[illegible]

SY35-0003-2

Systems

OS/VS
Catalog Management Logic

VS1 Release 2
VS2 Release 1

IBM®

Third Edition (January 1973)

This edition corresponds to Release 2 of OS/VS1 and to Release 1 of OS/VS2. All references to VSAM in this release are for planning purposes only.

A change to the text is indicated by a vertical line to the left of the change; a changed illustration is denoted by a vertical line to the left of the caption.

The information contained in this publication is subject to significant change. Any such change will be reported in subsequent editions or technical newsletters. Before using this publication in connection with the operation of IBM systems, consult the latest *IBM System/360 and System/370 SRL Newsletter,* GN20-0360, for the editions that are applicable and current.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office that serves you.

A form for reader's comments appears at the back of this publication. Address any additional comments concerning the publication to: IBM Corporation, Programming Publications, Department 27D, Post Office Box 1900, Boulder, Colorado 80302. Comments become the property of IBM.

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

You should be familiar with general programming techniques, OS/VS concepts and use, and System/370 before reading this book. If you are unfamiliar with the general concepts of catalog management, read:

- *OS/VS Data Management Services Guide*, GC28-3783, for the general concepts of data sets and catalog management.
- *OS/VS VSAM Programmer's Guide,* GC26-3806, for the general concepts of VSAM data sets and catalog management.
- *OS/VS Utilities*, GC35-0005, for information about IEHPROGM and generation data groups.
- *OS/VS Data Management for System Programmers,* GC28-0631, to see how catalog-management routines are invoked.

Other books that may be helpful to you are:

- *OS/VS1 Debugging Guide*, GC24-5093, which describes how to analyze a main storage dump from OS/VS1.
- *OS/VS2 Debugging Guide*, GC28-0632, which describes how to analyze a dump from OS/VS2.
- *OS/VS Service Aids,* GC28-0633, which describes several service aids and programs available under the operating system (notably for this book, HMASPZAP).
- *OS/VS DADSM Logic,* GY26-3787, which provides details about module IGG0553A and rotational positioning support (RPS).
- *OS/VS SAM Logic,* GY26-3788, which provides details about the BLDL routine.
- *OS/VS VSAM Logic*, SY26-3805, which provides the internal logic of the VSAM catalog routines, and describes the VSAM parameter list.
- *OS/VS1 System Data Areas*, SY28-0605, which shows the content of most of the operating system control blocks and tables.
- *OS/VS2 System Data Areas*, SY28-0606, which shows data areas for OS/VS2.

This book is divided into six chapters and a glossary:

- "Introduction" describes what the catalog is, how the catalog is structured, and how the catalog-management routines are invoked. The chapter defines terms that are used throughout the book.
- "Method of Operation" provides the design overview. This chapter describes the philosophy behind the programs. Emphasis is on the flow of data and the concepts of catalog management, rather than on the organization of the modules.

- "Program Organization" describes each module of catalog management and identifies the specific function that each performs to achieve the catalog management objectives. This chapter shows the logical flow from module to module and contains the flowcharts of the modules.
- "Microfiche Directory" contains a table that relates the information in this book to the listings on microfiche.
- "Data Areas" describes the record formats and work areas that are used by catalog management.
- "Diagnostic Aids" can help you when you're diagnosing a problem. This chapter shows how to dump and analyze the catalog, and how to determine what modules are used for a particular case.
- "Glossary" of terms and acronyms used in this publication is found just before the index.

Contents

Flowcharts

Summary of Amendments

VSAM capability is added. Three new modules (IGC0002F, IGG0CLCA, and IGG0CLCB) comprising Controller II, route catalog management requests to the appropriate catalog when both the OS/VS and VSAM catalogs are present. Controller I is present in the one-catalog environment (OS/VS only).

The initial load of both controllers is named IGC0002F.

If you are using Release 1 of OS/VS1 at this time, then ignore all changes that reflect VSAM capabilities.

Notes

Introduction

Catalog management is the part of OS/VS that enables a user to locate a data set by specifying only a data set name. Catalog management stores data set names with volume and device information in a catalog, itself a data set named SYSCTLG. In use, the name in the user's request is correlated with the appropriate volume identification. The catalog-management routines organize the catalog by inserting, removing, and locating entries in the catalog, and by formatting new catalogs.

Catalog management is a type 4 SVC. Under VS1, the catalog-management routines execute in the 1024-byte transient areas of the operating system nucleus. Under VS2, these routines execute in the Link Pack Area. System and machine requirements, therefore, are those of the operating system.

This chapter discusses the catalog, illustrates several examples of a catalog, and discusses how the catalog-management routines are invoked.

SYSCTLG Data Set

The catalog is always named SYSCTLG; thus, there can be only one OS/VS catalog on any one volume.

SYSCTLG data sets are formatted like partitioned data set (PDS) directories. Each physical block consists of an 8-byte key and a 256-byte data record. The format of the logical records contained in the physical block is shown in the chapter "Data Areas," page 61.

Names and Indexes

Every cataloged data set has a name; this name can be a *simple name* or a *qualified name.*

A simple name is a collection of up to eight EBCDIC characters. A qualified name is a string of names, separated by periods. The last name in the string is a simple name; the preceding names are *qualifiers.* Each qualifier can be up to eight characters long; the maximum total length of a qualified name is 44 characters, including the separating periods.

Each qualifier of a data set name corresponds to one *index* in the catalog. The structure of these indexes forms the catalog. The simple name of a data set is found in the lowest level of structure corresponding to the fully qualified name. The highest level of the catalog is called the *volume index;* the volume index is automatically built the first time a new catalog is used by the catalog management routines.

Names in an index level are arranged in EBCDIC collating sequence.

Each physical block of SYSCTLG can contain only one index level. An index level, however, can span more than one block. Multiple blocks of one index are chained together with the last logical record in each block. Contiguous blocks make use of the key to speed searching.

Figure 1 shows an example of a catalog containing only the volume index and one simple name. (We will build on this figure throughout this chapter.) As with any data set on a direct access volume, SYSCTLG is found by tracing through the Volume Label to the Volume Table of Contents (VTOC). The VTOC contains a Data Set Control Block (DSCB) for each data set on that volume (independent of whether that data set is cataloged). The Format 1 DSCB for SYSCTLG points to the catalog data set.

The catalog data set is shown as the shaded block in Figure 1. The first record of SYSCTLG contains the volume index. The entry in that volume index identifies the device type and serial number for the volume that contains the data set ALPHA.

Figure 2 shows a qualified name in the catalog. Cataloging the data set BETA.B.C adds another entry to the volume index. This entry points to the BETA index, which contains an entry for B. Following the pointer to the B index, we find an entry for C, which identifies the volume containing data set BETA.B.C.

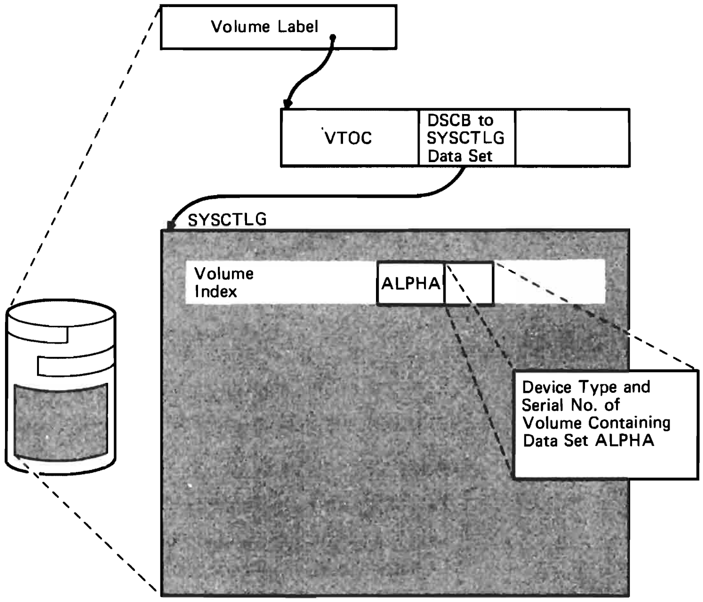


Figure 1. Example of a Catalog with One Simple Name

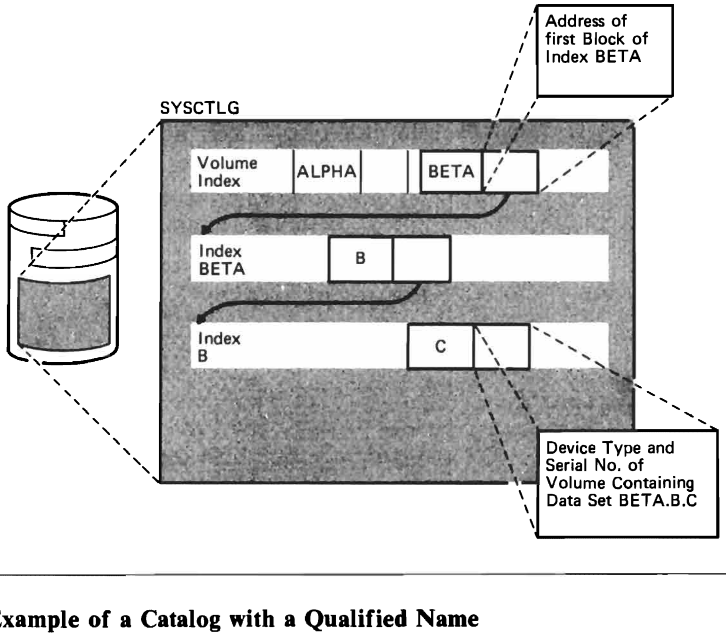

Address of first Block of Index BETA
SYSCTLG
Volume Index
ALPHA
BETA
Index BETA
B
Index B
C
Device Type and Serial No. of Volume Containing Data Set BETA.B.C

Figure 2. Example of a Catalog with a Qualified Name

Control Volumes

Any direct access volume supported by OS/VS may contain a catalog. A volume that contains a catalog is called a *control volume* (CVOL). The system residence volume always contains a catalog.

A user specifies which CVOL contains the catalog (perhaps by allowing the default to the system residence volume). For a data set to be available through the catalog, its name must appear in either the catalog on the specified CVOL or in a CVOL that is *connected* to the specified CVOL. When a CVOL is connected, there is a pointer in the volume index of the catalog on the specified CVOL. This identifies the connected CVOL and names a high-level index on the connected CVOL.

A search for a data set may start on the system residence volume; a CVOL pointer on the system residence volume can direct the search to a connected CVOL. The search starts again on the connected CVOL, using the fully qualified name in the CVOL's SYSCTLG data set. The search may alternately be started on a CVOL other than the system residence volume, as directed by the user.

Figure 3 shows a CVOL pointer in the catalog. This CVOL pointer connects another CVOL to this catalog. In Figure 3, an entry for Y has been placed in the volume index. This entry identifies the volume that contains the name Y in the volume index of its catalog.

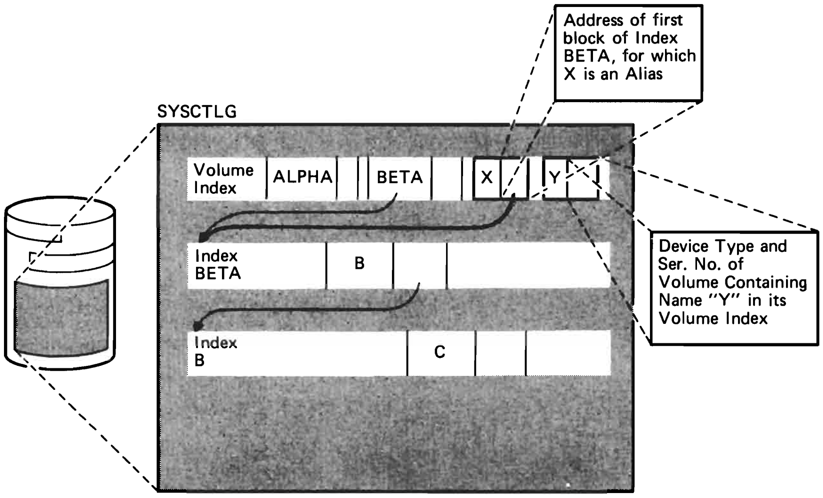

Address of first block of Index BETA, for which X is an Alias
SYSCTLG
Volume Index
ALPHA
BETA
X
Y
Index BETA
B
Index B
C
Device Type and Ser. No. of Volume Containing Name "Y" in its Volume Index

Figure 3. Example of a Catalog with an Alias and a Connected CVOL

Aliases

An *alias* is an alternative name for the highest-level index in a qualified data set name. There can be as many as 255 aliases to the first component of a qualified name.

Figure 3 also shows an alias entry. Here the name X is an alias for BETA, so an entry for X is placed in the volume index. The entry for BETA and the entry for X both point to index BETA. Note that only the high-level name can have an alias.

Generation Data Groups

A *generation data group* (GDG) is a collection of data sets that have a common external name, and that are related by the time sequence in which they were cataloged (that is, their generation). For example, a data set named LAB.PAYROLL(0) refers to the most recent data set of the group;

LAB.PAYROLL(-1) refers to the second most recent data set, etc. The number in parentheses is called the *relative number* of a specific generation.

The relative number relates to an *absolute name* for that generation of the data set. The absolute name is the simple name of the data set. The absolute name has the general form G*nnnn*V*mm,* where *nnnn* is the generation number, and *mm* is the version number. The last qualifier in a relative GDG name is the name of the *generation index*. The generation number in the absolute name is complemented before the name is added to the generation index, so the most recent name (highest generation number) is the first entry in the index.

Figure 4 shows a generation data group in the catalog. In this example, the external data set name BETA.B.F applies to a generation data group, consisting of generations of the same data set. Here the lowest qualifying name, F, does not point to the data set, but rather points to the generation index. This generation index consists of pointers to generations of the data set group BETA.B.F.

Entries of a Catalog

The logical records of the SYSCTLG data set are called *entries.* These entries are linked together to form indexes and to connect indexes together. Figure 5 shows and names each type of entry.

Each index contains one control entry and any number of pointer entries. The control entry, which is the first entry on each index, identifies that index. Other index entries link blocks of one index together, point to lower indexes, and point to data sets.

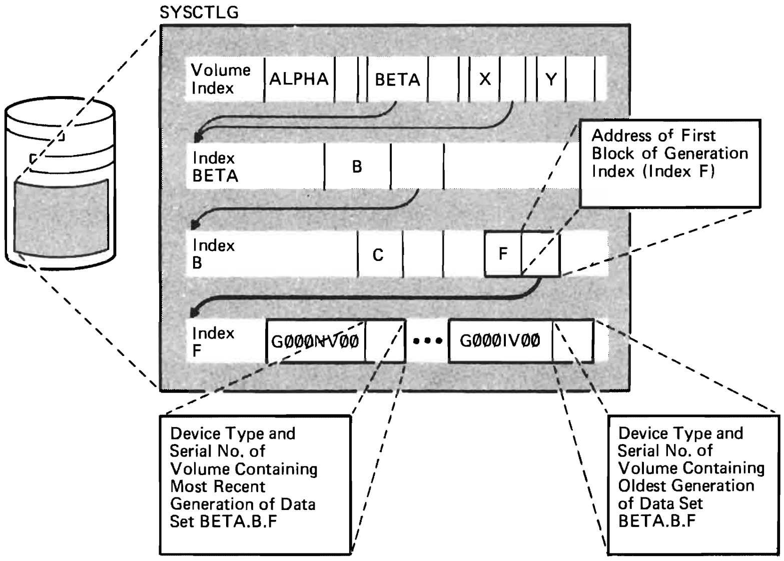

SYSCTLG
Volume Index
ALPHA
BETA
X
Y
Index BETA
B
Index B
C
F
Index F
G000NV00
•••
G000IV00
Address of First Block of Generation Index (Index F)
Device Type and Serial No. of Volume Containing Most Recent Generation of Data Set BETA.B.F
Device Type and Serial No. of Volume Containing Oldest Generation of Data Set BETA.B.F

Figure 4. Example of a Catalog with a Generation Data Group

Entry Name	*Abbreviation*	*Description*
Alias entry	AE	Contains an alternate name for the high-level qualifier of a data set name.
Control volume pointer entry	CVPE	Connects another control volume (CVOL) to this CVOL; contains a name that appears in the volume index of the connected CVOL.
Data set pointer entry	DSPE	Contains the simple name of a data set and provides the location of this data set.
Generation index pointer entry	GIPE	Points to the lowest index for a generation data group.
Index control entry	ICE	First entry of all indexes other than volume index; describes the index level for one qualifier of a data set name.
Index link entry	ILE	Links this block to the next block in a chain of blocks for one index.
Index pointer entry	IPE	Points to a lower-level index of this name.
Volume control block	VCB	A block of the catalog that contains the volume list for a data set that resides on more than five volumes.
Volume control block pointer entry	VCBPE	Points to a volume control block.
Volume index control entry	VICE	The first entry in the volume index (the highest-level index in the catalog); describes the volume index and controls allocation of space in the catalog on one volume.

Figure 5. Kinds of Entries in the Catalog

Using the Catalog Management Routines

Catalog management routines are invoked by the utility IEHPROGM, the job scheduler, the Time Sharing Option (TSO) of the operating system, or by macro statements in an Assembler language program.

IEHPROGM creates and deletes indexes, aliases, and generation indexes, and it catalogs and uncatalogs data sets according to specifications supplied by the user of IEHPROGM.

The job scheduler calls the catalog management routines when it must locate a data set, given only its name, or when the DISP parameter of a DD statement implies catalog action.

Dynamic allocation with TSO locates old data sets and catalogs new data sets. TSO command processors also call the catalog management routines to manipulate the catalog.

All of these invocations of catalog management are produced by using three macro instructions: LOCATE, CATALOG, and INDEX. These macros generate a reference to a parameter list (produced by the CAMLST macro) and an SVC 26 instruction.

One of the operands of the CAMLST macro instruction specifies the function to be performed. These functions are categorized as *locate functions* or *non-locate functions*. The locate functions are those that require read-only operations to the catalog (that is, the catalog is not changed). The non-locate functions are those that do require changes to the catalog.

The locate functions are:

- NAME, which locates a data set by name.
- BLOCK, which locates a block of SYSCTLG.

The non-locate functions are:

- BLDX, which creates one index level.
- BLDG, which creates a generation index.
- BLDA, which assigns an alias to an existing high-level name.
- DLTX, which deletes one index level.
- DLTA, which deletes an alias.
- LNKX, which connects a CVOL.
- DRPX, which disconnects a CVOL.
- CAT, which adds a data set name to the catalog.
- CATBX, which adds a data set name to the catalog and creates any missing index levels.
- UNCAT, which removes a data set name from the catalog.
- UCATDX, which removes a data set name from the catalog and deletes index levels that are no longer needed (except the high-level index).
- RECAT, which recatalogs a data set.

For more information about the use of a catalog, see *OS/VS Data Management for System Programmers,* GC28-0631 (the first chapter), or *OS/VS Utilities,* GC35-0005 (the section on IEHPROGM).

For more information about generation data groups, see *OS/VS Data Management Services,* GC28-3783.

Method of Operation

This chapter explains the design of catalog management.

In its simplest form, catalog management consists of three operations: read, setup, and write. Figure 6 illustrates this overview. Input to catalog management is shown in the block on the left, processing is shown in the center block, and output from catalog management is shown in the block to the right. The three operations are shown in the process block.

VS input to catalog management consists of a parameter list built by the caller with the CAMLST macro instruction, supervisor control blocks, and the existing catalog. The output from catalog management depends on the requested function. Locate functions result in a volume list returned in the caller's area. Non-locate functions result in an updated catalog.

VSAM input consists of a similar parameter list. For a description of the VSAM parameter list, see the chapter "Data Areas" in *OS/VS* VSAM *Logic*, SY26-3805.

When only the OS/VS catalog is present, catalog management, specifically Controller I, translates the VSAM input into CAMLST form. VSAM Superlocate, an internal Scheduler interface, is the only valid translatable list in Controller I. Output is translated to VSAM format.

When both the OS/VS and VSAM catalog are present, Controller II directs activity to the appropriate catalog and translates as required. Duplicate entry checking is always done to prevent a catalog entry from existing in both the OS/VS and VSAM catalog.

For a locate function, only the read operation is required. For a non-locate function, the read operation is used to locate the catalog block to which a change is to be made. The setup operation determines how the change is to be made, and the write operation updates the catalog accordingly.

The CATBX function is slightly more complex; it requires that setup and write be executed several times. I'll discuss that function in more detail after describing each of the operations.

Operation I, Read

Figure 7 illustrates the concepts of Operation I, Read. The input is that shown earlier in the overview. The caller's parameter list includes a function request, points to a data set name, and identifies the CVOL to use (possibly the system residence volume by default). If a function is a locate function, an area in the caller's module is made available for use as a return buffer. The usual SVC linkage for a type 3 SVC or type 4 SVC (catalog management is a type 4 SVC) provides pointers to several system control blocks. The VTOC of the CVOL is read to find the SYSCTLG data set and open it.

Operation I begins by routing—actually pre-processing—the request to the OS/VS or VSAM catalog.

In a one-catalog system, OS/VS only, VSAM Superlocate requests are translated and honored by Controller I.

In a two-catalog system, OS/VS and VSAM, Controller II routes locate-by-name requests as follows:

Case	*# of Ctlgs to Search*	*Parm List*	*Start Ctlg Search*
0	2	OS/VS	VSAM
1	2	VSAM	VSAM
2	1	OS/VS	VSAM
3	1	VSAM	VSAM
4	2	OS/VS	OS/VS
6	1	OS/VS	OS/VS

The case is determined by the decimal equivalent of the last three bits in CAMOPTN3 of the CAMLST (see page 67). Note that the CAMLST macro instruction sets these bits to zero, corresponding to case zero.

Non-locate requests are recognized by Controller II when the case bits are set to 0, 1, 3, and 6. The parameter list passed to the controller determines which catalog the request will be routed to; however, duplicate entry checks are made in the opposite catalog on all catalog-type requests.

Operation I continues by initializing a work area named WORKAREA. The data set SYSCTLG is opened by constructing a data control block (DCB) and a data extent block (DEB), which are shown as intermediate results in Figure 7. The BLDL macro instruction uses the DCB and DEB to read blocks of the catalog. The components of the name are isolated, and the high-level name is used as the search parameter of BLDL. BLDL begins by reading the block that contains the entry for the high-level name.

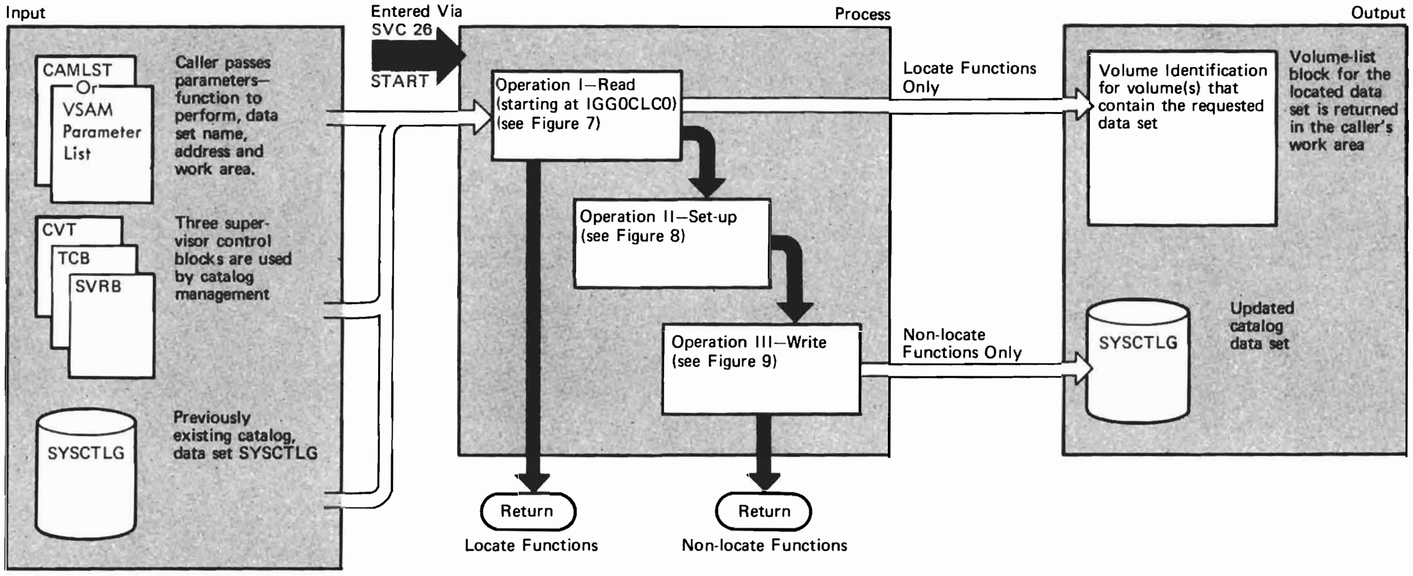
If the entry that is found is a CVOL pointer entry, the current CVOL is closed and the CVOL identified by the pointer is opened. The process repeats until the appropriate CVOL is open.

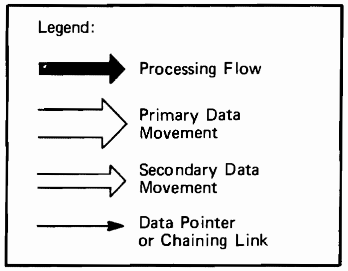
If the entry points to a lower level of the index structure, BLDL is called with the next component of the data set name as a search parameter. This process is repeated until the volume list is read (for locate functions), or the last valid level of the structure is found (for non-locate functions).

Insert A in Figure 7 shows how the blocks of the catalog are chained together and how keys are used in this chaining process. The blocks are visualized in this insert as several contiguous blocks in the catalog, with block addresses ascending from left to right. The last entry of any catalog block is the index link entry (ILE), which is used to chain one block of an index to the next block in the same index. When two blocks are chained together but are not contiguous, as in the case of the first and fourth blocks in the insert for index A, the first block of the pair has a key equal to X'FFFFFFFFFFFFFFFF'. BLDL issues a "search key high-or-equal" operation to the direct access device. When searching non-contiguous blocks, the high value of the key causes each block to be read and scanned for the requested name.

However, when two blocks are chained together and are contiguous, as is the case with index B with the second and third blocks of the insert, the first block of the pair has a key equal to the name in the next-to-last entry of that block. The "search key high-or-equal" operation stops on such a block only if the requested name could be in it; otherwise, the search continues with the next contiguous block.

Available blocks (that is, blocks that are unused) have a key equal to X'0000000000000000'. Thus an available block of the catalog can be found by searching for a zero key.

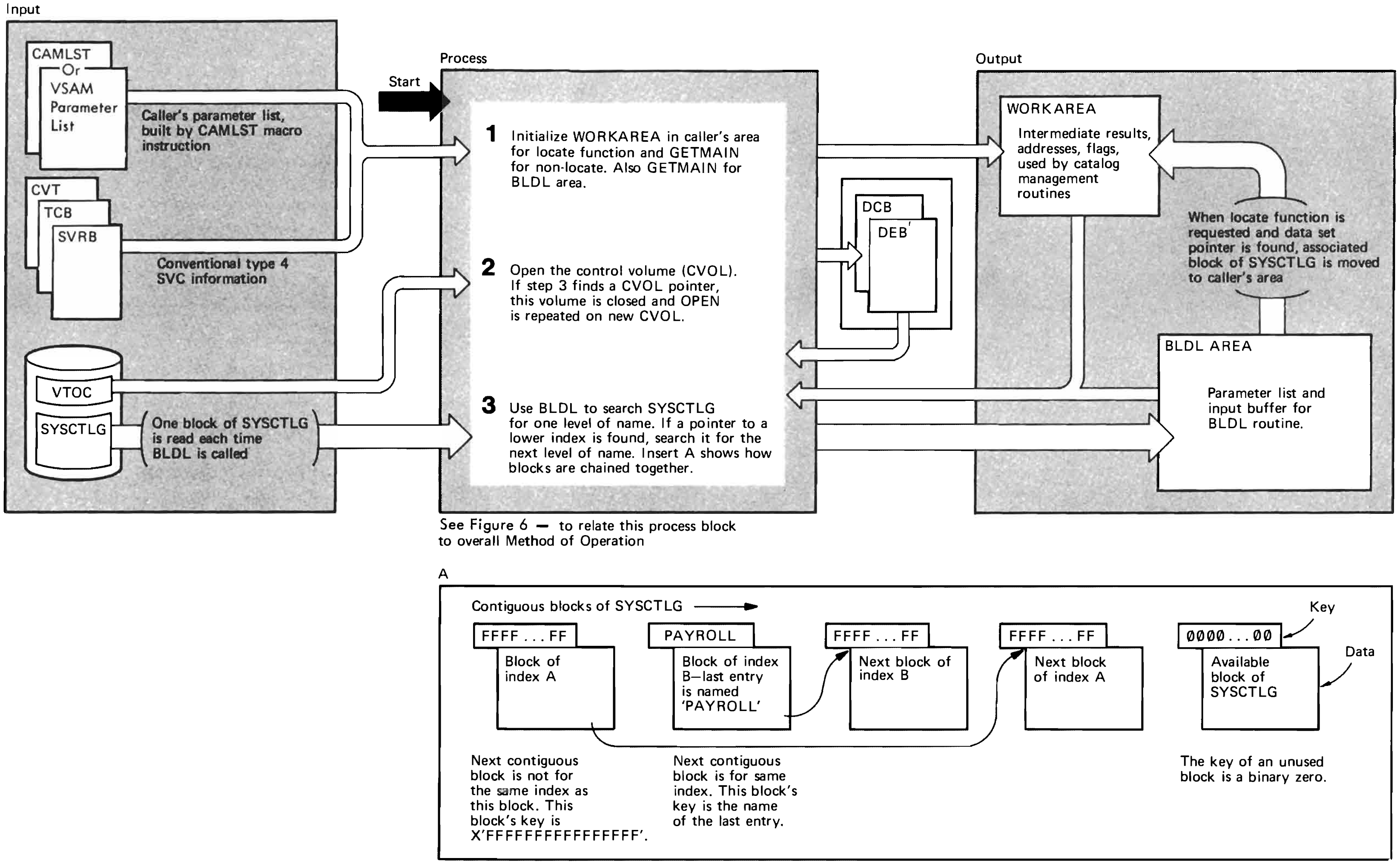

Input
CAMLST
Or
VSAM
Parameter
List
Caller passes parameters— function to perform, data set name, address and work area.
CVT
TCB
SVRB
Three supervisor control blocks are used by catalog management
SYSCTLG
Previously existing catalog, data set SYSCTLG
Entered Via SVC 26
START
Process
Operation I—Read (starting at IGG0CLC0) (see Figure 7)
Operation II—Set-up (see Figure 8)
Operation III—Write (see Figure 9)
Return
Locate Functions
Return
Non-locate Functions
Locate Functions Only
Non-locate Functions Only
Output
Volume Identification for volume(s) that contain the requested data set
Volume-list block for the located data set is returned in the caller's work area
SYSCTLG
Updated catalog data set



Notes:

- CAMLST is described on page 67
- SYSCTLG entries are described beginning on page 61
- Figure 7 is found on page 20
- Figure 8 is found on page 22
- Figure 9 is found on page 23
- CVT, TCB, and SVRB are described in *OS Control Blocks*, GC28-6628.

Figure 6. Catalog Management Overview


Input
CAMLST
Or
VSAM Parameter List
Caller's parameter list, built by CAMLST macro instruction
CVT
TCB
SVRB
Conventional type 4 SVC information
VTOC
SYSCTLG
One block of SYSCTLG is read each time BLDL is called
Start
Process
Output
WORKAREA
Intermediate results, addresses, flags, used by catalog management routines
DCB
DEB
When locate function is requested and data set pointer is found, associated block of SYSCTLG is moved to caller's area
BLDL AREA
Parameter list and input buffer for BLDL routine.
A
Contiguous blocks of SYSCTLG
Key
Data
FFFF . . . FF
Block of index A
PAYROLL
Block of index B—last entry is named 'PAYROLL'
FFFF . . . FF
Next block of index B
FFFF . . . FF
Next block of index A
0000 . . . 00
Available block of SYSCTLG
Next contiguous block is not for the same index as this block. This block's key is X'FFFFFFFFFFFFFFFF'.
Next contiguous block is for same index. This block's key is the name of the last entry.
The key of an unused block is a binary zero.

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

2 Open the control volume (CVOL). If step 3 finds a CVOL pointer, this volume is closed and OPEN is repeated on new CVOL.

3 Use BLDL to search SYSCTLG for one level of name. If a pointer to a lower index is found, search it for the next level of name. Insert A shows how blocks are chained together.

See Figure 6 — to relate this process block to overall Method of Operation

Notes:

- CAMLST is described on page 67
- WORKAREA is described on page 68
- BLDLAREA is described on page 72
- CVT, TCB, SVRB, VTOC, DCB, and DEB are described in OS/VS *System Data Areas*, SY28-0605.
- Figure 6 is found on page 19

- Operation I uses modules:

IGC0002F	pages 32 and 34
IGG0CLC0	page 40
IGG0CLC1	page 42
IGG0CLC2	page 44
IGG0CLC3	page 46

- Opening a CVOL uses modules:

IGC0002H	page 56
IGG0CLF2	page 58

Figure 7. Operation I—Read

Operation II, Setup

Figure 8 illustrates the setup operation. Input to this operation includes the name of an entry to be added or deleted from the catalog. This might be the simple name of a data set to be cataloged or removed from the catalog, or it might be an index name for a level to be created or deleted. Other information needed to build a new entry might be passed, depending on the request. The catalog block to which the change is to be made remains in a buffer from Operation I.

The setup process consists of either naming an entry to be deleted or constructing a new entry, then copying the catalog block one entry at a time from an input buffer to an output buffer. The new entry is merged into collating sequence with the old entries, or the named entry is bypassed, depending on the request.

Adding or deleting entries causes the following entries in the index to be displaced. For example, adding to a full block can result in an "overflow" of entries. These overflow entries must be added to the next block in the index chain. The effect of a change can thus *ripple* down the index chain. Similarly, a deletion can result in rippling in the other direction.

Operation III, Write

Figure 9 illustrates the write operation. Input to the operation is an updated catalog block and a relative track address to which this block is to be rewritten into SYSCTLG. Rippling can cause several blocks to be rewritten. Finally, the block containing the index control entry (ICE) and the volume index control entry (VICE) are updated and rewritten to reflect the changes made to the catalog. Output is the updated catalog.

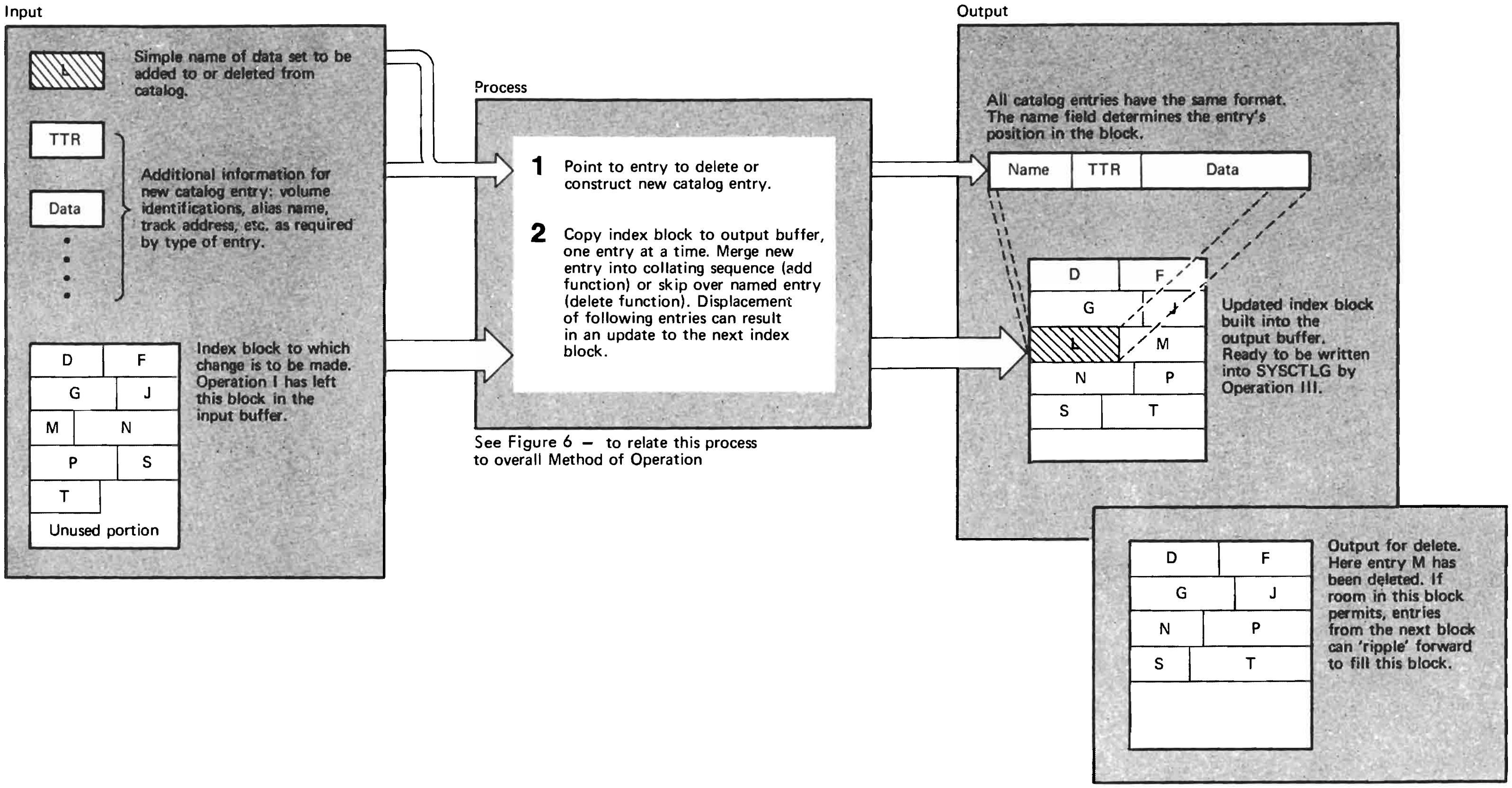
The insert to Figure 9 shows how the CATBX function is performed. In this case, the index levels to which a data set name is to be added do not exist; these index levels must also be created. This is done from the bottom up.

Step 1 shows the new data set pointer entry (DSPE) ready to be added to the catalog. A new index block is constructed and this DSPE is inserted into it. The newly constructed index block is written to the first available space in SYSCTLG, and the relative track address of that block is saved for step 2. The catalog is searched for another available space.

In step 2, a new index pointer entry (IPE) is created for the index block written in step 1, using the relative track address from step 1. If the next higher level of the index structure is also missing, another index block is created and written to SYSCTLG.

This process is repeated until a level is finally reached that does exist (step 3). An IPE is constructed for the highest new index and is added to the existing level.

For example, if the data set name A.B.C.DATASET were added to the catalog when none of the index levels A, B, or C exist, then step 1 would build index level C with a DSPE for A.B.C.DATASET in it. Step 2 would first build index level B with an IPE for C in it, and then index level A with an IPE for B in it. Finally, step 3 would insert an IPE for A into the volume index.


Input
L
Simple name of data set to be added to or deleted from catalog.
TTR
Data
Additional information for new catalog entry: volume identifications, alias name, track address, etc. as required by type of entry.
D
F
G
J
M
N
P
S
T
Unused portion
Index block to which change is to be made. Operation I has left this block in the input buffer.
Process
1 Point to entry to delete or construct new 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.
Output
All catalog entries have the same format. The name field determines the entry's position in the block.
Name
TTR
Data
D
F
G
J
L
M
N
P
S
T
Updated index block built into the output buffer. Ready to be written into SYSCTLG by Operation III.
D
F
G
J
N
P
S
T
Output for delete. Here entry M has been deleted. If room in this block permits, entries from the next block can 'ripple' forward to fill this block.

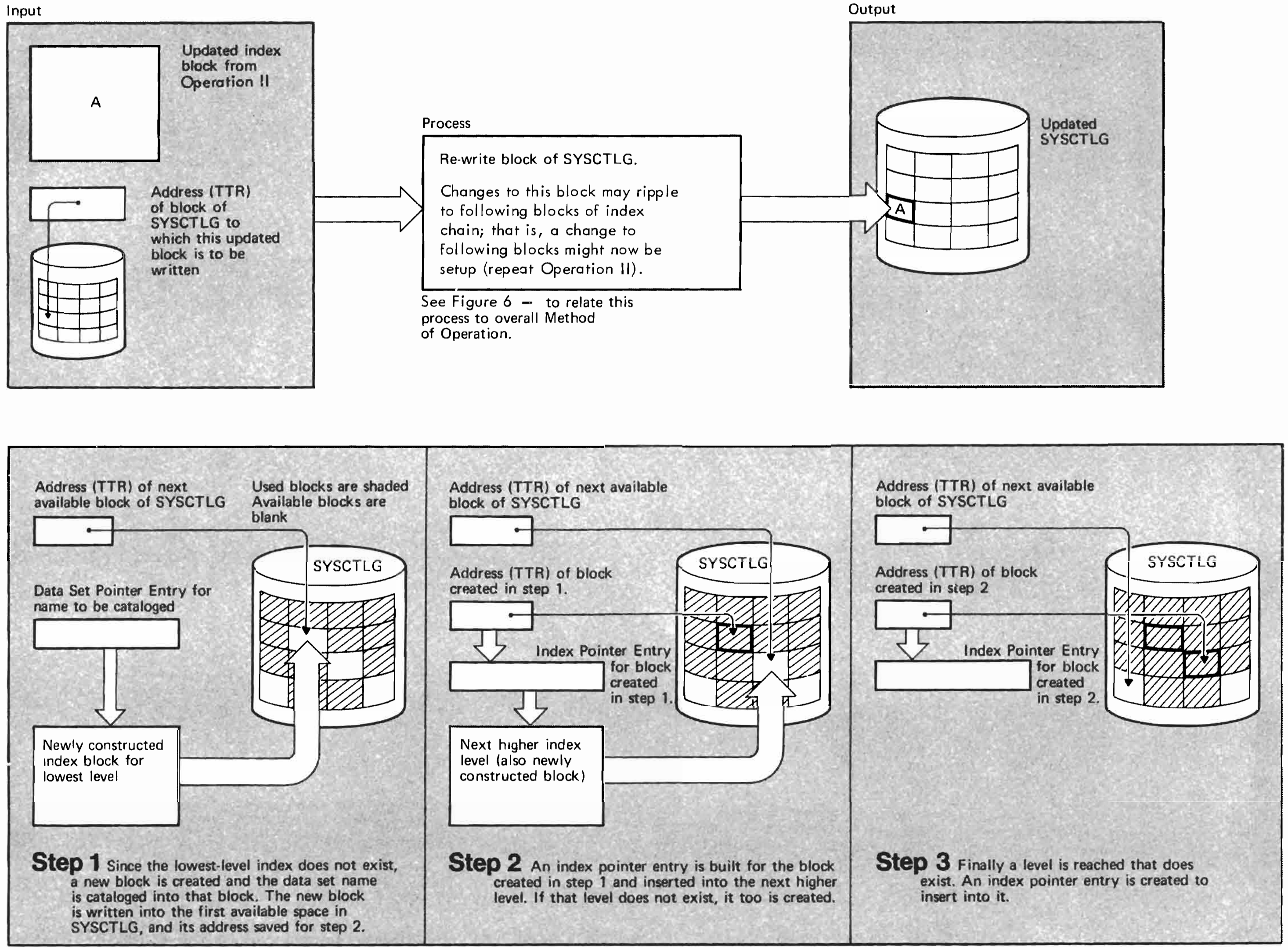
See Figure 6 — to relate this process to overall Method of Operation

Notes:

- SYSCTLG entries are described beginning on page 61
- BLDLAREA is described on page 72
- Figure 6 is found on page 19
- Operation II uses modules:

IGG0CLC3	page 46
IGG0CLC4	page 48
IGG0CLC5	page 50
IGG0CLC6	page 52

Figure 8. Operation II—Setup


Input
A
Updated index block from Operation II
Address (TTR) of block of SYSCTLG to which this updated block is to be written
Process
Output
Updated SYSCTLG
A
Address (TTR) of next available block of SYSCTLG
Used blocks are shaded Available blocks are blank
SYSCTLG
Data Set Pointer Entry for name to be cataloged
Newly constructed index block for lowest level
Address (TTR) of next available block of SYSCTLG
SYSCTLG
Address (TTR) of block created in step 1.
Index Pointer Entry for block created in step 1.
Next higher index level (also newly constructed block)
Address (TTR) of next available block of SYSCTLG
SYSCTLG
Address (TTR) of block created in step 2
Index Pointer Entry for block created in step 2.

Re-write block of SYSCTLG.

Changes to this block may ripple to following blocks of index chain; that is, a change to following blocks might now be setup (repeat Operation II).

See Figure 6 — to relate this process to overall Method of Operation.

Step 1 Since the lowest-level index does not exist, a new block is created and the data set name is cataloged into that block. The new block is written into the first available space in SYSCTLG, and its address saved for step 2.

Step 2 An index pointer entry is built for the block created in step 1 and inserted into the next higher level. If that level does not exist, it too is created.

Step 3 Finally a level is reached that does exist. An index pointer entry is created to insert into it.

Notes:

- Figure 6 is found on page 19.
- SYSCTLG entries are described beginning on page 61.
- Operation III uses modules:

  IGG0CLC3 page 46
  IGG0CLC5 page 50
  IGG0CLC6 page 52
  IGG0CLC7 page 54

Figure 9. Operation III—Write, with CATBX Detail

Program Organization

This chapter discusses the implementation of the catalog management routines, showing how the actual modules fit together.

The final authority of any program is the assembly or compilation listing. These module discussions complement the assembly-listing comments, and assume that the listings are at hand. You should have them available for any in-depth analysis.

Overall Organization

The three operations described in the previous chapter, "Method of Operation," form a conceptual image of catalog management. In reality, the broad operations of read, setup, and write merge together somewhat when packaged into modules. Still, the concept is useful, and it is continued in this chapter.

There are 11 or 13 modules in catalog management, depending on whether Controller I or Controller II is present. Eight modules and the controller form three phases, and two of the modules provide auxiliary services used by any phase.

Phase I equates to the read operation. It performs locate functions and the corresponding part of non-locate functions. Phase I consists of:

- Controller I or II, which routes the catalog management request to the appropriate routines. (This module does not exist in OS/VS1 Release 1.)
- IGG0CLC0, Initialization, which initializes work areas and opens the CVOL. (This module is named IGC0002F in OS/VS1 Release 1.)
- IGG0CLC1, Relative GDG and Alias, which resolves aliases and relative GDG numbers.
- IGG0CLC2, Locate, which searches the lower levels of the index structure.

Phase II begins the setup operation. It checks the validity of the requests against the existing entries in SYSCTLG, and it builds new entries to be added or it names entries to be deleted. Phase II consists of:

- IGG0CLC3, Update Initialization and Entry Building, which begins the update process by building new index blocks and routing the request as needed.
- IGG0CLC4, Entry Building, which builds data set pointer entries to add to the last valid level of the index.

Phase III completes the setup operation and performs the write operation. It merges entries into SYSCTLG blocks, deletes entries from blocks, and does most of the writing that is needed. Phase III consists of:

- IGG0CLC5, First Load of Update, which frees index blocks, frees volume control blocks (VCBs), and writes new VCBs.
- IGG0CLC6, Second Load of Update, which updates blocks, writes updated blocks to SYSCTLG, anu ripples the change as needed to the last block of the updated chain.
- IGG0CLC7, Third Load of Update and Error Handling, which writes the last updated block, updates the control entries, returns control to IGC0002F (to the caller, in OS/VS1 Release 1), and also handles error conditions.

The two service modules are:

- IGC0002H, SYSCTLG Open/Extend, which opens the SYSCTLG data set or gets the next extent of that data set, when needed.
- IGG0CLF2, SYSCTLG and BPAM Directory Formatter, which formats a new catalog and any new extents of a catalog.

Each of these modules operates in a transient area. IGC0002F, the entry point for catalog management, is invoked by a caller with an SVC 26; control is passed to the other eight modules of the phases via an XCTL macro instruction. IGC0002H, one of the service modules, is invoked via an SVC 28; it passes control to IGG0CLF2 via an XCTL macro instruction.

Figure 10 shows the organization of modules with relation to the phase structure and illustrates the overall flow of control from one module to another.

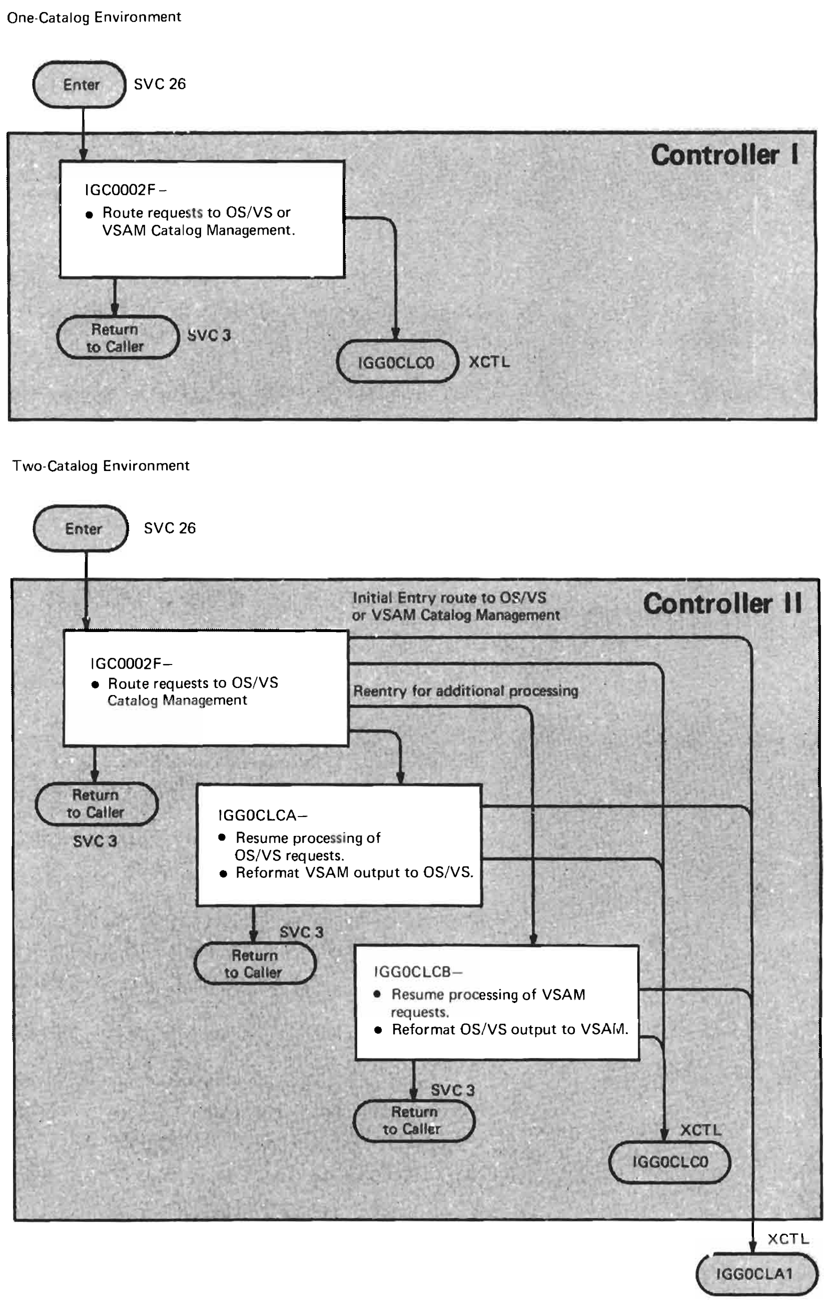

One-Catalog Environment
Enter
SVC 26
Controller I
IGC0002F -
• Route requests to OS/VS or VSAM Catalog Management.
Return to Caller
SVC 3
IGG0CLC0
XCTL
Two-Catalog Environment
Enter
SVC 26
Initial Entry route to OS/VS or VSAM Catalog Management
Controller II
IGC0002F–
• Route requests to OS/VS Catalog Management
Reentry for additional processing
Return to Caller
SVC 3
IGG0CLCA–
• Resume processing of OS/VS requests.
• Reformat VSAM output to OS/VS.
SVC 3
Return to Caller
IGG0CLCB–
• Resume processing of VSAM requests.
• Reformat OS/VS output to VSAM.
SVC 3
Return to Caller
XCTL
IGG0CLC0
XCTL
IGG0CLA1

Figure 10. (Part 1 of 2) Overall Program Organization

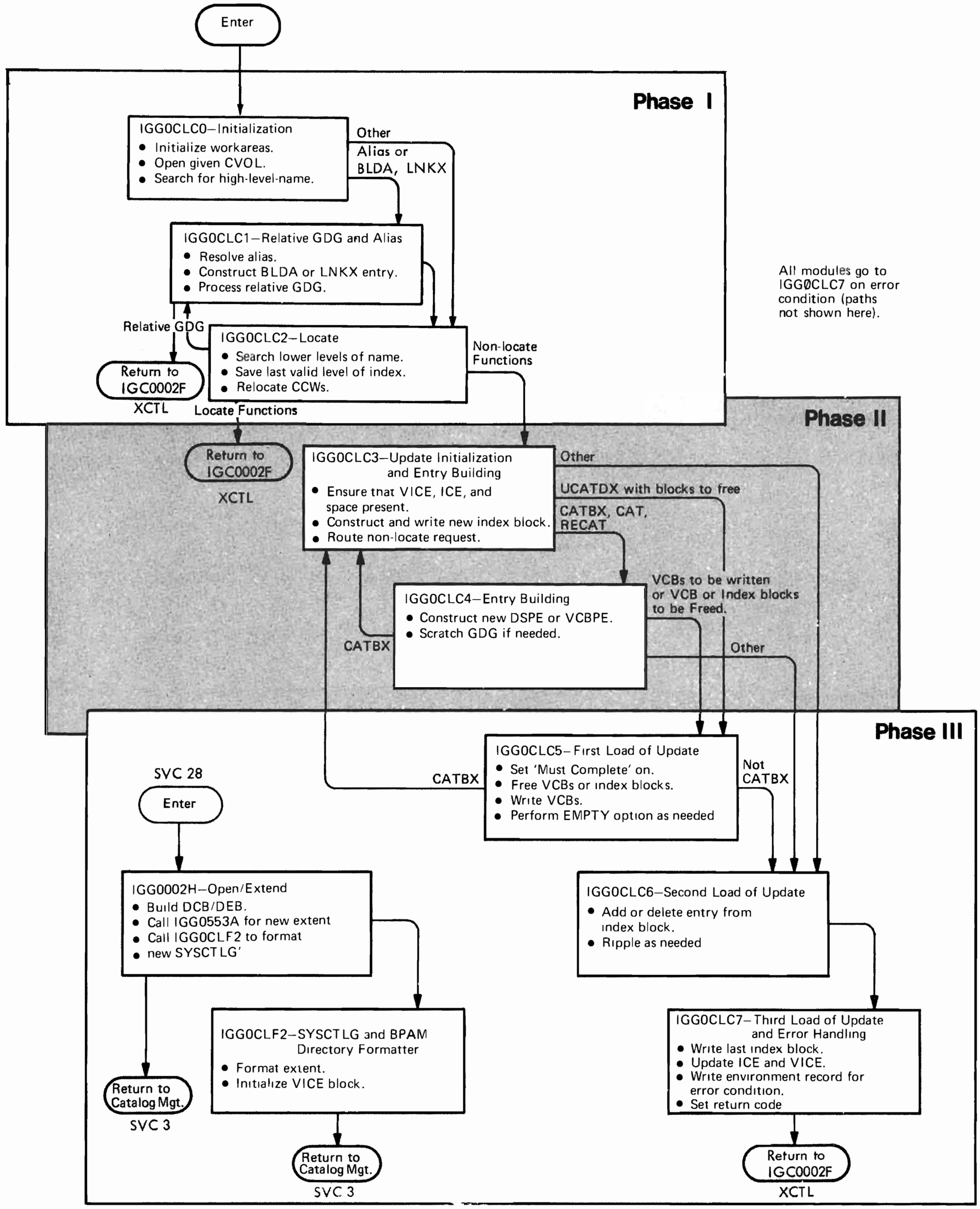


Figure 10. (Part 2 of 2) Overall Program Organization

An SVC 26 invokes the catalog management routines; control is then passed to IGC0002F, the catalog controller. The path that occurs through the remaining modules depends on both the particular function that is requested and the entries that are found in the catalog.

Each block in Figure 10 represents a module of the catalog management routines and contains a brief description of the functions that it performs. Each path is identified by the function/condition that it represents.

For example, follow the path for a BLDX function (request to create an index level) on a VS catalog. Assume that part of the index structure already exists; that is, this request extends an existing index structure.

Specifically, this is what each module does to accomplish the BLDX request:

- Entry to catalog management is at IGC0002F, which routes the request to IGG0CLC0.
- IGG0CLC0 initializes work areas, opens the CVOL, 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 to be added to that level.
- BLDX is a non-locate function, so control passes to IGG0CLC3. This module reads the control entries (VICE and ICE), constructs an index pointer entry (IPE), and routes the request via the arrow labeled "Other."
- IGG0CLC6 inserts the new IPE into the index block left by IGG0CLC2. IGG0CLC6 writes the updated block to SYSCTLG, 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 SYSCTLG.
- IGG0CLC7 writes the last block of the updated index chain, then reads, updates, and rewrites blocks containing the index control entry (ICE) and volume index control entry (VICE). Resources are released and control passes back to IGC0002F for return to the caller.

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

All of the VS catalog management modules are re-entrant. They use a common work space, initialized by IGG0CLC0 and called WORKAREA. (WORKAREA is described on page 68.)

Macros and Services Used

A small set of macros and services is used throughout the catalog management routines. The prologue commentary for each module lists the specific ones used in that module; the complete set is:

- BLDL, the routine used by Phase I to read blocks of SYSCTLG, which is formatted like a partitioned data set (PDS) directory. It is accessed through entry point IECPBLDL, whose address is found in field CVTPBLDL of the Communication Vector Table (CVT). In Phase I catalog management routines, this routine is used in a closed subroutine labeled "CALLBLDL".
- IECPCNVT, 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 CVT. In the catalog management routines, this routine is used in the closed subroutine labeled "TOABSL".
- IECPRLTV, 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 catalog management routines, this routine is used in a closed subroutine labeled "TORLTV".
- ENQ, a macro instruction used to control three resources: high-level name, volume index, and volume index control entry (VICE).

- DEQ, a macro instruction used to release resources obtained by the ENQ macro.
- EXCP, a macro instruction used for input/output operations in Phase II and Phase III modules. In the catalog management routines, this instruction is used in a closed subroutine labeled "IO".
- EXIT, a SVC instruction used to return to the caller.
- XCTL, a macro instruction used to pass control from one catalog management module to another.
- GETMAIN, a macro instruction used to allocate main storage. It is used by catalog management to obtain work areas and input/output buffers.
- WAIT, a macro instruction that allows an input/output operation to complete before continuing processing.
- FREEMAIN, a macro instruction used to release main storage used for work areas by catalog management.
- SCRATCH, a macro instruction used to delete a data set.
- DSECT macros IEFUCBOB and CVT, which describe system control blocks.

Character Dependency

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

Resource Enqueuing

Figure 11 shows when the resources of catalog management are enqueued in relation to the modules. 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.

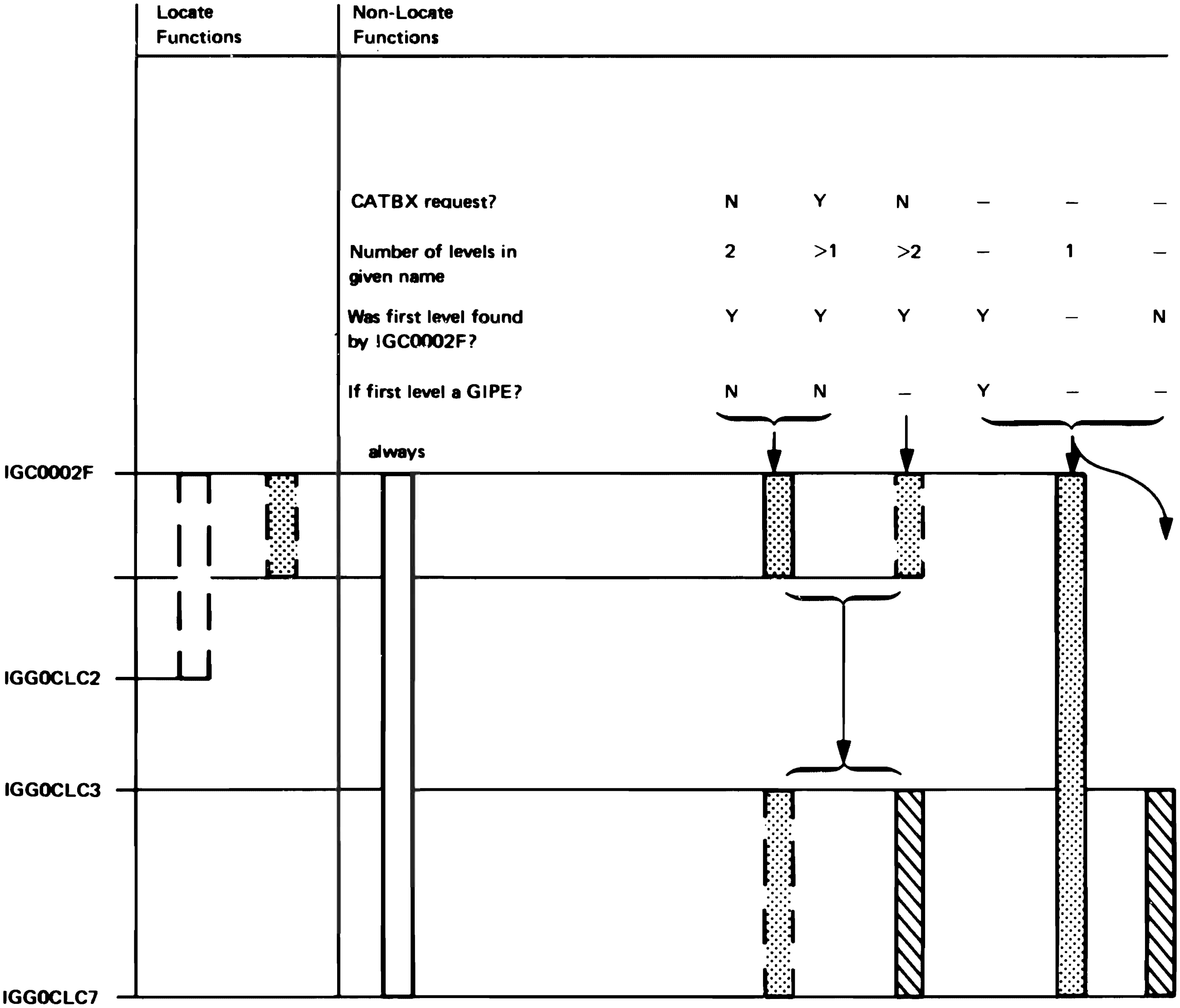
The major name for enqueuing is always "SYSCTLG". The minor name is the high-level name with the UCB address appended to it, "SYSCTLG" with the UCB address appended to it, or zeros with the UCB address appended to it.

Register Usage

With the exception of IGC0002H and IGG0CLF2, the catalog-management modules use a common set of registers. Module IGG0CLC0 initializes these registers, and their contents remain throughout. Contents of registers not described are considered destroyed.

Registers

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



CATBX request?	N	Y	N	–	–	–
Number of levels in given name	2	>1	>2	–	1	–
Was first level found by IGC0002F?	Y	Y	Y	Y	–	N
If first level a GIPE?	N	N	–	Y	–	–

Notes: "–" indicates "doesn't matter"

[‒ ‒]	Shared Enqueue
□	Exclusive Enqueue
□	High-level Name
░	Volume Index
▧	VICE

Order of enqueuing:
1. High-level Name
2. Volume Index
3. VICE (when applicable)

Missing combinations are error conditions

Figure 11. Resource Enqueuing

Module Descriptions

Each of the modules of catalog management is described in this section. The flowcharts are organized into two parts. Supporting text for the module appears beside each part.

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

IGC0002F: Controller I

Note: This module does not exist in OS/VS1 Release 1.

IGC0002F is the entry point for the catalog management routines, which are invoked by an SVC 26. IGC0002F is also entered from the following modules via XCTL, for return to the caller:

- IGG0CLC1
- IGG0CLC2
- IGG0CLC7

Registers

On Entry via SVC 26:

1 Address of caller's parameter list
3 Address of CVT
4 Address of SVRB

Functions

This module contains references to VSAM for planning purposes only. The module routes a request for catalog management to IGG0CLC0 without change when the caller is not VSAM. When the request is for a VSAM request, IGC0002F:

1. Translates the VSAM parameter list into CAMLST form and invokes normal catalog management routines (XCTL to IGG0CLC0).
2. Translates the results of the catalog search to VSAM format.

Internal Subroutines

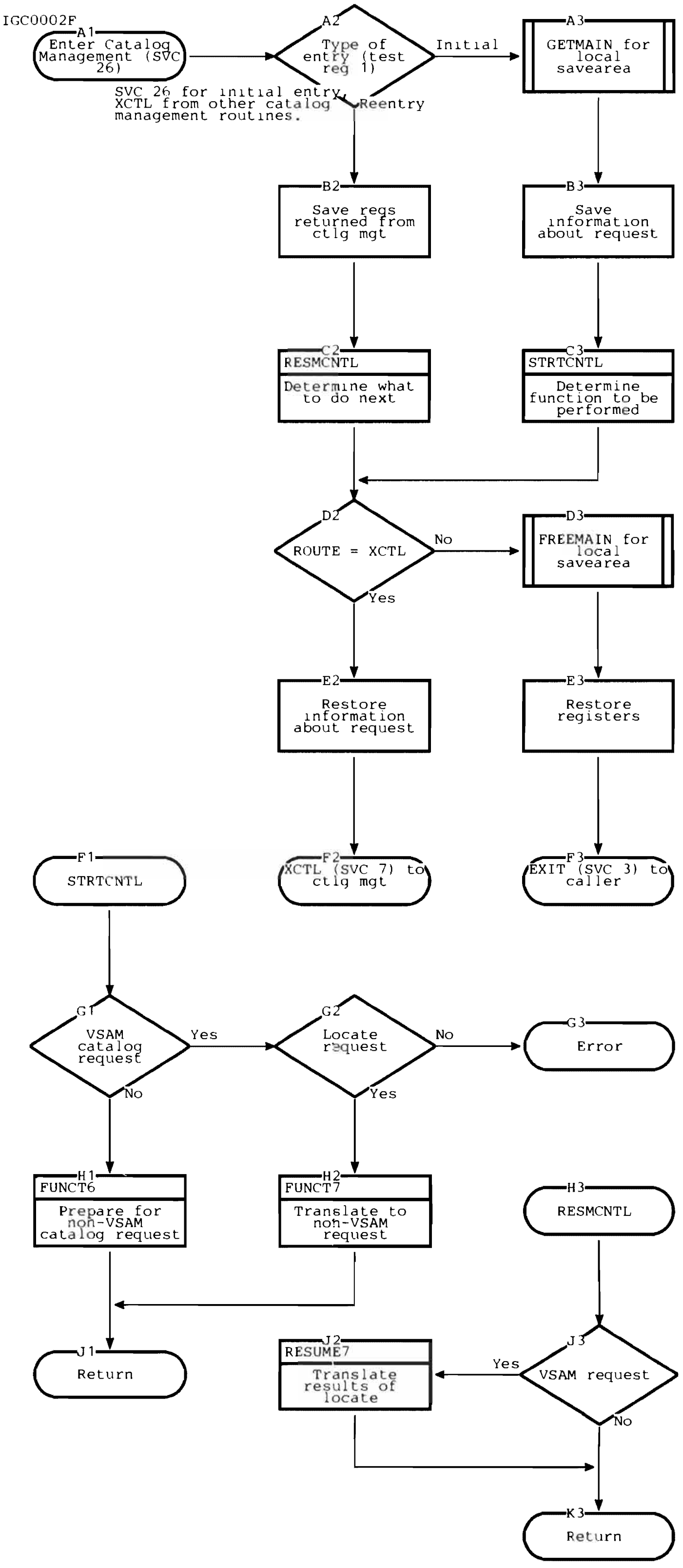
STRTCNTL handles the initial entry into IGC0002F by determining what functions are requested.

RESMCNTL handles subsequent re-entries of IGC0002F by determining if further functions are needed to satisfy the request.

FUNCT6 is used when the parameter list is already in the form of a CAMLST.

FUNCT7 is used when the parameter list is in VSAM form.

RESUME7 passes catalog information back to a VSAM caller, and controls further catalog searches when such is required.


IGC0002F
A1
Enter Catalog Management (SVC 26)
A2
Type of entry (test reg 1)
Initial
SVC 26 for initial entry, XCTL from other catalog management routines.
Reentry
A3
GETMAIN for local savearea
B2
Save regs returned from ctlg mgt
B3
Save information about request
C2
RESMCNTL
Determine what to do next
C3
STRTCNTL
Determine function to be performed
D2
ROUTE = XCTL
No
Yes
D3
FREEMAIN for local savearea
E2
Restore information about request
E3
Restore registers
F1
STRTCNTL
F2
XCTL (SVC 7) to ctlg mgt
F3
EXIT (SVC 3) to caller
G1
VSAM catalog request
Yes
No
G2
Locate request
No
Yes
G3
Error
H1
FUNCT6
Prepare for non-VSAM catalog request
H2
FUNCT7
Translate to non-VSAM request
H3
RESMCNTL
J1
Return
J2
RESUME7
Translate results of locate
J3
VSAM request
Yes
No
K3
Return

OSPL translates a VSAM parameter list into a CAMLST.

AMORD translates the results of a single catalog search into VSAM format.

CNTL2CAT handles cases when there are more than 20 volumes in the volume list.

AMORC converts a catalog management exception code to a VSAM.

FREERTN frees the work area that was obtained for a VSAM locate request.

Exits

Control passes via an XCTL macro instruction to IGG0CLC0.

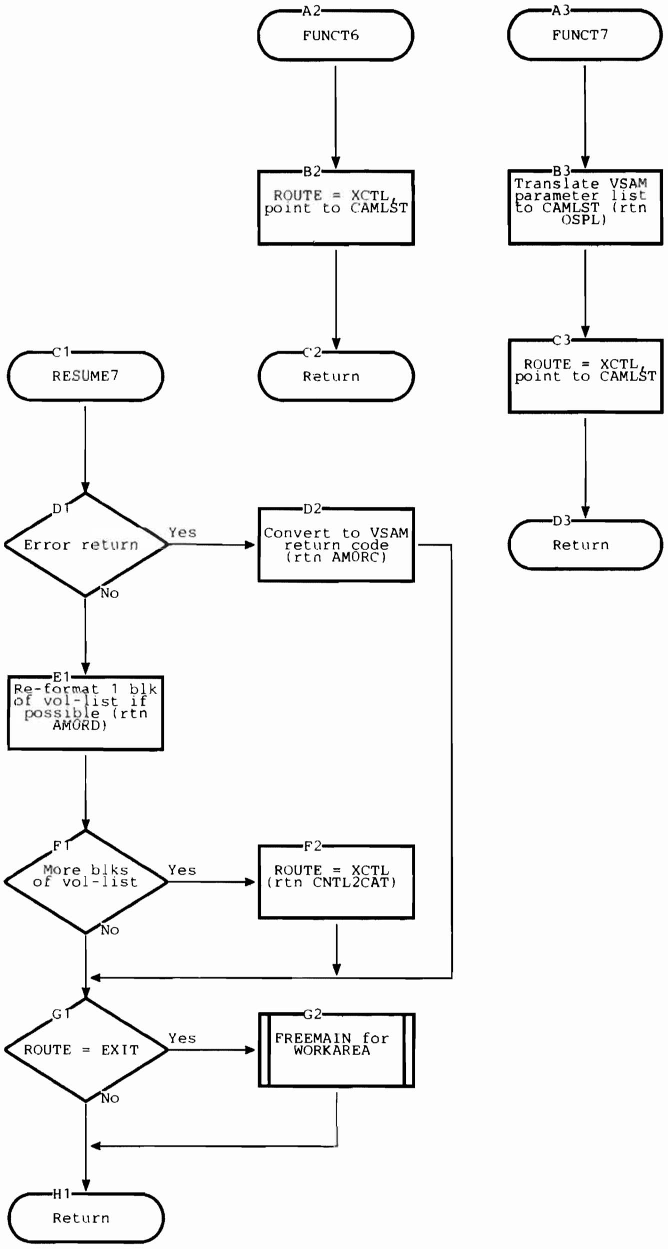
Error Conditions

Code	*Reason*
4	CVOL not mounted or does not contain SYSCTLG.
20	Syntax error in data set name.
24	Permanent input/output error.
28	Bad relative track address for SYSCTLG.
32	Bad address for caller's area.
36	Data set not found.
40	Output area provided by caller is too small.
48	Requested function is not supported.

References

Item	*Page number*
IGG0CLC0	40
CAMLSTD	67
Method of Operation Figure 7	20

CVT, TCB, SVRB, DCB, DEB, and UCB are described in *OS/VS1 System Data Areas*, SY28-0605, and in *OS/VS2 System Data Areas*, SY28-0606.



A2
FUNCT6
A3
FUNCT7
B2
ROUTE = XCTL, point to CAMLST
B3
Translate VSAM parameter list to CAMLST (rtn OSPL)
C1
RESUME7
C2
Return
C3
ROUTE = XCTL, point to CAMLST
D1
Error return
Yes
No
D2
Convert to VSAM return code (rtn AMORC)
D3
Return
E1
Re-format 1 blk of vol-list if possible (rtn AMORD)
F1
More blks of vol-list
Yes
No
F2
ROUTE = XCTL (rtn CNTL2CAT)
G1
ROUTE = EXIT
Yes
No
G2
FREEMAIN for WORKAREA
H1
Return

IGC0002F: Controller II

Note: This module exists only in OS/VS1 Release 2, and it exists *unless* you specify "VSAM=EXCLUDE" in the CTRLPROG macro at system generation.

IGC0002F is the entry point for the catalog management routines, which are invoked by an SVC 26. IGC0002F is also entered from the following modules:

- IGG0CLC1
- IGG0CLC2
- IGG0CLC7
- IGG0CLC9

Registers

On Entry via SVC 26 :

1	Address of caller's parameter list
3	Address of CVT
4	Address of TCB
5	Address of SVRB

On Re-entry via XCTL :

1	Negative number
2	Return code
12	Address of Controller's work area

Functions

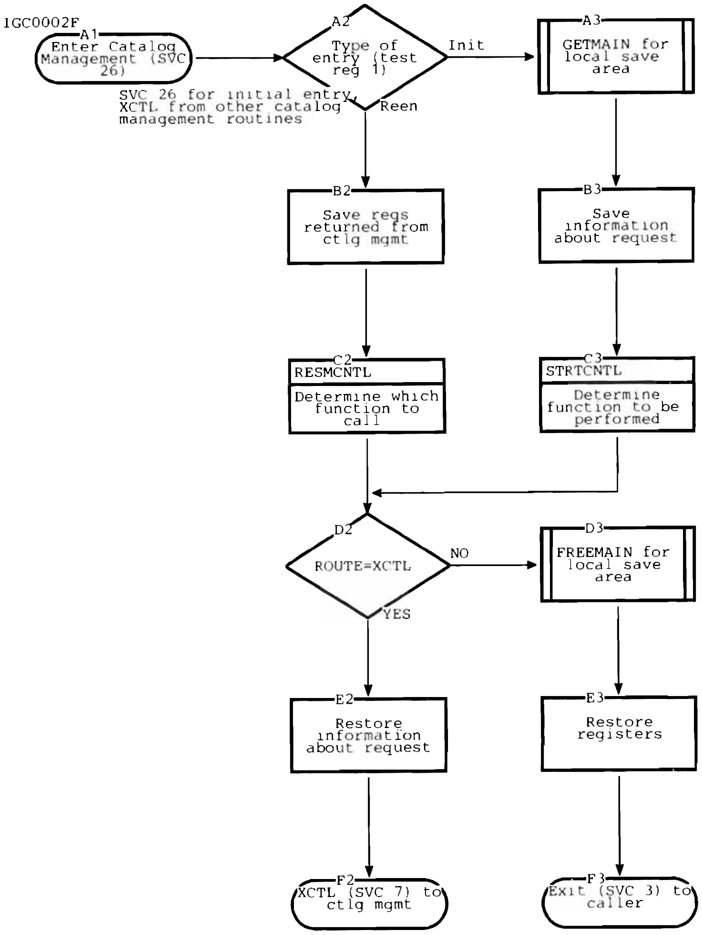
This module determines the type of request for catalog management and routes the request to the OS/VS (IGG0CLC0) or the VSAM (IGG0CLA1) catalog.

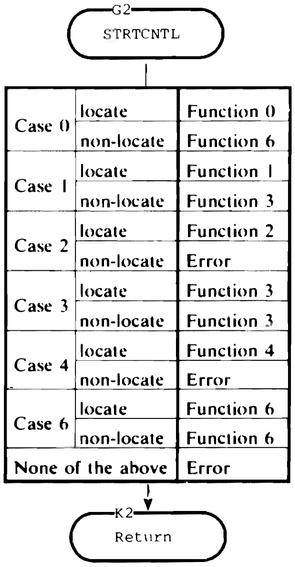
This module is also the entry point for all returns from either catalog. It determines the original type of request and routes to IGG0CLCA (original OS/VS requests) or IGG0CLCB (original VSAM requests), if additional processing is necessary.

The calling of initial and resume functions is based on the following:

	# of		*Start*
	Ctlgs	*Parm*	*Ctlg*
Case	*to Search*	*List*	*Search*
0	2	OS/VS	VSAM
1	2	VSAM	VSAM
2	1	OS/VS	VSAM
3	1	VSAM	VSAM
4	2	OS/VS	OS/VS
6	1	OS/VS	OS/VS

The case is determined by the decimal equivalent of the last three bits in CAMOPTN3 of the CAMLST (see page 67). Note that the CAMLST macro instruction sets these bits to zero, corresponding to case zero.


IGC0002F
A1
Enter Catalog Management (SVC 26)
SVC 26 for initial entry, XCTL from other catalog management routines
A2
Type of entry (test reg 1)
Init
Reen
A3
GETMAIN for local save area
B2
Save regs returned from ctlg mgmt
B3
Save information about request
C2
RESMCNTL
Determine which function to call
C3
STRTCNTL
Determine function to be performed
D2
ROUTE=XCTL
NO
YES
D3
FREEMAIN for local save area
E2
Restore information about request
E3
Restore registers
F2
XCTL (SVC 7) to ctlg mgmt
F3
Exit (SVC 3) to caller



Case 0	locate	Function 0
	non-locate	Function 6
Case 1	locate	Function 1
	non-locate	Function 3
Case 2	locate	Function 2
	non-locate	Error
Case 3	locate	Function 3
	non-locate	Function 3
Case 4	locate	Function 4
	non-locate	Error
Case 6	locate	Function 6
	non-locate	Function 6
None of the above		Error

Internal Subroutines

STRTCNTL handles the initial entry into IGG0002F to determine which function is required.

RESMCNTL handles subsequent entries into IGC0002F and routes to the appropriate module for addition processing.

FUNCT0 routes OS/VS requests to the VSAM catalog for a two-catalog search.

FUNCT2 routes directly to the VSAM catalog.

FUNCT3 routes VSAM DEFINE requests to the OS/VS catalog for a duplication check before any action is taken. All other requests go directly to the VSAM catalog.

FUNCT4 routes directly to the OS/VS catalog.

FUNCT6 routes OS/VS CATLG requests to the VSAM catalog for a duplication check before any action is taken.

AMOPL translates a CAMLST into a VSAM parameter list.

OSPL translates a VSAM parameter list into a CAMLST.

For a description of the VSAM parameter list, see the chapter "Data Areas" in *OS/VS VSAM Logic, SY26-3805.*

Exits

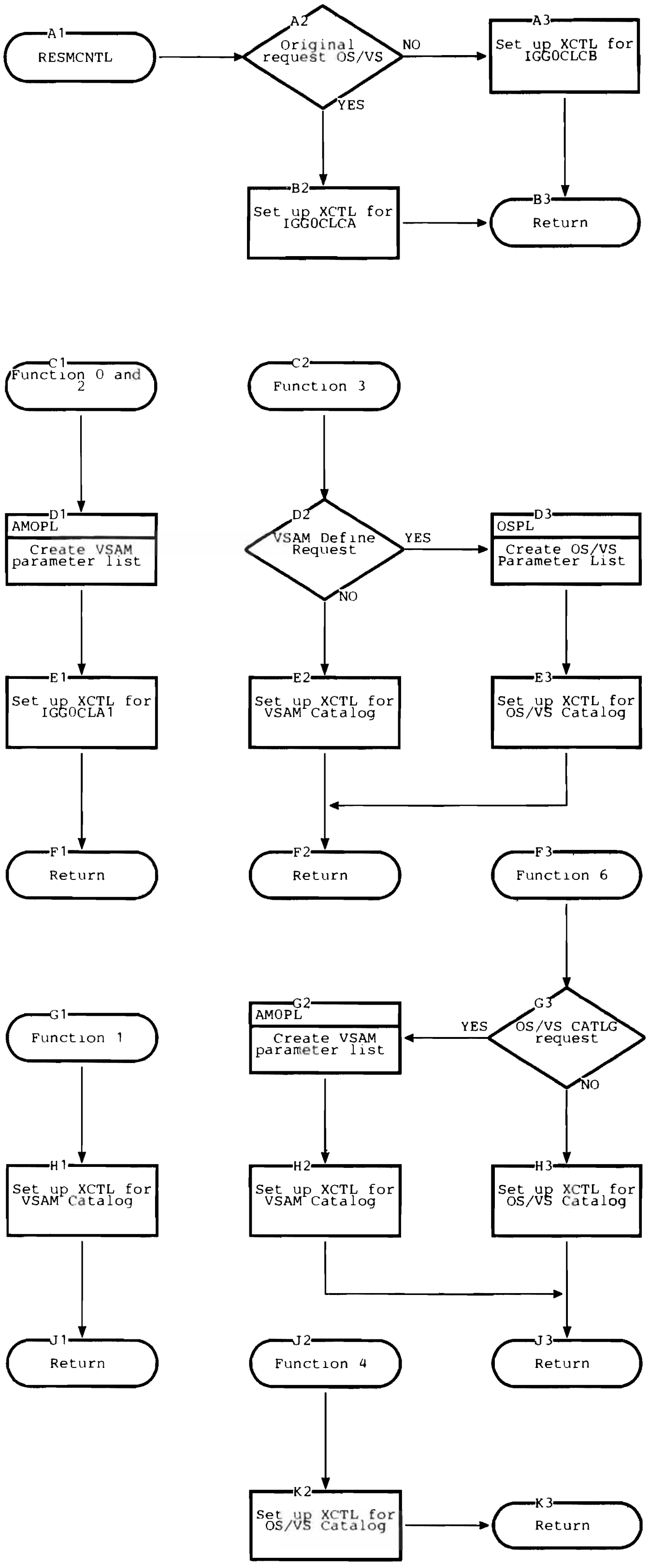
Control passes via an XCTL macro instruction to:

- IGG0CLCA
- IGG0CLCB
- IGG0CLC0
- IGG0CLA1

References

Item	*Page number*
CAMLSTD	67
IGG0CLCA	36
IGG0CLCB	38
IGG0CLC0	40
Method of Operation Figure 7	20

CVT, TCB, SVRB, DCB, DEB and UCB are described in *OS/VS1 System Data Areas,* SY28-0605, and in *OS/VS2 System Data Areas*, SY28-0606.



A1 RESMCNTL
A2 Original request OS/VS
NO
YES
A3 Set up XCTL for IGG0CLCB
B2 Set up XCTL for IGG0CLCA
B3 Return
C1 Function 0 and 2
C2 Function 3
D1 AMOPL Create VSAM parameter list
D2 VSAM Define Request
YES
NO
D3 OSPL Create OS/VS Parameter List
E1 Set up XCTL for IGG0CLA1
E2 Set up XCTL for VSAM Catalog
E3 Set up XCTL for OS/VS Catalog
F1 Return
F2 Return
F3 Function 6
G1 Function 1
G2 AMOPL Create VSAM parameter list
G3 OS/VS CATLG request
YES
NO
H1 Set up XCTL for VSAM Catalog
H2 Set up XCTL for VSAM Catalog
H3 Set up XCTL for OS/VS Catalog
J1 Return
J2 Function 4
J3 Return
K2 Set up XCTL for OS/VS Catalog
K3 Return

IGG0CLCA: Resume Processing of OS/VS Requests

IGG0CLCA is the entry point. Control comes from:

- IGC0002F after determining which catalog was originally requested.

Registers

12 Address of @DATD

Functions

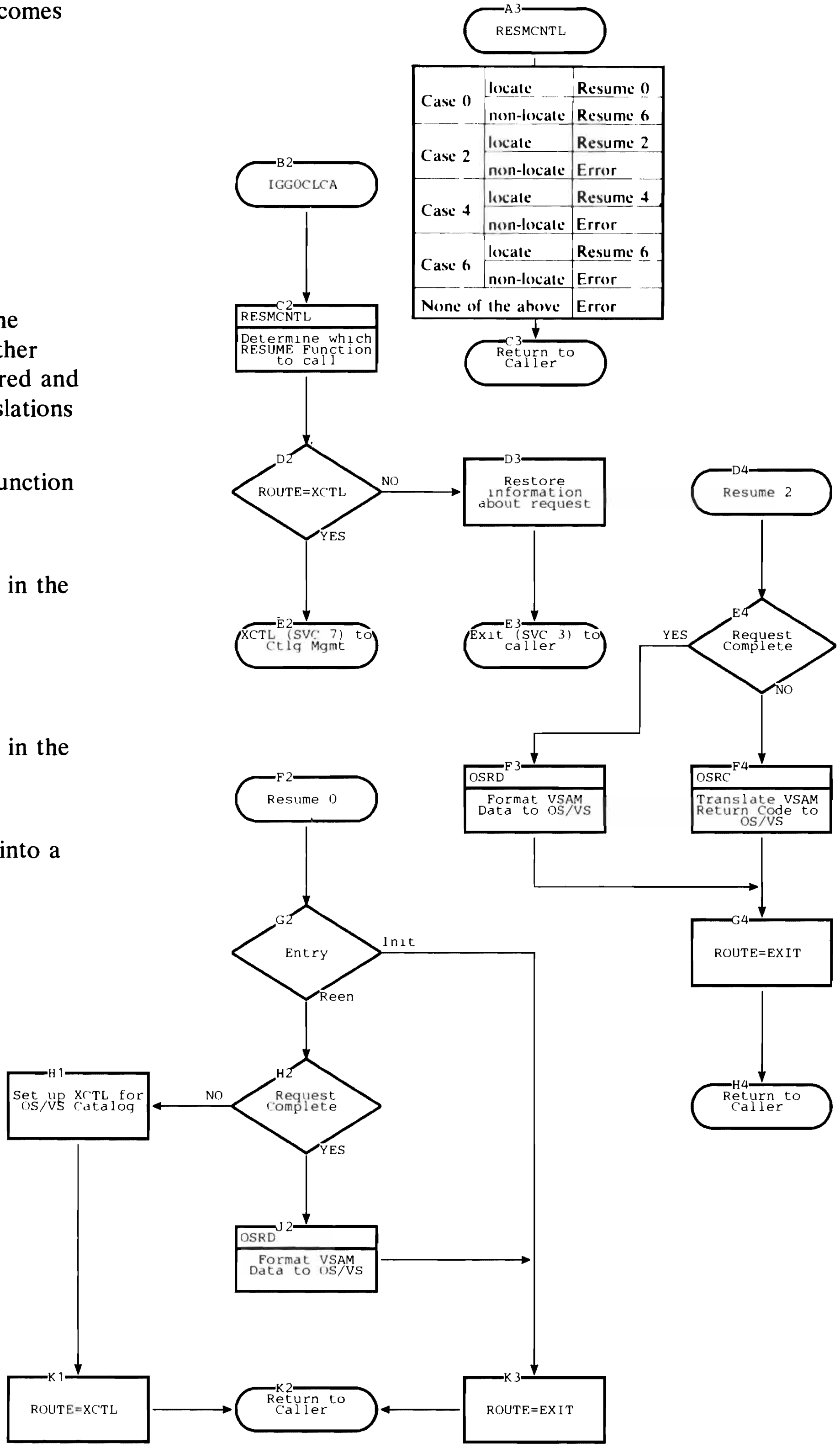
This module determines the status of the original request. It routes control to either catalog if additional processing is required and performs whatever conversions or translations are necessary.

RESMCNTL determines which resume function is required.

RESUME1 continues the processing of FUNCTION1 if the entry was not found in the VSAM catalog. If the entry was found, RESUME1 performs the necessary OS/VS-to-VSAM conversions.

RESUME3 continues the processing of FUNCTION3 if the entry was not found in the OS/VS catalog. If the entry was found, RESUME3 translates the return code.

OSPL translates a VSAM parameter list into a CAMLST.



Case 0	locate	Resume 0
	non-locate	Resume 6
Case 2	locate	Resume 2
	non-locate	Error
Case 4	locate	Resume 4
	non-locate	Error
Case 6	locate	Resume 6
	non-locate	Error
None of the above		Error

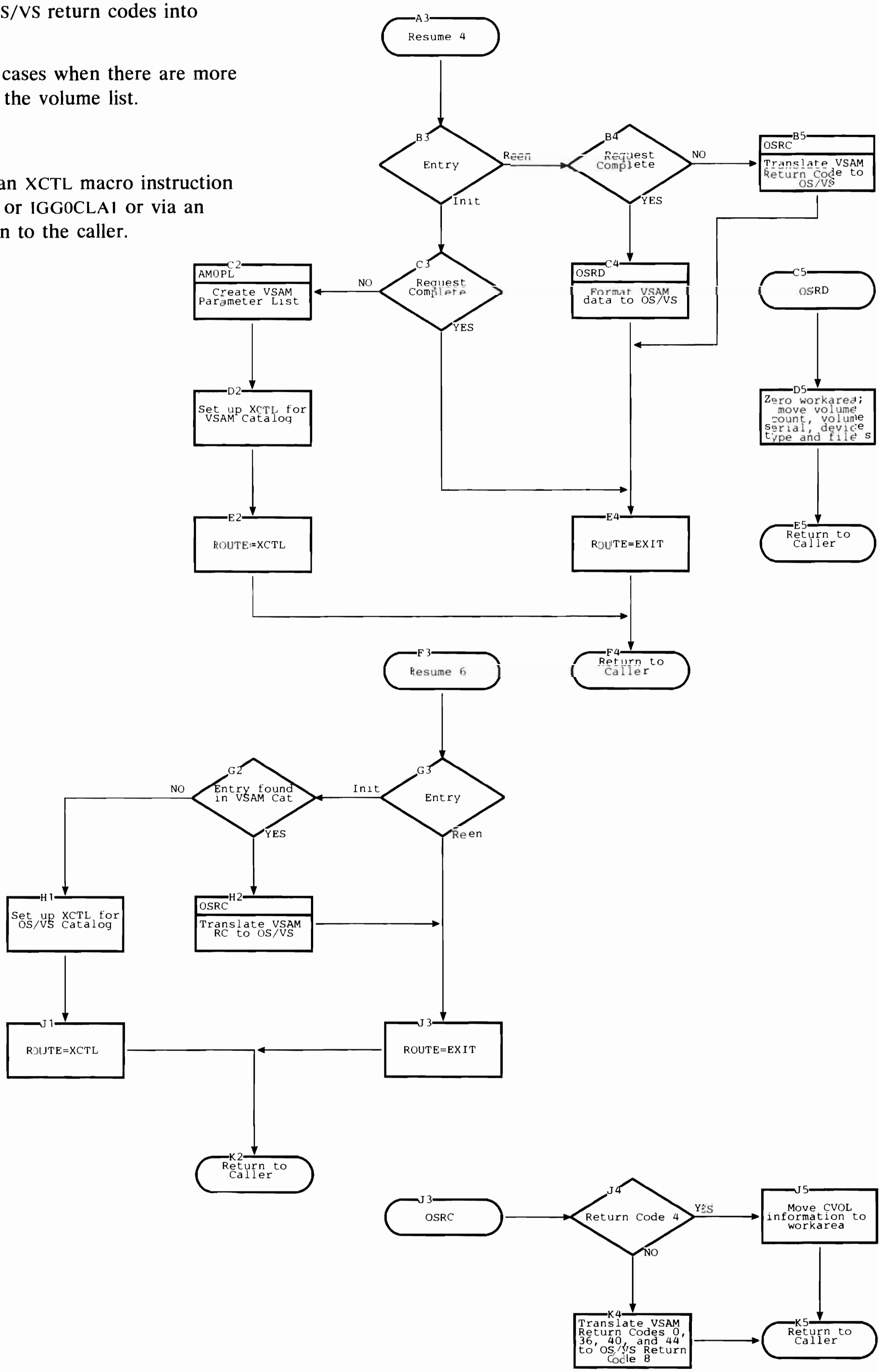
AMORD formats OS/VS information into the VSAM format.

AMORC translates OS/VS return codes into VSAM return codes.

CNTL2CAT handles cases when there are more than 20 volumes in the volume list.

Exits

Control passes via an XCTL macro instruction to either IGG0CLC0 or IGG0CLA1 or via an EXIT SVC instruction to the caller.



A3
Resume 4
B3
Entry
Reen
Init
B4
Request Complete
NO
YES
B5
OSRC
Translate VSAM Return Code to OS/VS
C2
AMOPL
Create VSAM Parameter List
NO
C3
Request Complete
YES
C4
OSRD
Format VSAM data to OS/VS
C5
OSRD
D2
Set up XCTL for VSAM Catalog
D5
Zero workarea; move volume count, volume serial, device type and file s
E2
ROUTE=XCTL
E4
ROUTE=EXIT
E5
Return to Caller
F3
Resume 6
F4
Return to Caller
G2
Entry found in VSAM Cat
NO
YES
Init
G3
Entry
Reen
H1
Set up XCTL for OS/VS Catalog
H2
OSRC
Translate VSAM RC to OS/VS
J1
ROUTE=XCTL
J3
ROUTE=EXIT
K2
Return to Caller
J3
OSRC
J4
Return Code 4
YES
NO
J5
Move CVOL information to workarea
K4
Translate VSAM Return Codes 0, 36, 40, and 44 to OS/VS Return Code 8
K5
Return to Caller

IGG0CLCB: Resume Processing of VSAM Requests

IGG0CLCB is the entry point. Control comes from:

- IGC0002F after determining which catalog was originally requested.

Registers

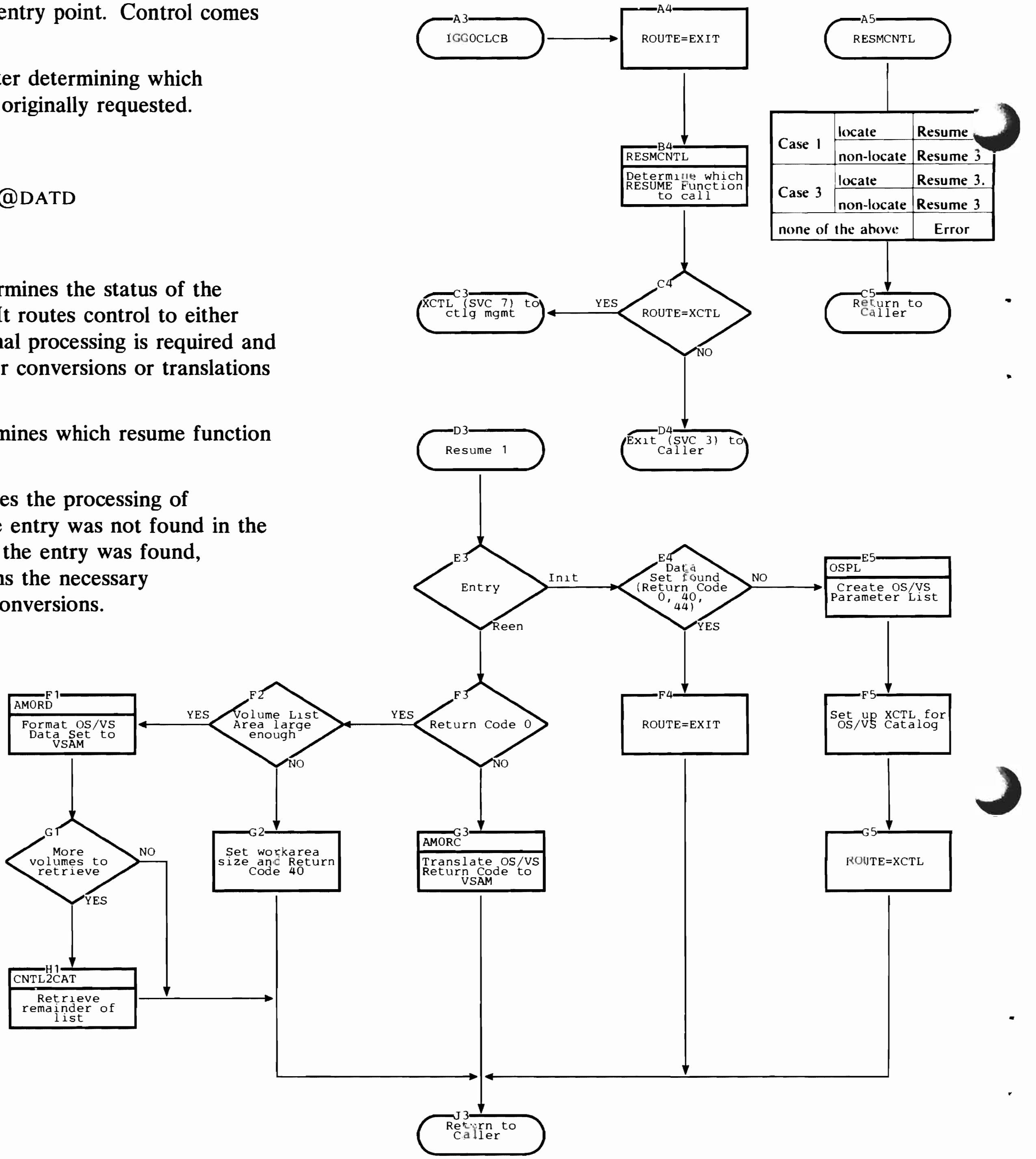
12 Address of @DATD

Functions

This module determines the status of the original request. It routes control to either catalog if additional processing is required and performs whatever conversions or translations are necessary.

RESMCNTL determines which resume function is required.

RESUME1 continues the processing of FUNCTION1 if the entry was not found in the VSAM catalog. If the entry was found, RESUME1 performs the necessary OS/VS-to-VSAM conversions.


A3
IGG0CLCB
A4
ROUTE=EXIT
A5
RESMCNTL
B4
RESMCNTL
Determine which RESUME Function to call
C3
XCTL (SVC 7) to ctlg mgmt
YES
C4
ROUTE=XCTL
NO
C5
Return to Caller
D3
Resume 1
D4
Exit (SVC 3) to Caller
E3
Entry
Init
Reen
E4
Data Set found (Return Code 0, 40, 44)
NO
YES
E5
OSPL
Create OS/VS Parameter List
F1
AMORD
Format OS/VS Data Set to VSAM
YES
F2
Volume List Area large enough
YES
F3
Return Code 0
NO
F4
ROUTE=EXIT
F5
Set up XCTL for OS/VS Catalog
G1
More volumes to retrieve
NO
YES
G2
Set workarea size and Return Code 40
G3
AMORC
Translate OS/VS Return Code to VSAM
G5
ROUTE=XCTL
H1
CNTL2CAT
Retrieve remainder of list
J3
Return to Caller

Case 1	locate	Resume
	non-locate	Resume 3
Case 3	locate	Resume 3.
	non-locate	Resume 3
none of the above		Error

RESUME3 continues the processing of FUNCTION3 if the entry was not found in the OS/VS catalog. If the entry was found, RESUME3 translates the return code.

OSPL translates a VSAM parameter list into a CAMLST.

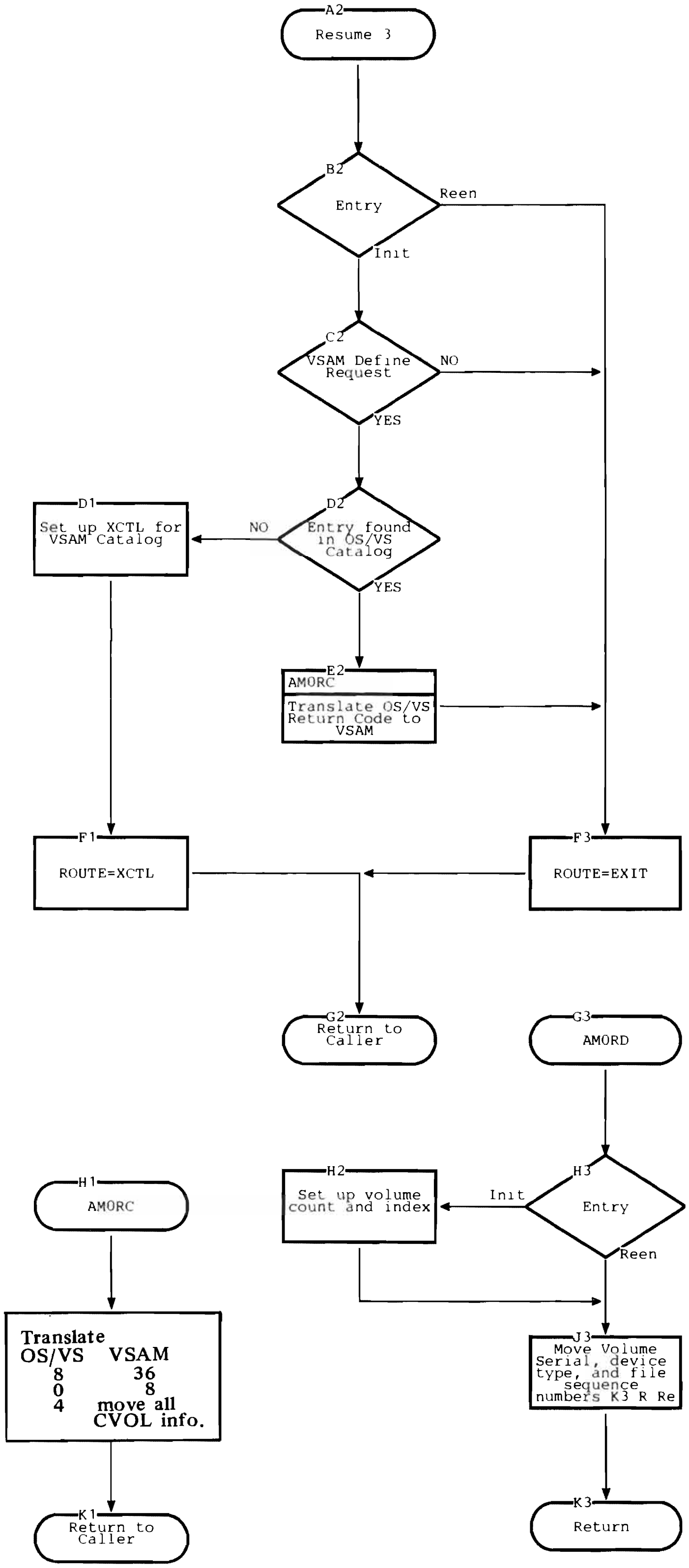
AMORD formats OS/VS information into the VSAM format.

AMORC translates OS/VS return codes into VSAM return codes.

CNTL2CAT handles cases when there are more than 20 volumes in the volume list.

Exits

Control passes via an XCTL macro instruction to either IGG0CLC0 or IGG0CLA1 or via an EXIT SVC instruction to the caller.



IGG0CLC0: Initialization

Note: This module is named IGC0002F in OS/VS1 Release 1.

IGG0CLC0 is the entry point. Control comes from IGC0002F.

Registers

On Entry:

1 Address of caller's parameter list (CAMLST)
3 Address of CVT
4 Address of TCB
5 Address of SVRB

On Exit:

2 Address of UCB
4 Base register for this module
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

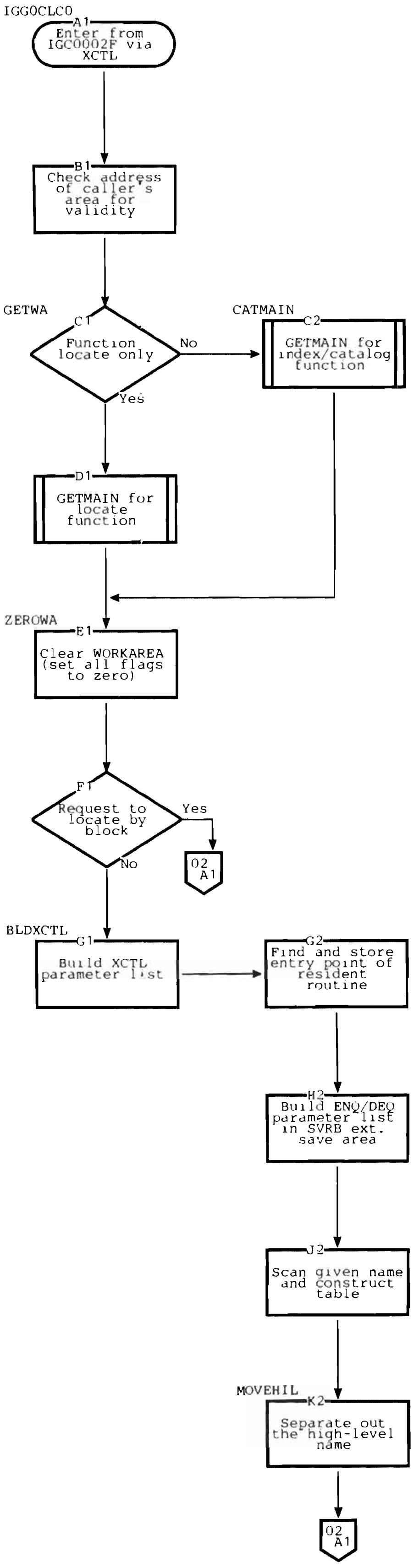
Functions

WORKAREA is the common workspace and communications area for all catalog management modules. 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 BLDL.

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 modules IGG0CLC0, IGG0CLC1, and IGG0CLC2. This 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 decomposed into its components. BLDLAREA is initialized for use as input/output buffers.

The UCB table is searched for device information about the given CVOL. GETMAIN is used to allocate space for a DCB and a DEB, and IGC0002H is used to open the CVOL (via an SVC 28).


IGG0CLC0
A1
Enter from IGC0002F via XCTL
B1
Check address of caller's area for validity
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
BLDXCTL
G1
Build XCTL parameter list
G2
Find and store entry point of resident routine
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

Note: The OPEN macro instruction is not used to open a CVOL. IGC0002H constructs a modified DCB/DEB for use by catalog management; IGC0002H is described on page 56. No CLOSE macro is issued to close a CVOL. 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 SYSCTLG. If BLDL returns a CVOL pointer entry (and the request is not LNKX or DRPX), the current CVOL is closed and control goes back to label UCBSRCH (block A1).

Internal Subroutines

None.

Exits

Control passes via an XCTL macro instruction to:

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

Control passes via an SVC 28 to IGC0002H to open the catalog, and returns to this module.

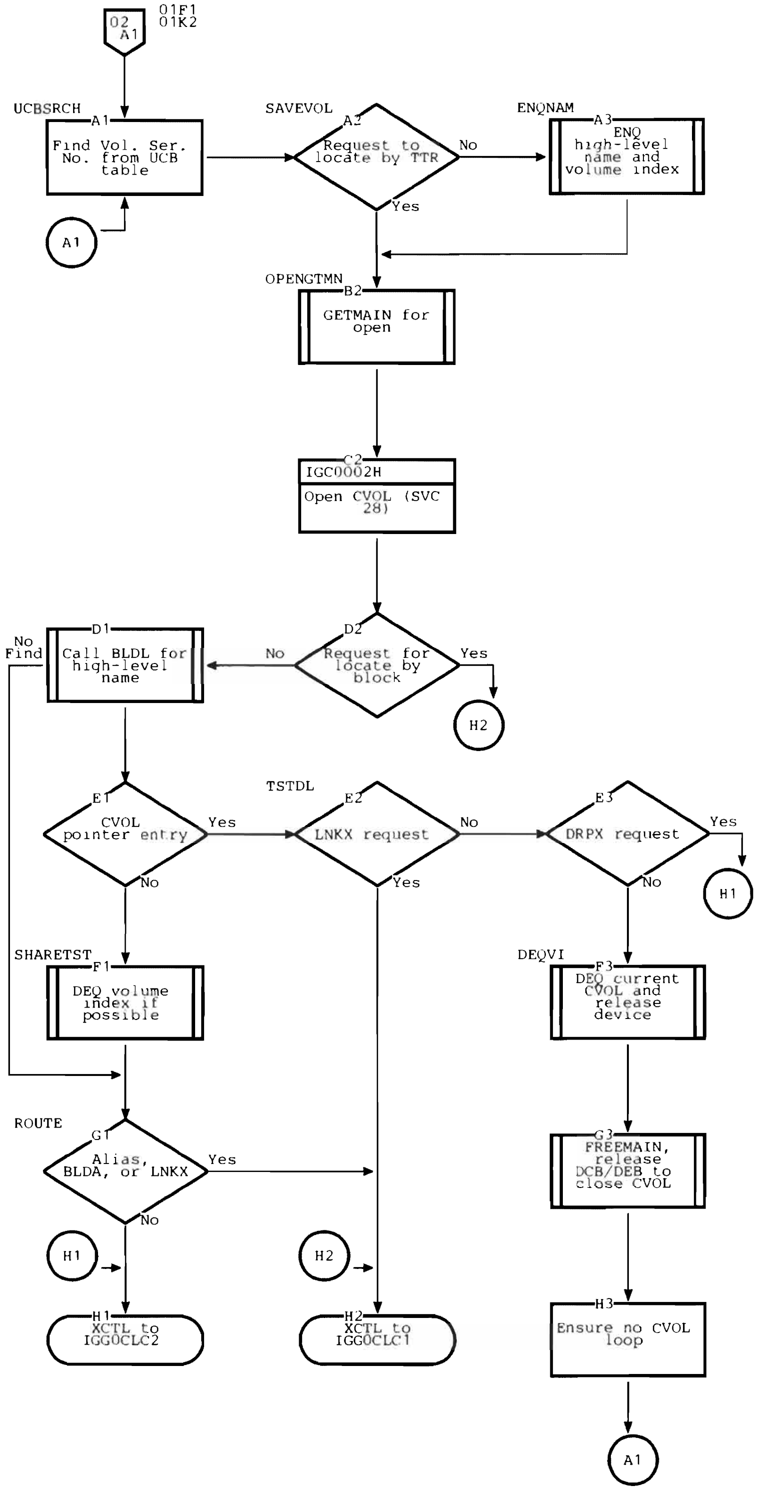
Error Conditions

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

References

Item	*Page number*
IGC0002F	32 and 34
IGG0CLC1	42
IGG0CLC2	44
IGG0CLC7	54
IGC0002H	56
SYSCTLG Entries	61
CAMLSTD	67
WORKAREA	68
BLDLAREA	72
Method of Operation Figure 7	20

CVT, TCB, SVRB, DCB, DEB, and UCB are described in *OS/VS1 System Data Areas*, SY28-0605, and *OS/VS2 System Data Areas*, SY28-0606.



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 module
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.

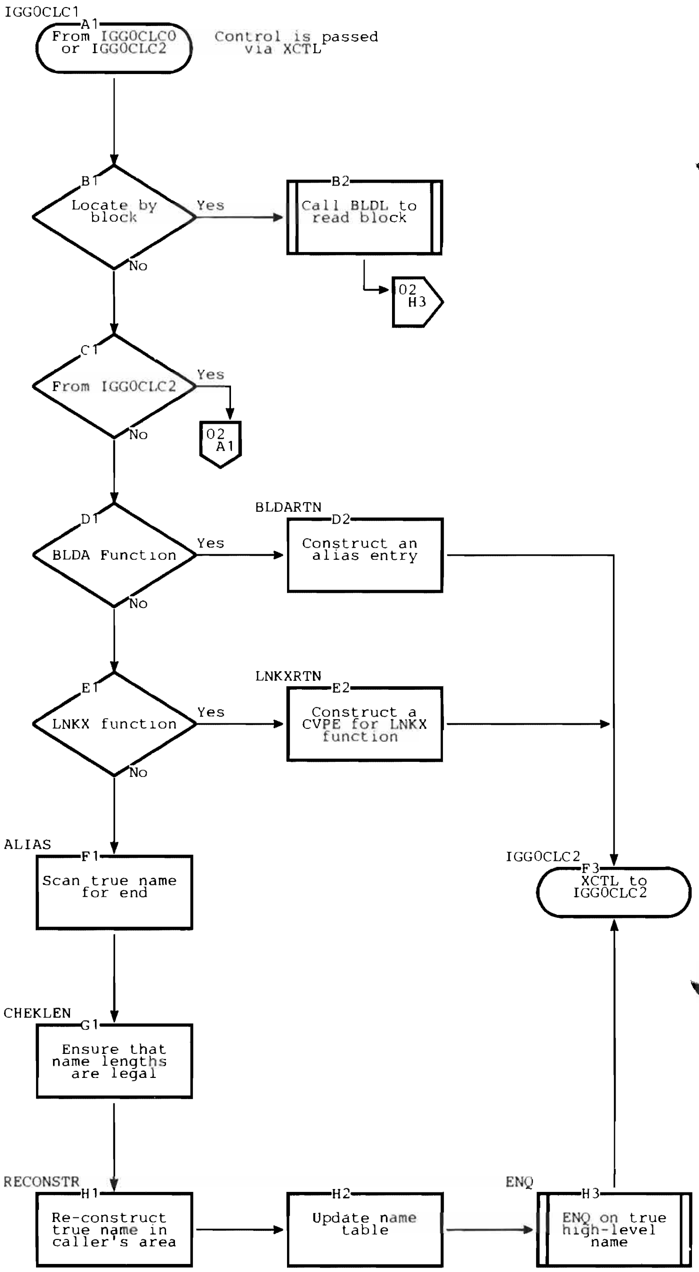
If the requested function is BLDA or LNKX, the appropriate entry is constructed and control passes to IGG0CLC2 for routine to the update modules.

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 then 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 module 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, etc.

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 G000nV00 (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 an XCTL macro instruction to:

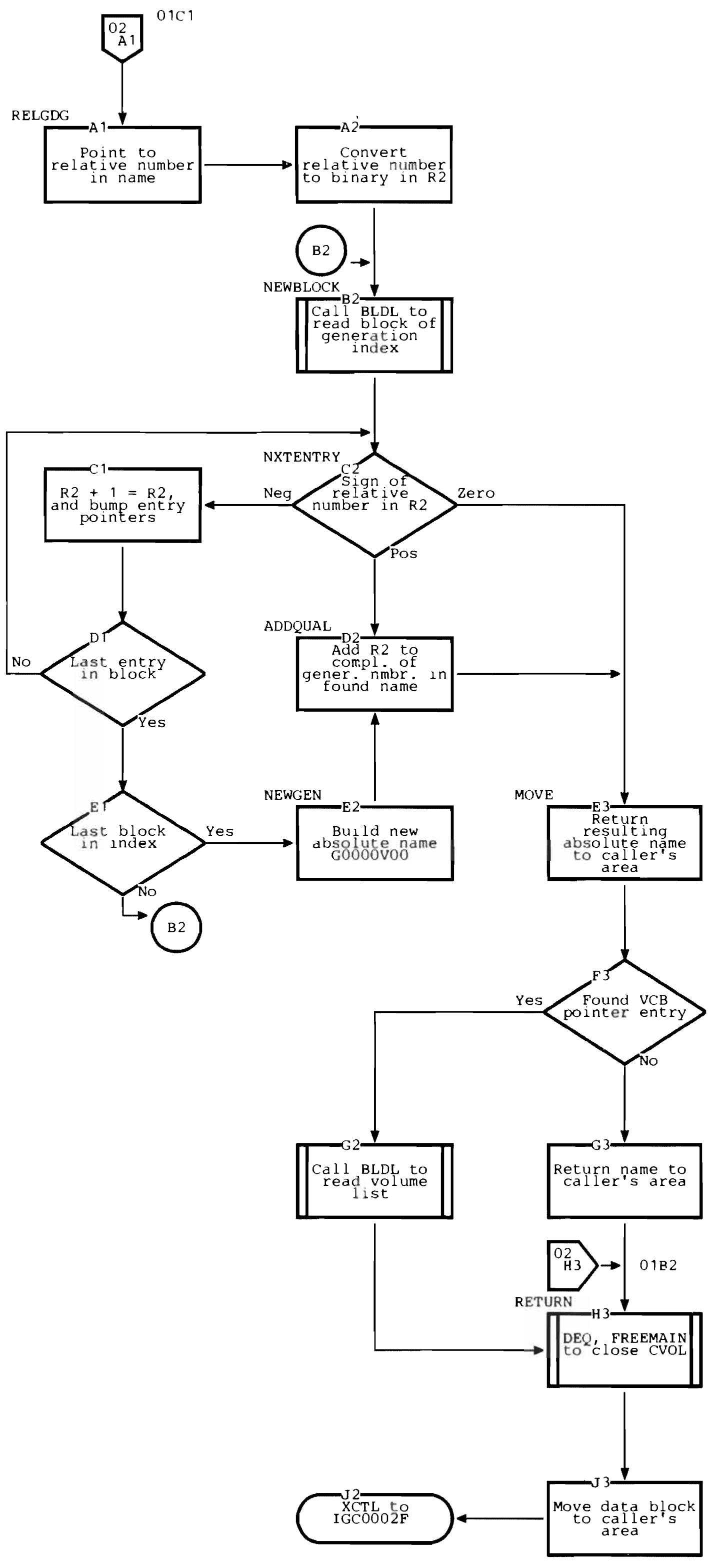
- IGC0002F after relative GDG processing.
- IGG0CLC7 for error conditions.
- IGG0CLC2 otherwise.

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.

References

Item	*Page number*
IGC0002F	32 and 34
IGG0CLC0	40
IGG0CLC2	44
IGG0CLC7	54
WORKAREA	68
Method of Operation Figure 7	20
Method of Operation Figure 8	22



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 compl. of gener. nmbr. in found name
NEWGEN
E2
Build new absolute name G0000V00
E1
Last block in index
Yes
No
B2
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
J3
Move data block to caller's area
J2
XCTL to IGC0002F

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 otherwise.

Registers

4	Base register for this module
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

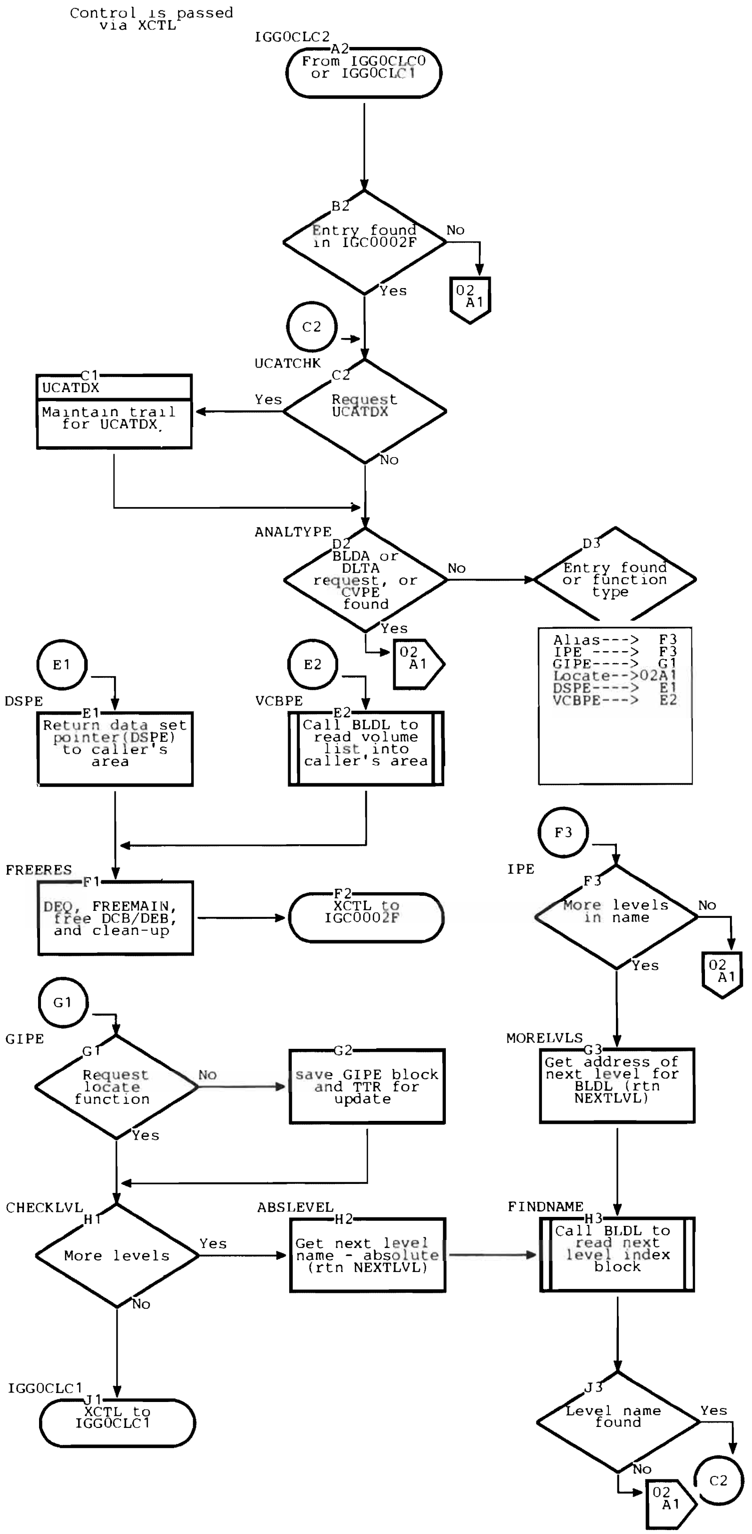
This module completes the locate functions, or finds the last valid index level for a non-locate function. BLDL is used to search index levels successively. At each index level, one component of the data set name is used. When locate-by-name is requested, BLDL is used with each component of the data set name as the search parameter. When BLDL returns an index pointer entry (IPE), use 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 modules. These CCW chains are moved to BLDLAREA.


Control is passed via XCTL
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
FREERES
F1
DEQ, FREEMAIN, free DCB/DEB, and clean-up
F2
XCTL to IGC0002F
F3
IPE
F3
More levels in name
No
02 A1
Yes
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
XCTL to IGG0CLC1
J3
Level name found
Yes
C2
No
02 A1

Internal Subroutines

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

CALLBLDL 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.

Exits

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

- IGC0002F along with the volume list for the data set name (via XCTL SVC instruction).
- IGG0CLC1 for relative GDG number.

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

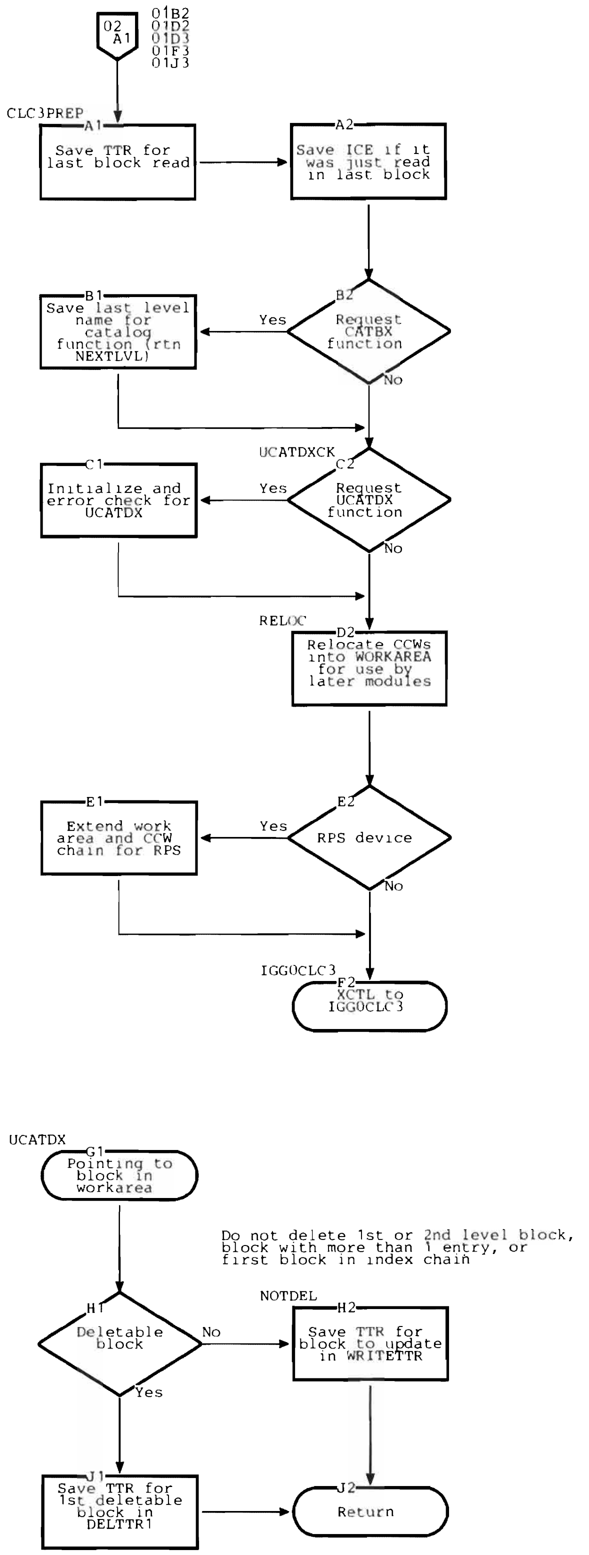
- IGG0CLC7 for an error condition.
- IGG0CLC3 otherwise.

Error Conditions

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

References

Item	*Page number*
IGC0002F	32 and 34
IGG0CLC0	40
IGG0CLC1	42
IGG0CLC3	46
IGG0CLC7	54
WORKAREA	68
BLDLAREA	72
Method of Operation Figure 7	20


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
XCTL 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
NOTDEL
H1
Deletable block
No
Yes
H2
Save TTR for block to update in WRITETTR
J1
Save TTR for 1st deletable block in DELTTR1
J2
Return

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 otherwise.

Registers

4	Base register for this module
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 SYSCTLG to ensure that there is enough space to make the required changes.

This module constructs new index levels for a BLDX 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 will delete it.

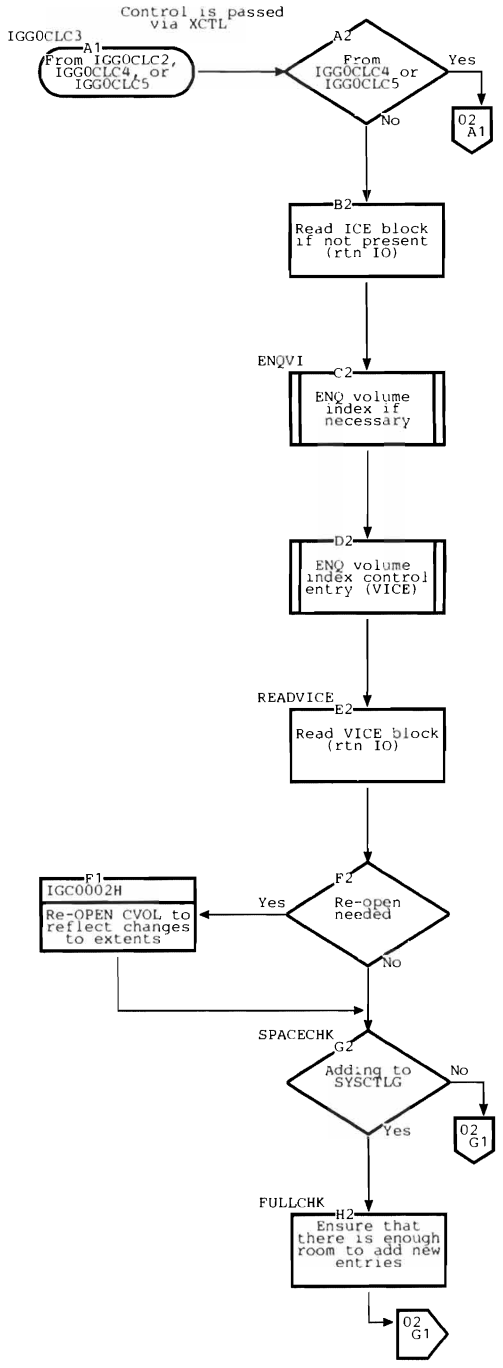
When CATBX is requested, IGG0CLC4 is called to construct the DSPE. Control returns to IGG0CLC3, where the required index levels are built and written into SYSCTLG. Each level results in an index pointer entry (IPE) that must be added to the next higher level. When an existing level is reached, control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

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

Internal Subroutines

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

WRTSRCH writes a new block to SYSCTLG and searches for another available block.


Control is passed via XCTL
IGG0CLC3
A1
From IGG0CLC2, IGG0CLC4, or IGG0CLC5
A2
From IGG0CLC4 or IGG0CLC5
Yes
No
02 A1
B2
Read ICE block if not present (rtn IO)
ENQVI
C2
ENQ volume index if necessary
D2
ENQ volume index control entry (VICE)
READVICE
E2
Read VICE block (rtn IO)
F1
IGC0002H
Re-OPEN CVOL to reflect changes to extents
F2
Re-open needed
Yes
No
SPACECHK
G2
Adding to SYSCTLG
No
02 G1
Yes
FULLCHK
H2
Ensure that there is enough room to add new entries
02 G1

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.

IO performs EXCP input/output. This subroutine invokes IGC0002H if a new extent of SYSCTLG is required.

Exits

Control passes via an XCTL macro instruction to:

- IGG0CLC4 when the requested function is CATBX, CAT, RECAT, or UNCAT.
- IGG0CLC5 when blocks of SYSCTLG need to be freed, or when new blocks have been written, but the requested prócess has been aborted.
- IGG0CLC7 for error conditions.
- IGG0CLC6 otherwise.

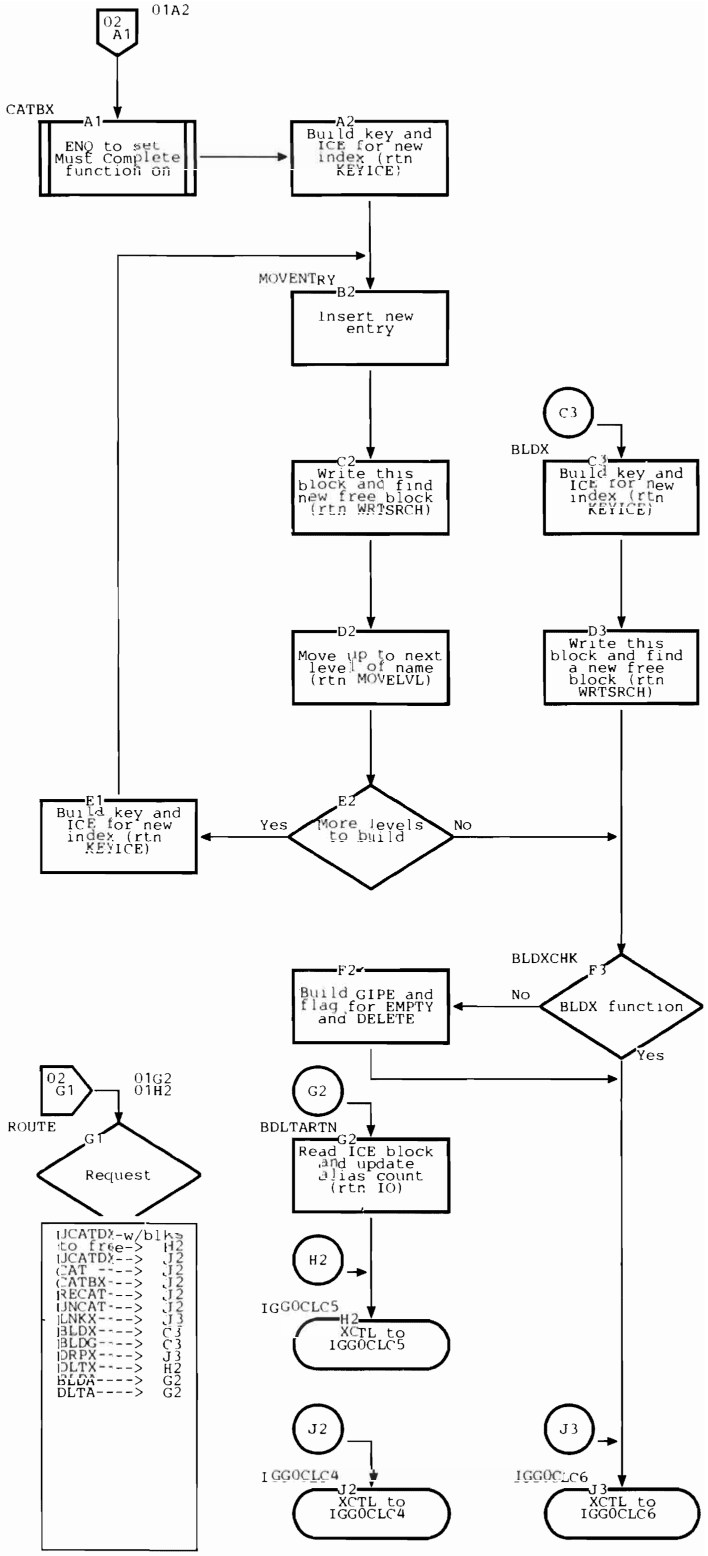
Control passes via an SVC 28 to IGC0002H when a new extent of SYSCTLG is required or when SYSCTLG must be re-opened, and returns to this module.

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 SYSCTLG to perform the requested function.
28	Permanent I/O error.

References

Item	*Page number*
IGG0CLC2	44
IGG0CLC4	48
IGG0CLC5	50
IGG0CLC6	52
IGG0CLC7	54
IGC0002H	56
SYSCTLG Entries	61
WORKAREA	68
BLDLAREA	72
Method of Operation Figure 8	22
Method of Operation Figure 9	23


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)
E2
More levels to build
Yes
No
BLDXCHK
F3
BLDX function
No
Yes
F2
Build GIPE and flag for EMPTY and DELETE
02 G1
01G2
01H2
ROUTE
G1
Request
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
G2
BDLTARTN
G2
Read ICE block and update alias count (rtn IO)
H2
IGG0CLC5
H2
XCTL to IGG0CLC5
J2
IGG0CLC4
J2
XCTL to IGG0CLC4
J3
IGG0CLC6
J3
XCTL to IGG0CLC6

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 module
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 module 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 options for GDG are processed.

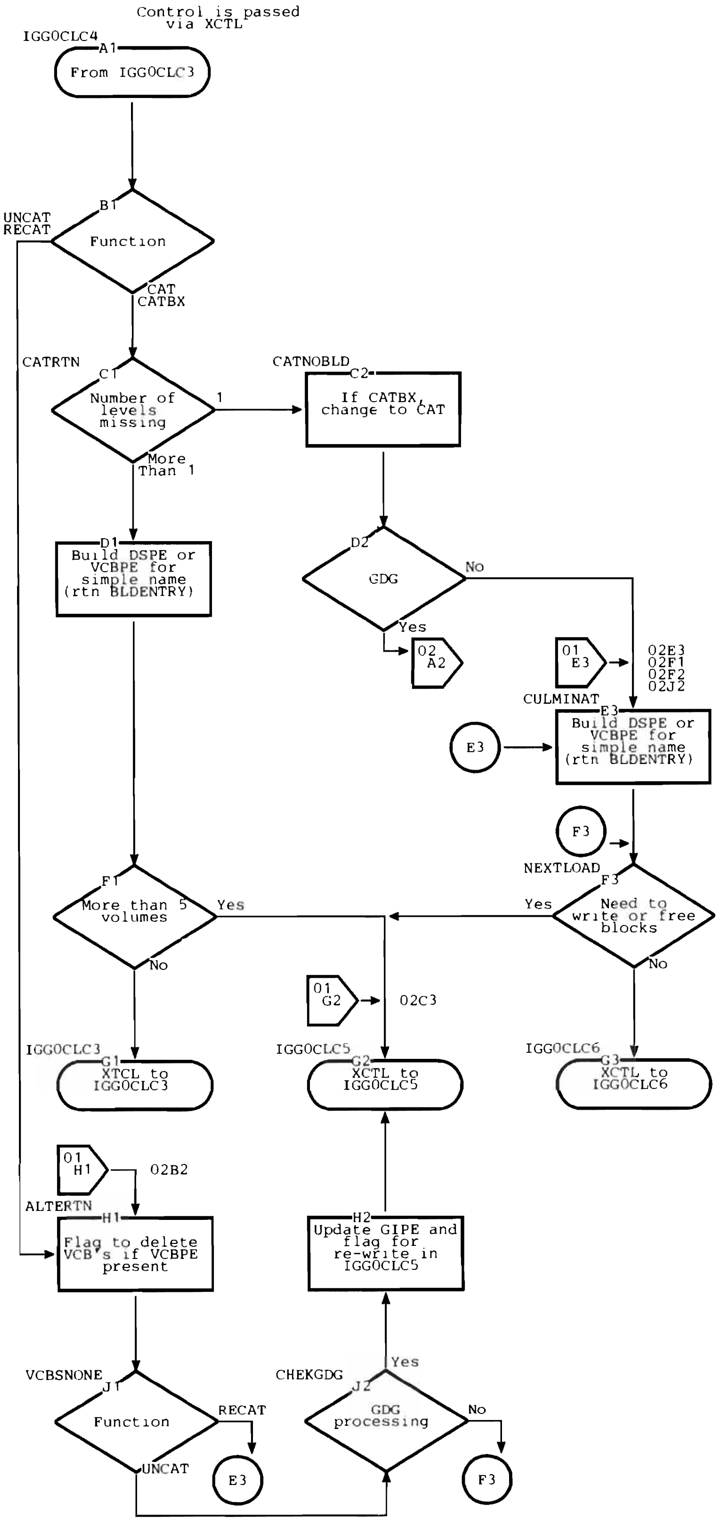
If EMPTY was specified, IGG0CLC5 will remove all entries from the generation index before adding the new entry. Otherwise, IGG0CLC5 will remove only the oldest entry before adding the new entry. This module (IGG0CLC4) flags what is to be done.

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

The RECAT and UNCAT functions are processed by naming the old entry. IGG0CLC6 will delete the old entry when it gets control. For RECAT, a new entry is also constructed. IGG0CLC6 will add this new entry to the catalog.

Internal Subroutines

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



IO performs EXCP input/output operations.

GET reads a block from SYSCTLG 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 an XCTL macro instruction to:

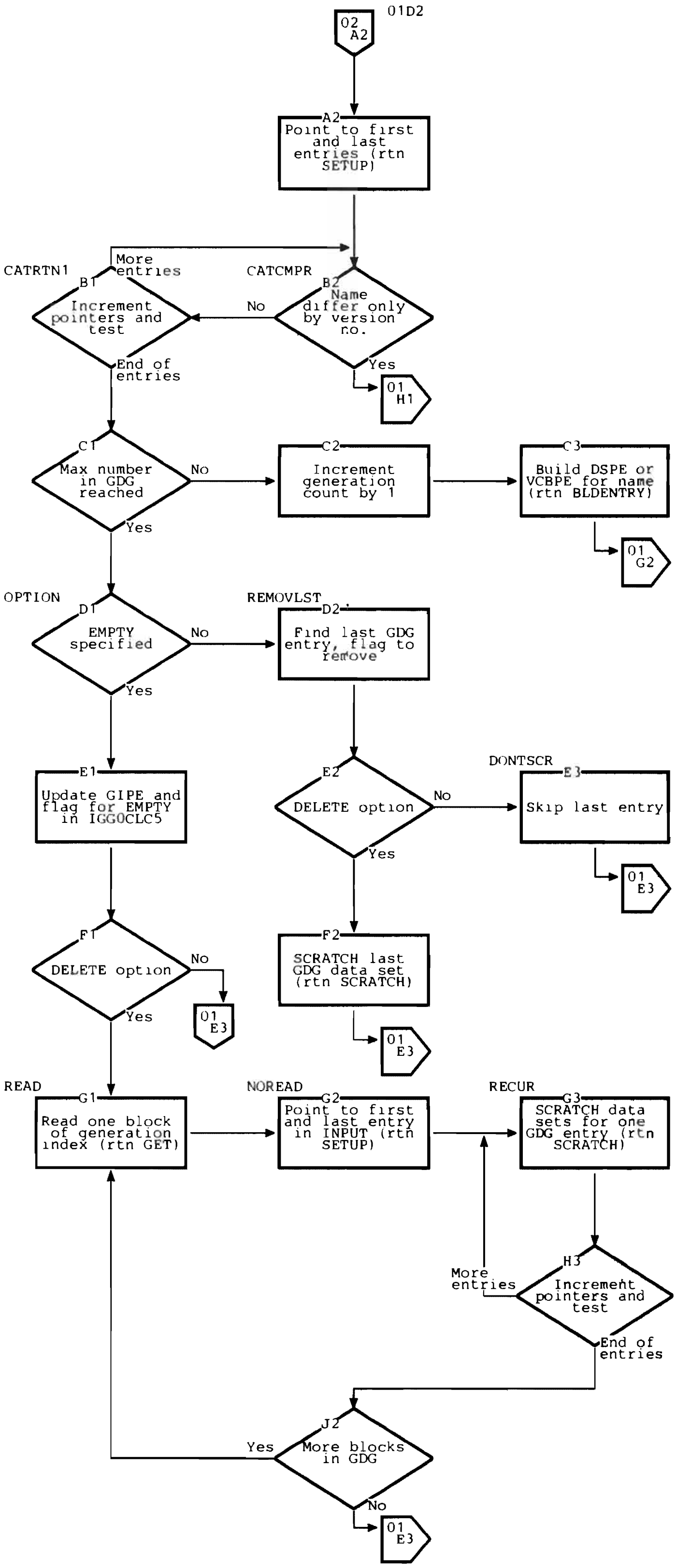
- IGG0CLC3 when CATBX is being performed.
- IGG0CLC5 when auxiliary reading or writing is required:
  - Volume control blocks (VCBs) need to be written.
  - VCBs or index blocks need to be freed.
  - The DELETE option of a GDG needs to be performed.
  - Updated GDG index blocks need to be rewritten.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

Error Conditions

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

References

Item	*Page number*
IGG0CLC3	46
IGG0CLC5	50
IGG0CLC6	52
IGG0CLC7	54
SYSCTLG Entries	61
WORKAREA	68
Method of Operation Figure 8	22


01D2
02 A2
A2 Point to first and last entries (rtn SETUP)
CATRTN1
B1 Increment pointers and test
More entries
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 IGG0CLC5
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)
H3 Increment pointers and test
More entries
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 catalog need to be written or freed.

Registers

4	Base register for this module
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 set this task in the "Must Complete" mode and ensure that any changes to SYSCTLG will be completed.

This module 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 SYSCTLG. They then have a key of zero, indicating that they are available for use.

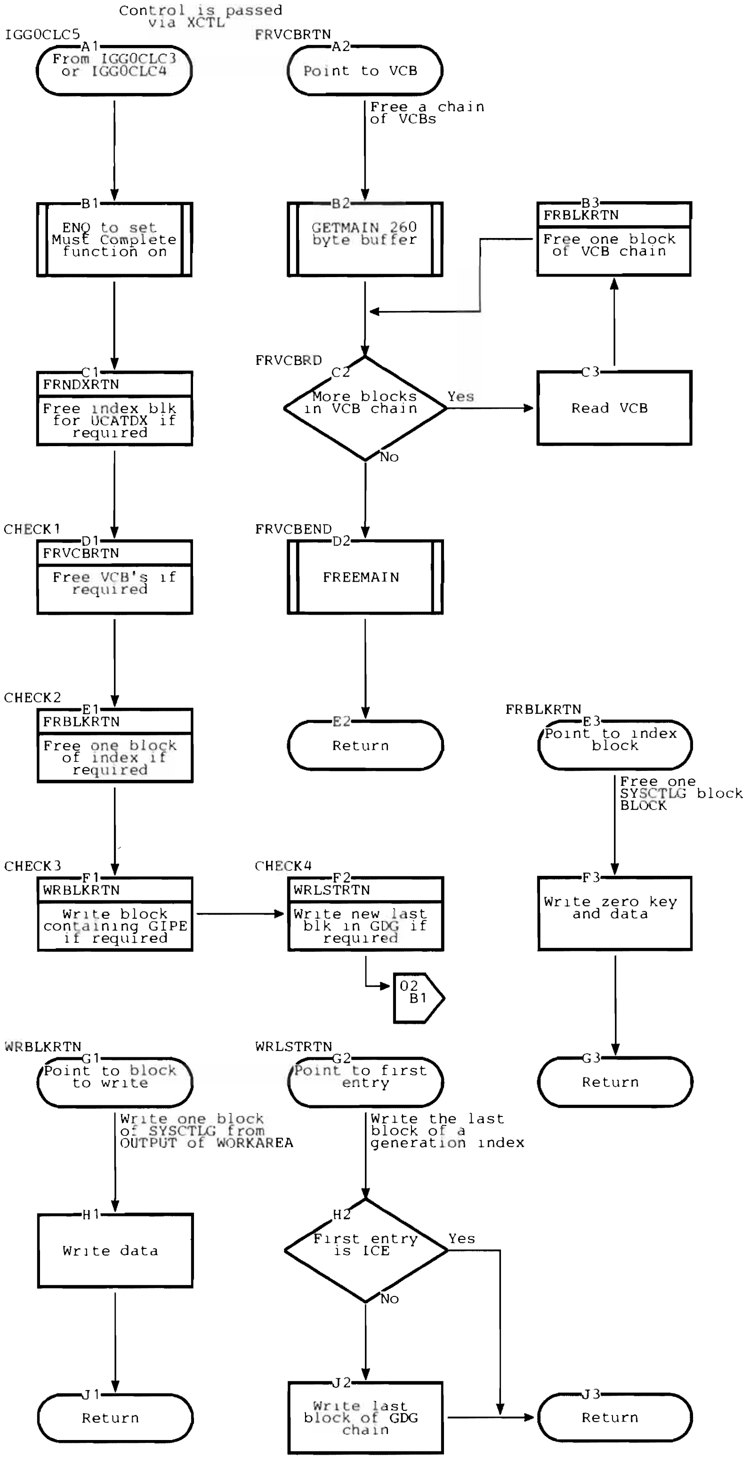
If changes have been made to a generation index, the block containing the generation index pointer entry (GIPE) must be updated. Likewise, the last block of the generation index may need to be rewritten.

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 contins more than five volumes, volume control blocks are constructed from that list and written to SYSCTLG.



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.

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

Exits

Control passes via an XCTL macro instruction to:

- IGG0CLC3 when the requested function is CATBX.
- IGG0CLC7 for error conditions.
- IGG0CLC6 otherwise.

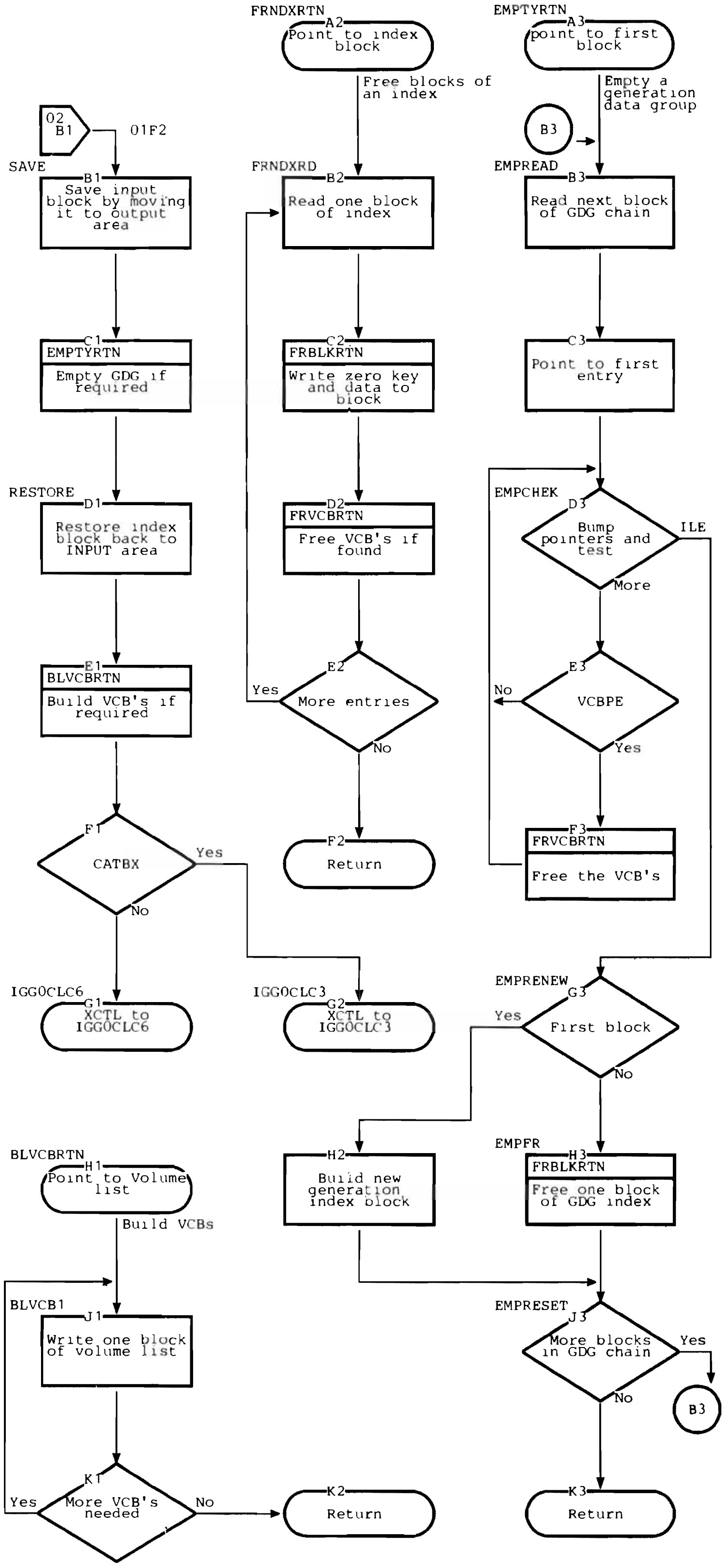
Control passes via an SVC 28 to IGC0002H when a new extent of SYSCTLG is required, and returns to this module.

Error Conditions

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

References

Item	*Page number*
IGG0CLC3	46
IGG0CLC4	48
IGG0CLC6	52
IGG0CLC7	54
IGC0002H	56
SYSCTLG Entries	61
WORKAREA	68
Method of Operation Figure 8	22
Method of Operation Figure 9	23



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.

Registers

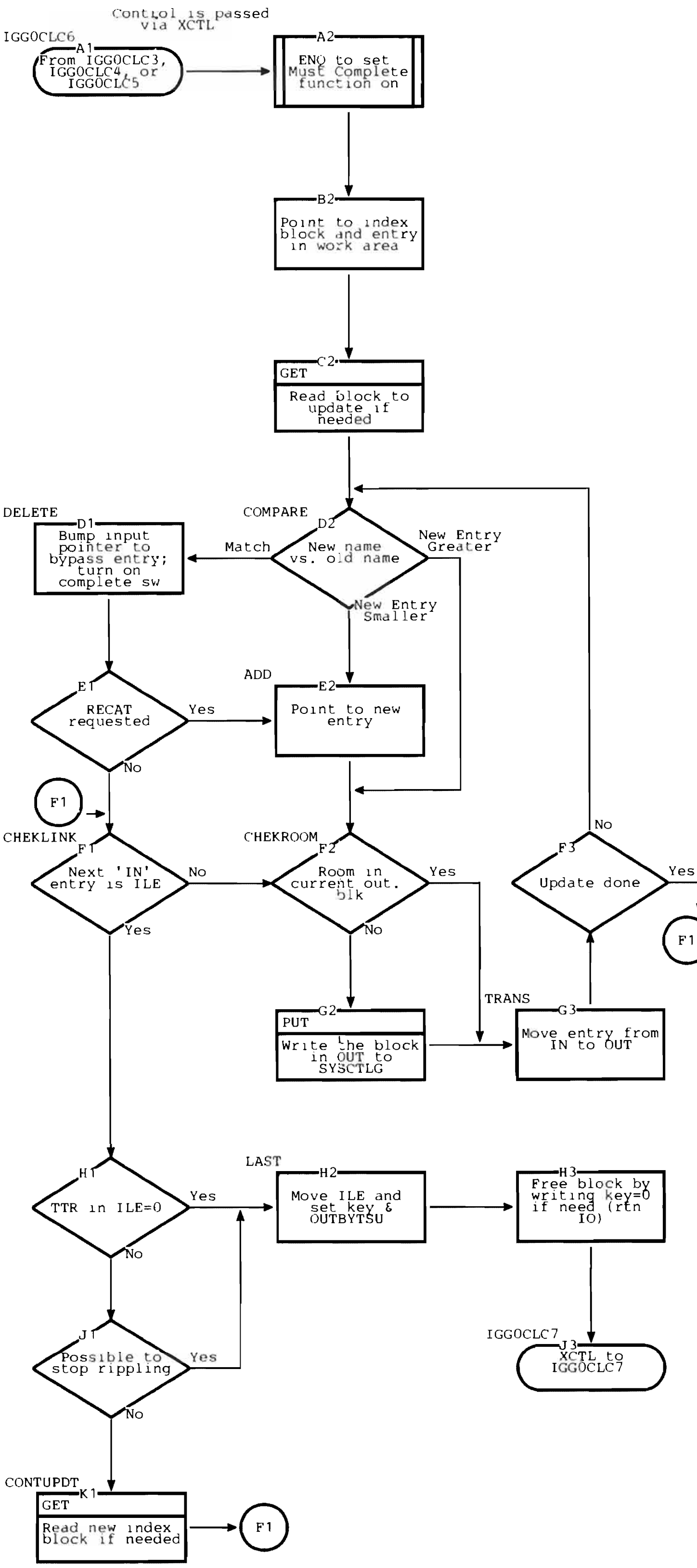
4 Base register for this module
6 Base register for WORKAREA DSECT
8 Base register for CAMLSTD DSECT
12 Linkage register for BAL instructions
14 Linkage register for BAL instructions

Functions

This module adds or deletes an entry to or from a given index block, as set up by earlier phases, and 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 possibilities 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 XCTL
IGG0CLC6
A1
From IGG0CLC3, IGG0CLC4, or IGG0CLC5
A2
END to set Must Complete function on
B2
Point to index block and entry in work area
C2
GET
Read block to update if needed
COMPARE
D2
New name vs. old name
Match
New Entry Greater
New Entry Smaller
DELETE
D1
Bump input pointer to bypass entry; turn on complete sw
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
Update done
No
Yes
F1
G2
PUT
Write the block in OUT to SYSCTLG
TRANS
G3
Move entry from IN to OUT
H1
TTR in ILE=0
Yes
No
LAST
H2
Move ILE and set key &amp; OUTBYTSU
H3
Free block by writing key=0 if need (rtn IO)
J1
Possible to stop rippling
Yes
No
IGG0CLC7
J3
XCTL to IGG0CLC7
CONTUPDT
K1
GET
Read new index block if needed
F1

Internal Subroutines

GET reads one block from an index in SYSCTLG.

PUT prepares and writes one block into an index in SYSCTLG.

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

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

IO performs EXCP I/O operations.

Exits

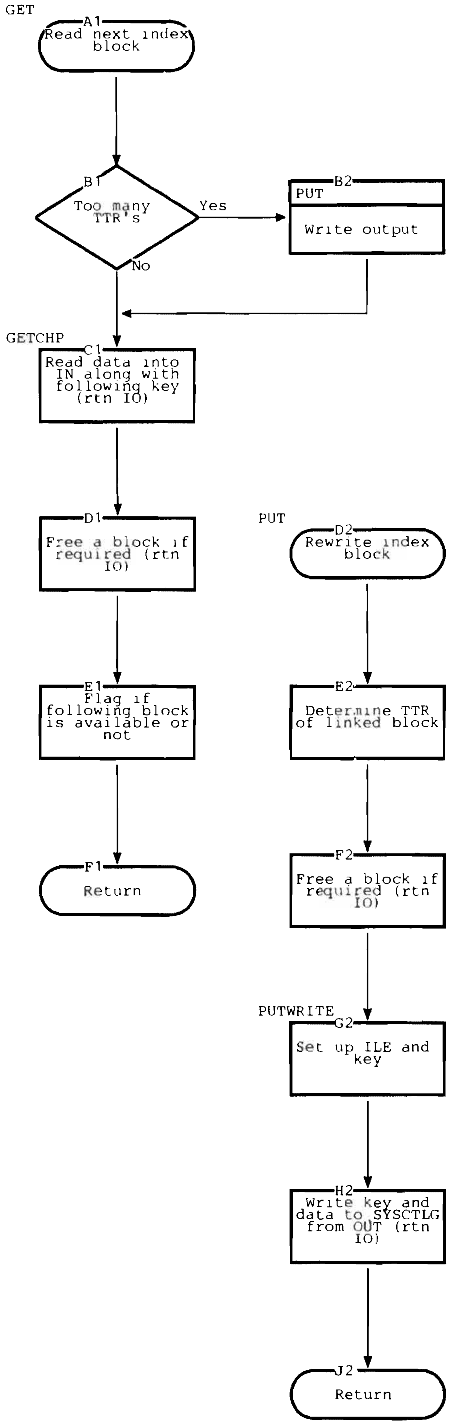
Control is always passed to IGG0CLC7 via an XCTL macro instruction.

Error Conditions

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

References

Item	*Page number*
IGG0CLC3	46
IGG0CLC4	48
IGG0CLC5	50
IGG0CLC7	54
SYSCTLG Entries	61
Method of Operation Figure 9	23


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 IO)
D1
Free a block if required (rtn IO)
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 IO)
PUTWRITE
G2
Set up ILE and key
H2
Write key and data to SYSCTLG from OUT (rtn IO)
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 module of catalog management when an error condition is discovered.

Registers

4 Base register for this module
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 specified the type of error encountered.

Functions

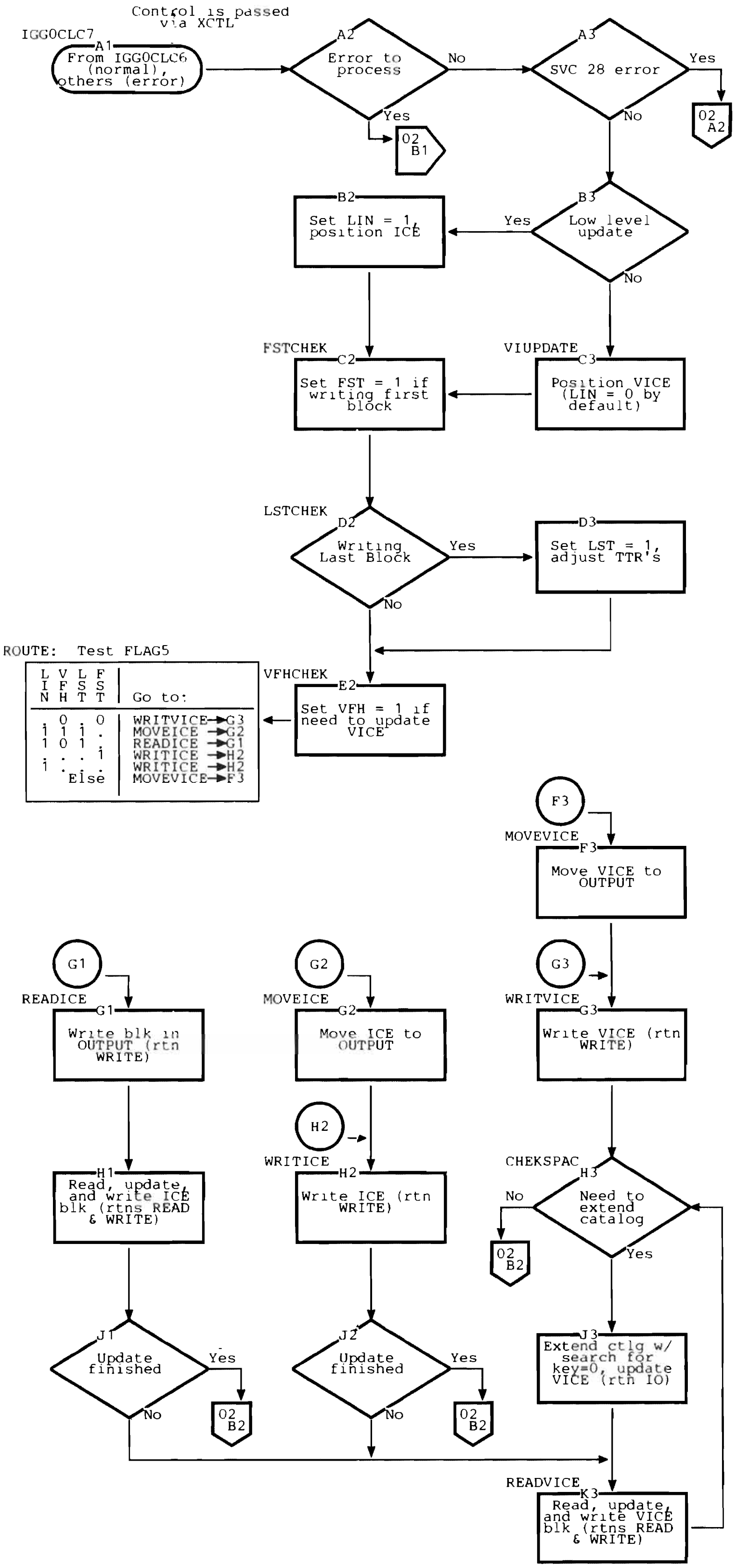
IGG0CLC7 completes the update process. The last block of an updated index is written to SYSCTLG.

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 SYSCTLG.

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

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 into an environment record and written to SYSCTLG. The exceptional return code is set and all resources are freed. Control returns to the caller of catalog management via an EXIT SVC instruction.


Control is passed via XCTL
IGG0CLC7
A1
From IGG0CLC6 (normal), others (error)
A2
Error to process
No
Yes
02 B1
A3
SVC 28 error
Yes
No
02 A2
B2
Set LIN = 1, position ICE
Yes
B3
Low level update
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
Yes
02 B2
J1
Update finished
Yes
No
02 B2
J2
Update finished
Yes
No
02 B2
J3
Extend ctlg w/ search for key=0, update VICE (rtn IO)
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

Internal Subroutines

READ reads one block from SYSCTLG.

WRITE writes one block to SYSCTLG.

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

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

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

Error Conditions

This module returns any exception code from another catalog management module to the caller. This exception code is passed to IGG0CLC7 in WORKAREA.

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

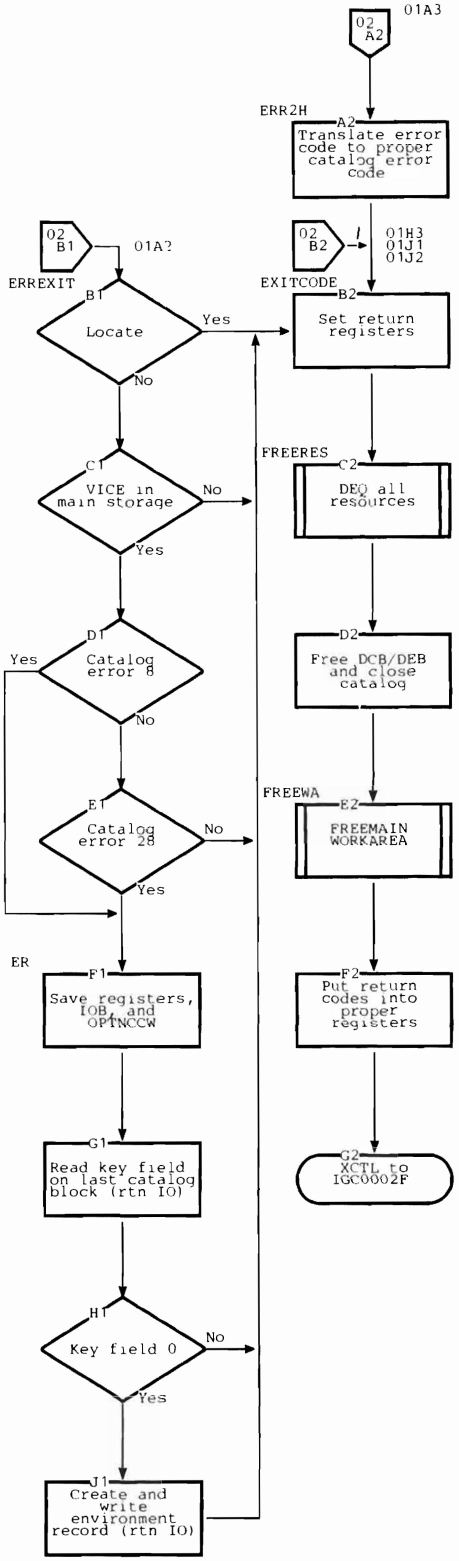
Exits

IGC0002H may be invoked via an SVC 28 when a new extent of SYSCTLG is required. Control returns to this module when one has been located.

Control returns to IGC0002F via an XCTL SVC instruction.

References

Item	*Page number*
IGC0002F	32 and 34
IGG0CLC6	52
IGC0002H	56
VICE	66
ICE	64
Environment Record	66
Method of Operation Figure 9	23



IGC0002H: SYSCTLG Open/Extend

IGC0002H is the entry point. Control comes via an SVC 28 from:

- IGG0CLC0 or IGG0CLC3 to open a CVOL.
- IGG0CLC3, IGG0CLC5, or IGG0CLC7 to extend SYSCTLG.

Control also comes via an XCTL macro instruction from IGG0553E after extending SYSCTLG.

Registers

On Entry for Opening:

0	Zero
1	Address of UCB table for CVOL
15	Address of area in which to build DCB/DEB chain

On Entry for Extending:

0	Address of DCB for SYSCTLG

On Entry after Extending:

0	Negative number
4	Address of EXTEND work area
9	Address of DCB for SYSCTLG
10	TTR for new extent of SYSCTLG
11	Address of UCB

On Exit:

1	Address of DCB/DEB chain

Functions

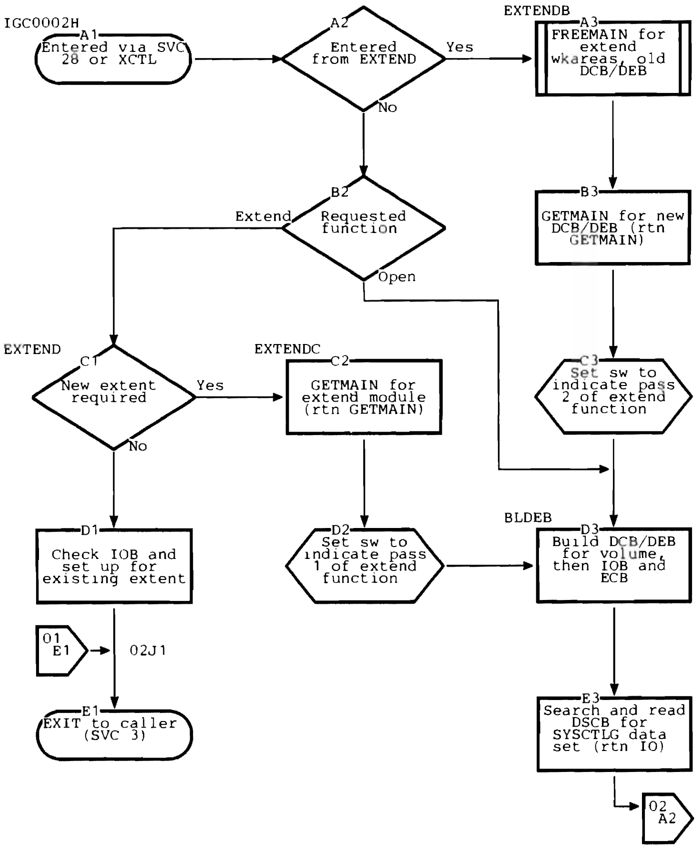
When this module is entered to open a CVOL, a data control block (DCB) and a data extent block (DEB) are built in the work area provided by IGG0CLC0. If the catalog is new, IGG0CLF2 is invoked to format it.

Note: The DCB/DEB constructed by this module is a modification of that described in *OS/VS1 System Data Areas* and in *OS/VS2 System Data Areas*. See Figure 12.

When this module is entered to allocate another extent of SYSCTLG, 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 main 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 intu the new area (the DCB/DEB is reopened).

The DCB/DEB built by this module differs from that described in *OS/VS1 System Data Areas* and d in *OS/VS2 System Data Areas*. These two blocks are merged together; that is, they overlap in the same area of main


IGC0002H
A1
Entered via SVC 28 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
02J1
E1
EXIT to caller (SVC 3)
E3
Search and read DSCB for SYSCTLG data set (rtn IO)
02 A2

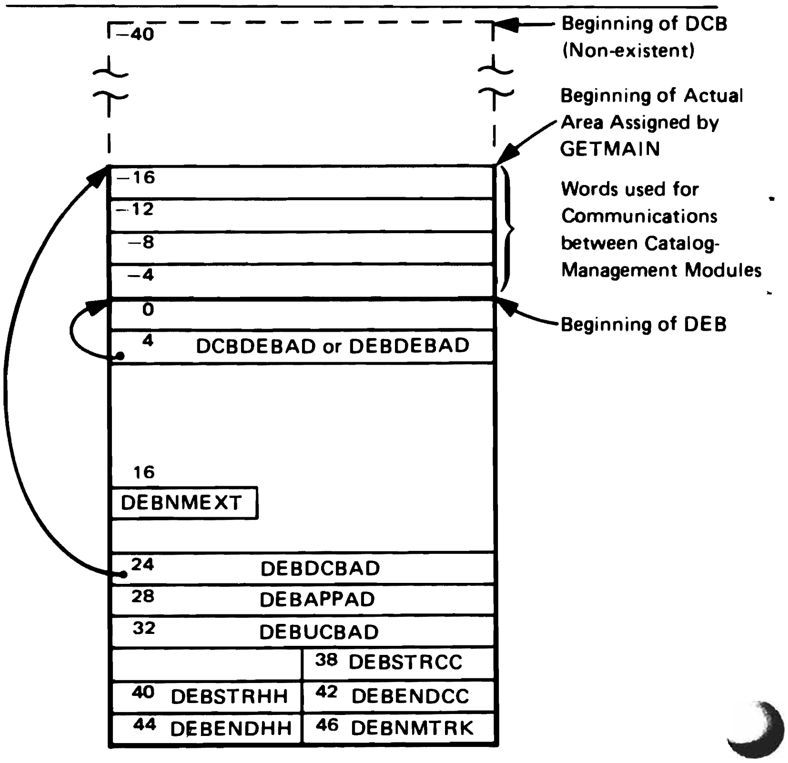


Figure 12. DCB/DEB Built by IGC0002H

storage, as shown in Figure 12. Main storage for the DCB/DEB is set to zeros before building; then only the fields that are shown are filled in. The first 20 bytes of the DCB do not exist (shown by the dotted line), so the area starts with offset 20 of the DCB. The DEB overlays the DCB at offset 40. The fields that are named are described in *OS/VS1 System Data Areas* and in *OS/VS2 System 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 an EXIT SVC instruction when IGC0002H completes its function.

Control returns to the caller via an EXIT SVC instruction for an error condition.

Control passes via an XCTL macro instruction to:

- IGG0553A when another extent is required. Control returns via an XCTL macro instruction to the entry point, IGC0002H (block A1 of part A).
- IGG0CLF2 when the catalog or new extent needs to be formatted. Control returns directly to the caller.

Error Conditions

Code	*Reason*
4	No extents are allocated.
8	No more extents are available.
12	Permanent I/O error.
72	DOS-to-OS conversion was unsuccessful (returned by IGG0533A).

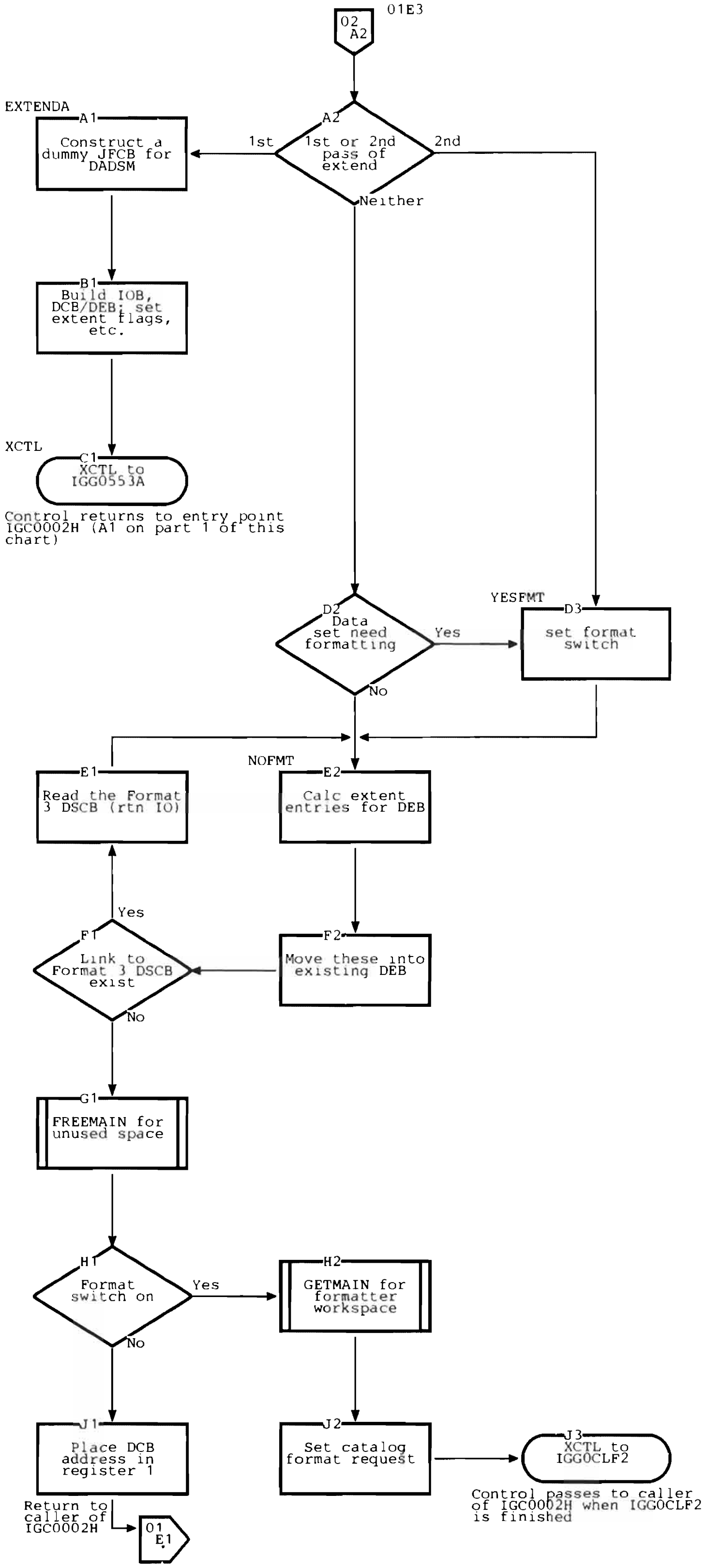
IGG0553A through IGG0553E are invoked by this module and can also return exception codes. For details about these modules, refer to the publication listed below.

References

Item	*Page number*
IGG0CLC0	40
IGG0CLC3	46
IGG0CLC4	48
IGG0CLC5	50
IGG0CLF2	58

IGG0553A through IGG0553E are described in *OS/VS DADSM Logic*, GY28-3787.

DCB and DEB are described in *OS/VS1 System Data Areas*, SY28-0605 and *OS/VS2 System Data Areas*, SY28-0606.



01E3
02 A2
EXTENDA
A1
Construct a dummy JFCB for DADSM
A2
1st or 2nd pass of extend
1st
2nd
Neither
B1
Build IOB, DCB/DEB, set extent flags, etc.
XCTL
C1
XCTL to IGG0553A
Control returns to entry point IGC0002H (A1 on part 1 of this chart)
D2
Data set need formatting
Yes
No
YESFMT
D3
set format switch
E1
Read the Format 3 DSCB (rtn IO)
NOFMT
E2
Calc extent entries for DEB
Yes
F1
Link to Format 3 DSCB exist
No
F2
Move these into existing DEB
G1
FREEMAIN for unused space
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
XCTL to IGG0CLF2
Control passes to caller of IGC0002H when IGG0CLF2 is finished

IGG0CLF2: SYSCTLG and BPAM Directory Formatter

IGG0CLF2 is the entry point. Control comes from IGC0002H.

Registers

0 Number of blocks to be written into a BPAM directory (zero when formatting SYSCTLG)

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

4 Address of work area

5 Data management count decrement value

6 Starting relative track address (TTR) when formatting SYSCTLG

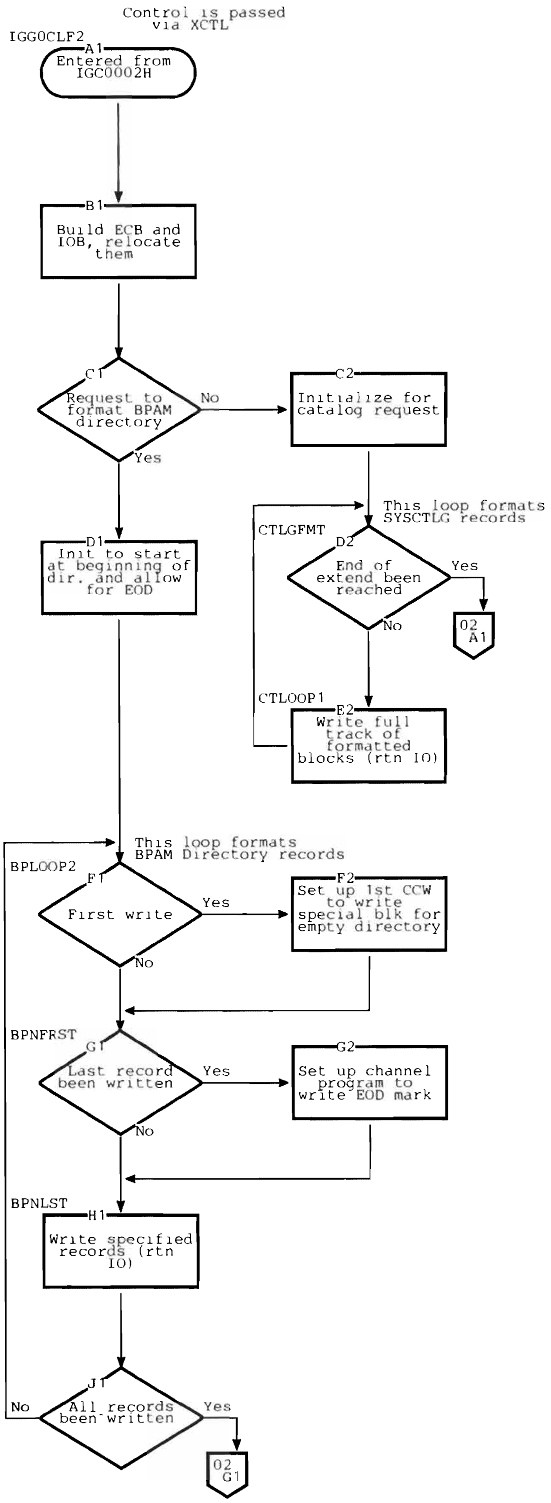
Functions

The data set is formatted into 256-byte blocks with 8-byte keys. When formatting a BPAM directory, the module writes one block at a time. When formatting SYSCTLG, the module writes several blocks each time.

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

If formatting is not being done during an open request, then this is an extent of an already existing SYSCTLG. The VICE is read, updated, and rewritten to reflect the new extent.

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



Control is passed via XCTL
IGG0CLF2
A1
Entered from IGC0002H
B1
Build ECB and IOB, relocate them
C1
Request to format BPAM directory
No
Yes
C2
Initialize for catalog request
This loop formats SYSCTLG records
CTLGFMT
D2
End of extend been reached
Yes
No
02 A1
D1
Init to start at beginning of dir. and allow for EOD
CTLOOP1
E2
Write full track of formatted blocks (rtn IO)
This loop formats BPAM Directory records
BPLOOP2
F1
First write
Yes
No
F2
Set up 1st CCW to write special blk for empty directory
BPNFRST
G1
Last record been written
Yes
No
G2
Set up channel program to write EOD mark
BPNLST
H1
Write specified records (rtn IO)
J1
All records been written
No
Yes
02 G1

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

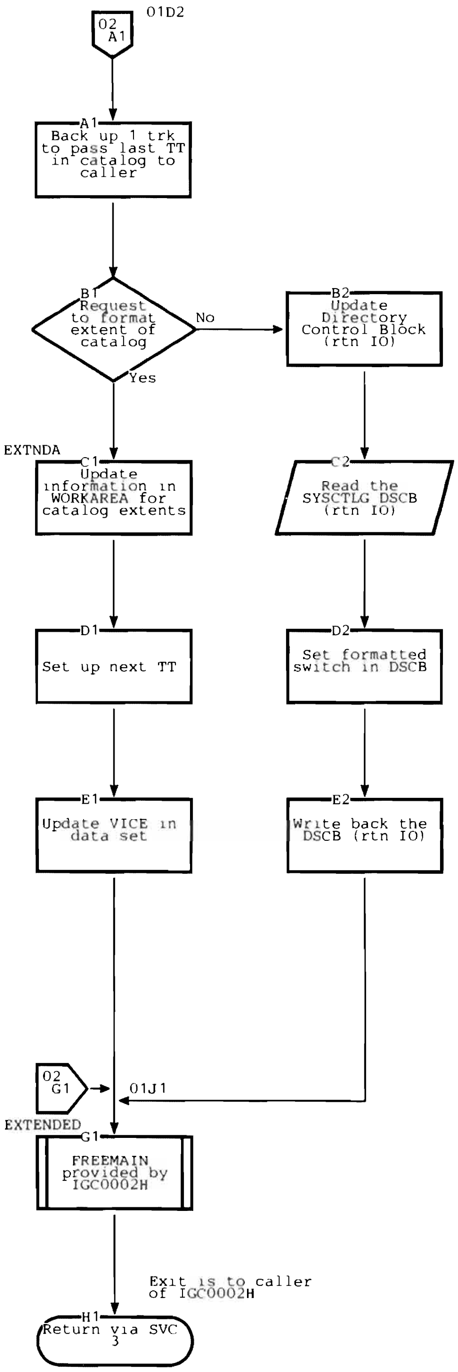
IGG0CLF2 returns one exception code, 12, which indicates that an I/O error has occurred. The caller of catalog management never sees this code.

Exits

Control is passed to the caller of IGC0002H via an EXIT SVC instruction.

References

Item	*Page number*
IGC0002H	56
VICE	66
WORKAREA	68



02 A1
01D2
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)
02 G1
01J1
EXTENDED
G1 FREEMAIN provided by IGC0002H
Exit is to caller of IGC0002H
H1 Return via SVC 3

This chapter helps you relate the discussions in this book to the assembly listings found on microfiche. The rightmost columns of this directory contain page number references to Method of Operation figures and/or flowcharts where more information can be found. The microfiche are found under the component name "Data Management."

Entry Point Name	Module/ Microfiche Name	Description	Method of Operation Page	Flowchart Page
IGG0002F	IGG0002F	Controller I to route catalog requests	20	32
IGG0002F	IGG0002F	Controller II to route catalog requests	20	34
IGG0CLCA	IGG0CLCA	Controller II to resume processing of OS/VS requests		36
IGG0CLCB	IGG0CLCB	Controller II to resume processing of VSAM requests		38
IGG0CLC0	IGG0CLC0	Initialization for catalog management routines	20	40
IGG0CLC1	IGG0CLC1	Relative GDG and alias processing	20	42
IGG0CLC2	IGG0CLC2	Locate processing	20	44
IGG0CLC3	IGG0CLC3	Update initialization and entry building	20, 22, 23	46
IGG0CLC4	IGG0CLC4	Entry building	22	48
IGG0CLC5	IGG0CLC5	First load of update	22, 23	50
IGG0CLC6	IGG0CLC6	Second load of update	22, 23	52
IGG0CLC7	IGG0CLC7	Third load of update, exit processing, and error handling	23	54
IGC0002H	IGC0002H	SYSCTLG open or extend	20	56
IGG0CLF2	IGG0CLF2	SYSCTLG and BPAM directory formatter	20	58

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 for the catalog management routines.

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.
242(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 (see description of CAMLSTD on page 67).
20(14)	1	ERMODMAP	Field MODMAP1 from WORKAREA, described on page 72.
21(15)	.1	ERFLAG1	Field FLAG1 from WORKAREA, described on page 70.
22(16)	..1	ERFLAG2	Field FLAG2 from WORKAREA, described on page 71.
23(17)	...1	ERFLAG3	Field FLAG3 from WORKAREA, described on page 70.
24(18)	2	ERERRCOD	Fields ERRCATSV and ERRLOCSV from WORKAREA, described on page 55.
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, a field described on page 72.
146(92)	..18	EROUTPUT	First entry in OUTPUT, a field described on page 72.
164(A4)	8	EROPTNCC	Field OPTNCCW from WORKAREA, described on page 72.
176(B4)	40	ERIOB	Field IOB from WORKAREA, described on page 72.
212(D4)	44	ERNAME	Fully qualified name provided by the caller.

CAMLSTD DSECT

CAMLSTD describes the parameter list provided by the caller of 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..	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.
		.... .1..	Search VS catalog first.
		.... ..1.	Search only one catalog.
		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 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.

RPSD DSECT

RPSD describes the CCW chain used for rotational positioning 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.

WORKAREA DSECT

WORKAREA serves all catalog modules. WORKAREA serves 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 area overlay other fields, and sections of an area can have more than one label. Figure 13 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.

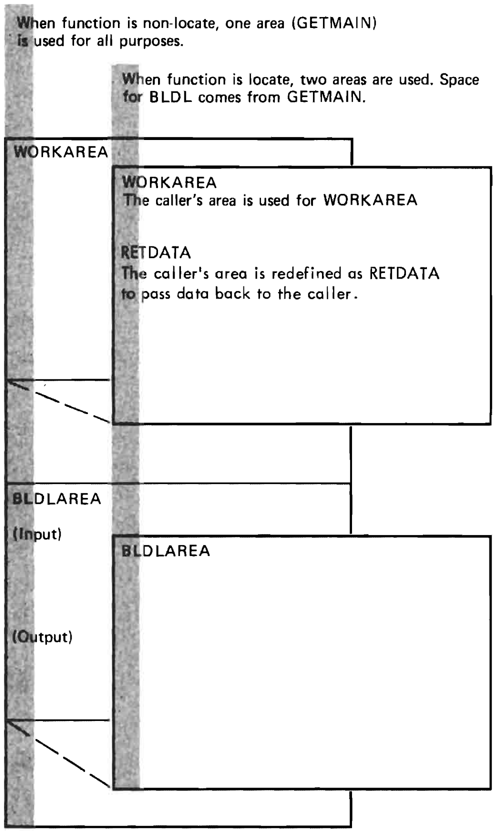

When function is non-locate, one area (GETMAIN) is used for all purposes.
When function is locate, two areas are used. Space for BLDL comes from GETMAIN.
WORKAREA
WORKAREA
The caller's area is used for WORKAREA
RETDATA
The caller's area is redefined as RETDATA to pass data back to the caller.
BLDLAREA
(Input)
BLDLAREA
(Output)

Figure 13. 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 in the entry that is being operated on, or the alias.
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 to which the alias in offset 4 relates.
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.

(cont.)

(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...	'(' has been found in the data set name.
		.... .1..	Last entry found was a CVOL pointer entry (CVPE).
		.... ..1.	Generation index pointer entry (GIPE) has been found.
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; see the description on page 64 for individual fields. Only bytes 8 through 15 are saved here.
136(88)	...9	VICE	Volume index control entry; see the description on page 66 for individual fields. 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	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.

(cont.)

(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 module IGC0002F or IGG0CLC0, because these modules are 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 extend 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 '1'.
200(C8)	16	XCTLPARM	Parameters for XCTL SVC.
216(D8)	12	LNKENTRY	General form in 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.
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.

(cont.)

(cont.)

Offset	Bytes and Alignment	Name and Content	Field Description and Meaning
258(102)	..1	ERRSV2H	Exceptional return code from module 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 'ƀ' or '('.
128(80)	8	RELNUMBR	Work area for Convert-to-Binary 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.
528(210)	120	BLDLCNT	Parameters for BLDL routine.
648(288)	4	BASESAVE	Save area for the register that would otherwise be destroyed by BLDL.
536(218)	44	RESALIAS	Work area used when resolving an alias name.
272(110)	256	INPUT	Input buffer for channel program.
272(110)	256	TRTABLE	Translate table used with TR instruction to analyze the given name.
528(210)	8	SIDE1	Search-ID-Equal CCW.
536(218)	8	TIC1	Transfer-In-Channel CCW.
544(220)	8	OPTNCCW	CCW that is changed to do the required input/output function.
552(228)	8	RC	Read-Count CCW.
560(230)	8	SKE	Search-Key-Equal CCW.
568(238)	8	TIC2	Transfer-In-Channel CCW.
576(240)	8	NOP	NOP CCW.
584(248)	4	ECB	Event control block for channel programs.
588(24C)	40	IOB	Input/output block for channel programs.
630(274)	8	RKD	Read-Key-Data CCW.
638(27C)	8	RD	Read-Data CCW.
644(284)	8	WKD	Write-Key-Data CCW.
652(28C)	264	OUTPUT	Output buffer for channel programs.

Diagnostic Aids

This chapter provides several aids that can be useful when diagnosing difficulties with the catalog management modules. These aids tell you:

- What to look for in a main storage dump.
- How to dump the SYSCTLG data set.
- A description of an *environment record* that is written when an error is detected.
- How to determine what modules are involved in a particular case and what the flow of logic is between them.

Reading Dumps

Two kinds of dumps can be used while diagnosing trouble with catalog management routines: main storage dumps and SYSCTLG 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/VS1 Debugging Guide,* GC24-5093, or *OS/VS2 Debugging Guide,* GC28-0632.

Main Storage Dump

Main storage dumps are rarely helpful. Unless the error that caused the ABEND was actually in one of the catalog management routines, the transient areas (or paging areas) will probably have been used by another module of the operating system.

If, however, an ABEND dump was produced because of an error in one of the catalog modules, 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 catalog management routines. This register points to the module that failed.

Register 6 Pointer to WORKAREA (described on page 68. The field MODMAP1 shows which modules have been entered; compare this to the expected path for the requested function. The section "Module Selection Charts," on page 76, shows the path for each function.

Register 8 Pointer to caller's CAMLST, described on page 67. Check the parameters in CAMLST carefully to be sure that the requests are valid.

Have the following items available before requesting additional assistance:

- Source or input listing for the use of catalog-management routines (be sure MSGLEVEL=(1,1)).
- Main storage dump produced by using a SYSABEND DD statement.
- Dump of SYSCTLG data set.

SYSCTLG Dump

There are several ways to dump a data set; this discussion assumes that HMASPZAP is used. (In VS1, the program is HMASPZAP; in VS2, the program is AMASPZAP.) HMASPZAP is a service-aid program that operates under the operating system. HMASPZAP is described in *OS/VS; Service Aids,* GC28-0633.

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

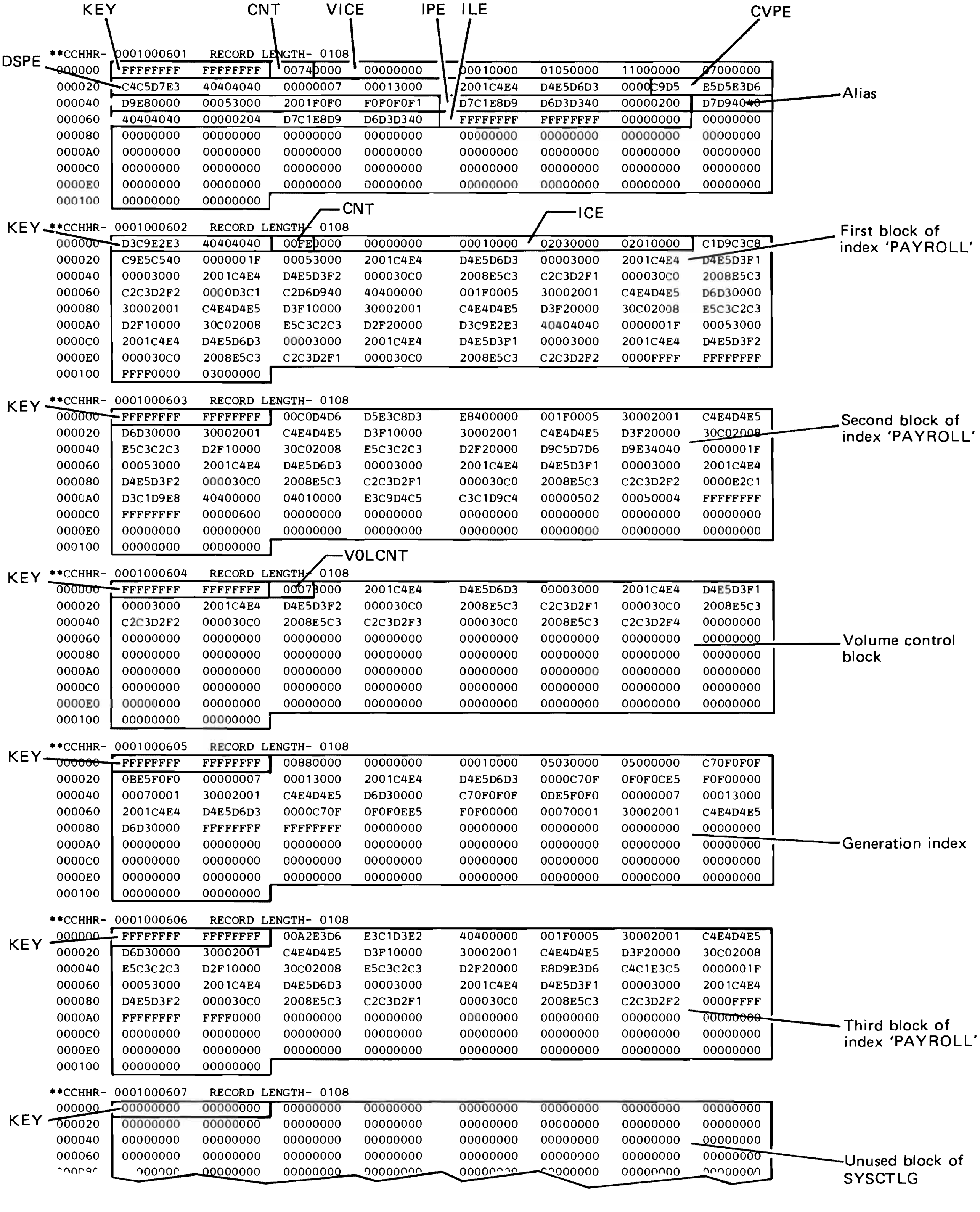
```
//DUMPSTEP  EXEC PGM=HMASPZAP
//SYSPRINT  DD  SYSOUT=A
//SYSLIB   DD  DSNAME=SYSCTLG,UNIT=xxxx,
//           VOL=SER=xxxxxx,DISP=OLD,
//           DCB=(KEYLEN=8)
//SYSIN   DD   *
 ABSDUMP ALL
/*
```

This JCL is used to dump the entire catalog; you can dump a portion 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 SYSCTLG Dump

Figure 14 shows an actual dump of the catalog. Entries in the volume index are outlined, and other blocks of the catalog are identified.



```
**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  C7F0F0F
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 14. Example of a SYSCTLG 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 on page 69.

Environment Record

Some error conditions cause an environment record to be written, whenever possible, to the last block of SYSCTLG. 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 SYSCTLG 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 SYSCTLG with DISP=OLD.
2. Add a step to your job to dump SYSCTLG. Follow the instructions under "SYSCTLG 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 SYSCTLG. 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)," on page 66. The description for each field relates this information to other data areas.

The field ERMODMAP contains seven bits that show which modules have been entered. IGC0002F and IGG0CLC0 are always entered; there are no bit switches for those modules.

As an example, if ERMODMAP equals X'66', then modules IGC0002F, IGG0CLC0, IGG0CLC2, IGG0CLC3, IGG0CLC6, and IGG0CLC7 were entered during the request that caused the environment record to be written. This is the sequence of modules that normally occurs with a request for BLDX.

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

Module Selection Charts

Figure 15 can help you determine which modules of the catalog management routines are involved in any given situation. The figure consists of several charts, which are modified reductions of Figure 10, found on page 26 Each chart shows the path through the catalog management routines for the functions noted on that chart. Figure 10 does *not* show the preprocessing done by the Controller. The blocks that are used are shaded. The specific path is shown by a bold arrow. Always enter module IGG0CLC0, which is the first load of the OS/VS catalog (upper left), then move down and to the right.

Page number references for flowcharts and module discussions are:

Module Name	*Page Number*
IGC0002F, Controller I	32
IGC0002F, Controller II	34
IGG0CLCA	36
IGG0CLCB	38
IGG0CLC0	40
IGG0CLC1	42
IGG0CLC2	44
IGG0CLC3	46
IGG0CLC4	48
IGG0CLC5	50
IGG0CLC6	52
IGG0CLC7	54
IGC0002H	56
IGG0CLF2	58

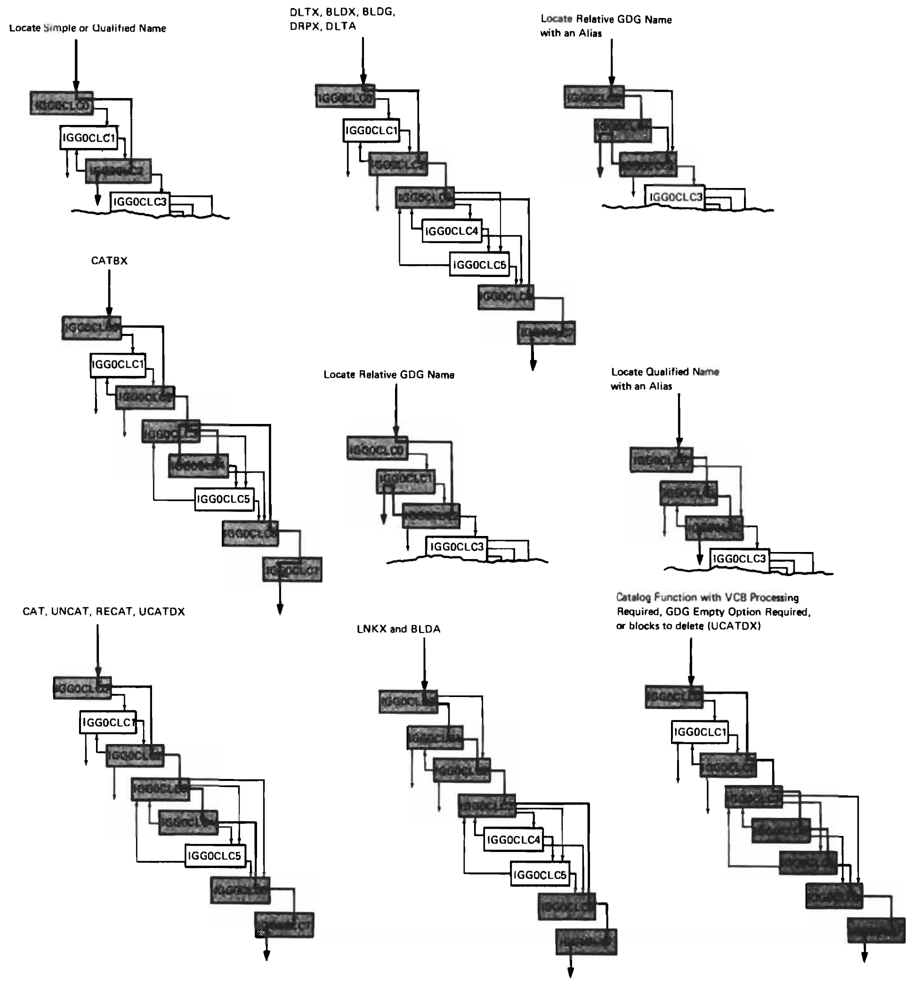

Locate Simple or Qualified Name
IGG0CLC0
IGG0CLC1
IGG0CLC2
IGG0CLC3
DLTX, BLDX, BLDG, DRPX, DLTA
IGG0CLC0
IGG0CLC1
IGG0CLC4
IGG0CLC5
Locate Relative GDG Name with an Alias
IGG0CLC3
CATBX
IGG0CLC1
IGG0CLC5
Locate Relative GDG Name
IGG0CLC0
IGG0CLC1
IGG0CLC3
Locate Qualified Name with an Alias
IGG0CLC3
CAT, UNCAT, RECAT, UCATDX
IGG0CLC1
IGG0CLC5
LNKX and BLDA
IGG0CLC4
IGG0CLC5
Catalog Function with VCB Processing Required, GDG Empty Option Required, or blocks to delete (UCATDX)
IGG0CLC1

Figure 15. Correlating Functions to Modules of Catalog Management

Glossary

This glossary contains definitions of words and acronyms that are used in this publication. Other data processing definitions can be found in *Data Processing Glossary,* GC20-1699.

alias: An alternative name for a data set. Only the high-level name of a qualified data set name may have an alias.

CAMLST: An Assembler language macro instruction that produces the parameter list for catalog management.

catalog: (noun) The collection of all data set indexes maintained by data management. (verb) To include the volume identification of a data set in the catalog.

CATALOG: An Assembler language macro instruction used to invoke the catalog management routines.

cataloged data set: 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.

connected: The property of a control volume that allows it to be found by searching another control volume.

control volume (CVOL): A volume that contains one or more of the indexes of the catalog.

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 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.

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

entry: A logical record of the catalog, data set SYSCTLG.

environment record: A 256-byte record that is written when the catalog management routines discover 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.

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 catalog that identifies the generations of a generation data group.

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

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

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

INDEX: An Assembler language macro instruction used to invoke the catalog management routines.

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

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

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

level: A conceptual relationship between indexes of the 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.

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

LOCATE: An Assembler language macro instruction used to invoke the catalog management routines.

must-complete: A facility of the operating system that ensures integrity whenever system resources are manipulated to modify information that is critical to the continued operation of the system.

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

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. Three unique resources are used by catalog management routines: the volume index, high-level name, and the volume index control entry (VICE).

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

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 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 catalog.

system residence volume: The volume on which the nucleus of the operating system and the highest level of the catalog are located.

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

transient area: An area of main storage contained within the operating system nucleus where supervisor routines may be temporarily loaded for execution.

true name: The high-level qualifier to which an alias is related.

uncatalog: To remove the volume identification of a data set from the 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 catalog entry that identifies a VCB for a named data set.

volume index: The highest level of index in the 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.

Index

The entries in this index are shown as they appear in the text of the book, that is, capitalized words are capitalized in text, lowercase words are lowercase in text, etc. When more than one reference is given for an entry, the primary reference is listed first.

Your Comments, Please . . .

OS/VS
Catalog Management Logic

SY35-0003-2

Your comments and constructive criticism regarding this publication will help us to improve it so that you may be better served. Each comment will be reviewed carefully by those responsible for the publication. Comments and suggestions become the property of IBM.

Thank you for your cooperation.

Comments:

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

SY35-0003-2

First Class
Permit No. 568
Boulder, Colorado

Business Reply Mail
No Postage Necessary If Mailed In The U. S. A.

Postage will be paid by:

IBM Corporation
Post Office Box 1900
Boulder, Colorado 80302

Attention:
Programming Publications

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)

SY35-0003-2

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)

Printed in U. S. A. SY35-0003-2

IBM Technical Newsletter

This Newsletter No. SN26-0912
Date July 28, 1978

Base Publication No. SY26-3826-3
File No. S370-30

Prerequisite Newsletters None

OS/VS2 Catalog Management Logic

This technical newsletter, a part of Release 3.7 of OS/VS2, incorporates and replaces all previous SU information in this publication. Please replace all identically numbered pages. These replacement pages remain in effect for any subsequent releases unless specifically altered. Pages to be inserted and removed are:

Cover - 12.1 (10.1, 12.1 added)
23 - 56
61 - 64
79, 80
85 - 90
95 - 102
107 - 116
127 - 132
139 - 142
147 - 154
169 - 176
197 - 206 (203.1 added)
225 - 276 (226.1 - 226.9 and 262.1 added)
283 - 286
291 - 300.2 (300.1, 300.2 added)
315, 316
335, 336
343 - 348.1 (348.1 added)
357 - 364
381, 382
387, 388
393 - 402

Each technical change is marked by a vertical bar to the left of the change.

Summary of Amendments

Changes included in this newsletter are summarized under "Summary of Amendments" following the list of diagrams.

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

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

Printed in U.S.A.