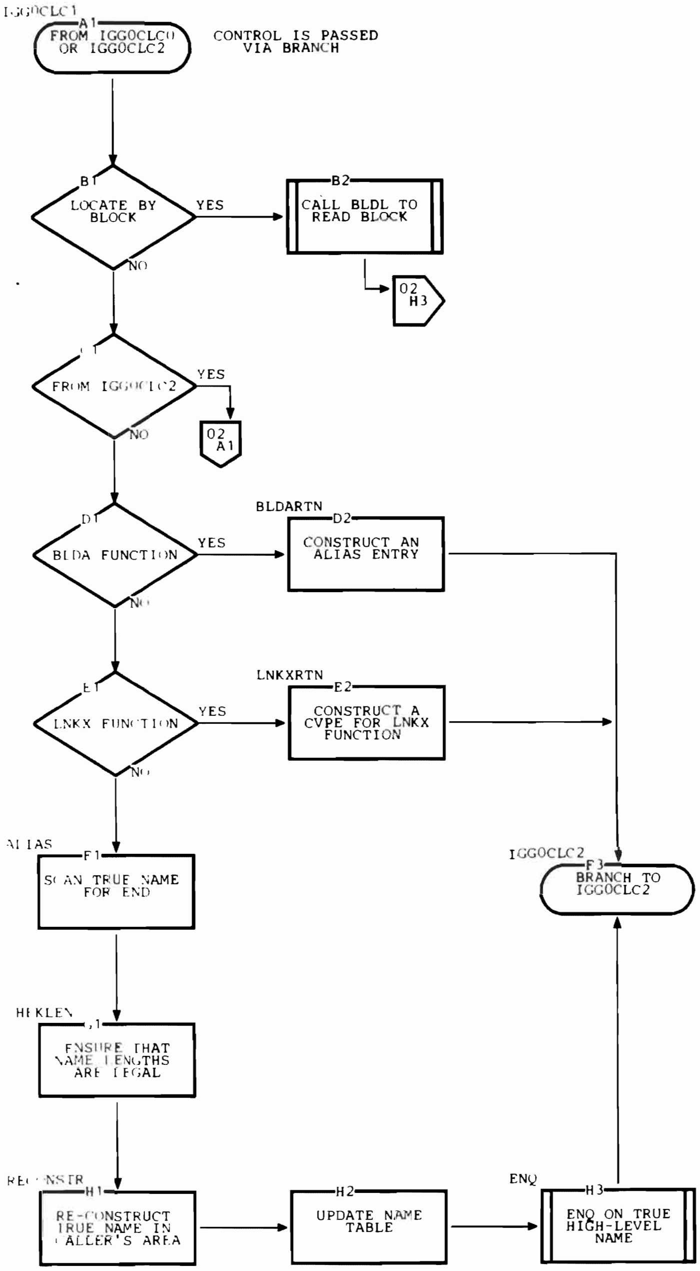
When control comes from IGG0CLC2, control goes to label RELGDG for relative GDG processing.

If the requested function is BLDA or LNKX, the appropriate entry is constructed and control passes to IGG0CLC2 to the update subroutines.

When an alias is discovered, the fully qualified name is reconstructed in the caller's name area, using the true name. The name table is updated to reflect the change, and the high-level name is re-enqueued.

Control comes from IGG0CLC2 when a relative GDG number is discovered in the data set name. This subroutine determines the absolute GDG name for the data set. If the request is a locate function, either the volume list for the data set or a new absolute GDG name is returned to the caller. Otherwise, an error condition exists and IGG0CLC7 is invoked.

The generation number in absolute GDG names is complemented before the names are added to the generation index. Therefore, the most recent entry (the highest generation number) is the first entry in the index, the second most recent entry is the second entry in the index, etc.



When the relative GDG number is negative or zero, an absolute GDG name from the generation index is returned to the caller along with the corresponding volume list. Zero corresponds to the first entry, −1 corresponds to the second entry, and so forth.

When the relative GDG number is positive, a new absolute GDG name is created and returned to the caller. If the generation index is empty, this name is G000*n*V00 (where *n* is the relative number). If the generation index is not empty, the relative GDG number is added to the generation number of the first entry to create the new absolute GDG name.

Internal Subroutines

CALLBLDL calls BLDL routine via entry point IECPBLDL.

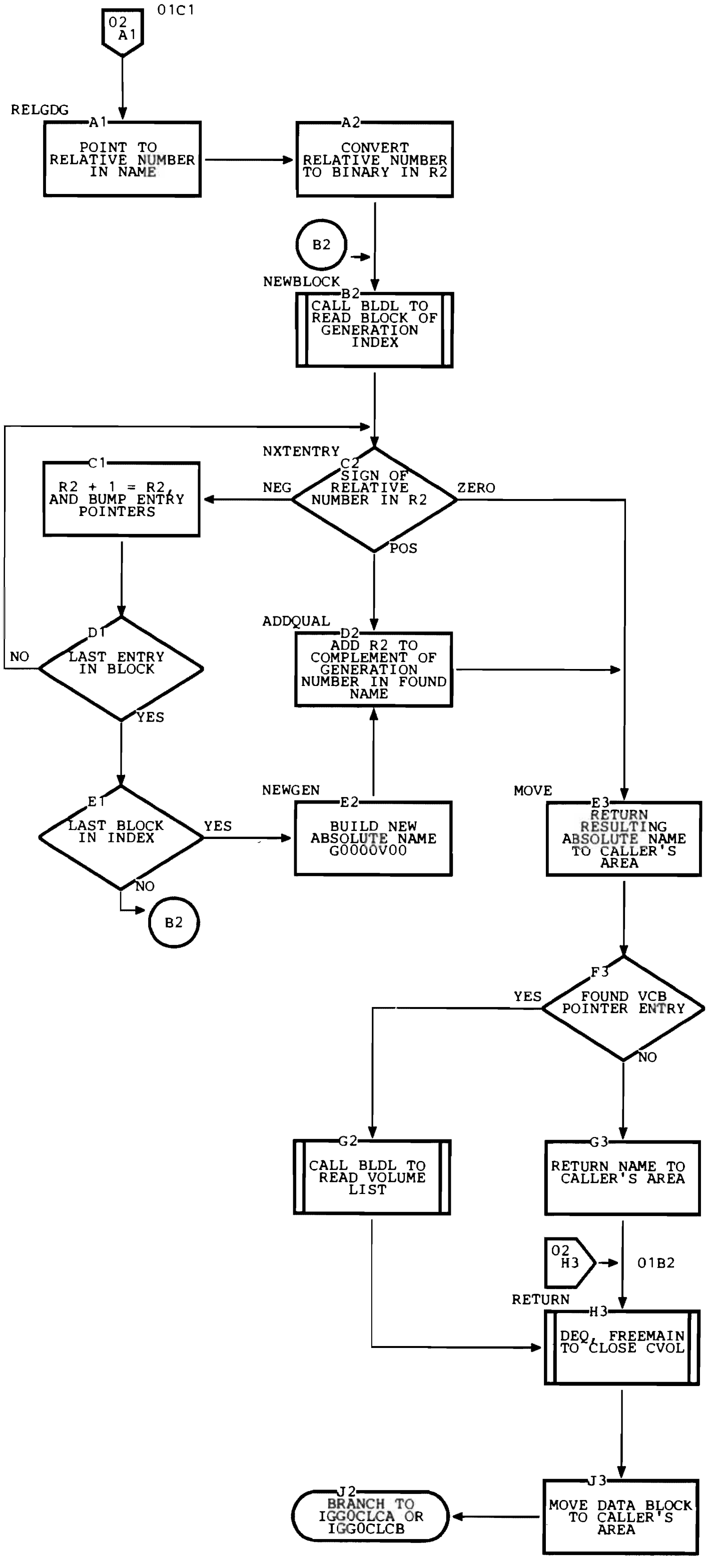
Exits

Control passes via a branch instruction to:

- IGG0CLCA or IGG0CLCB after relative GDG processing.
- IGG0CLC7 for error conditions.
- IGG0CLC2 for all other functions or conditions.

Error Conditions

Code	*Reason*
8	Name not found for locate function, or existing structure inconsistent with request for non-locate function.
20	Syntax error in data set name.
28	Permanent I/O error.



02 A1
01C1
RELGDG
A1
POINT TO RELATIVE NUMBER IN NAME
A2
CONVERT RELATIVE NUMBER TO BINARY IN R2
B2
NEWBLOCK
B2
CALL BLDL TO READ BLOCK OF GENERATION INDEX
NXTENTRY
C2
SIGN OF RELATIVE NUMBER IN R2
NEG
ZERO
POS
C1
R2 + 1 = R2, AND BUMP ENTRY POINTERS
D1
LAST ENTRY IN BLOCK
NO
YES
ADDQUAL
D2
ADD R2 TO COMPLEMENT OF GENERATION NUMBER IN FOUND NAME
E1
LAST BLOCK IN INDEX
YES
NO
B2
NEWGEN
E2
BUILD NEW ABSOLUTE NAME G0000V00
MOVE
E3
RETURN RESULTING ABSOLUTE NAME TO CALLER'S AREA
F3
FOUND VCB POINTER ENTRY
YES
NO
G2
CALL BLDL TO READ VOLUME LIST
G3
RETURN NAME TO CALLER'S AREA
02 H3
01B2
RETURN
H3
DEQ, FREEMAIN TO CLOSE CVOL
J2
BRANCH TO IGG0CLCA OR IGG0CLCB
J3
MOVE DATA BLOCK TO CALLER'S AREA

IGG0CLC2: Locate

IGG0CLC2 is the entry point. Control comes from:

- IGG0CLC1 after resolving an alias or constructing and entry for BLDA or LNKX request.

- IGG0CLC0 for all other functions or conditions.

Registers

4	Base register for this subroutine
6	Base register for WORKAREA DSECT
8	Base register for CAMLSTD DSECT
12	Linkage register for BALR instructions
13	Base register for BLDLAREA
14	Linkage register for BALR instructions

Functions

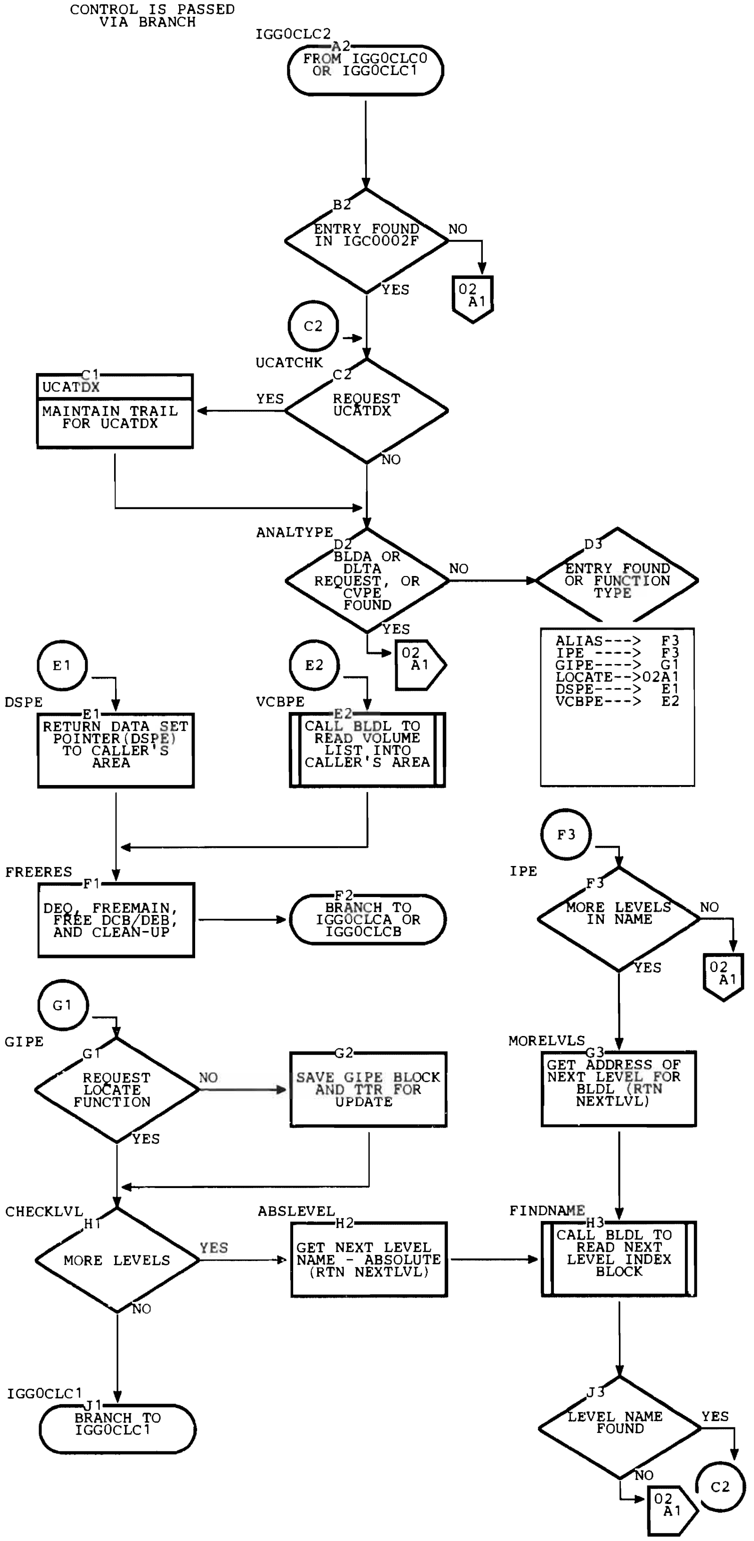
This subroutine completes the locate functions, or finds the last valid index level for a non-locate function. IECPBLDL (BLDL) is used to search index levels successively. At each index level, one component of the data set name is used. When locate-by-name is requested, BLDL is used with each component of the data set name as the search parameter. When BLDL returns an index pointer entry (IPE), IGG0CLC2 uses it to determine the track address for the next search. The search by BLDL continues with the next component of the name.

When BLDL returns a data set pointer entry (DSPE) or volume control block pointer entry (VCBPE), the corresponding volume list is returned to the caller.

When the request is for a non-locate function and BLDL fails to find the next level, the update process is initiated.

The last valid level of the existing index structure is saved to use while updating.

IGG0CLC2 contains skeletal channel programs that are used by the non-locate subroutines. These CCW chains are moved to BLDLAREA.



CONTROL IS PASSED VIA BRANCH
IGG0CLC2
A2
FROM IGG0CLC0 OR IGG0CLC1
B2
ENTRY FOUND IN IGC0002F
NO
02 A1
YES
C2
UCATCHK
C2
REQUEST UCATDX
YES
C1
UCATDX
MAINTAIN TRAIL FOR UCATDX
NO
ANALTYPE
D2
BLDA OR DLTA REQUEST, OR CVPE FOUND
NO
D3
ENTRY FOUND OR FUNCTION TYPE
ALIAS---> F3
IPE ----> F3
GIPE----> G1
LOCATE-->02A1
DSPE----> E1
VCBPE---> E2
YES
02 A1
E1
DSPE
E1
RETURN DATA SET POINTER(DSPE) TO CALLER'S AREA
E2
VCBPE
E2
CALL BLDL TO READ VOLUME LIST INTO CALLER'S AREA
F3
IPE
F3
MORE LEVELS IN NAME
NO
02 A1
YES
FREERES
F1
DEQ, FREEMAIN, FREE DCB/DEB, AND CLEAN-UP
F2
BRANCH TO IGG0CLCA OR IGG0CLCB
G1
GIPE
G1
REQUEST LOCATE FUNCTION
NO
G2
SAVE GIPE BLOCK AND TTR FOR UPDATE
YES
MORELVLS
G3
GET ADDRESS OF NEXT LEVEL FOR BLDL (RTN NEXTLVL)
CHECKLVL
H1
MORE LEVELS
YES
ABSLEVEL
H2
GET NEXT LEVEL NAME - ABSOLUTE (RTN NEXTLVL)
FINDNAME
H3
CALL BLDL TO READ NEXT LEVEL INDEX BLOCK
NO
IGG0CLC1
J1
BRANCH TO IGG0CLC1
J3
LEVEL NAME FOUND
YES
C2
NO
02 A1

Internal Subroutines

UCATDX maintains a TTR trail of blocks that can be deleted.

BLDLCALL calls BLDL to search for one name.

TORLTV converts an absolute address to a relative track address.

NEXTLVL gets the component of the data set name in order to search for the next level.

| RACHK performs RACF authorization checking.

Exits

When the request is a locate function, control passes to:

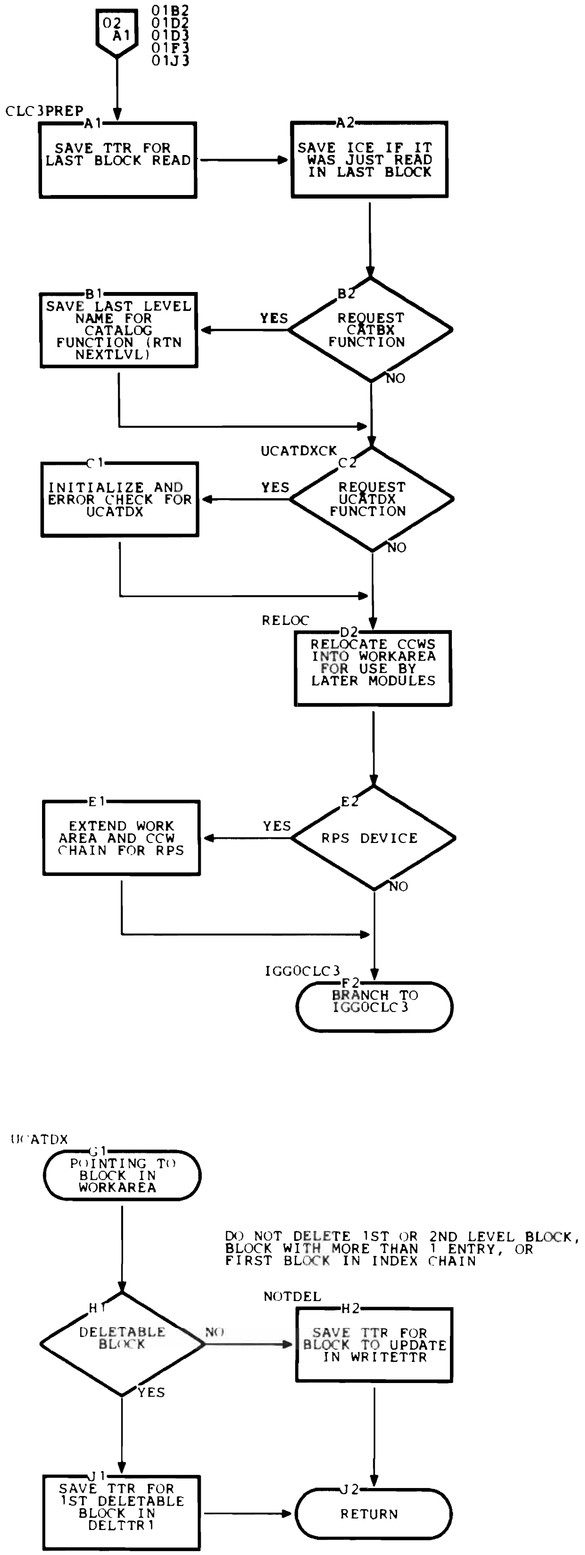
- IGG0CLCA or IGG0CLCB along with the volume list for the data set name.
- IGG0CLC1 for relative GDG number.

When the request is for a non-locate function, control passes to:

- IGG0CLC7 for an error condition.
- IGG0CLC3 for all other functions or conditions.

Error Conditions

Code	*Reason*
8	Name not found for locate request, existing structure inconsistent with non-locate request, or the last entry found was a CVPE with locate request.
12	Last entry found was an IPE or alias with locate request.
16	Non-existent index levels specified.
20	Syntax error in data set name.
28	Permanent I/O error.


02 A1
01B2
01D2
01D3
01F3
01J3
CLC3PREP
A1
SAVE TTR FOR LAST BLOCK READ
A2
SAVE ICE IF IT WAS JUST READ IN LAST BLOCK
B1
SAVE LAST LEVEL NAME FOR CATALOG FUNCTION (RTN NEXTLVL)
YES
B2
REQUEST CATBX FUNCTION
NO
UCATDXCK
C1
INITIALIZE AND ERROR CHECK FOR UCATDX
YES
C2
REQUEST UCATDX FUNCTION
NO
RELOC
D2
RELOCATE CCWS INTO WORKAREA FOR USE BY LATER MODULES
E1
EXTEND WORK AREA AND CCW CHAIN FOR RPS
YES
E2
RPS DEVICE
NO
IGG0CLC3
F2
BRANCH TO IGG0CLC3
UCATDX
G1
POINTING TO BLOCK IN WORKAREA
DO NOT DELETE 1ST OR 2ND LEVEL BLOCK, BLOCK WITH MORE THAN 1 ENTRY, OR FIRST BLOCK IN INDEX CHAIN
H1
DELETABLE BLOCK
NO
YES
NOTDEL
H2
SAVE TTR FOR BLOCK TO UPDATE IN WRITETTR
J1
SAVE TTR FOR 1ST DELETABLE BLOCK IN DELTTR1
J2
RETURN

IECPBLDL:

IECPBLDL is the entry point. Control comes from IGG0CLC1 or IGG0CLC2.

Registers

0	BLDL List address
1	DCB address
13	400 byte WORKAREA address
14	Return address

Functions

This subroutine searches the CVOL Catalog for a name, and returns the information stored in the directory associated with each name. The format of the directory and of the returned information is described in the IBM publication *OS/VS Data Management Services Guide*.

Exits

Control returns to the caller via a branch instruction when IECPBLDL completes its function.

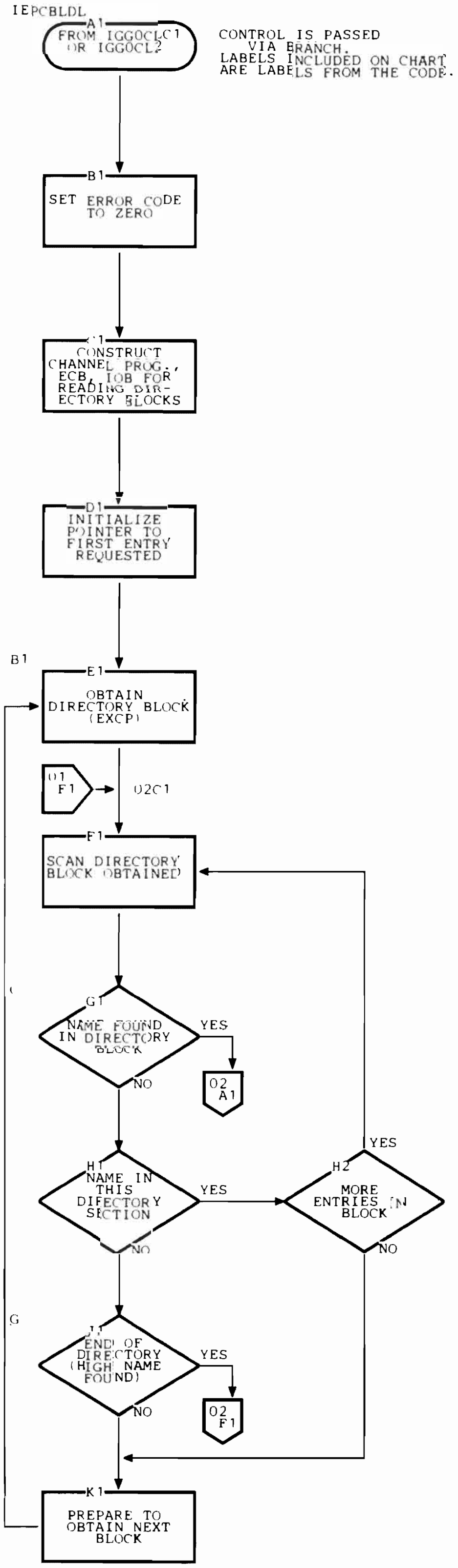
Control returns to the caller via a branch instruction for an error condition.

Internal Subroutines

None.

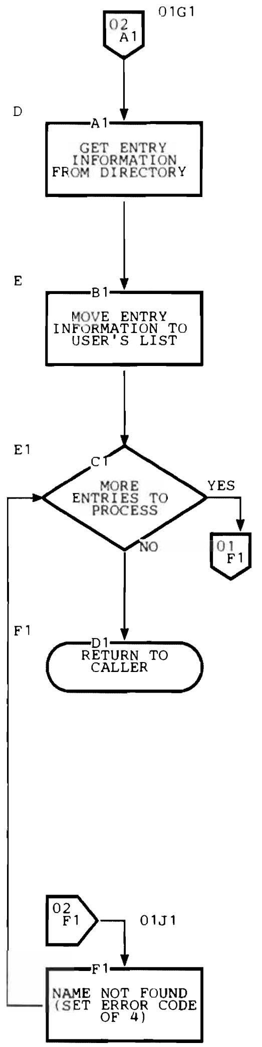
Error Conditions

Code	*Reason*
4	Entry not found
8	Permanent I/O error



CONTROL IS PASSED VIA BRANCH. LABELS INCLUDED ON CHART ARE LABELS FROM THE CODE.

IEPCBLDL
A1
FROM IGG0CLC1 OR IGG0CL2
B1
SET ERROR CODE TO ZERO
C1
CONSTRUCT CHANNEL PROG., ECB, IOB FOR READING DIR-ECTORY BLOCKS
D1
INITIALIZE POINTER TO FIRST ENTRY REQUESTED
B1
E1
OBTAIN DIRECTORY BLOCK (EXCP)
01 F1
02C1
F1
SCAN DIRECTORY BLOCK OBTAINED
G1
NAME FOUND IN DIRECTORY BLOCK
YES
NO
02 A1
H1
NAME IN THIS DIRECTORY SECTION
YES
NO
H2
MORE ENTRIES IN BLOCK
YES
NO
G
J1
END OF DIRECTORY (HIGH NAME FOUND)
YES
NO
02 F1
K1
PREPARE TO OBTAIN NEXT BLOCK


01G1
02 A1
D
A1
GET ENTRY INFORMATION FROM DIRECTORY
E
B1
MOVE ENTRY INFORMATION TO USER'S LIST
E1
C1
MORE ENTRIES TO PROCESS
YES
NO
01 F1
F1
D1
RETURN TO CALLER
02 F1
01J1
F1
NAME NOT FOUND (SET ERROR CODE OF 4)

IGG0CLC3: Update Initialization and Entry Building

IGG0CLC3 is the entry point. Control comes from:

- IGG0CLC4 after constructing a DSPE for a CATBX request.
- IGG0CLC5 after writing a volume control block and constructing a VCBPE for a CATBX function.
- IGG0CLC2 for all other functions or conditions.

Registers

4	Base register for this subroutine
6	Base register for WORKAREA DSECT
8	Base register for CAMLSTD DSECT
12	Linkage register for BAL instructions
14	Linkage register for BAL instructions

Functions

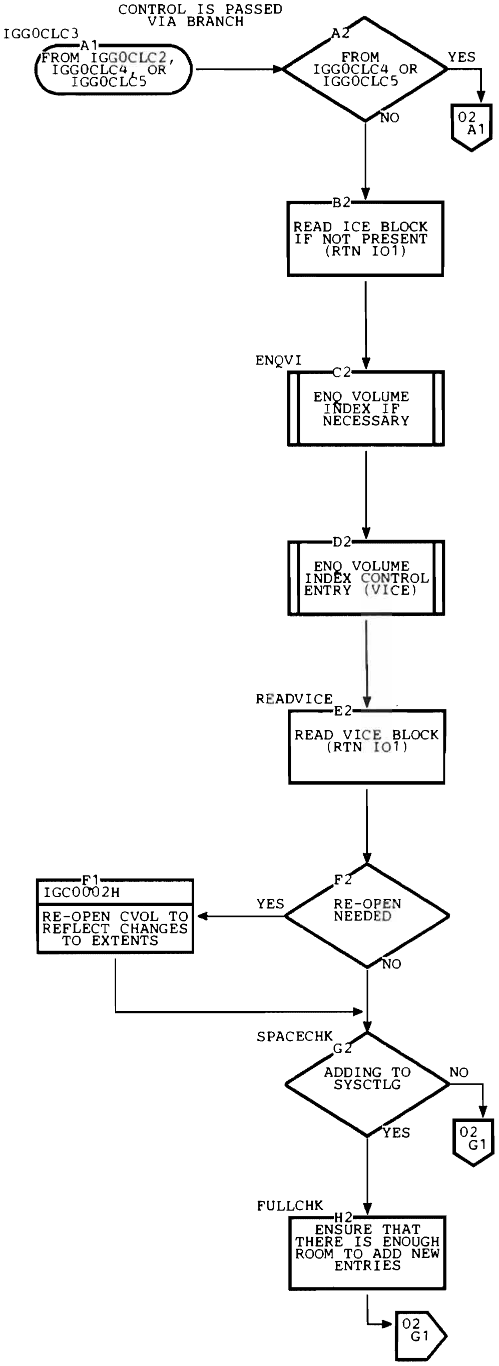
When entry is from IGG0CLC4 or IGG0CLC5, index levels for a CATBX request must be built. Control goes to label CATBX on the next subchart.

When control comes from IGG0CLC2, the index control entry (ICE), if not already present, and volume index control entry (VICE) are read. The request is checked against available space in the CVOL Catalog to ensure that there is enough space to make the required changes.

This module constructs new index levels for a CATBX function and constructs an index pointer entry for the new level to be added to the existing structure. When the requested function is DRPX or DLTA, the entry to be removed is named and IGG0CLC6 deletes it.

When CATBX is requested, IGG0CLC4 is called to construct the DSPE. Control returns to IGG0CLC3, where the required index levels are built and written into the CVOL Catalog. Each level results in an index pointer entry (IPE) that must be added to the next higher level. When an existing level is reached, control passes to IGG0CLC6.

IGG0CLC3 routes the update request to the subroutines that perform the appropriate function.


CONTROL IS PASSED VIA BRANCH
IGG0CLC3
A1
FROM IGG0CLC2, IGG0CLC4, OR IGG0CLC5
A2
FROM IGG0CLC4 OR IGG0CLC5
YES
NO
02 A1
B2
READ ICE BLOCK IF NOT PRESENT (RTN IO1)
ENQVI
C2
ENQ VOLUME INDEX IF NECESSARY
D2
ENQ VOLUME INDEX CONTROL ENTRY (VICE)
READVICE
E2
READ VICE BLOCK (RTN IO1)
F1
IGC0002H
RE-OPEN CVOL TO REFLECT CHANGES TO EXTENTS
YES
F2
RE-OPEN NEEDED
NO
SPACECHK
G2
ADDING TO SYSCTLG
NO
YES
02 G1
FULLCHK
H2
ENSURE THAT THERE IS ENOUGH ROOM TO ADD NEW ENTRIES
02 G1

Internal Subroutines

MOVELVL gets the component of the data set name for the current index lèvel from the name table.

WRTSRCH writes a new block to the CVOL Catalog and searches for another available block.

KEYICE constructs a new index block, with its ICE and key.

TOABSL converts a relative track address to an absolute track address.

TORLTV converts an absolute track address to a relative track address.

IO1 performs EXCP input/output. This subroutine invokes IGC0002H if a new extent of the CVOL Catalog is required.

Exits

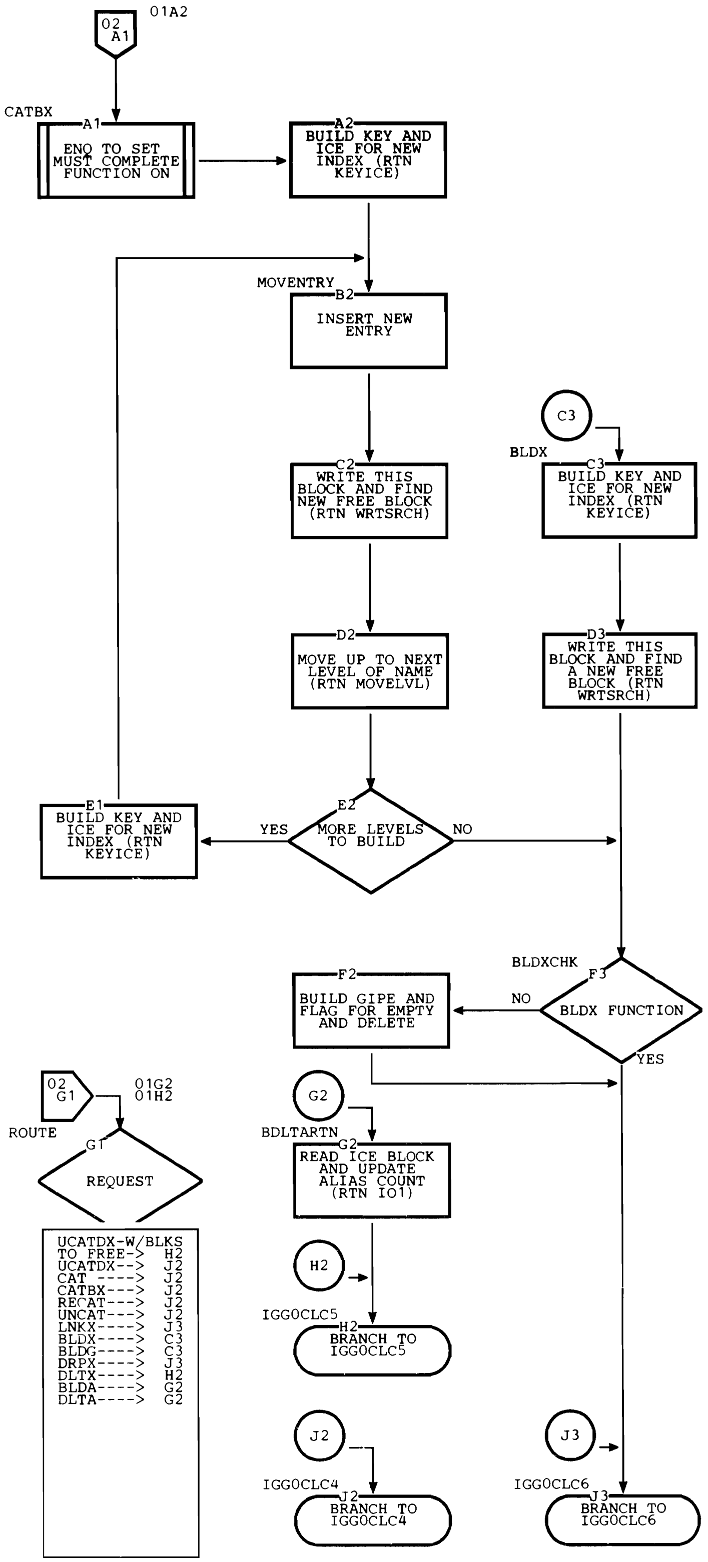
Control passes via a branch instruction to:

- IGG0CLC4 when the requested function is CATBX, CAT, RECAT, or UNCAT.
- IGG0CLC5 when blocks of the CVOL Catalog need to be freed, or when new blocks have been written, but the requested process has been aborted.
- IGG0CLC7 for error conditions.
- IGG0CLC6 for all other functions or conditions.

Control passes via a branch to IGC0002H when a new extent of the CVOL Catalog is required or when the CVOL Catalog must be re-opened, and returns to this subroutine.

Error Conditions

Code	*Reason*
8	Existing structure is inconsistent with the requested function.
12	Attempt to delete a non-empty index level.
20	Not enough space available in the CVOL Catalog to perform the requested function.
28	Permanent I/O error.



02 A1
01A2
CATBX
A1
ENQ TO SET MUST COMPLETE FUNCTION ON
A2
BUILD KEY AND ICE FOR NEW INDEX (RTN KEYICE)
MOVENTRY
B2
INSERT NEW ENTRY
C2
WRITE THIS BLOCK AND FIND NEW FREE BLOCK (RTN WRTSRCH)
C3
BLDX
C3
BUILD KEY AND ICE FOR NEW INDEX (RTN KEYICE)
D2
MOVE UP TO NEXT LEVEL OF NAME (RTN MOVELVL)
D3
WRITE THIS BLOCK AND FIND A NEW FREE BLOCK (RTN WRTSRCH)
E1
BUILD KEY AND ICE FOR NEW INDEX (RTN KEYICE)
YES
E2
MORE LEVELS TO BUILD
NO
BLDXCHK
F3
BLDX FUNCTION
NO
YES
F2
BUILD GIPE AND FLAG FOR EMPTY AND DELETE
02 G1
01G2
01H2
ROUTE
G1
REQUEST
G2
BDLTARTN
G2
READ ICE BLOCK AND UPDATE ALIAS COUNT (RTN IO1)
UCATDX-W/BLKS TO FREE-> H2
UCATDX--> J2
CAT ----> J2
CATBX---> J2
RECAT---> J2
UNCAT---> J2
LNKX----> J3
BLDX----> C3
BLDG----> C3
DRPX----> J3
DLTX----> H2
BLDA----> G2
DLTA----> G2
H2
IGGOCLC5
H2
BRANCH TO IGGOCLC5
J2
IGGOCLC4
J2
BRANCH TO IGGOCLC4
J3
IGGOCLC6
J3
BRANCH TO IGGOCLC6

IGG0CLC4: Entry Building

IGG0CLC4 is the entry point. Control comes from IGG0CLC3 when the requested function is CAT, CATBX, RECAT, or UNCAT.

Registers

4	Base register for this subroutine
6	Base register for WORKAREA DSECT
8	Base register for CAMLSTD DSECT
12	Linkage register for BAL instructions
14	Linkage register for BAL instructions

Functions

If the requested function is RECAT or UNCAT, control passes to label ALTERTN. If the request is for CAT or CATBX, control passes to label CATRTN.

This subroutine constructs a new DSPE or VCBPE. When there are more than five volumes in the volume list, IGG0CLC5 is invoked to write volume control blocks.

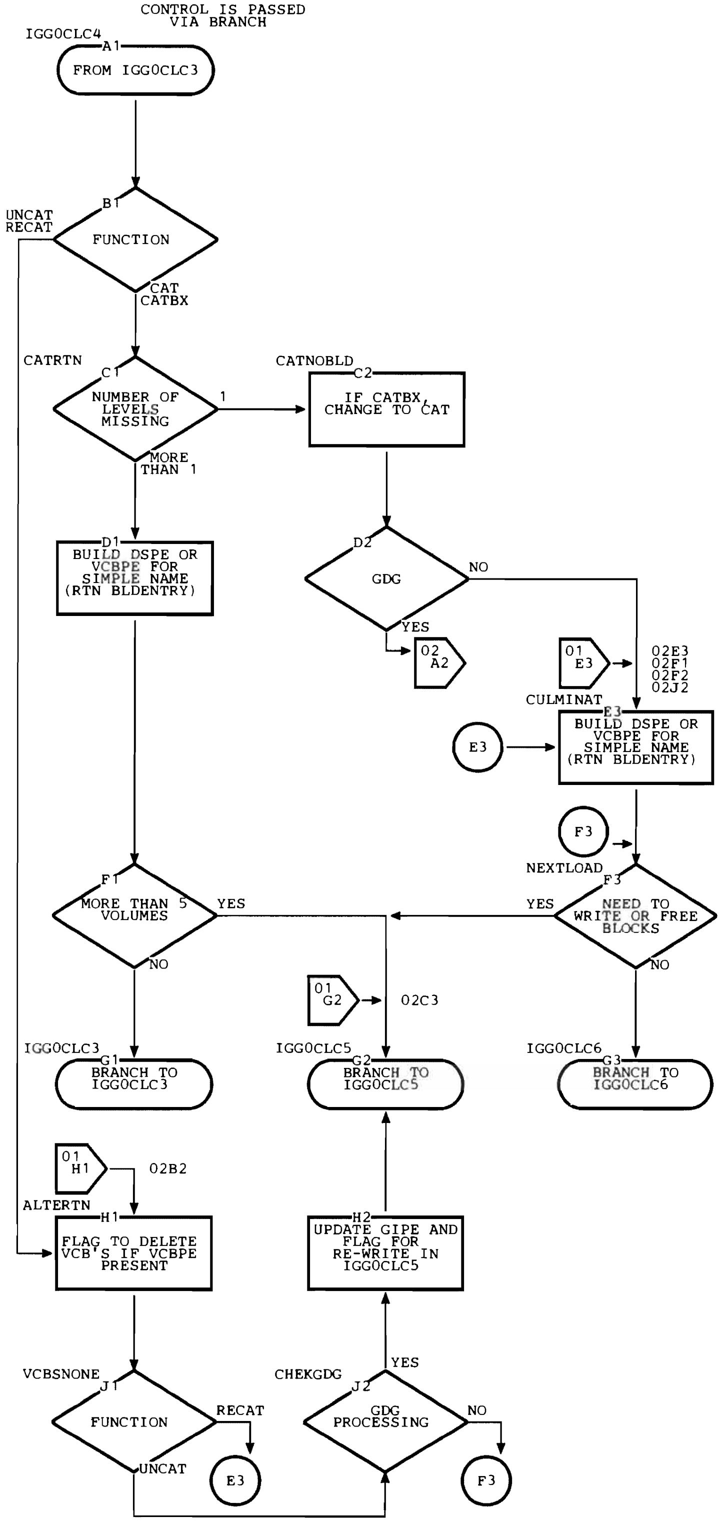
If the data set name is not for a generation data group, control passes to label CULMINAT. Part two of the flowchart deals with cataloging functions to a generation index. The new member of a GDG is checked against existing members to see if this is a new version of an existing member.

If the maximum number of entries that a generation index can hold is exceeded with this addition, the EMPTY and DELETE options for GDG are processed.

If EMPTY was specified, IGG0CLC5 will remove all entries from the generation index before adding the new entry. Otherwise, IGG0CLC5 will remove only the oldest entry before adding the new entry. IGG0CLC4 flags what is to be done.

If DELETE was specified, IGG0CLC4 issues the SCRATCH macro instruction on every data set name that will be removed by IGG0CLC5. If DELETE is not specified, nothing is scratched.

The RECAT and UNCAT functions are processed by naming the old entry. IGG0CLC6 deletes the old entry when it gets control. For RECAT, a new entry is also constructed. IGG0CLC6 adds this new entry to the CVOL Catalog.



Internal Subroutines

TOABSL2 converts an absolute track address to a relative track address.

IO2 performs EXCP input/output operations.

GET reads a block from the CVOL Catalog into the input buffer of BLDLAREA.

SETUP points to the first and last entry in an index block.

INCR bumps the pointer to the next entry in an index block.

BLDENTRY constructs a data set pointer entry (DSPE) or a volume control block pointer entry (VCBPE).

SCRATCH performs a SCRATCH macro instruction for one data set and its VCBs.

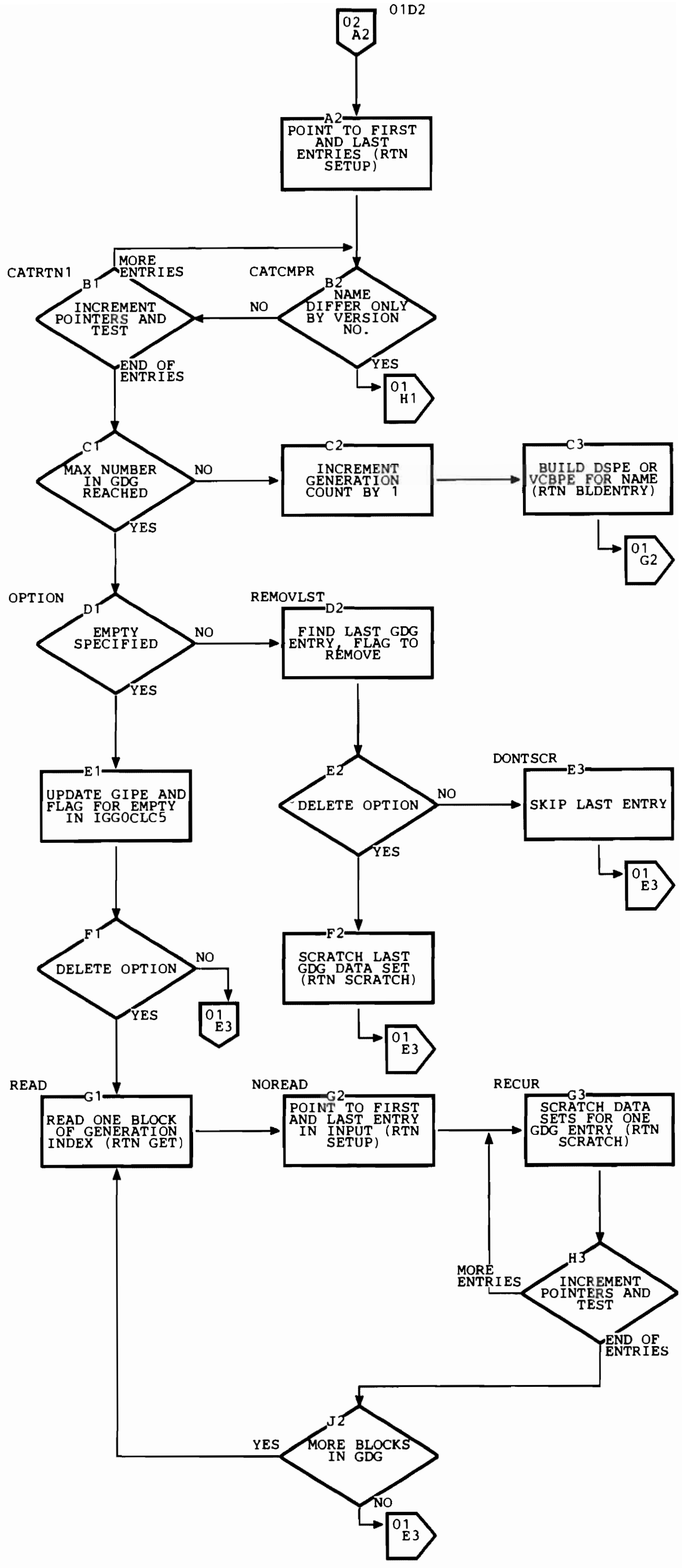
Exits

Control is passed via a branch instruction to:

- IGG0CLC3 when CATBX is being performed.
- IGG0CLC5 when auxiliary reading or writing is required:
  - Volume control blocks (VCBs) need to be written.
  - VCBs or index blocks need to be freed.
  - The DELETE option of a GDG needs to be performed.
  - Updated GDG index blocks need to be rewritten.
- IGG0CLC7 for error conditions.
- IGG0CLC6 for all other functions or conditions.

Error Conditions

Code	*Reason*
8	Existing structure is inconsistent with requested function.
16	Non-existent index level required.
24	Improperly námed GDG data set, or GDG data set to be added is older than existing GDG data sets.
28	Permanent I/O error.


01D2
02 A2
A2
POINT TO FIRST AND LAST ENTRIES (RTN SETUP)
MORE ENTRIES
CATRTN1
B1
INCREMENT POINTERS AND TEST
END OF ENTRIES
CATCMPR
B2
NAME DIFFER ONLY BY VERSION NO.
NO
YES
01 H1
C1
MAX NUMBER IN GDG REACHED
NO
YES
C2
INCREMENT GENERATION COUNT BY 1
C3
BUILD DSPE OR VCBPE FOR NAME (RTN BLDENTRY)
01 G2
OPTION
D1
EMPTY SPECIFIED
NO
YES
REMOVLST
D2
FIND LAST GDG ENTRY, FLAG TO REMOVE
E1
UPDATE GIPE AND FLAG FOR EMPTY IN IGGOCLC5
E2
DELETE OPTION
NO
YES
DONTSCR
E3
SKIP LAST ENTRY
01 E3
F1
DELETE OPTION
NO
YES
01 E3
F2
SCRATCH LAST GDG DATA SET (RTN SCRATCH)
01 E3
READ
G1
READ ONE BLOCK OF GENERATION INDEX (RTN GET)
NOREAD
G2
POINT TO FIRST AND LAST ENTRY IN INPUT (RTN SETUP)
RECUR
G3
SCRATCH DATA SETS FOR ONE GDG ENTRY (RTN SCRATCH)
MORE ENTRIES
H3
INCREMENT POINTERS AND TEST
END OF ENTRIES
J2
MORE BLOCKS IN GDG
YES
NO
01 E3

IGG0CLC5: First Load of Update

IGG0CLC5 is the entry point. Control comes from IGG0CLC3 or IGG0CLC4 when blocks of the CVOL Catalog need to be written or freed.

Registers

4	Base register for this subroutine
6	Base register for WORKAREA DSECT
8	Base register for CAMLSTD DSECT
12	Linkage register for BAL instructions
14	Linkage register for BAL instructions

Functions

ENQ is reissued to ensure that any changes to the CVOL Catalog will be completed.

This subroutine consists of a series of tests for required functions. Each test calls the appropriate internal subroutine to perform one function if it is required.

Chains of volume control blocks (VCBs) and index blocks are freed if possible; that is, they are set to zeros and rewritten into the CVOL Catalog. They then have a key of zero, indicating that they are available for use.

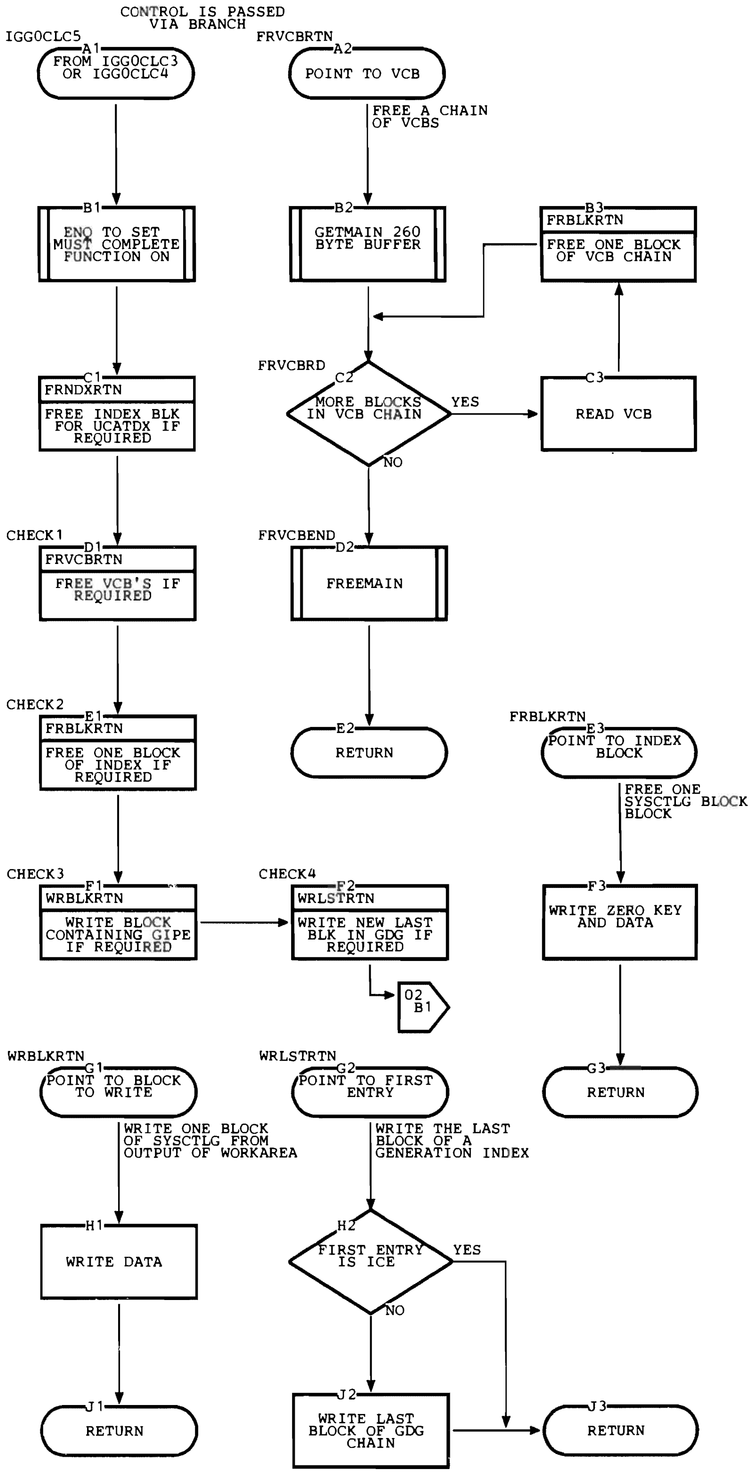
If changes have been made to a generation index, the block containing the generation index pointer entry (GIPE) must be updated. Likewise, the last block of the generation index may need to be rewritten.

If a generation index reached its maximum number of entries in IGG0CLC4 and the EMPTY option was specified, that option is processed. IGG0CLC4 will have already processed the DELETE option.

If the generation index is full and the EMPTY option was not specified, the name with the lowest generation number (the oldest data set) is removed from the index.

An UCATDX request can result in unneeded index blocks. Such blocks are freed.

If a CATBX function is requested and the volume list contains more than five volumes, volume control blocks are constructed from that list and written to the CVOL Catalog.



Internal Subroutines

WRBLKRTN, WRLSTRTN, EMPTYRTN, FRNDXRTN, FRVCBRTN, FRBLKRTN, and BLVCBRTN are shown on the flowchart.

SETUP points to the first and last entry in an index block.

INCR increments the pointer to the next entry in an index block.

TOABSL converts a relative track address to an absolute track address.

TORLTV converts an absolute track address to a relative track address.

IO3 performs EXCP input/output operations. This subroutine invokes IGC0002H if a new extent is required.

Exits

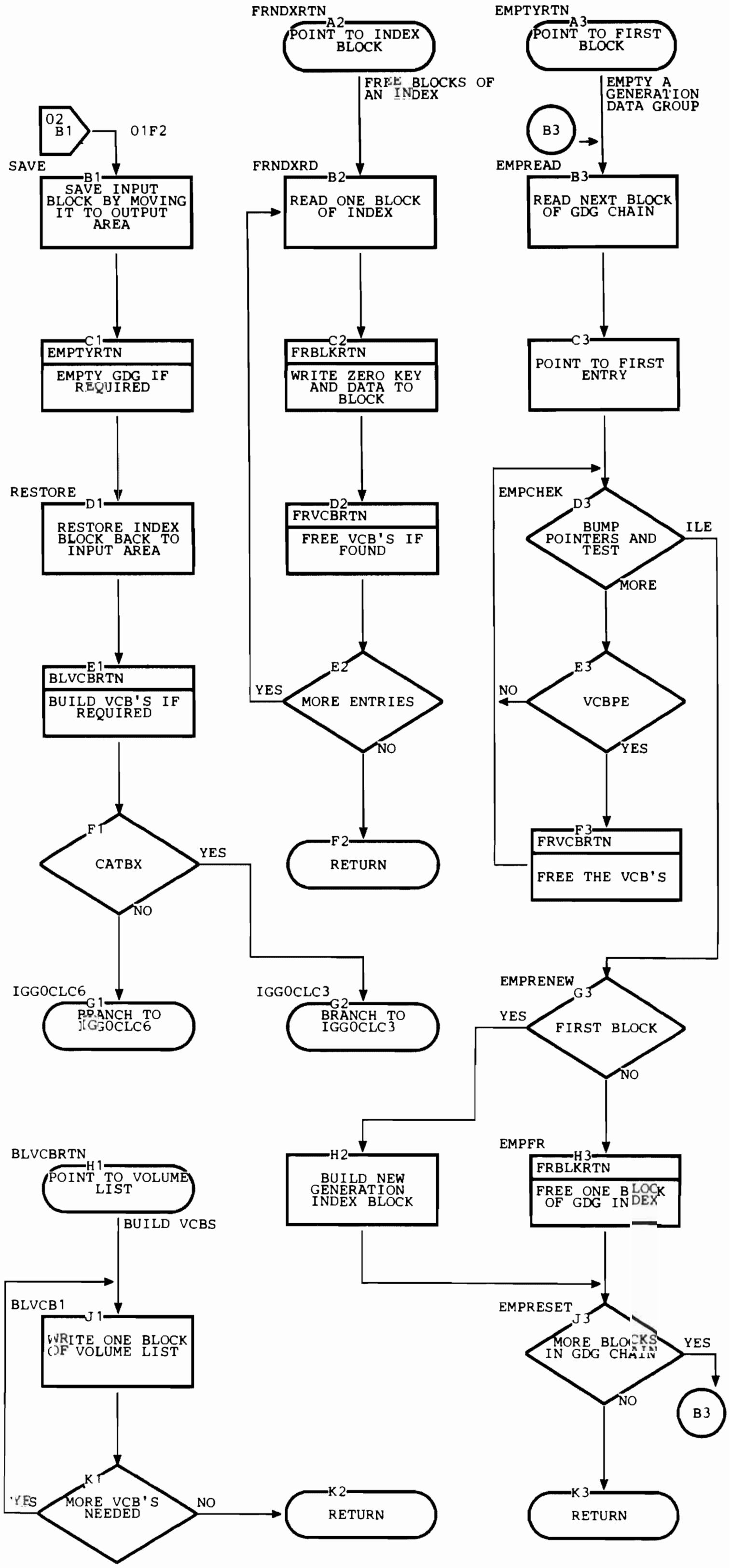
Control passes via a branch instruction to:

- IGG0CLC3 when the requested function is CATBX.
- IGG0CLC7 for error conditions.
- IGG0CLC6 for all other functions or conditions.

Control passes via a branch to IGC0002H when a new extent of the CVOL Catalog is required, and returns to this subroutine.

Error Conditions

Code	*Reason*
20	Not enough space available in the CVOL Catalog to perform the requested function.
28	Permanent I/O error.



IGG0CLC6: Second Load of Update

IGG0CLC6 is the entry point. Control comes from:

- IGG0CLC4 when the requested function is CAT, UNCAT, RECAT, or CATBX.
- IGG0CLC3 or IGG0CLC5 for all other requests or conditions.

Registers

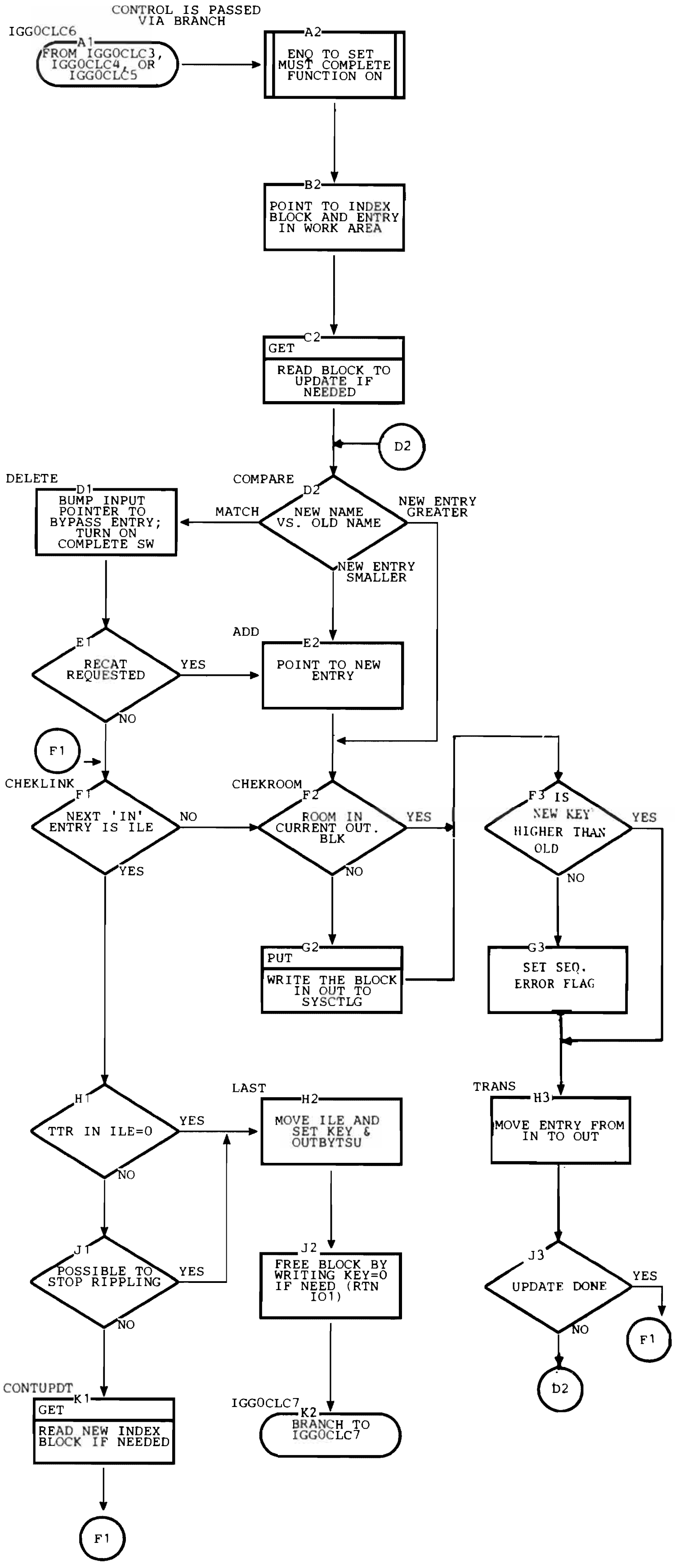
4	Base register for this subroutine
6	Base register for WORKAREA DSECT
8	Base register for CAMLSTD DSECT
12	Linkage register for BAL instructions
14	Linkage register for BAL instructions

Functions

This subroutine adds or deletes an entry to or from a given index block, as set up by earlier phases, and propagates (ripples) the change through the index chain as needed. Each entry is taken from the buffer INPUT and placed into the buffer OUTPUT until the collating sequence of the entry is equal to or greater than the name in the update request. If the request name is equal, that entry is skipped (delete function). If the request name is greater, the new entry is merged into OUTPUT (add function). Overflow entries become an add request for the next block in the chain.

Subroutines named GET and PUT are used for input/output. GET reads a block into INPUT, a field of WORKAREA, and initializes PUT. Entries are transferred from INPUT to OUTPUT, another field of WORKAREA. When all entries have been exhausted from INPUT, another block of the index is read from SYSCTLG.

When OUTPUT is full, a block is written to SYSCTLG from OUTPUT by the routine PUT. PUT checks all available records before writing the block and chooses the record of SYSCTLG that is most likely to result in contiguous blocks of one index. PUT tries to free any unneeded blocks; any unneeded block that PUT cannot free is later freed by GET.


CONTROL IS PASSED VIA BRANCH
IGG0CLC6
A1
FROM IGG0CLC3, IGG0CLC4, OR IGG0CLC5
A2
ENQ TO SET MUST COMPLETE FUNCTION ON
B2
POINT TO INDEX BLOCK AND ENTRY IN WORK AREA
C2
GET
READ BLOCK TO UPDATE IF NEEDED
D2
DELETE
D1
BUMP INPUT POINTER TO BYPASS ENTRY; TURN ON COMPLETE SW
COMPARE
D2
NEW NAME VS. OLD NAME
MATCH
NEW ENTRY GREATER
NEW ENTRY SMALLER
E1
RECAT REQUESTED
YES
NO
ADD
E2
POINT TO NEW ENTRY
F1
CHEKLINK
F1
NEXT 'IN' ENTRY IS ILE
NO
YES
CHEKROOM
F2
ROOM IN CURRENT OUT. BLK
YES
NO
F3
IS NEW KEY HIGHER THAN OLD
YES
NO
G2
PUT
WRITE THE BLOCK IN OUT TO SYSCTLG
G3
SET SEQ. ERROR FLAG
H1
TTR IN ILE=0
YES
NO
LAST
H2
MOVE ILE AND SET KEY & OUTBYTSU
TRANS
H3
MOVE ENTRY FROM IN TO OUT
J1
POSSIBLE TO STOP RIPPLING
YES
NO
J2
FREE BLOCK BY WRITING KEY=0 IF NEED (RTN IO1)
J3
UPDATE DONE
YES
NO
F1
D2
CONTUPDT
K1
GET
READ NEW INDEX BLOCK IF NEEDED
IGG0CLC7
K2
BRANCH TO IGG0CLC7
F1

Internal Subroutines

GET reads one block from an index in the CVOL Catalog.

PUT prepares and writes one block into an index in the CVOL Catalog.

TOABSL converts a relative track address to an absolute track address.

TORLTV converts an absolute track address to a relative track address.

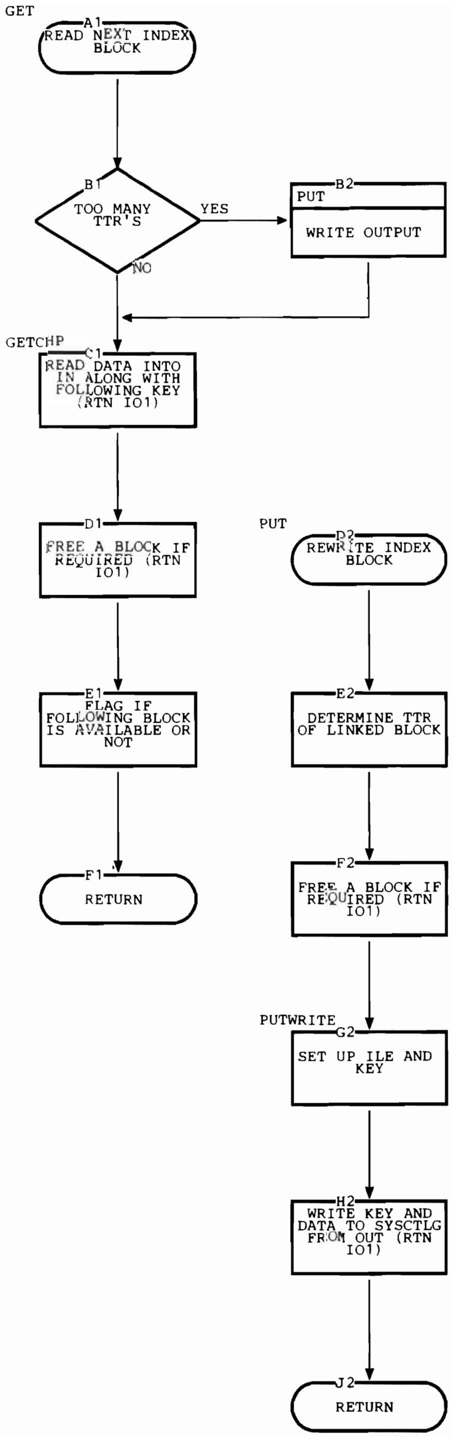
IO1 performs EXCP I/O operations.

Exits

Control is always passed to IGG0CLC7 via a branch instruction.

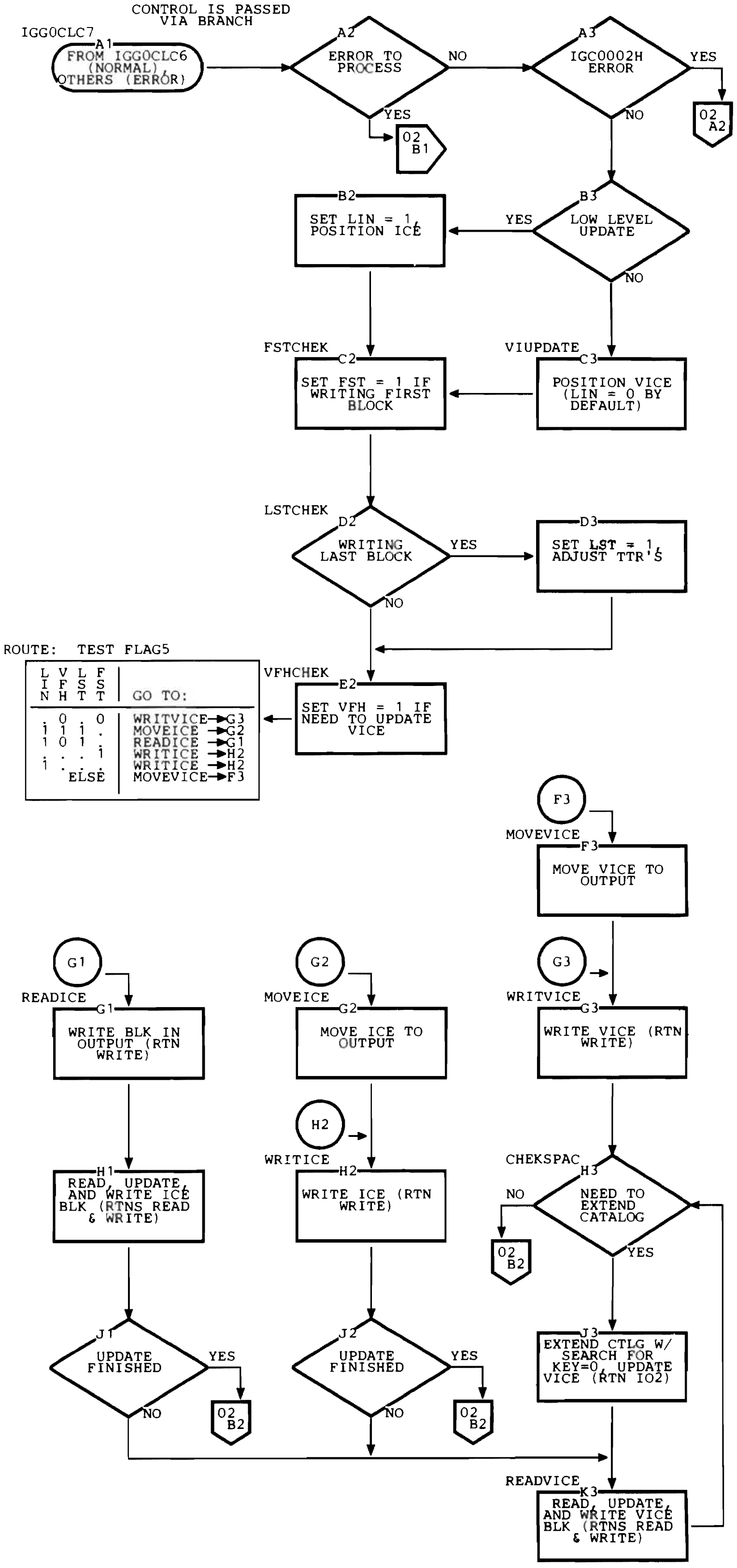
Error Conditions

The only exception code from this subroutine is 28, which indicates that a permanent input/output error has occurred.


GET
A1
READ NEXT INDEX BLOCK
B1
TOO MANY TTR'S
YES
NO
B2
PUT
WRITE OUTPUT
GETCHP
C1
READ DATA INTO IN ALONG WITH FOLLOWING KEY (RTN IO1)
D1
FREE A BLOCK IF REQUIRED (RTN IO1)
E1
FLAG IF FOLLOWING BLOCK IS AVAILABLE OR NOT
F1
RETURN
PUT
D2
REWRITE INDEX BLOCK
E2
DETERMINE TTR OF LINKED BLOCK
F2
FREE A BLOCK IF REQUIRED (RTN IO1)
PUTWRITE
G2
SET UP ILE AND KEY
H2
WRITE KEY AND DATA TO SYSCTLG FROM OUT (RTN IO1)
J2
RETURN

IGG0CLC7: Third Load of Update and Error Handling

IGG0CLC7 is the entry point. Control normally comes from IGG0CLC6, but can come from any subroutine of CVOL Catalog Management when an error condition is discovered.


CONTROL IS PASSED VIA BRANCH
IGG0CLC7
A1
FROM IGG0CLC6 (NORMAL) OTHERS (ERROR)
A2
ERROR TO PROCESS
NO
YES
02 B1
A3
IGC0002H ERROR
YES
02 A2
NO
B2
SET LIN = 1, POSITION ICE
B3
LOW LEVEL UPDATE
YES
NO
FSTCHEK
C2
SET FST = 1 IF WRITING FIRST BLOCK
VIUPDATE
C3
POSITION VICE (LIN = 0 BY DEFAULT)
LSTCHEK
D2
WRITING LAST BLOCK
YES
NO
D3
SET LST = 1, ADJUST TTR'S
VFHCHEK
E2
SET VFH = 1 IF NEED TO UPDATE VICE
F3
MOVEVICE
F3
MOVE VICE TO OUTPUT
G1
READICE
G1
WRITE BLK IN OUTPUT (RTN WRITE)
G2
MOVEICE
G2
MOVE ICE TO OUTPUT
G3
WRITVICE
G3
WRITE VICE (RTN WRITE)
H1
READ, UPDATE, AND WRITE ICE BLK (RTNS READ & WRITE)
H2
WRITICE
H2
WRITE ICE (RTN WRITE)
CHEKSPAC
H3
NEED TO EXTEND CATALOG
NO
02 B2
YES
J1
UPDATE FINISHED
YES
02 B2
NO
J2
UPDATE FINISHED
YES
02 B2
NO
J3
EXTEND CTLG W/ SEARCH FOR KEY=0, UPDATE VICE (RTN IO2)
READVICE
K3
READ, UPDATE, AND WRITE VICE BLK (RTNS READ & WRITE)

ROUTE: TEST FLAG5

LIN	VFH	LST	FST	GO TO:
.	0	.	0	WRITVICE→G3
1	1	1	.	MOVEICE →G2
1	0	1	.	READICE →G1
.	.	.	1	WRITICE →H2
1	.	.	.	WRITICE →H2
ELSE				MOVEVICE→F3

Registers

4 Base register for this subroutine
5 Pointer to SVRB extension
6 Base register for WORKAREA DSECT
8 Base register for CAMLSTD DSECT
12 Linkage register for BAL instructions
14 Linkage register for BAL instructions

On exit to the caller, the registers (except registers 0, 1, and 15) are restored by the supervisor.

Register 15 contains the exceptional return code. Registers 0 and 1 contain additional information that specifies the type of error encountered.

Functions

IGG0CLC7 completes the update process. The last block of an updated index is written to the CVOL Catalog.

The block containing the index control entry (ICE) is read, and the ICE is updated to reflect changes to the index. This block is rewritten to the CVOL Catalog.

The block containing the volume index control entry (VICE) is read, and the VICE is updated to reflect changes to the CVOL Catalog. This block is rewritten into the CVOL Catalog.

Tests are made before rewriting any block. If the block is both the last block of an index and the block containing the ICE, or the block containing the VICE, it is rewritten only once.

If an error is discovered, pertinent information is gathered from the WORKAREA and placed into an environment record and written to the CVOL Catalog. If the error is a sequence error, message IEC304I is written to the operator console. If the error is an I/O error on a non-locate operation, message IEC302I is written to the operator console. The exceptional return code is set and all resources are freed. Control returns to the caller of CVOL Catalog Management via a branch instruction.

Internal Subroutines

READ reads one block from the CVOL Catalog.

WRITE writes one block to the CVOL Catalog.

TOABSL converts a relative track address to an absolute track address.

TORLTV converts a relative track address to an absolute track address.

IO2 performs EXCP input/output operations. This subroutine invokes IGC0002H if a new extent of the CVOL Catalog is required.

Error Conditions

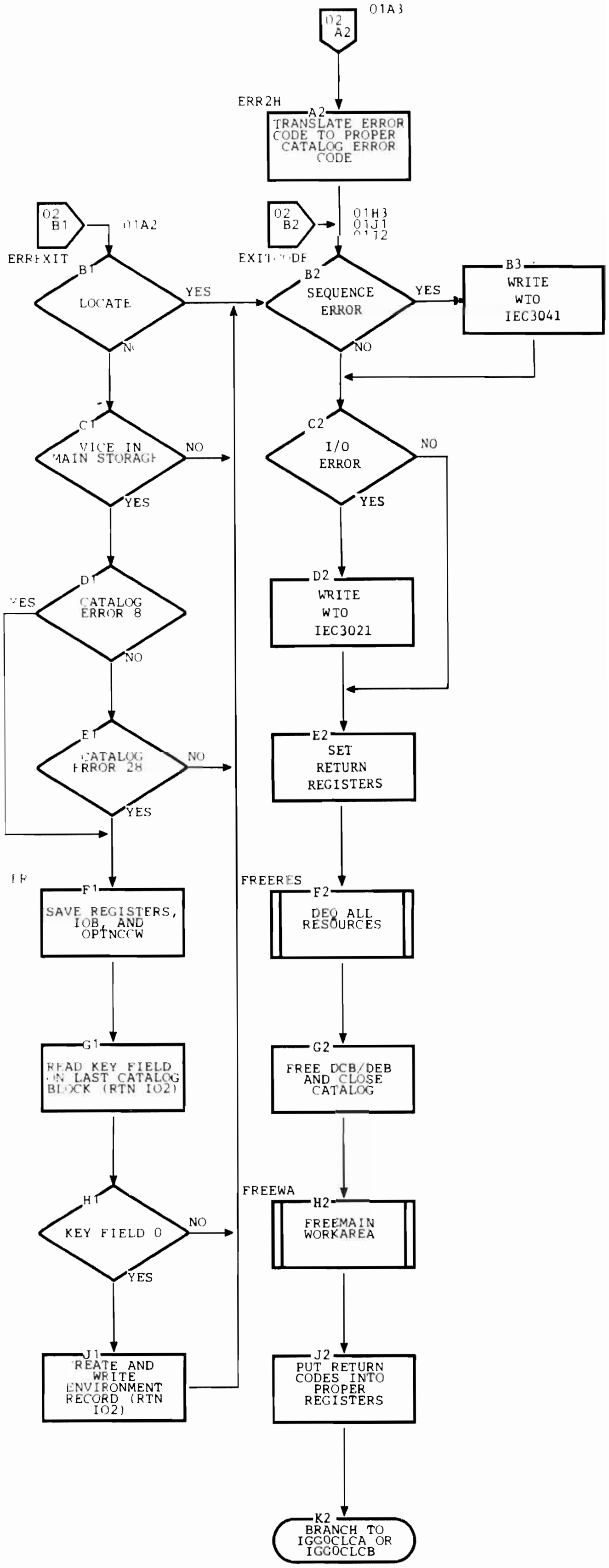
This subroutine returns any exception code from another CVOL Catalog Management CSECT to the caller. This exception code is passed to IGG0CLC7 in WORKAREA.

The only exception code from this subroutine is 28, which indicates that a permanent I/O error has occurred.

Exits

IGC0002H may be invoked via a branch when a new extent of the CVOL Catalog is required. Control returns to this subroutine when a new extent has been located.

Control returns to IGG0CLCA or IGG0CLCB via a branch instruction.



CSECT IGG0CLCF

IGC0002H: SYSCTLG Open/Extend

IGC0002H is the entry point. Control comes from:

- IGG0CLC0 or IGG0CLC3 to open a CVOL Catalog.
- IGG0CLC3, IGG0CLC5, or IGG0CLC7 to extend the CVOL Catalog.
- Control also comes via an XCTL macro instruction from IGG0553E after extending SYSCTLG.

Registers

On Entry for Opening:

0	Zero
1	Address of UCB for volume
8	Address of CAMLST
15	Address of area in which to build DCB/DEB chain

On Entry for Extending:

0	Address of DCB for the CVOL Catalog
8	Address of CAMLST

On Entry after Extending:

6	Address of SVRB
7	Address of Extend Work Area
8	Zero
9	Address of catalog DCB
10	UCB address

On Exit:

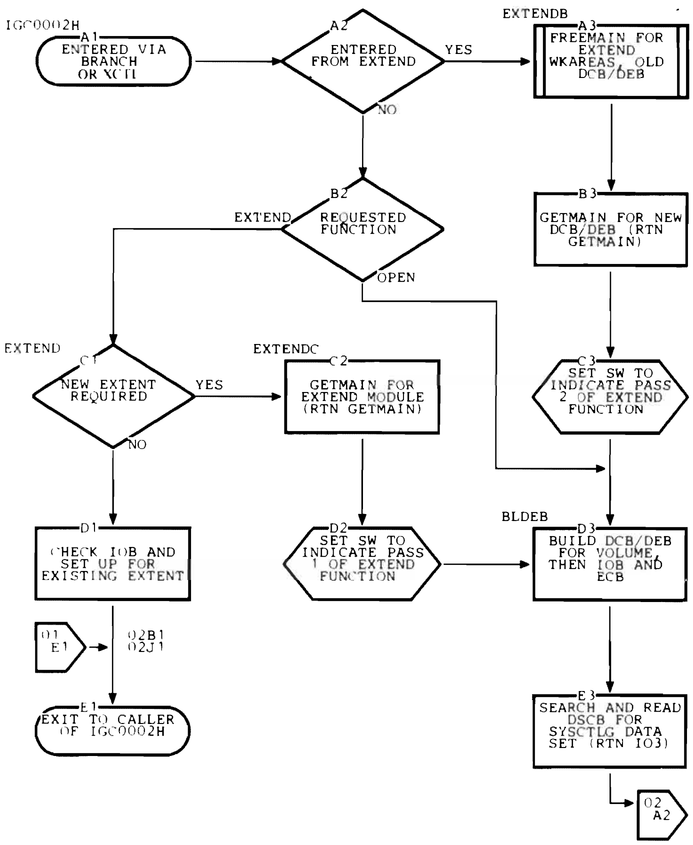
1	Address of DCB/DEB chain

Functions

When this subroutine is entered to open a CVOL Catalog, a data control block (DCB) and a data extent block (DEB) are built in the work area provided by IGG0CLC0. If the catalog is new, IGG0CLF2 is invoked to format it.

Note: The DCB/DEB constructed by this subroutine is a modification of that described in *OS/VS2 Data Areas.* These two blocks are merged together; that is, they overlap in the same area of main storage, as shown in Figure 3-9.

For SYSCTLG data sets that reside on MSS virtual volumes, an acquire for DASD space is issued.



IGC0002H
A1
ENTERED VIA BRANCH OR XCTL
A2
ENTERED FROM EXTEND
YES
NO
EXTENDB
A3
FREEMAIN FOR EXTEND WKAREAS, OLD DCB/DEB
B2
REQUESTED FUNCTION
EXTEND
OPEN
B3
GETMAIN FOR NEW DCB/DEB (RTN GETMAIN)
EXTEND
C1
NEW EXTENT REQUIRED
YES
NO
EXTENDC
C2
GETMAIN FOR EXTEND MODULE (RTN GETMAIN)
C3
SET SW TO INDICATE PASS 2 OF EXTEND FUNCTION
D1
CHECK IOB AND SET UP FOR EXISTING EXTENT
D2
SET SW TO INDICATE PASS 1 OF EXTEND FUNCTION
BLDEB
D3
BUILD DCB/DEB FOR VOLUME, THEN IOB AND ECB
01 E1
02B1
02J1
E1
EXIT TO CALLER OF IGC0002H
E3
SEARCH AND READ DSCB FOR SYSCTLG DATA SET (RTN IO3)
02 A2

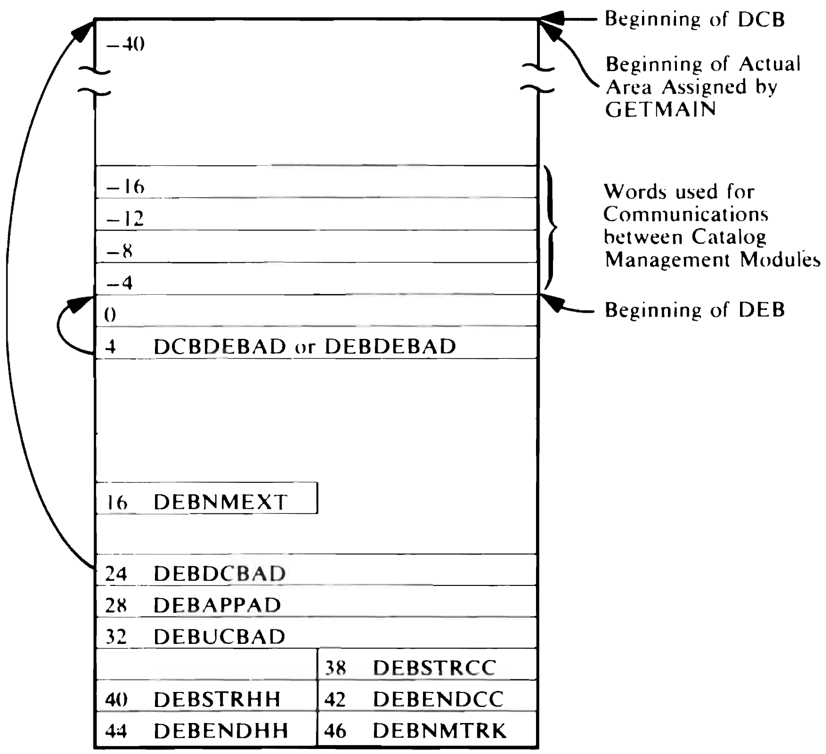


Figure 3-9. DCB/DEB Built by IGC0002H

When this subroutine is entered to cross to another extent of the CVOL Catalog, a test is made to see if another extent already exists. If so, WORKAREA is modified accordingly, and control returns to the caller.

When another extent does not exist, the virtual storage for the previous DCB/DEB is released and a new area is obtained with GETMAIN. IGG0553A is invoked to allocate a new extent and a new DCB/DEB is built into the new area (the catalog is reopened).

Main storage for the DCB/DEB is set to zeros before building; then only the fields that are shown are filled in. The DEB overlays the DCB at offset 40. The fields that are named are described in *OS/VS2 Data Areas.*

Internal Subroutines

GETMAIN gets main storage for the DCB/DEB.

IO performs EXCP input/output operations.

Exits

Control returns to the caller via a branch instruction when IGC0002H completes its function.

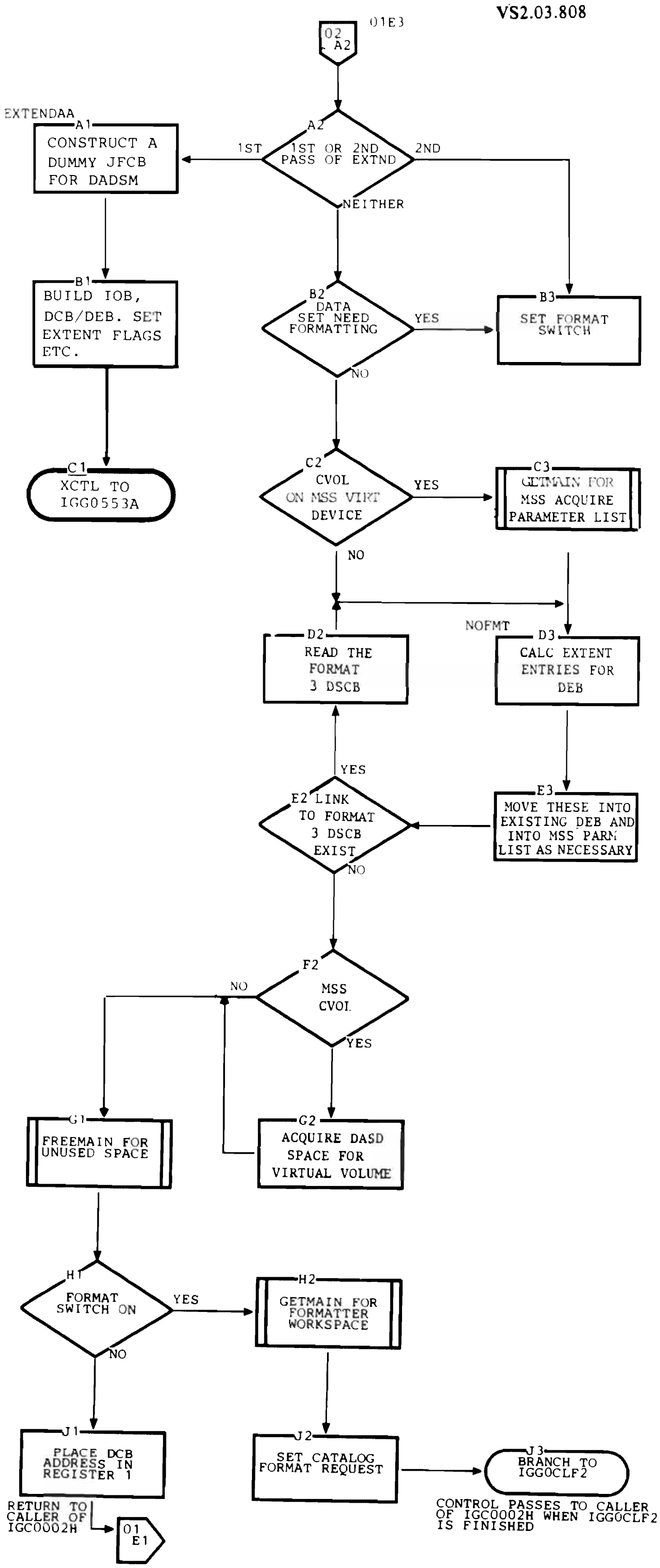
Control returns to the caller via a branch instruction for an error condition.

Control passes via XCTL to IGG0553A when another extent is required. Control returns via XCTL to entry point IGC0002H.

Control passes via a branch instruction to IGG0CLF2 when either the CVOL Catalog or a new extent needs to be formatted. Còntrol returns directly to the caller.

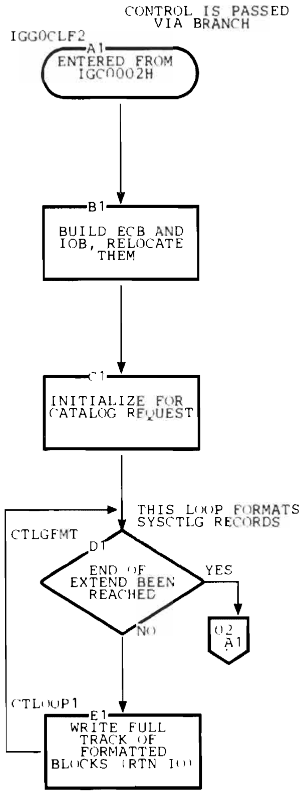
Error Conditions

Code	*Reason*
4	No extents are allocated or acquired.
8	No more extents are available.
12	Permanent I/O error.


01E3
02 A2
EXTENDAA
A1
CONSTRUCT A DUMMY JFCB FOR DADSM
1ST
A2
1ST OR 2ND PASS OF EXTND
2ND
NEITHER
B1
BUILD IOB, DCB/DEB. SET EXTENT FLAGS ETC.
B2
DATA SET NEED FORMATTING
YES
NO
B3
SET FORMAT SWITCH
C1
XCTL TO IGG0553A
C2
CVOL ON MSS VIRT DEVICE
YES
C3
GETMAIN FOR MSS ACQUIRE PARAMETER LIST
NO
NOFMT
D2
READ THE FORMAT 3 DSCB
D3
CALC EXTENT ENTRIES FOR DEB
YES
E2
LINK TO FORMAT 3 DSCB EXIST
NO
E3
MOVE THESE INTO EXISTING DEB AND INTO MSS PARM LIST AS NECESSARY
F2
MSS CVOL
NO
YES
G1
FREEMAIN FOR UNUSED SPACE
G2
ACQUIRE DASD SPACE FOR VIRTUAL VOLUME
H1
FORMAT SWITCH ON
YES
NO
H2
GETMAIN FOR FORMATTER WORKSPACE
J1
PLACE DCB ADDRESS IN REGISTER 1
RETURN TO CALLER OF IGC0002H
01 E1
J2
SET CATALOG FORMAT REQUEST
J3
BRANCH TO IGG0CLF2
CONTROL PASSES TO CALLER OF IGC0002H WHEN IGG0CLF2 IS FINISHED

IGG0CLF2: SYSCTLG Formatter

IGG0CLF2 is the entry point. Control comes from IGC0002H.


CONTROL IS PASSED VIA BRANCH
IGG0CLF2
A1
ENTERED FROM IGC0002H
B1
BUILD ECB AND IOB, RELOCATE THEM
C1
INITIALIZE FOR CATALOG REQUEST
THIS LOOP FORMATS SYSCTLG RECORDS
CTLGFMT
D1
END OF EXTEND BEEN REACHED
YES
NO
02 A1
CTLOOP1
E1
WRITE FULL TRACK OF FORMATTED BLOCKS (RTN IO)

Registers

0 Contains zeros when formatting the CVOL Catalog

1 Address of DCB for this data set

2 Number of blocks per track for this device

3 Number of bytes in work area passed to IGG0CLF2

5 Data management count decrement value

6 Starting relative track address (TTR) when formatting the CVOL Catalog

7 Address of work area

Functions

The data set is formatted into 256-byte blocks with 8-byte keys.

If the extent is being formatted during an open CVOL Catalog request, this is the first extent of a new CVOL Catalog. The first block is initialized by writing a volume index control entry (VICE) into it.

If formatting is not being done for the first extent, this is a new extent of an already existing CVOL Catalog. The VICE is read, updated, and rewritten to reflect the new extent.

The work area that is passed to IGG0CLF2 is freed before exit.

Internal Subroutines

CNVT converts a relative track address to an absolute track address.

IO performs EXCP input/output operations.

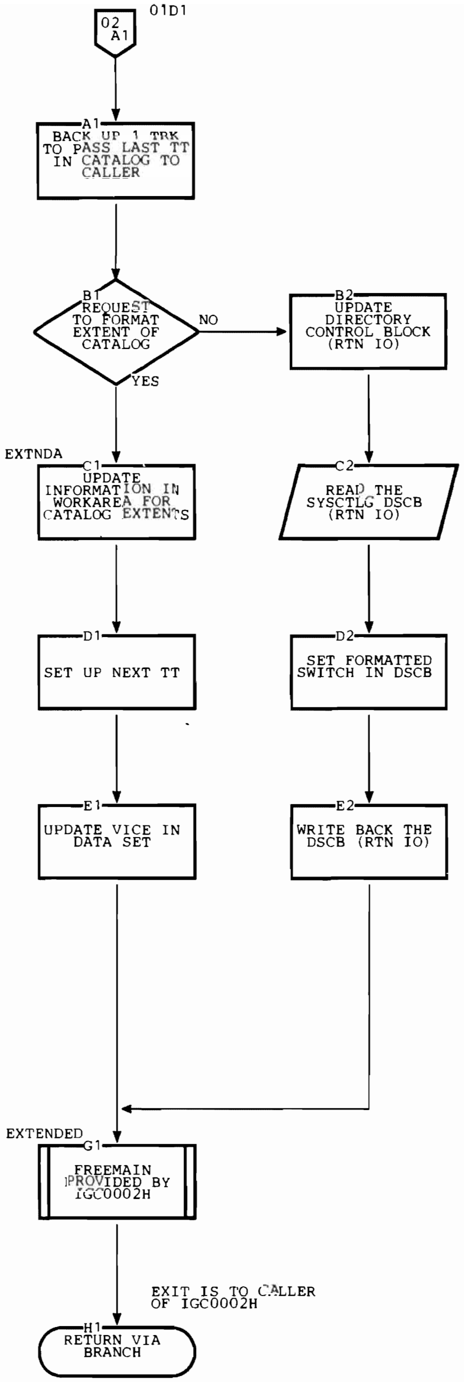
RELOC builds channel programs for input/output.

Error Conditions

IGG0CLF2 returns one exception code, 12, which indicates that an I/O error has occurred. The caller of CVOL Catalog Management never sees this code.

Exits

Control is passed to the caller of IGC0002H via a branch instruction.



01D1
02 A1
A1 BACK UP 1 TRK TO PASS LAST TT IN CATALOG TO CALLER
B1 REQUEST TO FORMAT EXTENT OF CATALOG
NO
YES
B2 UPDATE DIRECTORY CONTROL BLOCK (RTN IO)
EXTNDA
C1 UPDATE INFORMATION IN WORKAREA FOR CATALOG EXTENTS
C2 READ THE SYSCTLG DSCB (RTN IO)
D1 SET UP NEXT TT
D2 SET FORMATTED SWITCH IN DSCB
E1 UPDATE VICE IN DATA SET
E2 WRITE BACK THE DSCB (RTN IO)
EXTENDED
G1 FREEMAIN PROVIDED BY IGC0002H
EXIT IS TO CALLER OF IGC0002H
H1 RETURN VIA BRANCH

This chapter contains a directory to the microfiche listings for all the CSECTs and subroutines used by the CVOL Processor. This directory describes the contents of each CSECT and allows you to quickly find any desired code.

CSECT IGG0CLCA is written in PL/S-2, a high-level, proprietary system language. Listings produced for microfiche consist of the PL/S-2 source code, a cross-reference and attribute table, and the assembly code. See the IBM publication *Guide to PL/S II* for a more detailed explanation of PL/S and its listings.

CSECTs IGG0CLCB, IGG0CLCC, IGG0CLCD, IGG0CLCE, and IGG0CLCF use Assembler language. Listings produced for microfiche consist of the Assembler source code, a cross-reference and attribute table, and the assembly code. For more information on Assembler language, see the IBM publications *OS/VS-DOS/VS-VM/370 Assembler Language* and *OS/VS-VM/370 Assembler Programmer's Guide.*

Note: The listings use CPL, FVT, and FPL instead of CTGPL, CTGFV, and CTGFL, respectively. See *OS/VS2 Catalog Management Logic* for a description of these data areas.

In the following tables, the CSECT name appears in the first (leftmost) column. The second column contains an entry-point label or a subroutine label (internal procedure). The third column differentiates between entry points (EP) and procedures (PR). The fourth column describes the subroutine. For more information on the CSECTs and subroutines, refer to the following chapters in this publication: "Method of Operation" (Chapter 2), and "Program Organization" (Chapter 3).

CSECT	Subroutine	Use	Description
IGG0CLCA			CVOL Sharing Interface Mapper CSECT 1. This module is the first of two CSECTs that map VSAM and OS catalog functions to CVOL Catalog Management.
	IGG0CLCA	EP	Main entry point for this CSECT.
	IGG0CL1A	EP	Dynamically allocates a CVOL Catalog.
	DELETE	PR	Processes a VSAM-like delete request. This is accomplished by issuing an OS UCATDX request and optionally a SCRATCH SVC.
	DSCBTTR	PR	Processes a 'DSCBTTR' CTGFL.
	DSTYPNAM	PR	Processes a 'DSTYPNAM' CTGFL.
	ENTNAME	PR	Processes an 'ENTNAME' CTGFL.
	ENTYPE	PR	Processes an 'ENTYPE' CTGFL.
	ESTAEXIT	PR	Processes an ESTAE intercepted abend.
	FPLMV	PR	Processes the following repeating field CTGFLs: DEVTYP, VOLSER, FILESEQ, and CATVOL.
	GENLOC	PR	Processes a VSAM-like generic locate. Most of the processing is done by the second CSECT of the Interface Mapper (IGG0CLCB).
	GETSVCK	PR	Changes storage key via MODESET macro from user key to SVC key.
	GETUSERK	PR	Changes storage key via MODESET macro from SVC key to user key.
	LOCNAME	PR	Issues an OS LOCATE by NAME request.
	LOCTTR	PR	Issues an OS LOCATE by TTR request.
	OSREQ	PR	Sets up and executes an original OS CAMLST format request.
	RESCAN	PR	Searches the CVOL Catalog and determines if the specified data set is a generation data group type. If a generation index pointer entry (GIPE) is found, the GIPEPTR contains the address of the GIPE. Otherwise, the GIPEPTR contains zeros to indicate the absence of a GIPE.
	SLGDG	PR	Processes a SUPERLOCATE generation data group request with the base generation number supplied.
	SLGDGB	PR	Processes a SUPERLOCATE generation data group request to return the generation data group base value.
	SLGDGBL	PR	Searches for a new absolute generation number if the supplied relative generation number is less than zero.
	SLNAME	PR	Processes a normal superlocate request or a GDG ALL request.
	SLVOLST	PR	Fills the user's volume list area with volume serial numbers, device types, and file sequence numbers.
	SRCHPCCB	PR	Searches the PCCB (Private Catalog Control Block) chain to see if a PCCB for the needed catalog is already on the chain. If it is, the PCCBPTR points to it. If there is no PCCB, the PCCBPTR is zeroed.
	SUPERLOC	PR	Determines type of superlocate request and calls the appropriate procedure: SLGDG, base generation number supplied; SLGDGB, base only requested; or SLNAME, normal SUPERLOCATE.
	VLOC	PR	Processes a VSAM LOCATE or an Access Method Services LISTCAT.

CSECT	Subroutine	Use	Description
IGG0CLCB			This is the main processing module for Generic Locate. It searches the SYSCTLG data set using CVOL Catalog Management LOCATE and returns the names of all data sets that are found to have the requested high level qualifiers as the first part of the data set name.
	IGG0CLCB	EP	Only entry point for this CSECT.
	CIR	PR	This subroutine is a modified version of the OS/VS2 Release 1 module IKJEHCIR, which was used for the TSO LISTCAT command. CIR does the actual locates and builds the lists of qualifiers to be processed for CSECT IGG0CLCB.
	CODE00	PR	This subroutine gets control if LOCATE passes a return code of zero.
	DSNAMRT	PR	This subroutine gets control when a data set entry is found in the list from CIR. DSNAMRT checks to determine if a generation data group is being processed. If so, the generation portion of the simple name must be complemented.
	GDGROUT	PR	This subroutine is entered if a generation data group entry is found in the CIR list. It turns on the GDGSW switch so that the data set name entry routine (DSNAMRT) will know that the generation number in the simple name will need complementing. A check is made to see if any generations exist. If not, this entry is skipped. If any generations exist, the count of the number of generations cataloged is kept and decremented each time a generation name is processed. Control is passed to the index entry routine (INDEXRT) to read in the list of names through CIR. Register 6 points to the current entry.
	INDEXRT	PR	This subroutine gets control when an index entry is discovered in the list from CIR. It sets up a parameter list for CIR and uses subroutine OBTBLK to allocate another block for a new list of lower qualifiers. The new list is made current, and CURNTBLK points to the current CIR list.
	MAIN00	PR	This subroutine checks for a null list and returns to the caller with a return code of 4 if the index structure specified or the USERID had no data sets cataloged under it.
	MAIN01	PR	Entry is made to this subroutine with register 6 pointing to a new list element or entry. The list entry is analyzed and the appropriate routine is used to process it. Data set name entries are used to complete fully qualified data set names, and are returned in the caller's output area.
	OBTBLK	PR	This subroutine is used to obtain a new block to be used as a work area for CIR and to become the current block. If no free blocks are available, a conditional GETMAIN is issued and the new block is added on the chain. If the GETMAIN fails, control is returned to caller with a return code of 8.
	POINTER	PR	This subroutine updates the current entry pointer in the current block. The current block is determined by searching the chain for the first block with a zero entry pointer and then backing up one. The current entry type is determined, and the pointer is advanced accordingly. If the next entry is a link entry which contains a non-zero TTR, the CIR is called to provide the next block of entries. If the TTR is zero, the current block is released and the preceding block is considered. When all blocks are processed, that is, the current block equals the first block and the empty block equals zero, the WRAPUP routine is entered.

CSECT	Subroutine	Use	Description
	VCBROUT	PR	This subroutine is given control when a volume control block (VCB) entry is found in the list from CIR. A check is made to determine if a generation data group is being processed. If so, the generation portion of the simple name must be complemented. If there is no generation data group, the simple name is not complemented.
	WRAPUP	PR	This subroutine gets control when processing for IGG0CLCB is completed or an error resulting in termination occurs. It frees all the dynamic core obtained for IGG0CLCB.

CSECT	Subroutine	Use	Description
IGG0CLCC			CSECT IGG0CLCC performs the read operation for CVOL Catalog Management. It performs the locate functions and the locating part of the non-locate functions.
	IGG0CLCC	EP	Only entry point for CSECT IGG0CLCC.
	IGG0CLC0	PR	This subroutine initializes the work areas, opens the given CVOL Catalog, and searches for high level names.
	IGG0CLC1	PR	This subroutine resolves aliases, constructs BLDA or LNKX entries, and processes relative generation data groups.
	IGG0CLC2	PR	This subroutine searches lower levels of the name, saves last valid index levels, and relocates CCWs for use by CSECTs IGG0CLCD and IGG0CLCE.
	IECBLDL	PR	This subroutine searches for the qualified name in the CVOL Catalog.
	RACHK	PR	Performs RACF authorization checking via RACHECK macro for UNCATLG, RECATLG, DRPX, and CATLG-GDG requests.
IGG0CLCD			CSECT IGG0CLCD performs the setup operation. It checks the validity of the requests against the existing entries in the CVOL Catalog. It builds new entries to be added to the catalog, or it names entries to be deleted.
	IGG0CLCD	EP	Only entry point for CSECT IGG0CLCD.
	IGG0CLC3	PR	This subroutine ensures that VICE, ICE, and space are present. It constructs and writes new index blocks, and routes non-locate requests.
	IGG0CLC4	PR	This subroutine constructs new DSPEs or VCBPEs. It scratches generation data groups if requested. The EMPTY option for generation data groups allows the existing generations to be scratched before adding new ones.
	IGG0CLC5	PR	This subroutine frees index blocks, frees volume control blocks, and writes new volume control blocks. It also performs the EMPTY option as requested.
IGG0CLCE			CSECT IGG0CLCE performs the write operation. It merges entries into SYSCTLG blocks, deletes entries from blocks, and does most of the writing that is needed.
	IGG0CLCE	EP	Only entry point for CSECT IGG0CLCE.
	IGG0CLC6	PR	This subroutine updates blocks, writes updated blocks to SYSCTLG, and ripples a change as needed to the last block of the updated chain.
	IGG0CLC7	PR	This subroutine writes the last updated block, updates the control entries, returns control to CSECT IGG0CLCA or IGG0CLCB, whichever called CSECT IGG0CLCC. This subroutine also handles all error conditions for CSECTs IGG0CLCC, IGG0CLCD, and IGG0CLCE.
IGG0CLCF			CSECT IGG0CLCF performs three functions: it opens CVOLs, extends CVOLs, and formats new extents.
	IGG0CLCF	EP	Main entry point for CSECT IGG0CLCF.
	IGC0002H	PR	This subroutine opens the SYSCTLG data set and gets the next extent of that data set. For SYSCTLG data sets which reside on MSS virtual volumes, it acquires the DASD space using SVC 26.
	IGG0CLF2	PR	This subroutine formats new extents of a catalog.

The data areas and record formats in this chapter are described in four columns, which are interpreted as follows:

Offset

The numeric address of the field relative to the beginning of the area. The first number is the offset in decimal, followed (in parentheses) by the hexadecimal equivalent.

Bytes and Alignment

The size (number of bytes) of the field and its alignment relative to the fullword boundary.

Examples:

4 A 4-byte field beginning on a word boundary.

. . 3 A 3-byte field beginning on a halfword boundary and running into the next word.

. . . 2 A 2-byte field beginning at the low-order byte of a word and running into the next word.

Name and Content

A name that identifies the field. This name appears as a label in the assembly listings.

This column is also used to show the contents of the field or the bit settings of flag fields (the state of bits in a byte). When the column is used to show the state of the bits (0 or 1) in a flag byte, it is shown as follows:

.... The 8 bit positions (0-7) in a byte. For ease of scanning, the high-order (leftmost) 4 bits are separated from the low-order 4 bits.

x... .... A reference to bit 0.

1... .... Bit 0 is on.

0... .... Bit 0 is off.

.... ..xx A reference to bits 6 and 7.

Bit settings that are significant are shown and described. Bit settings that are not presently shown are understood to be reserved bits.

Field Description and Meaning

The use of the field.

SYSCTLG Entry Formats

This section describes the formats of the entries of SYSCTLG, along with the symbolic labels that are used to refer to their fields. The entries are arranged alphabetically.

Except for the volume control block (VCB), SYSCTLG entries have a similar format. These entries share a common definition for the first 12 bytes. The shared names are:

ENAME	ETTR	ETYPE
(8 bytes)	(3 bytes)	(1 byte)

Individually named fields follow either ETTR or ETYPE.

The entries in a SYSCTLG block begin in the third byte of the block. The first halfword of the block contains the binary number of the bytes that are used in this block, including the halfword count field.

Alias Entry (AE)

An alias entry defines an alternate name for the high-level qualifier of a data set name.

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	8	ENAME	Name; contains the alias of the high-level index whose relative track address is found at offset 8 of this entry.
8(8)	3	ETTR	Address; contains the relative track address (TTR) of the first block of the index named at offset 12 of this entry.
11(B)	...1	ETYPE X'04'	Type; indicates that this is an alias entry; also that four halfwords follow in the remainder of the entry.
12(C)	8	ETRUEN	True name; contains the name of the index whose alias appears at the beginning of this entry.

Control Volume Pointer Entry (CVPE)

A control volume pointer entry can appear only in volume indexes. Two forms are possible: the old form, created prior to Release 17 of IBM System/360 Operating System, and the new form, created since that release. Both forms are shown here.

Old CVOL Pointer Entry

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	8	ENAME	Name field; contains a high-level name that appears in the volume index of the control volume identified at offset 12 of this entry.
8(8)	3	ETTR X'000000'	Zero field.
11(B)	..1	ETYPE X'03'	Type; indicates that this is either an old CVOL pointer entry (CVPE), or an index control entry (ICE). An ICE always appears as the first record of an index level; a CVOL pointer entry always appears in the volume index. This is also the number of halfwords that follow in the remainder of the entry.
12(C)	6	EVOLIDO	Serial number of the control volume whose volume index contains an entry for the name found at the beginning of this entry.

New CVOL Pointer Entry

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	8	ENAME	Name; contains a high-level name that appears in the volume index of the control volume identified at offset 12 of this entry.
8(8)	3	ETTR X'000000'	Zero field.
11(B)	..1	ETYPE X'05'	Type; indicates that this is a new CVOL pointer entry (CVPE) or the volume index control entry (VICE). The VICE always appears as the first entry in the first block of SYSCTLG; a CVOL pointer never appears as the first entry of the first block. Also indicates that five halfwords follow in the remainder of the entry.
12(C)	4	EDEVTYP	Control volume device type; contains the binary device code of the control volume whose volume index contains an entry for the name found at the beginning of this entry.
16(10)	6	EVOLID	Serial number of the control volume whose volume index contains an entry for the name found at the beginning of this entry.

Data Set Pointer Entry (DSPE)

A data set pointer entry can appear in any index level. It contains the simple name of a data set and from one to five 12-byte fields, each of which identifies a volume on which the named data set resides.

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	8	ENAME	Name; contains the simple name of the data set whose volumes are identified at offset 12 of this entry.
8(8)	3	EDSCBTTR	Address; contains either binary zero or, when the data set resides on only one volume, the relative track address (TTR) of the data set control block (DSCB) for this data set in the volume table of contents (VTOC).
11(B)	...1	ETYPE X'07' X'0D' X'13' X'19' X'1F'	Type; indicates that this is a data set pointer entry (DSPE). Also indicates the number of halfwords that follow in the remainder of this entry.
12(C)	2	EVOLCNT	Volume count; contains the binary count of the number of volumes identified beginning at offset 14.
14(E)	..12 to 60	EDATA	Volume entries; contains from one to five 12-byte entries, each of which identifies one volume on which the data set resides. Catalog management neither uses nor checks the contents of this field.

Generation Index Pointer Entry (GIPE)

A generation index pointer entry can appear in any index except a generation index. It corresponds to the simple name used in the relative name for a GDG data set.

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	8	ENAME	Name; contains the name of the generation index to which this entry points.
8(8)	3	ETTR	Address; contains the relative track address of the first block of the generation index named in this entry, in the form TTR.
11(B)	...1	ETYPE X'02'	Type; indicates that this is a generation index pointer entry (GIPE). Also indicates that two halfwords follow in the remainder of this entry.
12(C)	1	EGFLAGS	Flags; contains the options specified by the creator of the generation data group:
		.... ..1.	DELETE option.
		1	EMPTY option.
13(D)	.1	EGMAXSIZ	Maximum count; contains a binary number specifying the maximum number of generations allowed in the generation index at one time.
14(E)	..2	EGCURSIZ	Current generation count; contains the binary number of generations currently cataloged in the index.

Index Control Entry (ICE)

The index control entry is the first entry in all indexes except the volume index.

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	8	INAME X'00...01'	Name; low value of binary 1 ensures that this is the first entry in the index.
8(8)	3	ILSTBLK	Last block address; contains the relative track address of the last block assigned to the index, in the form TTR.
11(B)	...1	ITYPE X'03'	Type; indicates that this is either an ICE or an old CVOL pointer. An ICE always appears as the first entry of an index; an old CVOL pointer always appears in the volume index. Also indicates the number of halfwords that follow in the remainder of the entry.
12(C)	3	IFSTBLK	First block address; contains the relative address of the block in which this entry appears, in the form TTR.
15(F)	...1	ILIASCNT	Number of aliases; contains a binary count of aliases assigned to the index. This count is always zero for indexes that are not high-level. An index cannot be deleted if this count is non-zero.
16(10)	2		Reserved.

Index Link Entry (ILE)

An index link entry is always the last entry in any index block. It is used to link blocks of one index into a chain.

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	8	ENAME X'FF...FF'	Name; high value (all bits on) ensures that this is the last entry in the index.
8(8)	3	ETTR	Link address; contains the relative track address of the next block of the same index, if there is one, in the form TTR. When this is the last (or only) block, this field contains binary zero.
11(B)	...1	ETYPE X'00'	Type; indicates that this is either an ILE or an IPE. The name field of an ILE always contains X'FFFFFFFFFFFFFFFF'; the name field of an IPE never does. Also indicates that there are no more halfwords in the entry.

Index Pointer Entry (IPE)

The index pointer entry can appear in any index except a generation index. It points to a lower index.

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	8	ENAME	Name; contains the name of the index to which this entry points.
8(8)	3	ETTR	Index address; contains the relative track address of the first block of the index named in this entry, in the form TTR.
11(B)	...1	ETYPE X'00'	Type; indicates that this is either an IPE or an ILE. The name field of an ILE always contains X'FFFFFFFFFFFFFFFF'; the name field of an IPE never does. Also indicates that there are no more bytes in the entry.

Volume Control Block (VCB)

A volume list can be recorded in one or more volume control blocks. Each volume control block is one block of the SYSCTLG data set, and can identify up to 20 volumes on which one data set is recorded.

Note: This block is different from other blocks of SYSCTLG. The first halfword does not contain the number of bytes used in the block as do other SYSCTLG blocks. The field VCBVOLCT, shown below, is the first halfword of the VCB block.

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	2	VCBVOLCT	Number of volumes; contains the number of volumes identified in this and subsequent volume control blocks. This number is reduced by 20 for each subsequent volume control block. For example, if a data set resides on 61 volumes, it uses four volume control blocks. This field of each block contains 61, 41, 21, and 1, respectively.
2(2)	..12 to 240	VCBVOLS	Volume identifications; contains from 1 to 20 12-byte entries, each of which identifies one of the volumes on which the data set resides. Catalog management neither uses nor inspects the content of these entries. Each 12-byte entry contains a 4-byte device code, a 6-byte volume serial number, and a 2-byte data set sequence number.
242(F2)	..10	X'00...00'	Zero field.
252(FC)	3		Chain address, contains the relative track address of the next volume control block, if there is one, in the form TTR. If this is the last (or only) block of the volume control block, this field contains binary zero.
255	1	X'00'	Zero field.

Volume Control Block Pointer Entry (VCBPE)

A volume control block pointer entry can appear in any index. It is used when a data set resides on more than five volumes.

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	8	ENAME	Name; contains the simple name of the data set whose volumes are identified in the volume control block that is pointed to by this entry.
8(8)	3	ETTR	Address; contains the relative track address of the volume control block identifying the volumes containing the data set named in this entry, in the form TTR.
11(B)	...1	ETYPE X'01'	Type; indicates that this is a volume control block pointer entry. Also indicates that one halfword follows in the remainder of this entry.
12(C)	2	X'0000'	Zero field.

Volume Index Control Entry (VICE)

The volume index control entry is always the first entry in the first block of data set SYSCTLG.

It is the control record for the entire data set, and acts as an ICE for the volume index.

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	8	VNAME X'00...01'	Name; always contains a binary one to ensure that this is the first entry of the volume index.
8(8)	3	VLSTBLK	Last block address; contains the relative track address of the last block of the volume index, in the form TTR.
11(B)	...1	VTYPE X'05'	Type; indicates that this is the volume index control entry or a new CVOL pointer entry. The volume index control entry is always the first entry of the first block of SYSCTLG; a CVOL pointer is never the first entry. Also indicates that five halfwords follow in the remainder of the entry.
12(C)	3	VCLSTBLK	Last block of the catalog; contains the relative track address of the last block in SYSCTLG, in the form TTR.
14(E)	..1	VHIREC	Contains the number of TTRs in VCLSTBLK. Note that this field is the last byte of VCLSTBLK (offset 12).
15(F)	...1	X'00'	Zero field.
16(10)	3	VFHOLE	First available block; contains the relative track address of the first unused block in SYSCTLG, in the form TTR.
19(13)	...1	X'00'	Zero field.
20(14)	2		Reserved.

Environment Record (EREC DSECT)

The environment record is written by module IGG0CLC7 under certain error conditions. This record is useful in diagnosing problems using the catalog management routines. Reading the environment record is described in the chapter, "Diagnostic Aids."

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	8		Reserved.
8(8)	8	ERTIME	Time stamp, as produced by the TIME macro instruction.
16(10)	4	ERCAMLST	First four bytes of the caller's parameter list produced by the CAMLST macro instruction.
20(14)	1	ERMODMAP	Field MODMAP1 from WORKAREA.
21(15)	.1	ERFLAG1	Field FLAG1 from WORKAREA.
22(16)	..1	ERFLAG2	Field FLAG2 from WORKAREA.
23(17)	...1	ERFLAG3	Field FLAG3 from WORKAREA.
24(18)	2	ERERRCOD	Fields ERRCATSV and ERRLOCSV from WORKAREA.
26(1A)	..14	ERNAMTTR	Level name, TTR, type, and volcnt; the first 14 bytes of a general entry.
40(28)	60	ERREGSV	Contents of general registers 0 through 14 at the time the environment record is written (register 15 is destroyed by module IGG0CLC7).
100(64)	28	ERWA1	Contents of WORKAREA from offset 12 bytes (label TTR) through offset 39 bytes.

(cont.)

(cont.)

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
128(80)	18	ERINPUT	First entry in INPUT.
146(92)	..18	EROUTPUT	First entry in OUTPUT.
164(A4)	8	EROPTNCC	Field OPTNCCW from WORKAREA.
176(B4)	40	ERIOB	Field IOB from WORKAREA.
212(D4)	44	ERNAME	Fully qualified name provided by the caller.

RPSD DSECT

RPSD describes the CCW chain used for rotational position sensing (RPS) support.

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	16	RPSCCW	Two double-words: RPSSS and RPSTIC.
0(0)	8	RPSSS	Set sector CCW.
8(8)	8	RPSTIC	TICs to normal channel program.
16(10)	16	RPSINPUT	Four words: RPSCNVT, RPSDDKR, RPSR1, and RPSPTR.
16(10)	4	RPSCNVT	Address of supervisor routine to convert sector value.
20(14)	4	RPSDDKR	Block size (DD, 256 bytes), key length (K, 8 bytes), and record number.
24(18)	4	RPSR1	Address of location of this DSECT during use.
28(1C)	4	RPSPTR	Type and address: the first byte contains the device type code, and the last three bytes contain the sector value.
32(20)	40	RPSAVE	10-word register save area.

Note: The listings use CPL, FVT, and FPL instead of CTGPL, CTGFV, and CTGFL, respectively. Please see *OS/VS2 Catalog Management Logic* for a description of these data areas, as well as PCCBs.

WORKCLCA Work Area

Controller III creates WORKCLCA. The CVOL Processor gains control via an XCTL with register 12 pointing to WORKCLCA. For more information on WORKCLCA at processor invocation, see Figure 3-1.

Offset	Bytes and Bit Pattern	Field Name	Description: Content, Meaning, Use
0 (0)	4	*	Reserved.
4 (4)	44	WKCATNM	Name of the non-VSAM entry that defines the CVOL Catalog in the VSAM Master Catalog.
48 (30)	44	WKCATANM	Alias name in the VSAM Master Catalog that is related to WKCATNM.
92 (5C)	6	WKCVOLVS	Volume serial number of CVOL Catalog.
98 (62)	..2	*	Reserved.
100 (64)	4	CVTPTR	Address of CVT.
104 (68)	4	TCBPTR	Address of TCB.
108 (6C)	4	SVRBSAV	Address of SVRB.
112 (70)	4	VSRC15	VSAM register 15 return code.
116 (74)	4	REG13SAV	CVOL Catalog Management register 13 save area address.
120 (78)	4	LIMIT	Limit of DO Loop.
124 (7C)	4	EXITSAV	Address of Exit Prologue.
128 (80)	4	CTGPLPTR	Address of VSAM CTGPL.
132 (84)	4	CTGFLPTR	Address of VSAM CTGFL.
136 (88)	4	CAMPLPTR	Address of OS CAMLST.
140 (8C)	4	PRMLSTSZ	Size of Dynamic Area to be freed for SVC 26.
144 (90)	72	XSAVAREA	Save area for all external references.
216 (D8)	20	WKCAMLST	CAMLST build area for calling CVOL Catalog Management.
	4	WKOPTNS	Option bytes.
	4	WKPTR1	Address of data set name.
	4	WKCVOLP	Address of CVOL = ZERO.
	4	WKPTR3	Address of the CVOL Catalog Management output area.
	4	WKDSCBP	Address of DSCB TTR.
236 (EC)	4	GIPEPTR	Address of Generation Index Pointer Entry (GIPE).
240 (F0)	4	PCCBPTR	Address of PCCB.
244 (F4)	4	SAVER1	Save area number of bytes in data set name.
248 (F8)	4	SAVER3	Save area for register 3.

Offset	Bytes and Bit Pattern	Field Name	Description: Content, Meaning, Use
252 (FC)	4	SAVER4	Save area for register 4.
256 (100)	4	SAVER6	Save area for register 6.
260 (104)	4	SAVER10	Save area for register 10 for ESTAE.
264 (108)	4	SAVER11	Save area for register 11 for ESTAE.
268 (10C)	4	SAVER12	Save area for register 12 for ESTAE.
272 (110)	4	*	Reserved.
276 (114)	4	SAVEI	Save area for I pointer.
280 (118)	4	OSRC15	CVOL Catalog Management register 15 return code.
284 (11C)	4	OSRC0	CVOL Catalog Management register 0 return code.
288 (120)	4		One of the following:
	4	RELNUM	Binary relative generation number.
	4	ENTCOUNT	CTGFL entry byte count.
292 (124)	4	LBASE	Binary located base number.
296 (128)	4	SBASE	Binary supplied base number.
300 (12C)	44	LOCDSN	Data Set Name hold area.
344 (158)	44	WKDSN	Data Set Name hold area.
388 (184)	1	WKBLANK	Blank character to stop TRT on WKDSN.
389 (185)	.3	WKDSCBT	DSCBTTR hold area.
392 (188)	1	KEYTYPE	Switch to indicate which key IGG0CLCA is currently operating under. X‘00’=SVC, X‘FF’=USER.
393 (189)	.1	OLDKEY	MODESET savekey area.
394 (18A)	..1	INCORESW	Switch to indicate type of block in storage. X‘00’=NAME, X‘FF’=TTR.
395 (18B)	...1	PCCBSW	DO WHILE controller.
373 (175)	1	ENQDEQSW	X‘00’=not enqueued, X‘FF’=enqueued (enqueuing on a chain of PCCBs).
396 (18C)	.3	*	Reserved.
400 (190)	8		One of the following:
	8	HOLDINDX	Index name save area.
	8	HOLDFPLN	CTGFL name being processed.
408 (198)	8	HOLDREL	GDG work area.
416 (1A0)	265	WKVOLST	Volume list area.
	2	WKVOLNUM	Number of volumes.
	..250	WKVOLS	Volume entries.
	3	WKNXTTTR	TTR to next block.
	...10	*	Reserved.
681 (2A9)	.3	*	Reserved.

Offset	Bytes and Bit Pattern	Field Name	Description: Content, Meaning, Use
684 (2AC)	2794	*	Entire 2794 bytes needed for OS CVOL CATBX or RECAT only.
684 (2AC)	256	TRTABLE	Translate and Test Table.
940 (3AC)	4	LKNP	LINK name pointer.
944 (3B0)	4	LKDP	LINK DCB pointer.
948 (3B8)	8	LKNM	Name of module being linked to.
956 (3C0)	16	ESTAELIST	ESTAE macro list form.
972 (3D0)	44	WKTMPCNM	Temporary catalog name.
3478 (D96)	2	*	Reserved.
3720 (D98)	72	WKCLIASV	IGG0CL1A save area.
3552 (DE0)	48	WKSHRPRM	Shared parameter area.
3552 (DE0)	4	ACCCBP	Pointer to Allocate Catalog Control options.
	4	ACCRWP1	Pointer to ACCRWP2.
	4	ACCJSCBP	Pointer to TCBJSCB.
	4	ACCCATP1	Pointer to ACCCATP2.
	4	ACCALSP1	Pointer to ACCALSP2.
	4	ACCDDNMP	Pointer to zero.
	4	ACCRWP2	Pointer to ACCRW.
	4	ACCCATP2	Pointer to WKCATNM Catalog Name.
	4	ACCALSP2	Pointer to WKCATANM Catalog Alias Name.
	4	ACCRW	Return data from Allocate Catalog control.
	2	ACCRETCD	Allocate Catalog Control Return Code.
	.2	ACCRESCD	Allocate Catalog Control Reason Code.
	2	ACCCB	Allocate Catalog Control bits.
	.2	*	Reserved.
3552 (DE0)	16	ENQPARMA	ENQ/DEQ parameter area.
3552 (DE0)	4	*	Area for TCB.
3556 (DE4)	12	ENQDEQPL	ENQ/DEQ parameter list.

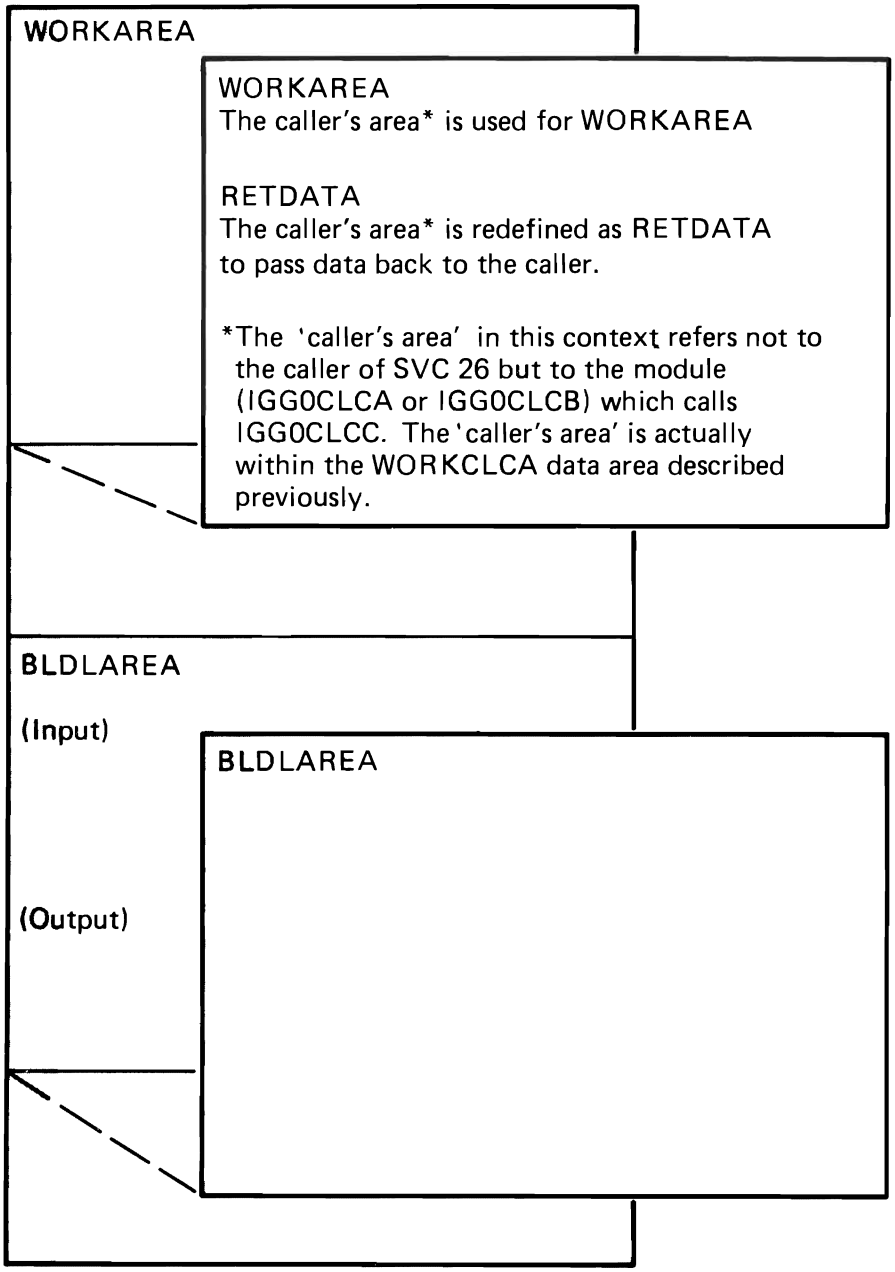
WORKAREA DSECT

WORKAREA serves all CVOL Processor catalog CSECTs as intermediate storage, communications area, and buffers. BLDLAREA is a portion of WORKAREA that serves the resident BLDL routines. For a locate function, BLDLAREA is separate from WORKAREA.

Many of the fields in the WORKAREA overlay other fields, and sections of an area can have more than one label. Figure 5-1 shows where these overlays occur, by label. The listing for any module shows more labels and more detail; only the most significant are shown here.

When function is non-locate, one area (GETMAIN) is used for all purposes.

When function is locate, two areas are used. Space for BLDL comes from GETMAIN.



The caller's area* is used for WORKAREA

The caller's area* is redefined as RETDATA to pass data back to the caller.

*The 'caller's area' in this context refers not to the caller of SVC 26 but to the module (IGG0CLCA or IGG0CLCB) which calls IGG0CLCC. The 'caller's area' is actually within the WORKCLCA data area described previously.

Figure 5-1. Data Area Hierarchy

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0 (0)	4	BLDLIST SAVETTR3	List parameter for BLDL or, when appropriate, the name of the last valid index level.
4 (4)	8	NAME ALIASNAM	Name or alias in the entry that is being operated on.
5 (5)	.4	GENNO	Generation number portion of an absolute GDG name.
12 (C)	3	TTR	Relative track address in the current entry, in the form TTR.
15 (F)	...1	TYPE	Type of entry; also the binary number of halfwords following in the remainder of the entry. TYPE is interpreted as:
		X'00'	Either an index pointer entry (IPE) or an index link entry (ILE). The name field of an ILE always contains X'FFFFFFFFFFFFFFFF'; the name field of an IPE never does.
		X'01'	Volume control block pointer entry (VCBPE).
		X'02'	Generation index pointer entry (GIPE).
		X'03'	Index control entry (ICE) or old CVOL pointer entry (CVPE). An ICE always appears as the first entry of the index; a CVPE always appears in the volume index.
		X'04'	Alias entry (AE).
		X'05'	Volume index control entry (VICE) or new CVOL pointer entry (CVPE). The VICE always appears as the first entry of the first block of the catalog; a CVPE always appears later in the volume index.
		X'07'	Data set pointer entry (DSPE with one volume identification).
		X'0D'	DSPE with two volumes.
		X'13'	DSPE with three volumes.
		X'19'	DSPE with four volumes.
		X'1F'	DSPE with five volumes.
16 (10)	8	TRUE	The true name related to the alias in offset 4.
16 (10)	2	VOLCNT	Number of volumes identified in DATA when the current entry is a data set pointer entry (DSPE).
16 (10)	62	DATA	Volume identifications for DSPE.
88 (58)	1	ERRCATSV	Error code generated for non-locate function.
89 (59)	.1	ERRLOCSV	Error code generated for locate function.
90 (5A)	..1	FLAG1	Switches declaring requested function.
		.1..	The index control entry (ICE) must be read.
		..1.	SYSCTLG has no more room during CATBX or BLDX function.
		...1	The DCB/DEB was freed by SVC 28 processing.
		.... 1...	CATBX request.
		.... .1..	UCATDX request.
		.... ..1.	Locate request.
		1	RECAT request.

(continued)

(cont.)

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
91 (5B)	...1	FLAG2	Switches used to specify flow of control.
		..1.	RPS device.
		...1	Alias entry has been found.
		.... 1...	Sequence error.
		.... .1..	Last entry found was a CVOL pointer entry (CVPE).
		.... ..1.	Generation index pointer entry (GIPE) has been found.
		1	Alias entry has been built.
92 (5C)	28	SAVEAREA	Save area for temporarily storing the contents of general purpose registers.
120 (78)	8	NEXTKEY NEXTCNT	The key or count of the next block beyond the one read.
128 (80)	10	ICE	Index control entry. Only bytes 8 through 15 are saved here.
136 (88)	...9	VICE	Volume index control entry. Only bytes 11 through 18 are saved here.
148 (94)	1	FLAG3	Switches to invoke functions of IGG0CLC5.
		1...	Absolute GDG name found.
		.1..	Free index blocks.
		..1.	Read a block for updating.
		...1	Process EMPTY option of generation data group (GDG).
		.... 1...	Write the last block of a GDG chain when the GDG is full and a new one is being added.
		.... .1..	Build volume control blocks (VCBs).
		.... ..1.	Free VCBs.
		1	Write a block.
149 (95)	.1	FLAG4	Switches to specify the flow of control in IGG0CLC6.
		1...	New entry has been inserted into block now in the work area. Updating is in process.
		.1..	The updated block has been written into SYSCTLG. Updating is complete.
		...1	The block following the block pointed to by field WRITETTR is free.
		.... ..1.	The first write has occurred.
		1	The block following the block pointed to by field LINKTTR is free.
150 (96)	..2	NAMLEN	Length of the full name given by caller minus 1.
152 (98)	4	NAMDELMP	Address of last delimiter in given name.
156 (9C)	4	NAMLSTP	Pointer to last displacement of given name in the name table.
161 (A1)	.1	FLAG5	Flag bits
		...1	CVOL has extended security.
			Switches to specify flow of control in IGG0CLC7:
		.... 1...	Low-level index is involved.
		.... .1..	VFHOLE needs to be updated.
		.... ..1.	LSTBLK needs to be updated.
		1	FSTBLK needs to be updated.

(continued)

(cont.)

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
162 (A2)	..1	MODMAP1	Trace of modules that have been entered. The appropriate bit is set to 1 as each module is entered. There is no bit for subroutine IGG0CLC0, because it is always entered before any other.
		1...	IGG0CLC1
		.1..	IGG0CLC2
		..1.	IGG0CLC3
		...1	IGG0CLC4
		.... 1...	IGG0CLC5
		.... .1..	IGG0CLC6
		.... ..1.	IGG0CLC7
164 (A4)	4	EPBLDL	Address of the entry point of the supervisor routine BLDL, IECPBLDL (copied from field CVTPBLDL of the CVT).
168 (A8)	4	BLDLISTP	Address of the list to be completed by BLDL (address of field BLDLIST, offset 0 of this DSECT).
172 (AC)	4	DCBADDR	Address of the data control block (DCB) for the control volume.
176 (B0)	4	DEBADDR	Address of the data extent block (DEB) for the control volume.
180 (B4)	4	FOUNDENT	Address of an entry in an input/output buffer.
184 (B8)	4	EPTORLTV	Address of the entry point IECPRLTV, a supervisor routine that converts absolute track addresses to relative track addresses (copies from field CVTPRLTV of the CVT).
188 (BC)	4	EPTOABSL	Address of the entry point IECPCNVT, a supervisor routine that converts relative track addresses to absolute track addresses (copied from field·CVTPCNVT of the CVT).
192 (C0)	4	SVRBEXTP	Address of the extension to the SVRB.
196 (C4)	4	ADDING	Address of new entry, meaningful only when bit 0 of FLAG4 is X'1'.
200 (C8)	4	SVBALREG	Branch and link register save area.
204 (CC)	12	*	Reserved
216 (D8)	12	LNKENTRY	General form of index link entry (ILE). The first eight bytes contain X'FFFFFFFFFFFFFFFF'.
224 (E0)	4	LINKTTR	Last four bytes of LNKENTRY; contains the TTR for this ILE.
228 (E4)	4	WRITETTR	Save area for relative address of block to be written.
232 (E8)	4	ICETTR	Relative track address of block that contains an index control entry (ICE).
236 (EC)	4	SAVETTR	Save area for any relative track address.
240 (F0)	4	READTTR	Save area for relative address of block to be read.
244 (F4)	4	CWAP	Pointer to catalog controller work area.
248 (F8)	2	NAMLF	Number of levels of the name that were found.
250 (FA)	..2	NAMLG	Number of levels in given name.
252 (FC)	4	DEVTYPE	Device-type portion of an identification.
256 (100)	1	THETA	Angular displacement value (theta) for rotational positioning support (RPS).
257 (101)	.1	INDEXLEN	Length of all levels given except the last. Used with SCRATCH macro instruction.

(continued)

(cont.)

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
258 (102)	..1	ERRSV2H	Exceptional return code from subroutine IGC0002H.
259 (103)	...6	VOLSN	Serial-number portion of a volume identification.
16 (10)	44	DSNAME	Data set name to be scratched when processing GDG data sets.
60 (3C)	12	SCRPARM	Parameter list for SCRATCH macro instruction.
72 (48)	4	SCRVOLS	Volume list for SCRATCH macro instruction.
32 (20)	44	NAMTABLE	Name table containing the length and displacement of each component of the given name.
76 (4C)	.1	NAMDELIM	Last delimiter in the given name, either ‘b̸’ or ‘ (’.
128 (80)	8	RELNUMBR	Work area for Convert-to-Binary (CVB) instruction used with relative GDG processing.
136 (88)	8	PKDNUMBR	Work area for PACK instruction used with relative GDG processing.
0 (0)	256	RETDATA	Volume list returned to caller.
252 (FC)	4	REDSCBT	Relative track address of the DSCB in the VTOC for a single-volume data set, as returned to the caller.
259 (103)	...6	RETCVOL	Serial number for the control volume containing the returned volume list.
265 (109)	3	VICESAVE	Save area for volume index control entry (VICE) information.
268 (10C)	4	BALREGS	Save area for register used in BAL instruction.
272 (110)	400	BLDLAREA	Work area for use by BLDL routine; for a locate function, WORKAREA is in two parts, and BLDLAREA is the second part.
272 (110)	48	SVAREA2H	Register save area for subroutine IGC0002H.
320 (140)	16	ESTAEPRM	ESTAE exit routine parameter list.
336 (150)	24	ESTAESVA	ESTAE information area for ESTAE error exit cleanup.
360 (168)	16	ESTAELST	ESTAE record parameter list.
632 (278)	120	BLDLCNT	Parameters for BLDL routine.
752 (2F0)	4	BASESAVE	Save area for the register that would otherwise be destroyed by BLDL.
640 (280)	44	RESALIAS	Work area used when resolving an alias name.
376 (178)	256	INPUT	Input buffer for channel program.
376 (178)	256	TRTABLE	Translate table used with TR instruction to analyze the given name.
632 (278)	8	SIDE1	Search-ID-Equal CCW.
640 (280)	8	TIC1	Transfer-In-Channel CCW.
648 (288)	8	OPTNCCW	CCW that is changed to do the required input/output function.
656 (290)	8	RC	Read-Count CCW.
664 (298)	8	SKE	Search-Key-Equal CCW.
672 (2A0)	8	TIC2	Transfer-In-Channel CCW.
680 (2A8)	8	NOP	NOP CCW.
688 (2B0)	4	ECB	Event control block for channel programs.
672 (2B4)	40	IOB	Input/output block for channel programs.
732 (2DC)	8	RKD	Read-Key-Data CCW.
740 (2E4)	8	RD	Read-Data CCW.
748 (2EC)	8	WKD	Write-Key-Data CCW.
756 (2F4)	264	OUTPUT	Output buffer for channel programs.

CAMLSTD DSECT

CAMLSTD describes the parameter list provided by the caller of CVOL Catalog Management. It maps the result of the CAMLST macro instruction.

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
0(0)	1	CAMOPTN1	First option byte.
		1...	Catalog is not on SYSRES.
		..1.	CAT or CATBX request.
		...1	RECAT request.
		.... 1...	UNCAT or UNCATDX request.
		.... ..1.	Locate-by-block request.
1(1)	.1	CAMOPTN2	Second option byte.
		1...	Do not allocate a catalog.
		.1..	BLDX or CATBX request.
		..1.	BLDG request.
		...1	BLDA request.
		.... 1...	LNKX request.
		.... .1..	DLTX or UCATDX request.
		.... ..1.	DSCB TTR has been specified.
		1	DLTA request.
2(2)	..2	CAMOPTN3	Third option byte.
		1...	DRPX request.
		.1..	Scratch GDG data sets.
		.... 1...	Empty generation index when maximum generation count is reached.
		0	VS CAMLST.
		1	VSAM parameter list.
3(3)	...1	CAMGEN	Maximum generation count.
4(4)	4	CAMPTR1	Address of the name field in caller's area. For locate-by-block, the name field contains a relative track address instead of a name.
8(8)	4	CAMCVOLP	Address of CVOL Catalog volume serial number (a 6-byte field).
12(C)	4	CAMPTR3	Address of caller's third parameter. Meaning depends on the function: Locate — Caller's 265-byte work area BLDA — 8-byte name field LNKX — 10-byte volume identification CAT, CATBX or RECAT — Volume list Other — Not used
16(10)	4	CAMDSCBP	Address of three-byte field containing the relative track address (TTR) for the Format 1 DSCB for the data set named through CAMPTR1.

This chapter provides several aids that can be useful when diagnosing difficulties with the CVOL Processor. Before you use the following diagnostic aids, be sure that the CVOL Processor received control as a result of your SVC 26 instruction. That is, make sure that the CVOL Catalog you are referencing is properly defined in the VSAM Master Catalog. Also make sure that the data set you are referencing is defined as an alias of the CVOL Catalog if you are not explicitly specifying the CVOL volume serial in your SVC 26 request. You can use the Access Method Services LISTCAT command to list the VSAM Master Catalog. Refer to the IBM publication *OS/VS2 Access Method Services* for more information on the LISTCAT command. Refer to the IBM publication, *OS/VS2 MVS CVOL Processor,* for more information on how to set up the CVOL Processor.

CSECT IGGOCLCA is written in PL/S-2, a high-level proprietary system language. Listings on microfiche consist of the PL/S-2 source code, a cross-reference and attribute table, and the assembly code. See the IBM publication *Guide to PL/S II* for a more detailed explanation of PL/S and its listings.

CSECTs IGGOCLCB, IGGOCLCC, IGGOCLCD, and IGGOCLCE are written in Assembler language. Of the various symbolic programming languages, Assembler language is the closest to machine language in form and content. Listings on microfiche consist of the Assembler source code, a cross-reference and attribute table, and the assembly code. For more information on Assembler language, see the IBM publications *OS/VS-DOS/VS-VM/370 Assembler Language* and *OS/VS-VM/370 Assembler Programmer's Guide.*

Subroutine Selection Charts for CSECTs IGGOCLCA and IGGOCLCB

Figure 6-1 can help you determine which subroutine of CSECT IGGOCLCA is involved in any given situation. The figure consists of several charts. Each chart shows the path through CSECT IGGOCLCA for the function(s) noted with that chart.

Only subroutine GENLOC calls CSECT IGGOCLCB. Therefore, the GENLOC chart shows the path through IGGOCLCB as well as the path through IGGOCLCA.

Note: The entry point for the CVOL Processor, CSECT IGGOCLCA, the subroutines IGGOCL1A and SRCHPCCB, and the external subroutine IEFAB4F5 are common to all of the functions represented in these charts.

OS CAMLST Format Request

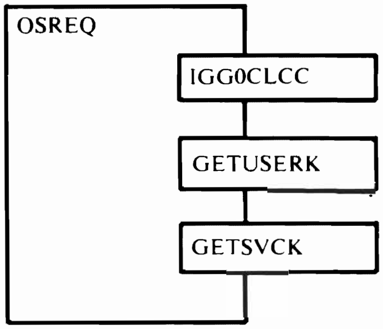


Figure 6-1. Subroutine Selection Charts for CSECTs IGG0CLCA and IGG0CLCB (Chart 1 of 7)

VSAM DELETE

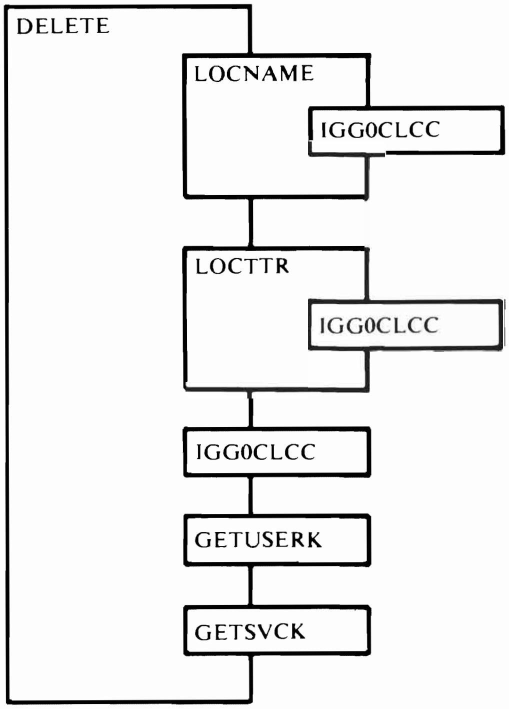


Figure 6.1. Subroutine Selection Charts for CSECTs IGG0CLCA and IGG0CLCB (Chart 2 of 7)

SUPERLOCATE with GDG Base Supplied

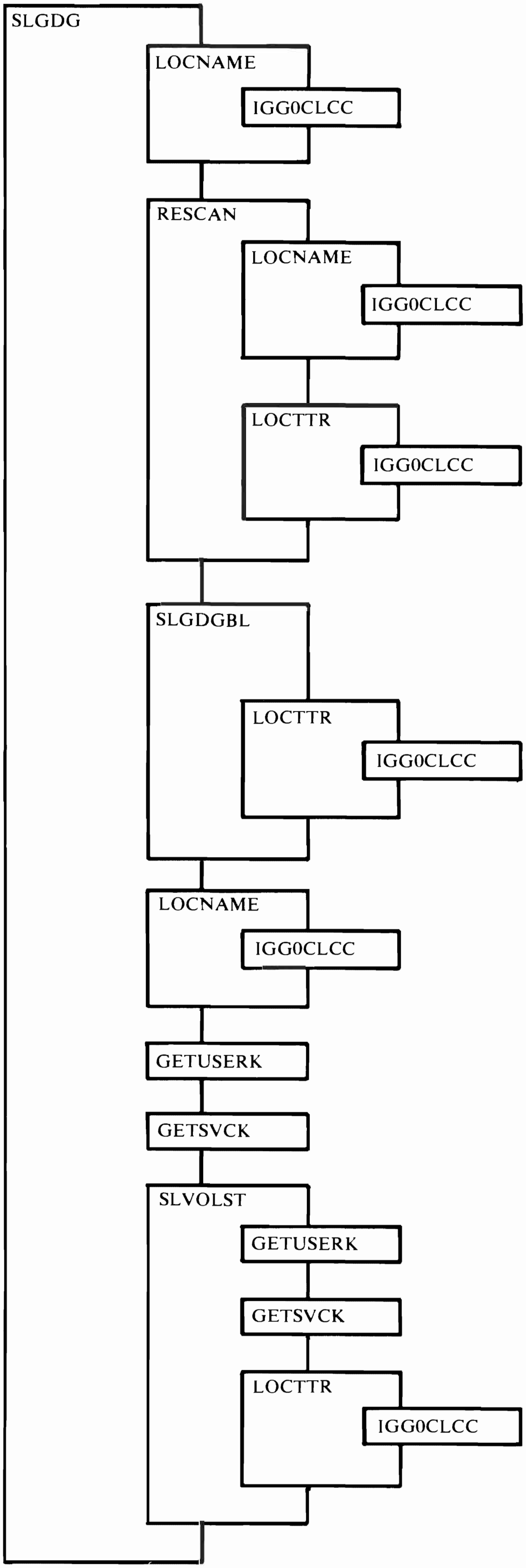


Figure 6.1. Subroutine Selection Charts for CSECTs IGG0CLCA and IGG0CLCB (Chart 3 of 7)

SUPERLOCATE GDG Base Only

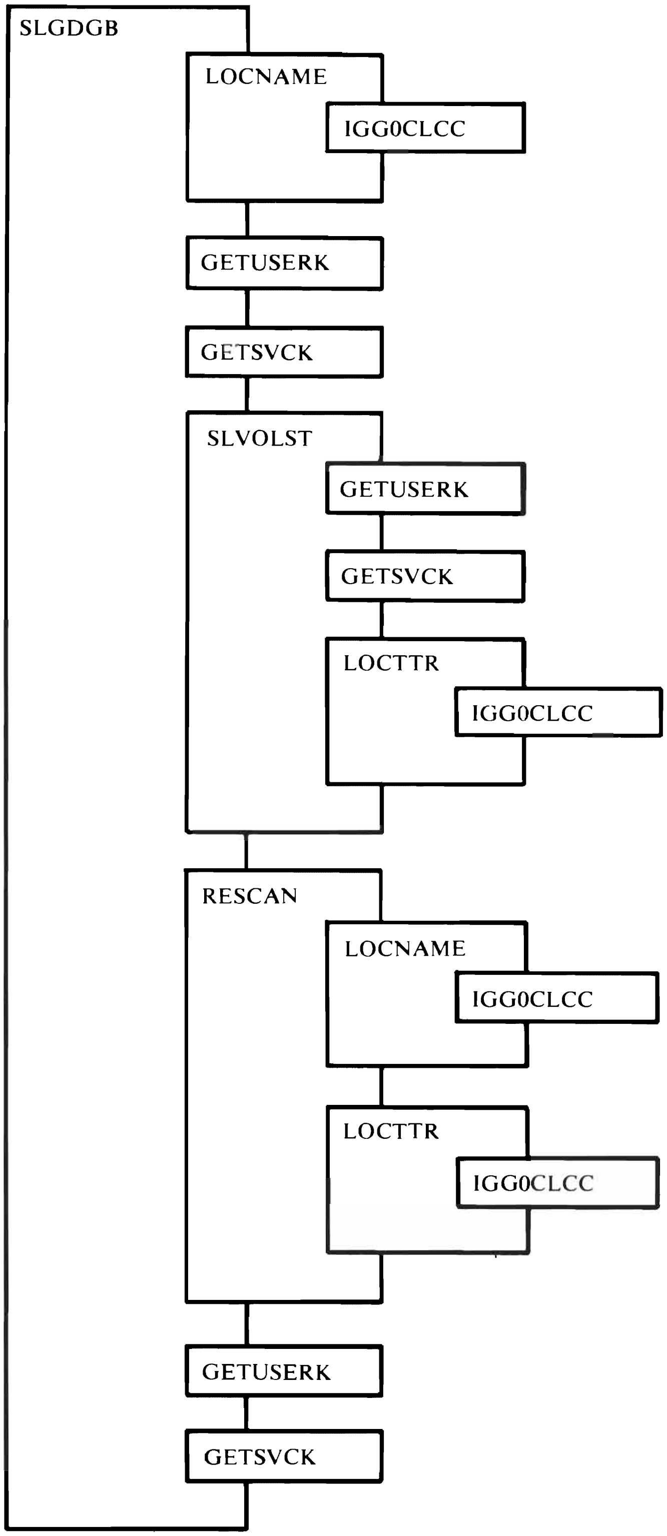


Figure 6.1. Subroutine Selection Charts for CSECTs IGG0CLCA and IGG0CLCB (Chart 4 of 7)

SUPERLOCATE - Normal SUPERLOCATE

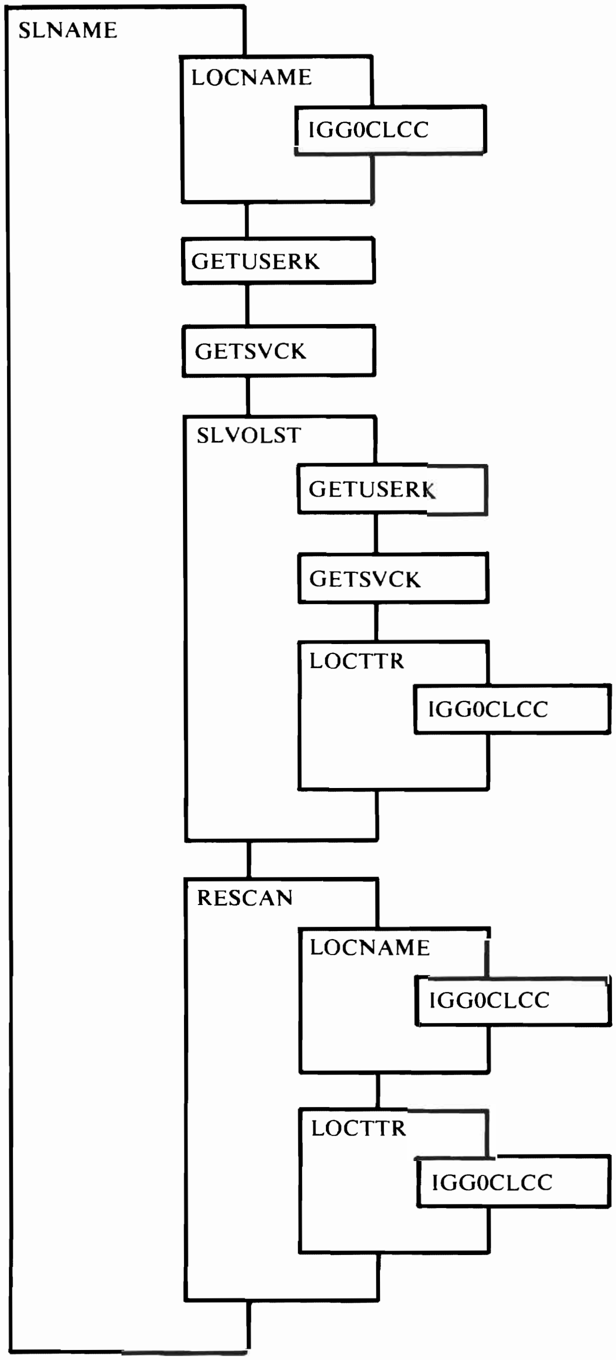


Figure 6.1. Subroutine Selection Charts for CSECTs IGG0CLCA and IGG0CLCB (Chart 5 of 7)

SUPERLOCATE with GENERIC LOCATE Specified

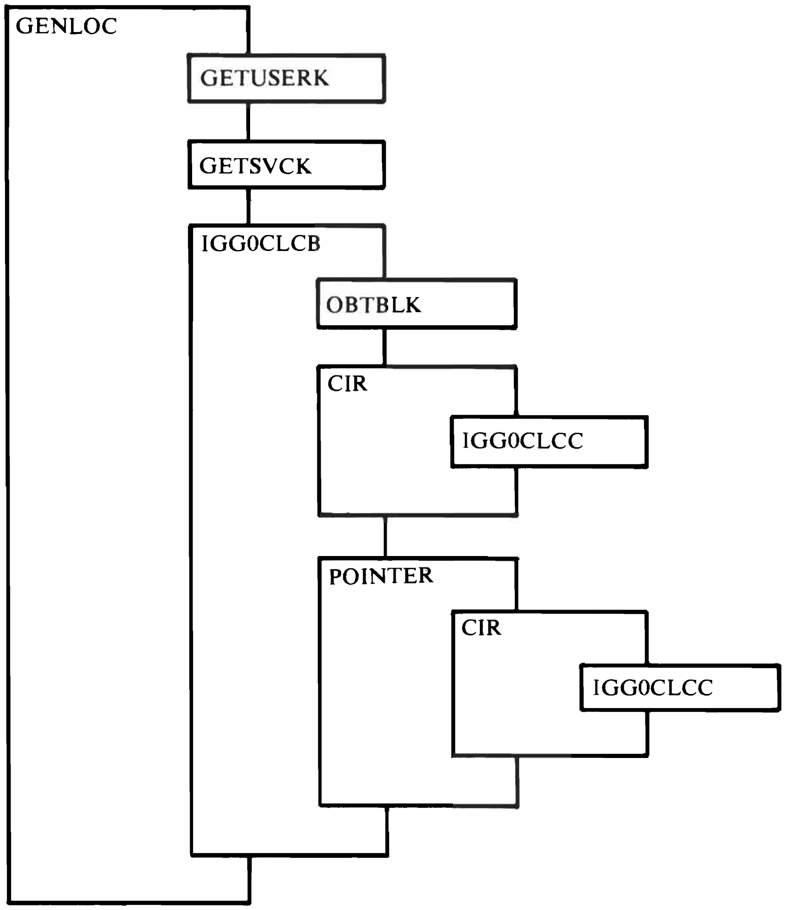


Figure 6.1. Subroutine Selection Charts for CSECTs IGG0CLCA and IGG0CLCB (Chart 6 of 7)

VSAM LOCATE OR VSAM LOCATE-LISTCAT (NOT-GET NEXT)

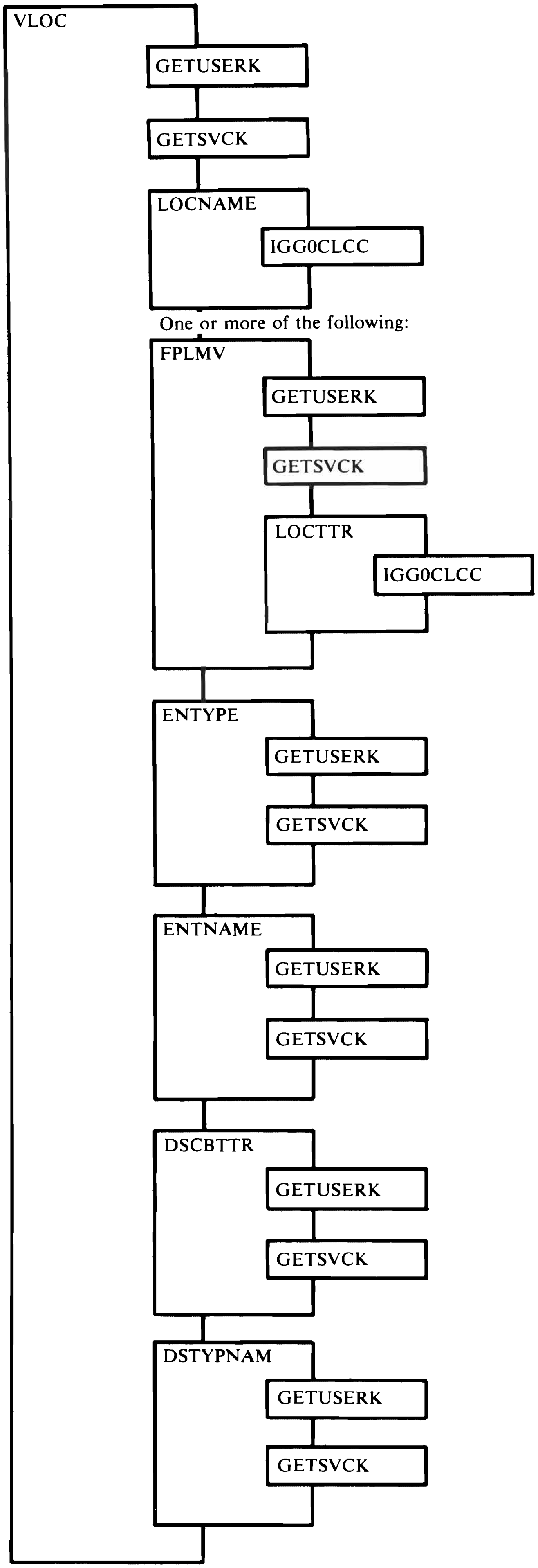


Figure 6.1. Subroutine Selection Charts for CSECTs IGG0CLCA and IGG0CLCB (Chart 7 of 7)

Subroutine Selection Charts for CSECTs IGG0CLCC, IGG0CLCD, and IGG0CLCE

Figure 6-2 can help you determine which subroutines of the CVOL Catalog Management CSECTs are involved in any given function. The figure consists of several charts that are modifications of Figure 3-7 of this publication. Each chart shows the path through the CVOL Catalog Management subroutines for the functions noted on that chart. The specific path is shown by an arrow. Always enter subroutine IGG0CLC0, which is the entry point for CSECT IGG0CLCC (upper left), then move down and to the right.

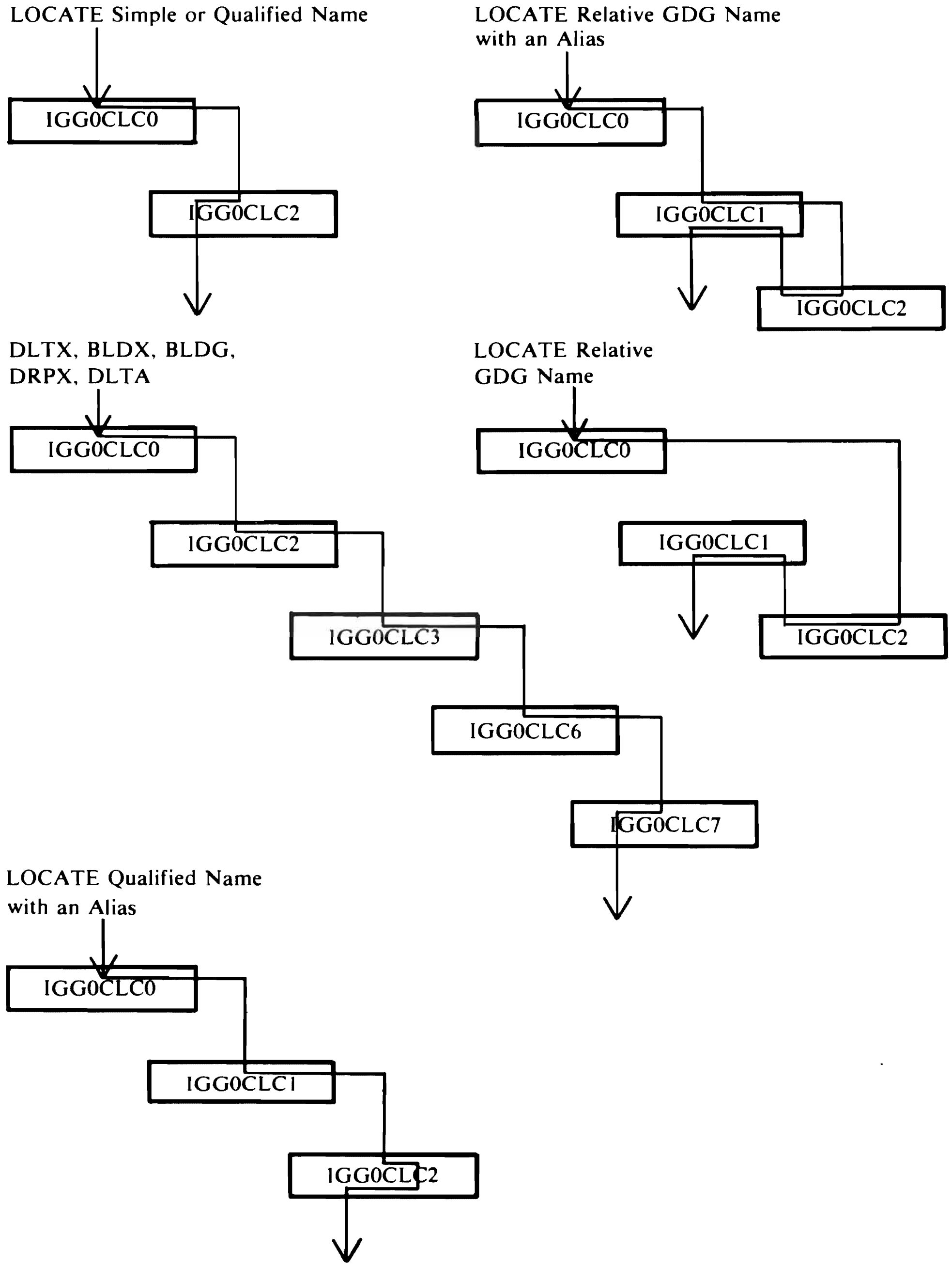


Figure 6-2. Subroutine Selection Charts for CVOL Catalog Management (Chart 1 of 3)

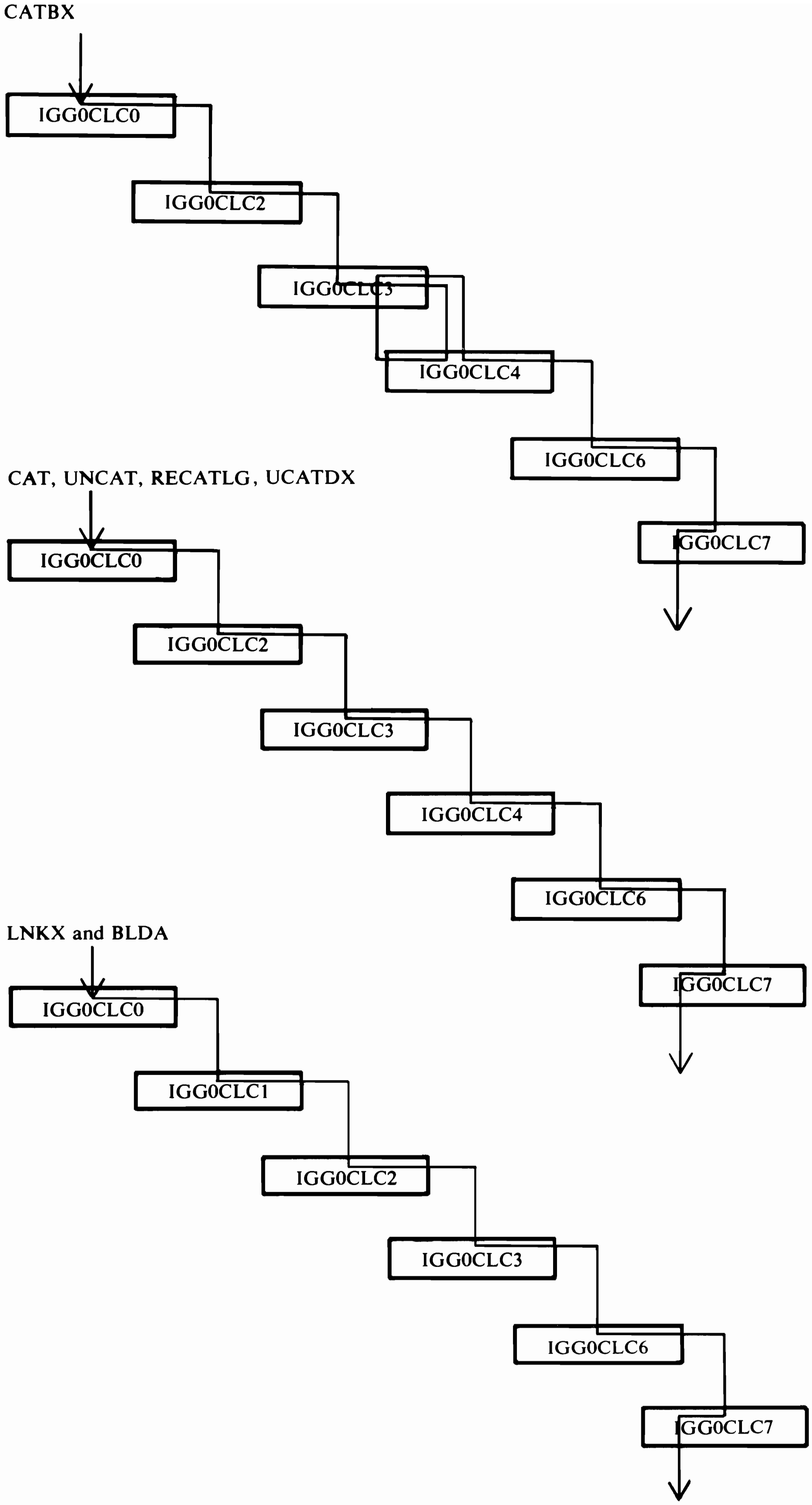


Figure 6.2. Subroutine Selection Charts for CVOL Catalog Management (Chart 2 of 3)

Catalog functions with VCB processing required, GDG Empty option required, or blocks to delete (UCATDX)

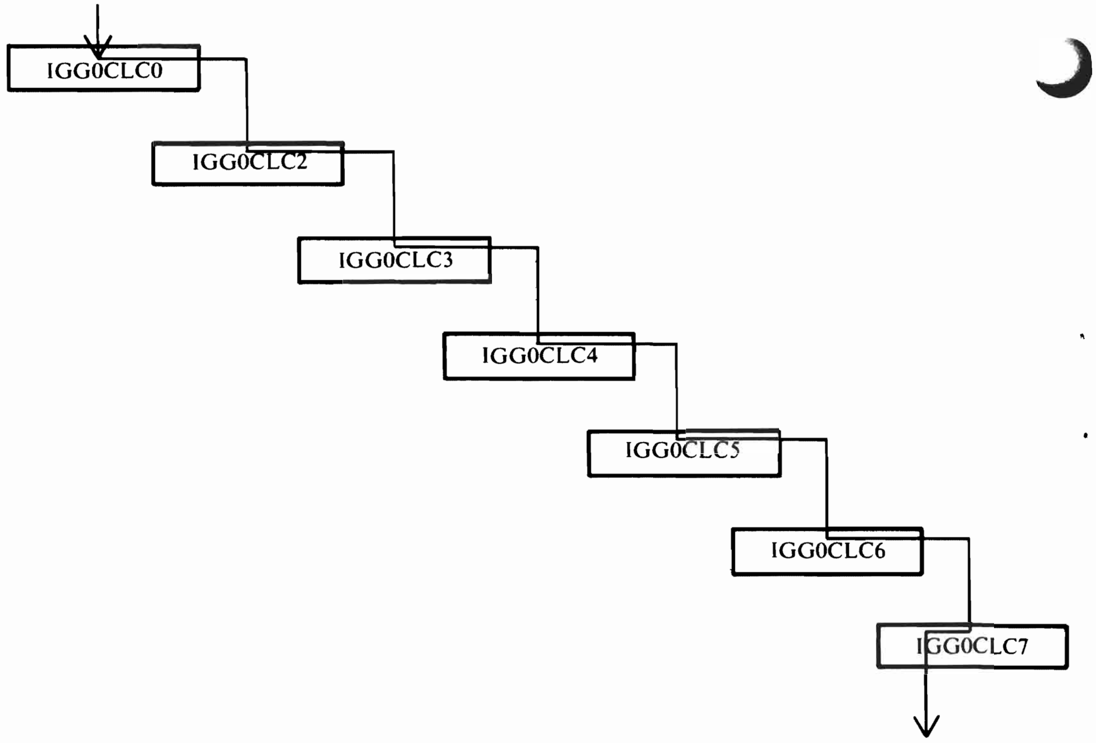


Figure 6.2. Subroutine Selection Charts for CVOL Catalog Management (Chart 3 of 3)

All of the CVOL Processor CSECTs use the ESTAE macro for error analysis. For more information on the ESTAE macro, refer to the IBM publication *OS/VS2 System Programming Library: Supervisor* for more information on the ESTAE macro. If you get an 11A Dump when running the CVOL Processor, the ESTAE macro determined that the error was caused by bad information in the parameter list passed by the SVC 26.

Have the following items available before requesting additional assistance:

- Source or input listing for the use of the CVOL Processor.
- Main storage dump produced by using a //SYSABEND DD statement.
- Dump of CVOL Catalog data set.

Two kinds of dumps can be used while diagnosing trouble with the CVOL Processor: main storage dumps and CVOL Catalog data set dumps. This section points out significant diagnostic clues to look for. It does not explain the full meaning of dumps; for that information, see *OS/VS2 System Programming Library: Debugging Handbook.*

Main Storage Dump

Each CSECT of the CVOL Processor has its own identifier. The identifier is eight characters long, IGG0CLCA for example, and appears in the first few bytes after the entry point to the CSECT.

If an ABEND dump was produced because of an error in one of the CVOL Catalog Management CSECTs, then look at the content of the general registers at the time of the ABEND. The most significant registers are:

Register 4	Base register for all CVOL Catalog Management subroutines, except IECPBLDL, which uses register 15 as a base register.
Register 6	Pointer to WORKAREA. The field MODMAP1 shows which subroutines have been entered; compare this to the expected path for the requested function. The section "Subroutine Selection Charts" for CVOL Catalog Management in Figure 6-2 shows the path for each function.
Register 8	Pointer to CAMLST passed to IGG0CLCC. This CAMLST may be either the original CAMLST (built by the issuer of the SVC 26 instruction), or a CAMLST built by the Interface Mappers.

Register Usage for the CVOL Processor

None of the CVOL Processor CSECTs use standard register linkage. Refer to the following lists for register used by each CSECT.

Register Usage for CSECT IGG0CLCA

Registers:

10	Second base register for CSECT
11	First base register for CSECT
12	Base register for WORKCLCA structure

Register Usage for CSECT IGG0CLCB

Registers:

11	Base register for CSECT
12	Base register for WORKCLCA structure

Register Usage for CSECTs IGG0CLCC, IGG0CLCD, and IGG0CLCE

Registers:

4	Base register for the CSECT
6	Base register for WORKAREA DSECT
8	Base register for CAMLSTD DSECT
12	Linkage register for BAL instructions
14	Linkage register for BAL instructions

CVOL Catalog Dump

There are several ways to dump a data set; this discussion assumes that AMASPZAP is used. AMASPZAP is a service-aid program that operates under the operating system. AMASPZAP is described in *OS/VS2 System Programming Library: Service Aids.*

To dump the catalog with AMASPZAP, use the following JCL, where the //SYSLIB DD card points to the CVOL to be dumped:

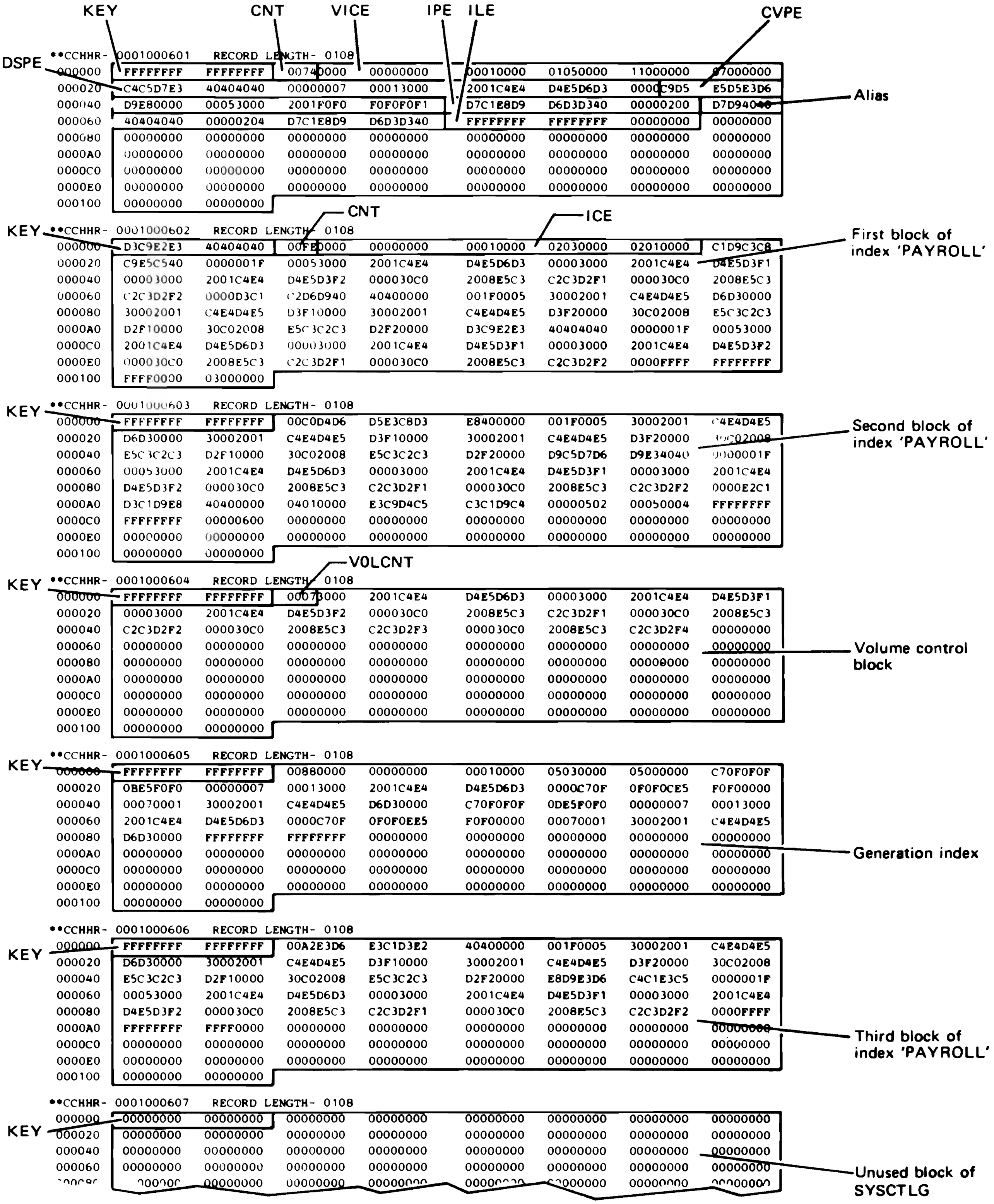
```
//DUMPSTEP  EXEC  PGM=AMASPZAP
//SYSPRINT  DD   SYSOUT=A
//SYSLIB   DD   DSNAME=SYSCTLG,UNIT=xxxx,
//          VOL=SER=xxxxxx,DISP=OLD,
//          DCB=(KEYLEN=8)
//SYSIN    DD   *
 ABSDUMP ALL
/*
```

This JCL is used to dump the entire catalog. You can dump a portion of the catalog by specifying beginning and ending track addresses.

The DCB parameter KEYLEN in the //SYSLIB DD statement formats the key as well as the data for each block. The key appears as the first two words of the first line of each block. The data for the block begins in the third word.

Example of a CVOL Catalog Dump

Figure 6-3 shows an actual dump of the catalog. Entries in the volume index are outlined, and other blocks of the catalog are identified.


KEY
CNT
VICE
IPE
ILE
CVPE
DSPE
Alias
KEY
CNT
ICE
First block of index 'PAYROLL'
KEY
Second block of index 'PAYROLL'
VOLCNT
KEY
Volume control block
KEY
Generation index
KEY
Third block of index 'PAYROLL'
KEY
Unused block of SYSCTLG

```
**CCHHR- 0001000601   RECORD LENGTH- 0108
000000   FFFFFFFF  FFFFFFFF  00740000  00000000   00010000  01050000  11000000  07000000
000020   C4C5D7E3  40404040  00000007  00013000   2001C4E4  D4E5D6D3  0000C9D5  E5D5E3D6
000040   D9E80000  00053000  2001F0F0  F0F0F0F1   D7C1E8D9  D6D3D340  00000200  D7D94040
000060   40404040  00000204  D7C1E8D9  D6D3D340   FFFFFFFF  FFFFFFFF  00000000  00000000
000080   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
0000A0   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
0000C0   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
0000E0   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
000100   00000000  00000000

**CCHHR- 0001000602   RECORD LENGTH- 0108
000000   D3C9E2E3  40404040  00FE0000  00000000   00010000  02030000  02010000  C1D9C3C8
000020   C9E5C540  0000001F  00053000  2001C4E4   D4E5D6D3  00003000  2001C4E4  D4E5D3F1
000040   00003000  2001C4E4  D4E5D3F2  000030C0   2008E5C3  C2C3D2F1  000030C0  2008E5C3
000060   C2C3D2F2  0000D3C1  C2D6D940  40400000   001F0005  30002001  C4E4D4E5  D6D30000
000080   30002001  C4E4D4E5  D3F10000  30002001   C4E4D4E5  D3F20000  30C02008  E5C3C2C3
0000A0   D2F10000  30C02008  E5C3C2C3  D2F20000   D3C9E2E3  40404040  0000001F  00053000
0000C0   2001C4E4  D4E5D6D3  00003000  2001C4E4   D4E5D3F1  00003000  2001C4E4  D4E5D3F2
0000E0   000030C0  2008E5C3  C2C3D2F1  000030C0   2008E5C3  C2C3D2F2  0000FFFF  FFFFFFFF
000100   FFFF0000  03000000

**CCHHR- 0001000603   RECORD LENGTH- 0108
000000   FFFFFFFF  FFFFFFFF  00C0D4D6  D5E3C8D3   E8400000  001F0005  30002001  C4E4D4E5
000020   D6D30000  30002001  C4E4D4E5  D3F10000   30002001  C4E4D4E5  D3F20000  30C02008
000040   E5C3C2C3  D2F10000  30C02008  E5C3C2C3   D2F20000  D9C5D7D6  D9E34040  0000001F
000060   00053000  2001C4E4  D4E5D6D3  00003000   2001C4E4  D4E5D3F1  00003000  2001C4E4
000080   D4E5D3F2  000030C0  2008E5C3  C2C3D2F1   000030C0  2008E5C3  C2C3D2F2  0000E2C1
0000A0   D3C1D9E8  40400000  04010000  E3C9D4C5   C3C1D9C4  00000502  00050004  FFFFFFFF
0000C0   FFFFFFFF  00000600  00000000  00000000   00000000  00000000  00000000  00000000
0000E0   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
000100   00000000  00000000

**CCHHR- 0001000604   RECORD LENGTH- 0108
000000   FFFFFFFF  FFFFFFFF  00073000  2001C4E4   D4E5D6D3  00003000  2001C4E4  D4E5D3F1
000020   00003000  2001C4E4  D4E5D3F2  000030C0   2008E5C3  C2C3D2F1  000030C0  2008E5C3
000040   C2C3D2F2  000030C0  2008E5C3  C2C3D2F3   000030C0  2008E5C3  C2C3D2F4  00000000
000060   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
000080   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
0000A0   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
0000C0   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
0000E0   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
000100   00000000  00000000

**CCHHR- 0001000605   RECORD LENGTH- 0108
000000   FFFFFFFF  FFFFFFFF  00880000  00000000   00010000  05030000  05000000  C7F0F0F0F
000020   0BE5F0F0  00000007  00013000  2001C4E4   D4E5D6D3  0000C70F  0F0F0CE5  F0F00000
000040   00070001  30002001  C4E4D4E5  D6D30000   C70F0F0F  0DE5F0F0  00000007  00013000
000060   2001C4E4  D4E5D6D3  0000C70F  0F0F0EE5   F0F00000  00070001  30002001  C4E4D4E5
000080   D6D30000  FFFFFFFF  FFFFFFFF  00000000   00000000  00000000  00000000  00000000
0000A0   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
0000C0   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
0000E0   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
000100   00000000  00000000

**CCHHR- 0001000606   RECORD LENGTH- 0108
000000   FFFFFFFF  FFFFFFFF  00A2E3D6  E3C1D3E2   40400000  001F0005  30002001  C4E4D4E5
000020   D6D30000  30002001  C4E4D4E5  D3F10000   30002001  C4E4D4E5  D3F20000  30C02008
000040   E5C3C2C3  D2F10000  30C02008  E5C3C2C3   D2F20000  E8D9E3D6  C4C1E3C5  0000001F
000060   00053000  2001C4E4  D4E5D6D3  00003000   2001C4E4  D4E5D3F1  00003000  2001C4E4
000080   D4E5D3F2  000030C0  2008E5C3  C2C3D2F1   000030C0  2008E5C3  C2C3D2F2  0000FFFF
0000A0   FFFFFFFF  FFFF0000  00000000  00000000   00000000  00000000  00000000  00000000
0000C0   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
0000E0   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
000100   00000000  00000000

**CCHHR- 0001000607   RECORD LENGTH- 0108
000000   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
000020   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
000040   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
000060   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
000080   00000000  00000000  00000000  00000000   00000000  00000000  00000000  00000000
```

Figure 6-3. Example of a CVOL Catalog Dump

The data portion of each block begins with a 16-bit binary number that tells how many bytes of the block are used (including the two bytes of this number). The catalog entries begin immediately thereafter. These entries are described in detail in the chapter "Data Areas."

The first entry of the first block is always the volume index control entry (VICE). The type of each entry can be determined from the byte at offset 11 of the entry; the type codes are described under the field TYPE in the WORKAREA DSECT found in Chapter 5 of this publication.

Environment Record

Some error conditions cause an environment record to be written, whenever possible, to the last block of the CVOL Catalog. The environment record is written when a non-locate function is requested, and the exceptional return code is 8 or 28. Here's how you can dump the catalog and examine this record to see what happened:

1. Reproduce the failure, but this time reserve the data set CVOL Catalog for your exclusive use, so that no other task can destroy the environment record before you can dump it. Do this by adding or modifying your JCL statements to include a DD statement for CVOL Catalog with DISP=OLD.
2. Add a step to your job to dump CVOL Catalog. Follow the instructions under "CVOL Catalog Dump."
3. Look at the VICE, which begins at offset two of the first physical block of the catalog. (Remember to allow for the key.) Field VCLSTBLK (offset 12 bytes in the VICE) contains the TTR for the last block in CVOL Catalog. This block contains the environment record.
4. Compute the absolute track address by using the cylinder-head numbers supplied for the first block and the TTR. TT is the relative track from the first block; R is the record number for that track.
5. The fields of the environment record are described in "Environment Record (EREC DSECT)," in the "Data Areas" chapter of this publication. The description for each field relates this information to other data areas.

The field ERMODMAP contains seven bits that show which subroutines have been entered. IGG0CLC0 is always entered; there are no bit switches for this subroutine.

As an example, if ERMODMAP equals X'76', then modules IGG0CLC0, IGG0CLC2, IGG0CLC3, IGG0CLC4, IGG0CLC6, and IGG0CLC7 were entered during the request that caused the environment record to be written. This is the sequence of subroutines that normally occurs with a request for CATBX.

Note: The environment record is not written for any error associated with a "catalog-full" condition.

Glossary

This glossary contains definitions of words and acronyms that are used in this publication. Other data processing definitions can be found in the *Data Processing Glossary.*

alias: An alternative name for a data set. In a CVOL Catalog, only the high-level name of a fully qualified data set name may have an alias.

cataloged data set: In a CVOL Catalog, a data set that is represented in an index or hierarchy of indexes that provides the means for locating the data set.

communication vector table (CVT): An operating system control block that provides the address of information in the nucleus to non-resident routines.

control volume (CVOL): An OS/VS Catalog that contains one or more of the indexes.

CVOL catalog: The collection of all data set indexes maintained by CVOL Catalog Management.

data control block (DCB): An operating system control block that describes the current use of the data set.

data extent block (DEB): A control block that describes the physical attributes of the data set.

data set: The major unit of data storage and retrieval in the operating system.

data set control block (DSCB): A label for a data set on a direct storage volume.

data set name: An identifier that unambiguously names a data set.

data set pointer entry (DSPE): A CVOL Catalog entry that identifies the volume on which a named data set resides.

dequeue: To remove a request for a resource from a list of requests.

DEQ: An Assembler language macro instruction used to remove control of one or more serially reusable resources from the active task. It can also be used to determine whether control of the resource is currently assigned to or requested for the active task.

enqueue: To build a list of requests for a named resource.

ENQ: An Assembler language macro instruction that requests the control program to assign control of one or more serially reusable resources to the active task. It is also used to determine the status of resource, that is, whether it is immediately available or in use, and whether control has been previously requested for the active task in another ENQ macro instruction.

entry: A logical record of a catalog.

environment record: A 256-byte record that is written when CVOL Catalog Management discovers an error. This record, which contains significant data that is present at the time of the error, is written to the last block of data set SYSCTLG for later analysis.

ESTAE: A Supervisor macro instruction used to extend the recovery capability of the STAE macro. ESTAE provides more levels of recovery than the STAE macro.

EXCP: An Assembler language macro instruction that requests the initiation of the I/O operations of a channel program.

FREEMAIN: An Assembler language macro instruction that releases one area of main storage that had previously been allocated to the job step as a result of a GETMAIN macro instruction.

generation: One member of a generation data group.

generation data group (GDG): A collection of historically related data sets.

generation index: An index of the CVOL Catalog that identifies the generations of a generation data group.

generation index pointer entry (GIPE): A CVOL Catalog entry that identifies a generation index.

GETMAIN: An Assembler language macro instruction that is used to allocate an area of main storage for use by the job step task.

high-level name: The first component of a qualified name. This name is found in a volume index of the CVOL Catalog.

index: A table in the CVOL Catalog structure that is used to locate data sets.

index control entry (ICE): The first entry of each index of the CVOL Catalog. This entry contains all control information about the index.

index link entry (ILE): The last entry of each block of the CVOL Catalog, used to link blocks of one index together in a chain.

index pointer entry (IPE): A CVOL Catalog entry that attaches a lower-level index to the index in which it is found.

level: A conceptual relationship between indexes of the CVOL Catalog. The index corresponding to the simple name of a data set is said to be the lowest level; the first component of a qualifier name is said to correspond to the highest-level index.

LINK: An Assembler language macro instruction that causes control to be passed to a specified entry point. The linkage relationship established is the same as that created by a BAL instruction.

locate: Pertaining to functions that do not change the status of a catalog; that is, read-only operations are performed.

MODESET: A Supervisor macro instruction used to change the system status by altering the PSW key or the mode indicator.

must-complete: An indication to the operating system that the event must be performed without interruption or waiting.

non-locate: Pertaining to functions that change the status of a catalog: that is, write operations are performed.

partitioned data set directory: The portion of a partitioned data set that provides a means of locating any of the members of the data set.

qualified name: A data set name consisting of a string of names separated by periods: for example, "TREE.FRUIT.APPLE," is a qualified name.

qualifier: Each component name in a qualified name other than the rightmost name. For example, "TREE" and "FRUIT" are qualifiers in "TREE.FRUIT.APPLE."

relative track address (TTR): A direct-access device address, expressed as a displacement in a data set. This address has the form TTR, where TT represents two hexadecimal digits specifying the track relative to the beginning of the data set, and R is one hexadecimal digit specifying the record on that track.

resource: Any facility of the computing system or operating system required by a job or task, including main storage, input/output devices, the central processing unit, data sets, and control processing systems.

RETURN: An Assembler language macro instruction that is used to return control to the calling CSECT and to signal normal termination of the returning CSECT.

ripple: Moving data from one block of a chain to the next, due to modification of data in a preceding block.

SAVE: An Assembler language macro instruction that causes the contents of the specified registers to be stored in the save area at the address contained in register 13.

SCRATCH: An Assembler language macro instruction that points to the CAMLST macro instruction. SCRATCH, the first operand of CAMLST, specifies that a data set be deleted.

simple name: The rightmost component of a qualified name. For example, "APPLE" is the simple name in "TREE.FRUIT.APPLE." The simple name corresponds to the lowest index level in the CVOL Catalog for the data set name.

supervisor request block (SVRB): An operating system control block containing program status information and general register contents.

SYSCTLG: The data set name of the CVOL Catalog.

system residence volume: The volume on which the nucleus of the operating system is located.

task control block (TCB): An operating system control block that contains information and pointers associated with the task in progress.

true name: In a CVOL Catalog, the high-level qualifier to which an alias is related.

uncatalog: To remove the catalog entry of a data set from a catalog.

volume control block (VCB): A block of the catalog that identifies as many as 20 volumes containing one data set.

volume control block pointer entry (VCBPE): A CVOL Catalog entry that identifies a VCB for a named data set.

volume index: The highest level of index in the CVOL Catalog structure. Entries in the volume index point to all lower indexes and simple names.

volume index control entry (VICE): The first entry in the volume index. The VICE describes the volume index and controls space allocation in SYSCTLG.

volume table of contents (VTOC): A table associated with a direct access volume that describes each data set on that volume and identifies all available space on the volume.

WAIT: An Assembler language macro instruction that informs the control program that the issuing program cannot continue until a specific event, represented by an event control block, has occurred.

XCTL: An Assembler language macro instruction that causes control to be passed to a specified entry point.

Index

SY26-3860-0

OS/VS2 CVOL Processor Logic - VS2.03.808 Printed in U.S.A. File No. S370-30 SY26-3860-0

IBM

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

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

OS/VS2 CVOL Processor Logic
SY26-3860-0

Reader's Comment Form

Your comments about this publication will help us to improve it for you Comment in the space below, giving specific page and paragraph references whenever possible. All comments become the property of IBM.

Please do not use this form to ask technical questions about IBM systems and programs or to request copies of publications. Rather, direct such questions or requests to your local IBM representative.

If you would like a reply, please provide your name, job title, and business address (including ZIP code).

Fold on two lines, staple, and mail. No postage necessary if mailed in the U.S.A. (Elsewhere, any IBM representative will be happy to forward your comments.) Thank you for your cooperation.

SY26-3860-0

Fold and Staple

First Class Permit
Number 6090
San Jose, California

Business Reply Mail

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

Postage will be paid by:

IBM Corporation
P.O. Box 50020
Programming Publishing
San Jose, California 95150

Fold and Staple

OS/VS2CVOL Processor Logic - VS2.03.808 Printed in U.S.A. File No. S370-30 SY26-3860-0

IBM®

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

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

OS/VS2 CVOL Processor Logic
SY26-3860-0

Reader's Comment Form

Your comments about this publication will help us to improve it for you Comment in the space below, giving specific page and paragraph references whenever possible. All comments become the property of IBM.

Please do not use this form to ask technical questions about IBM systems and programs or to request copies of publications. Rather, direct such questions or requests to your local IBM representative.

If you would like a reply, please provide your name, job title, and business address (including ZIP code).

Fold on two lines, staple, and mail. No postage necessary if mailed in the U.S.A. (Elsewhere, any IBM representative will be happy to forward your comments.) Thank you for your cooperation.

SY26-3860-0

Fold and Staple

First Class Permit
Number 6090
San Jose, California

Business Reply Mail

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

Postage will be paid by:

IBM Corporation
P.O. Box 50020
Programming Publishing
San Jose, California 95150

Fold and Staple

IBM®

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

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