Licensed Material — Property of IBM
LY24-5191-1
File No. S370-30

Program Product

VSE/VSAM VSAM Logic, Volume 1: Catalog Management, Open/Close, DADSM, IIP, Control Block Manipulation

Program Number 5746-AM2

Component 5745-SC-VSM

Release 2

IBM

Second Edition (December 1979)

This edition, LY24-5191-1, applies to Release 2, of Virtual Storage Extended/Virtual Storage Access Method (VSE/VSAM), Program Product 5746-AM2, and to subsequent releases and modifications until otherwise indicated in new editions or Technical Newsletters. Changes are periodically made to the information contained herein; before using this publication in connection with the operation of IBM systems, consult the *IBM System/370 and 4300 Processors Bibliography,* GC20-0001, for the editions that are applicable and current.

Summary of Amendments

For a list of changes, see page iii.

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

Summary of Amendments for VSE/VSAM VSAM Logic, Volume 1

Summary of Amendments for LY24-5191-1 Release 2

This major revision contains changes for the following items:

- DASD Sharing: Users can share files and catalogs across VSE systems as well as across partitions through the use of the Access Method Services SHAREOPTIONS parameter. VSAM now uses the supervisor lock table/lock file and the LOCK and UNLOCK macros to determine whether a file can be opened by multiple users. (This function was formerly performed by the Look-Aside Table and by issuing USE and RELEASE macros.)
- Catalog Management Improvement: The scanning method used by Catalog Management in retrieving catalog record fields (from the catalog field name dictionary) has been changed to reduce the path length for each scan and to reduce the number of scans required for a combination name.
- Dynamic Files: If a file is defined with the NOALLOCATION parameter, space is not allocated to it until it is opened. VSAM Open invokes Catalog Management to allocate space of the requested class.
- Default Volumes: Users can omit explicit volume lists in the DEFINE CLUSTER and DEFINE ALTERNATEINDEX commands if a default model exists for the component. Catalog Management generates a volume list from the default model.
- Partition/Processor Independence: Catalog Management generates a filename by removing the % and appending the partition-ID (partition independent file) or removing the %% and appending the processor-ID and model number (processor independent file).
- DEDICATE/ORIGIN: For DEDICATE, Catalog Management searches the VTOC and assigns all available tracks or blocks to VSAM for suballocation. For ORIGIN, Catalog Management assigns space in the VTOC based on the value specified in the ORIGIN parameter, rather than requiring an explicit specification in the EXTENT statements.
- Dynamic Assignment: Catalog Management does not require the user to specify device assignments (symbolic unit parameter in EXTENT statement). It requests the Supervisor to dynamically assign devices.
- Space Management for SAM Feature: This manual documents hooks in VSAM Open/Close for the Space Management for SAM Feature. For information about the internal logic of that feature, refer to *VSE/VSAM Space Management for SAM Feature Logic,* LY24-5204.

Various editorial changes are also included to improve the usefulness of this manual.

Preface

This logic manual is one of three volumes providing detailed information about VSE/VSAM. The three volumes are:

VSE/VSAM VSAM Logic, Volume 1: Catalog Management, Open/Close, DADSM, IIP, Control Block Manipulation, LY24-5191

VSE/VSAM VSAM Logic, Volume 2: Record Management, LY24-5192

VSE/VSAM Access Method Services Logic, LY24-5195

Except for record management, this volume contains all VSAM logic documentation. Specifically, the following topics are included:

- Control block manipulation
- OPEN/CLOSE
- ISAM Interface
- Catalog management
- Catalog management services
- DADSM

VSE/VSAM VSAM Logic, Volume 2 documents record management, I/O management, buffer management and EOV logic.

This manual is mainly intended for persons involved in program maintenance and for system programmers who are altering the program design. Logic information is not necessary for the operation of the programs described.

This manual and the code it supports should be viewed as a maintenance set. This means that the module prologues and comments contain certain types of information and that this manual contains other kinds of information. Thus, the listings provide the description of the internal logic of modules, and the manual uses Method of Operations diagrams to show what the functions of VSAM are and how the modules work together to carry out those functions. The term *data set* is used in this manual instead of *file* to conform to the program listings.

Effective use of this publication requires an understanding of system operation, PL/S language, assembler language, and its associated macros.

Organization of This Publication

This publication is organized in the following manner:

- *Section 1. Introduction* describes the major components of VSAM.
- *Section 2. Method of Operation* contains HIPO diagrams describing the VSAM components documented in this volume.
- *Section 3. Program Organization* describes the information contained in VSAM program listings and the relationship of the program structures to the issued macro. Flowcharts of catalog management modules are included.
- *Section 4. Directory* contains lists of phases, components, modules, routines, catalog external entry points, and data areas.
- *Section 5. Data Areas* describes control blocks used by the VSAM components documented in this volume, and describes VSAM data, index, and catalog records.
- *Section 6. Diagnostic Aids* contains diagnostic aids, such as error codes.

- *Glossary* defines terms relevant to VSAM.
- *Index* is a subject index to the publication.

Required Publications

The following publications should be read and understood before using this publication:

VSE/VSAM General Information, GC24-5143, explains basic VSAM-Extended concepts and facilities and how to use them.

Using VSE/VSAM Commands and Macros, SC24-5144, tells how to code VSAM macros in application programs and describes VSAM data management. Access Method Services commands and their use are also described.

Related Publications

Other publications that may be of interest in conjunction with this manual are:

VSE/VSAM Programmer's Reference, SC24-5145, describes installation and operating procedures, sysgen information, storage estimates, debugging techniques, performance tips, and recovery procedures.

VSE/VSAM Messages and Codes, SC24-5146, includes all messages and codes originated by VSAM and Access Method Services.

VSE/VSAM VSAM Logic, Volume 2, LY24-5192, describes record management, I/O management, buffer management, and EOV logic.

VSE/VSAM Access Method Services Logic, LY24-5195, documents the logic of Access Method Services.

VSE/VSAM Space Management for SAM Feature Logic, LY24-5204 documents the interface between that feature and VSAM open/close.

Using This Publication

This publication is designed to be used with the VSAM program listings. The diagrams in *Method of Operation* describe the major functions performed by VSAM; these diagrams are intended to be your key to a module name (and routine name, as appropriate) in the listing. See the *Method of Operation* chapter for a description of how to read these diagrams. For information on what is available in the program listings, see the chapter *Program Organization.*

The module directory in the *Directory* chapter lists the modules by symbolic name (all of which start with IKQ, IIP, IGG0, $S or $$B) and contains page references to the appropriate method of operation diagram or program structure that applies to each module. If you wish to see how modules are grouped according to component (such as open, catalog, etc.) see the component directory. The routine directory, where relevant, further shows how the modules are subdivided into routines.

The *Directory* chapter also contains the names of the catalog external entry points (which start with IGGP). These external entry points are cross-referenced in the module directory by module name. As a further aid, charts showing program flow for each catalog module are contained in the *Program Organization* chapter. The charts are numbered sequentially by an alphameric code that corresponds to the last two characters of the symbolic module name, for example, module IGG0CLAG is flowcharted in Chart AG.

The *Index* to this volume contains a list of all VSAM modules and indicates whether each is documented in Volume 1, Volume 2, or both.

Contents

Illustrations

Figures

Method of Operation Diagrams

Catalog Program Flowcharts

Acronyms and Abbreviations

AC	allocation chain
ACB	access method control block
ACC	AMS Catalog communication area
ACE	argument control entry
ADDR	address
ADR	addressed accessing
AIX	alternate index
AMBL	access method block list
AMCBS	access method control block structure
AMDSB	access method data statistics block
AMDTF	access method define the file (ISAM only)
ANCHT	anchor table
ARDB	address range definition block
AU	allocation unit
BCB	buffer control block
BCR	base cluster record
BHD	buffer header
BKPHD	block pool header
BKWD (also BWD)	backward
BLK	block
BSPH	buffer subpool header
BUFF	buffer
BUFH	buffer header
CA	control area
CAT	catalog
CAXWA	catalog auxiliary work area
CB	control block
CCA	catalog communications area
CCB	command control block
CCR	catalog control record
CI or CNV	control interval
CIDF	control interval definition field
CINV	control interval
CIWA (also CIW)	control interval work area
CKD	count-key-data
CM	catalog management
CMS	catalog management services
CNV or CI	control interval
COMREG	communications region
CP	channel program
CPA	channel program area
CPAH	channel program area header
CPL	catalog parameter list
CRA	catalog recovery area
CTGFL	catalog field parameter list
CTGFV	catalog field vector table
CTGPL	catalog parameter list
CVH	common VTOC handler
DADSM	direct-access device space management
DDname	data definition name
DIR	direct processing
DLBL	DASD label
DS	data set
DSA	dynamic storage area
DSCB	data set control block (in VSE, VTOC label)
DSN (also DSNAME)	data set name
DSORG	data set organization
DTF	define the file
ECB	event control block
EDB	extent definition block
EOD	end of data
EOF	end of file
EOV	end of volume
ESDS	entry-sequenced data set
EXCP	execute channel program
EXLST	exit list
FBA	fixed block architecture
FCDB	field control and data block
FKS	full key search
Fn	format n
FPL (also FL)	field parameter list
FS	free space
FVT	field vector table
FWD	forward
FXL	fix list
GEN	generic key search
GO	group occurrence
GOP	group occurrence pointer
ID	identifier
IIP	ISAM interface program
I/O	input/output
ISAM	indexed sequential access method
JIB	job information block
KEQ	search on key equal
KEY	keyed accessing
KGE	search on key greater or equal
KRDR	key range determination routine
KSDS	key-sequenced data set
KWTC	keyword type code
LOC	locate
LPMB	logical-to-physical mapping block
LRD	last record
LUB	logical unit block
MVE	move
n	number
NSP	note string position
NUB	no user buffer
NUP	no update
OAL	open ACB list
O/C/EOV	open/close/end of volume
OPNWA	open work area
PIB	program information block
PL/S	programming language/system
PLH	placeholder
PSW	program status word
PT or PTR	pointer
PUB	physical unit block
RAB	record area block
RBA	relative byte address
RDF	record definition field
REP	replication
Rn	register n
RPHD	resource pool header
RPL	request parameter list
RRDS	relative-record data set

RSCB	resource sharing control block
SCIB	search compressed index block
SEOF	software end of file
SEQ	sequential
SHRW	file sharing work area
SKP	skip sequential
SS	sequence set
SVC	supervisor call
THB	the hold block
TIC	transfer in channel
UBF	user buffer
UCAT	user catalog
UPD	update mode (or data modify)
USB	upgrade set block
USVR	user security verification routine
VOLID	volume identification
VRPPL	BLDVRP parameter list
VSAM	virtual storage access method
VSRT	VSAM shared resource table
VTOC	volume table of contents
WA	work area

Section 1. Introduction

Virtual Storage Access Method (VSAM) is an access method used with direct-access storage to provide fast storage and retrieval of data.

VSAM is divided into modules, which are logically grouped into the following components:

- Control block manipulation, which allows the user program to create, modify, display, and test the contents of some VSAM control blocks (the ACB, EXLST, and RPL, which are described under *Data Areas* in this publication), and to build or delete a VSAM resource pool.
- Open, which connects a user's program to a VSAM data set and builds the control blocks required to permit the user to read from and write to the data set.
- ISAM interface, which allows the user program to issue ISAM macro instructions to process records in a VSAM data set.
- Catalog management, which writes and updates catalog records. Catalog management processes the catalog to obtain information for Open, Close, end-of-volume, and Access Method Services.
- DADSM, which allows the system to maintain VTOC labels for data spaces. In VSAM, DADSM is used by the catalog to create and delete data spaces, both unique and nonunique.
- Record management, which reads and writes records in response to user-issued VSAM and ISAM macro instructions. This component also reads and writes records for the catalog management component. This component is documented in *VSE/VSAM VSAM Logic, Volume 2.*
- End of Volume, which mounts volumes and allocates space. End of Volume modifies the existing control blocks to reflect the newly mounted volumes and newly allocated space. This component is documented in *VSE/VSAM VSAM Logic, Volume 2.*
- Close, which disconnects a user's program from a data set and releases the data set's control blocks built by Open. Close also updates statistics in the VSAM catalog.
- Service aids, which enable program maintenance and Field Engineering personnel to obtain dumps, maintain VTOC labels, and load phases.

For a list of the modules in these components, see the *Directory* in this publication.

Section 2. Method of Operation

Reading Method of Operation Diagrams

Method of operation diagrams depict the internal functions of a programming system, in this case, an access method. The internal functions are categorized by the macro instructions issued by the user, such as the GENCB, MODCB, OPEN, GET, PUT, CLOSE and ENDREQ macro instructions.

Diagram AB shows the basic organization of the method of operation diagrams according to the macro instructions mentioned above. References lead from the high-level charts showing subfunctions required to carry a request to its completion.

Note the relationship of function (exemplified by the macro instructions) to component. Starting with an OPEN issued by the user, a logical progression is made from Open modules to supporting Catalog modules. When a record management macro instruction such as PUT is issued by the user, not only the Record Management modules are involved (which include modules that perform buffer and I/O management and end-of-volume processing) but the Catalog modules which, in turn, call upon the DADSM modules for space management.

The diagrams contain three blocks of information: input, processing, and output. The left-hand side of the diagram shows the data that serves as input to the processing steps on the center of the diagram, and the right-hand side shows the data that is output from the processing steps. Input is anything significant that program processing steps refer to or get. Processing is the steps that support the function or subfunction represented by the diagram. Output is any significant change effected by a processing step, for example, register contents, or control blocks created or modified. The processing steps are numbered and the numbers correspond to notes, if any, on the pages following the appropriate diagram(s). If notes are given, they include references to modules, routines, and/or labels shown on the extreme right-hand side of the diagram. These references are your link to the program listings. Figure 2.1 shows the symbols used in these diagrams and describes their meaning.

As an example of how to interpret a typical method of operation diagram, see page 2 of Diagram GA, which graphically depicts the CLOSE/TCLOSE functions. The left-hand side of the diagram shows the significant input required by the processing steps shown in the diagram. For example, register 0 points to a list of DTF pointers for an ISAM user and ACB pointers for a VSAM user. The data-set information in the ACB is input to steps 5 and 7 in the processing portion of the diagram. The processing portion of the diagram shows the processing steps required to fulfill the function described by the diagram. Note that the function described by one diagram may be performed by one or more VSAM modules; that is, the diagrams not only show program flow, but show the subfunctions that are required to carry out the function and that are subsequently shown in separate diagrams.

Note that some diagrams have more than one entry point. In Diagram GA, for example, there are three entry points:

(1) at step 1 for an automatic Close,
(2) at step 5 after a user-issued CLOSE macro
(3) at step 7 after a user-issued TCLOSE macro

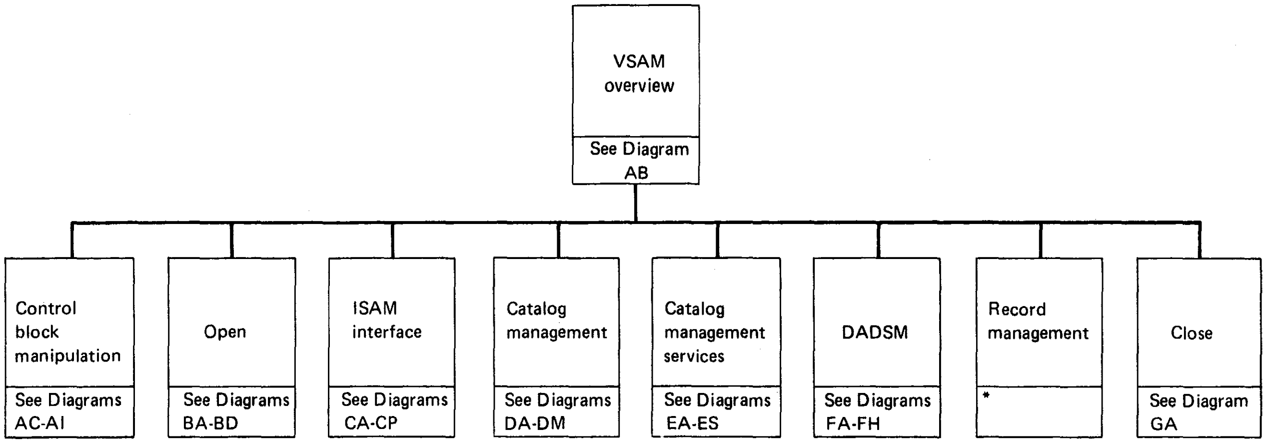
The notes provide details about the processing shown in the diagram. For example, note 5 tells what action the VSE Close Monitor takes (in step 5) when it examines the DTF-type field in the list of ACBs and DTFs passed by the user. The notes also name the modules and routines that perform the functions represented.

The diagrams are numbered in a sequence that follows the pattern *ccn,* where the first character, in general, represents a part of VSAM such as catalog management, the second character represents a category within catalog management, and the number represents the first, second, third, etc., page of that particular diagram. Thus, DE1 would be the first page (1) of the Locate function (E) for catalog management (D). See the list of diagrams for details.

	DATA MOVE -- shows input and output of data to and from the processing steps
	FLOW OF ACTION -- shows the flow of action through the processing steps
	ERROR FLOW -- shows the flow of action in the case of an error or for unusual conditions. This symbol is normally used only in the steps where an error causes a change of flow. Further error processing uses the normal symbols for flow of action.
	POINTERS -- these are the interconnections between the control blocks and data areas
	REFERENCE INDICATOR -- means *refer to this item*
A	I/O INDICATOR -- used where arrows had to be interrupted, or where one input or output is used for several steps. Circles within one diagram with identical letters are regarded as being logically connected.
1	ONPAGE CONNECTOR -- used together with the flow of action symbols to indicate a branch, within the same diagram, to the step number within the circle, regardless of whether the diagram is on more than one page.
	OFFPAGE CONNECTOR -- used for backward reference between diagrams.
	ADDRESS -- indicates, within control blocks, that this field contains the address of the specified field.

Figure 2.1 **Symbols used on method of operation diagrams**

Diagram AA1. Method of Operation Contents



* See VSE/VSAM VSAM Logic, Volume 2

Diagram AB1. VSAM Overview

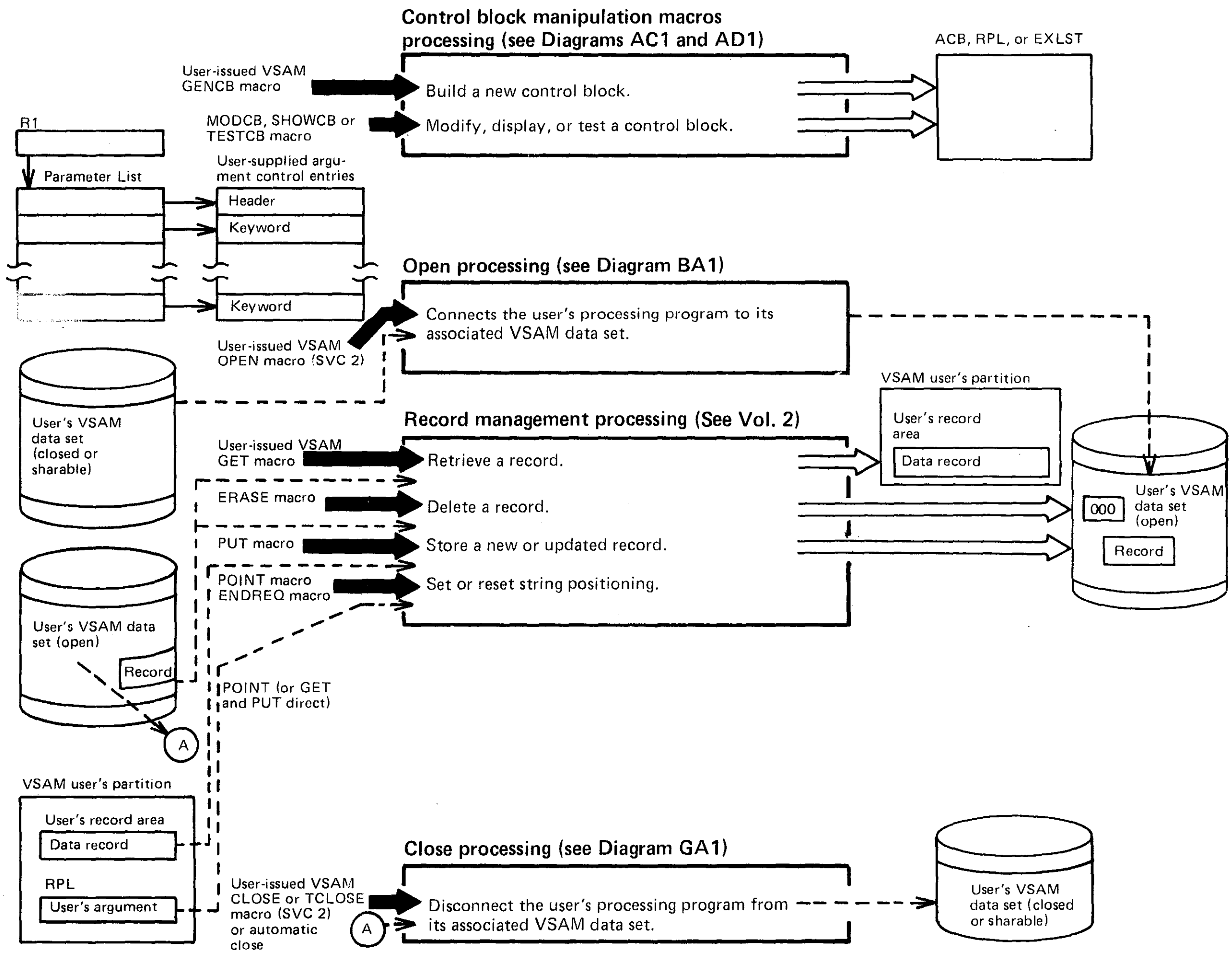
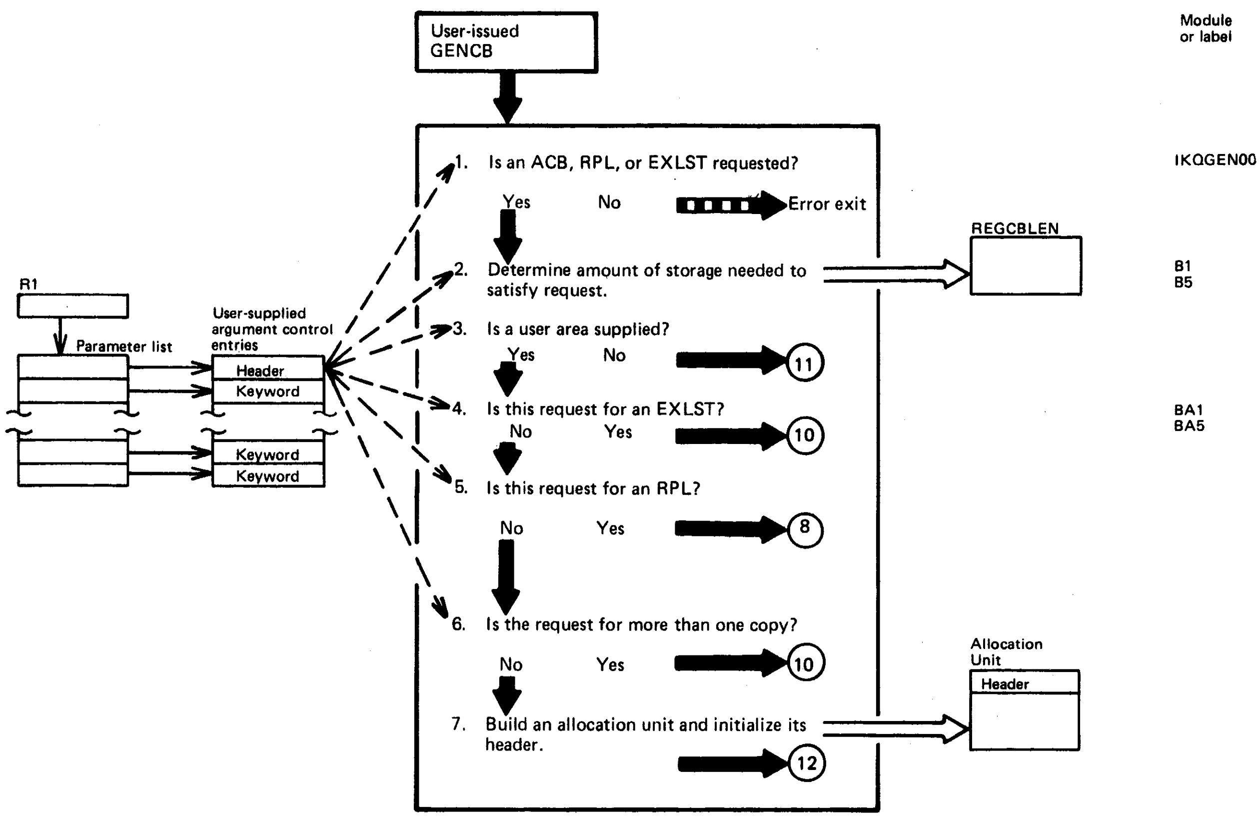


Diagram AC1. GENCB: Build a New Control Block



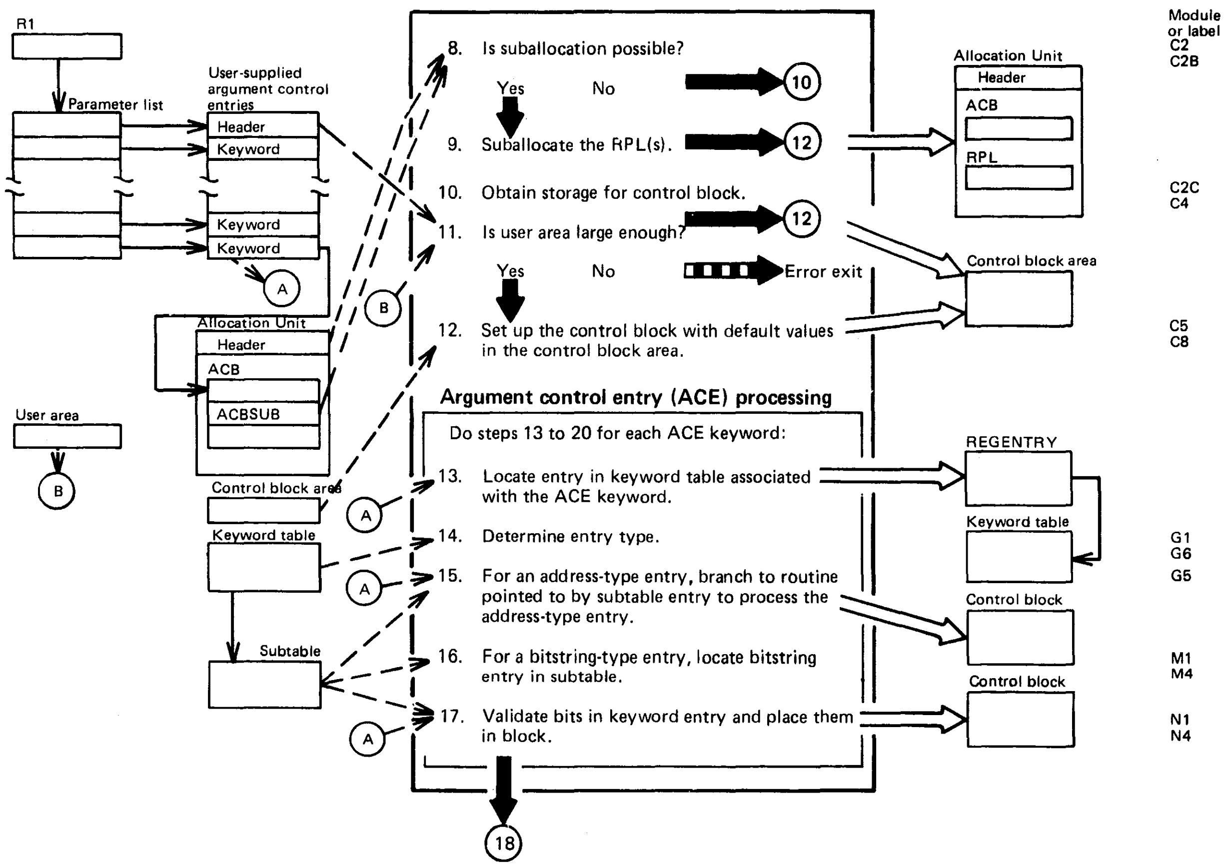
Notes for Diagram AC1

1 The GENCB macro is issued to create an ACB, RPL, or EXLST dynamically. If an ACB, RPL, or EXLST has not been requested, an error exit is taken.

2-3 The ACB and RPL are fixed-length control blocks but the EXLST length is variable. If a user area is supplied for an EXLST, this routine uses a minimum length of ten bytes to find out if the user area is large enough. If a user area is not supplied for an EXLST, this routine calculates the amount of space need for the EXLST and any copies the user requested.

4-7 If only a single ACB is requested, and no user area is supplied, an allocation unit is built to contain an allocation unit header, an ACB, an AMBL, and as many RPLs and PLHs as indicated by the STRNO parameter.

Diagram AC2. GENCB: Build a New Control Block



Notes for Diagram AC2

8-9. If an RPL is requested for an ACB which is suballocated, and if there is sufficient space in the ACB's allocation unit, the RPL is suballocated.

10. The routine issues a GETVIS macro to obtain the required storage for any block for which a user area is not provided.

12. The block is initialized to its default values. Information is subsequently added to the block as specified by the keyword entries.

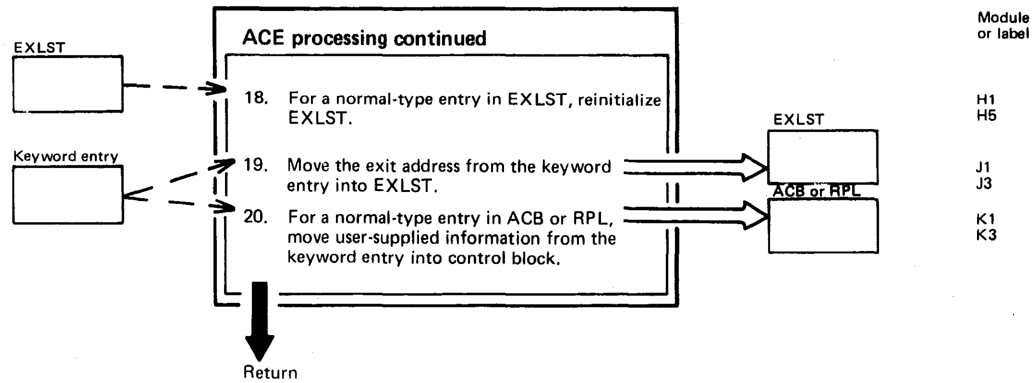
13. For the last argument entry, the high-order bit in the parameter list pointer is 1.

14. Three types of entries are identified in the keyword table: address, bitstring, and normal.

15. If the entry is an address type, the keyword table points to an offset in the subtable that contains a pointer to a routine that generates information to be placed in the appropriate control block field.

16-17. If the entry is a bitstring type, the keyword table points to a series of bit entries in the subtable. The indicated bits are placed in the block.

Diagram AC3. GENCB: Build a New Control Block



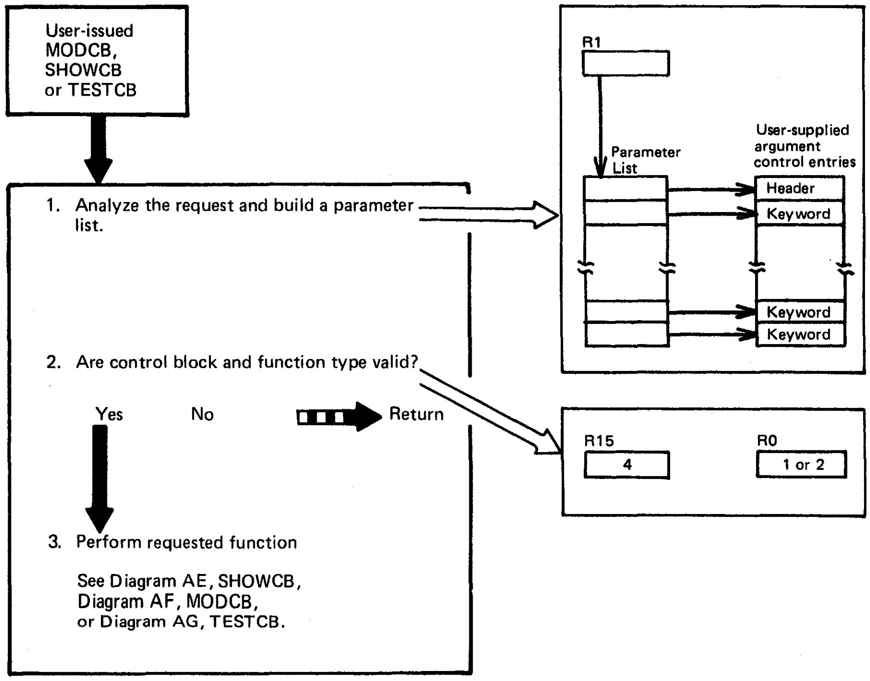
Notes for Diagram AC3

18-19. The size of the EXLST is adjusted to the actual amount of space required to satisfy the user request. If the user-supplied area is not large enough, a return is made to the user.

When the last keyword has been processed, any surplus space obtained by a GETVIS is reclaimed (surplus user-supplied space is not reclaimed). As each keyword is processed, the EXLST is checked to see if space is available at the proper offset for that keyword. If no slot is available, a GETVIS obtains additional space, provided the space was originally obtained by a GETVIS.

20. If the entry is a normal type, the value in the argument control entry is moved into the block.

Diagram AD1. MODCB, SHOWCB, TESTCB: Modify, Display, or Test a Control Block



Notes for Diagram AD1

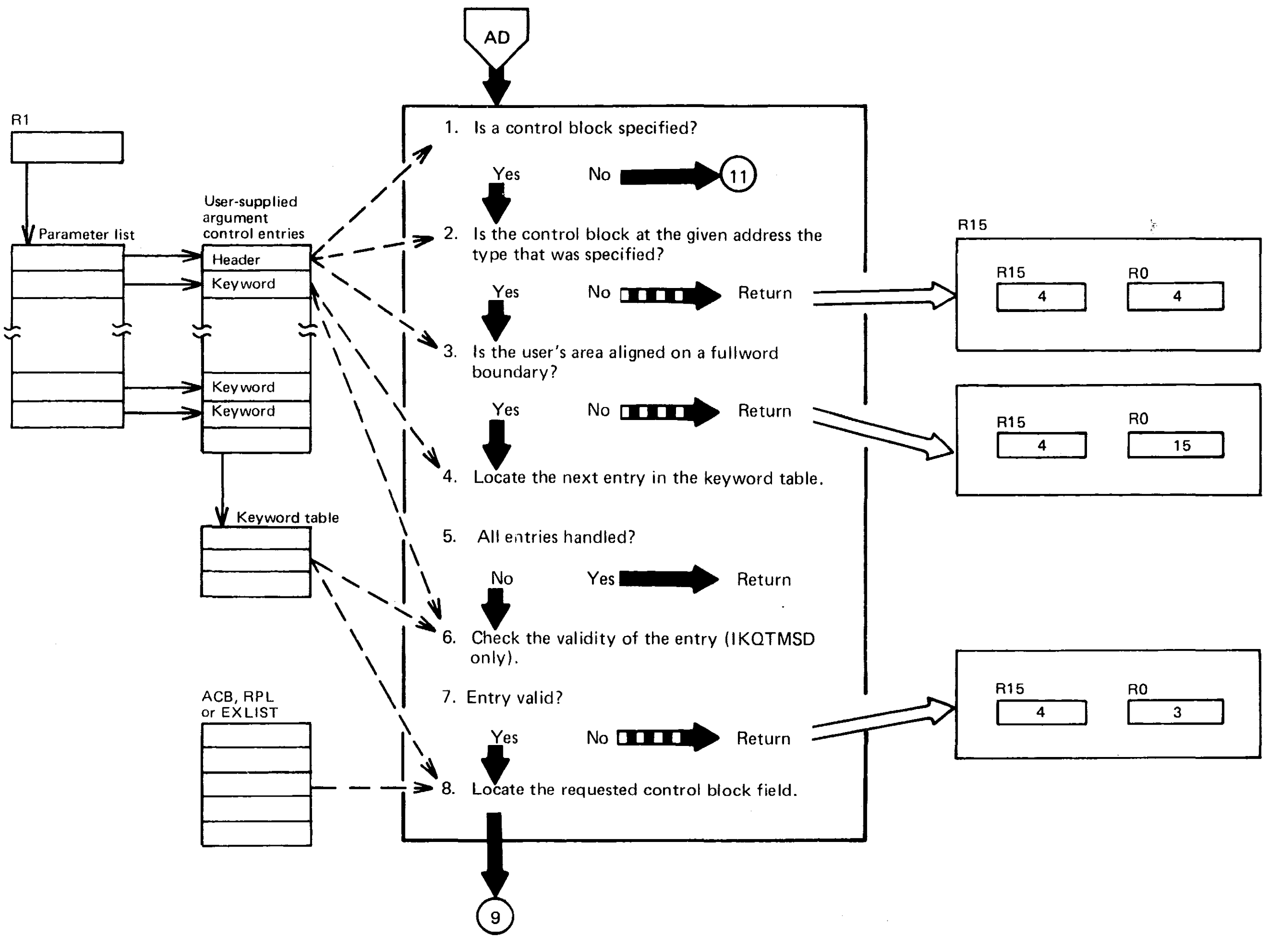
The MODCB, SHOWCB, and TESTCB macros are issued to modify, display, and test the ACB, RPL, or EXLST.

The macros are analyzed by the macro IKQCB2. If this macro actually builds the parameter list, it can ensure that the parameters are valid. In this case, control is then passed to the module IKQTMSF (F = fast = without diagnostics).

If the user provides a ready-built parameter list, the macro IKQCB2 cannot check the validity of the parameters and passes control to the module IKQTMSD (D = diagnostic).

Each module contains routines for processing all possible combinations of request and control blocks (such as SHOWCB for ACB, TESTCB for an RPL, etc.).

Diagram AE1. SHOWCB: Display a Control Block



Notes for Diagram AE1

The processing shown here can take place in module IKQTMSF or IKQTMSD (see notes for Diagram AD).

The subroutines, which have identical names in both modules, are:

Block	Label
ACB	SHOWACB
RPL	SHOWRPL
EXLST	SHOWEXL
None	SHOWNOB

Diagram AE2. SHOWCB: Display a Control Block

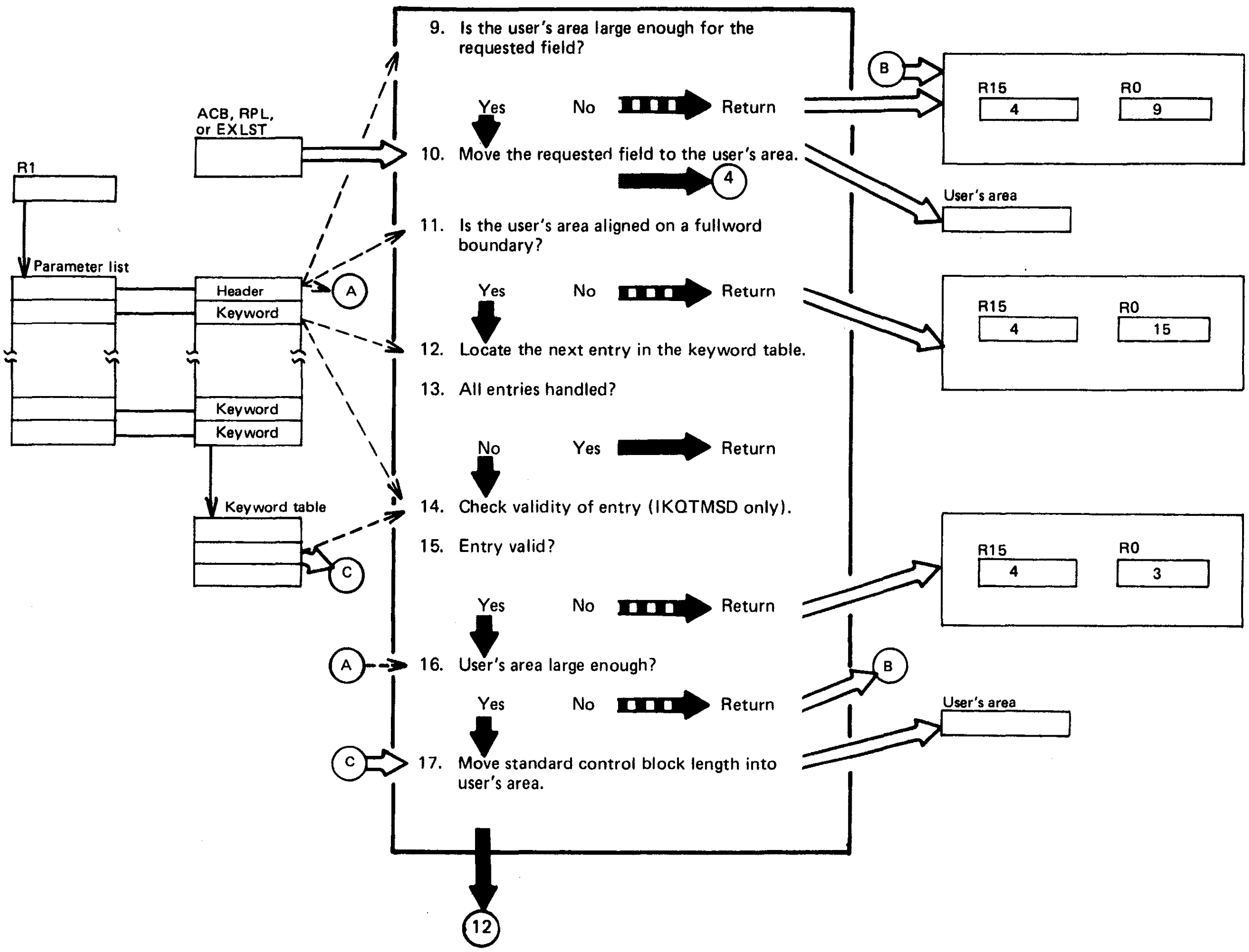
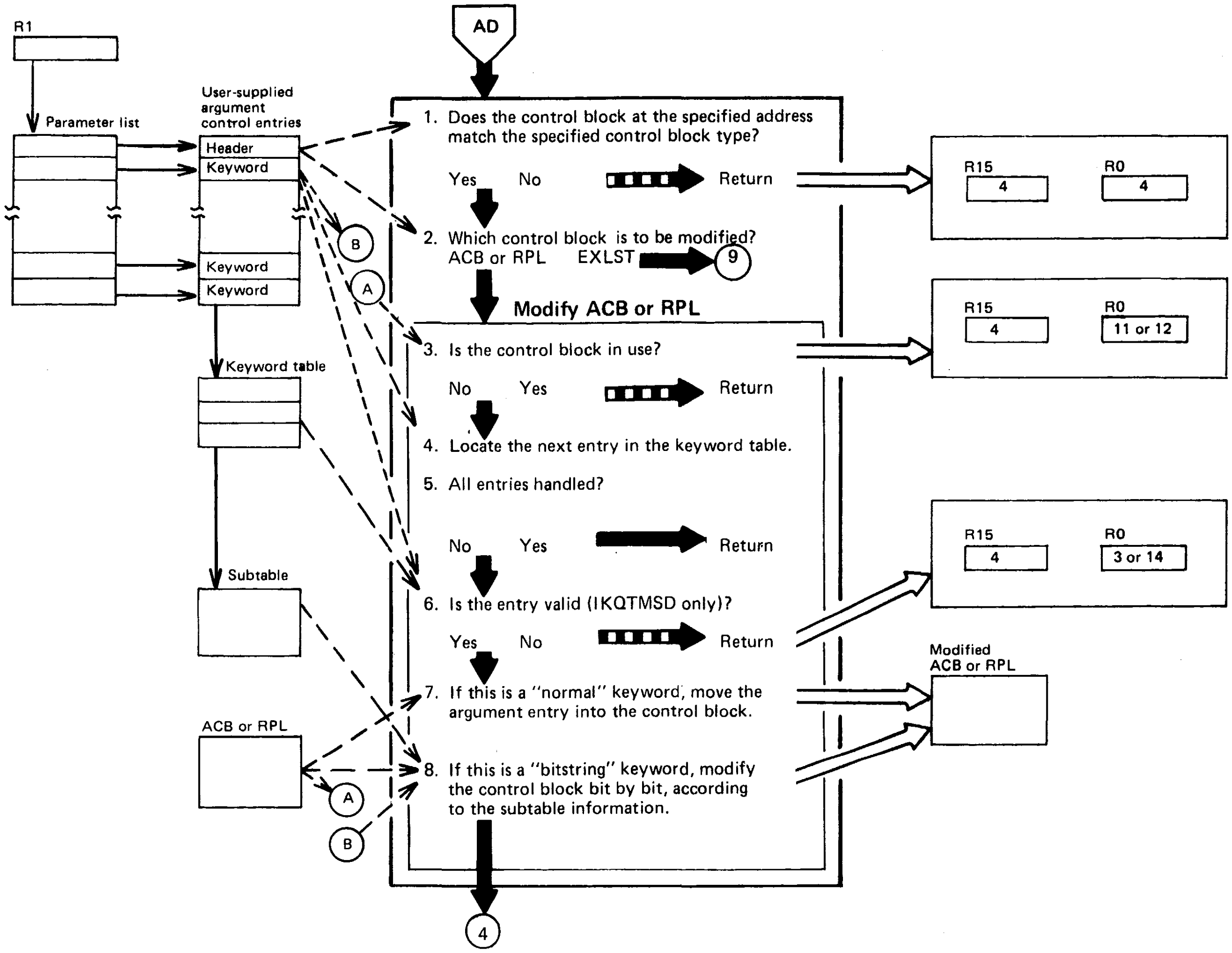


Diagram AF1. MODCB: Modify a Control Block



Notes for Diagram AF1

The processing shown here can take place in module IKQTMSF or IKQTMSD (see notes for Diagram AD).

The subroutines used for MODCB, which have identical names in both modules, are:

Block	Label
ACB	MODACB
RPL	MODRPL
EXLST	MODEXL

Diagram AF2. MODCB: Modify a Control Block

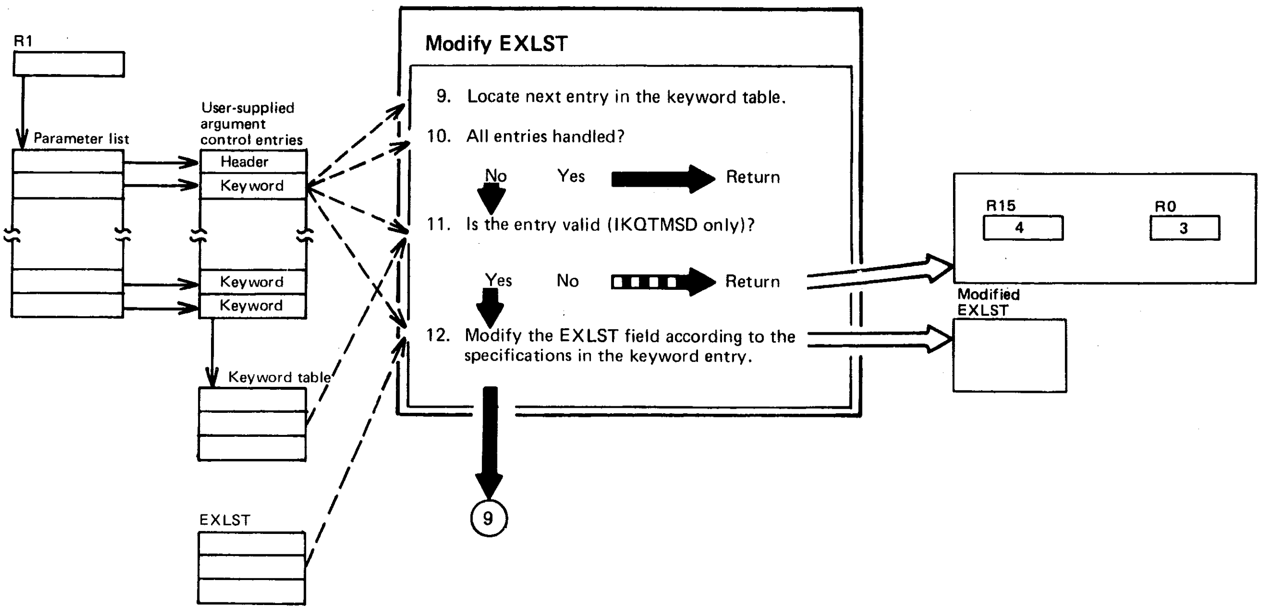
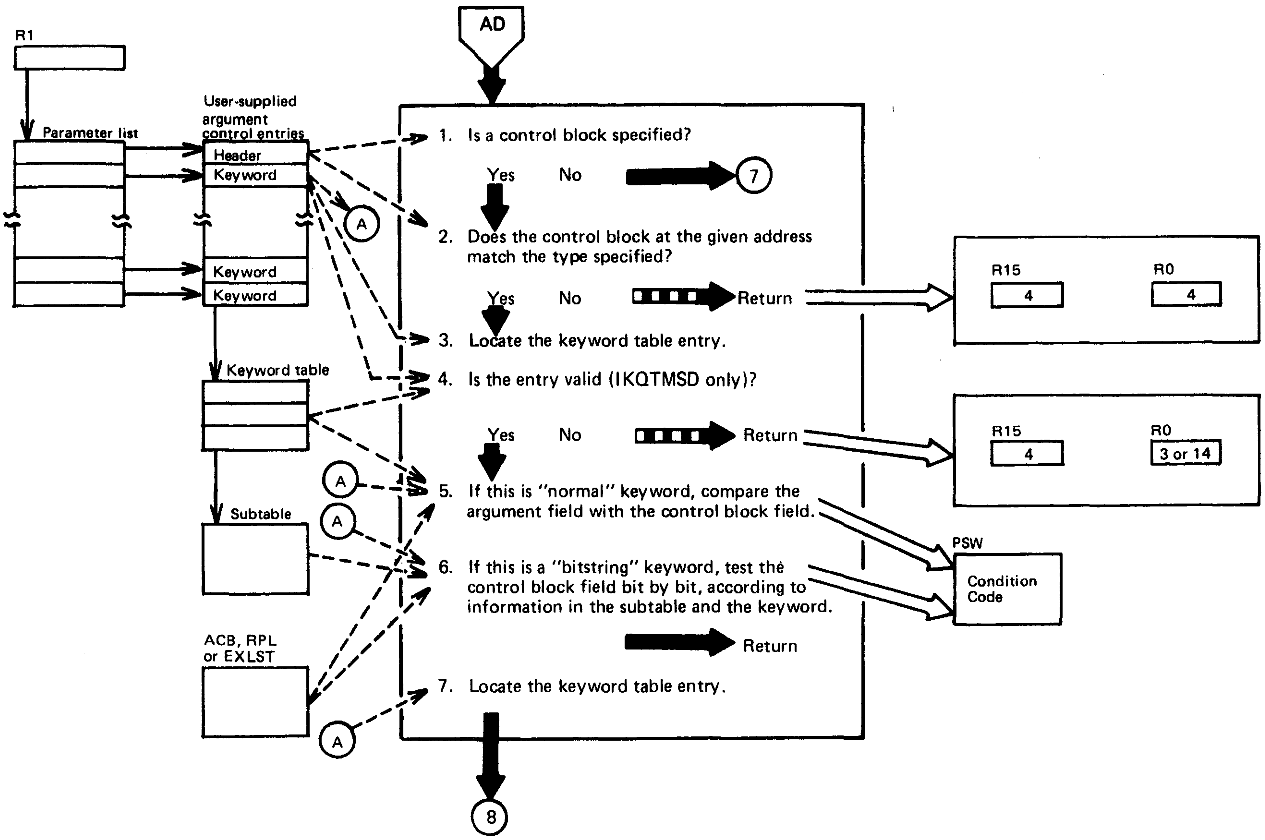


Diagram AG1. TESTCB: Test a Control Block



Notes for Diagram AG1

The processing shown here can take place in module IKQTMSF or IKQTMSD (see notes for Diagram AD).

The subroutines used for TESTCB, which have identical names in both modules, are:

Block	Label
ACB	TESTACB
RPL	TESTRPL
EXLST	TESTEXL
None	TESTNOB

Diagram AG2. TESTCB: Test a Control Block

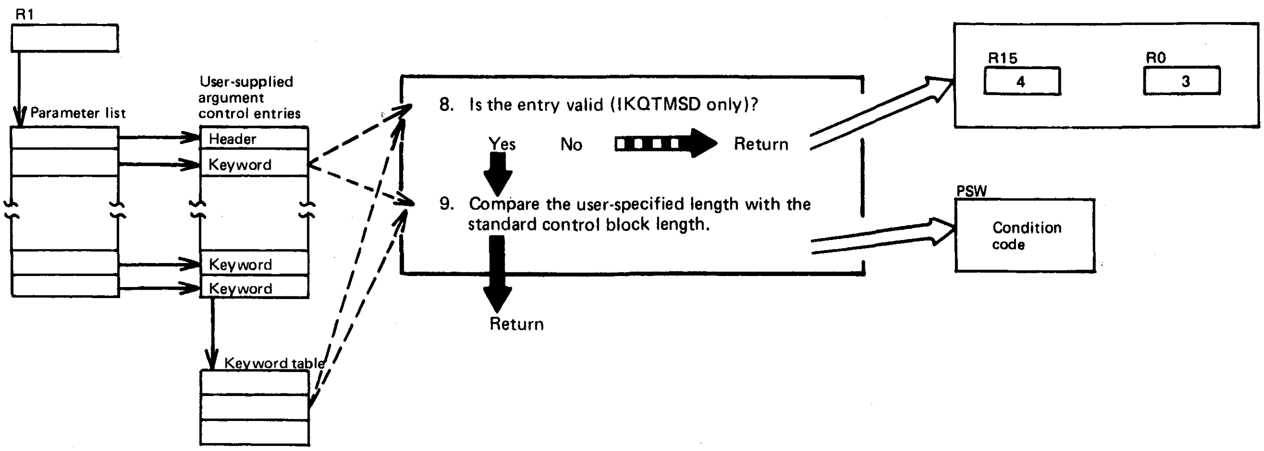
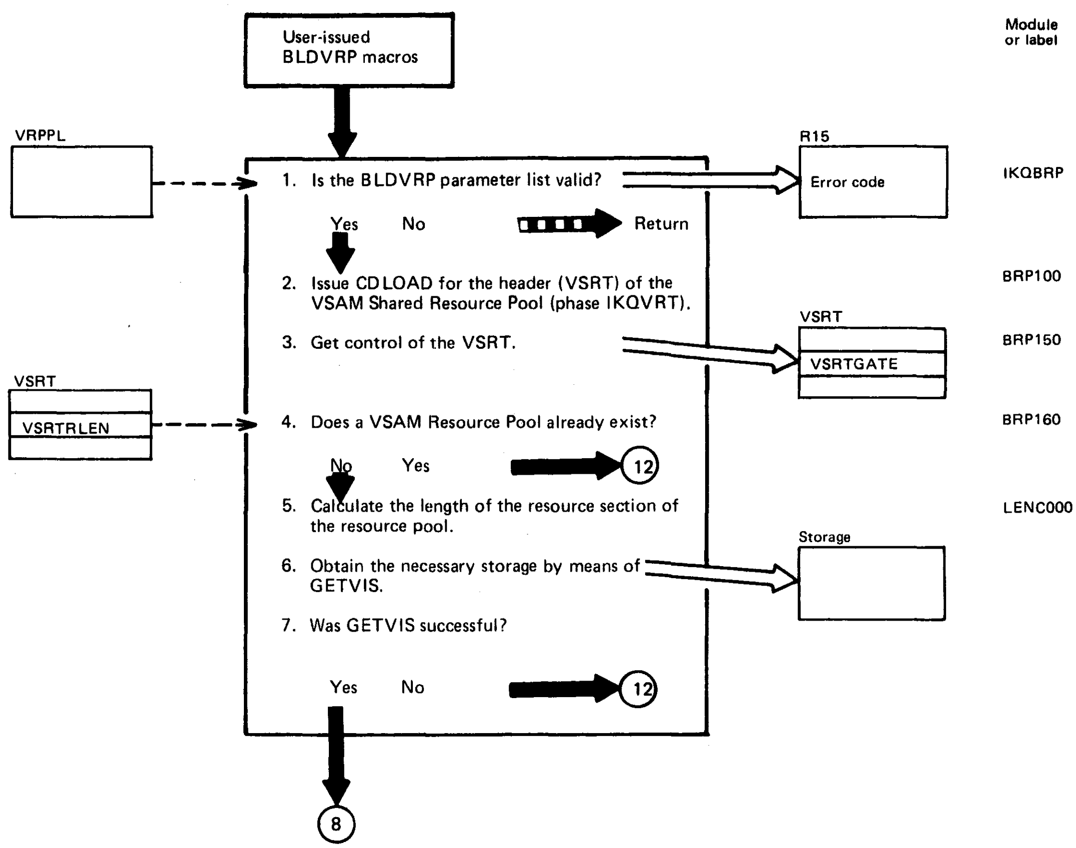


Diagram AH1. BLDVRP: Build VSAM Shared Resource Pool



Notes for Diagram AH1

2. The phase IKQVRT is an assembled CSECT that contains only the VSRT.

Diagram AH2. BLDVRP: Build VSAM Shared Resource Pool

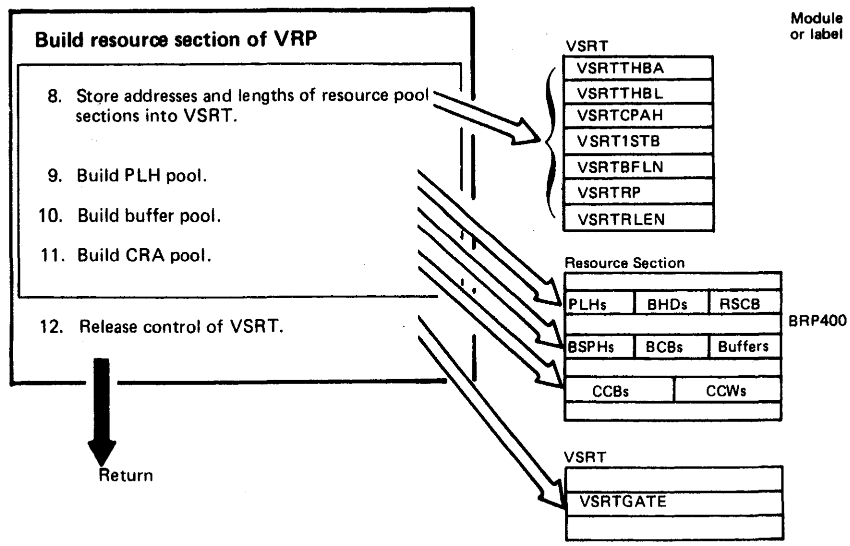
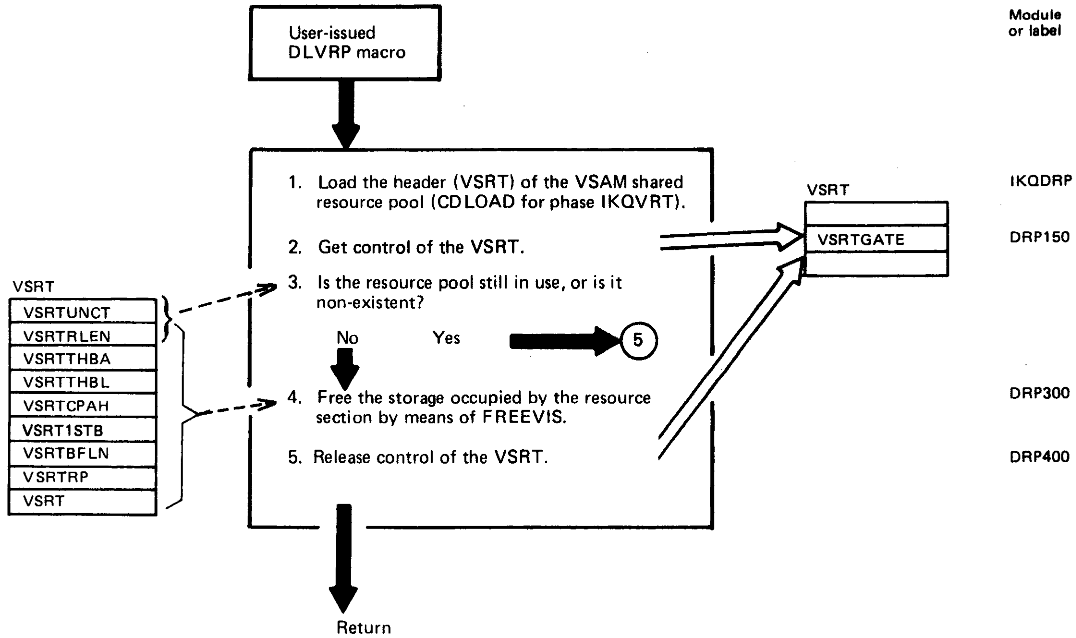


Diagram AI1. DLVRP: Delete VSAM Shared Resource Pool



Notes for Diagram AI1

1. For an existing VSRT, CDLOAD simply provides addressability to the VSRT.

Diagram BA1. OPEN: Connect a User's Program to a VSAM Data Set

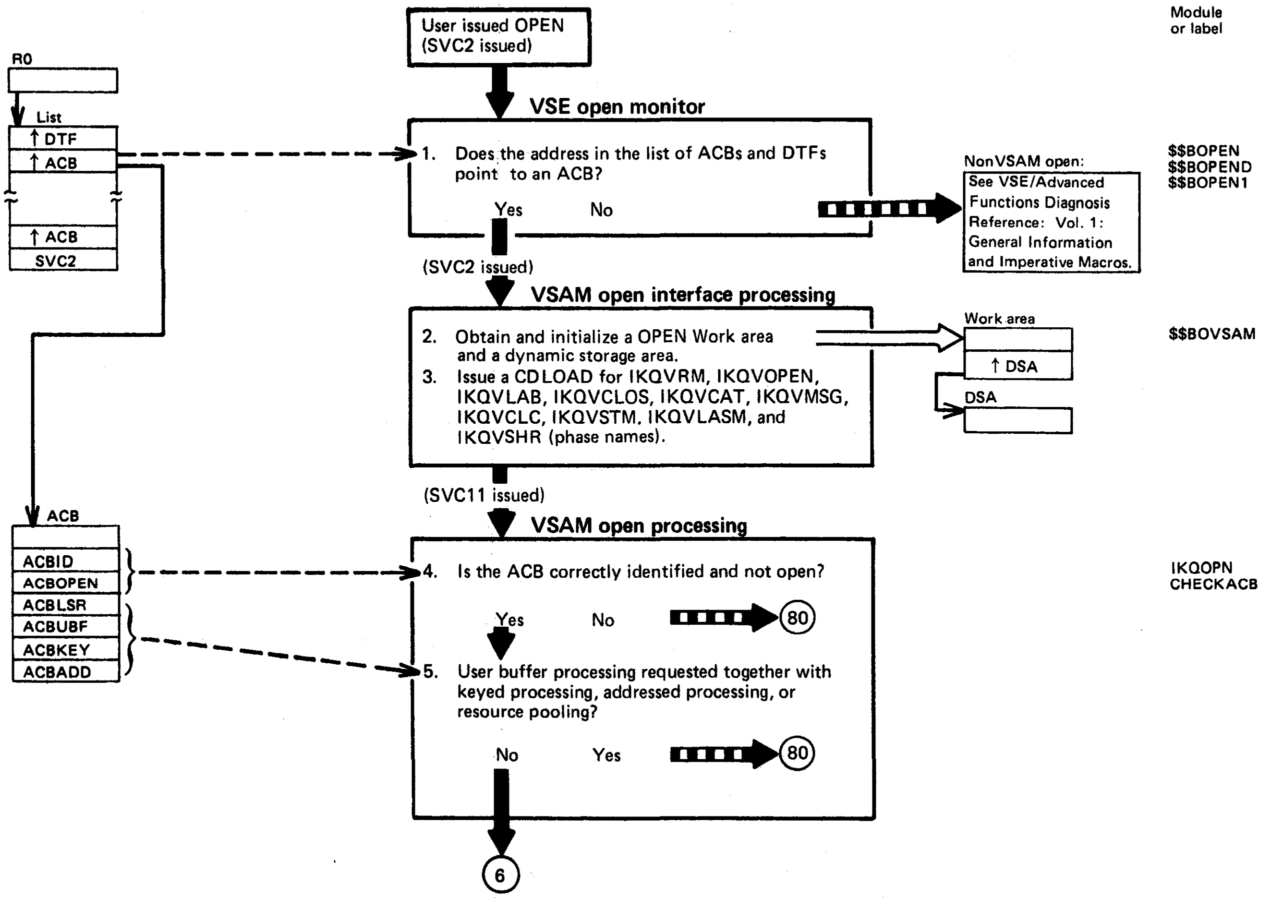


Diagram BA2. OPEN: Connect a User's Program to a VSAM Data Set

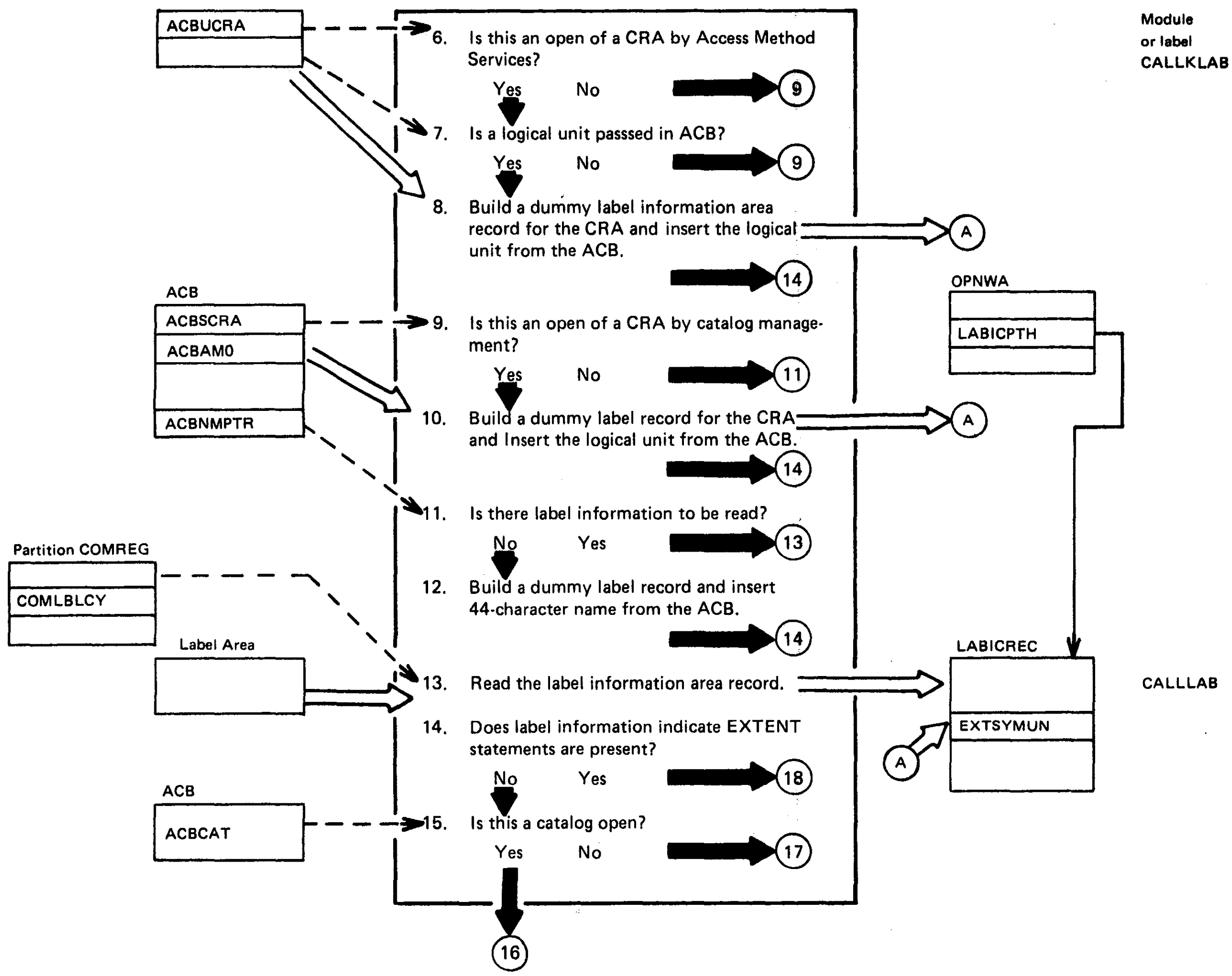


Diagram BA3. OPEN: Connect a User's Program to a VSAM Data Set

Module or label

ACB

ACBCAT
ACBDDNAM
ACBUCRA
ACBUAPTR

COMREG

COMFPROT

16. Move logical unit specification from CAXWA to label area record.

17. Put unspecified indication for logical unit in label area record.

18. Is this a catalog OPEN?
    Yes — No → 21

19. Is a unit specified for the catalog?
    No — Yes → 21

20. Insert SYSCAT as the symbolic unit. → A

21. Resolve file disposition from specifications in ACB and JCL. Check validity of file disposition.

22. Is the file name a default model name?
    No — Yes → 80

23. Is this a catalog open or CRA OPEN for recovery?
    Yes — No → 36

24. Is there a CAXWA?
    No — Yes → 26

25. Build and Initialize CAXWA.

26. Build channel program to read labels.

27. If DASD file protect is specified, set up extent block to allow access to entire catalog volume.

→ 28

ACB

ACBUAPTR

SVASSOC

CAXWA

CAXDRWP

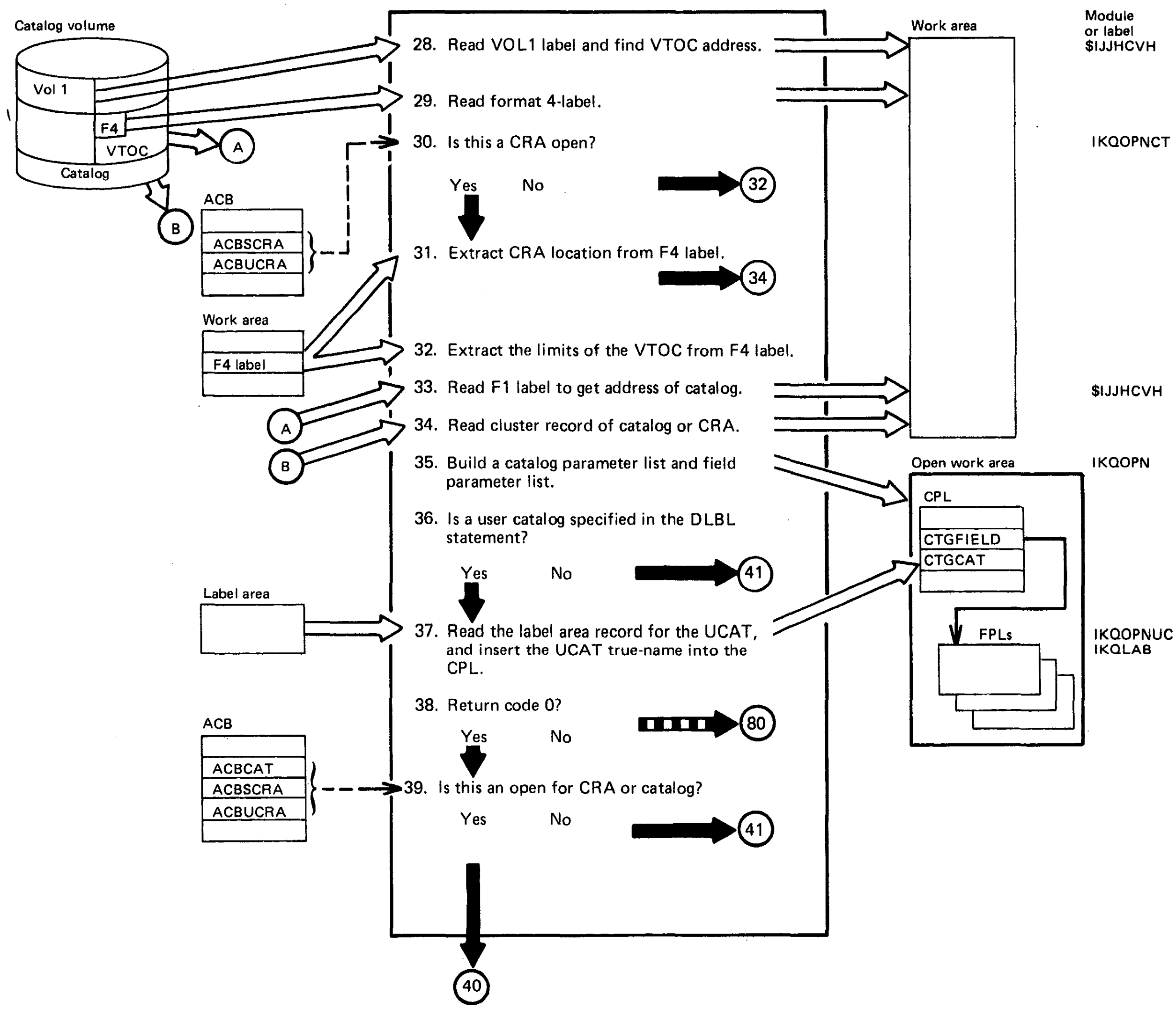
IKQOPNCT

Work area

CCB
CCW

IKQJIBSM

Diagram BA4. OPEN: Connect a User's Program to a VSAM Data Set

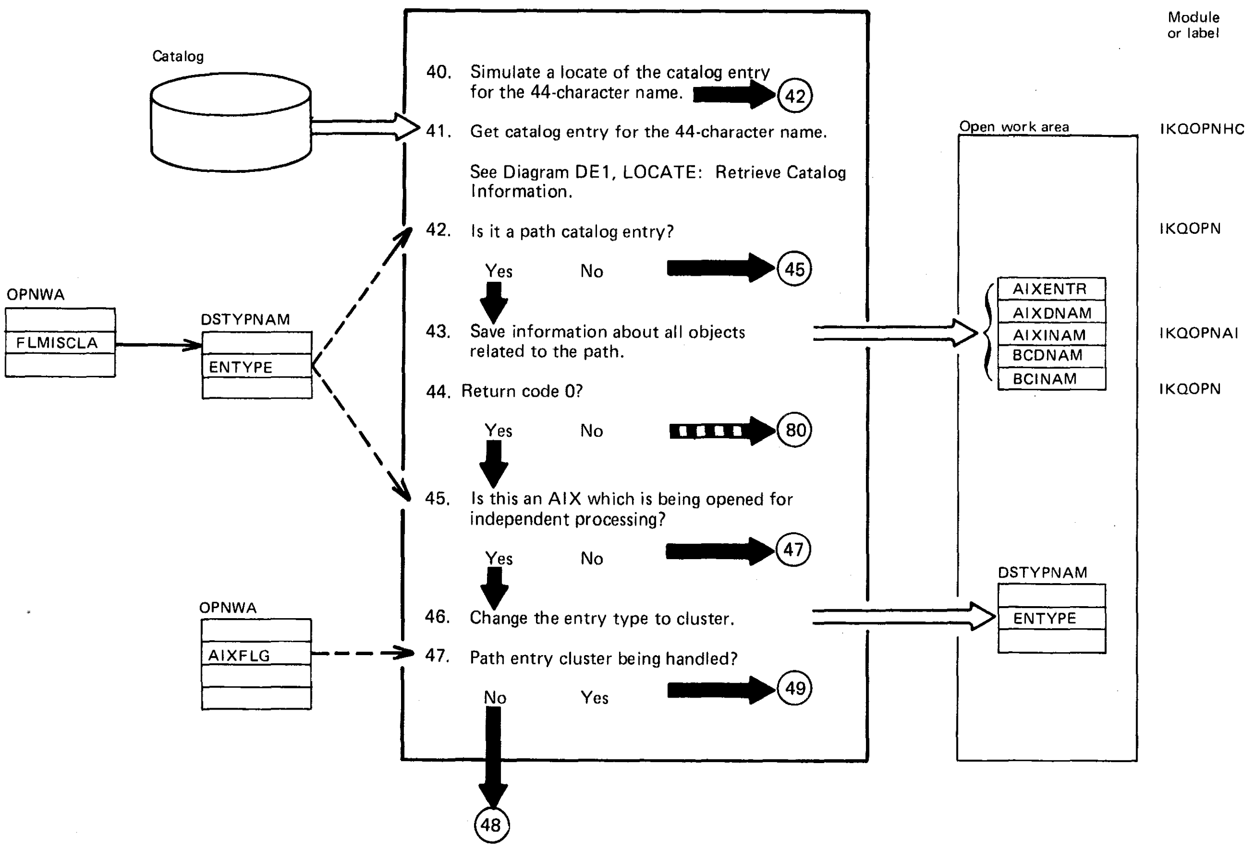


Notes for Diagram BA4

28. If the user catalog is not mounted before the first job is run that opens that catalog, the job is terminated.

35. The contents of the following combination names are fetched from the catalog:
NAMEDS
DSTYPNAM

Diagram BA5. OPEN: Connect a User's Program to a VSAM Data Set

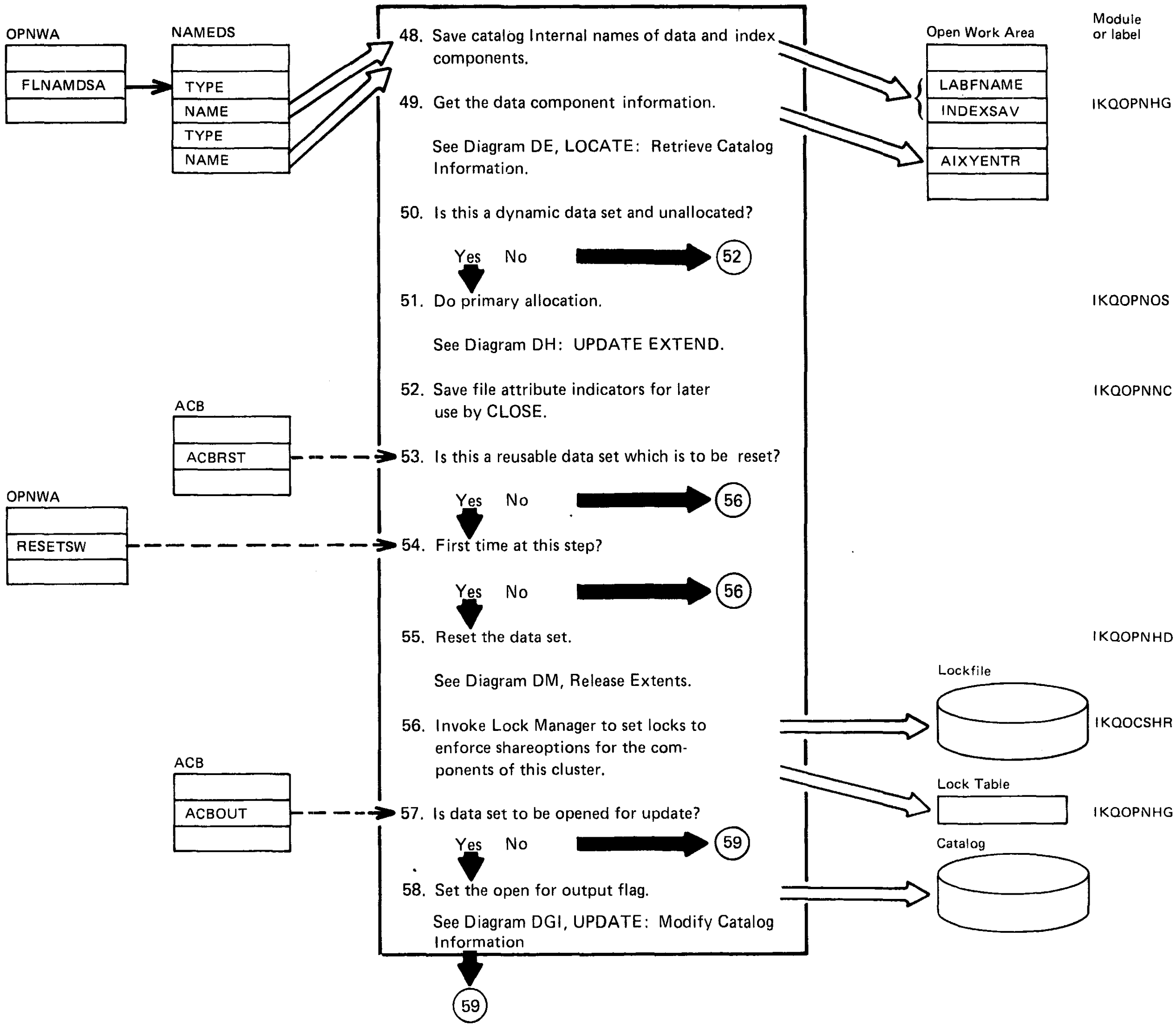


Notes for Diagram BA5

41. The contents of the following fields and combination names are fetched from the catalog:
    NAMEDS
    DSTYPNAM
    RGATTR
    CATACB

47. Path entry cluster is an AIX that is part of a VSAM path.

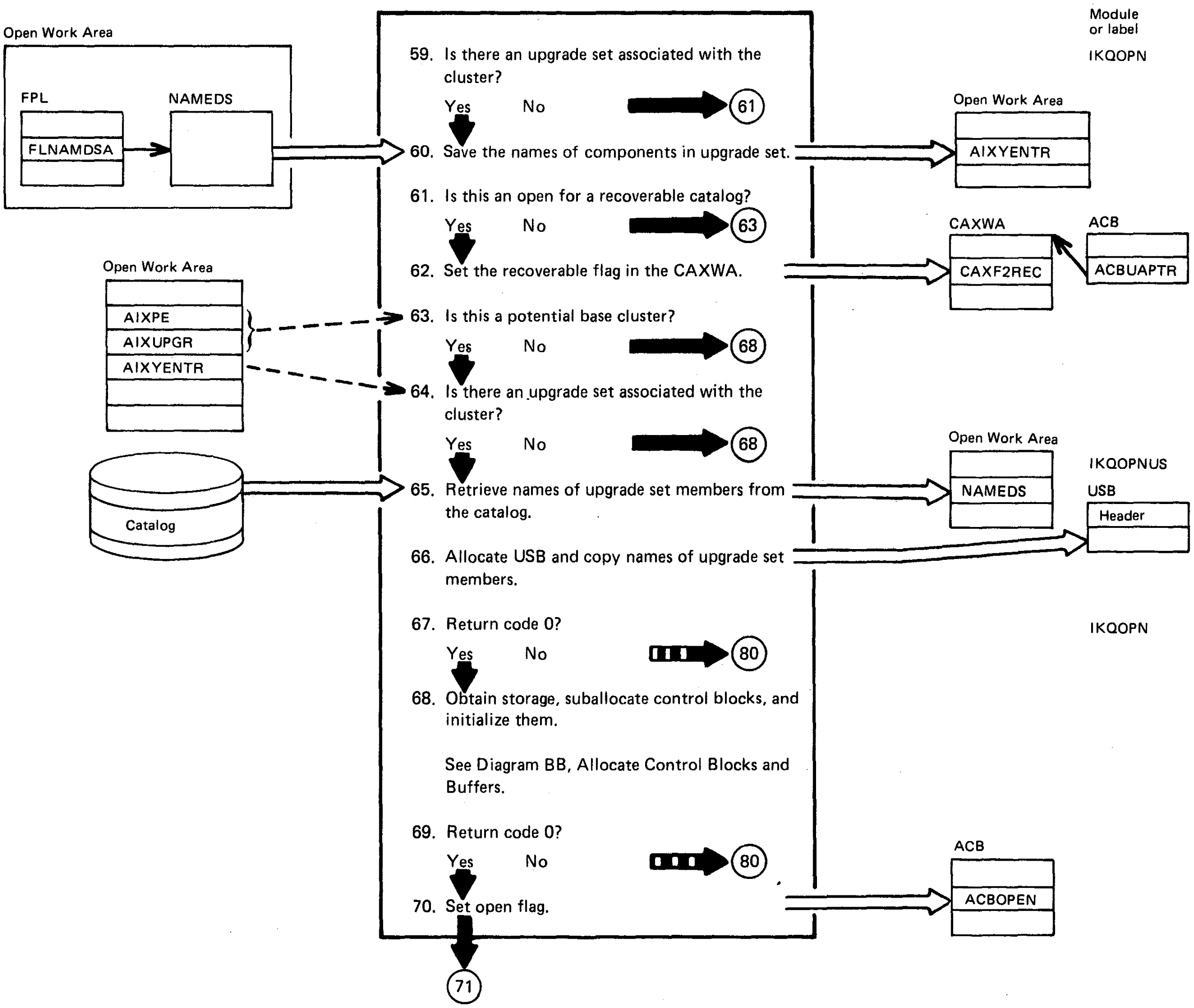
Diagram BA6. OPEN: Connect a User's Program to a VSAM Data Set



Notes for Diagram BA6

49. The contents of the following fields and combination names are fetched from the catalog:
    AMDSBCAT
    FNTVOL
    DSATTR
    OPENIND
    BUFSIZE
    HURBADS
    USERINFO
    NAMEDS
    EXCPEXIT
    DSETEXDT
    CATACB

Diagram BA7. OPEN: Connect a User's Program to a VSAM Data Set



Notes for Diagram BA7

63. Base cluster is a cluster that is not PE or UPGR.

64. The contents of combination name NAMEDS are fetched from the catalog.

Diagram BA8. OPEN: Connect a User's Program to a VSAM Data Set

		Module or label
ACB: ACBCRA, ACBCAT, ACBUCRA, ACBOPEN	71. Is this an open for catalog or CRA? Yes / No → 73	OPNRAPUP
	72. Indicate to catalog that open is complete. See Diagram DE, LOCATE: Dummy locate of data set in catalog.	
OPNWA: AIXFLG, USACBAD	73. Is an AIX structure being handled and current open completed correctly? Yes / No → 76	NEXT CLU
	74. Switch to next cluster. See Diagram BC, Switch to Next Cluster.	IKQOPNNC
OPNWA: AIXBASE, AIXMUS	75. Is the base cluster of a path or an upgrade set member still to be opened? No / Yes → 49	
ACB: ACBSCRA, ACBCAT, ACBUCRA, ACBOPEN	76. Is this an open for catalog or CRA? No / Yes → 79	
	77. Is ACBOPEN set? Yes / No → 79	
	78. Build an entry in the OAL. → OAL	
	→ 79	

Notes for Diagram BA8

72. This releases resources obtained by the catalog.

74. This can be a base cluster of a path or upgrade set member(s) for a base cluster.

Diagram BA9. OPEN: Connect a User's Program to a VSAM Data Set

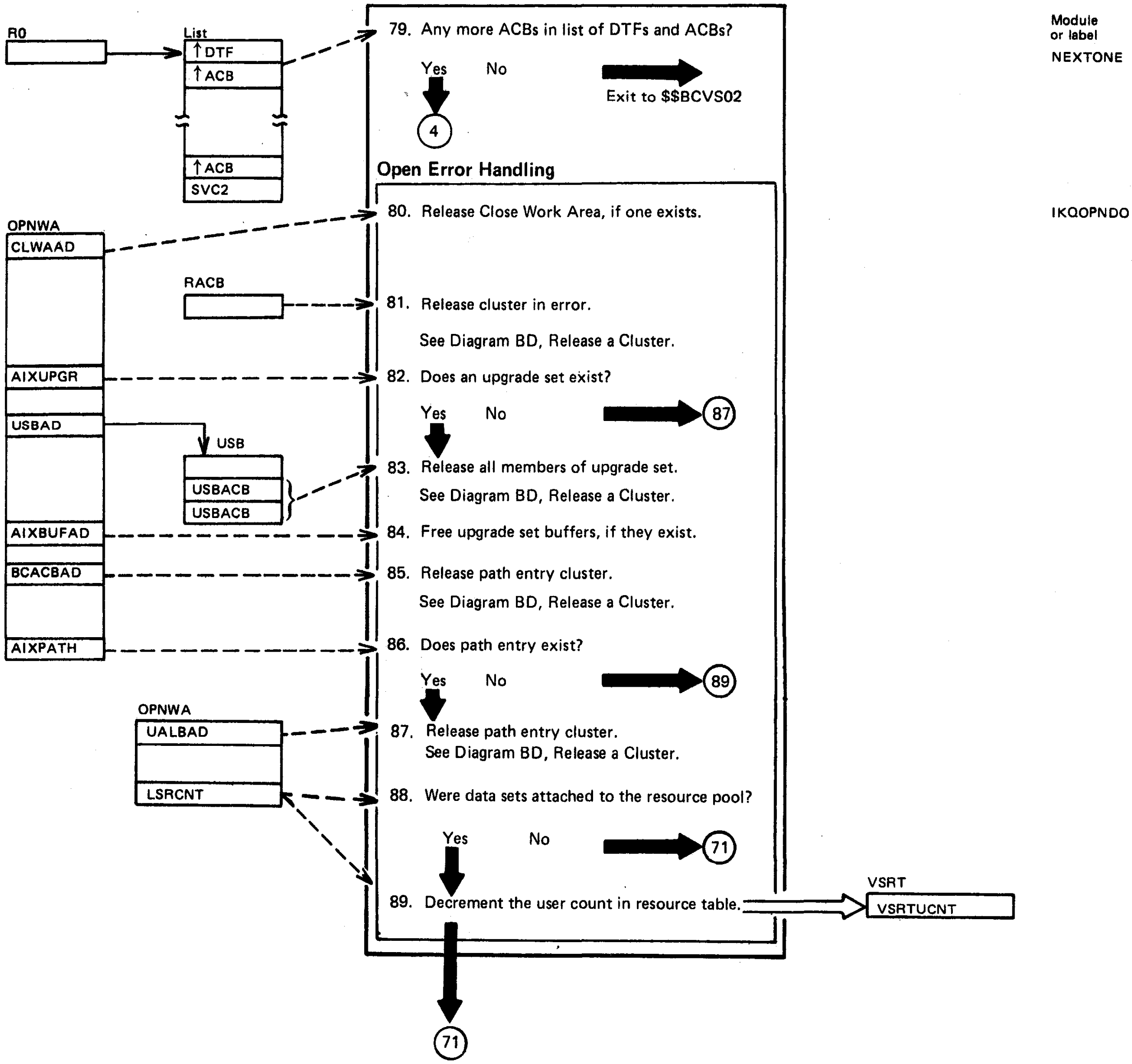
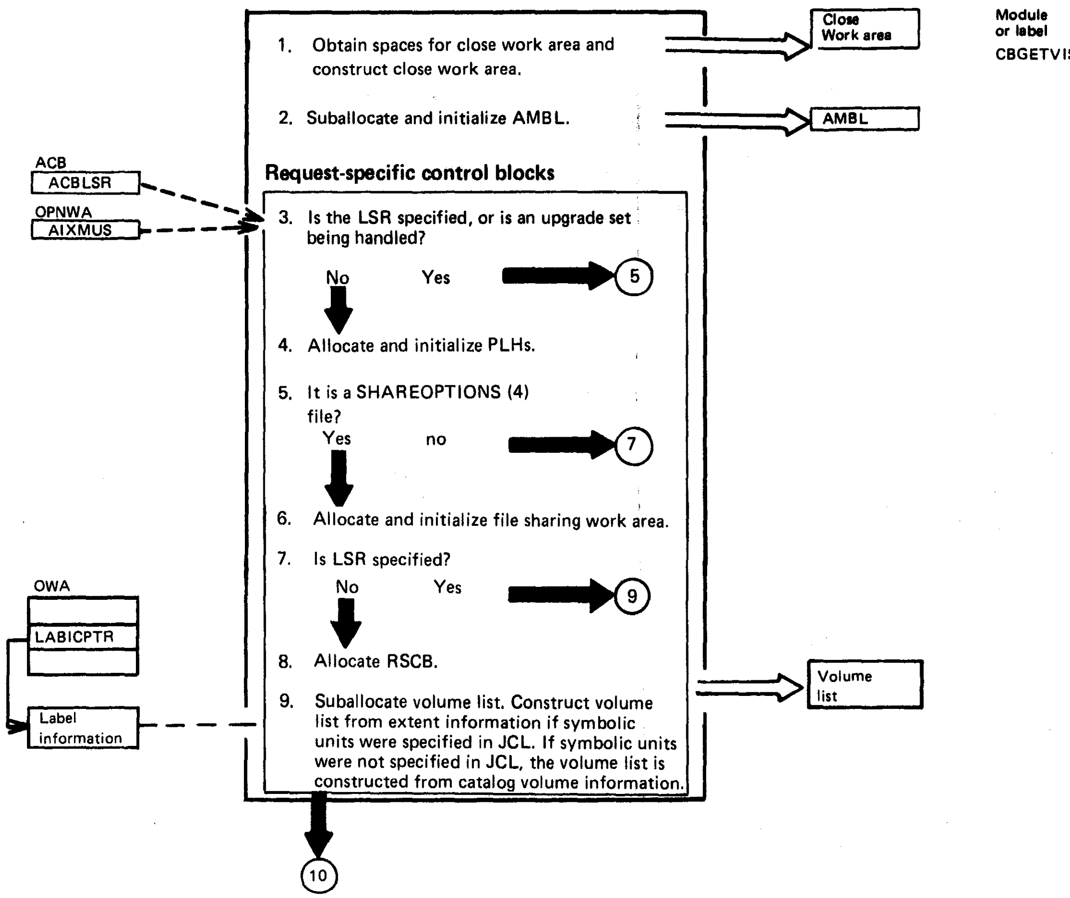


Diagram BB1. Allocate Control Blocks and Buffers



Notes for Diagram BB1

4. PLHs for data sets with LSR already exist in the resource pool. Members of an upgrade set share one PLH which was allocated by IKQOPNNC. The RSCB is not allocated for LSR because it is a member of the resource pool.

Diagram BB2. Allocate Control Blocks and Buffers

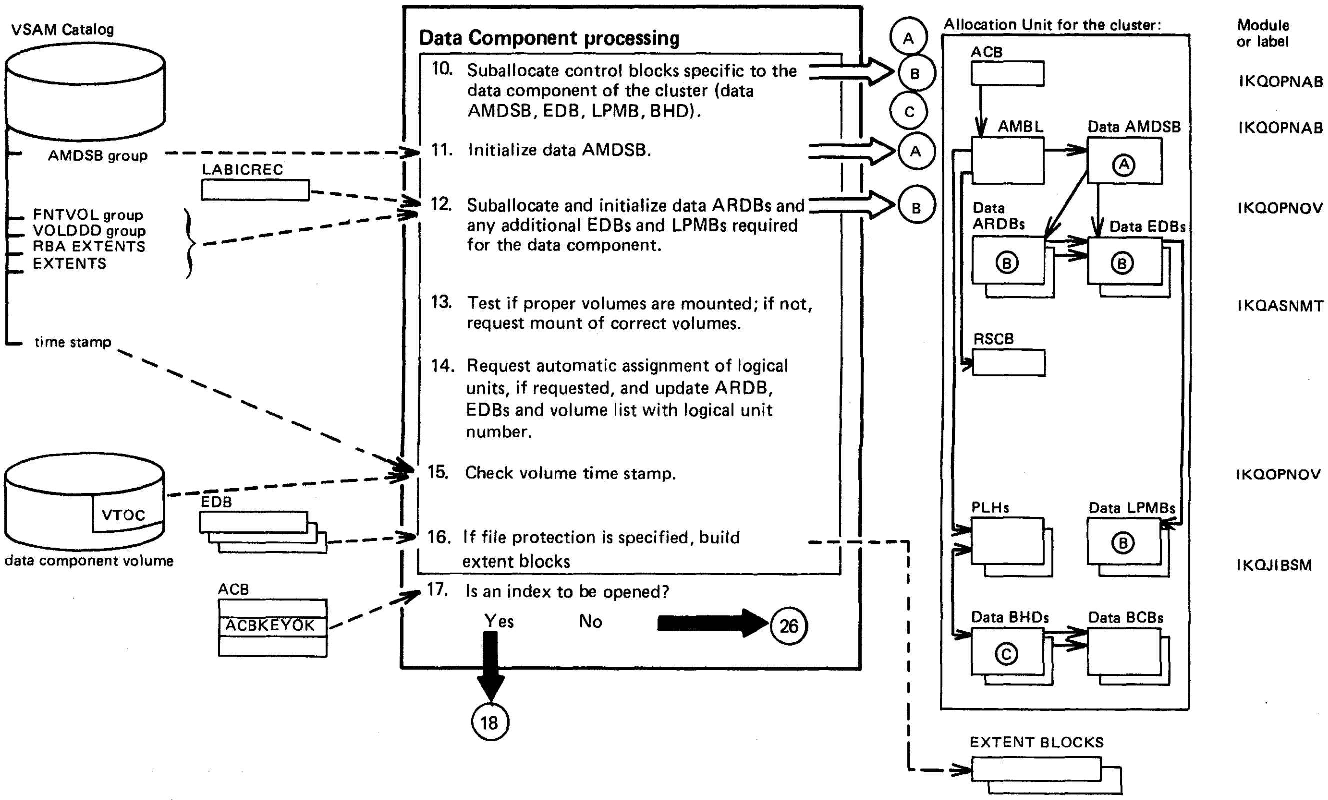


Diagram BB3. Allocate Control Blocks and Buffers

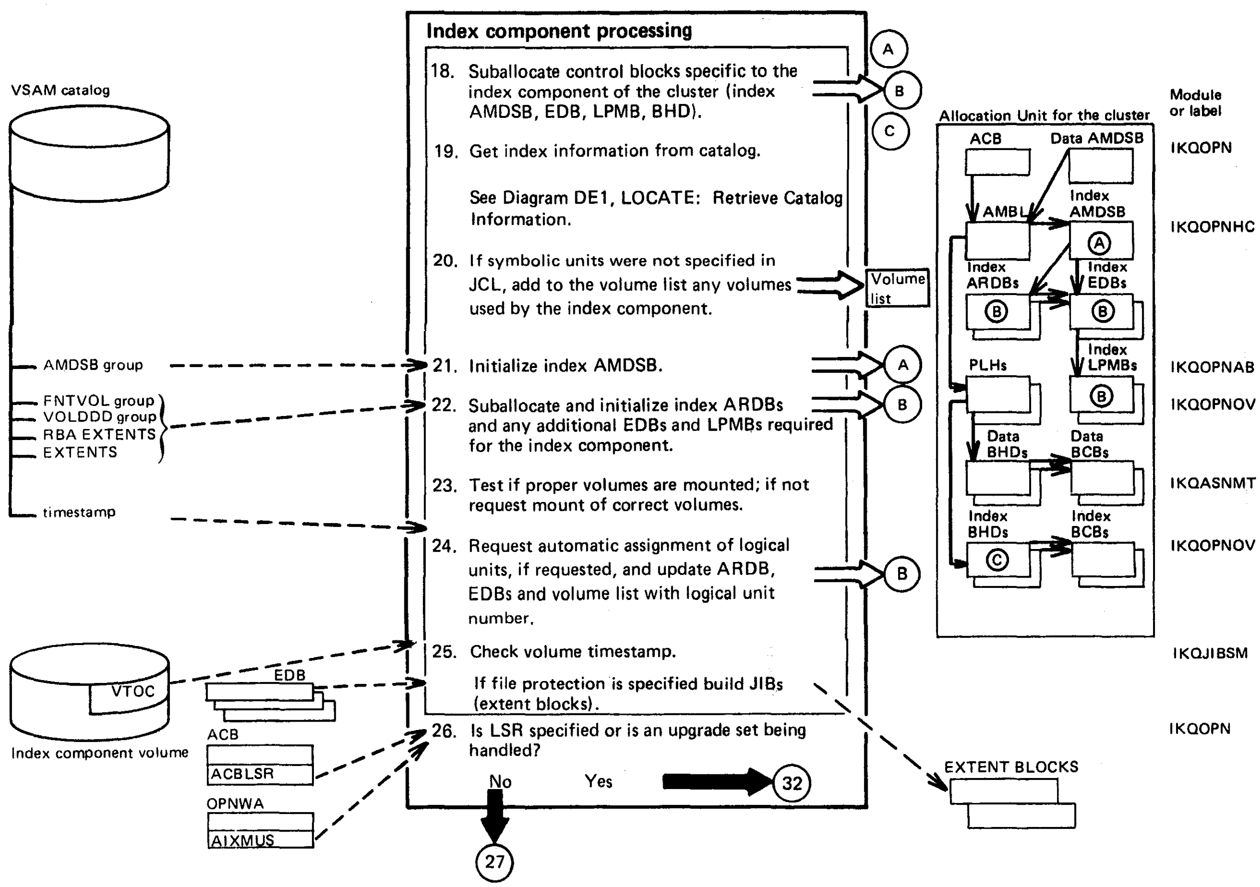
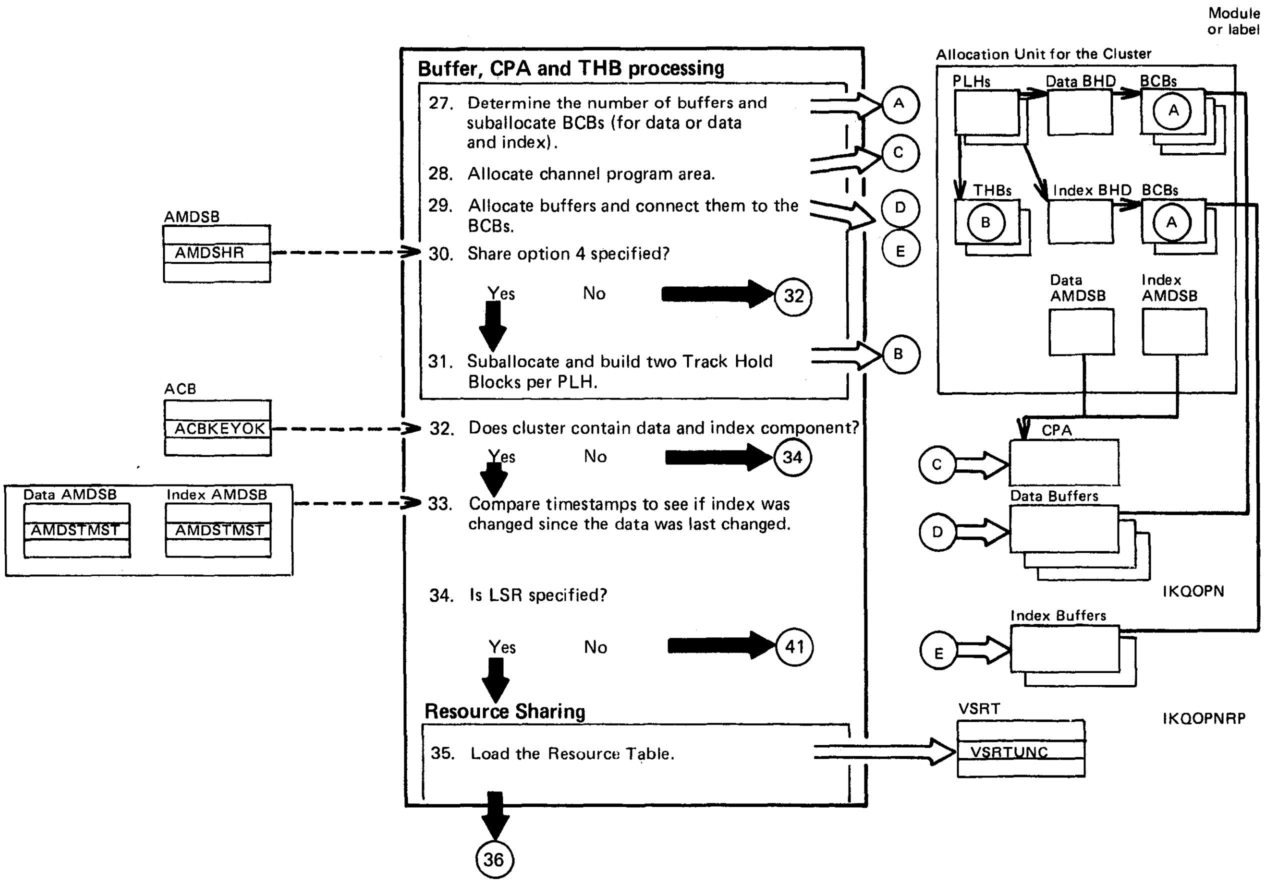


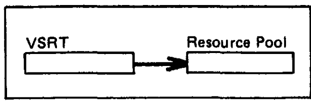
Diagram BB4. Allocate Control Blocks and Buffers



Notes for Diagram BB4

30. If the data set was defined SHAREOPTIONS (4) but the supervisor was not generated with track hold support, then the AMDSB is altered in storage to reflect SHAREOPTIONS (2). The AMDSB group occurrence in the catalog is not modified.

Diagram BB5. Allocate Control Blocks and Buffers



37. Attach PLH-pool to the cluster.

38. Attach buffer subpool to the cluster (data or data and index).

39. Attach channel program area.

40. Increase the use count.

41. Return.

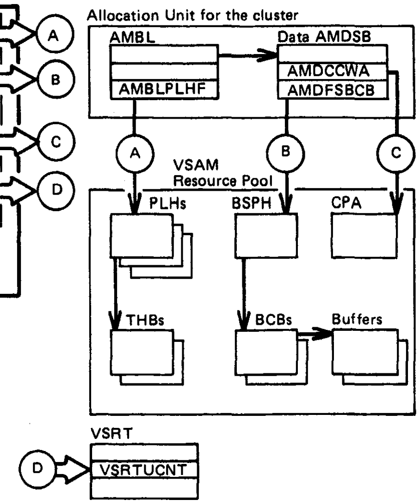


Diagram BC1. Switch to Next Cluster

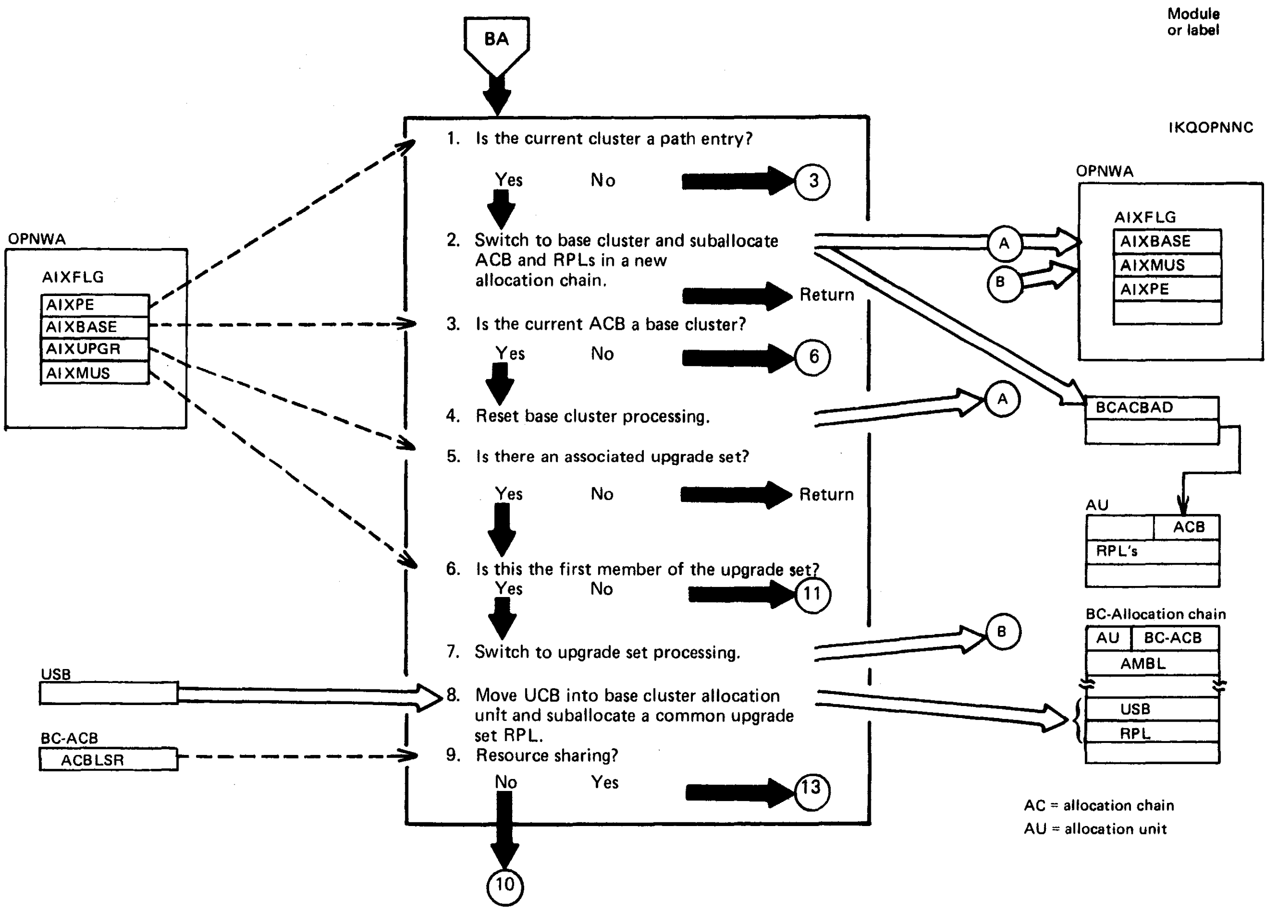
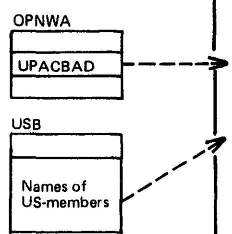


Diagram BC2. Switch to Next Cluster



10. Suballocate common US control blocks in base cluster allocation unit. → 14

11. Resource sharing?
    No ↓ Yes → 13

12. Attach common upgrade set control blocks to the control blocks for this upgrade set member.

13. Attach this upgrade set member to the USB.

14. Search for next (first) member of upgrade set.

15. Have all members of the upgrade set been handled?
    No ↓ Yes → 17

16. Switch to next member, and suballocate ACB in a new allocation chain.
    → Return

17. Resource sharing?
    No ↓ Yes → 19

18. Allocate minimum number of common upgrade set buffers, and attach them to the common upgrade set control blocks.

19. Reset upgrade set handling flag.

→ Return

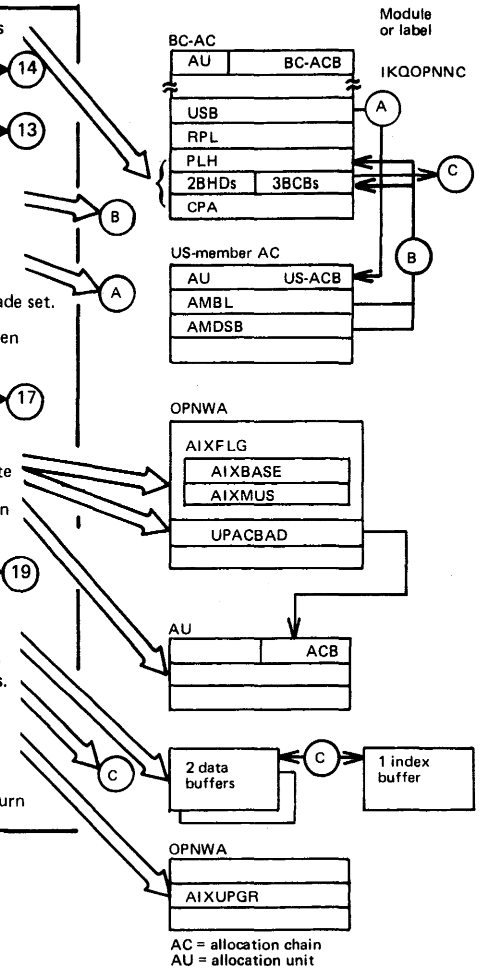


Diagram BD1. Release a Cluster

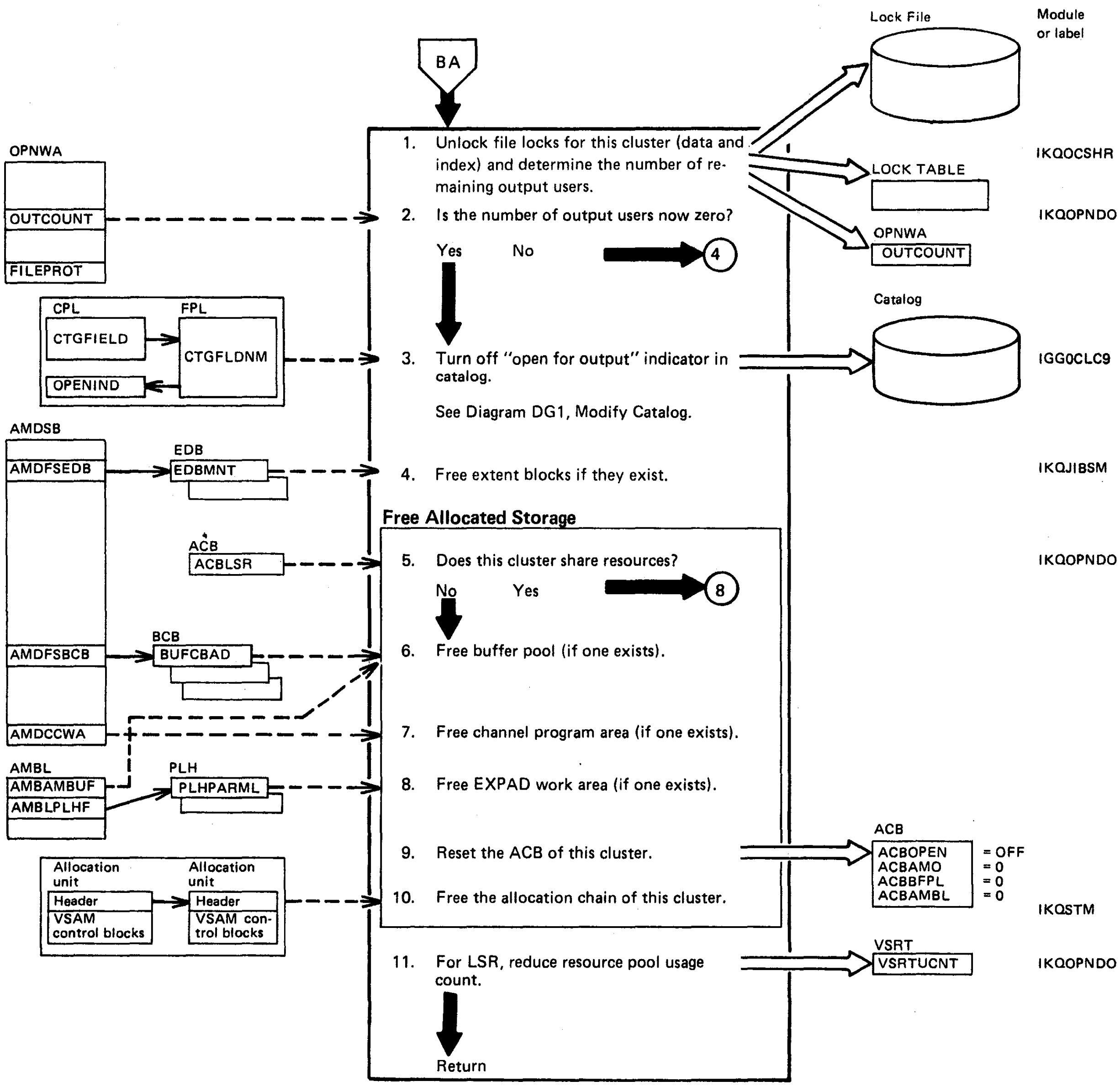


Diagram CA1. ISAM Interface Contents

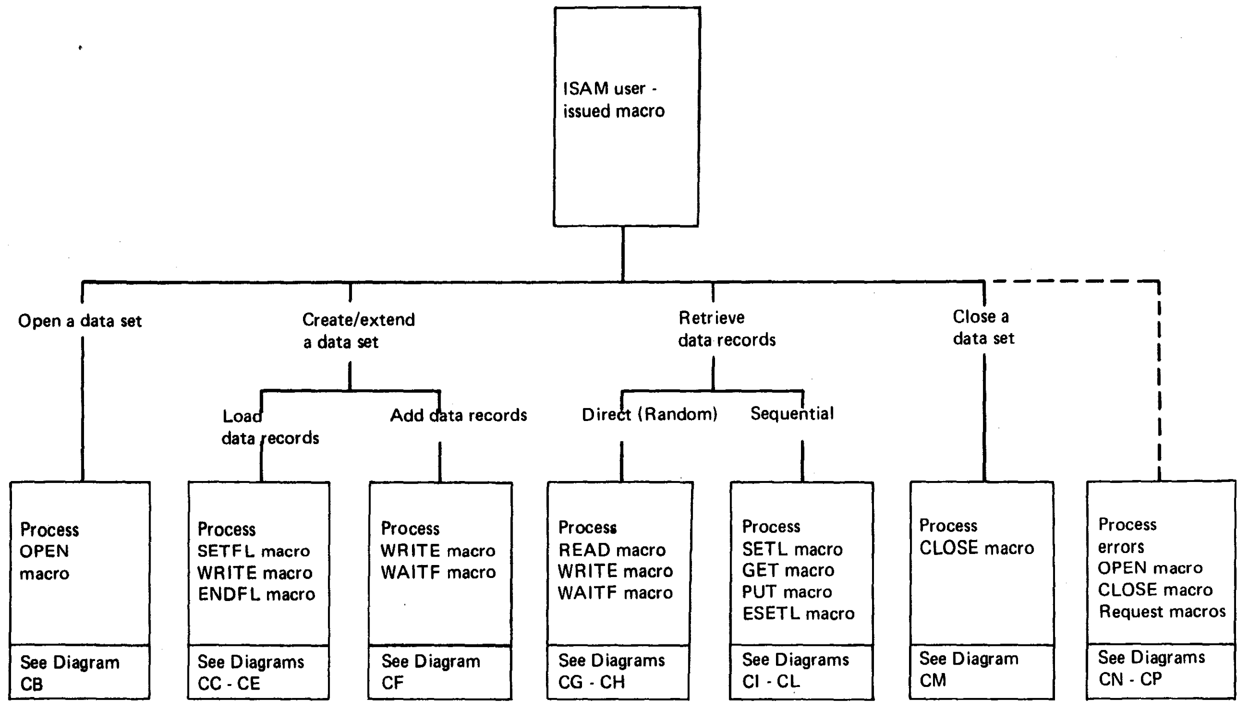
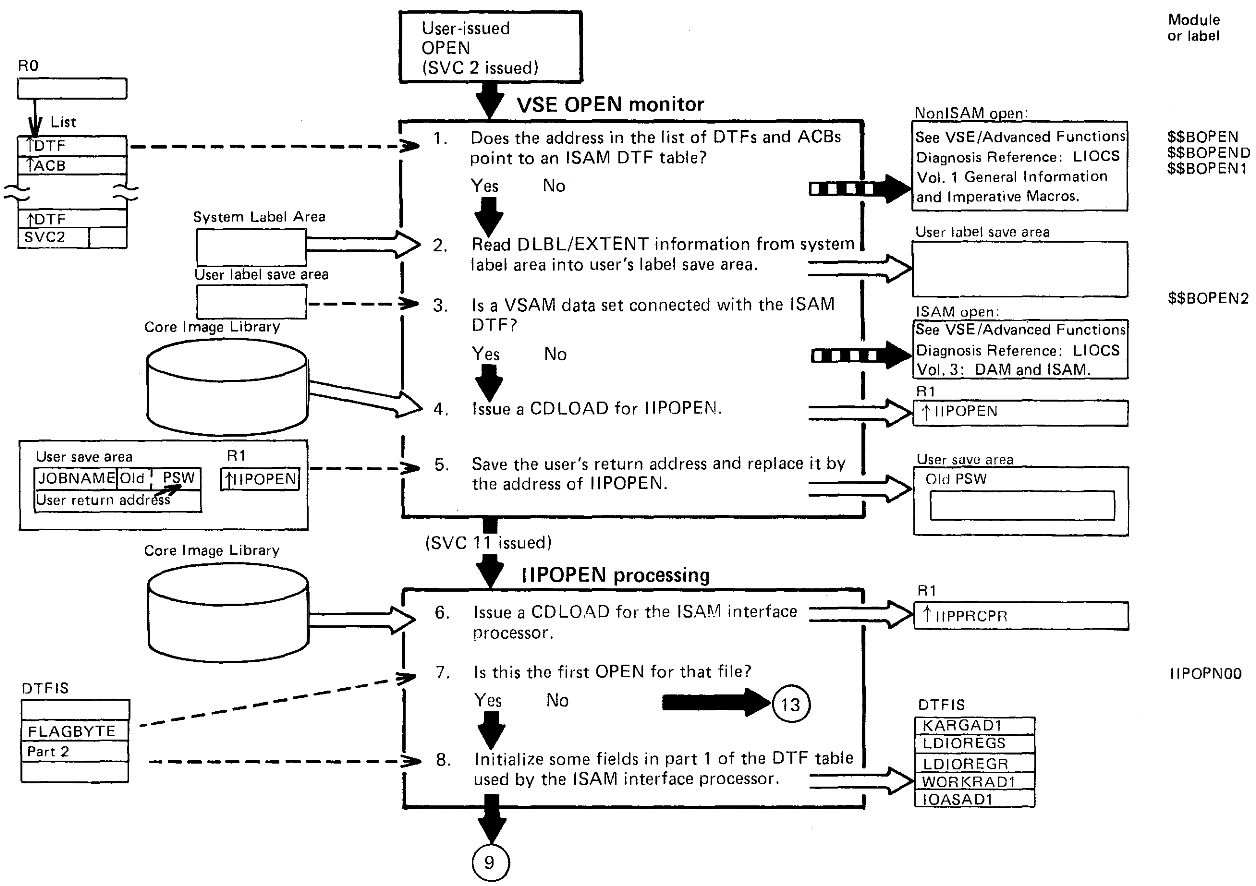


Diagram CB1. OPEN: Connect a User's ISAM Program to a VSAM Data Set



Notes for Diagram CB1

1. The VSE OPEN Monitor examines the DTF -type field (offset 20 of the address passed in the list) of the DTFs or ACBs. If the byte indicates an ISAM file (X'24', X'25', X'26', or X'27'), an SVC 2 is issued and $$BOPEN2 is fetched into the B-transient area. (If the file is not an ISAM file, the regular OPEN continues).

   The list pointed to by register 0 may consist of pointers to DTF tables and/or ACBs. Register 0 is provided by the user program.

3. LABEL TYPE'A' means that an ISAM file is connected with a VSAM data set.

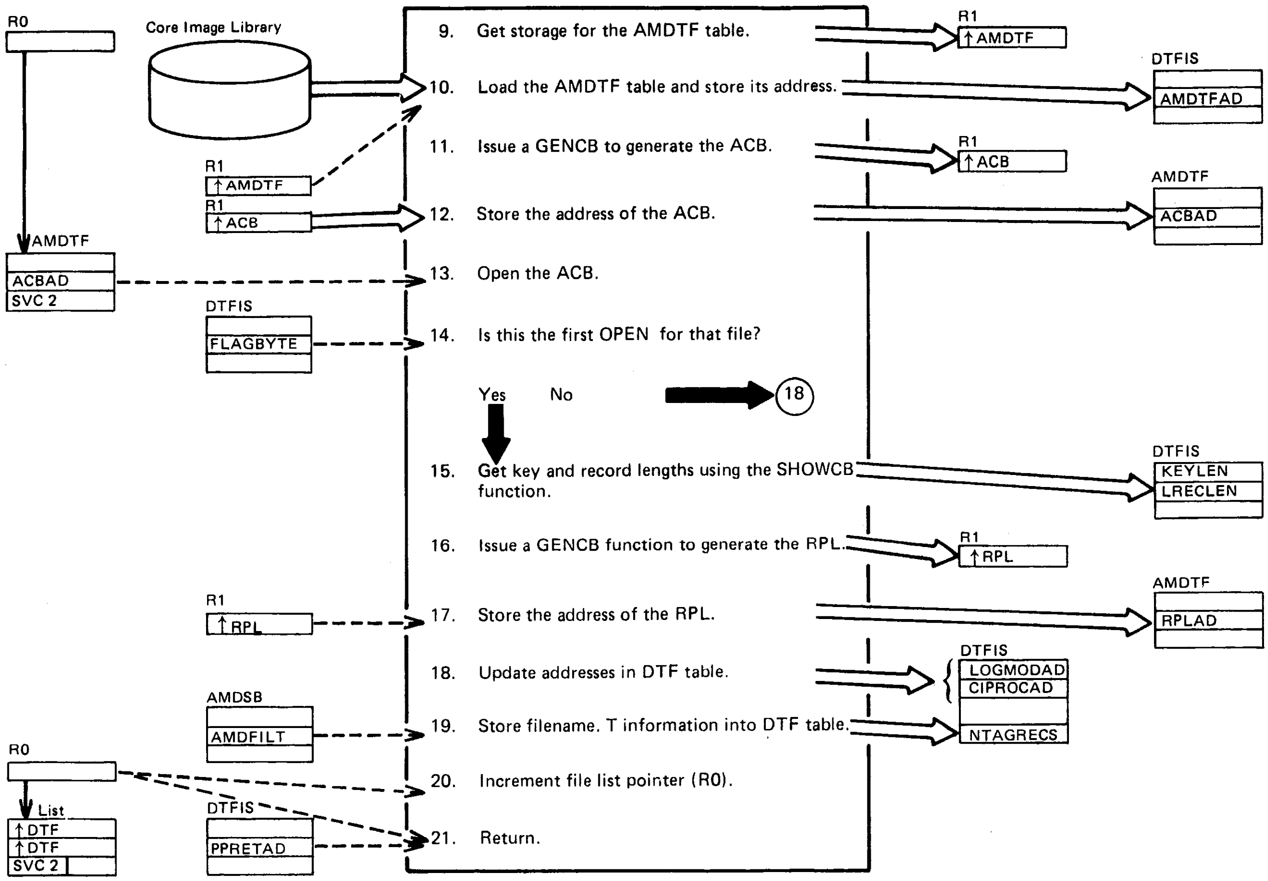
4. IIPOPEN is loaded into the caller's partition, and its address is returned in R1.

6. IIPPRCPR is loaded into the caller's partition, and its address is returned in R1.

7. Bit X'80' in FLAGBYTE is set by IIPOPEN after the first successful OPEN for the VSAM data set.

8. The information needed to initialize the fields in part 1 of the DTF table is derived from part 2 of that table. The only purpose of this transformation is to provide better access to these items for the ISAM Interface Processor.

Diagram CB2. OPEN: Connect a User's ISAM Program to a VSAM Data Set



Notes for Diagram CB2

10. The AMDTF table contains the parameter lists for the GENCB ACB, GENCB RPL, SHOWCB, and MODCB RPL macros, and the ERREXT parameter list used by the ISAM program in case of errors. The AMDTF table is loaded via the LOAD macro.

11. The parameter list in the AMDTF table for the GENCB macro is completed as follows:

    - Copy the filename from the DTF table
    - Specify the MACRF element according to the IOROUT parameter in DTFIS:

IOROUT	KEY	DIR	SEQ	IN	OUT
LOAD	X		X		X
ADD	X	X			X
ADDRTR	X	X	X	X	X
RETRVE	X	X	X	X	X

13. The address of the ACB (pointed to by R0) is followed by a non-zero byte to make sure that control returns to IIPOPEN after opening the ACB.

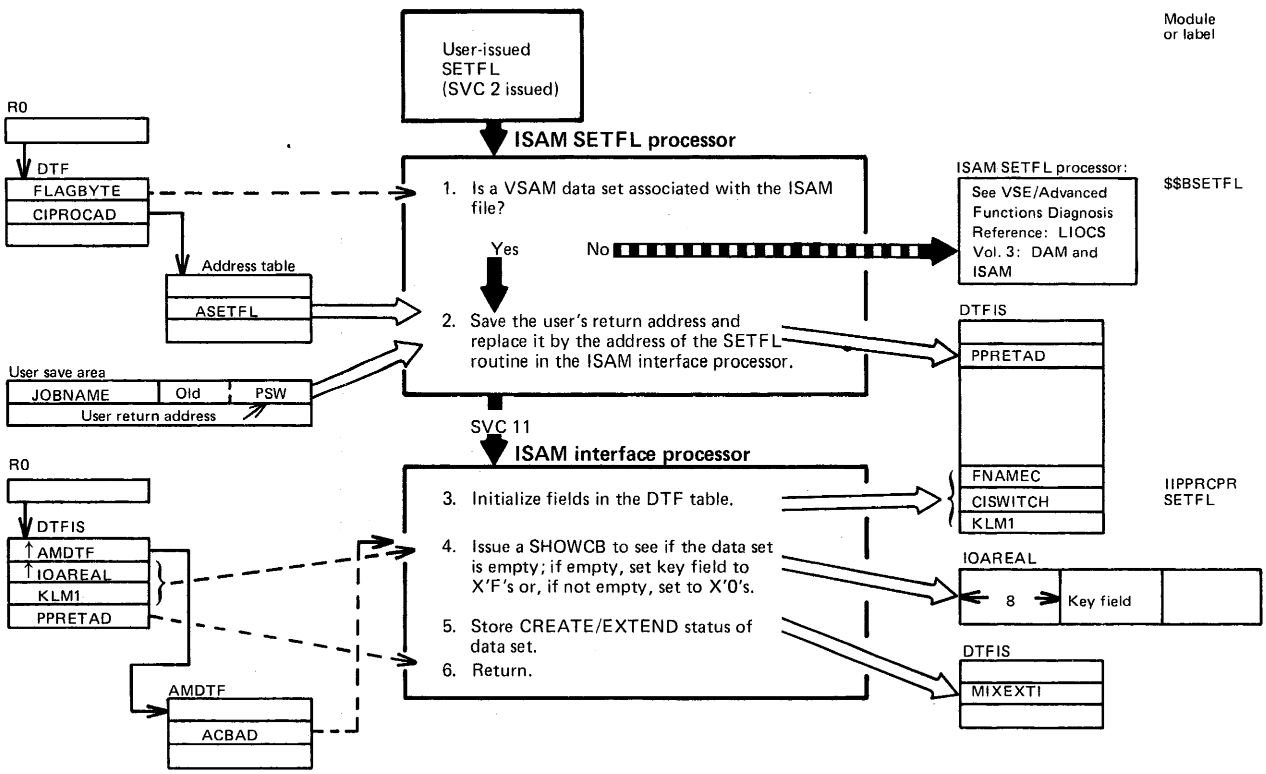
15. Since key length and record length are not generated in the DTFIS macro expansion for retrieve files, this information is extracted from the ACB via a SHOWCB. The field LRECLEN eventually contains the VSAM record length for blocked records and the VSAM record length diminished by the key length for unblocked records.

16. The parameter list for the GENCB macro in the AMDTF table is completed by storing the key length, record length, and ACB address in it.

18. The logic module address in the DTF table (LOGMODAD) is replaced by the address of the LOAD branch vector in IIPPRCPR or the address of the general branch vector, depending on whether the ISAM file is a LOAD file. The beginning address of IIPPRCPR is also stored (CIPROCAD). The LOGMODAD field is used to pass control from the user to data management. The address of the ISAM module is replaced by the branch vector address in IIPPRCPR, and control automatically goes to the ISAM Interface Processor instead of the ISAM logic module. CIPROCAD is referenced when the user issues a SETFL, ENDFL, or SETL macro to pass control from the $$B-phases to the ISAM interface processor (CIPROCAD is a pointer to the address list at the very beginning of IIPPRCPR).

21. If there is no further element in the list, control is returned to the instruction in the user program that follows the SVC 2. Otherwise, control is returned to the SVC 2 instruction.

Diagram CC1. Load Data Records: SETFL



Notes for Diagram CC1

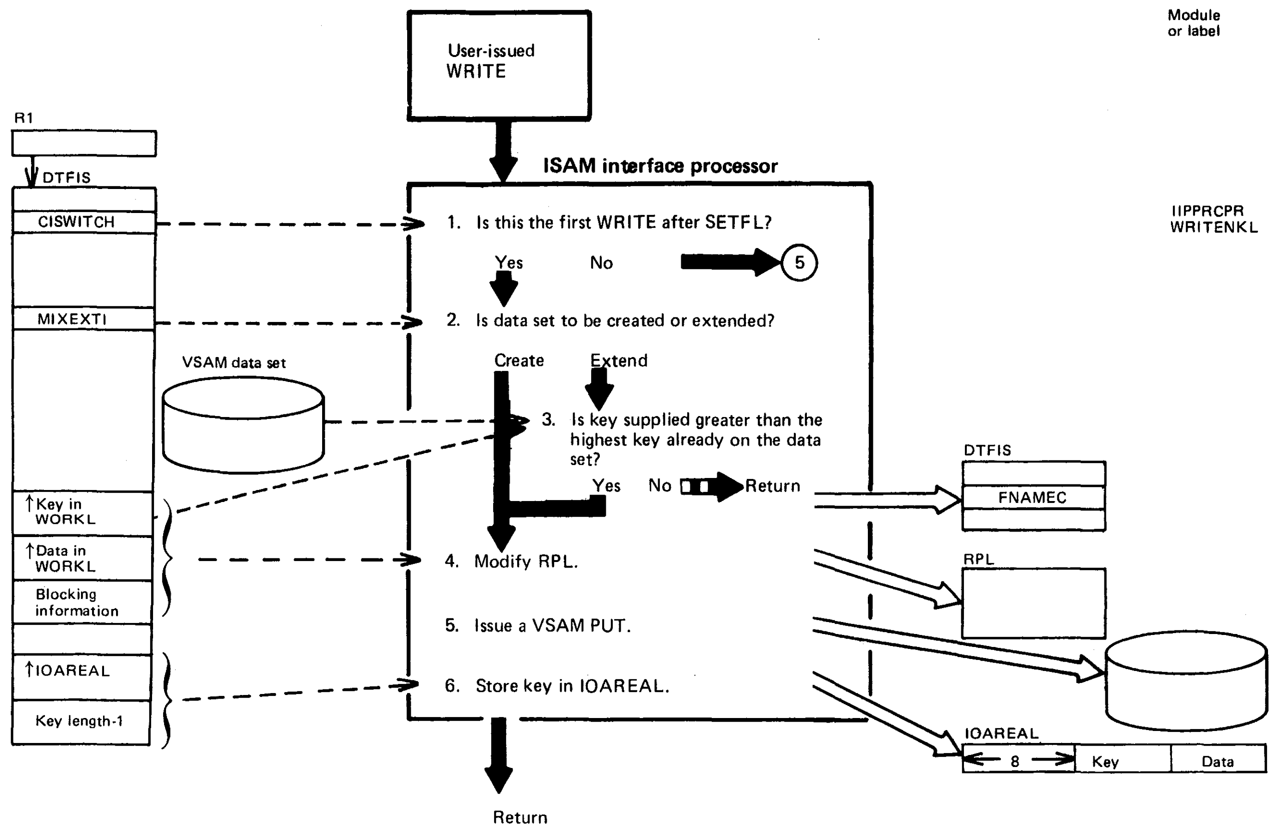
1. Bit 0 in the flag byte (offset 16 in the DTF table) is set by IIPOPEN after a successful OPEN of the data set.

3. Fields in the DTF are set as follows:

   - FNAMEC to X'00'
   - CISWITCH to 'LOAD' and 'FIRST WRITE'
   - KLM1 is initialized with 'key length – 1'. The key length is derived from the field KEYLEN which was initialized by IIPOPEN.

4. The key information in this form in IOAREAL is used after SETFL by various problem programs, for example, PL/I.

Diagram CD1. Load Data Records: WRITE



Notes for Diagram CD1

1. Control is passed from the user to the ISAM interface processor via the LOGMODAD field in the DTF table, which, after OPEN, contains the address of the LOAD branch vector in the ISAM interface processor.

1-2. The two switches referenced in these steps are set by the ISAM interface processor routine SETFL.

3. To perform the sequence check, a dummy GET is issued that is prepared by a MODCB with the OPTCD code: KEY, DIR, SYN, NUP, KGE, FKS, LOC.

   If the return code following the GET operation is zero, this indicates that a sequence error has occurred, that is, that a record with a key equal to or greater than that in WORKL is already on the data set.

   If the return code following the GET operation is not zero, control is passed to the general error routine ERGN in IIPPRCPR. This routine analyzes the error. If the error is "no record found" and the "first write" bit in CISWITCH is on, the error is ignored and normal processing continues.

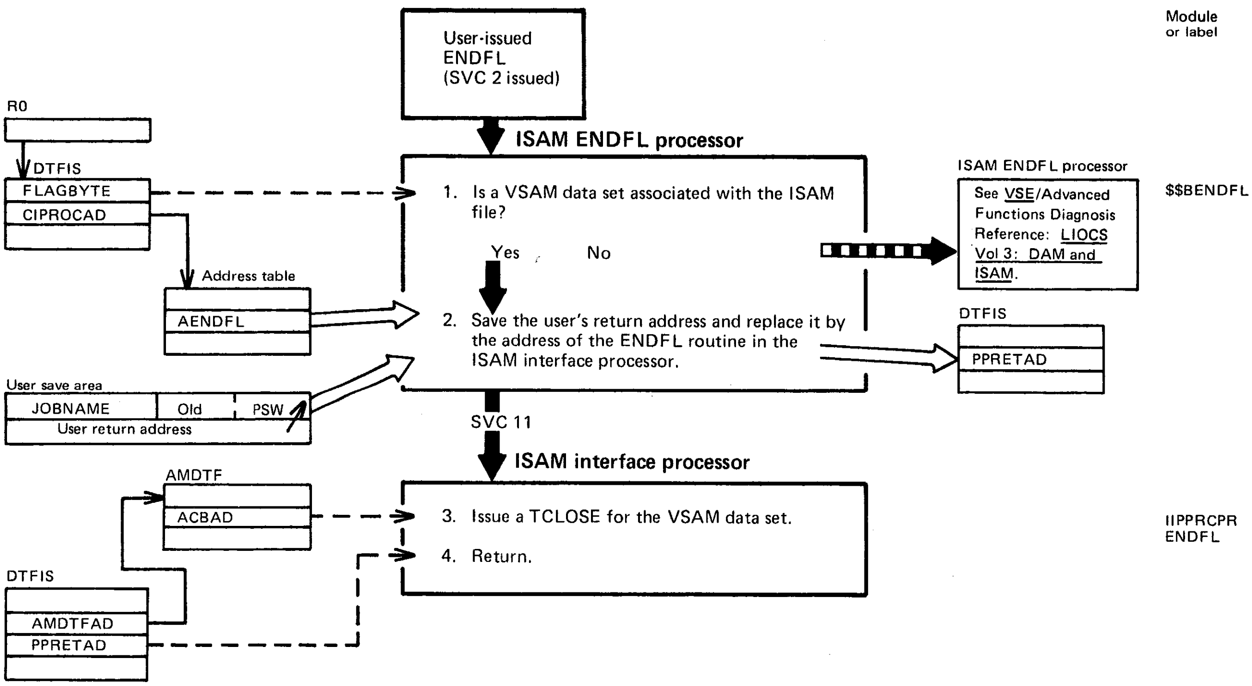
   Note: The sequence check for subsequent PUTs is done by VSAM.

4. A MODCB is issued with the following OPTCD code: KEY, SEQ, SYN, NUP, MVE.

   The AREA element in the MODCB parameter list is initialized. It specifies the address of WORKL. Since the address of WORKL is not supplied in the DTF macro expansion for LOAD files, it is derived from the address of the key in WORKL if RECFORM=FIXUNB and the address of the data in WORKL if RECFORM=FIXBLK.

6. The key is stored in IOAREAL mainly for the benefit of problem programs (for example, PL/I) which may check this field.

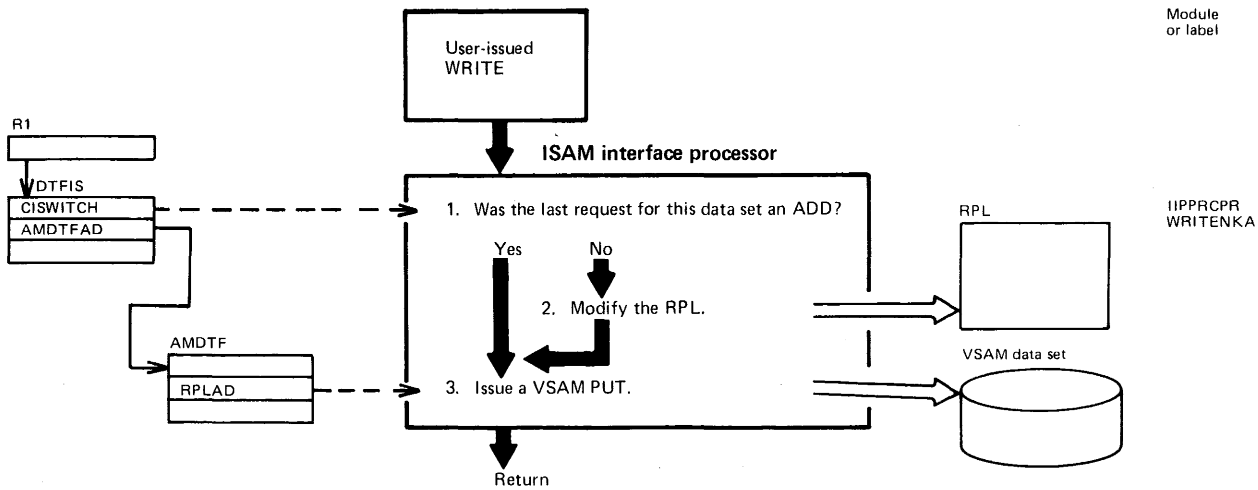
Diagram CE1. Load Data Records: ENDFL

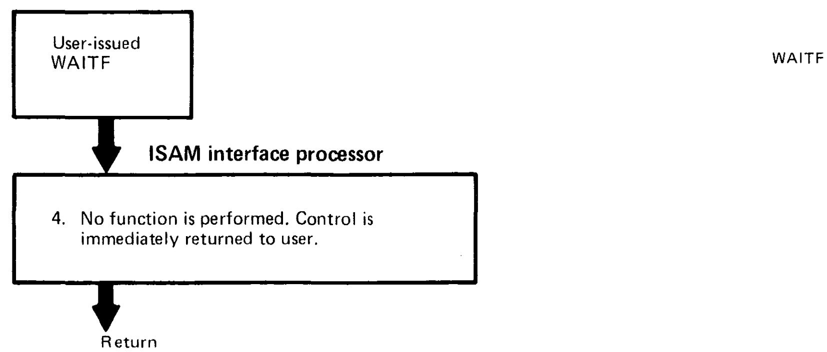


Notes for Diagram CE1

1. Bit 0 in the flag byte (offset 16 in the DTF table) is set by IIPOPEN after a successful OPEN of the VSAM data set.

Diagram CF1. Add Data Records





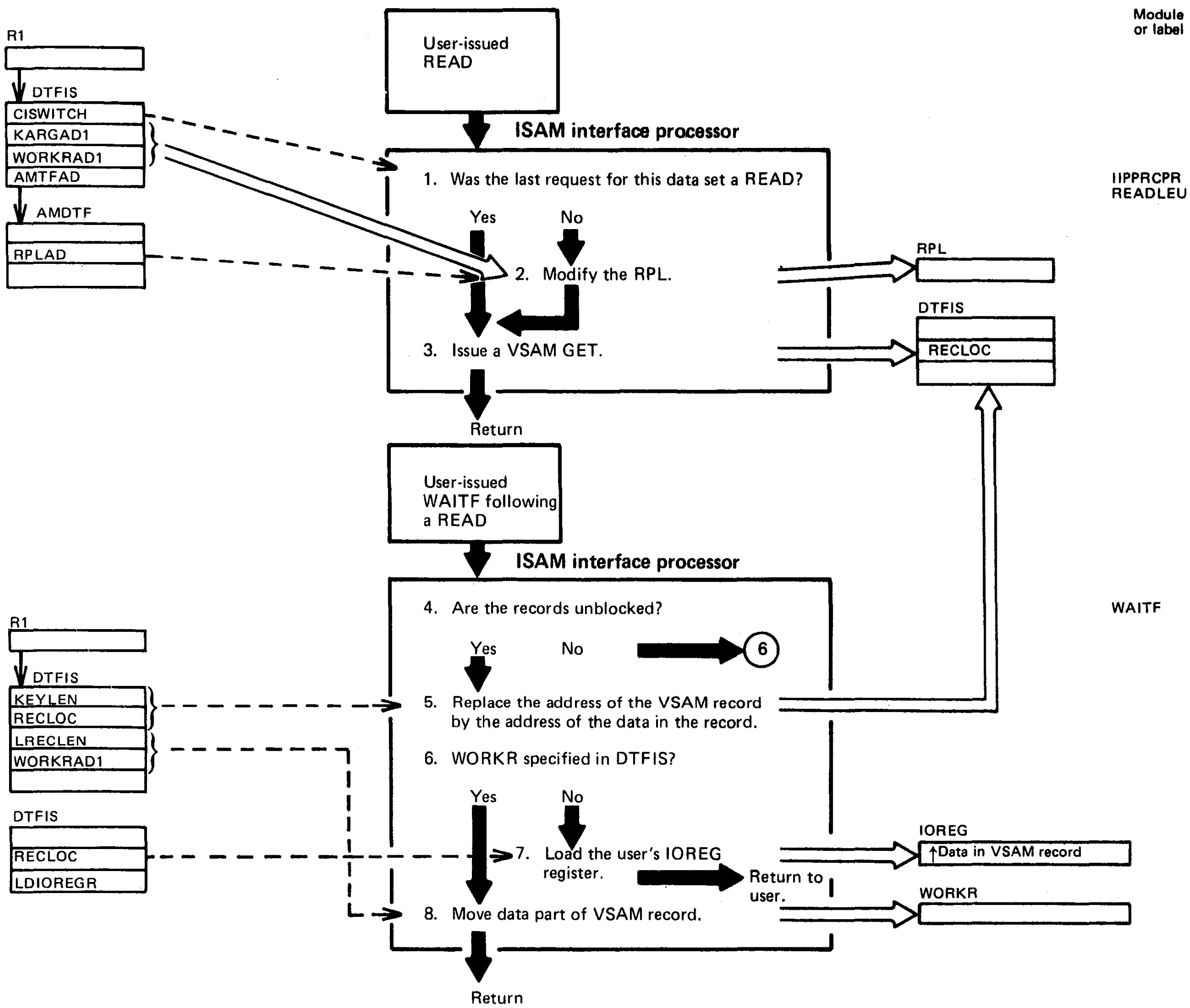
Notes for Diagram CF1

1. Control is passed from the user to the ISAM interface processor via the LOGMODAD field in the DTF table, which, after OPEN, contains the address of the general branch vector in the ISAM interface processor.

2. A MODCB is issued with the following OPTCD code: KEY, DIR, SYN, NUP, MVE.

4. See note 1.

Diagram CG1. Direct (Random) Processing of Data Records: READ



Notes for Diagram CG1

1. Control is passed from the user to the ISAM interface processor via the LOGMODAD field in the DTF table, which, after OPEN, contains the address of the general branch vector in the ISAM interface processor.

2. The records are processed by VSAM in MOVE mode when WORKR= name and RECFORM=FIXBLK are specified in the DTFIS. The OPTCD parameters for the RPL modifications are KEY, DIR, SYN, UPD, KEQ, FKS, and MVE.

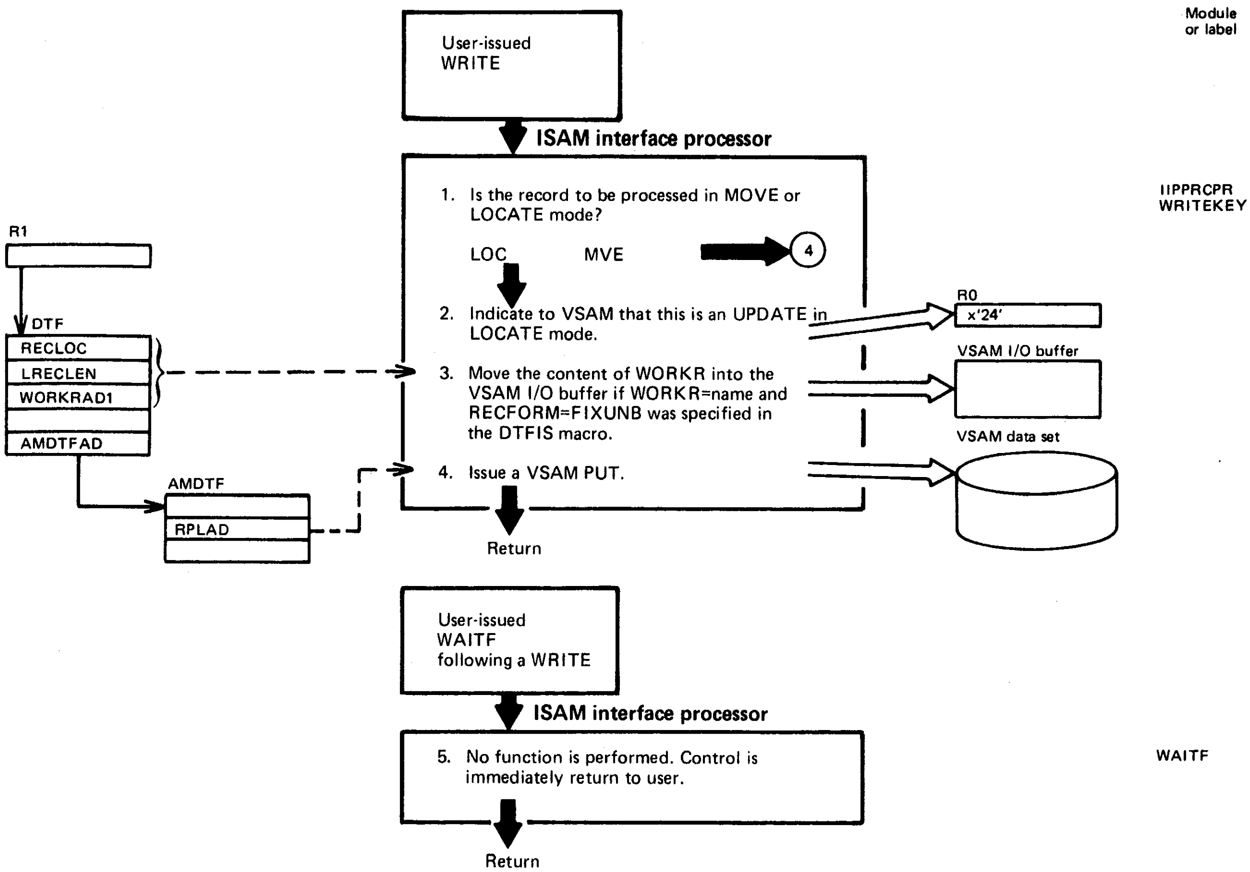
   The records are processed by VSAM in LOCATE mode in all other cases, and the OPTCD parameters are KEY, DIR, SYN, UPD, KEQ, FKS, and LOC.

   The WORKRAD1 field in the DTF table (which contains a pointer to WORKR) used for MOVE mode is initialized by IIPOPEN, as well as KARGAD1 (which contains a pointer to the key).

3. If LOCATE mode is specified, VSAM returns the address of the VSAM record in the DTF after the GET operation.

4. See note 1.

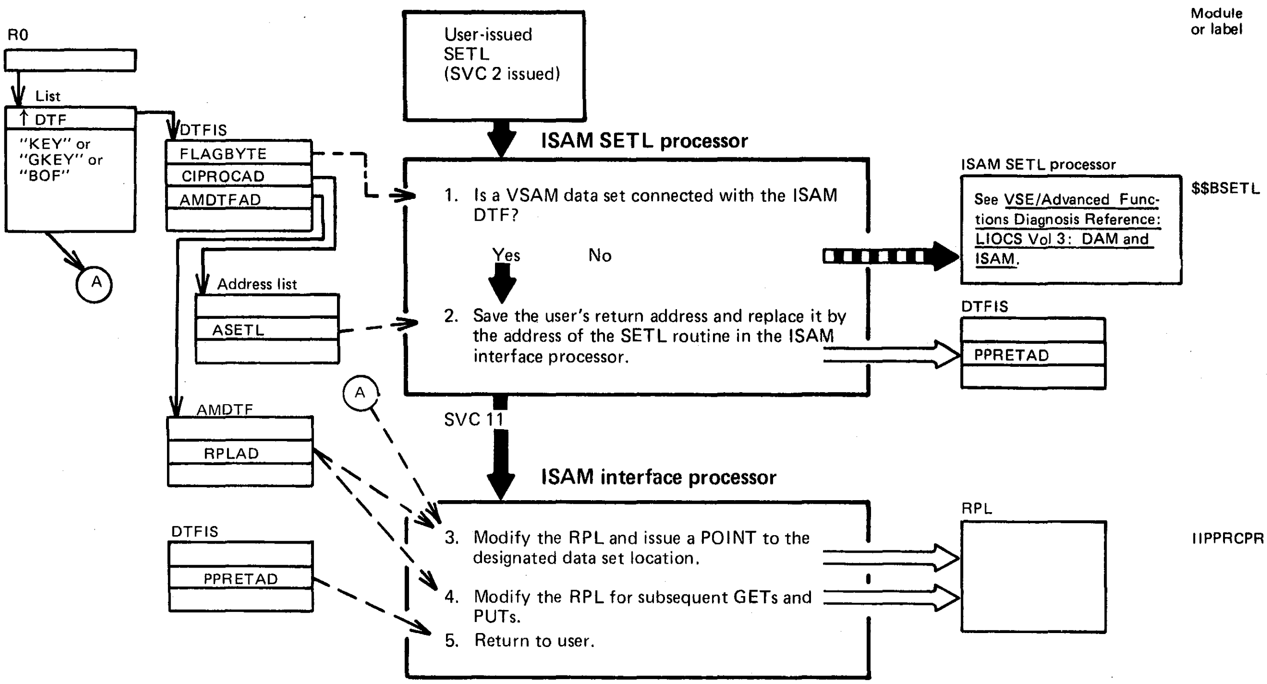
Diagram CH1. Direct (Random) Processing of Data Records: WRITE



Notes for Diagram CH1

1. See notes 1 and 2 for Diagram CG1.
2. Since VSAM does not allow update of records in LOCATE mode on the one hand and the ISAM interface processor, on the other hand, needs update in LOCATE mode, a special PUT is issued which, by means of the value passed in register 0, indicates to VSAM that the request comes from the ISAM interface program.
5. See note 1 for Diagram CG1.

Diagram CI1. Sequential Processing of Data Records: SETL



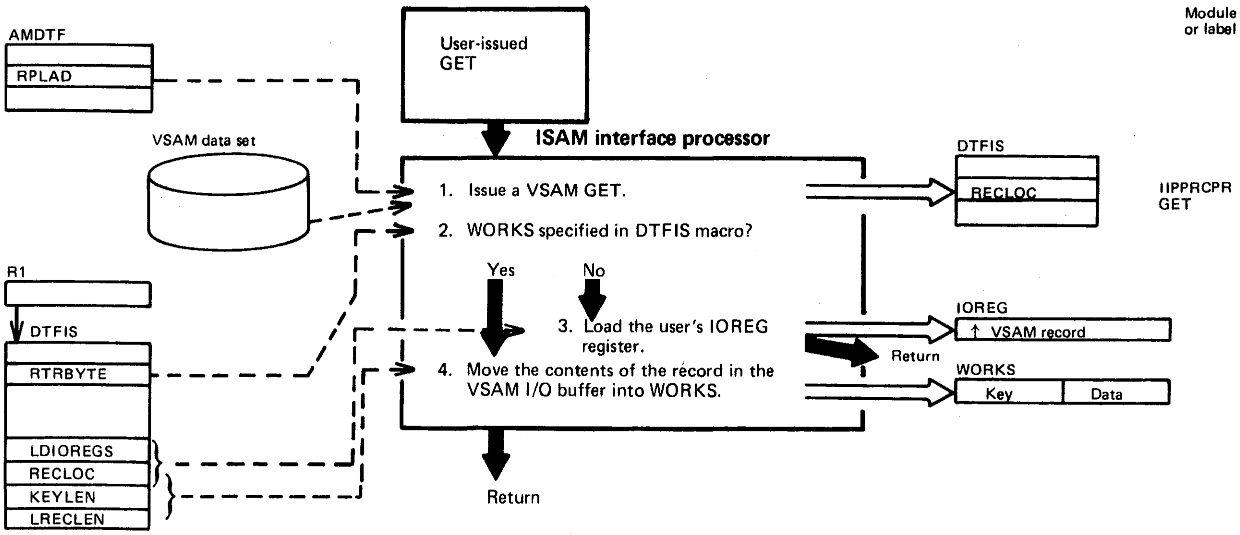
Notes for Diagram CI1

1. Bit 0 in the flag byte (offset 16 in the DTF table) is set by IIPOPEN after a successful OPEN of the data set.

3-5. A MODCB is issued with the following OPTCD parameters to prepare for subsequent GETs and PUTs: KEY, SEQ, SYN, UPD, and LOC.

Thus, all records are to be processed in LOCATE mode.

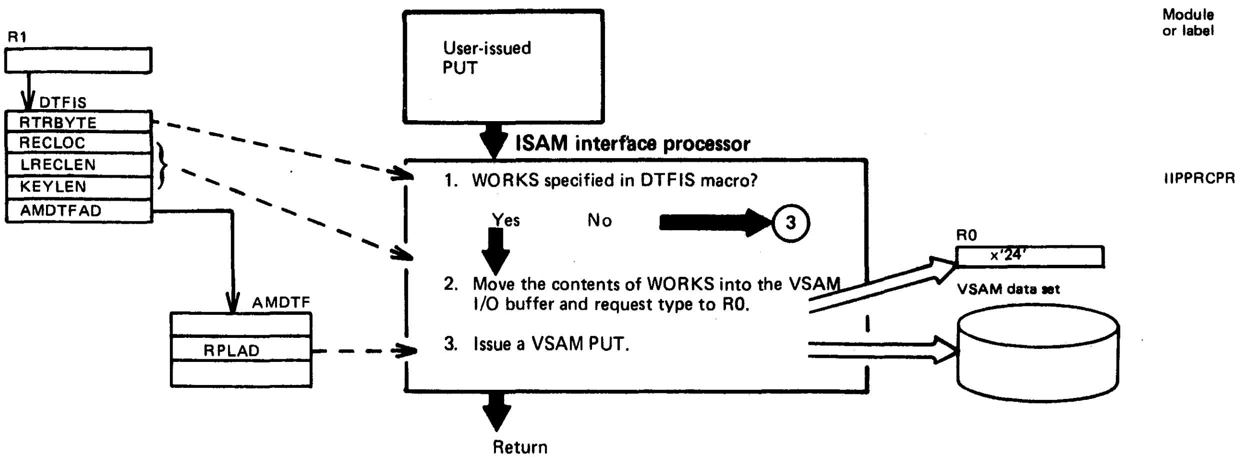
Diagram CJ1. Sequential Processing of Data Records: GET



Notes for Diagram CJ1

1. Control is passed from the user to the ISAM interface processor via the LOGMODAD field in the DTF table, which, after OPEN, contains the address of the general branch vector in the ISAM interface processor.

Diagram CK1. Sequential Processing of Data Records: PUT

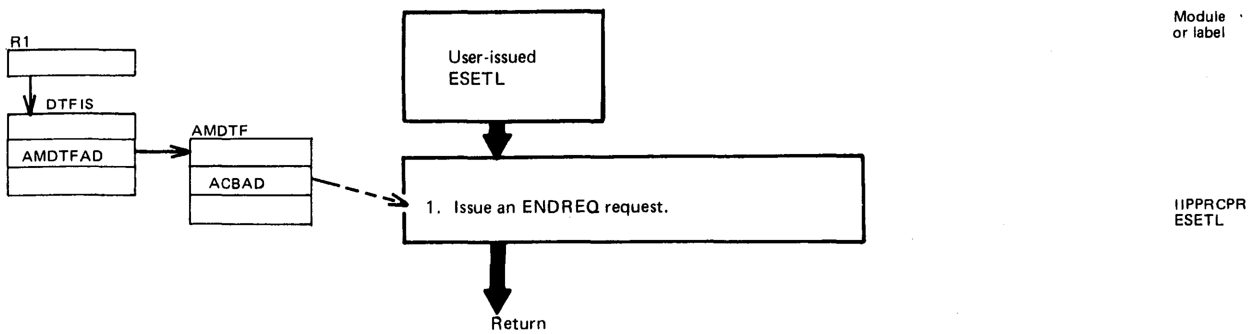


Notes for Diagram CK1

1. Control is passed from the user to the ISAM interface processor via the LOGMODAD field in the DTF table, which, after OPEN, contains the address of the general branch vector in the ISAM interface processor.

1-3. Since VSAM does not allow update of records in LOCATE mode on one hand and the ISAM interface processor, on the other hand, needs update in LOCATE mode, a special PUT is issued which, by means of the value passed in R0, indicates to VSAM that the request comes from the ISAM interface program.

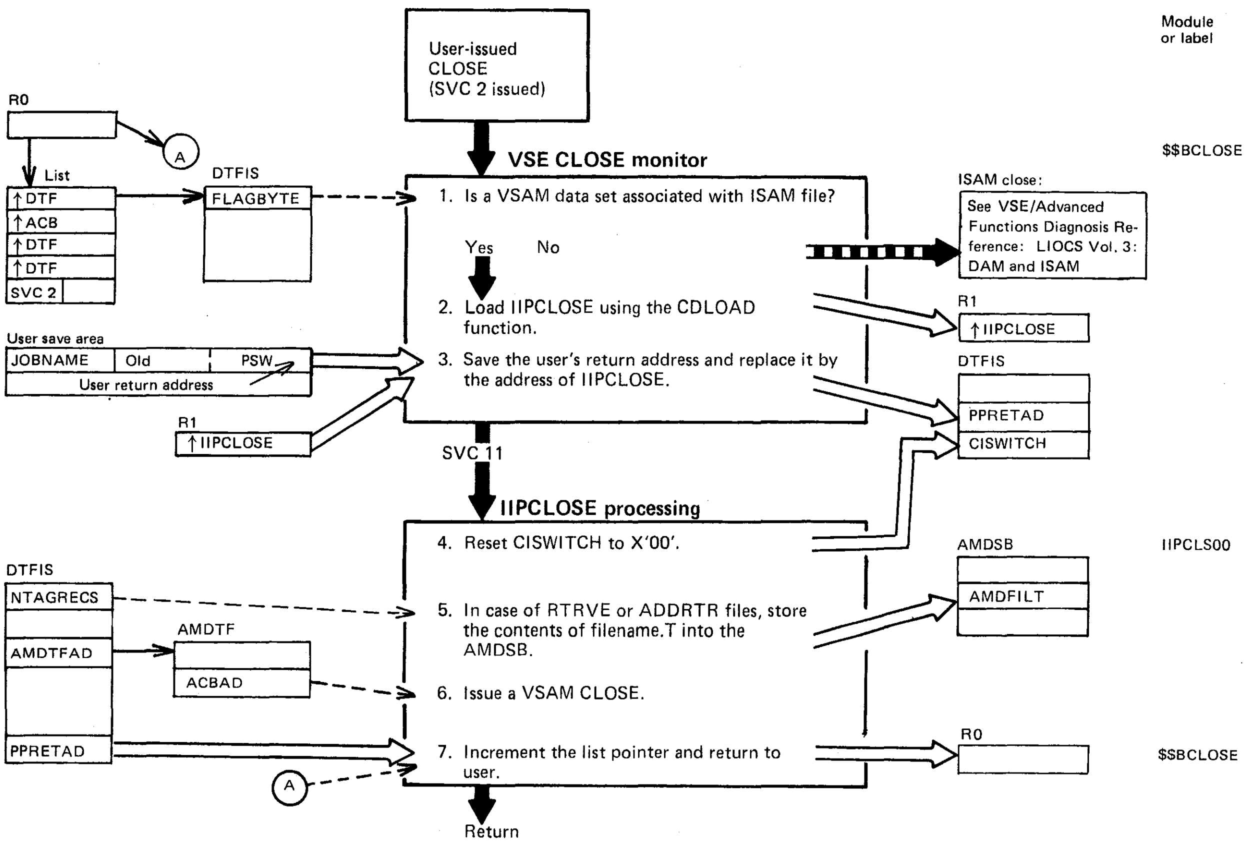
Diagram CL1. Sequential Processing of Data Records: ESETL



Notes for Diagram CL1

1. Control is passed from the user to the ISAM interface processor via the LOGMODAD field in the DTF table, which, after OPEN, contains the address of the general branch vector in the ISAM interface processor.

Diagram CM1. CLOSE: Disconnect a User's ISAM Program from a VSAM Data Set

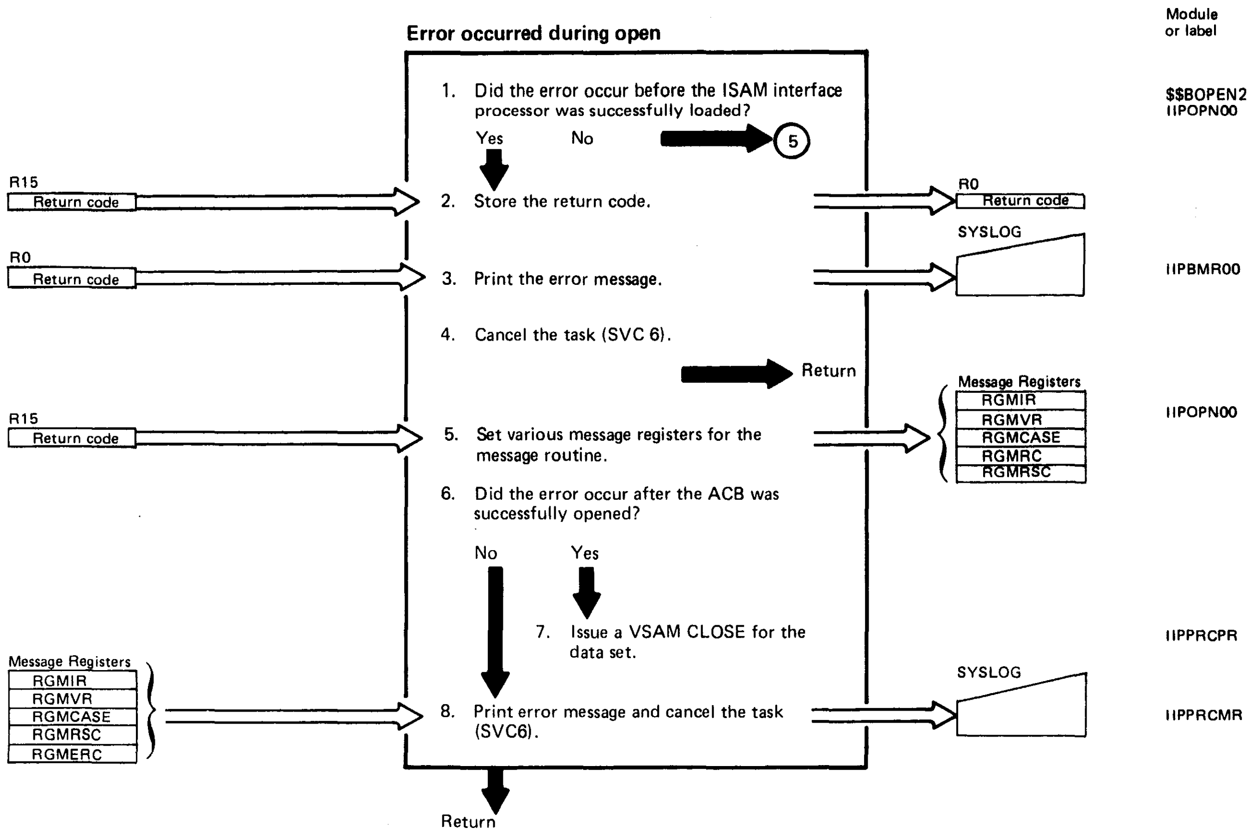


Notes for Diagram CM1

1. Bit 0 in the flag byte (offset 16 in the DTF table) is set by IIPOPEN after a successful OPEN of the data set.

7. When no elements are left in the list, control is returned to the instruction immediately following the SVC 2 in the user program. Otherwise, control is returned to the SVC 2.

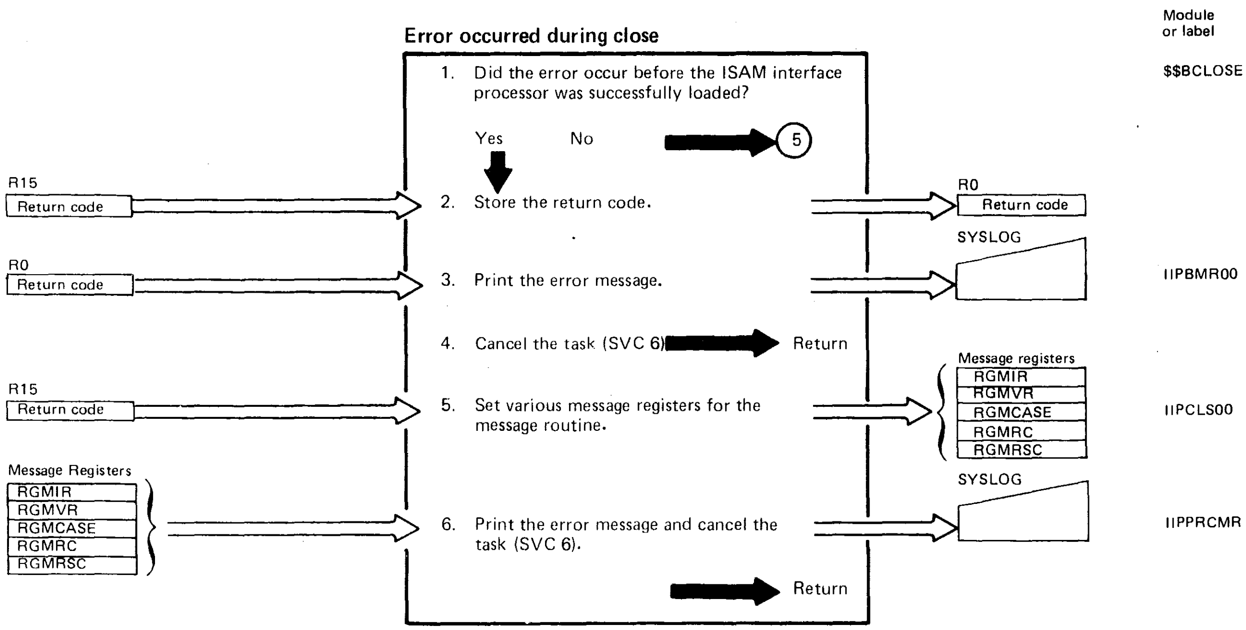
Diagram CN1. ISAM Interface Error Processing: OPEN



Notes for Diagram CN1

1. If the ISAM interface processor is not loaded, a separate message routine must be used.
2. Return code is moved to R0, in preparation for the message routine.
3. Return code loaded into R0 during step 2 determines message.
5. Registers are set up for IIPPRCMR.
7. Data set must be closed.
8. Uses registers set up in step 5.

Diagram CO1. ISAM Interface Error Processing: CLOSE



Notes for Diagram CO1

1. If the ISAM interface processor is not loaded, a separate message routine must be used.
2. Return code is moved to R0, in preparation for the message routine.
3. Return code loaded into R0 during step 2 determines message.
5. Registers are set up for IIPPCRMR
6. Uses registers set up in step 5.

Diagram CP1. ISAM Interface Error Processing: Request Macros

Module or label

R15
Return code

Message Registers
RGMIR
RGMVR
RGMCASE
RGMRC
RGMRSC

Error occurred during functions other than open or close

1. Indicate ISAM and failing VSAM request. — IIPPRCPR
2. Did the error occur after a MODCB or SHOWCB, or was the return code from an action macro (R15=4)?
   Yes   No → 6
3. Issue VSAM CLOSE for the data set.
4. Print the error message. → SYSLOG
5. Cancel the task (SVC 6). — IIPPRCMR
   → Return — IIPPRCPR
6. Issue a SHOWCB for the feedback information in order to analyze the error.
7. SHOWCB failed?
   No   Yes
   8. Print the error message and cancel the task (SVC 6). → SYSLOG — IIPPRCMR IIPPRCPR
      → Return
9. Was it a physical or logical error?
   Logical → 10   Physical → 14

Diagram CP2. ISAM Interface Error Processing: Request Macros

Module or label

Error occurred during functions other than open or close (continued)

10. Was the error one of the following?
    - End of data set
    - Duplicate record
    - Record out of sequence
    - No DASD space available
    - No record found

    Yes / No

    No:

11. Issue a VSAM CLOSE for the data set. — IIPPRCMR
12. Print error message and cancel task (SVC 6). — IIPPRCPR

    Return

    Yes:

13. Store information into filename.C.
14. Did a READ or WRITE error occur on a data, index, or sequence set?

    Yes / No

    No:

15. Issue a VSAM CLOSE for the data set. — IIPPRCMR
16. Print error message and cancel task (SVC 6). — IIPPRCPR

    Return

    Yes:

17. Indicate irrecoverable I/O error in filename.C.

(18)

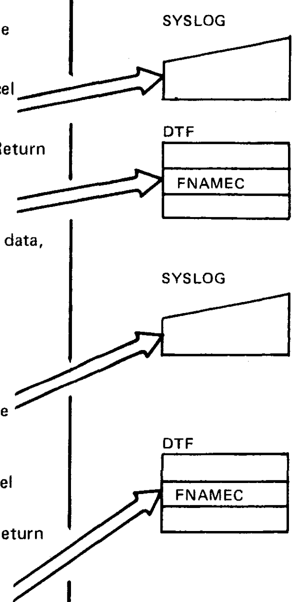


Diagram CP3. ISAM Interface Error Processing: Request Macros

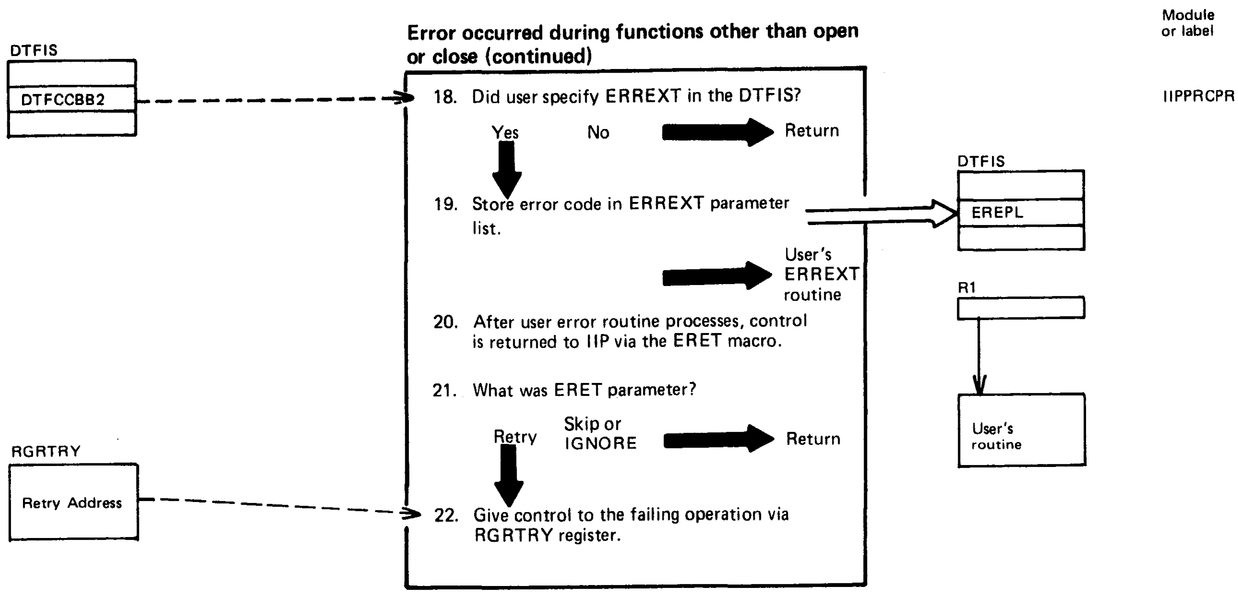


Diagram DA1. Catalog Management Contents

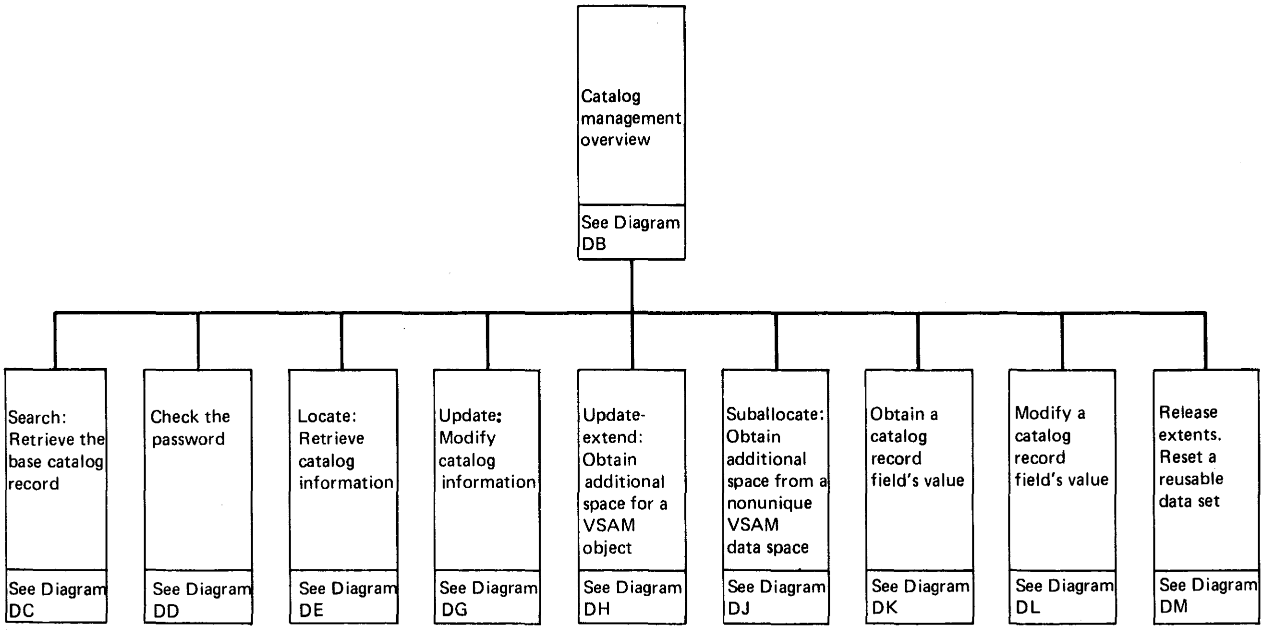
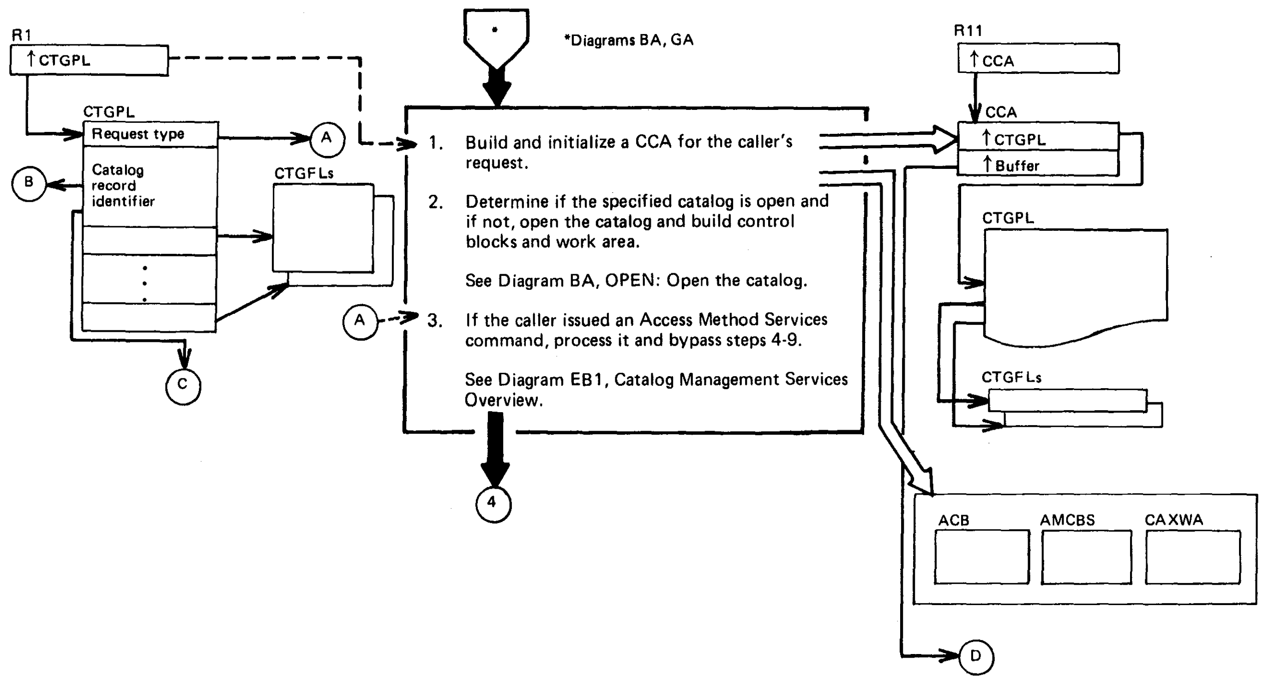


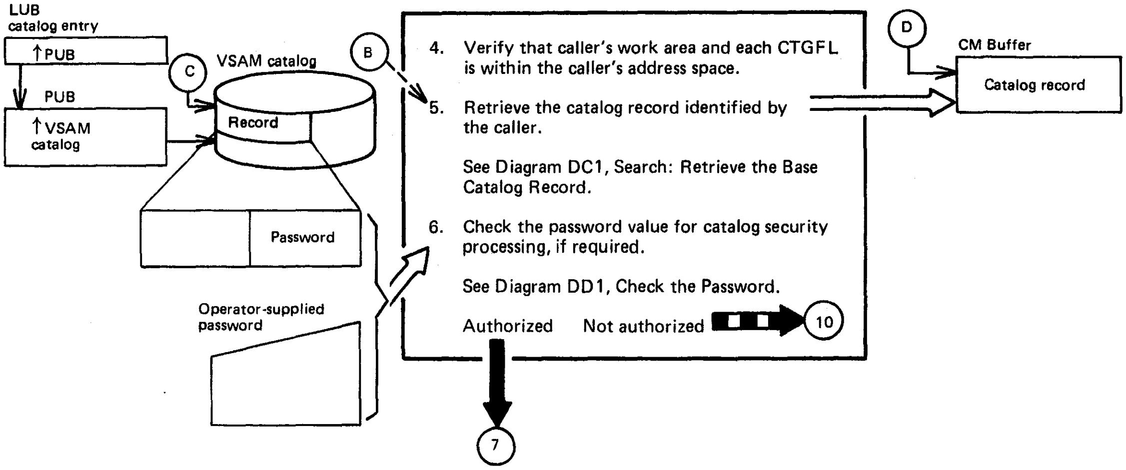
Diagram DB1. Catalog Management Overview



Notes for Diagram DB1

Description	Module	Procedure
VSAM catalog management (CM) is called with a CATLG macro instruction by VSAM Open, Close, and end-of-volume routines. In addition, a user's program can process VSAM catalog records by issuing an Access Method Services request. Access Method Services also issues the CATLG macro, which translates the request into an SVC 65 and a catalog parameter list (CTGPL).		
The CATLG macro instruction checks to see if the called VSAM CM module is in storage. If it isn't, the module is loaded from the Core Image Library. Register 1 contains the address of the caller's CTGPL. The CTGPL identifies which catalog record to process and what process to perform.		
1. The catalog control area (CCA) contains data about catalog records retrieved to process the request. The CCA also contains a register save area that shows the flow of control between CM routines used to process the request.	IGG0CLC9	BLDCCA
Each time a CM routine is called by another CM routine, the contents of registers 12, 13, and 14 are put in the CCA's register save area. See the Data Areas section of this manual for details about the CCA and CTGPL, and the Diagnostic Aids section for information about the CCA register save area.		
The catalog driver is then called to determine what request was issued and which routine processes the request.	IGG0CLAB	IGGPACDV
2. The master catalog's AMCBS is built, if it has not already been built. A check is then made to see if the master catalog is open. If it is not open, IGG0CLAD calls $$BOPEN to open the master catalog and then builds the master catalog's ACB and CAXWA. Control is returned to IGG0CLAC and then to IGG0CLAB.	IGG0CLAC IGG0CLAD $$BOPEN IGG0CLAD	IGGPMC0 IGGPMC02 IGGPMC02
3. An Access Method Services command is translated into a catalog management services (CMS) request to define (create), alter, delete, or list catalog records. IGG0CLAT then returns to IGG0CLAB.	IGG0CLAT	IGGPCDVR

Diagram DB2. Catalog Management Overview

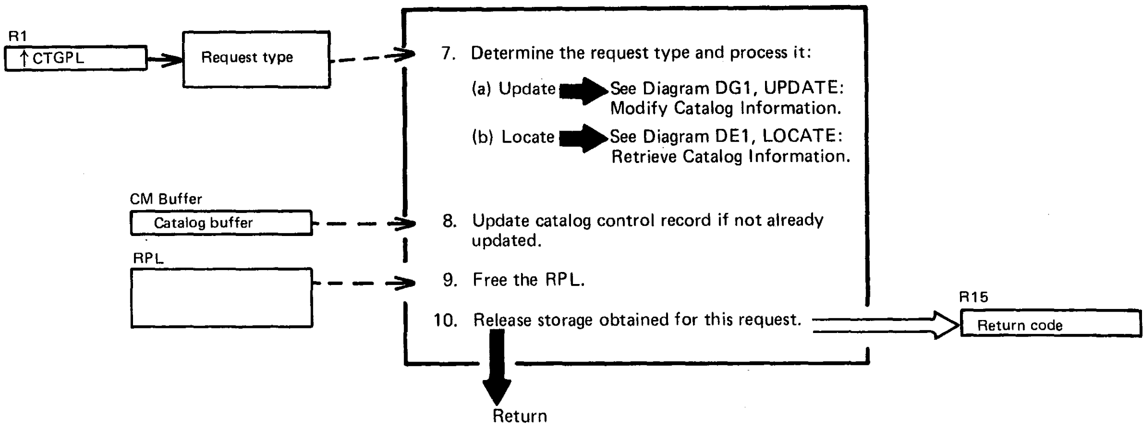


Notes for Diagram DB2

	Description	Module	Procedure
4.	The caller's work area and each CTGPL are checked to ensure that they are within the caller's address space.	IGG0CLAY	IGGPSCNC
	The field-name value in the field parameter list (CTGFL) is used to obtain dictionary data that defines the field's characteristics and location within the record. Control is returned to IGG0CLAB.	IGG0CLAY	IGGPSCNC
5.	The catalog record is identified by the caller's DSNAME value, volume serial number, or control interval number. If the CTGPL's catalog identifier addresses the record's control interval number, the catalog record can be retrieved without a search of the catalog's index. Control returns to the calling function.	IGG0CLFH	IGGPSCAT
6.	The caller's request type determines the password value that, when supplied by the operator or input stream, allows the VSAM CM routines to complete the caller's request.	IGG0CLBM	IGGPCKAU

Description	Module	Procedure
The catalog cluster record is retrieved for catalog security processing. The password group occurrence data must also be retrieved, if it exists. Control is then returned to the catalog calling function.	IGG0CLEG IGG0CLAZ	IGGPGET IGGPEXT

Diagram DB3. Catalog Management Overview

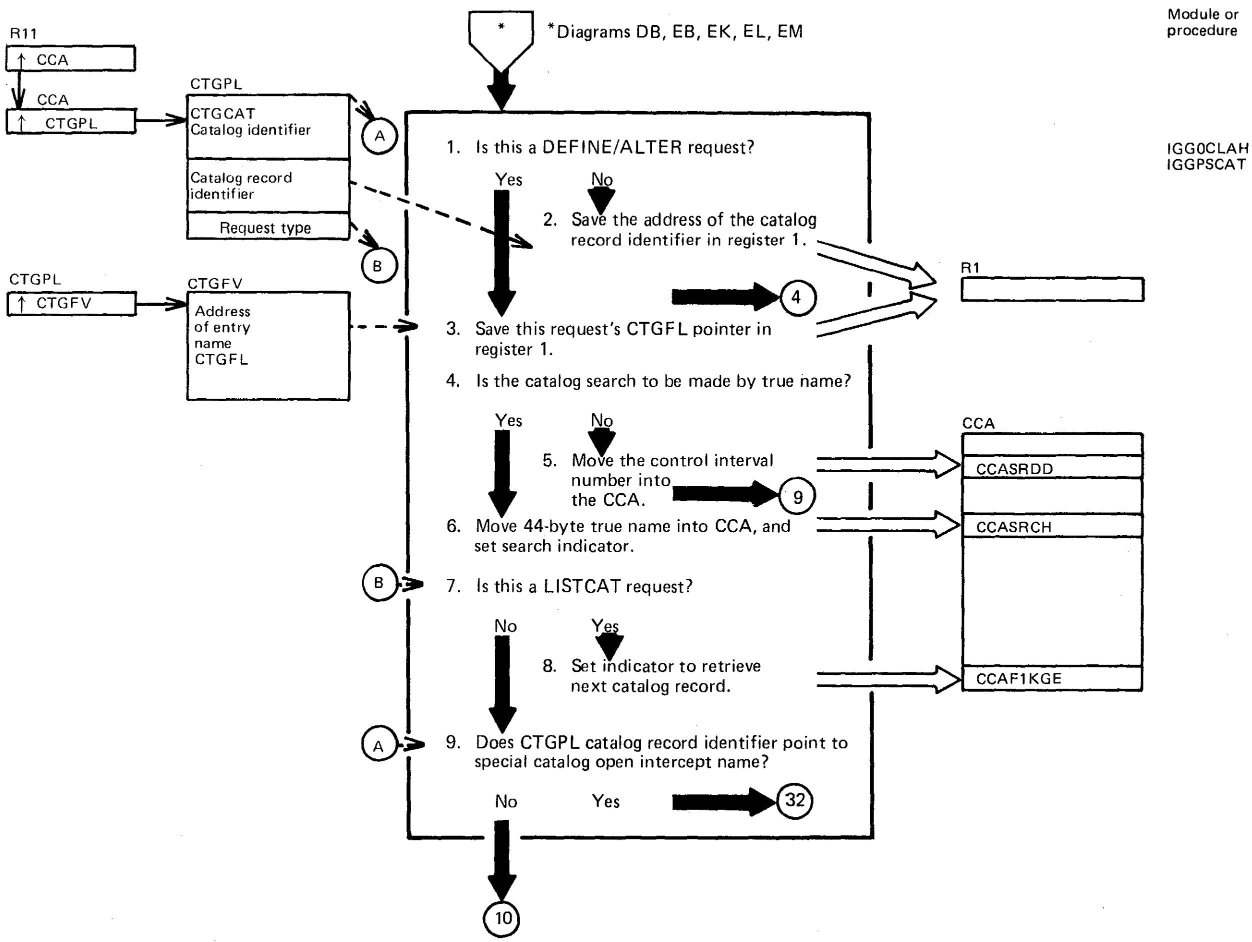


Notes for Diagram DB3

Description	Module	Procedure
7(a) An Update request modifies information in a catalog record. An Update request can also obtain direct-access space for the data set or index identified by the DSNAME value. Control is then returned to the catalog calling functions.	IGG0CLAV IGG0CLBB	IGGPUPD IGGPUPDE
(b) A Locate request retrieves information from the catalog record. IGG0CLBA is called to determine those occurrences from which field data should be retrieved. Control is then returned to the catalog calling functions.	IGG0CLAZ IGG0CLBA	IGGPLOC IGGPTSTS IGGPGVAL IGGPGREC
8. When the VSAM catalog driver (IGG0CLAB) returns to the catalog first load module, IGGPRCU finds out if the CCR has been updated in storage. If so, IGGPCCCR is called to update the CCR and turns to caller.	IGG0CLC9 IGG0CLEG	IGGPRCU IGGPCCCR

Description	Module	Procedure
9. IGGPRPLF frees the RPL for other requests and returns to the catalog first load module.	IGG0CLAB	IGGPRPLF
All storage obtained for work areas and control blocks is freed and returned to the system. A return code is set in register 15, control returns to the caller.	IGG0CLC9	IGGPRCU

Diagram DC1. Search: Retrieve the Base Catalog Record



Notes for Diagram DC1

The CTGPL's catalog identifier field, set by the caller, can contain the address of a catalog's ACB, the address of a catalog's DSNAME, or 0.

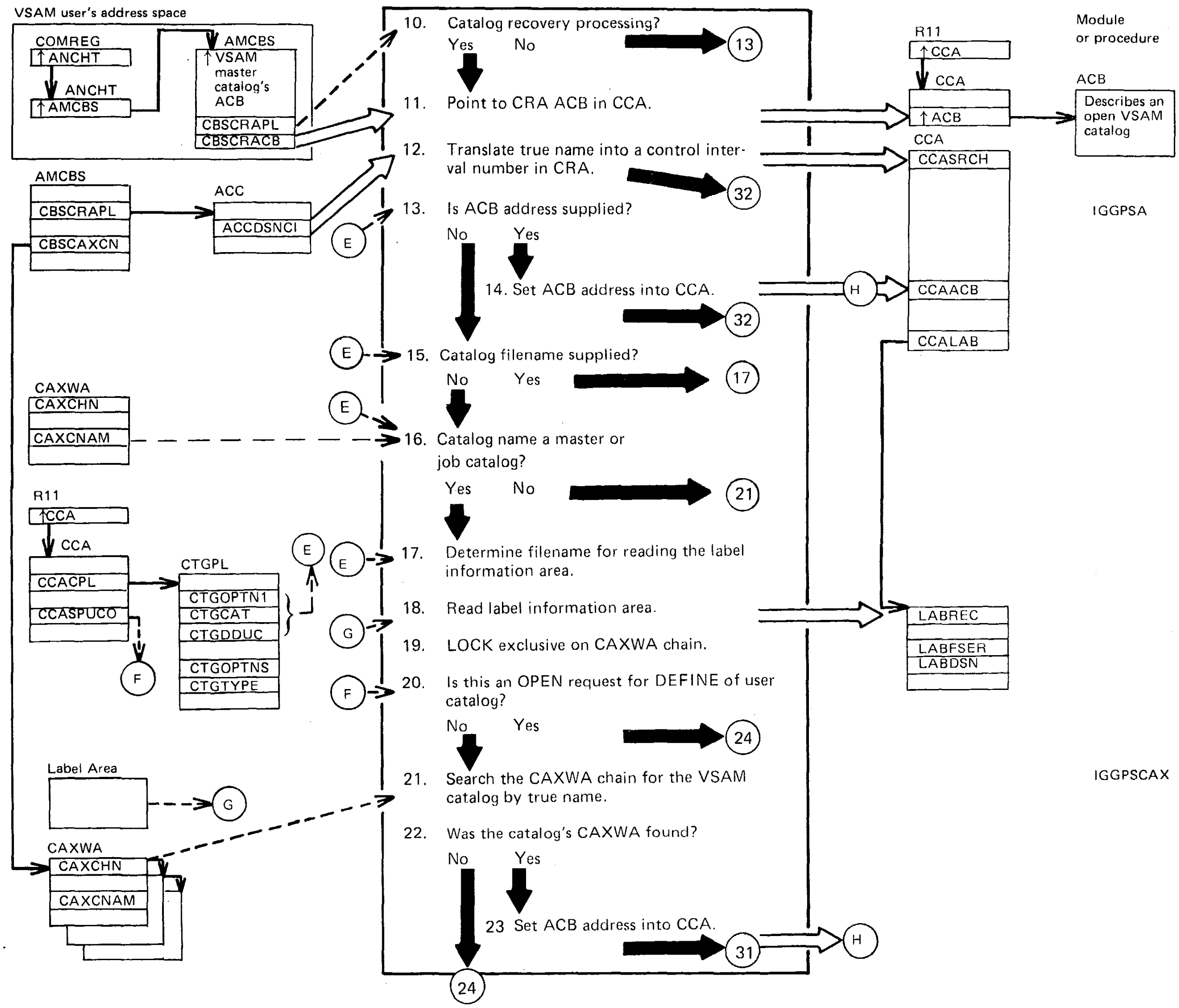
See the Data Areas section of this manual for details about the CCA, ACB, and CTGPL.

The CMS DEFINE routine calls the search catalog routine to confirm that when a caller wants to create a VSAM cluster or catalog, the new cluster or catalog DSNAME is not duplicated in the catalog. The caller (CMS DEFINE) expects the "no record found" return code.

4. If the catalog record identifier is not a true name, it must be a control interval number. In this case, the catalog identifier (CTGCAT) must contain the address of the catalog's ACB.

8. The caller did not know the address of the catalog's ACB and IGG0CLFH must locate it.

Diagram DC2. Search: Retrieve the Base Catalog Record



Notes for Diagram DC2

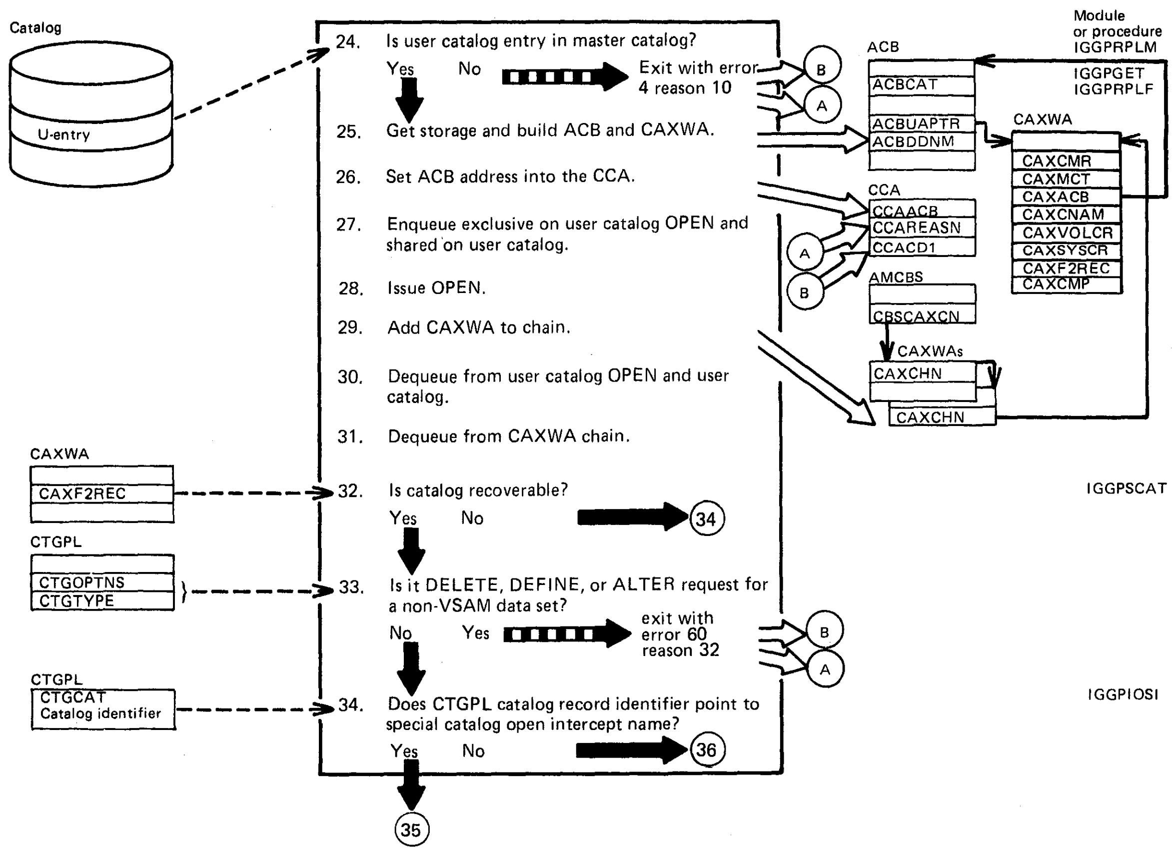
10- 12. If the AMCBS indicates the presence of IDCDF60 (an Access Method Services parameter list), catalog management must use the CRA to retrieve data set information. This is the type of processing used by the EXPORTRA and IMPORTRA functions.

13- 18. If the CTGPL's catalog identifier field CTGCAT and/or the CTGPL's pointer to the user catalog's filename CTGDDUC contain 0, then the job catalog is searched if it exists, and if it does not, the master catalog is searched. Otherwise the specified user catalog is used. The AMCBS contains the address of the CAXWA chain.

20. For DEFINE of user catalog, the CAXWA chain cannot contain the requested CAXWA. Therefore no scanning is necessary.

21. 22. The CAXWA chain contains only CAXWAs referring to the ACBs of open catalogs or CRAs.

Diagram DC3. Search: Retrieve the Base Catalog Record

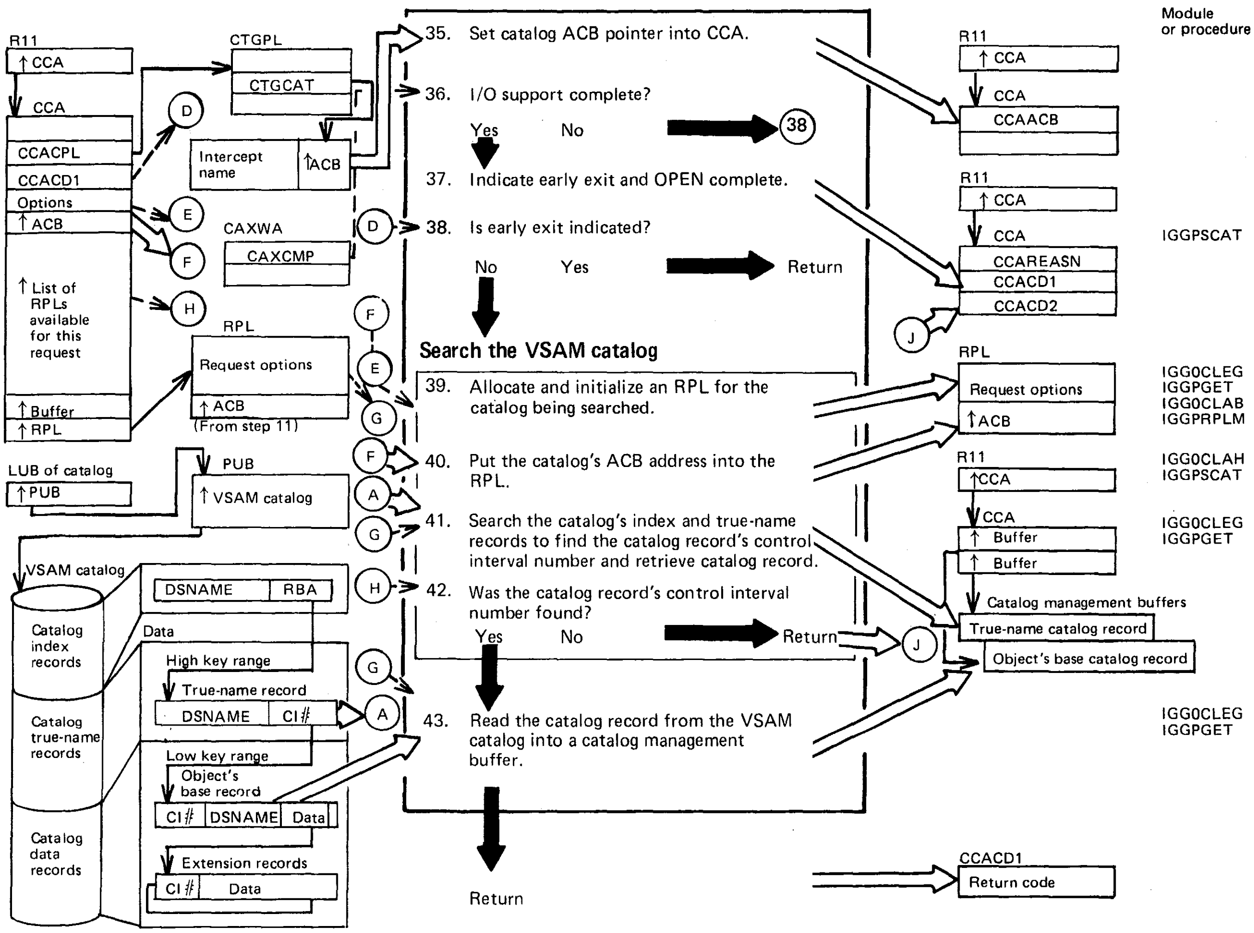


Notes for Diagram DC3

24. If the CAXWA was not found, then the corresponding user-catalog must be opened, if an entry exists in the master catalog.

28. OPEN recursively calls the search catalog module. Procedure IGGPIOSI takes account of OPEN-build case.

Diagram DC4. Search: Retrieve the Base Catalog Record



Notes for Diagram DC4

39. If CTGCAT contains a pointer to a 44-character name, it must be the data set name of the catalog as contained in the CAXWA.

    If the CTGPL's catalog record identifier field addresses the record's control interval number, the catalog record can be retrieved without a search of the catalog's index.

    The search catalog routine assigns an RPL to the caller. Catalog management (CM) routines issue GET and PUT macro instructions to retrieve and write catalog records. Each record management request (GET, PUT, etc.) needed to satisfy the caller's CM request refers to the RPL. The RPL is initialized for a caller and used as often as necessary to process the caller's CM request. When the caller's CM request is completed, the RPL is assigned to another caller.

40. At this time, the CCA (CCAACB) contains the ACB address of the catalog or CRA.

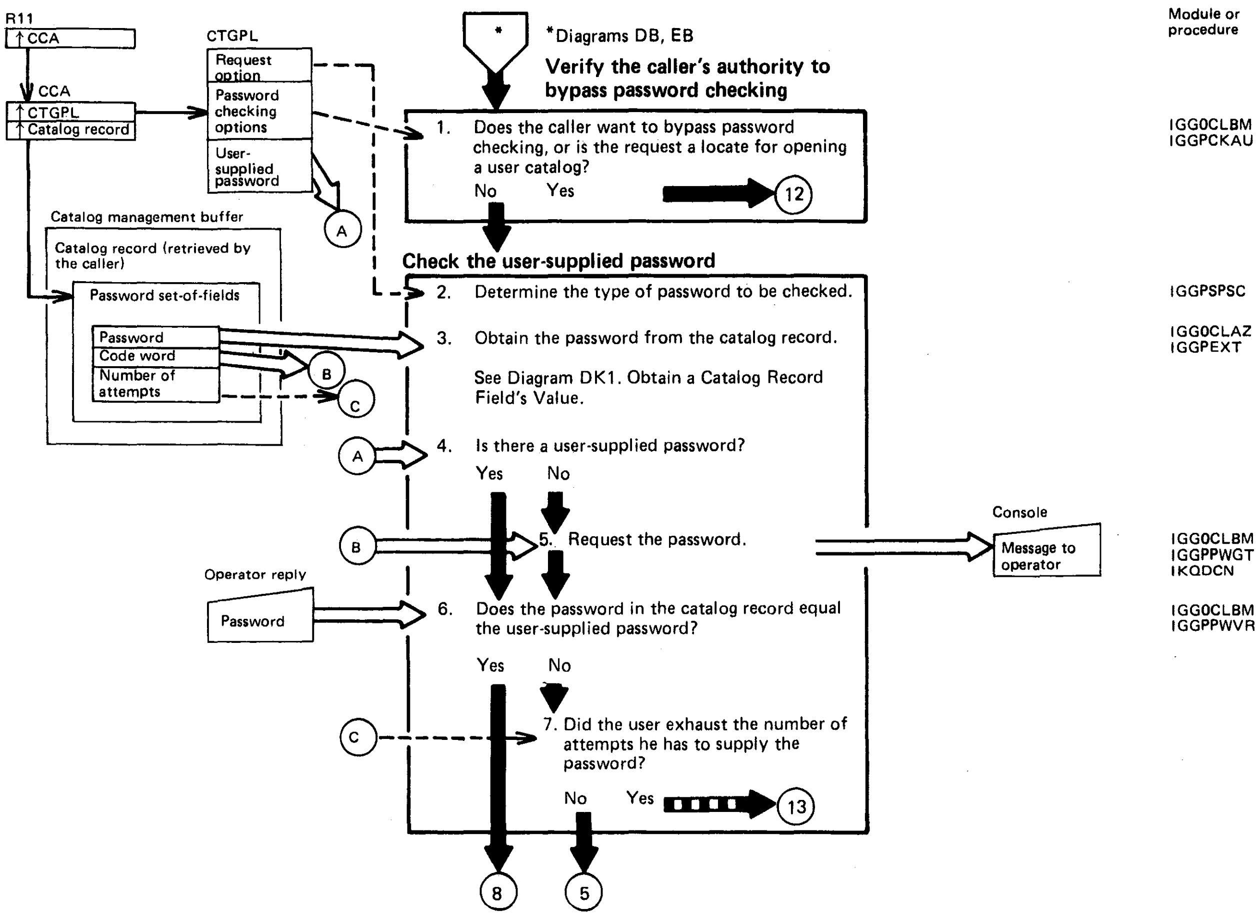
41. The goal of the search is to find the true-name record identified by the DSNAME or the volume serial number. The true-name record contains the cluster's DSNAME or volume serial number and the control interval number of the cluster or volume catalog record.

    The search catalog routine sets the "no record found" error code in the field CCACD2 of the CCA and returns to the caller. If the VSAM catalog has been unsuccessfully searched, the search catalog routine returns to the caller with the same error code set.

    See the Diagnostic Aids section of this manual for CM error codes.

43. The catalog record is located by its control interval number and read into a CM buffer. The buffer's address is put into the CCA.

Diagram DD1. Check the Password



Notes for Diagram DD1

When the VSAM Open routine (IKQOPN) calls VSAM catalog management to retrieve a cluster catalog record, the password checking routine (IGG0CLBM) confirms the user's authorization to gain access to the cluster.

When an Access Method Services routine calls a catalog management services routine (see Diagram EB1, step 1), the password checking routine confirms the user's password to gain access to the VSAM catalog or a specific catalog record.

The catalog record containing the password(s) is available in the buffer addressed in the caller's CCA.

The type of processing that the user is allowed to do with the data set is determined by the password:

Master password: The user is allowed to modify passwords and catalog records that describe his data set, and to process his data set's control intervals and records.

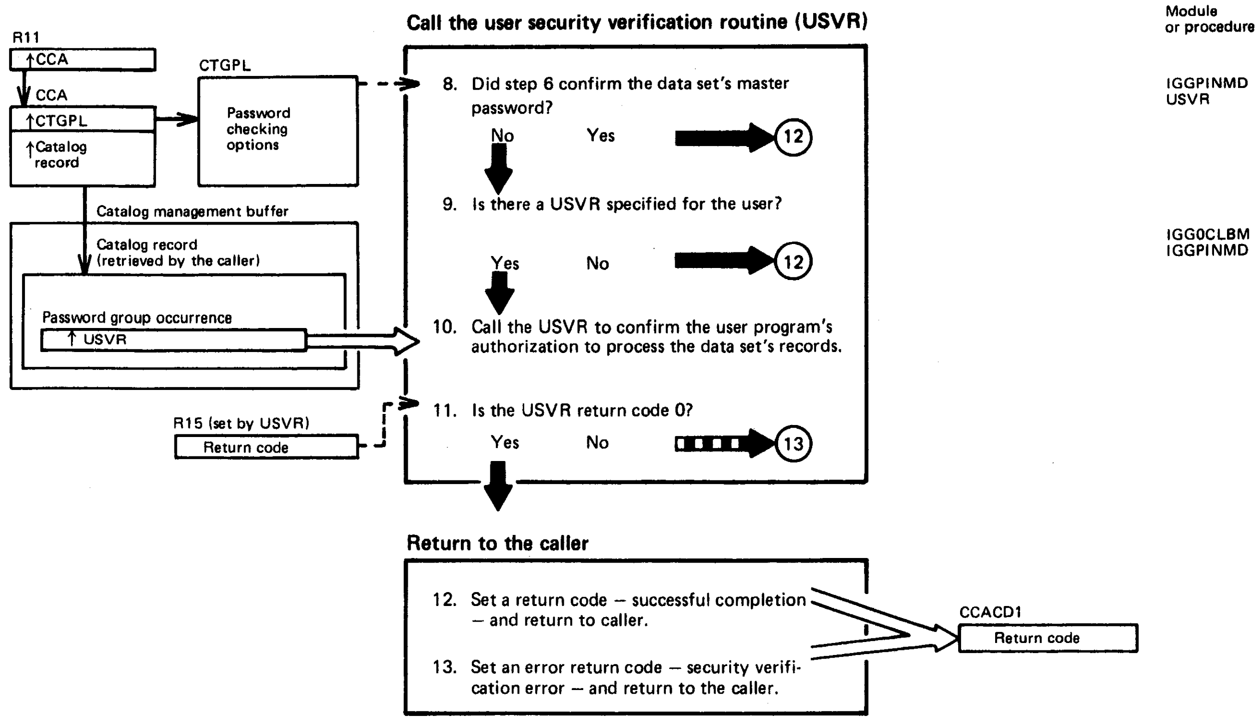
Control-interval password: The user is allowed to process the data set's control intervals as well as its records.

Update password: The user is allowed to process his data set's records.

Read-only password: The user is allowed to read, but not to write (add or update), records in his data set.

1. If the user's password has been verified during a previous CM request, the caller (VSAM Open or CMS) can set the CTGPL's bypass-password-checking flag on.
2. The caller can indicate the minimum level of password to be verified with the CTGPL, but the password checking routine determines the type of password required for the request.
3. The password is in the password group occurrence in the catalog record.
5. The console operator can reply to the VSAM request for a password message with a password. If the operator replies with CANCEL or EOB, error code 56 is returned and module IGG0CLC9 cancels the job.

Diagram DD2. Check the Password



Notes for Diagram DD2

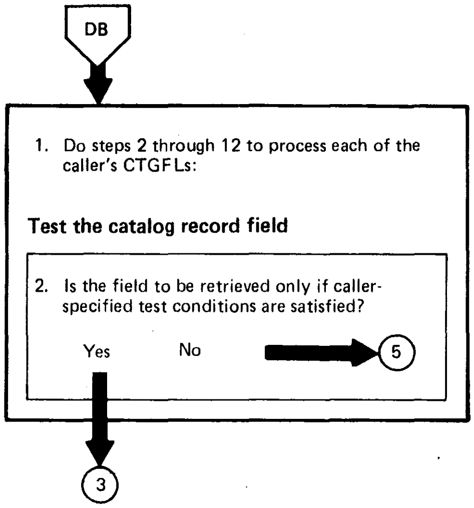
8. If the user supplied the correct master password, the user security verification routine (USVR), if it exists, is bypassed. If a USVR exists, the USVR exit is taken even though the user provided another type of password correctly.

9. If a user security verification routine exists for the user, its name is in the catalog record's password group occurrence.

   See "Data Areas" for details about the cluster catalog record and password group occurrence.

10. The user security verification routine is an installation-supplied routine that confirms a user's authorization to gain access to the data set. The USVR confirms that the user satisfied the installation's security verification criteria.

Diagram DE1. Locate: Retrieve Catalog Information

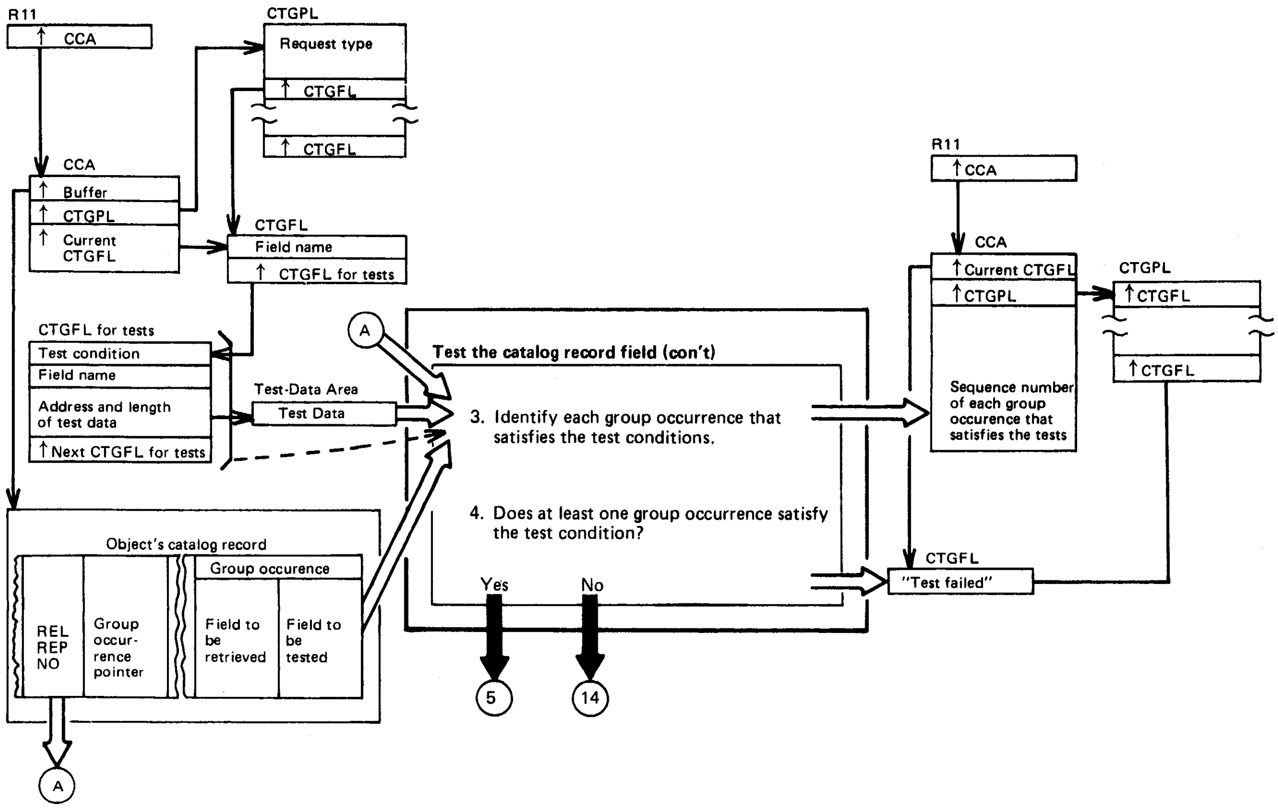


Notes for Diagram DE1

Description	Module	Procedure
The VSAM Open routine (IKQOPN) issues the CATLG instruction to obtain data set and volume information about the user's data set and index.		
The VSAM end-of-volume routine (IKQEDX) issues the CATLG macro instruction to obtain volume information about the extents added to the user's data set.		
When the caller issues a CATLG macro instruction, register 1 points to the caller's CTGPL. The CTGPL's request options are decoded and the base catalog record is retrieved for the request.		
1. The Locate routine processes each CTGFL associated with the caller's CTGPL and returns as much caller-requested data (in the caller's work area) as the caller's test conditions and work area size permit.	IGG0CLAZ	IGGPLOC

Description	Module	Procedure
2. The caller's CTGFL list (CTGFIELD in the CTGPL) contains the address of each CTGFL required to satisfy the caller's need for catalog information. Each CTGFL describes one of the catalog record fields to be retrieved. Each CTGFL is completely processed before the next one is started.		
A caller might make conditional requests for retrieval of catalog record fields. For example, a chain of CTGFLs might be supplied with the request and processed together. The first CTGFL identifies a field to be retrieved and points to subsequent CTGFLs that contain the names of the catalog fields to be tested, the test conditions (equal, low, high, etc.) and the address and length of the caller's test data area. The catalog record fields identified by the second and subsequent CTGFLs are compared to (or tested against) the caller's data. If the comparison satisfies the test conditions, the catalog record field specified by the first CTGFL is retrieved.		

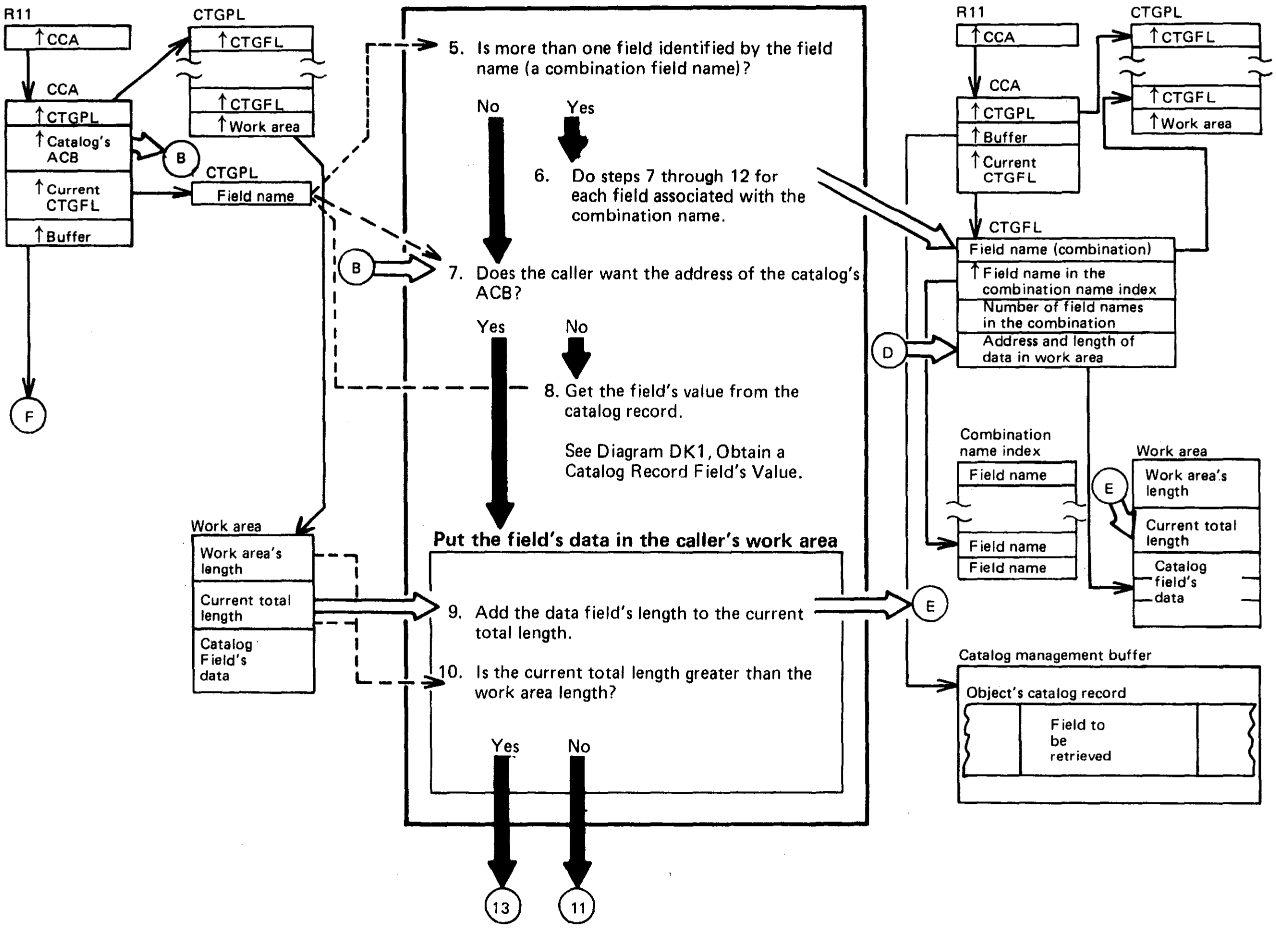
Diagram DE2. Locate: Retrieve Catalog Information



Notes for Diagram DE2

Description	Module	Procedure
3. If the caller wants to retrieve a catalog record's header field, the field's data is retrieved if all tests are satisfied. If the caller wants to retrieve a field from one of the group occurrences that follow the header field, the field's data is retrieved from each group occurence that satisfies all tests. The sequence number of each group occurrence that satisfies the tests is put in the CCA. When all group occurrences have been tested, the sequence numbers in the CCA are used to identify each group occurrence that contains caller-requested data.	IGG0CLBA	IGGPTSTS IGGPGREC
4. If none of the group occurrences satisfy the test conditions specified by test CTGFLs, the next CTGFL in the catalog parameter list (CTGPL) is processed.	IGG0CLAZ	IGGPSCNF

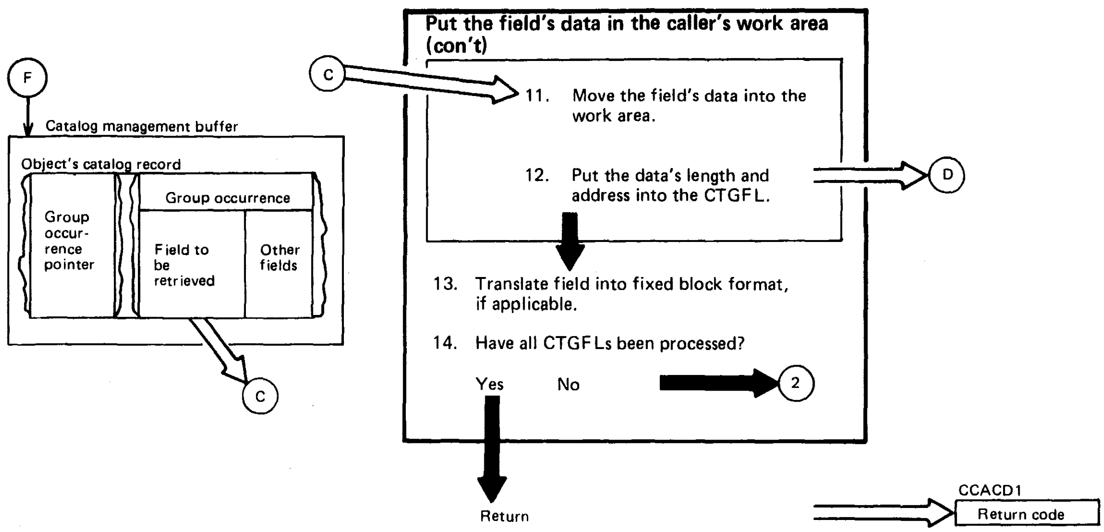
Diagram DE3. Locate: Retrieve Catalog Information



Notes for Diagram DE3

Description	Module	Procedure
5. A combination name refers to a set of related catalog field names, and is used by the caller instead of a separate CTGFL for each field name.	IGG0CLAZ	IGGPLOC2
6. The combination name index has an entry for each field name in the combination. The Locate routine processes each field name entry in the combination name index sequentially, starting at the address of the first field name entry for the combination, and ending when the number of entries processed equals the number of field names associated with the combination name. The test sequence, if any, associated with a combination-name CTGFL is done only once, not once for each field name in the combination.	IGG0CLAZ	IGGPLOC2
7. The address of the catalog's ACB is in the CCA. All other catalog record fields that the caller can request are in the catalog record. Each catalog record field is identified by its field name.	IGG0CLAZ	IGGPLOC2
8. Diagram DK1 shows how the requested catalog record field (specified by its field name in the CTGFL) is located for the Locate routine.	IGG0CLBA	IGGPGVAL IGGPGREC
9. The first two fields in the caller's work area specify the number of bytes the caller allocated to the work area and the number of bytes that contain catalog record field data (the current total length field).	IGG0CLAZ	IGGPLOC2 IGGPSHIN
10. If the current total length exceeds the work area length, the current total length field is updated with the length of the catalog record data, but the data itself is not moved in the caller's work area.	IGG0CLAZ	IGGPSHIN

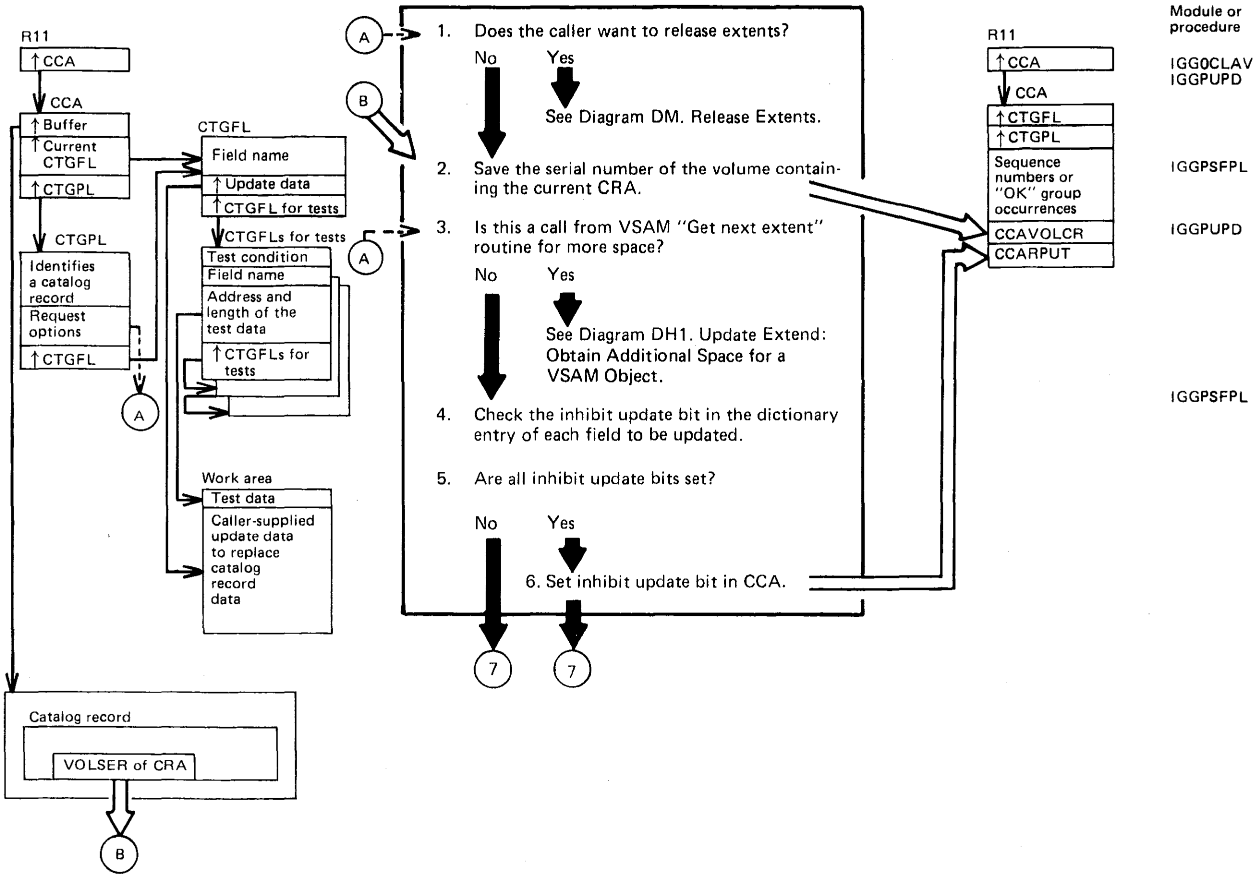
Diagram DE4. Locate: Retrieve Catalog Information



Notes for Diagram DE4

Description	Module	Procedure
11. The Locate routine puts the beginning address and the length of the catalog field into the CTGFL's field-data entry.	IGG0CLAZ	IGGPSHIN
12. The CTGFL's field-data entry contains the beginning address and length of the data in the caller's work area. When control is returned to the caller, the caller can use the field-data entry to locate a specific field's data in the work area.	IGG0CLAZ	IGGPSHIN
13. If the dictionary information indicates that the field value is to be returned in fixed block format (units of blocks rather than tracks and/or cylinders), translate the field value.	IGG0CLEZ	IGGPLTRN

Diagram DG1. Update: Modify Catalog Information



Notes for Diagram DG1

The VSAM Close routine (IKQCLO) issues the CATLG macro instruction to modify the data set and index statistics maintained in the catalog record's copy of the AMDSB.

The VSAM "Get new extent" routine (IKQNEX) issues the CATLG macro instruction to obtain more space for a data set.

When the caller issues the CATLG macro instruction, register 1 points to the caller's CTGPL. The CTGPL request options are decoded, and the base catalog record is retrieved for the request.

1. If this is a call to release secondary extents (for a reusable data set), routine IGGPRELE in module IGG0CLCB is called.

2. This information is saved in order to be able to supply it for extension records.

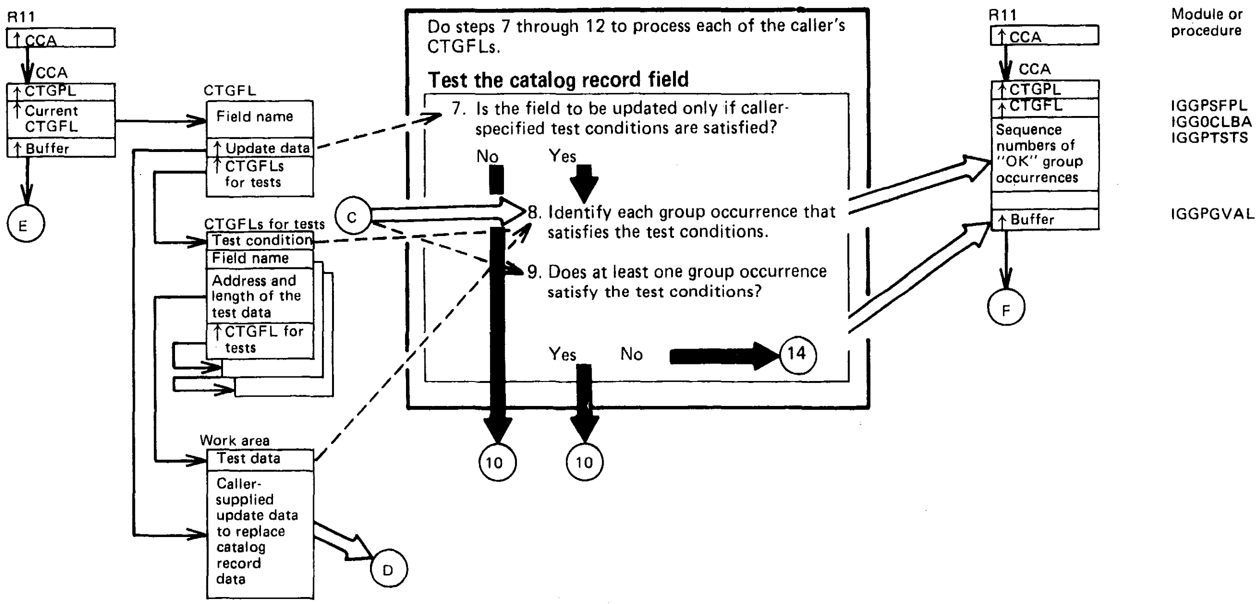
3. If more space is required for the data set, the UPDATE-Extend routine (IGG0CLBB) processes the caller's Update request and returns directly to the caller (the VSAM "Get new extent" routine). See Diagram DH for information about UPDATE-Extend processing.

4. Scan through the dictionary information of all fields to be updated.

5. If there are one or more fields in the list that require CRA update, the inhibit CRA update bit in the CCA (which was initially set "on") is cleared.

Steps 7 through 12 are performed to update each of the catalog record fields identified by the caller's CTGFL.

Diagram DG2. Update: Modify Catalog Information



Notes for Diagram DG2

7. The caller's CTGFL list (CTGFIELD in the CTGPL) contains the address of each CTGFL needed to satisfy the caller's updating requirements. Each CTGFL describes one of the catalog record fields to be updated. Each CTGFL is completely processed before the next one is started.

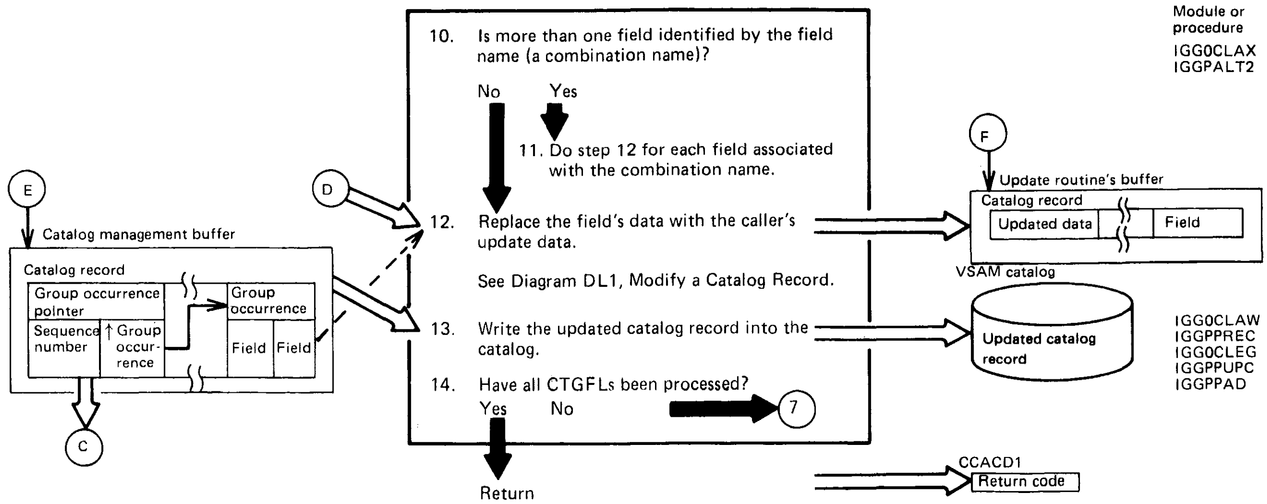
   The caller may want to update a field only if another field's value, when compared to the caller's test value, satisfies the caller's test conditions. If so, the caller builds a CTGFL that contains the name of the catalog field to be tested, the test conditions (equal, high, low, etc.), and the address and length of the caller's test value. If a CTGFL contains the address of another CTGFL, the second CTGFL describes a catalog record field that is to be compared to the caller's data. If the comparison satisfies the test conditions, the catalog record field specified by the first CTGFL is updated with the caller's data.

8. If the caller wants to update a catalog record's header field, the field's data is updated with the caller's data if all tests are satisfied.

   If the caller wants to update a field from one of the group occurrences that follow the header field, the field's data is updated with the caller's data for each group occurrence field that satisfies all tests. The group occurrence that contains the field to be updated can also be identified by its sequence number.

   The sequence number of each group occurrence that satisfies the tests is put in the CCA. When all group occurrences have been tested, the sequence numbers are used to identify each group occurrence that contains caller-requested data.

Diagram DG3. Update: Modify Catalog Information



Notes for Diagram DG3

10. A combination name refers to a set of related catalog field names, and is used by the caller instead of a separate CTGFL for each field name.

11. The CCA's combination name index has an entry for each field name in the combination. The Update routine processes each field name entry in the combination name index sequentially, starting at the address of the first field name entry for the combination, and ending when the number of entries processed equals the number of field names associated with the combination name.

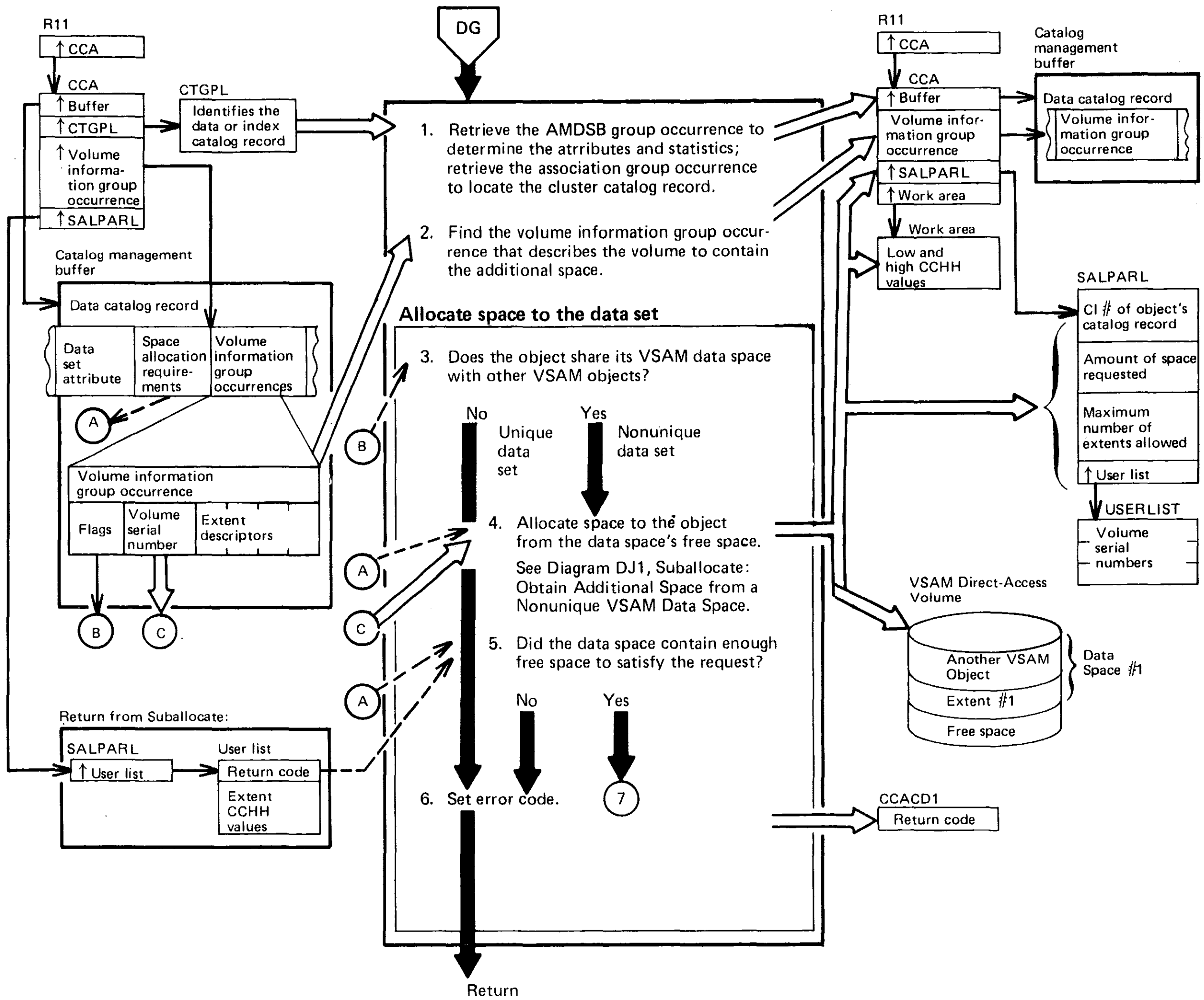
    The combination name's CTGFL contains the beginning address and the total length of the group of update data fields in the caller's work area.

    The test sequence, if any, associated with a combination-name CTGFL is done only once, not once for each field name in the combination.

13. When the catalog record is updated (in a buffer in the Update routine's virtual storage), the Update routine sets the "must write" flag on to indicate that the I/O manager must write the buffer from storage into the catalog before the buffer can be made available to contain another catalog record. When the caller's Update routine needs the buffer to process another catalog record associated with the request, the Update routine writes the catalog record from the buffer into the VSAM catalog (on a direct-access storage device).

Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT (the update inhibit bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.

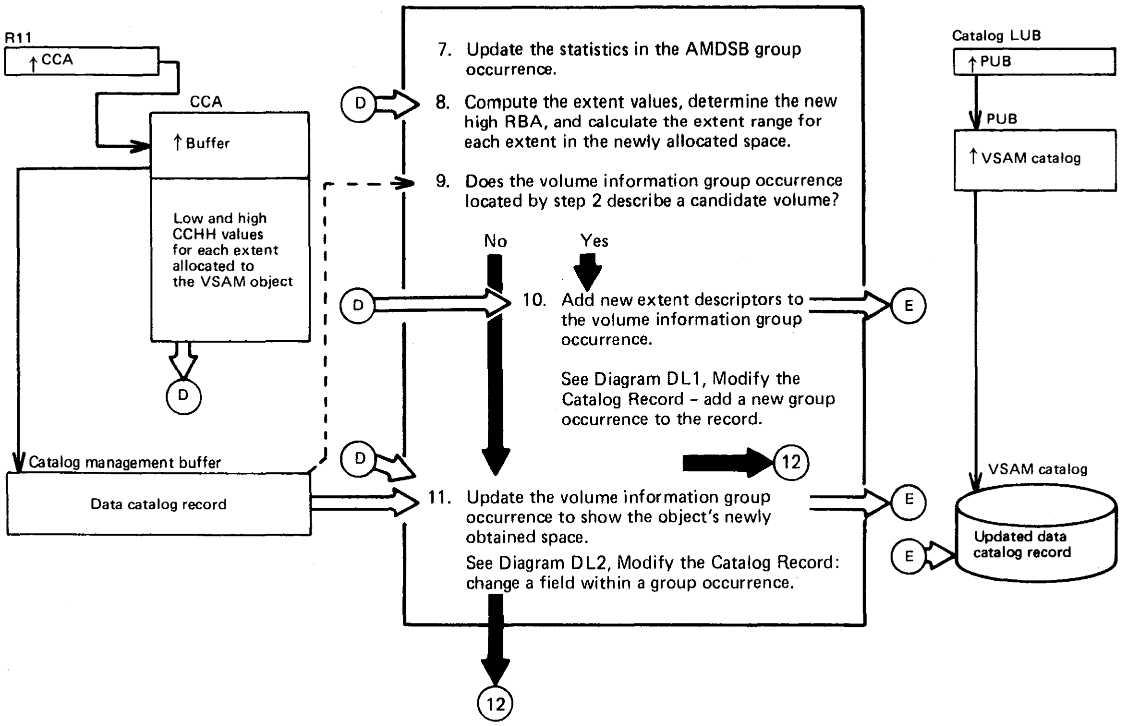
Diagram DH1. Update-Extend: Obtain Additional Space For a VSAM Object



Notes for Diagram DH1

	Description	Module	Procedure
	The UPDATE-Extend routine is called whenever a VSAM object (data set, index, or catalog) needs more space to store its records.		
	The VSAM "Get new extent" routine calls the CM Update routine, and an amount of space (based on the object's direct-access space allocation requirements) is allocated from a shared VSAM data space that has enough free space to satisfy the allocation requirements.	IGG0CLBB IGG0CLBC	IGGPUPDE IGGPINIT
2.	The volume information group occurrence is identified by either an RBA or a key value.	IGG0CLAZ IGG0CLBB IGG0CLBC	IGGPEXT IGGPUPDE IGGPSVOL
3.	A shared (nonunique) VSAM data space contains all or parts of two or more VSAM objects. A unique VSAM data space contains all or part of only one VSAM object, and is not allowed to contain records of another object.	IGG0CLBB IGG0CLEG	IGGPUPDE IGGPUALL IGGPGET
4.	If the object shares its data space with other VSAM data sets or indexes, there might be enough free space in one of the data spaces on the volume to satisfy the object's direct-access space allocation requirements.	IGG0CLBB IGG0CLAR	IGGPUPDE IGGPCSAL IGGPSALL

Diagram DH2. Update-Extend: Obtain Additional Space for a VSAM Object

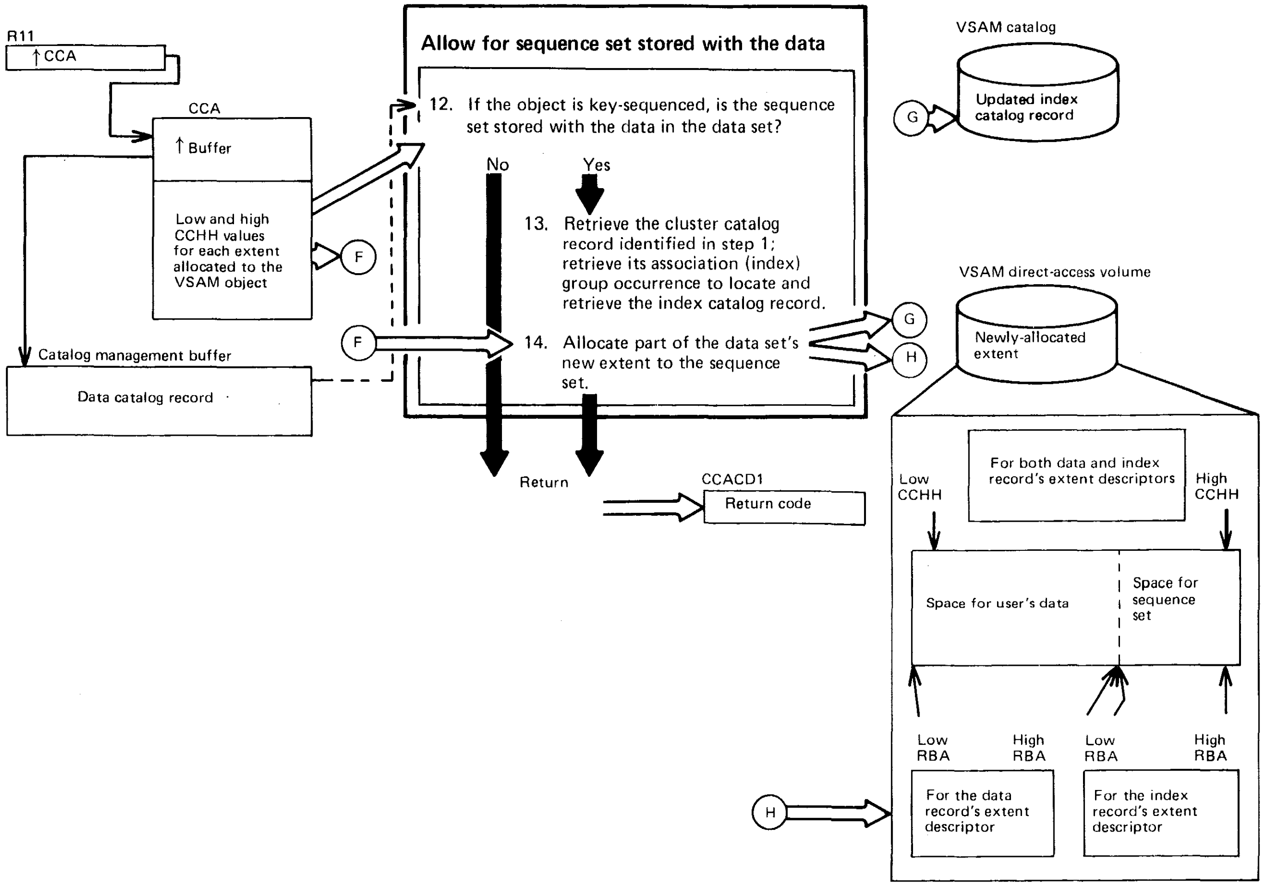


Notes for Diagram DH2

	Description	Module	Procedure
7.	The object's catalog record contains a volume information group occurrence to describe the object's space on each volume that contains a part of the object. If the object's newly obtained extent is on a new volume, the UPDATE-Extend routine builds a volume information group occurrence to describe the new volume and extent. Otherwise, an existing volume information group occurrence is updated with the high-allocated RBA and extent information in the form:	IGG0CLBB IGG0CLAV	IGGPMVOL IGGPMOD

SS CCHH CCHH DDDD DDDD
where:
- SS identifies the VSAM data space
- CCHH are the low and high cylinder and track addresses
- DDDD are the low and high RBAs

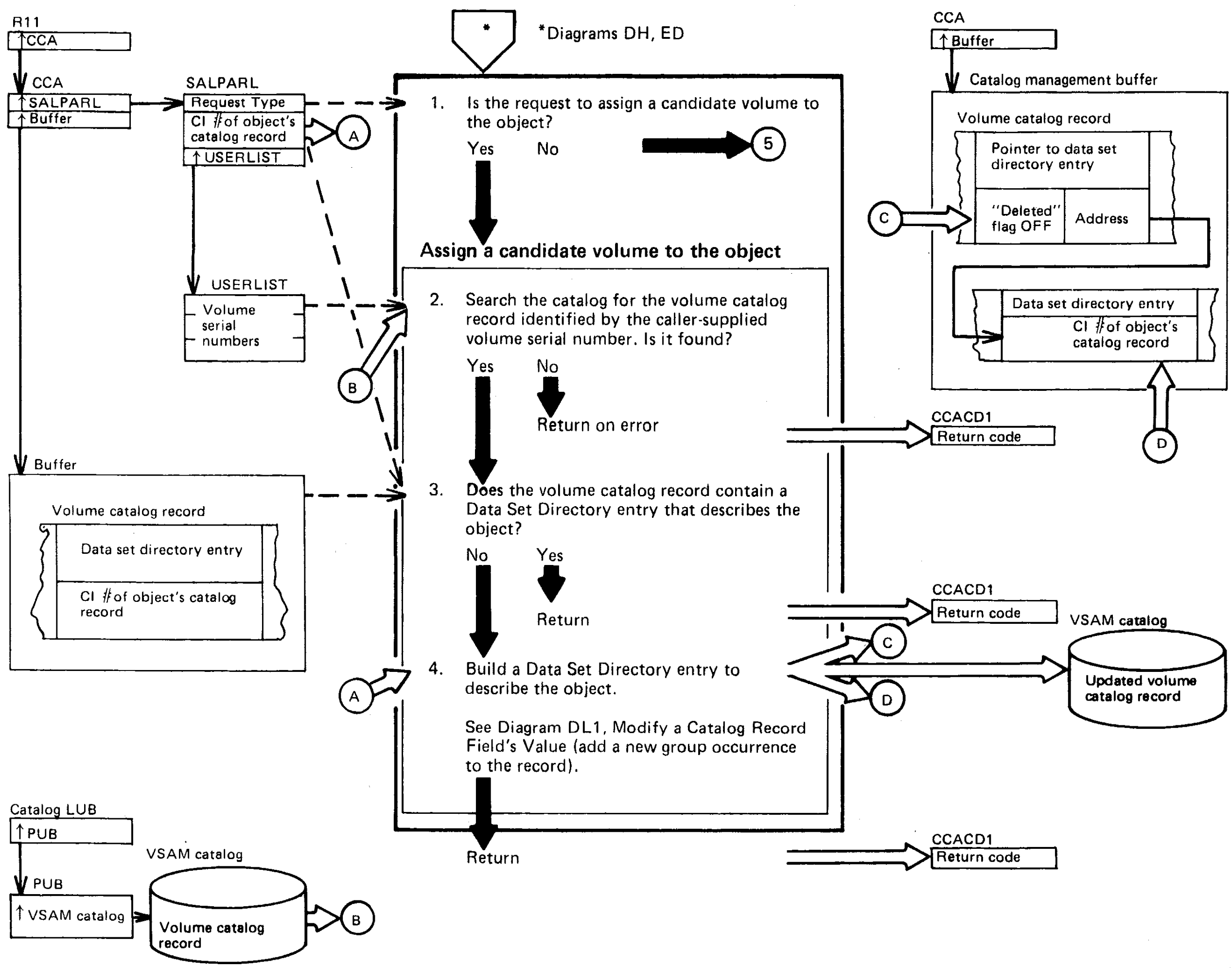
Diagram DH3. Update-extend: Obtain Additional Space for a VSAM Object



Notes for Diagram DH3

Description	Module	Procedure
14. The low and high addresses in the index catalog record's volume information group occurrence are those of the extent obtained for the data set. The low and high RBA values are for the sequence set.	IGG0CLBB	IGGPSSWD
	IGG0CLEG	IGGPGET
	IGG0CLBB	IGGPSSWD
	IGG0CLBC	IGGPINIT
	IGG0CLBB	IGGPSSWD
	IGG0CLBC	IGGPSVOL

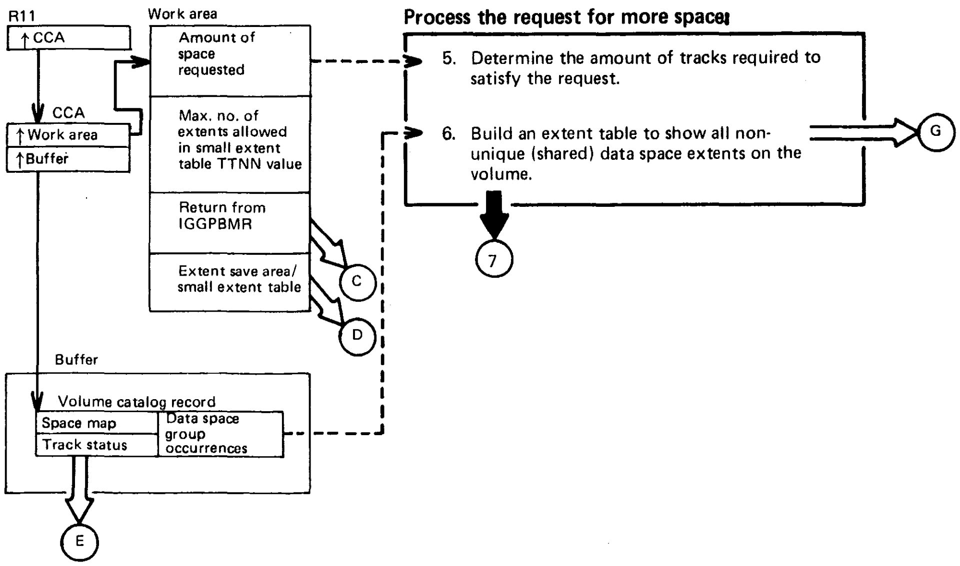
Diagram DJ1. Suballocate: Obtain Additional Space From a Nonunique VSAM Data Space



Notes for Diagram DJ1

Description	Module	Procedure
The suballocate routine, IGG0CLAR, is called to assign a candidate volume to a VSAM object (data set, index, or catalog) and to assign available spaces on the caller-specified volume. The caller, either the UPDATE-Extend routine (Diagram DH) or the DEFINE Cluster routine (Diagram ED) builds a list of volume serial numbers to identify each volume to be assigned to the object as a candidate volume. If the caller requests space allocated to the object, the list contains one volume serial number.		
Note: Although space requests on FBA devices are externally specified in BLOCKS (or RECORDS), catalog management processes them as track and cylinder addresses.		
1. A candidate volume is available to contain a VSAM object's catalog records, but no space is allocated to the data space from the volume as yet.	IGG0CLAR	IGGPSALL
2. The volume must be owned by the same catalog that contains the object.	IGG0CLAR IGG0CLEG	IGGPSALL IGGPGET
3. The volume catalog record already contains a data set directory entry, the volume is either already assigned to the VSAM object as a candidate volume or has some of its space allocated to the VSAM object.	IGG0CLAR IGG0CLAZ	IGGPSALL IGGPEXT
4. The new data set directory is added to the volume catalog record.	IGG0CLAR IGG0CLAG IGG0CLAR IGG0CLAV	IGGPSALL IGGPISCI IGGPSALL IGGPMOD

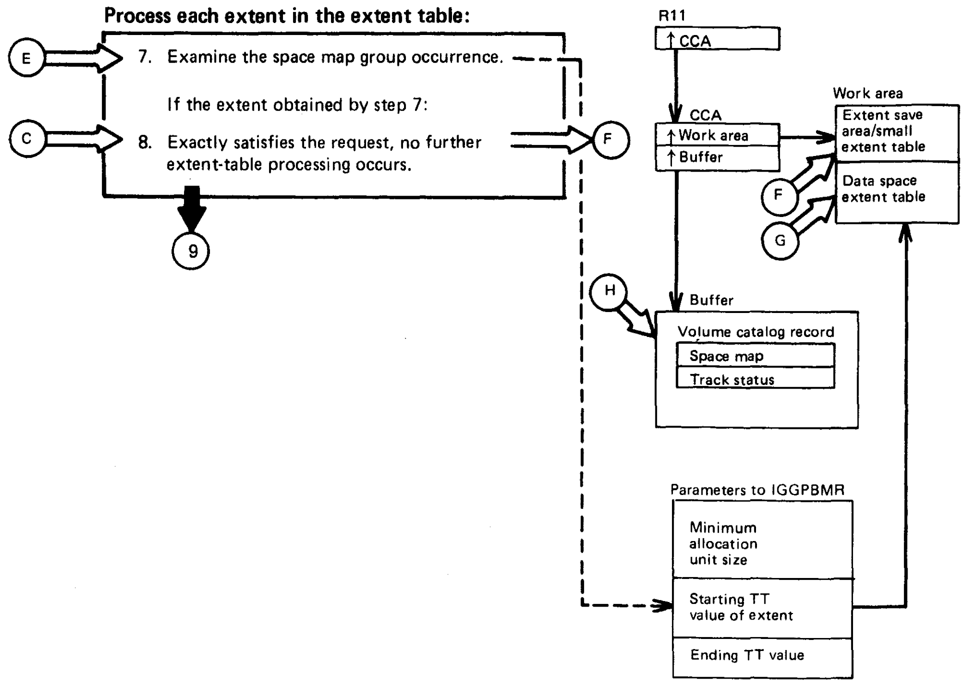
Diagram DJ2. Suballocate: Obtain Additional Space From a Nonunique VSAM Data Space



Notes for Diagram DJ2

Description	Module	Procedure
5. If the amount of space requested is a number of cylinders, convert it to a number of tracks.	IGGOCLAR IGGOCLAU	IGGPSALL IGGPSALS
6. The extent table is built by retrieving each extent descriptor (from each data space occurrence) that might contain enough free space to satisfy the request's minimum allocation requirement (the number of tracks in one control area).	IGGOCLAZ	IGGPEXT

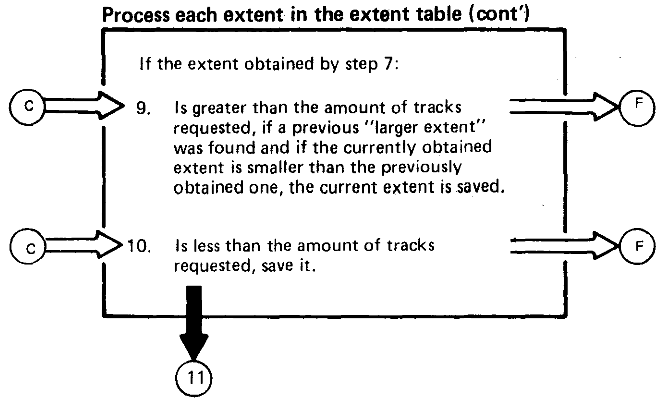
Diagram DJ3. Suballocate: Obtain Additional Space From a Nonunique VSAM Data Space



Notes for Diagram DJ3

Description	Module	Procedure
7. Each extent descriptor in the extent table is processed beginning with the lowest extent starting track number (TT) in the table, in ascending sequence, until all extent descriptors have been processed.	IGG0CLAU	IGGPSALS IGGPEDS
	IGG0CLBR	IGGPBMR
IGGPBMR examines each extent to find an amount of contiguous unallocated tracks at least as large as the request's minimum allocation unit. IGGPBMR examines the space map group occurrence, starting at bit position TT (track indicator) and ending at bit position TT plus the number of tracks in the extent (NN) minus 1. If IGGPBMR finds a large enough amount of unallocated tracks, it returns to IGGPEDS with the beginning track number (TT) and the number of tracks (NN). If the data space's extent contains another amount of unallocated tracks at least as large as the request's minimum allocation unit, IGGPEDS calls IGGPBMR to examine the rest of the data space's extent.		
8. If the extent returned by IGGPBMR is the exact number of tracks required to satisfy the caller's request, no further extent table processing is done. Larger or smaller extents obtained from previous extent table entries are ignored.	IGG0CLAU	IGGPEDS

Diagram DJ4. Suballocate: Obtain Additional Space From a Nonunique VSAM Data Space



Notes for Diagram DJ4

Description	Module	Procedure
9. If the extent returned by IGGPBMR is larger than the amount of tracks required to satisfy the request, the extent is saved if either:	IGG0CLAU	IGGPEDS

- No other larger-than-requested-amount extent has been returned yet, or
- The current extent is smaller than a previously obtained larger-than-requested-amount extent.

In either case, only one larger-than-requested-amount extent value is saved. The small extent table (built in step 10) is ignored and no longer used.

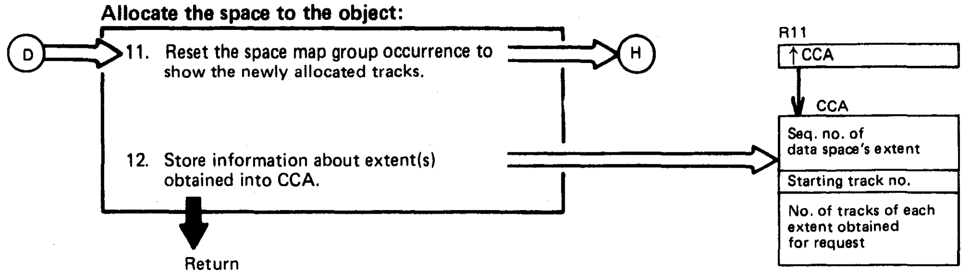
Description	Module	Procedure
10. If the extent returned by IGGPBMR is smaller than the amount of tracks required to satisfy the request, its TTNN value is adjusted so that TT is on a cylinder boundary. If NN is now at least as large as the request's minimum allocation unit (number of tracks for one control area), the extent is saved in the small extent table if:	IGG0CLAU	IGGPEDS

- The table has fewer than five entries (or a caller-specified maximum less than five) in it, or
- The table is full and the current extent's NN value is greater than the table's smallest extent. The current extent replaces the table's smallest extent.

In either case, the extent is not put in the small extent table if it is too small (adjusted NN is less than the minimum allocation unit) or if a larger-than-requested-amount extent already exists (see step 9).

If, after all data spaces have been examined, the total of the NN values in the small extent table is less than the amount required to satisfy the request, no space is allocated to the object.

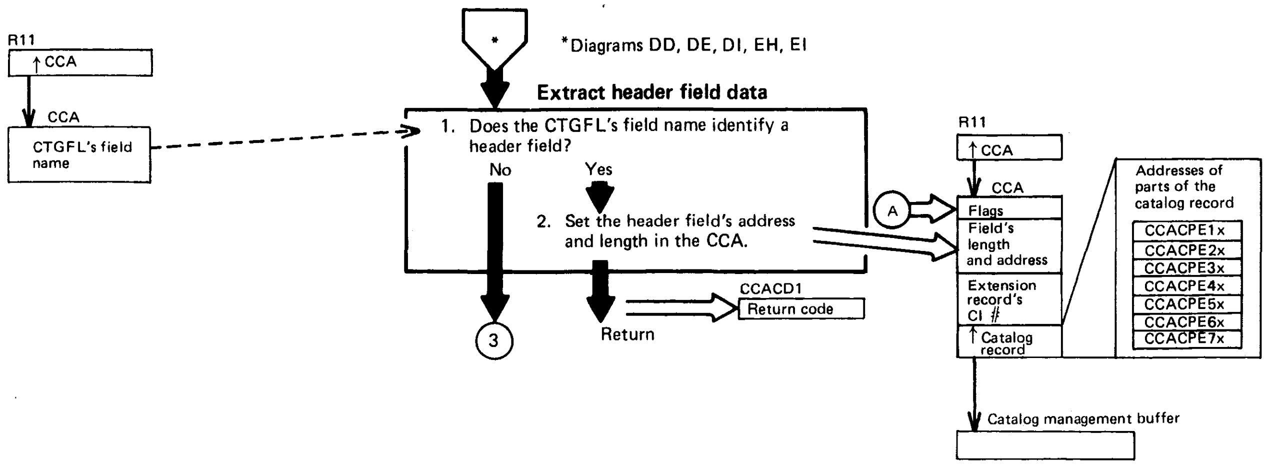
Diagram DJ5. Suballocate: Obtain Additional Space From a Nonunique VSAM Data Space



Notes for Diagram DJ5

Description	Module	Procedure
11. If the selected extent is larger than or equal to the amount of space requested, IGGPBMR adjusts the space map group occurrence starting at bit position TT (track indicator) by turning off NN bits (NN is the exact number of tracks required to satisfy the request).	IGG0CLAU IGG0CLBR	IGGPSALS IGGPBMR
If the space is allocated to an object from a number of extents, the small extent table is sorted so that the largest NN value is first, the smallest last. IGGPBMR then adjusts the space map group occurrence for each TTNN value in the small extent table, until the amount of allocated tracks equals the amount of tracks requested.		
12. IGGPSALS returns the sequence number of the data space's extent, starting track number, and number of tracks of each extent obtained for the request. The caller uses this information to build extent descriptor entries in the VSAM object's volume information group occurrence.	IGG0CLAU	IGGPSALS

Diagram DK1. Obtain a Catalog Record Field's Value



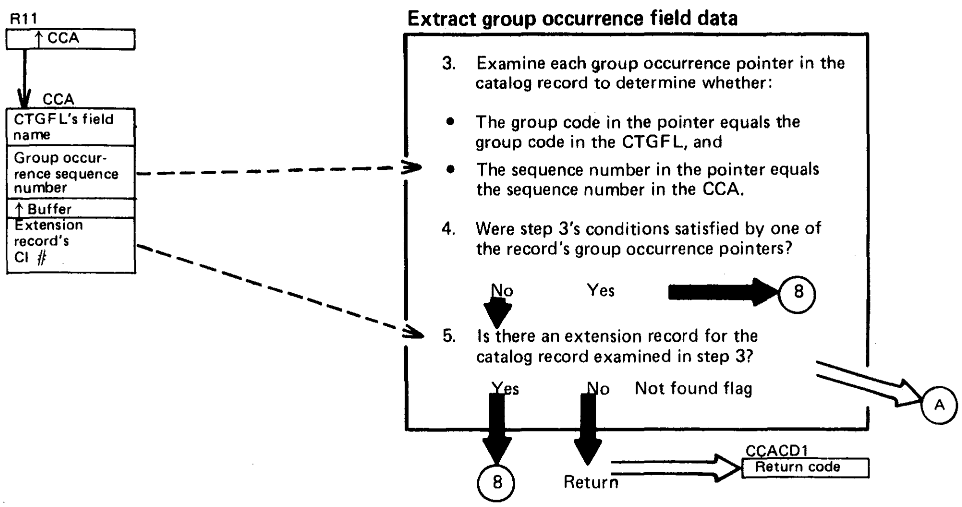
Notes for Diagram DK1

Description	Module	Procedure
The obtain-field-value routine is called by other catalog management (CM) routines to obtain the location and length of a field in a catalog record. The record is in a CM buffer in main storage. The following results could occur:		

- The field is entirely contained in the record in the buffer, and the field's address and length are set in the caller's CCA.
- The field is partially contained in the record in the buffer, and the field's address and partial length are set in the caller's CCA. The CCA also has the "not complete" flag on and contains the control-interval number of the catalog record's extension, which contains more of the field.
- The field is not retrieved because it doesn't exist in the caller-specified group occurrence, or because there are no more group occurrences in the record, or because no buffer space is available to contain extension record.

Description	Module	Procedure
1. The derived volume entry fields are retrieved.	IGG0CLBS	IGGPXVAL
The field-name dictionary is a read-only catalog management table. The catalog field name dictionary contains an entry for each type of catalog record field, based on its field name. The caller puts the dictionary entry identified by the CTGFL's field name into the CCA before calling the obtain-field-value routine.	IGG0CLBA	IGGPGVAL
If the field name identifies a header field, the field's type code (in its dictionary entry) is 0. A nonzero group code identifies the group occurrence that contains the field identified by the CTGFL's field name.		
If the field name identifies a header field and the field is fixed-length, it is at a fixed displacement from the beginning of the catalog record.		
The field's address is obtained by adding the displacement in the CTGFL's dictionary entry to the beginning address of the record (the contents of CCACPE1x). The field's length is part of the CTGFL's dictionary entry.		

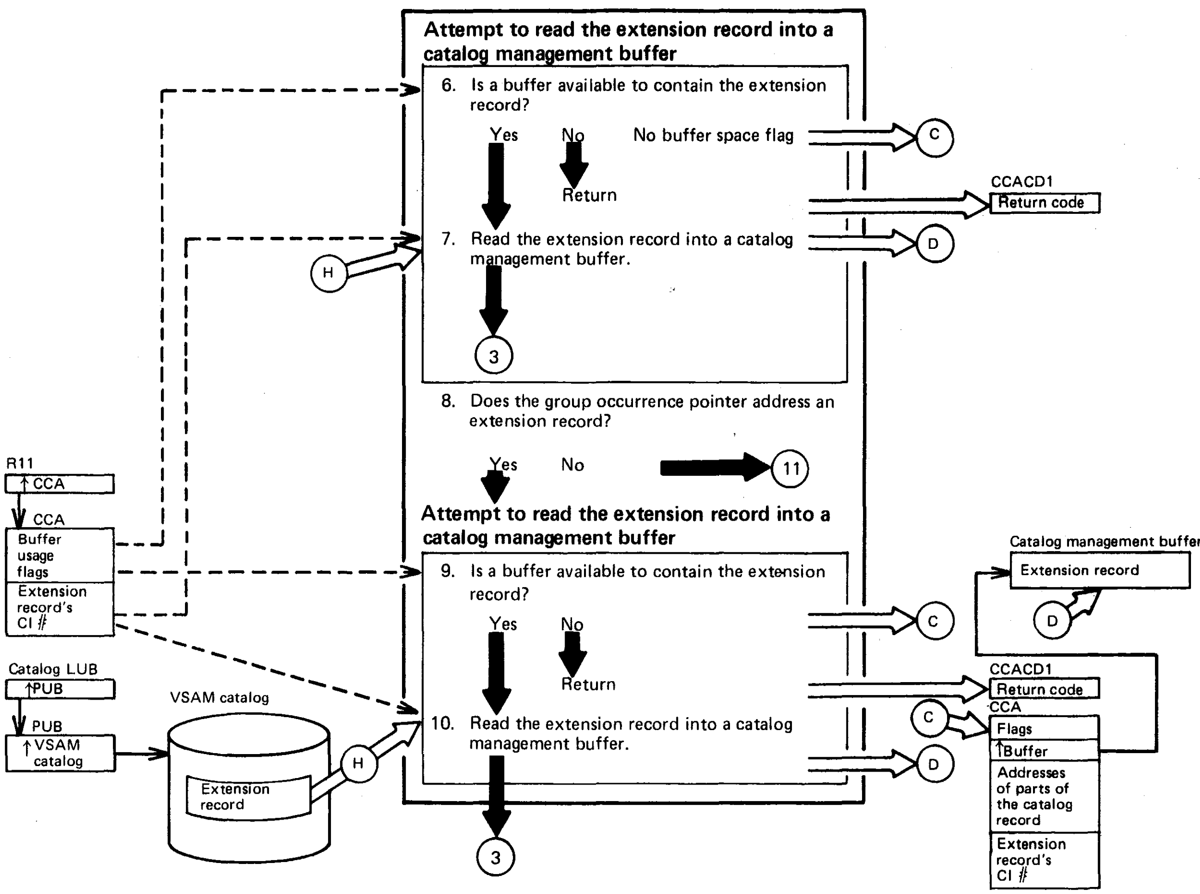
Diagram DK2. Obtain a Catalog Record Field's Value



Notes for Diagram DK2

	Description	Module	Procedure
3.	The group occurrence pointer (GOP) is used to locate a group occurrence. The GOPs are grouped together by a group code. Within each group of GOPs, the pointers are ordered by sequence number.	IGG0CLBA	IGGPLVAL
4.	If the caller-specified GOP, identified by its sequence number, is found, its displacement field and flag field specify the location of its group occurrence as: • The number of bytes from the beginning of the record's group occurrences (the contents of CCACPE3x plus the GOP's displacement field value), or • The control-interval number of the extension record that contains the group occurrence. The extension record contains a GOP that specifies the group occurrence's location as a number of bytes from the beginning of the record's group occurrences.	IGG0CLBA	IGGPLVAL
5.	The actual group occurrences containing the field to be retrieved may be contained in an extension record.	IGG0CLBA	IGGPLVAL

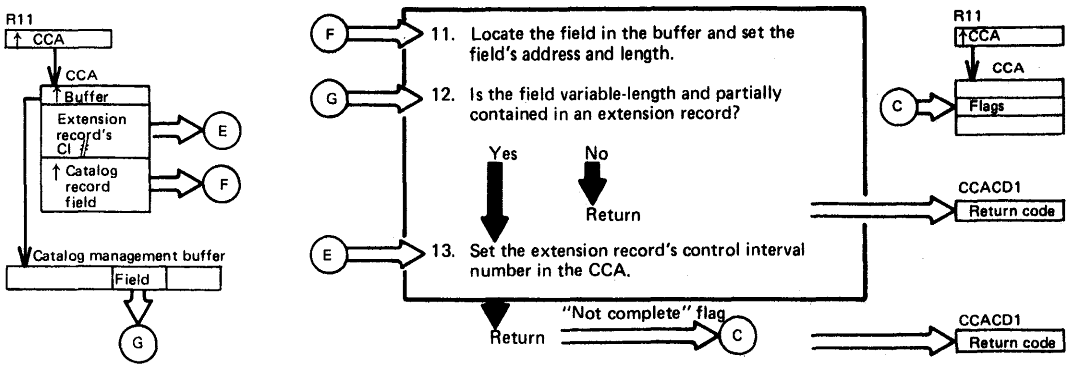
Diagram DK3. Obtain a Catalog Record Field's Value



Notes for Diagram DK3

	Description	Module	Procedure
6.	Each catalog record in the CM buffer is identified by a record area block (RAB) within the CCA. The RAB contains flags that indicate whether or not the buffer can be used to contain another record. If the RAB's "must write" flag is on, the buffer cannot be used for another record until its contents have been written into the catalog.	IGG0CLBA IGG0CLAW IGG0CLBA IGG0CLEG	IGGPGREC IGGPPREC IGGPGREC IGGPGET
	Each CCA contains six record area blocks (RABs). Each CM request can use a maximum of five buffers. If all buffers are filled and cannot be released, the obtain-field-value routine sets the CCA's "no buffer space" flag on.	IGG0CLBA	IGGPLVAL
7.	The CCA's not found flag is set off before returning to step 3 to examine the extension record's GOPs.	IGG0CLBA IGG0CLEG	IGGPGREC IGGPGET
8.	If the GOP contains the control-interval number of an extension record, the group occurrence is contained on that extension record.	IGG0CLBA	IGGPGVAL
9.	See step 6.		
10.	See step 7.		

Diagram DK4. Obtain a Catalog Record Field's Value

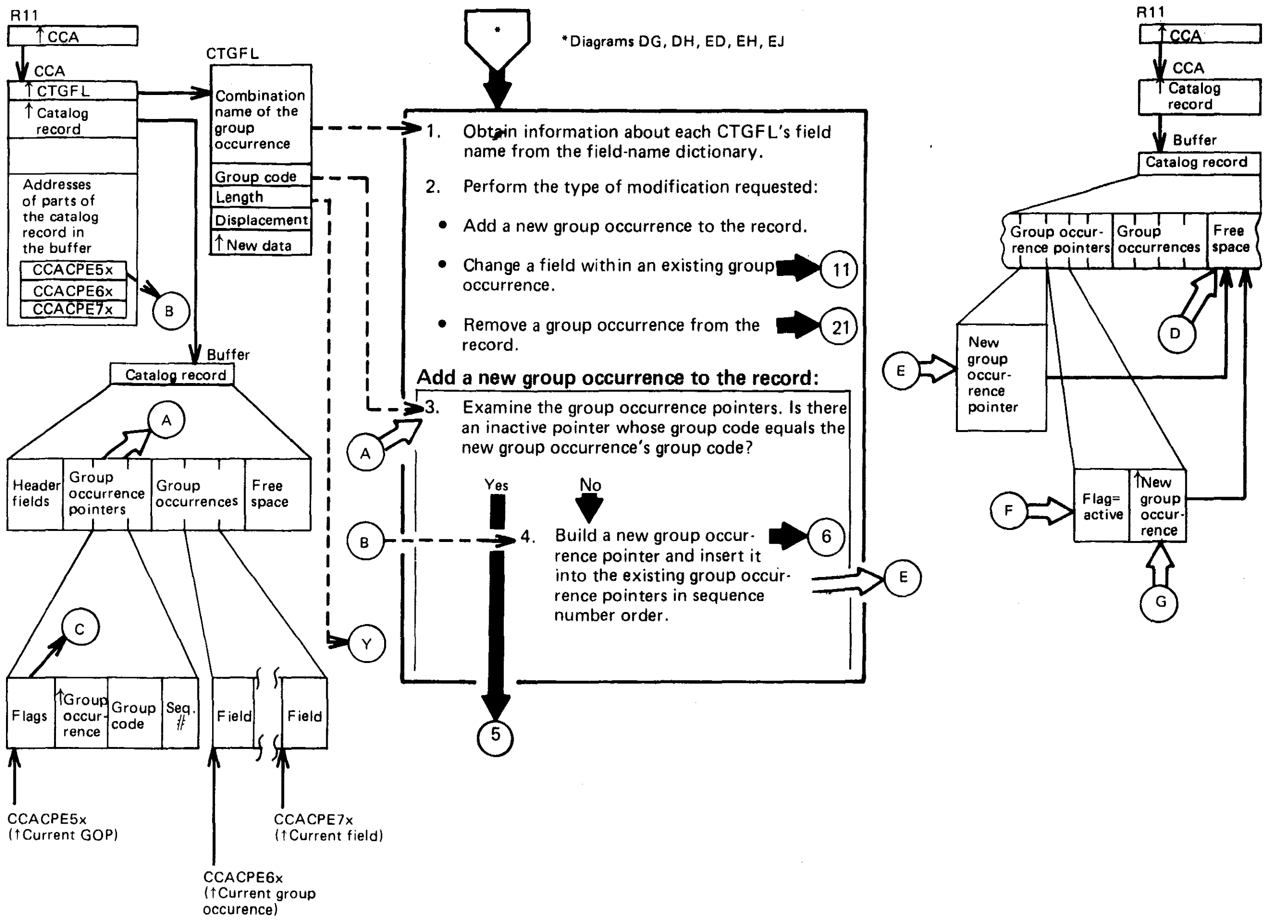


Notes for Diagram DK4

Description	Module	Procedure
11. The field's length is obtained from the CTGFL's dictionary entry (for a fixed-length field) or the first two bytes of the field (which are the length bytes of a variable length field). The field's address is the sum of the address of the group occurrence and the number of bytes from the beginning of the group occurrence.	IGG0CLBA	IGGPLVAL
12. A variable-length field might be partially contained in an extension record. If so, the field's length is greater than the number of bytes remaining in the record.	IGG0CLBA	IGGPGVAL
13. The caller's information requirements might be satisfied with the part of the field that is currently available. If not, the caller (a CM routine) returns to IGG0CLBA to obtain the value of the next part of the field from the extension record.	IGG0CLBA	IGGPLVAL

Description	Module	Procedure
The caller can move that part of the field currently in the buffer into a work area. If the rest of the field is required, the caller can return to IGG0CLBA to retrieve the extension record.		

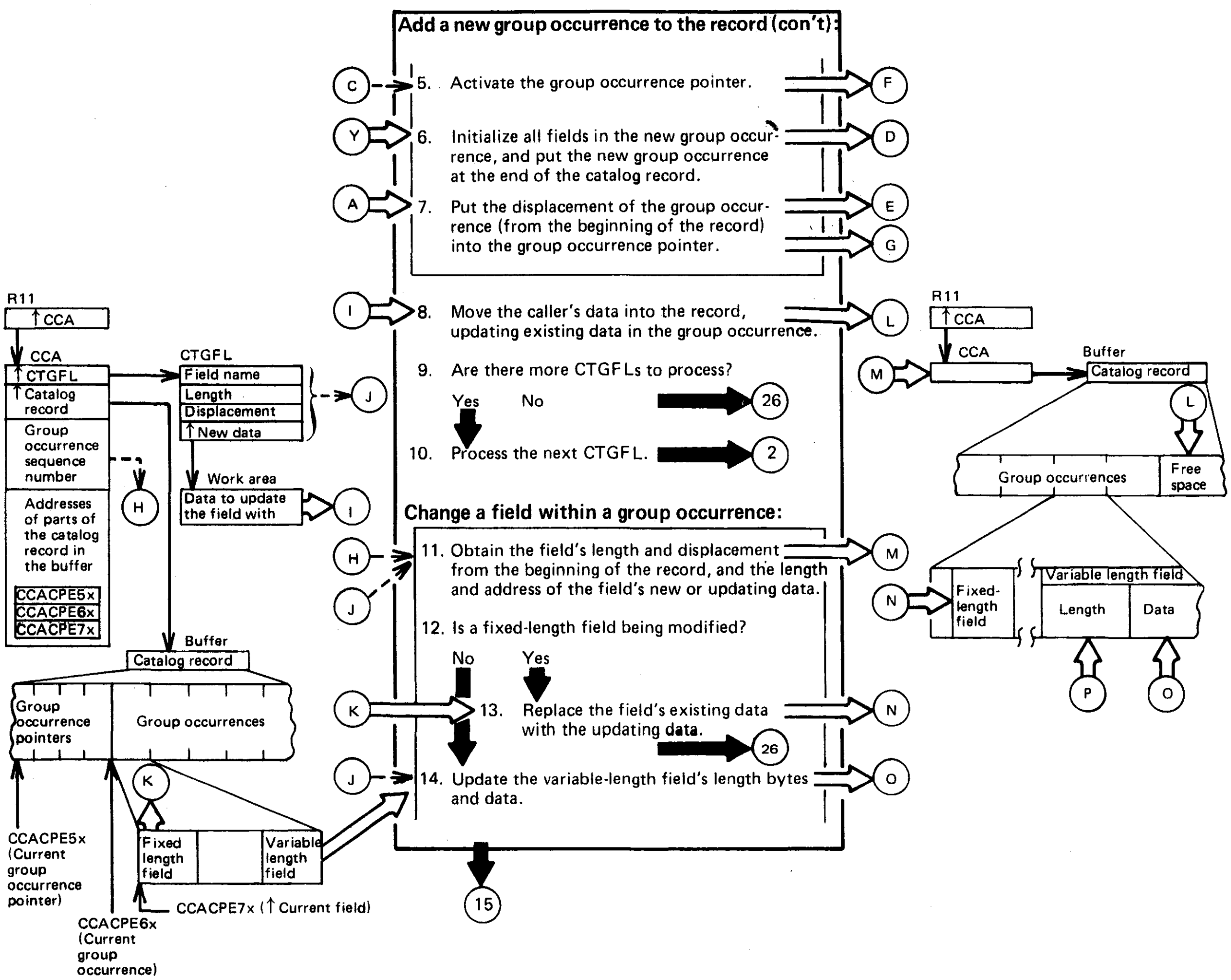
Diagram DL1. Modify a Catalog Record Field's Value



Notes for Diagram DL1

	Description	Module	Procedure
1.	The modify routine initializes each CTGFL with the dictionary entry associated with the CTGFL's field-name value.	IGG0CLAV IGG0CLAY	IGGPMOD IGGPSCNC
2.	The field parameter list (CTGFL) contains the field's name, type code, length, and displacement from the beginning of the record or group occurrence. If the field exists, it is either a header field (group code 0) or a field within a group occurrence. If the caller supplied modifying data and test conditions, the field is being altered. If the caller supplied modifying data and no test conditions, a group occurrence is to be added to the records. If the caller identified a group occurrence combination field name but didn't supply modifying data, the group occurrence is being deleted.	IGG0CLAV IGG0CLBA IGG0CLAV IGG0CLBT IGG0CLAV IGG0CLBT IGG0CLAV IGG0CLBT IGG0CLAV IGG0CLBA	IGGPMOD IGGPSFPL IGGPTSTS IGGPSFPL IGGPXLT2 IGGPSFPL IGGPXDGO IGGPSFPL IGGPXEL2 IGGPSFPL IGGPGREC
3.	A new group occurrence requires a group occurrence pointer (GOP).	IGG0CLAV IGG0CLAW	IGGPSFPL IGGPADGO
4.	If a new GOP is built, it is put into the catalog record at the end of its group of GOPs. The GOPs are grouped by group code and, within the group-code group, are ordered by sequence number. If the new GOP causes the catalog record to overflow, an extension record is obtained from the catalog's free control intervals. All group occurrences in the original record are put into the extension record. The GOPs displacement value (in the original record) is replaced with the control interval number of the extension record. In addition, a GOP is built and put into the extension record for each group occurrence in the extension record. The GOP in the extension record contains the displacement from the beginning of the record to its group occurrence. If the new GOP causes the catalog record to overflow and the catalog record contains only GOPs, an extension record is obtained to contain the new GOP. The original record's extension field contains the control interval number of the extension record.	IGG0CLBA IGG0CLAW IGG0CLAG IGG0CLAW IGG0CLBA	IGGPGREC IGGPADGO IGGPAXCI IGGPADGO IGGPGREL IGGPGREC

Diagram DL2. Modify a Catalog Record Field's Value



Notes for Diagram DL2

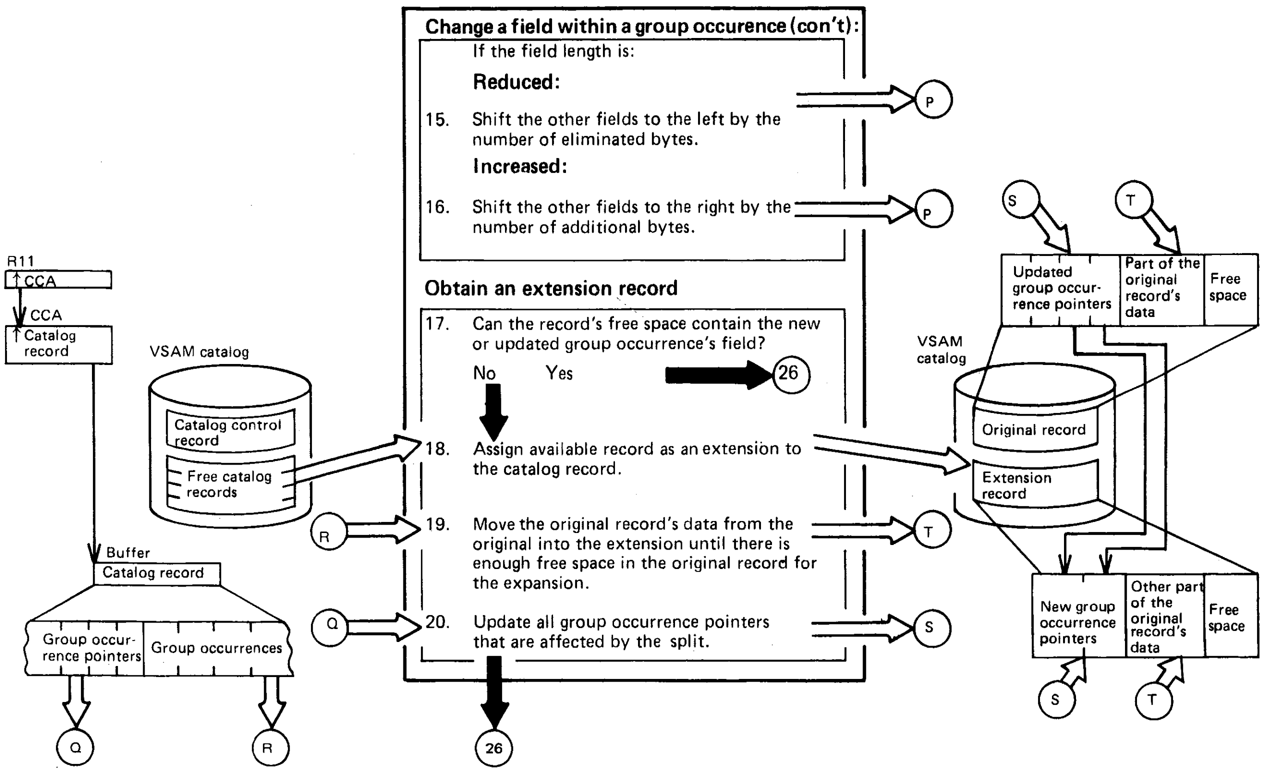
	Description	Module	Procedure
5.	This routine activates the GOP by setting its inactive flag off.	IGG0CLAW IGG0CLAG IGG0CLAW IGG0CLBA IGG0CLAW IGG0CLAX IGG0CLAW	IGGPAGOP IGGPAXCI IGGPAGOP IGGPGREC IGGPAGOP IGGPMGO IGGPAGOP IGGPIGOP
6.	The new group occurrence contains fixed-length fields and, possibly, variable-length fields. If the new group occurrence causes the record to overflow, an extension record is obtained to contain the new group occurrence.	IGG0CLAW	IGGPADGO
8.	Replace the initial field values (from step 6) with the caller-supplied values addressed by the CCA.	IGG0CLAW	IGGPADGO IGGPMVGO
9.	If there are no more CTGFLs to process, IGGPSFPL calls IGGPPREC to write each updated catalog record into the catalog.	IGG0CLAV IGG0CLAW	IGGPSFPL IGGPPREC
11.	The CCACPE7x field contains the field's address. The CTGFL contains the address and length of the data with which to update the field.	IGG0CLAX IGG0CLBA	IGGPALT2 IGGPGVAL
12.	The CTGFL flags field (from the catalog field name directory) specifies field type.	IGG0CLAX	IGGPALT2
13.	The CTGFL contains the length and address of the updating data. The data is in the caller's work area.	IGG0CLAX	IGGPALT2

Diagram DL3. Modify a Catalog Record Field's Value

Notes for Diagram DL2

Description	Module	Procedure
14. The CTGFL flags field (from the catalog field name directory) specifies field type. If the length of the data to update the field within the CTGFL isn't equal to the field's length in the CCA, the variable-length field's length is either increased or decreased, causing a corresponding reduction or increase in the catalog record's amount of free space.	IGG0CLAX	IGGPALT2
		IGGPMVAR
		IGGPMBGO
		IGGPMVAR
	IGG0CLBA	IGGPGVAL
	IGG0CLAX	IGGPMVAR
	IGG0CLAG	IGGPAXCI
	IGG0CLAX	IGGPMVAR
	IGG0CLAW	IGGPGNEX
	IGG0CLAX	IGGPMVAR
	IGG0CLBA	IGGPGREC
Note: If the catalog is recoverable, the call to IGG0CLAG or IGG0CLEG will result in a CRA update unless bit CCARPUT (the update inhibit bit) is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.	IGG0CLAX	IGGPMVAR
		IGGPMGO
		IGGPMVAR
	IGG0CLBW	IGGPDEIN
		IGGPRISE
	IGG0CLAG	IGGPAXCI
	IGG0CLBW	IGGPRISE
	IGG0CLBA	IGGPGVAL
	IGG0CLBW	IGGPRISE
	IGG0CLAX	IGGPSHNK
	IGG0CLBW	IGGPRISE
	IGG0CLAW	IGGPPREC
	IGG0CLBW	IGGPDEIN
		IGGPSINK
	IGG0CLAX	IGGPSHNK
	IGG0CLBW	IGGPSINK
	IGG0CLAX	IGGPEXPD
	IGG0CLBW	IGGPDEIN
		IGGPSNK2
	IGG0CLAG	IGGPAXCI
	IGG0CLBW	IGGPSNK2
		IGGPDOWN
	IGG0CLAG	IGGPAXCI
	IGG0CLBW	IGGPDOWN
	IGG0CLBA	IGGPGVAL
	IGG0CLBW	IGGPDOWN
	IGG0CLAW	IGGPGNEX
	IGG0CLBW	IGGPDOWN
	IGG0CLAX	IGGPSHNK
	IGG0CLBW	IGGPDOWN
	IGG0CLAX	IGGPEXPD
	IGG0CLBW	IGGPDOWN
	IGG0CLAW	IGGPPREC
	IGG0CLBW	IGGPSNK2
	IGG0CLAW	IGGPPREC
	IGG0CLBW	IGGPSNK2
	IGG0CLBA	IGGPGREC
	IGG0CLBW	IGGPSNK2
	IGG0CLEG	IGGPPDE
	IGG0CLBW	IGGPDEIN
	IGG0CLBA	IGGPGVAL
If the modification occurs in a base catalog record, the affected group occurrence is moved from the base catalog record into an extension record. If the available space pointer doesn't have an extension control interval number, an extension control interval number is assigned to the caller.	IGG0CLAX	IGGPMBGO
	IGG0CLAG	IGGPAXCI
	IGG0CLAX	IGGPMBGO
		IGGPMGO
The length of the group occurrence to be moved is computed. The group occurrence and the GOPs are moved and updated.	IGG0CLAX	IGGPMGO
	IGG0CLAV	IGGPSGOP
	IGG0CLAX	IGGPMGO
	IGG0CLAW	IGGPIGOP
The variable-length field's length bytes are replaced with the length of the data with which to update the field (in the CTGFL).	IGG0CLAX	IGGPMGO
		IGGPCGO
		IGGPMGO
		IGGPDGO
		IGGPMGO
		IGGPDGOP

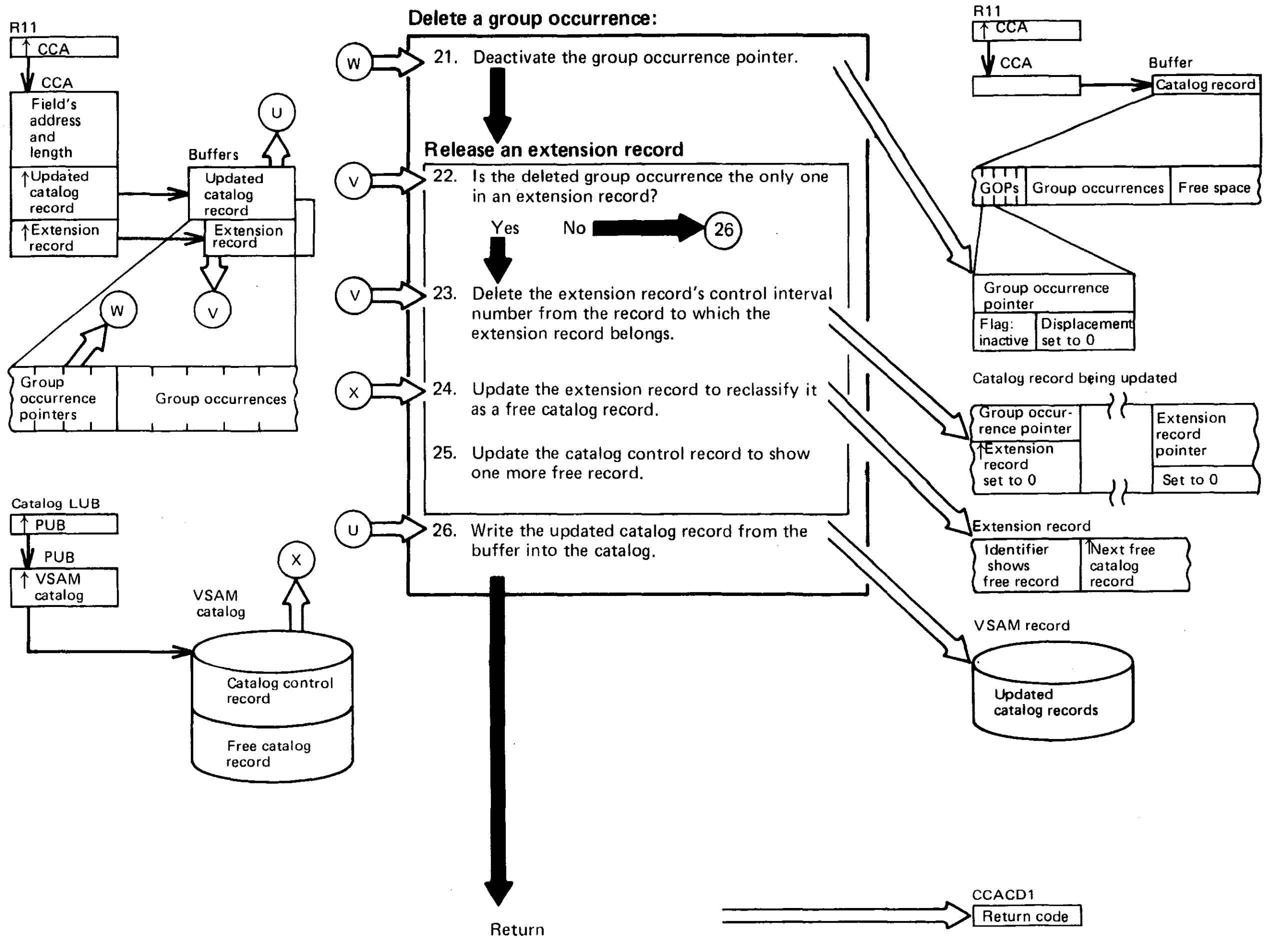
Diagram DL4. Modify a Catalog Record Field's Value



Notes for Diagram DL4

	Description	Module	Procedure
15.	The eliminated bytes at the end of the record are added to the record's free space.	IGG0CLAX	IGGPMVAR IGGPSHNK
16.	The additional bytes are obtained by reducing the record's free space. If the increased length causes the catalog record to overflow, an extension record is obtained. The original record's data is split so that part remains in the original record and part is moved into the extension record. Each associated GOP is updated to show the new position of its group occurrence.	IGG0CLAX	IGGPMVAR IGGPEXPD
18.	The catalog control record (CCR) contains pointers to two chains: one of unassigned records (CIs) and one of free (F type) records. The unassigned records are used first, down to a minimum "reserve" of 4 (needed for catalog extension), after which the requests are filled from the free record chain.	IGG0CLAX	IGGPALT2
19.	The group occurrence data is split so that part remains in the original record and part is moved into the extension record.	IGG0CLAX	IGGPALT2
20.	Each GOP is updated to show the new position of its group occurrence.	IGG0CLAX	IGGPALT2

Diagram DL5. Modify a Catalog Record Field's Value



Notes for Diagram DL 5

	Description	Module	Procedure
21.	The GOP's inactive flag is set on, thereby deactivating it. The GOP's type code and sequence number fields are unchanged. The displacement field is set to 0.	IGG0CLAV	IGGPDEL2
22.	If an extension record contains only deleted data, the record is reclaimed by CM as a free control interval.	IGG0CLAV	IGGPDEL2
23-25.	The CCR contains the count of free control intervals available to the catalog and the control interval number of the first free control interval. All free control intervals are chained together. When a control interval is added to the free control interval chain, CM puts the CCR's free control interval number into the new free control interval and puts the new free control interval's control-interval number into the CCR. The CCR's free control-interval count is increased by 1.	IGG0CLAV IGG0CLAX IGG0CLAV IGG0CLBA IGG0CLAV IGG0CLAW IGG0CLAV IGG0CLEG IGG0CLAV IGG0CLAX	IGGPDEL2 IGGPDGO IGGPDEL2 IGGPGREC IGGPDEL2 IGGPPREC IGGPDEL2 IGGPPDE IGGPDEL2 IGGPDGOP

Description	Module	Procedure
If a variable-length field within a group occurrence spans several physical records, multiple extension records must be deleted. Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT ("update inhibit") is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.		

Diagram DM1. Release Extents

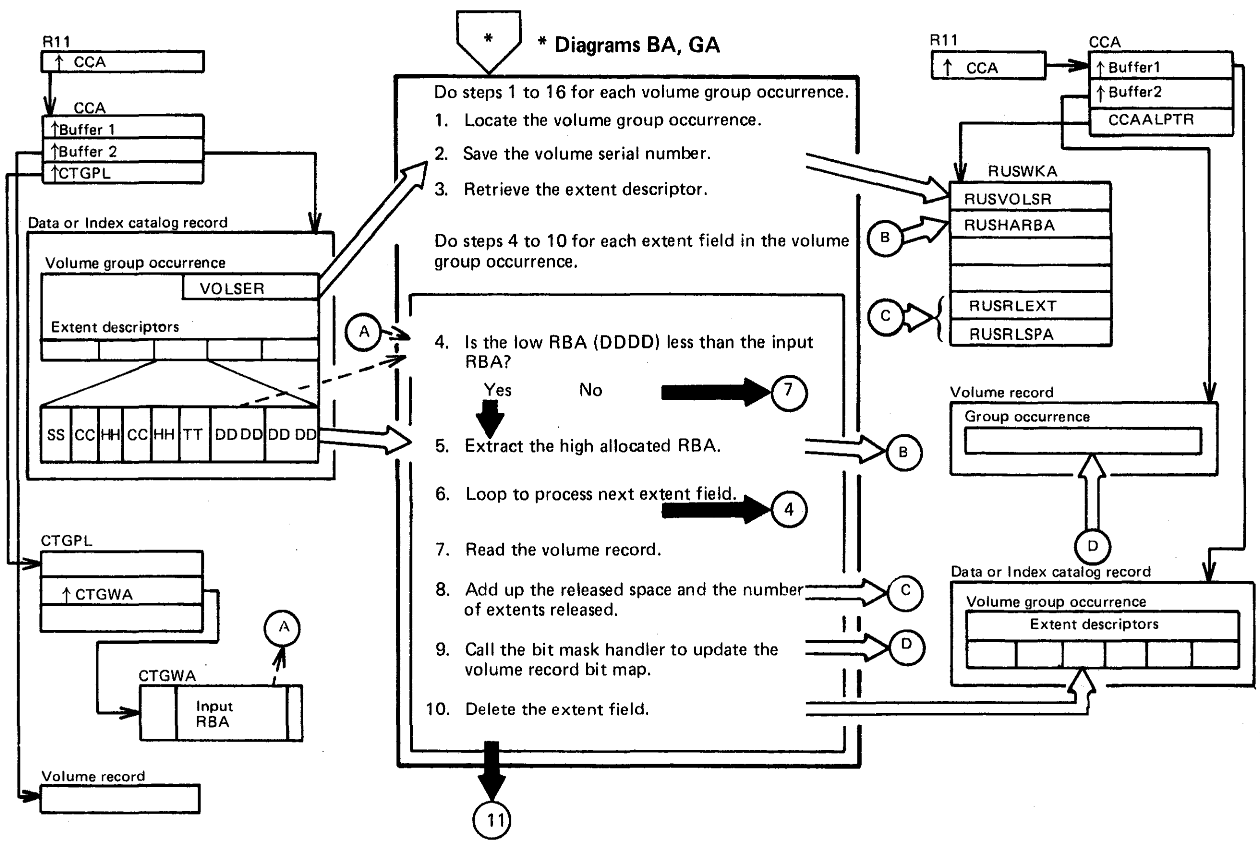


Diagram DM2. Release Extents

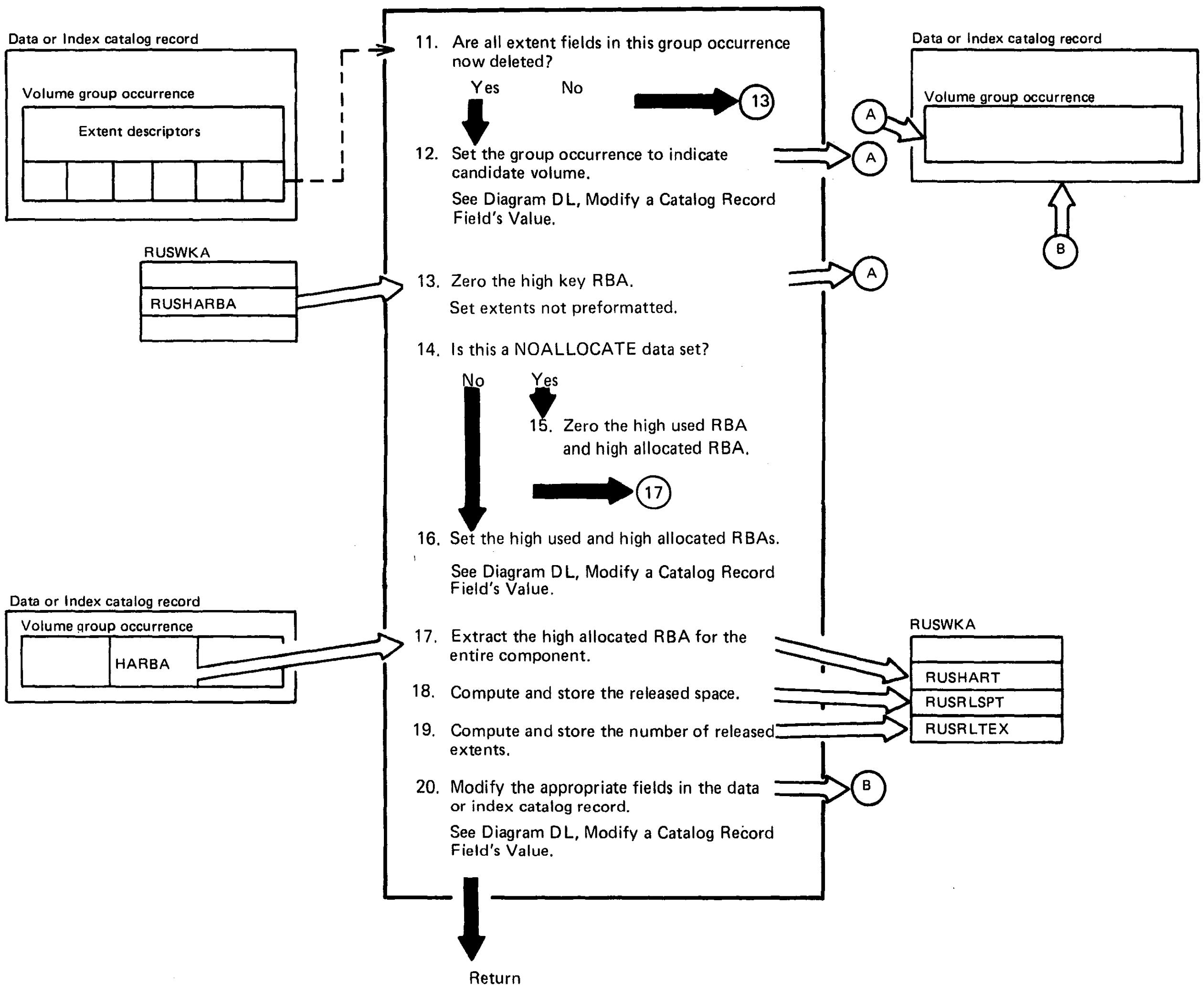


Diagram EA1. Catalog Management Services Contents

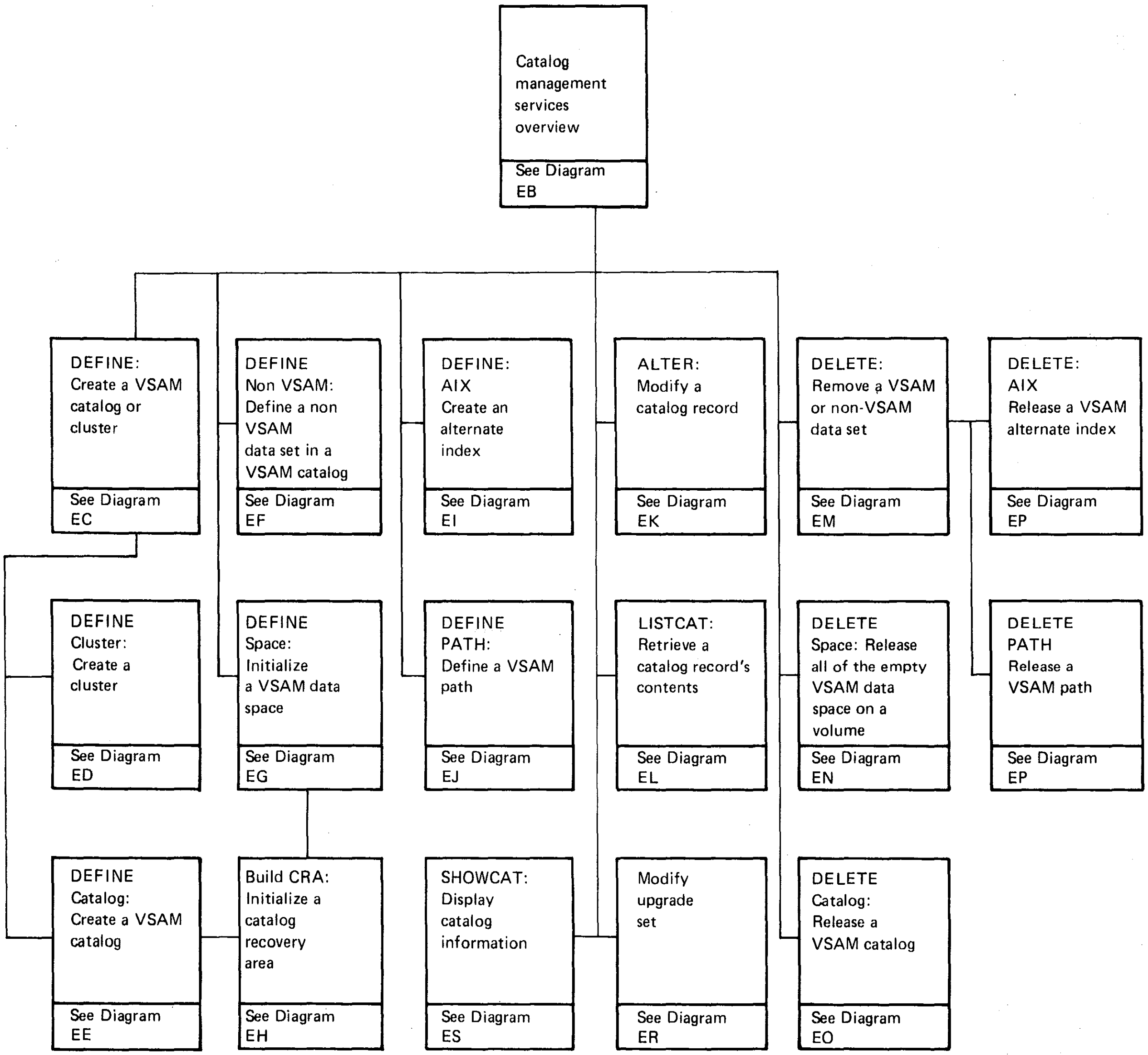
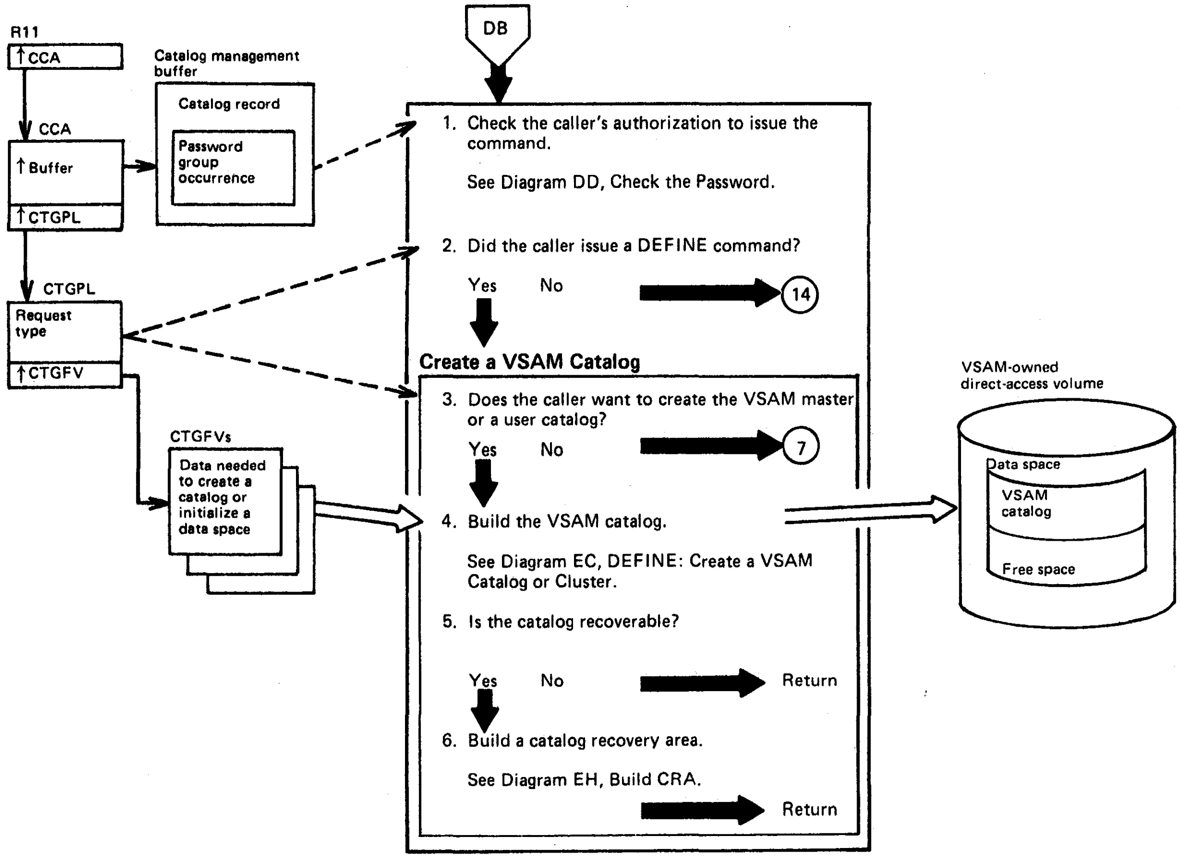


Diagram EB1. Catalog Management Services Overview



Notes for Diagram EB1

2. When using the DEFINE command with DEDICATE ORIGIN or default ORIGIN, module IGGOCLFD Dedicated VSAM Volume is called.

Diagram EB2. Catalog Management Services Overview

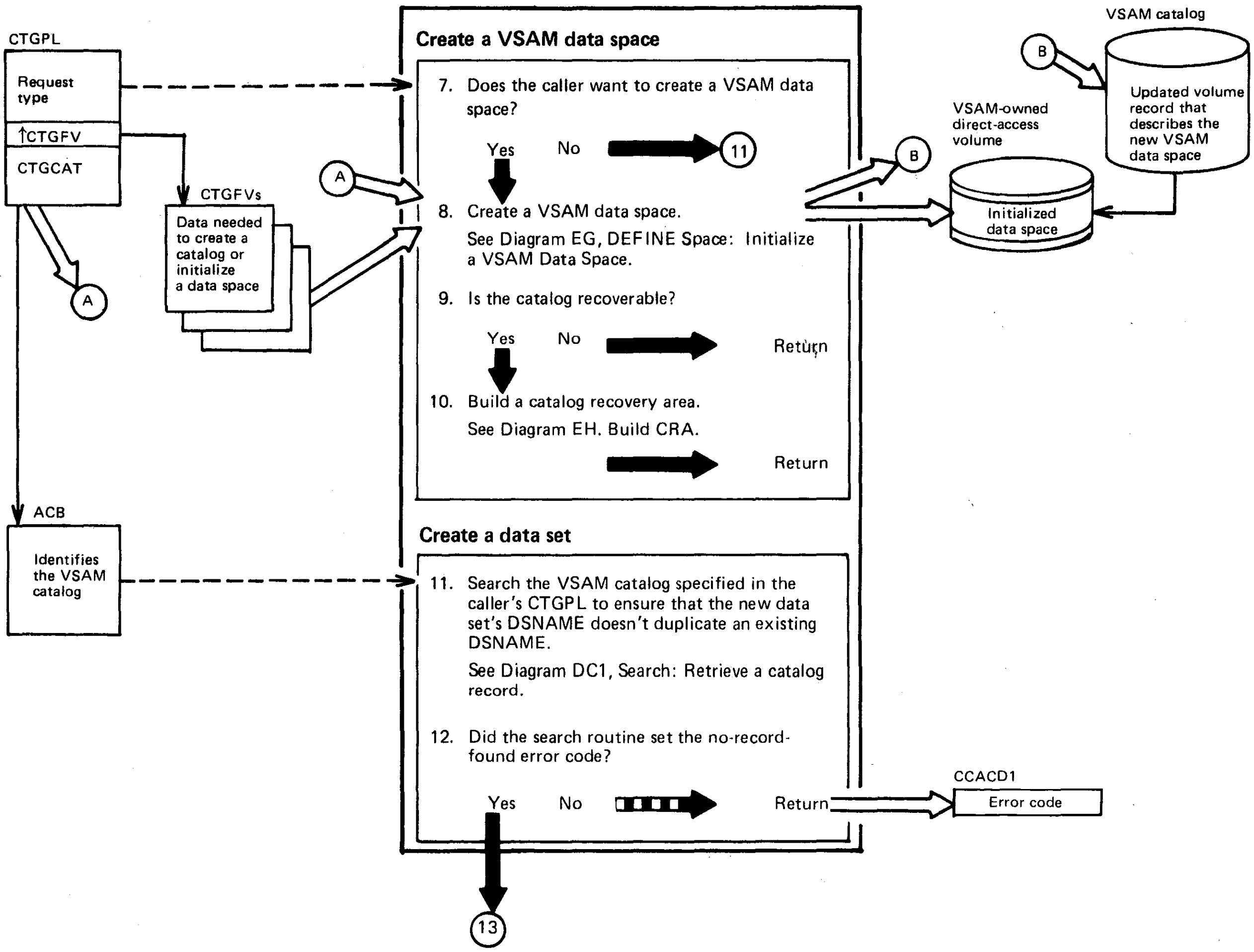


Diagram EB3. Catalog Management Services Overview

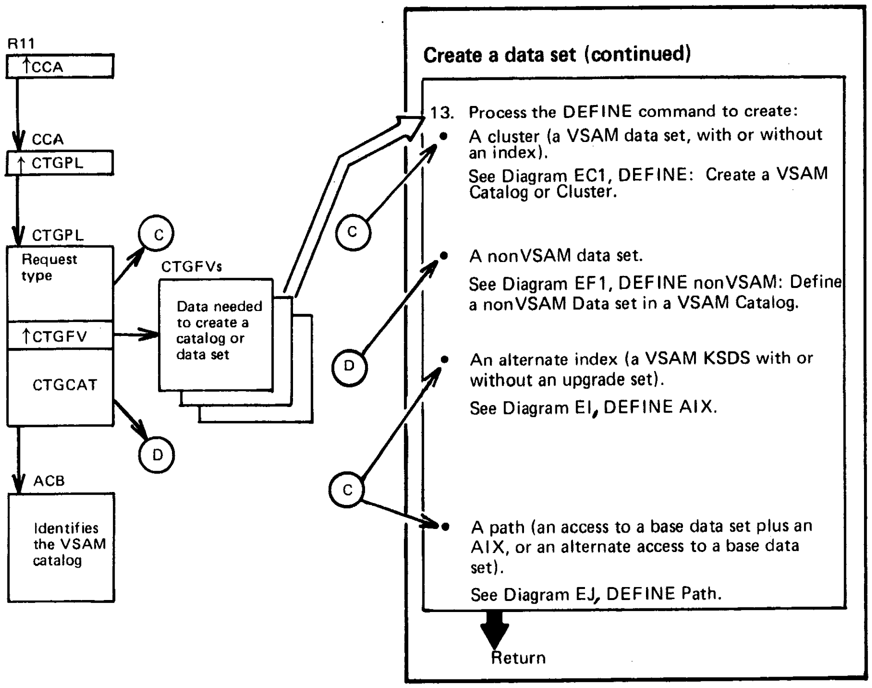
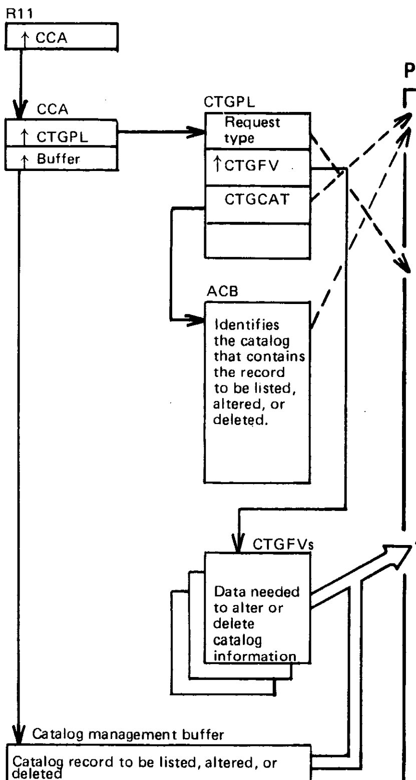


Diagram EB4. Catalog Management Services Overview



Process an ALTER, DELETE, or LISTCAT

14. Search the VSAM catalog specified in the caller's CTGPL, and retrieve the catalog record identified by the caller-supplied DSNAME.

    See Diagram DC1, Search: Retrieve a Catalog Record.

15. Process the command to:

    - Modify a catalog record.
      See Diagram EK, ALTER: Modify a Catalog Record.
    - Delete a cluster or nonVSAM data set.
      See Diagram EM, DELETE: Remove a catalog record.
    - Delete all empty data spaces on a volume.
      See Diagram EN, DELETE Space: Release all Empty Data Spaces on a Direct-access Volume.
    - Delete a VSAM catalog.
      See Diagram EO, DELETE Catalog: Release a VSAM Catalog.
    - Delete a VSAM alternate index.
      See Diagram EP, DELETE AIX.
    - Delete a VSAM path.
      See Diagram EP, DELETE Path.
    - Retrieve information from a catalog.
      See Diagram EL, LISTCAT: List the catalog contents.

Return

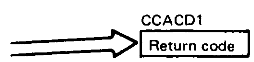
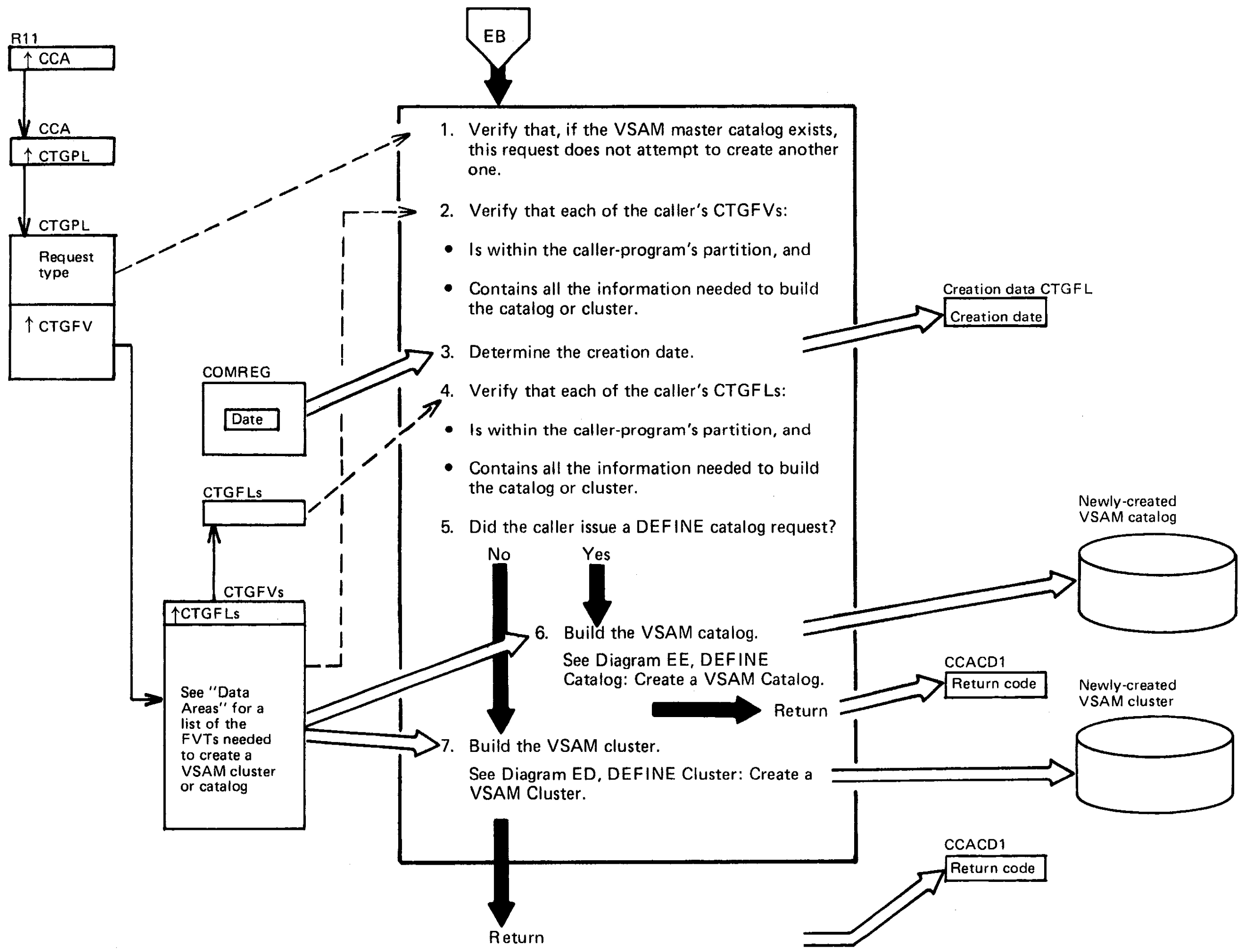


Diagram EC1. DEFINE: Create a VSAM Catalog or Cluster



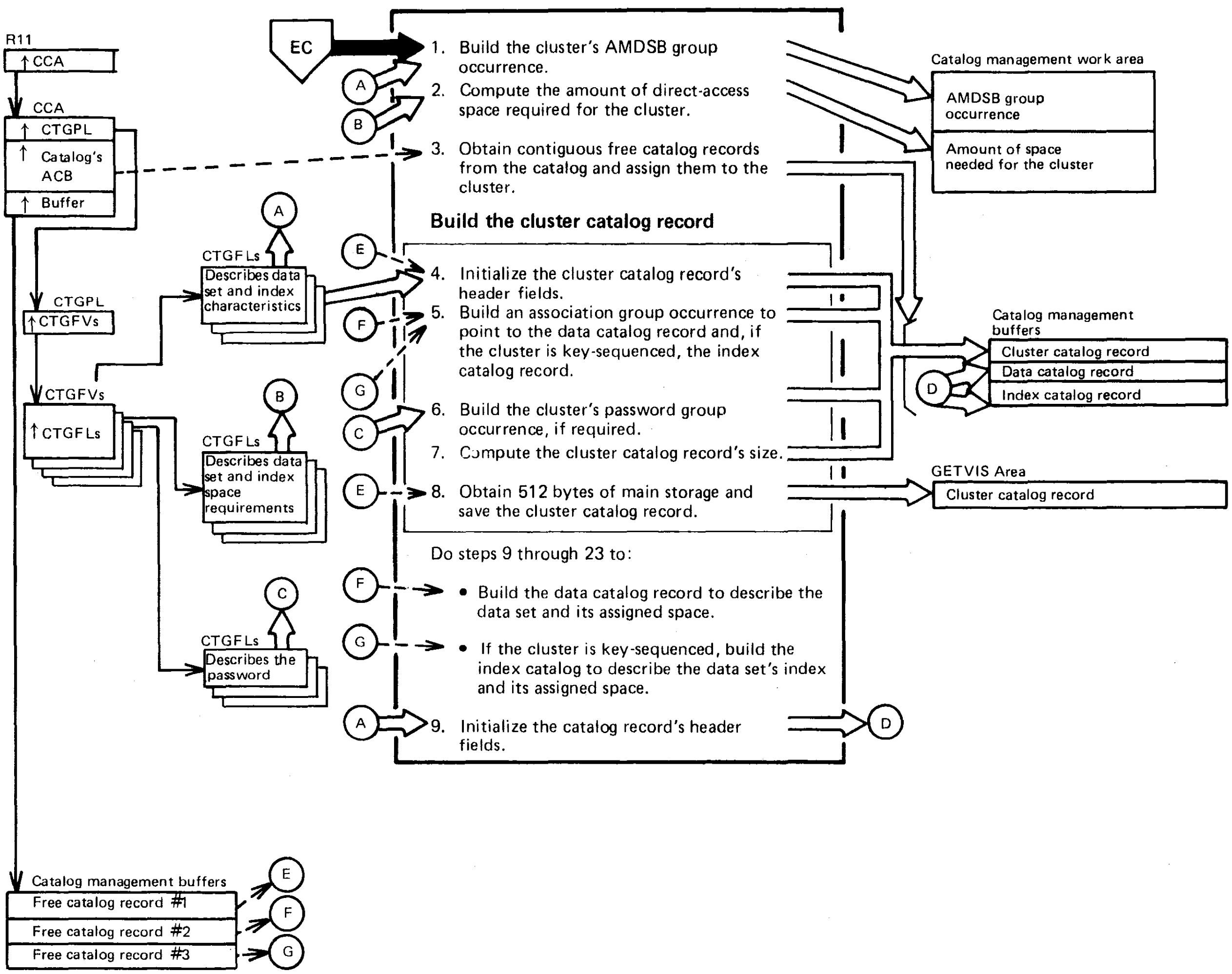
Notes for Diagram EC1

When the user issues the Access Method Services DEFINE command to create a catalog or a cluster, the CMS DEFINE initial processing routine insures that the caller has provided all the information necessary to create a catalog or a cluster. The field CCACD1 is checked for error codes. If no errors are found, CCACD1 will be 0. See the module prologue for IGG0CLAL for specific error conditions.

Description	Module	Procedure
1. Only one master catalog is permitted. An attempt to define a second master catalog will result in an error.	IGG0CLAL	IGGPDEF
2. CTGFVs and CTGFLs are checked by internal procedures in IGG0CLAL:		
Cluster CTGFV	IGG0CLAL	IGGPDCWC
Caller's work area		IGGPDCWC
Data CTGFV structure		IGGPDFSC
Index CTGFV structure		IGGPDFSC
Data DSNAME CTGPL		IGGPDDNP
Index DSNAME CTGPL		IGGPDDNP
Data space CTGFV		IGGPDBVC
Catalog space CTGPL	IGG0CLAL	IGGPDCSF
Key range CTGPL		IGGPDRPG
File CTGFV		IGGPDEF
3. Get the current date from COMREG and insert it in the creation date in the CTGFL.	IGG0CLAL IGG0CLEG	IGGPDEDE IGGPGET

Description	Module	Procedure
4. Calculate the amount of work area used or needed. The password and owner-ID are checked in the CTGFL.	IGG0CLAL	IGGPDDEP IGGPDSTY IGGPDEF
Space parameter CTGFLs Buffer-size CTGFLs Average-record size CTGFLs	IGG0CLBX	IGGPDSPF IGGPDBSF IGGPDALR
6.	IGG0CLAN IGG0CLAP	IGGPDSCB IGGPDCDA
7.	IGG0CLAN IGG0CLAS IGG0CLAN IGG0CLAJ	IGGPDCSB IGGPDRDA IGGPDSCB IGGPDEFC IGGPDSCB IGGPDBDI
If a volume list must be generated, call IGG0CLFB. The CTGFLs are then checked and completed by IGG0CLAN.	IGG0CLFB IGG0CLAN	IGGPGVL IGGPDSCB

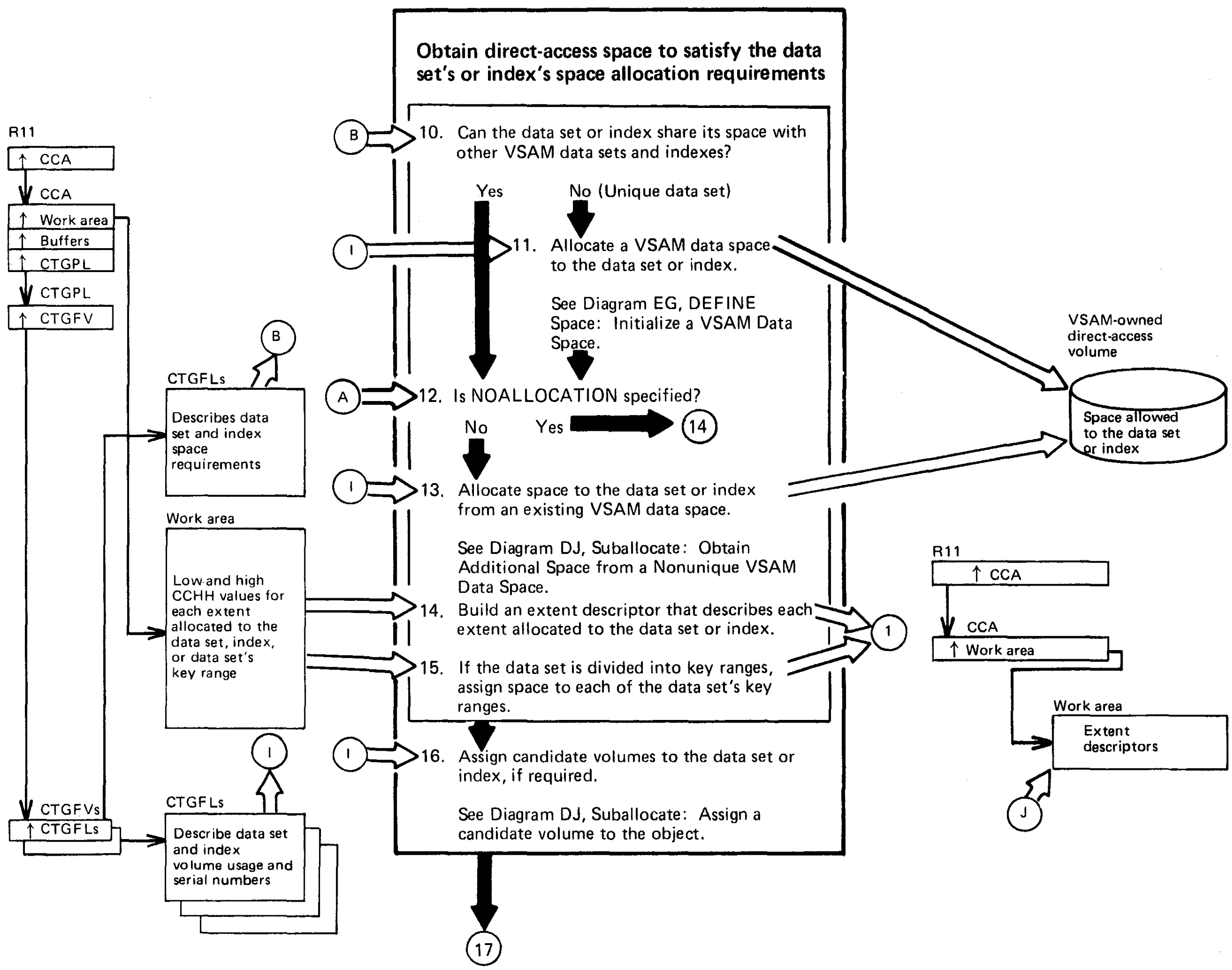
Diagram ED1. DEFINE Cluster: Create a VSAM Cluster



Notes for Diagram ED1

	Description	Module	Procedure
1.	The AMDSB contains the cluster's statistics and fixed characteristics. Each time the cluster is opened, the AMDSB is retrieved from the data catalog record. When the cluster is closed, the AMDSB is updated and rewritten into the data catalog record.	IGG0CLAN	IGGPDSCB IGGPDRDA
2.	The field vector table (CTGFV) contains addresses of buffer-size and record-length field parameter lists (CTGFLs). This data is used to determine the data set's control-interval and control-area size. If the data set is key-sequenced, other buffer-size and record-length CTGFLs determine the index's control-interval and control-area size. If the key-sequenced data set is divided into key ranges, the size of each key range is determined. If no errors are detected, then control is returned to IGG0CLAN.	IGG0CLBX IKQVDTPE IGG0CLBX IGG0CLEG IGG0CLAN	IGGPDSPC IGGPDDCE IGGPDDCE IGGPGET IGGPDRDA
	IGG0CLBY is called to perform space parameter calculations.	IGG0CLBY	IGGPDRSP
3.	A user's data set is described by a cluster catalog record, a data catalog record, and, if the data set is key-sequenced, an index catalog record.	IGG0CLAN	IGGPDCCE
	IGG0CLAG is called to get control-interval numbers assigned.	IGG0CLAG	IGGPAOCI
8.	For recoverable catalogs, the CRA volume has to be known. The cluster record is therefore saved until this volume is known.	IGG0CLAN	IGGPDCCE

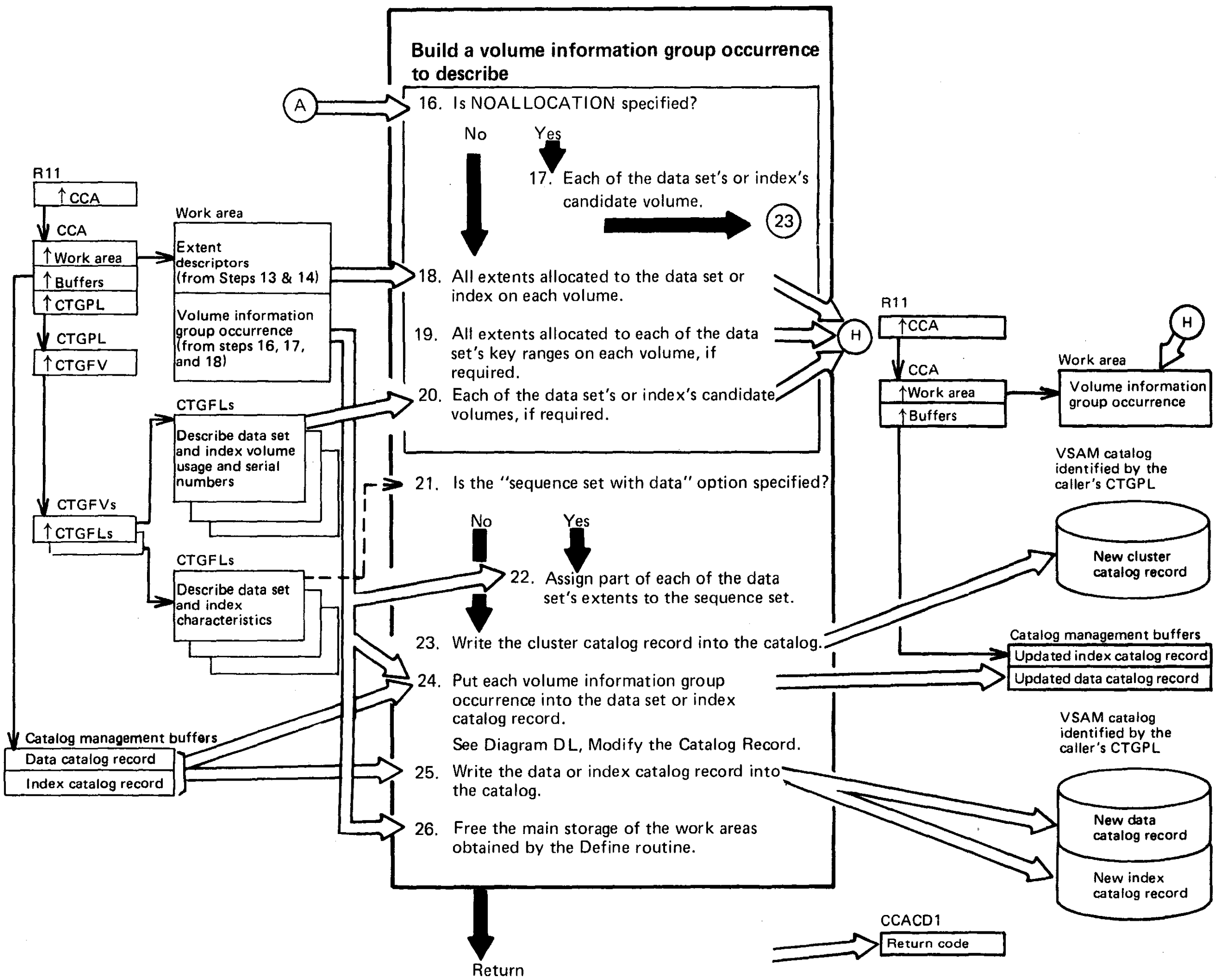
Diagram ED2. DEFINE Cluster: Create a VSAM Cluster



Notes for Diagram ED2

	Description	Module	Procedure
10.		IGG0CLAN IGG0CLAJ	IGGPDSCB IGGPDBDI IGGPDSPO
11.	A data space entry is added to the volume catalog record, and the data set's name is added to the volume catalog record's data set directory.	IGG0CLAJ IGG0CLAQ	IGGPDSPO IGGPDEFS
13.	The data set's name is added to the volume catalog record's data set directory, and the volume catalog record's data space entry is updated to show the cylinders and tracks allocated to the new data set or index.	IGG0CLAJ IGG0CLAR	IGGPDSPO IGGPSALL
14.		IGG0CLAJ	IGGPDSEX
15.	Each key range is assigned physical space and space allocation continues for each range.	IGG0CLAJ	IGGPDSPO
16.	A candidate volume is available to contain part of the cluster if more space is needed later. None of the candidate volume's space is allocated to the data set or index when the cluster is created.	IGG0CLAJ IGG0CLAR IGG0CLAJ IGG0CLAK	IGGPDCNV IGGPSALL IGGPDBDI IGGPDCMB

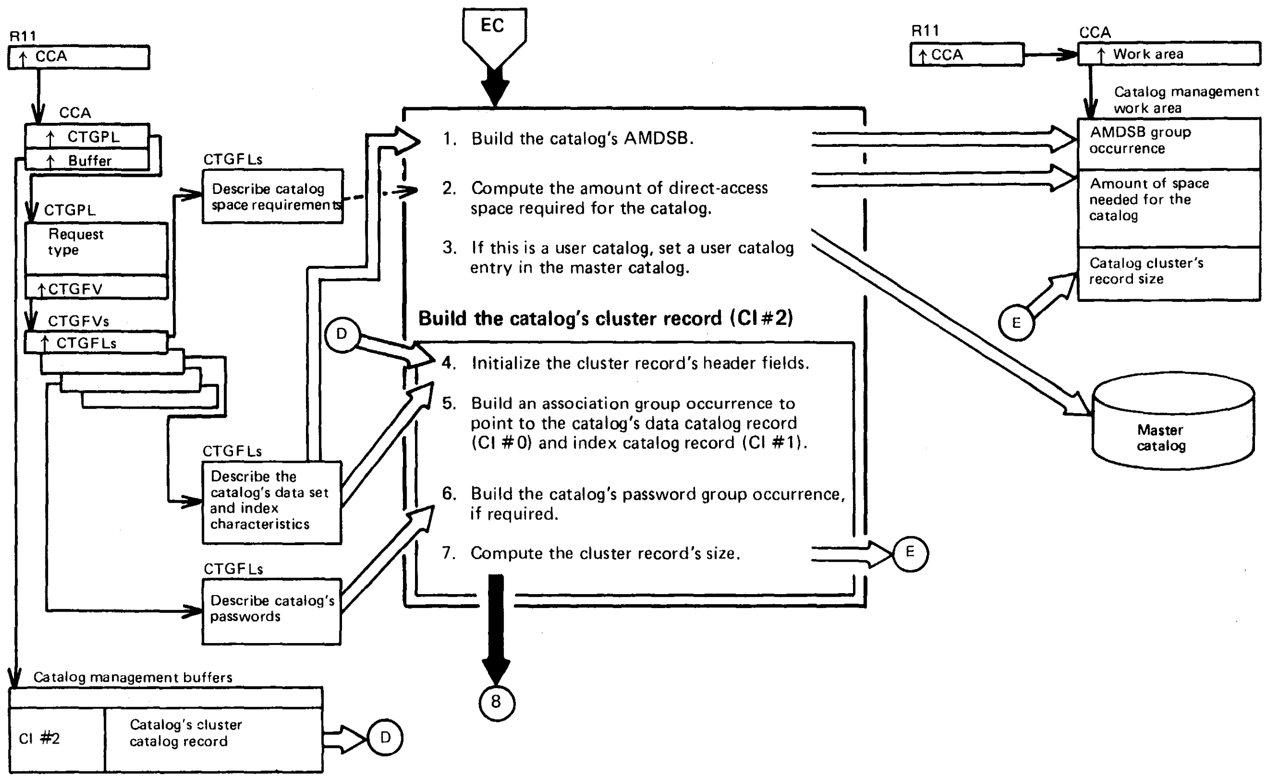
Diagram ED3. DEFINE Cluster: Create a VSAM Cluster



Notes for Diagram ED3

Description	Module	Procedure
16. If NOALLOCATION data set construct the volume information group occurance for all the candidate volumes of the data set.	IGGOCLAK	IGGPDCMB IGGPDBCV
18. Each volume that contains a part of the data set or index is described by a volume information group occurrence within the data or index catalog record.	IGGOCLAK	IGGPDBVO
19. If the data set is divided into key ranges, each key range's space on a volume is described in a separate volume information group occurrence. If the key range's space is on more than one volume, each volume that contains part of the key range's space is described in a separate volume information group occurrence.	IGGOCLAK	IGGPDRNG
20. Construct the volume information group occurrence for all candidate volumes of the data set.	IGGOCLAK	IGGPDBCV
22. If the "sequence set with data" (IMBED) option is specified, part of the data set's space is allocated to the index for sequence-set records. The low and high CCHH values in the index record's volume information group occurrence are those of the extent obtained for the data set. The low and high RBA values are for the sequence set and are relative to the index addresses.	IGGOCLAK IGGOCLAK	IGGPDSSP IGGPDBCV
23. The module calls the clear buffer routine to write the cluster record into the catalog.	IGGOCLCB IGGOCLEG	IGGPCLBF IGGPPAD
Note: If the catalog is recoverable, the call to IGGOCLEG will result in a CRA update unless bit CCARPUT ("update inhibit") is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.	IGGOCLCY IGGOCLAV	IGGPDMOP IGGPMOD
24. The modify routine (IGGOCLAV) inserts each volume information group occurrence into the record.	IGGOCLCY	IGGPDMOP
25. A CM routine writes the completed data or index catalog record into the VSAM catalog and frees the CM buffer that contains the record.	IGGOCLCY IGGOCLAV	IGGPDMOP IGGPMOD
26. Free all unneeded storage resources.	IGGOCLA8	IGGPDFRS

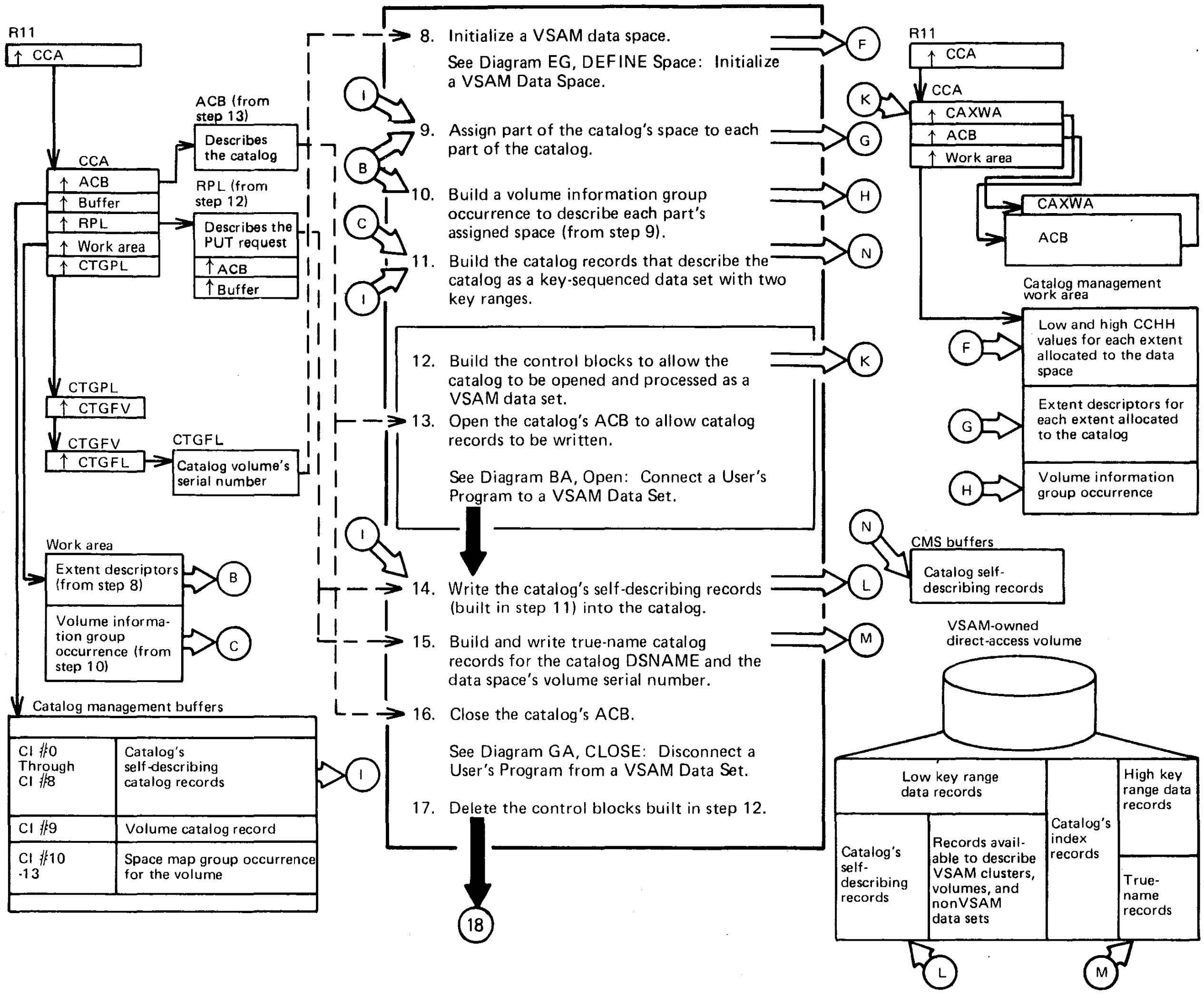
Diagram EE1. DEFINE Catalog: Create a VSAM Catalog



Notes for Diagram EE1

	Description	Module	Procedure
1.	The AMDSB contains the catalog's statistics and fixed characteristics. Each time the catalog is opened, the AMDSB is retrieved from the catalog's data catalog record (CI #0). When the catalog is closed, the AMDSB is updated and rewritten into the data catalog record.	IGG0CLAP IKQVDTPE	IGGPDCDA IGGPDCVS
2.	The field vector table (CTGFV) contains addresses of buffer-size and record-length CTGFLs. This data is used to determine the catalog's control-interval and control-area size, and the amount of space required for the catalog.	IGG0CLAP	IGGPDCSP
3.	User catalogs are internally identical to master catalogs. The only difference is that the master catalog contains an entry pointing to the user catalog.		
5.	The cluster catalog contains an associated data set group occurrence to locate the catalog's data catalog record (CI #0) and an associated index group occurrence to locate the catalog's index catalog record (CI #1).	IGG0CLAN	IGGPDCCE

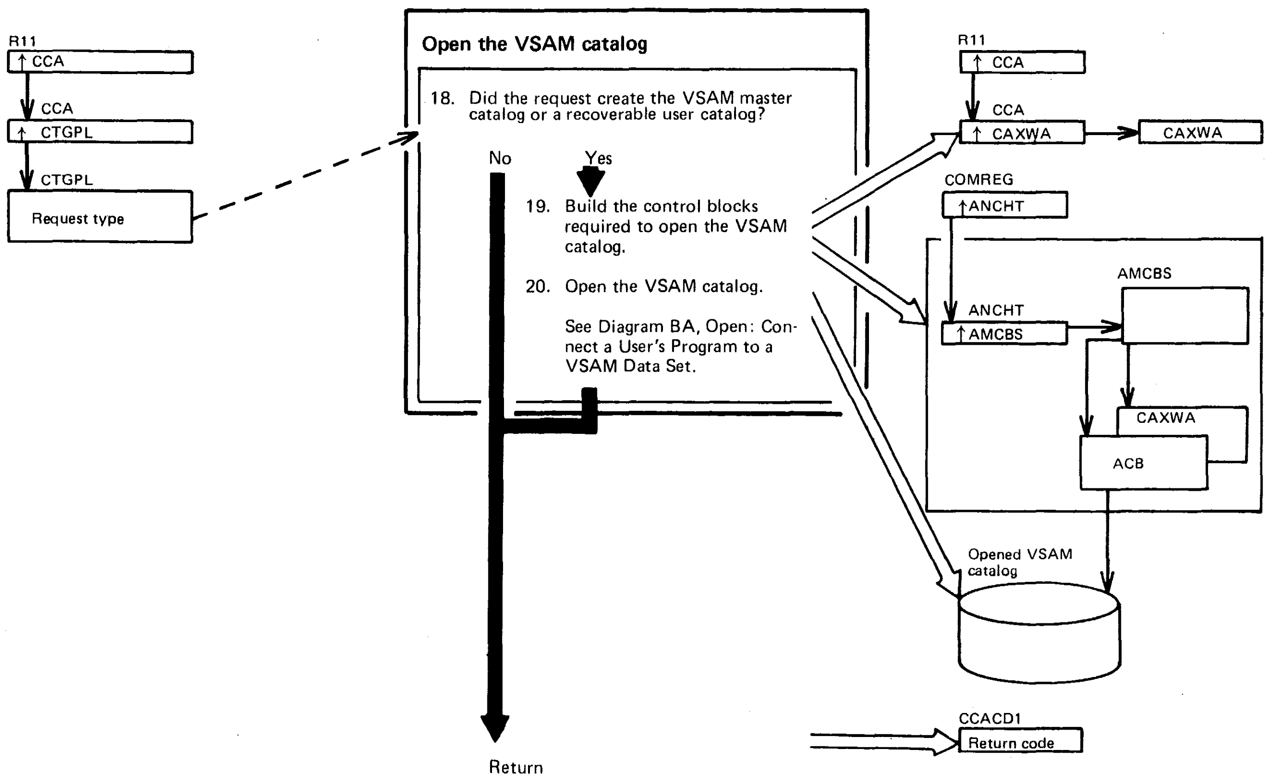
Diagram EE2. DEFINE Catalog: Create a VSAM Catalog



Notes for Diagram EE2

	Description	Module	Procedure
8.	The VSAM catalog is always built in a data space that can contain other VSAM data sets and indexes. A new data space is allocated to the VSAM routine, and the data space is assigned to the new data set.	IGG0CLAS IGG0CLAQ	IGGPDCSP IGGPDEFS
	A data set directory entry containing the data set's DSNAME is added to the volume catalog record.	IGG0CLAS IGG0CLAR	IGGPDCSP IGGPSALL
9.	The catalog might contain records that describe the user's VSAM data sets, the user's nonVSAM data sets, and direct-access volumes.	IGG0CLES	IGGPDCLD
10.		IGG0CLES	IGGPDCVO
	See "Data Areas" for details about the first ten records in the catalog. These records define the catalog as a key-sequenced VSAM data set, describe the space allocated to the catalog's data records, index records, and true-name records, describe the free space within the catalog, and describe the allocated tracks on the catalog volume.	IGG0CLAS IGG0CLAV	IGGPDCBE IGGPDCEB IGGPDCME IGGPMOD
12.		IGG0CLAE	IGGPDCOC IGGPDCCB
13.		$$BOPEN	
14.		IGG0CLAE IGG0CLCG	IGGPDCOC IGGPDCPR IGGPXIO
15.		IGG0CLAE	IGGPDCOC
16.		$$BCLOSE	
17.		IGG0CLAE	IGGPDCOC

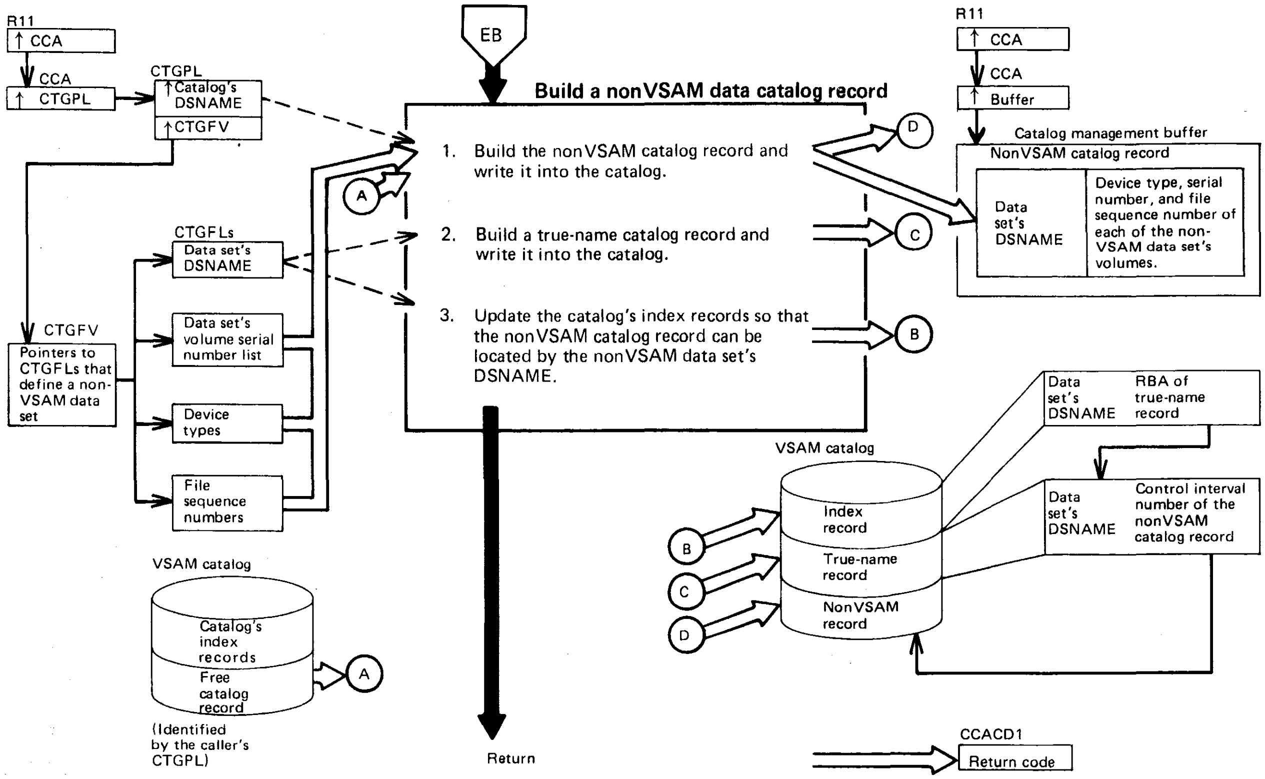
Diagram EE3. DEFINE Catalog: Create a VSAM Catalog



Notes for Diagram EE3

Description	Module	Procedure
19.	IGG0CLAD	IGGPMCO2
20.		IGGPDCOC

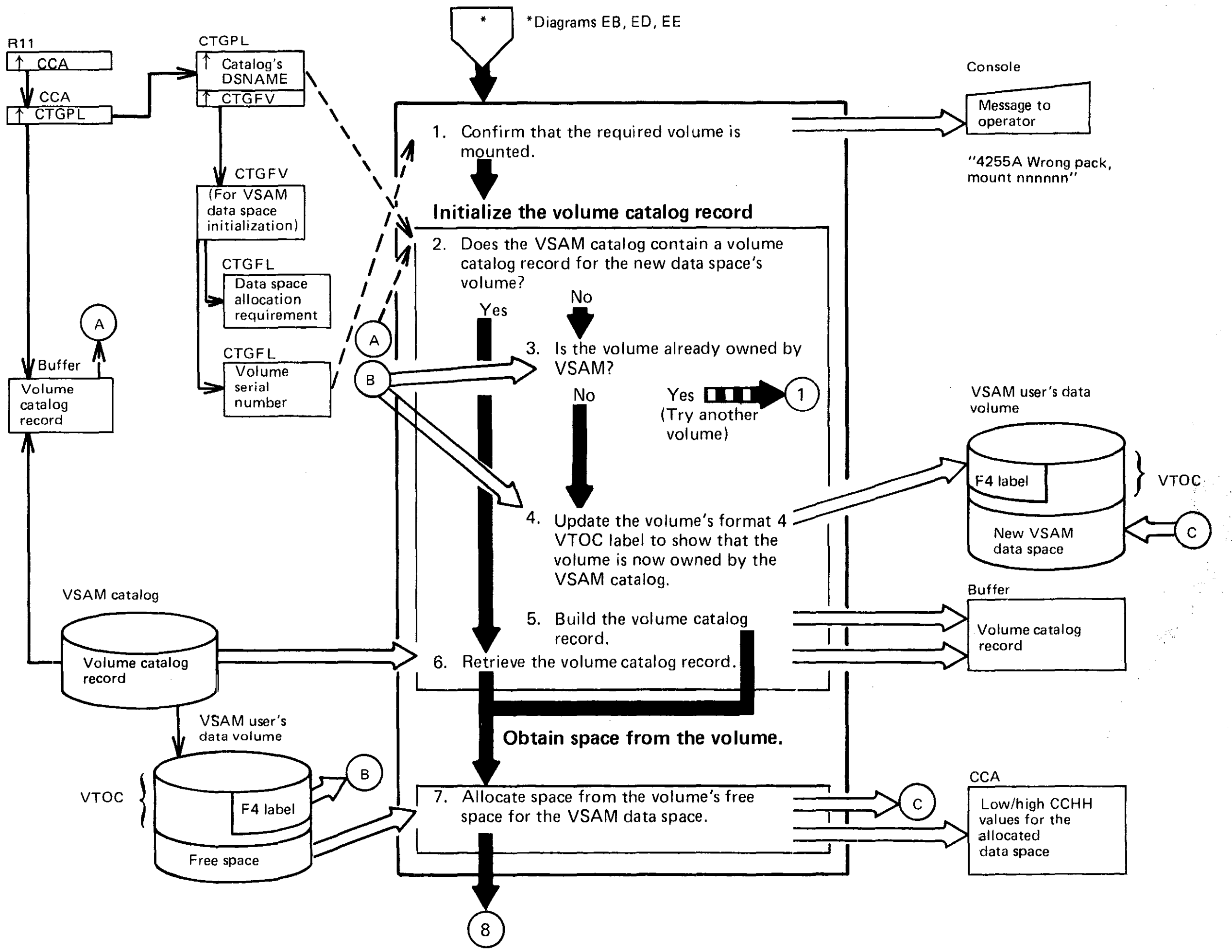
Diagram EF1. DEFINE NonVSAM: Define a NonVSAM Data Set in a VSAM Catalog



Notes for Diagram EF1

	Description	Module	Procedure
1.	The define nonVSAM routine transfers the nonVSAM catalog record from a CM buffer in storage to the VSAM catalog specified by the caller's DEFINE command.	IGG0CLBH	IGGPDEFA IGGPDAIN
	Control interval numbers are assigned for the nonVSAM data set.	IGG0CLBH IGG0CLAG	IGGPDAIN IGGPAOCI
	The 8-character device name is converted into a 4-character device code, using the device name table.	IGG0CLBH IKQDNT	IGGPDAIN IGGPDANL
	IGG0CLAV is called to add volume occurrences.	IGG0CLBH IGG0CLAV	IGGPDAVO IGGPMOD
	If an error occurs during the define process, IGG0CLAN is called to back out any CI numbers assigned and any entries put into the VSAM catalog.	IGG0CLBH IGG0CLAN	IGGPDAVO IGGPDEFA

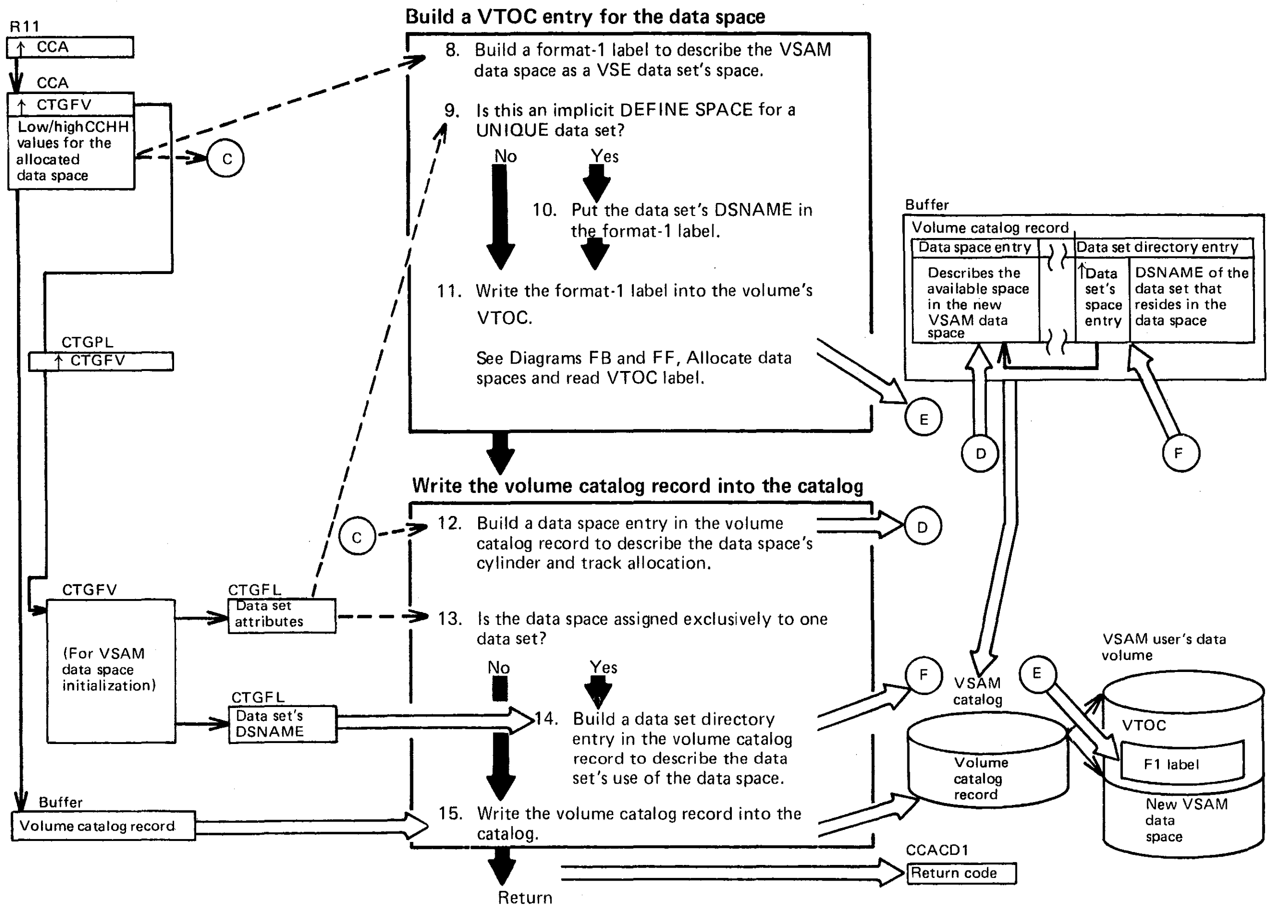
Diagram EG1. DEFINE Space: Initialize a VSAM Data Space



Notes for Diagram EG1

	Description	Module	Procedure
1.		IGG0CLAQ IGG0CLA6 IKQASNMT IKQVTC00	IGGPDEFS IGGPVMTV
2.	If a volume catalog record exists for the volume and if the volume already contains a VSAM data space, a data space group occurrence is added to the volume catalog record to describe the new VSAM data space.		
3.	If this volume is already owned by VSAM, error code decimal 148 is issued.	IGG0CLA6	IGGPF4PR
4.	If the volume is a candidate volume (one that is eligible to contain a VSAM data space, but doesn't as yet), the volume's format-4 VTOC label is updated to show that the VSAM catalog is now the volume owner.	IGG0CLAQ IGG0CLA6 IGG0CLBU	IGGPDEFS IGGPF4PR IGGPF4RD IGGPF4DQ IGGPF4WR
5.	See "Data Areas" for details about the volume catalog record.	IGG0CLA6	IGGPIVER
6.	The volume catalog record is identified by the volume's serial number.	IGG0CLAQ IKQVDTPE	IGGPDEFS

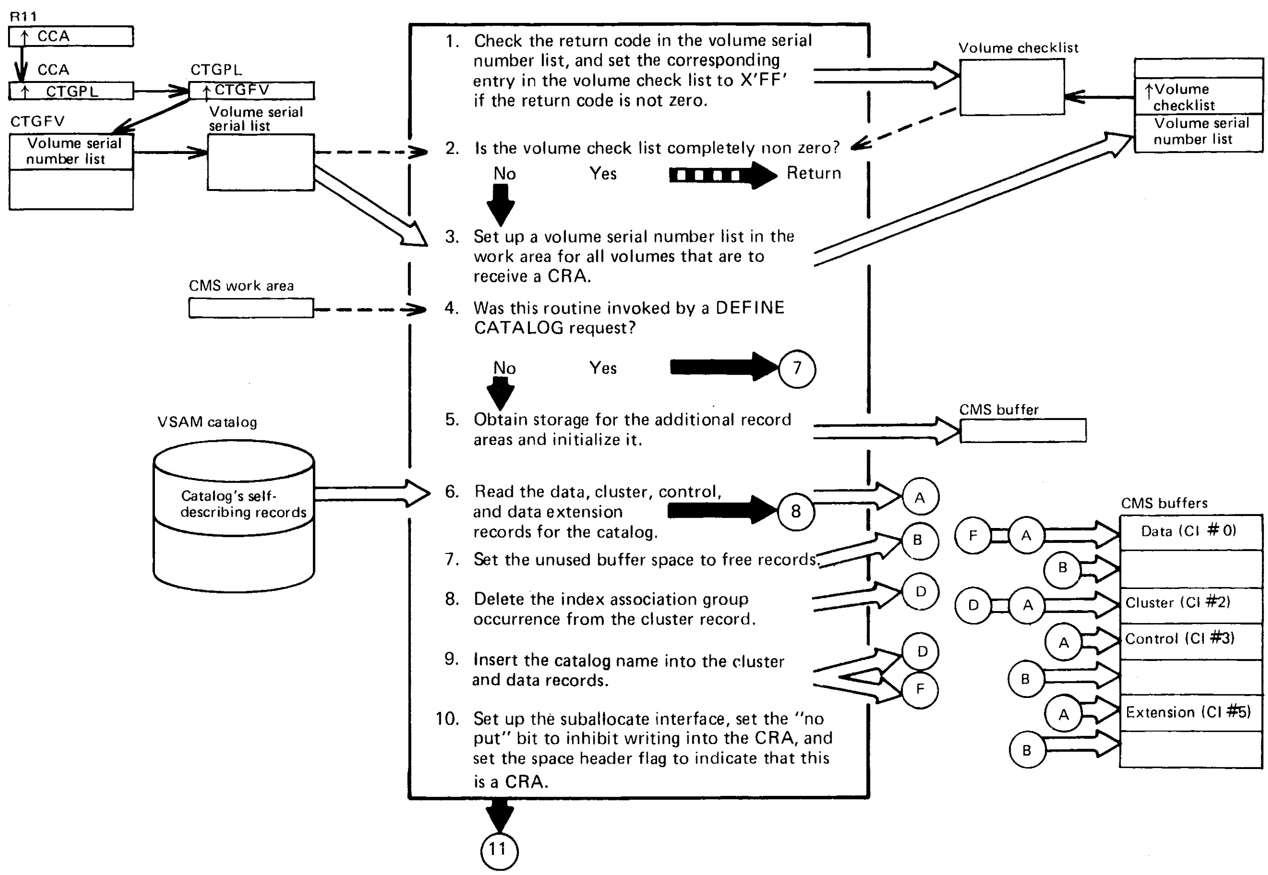
Diagram EG2. DEFINE Space: Initialize a VSAM Data Space



Notes for Diagram EG2

Description	Module	Procedure
10. A VSAM data space assigned exclusively to one data set is described by a format-1 label that contains the data set's DSNAME. If a data space can be assigned to more than one data set, its format-1 label contains a DSNAME generated by the define space routine.	IGG0CLAQ	IGGPDEFS
11. Prior to calling IKQALL00, IGG0CLAQ sets the address of the DADSM exit routine, IKQDXT (located in IGG0CLAQ), into the DADSM parameter list. This routine is called by DADSM during space allocation to place the Access Method Services-specified secondary space quantity and an expiration date into those fields of the format-1 label before it is written into the VTOC. The secondary space quantity is for OS/VS compatibility only. This information is not used by VSE.	IGG0CLEG IGG0CLAQ IKQALL00 IGG0CLAQ IGG0CLA6 IKQVTC00 IGG0CLA6 IKQRDS00 IGG0CLA6 IKQASNMT	IGGPPAD IGGPDEFS IGGPDEFS IGGPCOBT IGGPCOBT IGGPCOBT
Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.		
12. See "Data Areas" for details about the data space entry.	IGG0CLA6 IGG0CLAV	IGGPDFS2 IGGPCSHG IGGPCSDG IGGPDSMD IGGPMOD
14. The volume catalog record contains the identifier of each data set that resides in part or full on the volume.	IGG0CLA6	IGGPCDSD

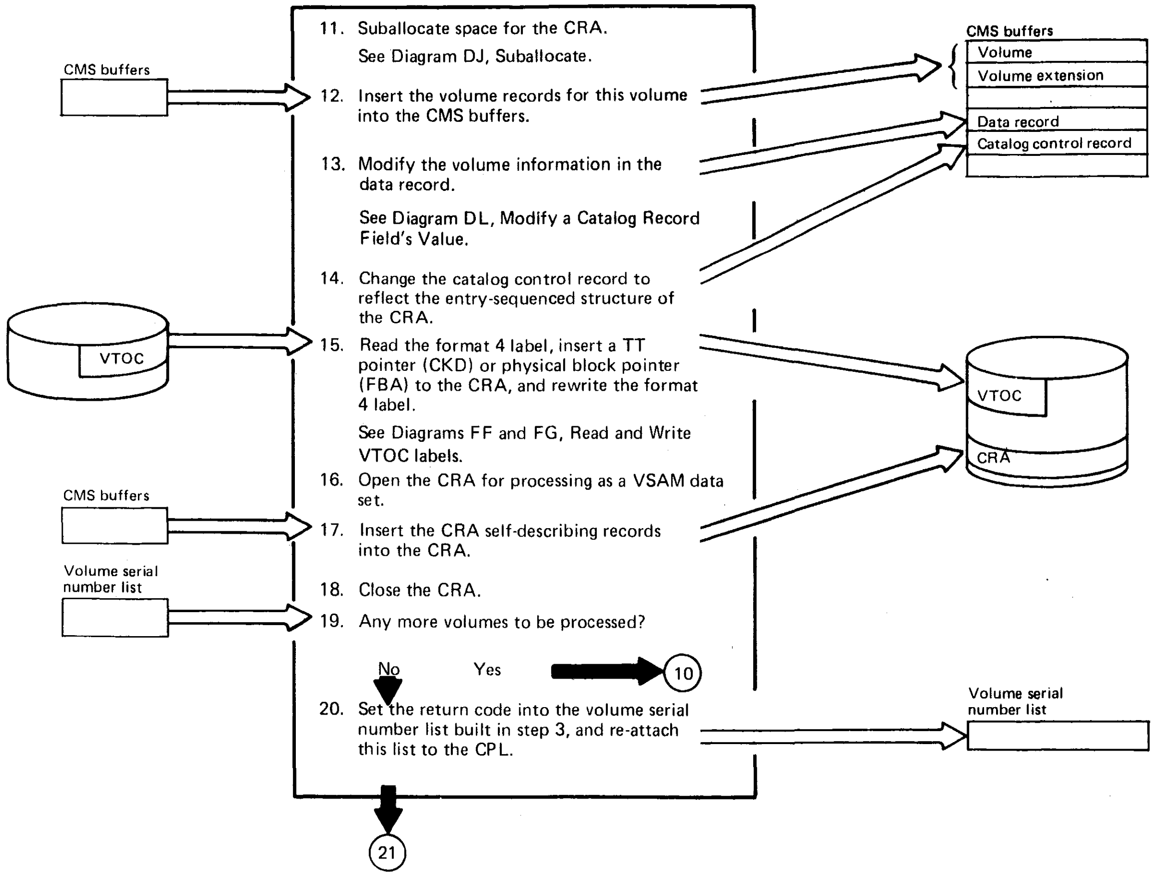
Diagram EH1. DEFINE CRA: Create a Catalog Recovery Area



Notes for Diagram EH1

Step	Module	Procedure
1-3.	IGG0CLCR	IGGPCADR
4-5.	IGG0CLCR	IGGPBCRA
6.	IGG0CLCR IGG0CLEG	IGGPBCRA IGGPGET
7.	IGG0CLCR	IGGPBCRA
8.	IGG0CLCR IGG0CLAV	IGGPCRDI IGGPMOD
9-10.	IGG0CLCS	IGGPCRVL

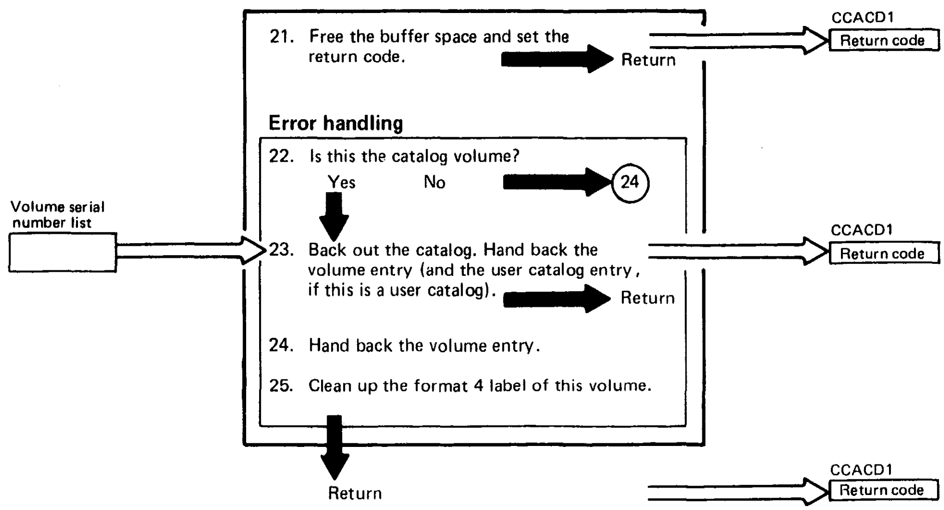
Diagram EH2. DEFINE CRA: Create a Catalog Recovery Area



Notes for Diagram EH2

Step	Module	Procedure
11.	IGG0CLAR	IGGPSALL
12.	IGG0CLCS IGG0CLEG IGG0CLCS	IGGPCRVL IGGPGET IGGPCRVL
13.	IGG0CLAS IGG0CLCS IGG0CLAV	IGGPDCRC IGGPCRVL IGGPMOD
14.	IGG0CLCS	IGGPCRVL
15.	IGG0CLA6 IGG0CLBU IGG0CLCS IGG0CLBU	IGGPVMTV IGGPF4RD IGGPCRVL IGGPF4WD
16.	IGG0CLCS IGG0CLCO	IGGPPRDS IGGPCRAO
17.	IGG0CLCG IGG0CLCS	IGGPPXIO IGGPPRDS
18.	$$BCLOSE	——
19.	IGG0CLCS	IGGPCRVL
20.	IGG0CLCR	IGGPBCRA

Diagram EH3. DEFINE CRA: Create a Catalog Recovery Area



Notes for Diagram EH3

Step	Module	Procedure
21.	IGG0CLCR	IGGPCADR
22.	IGG0CLCR	IGGPBCRA
23.	IGG0CLCR	IGGPBCRA
	IGG0CLAE	IGGPDCBO
	$$BCLOSE	–
	IGG0CLCO	IGGPSCAX
	$$BCLOSE	–
24.	IGG0CLCR	IGGPBCRA
	IGG0CLCR	IGGPCRBO
	IGG0CLEG	IGGPGET
	IGG0CLEG	IGGPPDE
25.	IGG0CLBU	IGGPF4RD
	IGG0CLBU	IGGPF4WR
	IGG0CLAF	IGGPSKSP

Note: Control is passed to the error handling routine (Step 22) whenever an error is detected.

Diagram EI1. DEFINE AIX: Create an Alternate Index

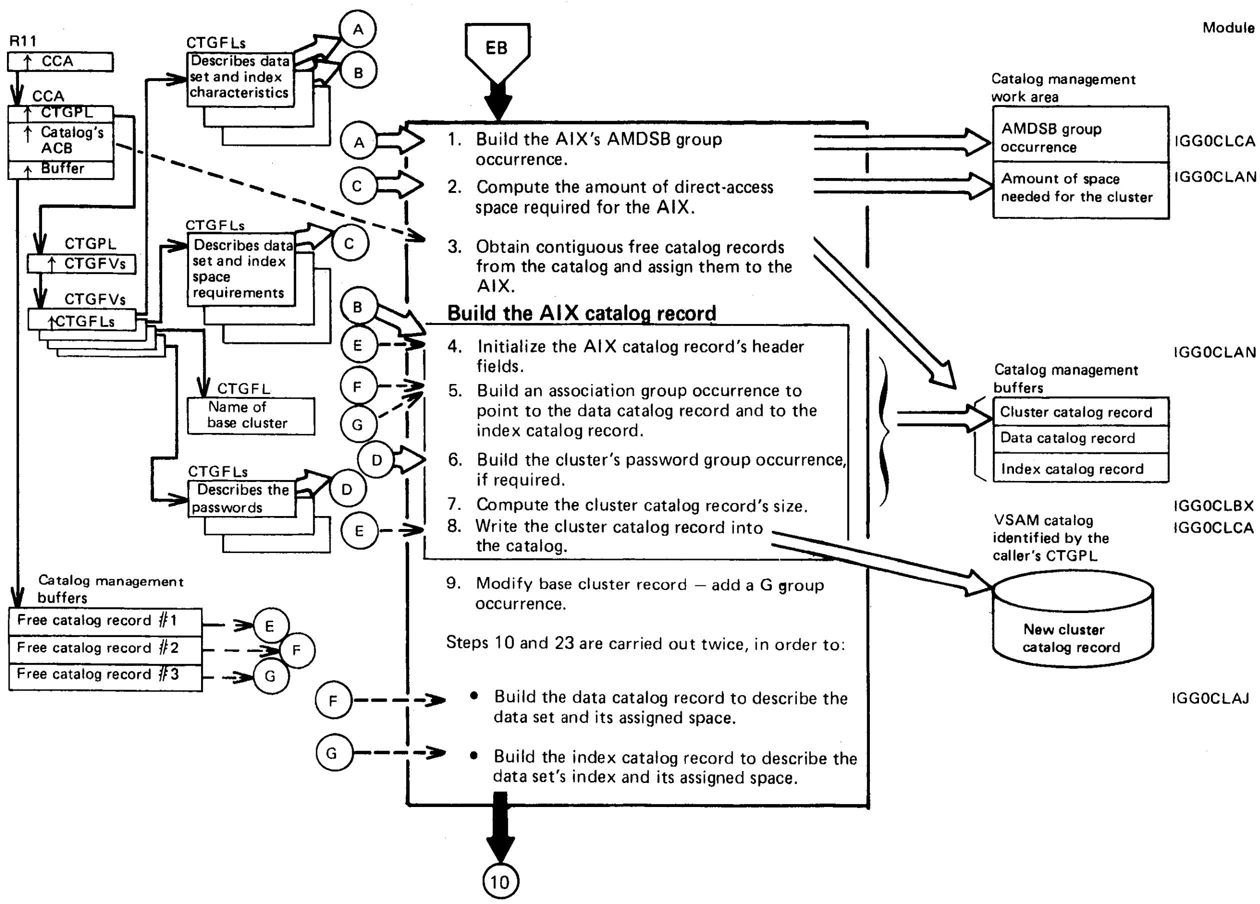


Diagram EI2. DEFINE AIX: Create an Alternate Index

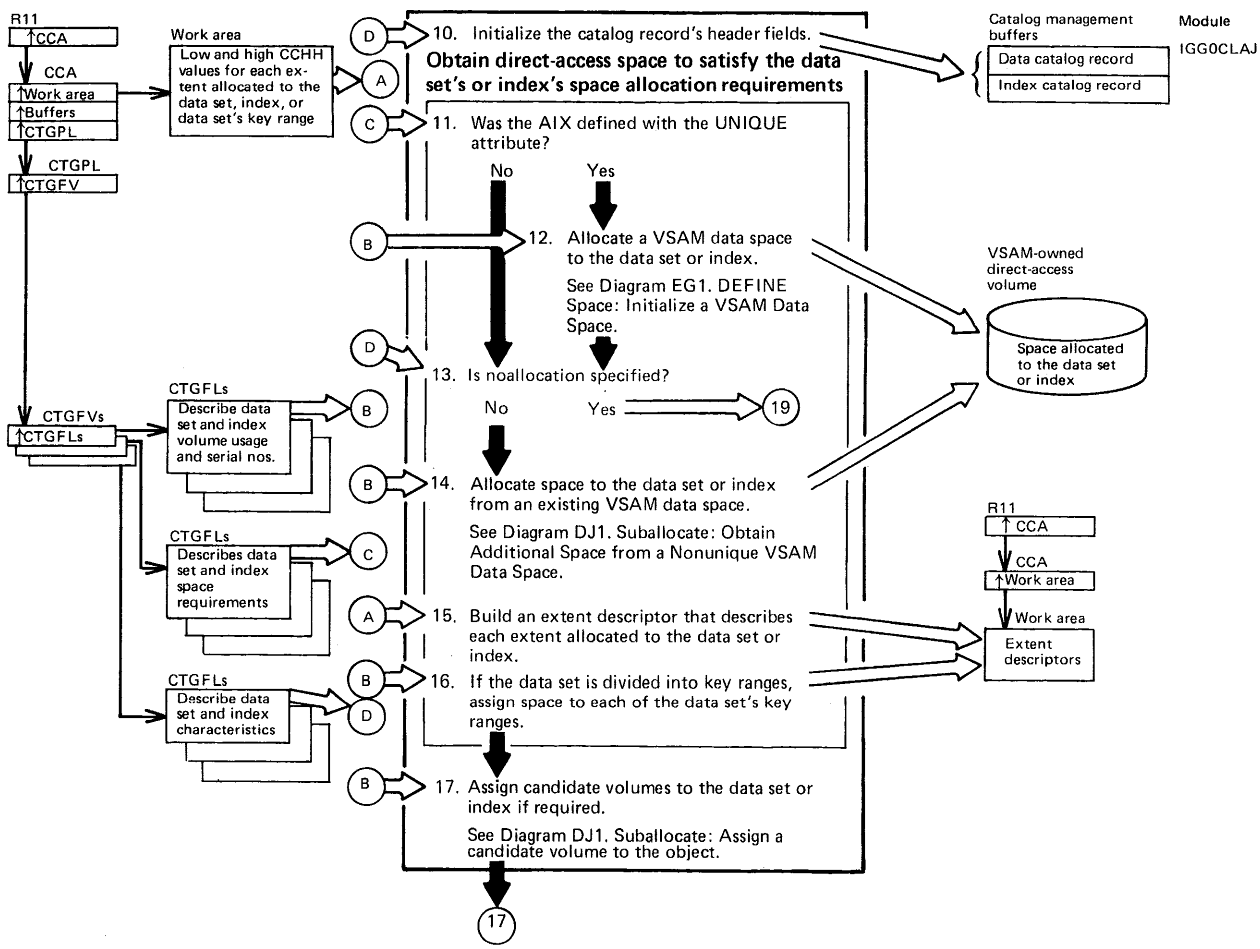


Diagram EI3. DEFINE AIX: Create an Alternate Index

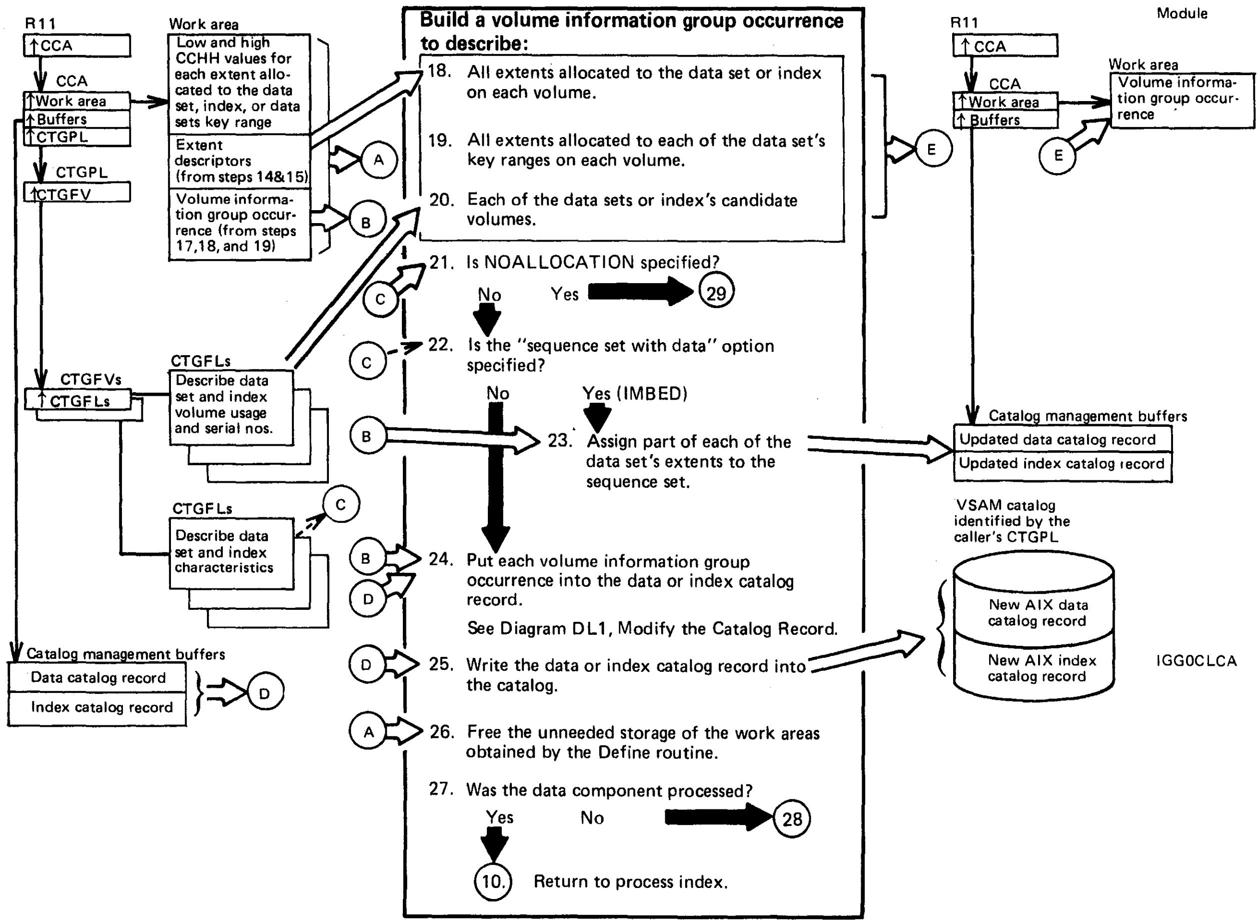


Diagram EI4. DEFINE AIX: Create an Alternate Index

CTGFL

Describe data set and index characteristics

28. Is NOALLOCATION Specified?

No — Yes → Return

29. Is this AIX to belong to the upgrade set?

Yes — No → Return

30. Add the AIX to the upgrade set.
See Diagram ER, Modify the Upgrade Set.

Return

Module
IGG0CLCA
IGG0CLCD

Diagram EJ1. DEFINE Path: Create a VSAM Path

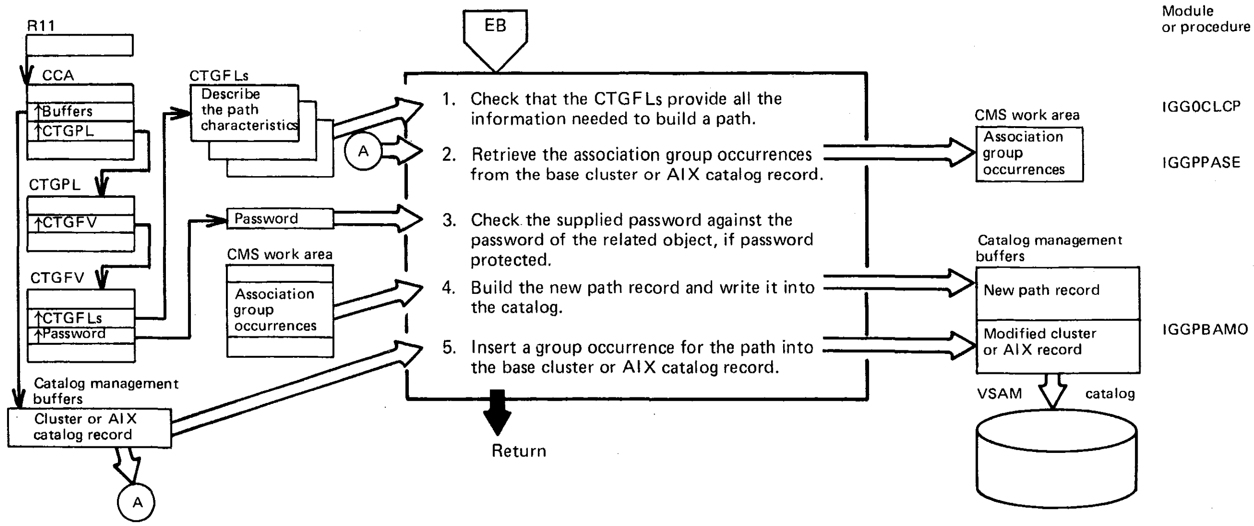
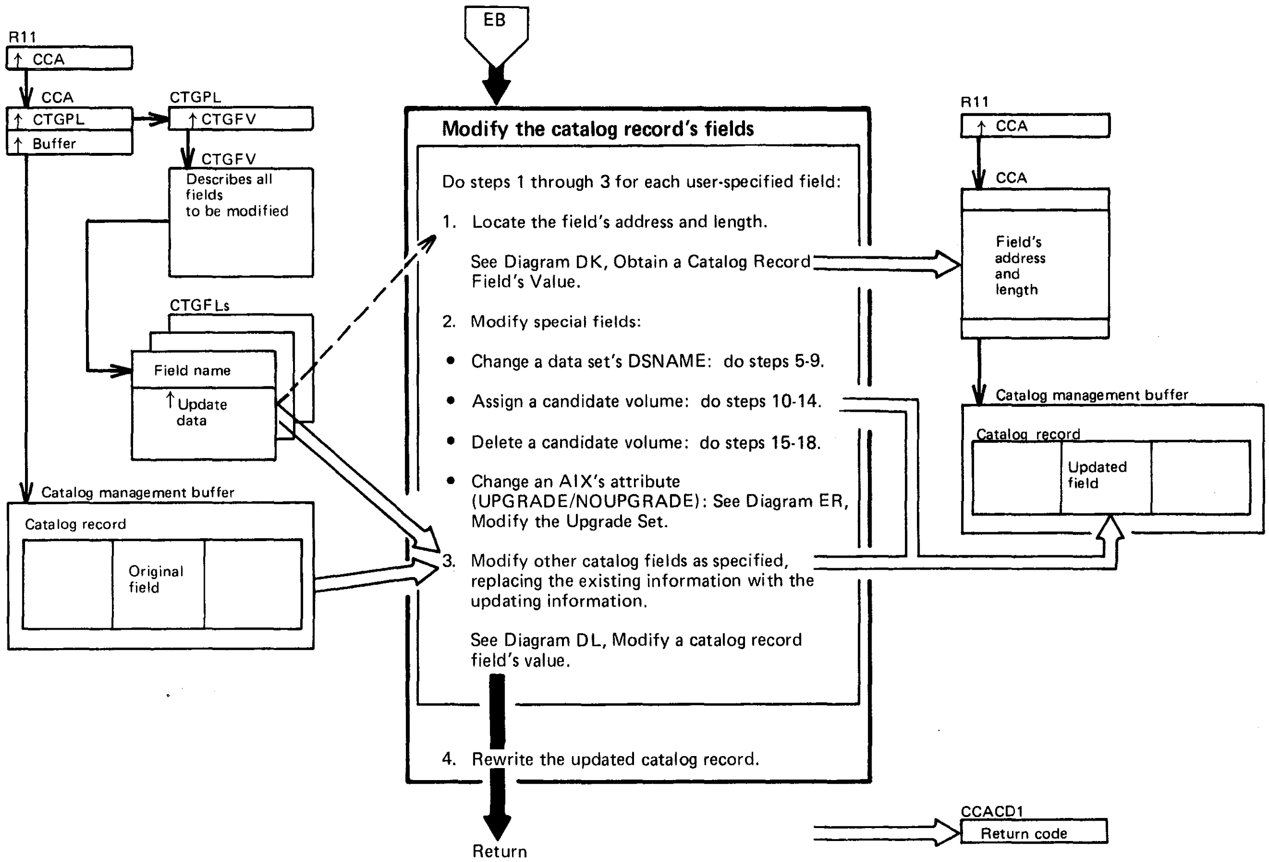


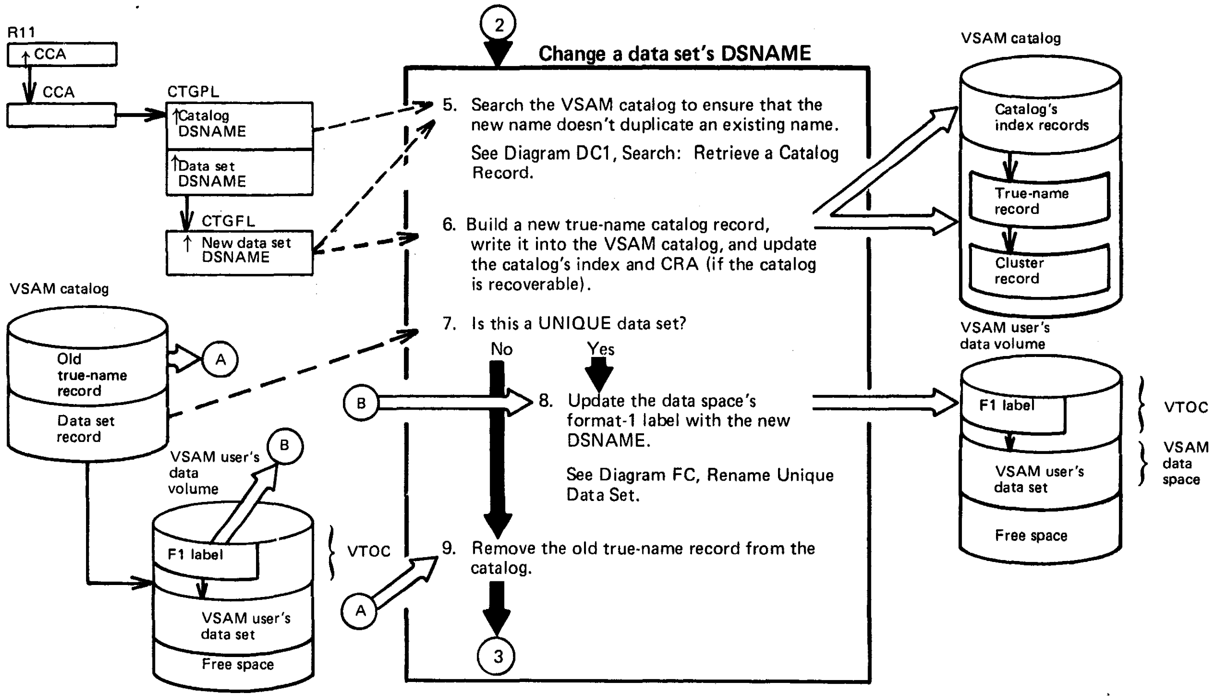
Diagram EK1. ALTER: Modify a Catalog Record



Notes for Diagram EK1

Description	Module	Procedure
The ALTER command enables the user to modify some of the information he established when he created a VSAM data set.		
2. When the data set name or allocated candidate volumes are changed, other catalog records besides the data set catalog record must be updated.	IGG0CLBD	IGGPALT
	IGG0CLEG	IGGPGET
	IGG0CLBD	IGGPALT
	IGG0CLEG	IGGPPUPC
	IGG0CLBD	IGGPALT
	IGG0CLBE	IGGPALVL
	IGG0CLBD	IGGPALT
		IGGPALMD
	IGG0CLAV	IGGPMOD

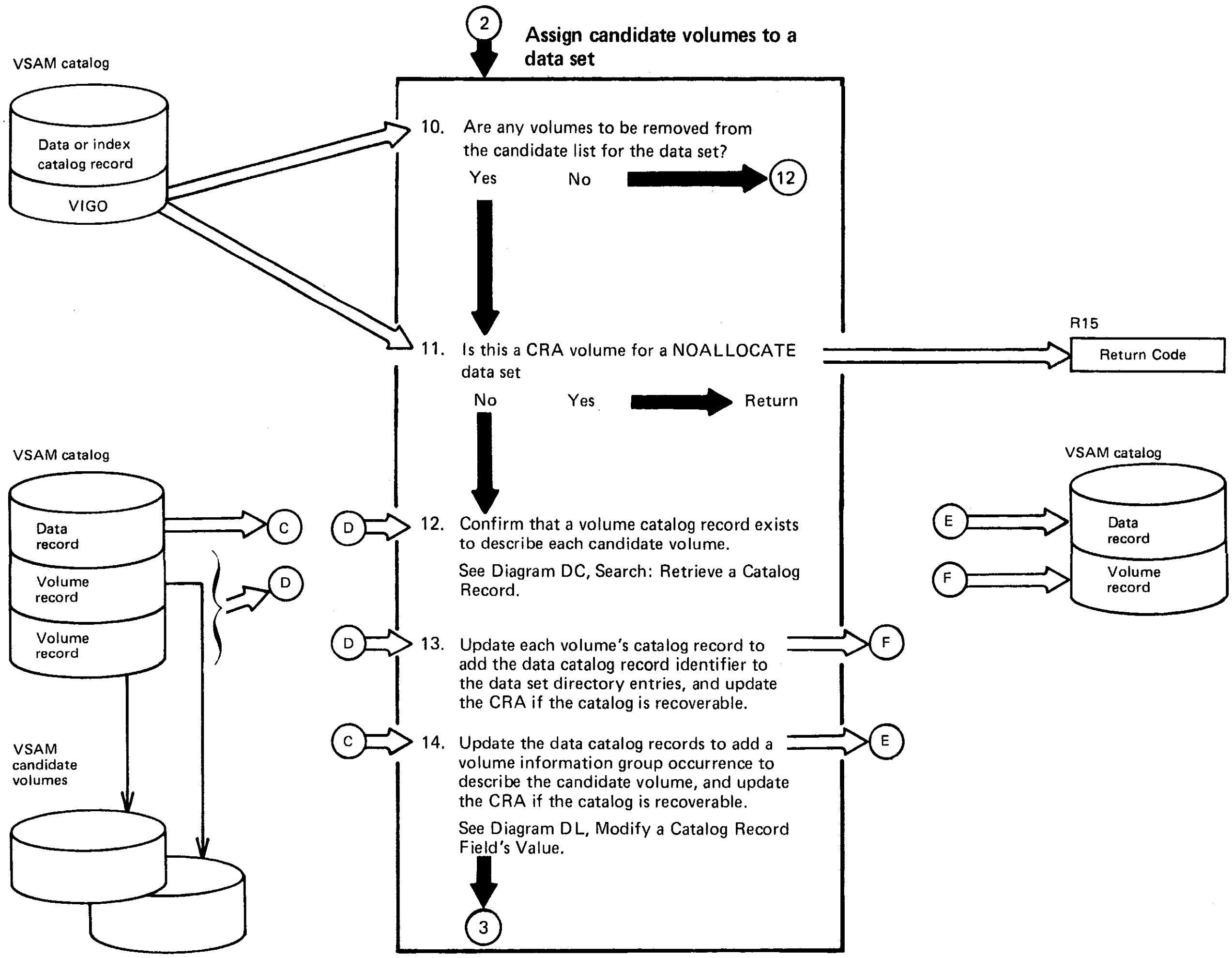
Diagram EK2. ALTER: Modify a Catalog Record



Notes for Diagram EK2

	Description	Module	Procedure
5.	The catalog specified by the ALTER command's CATALOG parameter is examined. The new name to be added must not exist in the catalog.	IGG0CLBD IGG0CLEG IGG0CLBD IGG0CLEG IGG0CLBD IGG0CLEG	IGGPALNM IGGPGET IGGPALNM IGGPPAD IGGPALNM IGGPPDE
6.	A new entry is placed in the high key range portion of the catalog.	IGG0CLBD	IGGPALNM
7.	A determination must be made as to whether or not the data set is unique.		
8.	All volumes that contain a format-1 label must have their names modified in the VTOC label.	IKQREN00	
9.	The name and control-interval number fields in the data set's true-name record are set to 0 and the record identifier field of the catalog record pointed to by the true-name entry is set to C'F'.	IGG0CLBD IGG0CLAZ IGG0CLBD IGG0CLBE	IGGPALF1 IGGPALGV IGGPEXT IGGPALT IGGPALVL

Diagram EK3. ALTER: Modify a Catalog Record

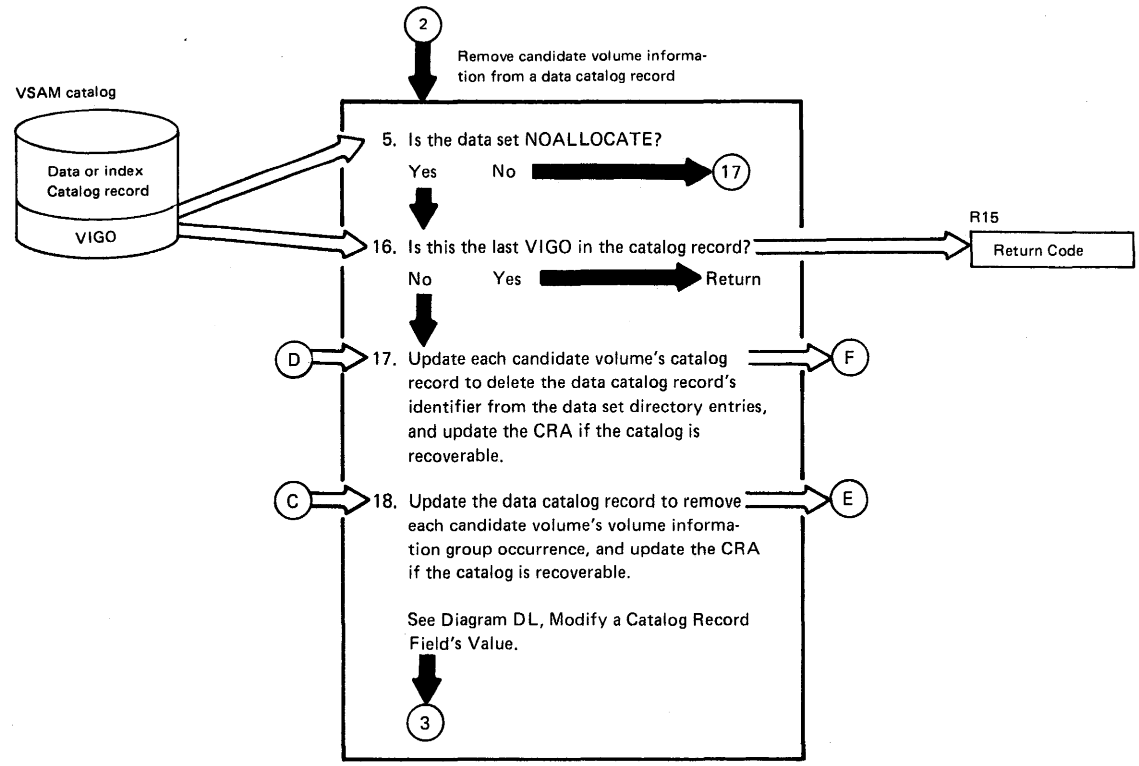


Notes for Diagram EK3

Description	Module	Procedure
12. IGGPALVA calls IGGPSALL (IGG0CLAR) to assign the candidate volume to the data set. If a volume catalog record does not exist for the candidate volume, the suballocate routine (IGG0CLAR) returns an error code.	IGG0CLAZ	IGGPEXT
	IGG0CLBE	IGGPALVL
		IGGPALVA
	IGG0CLAR	IGGPSALL
See Diagram DJ for details of the suballocate routine.		
13. The volume catalog record contains a data set directory that describes each VSAM data set's use of the volume's VSAM space.	IGG0CLBE	IGGPALVA
	IGG0CLAZ	IGGPEXT

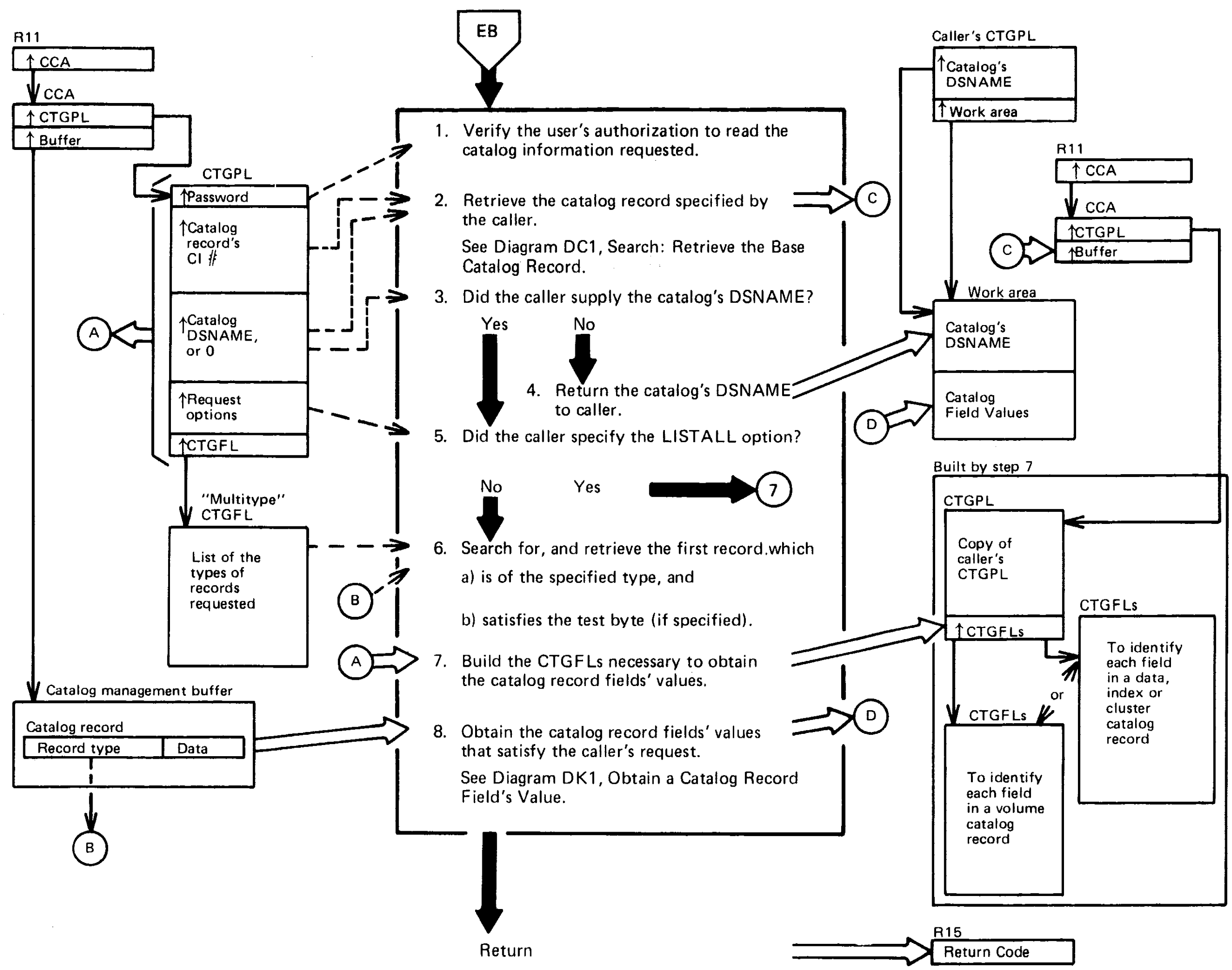
Description	Module	Procedure
14.	IGG0CLBE	IGGPALVA
		IGGPALSA
	IGG0CLAR	IGGPSALL
	IGG0CLBE	IGGPALSA
	IGG0CLAV	IGGPMOD
	IGG0CLBE	IGGPALSA
		IGGPALEC
	IGG0CLEG	IGGPGET
	IGG0CLBE	IGGPALVR

Diagram EK4. ALTER: Modify a Catalog Record



Description	Module	Procedure
17.	IGGOCLBN	IGGPALVR
18. Note: If the catalog is recoverable, the call to IGGOCLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.	IGGOCLBN	IGGPALVR
	IGGOCLAV	IGGPMOD
	IGGOCLBN	IGGPALVR
	IGGOCLAZ	IGGPEXT
	IGGOCLBN	IGGPALVR
		IGGPALVE
	IGGOCLEG	IGGPGET
	IGGOCLBN	IGGPALVE
	IGGOCLAV	IGGPMOD
	IGGOCLBN	IGGPALVE
	IGGOCLEG	IGGPPUPC
	IGGOCLBN	IGGPALVE
		IGGPALVR
	IGGOCLAV	IGGPMOD
	IGGOCLBN	IGGPALVR
	IGGOCLBE	IGGPALEC

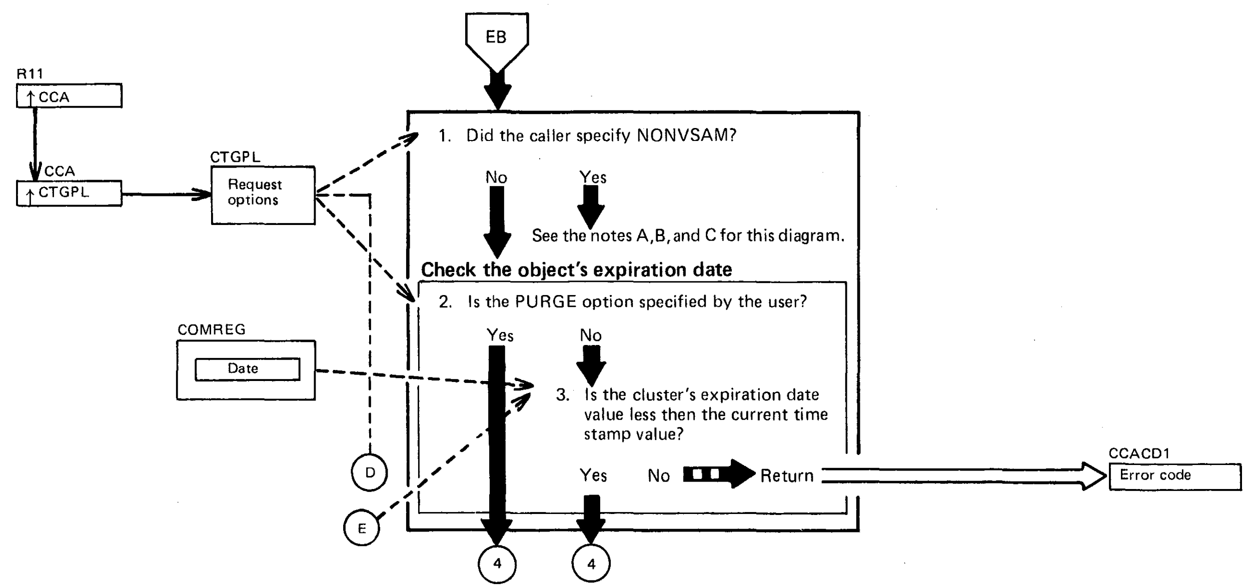
Diagram EL1. LISTCAT: Retrieve a Catalog Record's Contents



Notes for Diagram EL1

Description	Module	Procedure
The LISTCAT command allows the user to list all or a part of a VSAM catalog's contents.		
This routine, however, can return only information from one record (including extension records, if any are present) each time it is called. It is AMS's responsibility to specify the starting point for the search operation, so that records which have already been processed are not retrieved again. This applies regardless of whether LISTALL has been specified or only certain types of records are to be handled.		
1.	IGG0CLBQ IGG0CLBM	IGGPLSTC IGGPCKAU
2. The caller can request the information contained in a specific catalog record by providing the record's DSNAME (for a cluster, nonVSAM data set, or catalog) or volume serial number (for a volume).	IGG0CLBQ	IGGPLSTC
See DOS/VS VSAM-Extended: Using VSAM-E Commands and Macros, for details about the ENTRIES parameter.		
The true-name catalog records contain the DSNAME of each cluster and non-VSAM data set described in the catalog. Each true-name record also contains the control-interval number of its associated catalog record.		
6.	IGG0CLBQ IGG0CLEG	IGGPLSTC IGGPGET
8.	IGG0CLBQ IGG0CLAZ	IGGPLSTC IGGPEXT

Diagram EM1. DELETE: Remove a VSAM or NonVSAM Data Set

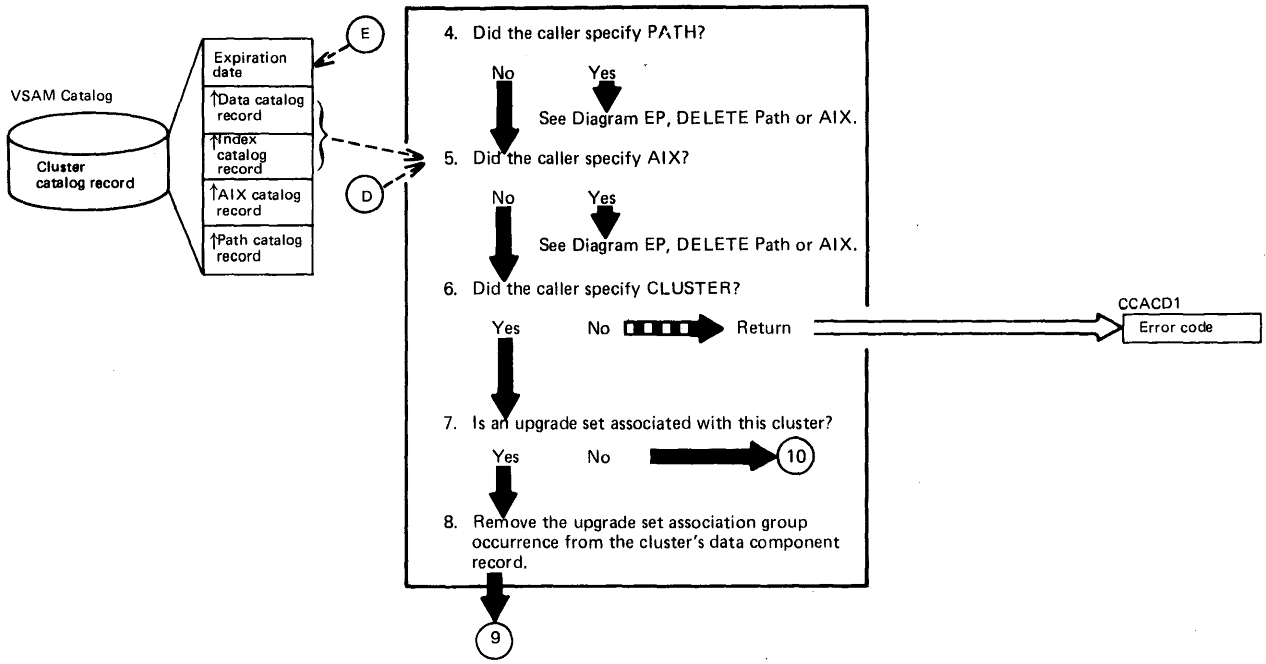


Notes for Diagram EM1

Description	Module (for nonVSAM)	Procedure
A. For CTGTYPE = A, control is transferred to the delete alien driver.	IGG0CLBG	IGGPDEL
		IGGPDELA
		IGGPDEL
B. The VTOC of the volume in which the nonVSAM resides is updated only if "scratch" is specified. (The volume information is extracted from the 'A' record. A check is done if the volume is mounted, and the label is scratched from the VTOC).	IGG0CLBG	IGGPDELA
	IGG0CLA7	IGGPDEMV
	IGG0CLAZ	IGGPEXT
	IGG0CLA7	IGGDEMV
	IGG0CLBG	IGGPDELA
	IGG0CLA7	IGGPDVMV
	IKQASNMT	
	IGG0CLA7	IGGPDVMV
	IGG0CLBG	IGGPDELA
	IGG0CLA7	IGGPDUSC
	IKQSCR00	
	IGG0CLA7	IGGPDUSC
C. The 'A' record is deleted from the catalog.	IGG0CLBG	IGGPDELA
	IGG0CLAN	IGGPDUND
	IGG0CLEG	IGGPDDE
	IGG0CLAN	IGGPDUND
	IGG0CLBG	IGGPDELA

Description	Module	Procedure
Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.		
The DELETE command enables the user to remove from the catalog all information about a specified cluster or nonVSAM data set.		
1. The catalog record identifier is examined to determine the record type and verify that the contents of the parameter list field CTGTYPE are correct. For processing of a nonVSAM delete (CTGTYPE = A), see notes A,B, and C.	IGG0CLBG	IGGPDEL
2. If the user specified PURGE, the data set's expiration date is ignored.	IGG0CLBG	IGGPDEL
3. If the user who created the data set specified an expiration date, the data set cannot be deleted until after that date (unless the PURGE parameter is specified: see step 2).	IGG0CLBG	IGGPDEL

Diagram EM2. DELETE: Remove a VSAM or NonVSAM Data Set

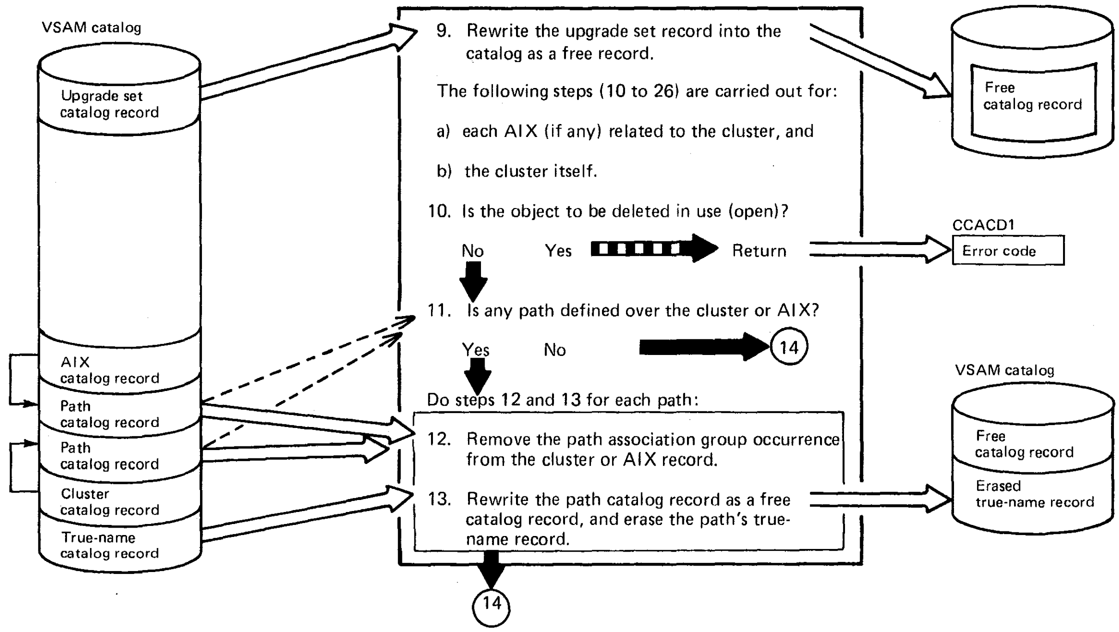


Notes for Diagram EM2

	Description	Module	Procedure
4.	If the user wants to delete a path, control is transferred to the delete path driver.	IGG0CLBG IGG0CLCX IGG0CLBG	IGGPDEL IGGPDELX IGGPDEL
5.	If the user wants to delete an alternate index, control is transferred to the delete AIX driver.	IGG0CLBG IGG0CLCX IGG0CLBG	IGGPDEL IGGPDELX IGGPDEL
6.	If the user wants to delete a cluster, control is transferred to the delete cluster driver.	IGG0CLBG	IGGPDEL IGGPDLCL IGGPDEL
7.	The upgrade set is retrieved via the cluster's data record containing the upgrade set (Y) association. In order to avoid having to update the Y record each time an AIX related to this cluster is deleted, the Y record is deleted at this point.	IGG0CLBG IGG0CLCX IGG0CLBG	IGGPDLCL IGGPDELY IGGPDLCL
8.		IGG0CLCX IGG0CLAV IGG0CLCX	IGGPDELY IGGPMOD IGGPDELY

Description	Module	Procedure

Diagram EM3. DELETE: Remove a VSAM or NonVSAM Data Set



Notes for Diagram EM3

	Description	Module	Procedure
9.	Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.	IGG0CLCX IGG0CLEG IGG0CLCX	IGGPDELY IGGPPDE IGGPDELY

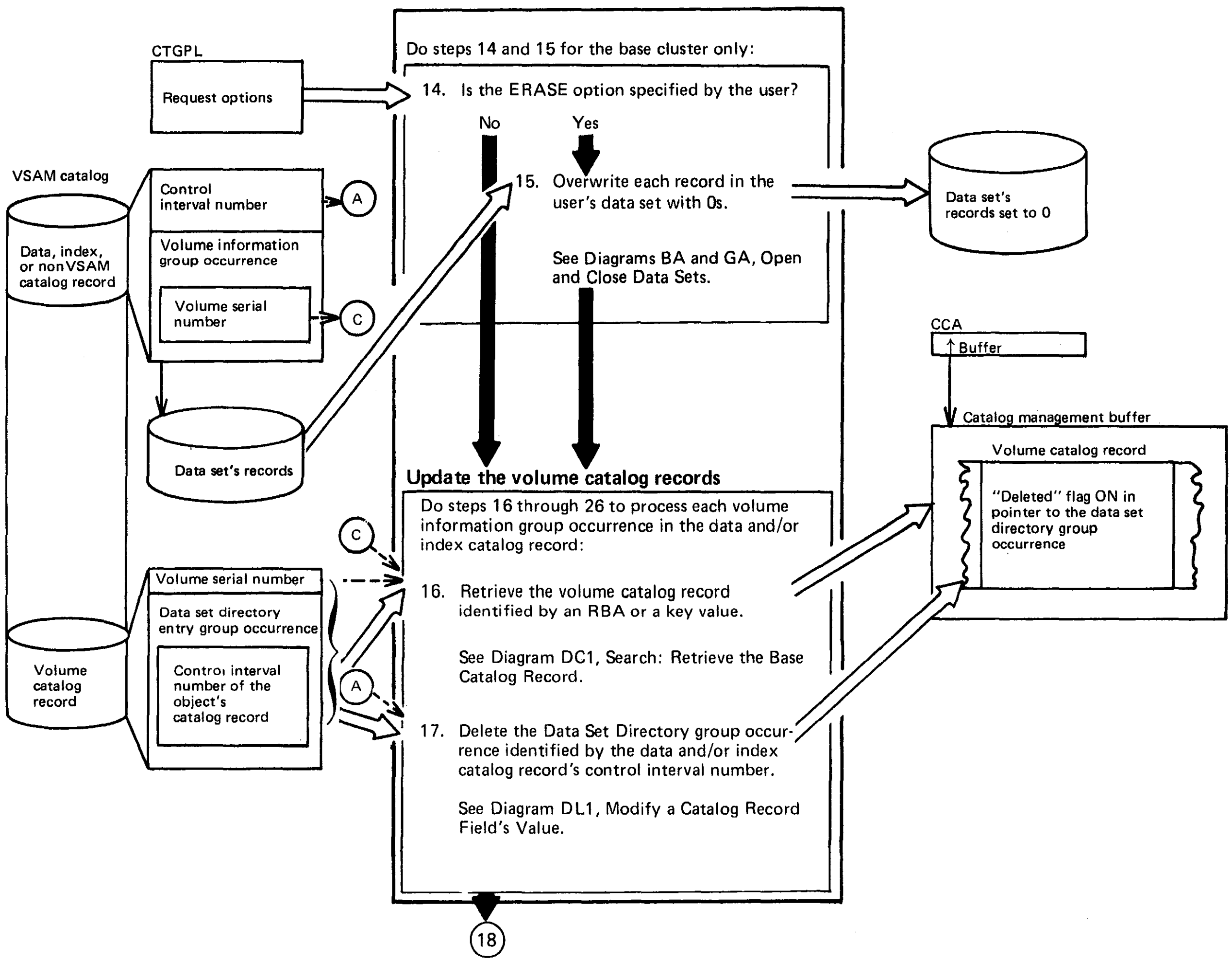
10. A delete request is rejected if the data set is open. The information indicating that a data set is open is kept in the Lock Table.

	for the cluster itself: Module	Procedure	for each AIX related to the cluster: Module	Procedure
10.	IGG0CLBG	IGGPDLCL	IGG0CLCX	IGGPDAIX
	IGG0CLCX	IGGPDELO	IGG0CLCX	IGGPDELO
	IGG0CLCX	IGGPDELO	IGG0CLCX	IGGPDELO
	IGG0CLBG	IGGPDLCL	IGG0CLCX	IGGPDAIX
12.	IGG0CLBG	IGGPDLCL	IGG0CLCX	IGGPDAIX

13. Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.

for the cluster itself: Module	Procedure	for each AIX related to the cluster: Module	Procedure
IGG0CLAV	IGGPMOD	IGG0CLAV	IGGPMOD
IGG0CLBG	IGGPDLCL	IGG0CLCX	IGGPDAIX
IGG0CLBG	IGGPDLCL	IGG0CLCX	IGGPDAIX
IGG0CLEG	IGGPPDE	IGG0CLEG	IGGPPDE
IGG0CLBG	IGGPDLCL	IGG0CLCX	IGGPDAIX

Diagram EM4. DELETE: Remove a VSAM or NonVSAM Data Set

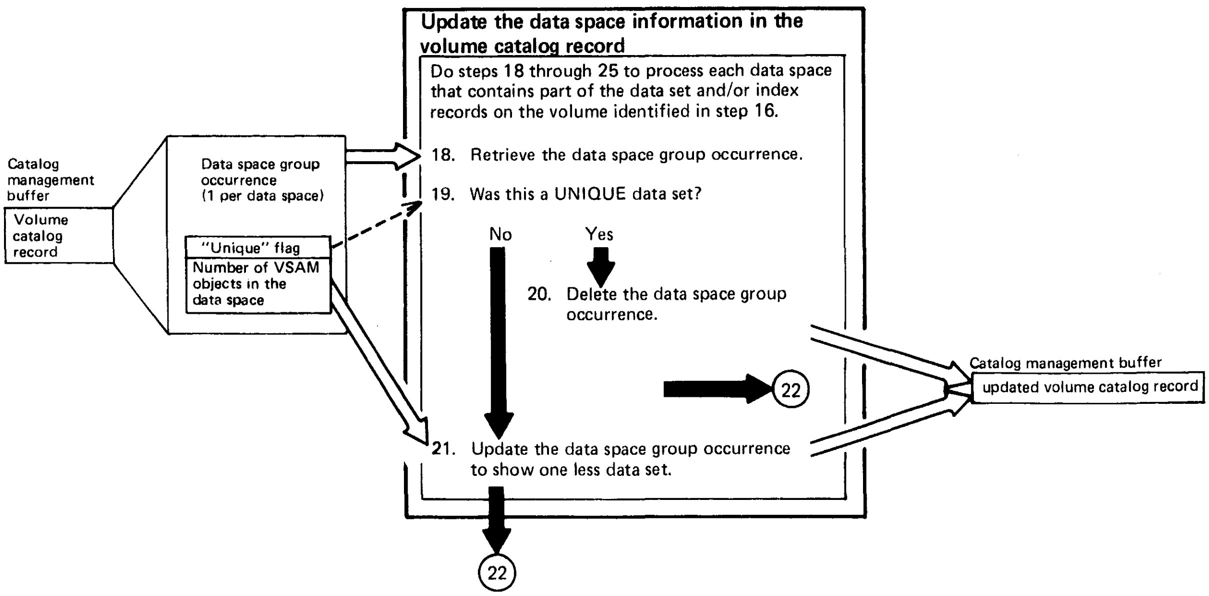


Notes for Diagram EM4

Description	Module	Procedure
14. Each of the cluster data set's records is sequentially retrieved and overwritten with 0s. SAM ESDSs with NOCIFORMAT are not overwritten with zero's.	IGG0CLBG IGG0CLA7 $$BOPEN IGG0CLA7 $$BCLOSE IGG0CLA7 IGG0CLBG	IGGPDLDS IGGPERAS IGGPERAS IGGPERAS IGGPDLDS
16.	IGG0CLBG IGG0CLEG IGG0CLBG	IGGPDLDS IGGPGET IGGPDLDS
17. The volume catalog record also contains a data set directory group occurrence to describe each VSAM data set that is contained, partially or completely, on the volume. If the volume is a candidate volume for a data set or index, the data set or index is also described by a data set directory group occurrence.	IGG0CLBG IGG0CLA7 IGG0CLAZ IGG0CLA7 IGG0CLAZ IGG0CLA7 IGG0CLAV	IGGPDLDS IGGPVMSC IGGPDEMV IGGPEXT IGGPDEMV IGGPVMSC IGGPDEDD IGGPEXT IGGPDEDD IGGPMOD

Description	Module	Procedure
	IGG0CLA7	IGGPDEDD IGGPVMSC

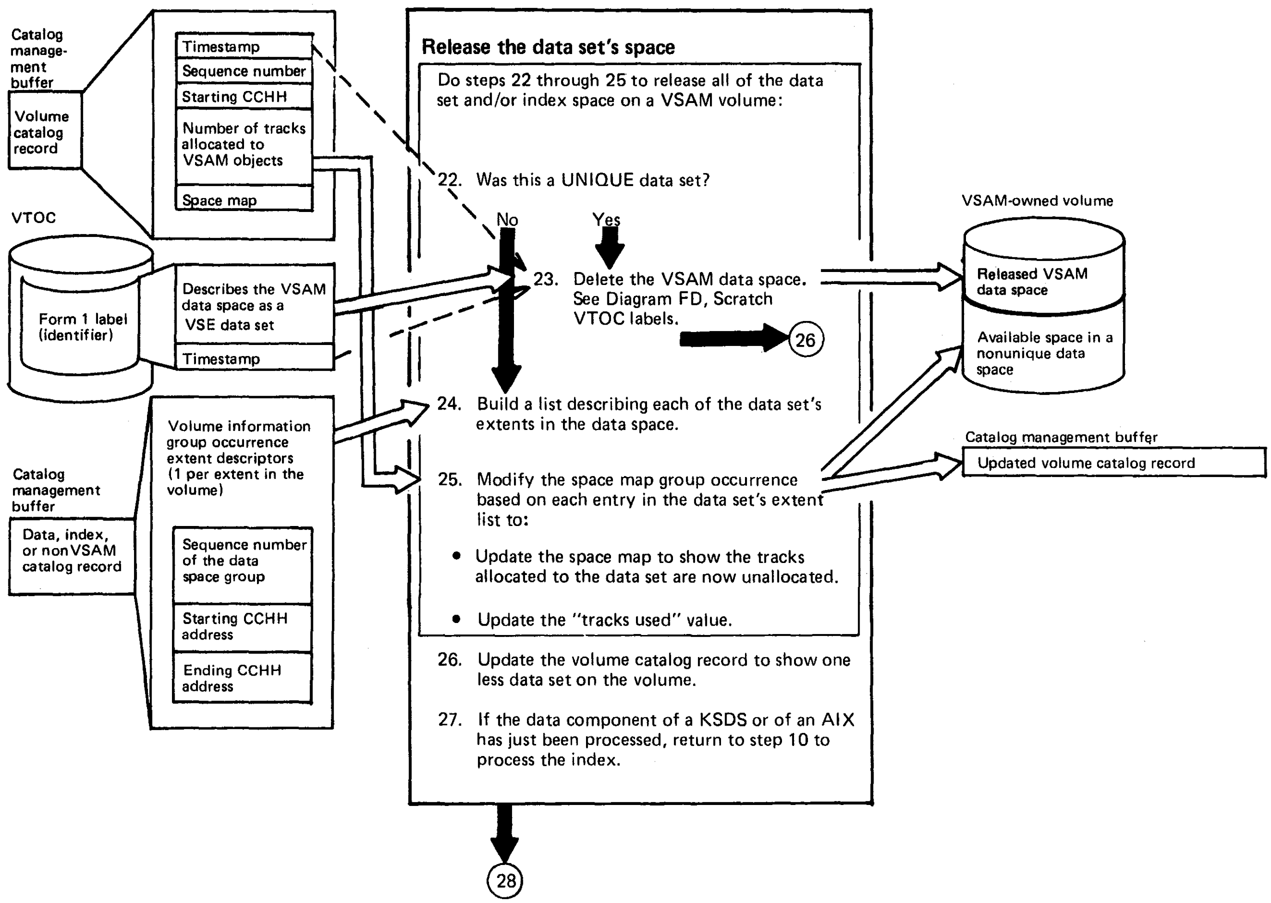
Diagram EM5. DELETE: Remove a VSAM or NonVSAM Data Set



Notes for Diagram EM5

	Description	Modules	Procedure
18-21	The volume catalog record contains a data space group occurrence to describe each VSAM data space on the volume.	IGG0CLA7	IGGPVMSC
			IGGPDESH
		IGG0CLAZ	IGGPEXT
		IGG0CLA7	IGGPDESH
		IGG0CLAV	IGGPMOD
		IGG0CLA7	IGGPDESH
			IGGPVMSC

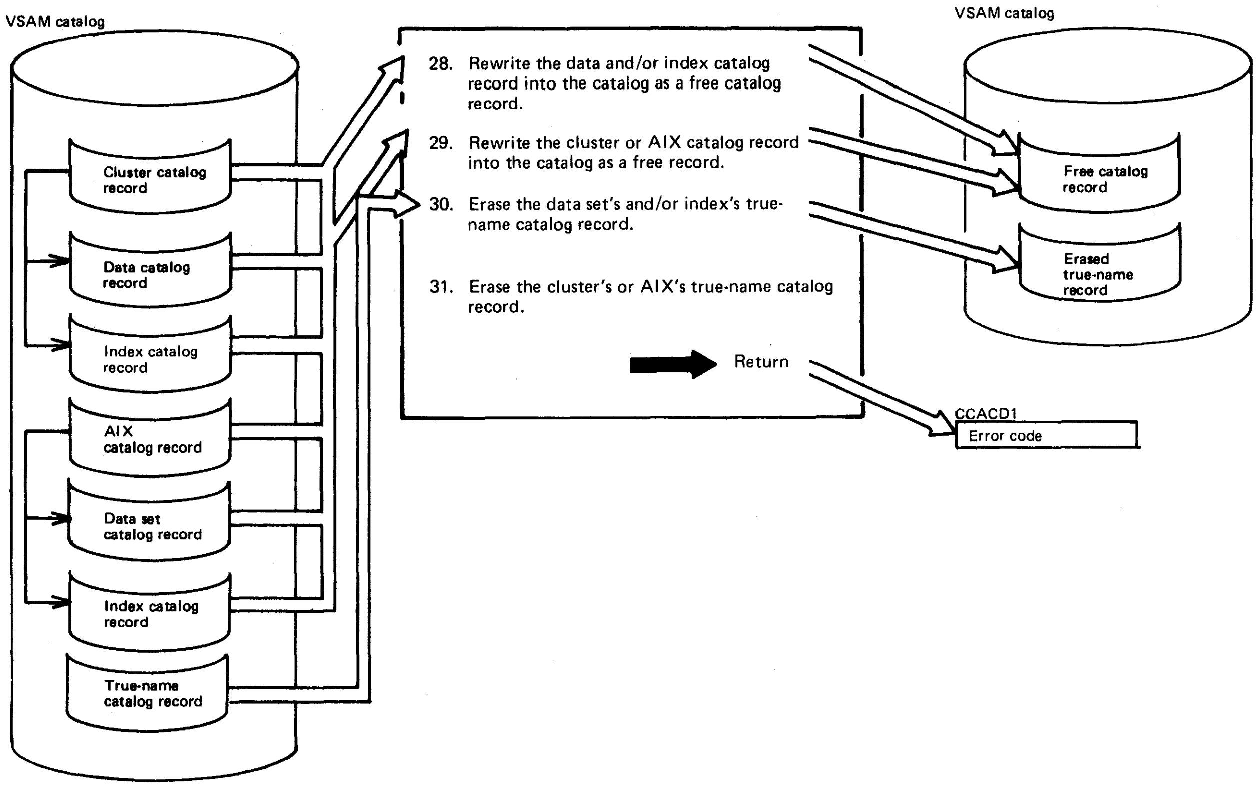
Diagram EM6. DELETE: Remove a VSAM or NonVSAM Data Set



Notes for Diagram EM6

Description	Module	Procedure
23. If the data or index space is unique, its data space is also deleted. Before the data space is deleted, the volume containing it is mounted.	IGG0CLA7	IGGPVMSC
		IGGPDVMV
	IKQASNMT	
	IGG0CLA7	IGGPDVMV
		IGGPVMSC
The volume containing the data space is optionally specified by the FILE parameter.		IGGPDUSC
	IKQSCR00	
	IGG0CLA7	IGGPDUSC
The extents in the data space's format-1 label and format-3 label, if present, are scratched from the VTOC.		IGGPVMSC
24. Each entry in the list identifies one of the data set or index extents in one of the data spaces on the volume.	IGG0CLA7	IGGPVMSC
		IGGPDEVG
	IGG0CLAZ	IGGPEXT
	IGG0CLA7	IGGPDEVG
		IGGPVMSC
25. Each of the data space's extents is described in the data space group occurrence.	IGG0CLA7	IGGPVMSC
	IGG0CLBF	IGGPSSCR
		IGGPESDG
	IGG0CLAZ	IGGPEXT
	IGG0CLBF	IGGPSSCR
		IGGPSCAN
		IGGPMRLC
	IGG0CLAV	IGGPMOD
	IGG0CLBF	IGGPSSCR
	IGG0CLA7	IGGPVMSC
27.	IGG0CLBG	IGGPDLDS

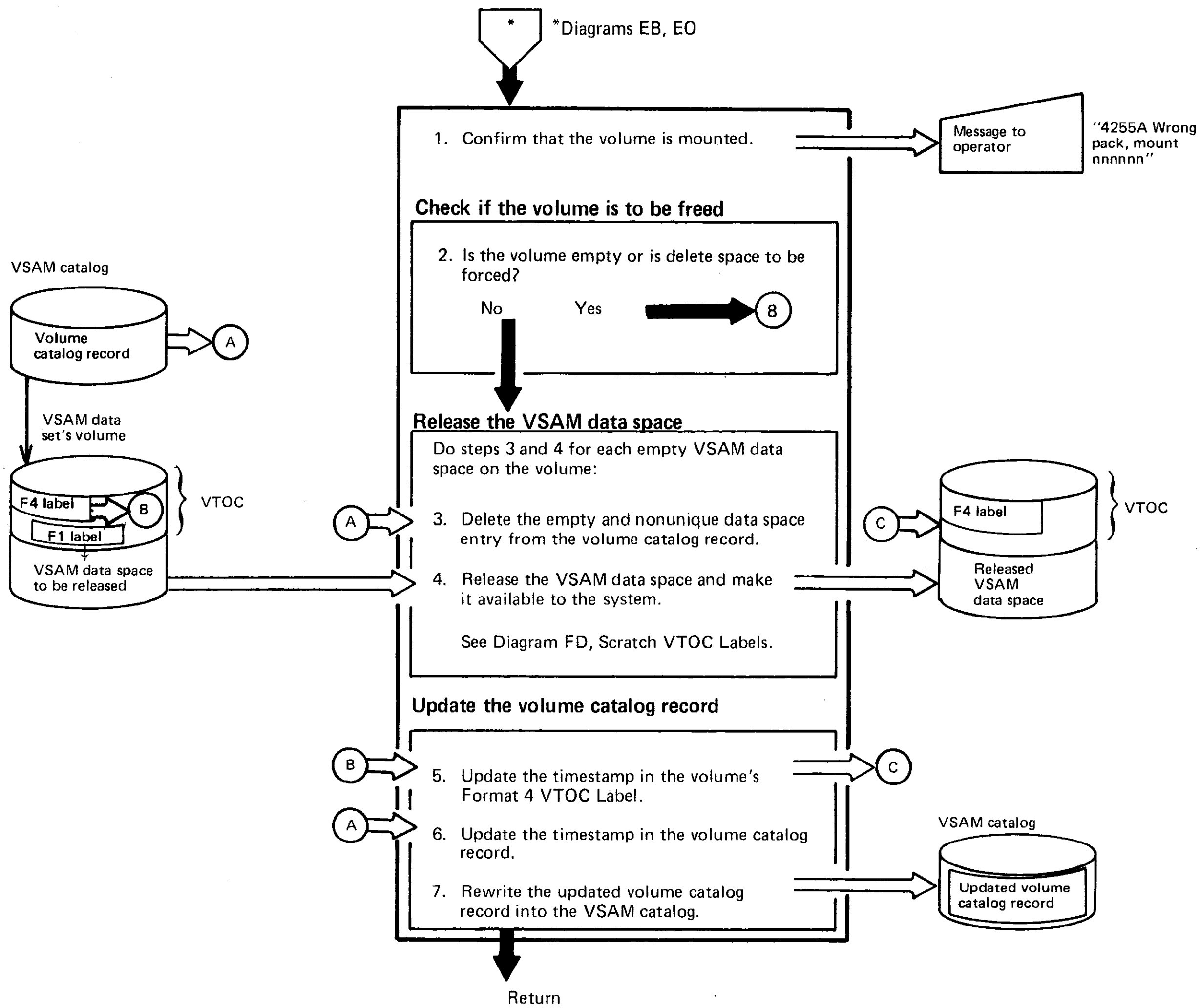
Diagram EM7. DELETE: Remove a VSAM or NonVSAM Data Set



Notes for Diagram EM7

	Description	Module	Procedure
28-31.	The delete routines erase the data set's true-name record and delete all references to the data set's DSNAME in the catalog's index.	IGG0CLBG	IGGPDLDS
		IGG0CLAN	IGGPDUND
			IGGPDEUN
		IGG0CLEG	IGGPGET
			IGGPDDE
		IGG0CLAN	IGGPDEUN
			IGGPDUND
		IGG0CLBG	IGGPDLDS

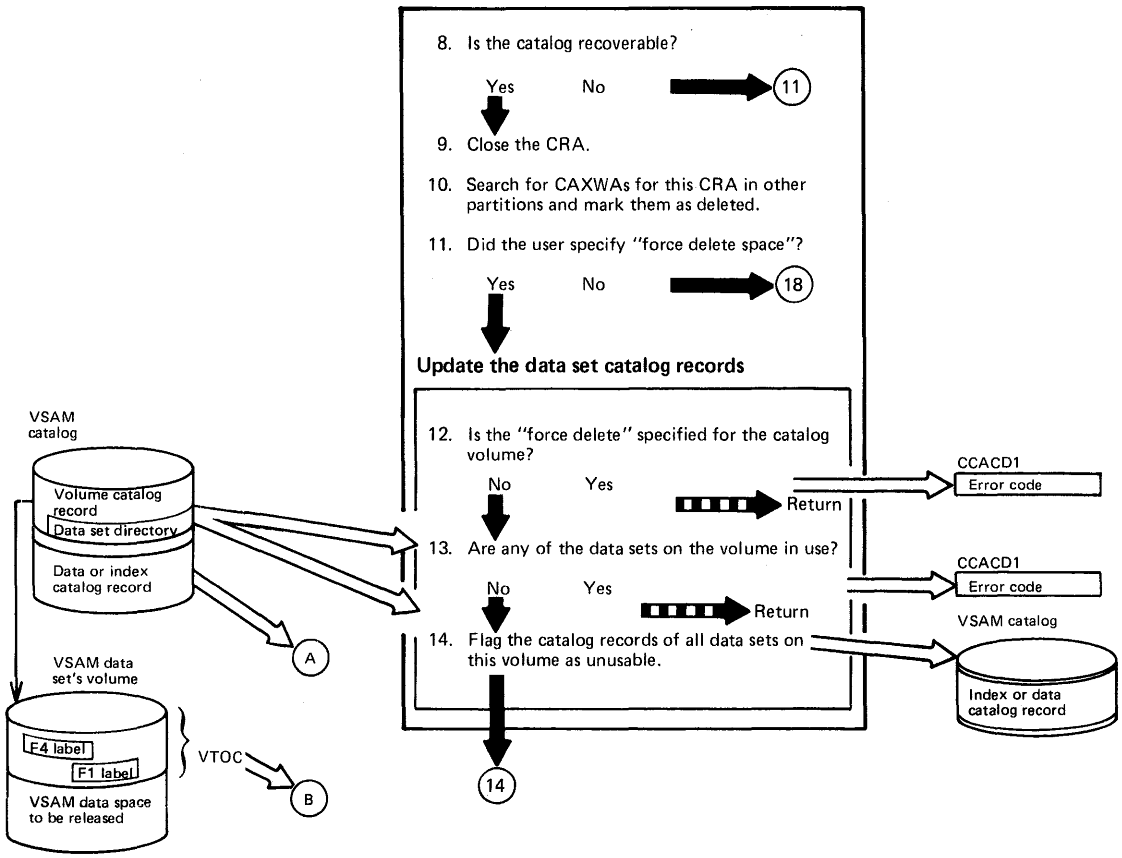
Diagram EN1. DELETE Space: Release All of the Empty VSAM Data Space on a Volume



Notes for Diagram EN1

	Description	Module	Procedure
	The DELETE space command enables the user to release all VSAM data spaces on a specified volume.		
1.	If the volume isn't already mounted and available for use, the DELETE space routine issues the appropriate mount message to the operator.	IGG0CLBL IKQASNMT IGG0CLBL IKQVDTPE	IGGPDELS IGGPDLVM IGGPDLVM
2.	A volume is empty when its volume catalog record contains no data set directory group occurrence (which normally describes data sets on the volume). "Force" is an option the user may specify to delete a volume even though there are still data sets on this volume.	IGG0CLBL IGG0CLAZ IGG0CLBL	IGGPDELS IGGPDLVC IGGPEXT IGGPDLVC IGGPDELS
3.	The volume catalog record contains a data space group occurrence to describe each VSAM data space on the volume.	IGG0CLBL IGG0CLAZ IGG0CLBL IGG0CLAV IGG0CLBL IGG0CLAV	IGGPDLCD IGGPEXT IGGPDLSH IGGPMOD IGGPDLSD IGGPMOD
4.	The DADSM routine IKQSCR00 releases VSAM data space and makes its space available to other VSE system users.	IGG0CLBL IKQSCR00	IGGPDLSC
5.	The timestamp in the VTOC is updated to indicate when the last change to the VTOC was made by VSAM.	IGG0CLBL IGG0CLBU IGG0CLBL IGG0CLBU IGG0CLBL	IGGPDELS IGGPDLTS IGGPF4RD IGGPDLTS IGGPF4WR IGGPDLTS
6.		IGG0CLBL	IGGPDELS
7.	Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.	IGG0CLBL IGG0CLEG IGG0CLBL	IGGPDELS IGGPPUPC IGGPDELS

Diagram EN2. DELETE Space: Release All of the Empty VSAM Data Space on a Volume

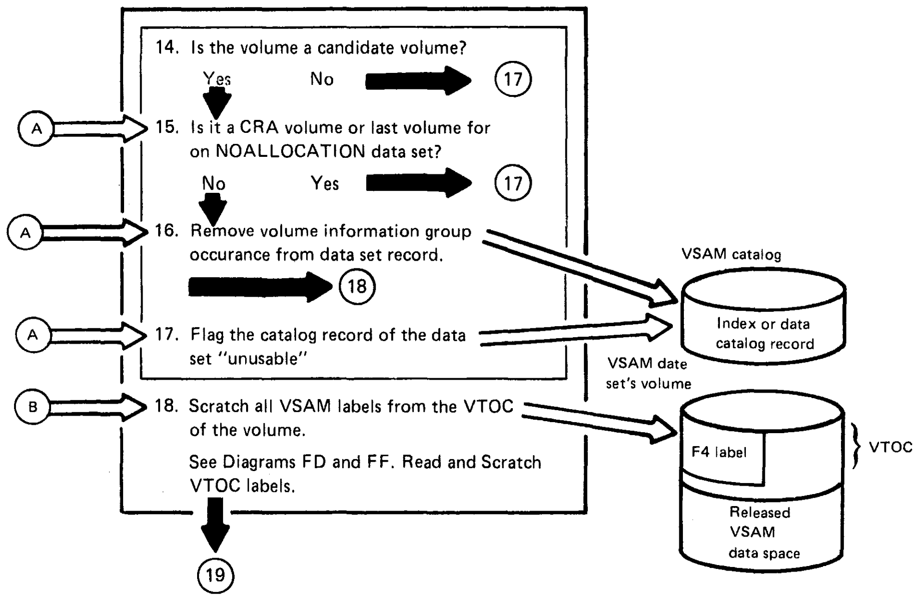


Notes for Diagram EN2,

	Description	Module	Procedure
8.	If a volume owned by a recoverable catalog is to be freed, its CRA will be scratched. Care must thus be taken that no partitions will use this CRA again.	IGG0CLBL	IGGPDELS
9.	By closing the CRA, all control blocks in the requesting partition will be freed.	IGG0CLBL . IGG0CLCO IGG0CLBL $$BCLOSE IGG0CLBL	IGGPDELS IGGPDLCR IGGPSCAX IGGPDLCR IGGPDLCR
10.	The control blocks for the CRA are erased at the end of the current job (which issued the DELETE SPACE command). If, however, a job in another partition issues a DEFINE SPACE for the same volume before the end of the current job, the DEFINE SPACE routine must know that is has to build a new CRA. For this reason, the CRA's control blocks in all other partitions are marked invalid.	IGG0CLBL $$BCLCRA IGG0CLBL IGG0CLBL	IGGPDLCR IGGPDLCR IGGPDELS

	Description	Module	Procedure
13.	As long as any of the data sets on the volume is in use, the forced delete space is rejected.	IGG0CLBL IGG0CLCL IGG0CLAZ IGG0CLCL IGG0CLCX IGG0CLCL	IGGPDELS IGGPDLSF IGGPDXDS IGGPEXT IGGPDLSF IGGPDELO IGGPDLSF
14.	The catalog records of all data sets on the volume are flagged as unusable. Since their space will be partly or completely gone, they are thus marked to inhibit any open requests for output, and to warn CMS delete cluster. (CRA duplicates may be gone; volume occurences may point to non-existing volumes.) Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.	IGG0CLCL IGG0CLAZ IGG0CLCL IGG0CLEG IGG0CLCL IGG0CLEG IGG0CLCL IGG0CLAV IGG0CLCL	IGGPDLSF IGGPDXDS IGGPEXT IGGPDLSF IGGPGET IGGPDLSF IGGPPUPC IGGPDLSF IGGPDMDS IGGPMOD IGGPDLSF

Diagram EN3. DELETE Space: Release All of the Empty VSAM Data Space on a Volume



Notes for Diagram EN3.

Description

14. The catalog records of all data sets with space on this volume are flagged as unusuable, since their space will be partly or completely gone, they are thus marked to inhibit any open requests for output, and to warn CMS delete cluster. If this is a candidate volume, the data set remains usable and the volume is removed from the data set's candidate list. (CRA duplicates may be gone: volume occurances may point to non-calsting volumes.)

    Note: If the catalog is recoverable, the call to IGGOGLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.

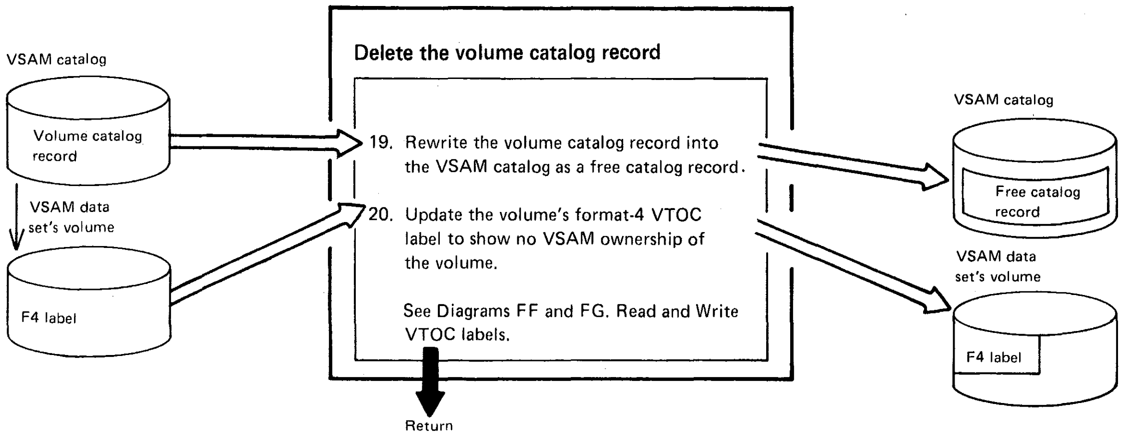
Module	Procedure
IGGOCLCL	IGGPDLSF
	IGGPDXDS
IGGOCLAZ	IGGPEXT
IGGOCLCL	IGGPDLSF
IGGOCLEG	IGGPGET
IGGOCLCL	IGGPDLSF
IGGOCLEG	IGGPPUPC
IGGOCLCL	IGGPDLSF
	IGGPDMDS
IGGOCLAV	IGGPMOD
IGGOCLCL	IGGPDLSF

18. The VTOC is scanned for VSAM VTOC labels, and all VSAM VTOC labels are scratched thus returning all space occupied by VSAM to the VTOC.

empty vol.:	force delete:	empty vol.:	force delete:
IGGOCLBL	IGGOCLCL	IGGPDELS	IGGPDLS
	IGGOCLCL		IGGPDVSC
	IKQVTC00		
	IGGOCLCL		IGGPDVSC
	IKQRDS00		
	IGGOCLCL		IGGPDVSC
	IKQSCR00		
	IGGOCLCL		IGGPDVSC
	IKQVTC00		
	IGGOCLCL		IGGPDVSC
	IGGOCLCL		IGGPDSF
	IGGOCLBL		IGGPDELS

	Volume only being used for data space (No candidate)	Volume only being used as candidate (no file space)	Volume being used as both candidate and file space	Volume has no data set directory group occurance
Delete Space (not forced)	DELETE fails	DELETE fails	DELETE fails	DELETE successful
Delete Space (Forced)	• Files having space on volume are marked unusable. • DELETE successful	• Candidate volume group occurances (VGO) are deleted if - not last VGO for file - not CRA vol for file else file marked ususable. • DELETE successful	• Files having space on the vol are marked unusable • Candidate vol G.O.'s are deleted if - not V.G.O. for file - not CRA vol for file else, the file is marked ususable. • DELETE is successful	DELETE successful

Diagram EN4. DELETE Space: Release All of the Empty VSAM Data Space on a Volume



Notes for Diagram EN4

Module		Procedure	
empty vol.:	force delete:	empty vol.:	force delete:

19. Reset the volume record and all its extension records so that it is available for future assignment.

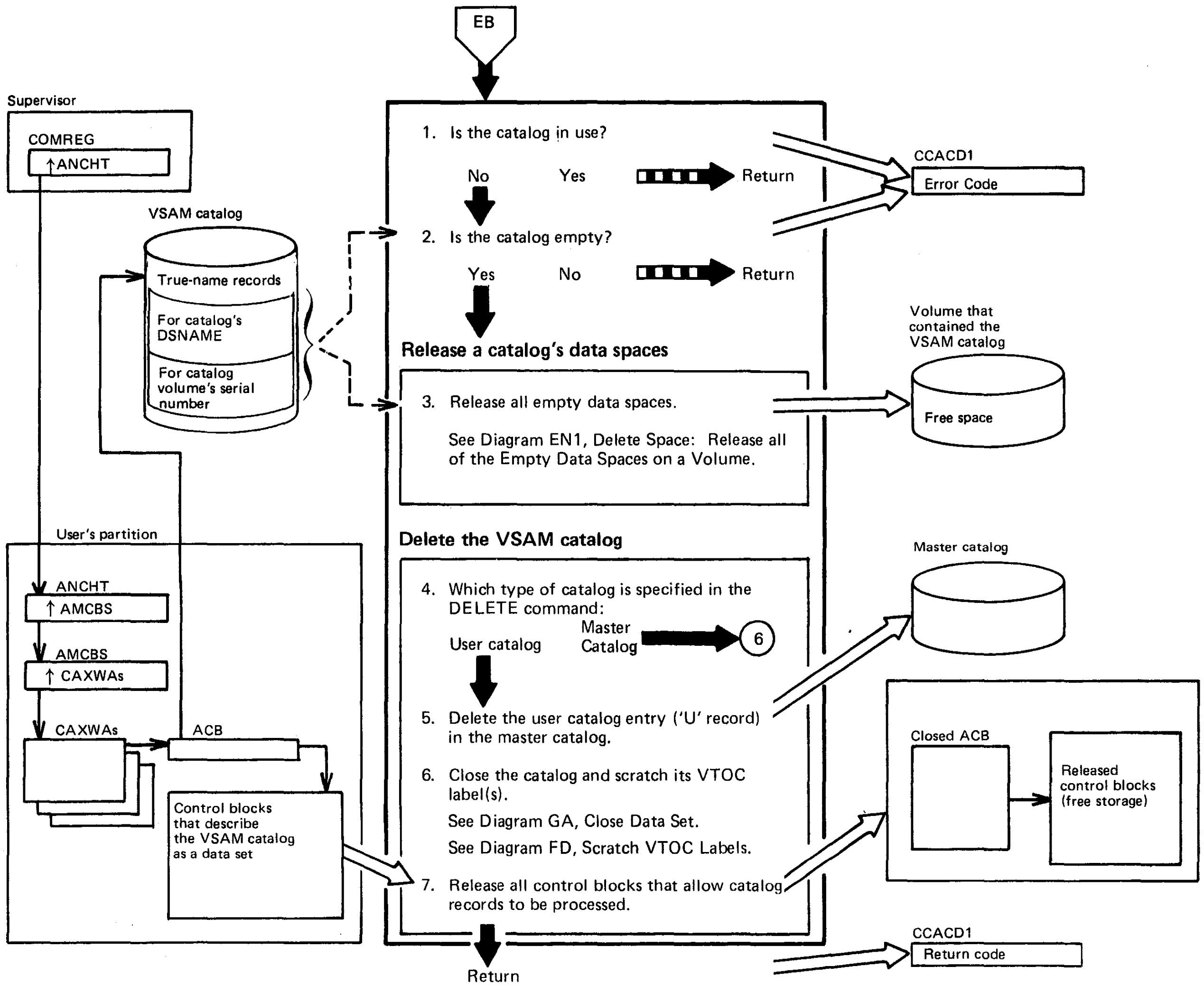
IGGOCLBL	IGGPDELS
	IGGPDLET
IGGOCLEG	IGGPGET
IGGOCLBL	IGGPDLET
IGGOCLEG	IGGPPDEC
IGGOCLBL	IGGPDLET

20. The format-4 label is the first entry in a direct-access volume's VTOC. It contains the volume owner's identification and information on how the volume is used.

Note: If the catalog is recoverable, the call to IGGOCLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is Set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.

IGGOCLBL	IGGPDLET
IGGOCLBU	IGGPF4RD
IGGOCLBL	IGGPDLET
IGGOCLBU	IGGPF4WR
IGGOCLBL	IGGPDLET
	IGGPDELS

Diagram EO1. DELETE Catalog: Release a VSAM Catalog

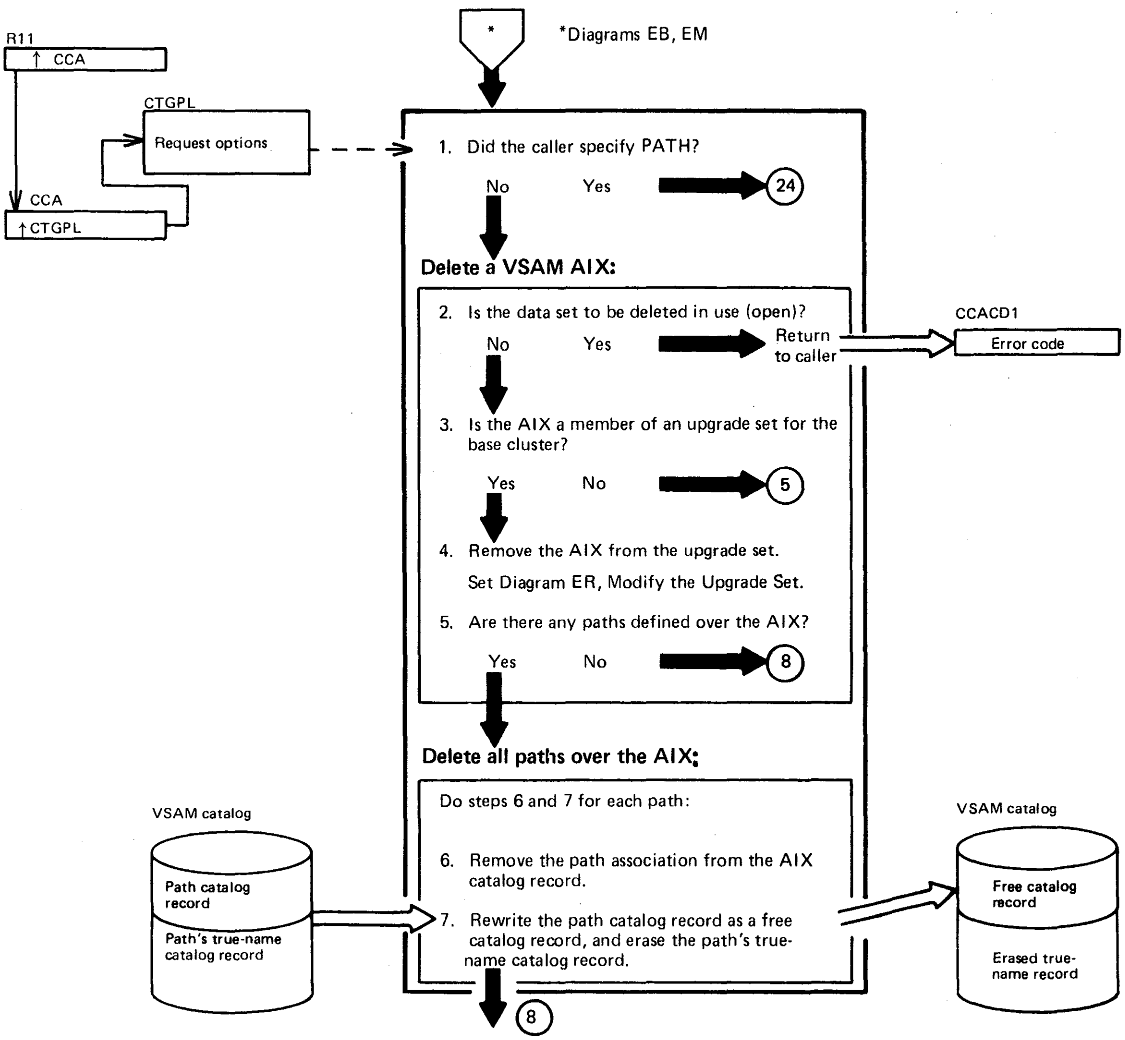


Notes for Diagram EO1

	Description	Module	Procedure
	The DELETE catalog command enables the user to release a catalog's space and make it availiable to other VSE system users. The catalog must be empty or the request is rejected.		
1.	By attempting to LOCK the catalog for exclusive useit may be determined wether it is currently in use.	IGG0CLAF	
2.	If the catalog contains more than two true-name records, it is not empty and cannot be deleted.	IGG0CLAF	IGGPDELC
3.	The volume catalog record contains an entry for each VSAM data space allocated on the volume. Each entry contains the data necessary to free the data space.	IGG0CLAF	IGGPSDSP
4.	One of these type must be specified for a DELETE CATALOG operation.		
5.	Each user catalog has an entry in the master catalog.		

	Description	Module	Procedure
6.	The communications region (COMREG) points to the ANCHT which points to the AMCBS. The AMCBS points to the control blocks that describe the VSAM catalog to the system.	IGG0CLAF	IGGPDELC
		IGG0CLAL	IGGPDBVC
		IGG0CLAF	IGGPDELC
		IGG0CLEG	IGGPPDE
		IGG0CLAF	IGGPDELC
		IGG0CLEG	IGGPGET
		IGG0CLAF	IGGPDELC
	Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.	IKQVDTPE	
		IGG0CLAF	IGGPDELC
		IGG0CLAZ	IGGPEXT
		IGG0CLAF	IGGPDELC
		$$BCLOSE	
		IGG0CLAF	IGGPDELC
			IGGPSDSP
		IKQSCR00	
		IGG0CLAF	IGGPDELC
		IGG0CLBU	IGGPF4RD
		IGG0CLAF	IGGPDELC
		IKQASNMT	
		IGG0CLAF	IGGPDELC
		IGG0CLBU	IGGPF4WR

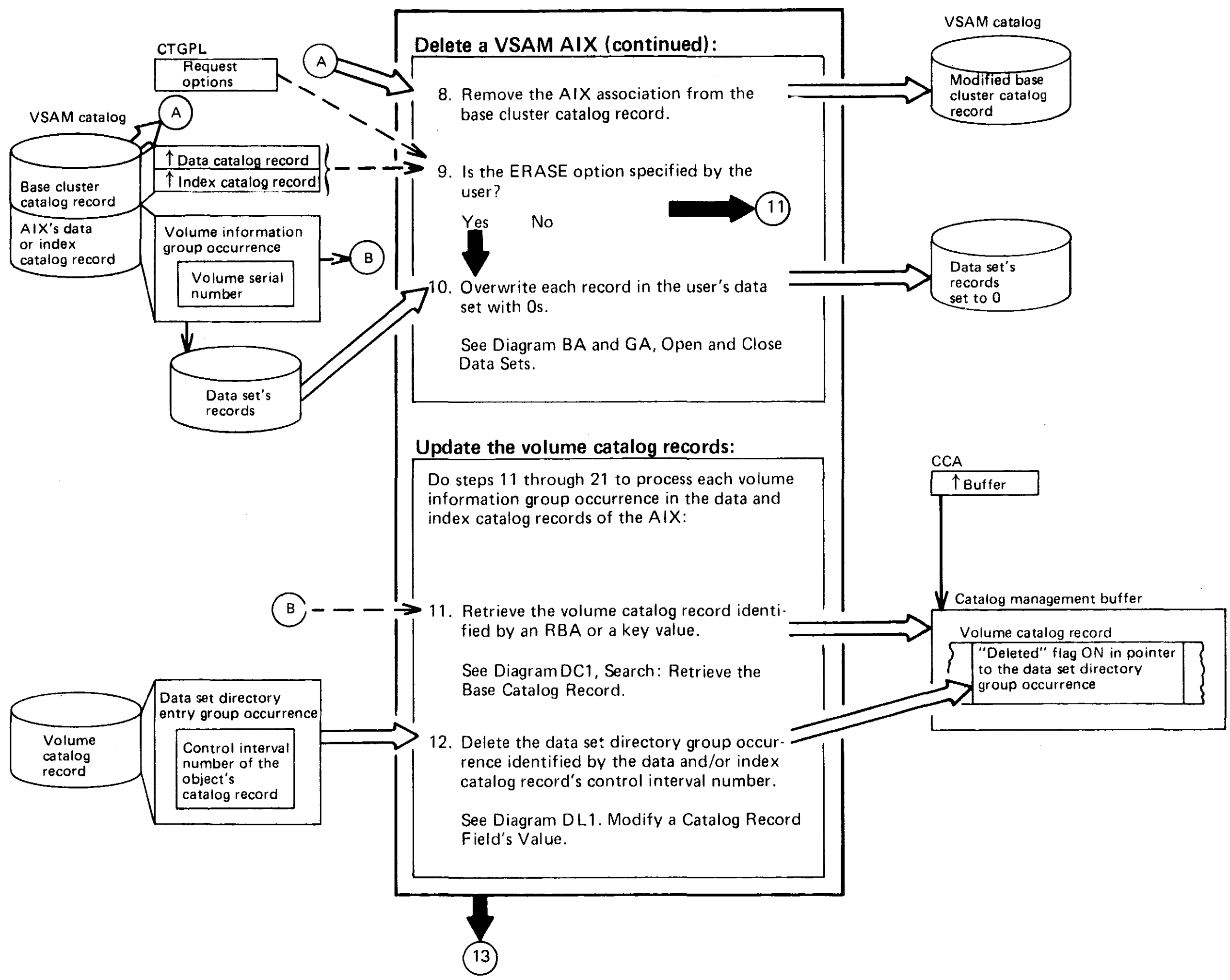
Diagram EP1. DELETE AIX or Path: Release an Alternate Index or a VSAM Path



Notes for Diagram EP1

Description	Module	Procedure
The DELETE command enables the user to remove from the catalog all information about a specified AIX or path.		
1. If the user wants to delete a path, control is transferred to the delete path driver.	IGG0CLBG IGG0CLCX IGG0CLBG	IGGPDEL IGGPDELP IGGPDEL
2. A delete request is rejected if the data set is open. The information whether a data set is open is kept in the look-aside table.	IGG0CLCX IGG0CLCX IKQLASMD IGG0CLCX IGG0CLCX	IGGPDELX IGGPDELO IGGPDELO IGGPDELX
3-4. The upgrade set is retrieved via the AIX's base cluster data component containing the upgrade set association. In case this AIX is is the last member in the upgrade set, the upgrade set and its association are deleted. Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.	IGG0CLCX IGG0CLEG IGG0CLCX IGG0CLAZ IGG0CLCX IGG0CLCD IGG0CLEG IGG0CLCD IGG0CLAZ IGG0CLCD IGG0CLEG IGG0CLCD IGG0CLAV IGG0CLCD IGG0CLEG IGG0CLCD IGG0CLAV	IGGPDELX IGGPGET IGGPDELX IGGPEXT IGGPDELX IGGPUPG IGGPGET IGGPUPG IGGPEXT IGGPUPG IGGPGET IGGPUPG IGGPMOD IGGPUPG IGGPPDEC IGGPUPG IGGPMOD
5.	IGG0CLCX IGG0CLAZ IGG0CLCX	IGGPDELX IGGPEXT IGGPDELX
6. The path association is removed from its AIX before the path is actually deleted. Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.	IGG0CLCX IGG0CLAV IGG0CLCX	IGGPDPTH IGGPMOD IGGPDPTH
7.	IGG0CLCX IGG0CLEG IGG0CLCX	IGGPDPTH IGGPPDE IGGPDPTH

Diagram EP2. DELETE AIX or Path: Release an Alternate Index or a VSAM Path

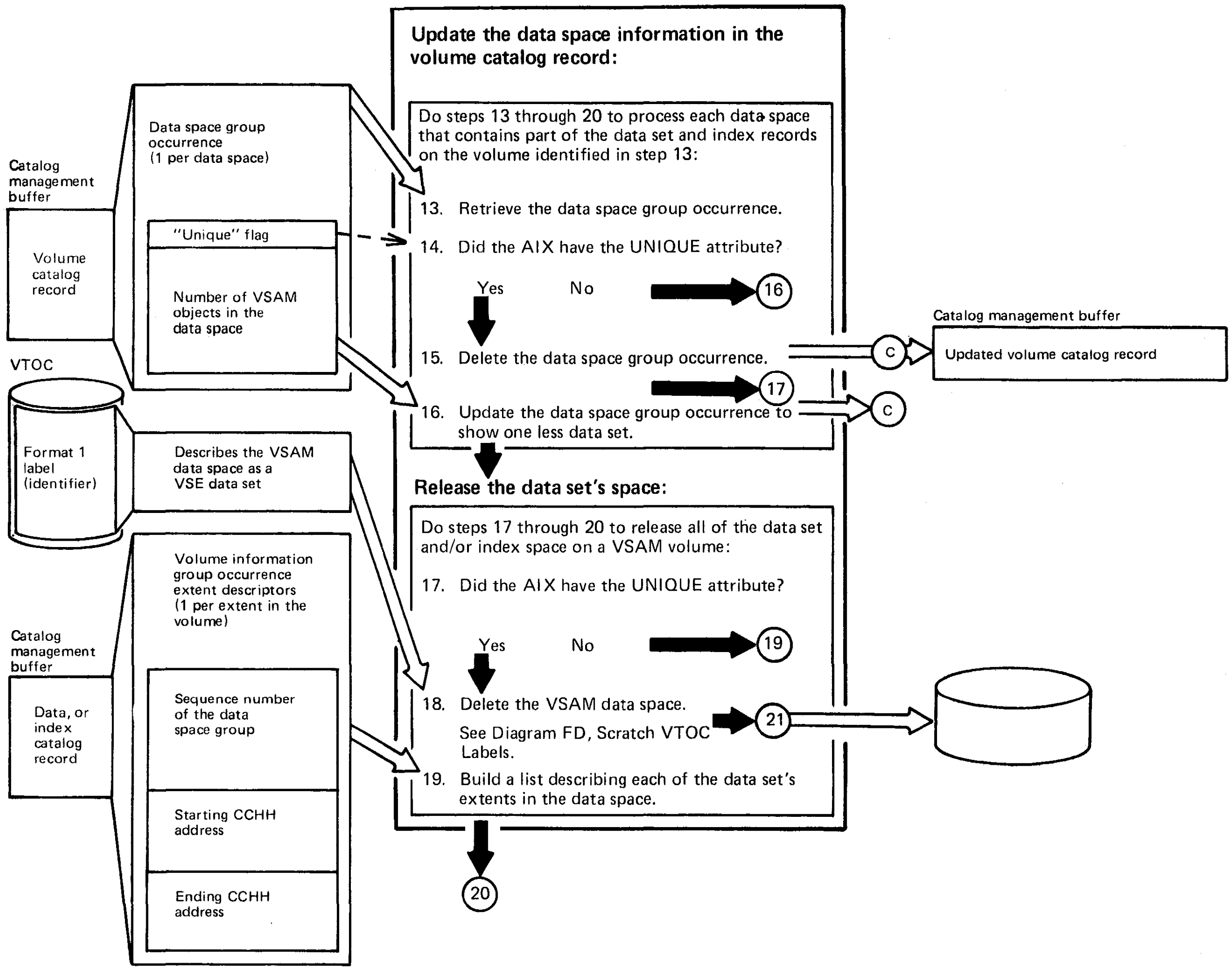


Notes for Diagram EP2

	Description	Module	Procedure
8.	The AIX association is removed from its base cluster before the AIX is actually deleted.	IGG0CLCX IGG0CLAV IGG0CLCX	IGGPDELX IGGPMOD IGGPDELX
10.	Each of the cluster data set's records is sequentially retrieved and overwritten with 0s.	IGG0CLBG IGG0CLA7 $$BOPEN IGG0CLA7 $$BCLOSE IGG0CLA7 IGG0CLBG	IGGPDLDS IGGPERAS IGGPERAS IGGPERAS IGGPDLDS
11.		IGG0CLBG IGG0CLEG IGG0CLBG	IGGPDLDS IGGPGET IGGPDLDS
12.	The volume catalog record also contains a data set directory group occurrence to describe each VSAM data set that is contained, partially or completely, on the volume. If the volume is a candidate volume for a data set or index, the data set or index is also described by a data set directory group occurrence.	IGG0CLBG IGG0CLA7 IGG0CLAZ IGG0CLA7 IGG0CLAZ	IGGPDLDS IGGPVMSC IGGPDEMV IGGPEXT IGGPDEMV IGGPVMSC IGGPDEDD IGGPEXT

Description	Module	Procedure
	IGG0CLA7 IGG0CLAV IGG0CLA7	IGGPDEDD IGGPMOD IGGPDEDD IGGPVMSC

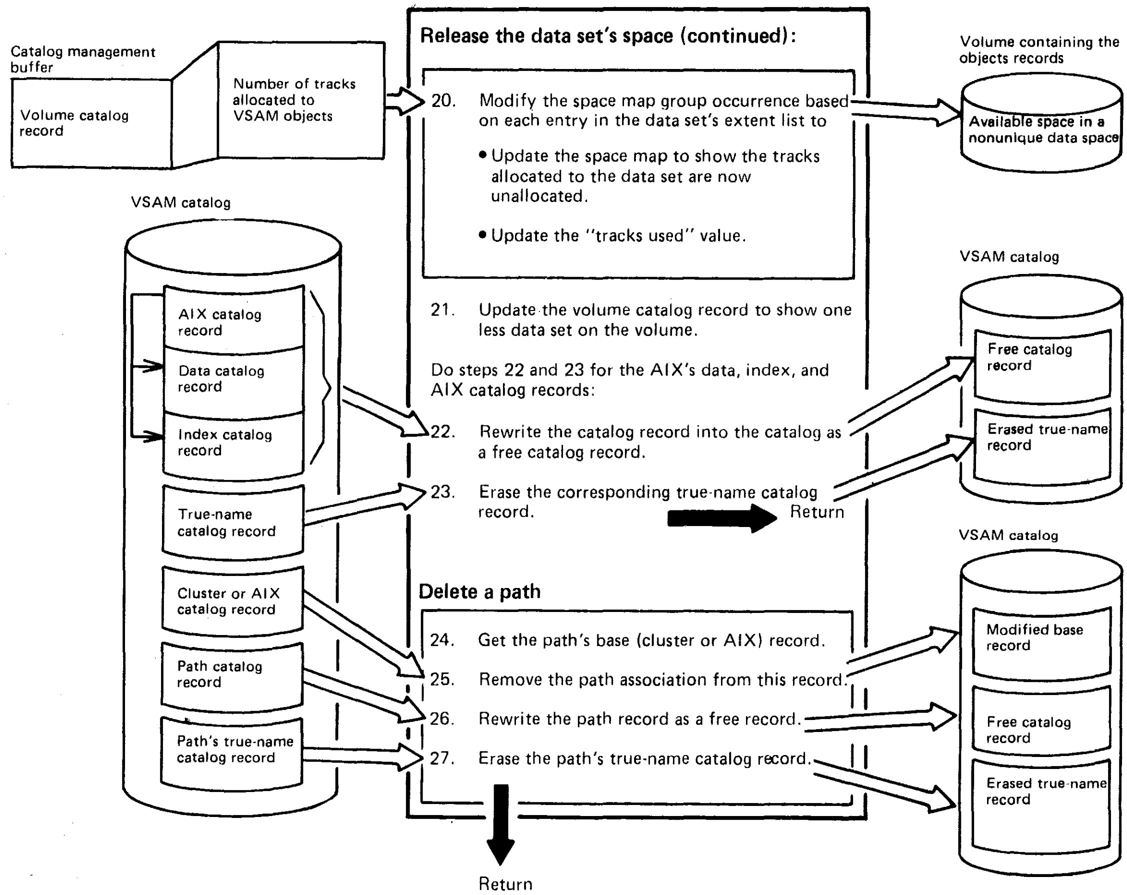
Diagram EP3. DELETE AIX or Path: Release an Alternate Index or a VSAM Path



Notes for Diagram EP3

	Description	Module	Procedure
13-16.	The volume catalog record contains a data space group occurrence to describe each VSAM data space on the volume.	IGG0CLA7	IGGPVMSC
			IGGPDESH
		IGG0CLAZ	IGGPEXT
		IGG0CLA7	IGGPDESH
		IGG0CLAV	IGGPMOD
		IGG0CLA7	IGGPDESH
			IGGPVMSC
18.	If the data set or index space is unique, its data space is also deleted. Before the data space is deleted, the volume containing it is mounted.	IGG0CLA7	IGGPVMSC
			IGGPDVMV
		IKQASNMT	
		IGG0CLA7	IGGPDVMV
			IGGPVMSC
	The volume containing the data space is specified by the FILE parameter.		IGGPDUSC
		IKQSCR00	
	The extents in the data space's format-1 (identifier) label and format-3 (extension) label are scratched from the VTOC.	IGG0CLA7	IGGPDUSC
			IGGPVMSC
19.	Each entry in the list identifies one of the data set or index extents in one of the data spaces on the volume.	IGG0CLA7	IGGPVMSC
			IGGPDEVG
		IGG0CLAZ	IGGPEXT
		IGG0CLA7	IGGPDEVG
			IGGPVMSC

Diagram EP4. DELETE AIX or Path: Release an Alternate Index or a VSAM Path



Notes for Diagram EP4

	Description	Module	Procedure
20.	Each of the data space's extents is described in the data space group occurrence.	IGG0CLA7	IGGPVMSC
		IGG0CLBF	IGGPSSCR
			IGGPESDG
		IGG0CLAZ	IGGPEXT
		IGG0CLBF	IGGPSSCR
			IGGPSCAN
			IGGPMRLC
		IGG0CLAV	IGGPMOD
		IGG0CLBF	IGGPSSCR
		IGG0CLA7	IGGPVMSC
21.		IGG0CLBG	IGGPDLDS
22-23.	The delete routine erases the data set's true-name record and deletes all references to the data set's DSNAME in the catalog's index.	IGG0CLBG	IGGPDLDS
		IGG0CLAN	IGGPDUND
			IGGPDEUN
		IGG0CLEG	IGGPGET
			IGGPDDE
	Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.	IGG0CLAN	IGGPDEUN
		IGG0CLBG	IGGPDLDS
24.	The base cluster or AIX association is extracted from the path and the base record is read.	IGG0CLCX	IGGPDELP
		IGG0CLAZ	IGGPEXT
		IGG0CLCX	IGGPDELP
		IGG0CLEG	IGGPGET
		IGG0CLCX	IGGPDELP
25.		IGG0CLCX	IGGPDELP
		IGG0CLAV	IGGPMOD
		IGG0CLCX	IGGPDELP
26-27.	Note: If the catalog is recoverable, the call to IGG0CLEG will result in a CRA update unless bit CCARPUT (the "update inhibit" bit) is set to '1'. This bit is set in the calling procedures, which decide whether CRA updating is necessary for a given operation.	IGG0CLCX	IGGPDELP
		IGG0CLEG	IGGPPDE
		IGG0CLCX	IGGPDELP

Diagram ER1. Modify the Upgrade Set

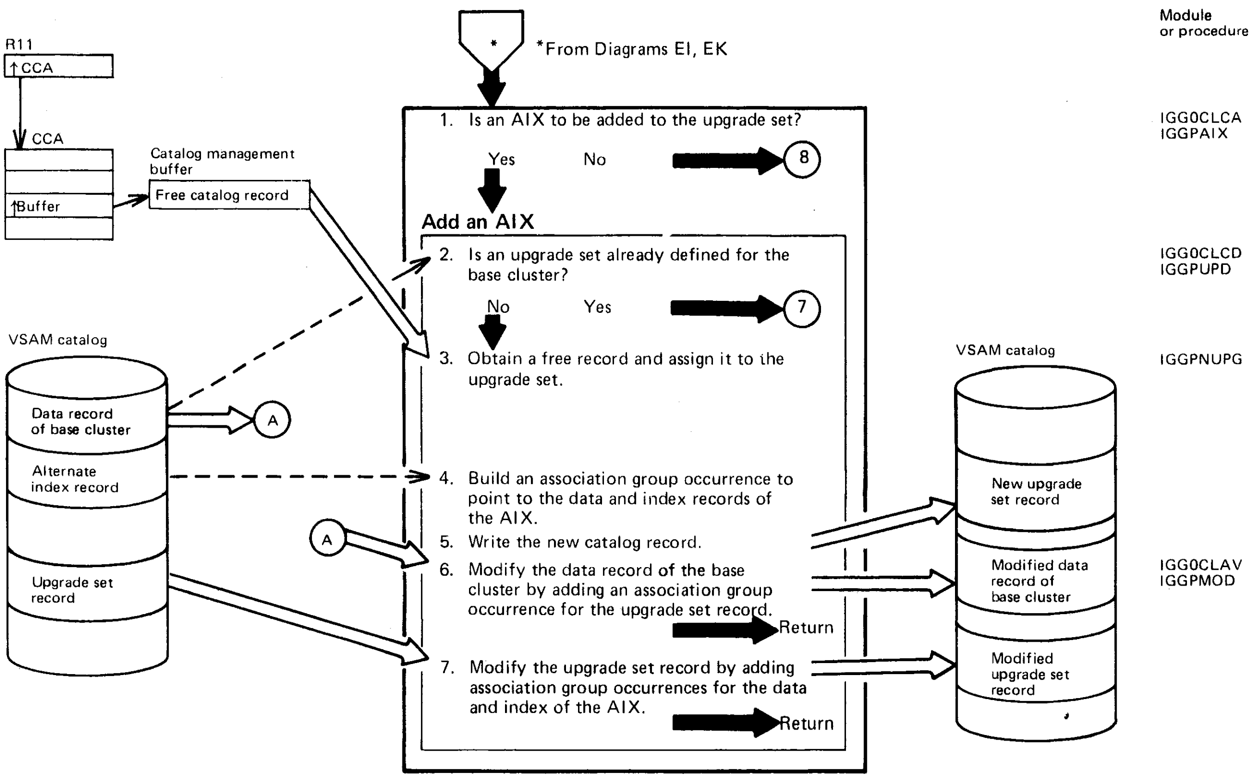


Diagram ER2. Modify the Upgrade Set

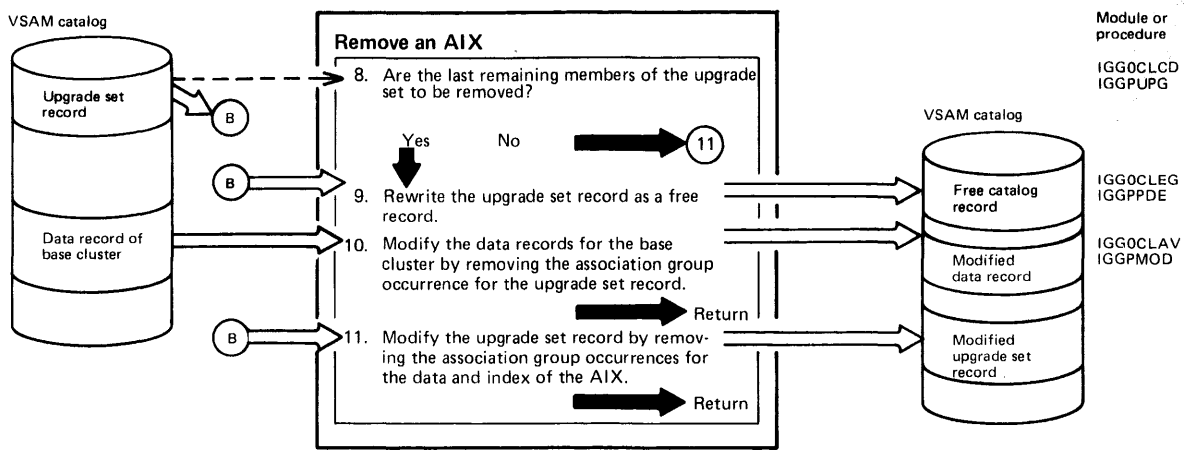
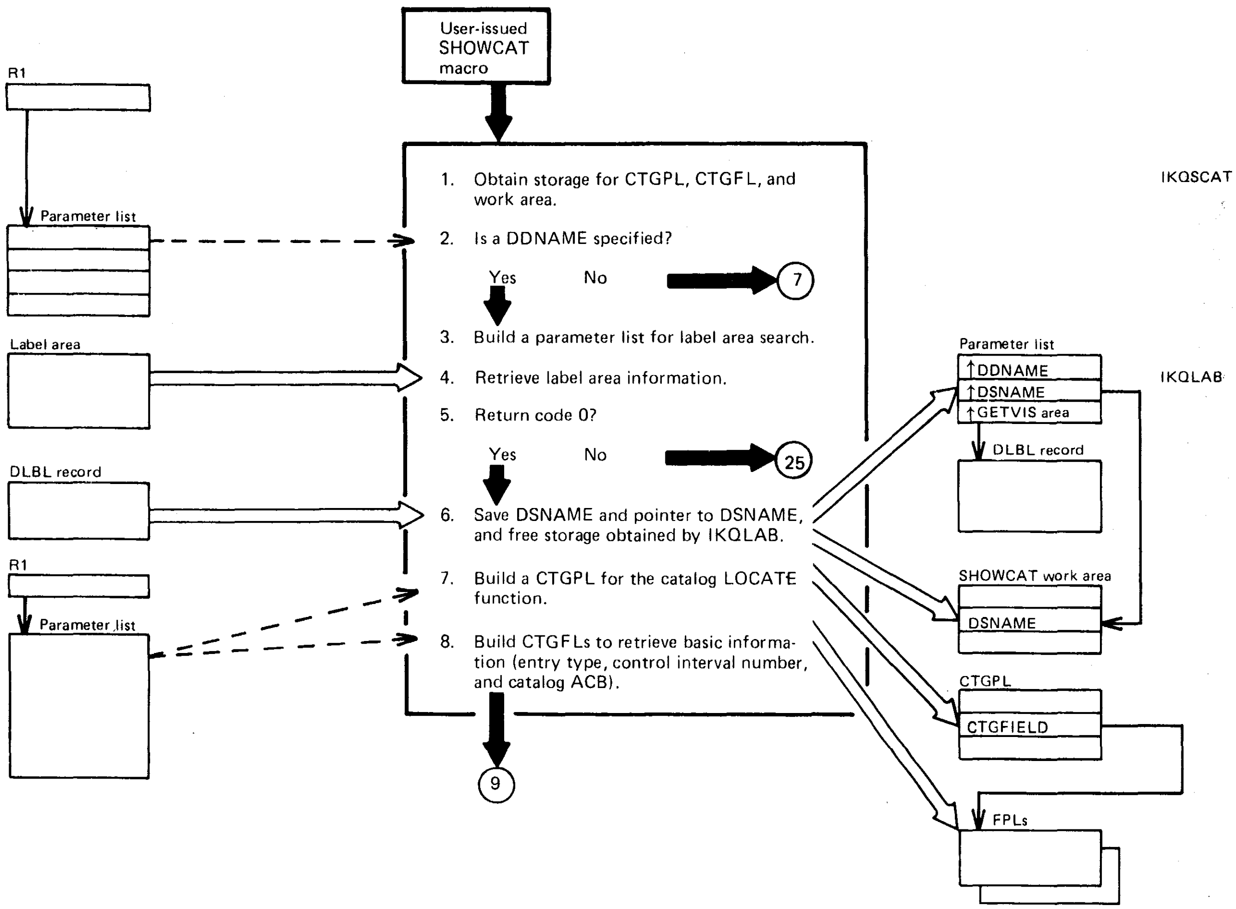


Diagram ES1. SHOWCAT: Display Catalog Information

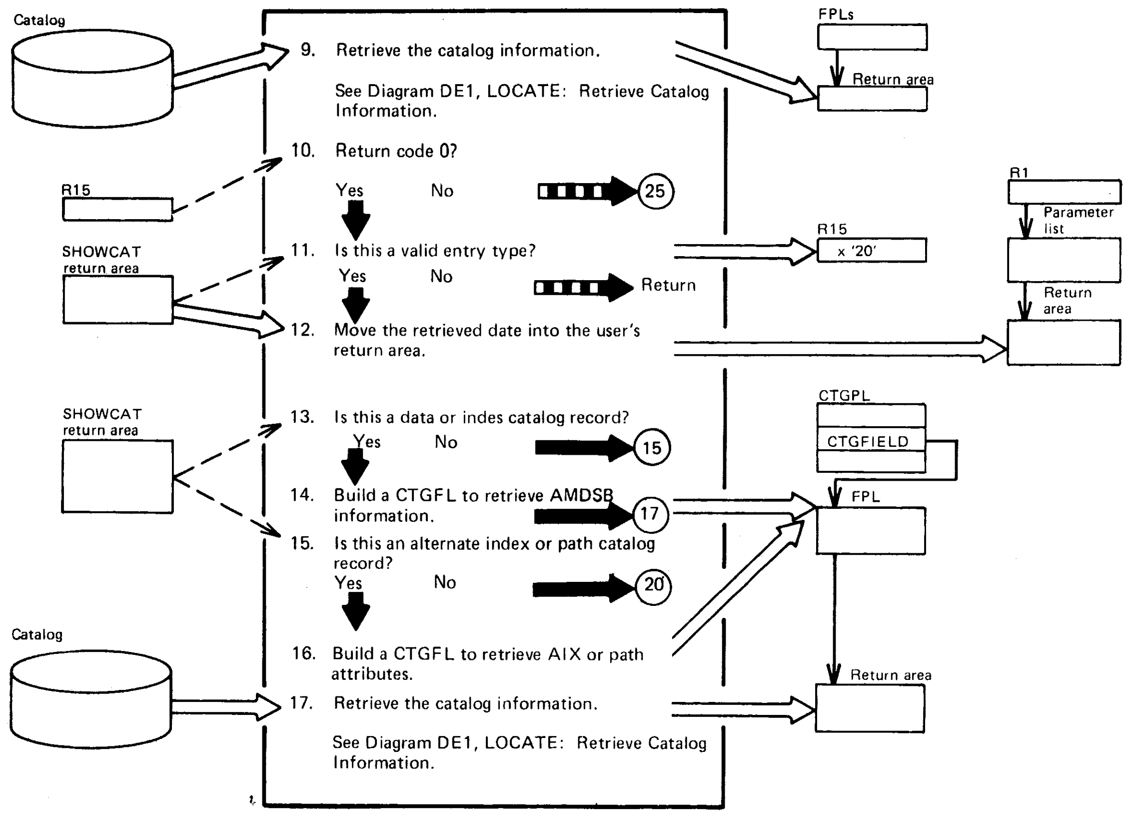


Notes for Diagram ES1

1. 256 bytes of storage are needed.

8. If the CI number was specified, the contents of catalog fields ENTYPE and CATACB are retrieved. If the true-name was specified, the field DSTYPNAM must be retrieved instead of ENTYPE, in order to find the CI number.

Diagram ES2. SHOWCAT: Display Catalog Information

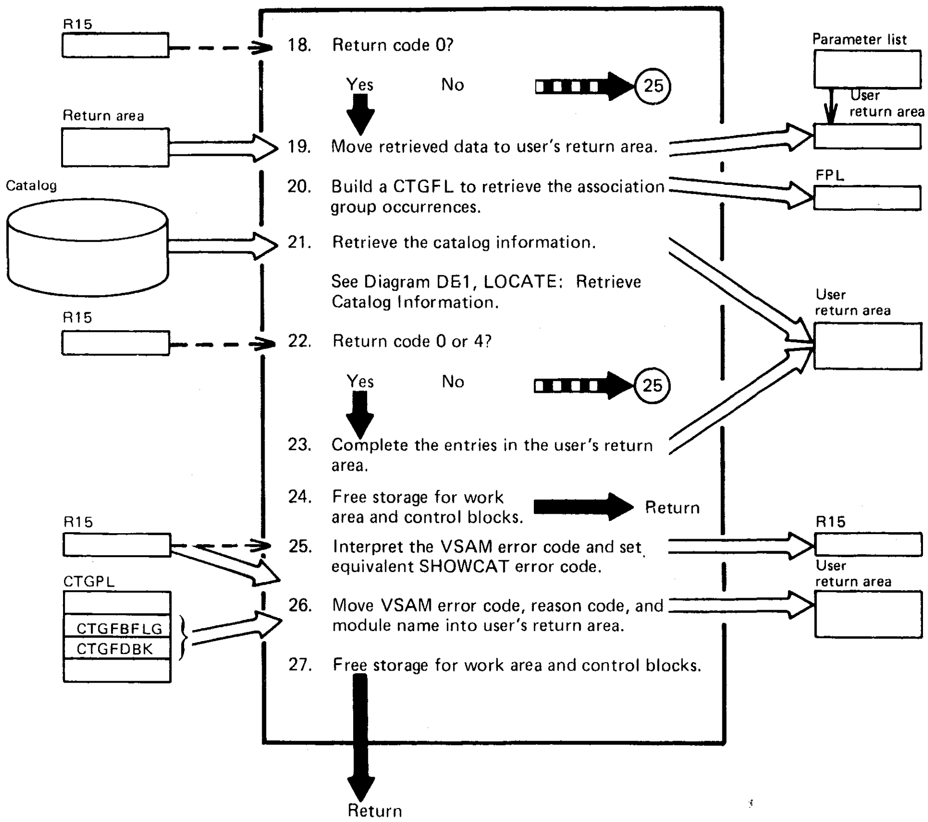


Notes for Diagram ES2

11. Only the following catalog record types may be retrieved with the SHOWCAT macro:

    Cluster
    Data
    Index
    Alternate index
    Path
    Upgrade set

Diagram ES3. SHOWCAT: Display Catalog Information



Notes for Diagram ES3

22. Return code 4 indicates that the return area was too small to accept all group occurrences; the information retrieved (as much as would fit in the return area) is passed to the user.

23. Information saved before the LOCATE is restored and the length counts are updated.

Diagram FA1. DADSM Contents

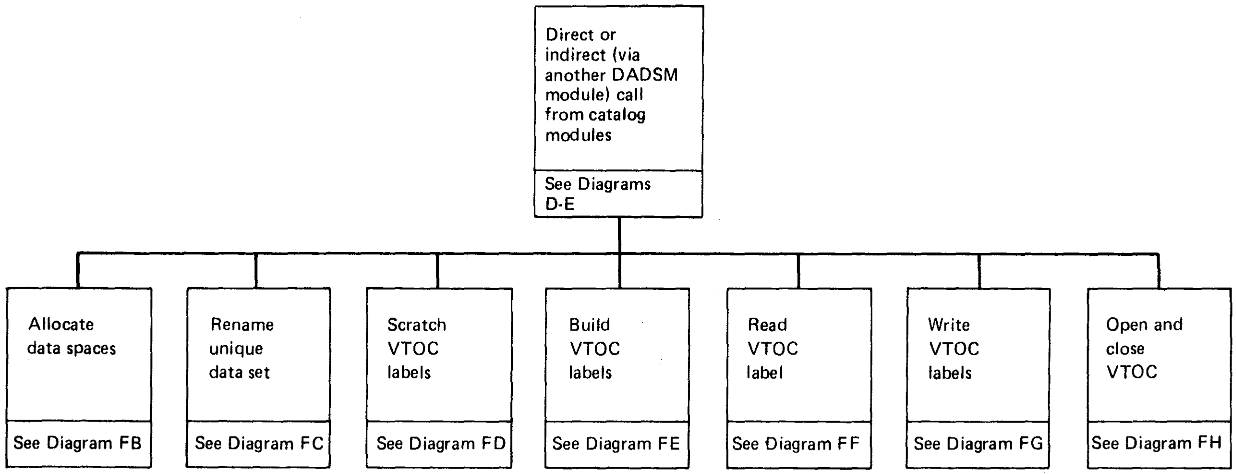
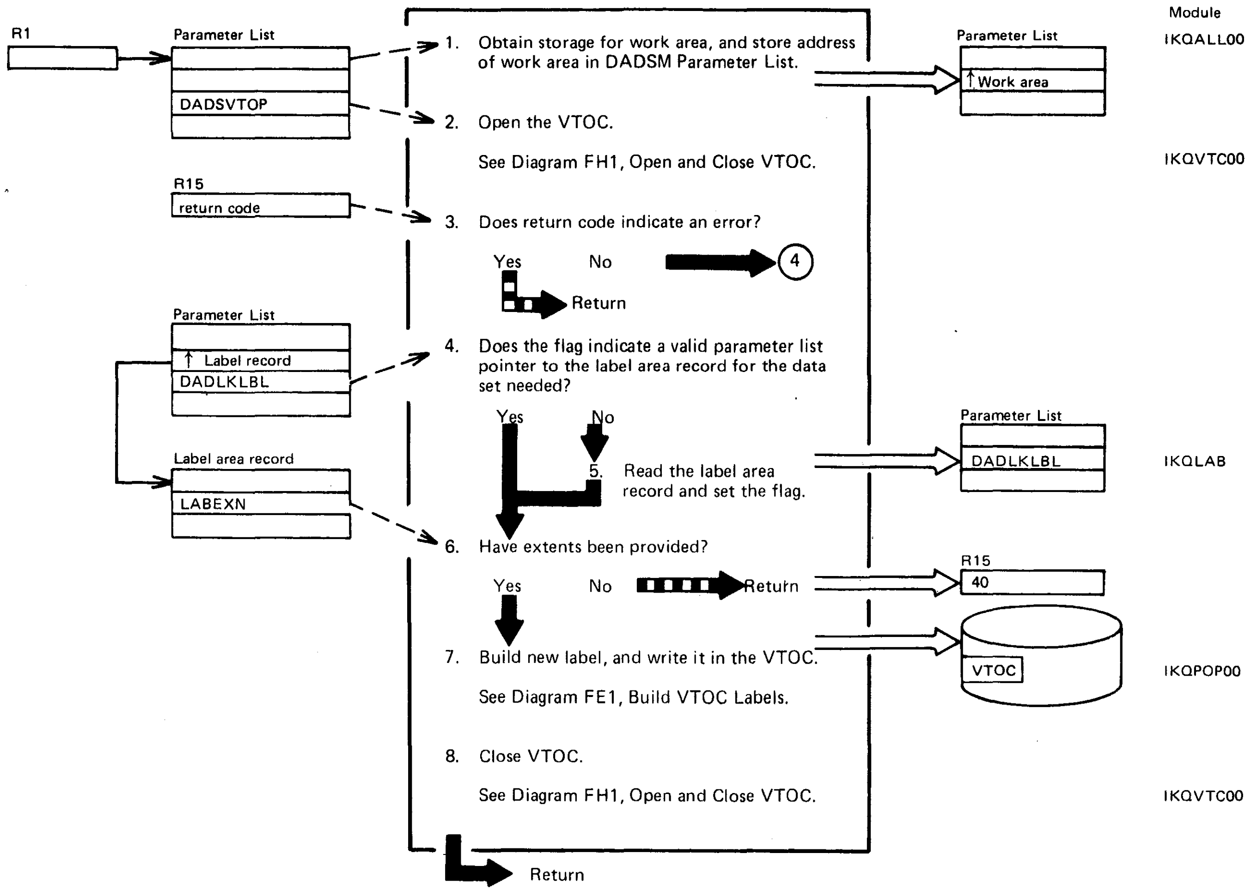


Diagram FB1. Allocate Data Spaces



Notes for Diagram FB1

2. A JIB must be built if the file protect feature is present.

7. After the new label is built, the DADSEXIT field in the DADSM Parameter List is checked. If this field contains an address of a DADSM exit routine, that exit is taken.

Diagram FC1. Rename Unique Data Set

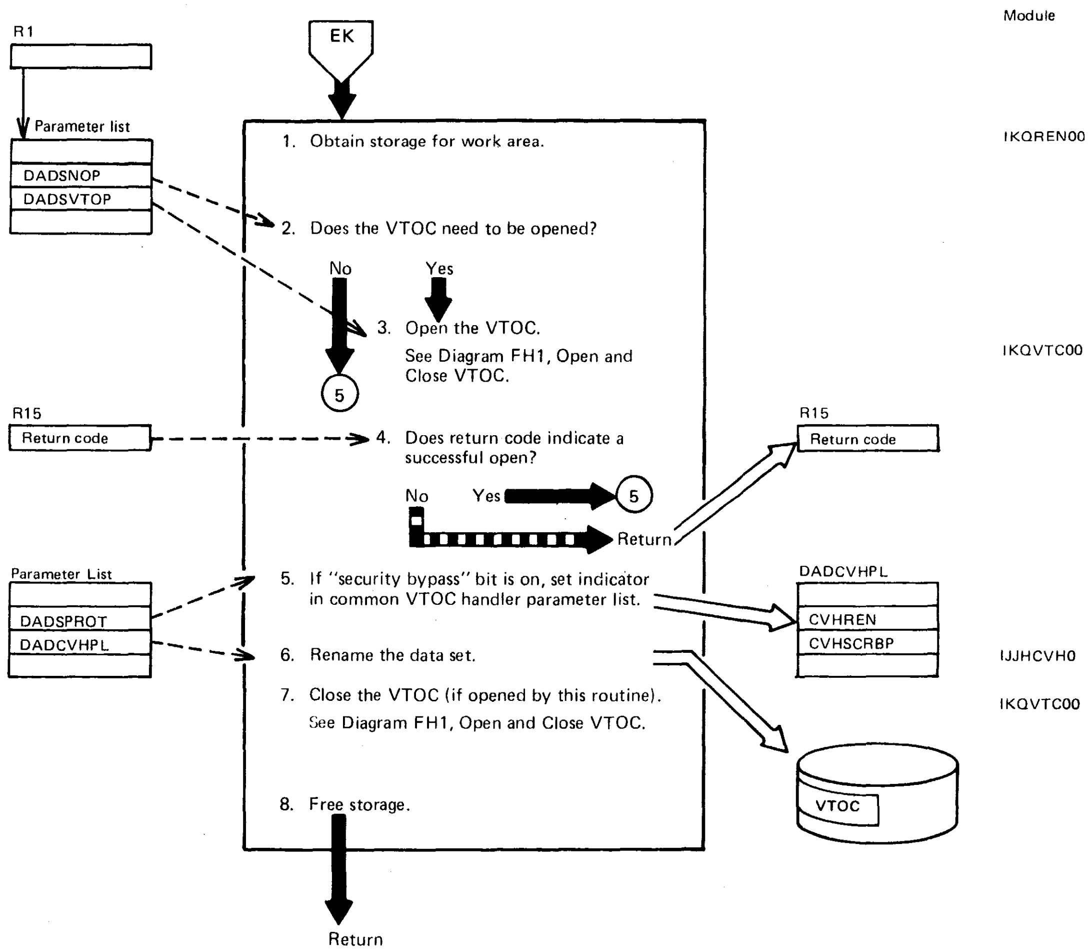


Diagram FD1. Scratch VTOC Labels

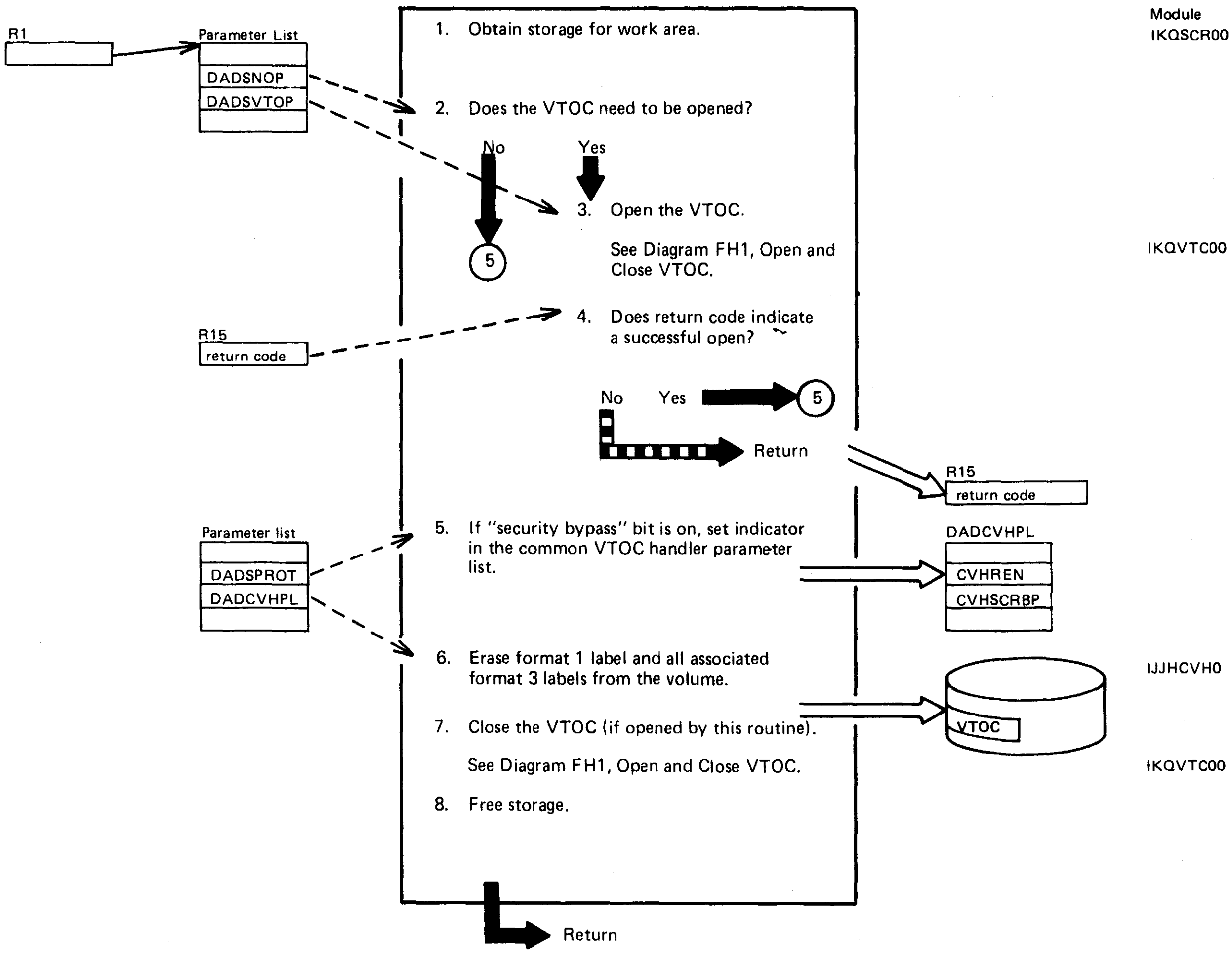


Diagram FE1. Build VTOC Labels

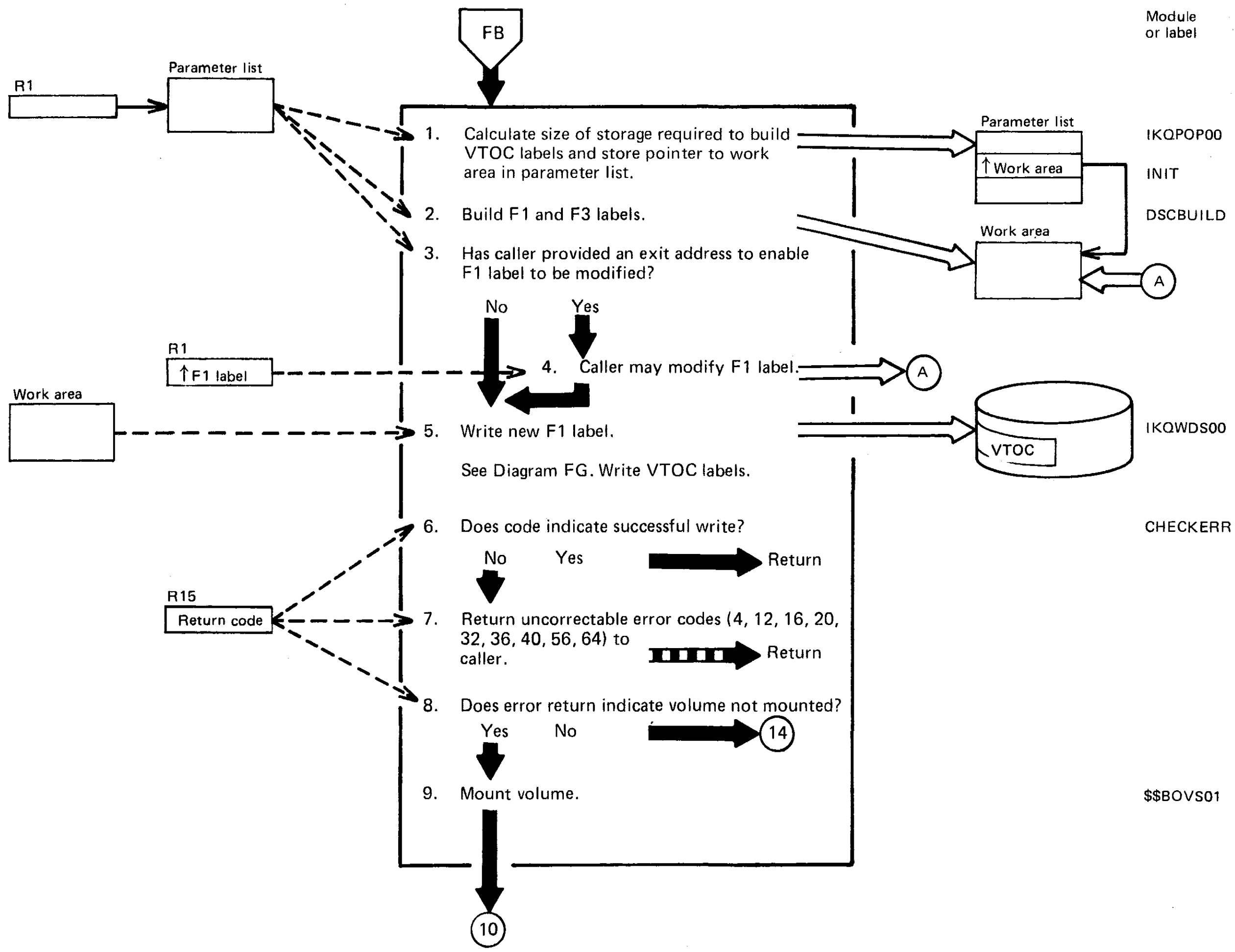


Diagram FE2. Build VTOC Labels

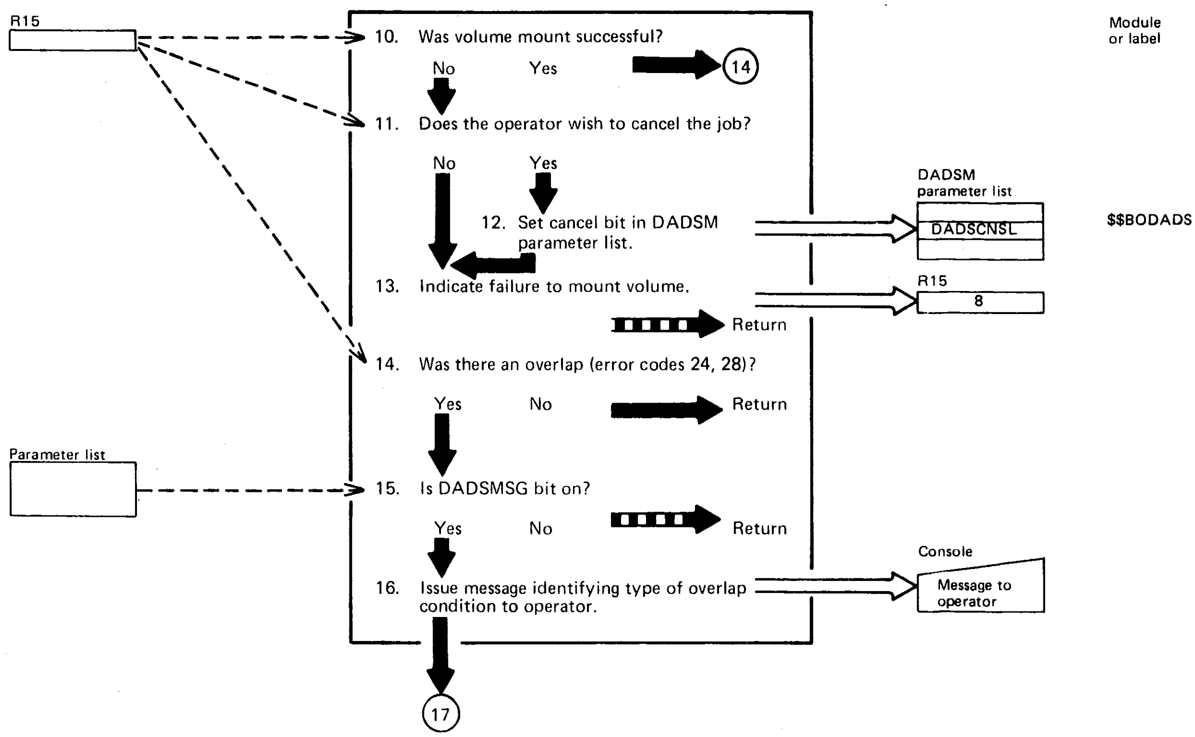


Diagram FE3. Build VTOC Labels

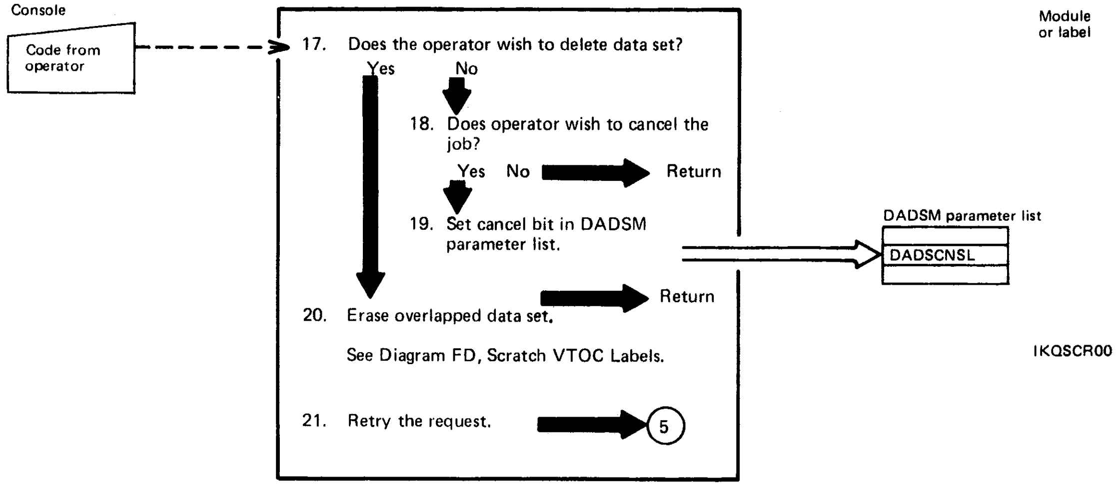


Diagram FF1. Read VTOC Label

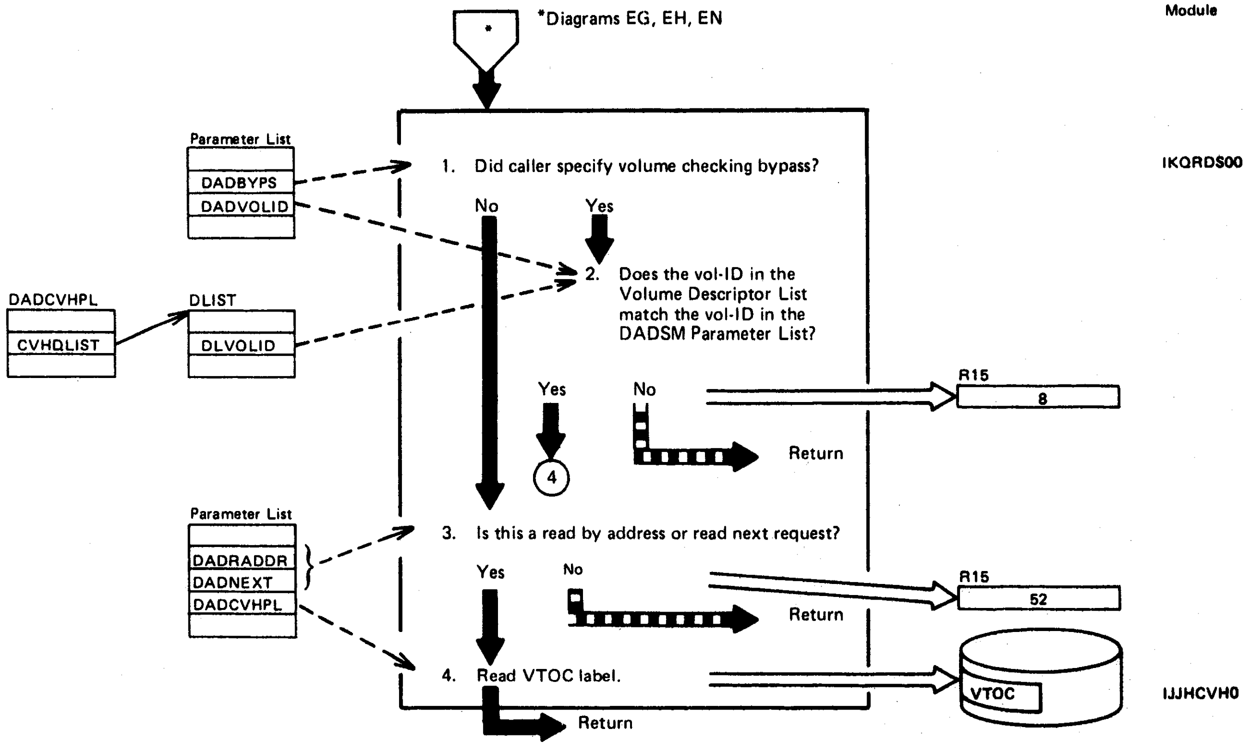


Diagram FG1. Write VTOC Labels

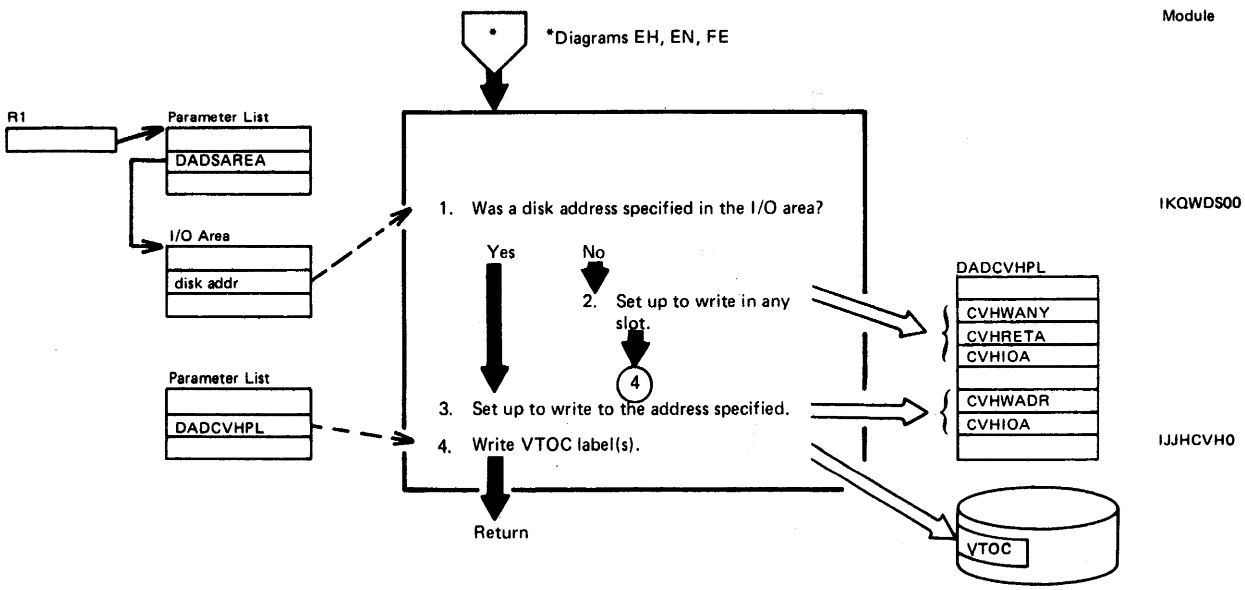


Diagram FH1. Open and Close VTOC

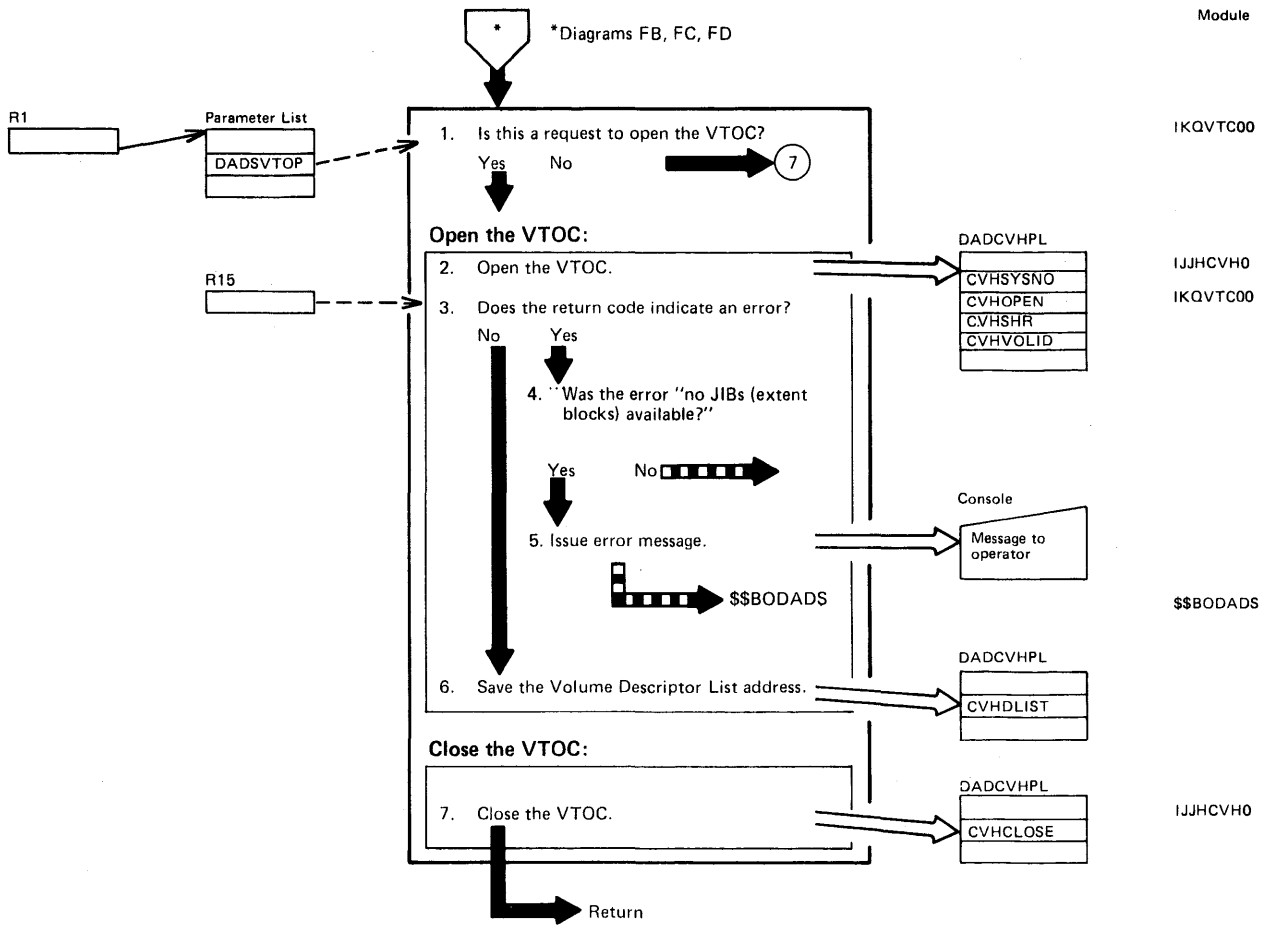
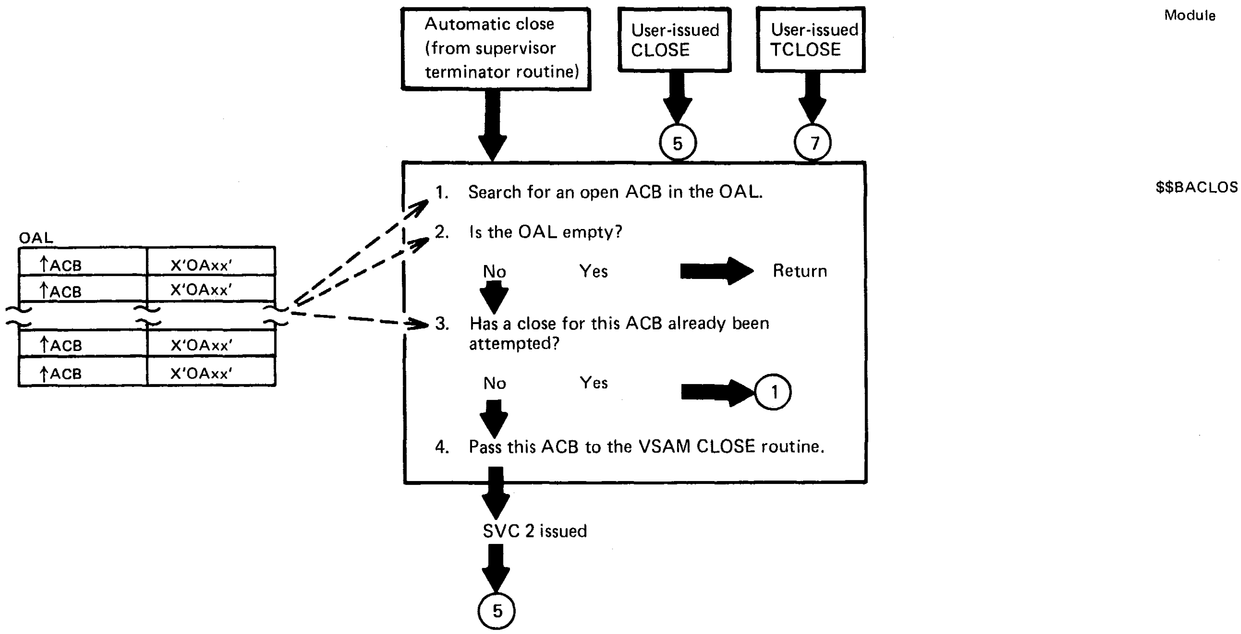


Diagram GA1. CLOSE or TCLOSE: Disconnect a User's Program from a VSAM Data Set



Notes for Diagram GA1

Note: When the user program issues a CLOSE or TCLOSE macro against an ACB or a DTF, or the user program SYNAD routine issues a CLOSE macro against an ACB, or the VSE end-of-job routines initiate an automatic close, an SVC 2 is generated.

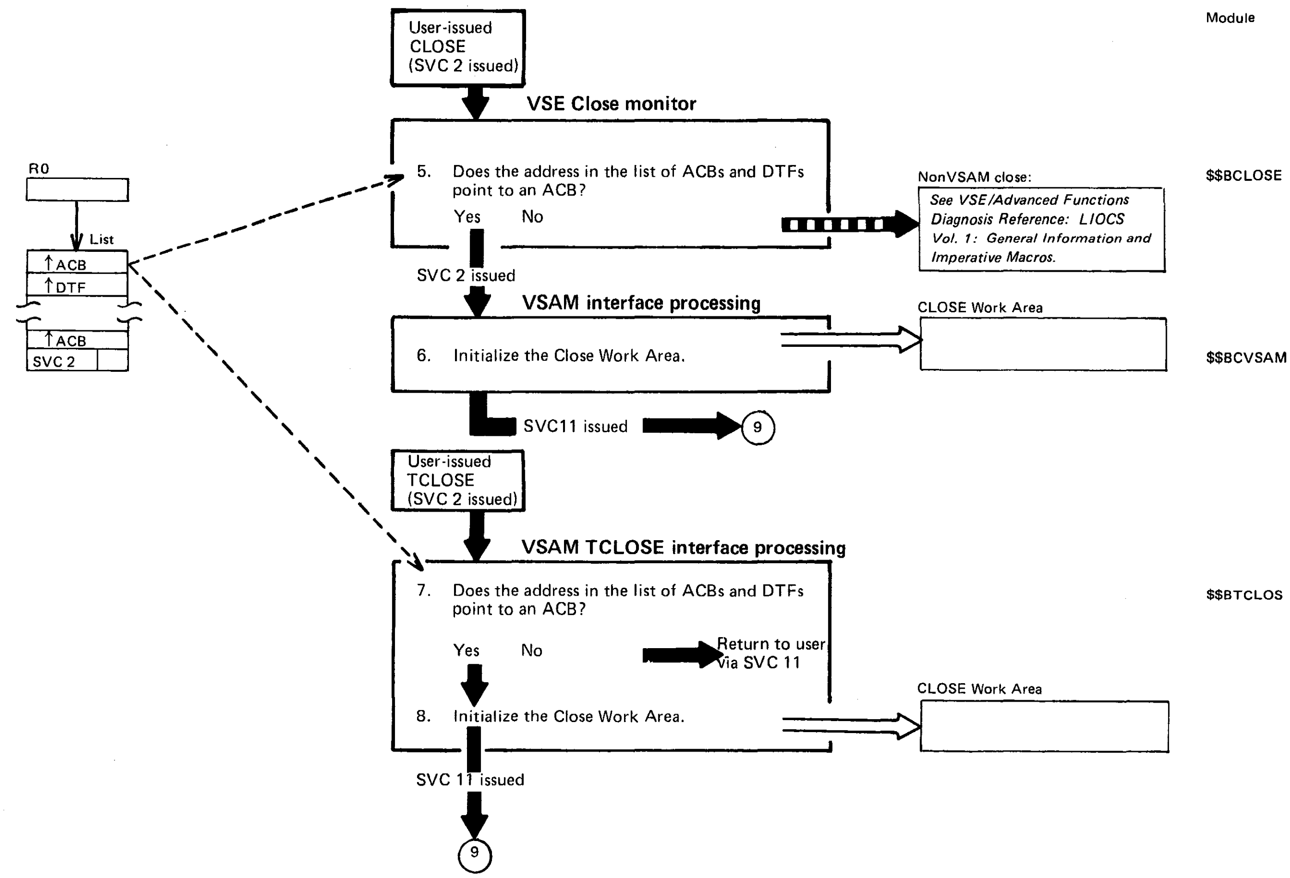
1. $$BACLOS is called by the supervisor during end-of-job processing to initiate automatic closing of ACBs that were not closed by the user's program.

2. If the OAL is empty or if automatic close has been disabled by using IKQVEDA, $$BACLOS exits to the supervisor terminator routine via SVC2 for $$BEOJ4.

3. A flag in the OAL entry indicates whether an attempt has already been made to close this ACB. If this is the case, the ACB is skipped, to avoid recurring attempts to close the same ACB, which would lead to a program loop.

4. The ACB is passed to the close routine as an ACB list, containing only one ACB.

Diagram GA2. CLOSE or TCLOSE: Disconnect a User's Program from a VSAM Data Set



Notes for Diagram GA2

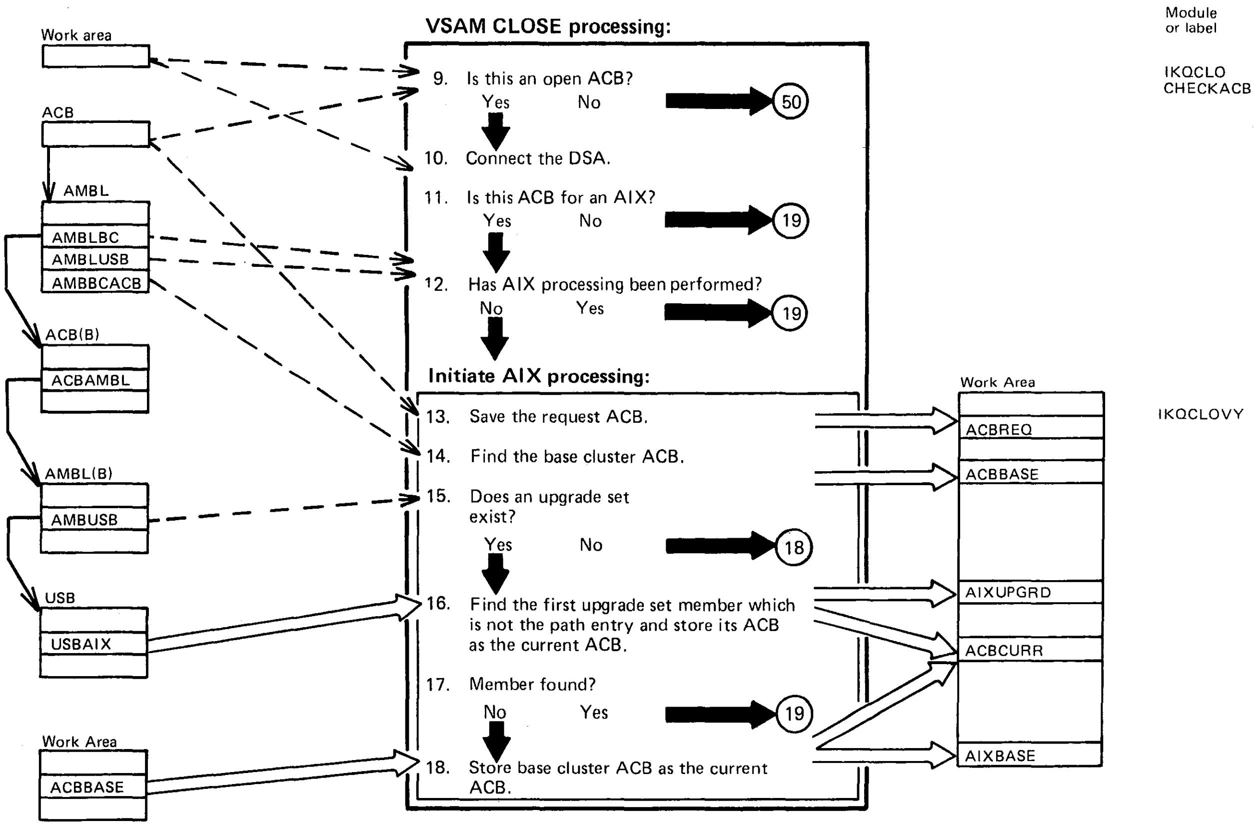
5. The VSE Close Monitor examines the DTF-type field (offset 20 from the address passed in the list) of the ACB or DTF. If the byte indicates an ACB (X'28'), an SVC 2 is issued and $$BCVSAM is fetched into the B-transient area. The list may consist of all DTFs, all ACBs, or a mixture. It is passed to the VSE Close Monitor by the user program via a pointer in register 0.

6. The VSAM interface module obtains and initializes a work area in which it sets a flag to indicate a CLOSE macro was issued. Pointers are saved to the current list entry and the VSE communication region in the work area. It copies the user's PSW and registers into the work area. It loads the VSAM Close module and then issues an SVC 11 to branch to it.

7-8. If the user program issued a TCLOSE (temporary close) macro against an ACB, the temporary close module is fetched into the B-transient area.

The VSAM TCLOSE interface module examines the DTF-type field (offset 20 of the address passed in the list) of the ACB or DTF. If the byte indicates an ACB (X'28') and the ACB indicates that deferred write operations are not possible, this module obtains and initializes a work area in which it sets a flag to indicate a TCLOSE macro was issued. Pointers are saved to the current list entry, and the system communication region in the work area. It copies the user's PSW and registers into the work area. It loads the VSAM Close module and then issues an SVC 11 to branch to it.

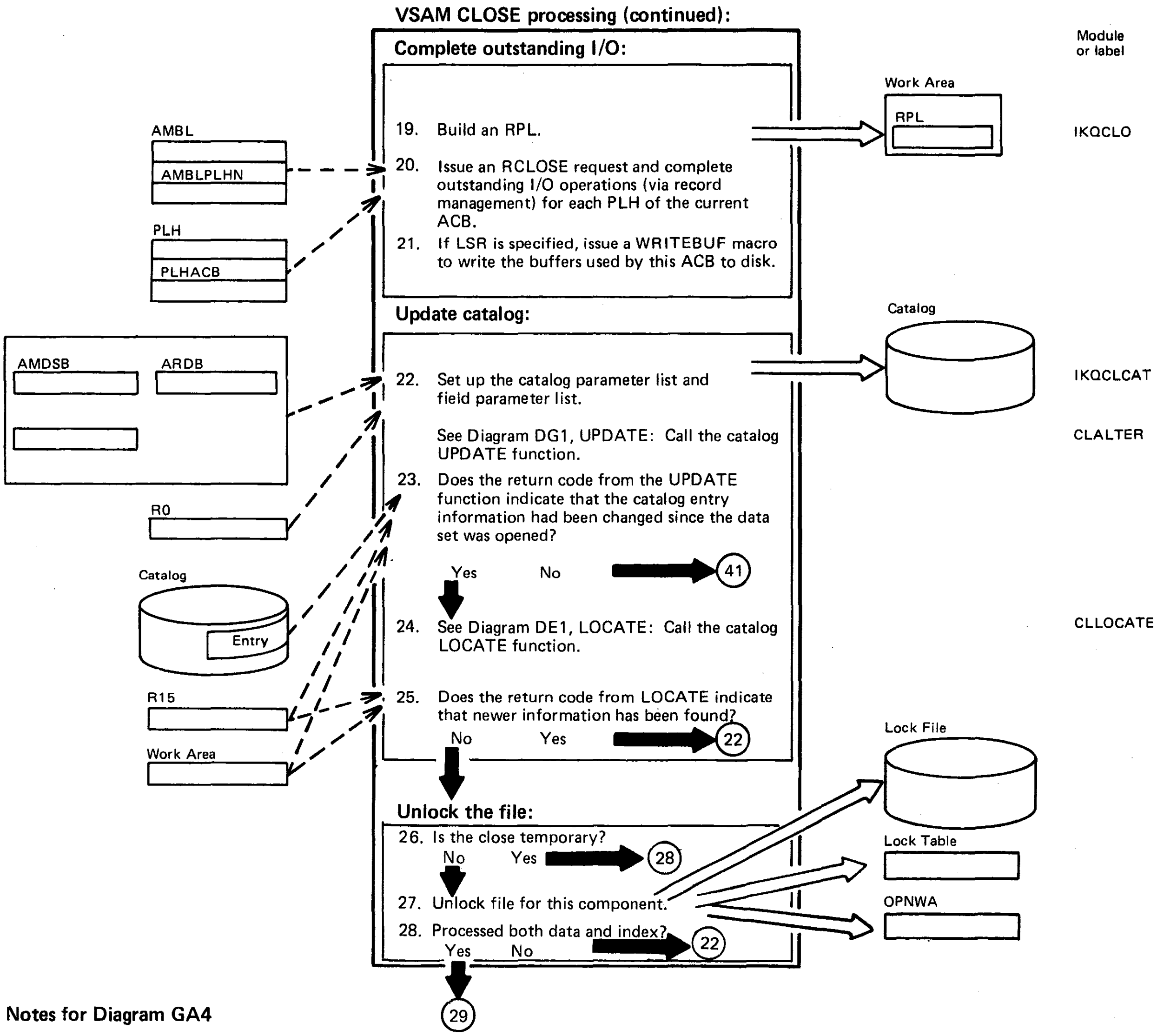
Diagram GA3. CLOSE or TCLOSE: Disconnect a User's Program from a VSAM Data Set



Notes for Diagram GA3

9. The ACB identifier field is checked for an X'A0'. The ACB open flag is also checked. If the ID is incorrectly specified or the open flag is off, an error code is set in the work area.

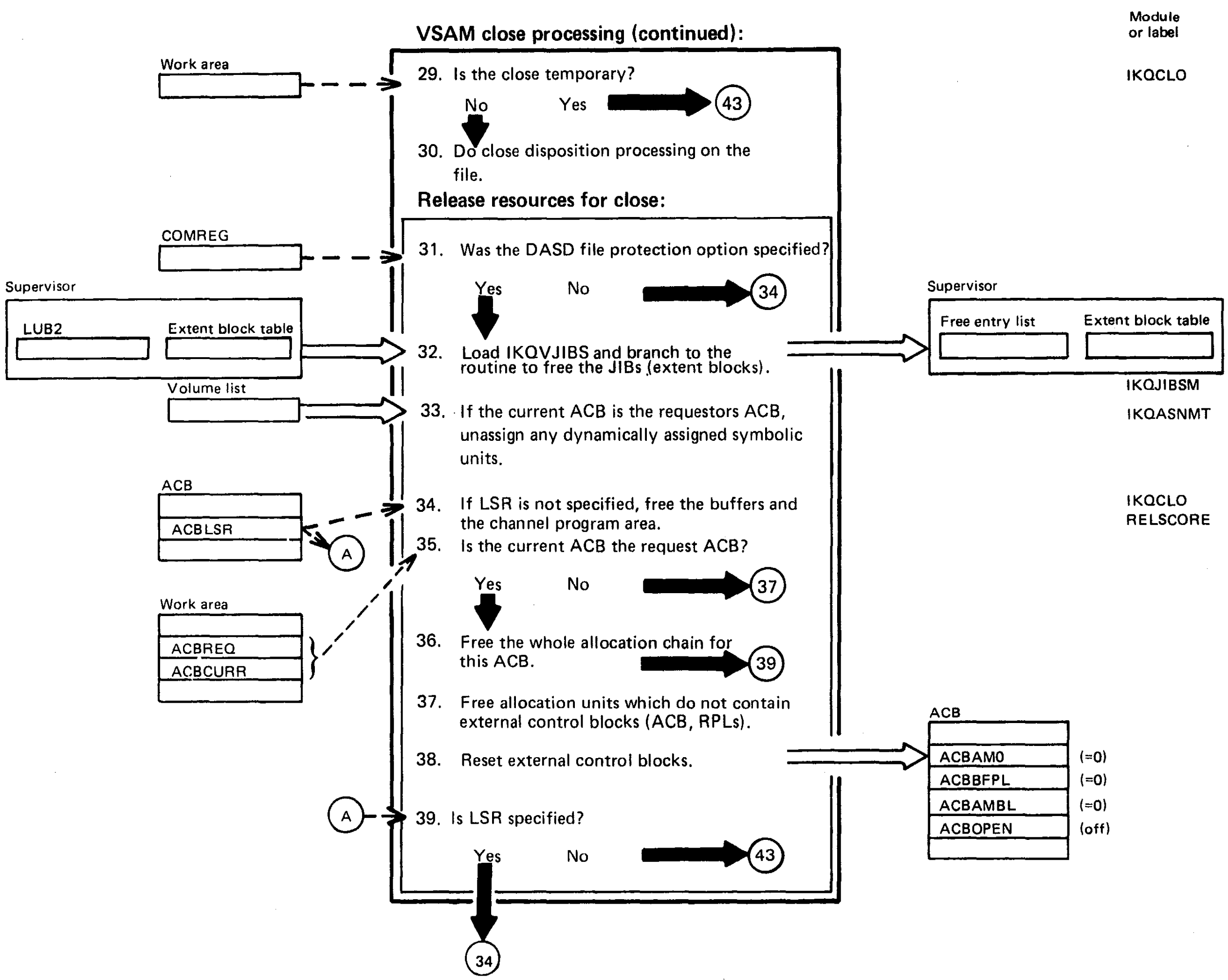
Diagram GA4. CLOSE or TCLOSE: Disconnect a User's Program from a VSAM Data Set



Notes for Diagram GA4

	Description	Module	Routine
19.	BUFMGT issues an RCLOSE to IKQVSM for each string. Dummy RPLs are built and passed by BUFMGT to IKQVSM, which then passes control to IKQRCL00.	IKQCLO IKQVSM	BUFMGT
	If load mode with speed option has been specified by the user's program (which means that a control interval is written to the disk only when the control interval has been filled with logical records), the remainder of the control area must be formatted and an SEOF record (all 0s) placed at the end of the area.	IKQRCL00 IKQPFO00	
	For a key-range data set, IKQCIS00 formats all remaining unformatted key ranges.	IKQCIS00	
	Note: In recovery mode, each control area is formatted with empty control intervals, and the control area is terminated with an SEOF prior to loading any records into the control area. Hence, in this case, it is not necessary to call IKQPFO00.		
	IKQBFA00 is always entered from IKQRCL00 to complete outstanding I/O requests. All current read operations are finished and all pending update or write operations are done. If an error occurs while I/O is being completed, IKQVSM returns a nonzero error code in register 15, and BUFMGT sets an error code in the work area.	IKQBFA00	
22.	If R0=0, IKQCLCAT has been called by Close; if R0=4, is has been called by Record Management.	IKQCLCAT	
23.	To update the permanent data set information in the catalog, Close utilizes the work area for the catalog parameter and field parameter lists.	IKQCLCAT	CLCATLG CLALTER
23-24.	If the information in the catalog entry had been changed since the data set had been opened (that is, between Open and Close another user had processed the data set and information had been altered), the latest copy of the entry must be located. If the data set has an index, repeat steps 22-28 for the index.	IKQCLCAT	CLCATLG CLLOCATE

Diagram GA5. CLOSE or TCLOSE: Disconnect a User's Program from a VSAM Data Set



Notes for Diagram GA5

	Description	Module	Routine
29.	A temporary Close does not release the JIB extents.		
30.	DISP = KEEP (specified explicitly or defaulted) or DISP = DATE when the file is not expired require no action. Otherwise, delete processing is done depending on file characteristics:	IKQCLRDD	
	• NOALLOCATION files → Release extents	IKQCLRDD	RELEASE
	See Diagram DM; Release Extents		
	• Implicit define files → Implicit delete	IKQOCIMR	
	• Other reusable files → Reset file to empty	IKQCLRDD	
32.	If the DASD file-protection option has been specified, the extent blocks for each extent of the data set are freed. The IKQJIBSM routine is called to release the extents. This processing is repeated for all extents. If the data set has an index, this step processes the index also.		

	Description	Module	Routine
34.	Storage obtained by Open and/or end-of-volume for LPMBs (other than the first LPMB), EDBs (other than the first EDB), ARDBs, BCBs, and buffers is released. If the data set has an index, this step processes the index also.		
35-37.	Allocation units containing the user's external control blocks may not be released. The whole allocation chain may be released if the first allocation unit contains only internal control blocks (AMBL allocation chain) or if the external control blocks were created by OPEN (base cluster ACB, upgrade set members ACBs, etc.).		

Diagram GA6. CLOSE or TCLOSE: Disconnect a User's Program from a VSAM Data Set

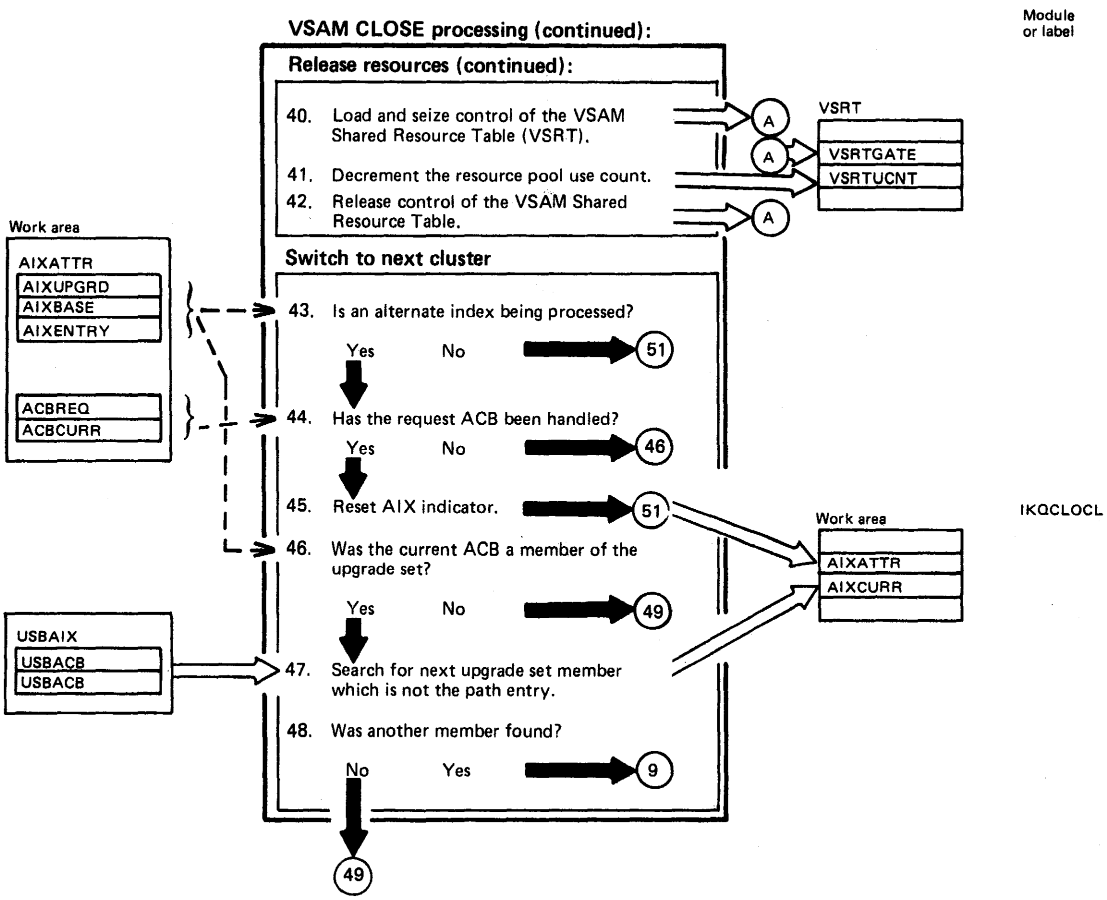
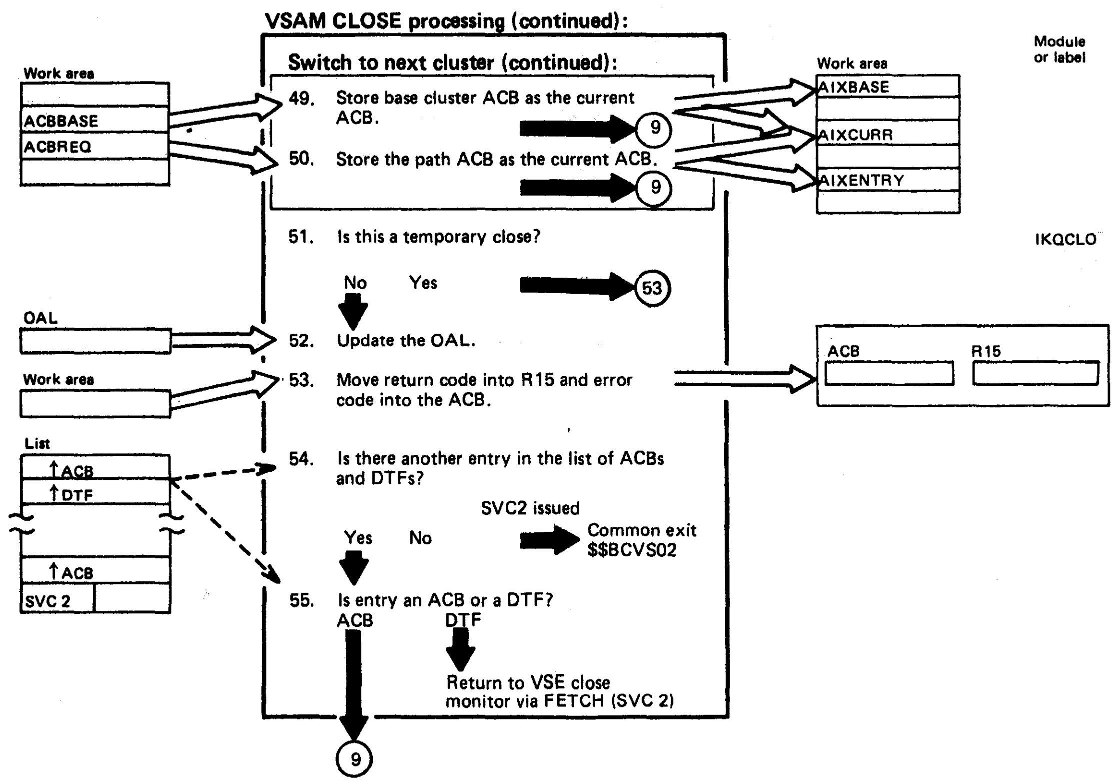


Diagram GA7. CLOSE or TCLOSE: Disconnect a User's Program from a VSAM Data Set



Notes for Diagram GA7

52. The OAL entry for this data set is set "inactive" and the count of open ACBs in the OAL is decremented.

53. The work area error code field is checked, and the error return code is moved into the ACB error flag field. An error indicator is also set in register 15 of the user's register save area.

54. If there are no more entries in the list, control is passed to $$BCVS02, the VSAM common exit module, via an SVC 2.

55. For a normal (user-issued) CLOSE or TCLOSE, $$BCVS02 returns control to the user. For an automatic close, it returns control to $$BACLOS.

Section 3. Program Organization

VSAM program listings are the key to VSAM's organization. You get into the listings from the method of operation diagrams. Once you have located the module or routine name that interests you in the diagrams, you are ready to turn to the listing to find the additional information you require.

Module Prologues

Each VSAM module listing begins with a description of the module, called the module prologue. The information contained in VSAM prologues is described in the topics that follow.

Module name: The external procedure name of the module (for example, IKQOPN).

Descriptive name: The English name of the module (for example, VSAM Open).

Status: The version and release level of the module.

Function: A brief step-by-step explanation of the functions performed by this module. Function is divided into steps so that you may more easily locate the routine responsible for each step.

Notes: A generalized heading that includes (1) any dependencies, for example, CPU model or features, that will affect the operation of this module, (2) any restrictions that apply to this module, (3) symbols used to represent registers and register usage, (4) symbolic name of the maintenance area for this module and whether the maintenance area is used or reserved, and (5) any special terms and acronyms that are used within this module that are not necessarily used elsewhere in the documentation.

Module Type: A description of the type of this module (for example, procedure or macro) the name of the compiler used/required to create this module, the amount of storage required by this module for executable code and associated data, and the attributes of the module (for example, reentrant or read-only).

Entry point: The name of the point at which control can enter this module, the conditions of entry, the calling sequence by which control was given, including any parameters passed and the names of modules that may enter at this entry point.

Input: A description of anything this module gets or references, such as registers, control blocks, or data. The means by which this module gains access to the input is included.

Output: A description of registers, control blocks, and data areas at output; any messages issued as a result of this module's processing are included.

Exit-normal: A description of conditions at and reasons for normal exit from this module, including the names of modules called by this module.

Exit-error: A description of conditions at and reasons for any error exit from this module.

External references: A list of modules, data areas, etc., defined outside of or accessible outside of this module.

Tables: A list of all local tables and work areas, that is, data areas built and used only within this module.

Macros: A description of system macros used by this module.

Change activity: A list of any change activity to this module.

Routine Prologues

The numbered steps in the module prologue FUNCTION heading are your link to the routine prologues. Routine prologues contain (1) an expanded description of the processing steps shown in the module prologues, (2) input to the routine, and (3) output from the routine.

Program Structures and Catalog Program Flowcharts

The following group of program structures and, for the catalog modules, program flowcharts, shows how the VSAM program is organized. These structures link modules together from the time a macro instruction is issued by the user program to the time that control exits from VSAM. The structures are ordered by user-issued macro instructions and the verify function in a way similar to the organization of method of operation diagrams. In addition, program structures are also shown for significant subfunctions required to complete processing of a macro instruction. The subfunctions included in this volume are the ISAM interface, open/close, catalog management, and DADSM.

The flowcharts are arranged in alphabetical order according to the last two alphameric characters of the module name. The title of the flowchart also has a number, in the third position, which is the page number within the flowchart. Module IGG0CLAF is thus shown on two pages — Chart AF1 and Chart AF2. Off-page connectors between the pages contain the page number and the block location. For example, the off-page connector at block CI in chart AF1 contains "02 J1", which refers to block J1 on chart AF2. As the flowcharts are intended to show the calling sequence rather than the internal logic, not all procedures are documented. Only those procedures which call other procedures are shown. This leads to two different types of cross-references in the flowcharts. Look, for example, at Chart AF2, blocks H1 and H2. In H1, the procedure IGGPF4WR is called, with the cross-reference BU1A2. This means that the procedure is located in module IGG0CLBU and documented in Chart BU1, starting at block A2. In H2, in contrast, the cross-reference for procedure IGGPDLER is simply AF. This means that the procedure is located in module IGG0CLAF, but is not documented.

Figure 3.1 shows the symbols used on the structures and describes their meanings.

————————	Indicates that a module is called and returns to calling module
————————►	Indicates that a module does not return to calling module
— — — — — —	Indicates that a module is called under certain conditions and then returns to calling module
— — — — — — ►	Indicates that a module is called under certain conditions and does not return to calling module
UPPER CASE	Indicates that a module is executed and calls one or more modules before returning
lower case	Indicates that a module is executed and then returns to the calling module

Figure 3.1 **Graphic symbols used in program structures**

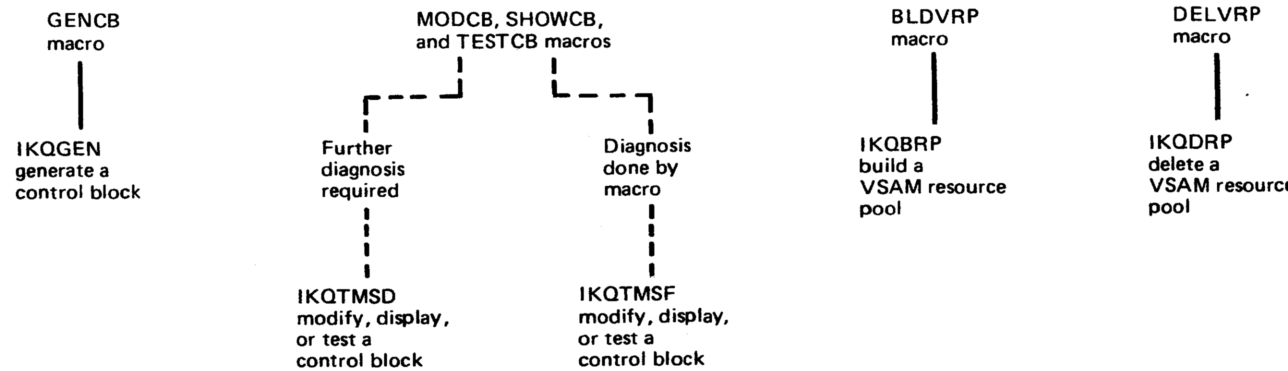


Figure 3.2 Program structure to process control block manipulation macros

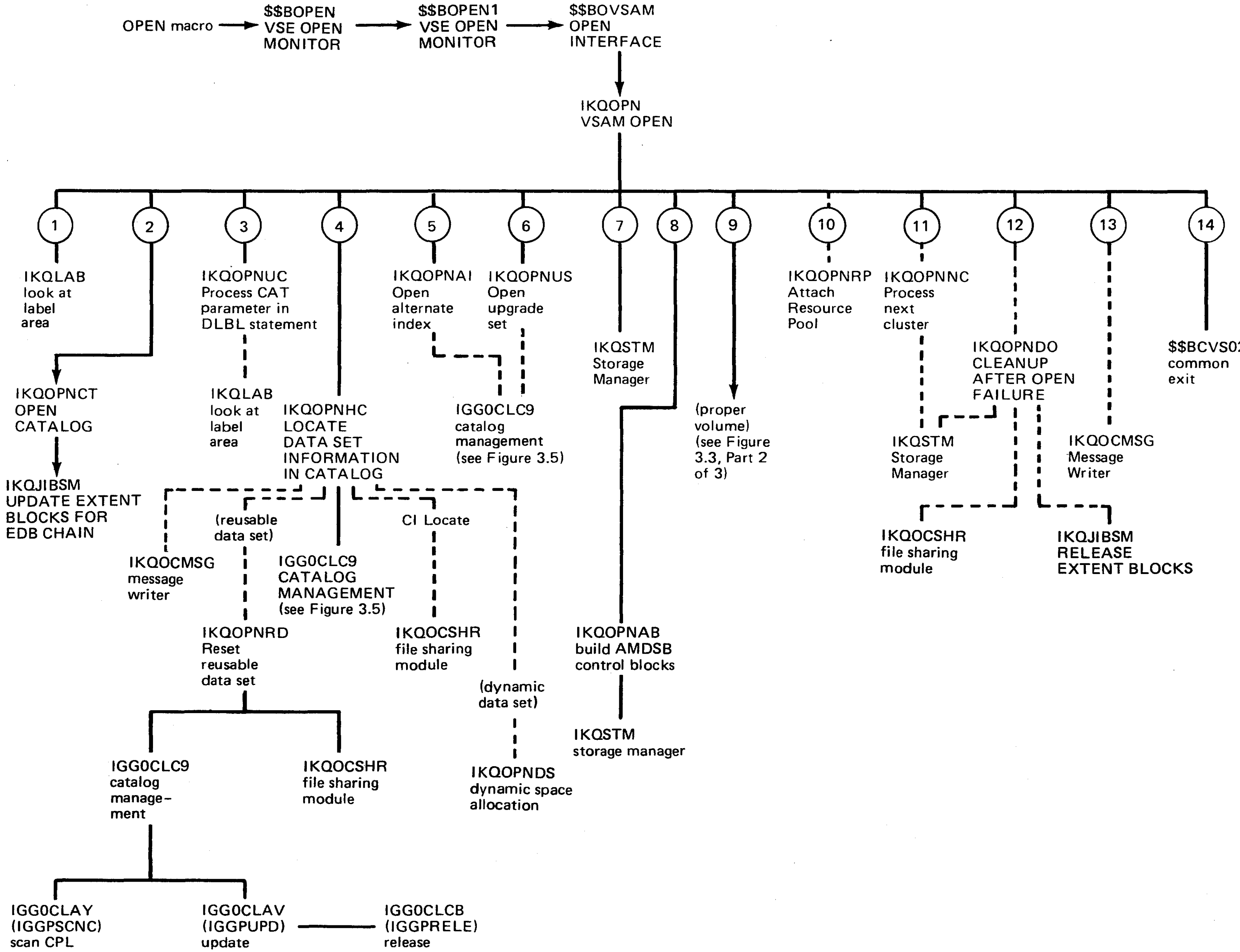


Figure 3.3 Program structure to process an OPEN (part 1 of 3)

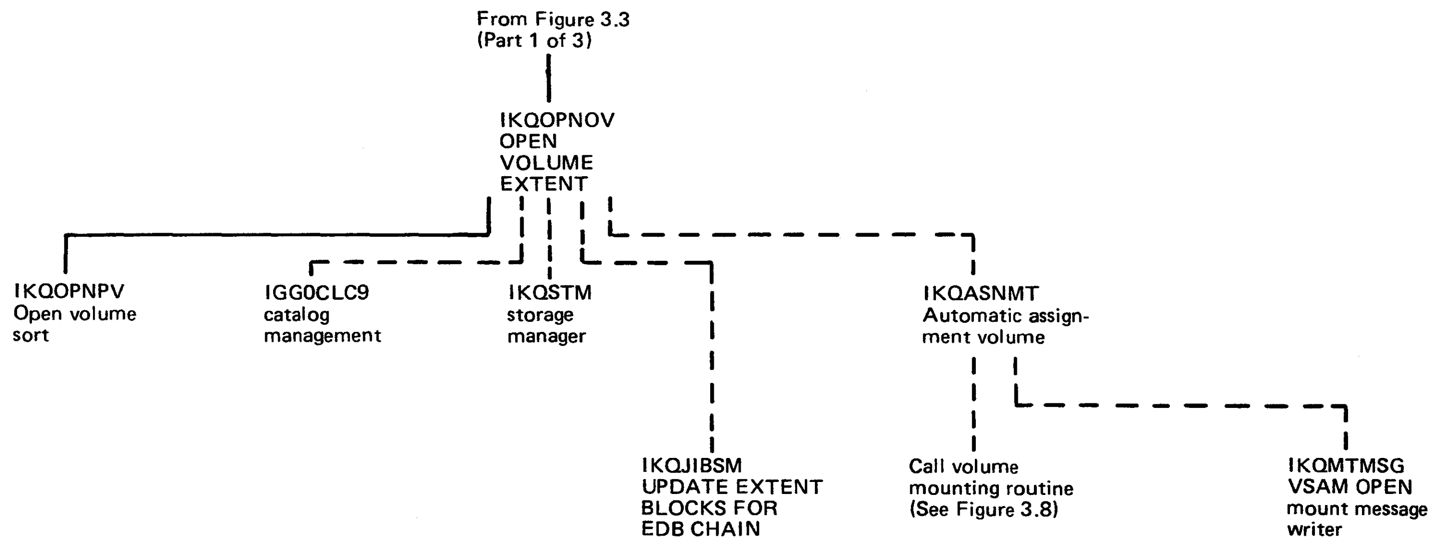


Figure 3.3 Program structure to process an OPEN (part 2 of 3)

Figure 3.3 Program structure to process an OPEN (part 3 of 3)

1. IKQLAB reads the label area and establishes the connection between data set name and file name.

2. IKQOPNCT is called if a catalog or a catalog recovery area is to be opened. IKQOPNCT finds in VTOC the address of the catalog cluster record or the CRA cluster record and reads it.

3. IKQOPNCT checks if a user catalog is needed to open the cluster. It obtains user catalog label information via IKQLAB if required.

4. IKQOPNHC is called to locate information in the catalog concerning the cluster to be opened. It resets reusable data sets via IKQOPNRD. It checks whether the cluster can be opened according to the sharing conditions via IKQLASMD.

5. IKQOPNAI is called if the cluster to be opened is an alternate index cluster. IKQOPNAI retrieves the cluster record of the base cluster identified by the AIX record from catalog.

6. IKQOPNUS is called whenever a possbile base cluster is processed. IKQOPNUS retrieves information concerning the upgrade set from the catalog.

7. IKQSTM is called to allocate VSAM record management control blocks.

8. IKQOPNAB is called to do validity checking and build the AMDSB control block structure for a single component.

9. IKQOPNOV is called to process the cluster's extent information. It retrieves the extent information from the catalog, obtains extent blocks via IKQJIBSM, checks if the proper volumes are mounted, and builds the control blocks via IKQSTM.

10. IKQOPNRP is called when resource sharing is requested. It attaches the cluster's control block structure to the resource pool.

11. IKQOPNNC is called when an alternate index structure is to be opened. It decides which cluster is to be opened next and creates ACB and RPL via IKQSTM.

12. IKQOPNDO is called whenever an open error occurred. It reduces the open count via IKQLASMD. It resets the open indication in catalog. It releases the extent blocks via IKQJIBSM. It frees the allocated storage via IKQSTM.

13. IKQOCMSG is called whenever a message has to be written.

14. $$BCVS02 returns-control to the user.

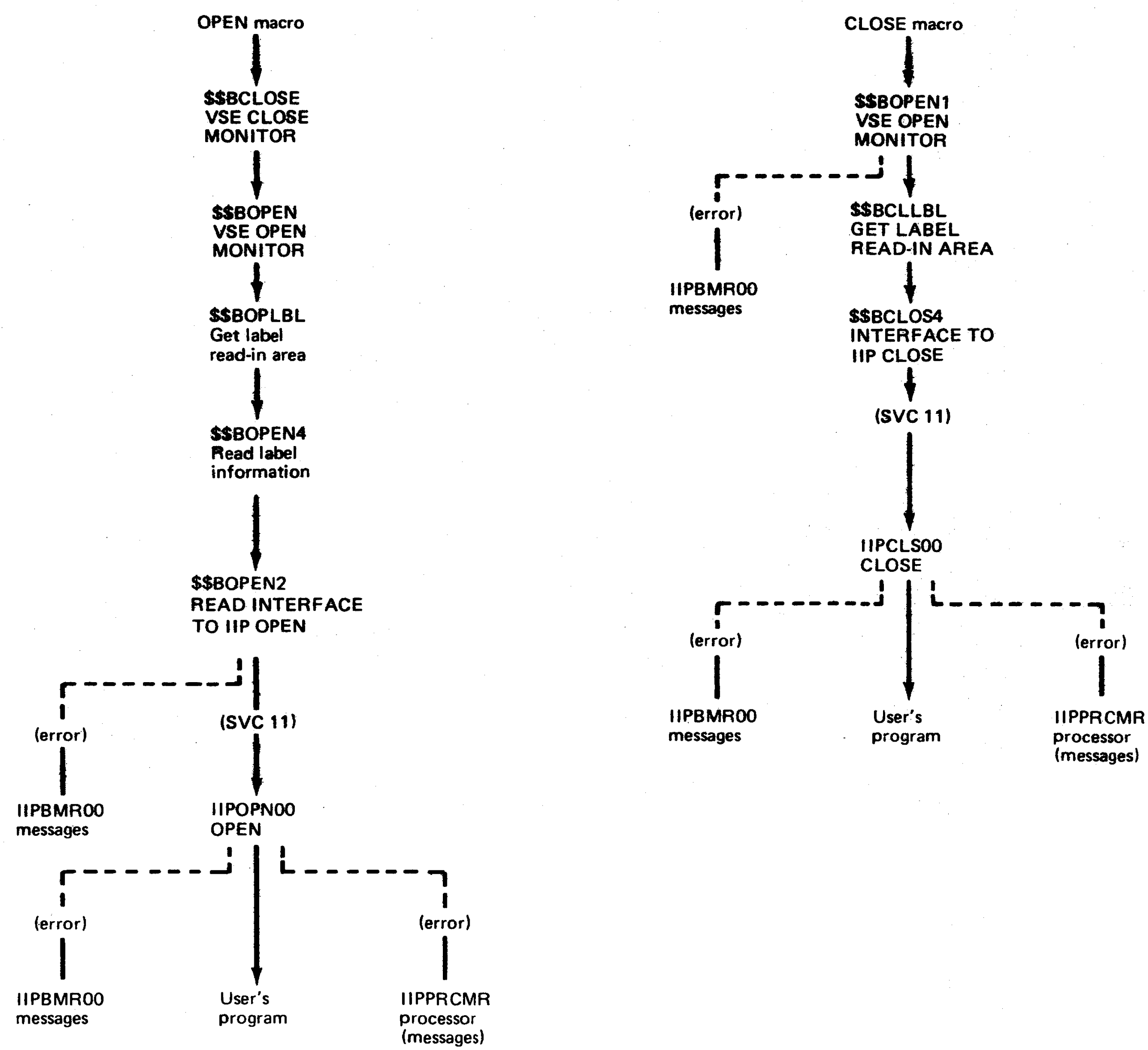


Figure 3.4 Program structure to process ISAM interface macros (part 1 of 2)

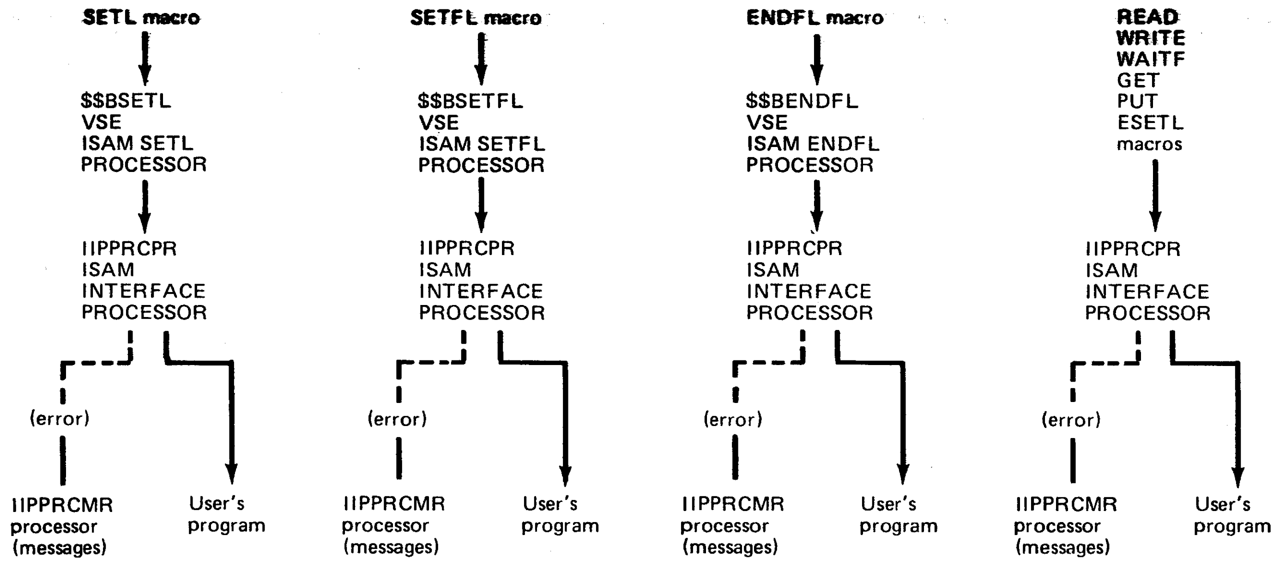


Figure 3.4 Program structure to process ISAM interface macros (part 2 of 2)

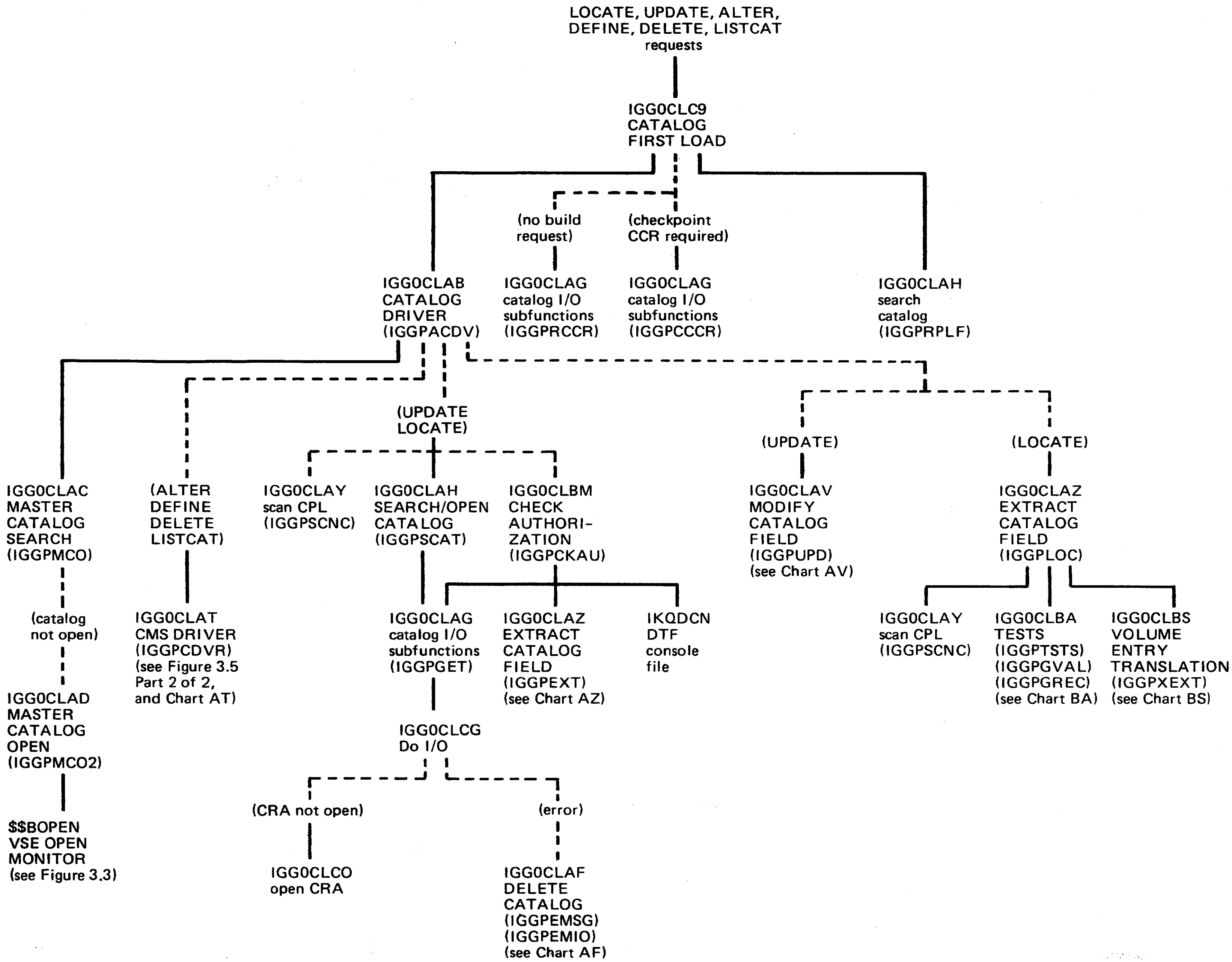


Figure 3.5 Program structure to process catalog management requests (part 1 of 2)

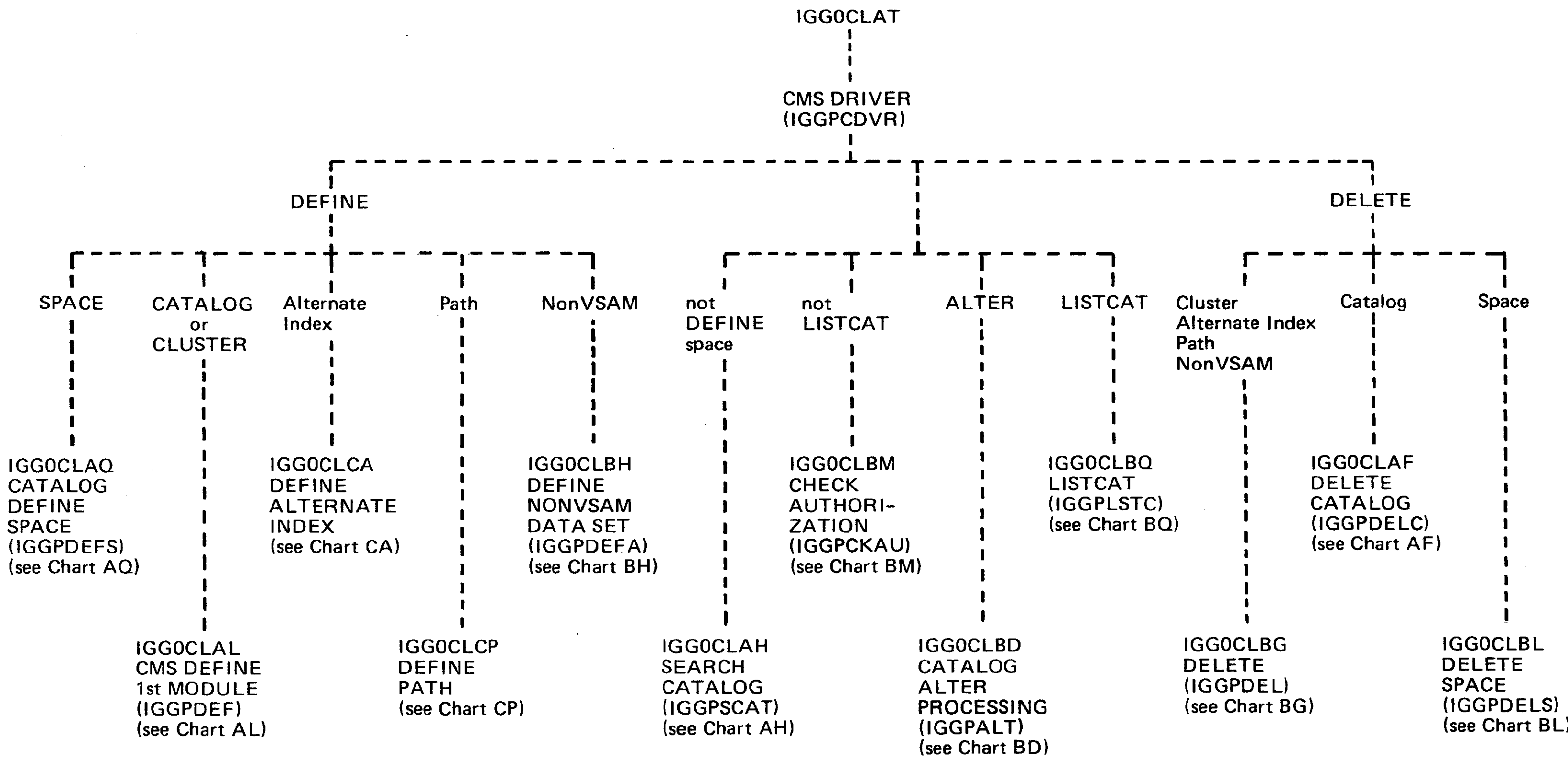


Figure 3.5 Program structure to process catalog management requests (part 2 of 2)

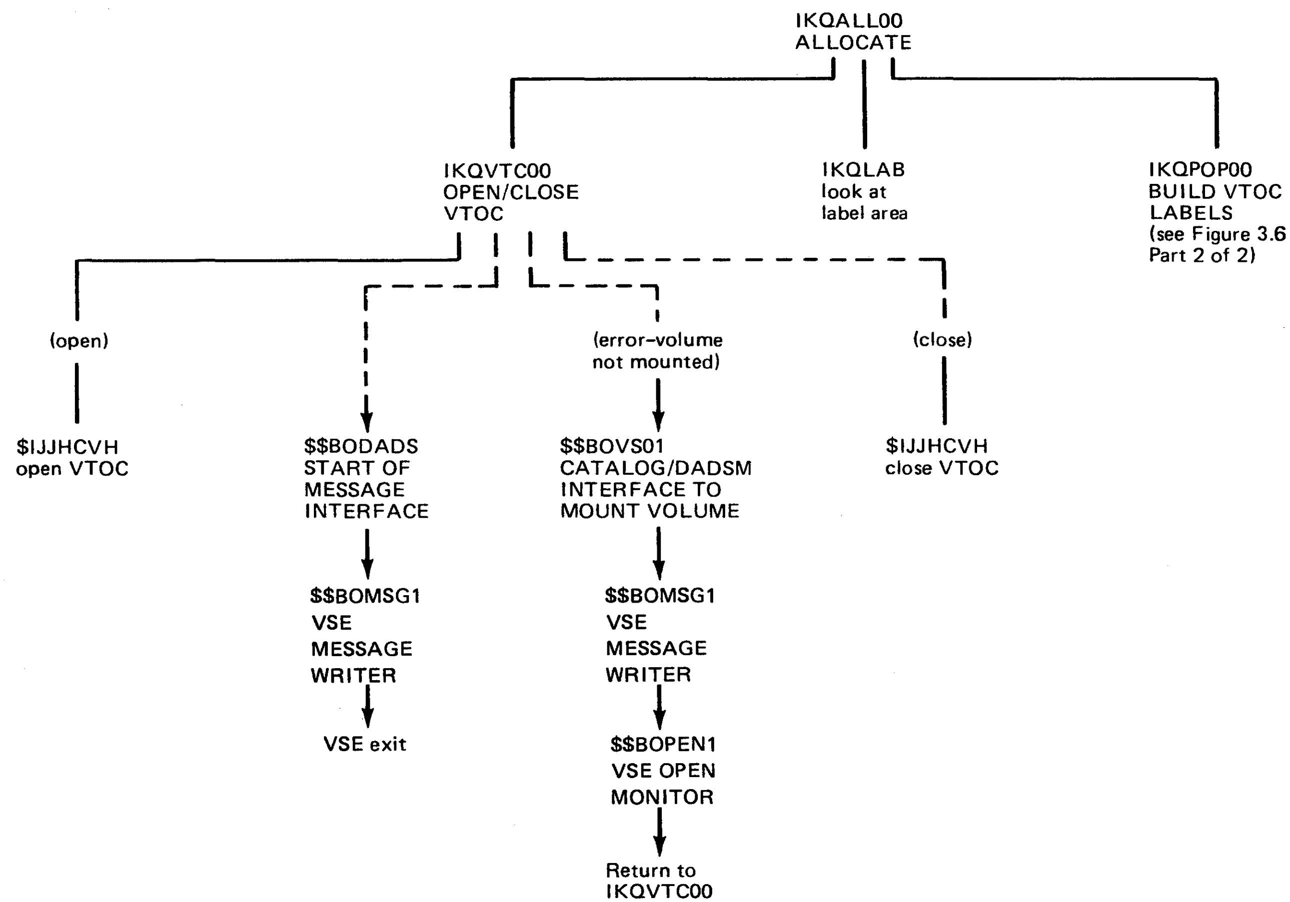
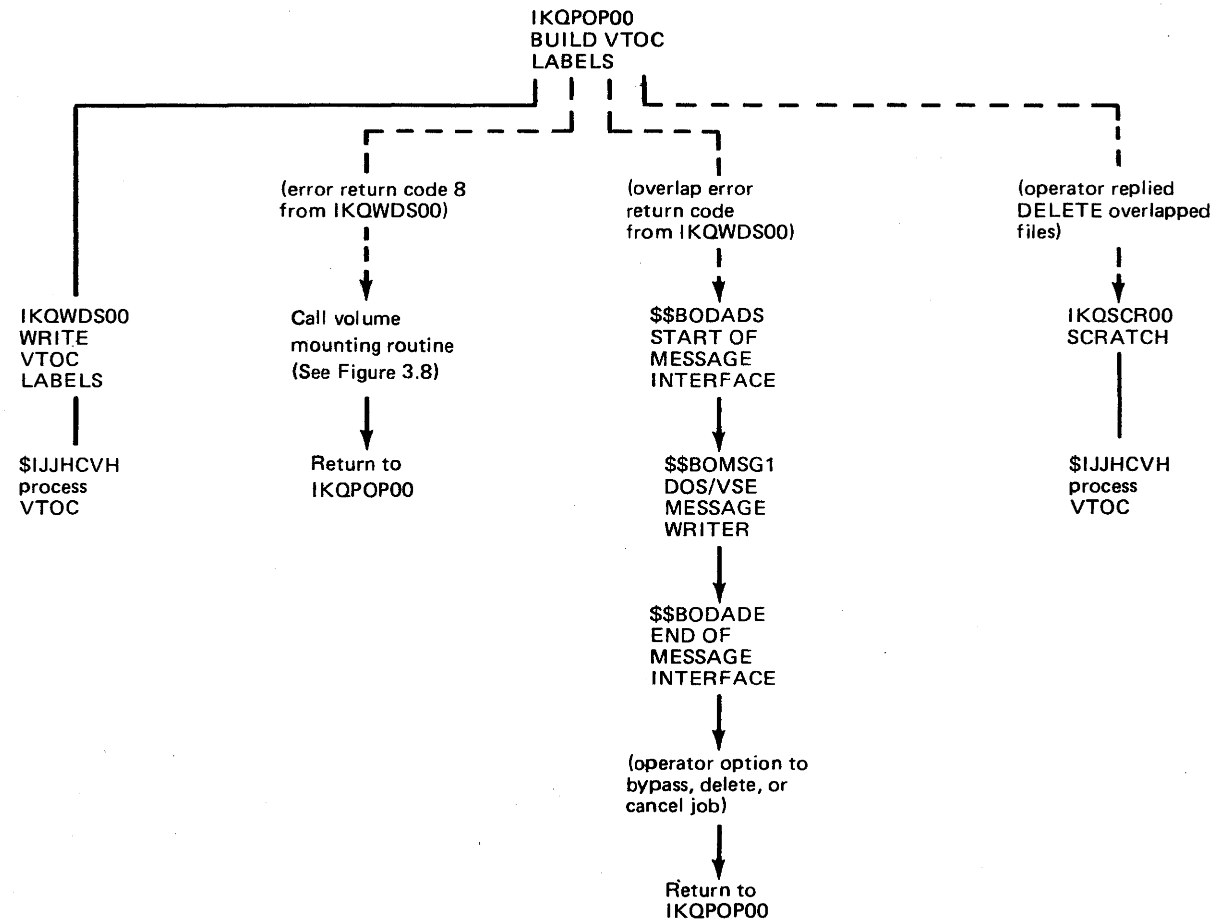


Figure 3.6 Program structure to process DADSM (part 1 of 2)

Figure 3.6 Program structure to process DADSM (part 2 of 2)



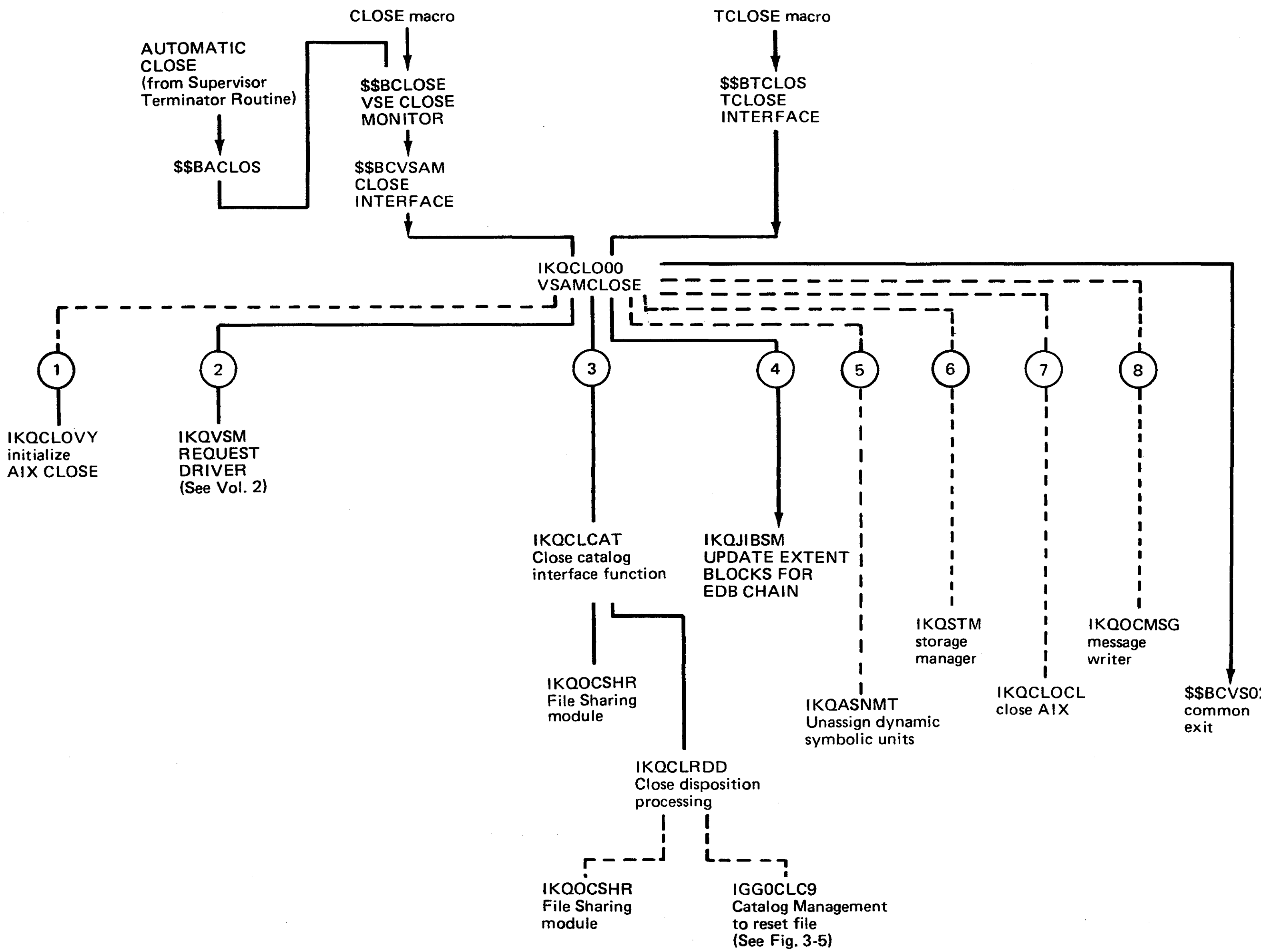


Figure 3.7 Program structure to process a CLOSE or TCLOSE (part 1 of 2)

Figure 3.7 Program structure to process a CLOSE or TCLOSE (part 2 of 2)

1 IKQCLOVY is called if an alternate index structure is to be closed: i.e. a path entry or an upgrade set exists. IKQCLOVY determines which cluster is to be closed first.

2 IKQVSM is called to complete outstanding I/O. For LSR (local shared resources) deferred I/O is completed.

3 IKQCLCAT is called to update the high water marks and the statistics for the data set to be closed in catalog.

4 IKQJIBSM is called to delete Extent Blocks.

5 IKQASNMT is called to unassign. Dynamic assignment occurred if a symbolic unit was not specified on the EXTENT statement.

6 IKQSTM is called for CLOSE request to free the virtual storage occupied by the control blocks of the data set to be closed.

7 IKQCLOCL is called if an alternate index structure is to be closed. IKQCLOCL determines the next cluster to be closed.

8 IKQOCMSG is called whenever an error turns up during close. IKQOCMSG writes a message to the user.

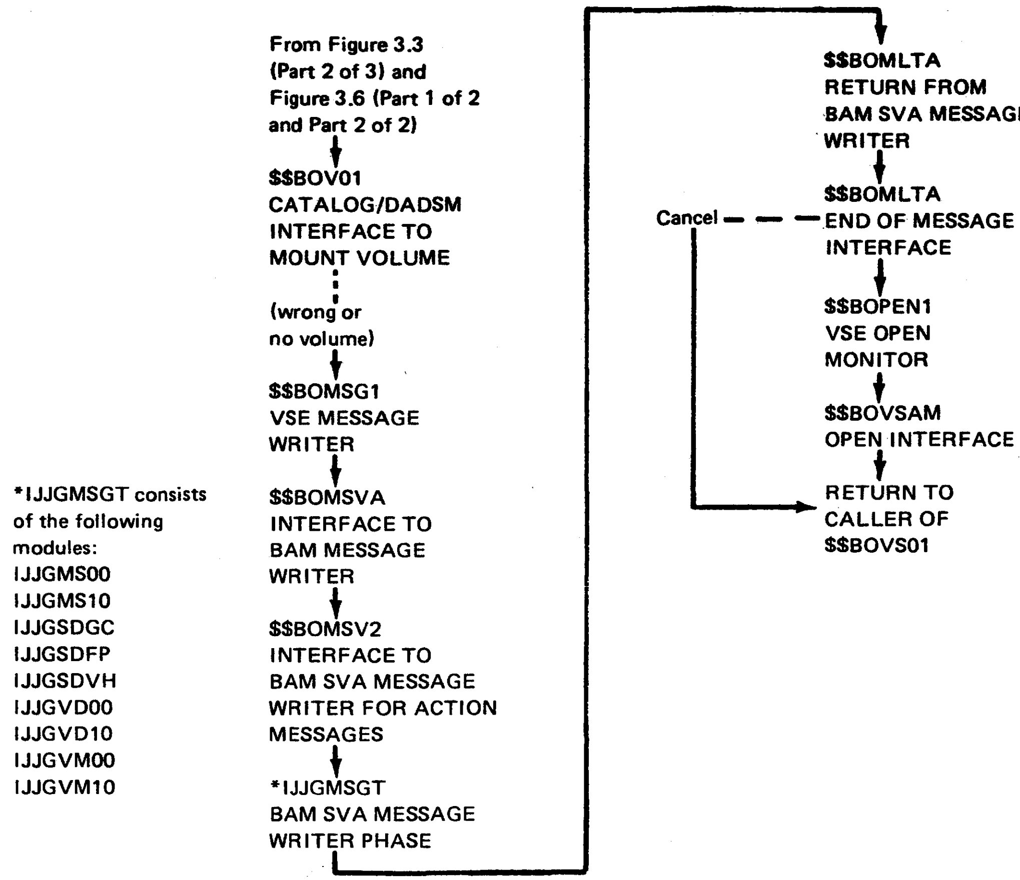


Figure 3.8 Volume mounting routine

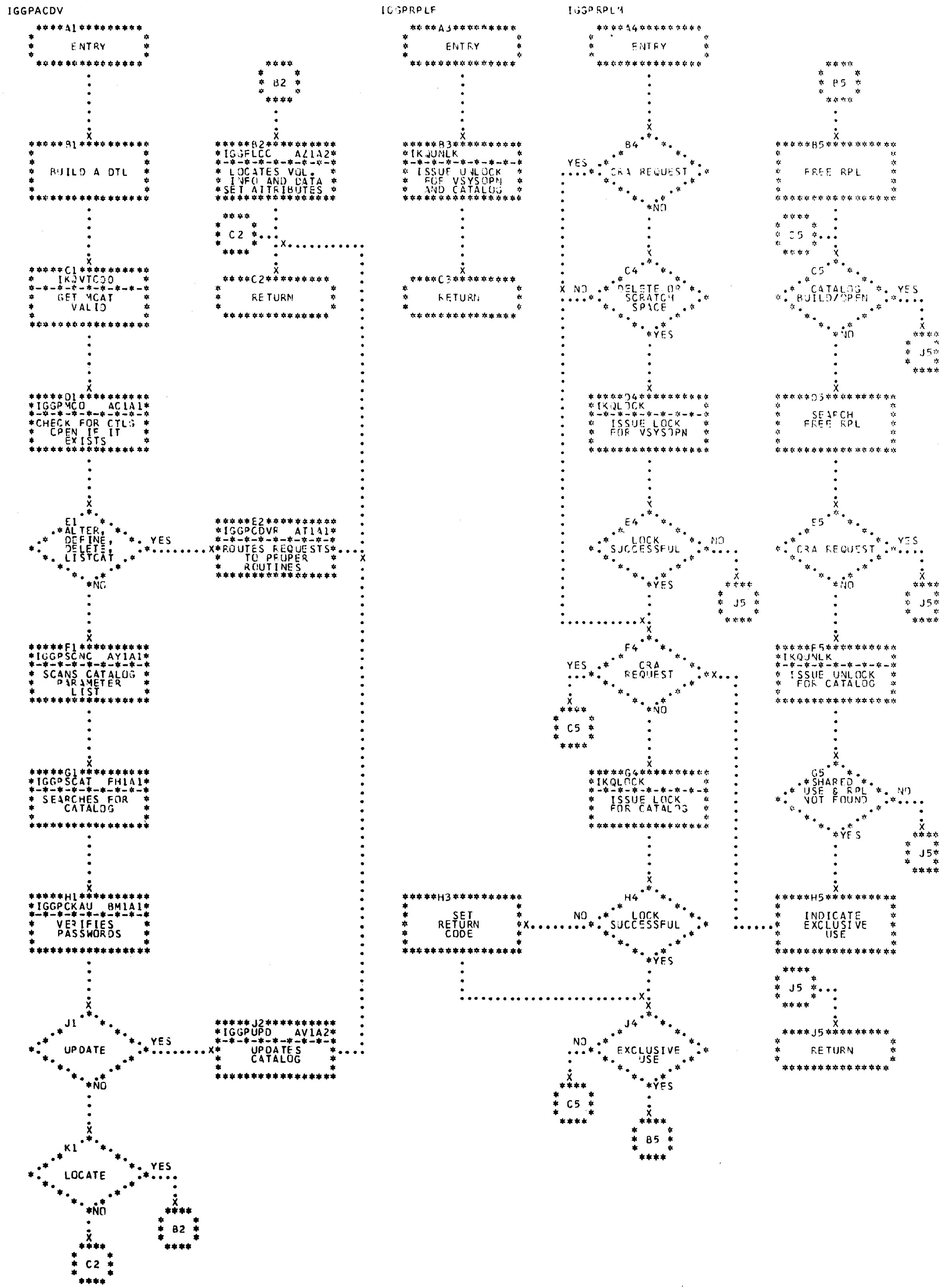


Chart AB1. Catalog Driver (IGG0CLAB)

```
IGGPMCO
       *****A1**********
       *               *
       *     ENTRY     *
       *               *
       *****************
               |
               |
               V
       *****B1**********
       *IKQLOCK        *
       *-*-*-*-*-*-*-*-*
       *     LOCK      *
       *    VSYSMCO    *
       *               *
       *****************
               |
               |
               V
              .*.
          C1 .   *.
           .*       *.
   YES  .*   MASTER    *.
 ,-----*. CATALOG OPEN  .*
 |        *.          .*
 |          *.      .*
 |            *.  .*
 |              *NO
 |               |
 |               V
 |              .*.
 |         D1  .   *.
 |           .*      *.
 |YES     .*  DEFINE   *.
 |<---*.     MASTER    .*
 |        *. CATALOG .*
 |          *.     .*
 |            *. .*
 |             *NO
 |               |
 |               V
 |     *****E1**********
 |     *IGGPMOD2  AD1A1*
 |     *-*-*-*-*-*-*-*-*
 |     * BUILD ACB AND *
 |     *ISSUE OPEN FOR *
 |     *MASTER CATALOG *
 |     *****************
 |               |
 L-------------->|
                 |
                 V
       *****F1**********
       *IKQUNLK        *
       *-*-*-*-*-*-*-*-*
       *    UNLOCK     *
       *    VSYSMCO    *
       *               *
       *****************
               |
               |
               V
       *****G1**********
       *               *
       *    RETURN     *
       *               *
       *****************
```

Chart AC1. Master Catalog Search (IGG0CLAC)

```
IGGPMCO2
 *****A1**********
 *               *
 *     ENTRY     *
 *               *
 *****************
         |
         |
         V
 *****B1**********
 *$$BOPEN        *
 *-*-*-*-*-*-*-*-*
 *               *
 *SEE FIGURE 3.3 *
 *               *
 *****************
         |
         |
         V
 *****C1**********
 *IKQUNLK        *
 *-*-*-*-*-*-*-*-*
 *               *
 *UNLOCK VSYSMCO *
 *               *
 *****************
         |
         |
         V
 *****D1**********
 *               *
 *    RETURN     *
 *               *
 *****************
```

Chart AD1. Master Catalog Open (IGG0CLAD)

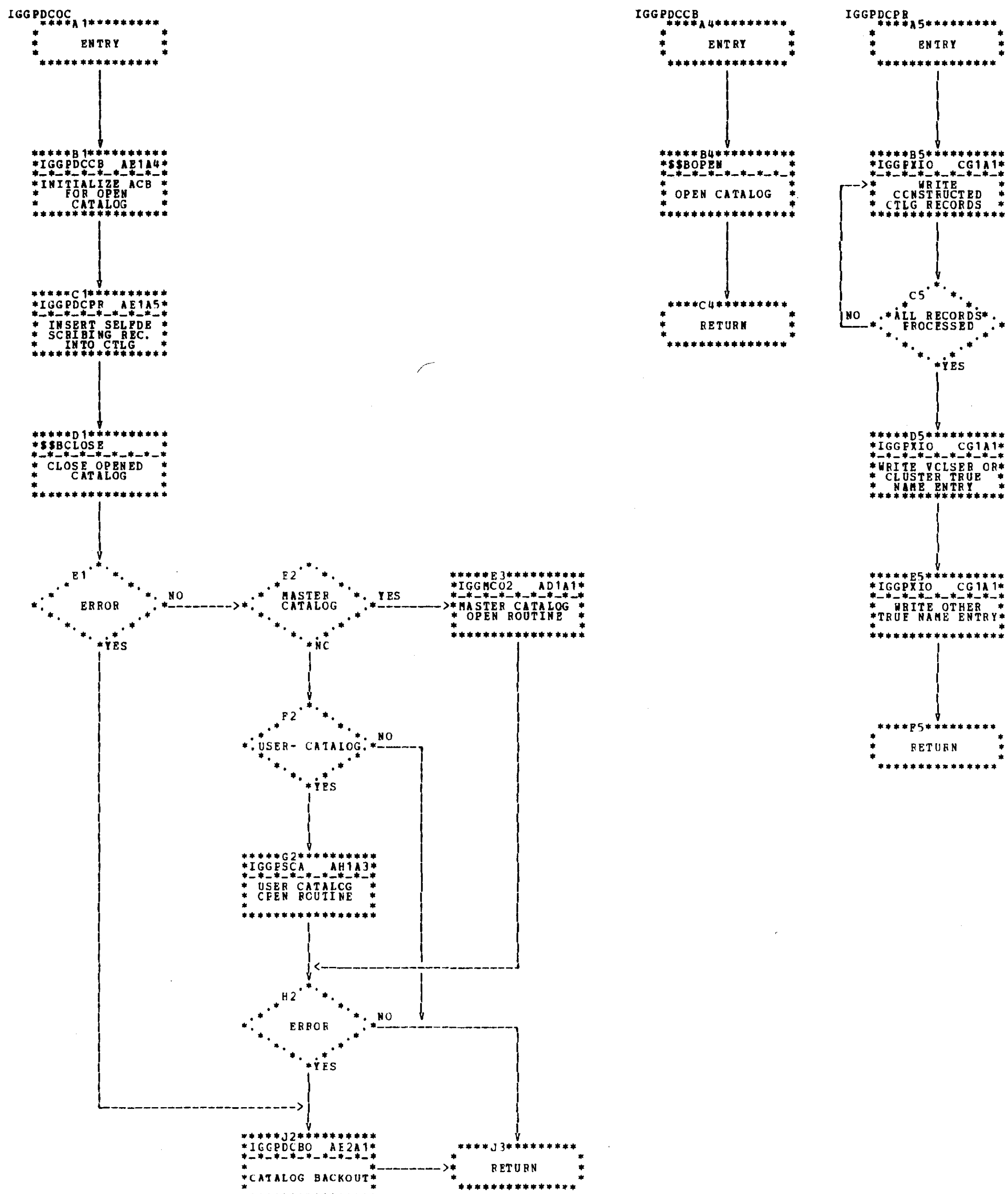


Chart AE1. Catalog Build and Open (IGG0CLAE)

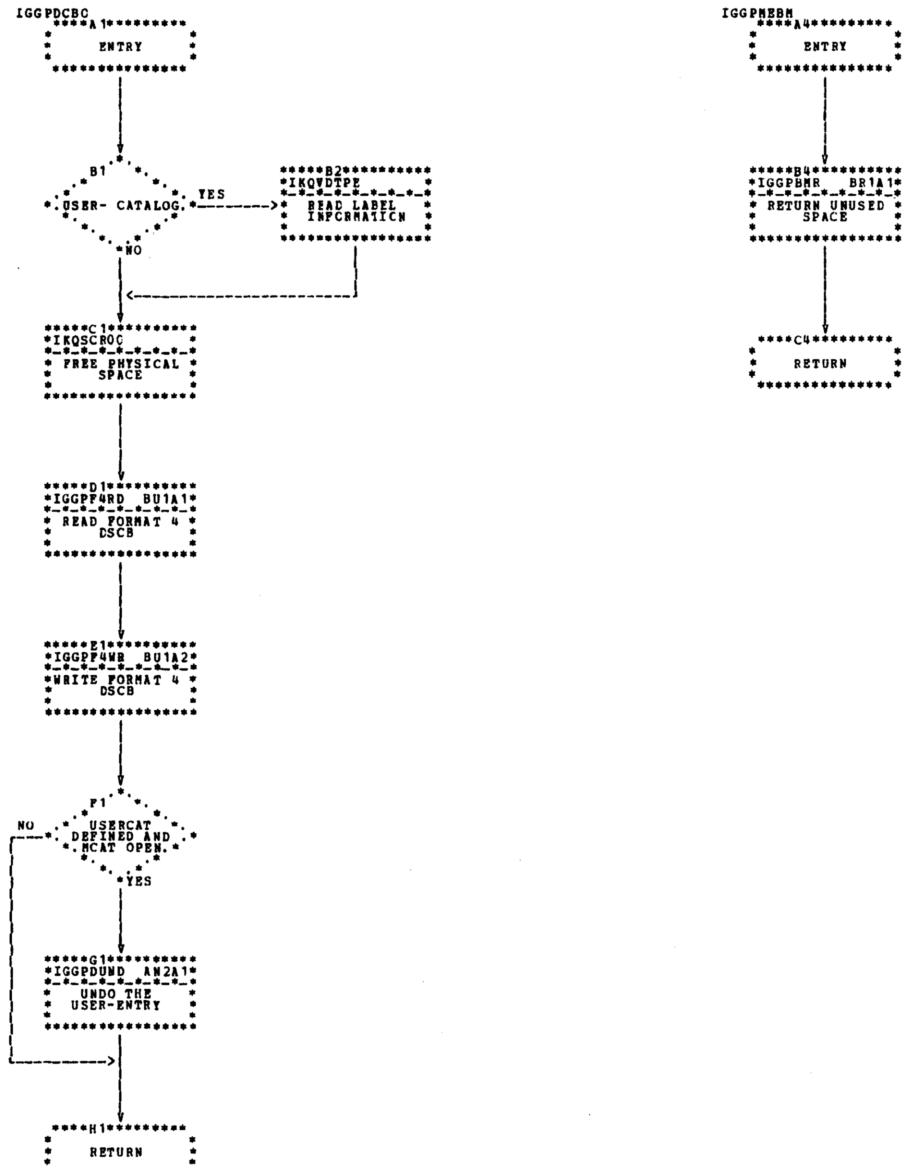


Chart AE2. Catalog Build and Open (IGG0CLAE)

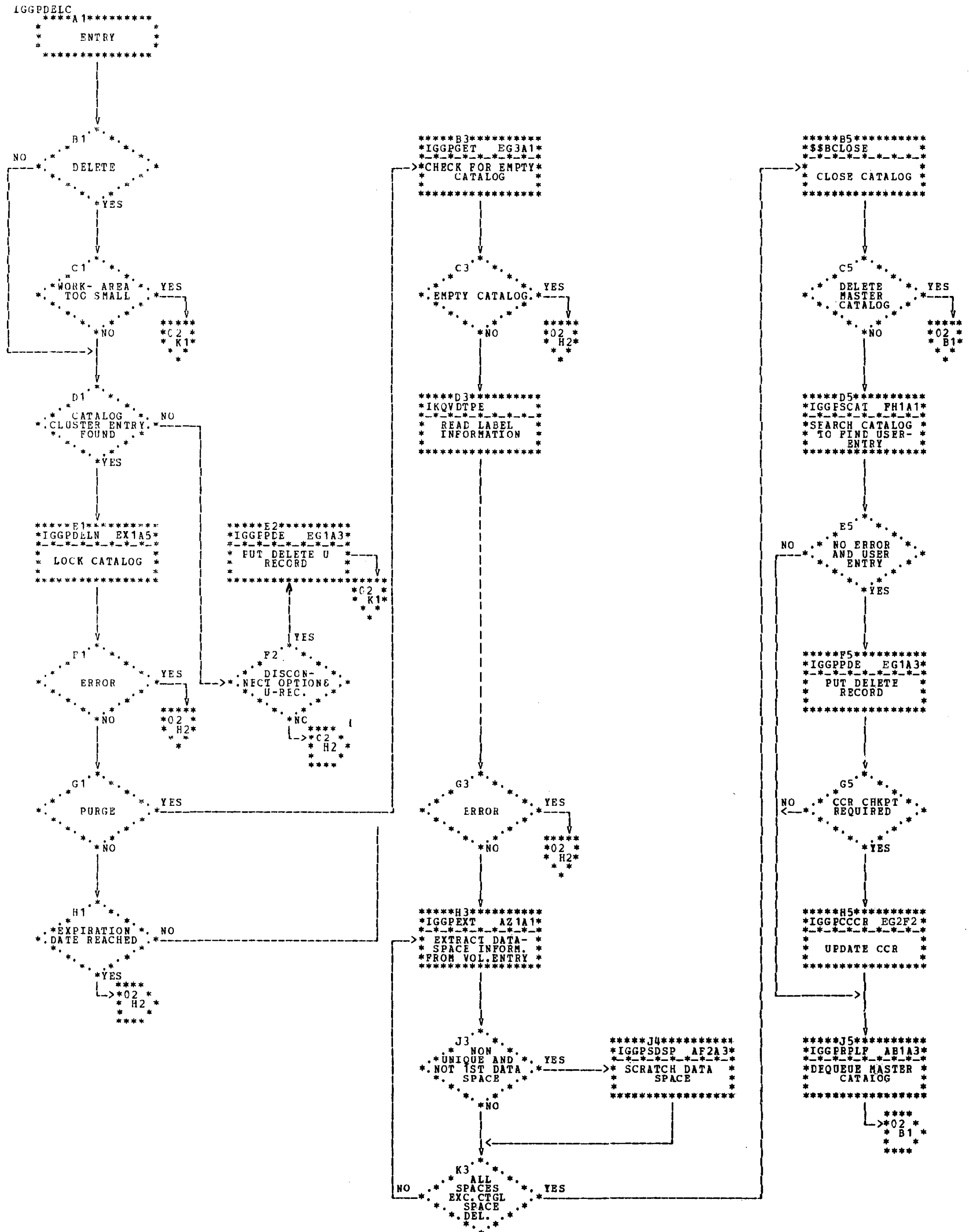


Chart AF1. CMS Delete Catalog (IGG0CLAF)

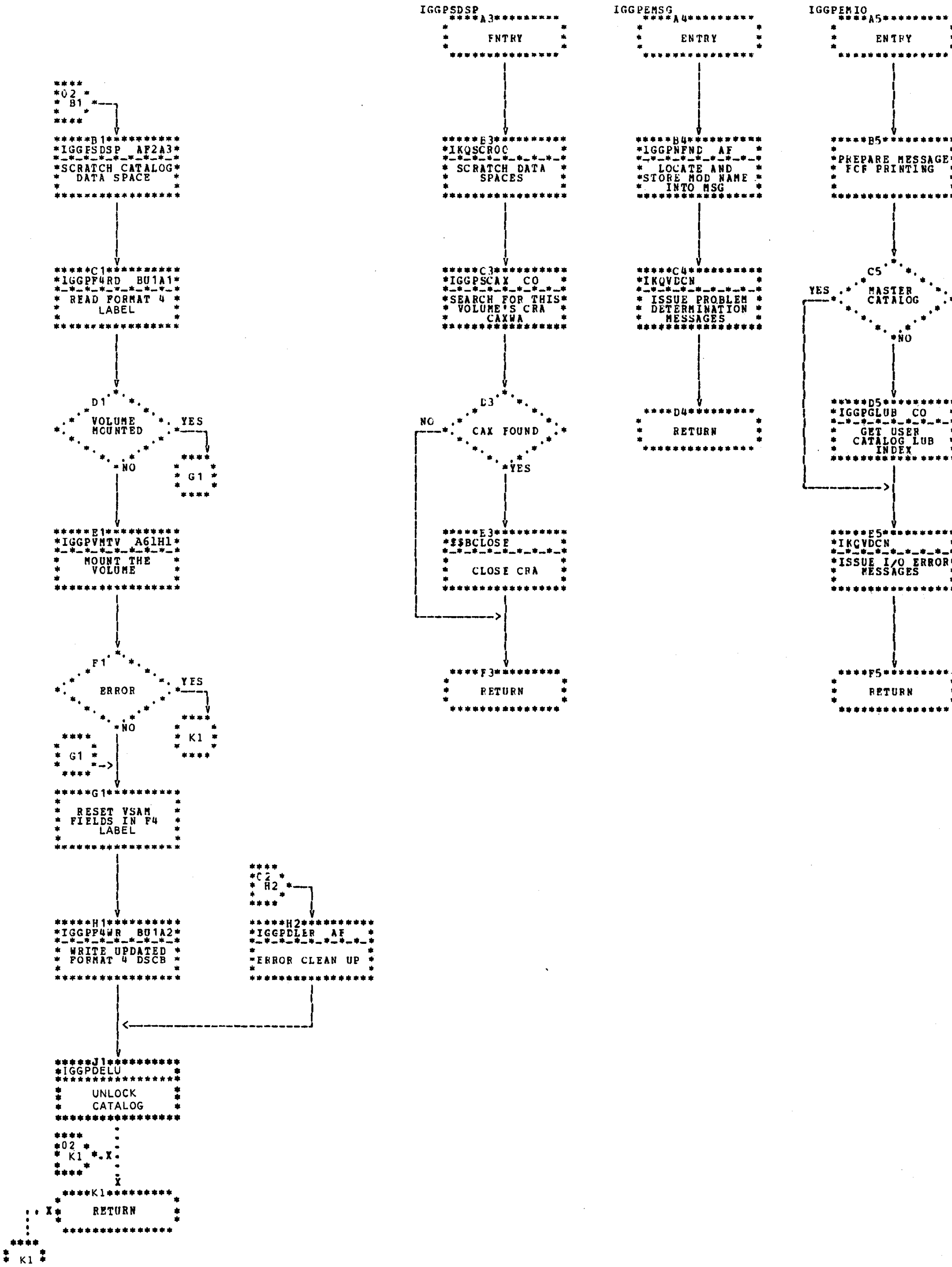


Chart AF2. Catalog Delete Catalog (IGG0CLAF)

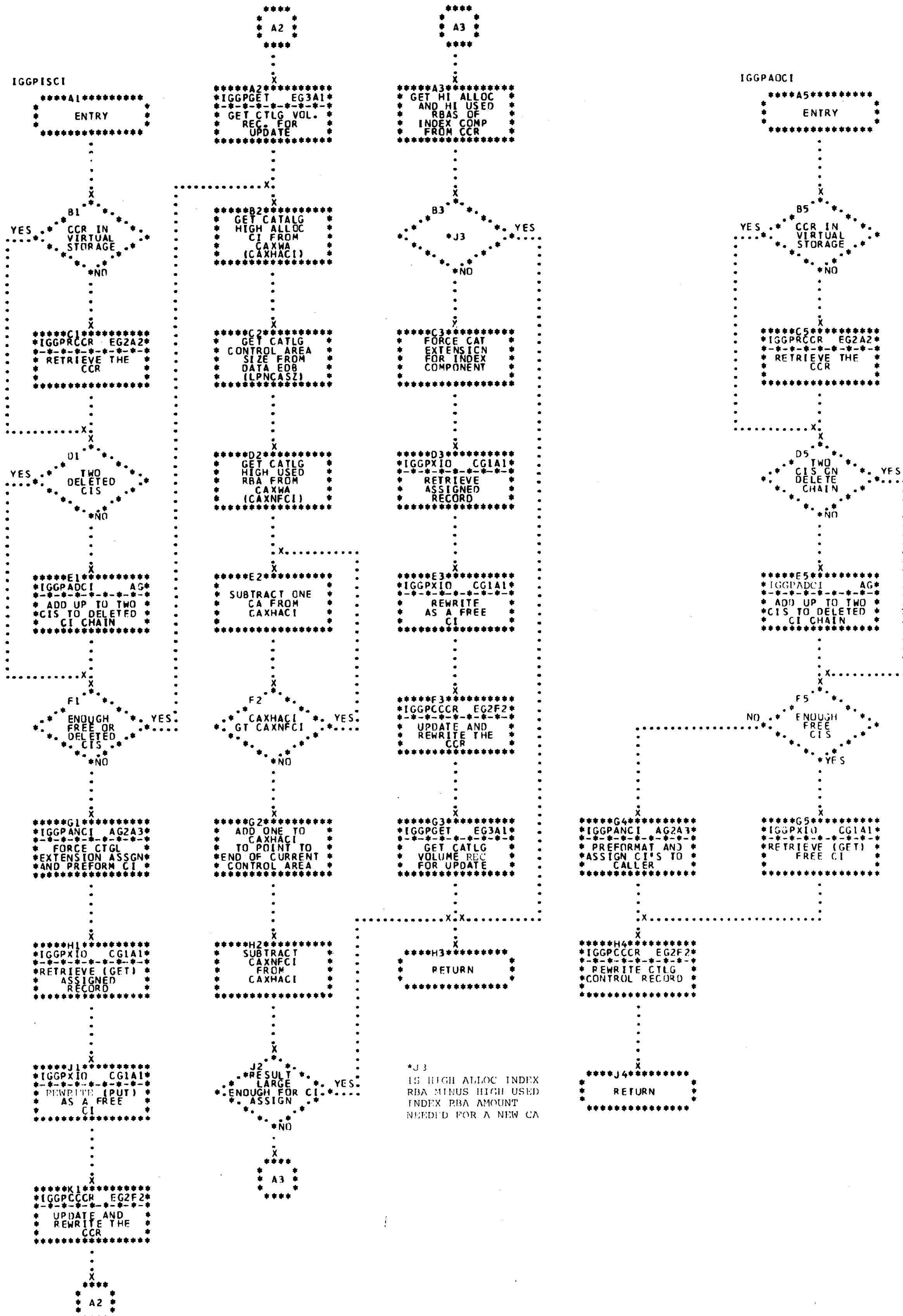


Chart AG1. Catalog I/O Subfunction (IGG0CLAG)

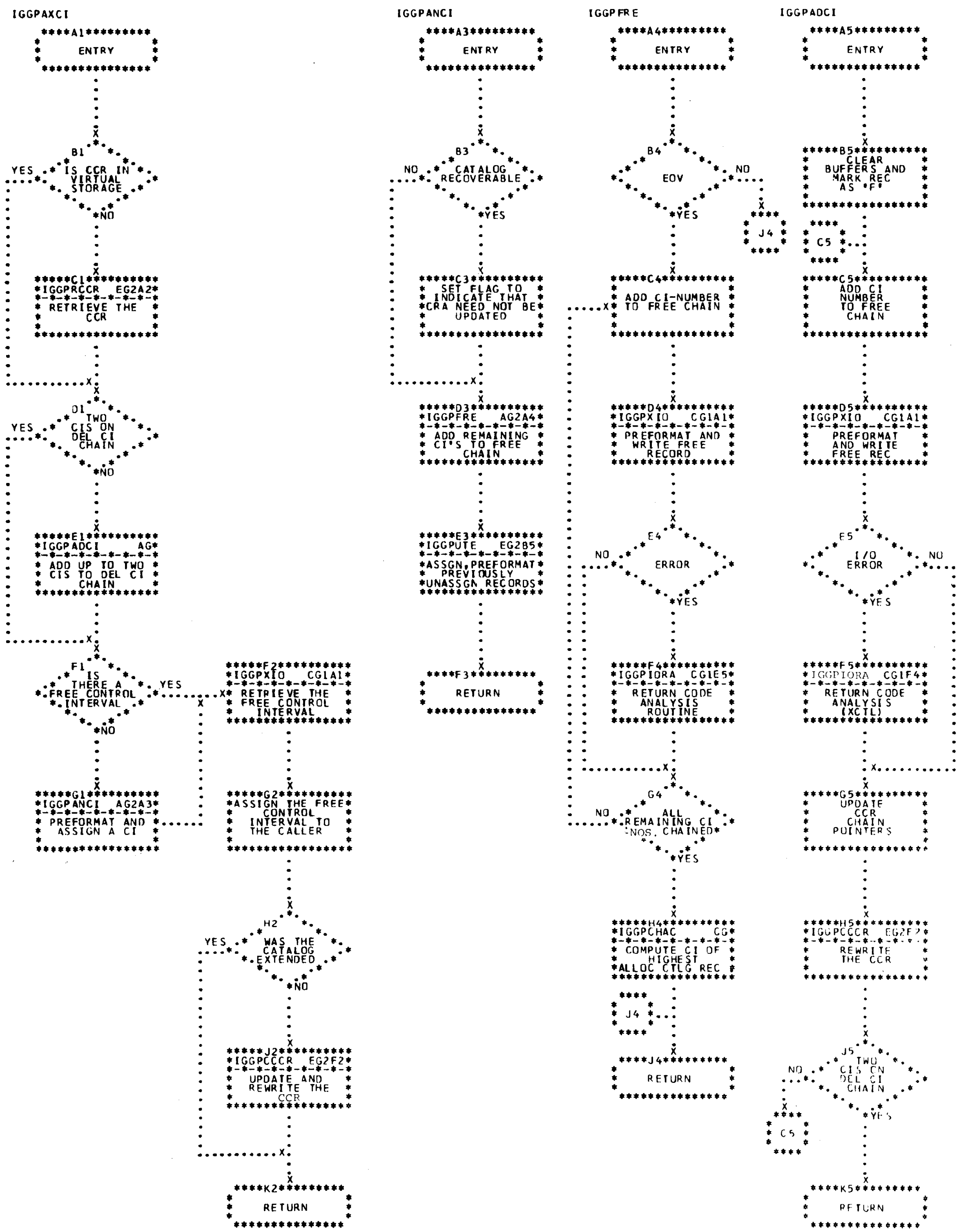


Chart AG2. Catalog I/O Subfunction (IGG0CLAG)

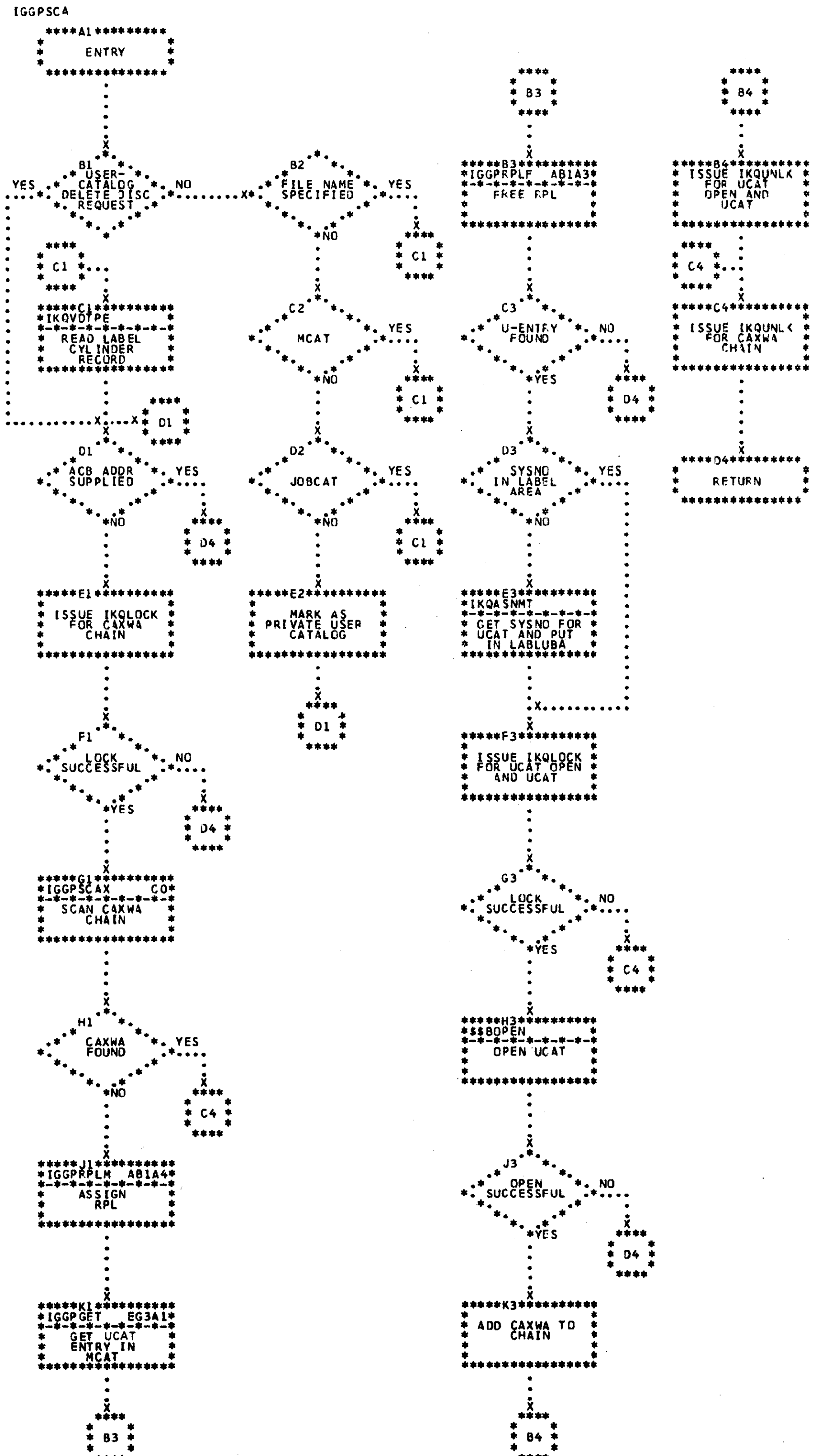


Chart AH1. Search/Open Catalog (IGG0CLAH)

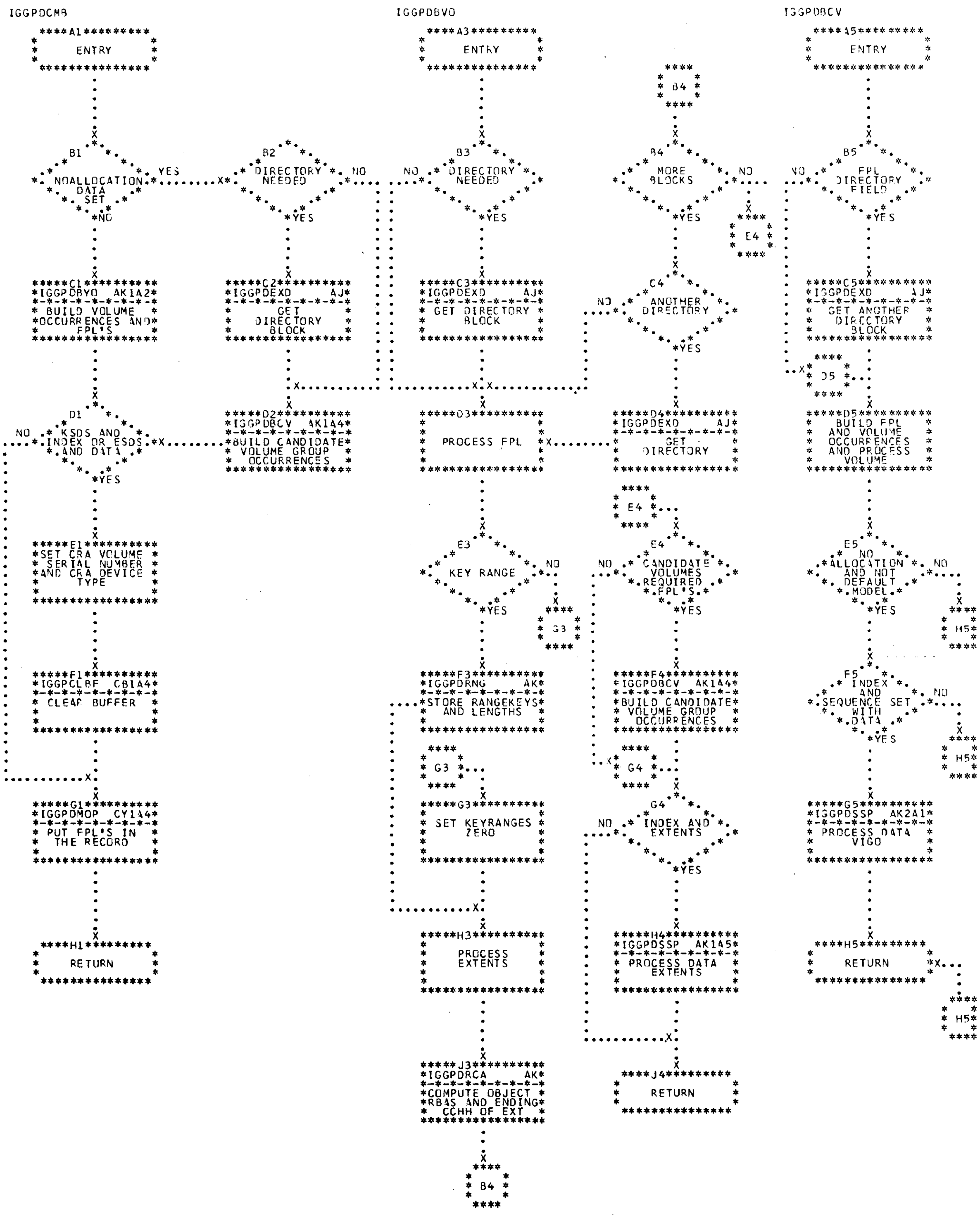


Chart AJ1. Define - Build Data and Index Entries (IGG0CLAJ)

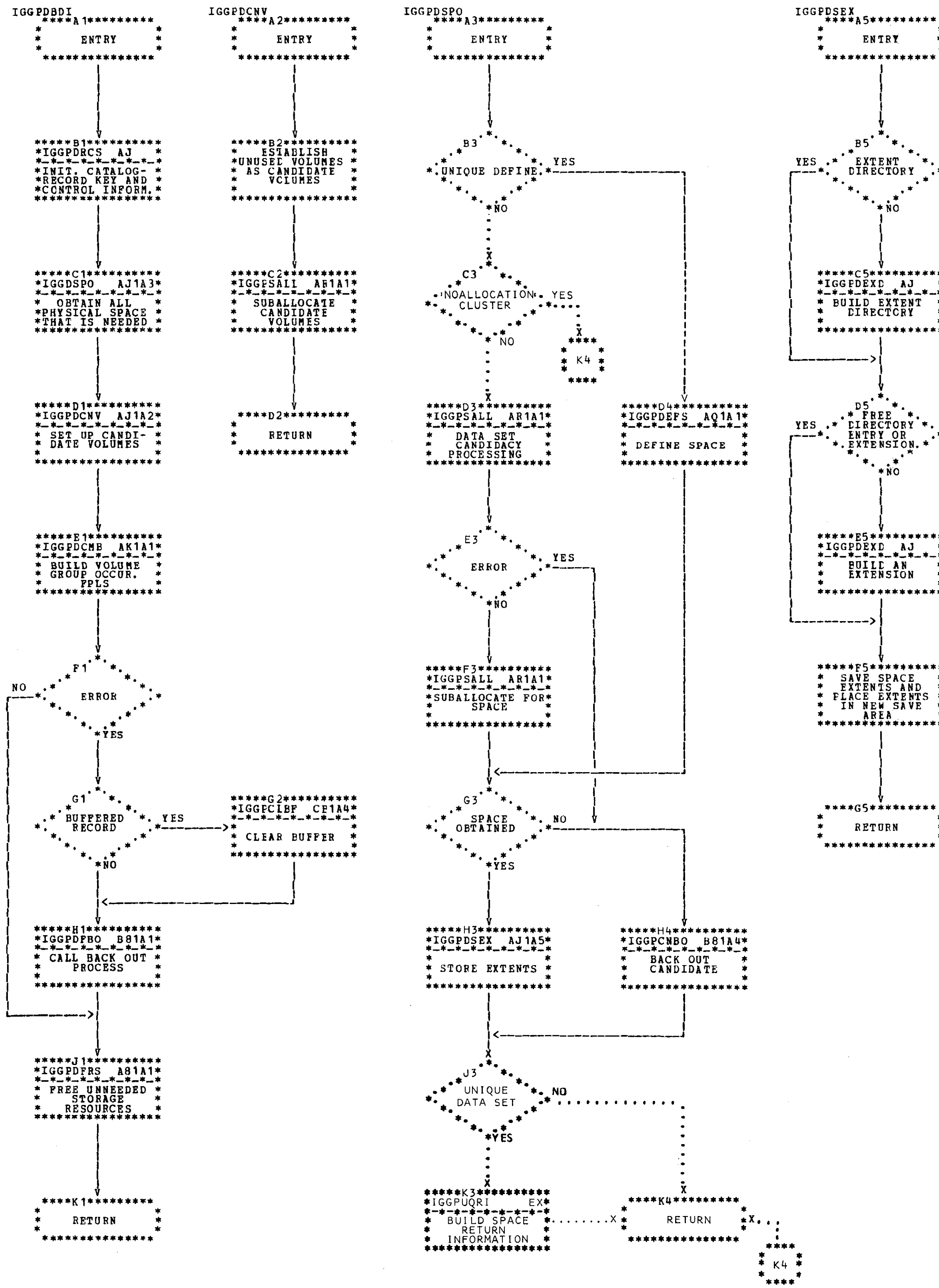


Chart AK1. Complete Define of an Entry (IGG0CLAK)

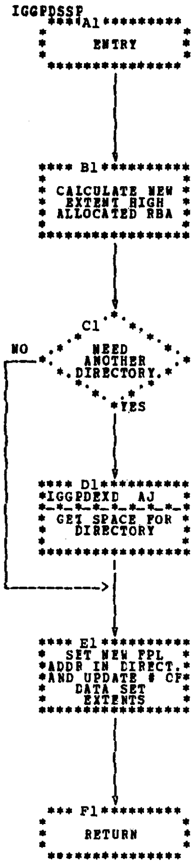


Chart AK2. Complete Define of an Entry (IGG0CLAK)

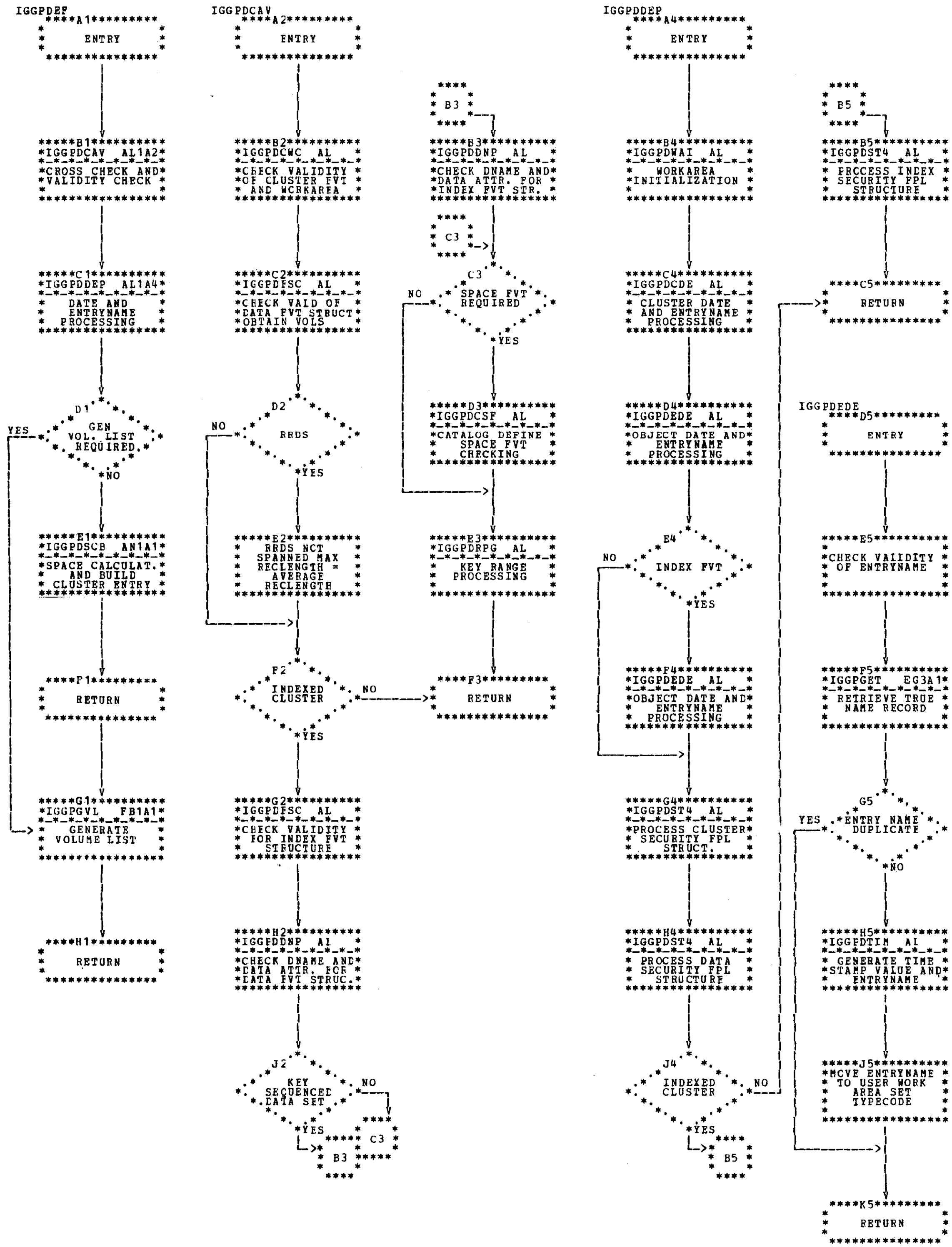


Chart AL1. CMS Define - First Module (IGG0CLAL)

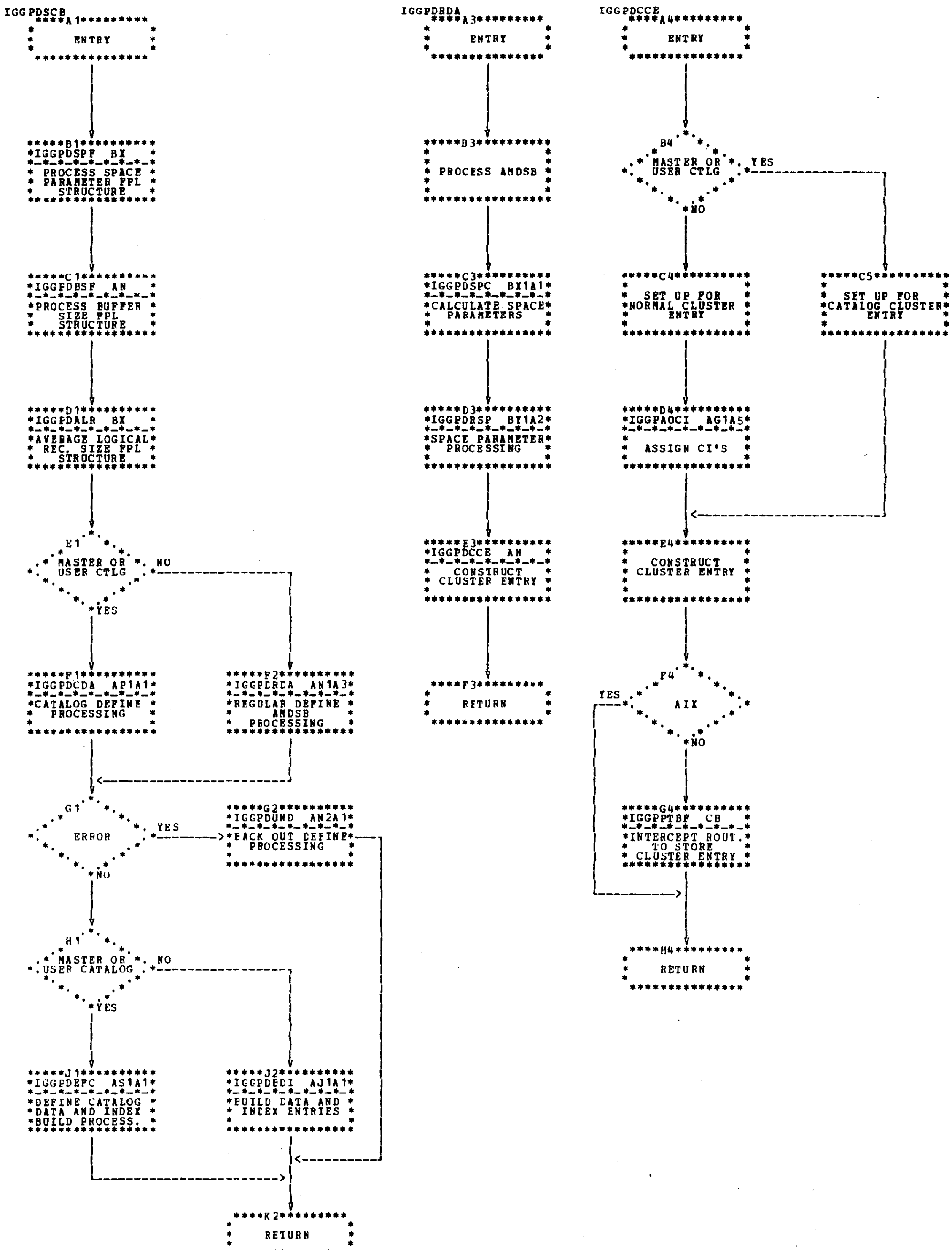


Chart AN1. CMS Define - Second Module (IGG0CLAN)

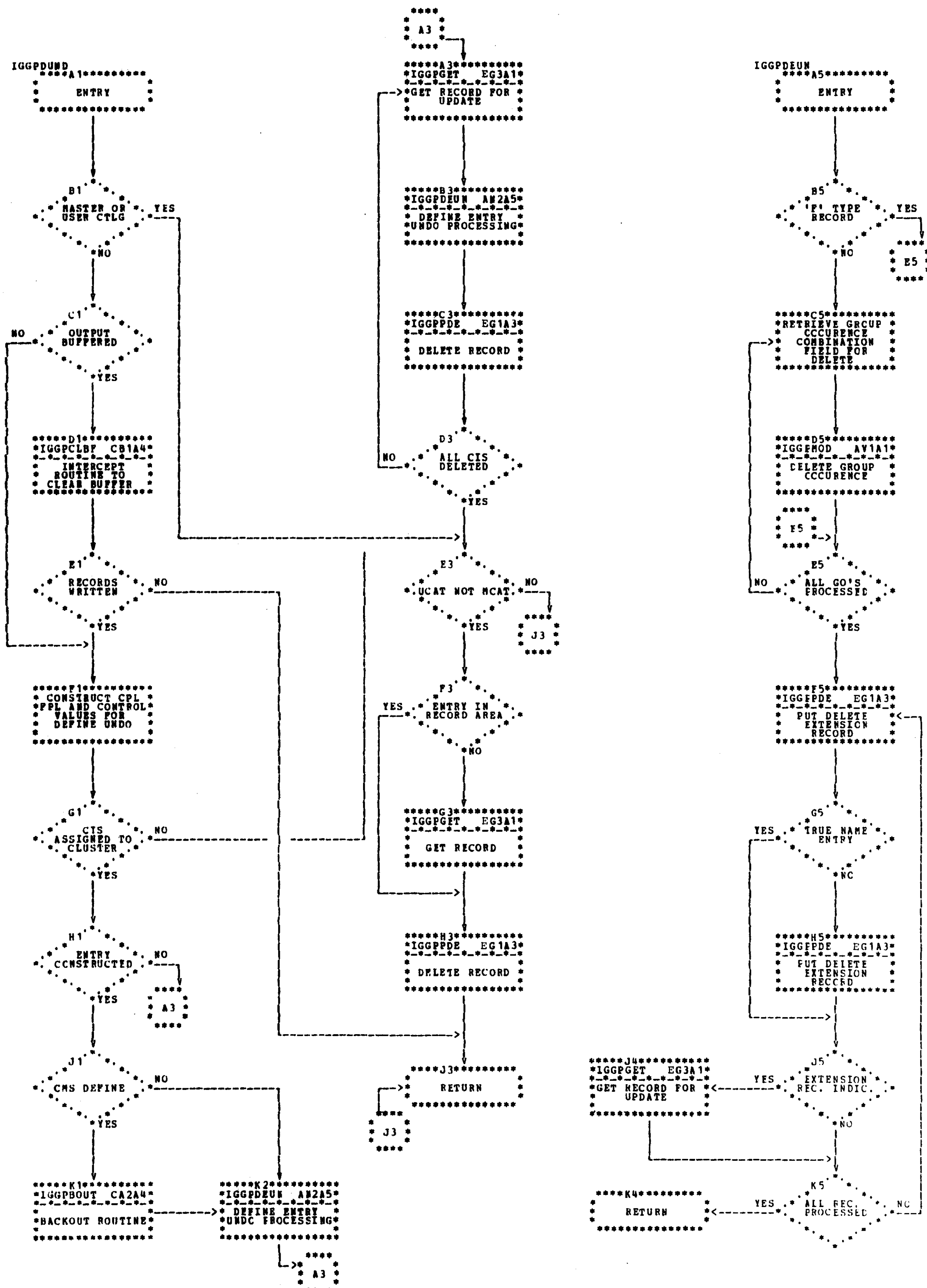


Chart AN2. CMS Define - Second Module (IGG0CLAN)

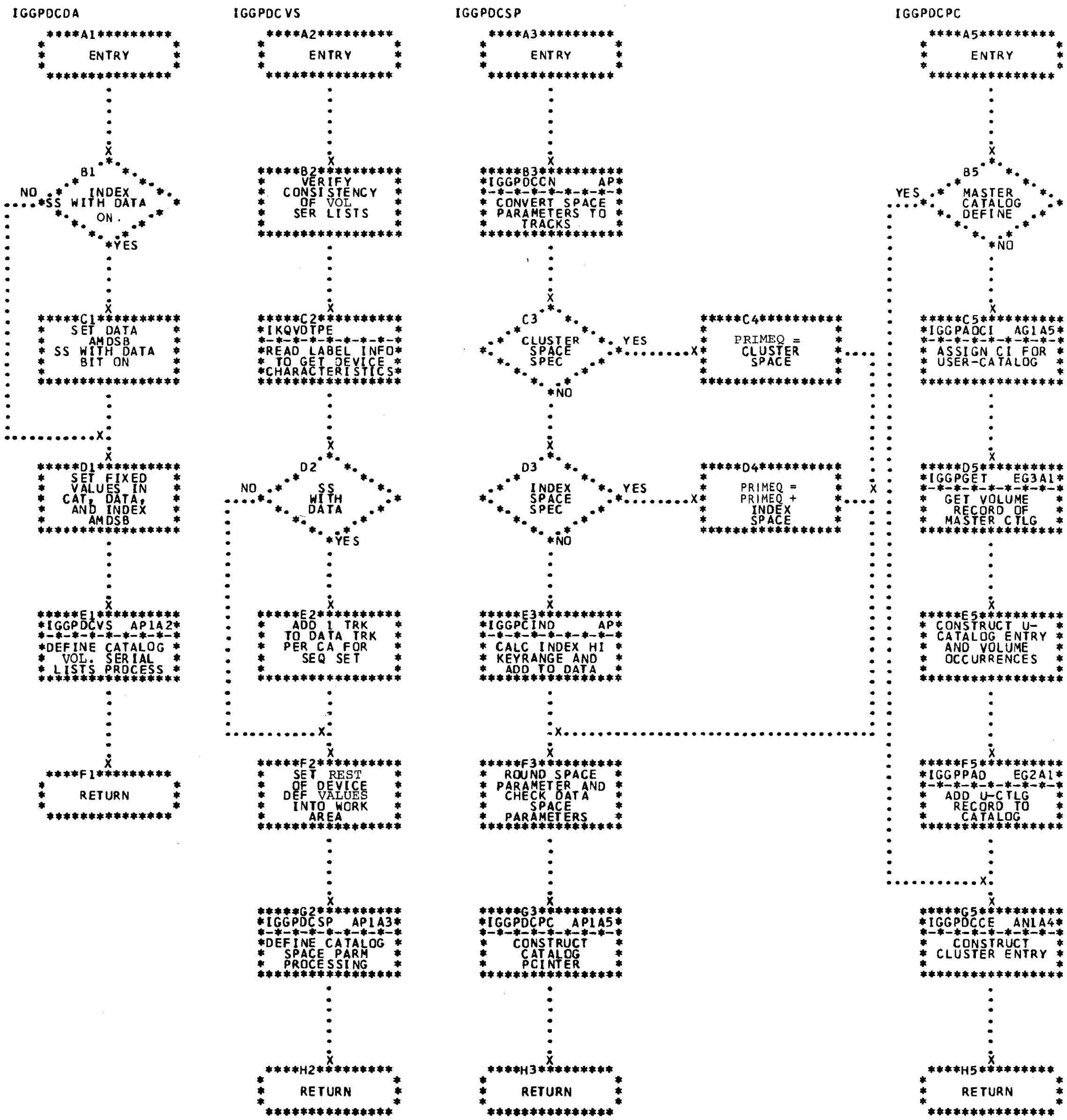


Chart AP1. CMS Define - Third Module (IGG0CLAP)

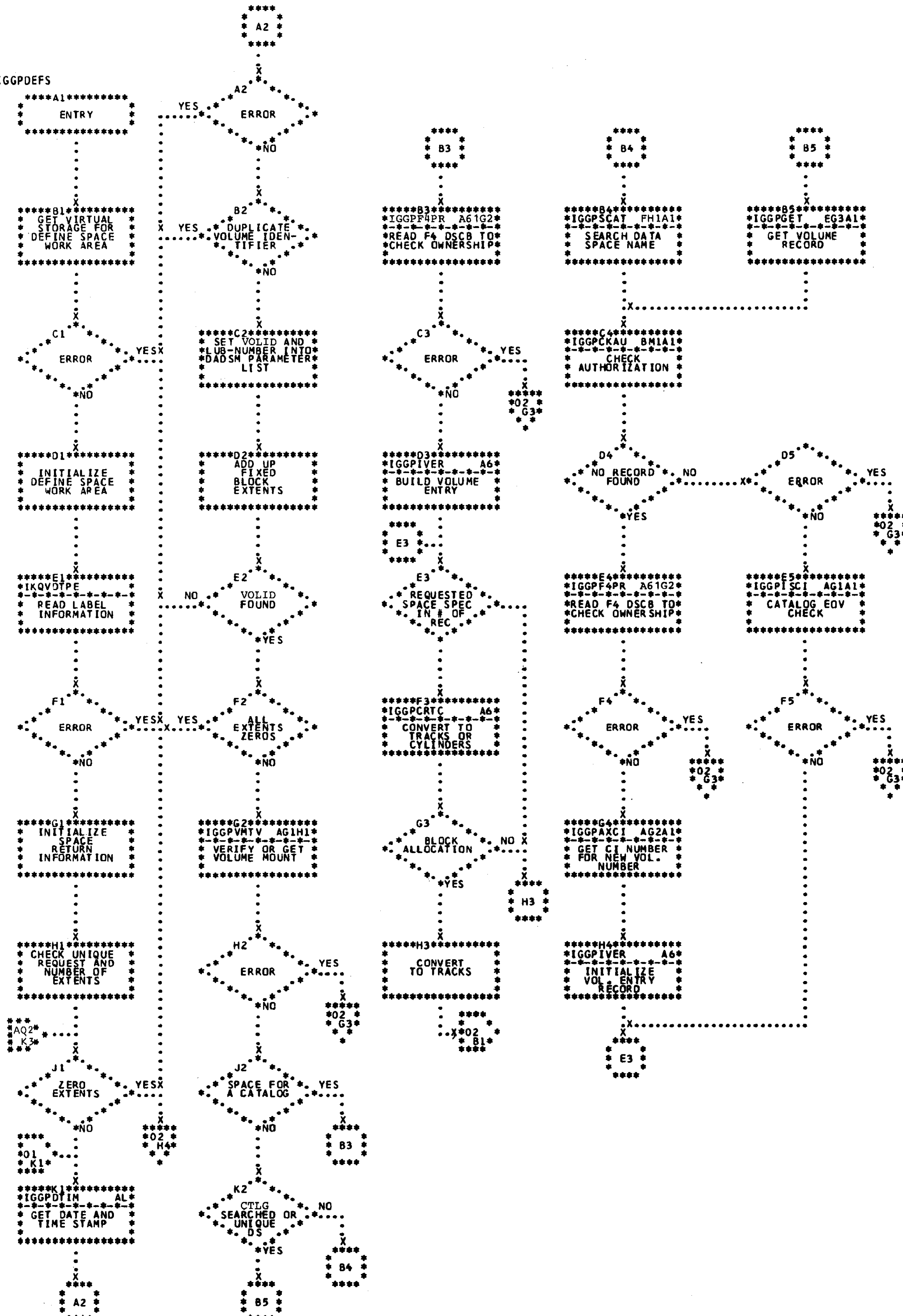


Chart AQ1. CMS Define Space - First Load (IGG0CLAQ)

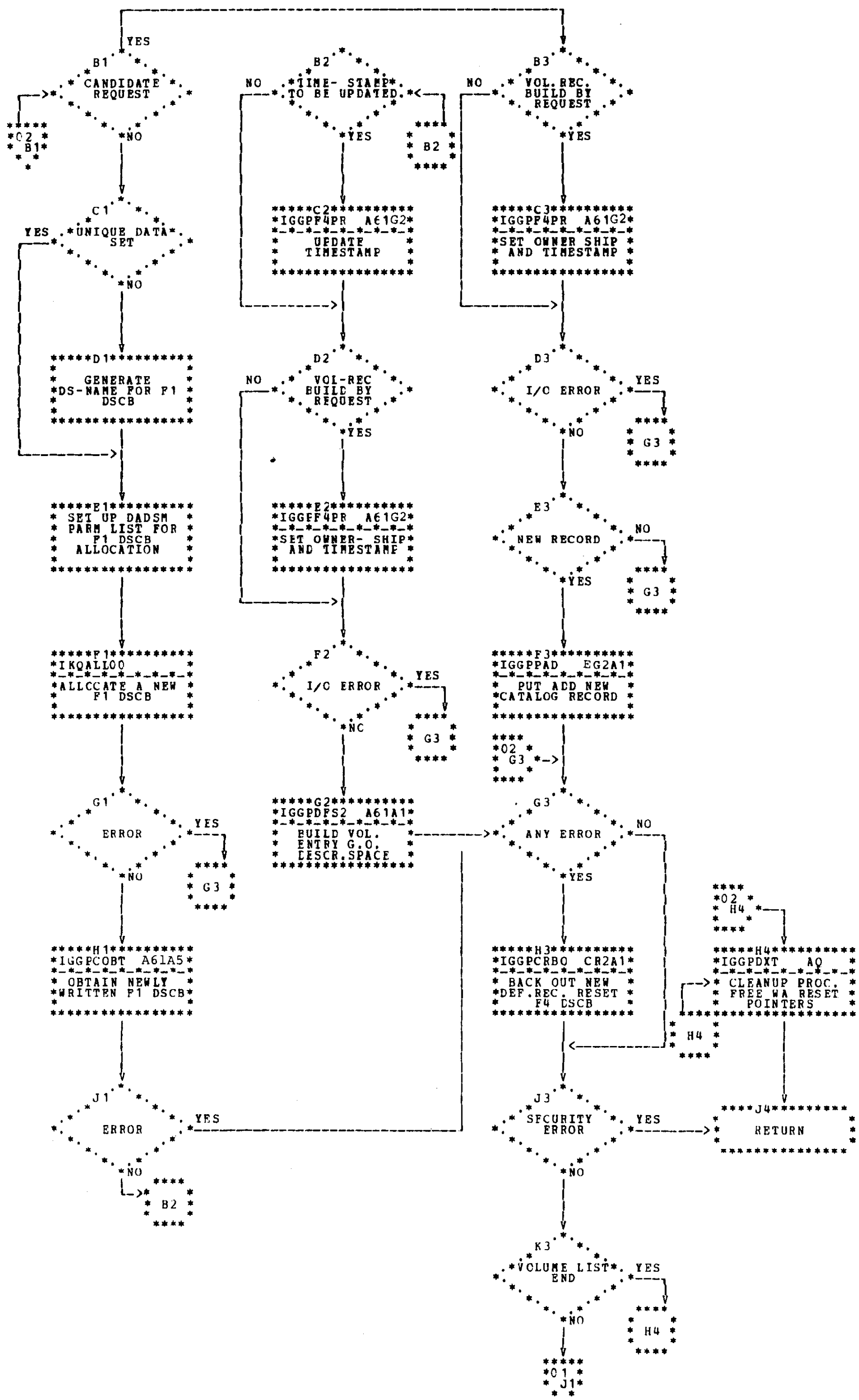


Chart AQ2. CMS Define Space - First Load (IGG0CLAQ)

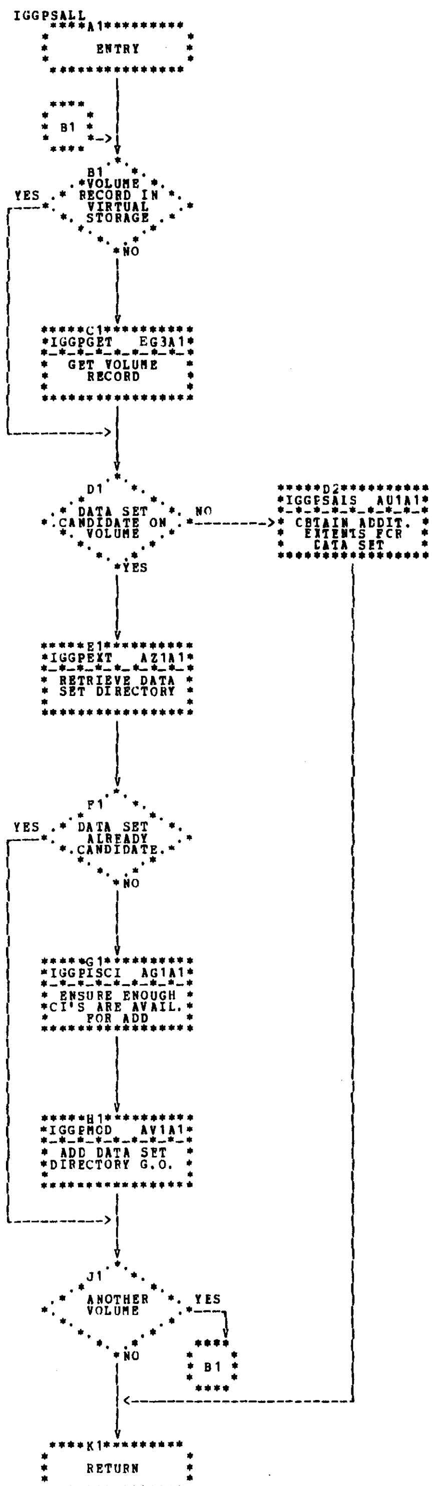
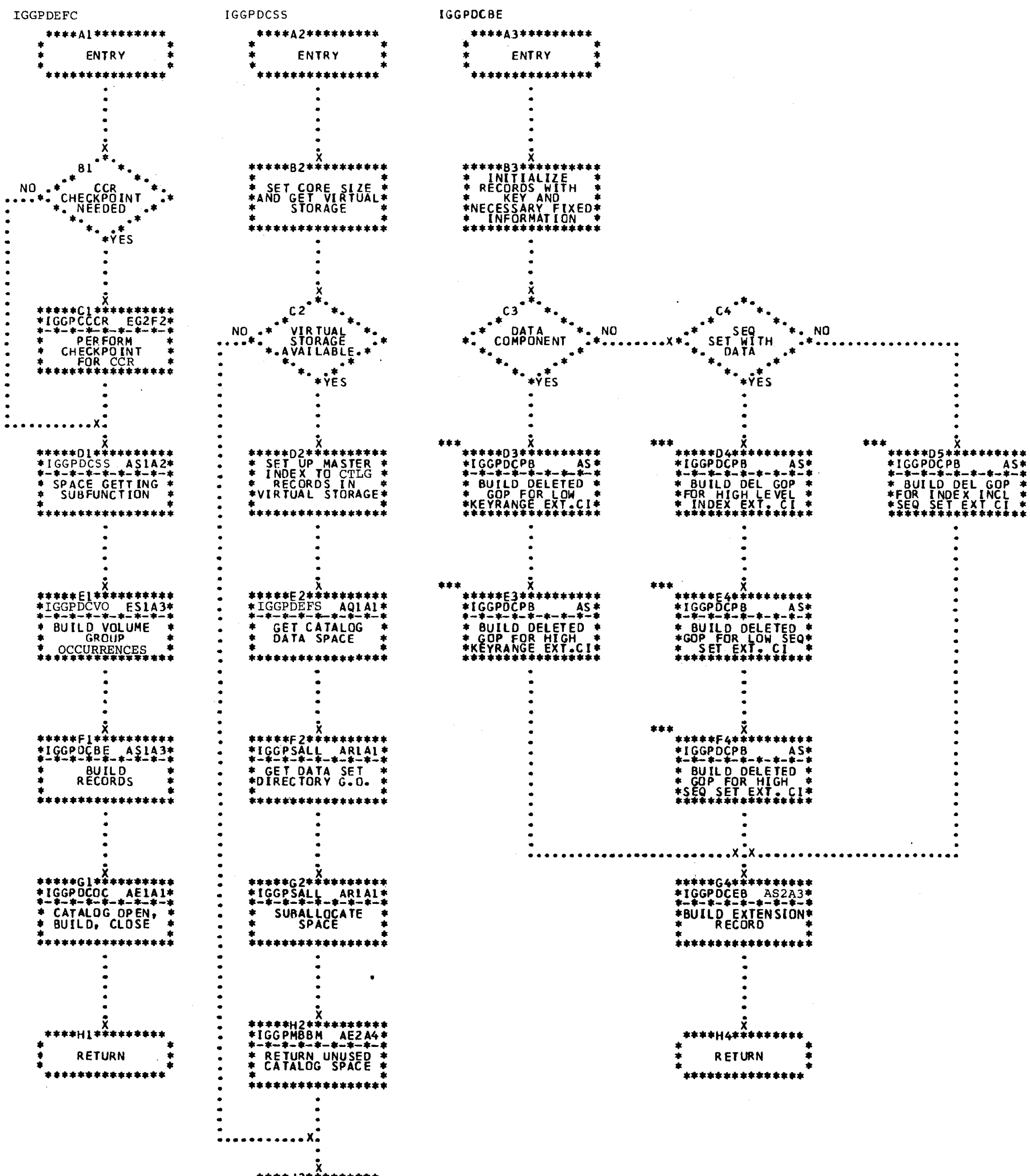


Chart AR1. Space Suballocation (IGG0CLAR)



*** THIS PROCEDURE BUILDS 'DELETED' GOP'S WHICH POINT TO THE D AND I EXTENSION RECORDS IN THE SELF-DESCRIBING PART OF THE CATALOG, AND CAN BE ACTIVATED WHEN THEY ARE NEEDED.

Chart AS1. Catalog Definition Processing (IGG0CLAS)

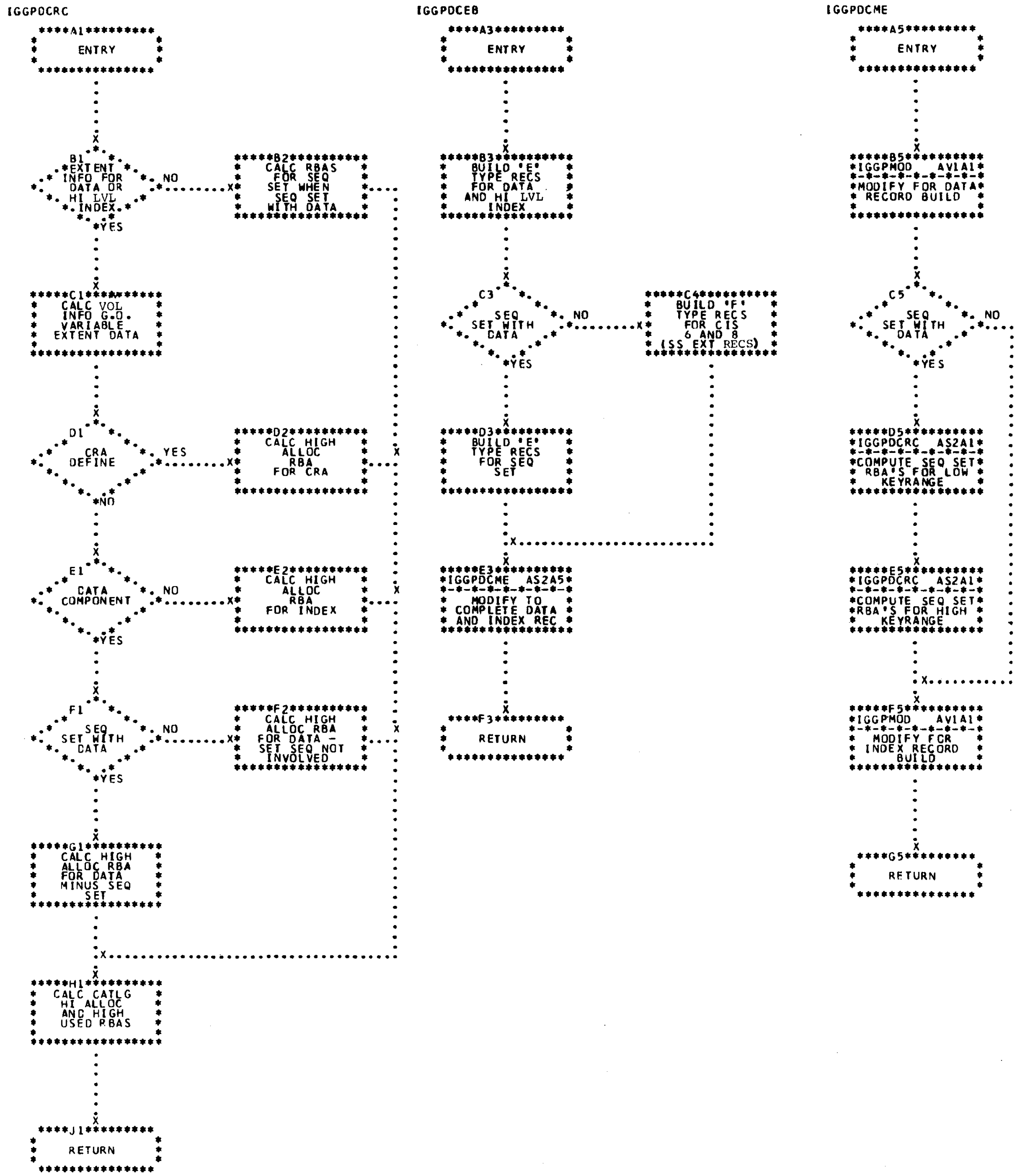


Chart AS2. Catalog Definition Processing (IGG0CLAS)

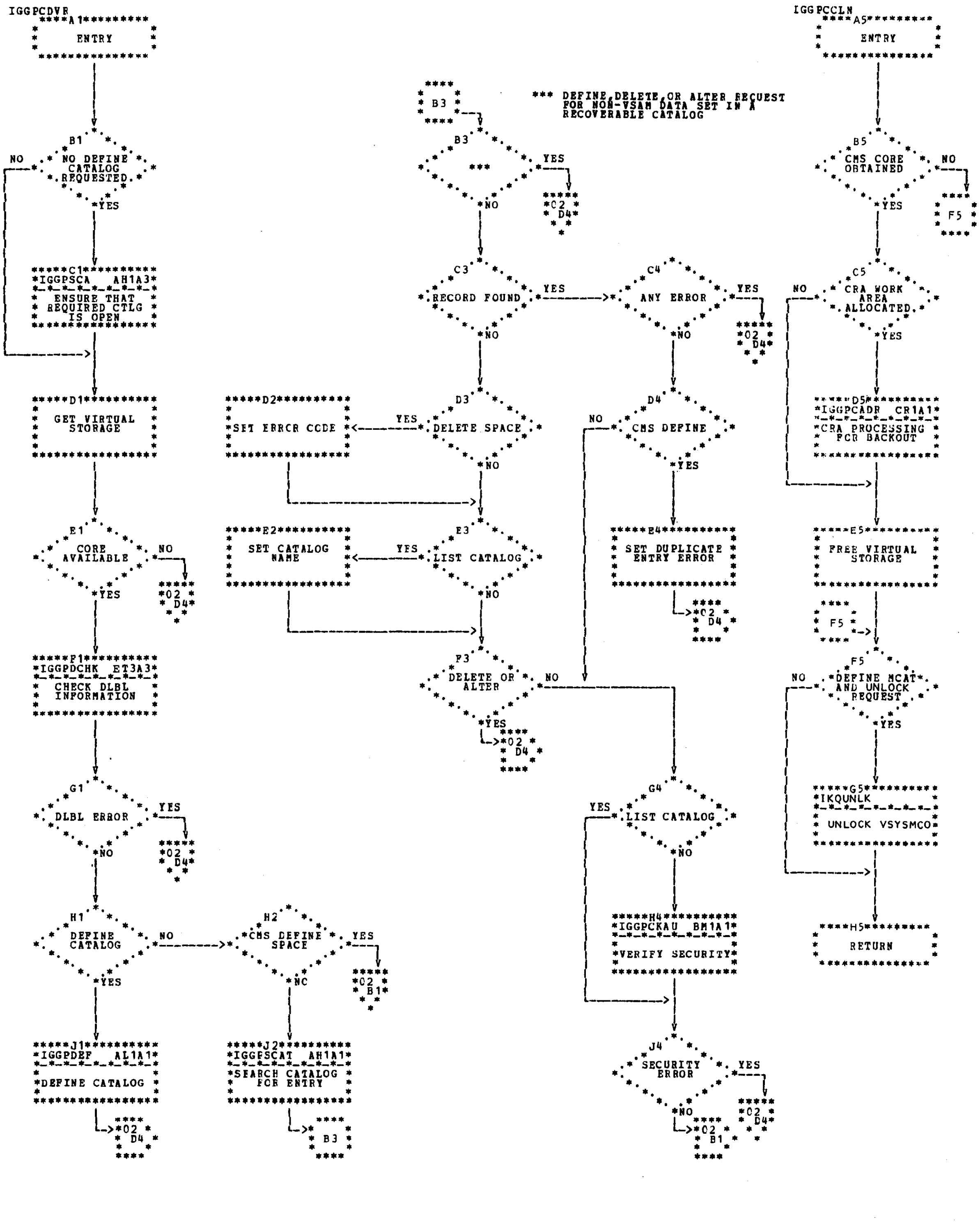


Chart AT1. CMS Driver (IGG0CLAT)

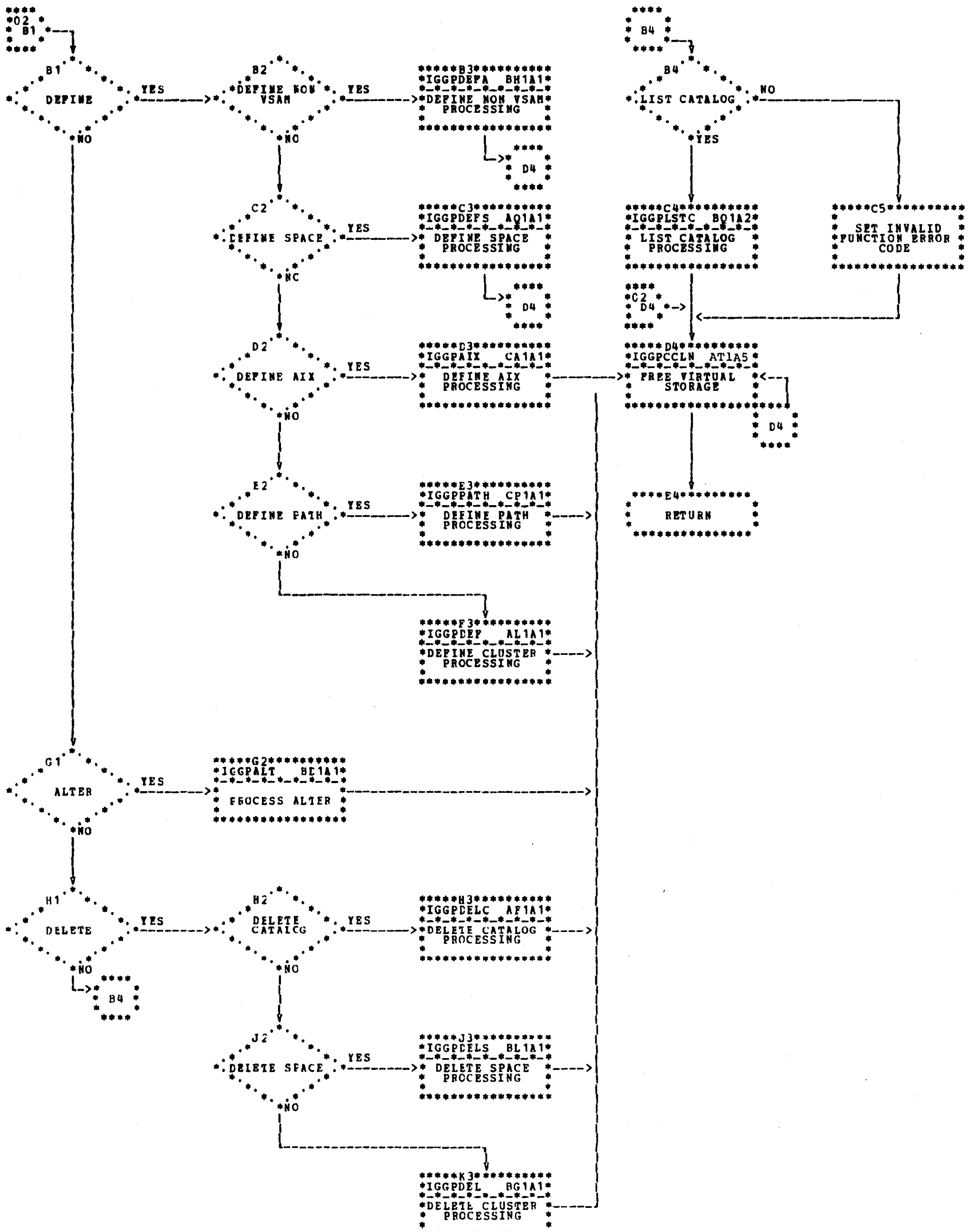


Chart AT2. CMS Driver (IGG0CLAT)

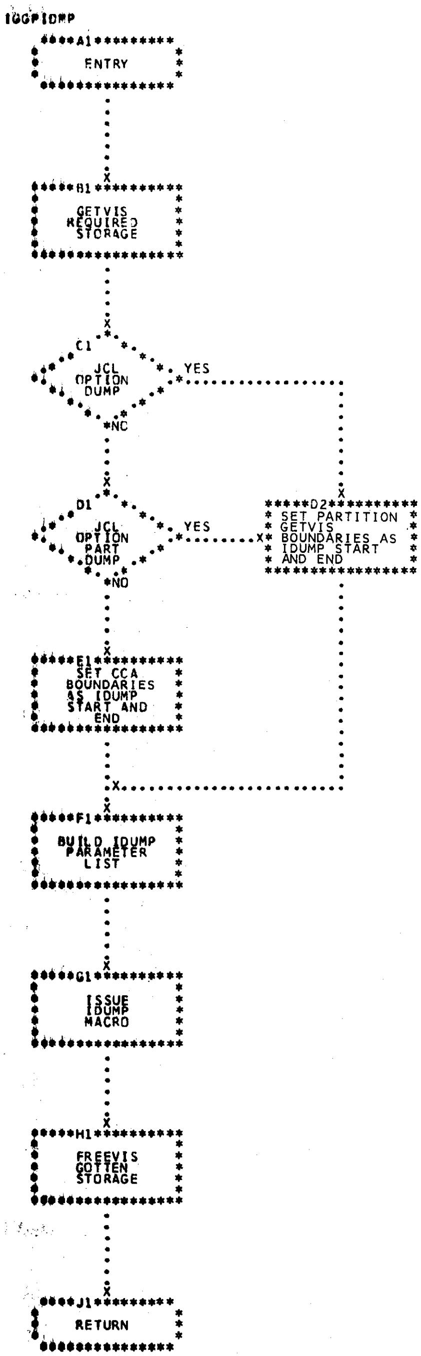


Chart AT3. CMS Driver (IGG0CLAT)

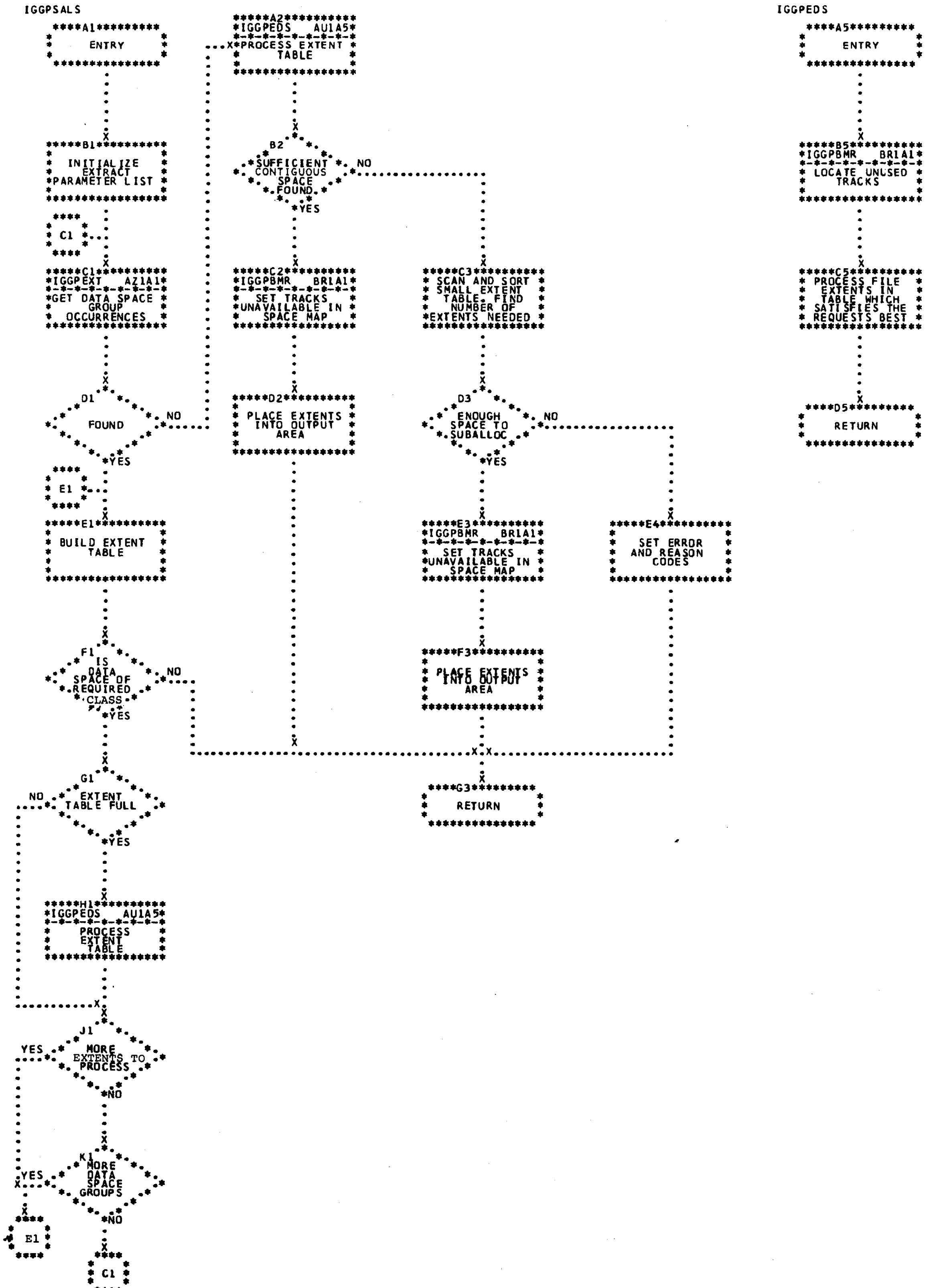


Chart AU1. Suballocation Routine (IGG0CLAU)

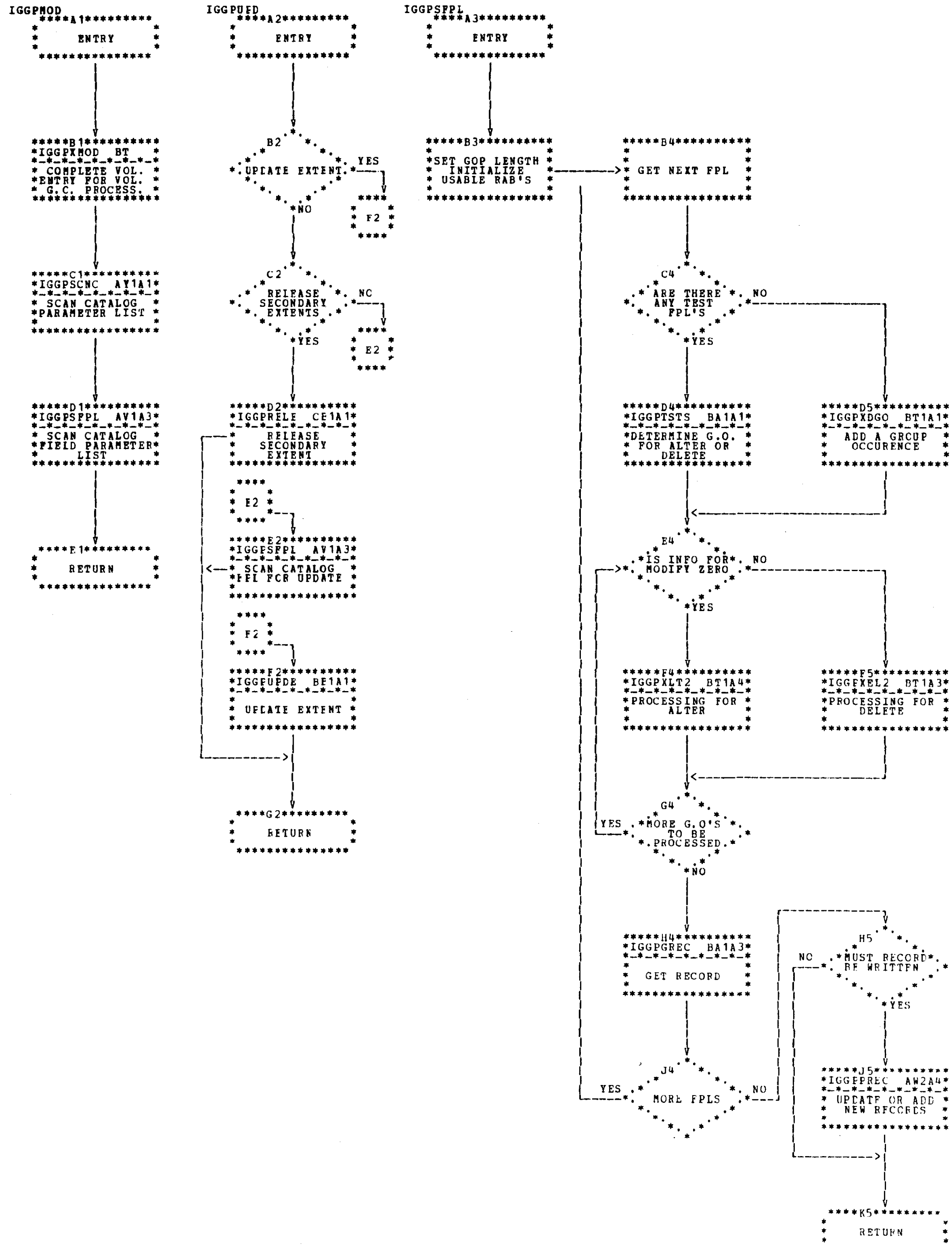


Chart AV1. Modify Catalog Field (IGG0CLAV)

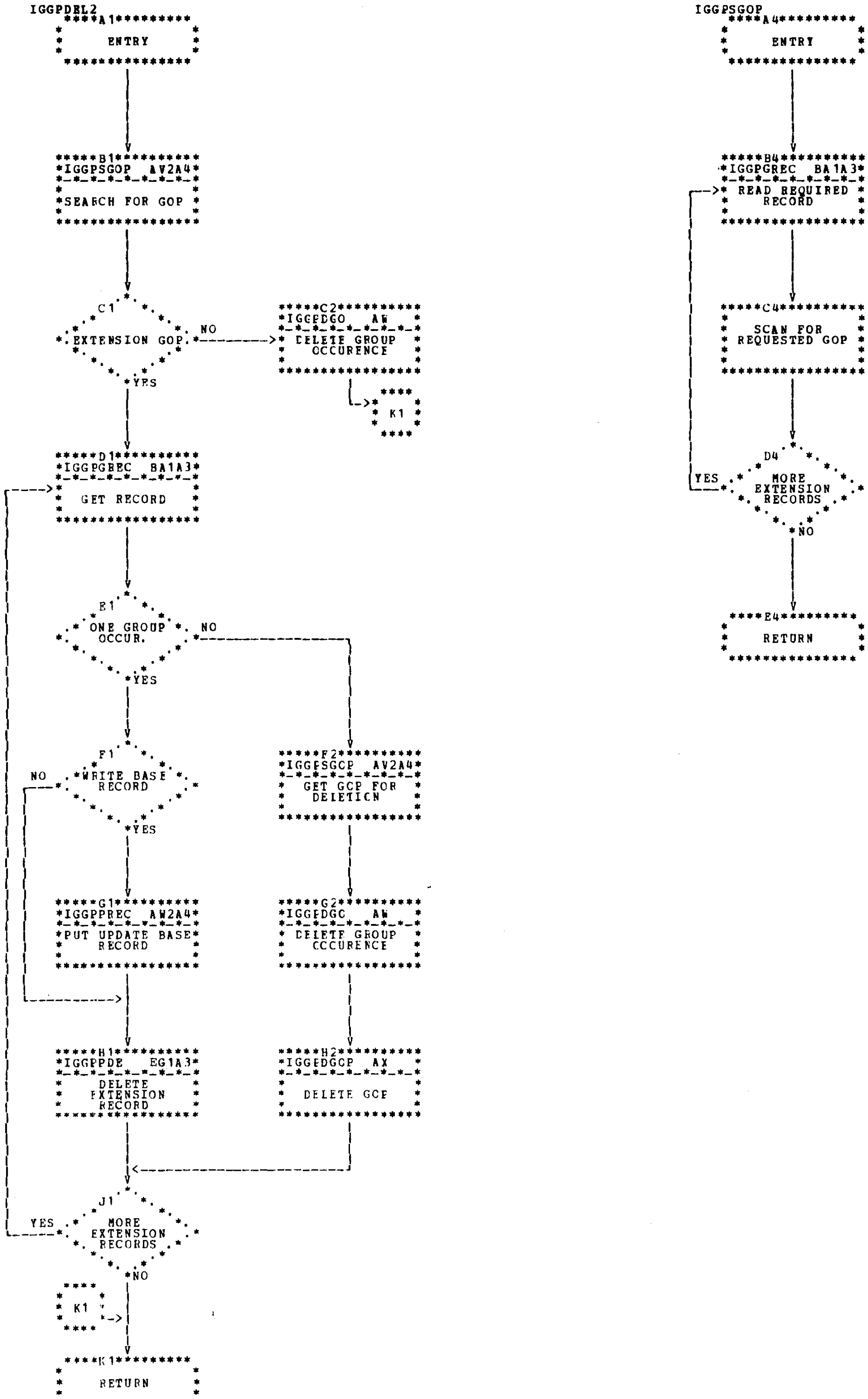


Chart AV2. Modify Catalog Field (IGG0CLAV)

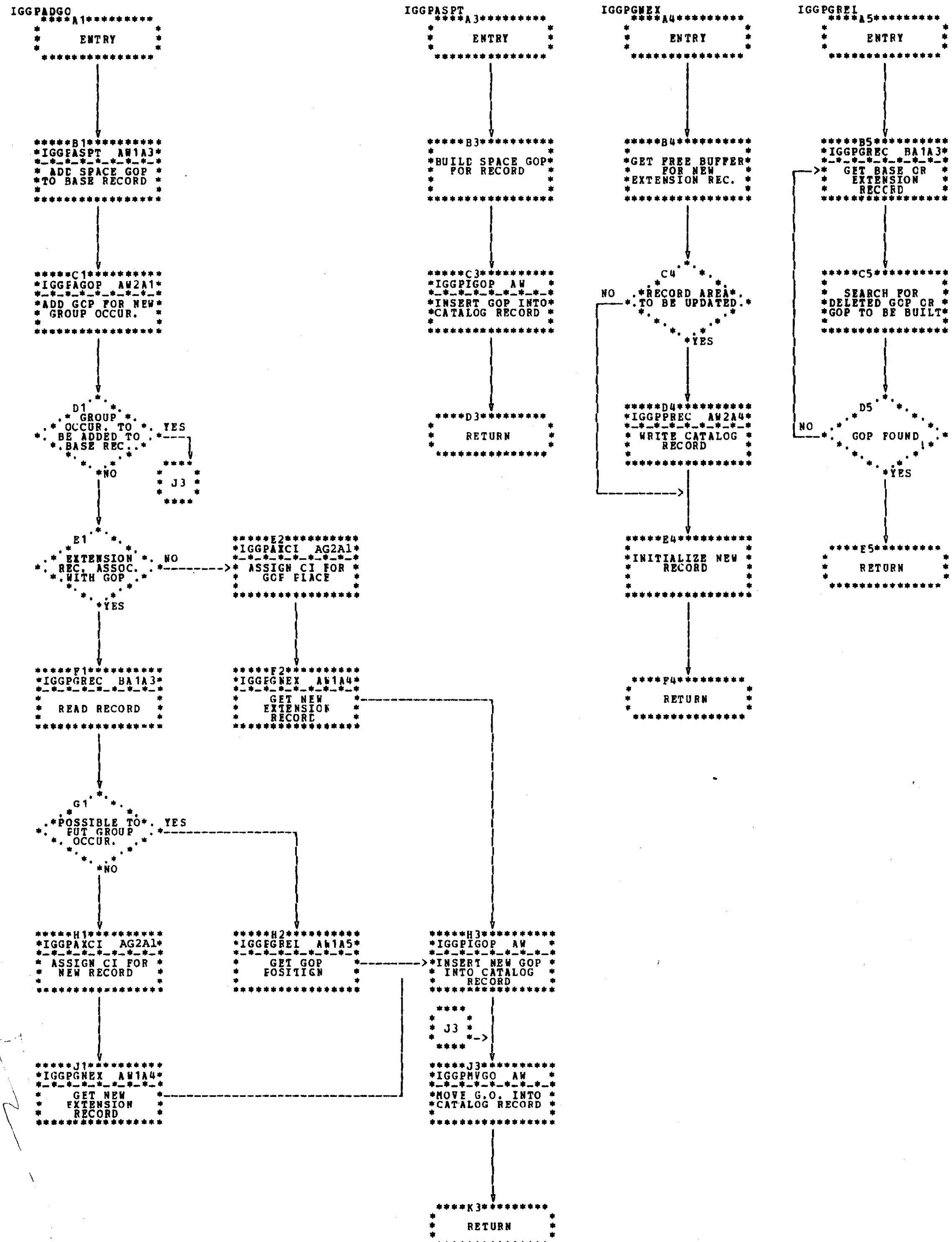


Chart AW1. Add Group Occurrence (IGG0CLAW)

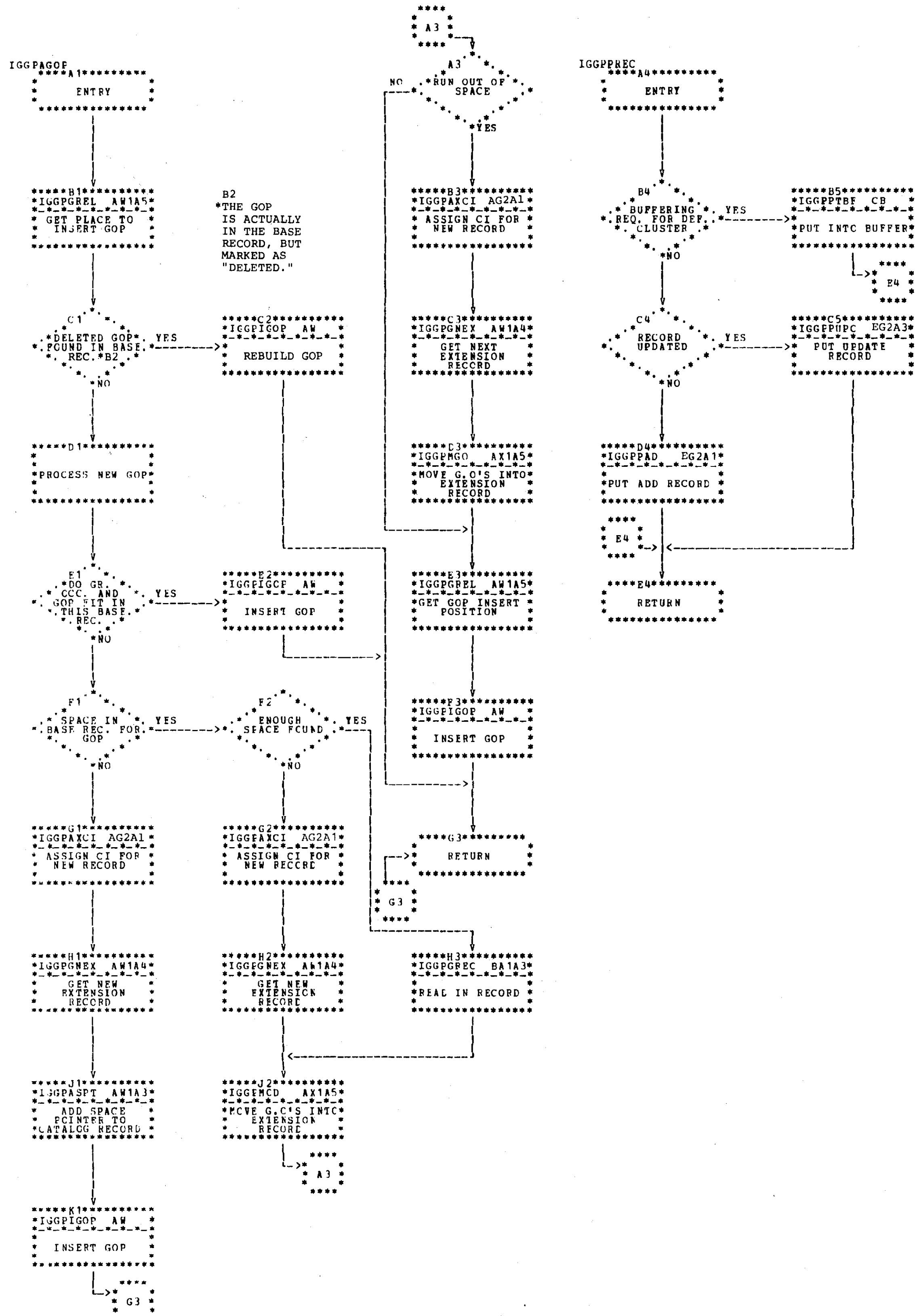


Chart AW2. Add Group Occurrence (IGG0CLAW)

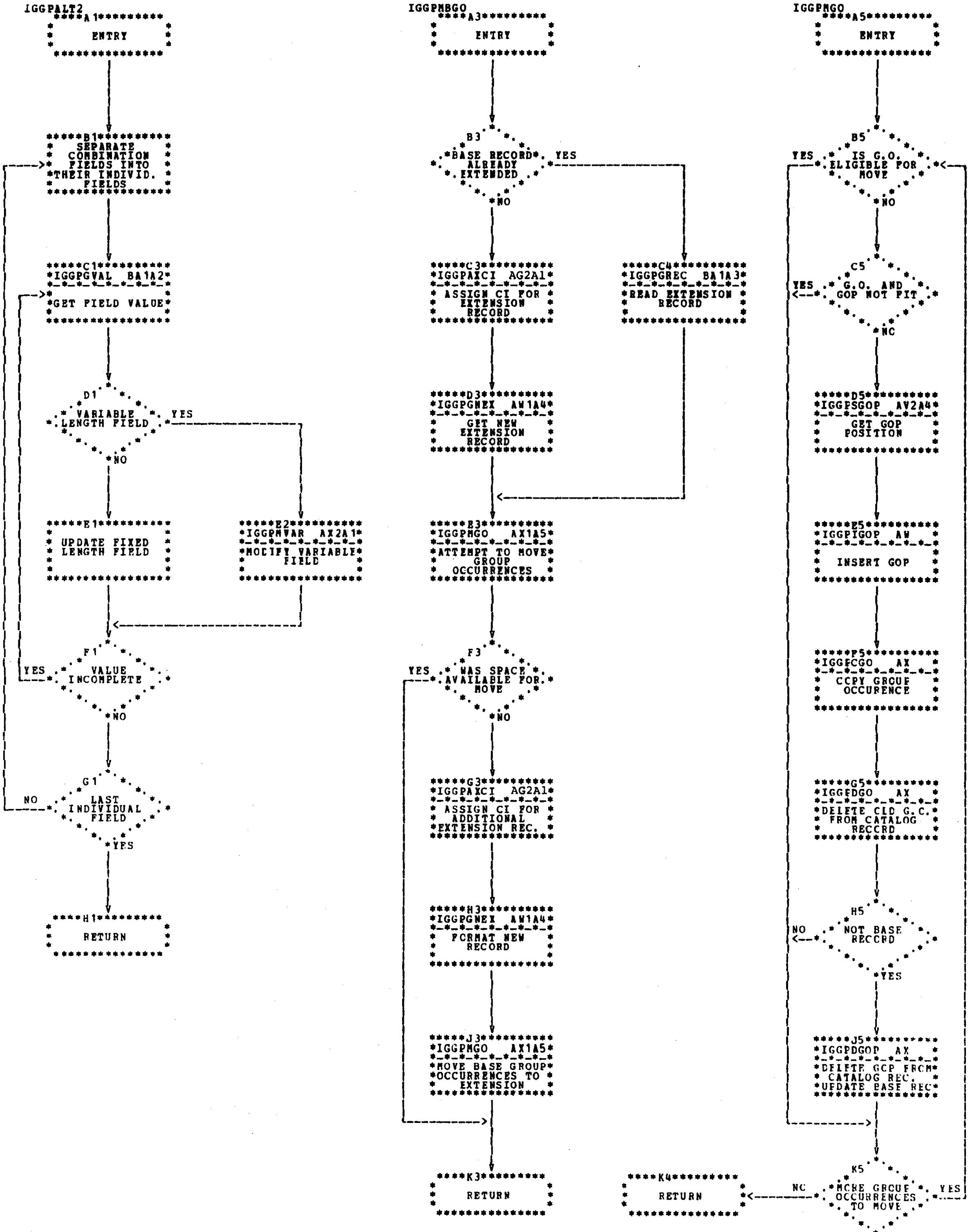


Chart AX1. Alter Catalog Field (IGG0CLAX)

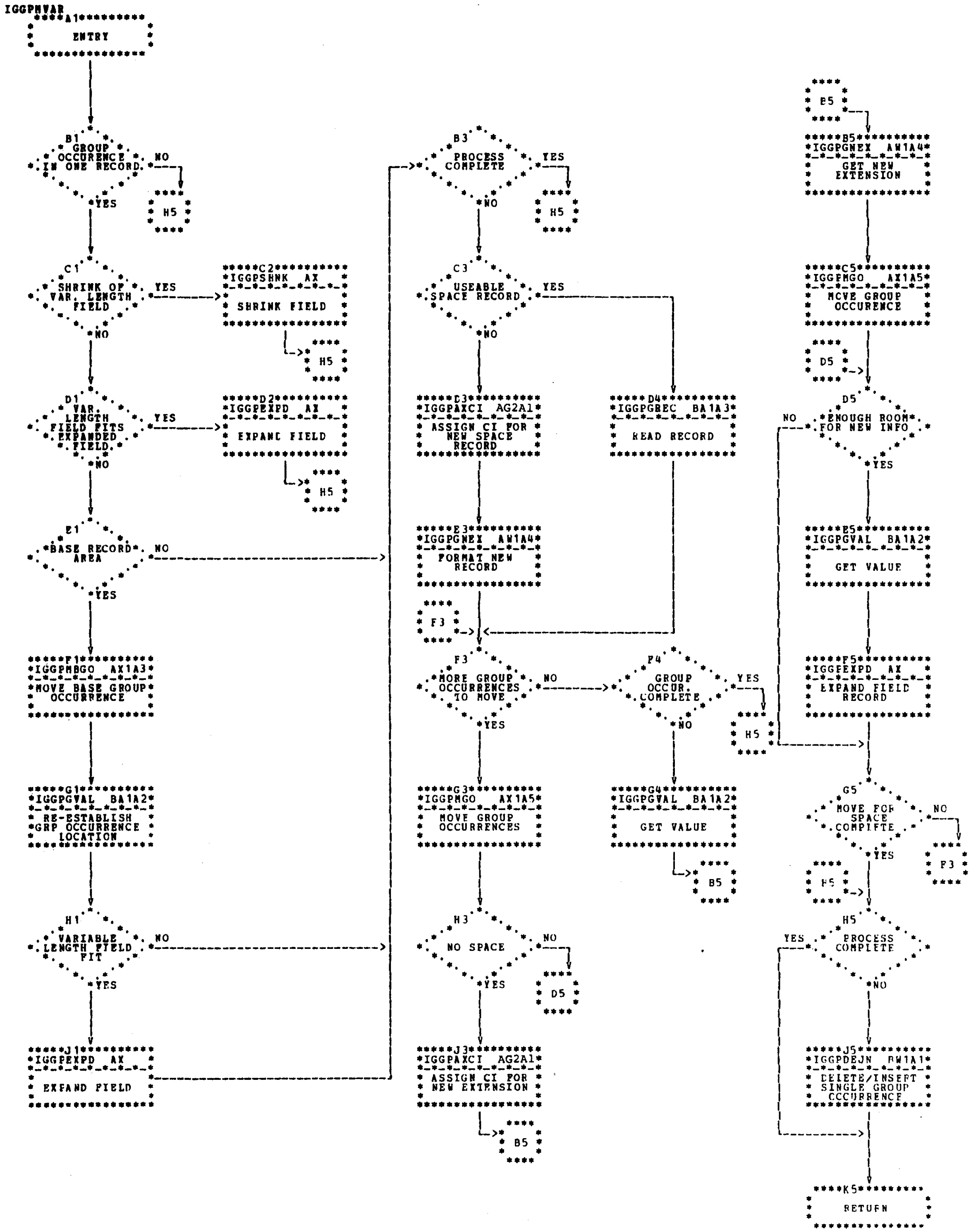


Chart AX2. Alter Catalog Field (IGG0CLAX)

```
IGGPSCNC
    ****A 1**********
    *               *
    *     ENTRY     *
    *               *
    *****************
            |
            |
            v
   *****B1**********
   *INITIALIZE CCA *
   *  TO POINT TO  *
   *    CATALOG    *
   *  DICTIONARY   *
   *               *
   *****************
            |
            |
            v
   *****C1**********
   *               *
   * SCAN CPL AND  *
   *FPL FOR FIELD- *
   *     NAMES     *
   *               *
   *****************
            |
            |
            v
   *****D1**********
   *               *
   *SCAN DICTIONARY*
   *WITH FIELD NAME*
   *               *
   *               *
   *****************
            |
            |
            v
   *****E1**********
   *    INSERT     *
   *  DICTIONARY   *
   *  INFORMATION  *
   *   INTO FPL    *
   *               *
   *****************
            |
            |
            v
    ****F1**********
    *               *
    *    RETURN     *
    *               *
    *****************
```

Chart AY1. Scan Catalog Parameter List (IGG0CLAY)

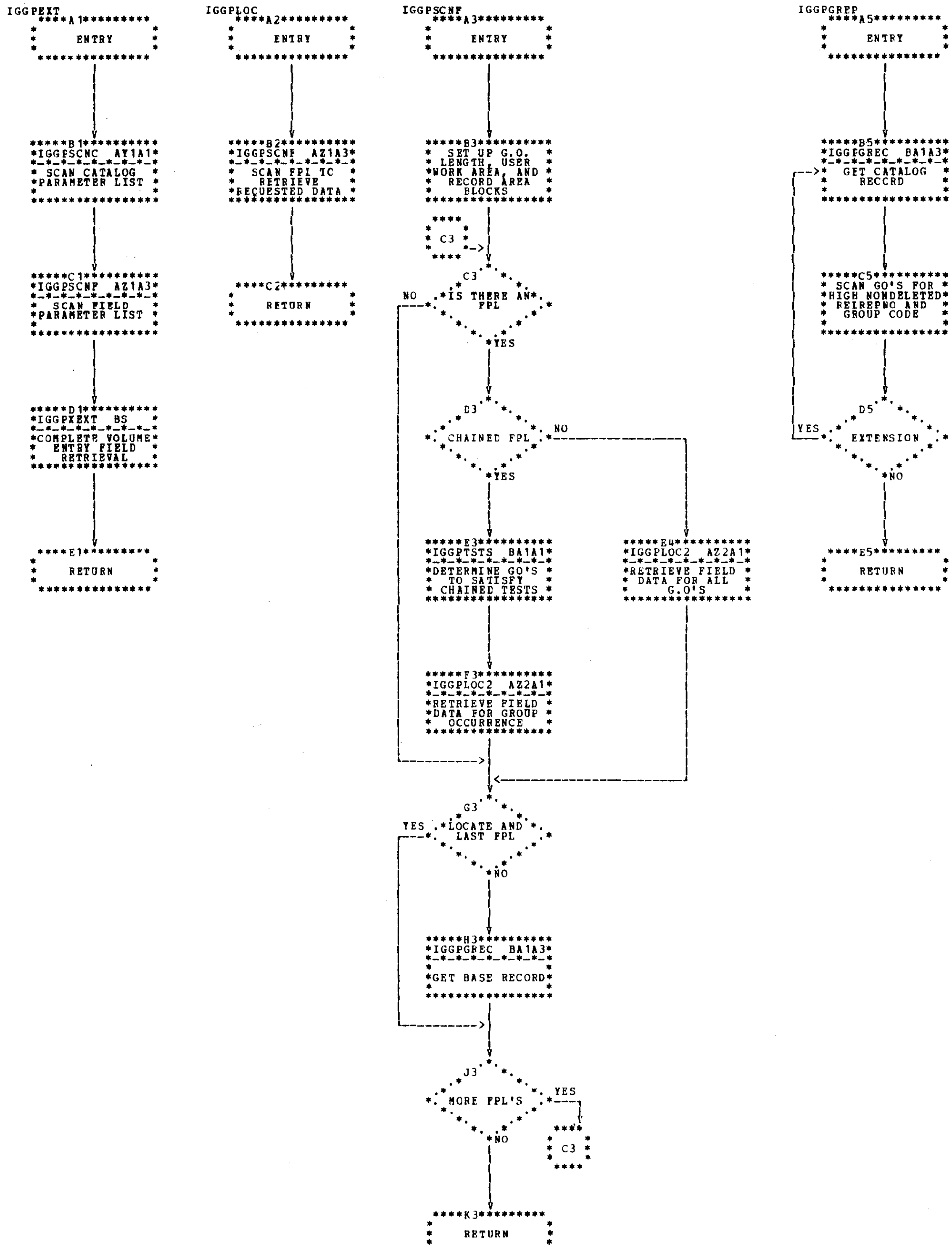


Chart AZ1. Extract Catalog Field (IGG0CLAZ)

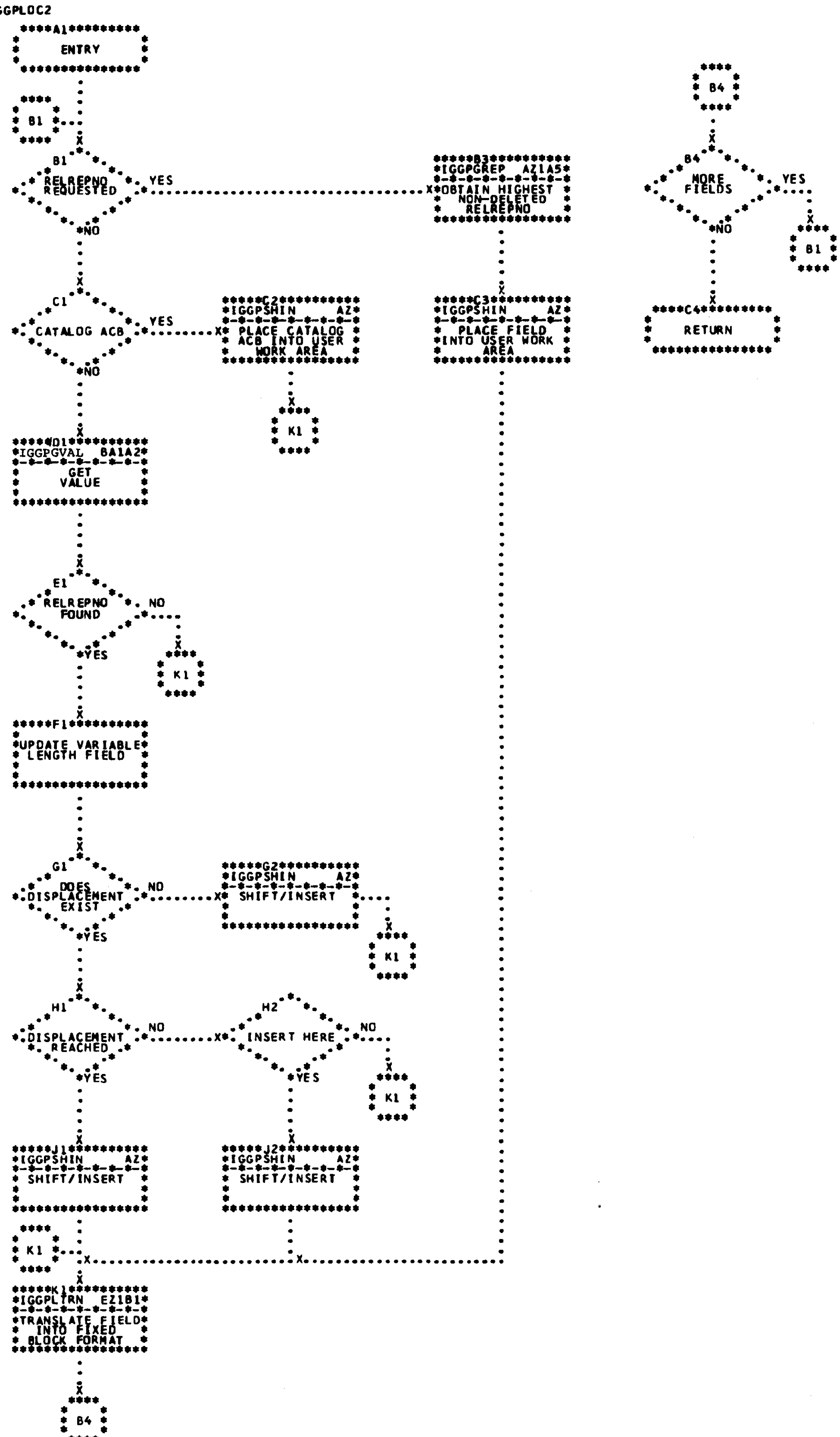


Chart AZ2. Extract Catalog Field (IGG0CLAZ)

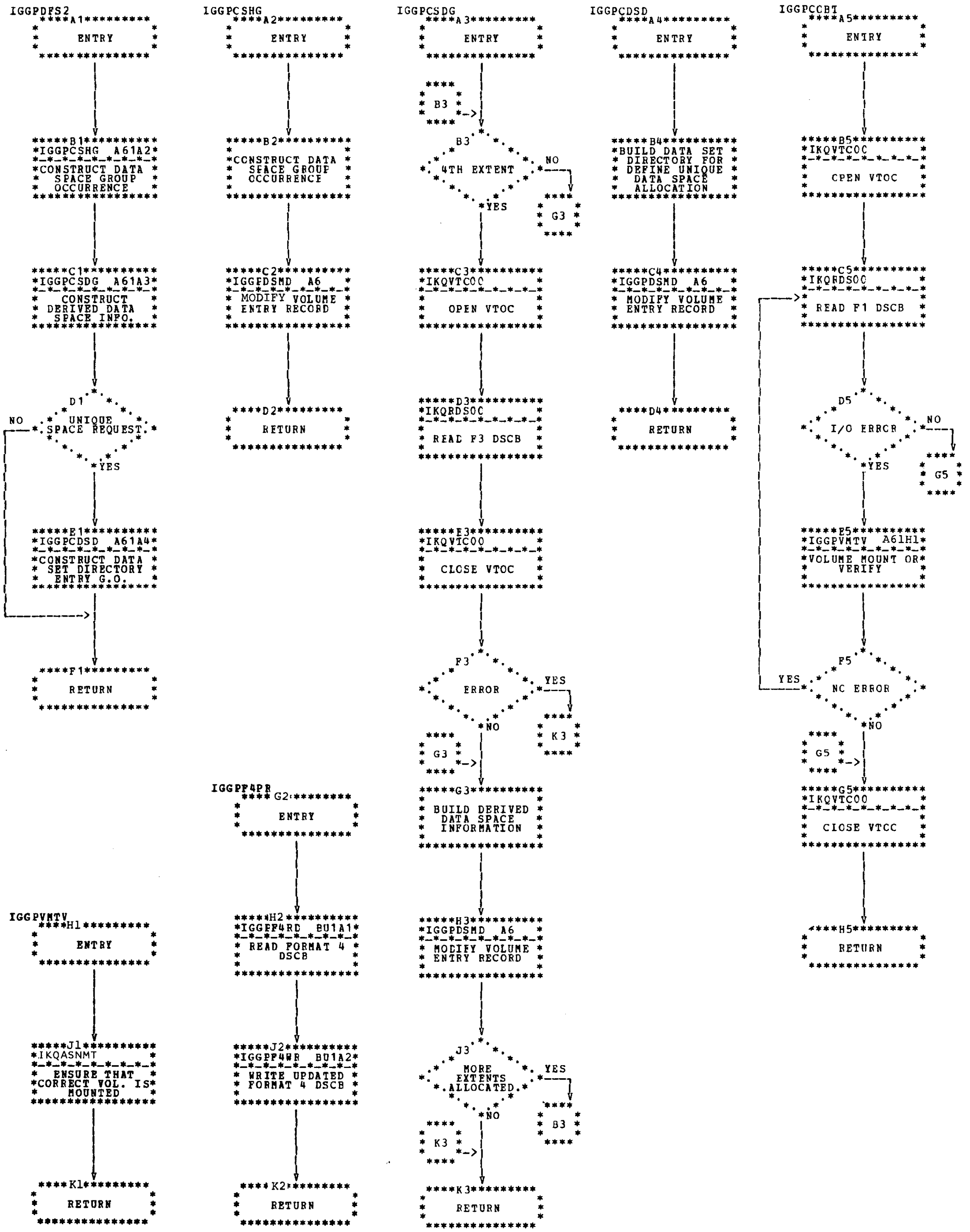


Chart A61. Define Space - Second Load (IGG0CLA6)

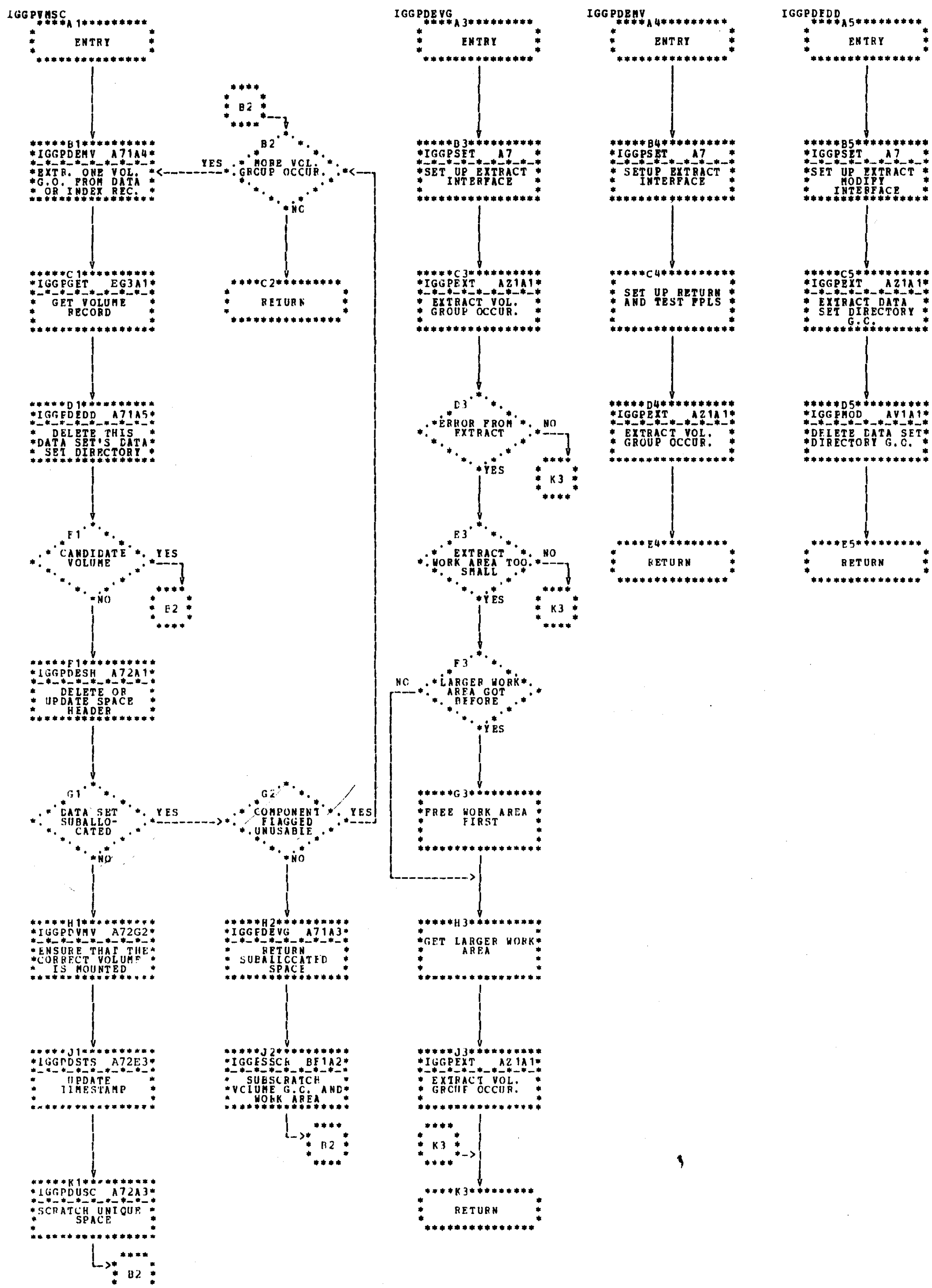


Chart A71. CMS Delete - Second Module (IGG0CLA7)

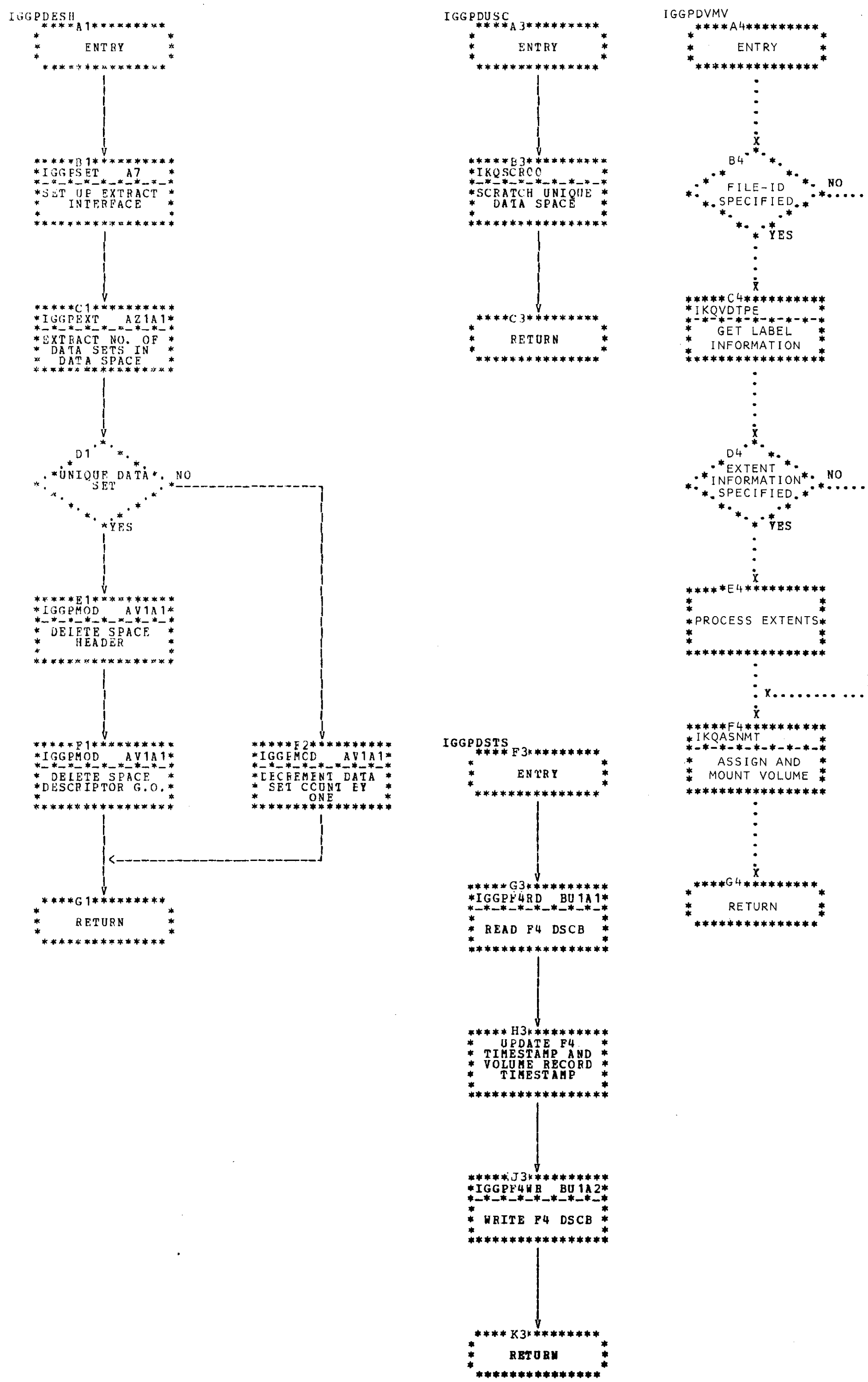


Chart A72. CMS Delete - Second Module (IGG0CLA7)

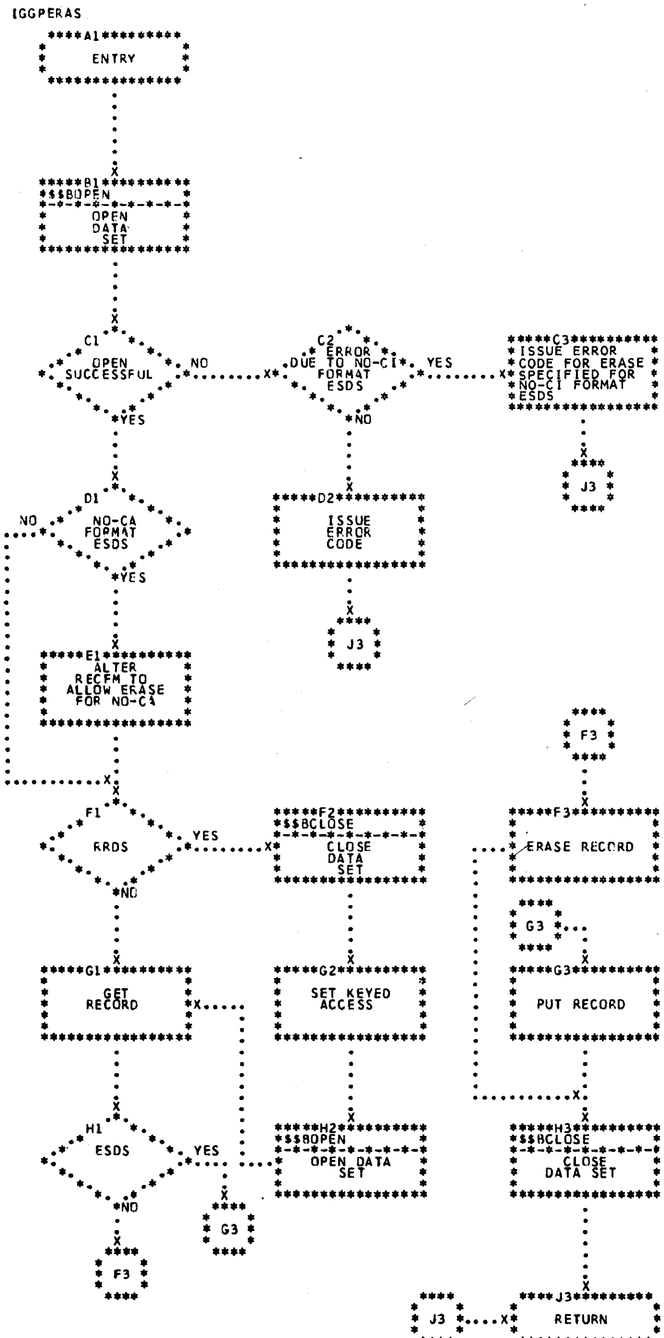


Chart A73. CMS Delete - Second Module (IGG0CLA7)

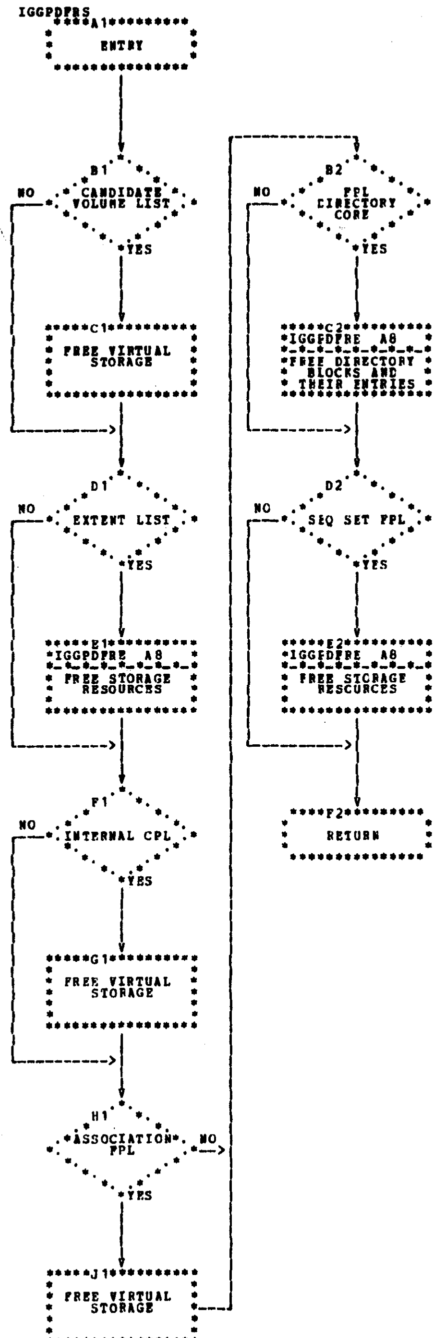


Chart A81. Clean Up of Storage Resources (IGG0CLA8)

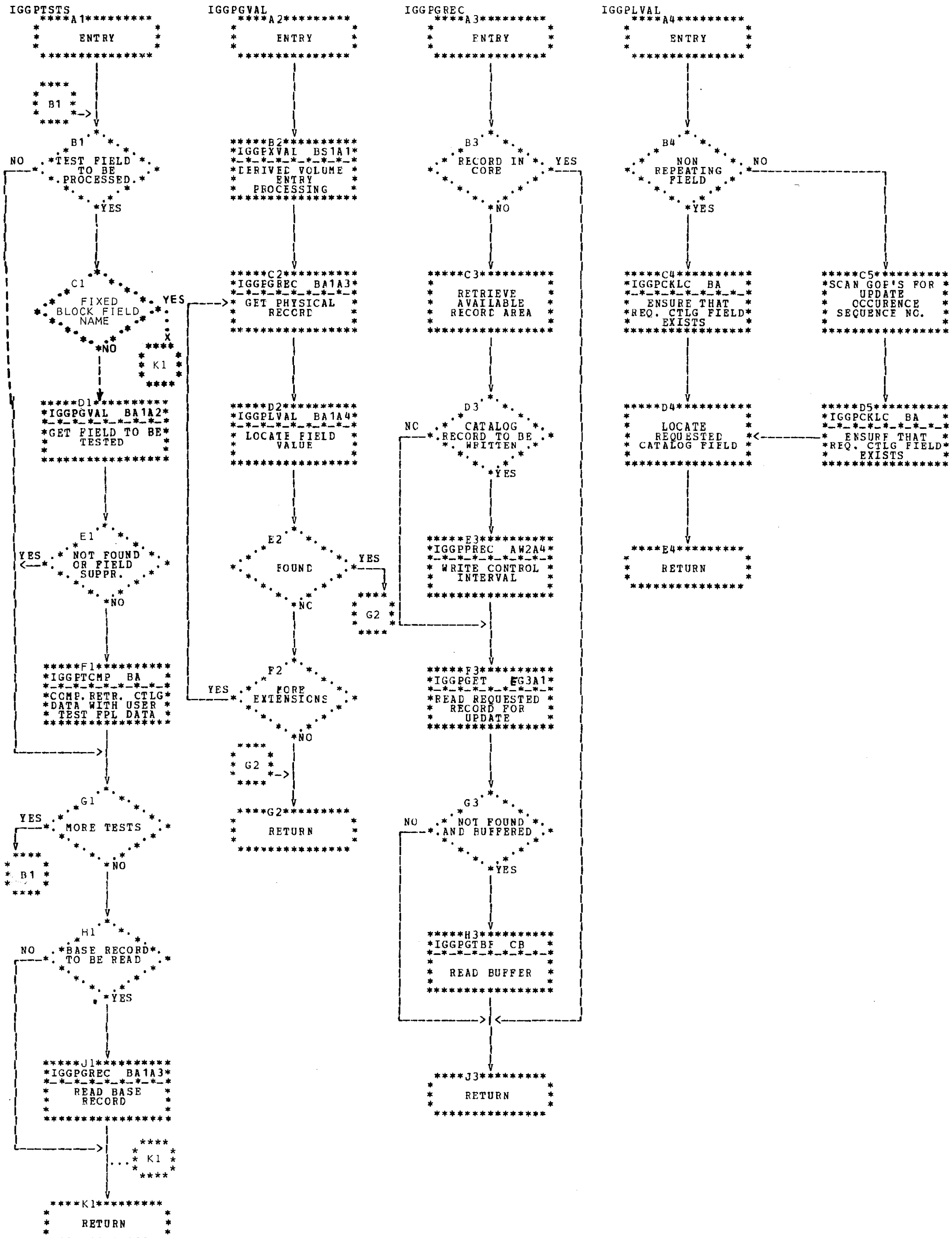


Chart BA1. Tests (IGG0CLBA)

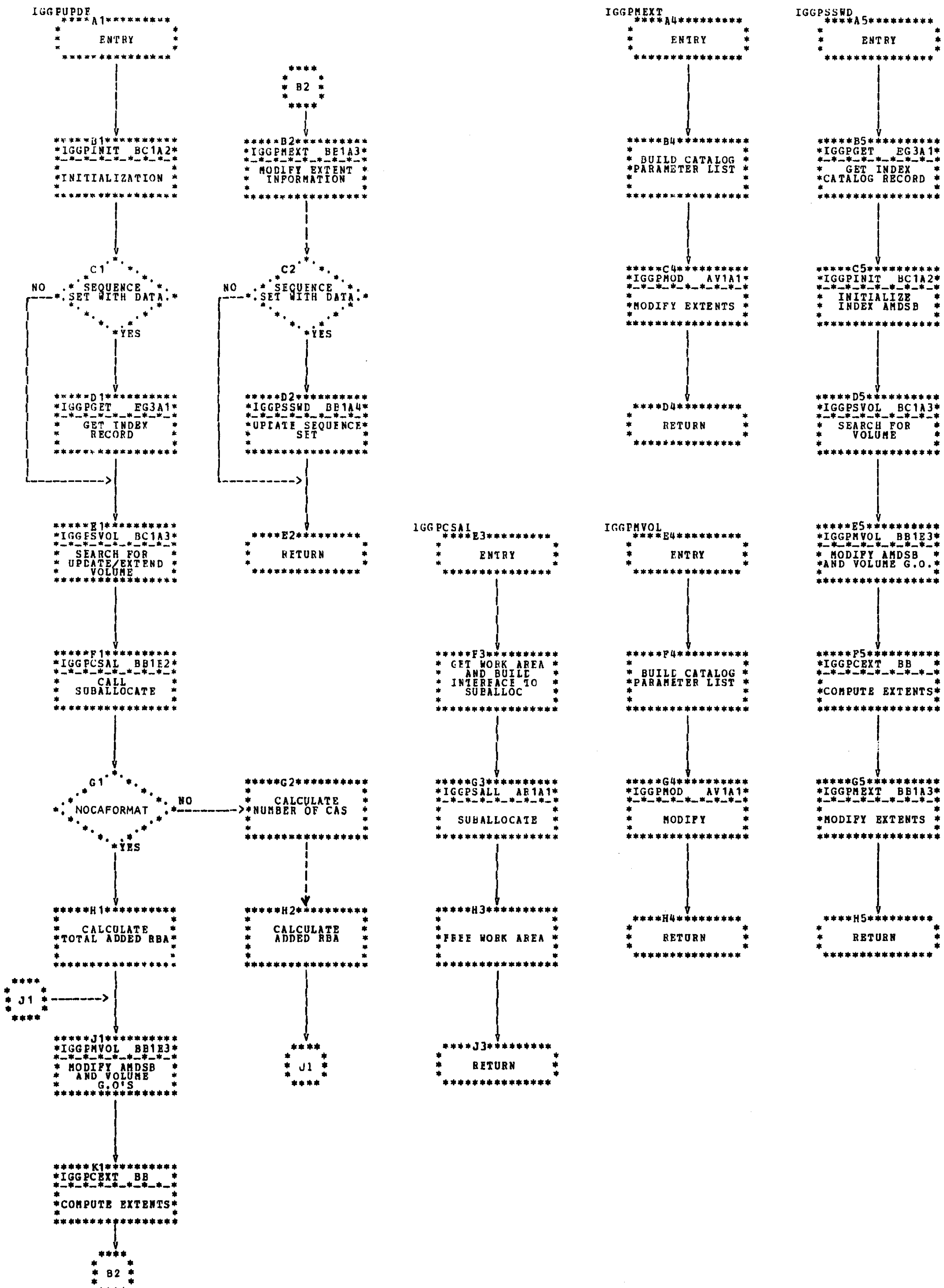


Chart BB1. Update Extend (IGG0CLBB)

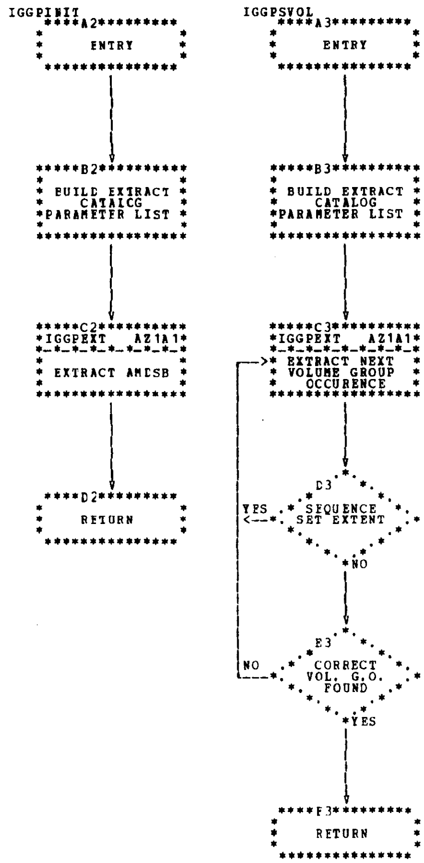


Chart BC1. Update Extend Initialization (IGG0CLBC)

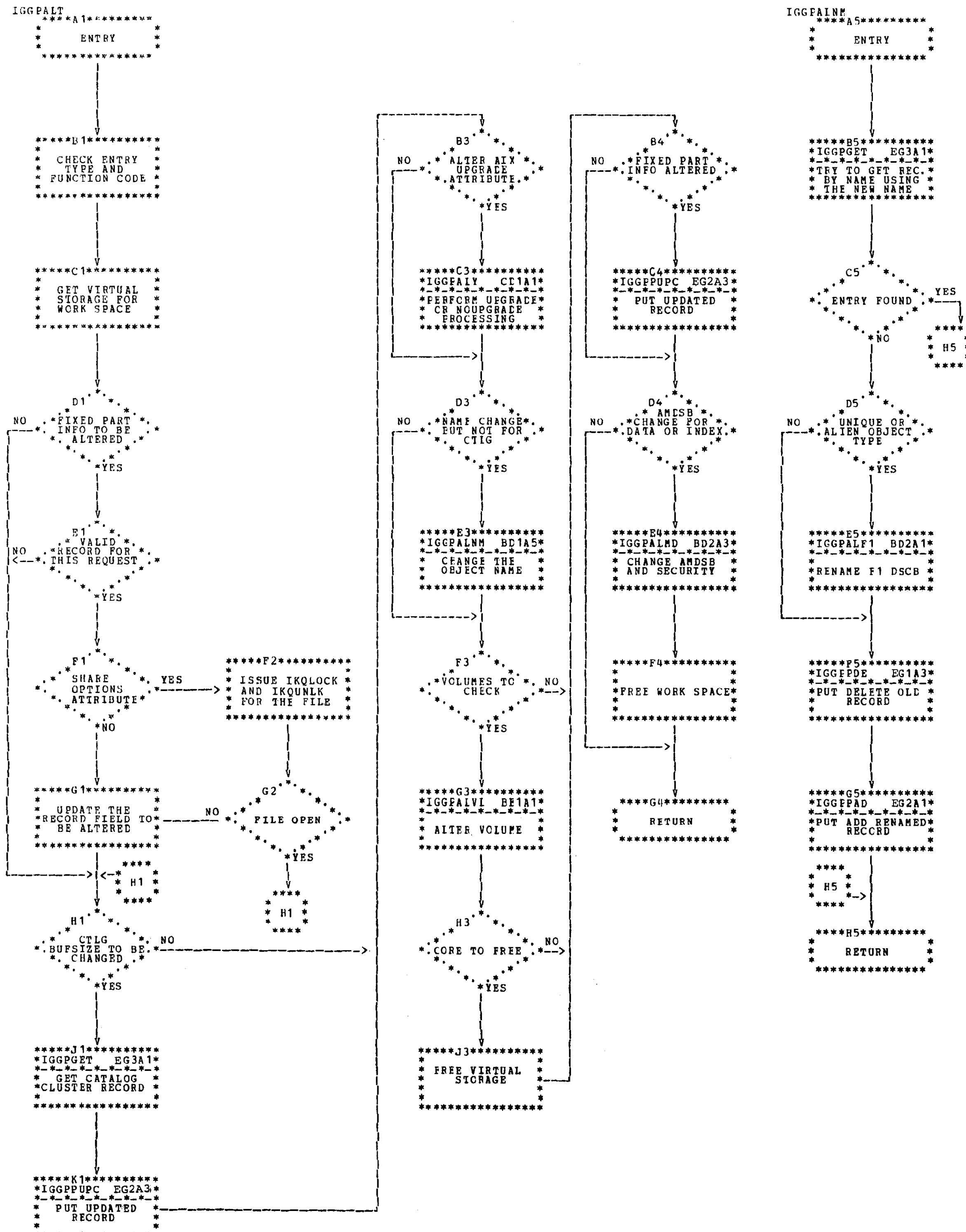


Chart BD1. CMS Alter - First Module (IGG0CLBD)

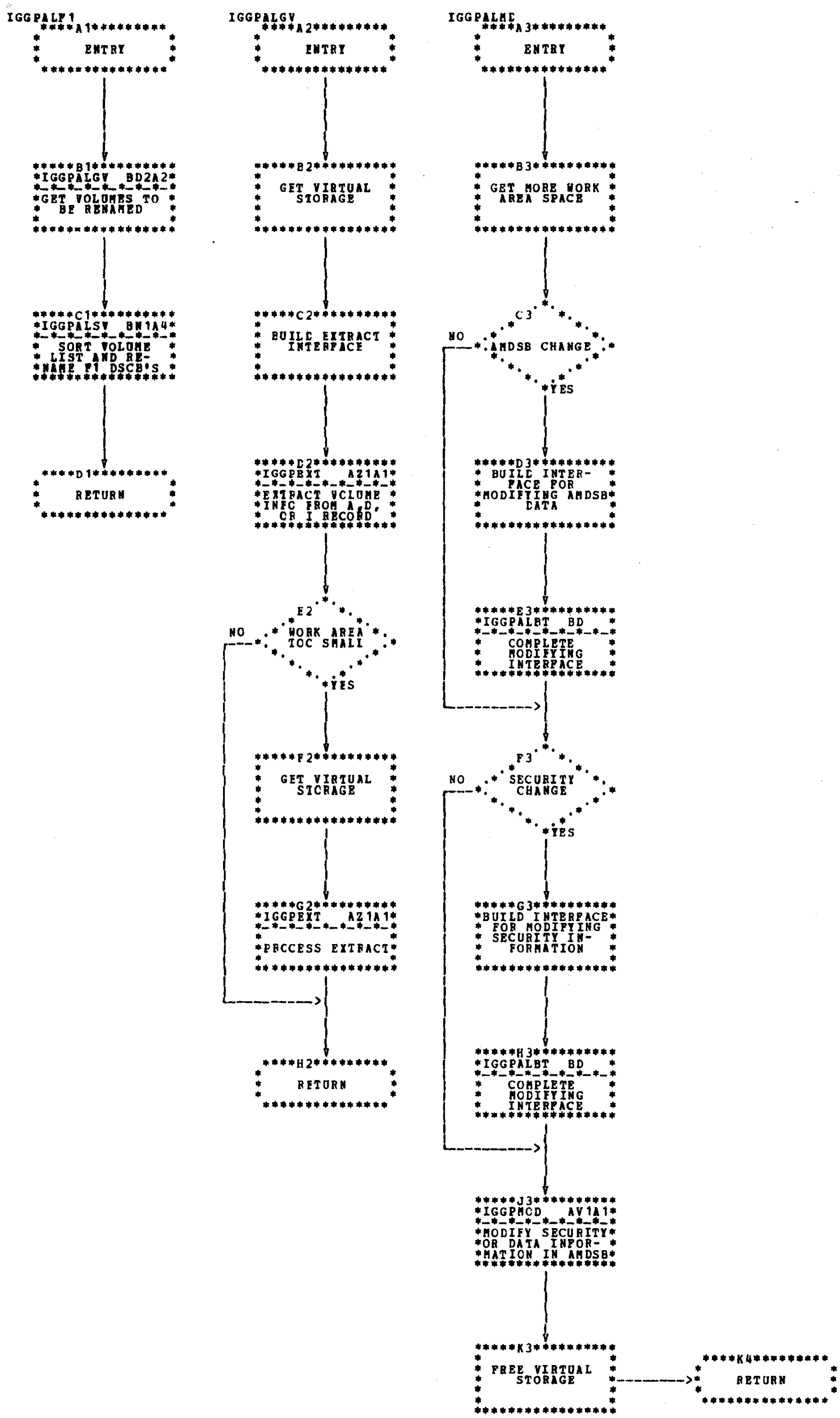


Chart BD2. CMS Alter - First Module (IGG0CLBD)

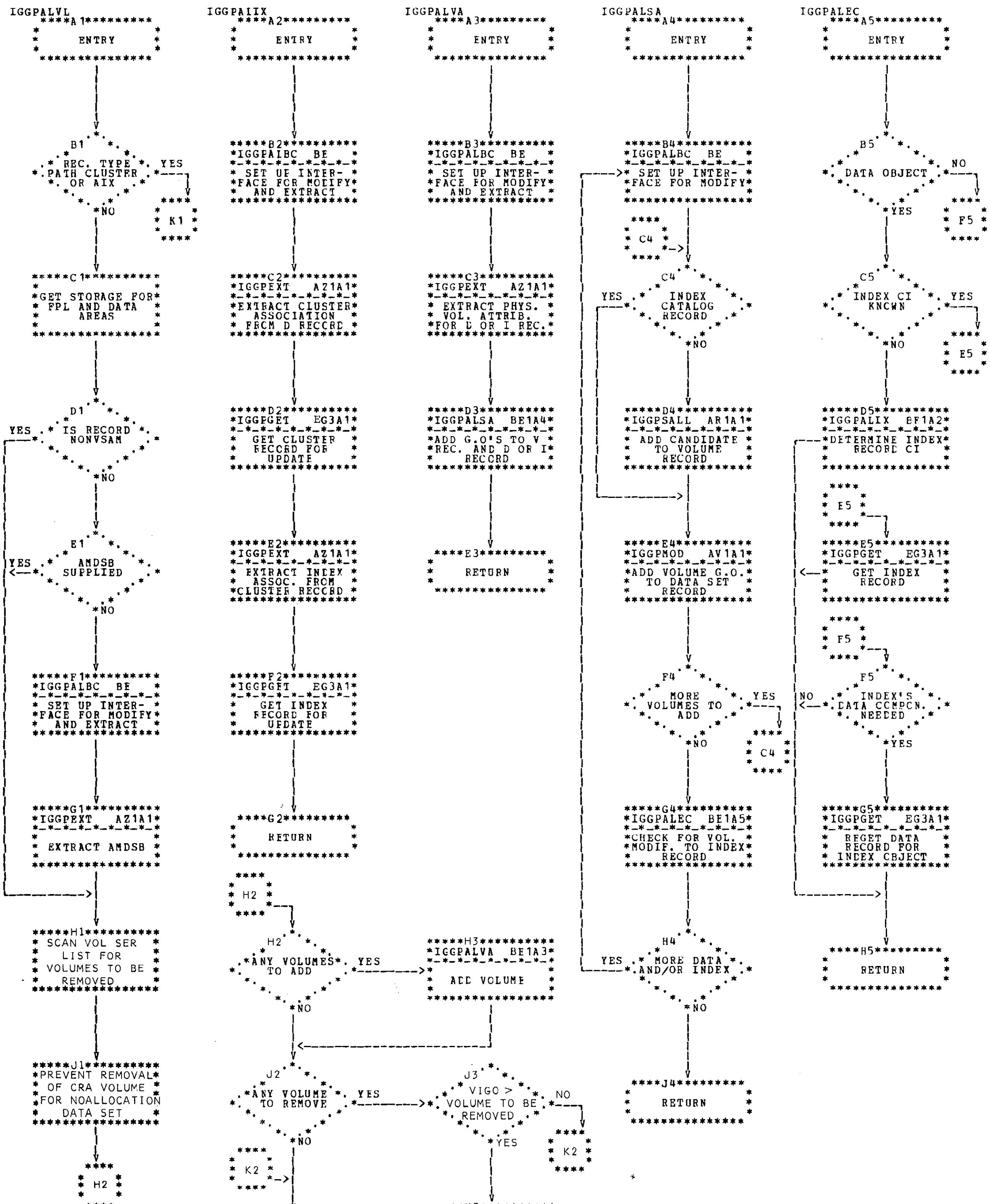


Chart BE1. CMS Alter - Third Module (IGG0CLBE)

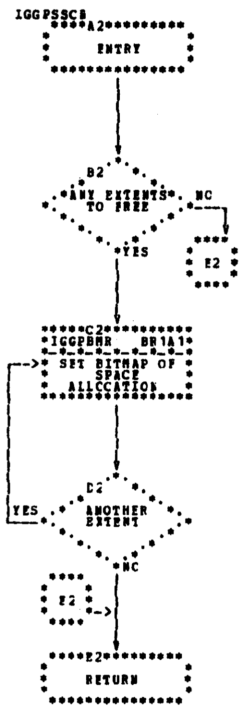


Chart BF1. Subscratch Routine (IGG0CLBF)

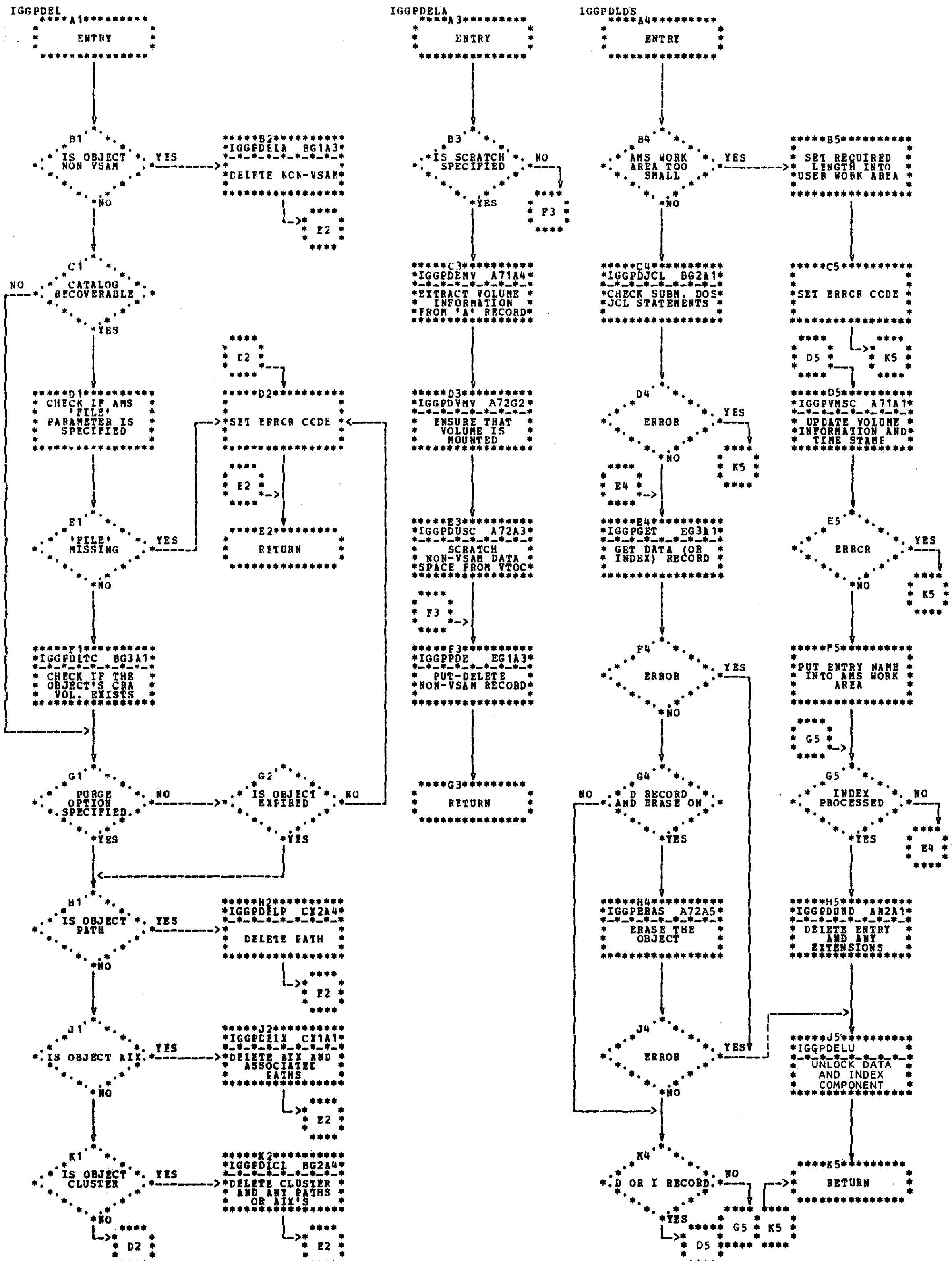


Chart BG1. CMS Delete - First Module (IGG0CLBG)

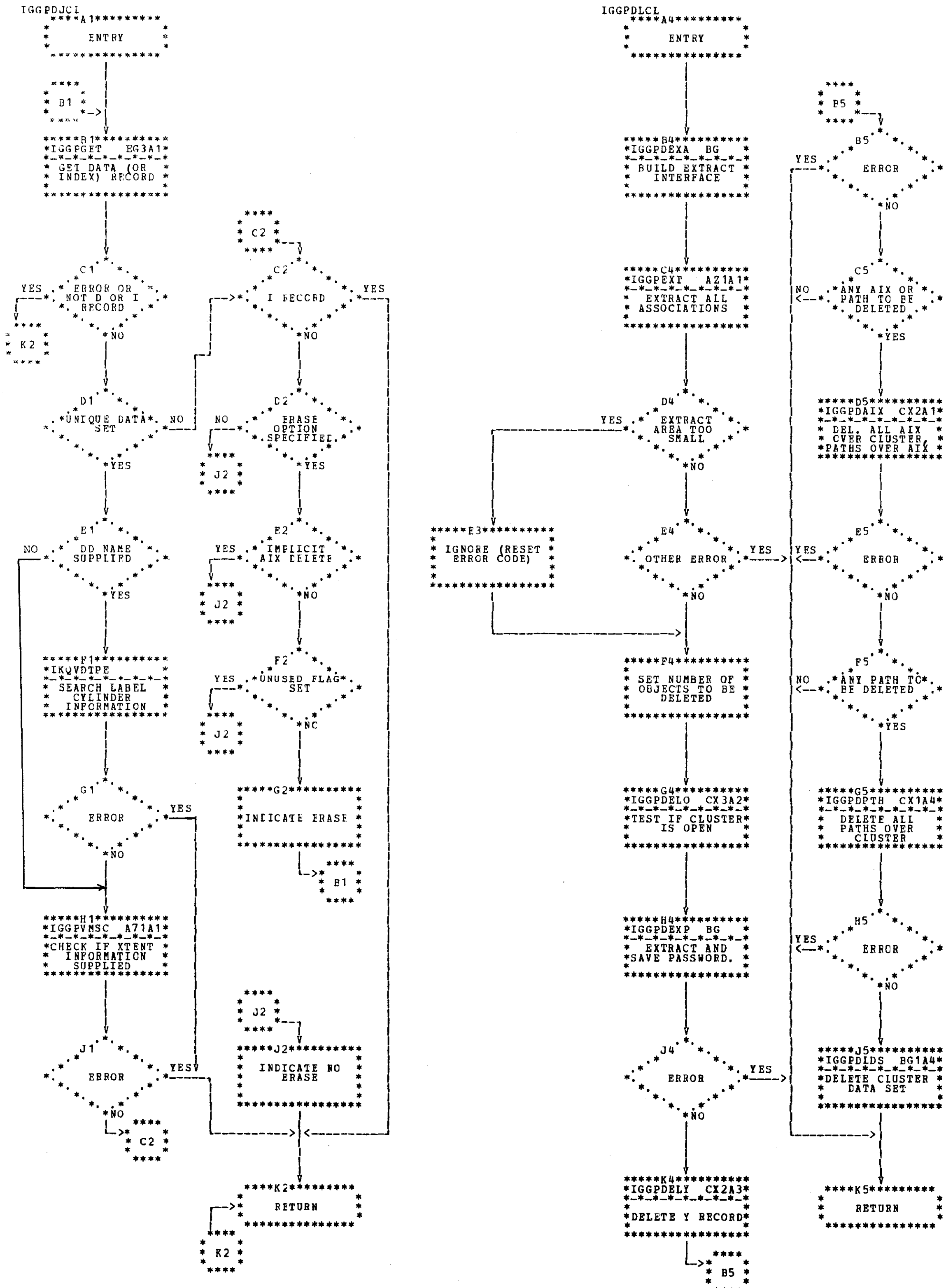


Chart BG2. CMS Delete - First Module (IGG0CLBG)

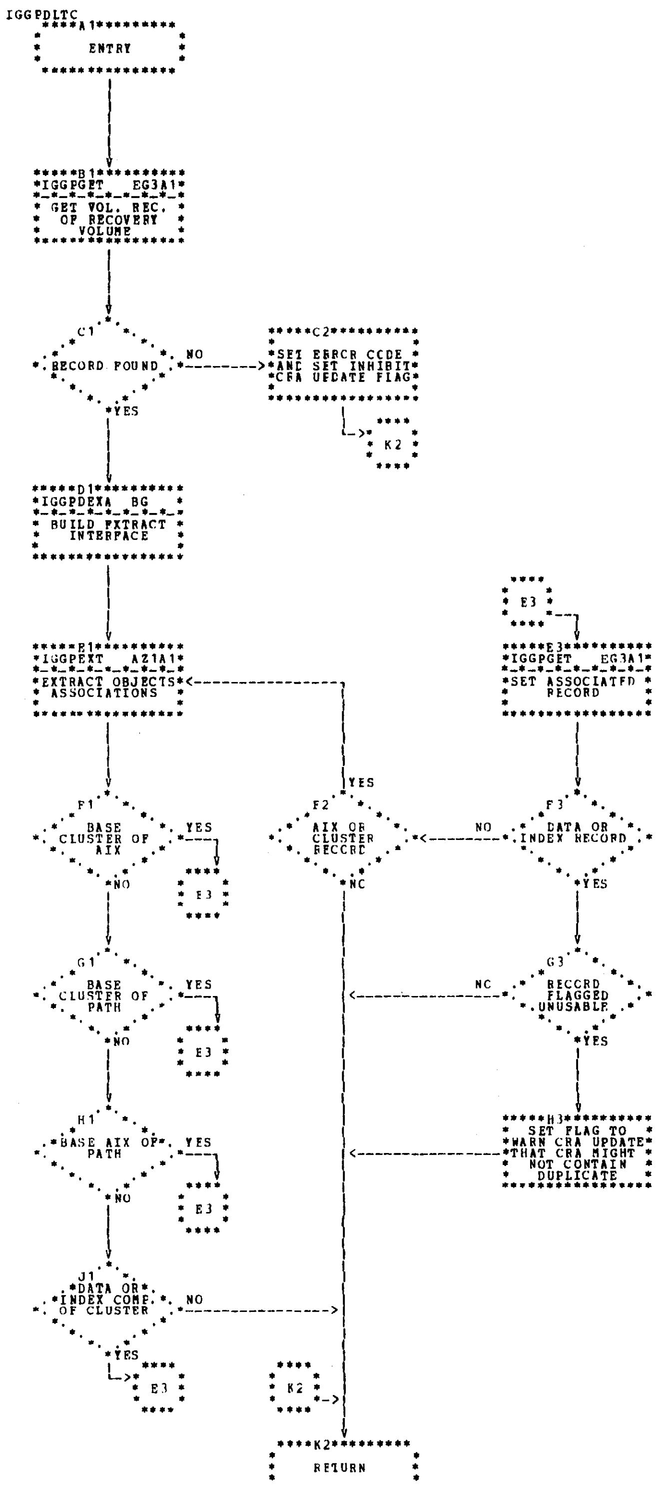


Chart BG3. CMS Delete - First Module (IGG0CLBG)

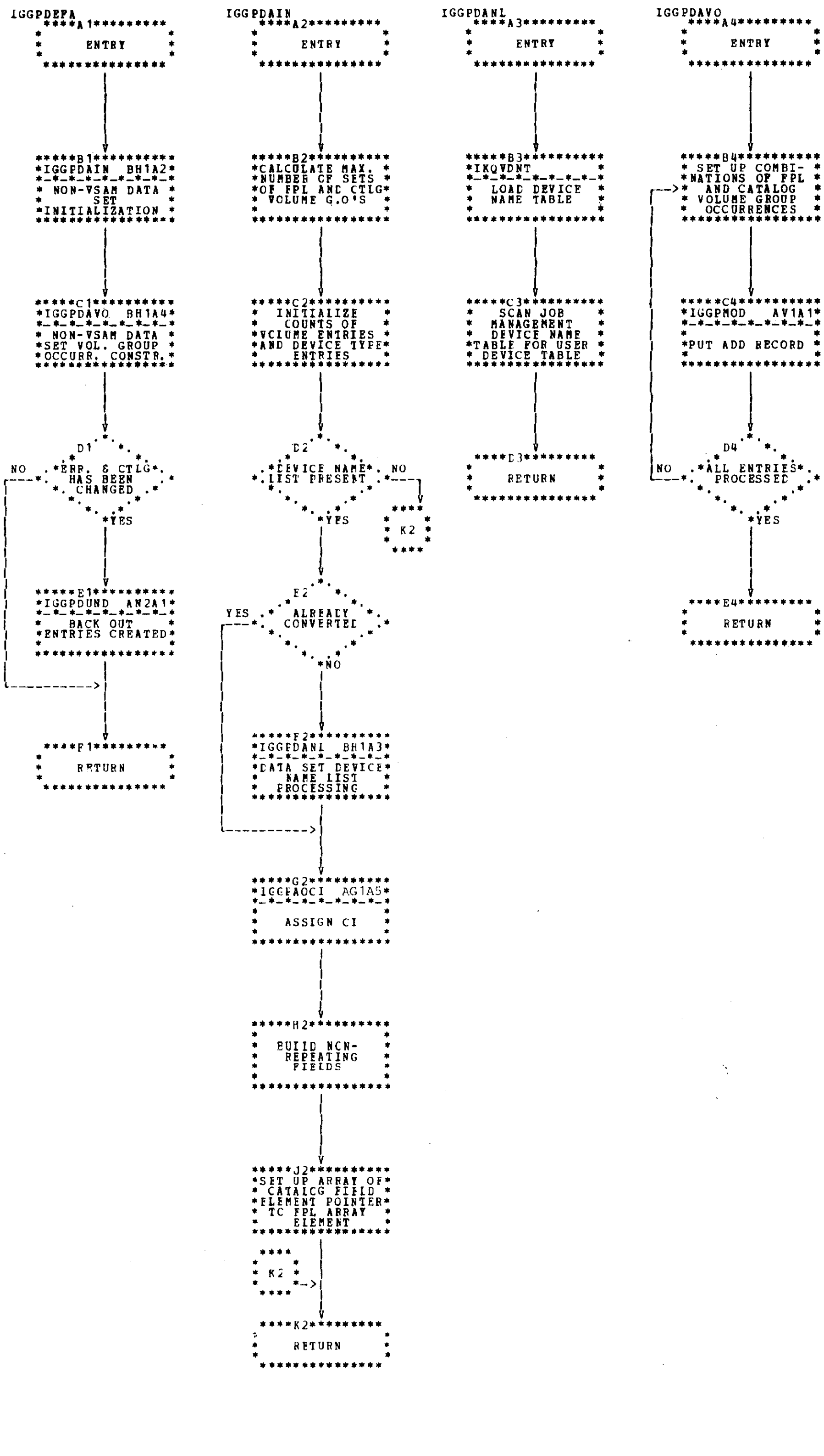


Chart BH1. Define NonVSAM Data Set (IGG0CLBH)

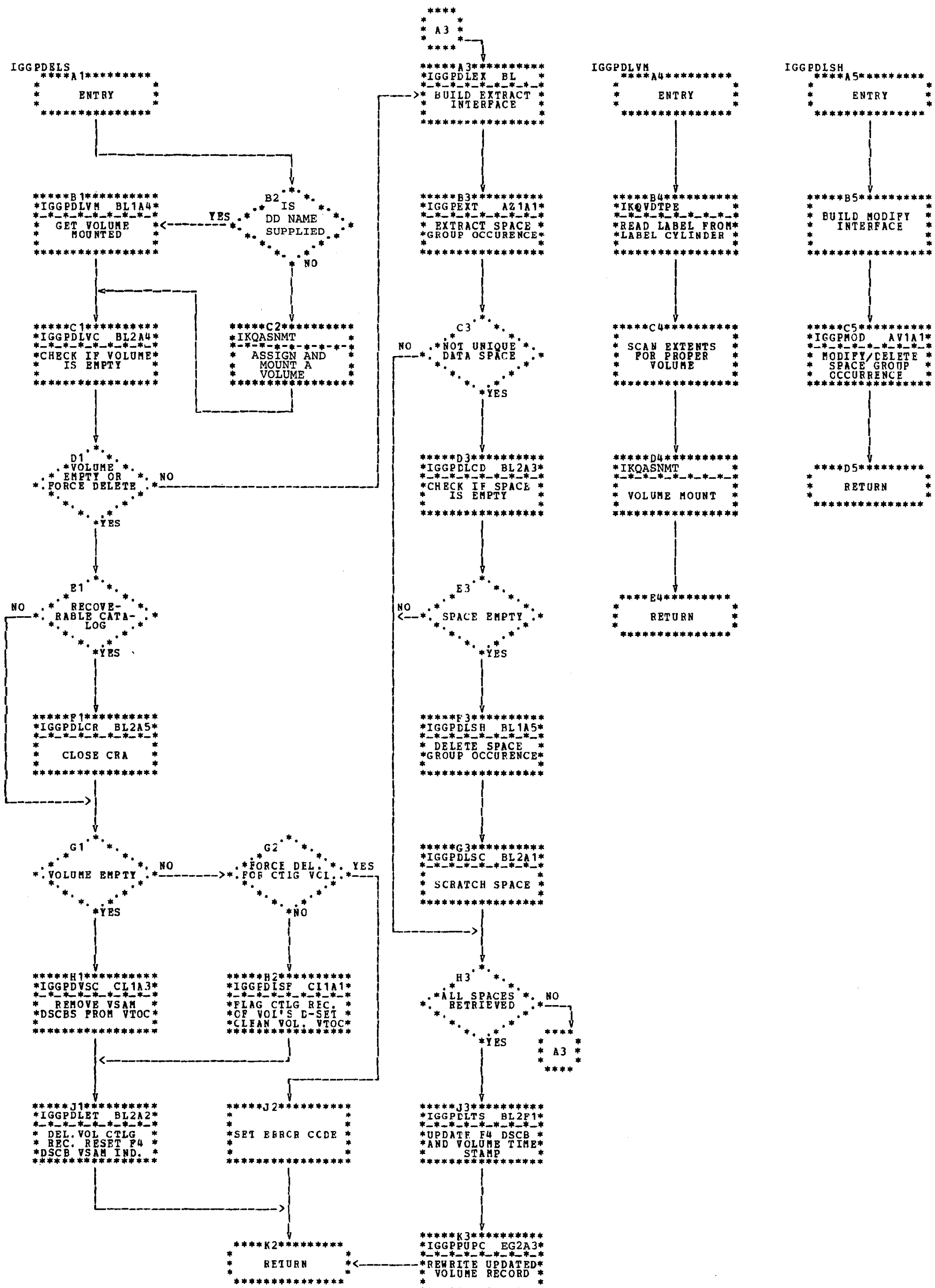


Chart BL1. CMS Delete Space - First Module (IGG0CLBL)

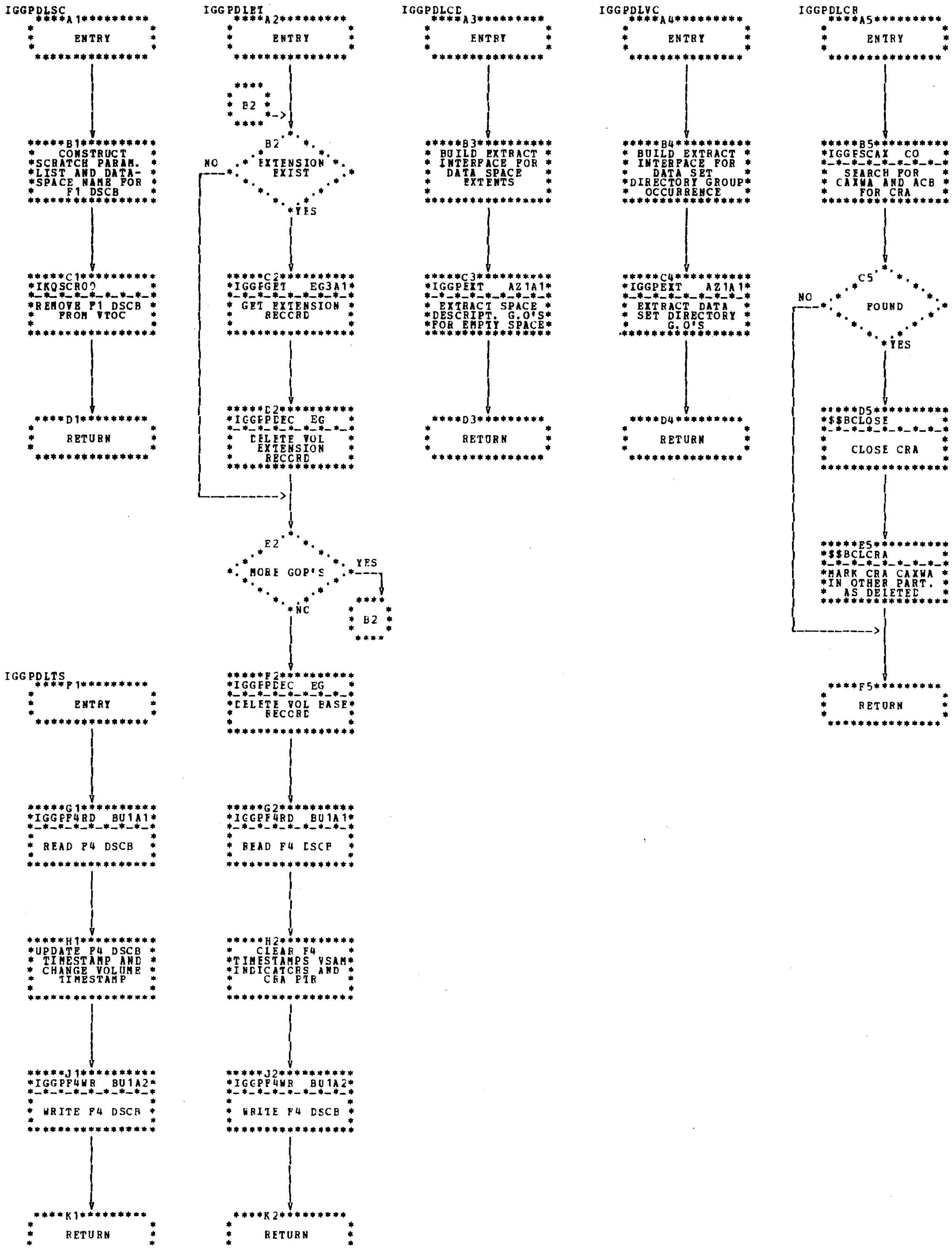


Chart BL2. CMS Delete Space - First Module (IGG0CLBL)

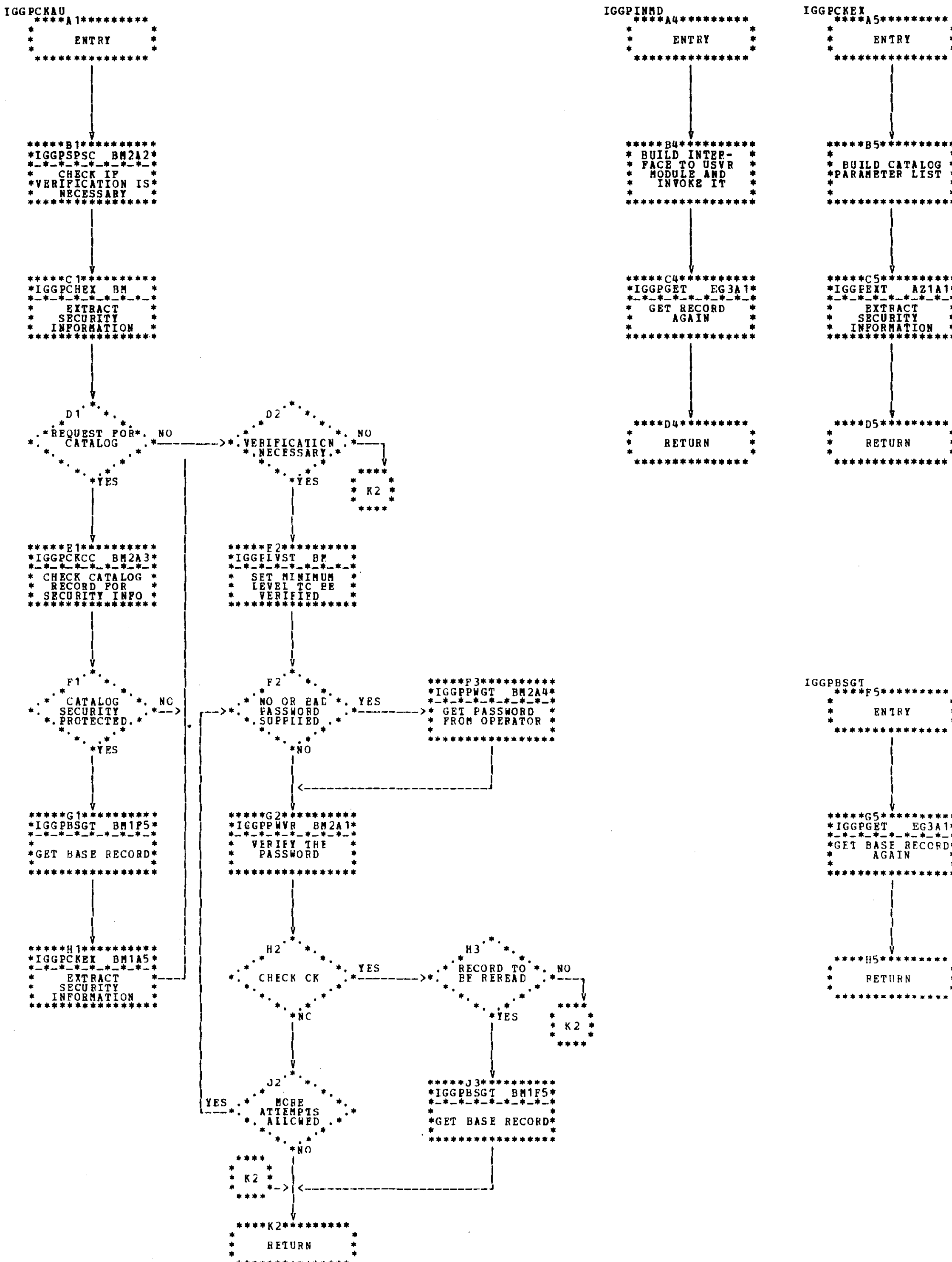


Chart BM1. Check Authorization (IGG0CLBM)

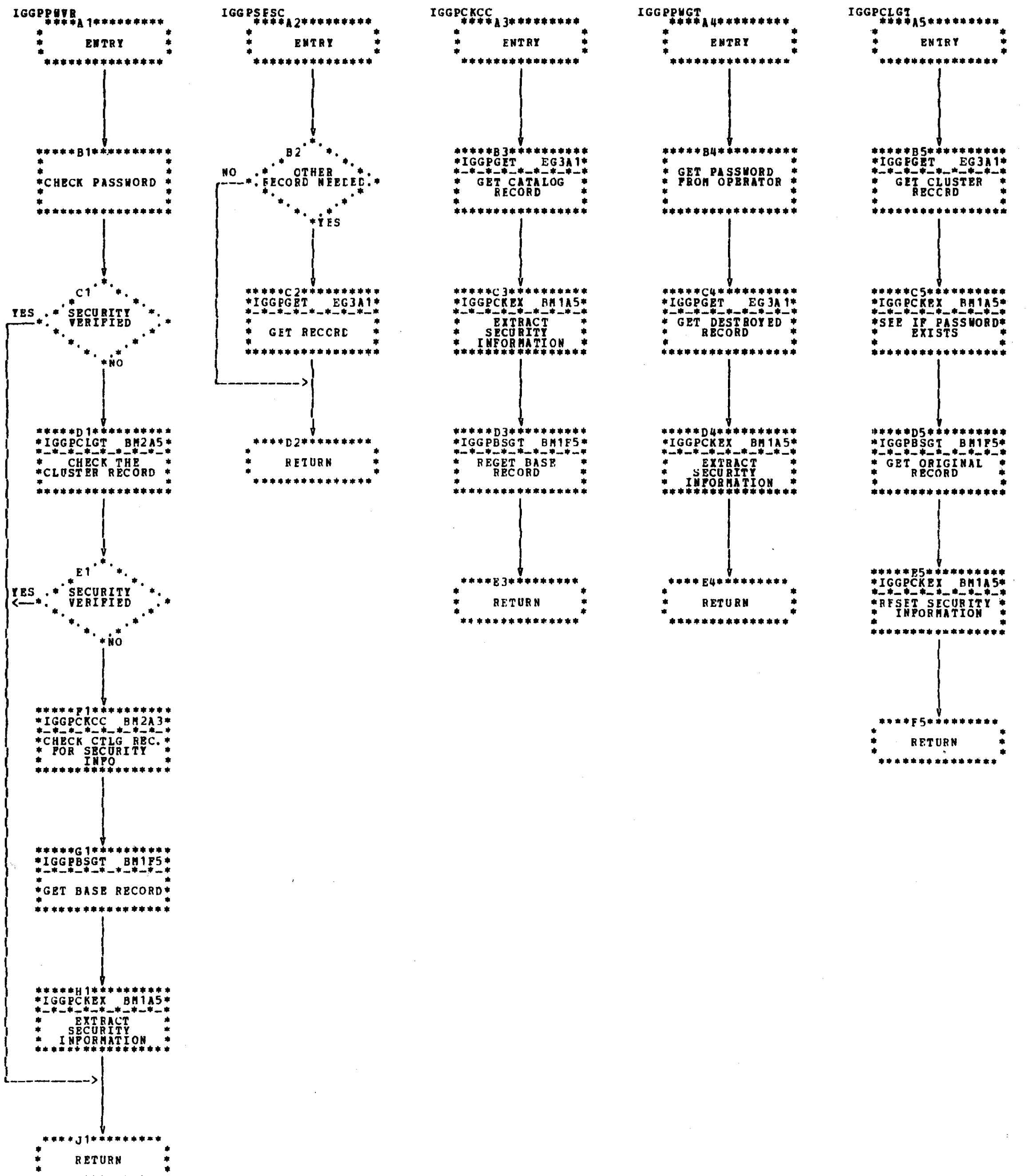


Chart BM2. Check Authorization (IGG0CLBM)

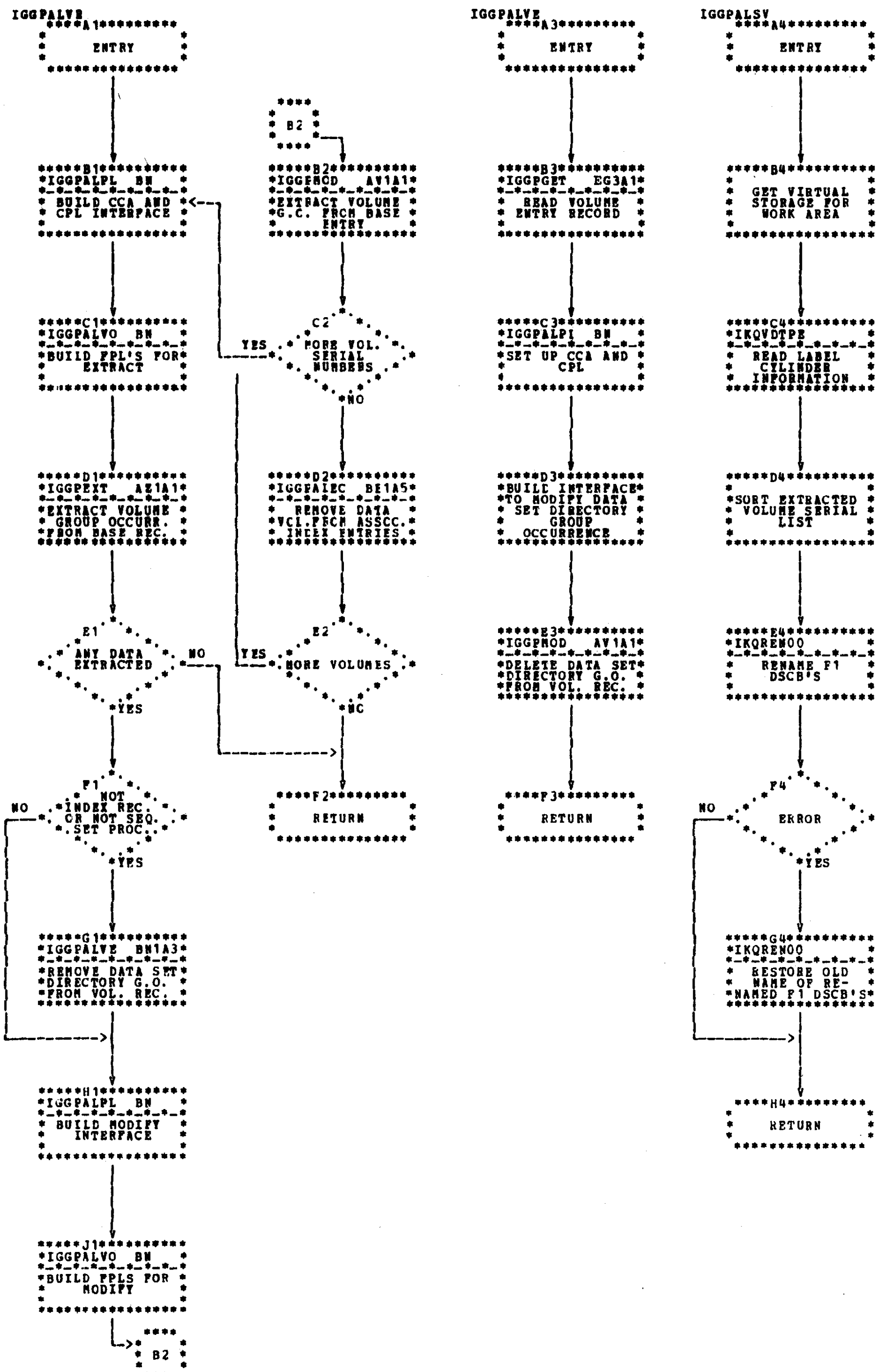


Chart BN1. CMS Alter - Second Module (IGG0CLBN)

```
IGGPLSTC
   ****A2*********
  *               *
  *     ENTRY     *
  *               *
   ***************
          |
          |
          |
          |
          V
         .*.
    B2  .   *.
      .* NEXT  *.
    .* RECORD OF *.  YES
  *. MULTIPLE LIST. *---------->*
    *. NEEDED    .*
      *.       .*
        *.   .*
          *NO
          |
          |
          |<-----------------------
          |
          V
  *****C2**********
  *IGGPCKAU   BM1A1*
  *-*-*-*-*-*-*-*-*
  *    SECURITY    *
  *  VERIFICATION  *
  *                *
  *****************
          |
          |
          |
          |
          V
  *****D2**********
  *IGGPEXT    AZ1A1*
  *-*-*-*-*-*-*-*-*
  * GET DATA REC.  *
  * FROM APPROPR.  *
  *     FIELDS     *
  *****************
          |
          |
          |
          |
          V
   ****E2*********
  *               *
  *    RETURN     *
  *               *
   ***************
```

```
  *****B3**********
  *IGGPGET    EG3A1*
  *-*-*-*-*-*-*-*-*
  *                *
  *GET NEXT RECORD*
  *                *
  *****************
```

Chart BQ1. List Catalog (IGG0CLBQ)

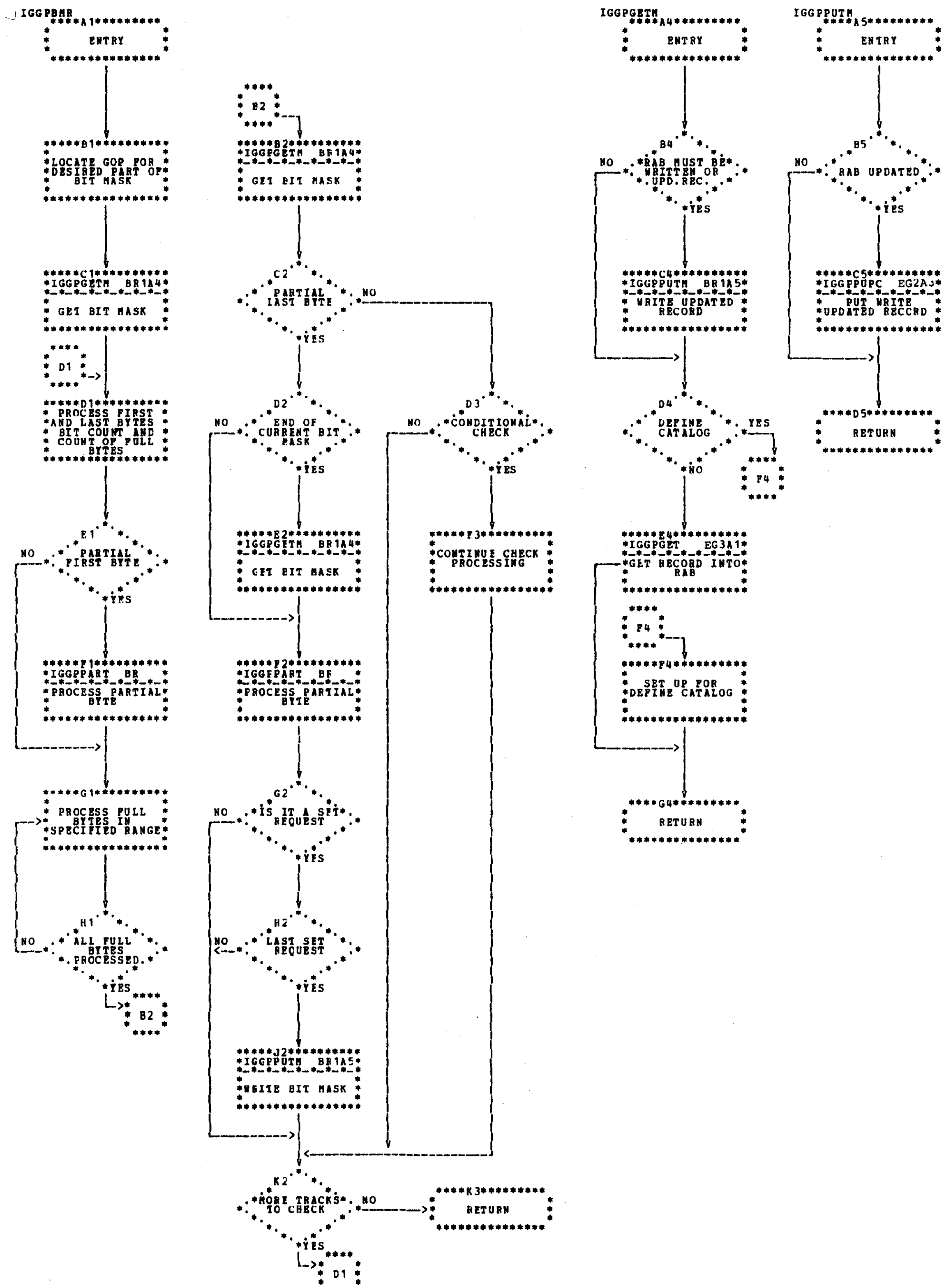


Chart BR1. Suballocate Bit Map Handler (IGG0CLBR)

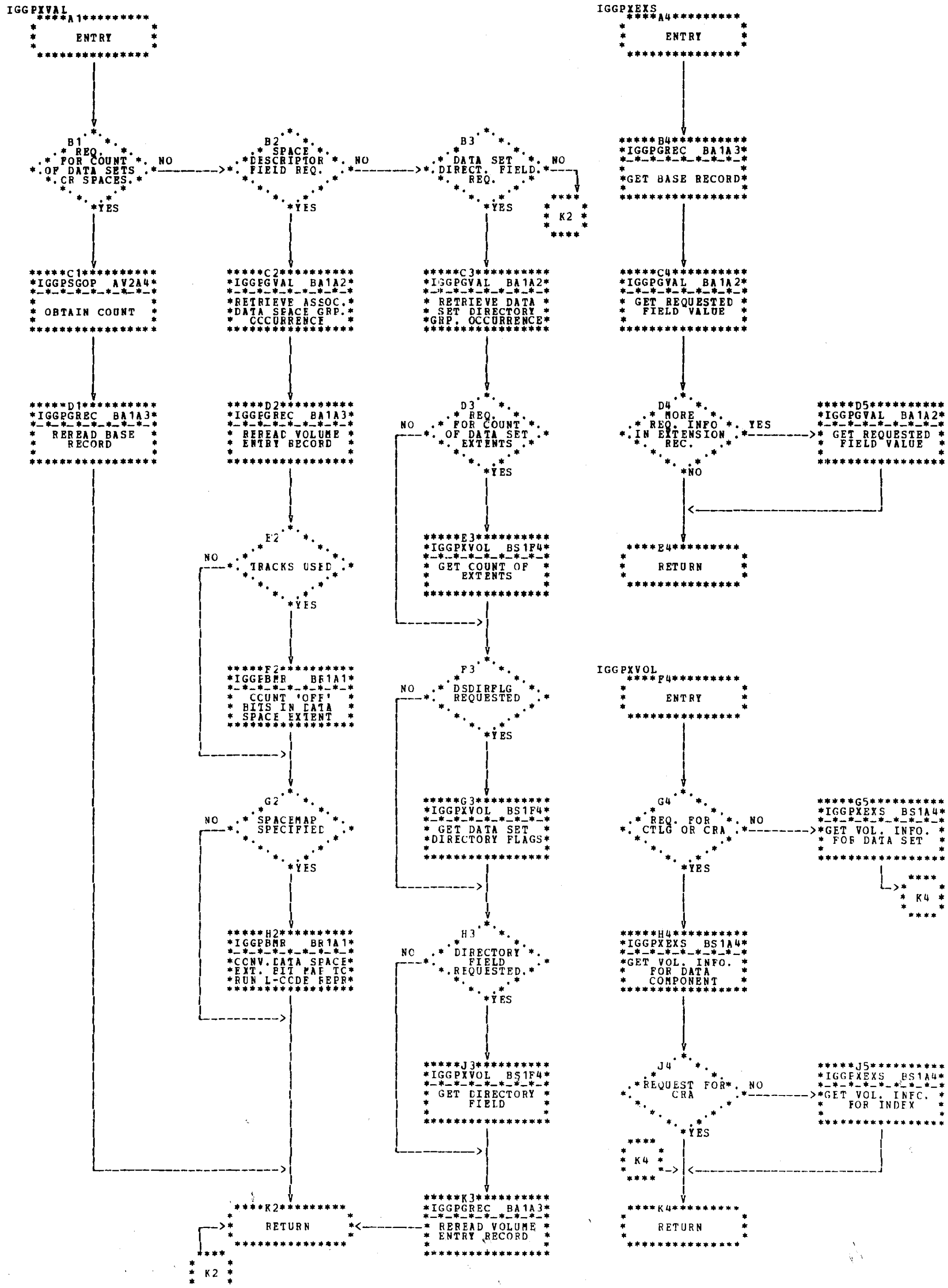


Chart BS1. Volume Entry Translation (IGG0CLBS)

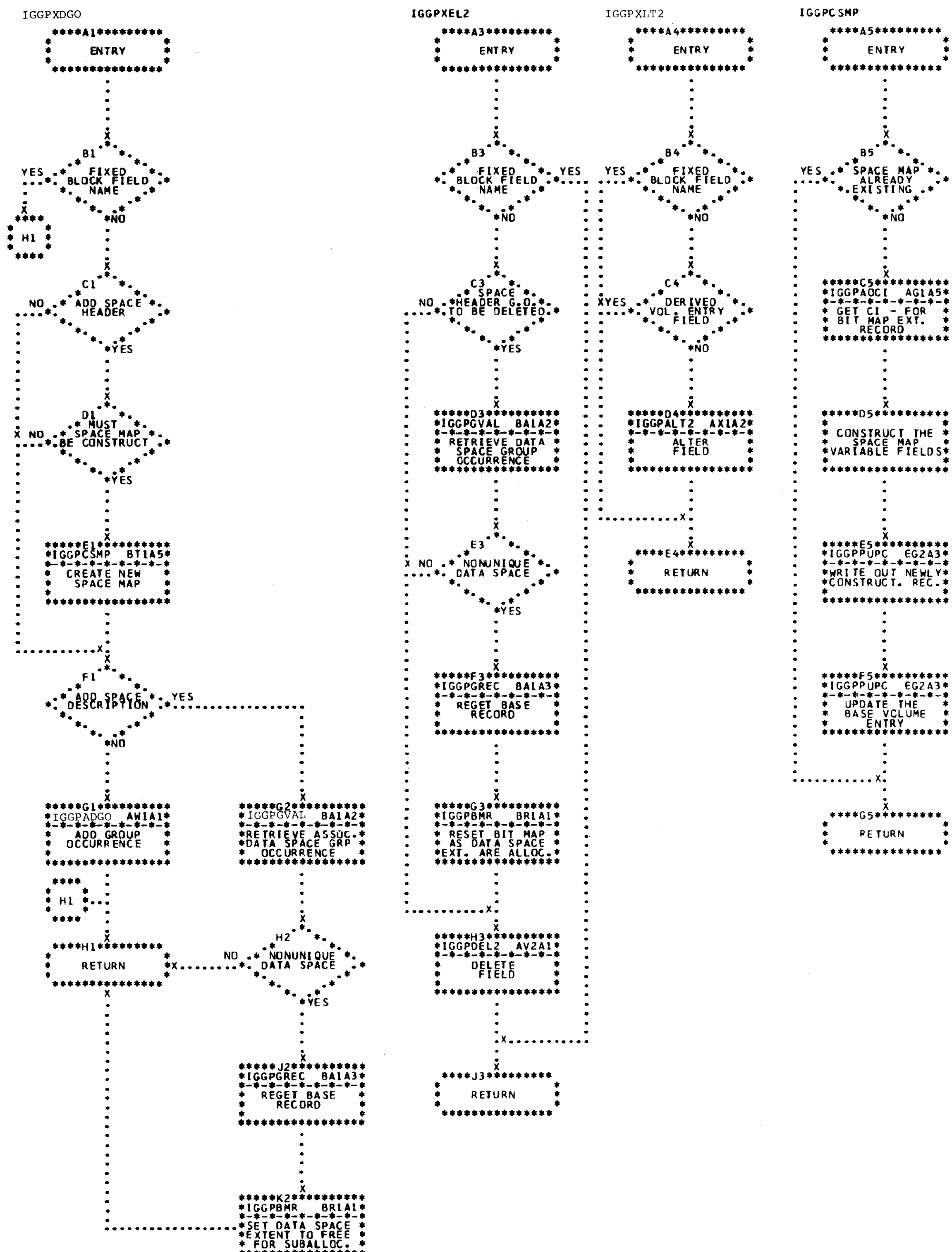


Chart BT1. Modify Volume Entry Translation (IGG0CLBT)

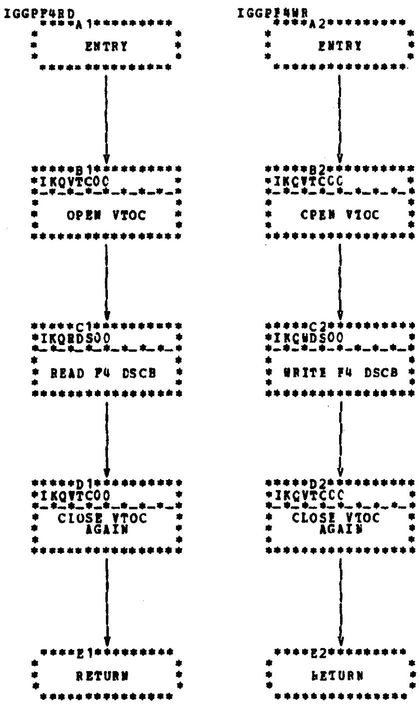


Chart BU1. Catalog Read/Write Format-4 VTOC Label (IGG0CLBU)

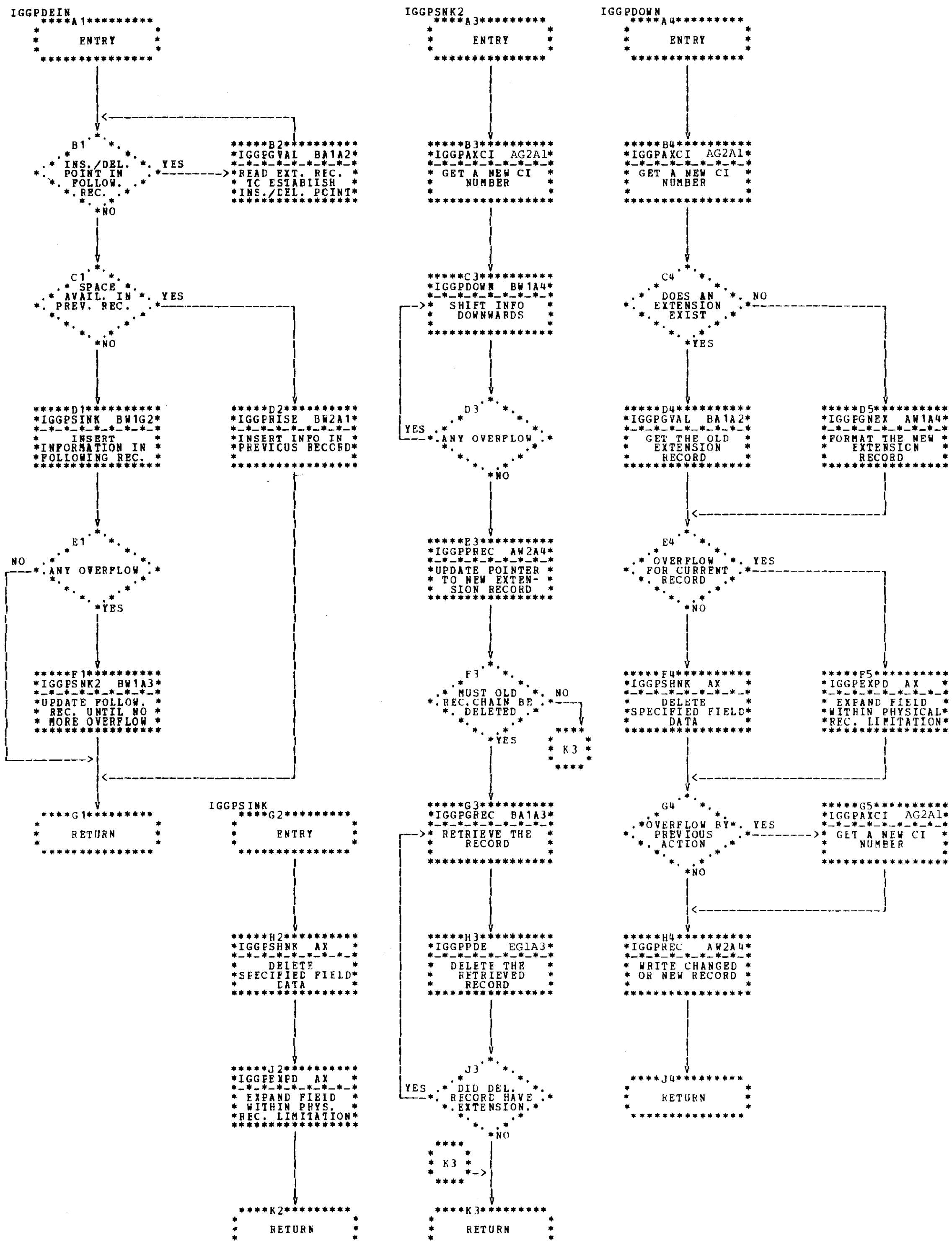


Chart BW1. Delete/Insert Subfunction (IGG0CLBW)

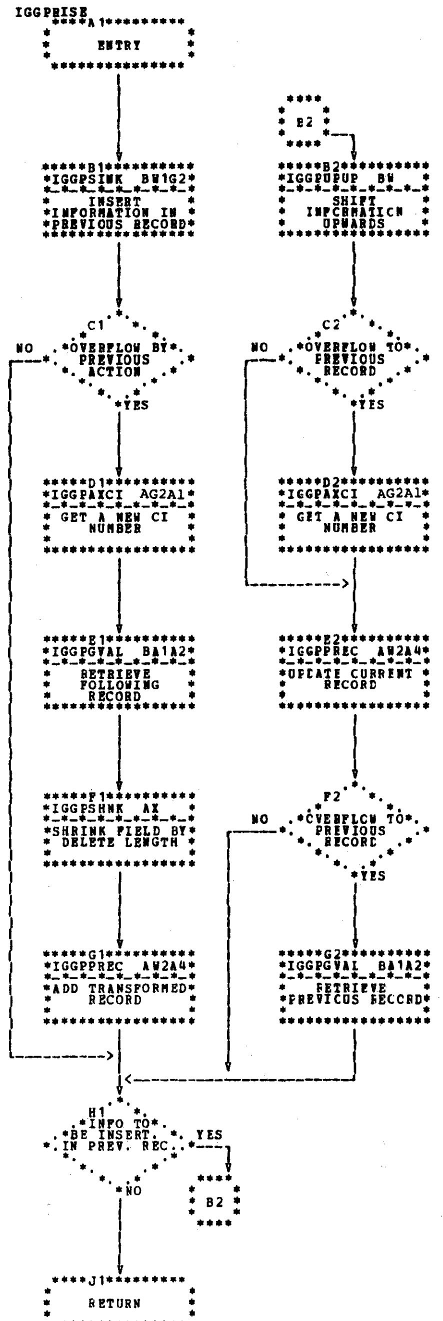


Chart BW2. Delete/Insert Subfunction (IGG0CLBW)

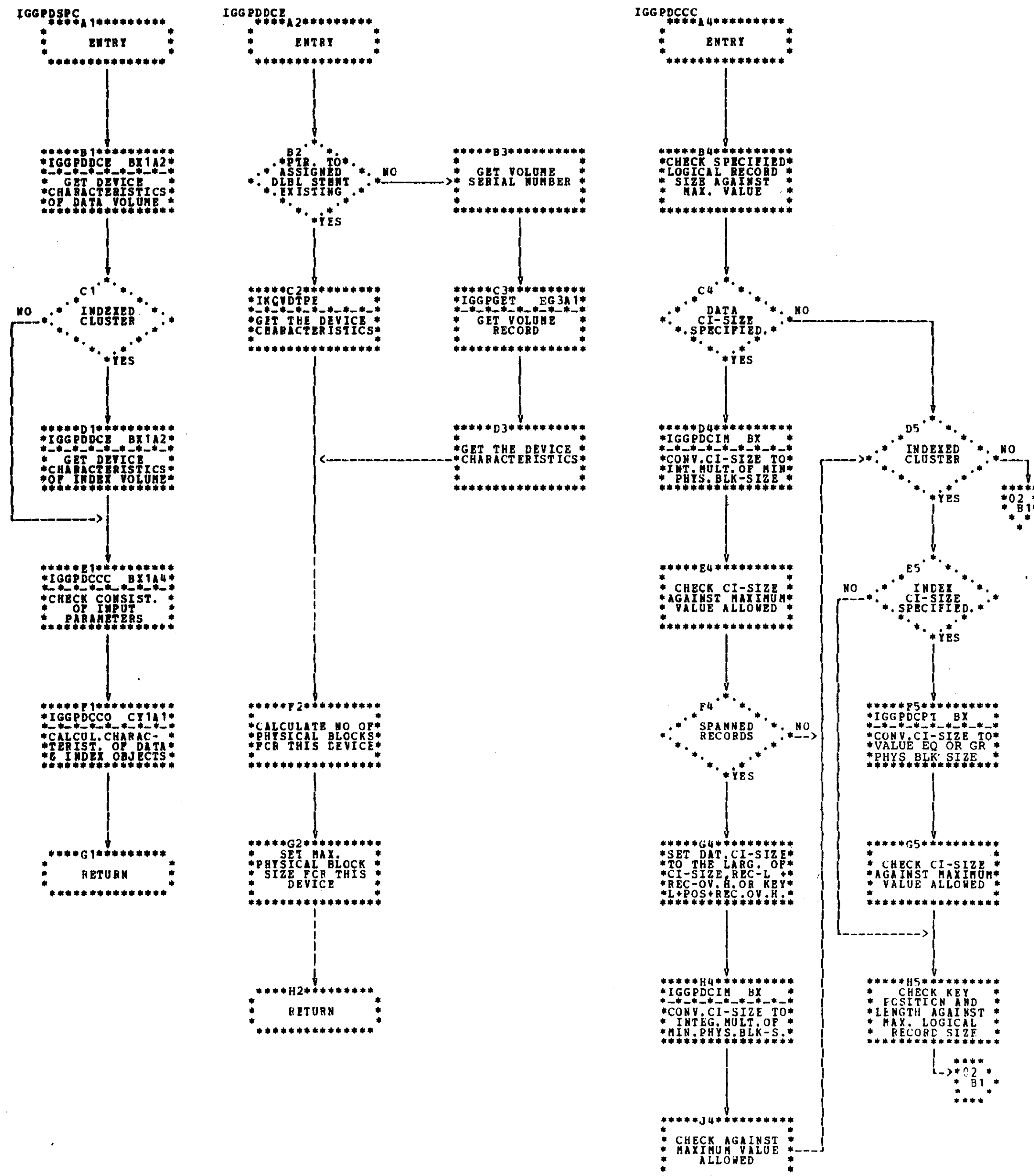


Chart BX1. CMS Define - Fourth Module (IGG0CLBX)

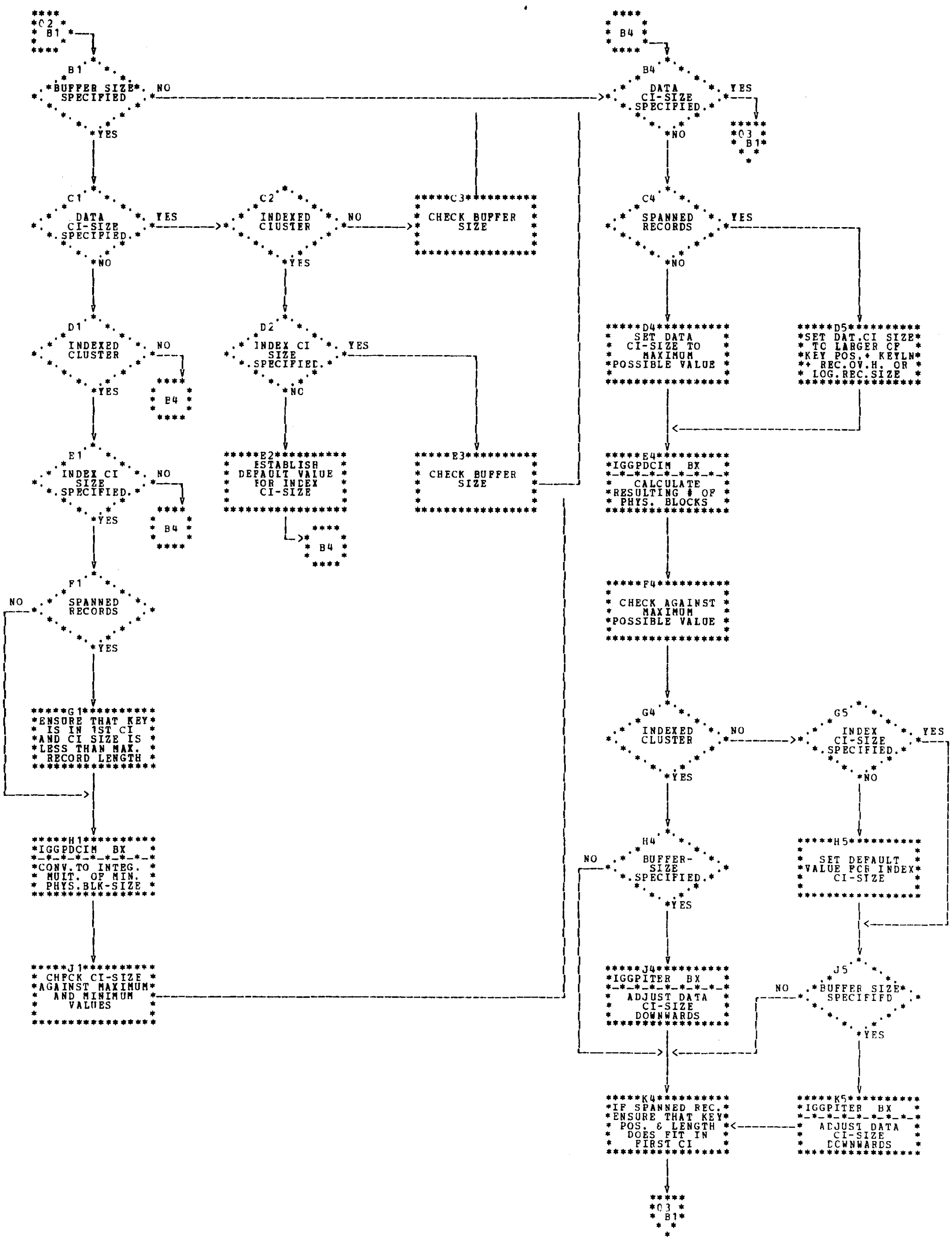


Chart BX2. CMS Define - Fourth Module (IGG0CLBX)

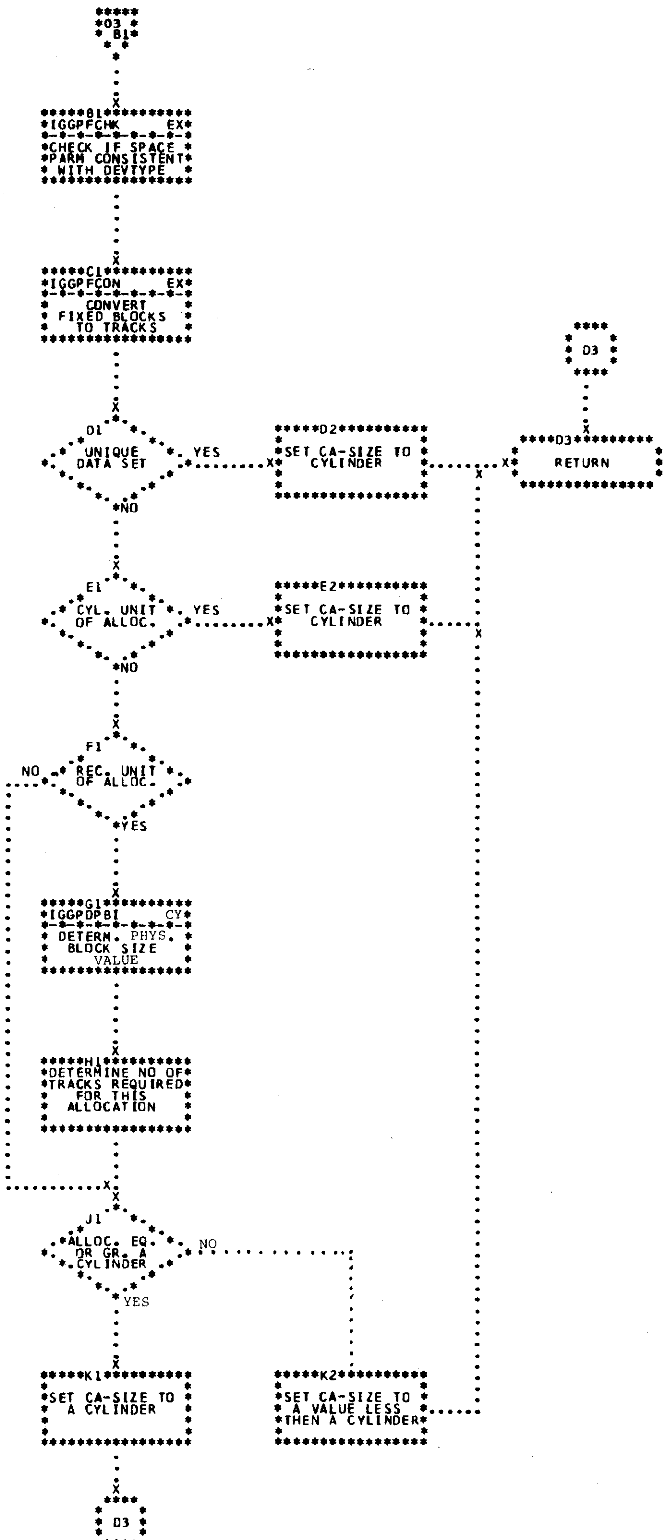


Chart BX3. CMS Define - Fourth Module (IGG0CLBX)

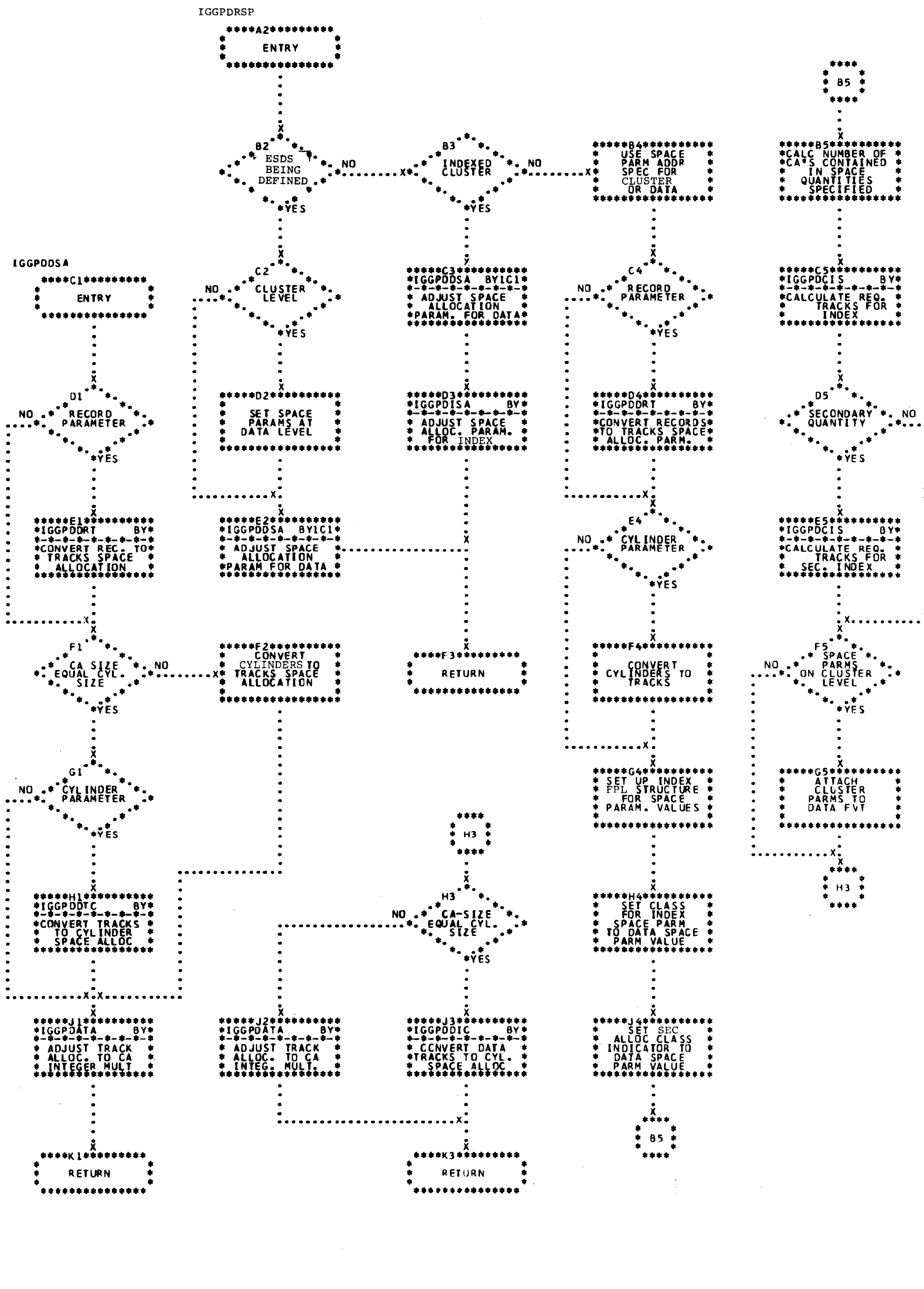


Chart BY1. CMS Define - Fifth Module (IGG0CLBY)

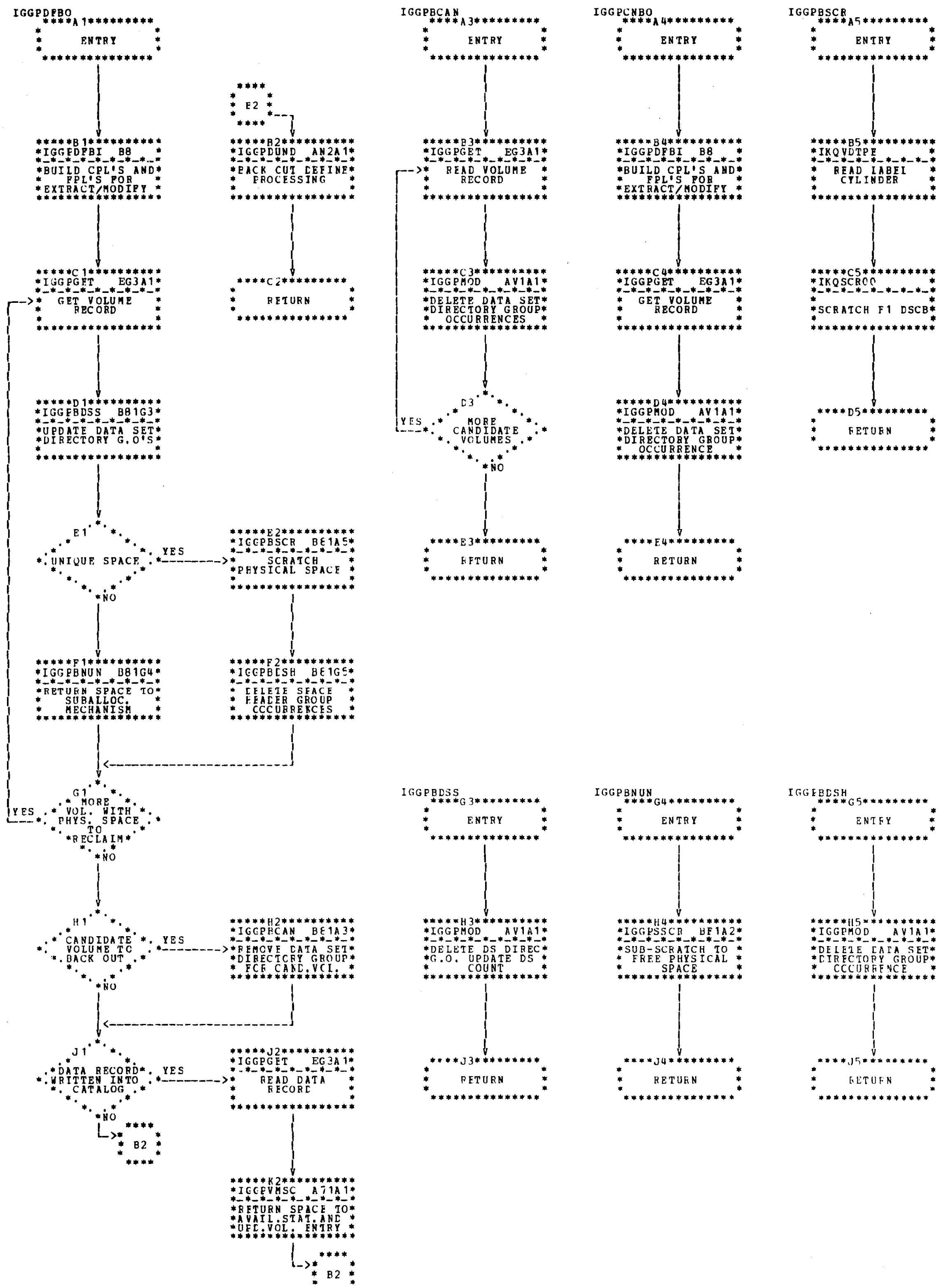


Chart B81. Define Space Recovery (IGG0CLB8)

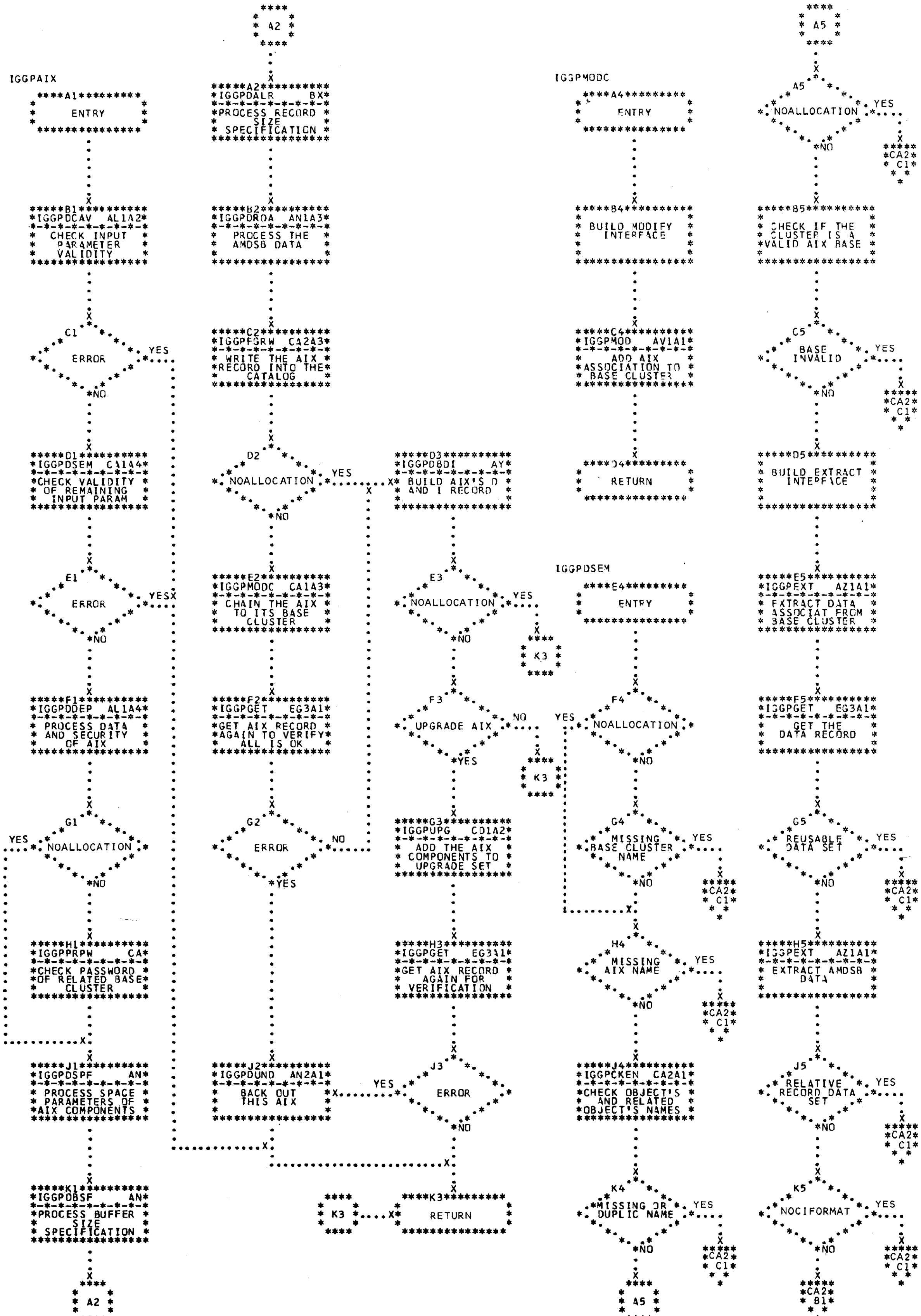


Chart CA1. Define Alternate Index (IGG0CLCA)

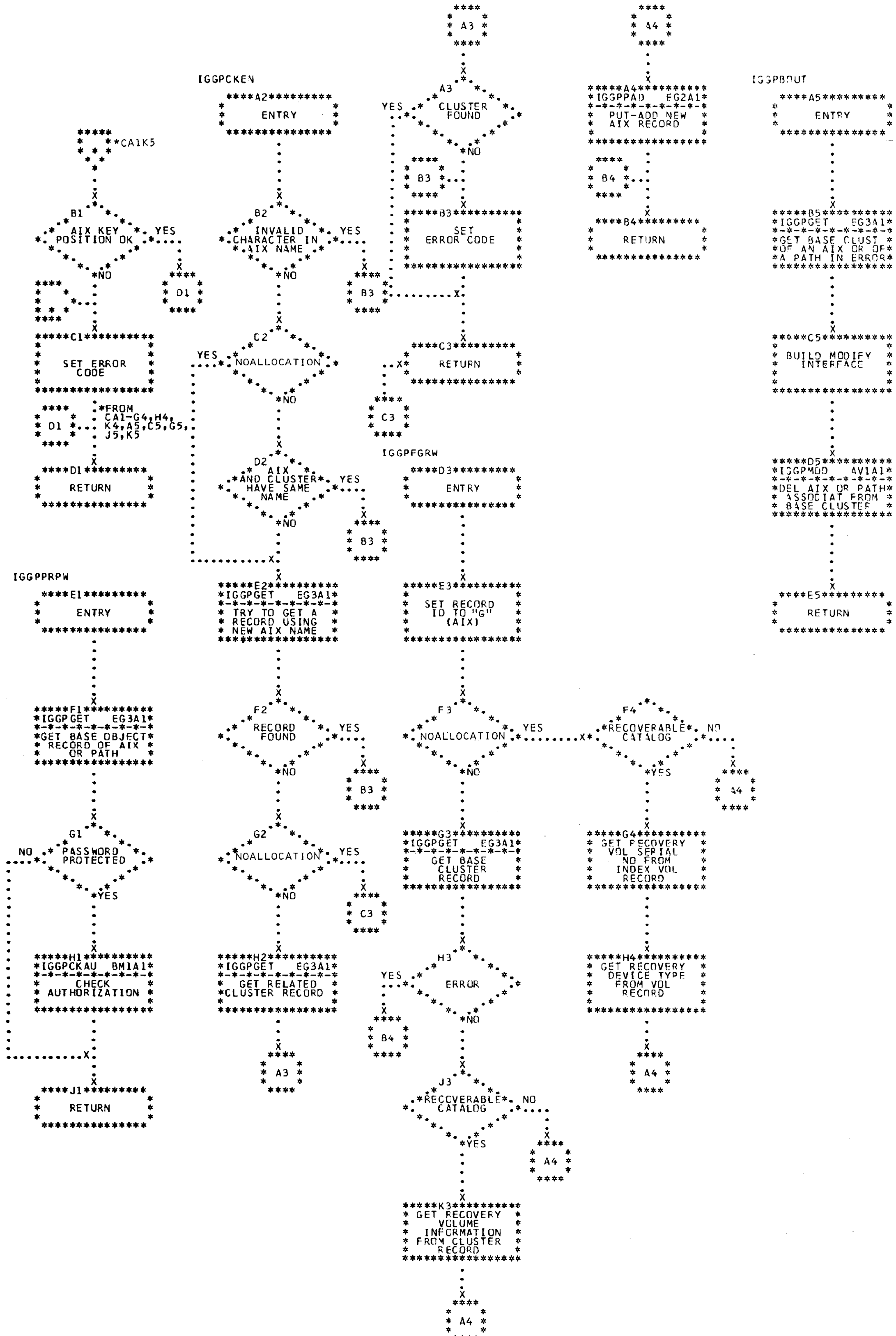


Chart CA2. Define Alternate Index (IGG0CLCA)

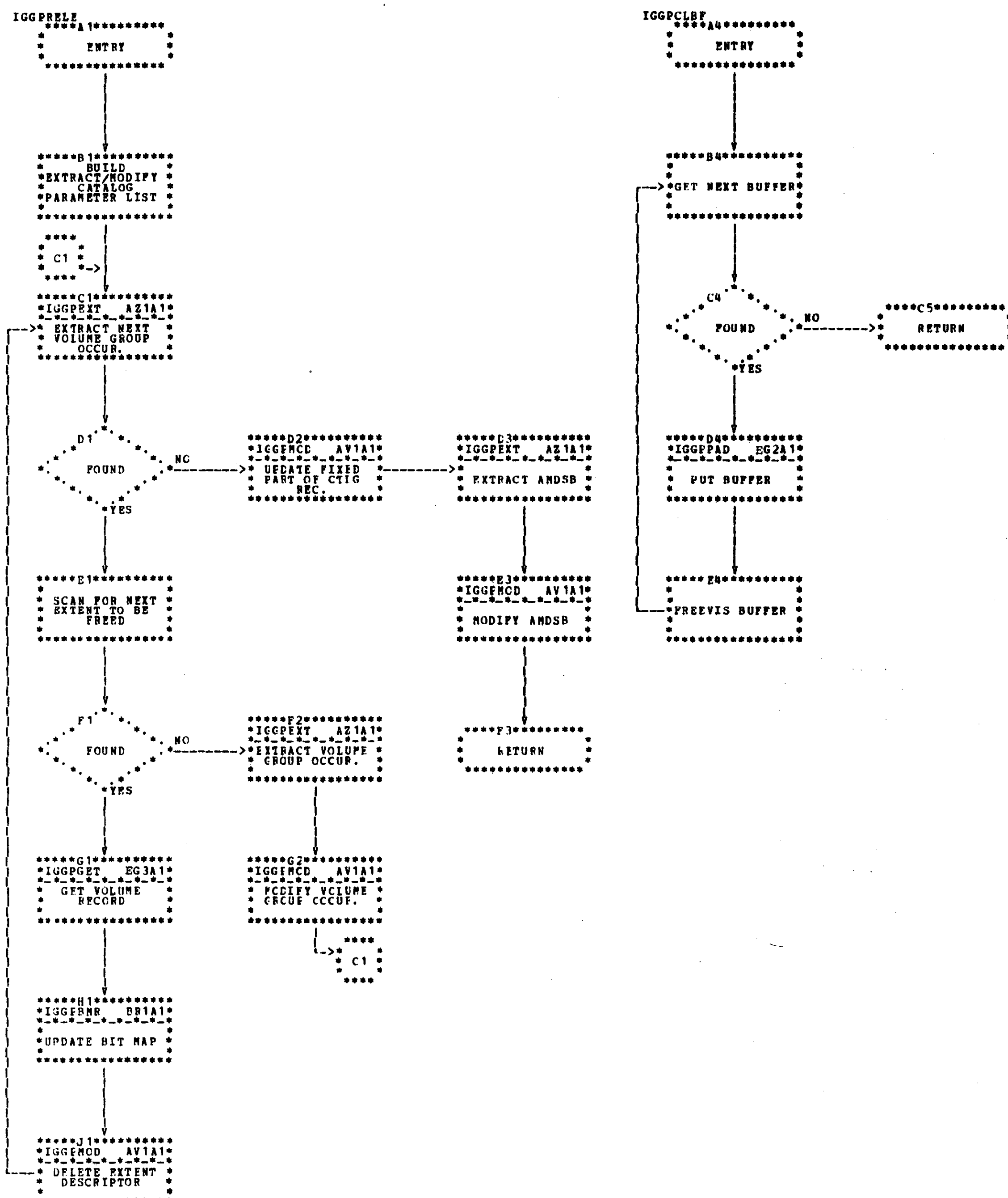


Chart CB1. Release Function (IGG0CLCB)

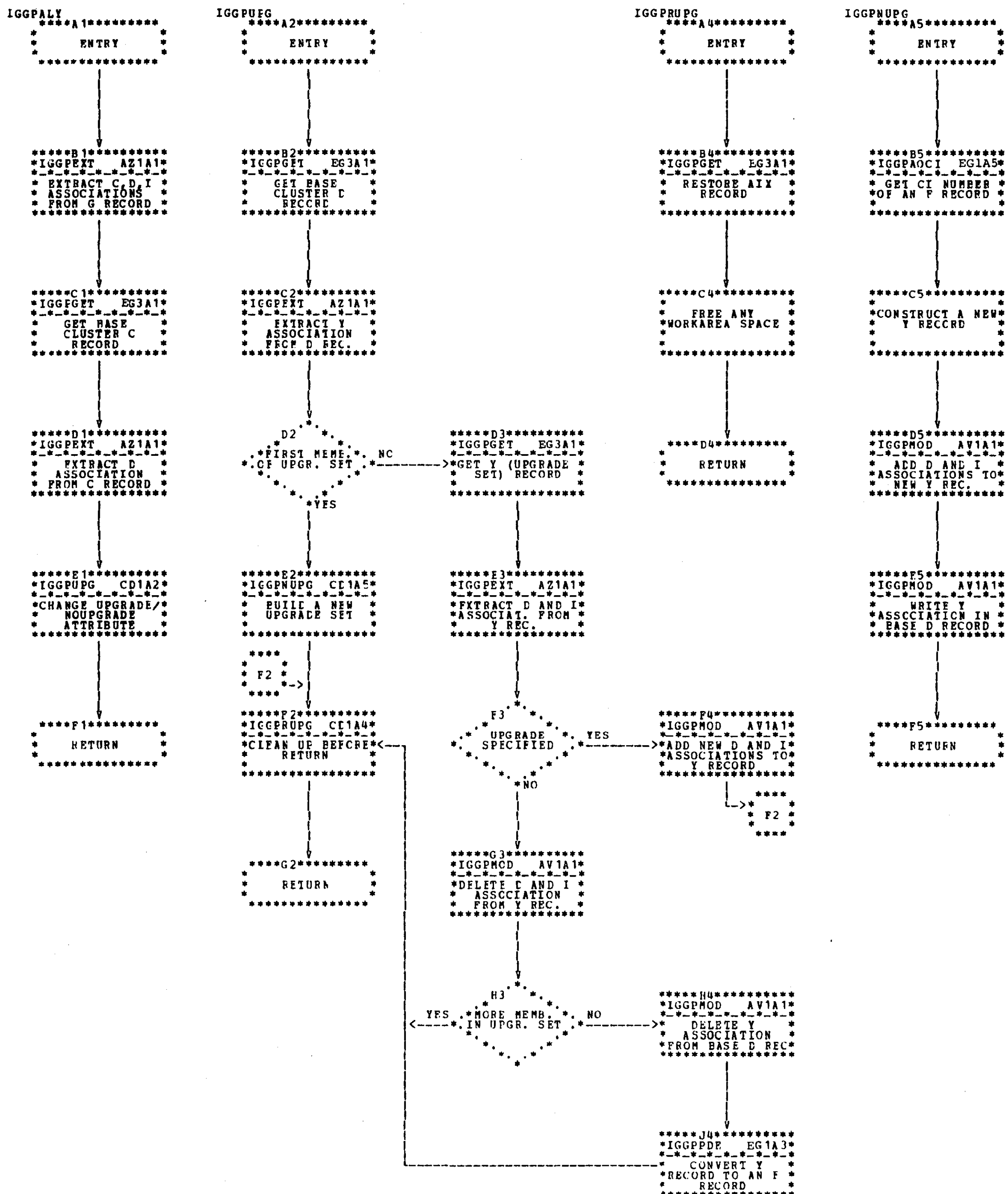


Chart CD1. CMS Alter - Fourth Module (IGG0CLCD)

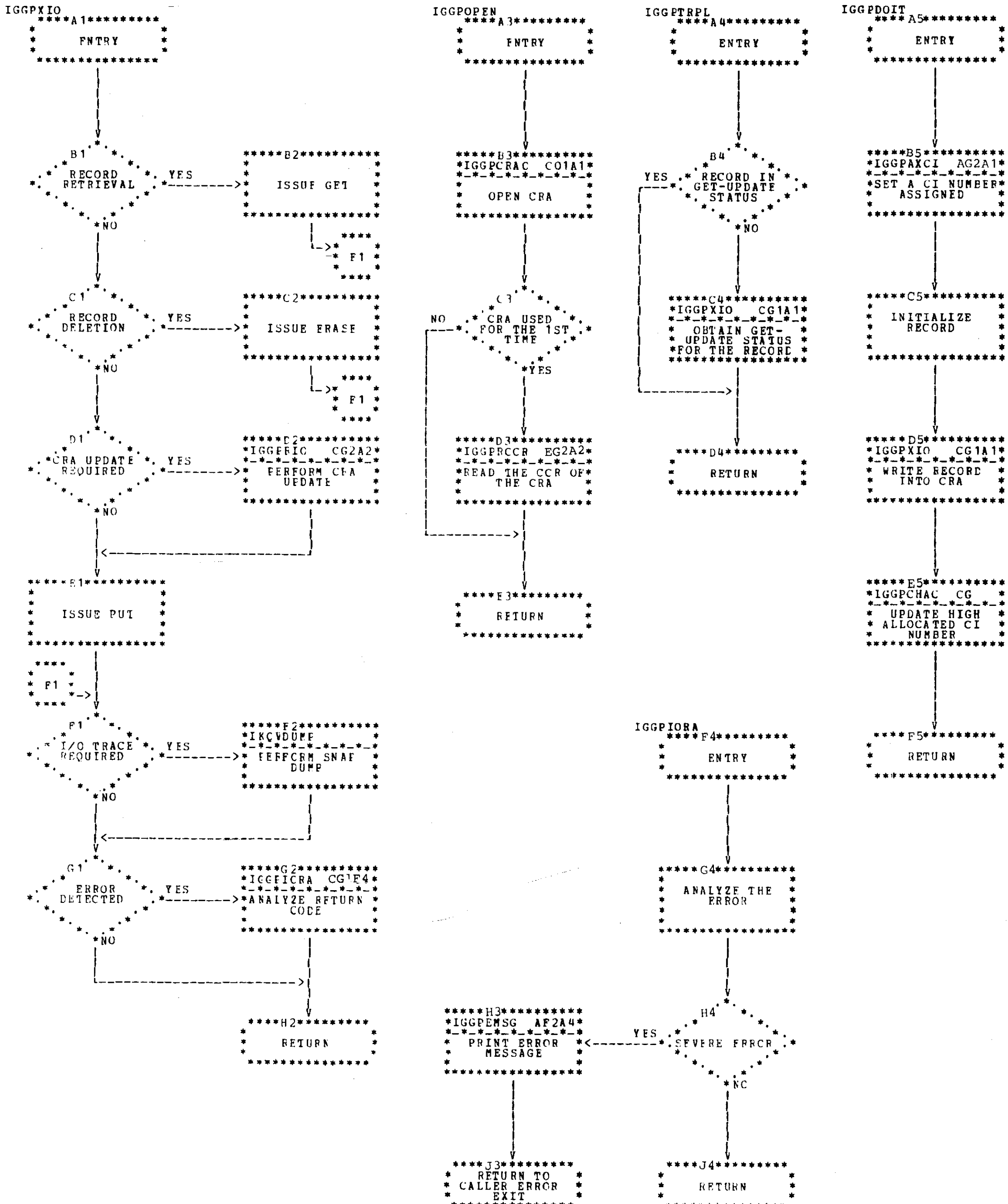


Chart CG1. Catalog I/O Subroutine - Second Module (IGG0CLCG)

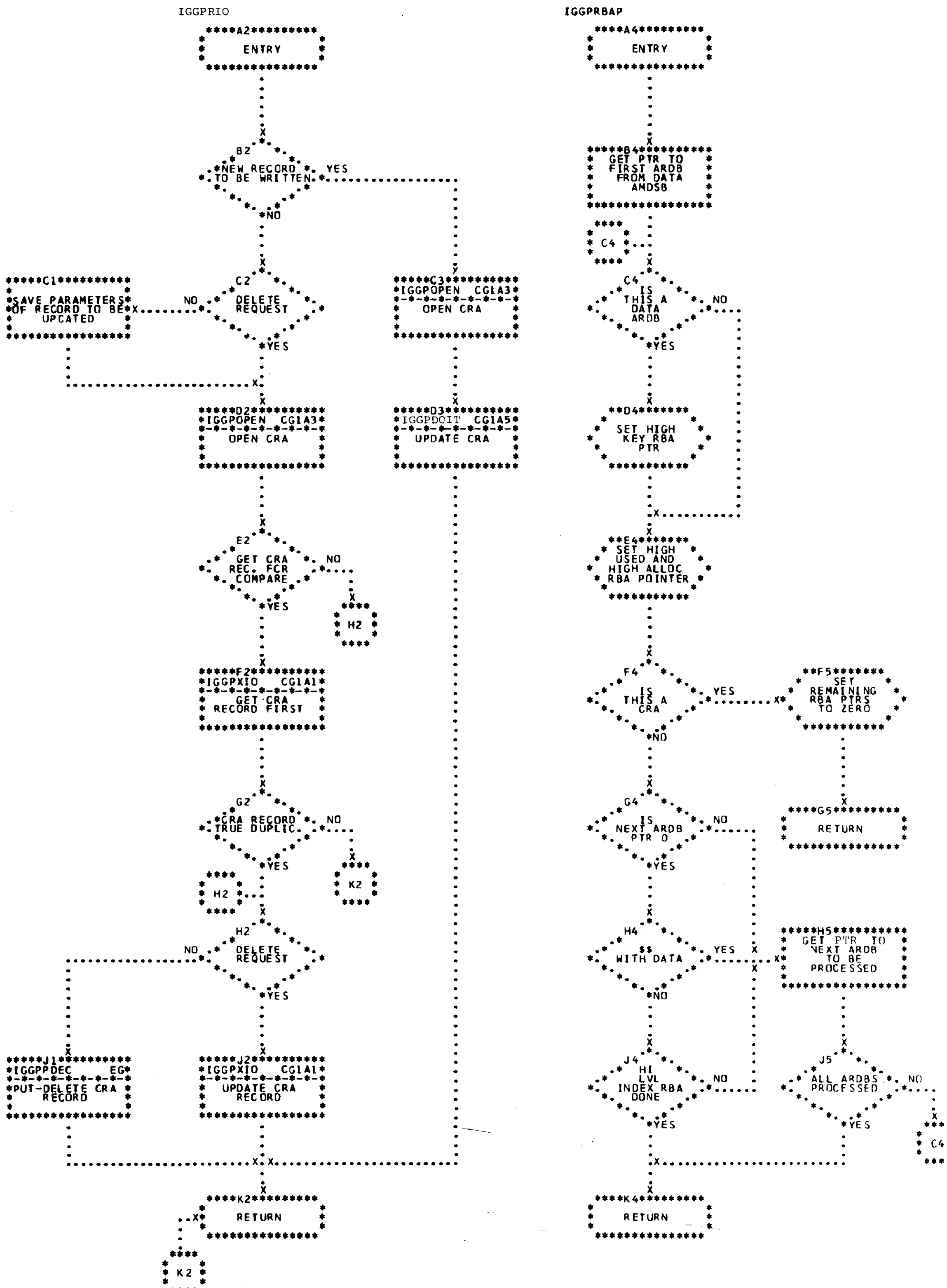


Chart CG2. Catalog I/O Subroutine Second Module (IGG0CLCG)

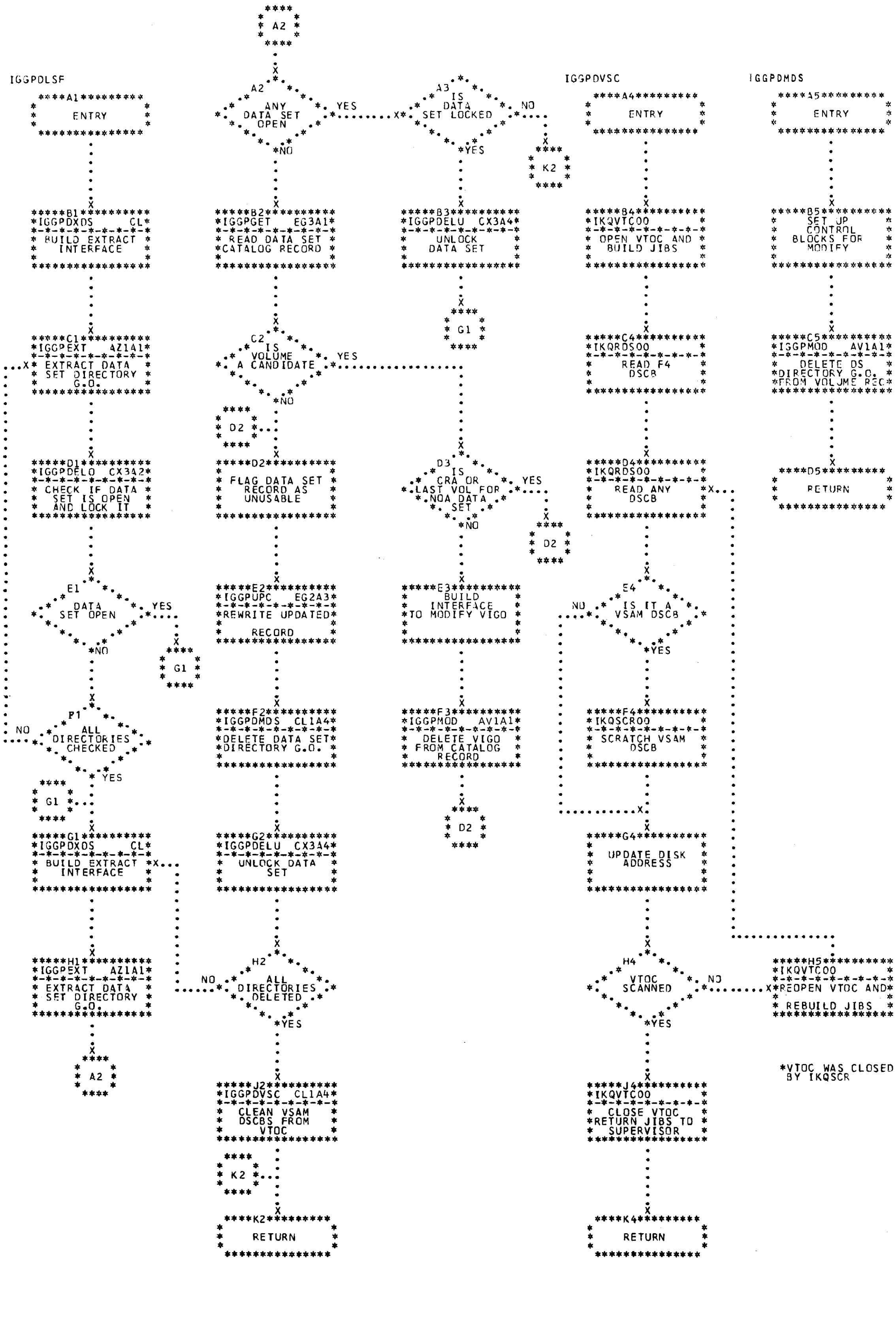


Chart CL1. CMS Delete Space - Second Module (IGG0CLCL)

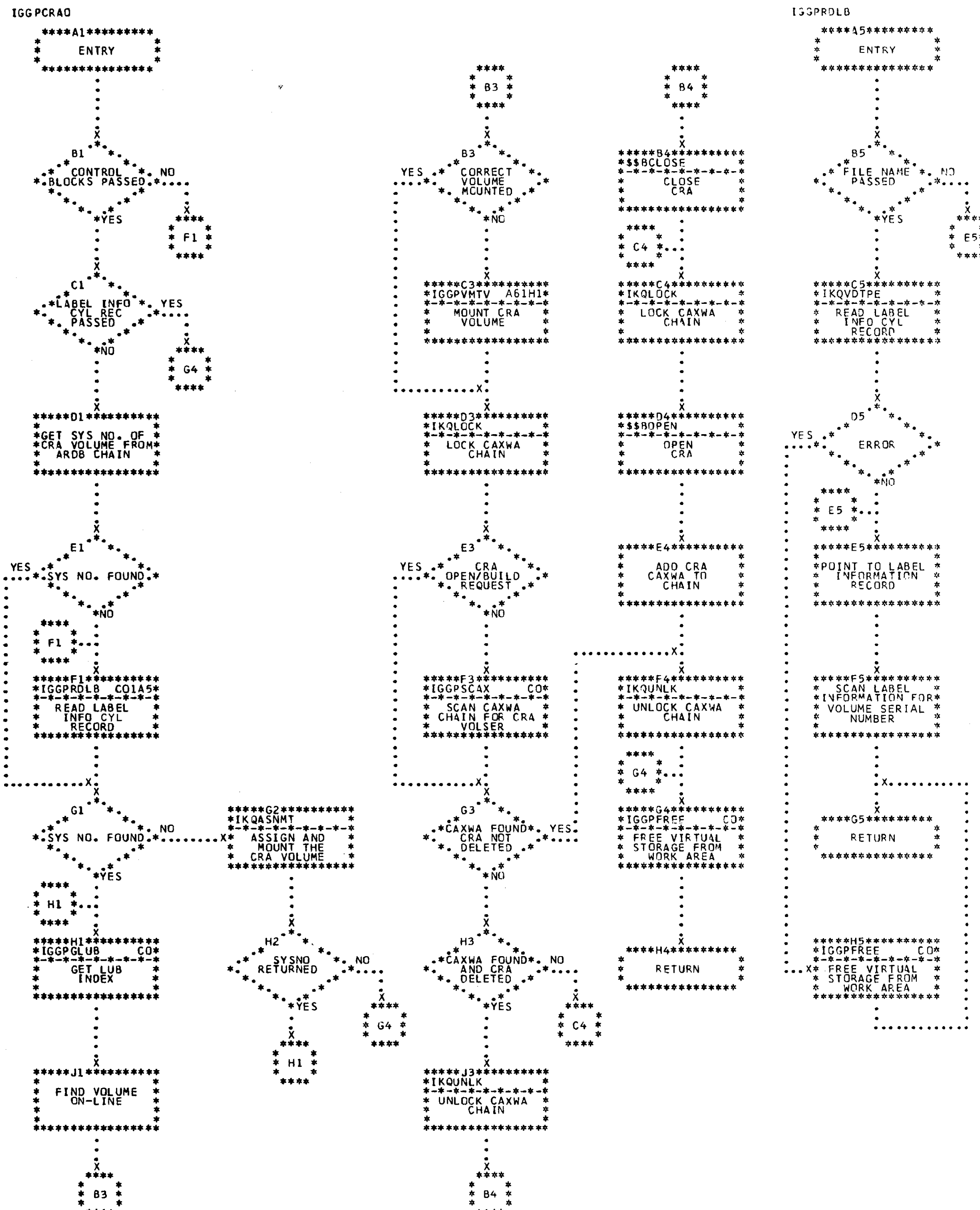


Chart CO1. Open Catalog Recover Area (IGG0CLCO)

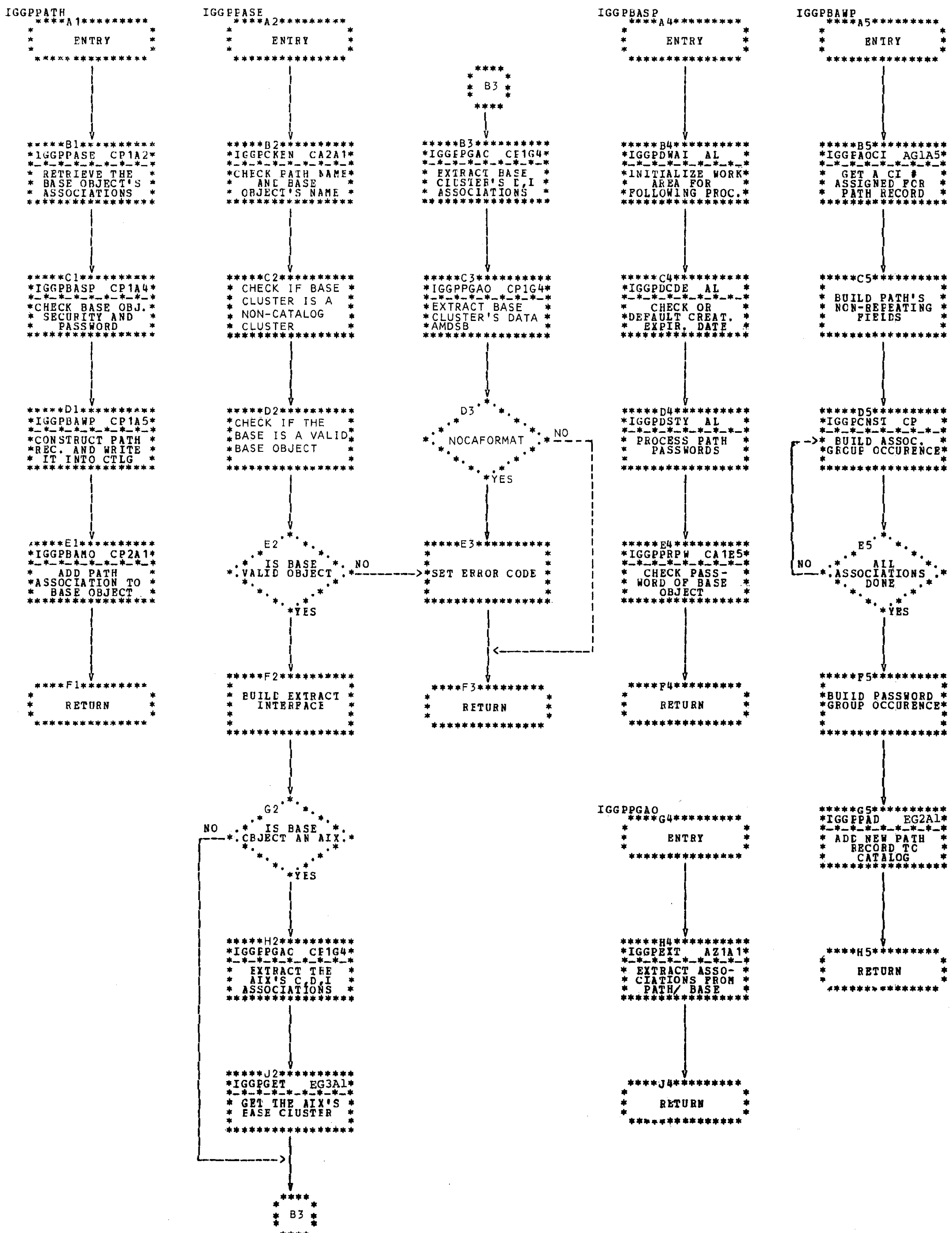


Chart CP1. Define Path (IGG0CLCP)

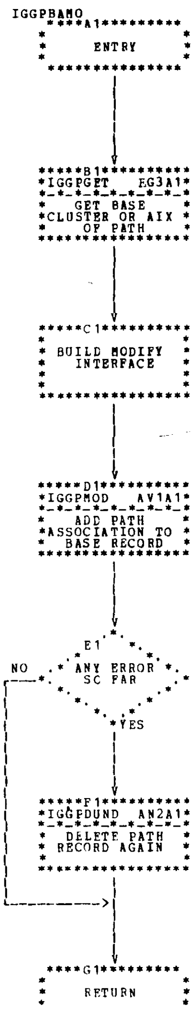


Chart CP2. Define Path (IGG0CLCP)

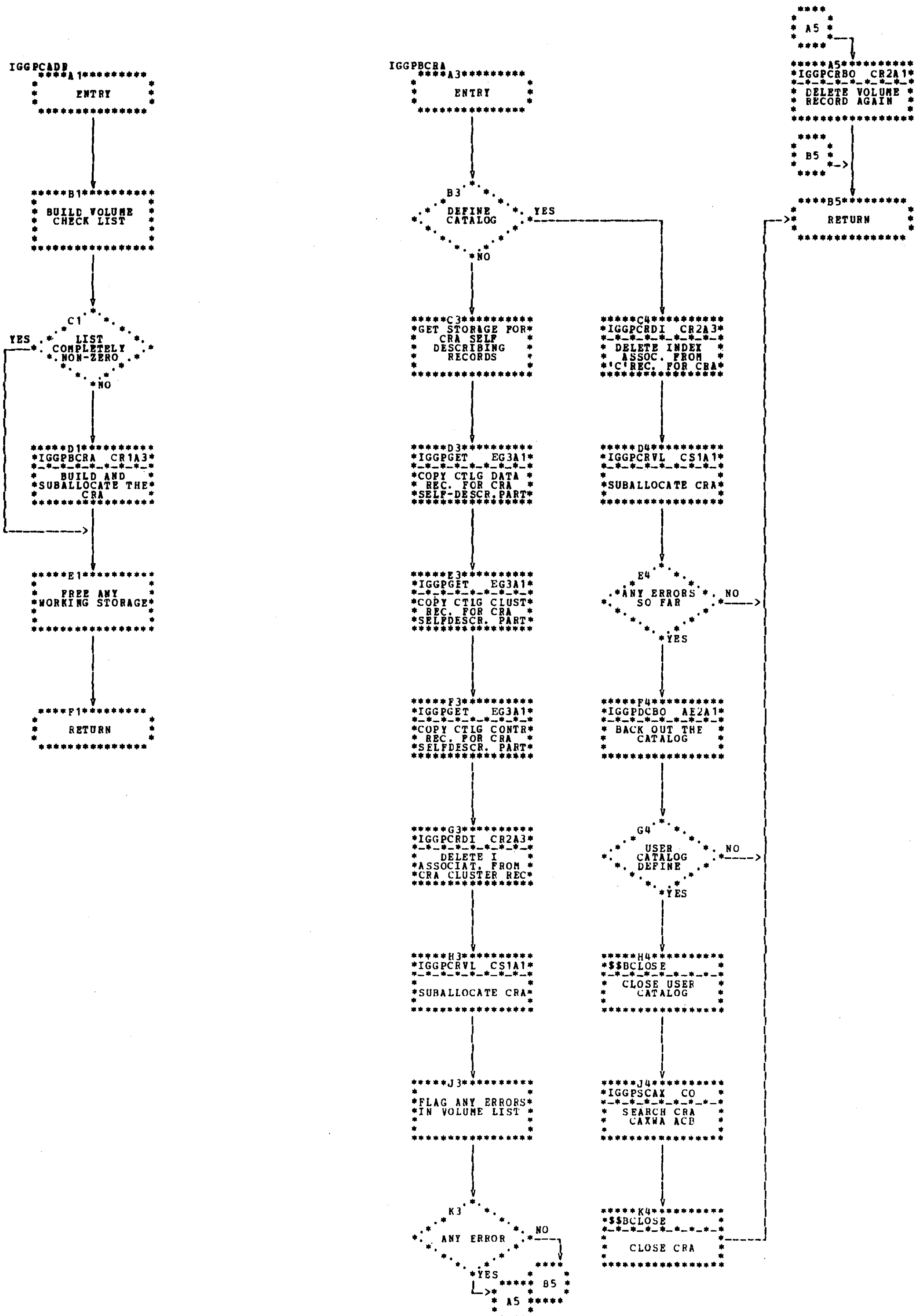


Chart CR1. Define Catalog Recovery Area - First Module (IGG0CLCR)

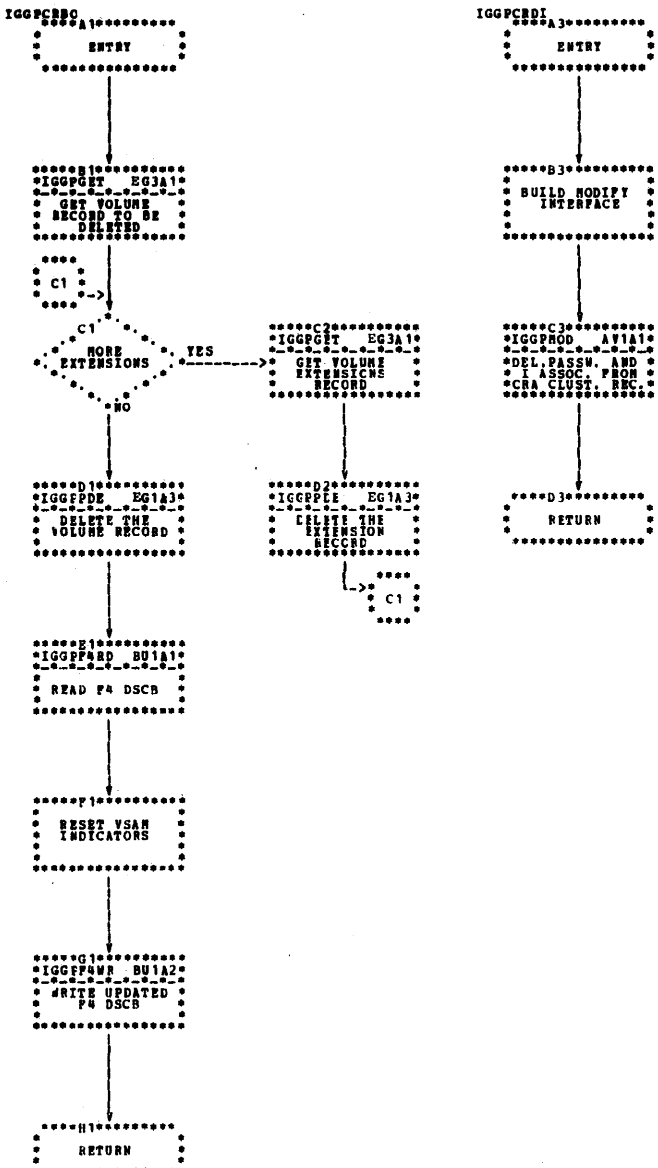


Chart CR2. Define Catalog Recovery Area - First Module (IGG0CLCR)

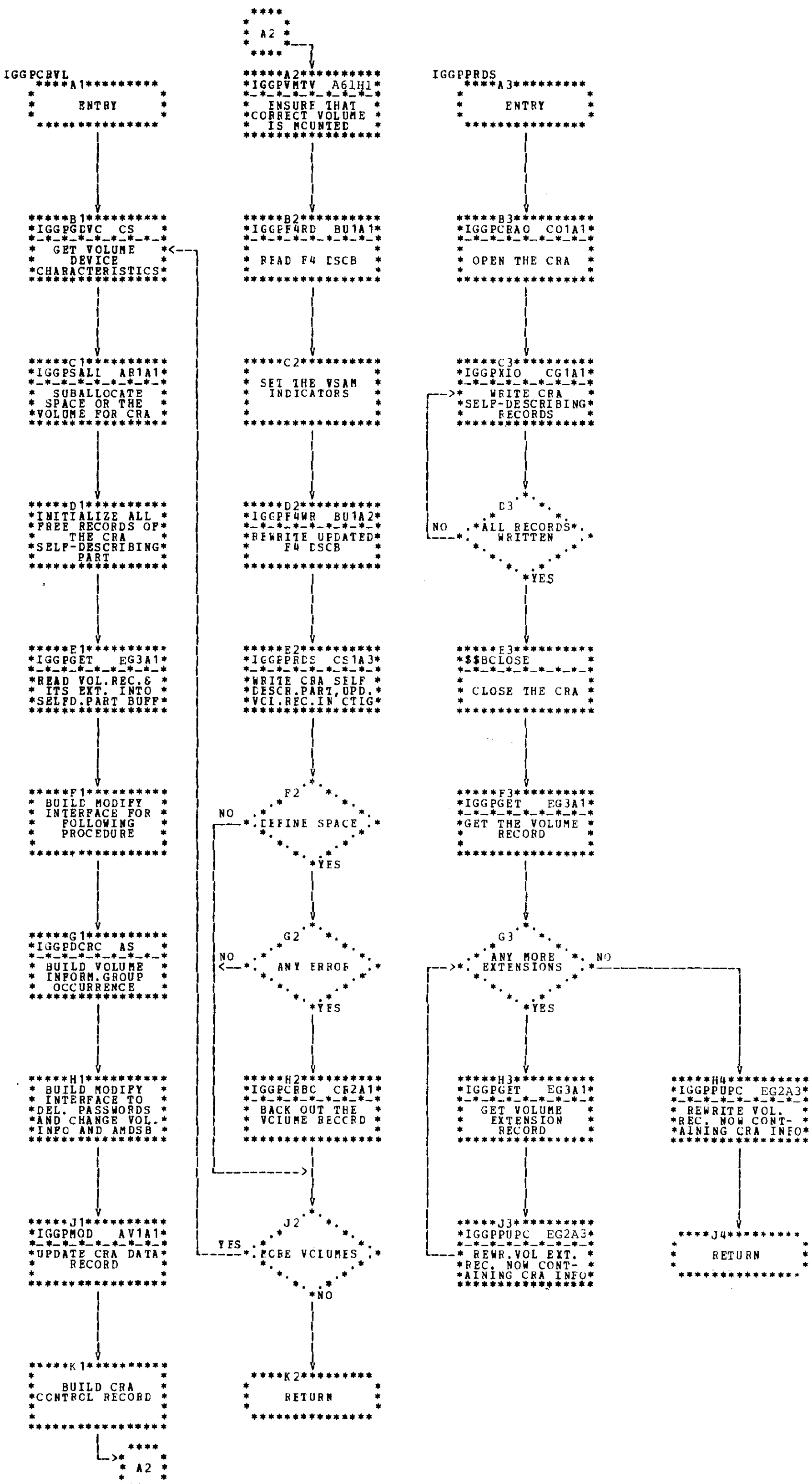


Chart CS1. Define Catalog Recovery Area - Second Module (IGG0CLCS)

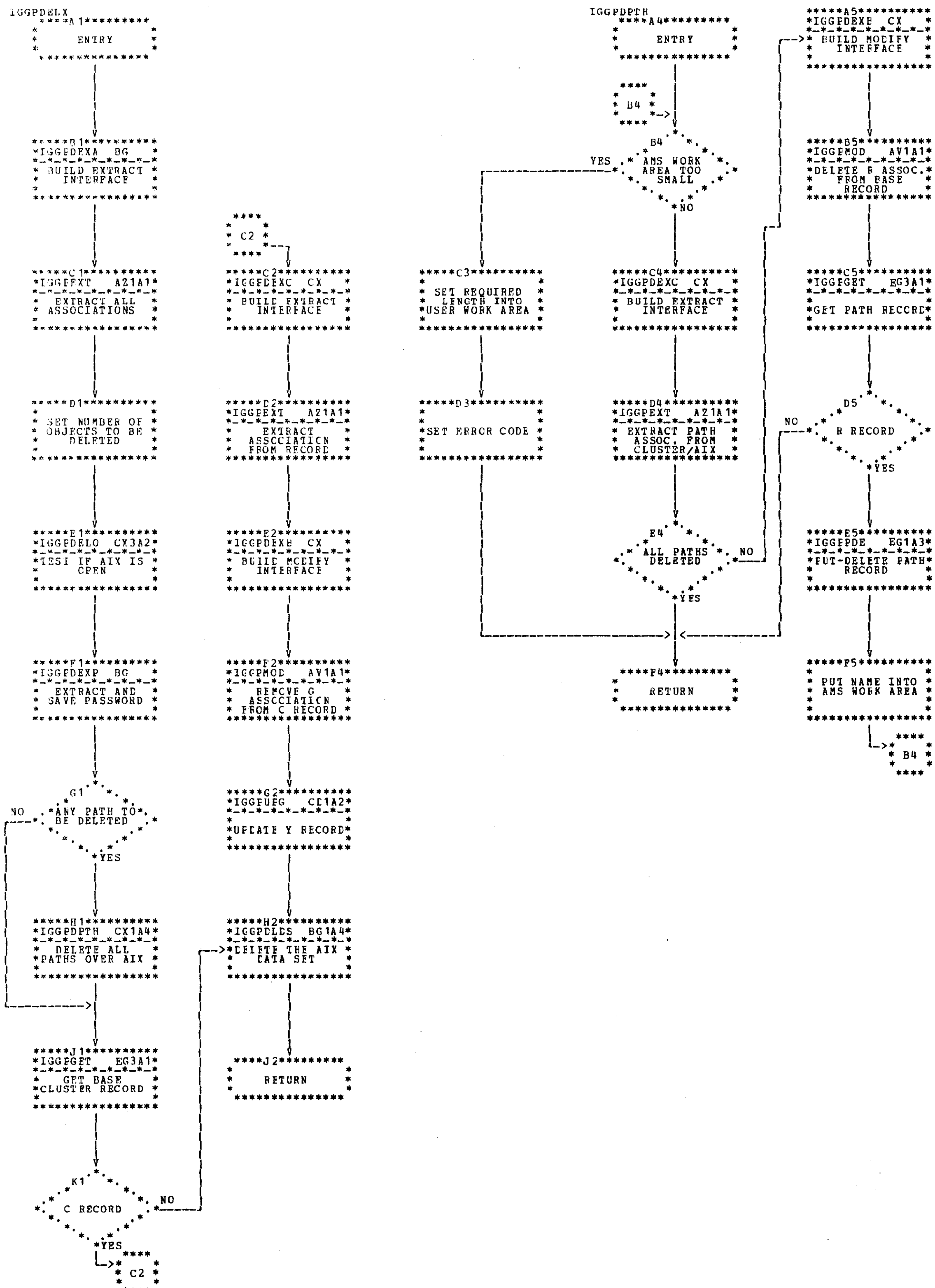


Chart CX1. CMS Delete - Third Module (IGG0CLCX)

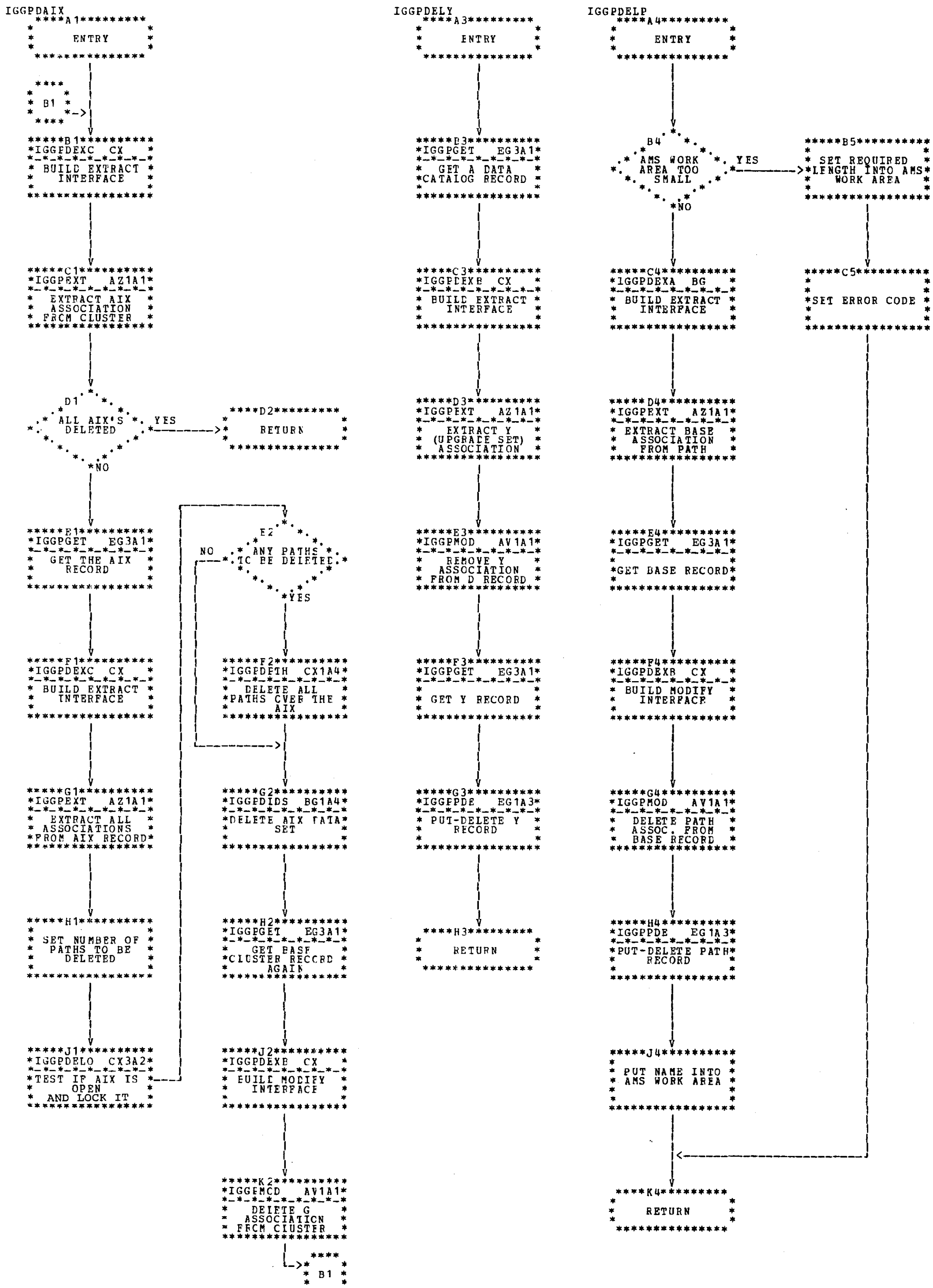


Chart CX2. CMS Delete - Third Module (IGG0CLCX)

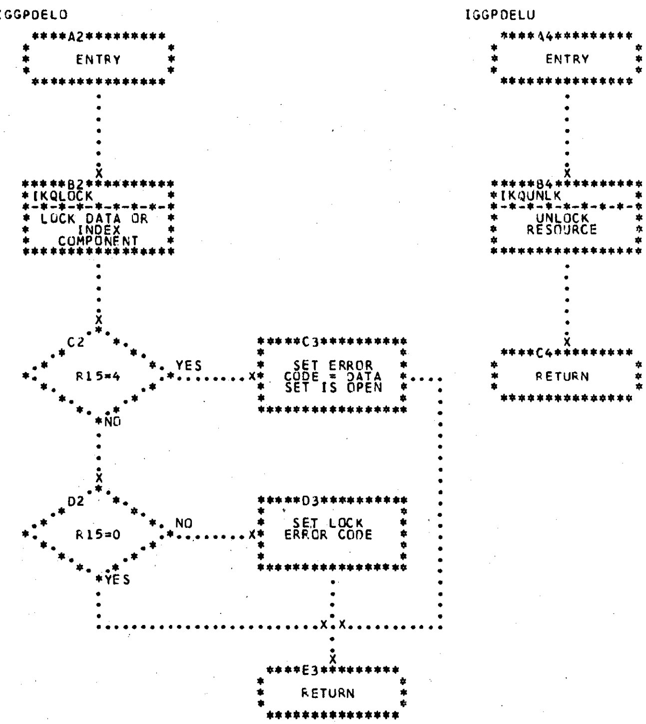


Chart CX3. CMS Delete - Third Module (IGG0CLCX)

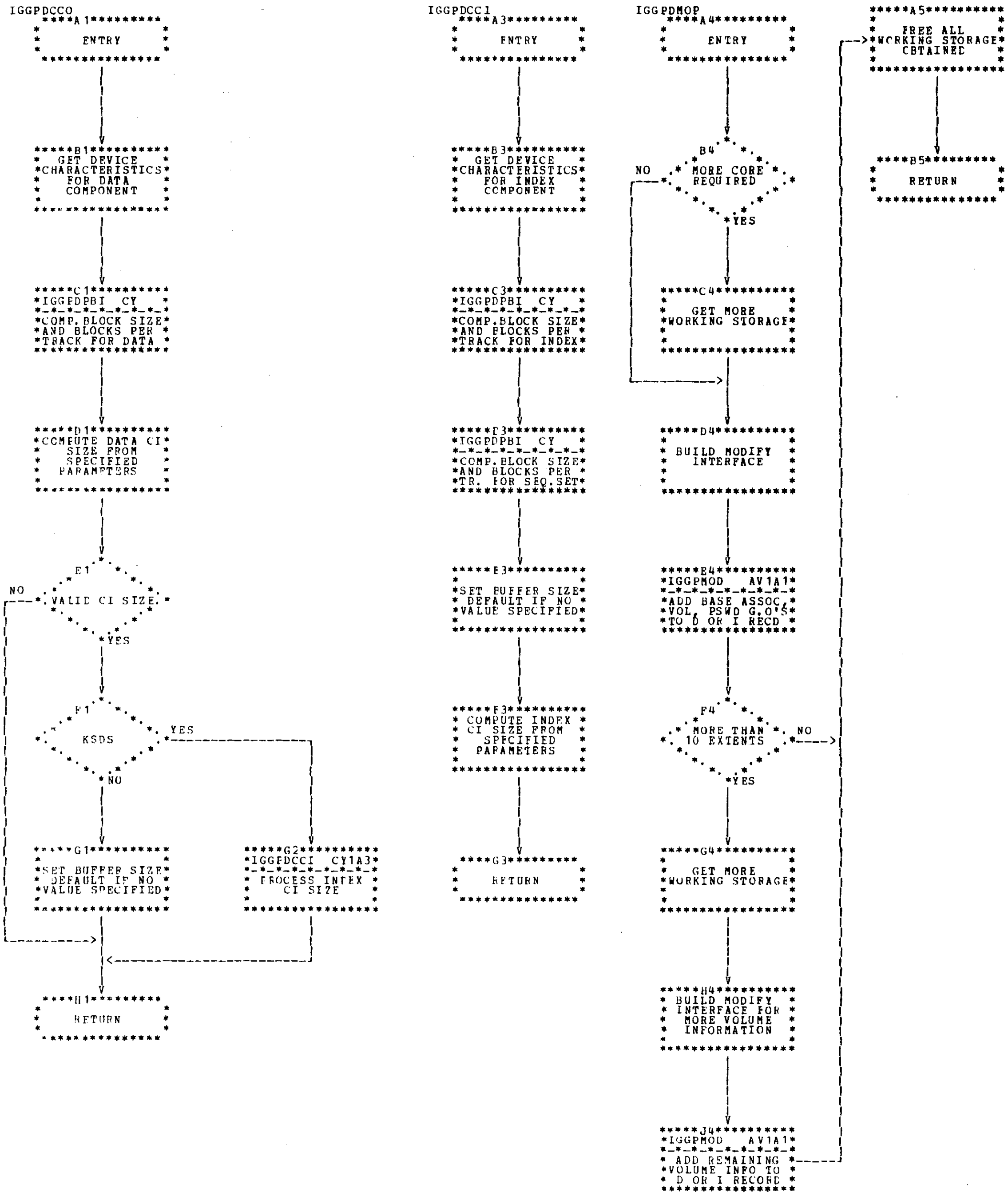


Chart CY1. CMS Define - Sixth Module (IGG0CLCY)

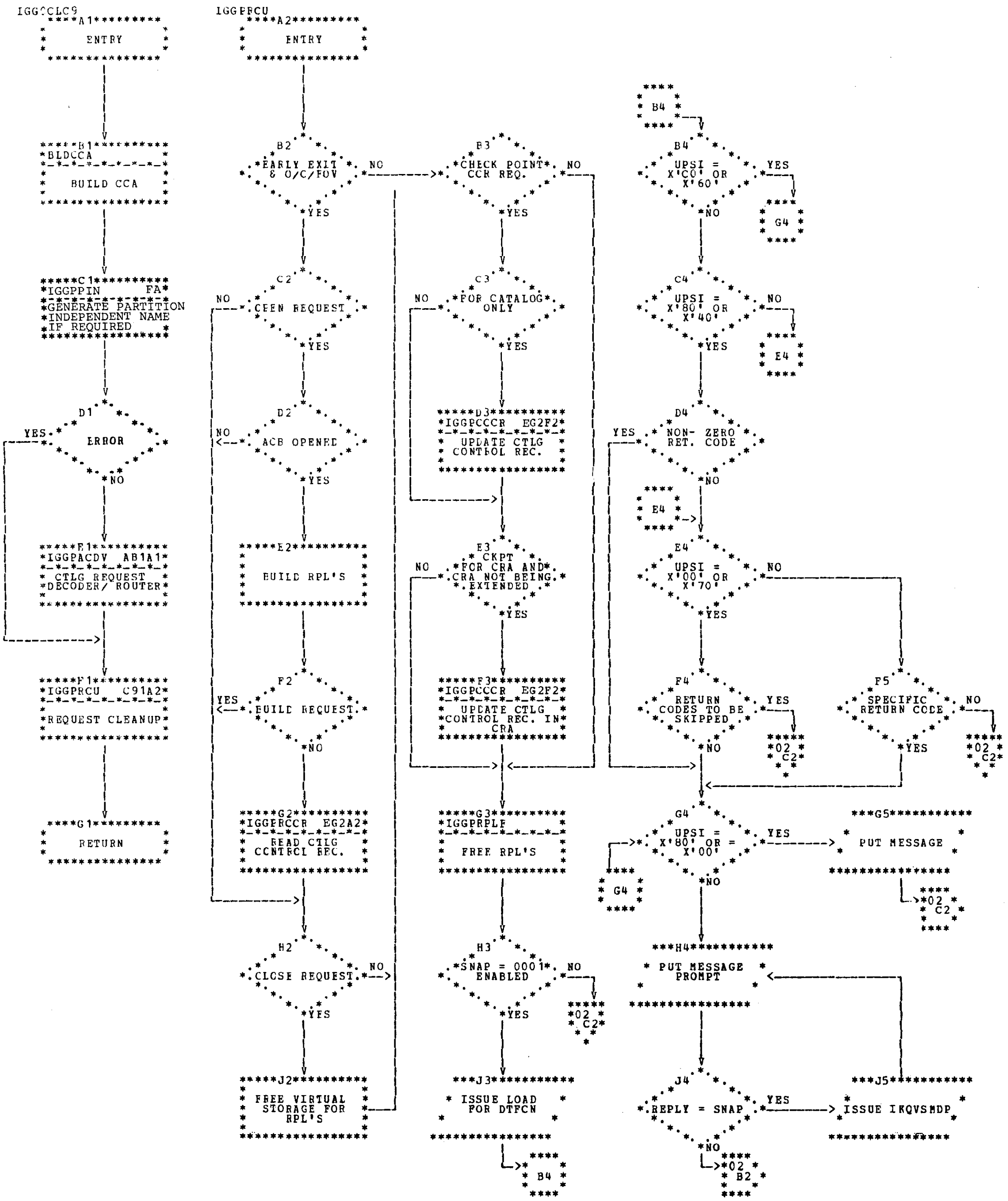


Chart C91. Catalog First Module (IGG0CLC9)

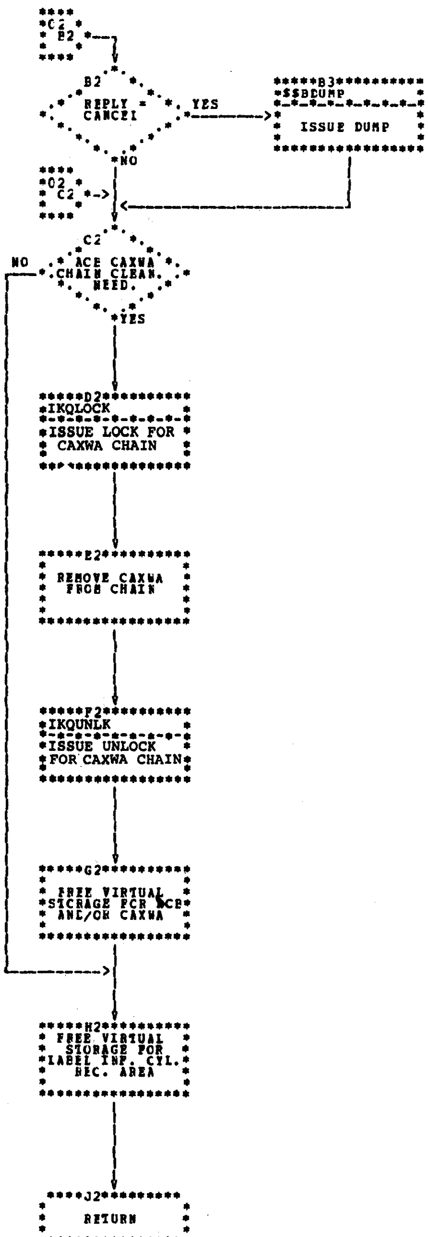


Chart C92. Catalog First Module (IGG0CLC9)

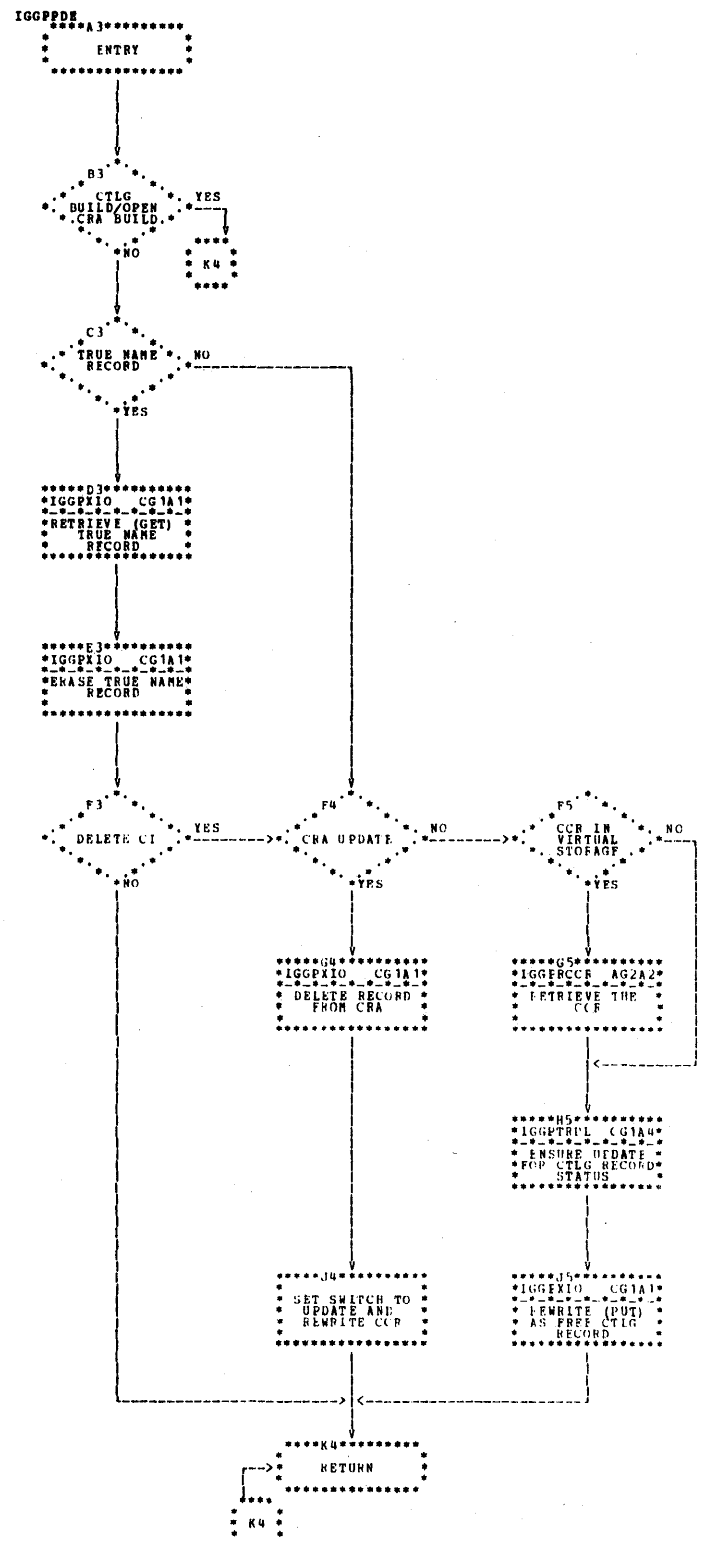


Chart EG1. Catalog I/O Subfunction (IGG0CLEG)

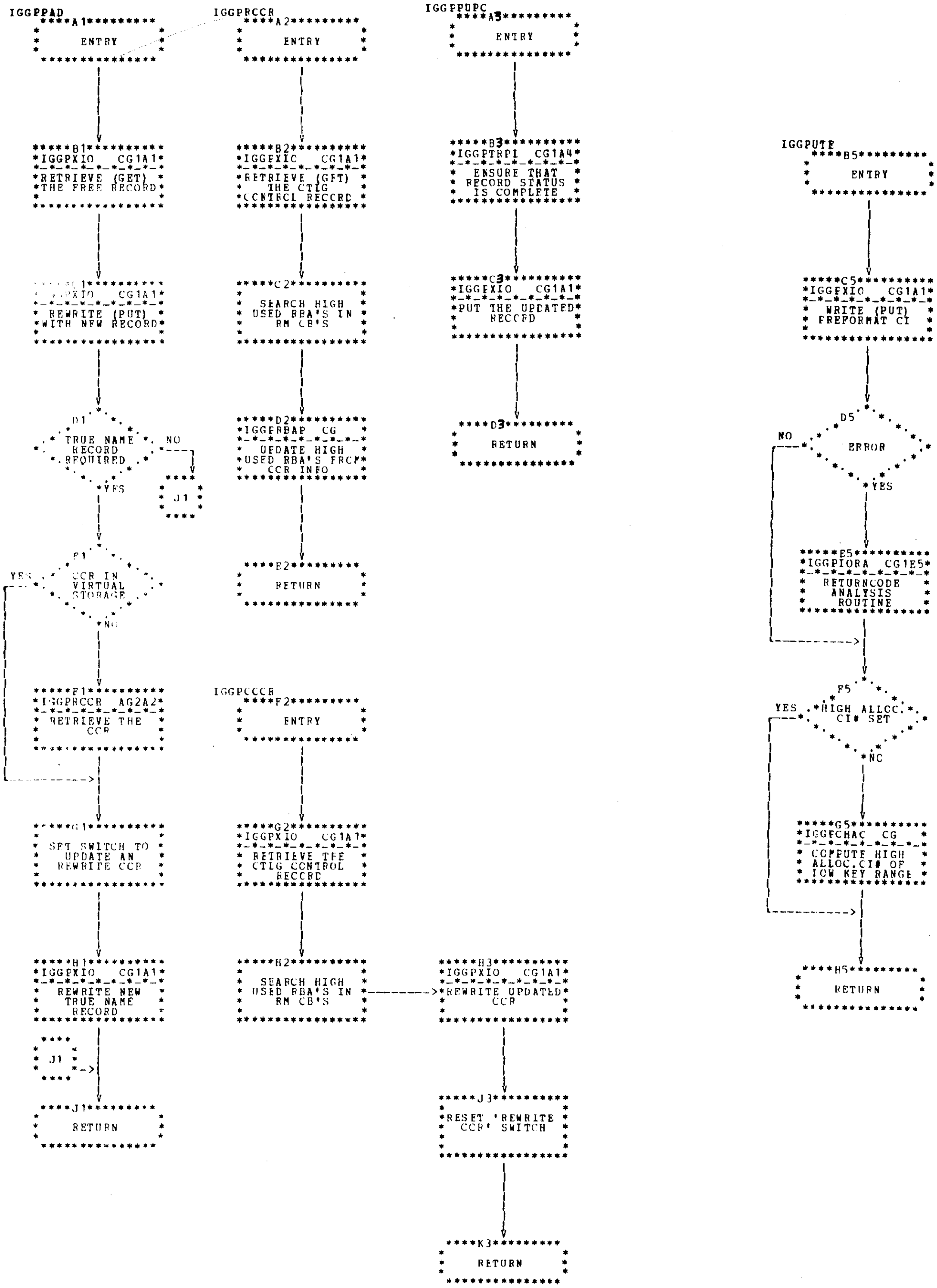


Chart EG2. Catalog I/O Subfunction (IGG0CLEG)

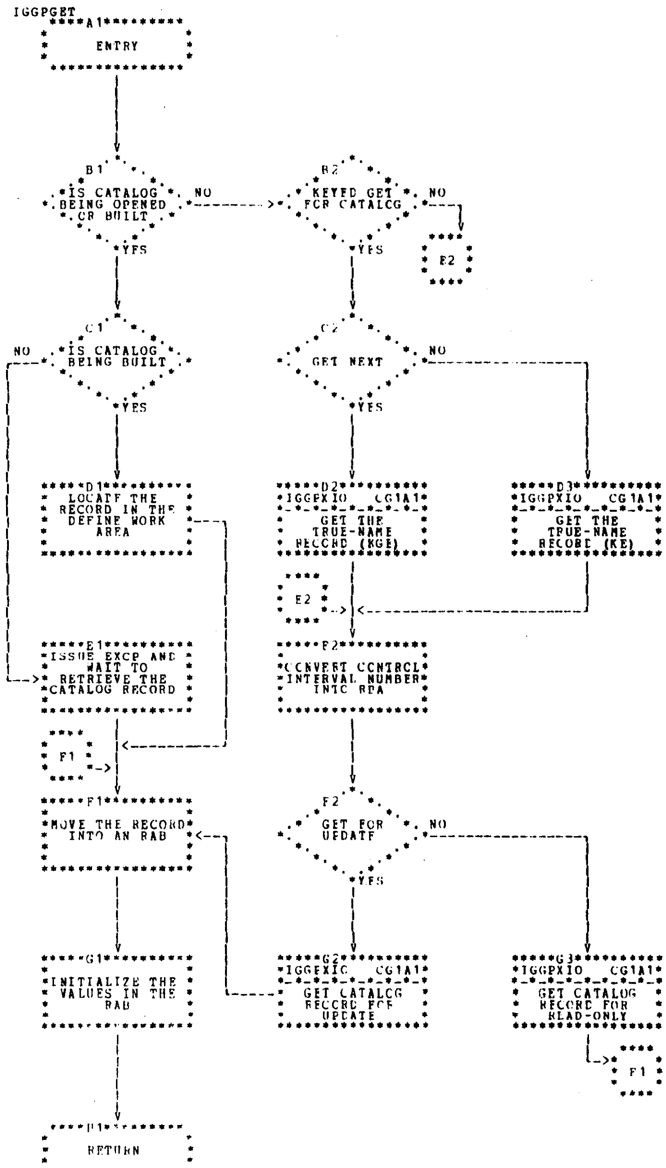


Chart EG3. Catalog I/O Subfunction (IGG0CLEG)

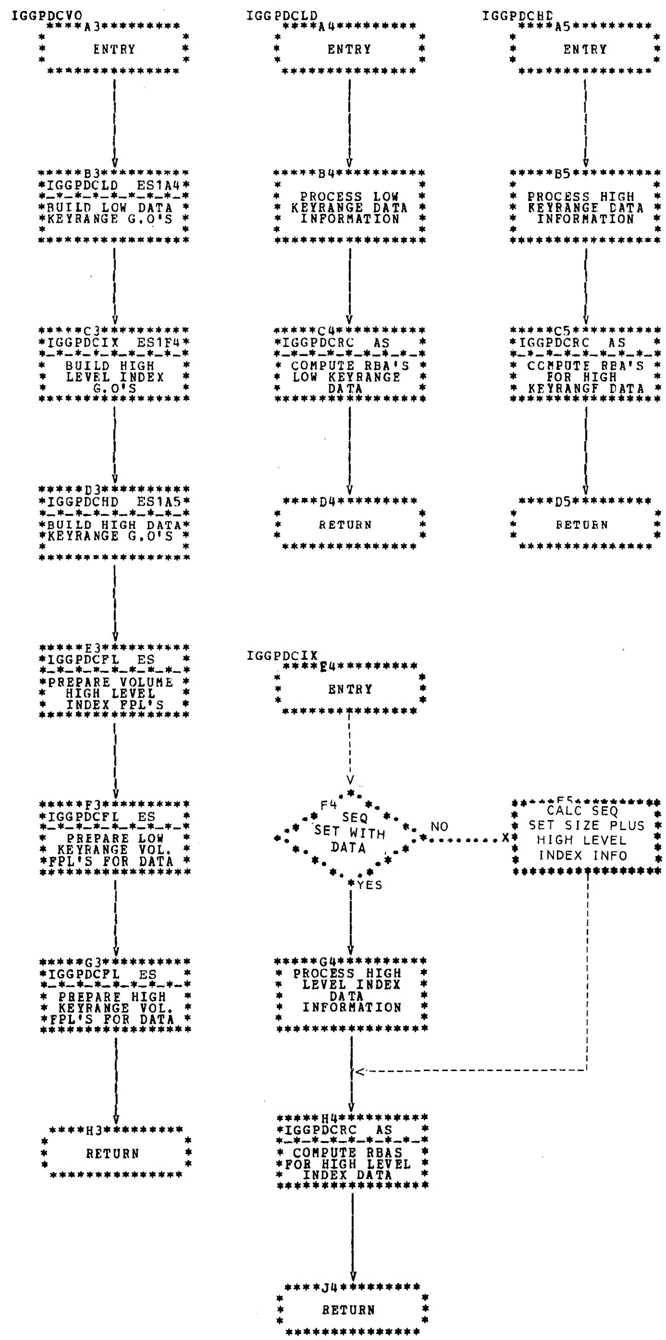


Chart ES1. Catalog Definition Processing (IGG0CLES)

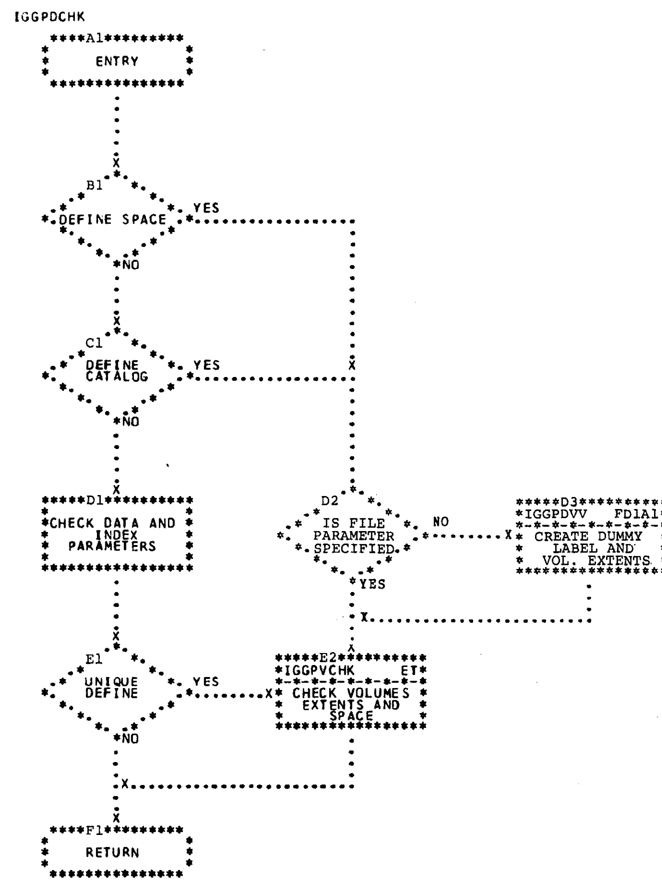


Chart ET1. CMS Request Validity Checking (IGG0CLET)

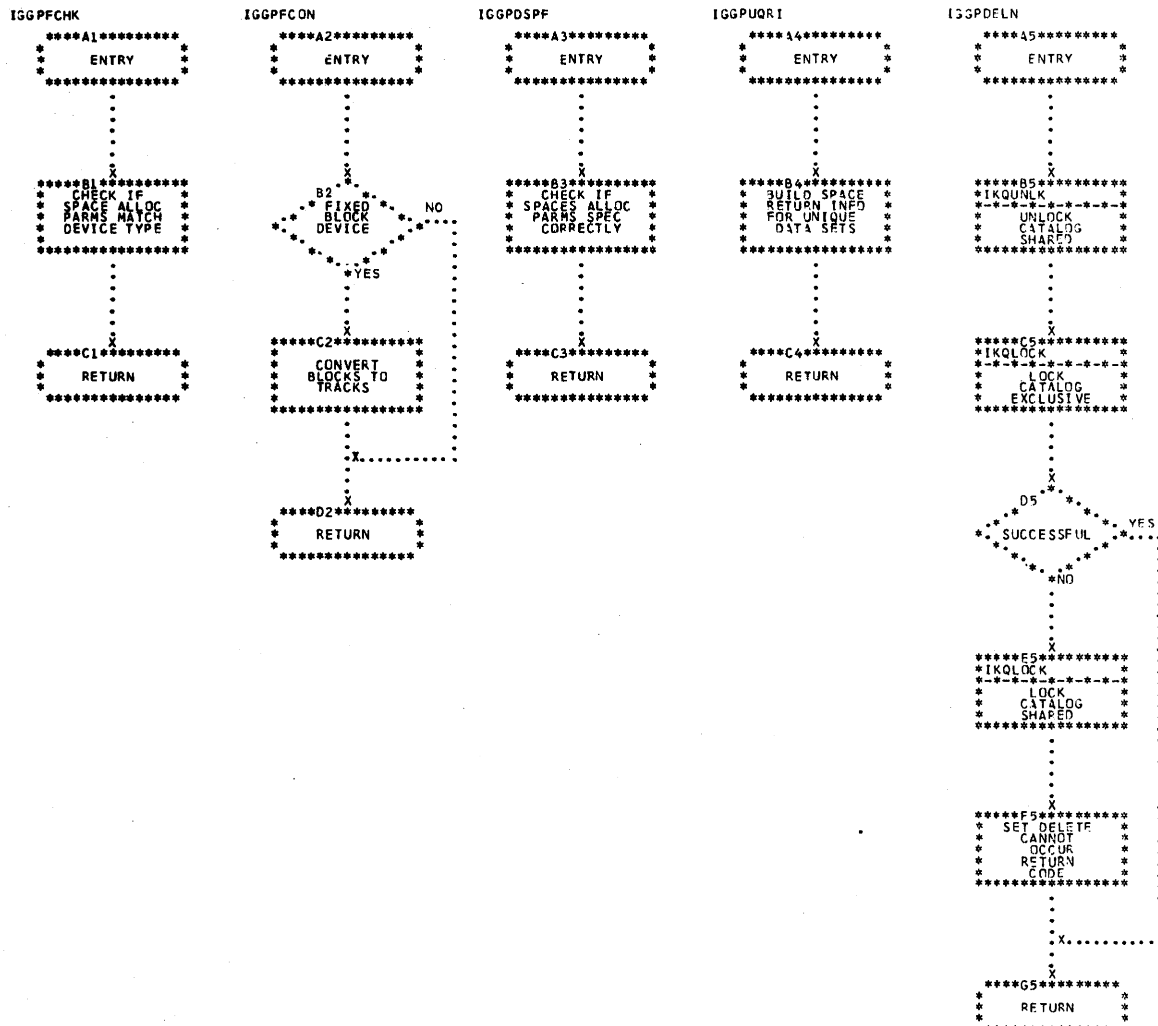


Chart EX1. CMS Define Block Conversion (IGG0CLEX)

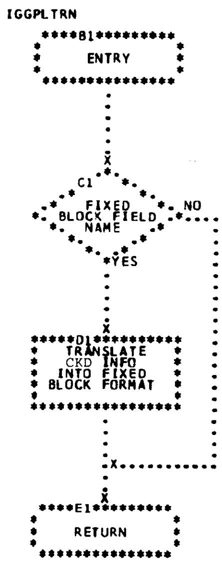


Chart EZ1. CKD to Fixed Block Translation (IGGOCLEZ)

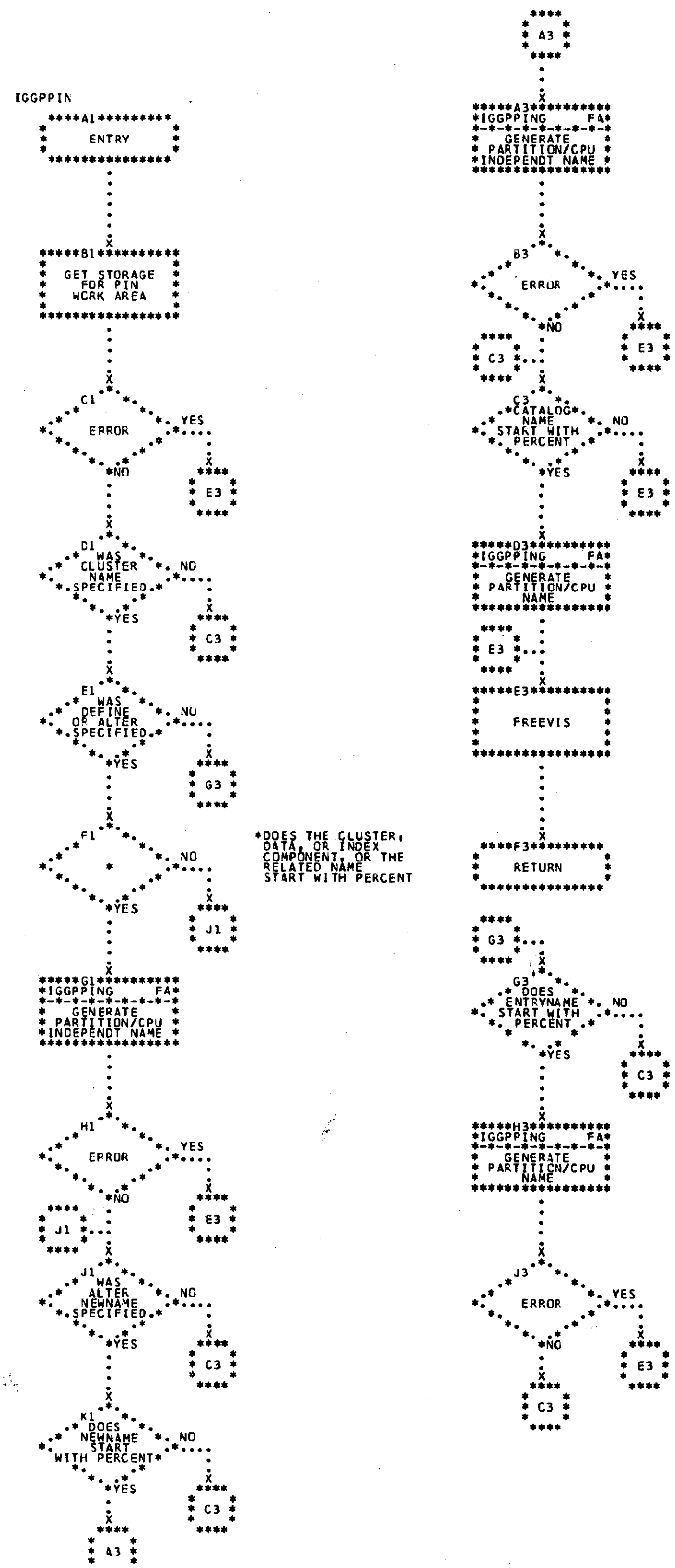


Chart FA1. Generate Partition/CPU Independent Name (IGG0CLFA)

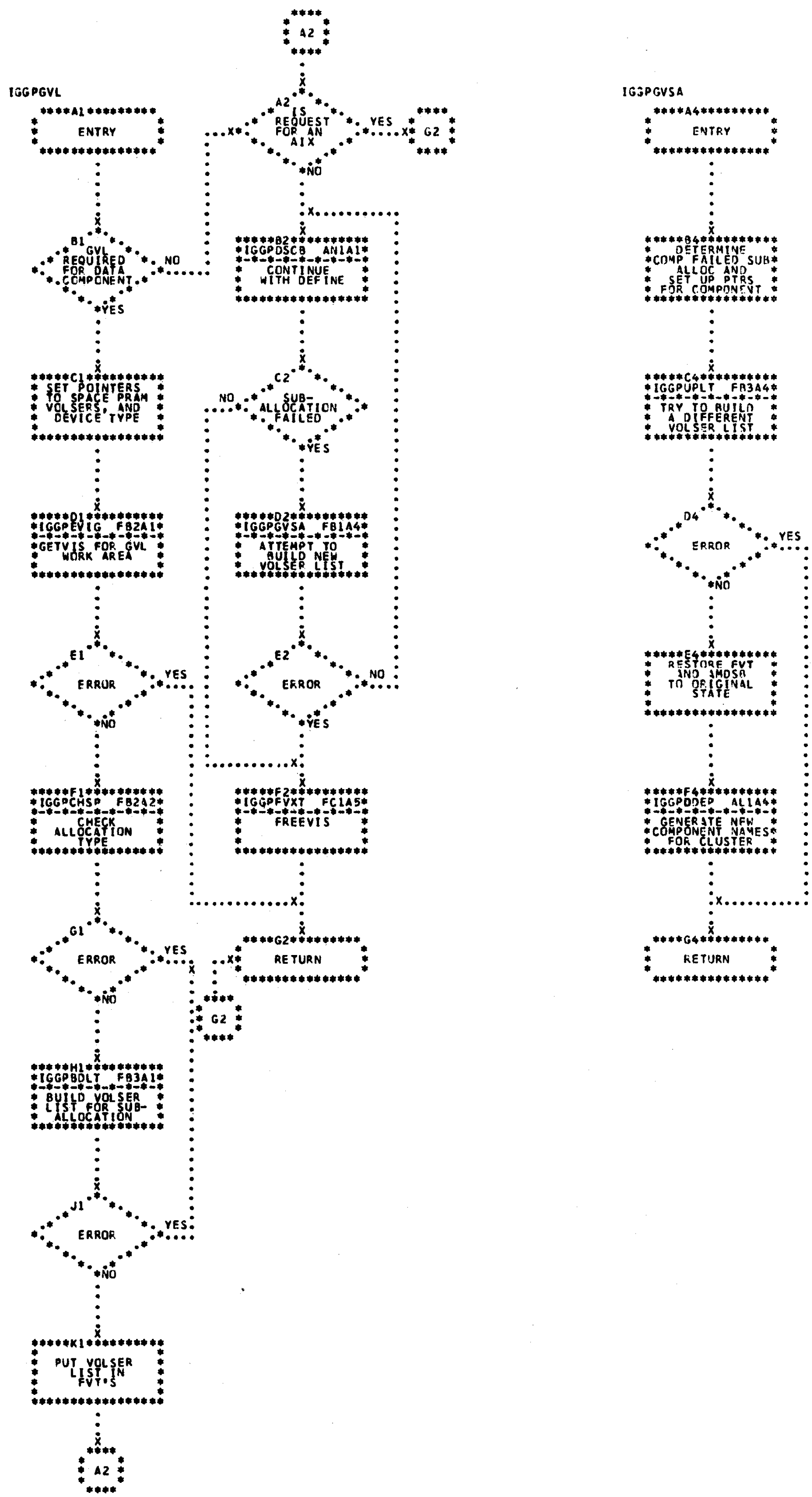


Chart FB1. Generate Volume List (1st Module) (IGG0CLFB)

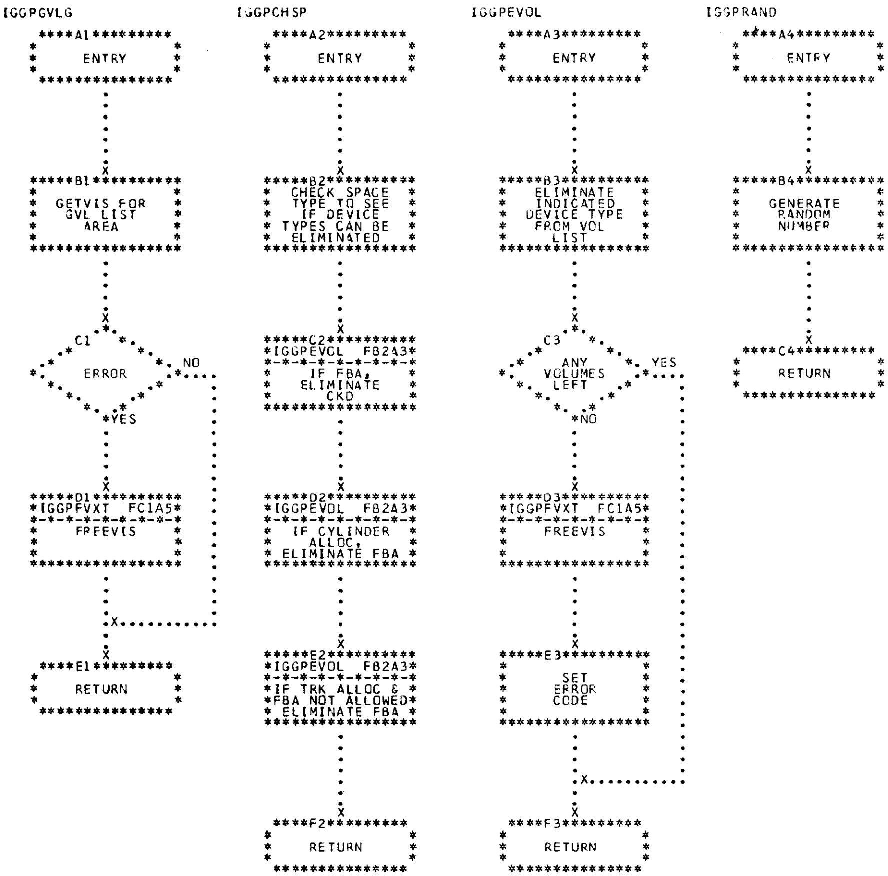


Chart FB2. Generate Volume List (1st Module) (IGG0CLFB)

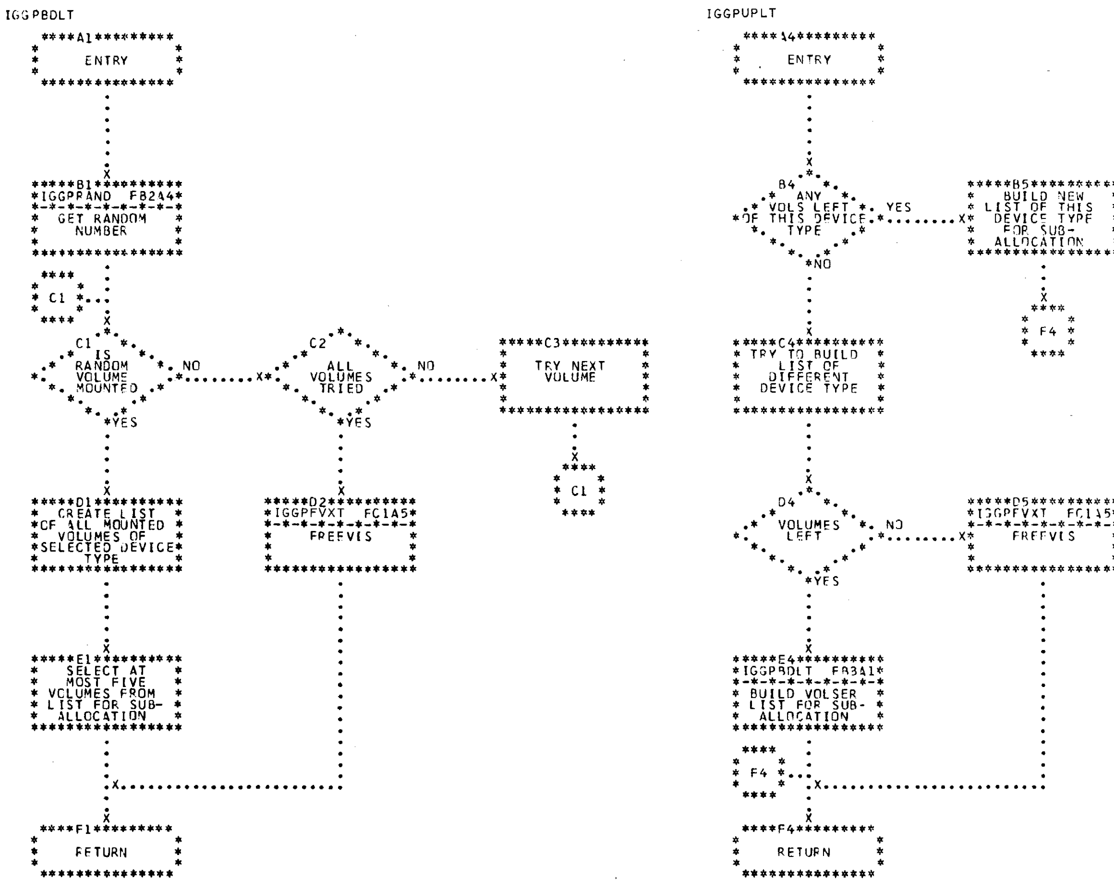


Chart FB3. Generate Volume List (1st Module) (IGG0CLFB)

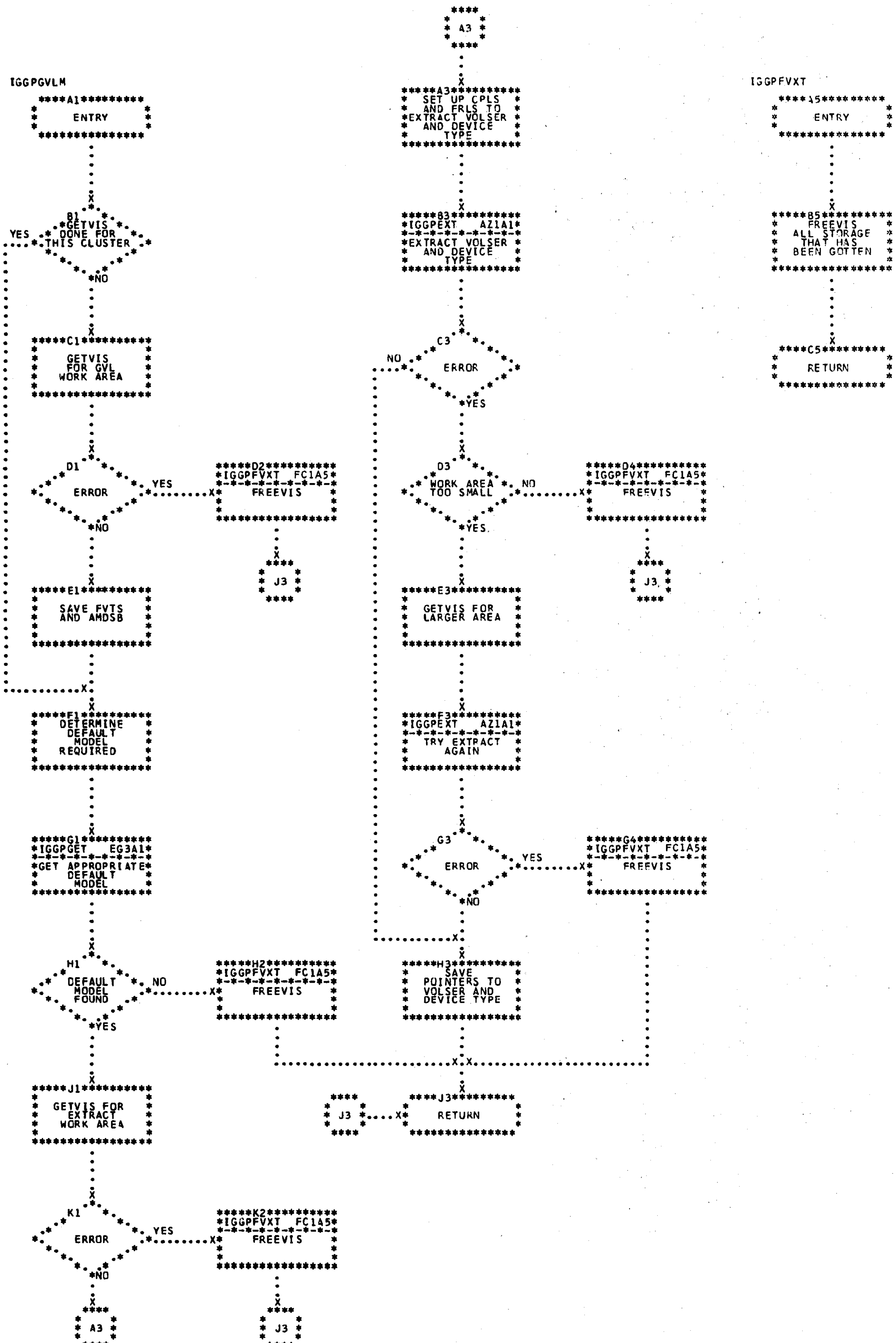


Chart FC1. Generate Volume List (2nd Module) (IGG0CLFC)

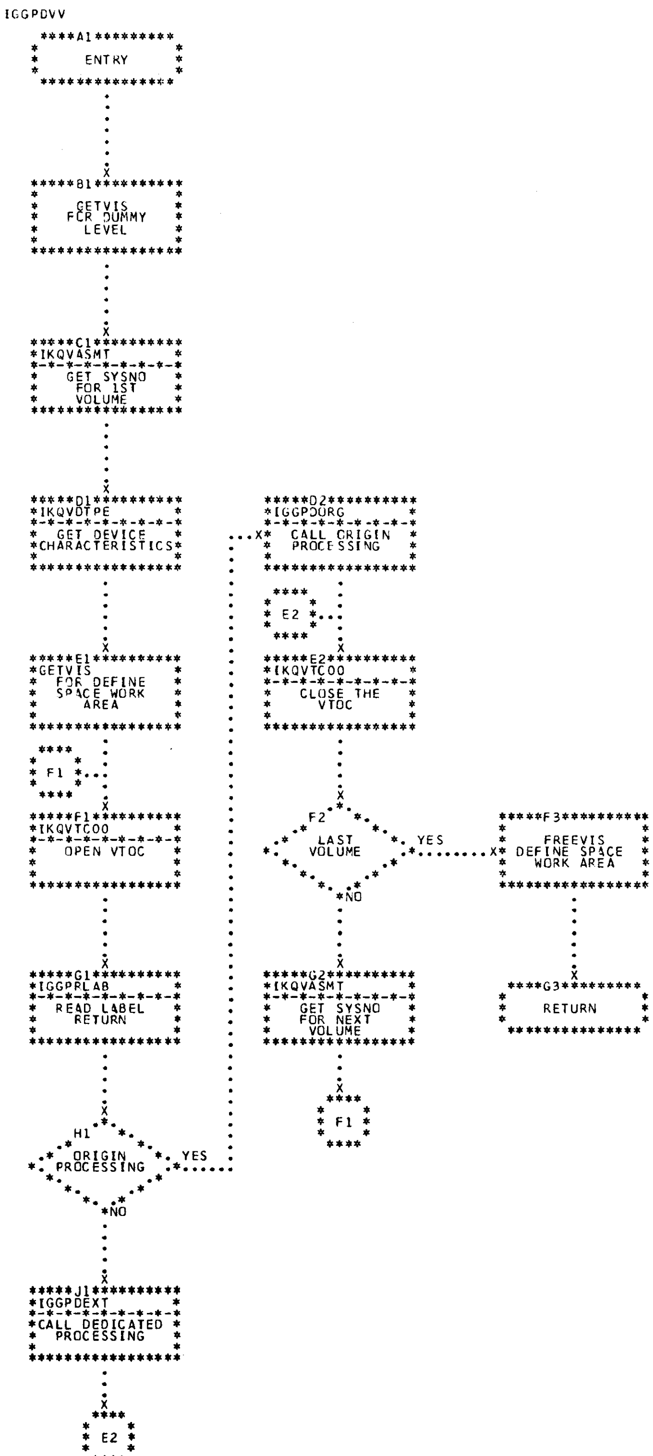


Chart FD1. Dedicated VSAM Volume (1st Module) (IGG0CLFD)

```
IGGPDEXT
 ****A1*********
 *             *
 *    ENTRY    *
 *             *
 ***************
        :
        :
        X
       .*.
   B1 .   *.
    .*  IS   *.
 THIS REQUEST  *. NO
 *.  CATALOG  .*....
   *. DEFINE .*      :
     *.    .*        X
       *. .*        ****
        *YES       *    *
        :          * E1 *
        :          *    *
        X           ****
 *****C1*********
 *              *
 *  CALCULATE   *
 *   MINEXT     *
 * FOR CATALOG  *
 *              *
 ****************
        :
        :
        X
 *****D1*********
 *IGGPSPC  FQ1F1*
 *-*-*-*-*-*-*-*-*
 *  CALCULATE   *
 *    TRACK     *
 *  ALLOCATION  *
 ****************
  ****  :
 *    * :
 * E1 *..:
 *    * :
  ****  X
 *****E1*********
 *              *
 *  SCAN BITMAP *
 *  TO BUILD UP *
 * TO 16 EXTENTS*
 *              *
 ****************
        :
        :
        X
       .*.
   F1 .   *.
    .*  IS   *.
   .*  MORE    *.  YES              *****F2**********
 *SPACE REQUIRED.*........X*IGGPLBLS  FQ1A3*
   *.   FOR   .*            *-*-*-*-*-*-*-*-*
     *.LABEL.*              *  GETVIS MORE *
       *. .*                *  LABEL SPACE *
        *NO                 *              *
        :                   ****************
        :                          :
        :X.........................:
        X
       .*.
   G1 .   *.                *****G2**********
    .*  IS   *.             *IGGPSPC  FQ1F1*
   .*   IT     *.  YES      *-*-*-*-*-*-*-*-*
 *.   DEFAULT  .*........X*  CALCULATE    *
   *.  ORIGIN .*            *    TRACK     *
     *.     .*              *  ALLOCATION  *
       *. .*                ****************
        *NO                        :
        :                          :
        :X.........................:
        X
 *****H1*********
 *              *
 *    BUILD     *
 *  EXTENTS IN  *
 *    LABEL     *
 *              *
 ****************
        :
        :
        X
 ****J1*********
 *             *
 *    RETURN   *
 *             *
 ***************
```

Chart FE1. Dedicated VSAM Volume (2nd Module) (IGG0CLFE)

```
IGGPDORG
   ****A1*********
  *               *
  *     ENTRY     *
  *               *
   ***************
          :
          :
          :
          X
  *****B1**********
  *IGGPSPC    FQ1F1*
  *-*-*-*-*-*-*-*-*
  * CCNVERT ALLOC  *
  * AND BEGINNING  *
  * LOC TO TRACKS  *
  *****************
          :
          :
          X
  *****C1**********
  *FIND BEGINNING  *
  * LOCATION AND   *
  *  CHECK IF IN   *
  *      USE       *
  *                *
  *****************
          :
          :
          X
  *****D1**********
  *   FIND END     *
  *     TRACK      *
  *  LOCATION OF   *
  *    EXTENT      *
  *                *
  *****************
          :
          :
          X
  *****E1**********
  *                *
  *     BUILD      *
  *   EXTENT IN    *
  *     LABEL      *
  *                *
  *****************
          :
          :
          X
   ****F1*********
  *               *
  *    RETURN     *
  *               *
   ***************
```

Chart FF1. ORIGIN for VSAM Volume (IGG0CLFF)

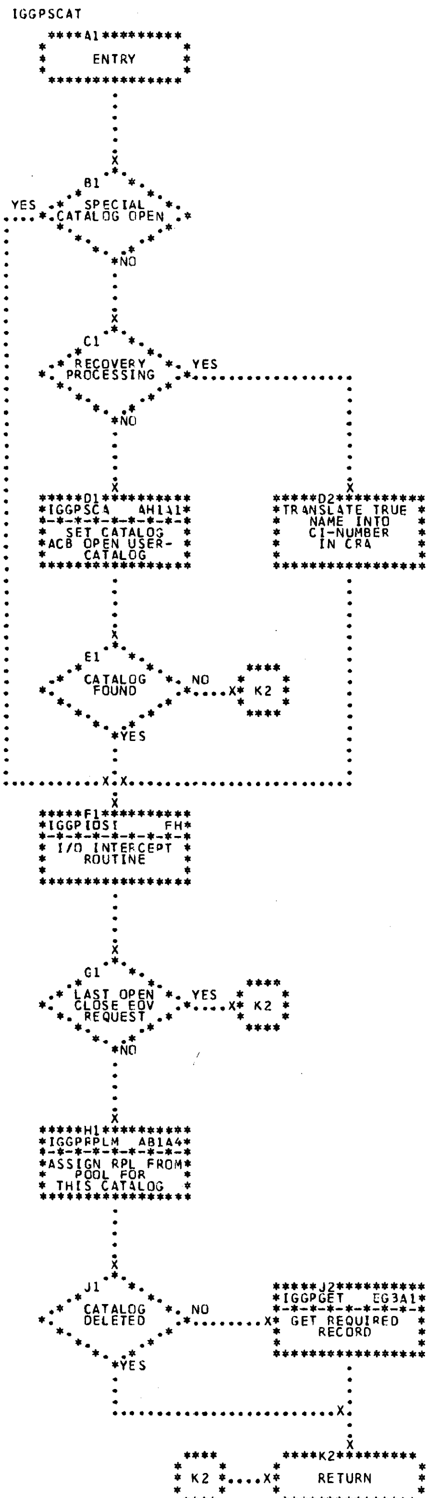


Chart FH1. Search/Open Catalog (IGG0CLFH)

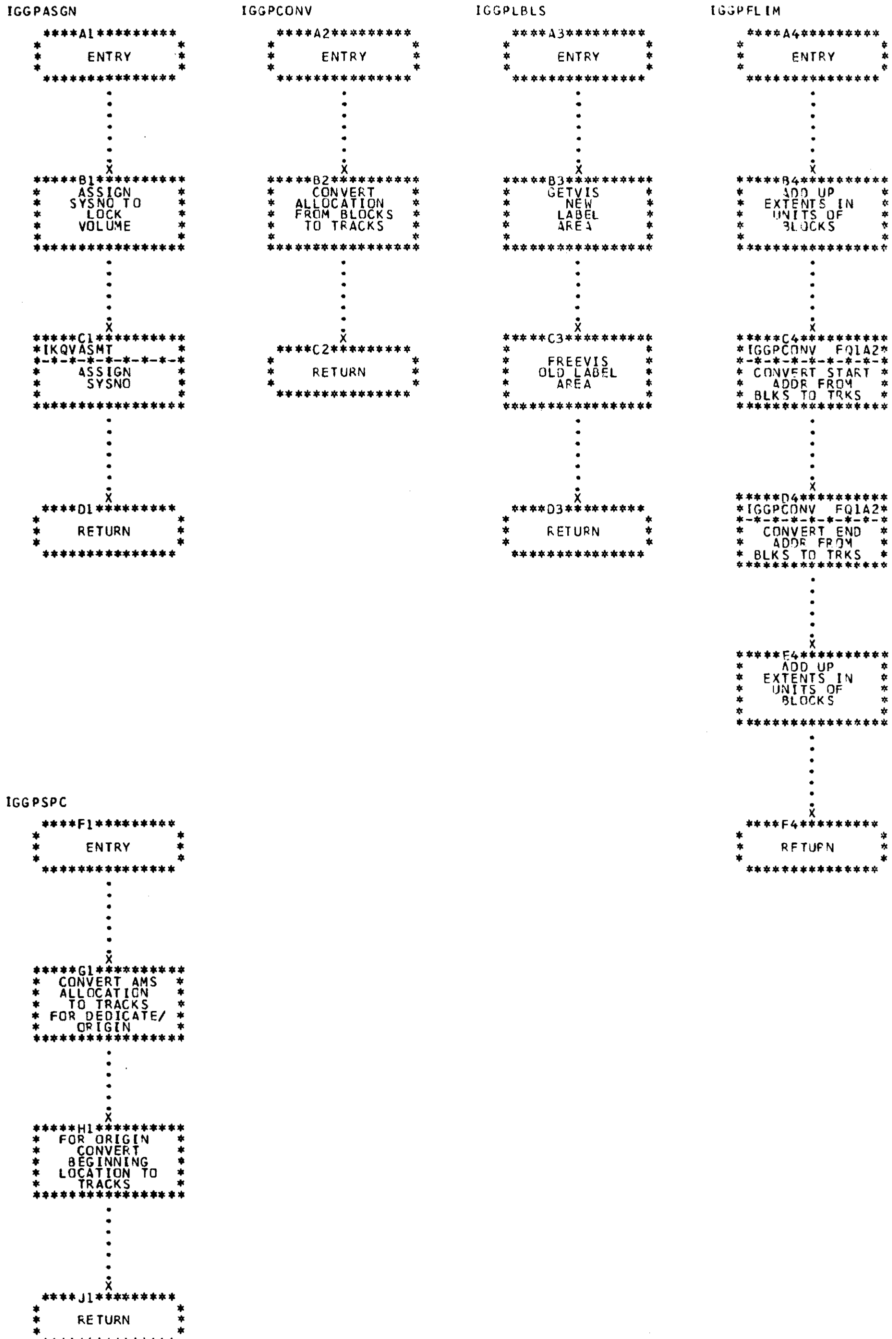


Chart FQ1. CMS Define Space (Third Load) (IGG0CLFQ)

Section 4. Directory

This section contains the following cross-reference material:

- VSAM Phase-to-Module Index
- IIP Phase-to-Module Index
- Component Index
- Module Directory
- Routine Directory
- Catalog External Entry Points
- Data Area Directory

VSAM Phase-to-Module Index

The core image library contains the VSAM phases. Their names are identifiable by IKQV,$S, or $$B. Packaged within the phases are the VSAM modules, identifiable by the leading characters IKQ, IGG0, $S or $$B. Two service aid phases, IKQVDU and IKQVEDA are not included in the link-edit of VSAM and must be placed in the core image library by executing a job described in *Service Aids*.

The following list includes the phase names and the names of the modules included within each phase.

Phase name	Module name(s)			
IKQFTIND	IKQFTIND IKQFT1 IKQFT2 IKQFT3			
IKQVASMT	IKQASNMT IKQMTMSG			
IKQVBRP	IKQBRP			
IKQVCAT	IGG0CLAB IGG0CLAC IGG0CLAD IGG0CLAE IGG0CLAF IGG0CLAG IGG0CLAH IGG0CLAJ IGG0CLAK IGG0CLAL IGG0CLAN IGG0CLAP IGG0CLAQ IGG0CLAR IGG0CLAS IGG0CLAT IGG0CLAU IGG0CLAV IGG0CLAW	IGG0CLAX IGG0CLAY IGG0CLAZ IGG0CLA6 IGG0CLA7 IGG0CLA8 IGG0CLBA IGG0CLBB IGG0CLBC IGG0CLBD IGG0CLBE IGG0CLBF IGG0CLBG IGG0CLBH IGG0CLBL IGG0CLBM IGG0CLBN IGG0CLBQ IGG0CLBR	IGG0CLBS IGG0CLBT IGG0CLBU IGG0CLBW IGG0CLBX IGG0CLBY IGG0CLB8 IGG0CLCA IGG0CLCB IGG0CLCD IGG0CLCG IGG0CLCL IGG0CLCO IGG0CLCP IGG0CLCR IGG0CLCS IGG0CLCX IGG0CLCY IGG0CLC9	IGG0CLEG IGG0CLES IGG0CLET IGG0CLEX IGG0CLEZ IGG0CLFA IGG0CLFB IGG0CLFC IGG0CLFD IGG0CLFE IGG0CLFF IGG0CLFH IGG0CLFQ IKQALL00 IKQPOP00 IKQRDS00 IKQREN00 IKQSCR00 IKQVTC00 IKQWDS00
IKQVCLC	IKQCLCAT			
IKQVCLIF	IKQCLIF			
IKQVCLOC	IKQCLOCL			
IKQCLOS	IKQCLO			
IKQVCLOV	IKQCLOVY			
IKQVDCN	IKQDCN			
IKQVDNT	IKQDNT			
IKQVDRP	IKQDRP			
IKQVDTPE	IKQVDTPE			
IKQVDU	IKQVDU			
IKQVDUMP	IKQDUMP IKQDUMPC			
IKQVEDA	IKQVEDA			
IKQVEDX	IKQEDX			
IKQVEOV	IKQEOV			
IKQVGEN	IKQGEN			
IKQVJIBS	IKQJIBSM			
IKQVLAB	IKQLAB			
IKQVCLRD	IKQCLRDD			
IKQVMSG	IKQOCMSG			
IKQVNEX	IKQNEX			
IKQVOPEN	IKQOPN IKQOPNAB IKQOPNAI IKQOPNCT IKQOPNDO IKQOPNDS IKQOPNHC	IKQOPNNC IKQOPNOV IKQOPNPV IKQOPNRD IKQOPNRP IKQOPNUC IKQOPNUS		
IKQVPBF	IKQPBF00			
IKQVRBA	IKQRBA			

Figure 4.1 VSAM phase-to-module index (part 1 of 2)

Phase name	Module name(s)			
IKQVRM	IKQAIX	IKQERX	IKQKRD	IKQRTV
	IKQBFA00	IKQGCI	IKQLCD	IKQSCN
	IKQBFB00	IKQGNX00	IKQLCN	IKQSFT
	IKQBFC00	IKQGPT	IKQLCP	IKQSPM00
	IKQBFD	IKQINT	IKQLNA	IKQSRG
	IKQBLD	IKQIOA	IKQMDY	IKQSRT
	IKQCAS00	IKQIOB	IKQNCA00	IKQSRU
	IKQCIL	IKQIOC	IKQPFO00	IKQUPD
	IKQCIR	IKQIOD	IKQRCL00	IKQUPG
	IKQCIS00	IKQIXE00	IKQRQA	IKQVFY
	IKQCIU	IKQIXF00	IKQRQB	IKQVSM
	IKQDDR	IKQIXS00	IKQRQC	
	IKQERH	IKQJRN	IKQRRP	
IKQVRT	IKQVRT			
IKQVSCAT	IKQSCAT			
IKQVSHR	IKQOCSHR			
IKQVSTM	IKQSTM			
IKQVTMS	IKQTMSD			
	IKQTMSF			
$SVAVSAM	$SVAVSAM			
$$BACLOS	$$BACLOS			
$$BCLCRA	$$BCLCRA			
$$BCVSAM	$$BCVSAM			
$$BCVS02	$$BCVS02			
$$BCVS03	$$BCVS03			
$$BCVS04	$$BCVS04			
$$BODADE	$$BODADE			
$$BODADS	$$BODADS			
$$BOVSAM	$$BOVSAM			
$$BOVS01	$$BOVS01			
$$BTCLOS	$$BTCLOS			

Figure 4.1 **VSAM phase-to-module index (part 2 of 2)**

IIP Phase-to-Module Index

The core image library contains the ISAM Interface Program phases, identifiable by the first three characters IIP or $$B. Packaged within the phases are the IIP modules. The following list includes the phase names and the names of the modules included within each phase.

Phase name	Module name(s)
$$BOCISC	IIPBMR00
IIPCLOSE	IIPCLS00
IIPOPEN	IIPOPN00
IIPPROC	IIPPRCPR
	IIPPRCMR
IIPAMDTF	IIPAMT00

Figure 4.2 **IIP phase-to-module index**

Component Index

VSAM is logically grouped into components, each of which consists of several modules. This index (Figure 4.3) lists these components in the following order: catalog, control block manipulation, open/close/EOV, DADSM, ISAM interface, and service aids.

Component	Module name	Module function
Catalog	IGG0CLAB	Act as switching station for various catalog routines
	IGG0CLAC[1]	Check whether VSAM master catalog is open
	IGG0CLAD[1]	Open VSAM master catalog
	IGG0CLAE[2]	Open and create VSAM catalog and write self-describing catalog records
	IGG0CLAF	Delete VSAM catalog
	IGG0CLAG	Perform VSAM catalog I/O subfunctions, part 1
	IGG0CLAH	Search VSAM catalog for required entry
	IGG0CLAJ[3]	Build data and index entries and allocate space
	IGG0CLAK[3]	Build data and index entries and construct fields in records
	IGG0CLAL[4]	Perform general Define processing
	IGG0CLAN[4]	Perform Define processing and construct cluster entry
	IGG0CLAP[4]	Perform Define processing and check AMDSBs, volume lists, and space parameters
	IGG0CLAQ[5]	Define VSAM data space
	IGG0CLAR[6]	Initialize for VSAM space suballocation
	IGG0CLAS[2]	Define catalog, allocate physical space, and initialize preliminary records
	IGG0CLAT	Act as Access Method Services request dispatcher to catalog functions
	IGG0CLAU[6]	Suballocate VSAM space
	IGG0CLAV[7]	Modify VSAM catalog fields
	IGG0CLAW[7]	Add new VSAM catalog fields
	IGG0CLAX[7]	Alter VSAM catalog fields
	IGG0CLAY[7-8]	Initialize and scan catalog parameter list
	IGG0CLAZ[8]	Extract VSAM catalog fields
	IGG0CLA6[5]	Define VSAM space
	IGG0CLA7[9]	Delete an entry from catalog and, if a unique data set on more than one volume, mount other volume and delete an entry from catalog for that volume

1 IGG0CLAC and IGG0CLAD are related master catalog open processing modules.
2 IGG0CLAE, IGG0CLAS, and IGG0CLES are related Define (catalog build and open) modules.
3 IGG0CLAJ, IGG0CLAK, and IGG0CLA8 are related Define modules.
4 IGG0CLAL, IGG0CLAN, IGG0CLAP, IGG0CLBX, IGG0CLBY, and IGG0CLEX are related modules commonly known as the Define routine.
5 IGG0CLAQ and IGG0CLA6 are related Define space modules.
6 IGG0CLAR and IGG0CLAU are related space suballocation modules.
7 IGG0CLAV, IGG0CLAY, IGG0CLBA, IGG0CLAW, IGG0CLAX, IGG0CLBW, IGG0CLBT and IGG0CLBS are related modules commonly known as the Modify routine.
8 IGG0CLAZ, IGG0CLAY, IGG0CLBA, and IGG0CLEZ are related modules commonly known as the Extract routine.
9 IGG0CLBG, IGG0CLA7 and IGG0CLCX are related delete catalog entry modules.
10 IGG0CLBD, IGG0CLBE, IGG0CLBN, and IGG0CLCD are related Alter processing modules.
11 IGG0CLFB and IGG0CLFC are related Generate Volume List modules.

Figure 4.3 **Component index (part 1 of 4)**

Component	Module name	Module function
Catalog	IGGOCLA8[3]	Perform Define processing and free storage resources
	IGGOCLBA[7-8]	Test VSAM catalog fields
	IGGOCLBB[9]	Extend VSAM data sets
	IGGOCLBC[9]	Initialize for extending VSAM data sets
	IGGOCLBD[10]	Alter an entry in catalog except when processing volumes
	IGGOCLBE[10]	Alter a volume entry and add data set directory to volume entry
	IGGOCLBF	Release space to catalog
	IGGOCLBG[9]	Delete an entry from catalog
	IGGOCLBH	Define a non-VSAM entry in VSAM catalog
	IGGOCLBL	Delete a VSAM data space, mount volume, process F4 labels, remove data from volume record, and scratch DASD space
	IGGOCLBM	Check authorization of catalog user, prompt terminal, and compare password
	IGGOCLBN[10]	Remove volumes for Alter processing and remove data set directories from volume entry
	IGGOCLBQ	List contents of catalog
	IGGOCLBR	Perform bit manipulation against VSAM space bit map
	IGGOCLBS[7]	Retrieve derived VSAM catalog fields
	IGGOCLBT[7]	Modify derived VSAM catalog fields
	IGGOCLBU	Read and/or write F4 labels
	IGGOCLBW[7]	Modify VSAM catalog by deleting or inserting fields
	IGGOCLBX[4]	Define data set entries and calculate size
	IGGOCLBY[4]	Define data set entries and calculate space
	IGGOCLB8	Back-out Define processing and restore allocated space
	IGGOCLCA	Define alternate index
	IGGOCLCB	Release function
	IGGOCLCD	CMS alter (4th module)
	IGGOCLCG	I/O subroutine (2nd module)
	IGGOCLCL	CMS delete space (2nd module)
	IGGOCLCO	Open catalog recovery area
	IGGOCLCP	Define path
	IGGOCLCR	Define CRA (first module)
	IGGOCLCS	Define CRA (second module)
	IGGOCLCX	CMS delete (3rd module)
	IGGOCLCY	CMS define (6th module)
	IGGOCLC9	Act as general interface and build CCA
	IGGOCLEG	Perform VSAM catalog I/O subfunctions, part 2
	IGGOCLES[2]	Define catalog, build volume occurrences
	IGGOCLET	Check space allocation parameters
	IGGOCLEX[4]	Convert blocks to tracks
	IGGOCLEZ[8]	Translate CKD data into fixed block format
	IGGOCLFA	Generate partition/processor independent name
	IGGOCLFB[11]	Generate a volume list from a default model
	IGGOCLFC[11]	Extract volume information from a default model for use in IGGOCLFB
	IGGOCLFD	Define dedicated VSAM volume
	IGGOCLFE	Define space DEDICATE Load 2
	IGGOCLFF	Define space ORIGIN
	IGGOCLFH	Search VSAM catalog for required entry
	IGGOCLFQ	CMS define space Load 3
	IKQDCN	Define console file
	IKQDNT	Define device name and characteristics table
	IKQSCAT	Display catalog information
	IKQVDTPE	Catalog device type and label area routine
	$$BCLCRA	Mark deleted CRA
CB Manip.	IKQGEN	Build ACB, RPL, or EXLST
	IKQTMSD	Test, modify, or display ACB, RPL, or EXLST (with diagnosis of input)
	IKQTMSF	Test, modify, or display ACB, RPL, or EXLST (without diagnosis of input)

Figure 4.3 **Component index (part 2 of 4)**

Component	Module name	Module function
O/C/EOV	IKQASNMT	Request volume mounting and logical assignment
	IKQBRP	Build VSAM resource pool
	IKQCLCAT	Update permanent data set information in the catalog
	IKQCLIF	Dynamic storage area close interface
	IKQCLO	Disconnect a user's program from a VSAM data set
	IKQCLOCL	Alternate index clean up
	IKQCLOVY	Alternate index evaluation
	IKQCLRDD	Close file disposition processing
	IKQDRP	Delete VSAM resource pool
	IKQEDX	Extend an EDB when the control blocks need to reflect additional space
	IKQEOV	Mount a volume when the required volume is not mounted
	IKQJIBSM	Build and delete JIBs (extent blocks)
	IKQLAB	Read label information area record (DLBL/EXTENT statements)
	IKQMTMSG	Open/close message writer (with operator response)
	IKQNEX	Get a new extent when space is needed
	IKQOCMSG	Open/Close message routine
	IKQOCSHR	Lock resources required to enforce the file's share option
	IKQOPN	Connect a user's program with a VSAM data set
	IKQOPNAB	Build AMDSB control block structure
	IKQOPNAI	Alternate index initialization
	IKQOPNCT	Open a catalog by means of special processing
	IKQOPNDO	Clean up after a failure to open a data set
	IKQOPNDS	Do primary allocation for dynamic data set
	IKQOPNHC	Locate data set information in catalog
	IKQOPNNC	Next cluster
	IKQOPNOV	Build ARDB, EDB, LPMB and call IKQJIBSM for volume and extent processing
	IKQOPNPV	Sort volume entries from catalog
	IKQOPNRD	Reset reusable data set
	IKQOPNRP	Attach data set to resource pool
	IKQOPNUC	User catalog open
	IKQOPNUS	Alternate index upgrade set determination
	IKQRBA	Update the catalog
	IKQSTM	Storage management
	IKQVRT	VSAM Shared Resource Table
	$$BACLOS	Automatic close
	$$BCVSAM	Provide an interface between VSE and VSAM when a data set is closed
	$$BCVS02	Provide common exit processing for VSAM modules
	$$BOVSAM	Provide an interface between VSE and VSAM when a data set is opened
	$$BOVS01	Provide an interface to the VSE message writer to get a volume mounted for open or catalog/DADSM processing
	$$BTCLOS	Provide an interface between VSE and VSAM when a data set is temporarily closed
DADSM	IKQALL00	Create a new F1 VTOC label (and F3 VTOC labels) from the system label area record
	IKQPOP00	Build F1 and any needed F3 VTOC labels from label area record (subfunction or IKQALL00)
	IKQRDS00	Read VTOC records either by key or disk address
	IKQREN00	Rename a specified F1 VTOC label

Figure 4.3 **Component index (part 3 of 4)**

Component	Module Name	Module Function
	IKQSCR00	Remove an F1 label (and any associated F3 labels) from the VTOC
	IKQVTC00	Open or close VTOC
	IKQWDS00	Write VTOC records either by key or disk address
	$$BODADE	Interface to DADSM from the VSE message writer
	$$BODADS	Interface to the VSE message writer from DASDM
ISAM Interface	IIPAMT00	Map a skeleton of the AMDTF table
	IIPBMR00	Issue an error message if a failure occurs when an ISAM program is trying to open or close a VSAM data set
	IIPCLS00	Close a VSAM data set for an ISAM program
	IIPIIP00	Link-edit phase and include statements
	IIPOPN00	Open a VSAM data set for an ISAM program
	IIPPRCMR	Issue error messages and cancel tasks in case an error occurred in IIP; issue a VSAM CLOSE for the data set if an error occurred during function other than Open or Close
	IIPPRCPR	Transform an ISAM request into an equivalent VSAM request
Service Aids	IKQCLEAN	DADSM utility
	IKQDUMP	Dump non-catalog control blocks
	IKQDUMPC	Dump catalog control blocks
	IKQVEDA	Enable and disable VSAM snap dump routine
	$$BCVS03	Load a phase
	$$BCVS04	I/O routine for IKQVEDA
Feature Indicator	IKQFTIND	Feature indicator module
	IKQFT1	Space Management Feature
	IKQFT2	Reserved
	IKQFT3	Reserved

Figure 4.3 **Component index (part 4 of 4)**

Module Directory

The module directory (Figure 4.4) is organized alphabetically by symbolic module name. It lists the descriptive name, the component to which that module belongs, the method of operation diagram and program structure figure numbers in which that module is referenced, and the external entry point(s).

Module name	Descriptive name	Component	Diag.#	Structure Figure 3.x	External entry points
IGG0CLAB	Catalog driver	Catalog	DB, DC	5	IGGPACDV IGGPRPLF IGGPRPLM
IGG0CLAC	Master catalog search	Catalog	DB	5	IGGPMCO
IGG0CLAD	Master catalog open	Catalog	DB, EE	5	IGGPMCO2
IGG0CLAE	Define catalog open and build	Catalog	EE, EH	-	IGGPDCOC IGGPMEBM IGGPDCBO
IGG0CLAF	Delete catalog	Catalog	EH, EO	5	IGGPDELC IGGPEMIO IGGPEMSG IGGPSDSP
IGG0CLAG	Catalog I/O subfunctions	Catalog	DJ, DL, ED, EF	5	IGGPISCI IGGPAOCI IGGPAXCI
IGG0CLAH	Search catalog (2nd module)	Catalog	DB-DC	5	IGGPSCA
IGG0CLAJ	Define and build data and index entries	Catalog	EC-ED,EI	-	IGGPDBDI IGGPDEXD
IGG0CLAK	Complete define of an entry	Catalog	ED	-	IGGPDCMB
IGG0CLAL	CMS define, 1st module	Catalog	EC, EO	5	IGGPDEF IGGPDTIM IGGPDCAV IGGPDDEP IGGPDCDE IGGPDWAI IGGPDSTY
IGG0CLAN	CMS define, 2nd module	Catalog	EC-EF,EI, EM,EP	-	IGGPDSCB IGGPDBSF IGGPDRDA IGGPDCCE IGGPDUND
IGG0CLAP	CMS define, 3rd module	Catalog	EC,EE	-	IGGPDCDA
IGG0CLAQ	Catalog define space	Catalog	ED-EE,EG	5	IGGPDEFS
IGG0CLAR	Suballocate	Catalog	DH, DJ, ED, EE, EH, EK	-	IGGPSALL
IGG0CLAS	VSAM catalog definition processing	Catalog	EC, EE, EH	-	IGGPDEFC IGGPDCRC
IGG0CLAT	CMS driver	Catalog	DB	5	IGGPCDVR
IGG0CLAU	Suballocation	Catalog	DJ	-	IGGPSALS

Figure 4.4 **Module directory (part 1 of 6)**

Module name	Descriptive name	Component	Diag.#	Structure Figure 3.x	External entry points
IGG0CLAV	Modify catalog field	Catalog	DB, DG, DH, DJ, DL, ED-EH, EK, EM EN, EP, ER	3,5	IGGPMOD IGGPUPD IGGPSGOP IGGPDEL2
IGG0CLAW	Add group occurrence (modify)	Catalog	DG, DK, DL	-	IGGPADGO IGGPGNEX IGGPIGOP IGGPPREC
IGG0CLAX	Alter catalog field	Catalog	DG, DL	-	IGGPALT2 IGGPEXPD IGGPSHNK IGGPDGOP IGGPMGO IGGPDGO
IGG0CLAY	Scan CPL	Catalog	DB, DL	3, 5	IGGPSCNC
IGG0CLAZ	Extract catalog field	Catalog	DB, DD DE, DH DJ, EK-EP	5	IGGPEXT IGGPLOC
IGG0CLA6	CMS define space (part 2)	Catalog	EG, EH	-	IGGPCOBT IGGPCRTC IGGPIVER
IGG0CLA7	CMS delete (part 2)	Catalog	EM, EP	-	IGGPVMSC IGGPDUSC IGGPDEMV IGGPDVMV IGGPERAS
IGG0CLA8	Define clean up	Catalog	ED	-	IGGPDFRS
IGG0CLBA	Tests	Catalog	DB, DE, DG, DK DL	5	IGGPTSTS IGGPGVAL IGGPGREC
IGG0CLBB	Update extend	Catalog	DB, DH	-	IGGPUPDE
IGG0CLBC	Update extend initialization	Catalog	DH	-	IGGPINIT IGGPSVOL
IGG0CLBD	Catalog alter processing	Catalog	EK	5	IGGPALT
IGG0CLBE	Alter volume processing	Catalog	EK	-	IGGPALVL IGGPALEC
IGG0CLBF	Subscratch	Catalog	EM, EP	-	IGGPSSCR
IGG0CLBG	Delete	Catalog	EM, EP	5	IGGPDEL IGGPDLDS IGGPDEXA IGGPDEXP IGGPDLXT
IGG0CLBH	Define non-VSAM data set	Catalog	EF	5	IGGPDEFA
IGG0CLBL	Delete space	Catalog	EN	5	IGGPDELS
IGG0CLBM	Check authorization	Catalog	DB, DD, EL	5	IGGPCKAU
IGG0CLBN	Catalog alter, remove volume processing	Catalog	EK	-	IGGPALVR IGGPALSV
IGG0CLBQ	LISTCAT processing	Catalog	EL	5	IGGPLSTC
IGG0CLBR	Suballocate bit mask handler	Catalog	DJ	-	IGGPBMR
IGG0CLBS	Volume entry translation	Catalog	DK	5	IGGPXVAL IGGPXEXT

Figure 4.4 **Module directory (part 2 of 6)**

Module name	Descriptive name	Component	Diag.#	Structure Figure 3.x	External entry points
IGG0CLBT	Modify volume entry translation	Catalog	DL	-	IGGPXMOD IGGPXLT2 IGGPXEL2 IGGPXDGO
IGG0CLBU	Catalog read/write F4 DSCB	Catalog	EG, EH, EN, EO	-	IGGPF4RD IGGPF4WR
IGG0CLBW	Delete/insert (modify)	Catalog	DL	-	IGGPDEIN
IGG0CLBX	CMS define, 4th module	Catalog	EC, ED, EI	-	IGGPDSPC IGGPDALR
IGG0CLBY	CMS define, 5th module	Catalog	ED	-	IGGPDRSP
IGG0CLB8	Define, space recovery	Catalog	-	-	IGGPDFBO IGGPCNBO
IGG0CLCA	Define AIX	Catalog	EI, ER	5	IGGPAIX IGGPPRPW IGGPCKEN IGGPBOUT
IGG0CLCB	Release function	Catalog	ED	3	IGGPRELE IGGPPTBF IGGPCLBF IGGPGTBF
IGG0CLCD	CMS alter, 4th module	Catalog	EI, EP, ER	-	IGGPUPG IGGPALY
IGG0CLCG	VSAM catalog I/O subroutine (2nd load)	Catalog	EE, EH	5	IGGPRBAP IGGPIORA IGGPCHAC IGGPTRPL IGGPXIO
IGG0CLCL	CMS delete space (2nd module)	Catalog	EN	-	IGGPDLSF IGGPDVSC
IGG0CLCO	Open CRA	Catalog	EH, EN	5	IGGPCRAO IGGPCRAP IGGPGLUB IGGPSCAX
IGG0CLCP	Define path	Catalog	EJ	5	IGGPPATH
IGG0CLCR	Define CRA (1st module)	Catalog	EH	-	IGGPCADR IGGPCRBO IGGPGDVC
IGG0CLCS	Define CRA (2nd module)	Catalog	EH	-	IGGPCRVL
IGG0CLCX	CMS delete (3rd module)	Catalog	EM, EN, EP	-	IGGPDELX IGGPDPTH IGGPDELP IGGPDAIX IGGPDELY IGGPDELO IGGPDELU
IGG0CLCY	CMS define (6th module)	Catalog	ED	-	IGGPDCCO IGGPDPBI IGGPDMOP
IGG0CLC9[1]	Catalog first load	Catalog	BD, DB	3, 5	IGG0CLC9
IGG0CLEG	Catalog I/O subfunctions	Catalog	DB, DC, DG-DL, EC, ED, EG, EH, EK-ER	-	IGGPCCCR IGGPGET IGGPPAD IGGPPDE IGGPPDEC IGGPRCCR IGGPPUPC

[1]Refer to *VSE/VSAM VSAM Logic, Volume 2* for additional documentation.

Figure 4.4 Module directory (part 3 of 6)

Module name	Descriptive name	Component	Diag.#	Structure Figure 3.x	External entry points
IGG0CLES	VSAM catalog definition processing	Catalog	EE	-	IGGPDCVO
IGG0CLET	Space allocation	Catalog	-	-	IGGPFLIM IGGPDCHK
IGG0CLEX	Block-to-track translation	Catalog	-	-	IGGPDELN IGGPFCHK IGGPFCON IGGPDSPF IGGPITER IGGPUQRI
IGG0CLEZ	Track-to-block translation	Catalog	DB, DE	-	IGGPLTRN
IGG0CLFA	Generate partition/processor independent name	Catalog	-	-	IGGPPIN
IGG0CLFB	Generate volume list from default model	Catalog	EC	-	IGGPGVL IGGPGVSA
IGG0CLFC	Extract volume information from default model	Catalog	-	-	IGGPFVXT IGGPGVLM
IGG0CLFD	Dedicated VSAM volume label processing (Part 1)	Catalog	-	-	IGGPDVV
IGG0CLFE	Dedicated VSAM volume label processing (Part 2)	Catalog	-	-	IGGPDEXT
IGG0CLFF	ORIGIN for VSAM volume	Catalog	-	-	IGGPDORG
IGG0CLFH[2]	Search catalog (1st module)	Catalog	DB	-	IGGPSCAT
IGG0CLFQ	CMS Define Space (3rd Load)	Catalog	-	-	IGGPASGN IGGPCONV IGGPFLIM IGGPLBLS IGGPSPC
IIPAMT00	AMDTF (control block)	ISAM interface	-	-	IIPAMT00
IIPBMR00	$$B message routine	ISAM interface	CN, CO	4	IIPBMR00
IIPCLS00	Close	ISAM interface	CM, CO	4	IIPCLS00
IIPIIP00	Phase and include statements	ISAM interface	-	-	
IIPOPN00	Open	ISAM interface	CB, CN	4	IIPOPN00
IIPPRCMR	Processor (messages)	ISAM interface	CN-CP	4	IIPPRCMR
IIPPRCPR	Processor (request translator)	ISAM interface	CC-CL, CN, CP	4	IIPPRCPR
IKQALL00	Allocate data spaces	DADSM	EG, FB	6	IKQALL00
IKQASNMT	Mount volume and assign a logical unit	O/C/EOV	BB, EG EM-EP, FE,GA	3, 7	IKQASNMT
IKQBFA00[1]	Buffer manager	Rec. Mgmt.	GA	7	IKQBFA00
IKQBFB00[1]	LSR buffer manager	Rec. Mgmt.	-	7	IKQBFB00
IKQBRP	Build resource pool	O/C/EOV	AH	2	IKQBRP
IKQCIS00[1]	Control interval split	Rec. Mgmt.	GA	7	IKQCIS00
IKQCLCAT[1]	Close catalog interface function	O/C/EOV	GA	7	IKQCLCAT
IKQCLEAN	VTOC maintenance utility	Serv. aids	-	-	IKQCLEAN
IKQCLIF	DSA close interface function	O/C/EOV	-	-	IKQCLIF

[1]Refer to *VSE/VSAM VSAM Logic, Volume 2* for additional documentation.

[2]IGG0CLFH and IGG0CLAH are related catalog search modules.

Figure 4.4 Module directory (part 4 of 6)

Module name	Descriptive name	Component	Diag.#	Structure Figure 3.x	External entry points
IKQCLNLK	Phase and include statement	VSAM incl.	-	-	IKQCLEAN
IKQCLO	Close function	O/C/EOV	GA	7	IKQCLO00
IKQCLOCL	AIX clean up	O/C/EOV	GA	7	IKQCLOCL
IKQCLOVY	AIX evaluation	O/C/EOV	GA	7	IKQVCLOV
IKQCLRDD	Close disposition processing	O/C/EOV	GA	7	IKQVCLRD
IKQDCN	DTF console file	Catalog	DD	5	IKQDCN
IKQDNT	Device name table	Catalog	EF	-	IKQDNT
IKQDRP	Delete resource pool	O/C/EOV	AI	2	IKQDRP
IKQDUMP	Block dump	Serv. aids	-	-	IKQDUMP IKQDUMPP
IKQDUMPC	Dump catalog control blocks	Serv. aids	-	-	IKQDUMPC
IKQEDX[1]	EDB extend	O/C/EOV	-	-	IKQEDX00
IKQEOV[1]	Mount volume	O/C/EOV	-	-	IKQEOV00
IKQERH[1]	Error handler	Rec. Mgmt.	-	7	IKQERH
IKQERX[1]	VSAM error exit	Rec. Mgmt.	-	7	IKQERX
IKQFTIND	Feature indicator	Feature	-	-	-
IKQFT1	Space Management Feature	Feature	-	-	-
IKQFT2	Reserved	Feature	-	-	-
IKQFT3	Reserved	Feature	-	-	-
IKQGEN	GENCB: Build a new control block	CB Manip.	AC	2	IKQGEN00
IKQJIBSM[1]	Build and delete extent blocks (JIBs)	O/C/EOV	BA, BB, BD, GA	3, 7	IKQJIBSM
IKQLAB	Look at label cylinder	O/C/EOV	BA, ES, FB	3, 6	IKQLAB
IKQMTMSG	O/C message writer with operator reply	O/C/EOV	BB	3	IKQMTMSG
IKQNEX[1]	Get new extent	O/C/EOV	-	-	IKQNEX00
IKQOCIMR[3]	VSE/VSAM Space Management for Sam Feature	-	-	-	IKQOCIMR
IKQOCMSG[1]	Open/Close message routine	O/C/EOV		3, 7	IKQOCMSG
IKQOCSHR	File sharing control	O/C/EOV	BA, BD, GA	3, 7	IKQOCSHR
IKQOPN	Open function	O/C/EOV	BA, BB	3	IKQOPN
IKQOPNAB	Build AMDSB	O/C/EOV	BB	3	IKQOPNAB
IKQOPNAI	Alternate index initialization	O/C/EOV	BA	3	IKQOPNAI
IKQOPNCT	Open catalog	O/C/EOV	BA	3	IKQOPNCT
IKQOPNDO	Clean up after open failure	O/C/EOV	BA, BD	3	IKQOPNDO
IKQOPNDS	Do primary allocation for dynamic data set	O/C/EOV	BA	3	IKQOPNDS
IKQOPNHC	Locate data set information in catalog	O/C/EOV	BA	3	IKQOPNHC
IKQOPNNC	Next cluster	O/C/EOV	BA, BC	3	IKQOPNNC
IKQOPNOV	Open volume extent	O/C/EOV	BB	3	IKQOPNOV
IKQOPNPV	Sort volume entries	O/C/EOV	-	3	IKQOPNPV
IKQOPNRD	Reset reusable data set	O/C/EOV	BA	3	IKQOPNRD
IKQOPNRP	Attach data set to resource pool	O/C/EOV	BB	3	IKQOPNRP
IKQOPNUC	User catalog	O/C/EOV	BA	3	IKQOPNUC
IKQOPNUS	Alternate index upgrade set determination	O/C/EOV	BA	3	IKQOPNUS
IKQOPNVC[3]	VSE/VSAM Space Management for SAM Feature	-	-	-	IKQOPNVC
IKQPFO00[1]	Format data CA or index CNV	Rec. Mgmt.	GA	7	IKQPFO00
IKQPOP00	Build VTOC labels	DADSM	FB, FE	6	IKQPOP00
IKQRBA[1]	Update catalog for sharing	O/C/EOV	-	-	IKQRBA00

1 Refer to *VSE/VSAM VSAM Logic, Volume 2* for additional documentation.

2 IGG0CLFH and IGG0CLAH are related catalog search modules.

3 Refer to *VSE/VSAM Space Management for SAM Logic* for additional documentation.

Figure 4.4 Module directory (part 5 of 6)

Module name	Descriptive name	Component	Diag.#	Structure Figure 3.x	External entry points
IKQRCLOO[1]	Record management close	Rec. Mgmt.	GA	7	IKQRCLOO
IKQRDSOO	Read VTOC labels	DADSM	EG, EN, FF	-	IKQRDSOO
IKQRENOO	Rename data set	DADSM	EK, FC	-	IKQRENOO
IKQRQA[1]	Request analyzer 1	Rec. Mgmt.	-	7	IKQRQA
IKQRQB[1]	Request analyzer 2	Rec. Mgmt.	-	7	IKQRQB
IKQRQC[1]	Request analyzer 3	Rec. Mgmt.	-	-	IKQRQC
IKQSCAT	SHOWCAT: Display catalog information	Catalog	ES	-	IKQSCAT
IKQSCROO	Scratch VTOC labels	DADSM	EM-EP, FD, FE	6	IKQSCROO
IKQSFT[1]	Shift	Rec. Mgmt.	-	7	IKQSFT
IKQSMACL[3]	VSE/VSAM Space Management for SAM Feature	-	-	-	IKQSMACL
IKQSTM	Storage manager	O/C/EOV	BD	3, 7	IKQSTM
IKQTMSD	MODCB, SHOWCB, TESTCB: Modify, display, or test a control block	CB. Manip.	AD-AG	2	IKQTMSD
IKQTMSF	MODCB, SHOWCB, TESTCB: Modify, display, or test a control block	CB. Manip.	AD-AG	2	IKQTMSF
IKQVDTPE	Device type routine	Catalog	ED, EE, EG, EN, EO	3	IKQVDTPE
IKQVEDA	Enable and disable VSAM diagnostic aids	Serv. Aids	-	-	IKQVEDA
IKQVSMLK	Phase and include statements	VSAM	-	-	-
IKQVSM[1]	VSAM request driver	Rec. Mgmt.	GA	-	IKQVSM
IKQVTCOO	Open/Close VTOC	DADSM	EG, EN, FB-FD, FH	6	IKQVTCOO
IKQWDSOO	Write VTOC labels	DADSM	FE, FG	6	IKQWDSOO
$SVAVSAM	SVA module list	VSAM	-	-	$SVAVSAM
$$BACLOS	Automatic close	O/C/EOV	GA	7	$$BACLOS
$$BCLCRA	Mark deleted CRA	Catalog	EN	-	$$BCLCRA
$$BCVSAM	Close interface	O/C/EOV	GA	7	$$BCVSAM
$$BCVS02	Common exit	O/C/EOV	-	3, 7	$$BCVS02
$$BCVS03	LKMOD routine	Serv. aids	-	-	$$BCVS03
$$BCVS04	I/O routine for IKQVEDA	Serv, aids	-	-	$$BCVS04
$$BODADE[1]	End of message interface	DADSM	-	3, 6	$$BODADE
$$BODADS	Start of message interface	DADSM	FE	6	$$BODADS
$$BOVSAM[1]	Open interface	O/C/EOV	BA	3	$$BOVSAM
$$BOVS01	Catalog/DADSM interface to mount volume	O/C/EOV	FE	3, 6,8	$$BOVS01
$$BTCLOS	TCLOSE interface	O/C/EOV	GA	7	$$BTCLOS

1 Refer to *VSE/VSAM VSAM Logic, Volume 2* for additional documentation.

3 Refer to *VSE/VSAM Space Management for SAM Logic* for additional documentation.

Figure 4.4 **Module directory (part 6 of 6)**

Routine Directory

Some of the VSAM modules contain several routines which are listed alphabetically by the entry points along with the appropriate module. Figure 4.5 contains catalog management modules.

Entry point	Module name	Procedure description
IGGPACDV	IGG0CLAB	Catalog management driver
IGGPADGO	IGG0CLAW	Add group occurrence
IGGPAIX	IGG0CLCA	Define AIX
IGGPALEC	IGG0CLBE	Check for index or data and sequence set with data
IGGPALSV	IGG0CLBN	Rename DSCBs for alter data set name function
IGGPALT	IGG0CLBD	ALTER processing
IGGPALT2	IGG0CLAX	ALTER catalog record's field value
IGGPALVL	IGG0CLBE	ALTER: Volume processing
IGGPALVR	IGG0CLBN	ALTER: Remove volume processing
IGGPALY	IGG0CLCD	Get and initialize work area for upgrading/no-upgrading routine
IGGPAOCI	IGG0CLAG	Assign contiguous control intervals
IGGPASGN	IGG0CLFQ	Assign symbolic unit to volume
IGGPAXCI	IGG0CLAG	Assign one control interval
IGGPBMR	IGG0CLBR	Suballocate bit mask handler
IGGPBOUT	IGG0CLCA	Backout AIX or path association group occurrence
IGGPCADR	IGG0CLCR	Build CRA
IGGPCCCR	IGG0CLEG	Checkpoint the catalog control record (CCR)
IGGPCDVR	IGG0CLAT	Catalog management services common processing
IGGPCHAC	IGG0CLCG	Compute RBAs for next extent
IGGPCKAU	IGG0CLBM	Check the caller's authorization to access the catalog record
IGGPCKEN	IGG0CLCA	Check entry name and entry name of related object
IGGPCLBF	IGG0CLCB	Clear buffer
IGGPCNBO	IGG0CLB8	Remove candidate volume occurrences
IGGPCOBT	IGG0CLA6	Obtain VTOC entries
IGGPCONV	IGG0CLFQ	Convert blocks to tracks
IGGPCRAO	IGG0CLCO	Open catalog recovery area (CRA)
IGGPCRAP	IGG0CLCO	CRA password checking
IGGPCRBO	IGG0CLCR	Back out volume record and reset TT-pointer
IGGPCRTC	IGG0CLA6	Convert records to tracks
IGGPCRVL	IGG0CLCS	Suballocate CRA and write initial CRA records
IGGPDAIX	IGG0CLCX	Implicit delete AIX (for DEL cluster request)
IGGPDALR	IGG0CLBX	Average logical record size FPL structure processing
IGGPDBDI	IGG0CLAJ	DEFINE: Build the data set and index catalog records of a cluster
IGGPDBSF	IGG0CLAN	Buffer size FPL structure processing
IGGPDCAV	IGG0CLAL	Cross check and validity check
IGGPDCBO	IGG0CLAE	Define space backout
IGGPDCCE	IGG0CLAN	DEFINE: Build the cluster's catalog record
IGGPDCCO	IGG0CLCY	Determination of data and index characteristics
IGGPDCDA	IGG0CLAP	DEFINE Catalog processing (2 of 2)
IGGPDCDE	IGG0CLAL	Date and entry name processing
IGGPDCHK	IGG0CLET	Input validity check
IGGPDCMB	IGG0CLAK	DEFINE: Completion (build the volume information group occurrence)
IGGPDCOC	IGG0CLAE	DEFINE Catalog: Catalog open, build, and close

Figure 4.5 **External entry points of catalog management modules (part 1 of 4)**

Entry point	Module name	Procedure description
IGGPDCRC	IGG0CLAS	Compute RBAs of data space
IGGPDCVO	IGG0CLES	Build volume group occurrences
IGGPDDEP	IGG0CLAL	Date and entry name processing
IGGPDEF	IGG0CLAL	DEFINE common processing
IGGPDEFA	IGG0CLBH	DEFINE non-VSAM processing
IGGPDEFC	IGG0CLAS	DEFINE Catalog processing (1 of 2)
IGGPDEFS	IGG0CLAQ	DEFINE Space processing
IGGPDEIN	IGG0CLBW	Modify: Delete/insert processing
IGGPDEL	IGG0CLBG	DELETE Cluster/non-VSAM processing
IGGPDELC	IGG0CLAF	DELETE Catalog processing
IGGPDELN	IGG0CLEX	Check if catalog is open
IGGPDELO	IGG0CLCX	Check if cluster or AIX data set is open
IGGPDELP	IGG0CLCX	Delete path driver
IGGPDELS	IGG0CLBL	DELETE Space processing
IGGPDELU	IGG0CLCX	Unlock cluster or AIX
IGGPDELX	IGG0CLCX	Delete AIX driver
IGGPDELY	IGG0CLCX	Delete and upgrade set (for DEL cluster request)
IGGPDEL2	IGG0CLAV	Delete a group occurrence
IGGPDEMV	IGG0CLA7	DELETE: Extract the volume information group occurrence
IGGPDEXA	IGG0CLBG	Build interface to extract cluster, AIX, or path associations
IGGPDEXD	IGG0CLAJ	Delete work area
IGGPDEXP	IGG0CLBG	Extract the password of a cluster or an AIX
IGGPDEXT	IGG0CLFE	DEDICATE processing
IGGPDFBO	IGG0CLB8	DEFINE: Space recovery
IGGPDFRS	IGG0CLA8	Free unused and unneeded storage resources
IGGPDFS2	IGG0CLA6	DEFINE Space: Build the space header, space descriptor group, and data set directory entry group occurrences
IGGPDGO	IGG0CLAX	MODIFY: Delete group occurrence processing
IGGPDGOP	IGG0CLAX	MODIFY: Delete group occurrence pointer processing
IGGPDLDS	IGG0CLBG	Delete the space of the cluster or AIX
IGGPDLSF	IGG0CLCL	Forced delete space
IGGPDLXT	IGG0CLBG	Delete clean up routine (exit)
IGGPDMOP	IGG0CLCY	Complete entry construction process
IGGPDORG	IGG0CLFF	ORIGIN processing
IGGPDPBI	IGG0CLCY	Determine physical block size index value
IGGPDPTH	IGG0CLCX	Implicit delete path (for DEL cluster or DEL AIX request)
IGGPDRDA	IGG0CLAN	Regular define AMDSB processing
IGGPDRSP	IGG0CLBY	DEFINE Cluster processing (5th module)
IGGPDSCB	IGG0CLAN	DEFINE common processing (space calculations and build the cluster catalog record)
IGGPDSPC	IGG0CLBX	DEFINE Cluster processing (4th module)
IGGPDSPF	IGG0CLEX	Space parameter FPL structure processing
IGGPDSTY	IGG0CLAL	Security FPL structure checking
IGGPDTIM	IGG0CLAL	DEFINE: Call the system timer
IGGPDUND	IGG0CLAN	DEFINE: Undo the previous processing
IGGPDUSC	IGG0CLA7	DELETE: Scratch the data space (format-1 label) from the volume's VTOC
IGGPDVMV	IGG0CLA7	DELETE: Mount and verify volumes
IGGPDVSC	IGG0CLCL	Clean VTOC from VSAM spaces
IGGPDVV	IGG0CLFD	DEFINE DEDICATE/ORIGIN VSAM volume label processing
IGGPDWAI	IGG0CLAL	Work area initialization
IGGPEMIO	IGG0CLAF	I/O error message writer
IGGPEMSG	IGG0CLAF	Error message writer
IGGPERAS	IGG0CLA7	DELETE: Erase data component
IGGPEXPD	IGG0CLAX	Expand a catalog record's variable-length field
IGGPEXT	IGG0CLAZ	Extract processing
IGGPFCHK	IGG0CLEX	Check FVT space parameter consistency
IGGPFCON	IGG0CLEX	Convert blocks to tracks in all FVTs
IGGPFLIM	IGG0CLFQ	Convert blocks to tracks or cylinders
IGGPFVXT	IGG0CLFC	FREEVIS of GVL storage
IGGPF4PR	IGG0CLA6	Read format-4 label and either set or reset time stamps, CRA pointer and ownership

Figure 4.5 **External entry points of catalog management modules (part 2 of 4)**

Entry point	Module name	Procedure description
IGGPF4RD	IGG0CLBU	Read the format-4 label
IGGPF4WR	IGG0CLBU	Write the format-4 label
IGGPGDVC	IGG0CLCR	Get device characteristics for define CRA
IGGPGET	IGG0CLEG	Get catalog record: Call record management to retrieve a catalog record
IGGPGLUB	IGG0CLCO	Get LUB index associated with CRA volume
IGGPGNEX	IGG0CLAW	Get an available RAB and format a new catalog extension record
IGGPGREC	IGG0CLBA	Retrieve a catalog record
IGGPGTBF	IGG0CLCB	Get buffer
IGGPGVAL	IGG0CLBA	Locate a catalog record field
IGGPGVL	IGG0CLFB	Generate volume list
IGGPGVLM	IGG0CLFC	Extract volume information from the default model
IGGPGVSA	IGG0CLFB	Build volume list after sub-allocation failed
IGGPIDMP	IGG0CLAT	Issue an IDUMP
IGGPIGOP	IGG0CLAW	Insert a group occurrence pointer
IGGPINIT	IGG0CLBC	Update-Extend: Initialization
IGGPIORA	IGG0CLCG	Error code analyzer
IGGPISCI	IGG0CLAG	Insure control-interval availability
IGGPITER	IGG0CLEX	Adjust data CI size
IGGPIVER	IGG0CLA6	Initialize volume entry record
IGGPLBLS	IGG0CLFQ	GETVIS space for dummy label
IGGPLOC	IGG0CLAZ	Locate processing
IGGPLSTC	IGG0CLBQ	LISTCAT processing
IGGPLTRN	IGG0CLEZ	Translate CKD field into fixed block format
IGGPMCO	IGG0CLAC	DEFINE Catalog: Master catalog build and open (1 of 2)
IGGPMCO2	IGG0CLAD	DEFINE Catalog: Master catalog build and open (2 of 2)
IGGPMEBM	IGG0CLAE	Handle multiple extents for catalog open and build
IGGPMGO	IGG0CLAX	Move group occurrence from one extension record into another
IGGPMOD	IGG0CLAV	Modify common processing
IGGPPAD	IGG0CLEG	PUT-Add: Call record management to write a new catalog record
IGGPPATH	IGG0CLCP	Define path
IGGPPDE	IGG0CLEG	ERASE: Call record management to erase a catalog record
IGGPPDEC	IGG0CLAG	Delete catalog record
IGGPPIN	IGG0CLFA	Create partition/processor independent name
IGGPPREC	IGG0CLAW	Call PUT-Add or PUT-Update to write a catalog record
IGGPPRPW	IGG0CLCA	Check password of related object
IGGPPTBF	IGG0CLCB	Put into buffer
IGGPPUPC	IGG0CLEG	PUT-Update: Call record management to rewrite a catalog record
IGGPRBAP	IGG0CLCG	Scan the ARDBs and AMDSBs for all RBAs
IGGPRCCR	IGG0CLEG	Read catalog control record
IGGPRELE	IGG0CLCB	Release function
IGGPRPLF	IGG0CLAB	Dequeue the catalog
IGGPRPLM	IGG0CLAB	Assign RPLs from the catalog RPL pool
IGGPSALL	IGG0CLAR	Suballocate: Candidate volume assignment
IGGPSALS	IGG0CLAU	Suballocate: Space assignment
IGGPSCA	IGG0CLAH	Set catalog ACB address/open user catalog
IGGPSCAT	IGG0CLFH	Search catalog processing
IGGPSCAX	IGG0CLCO	Scan CAXWA chain
IGGPSCNC	IGG0CLAY	Initial CTGPL processing
IGGPSDSP	IGG0CLAF	Remove space and close CRA

Figure 4.5 **External entry points of catalog management modules (part 3 of 4)**

Entry point	Module name	Procedure description
IGGPSGOP	IGG0CLAV	Retrieve the group occurrence pointer
IGGPSHNK	IGG0CLAX	Shrink a catalog record's variable-length field
IGGPSPC	IGG0CLFQ	Convert track allocation for DEDICATE and ORIGIN
IGGPSSCR	IGG0CLBF	Subscratch: Release a cluster's space within a VSAM data space
IGGPSVOL	IGG0CLBC	Search for the volume information group occurrence
IGGPTRPL	IGG0CLCG	Test RPL last used
IGGPTSTS	IGG0CLBA	CTGFL-for-tests processing
IGGPUPD	IGG0CLAV	Update catalog field
IGGPUPDE	IGG0CLBB	Update-Extend processing
IGGPUPG	IGG0CLCD	Add an AIX to the upgrade set (UPGRADE) or delete an AIX from the upgrade set (NOUPGRADE)
IGGPUQRI	IGG0CLEX	Build space return information for unique data set
IGGPVMSC	IGG0CLA7	DELETE: Delete all space information in the volume catalog record
IGGPVMTV	IGG0CLA6	Volume mount
IGGPXDGO	IGG0CLBT	Add derived group occurrence
IGGPXEL2	IGG0CLBT	Delete derived group occurrence
IGGPXEXT	IGG0CLBS	Extract derived group occurrence
IGGPXIO	IGG0CLCG	I/O routine for catalog and CRA
IGGPXLT2	IGG0CLBT	Alter derived field value
IGGPXMOD	IGG0CLBT	Modify derived group occurrence
IGGPXVAL	IGG0CLBS	Get derived field value
IGG0CLC9	IGG0CLC9	Catalog management first load

Figure 4.5 **External entry points of catalog management modules (part 4 of 4)**

Control Block Directory

The control block directory (Figure 4.6) contains a short entry for each of the most important VSAM control blocks for the VSAM components documented in this volume, giving the length and the purpose of each block.

Data Area	Total size	Purpose
ACB	68 bytes	To describe a VSAM cluster
AMBL	112 bytes	To connect an ACB to the PLH and AMDSB(s)
AMCBS	28 bytes	To contain addresses of CAXWA chain, master and job catalog ACBs, and recovery information
AMDSB	200 bytes	To record data set status and statistics (not including buffer header and first EDB)
AMDTF	469 bytes	To contain save areas, lists, addresses for ISAM interface programs, and the error message build area
CAXWA	156 bytes	To contain pointers to control blocks and work areas needed when a catalog is being processed
CCA	1456 bytes	To contain information about the catalog being processed and about the catalog record and its extensions
CTGFL*(also known as FPL or FL)	24 bytes + field pointers	To contain catalog field name, address, and length
CTGFV*(also known as FVT)	92 bytes	To contain addresses of user-supplied information fields and lists
CTGPL*(also known as CPL)	40 bytes + field pointers	To contain a description of the call for catalog management services
DASDM parameter list	168 bytes	To contain the input parameters for the DASD Space Management routines
DTFIS	variable depending on function performed	To describe an ISAM file
EDB	52 bytes	To contain the extent descriptions
OAL	24 bytes + 8 × number of open ACBs (max. 14)	To contain all opened VSAM ACBs
OPNWA	2172 bytes	To contain information needed when a data set is being opened
RPL	52 bytes	To contain user request information and error feedback information
VRPPL	20 bytes	To contain the input parameters needed for building the VSAM Resource Pool
VSRT	76 bytes	To contain information and pointers for the resource pool

* Rebuilt for each use, that is, not permanent.

Figure 4.6 **Control block directory**

Section 5: Data Areas

This section deals with the internal data areas of VSAM, describing their formats, functions, and interrelationships. It is assumed that you are familiar with the basic structure of VSAM, such as the types of data sets, the structure of indexes, the concept of the catalog, etc., as these are described in *Using VSE/VSAM Commands and Macros*.

The section is divided into two parts:

- Descriptions of the VSAM data set, index, alternate index, and catalog.
- Description and format of the VSAM control blocks, together with figures showing their interrelationships.

VSAM control blocks that do not appear in this volume are documented in *VSE/VSAM VSAM Logic, Volume 2*.

VSAM Data Set

A VSAM data set is a collection of records grouped into control intervals. Control intervals are grouped into larger units called control areas. If the VSAM data set is key-sequenced, then the control interval(s) in which it resides are pointed to by entries in an associated index. The VSAM stored record, control interval, control area, and index are described in the topics that follow.

VSAM Record

Records are normally treated by VSAM as variable-length records. Records can be spanned across control intervals within a control area, and their maximum size is thus equal to the length of a control area. The only exception to this is a relative-record data set, whose records must have a fixed length.

Control Interval

A control interval is a continuous area of auxiliary storage that VSAM uses for storing records. The control interval is the unit of information that VSAM transfers between virtual and auxiliary storage.

The length of each control interval is an integral multiple of block size. The size of a control interval is determined by the system from the size of the records, user-specified minimum buffer size, device characteristics, and the user-specified percentage of free space. You can specify the size of the control interval, but it must be within limits acceptable to VSAM. Control interval length must be in the range 512 to 32,768. If the length is between 512 and 8,192, the value must be a multiple of 512; if the length is between 8,193 and 32,768, the value must be a multiple of 2,048.

Data records are put in the low-address portion of the control interval. Control information about each data record is put in the high-address portion of the control interval. The combination of a data record and its control information, though they are not physically adjacent, is called a stored record. The control information in a control interval consists of a Control Interval Definition Field and one or more Record Definition Fields. Figure 5.1 shows the format of a control interval.

Record 1	Record 2	Record 3	Free Space	RDF 3	RDF 2	RDF 1	CIDF

Figure 5.1 Control interval format

Control Interval Definition Field

The Control Interval Definition Field (CIDF) describes the control interval. Its format is shown in Figure 5.2.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	2	CIDFDD	Free space offset (binary) Displacement from the beginning of the control interval to the beginning of the free space[1]
2	2	2	CIDFLL	Free space length (binary) Length of the free space area within this control interval[1]
2	2	1...	CIDFDDP	The process of moving records from this control interval to another has not completed; duplicate records may exist.

[1] If the CIDF contains only 0s, *end-of-data-set* or *end-of-key-range* is indicated; either the end of the data set was detected or the end of a key range in a key-sequenced data set was detected when the data set was to be divided between volumes. Information in the volume group occurrence (see VOLFLG) in the data set's catalog record helps to differentiate between the end-of-data-set and end-of-key-range conditions.

Figure 5.2 Control interval definition field format

Record Definition Field

The Record Definition Fields (RDFs) describe the records in the control interval. They are inserted into the control interval from right to left, which means that the rightmost RDF describes the leftmost data record.

There is normally one RDF for each record, except in two special cases. These are:

- When two or more consecutive records in the control interval have the same length. In this case, two RDFs are used to describe the whole group of records. The first (right-hand) RDF describes the characteristics of the records, and the second (left-hand) RDF contains a count of the number of records.

  Note that this is true only for key-sequenced and entry-sequenced data sets. The slots or records in a relative record data set have a fixed length, but specific information is required for each one. The records cannot, therefore, be grouped, and one RDF is required for each record.

- When the record is spanned. In this case, only one segment of one record can be located in the control interval. Nevertheless, two RDFs are used. The first (right-hand) RDF describes the record segment, and the second (left-hand) RDF contains a "level number", which is used for data integrity checking. This number is assigned and updated by VSAM whenever the spanned record is processed. The level number in all segments of a spanned record will always be the same, unless an error has occurred.

The format of an RDF is shown in Figure 5.3.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	1	RDFFLAG	Flag byte
			RDFEXT	RDF extension flag
		.0..		There is no RDF to the left of this RDF that contains additional information about record(s) described in this RDF.
		.1..		There is an RDF to the left of this RDF that contains additional information about the record(s) described in this RDF. Byte two and three of the RDF to the left contain the following information: • if there are more consecutive records than one of fixed length they contain the number of these records beginning with the record associated with the previous (to the right) RDF (see replication count flag) • in the case of spanned records they contain the level number
		..00		This is the only segment of a stored record.
		..11	RDFSRM	The RDF to the left contains information about spanned records (middle segment)
		..10	RDFSRL	The same as above but last segment
		..01	RDFSRF	The same as above but first segment
			RDFREPL	Replication count flag
		.... 0...		The second and the third bytes of this RDF contain the data record's length
		.... 1...		This RDF contains additional information about the record(s) described in the RDF to the right.
		.... x...	RDFRESL	Empty slot indicator (for relative record processing where one RDF is associated with one slot in the control interval - no extended RDFs)
		.... .1..		The record in the corresponding slot is invalid (it has been deleted or not yet inserted)
		.... .0..		The record in the corresponding slot is valid
				Depending on the kind of record(s) described, byte two and three of an RDF contain one of the following values:

Figure 5.3 Record definition field format (part 1 of 2)

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
1	1	2	RDFLL	Length field Always present in the rightmost (or only) RDF for a record or group of fixed-length records. These bytes contain the data record's length or the length of the segment of a spanned record.
1	1	2	RDFCOUNT	Count field These bytes contain the number of consecutive fixed-length records. It is a type of RDF that contains additional information about the records described in the RDF to the right. Byte 0 (of the RDF to the right) Bit 4 = 1, Bits 2, 3 = 0
1	1	2	RDFSRLVL	Level number These bytes contain the level number for spanned records. It is a type of RDF that contains additional information about the records described in the RDF to the right. Byte 0 (of the RDF to the right) Bit 4 = 1, Bits 2,3 = 11 or 10 or 01

Figure 5.3 **Record definition field format (part 2 of 2)**

Control Area

A control area consists of control intervals; the number of control intervals in a control area is determined by VSAM. The control area is the amount of space that VSAM preformats so that data integrity is ensured for records added to a data set.

Control areas are also used to simplify and localize the movement of records when records are inserted in a key-sequenced data set. If an insertion requires a free control interval and there isn't one, a control-area split results. VSAM establishes a new control area and moves the contents of approximately half of the full control area to free control intervals in the new control area. The new records, as their keys dictate, are then inserted into one of the two control areas. The control area has no specific control information.

Index

An index is created at the same time as a key-sequenced data set. The index structure exists in its own address space and consists of one or more levels. The lowest level or *sequence set* consists of one or more index records. There is an index record in the sequence set for each formatted control area. Within a sequence-set record there is either an index entry or a free data control interval pointer for each control interval in the control area. (Free data control interval pointers are discussed later in this section.) The key in each entry of a sequence set record is the same as the key of the last (highest) entry in the corresponding control interval. To save space, VSAM compresses the keys in the index.

The upper levels of the index are collectively called the index set, and contain index entries which point to the next lower level of the index. Figure 5.4 shows a simple index structure.

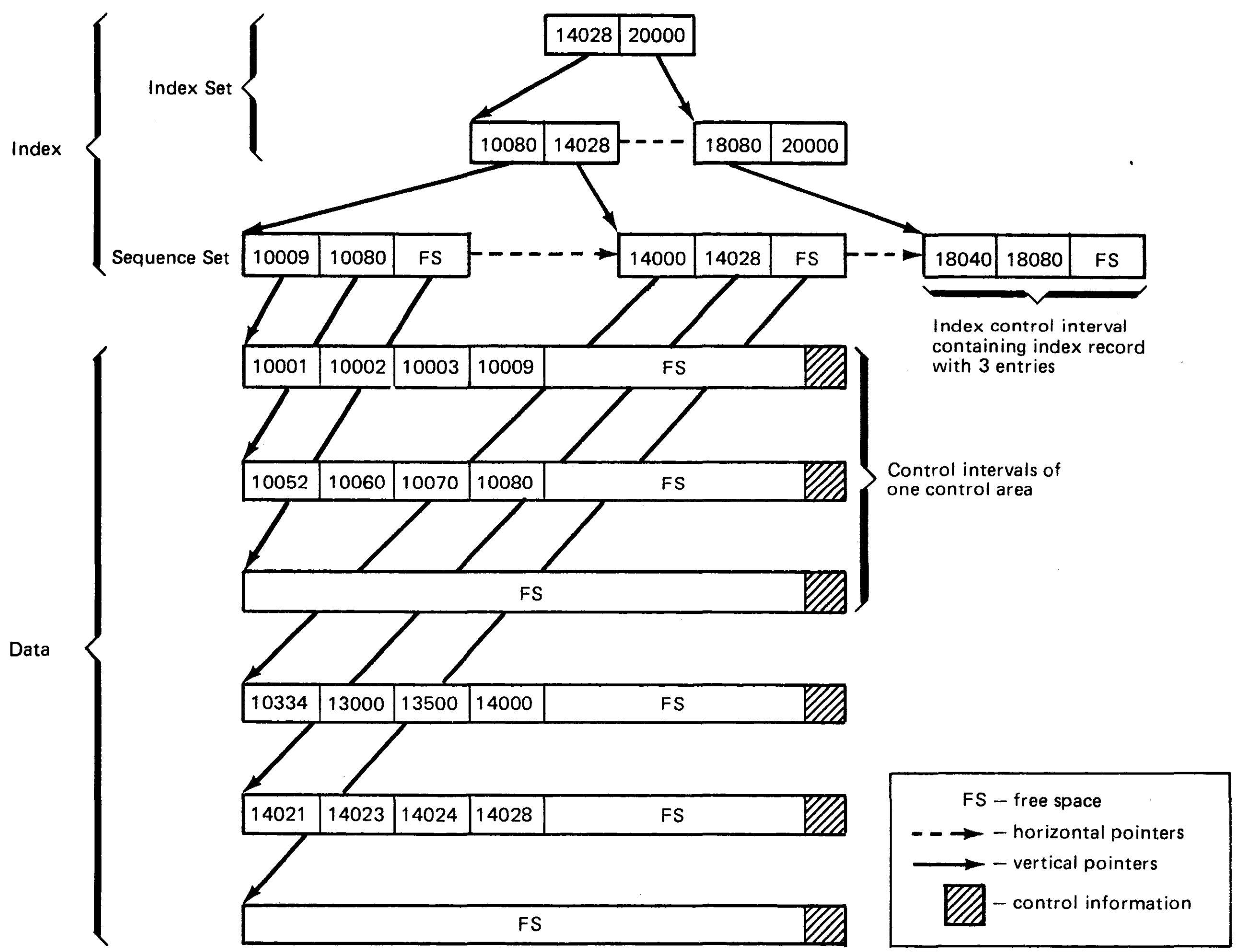


Figure 5.4 **Example of a simple VSAM index**

Index Record

The index records and control intervals are fully compatible with VSAM data records and control intervals, and are handled by record-management modules in the same way. The only differences between index records and data records are:

- There is only a single index record in an index control interval (and thus only one RDF).
- The internal format of an index record is fixed. This format is shown in the example of Figure 5.5, and its various parts are discussed.

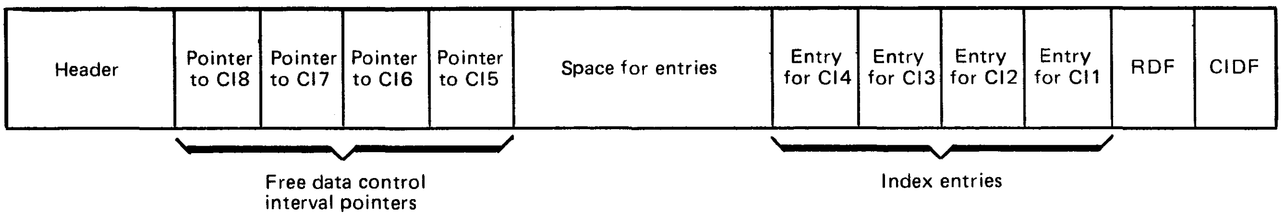


Figure 5.5 **Example of an index control interval**

Index Record Header

The index record header contains the information needed to insert index entries, to locate entries within the record, and to convert pointers into RBAs. The format of the index record header is shown in Figure 5.6.

Offset Dec	Offset Hex	Field Name	Field Size	Description
0	0	IXRL	2	Length in bytes, of the index record, including this field.
2	2	IXCINL	1	Length, in bytes, of the control information (the IXENTRYF, IXENTRYL, and IXENTRYP fields) in each index entry.
3	3	IXPMASK	1	Length of the pointers to free data control intervals in this index record[1]. This field is used as a mask for insert character (store character) under mask instructions that are used to access pointers. The value contained in this field specifies the length of these pointers, as follows: B'0001' 1-byte pointer B'0011' 2-byte pointer B'0111' 3-byte pointer
4	4	IXBASRBA	4	For a sequence-set index record, the RBA of a data control area that contains data to be referenced. This RBA and index-entry pointers are used together to calculate the 4-byte RBA of another index record or of a data control interval (0 for high-level indexes).
8	8	IXNXTIR	4	Pointer to the logically next index record in this index level. (Horizontal pointer)
12	C		4	Reserved (0).
16	10	IXLVLNO	1	Index level number. A sequence-set index is assigned a value of 1; the next higher-level index is assigned a value of 2; etc.
17	11		1	Reserved (0).
18	12	IXINSOS	2	Displacement from the beginning of this record to the space available for inserting index entries. For higher-level indexes, the entry space immediately follows the record header; for sequence-set indexes, the entry space follows the record header and free data-control-interval pointers.
20	14	IXLENTRY	2	Displacement from the beginning of this record to the last (high-key) entry in the index record.[2]
22	16	IXFSECTN	2	Displacement from the beginning of this record to the first (low key) section entry in the index record.[2]

[1] Pointers are allowed to vary in length to conserve index space. If, for example, the number of items to be referenced by an index record is less than 256, a one-byte pointer can be used; if the number is greater than 256 and less than 65,536, a two-byte pointer can be used; and if the number is greater than 65,536, a three-byte pointer can be used.

[2] This displacement is to the F (front-key compression count) byte of the entry, not to the beginning of the entry.

Figure 5.6 **Index record header format**

Free Data-Control-Interval Pointers

Free data-control-interval pointers, which exist only in sequence-set index records, are used to calculate the RBAs of available data control intervals. The length of a pointer is specified in the record header.

When the index is first built, and before records have been loaded into the data set, the index records of the sequence set contain one free data control interval pointer for each data control interval.

VSAM always uses the rightmost free data-control-interval pointer when a data control interval is needed. The value of the pointer is set to 0 when the control interval is used. As pointers are set to 0, the displacement to space that is available for index entries (contained in the record header) is adjusted by the length of the free data-control-interval pointer. In this way, space used by free data-control-interval pointers is made available for index entries when the pointers are no longer required.

The example in Figure 5.5 shows a sequence set record for a control area with eight control intervals. Of these eight, the first four are now occupied by data, and the last four are still free.

Index Entries

The index entries are the link between the index and the data set. They contain the key, the pointer to the data control interval containing the data record, and information about key compression. The format of an index entry is shown in Figure 5.7.

Field Size (in bytes)	Field Name	Description
Variable	IXKEY	Key characters that determine the sequence of records in a key-sequenced data set.
1	IXENTRYF (F byte)	Front-key compression count, that is, the number of characters by which the beginning of the key has been compressed.
1	IXENTRYL (L byte)	Length of the IXKEY field.
1-3	IXENTRYP (P field)	Pointer to an index or data control interval. This value is the number of the CI within the CA (for example '4' for the fifth CI). To calculate the RBA of the CI, this value must be multiplied by the CI size and added to the contents of IXBASRBA.

Figure 5.7 **Index entry format**

Index Entries for Spanned Records

Since spanned records extend across two or more data control intervals, their index entries, sometimes called 'complex index entries', consist of a series of 'normal' entries (one for each data control interval). These entries, in turn, are basically standard index entries, but they have some special features:

- The key is contained only in the entry for the last segment of the spanned records, whose F byte contains the actual key compression count.
- The entries for all other segments contain no key, and their F byte contains a compression count equal to the key length, thus indicating a key length (in the entry) of zero.
- Each entry contains a pointer to its associated segment (or data control interval).

Index Entry Sections

To save time when searching index records for a given key, the index entries are grouped into sections. This allows a rapid search, scanning only the highest key in each section, to locate the correct section, which is then searched for the correct key.

A section is defined by a two-byte field to the left of the high-key entry in the section. This field contains the displacement from the F byte of the high-key entry in this section to the F byte of the high-key entry in the next section (to the left). The index record header contains a pointer to the F byte in the high-key entry in the first section.

For technical reasons, this division of the index entries into sections is not carried out until a control interval split is necessary in a control area. There will thus be no section definition fields in the index of a freshly loaded data set, and only some of the sequence set records in an 'older' data set will have such fields.

Alternate Index

The alternate index (AIX) provides an alternate means of access, using different keys, to the data records in the base cluster, which can be a key-sequenced or entry-sequenced data set, but not a relative-record data set. The alternate index itself is a key-sequenced data set. The index component of the AIX is identical in structure, format, and function to the index of any other key-sequenced data set. The basic structure of the data component of the AIX is also identical to that of a normal key-sequenced data set, as far as CIDFs and RDFs are concerned.

The only difference in format between the AIX and a normal key-sequenced data set concerns the records in the data component of the AIX, which have a fixed format, shown in Figure 5.8. These records form the logical connection between the AIX and the base cluster, and contain control information, the alternate key, and one or more pointers to the base cluster. If this base cluster is a key-sequenced data set, the pointers consist of the prime keys of the required data records, which are then located by means of the base cluster index. If the base cluster is an entry-sequenced data set, which has no index, the pointers are relative-byte addresses (RBAs) of the required records, which can then be located directly.

As it is possible to have more than one pointer in an AIX record, the length of such a record can vary. In extreme cases, it may be greater than the control interval length, and the record is treated as a spanned record.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	1	AIXFG	Flag byte
		xxxx xxx.		Reserved
		1	AIXPKP	Prime key pointers are used
		0		RBA pointers are used
1	1	1	AIXPL	Pointer length (in binary)
2	2	2	AIXPC	Number of pointers in this record (in binary)
4	4	1	AIXKL	Length of alternate key (in binary)
5	5	Note 1	AIXKY	Alternate key
Note 2		Note 3	AIXPT	First pointer to base cluster

Note 1: The length of this field is specified in AIXKL

Note 2: The displacement of this field is 5 + the length of AIXKY

Note 3: The length of this field is specified in AIXPL

Figure 5.8 **Alternate index record format**

Catalog

VSAM employs two types of catalogs - the master catalog and user catalogs. The internal structure and format of the two types is identical; the only difference is that the master catalog contains an entry for each user catalog.

Purpose

The VSAM catalog is built and processed by catalog-management modules. Catalog-management modules, via the catalog, enable a user to locate a data set, volume, index, or cluster by specifying a name or volume serial number. In addition, the VSAM catalog provides VSAM with the information required to allocate space for data sets, verify authorization to gain access to them, compile usage statistics on them, and relate RBAs to physical locations within data sets. The catalog indicates, therefore, much more than the simple location of data sets. The catalog maintains the relationship between a key-sequenced data set and its index, or between any data set and its alternate index(es), describes the location of VSAM data spaces and the data sets that reside in them, and describes the space that is available for new data sets.

Structure

The VSAM catalog is conceptually a key-sequenced VSAM data set divided into two key ranges. VSAM data set processing options, as record replication and sequence set with data, are utilized in both key ranges of the catalog. The catalog record size is 505 bytes in the low key range and 47 bytes in the high key range; the catalog control interval size is 512 bytes. Figure 5.9 shows the VSAM catalog. The figure shows:

- The low key range of the catalog, shown on the left, contains records that describe objects, that is, data sets, indexes, volumes, and clusters.
- The high key range of the catalog, shown on the right, contains the true name (a data set name or volume serial number) of an object specified by the user.
- The index, shown in the middle, points to both the low and high key ranges of the catalog.

With the exception of catalog records that are built when the catalog is created and describe the catalog itself, catalog records are built as objects are cataloged. The order of the records depends upon which portion of the catalog the records belong to. If the catalog records reside in the low key range of the catalog, they are ordered according to control-interval number. As objects are cataloged, available control intervals are used. If the catalog records reside in the high key range of the catalog, they are ordered according to their true name (data-set name or volume serial number).

Catalog management relies on record management for all record retrieval and storage. When a user specifies a data-set name, record management uses the index to retrieve a catalog record that contains the data-set name (in the high key range of the catalog); that record, in turn, contains the control-interval number of the catalog record that describes the data set. Catalog management converts the control-interval number to an RBA in the low key range of the catalog.

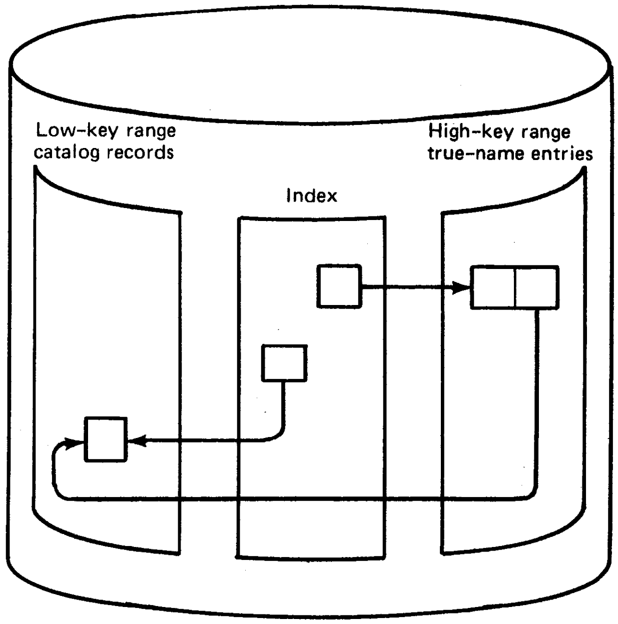


Figure 5.9 **Parts of the VSAM catalog**

Catalog Records which Describe the Catalog

Catalog records that describe the catalog as a data set are in fixed positions at the beginning of the catalog. Figure 5.10 shows the control-interval numbers of records that describe the catalog, the kind of catalog record each is, and the contents of each. The various types of catalog record are described later in this section.

Note that the self-describing records of the catalog do not contain CRA information (bytes 5-17). They do however use the release indicator (byte 4).

When the catalog is built, there are two True Name records. One contains the catalog's volume serial number and points to control-interval 9. The other contains the catalog's name and points to control-interval 2.

Types of Catalog Records

There are various types of catalog records. They are shown below, grouped according to the key range of the catalog in which they are located.

Control Interval Number	Record Type	Contents
0	Data	Description of the data portion of the catalog (low and high key ranges)
1	Index	Description of the index portion of the catalog
2	Cluster	Description of the catalog as a key-sequenced VSAM cluster. This catalog record contains the catalog's password information group occurrence.
3	Control	Catalog control record (CCR), which describes the catalog's free control intervals within the low key range.
4	Extension	Extension of the catalog-index record (control interval #1). This Extension record contains a description of the high-level index extents of the catalog.
5	Extension	Extension of the catalog-data record (control interval #0). This Extension record contains a description of the low key range data extents of the catalog.
6	Extension	Extension of the catalog-index record (control interval #1). For catalogs with embedded sequence-set records, this Extension record contains a description of the index sequence-set extents for the low key range of the catalog. For non-imbedded catalogs, this record is marked as free.
7	Extension	Extension of the catalog-data record (control interval #0). This Extension record contains a description of the extents of the True Name records in the high key range of the catalog.
8	Extension	Extension of the catalog index record (control interval #1). For catalogs with embedded sequence-set records, this Extension record contains a description of the index sequence-set extents for the high key range of the catalog. For non-imbedded catalogs, this record is marked as free.
9	Volume	Description of the track allocation and VSAM data spaces on this volume.
10,11...	Volume Extension	As many volume extension records as are necessary to describe the total space on the volume.

Figure 5.10 Catalog records that describe the catalog

High Key Range of the Catalog

The high key range of the catalog contains 47-byte True Name records. The True Name records associate user-specified name or volume serial numbers with the control-interval number of the catalog record that describes the specified object.

Low Key Range of the Catalog

Each catalog record in this part of the catalog occupies a full control interval and each contains the number of the control interval in which it resides. Each catalog record also contains the record type of the record. The low key range of the catalog is made up of the following types of records:

A: NonVSAM record, which describes a data set organized differently from VSAM. There is one nonVSAM record for each nonVSAM data set cataloged. Sometimes called 'Alien' record.

C: Cluster record, which describes a VSAM data-set cluster. This record contains the control-interval number of a Data record and, if the VSAM data set is a key-sequenced data set, the control-interval number of an Index record. There is one Cluster record for each VSAM cluster cataloged.

D: Data record, which describes the data component of a catalog, cluster, or AIX. There is one data record for each data set cataloged.

E: Extension record, which contains overflow information from another catalog record (except type 'V'). There are as many Extension records as are required to contain overflow information.

F: Free record, which marks the control interval in which it resides as available for use as another kind of catalog record. There is one Free record for each previously assigned control interval that is available for use.

G: Alternate Index record, which describes an alternate index. There is one such record for each alternate index cataloged.

I: Index record, which describes the index component of a catalog, cluster, or AIX. There is one index record for each index cataloged.

L: Control record, which describes the free control intervals in the low key range of the catalog. The Control record is the fourth record in the catalog.

R: Path record, which describes a VSAM path. There is one such record for each path cataloged.

U: User Catalog record, which describes a user catalog. One user catalog record is present in the master catalog for each user catalog which is cataloged.

V: Volume record, which describes each VSAM data space on a volume, the data sets that reside in the data space, and the space available within the data space. There is one Volume record for each volume controlled by this catalog.

W: Volume Extension record, which is used to extend volume records as required.

Y: Upgrade set record, which describes an upgrade set. There is one such record for each upgrade set cataloged.

Catalog Recovery Area

For a catalog defined with the recovery attribute, a Catalog Recovery Area (CRA) is reserved on each volume owned by the catalog. The CRA on each volume is conceptually an entry-sequenced data set which contains a self-describing part similar to that of the catalog and copies of catalog records describing the data sets on the volume. These copies are immediately updated whenever the original records in the catalog are changed.

Self-Describing Part of the CRA

This part of the CRA contains those records which are necessary to describe the CRA. They are basically the same records as those in the self-describing part of a catalog, except that the records describing an index are not needed, because the CRA is an entry-sequenced data set. The unused control intervals contain free records. Figure 5.11 shows the format of the self-describing part of the CRA.

Control Interval number	Record type	Contents
0	Data	Description of the data portion of the CRA The field CRAVOL in this record contains the volume serial number of the catalog which owns the volume.
1	Free record	
2	Cluster	Description of the CRA as an entry-sequenced VSAM cluster. This record contains the name of the catalog which owns the volume.
3	Control	Catalog control record which describes the free control intervals in the CRA
4	Free record	
5	Extension	Extension of the CRA data record (CI#0)
6	Free record	
7	Free record	
8	Free record	
9	Volume	Description of the track allocation for the CRA
10	Volume extension	
11	Volume extension	Extensions of the volume record (CI#9)
12	Volume extension	

Figure 5.11 **Self-describing part of the CRA**

Copies of Catalog Records

All catalog records which describe data sets or volumes are duplicated on specific volumes, as shown below:

A volume record is duplicated in the CRA of the volume which it describes.

All records concerning a key-sequenced data set or its alternate index (Cluster, Alternate Index, Data, Index, Path, and Upgrade Set records) are duplicated in the CRA of the first volume on which space was allocated for the index of the base cluster.

All records concerning an entry-sequenced data set or its alternate index, or a relative record data set, (Cluster, Alternate Index, Data and Index for the AIX, Path, and Upgrade Set records) are duplicated on the first volume on which space was allocated for the data component of the base cluster.

If a volume is imported from an OS/VS VSAM system, the CRA may contain other records, such as records which describe a nonVSAM data set. These records are not, however, used by VSE/VSAM.

Catalog Record Formats

True-name Catalog Record

The True Name record associates the volume serial number, data-set name, or cluster name specified by the user with the control-interval number of the catalog record that describes the object (volume, cluster, nonVSAM [alien] data set, data component, index component, path, or alternate index). True Name records are contained in the high key range part of the catalog and are pointed to by the catalog's index records. The True Name record is retrieved using key-sequenced processing. The catalog-management modules convert the control-interval number in the True Name record to an RBA which can be used to retrieve the associated record in the low key range.

True Name records are 47 bytes long; several might be contained in a catalog's control interval (512 bytes). The format of that record is shown in Figure 5.12.

Offset Hex	Offset Dec	Bytes	Description
0	0	44	Name of a data set or cluster, filled on the right with blanks, or a volume serial number, filled on the right with zeros, specified by the user.
44	2C	3	Control-interval number of the catalog record that describes the object.

Figure 5.12 **True-name catalog record format**

NonVSAM Catalog Record

The nonVSAM catalog record describes a nonVSAM data set. Figure 5.13 shows the format of a nonVSAM catalog record.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	44		Key and ID area
0	0	1		Binary zeros
1	1	3	ENTIDNO	Control interval number of entry
4	4	1	RELIND	Release indicator; X'00' This record was created with a DOS/VS Release prior to Release 31. X'01' This record was created with DOS/VS Release 31 or later.
5	5	6	CRAVOL	CRA volume serial number
11	B	3	CRAIDNO	CRA control interval number
14	E	4	CRADEVT	CRA device type
18	12	26		Binary zeros
44	2C	1	ENTYPE	Record type - 'A'
45	2D	2		Length of information contained in record
47	2F	1		Number of variable-length fields that precede the pointer to an Extension record. Always zero.
48	30	1		Length of the fixed-length fields in this record, excluding any fixed-length fields that follow displacement 92 (5C). This value is always equal to the displacement from the beginning of the record to the pointer to an Extension record.
49	31	44	ENTNAME	Entry name (data set name)
The following six-byte entry contains control information for the group occurrence pointers that follow it.[1]				
93	5D	5		Pointer to Extension record. If this record is not continued in an Extension record, this field contains zeros.
98	62	1		The number of group occurrence pointers that follow.

Figure 5.13 **NonVSAM catalog record format (part 1 of 2)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
				Group occurrence pointer (repetitive)
Bit 0 and 1 of Byte 3 identify the group occurrence further:				
		00xx xxxx		Pointer to a group occurrence within the record. **Byte** **Meaning** 0 Reserved 1-2 Displacement of the group occurrence from the beginning of all group occurrences in this record 3 Bits 0 and 1 are set to zero. Bits 2 through 7 contain a code describing the group occurrence pointed to. 4 Sequence number of the group occurrence pointed to.
		10xx xxxx		Pointer to a group occurrence contained in an Extension record. **Byte** **Meaning** 0-2 Control-interval number of the Extension record that contains this group occurrence. 3 Bits 0 and 1 are set to 1 and 0, respectively. Bits 2 through 7 contain a code describing the group occurrence pointed to. 4 Sequence number of the group occurrence pointed to.
		01xx xxxx		Pointer to a group occurrence which has been deleted. The code in bits 2 through 7 of byte 3 and the sequence number in byte 4 have been retained however.

[1] For further information see section *Group occurrences in catalog records.*

Figure 5.13 NonVSAM catalog record format (part 2 of 2)

Cluster Catalog Record

The Cluster record describes a data set and its index. Figure 5.14 shows the format of a cluster catalog record.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	44		Key and ID area
0	0	1		Binary zeros
1	1	3	ENTIDNO	Control interval number of entry
4	4	1	RELIND	Release indicator; X'00' This record was created with a DOS/VS Release prior to Release 31. X'01' This record was created with DOS/VS Release 31 or later.
5	5	6	CRAVOL	CRA volume serial number
11	B	3	CRAIDNO	CRA CI number
14	E	4	CRADEVT	CRA device type
18	12	26		Binary zeros
44	2C	1	ENTYPE	Record type - 'C' for a cluster record
45	2D	2		Record length
47	2F	1		Number of variable-length fields that precede the pointer to an Extension record. Always zero.
48	30	1		Length of the fixed-length fields in this record, excluding any fixed-length fields that follow displacement 107 (6B). This value is always equal to the displacement from the beginning of the record to the pointer to an Extension record.
49	31	44	ENTNAME	Entry name (Normally true name of the cluster. If this record is in the self-describing part of the CRA, it contains the name of the catalog which owns this volume)
93	5D	8	OWNERID	Owner of data set
101	65	3	DSETCRDT	Data set creation data (ydd)
104	69	3	DSETEXDT	Data set expiration date (ydd)
107	6B	1	CATTR	Reserved (for OS)
The following six-byte entry contains control information for the group occurrence pointers that follow it.[1]				
108	6C	5		Pointer to Extension record. If this record is not continued on an Extension record, this field contains zeros.
113	71	1		The number of group occurrence pointers that follow.

Figure 5.14 Cluster catalog record format (part 1 of 2)

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
				Group occurrence pointer (repetitive)
Bit 0 and 1 of Byte 3 identify the group occurrence further:				
		00xx xxxx		Pointer to a group occurrence within the record.
				Byte **Meaning**
				0 Reserved
				1-2 Displacement of the group occurrence from the beginning of all group occurrences in this record.
				3 Bits 0 and 1 are set to zero. Bits 2 through 7 contain a code describing the group occurrence pointed to.
				4 Sequence number of the group occurrence pointed to by code.
		10xx xxxx		Pointer to a group occurrence contained in an Extension record.
				Byte **Meaning**
				0-2 Control-interval number of the Extension record that contains this group occurrence.
				3 Bits 0 and 1 are set to 1 and 0, respectively. Bits 2 through 7 contain a code that describes the group occurrence pointed to.
				4 Sequence number of the group occurrence pointed to by code.
		01xx xxxx		Pointer to a group occurrence which has been deleted.
				The code in bits 2 through 7 of byte 3 and the sequence number in byte 4 have been retained however.

[1] For further information see section *Group occurrences in catalog records.*

Figure 5.14 Cluster catalog record format (part 2 of 2)

Data and Index Catalog Record

Data and Index records describe data sets and their indexes. Figure 5.15 shows the format of the data and index records.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	44		Key and ID area
0	0	1		Binary zeros
1	1	3	ENTIDNO	Control interval number of entry
4	4	1	RELIND	Release indicator; X'00' This record was created with a DOS/VS Release prior to Release 31. X'01' This record was created with DOS/VS Release 31 or later.
5	5	6	CRAVOL	CRA volume serial number
11	B	3	CRAIDNO	CRA control interval number
14	E	4	CRADEVT	CRA device type
18	12	26		Binary zeros
44	2C	1	ENTYPE	Record type - 'D' for a data record or 'I' for an index record
45	2D	2		Record length
47	2F	1		Number of variable-length fields that precede the pointer to an Extension record. Always zero.
48	30	1		Length of the fixed-length fields in this record, excluding any fixed-length fields following displacement 142 (8E). This value is always equal to the displacement from the beginning of the record to the pointer to an Extension record.
49	31	44	ENTNAME	For a Data or Index record, the data-set name
93	5D	8	OWNERID	Owner of the data set, specified when the data set was defined.
101	65	3	DSETCRDT	Data-set creation data, in packed-decimal form YDD, specified when the data set was defined.
104	68	3	DSETEXDT	Data-set expiration date, in packed-decimal form YDD, specified when the data set was defined.
107	6B	1	ATTR1	Data-set attributes, which are defined in Access Method Services commands, as follows:
		1...		Speed – recovery features will be minimized or omitted in order to optimize operating speed during initial loading
		.1..		Unique component. Entire space occupied by a unique component.
		..1.		Reusable data set
		...1		Erase the component upon deletion
		.... 1...		Recoverable catalog
		.... .1..		Inhibit update
		.... ..1.		This component has been temporarily exported
		x		Reserved for OS/VS MVM

Figure 5.15 **Data and index catalog record format (part 1 of 4)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
108	6C	1	ATTR2	Data-set sharing attributes as follows: Cross-partition sharing:
		00..		The data set can be shared by READ users *or* it can be used by one UPDATE/OUTPUT user.
		01..		The data set can be shared by READ users *and* one UPDATE/OUTPUT user.
		10..		The data set can be fully shared
		11..		The data set can be fully shared; with assistance supplied by VSAM. Cross-system sharing (set by DOS but used only by OS):
		..00		The data set can be shared by READ users *or* it can be used by one UPDATE/OUTPUT user.
		..01		The data set can be shared by READ user *and* one UPDATE/OUTPUT user.
		..10		The data set can be fully shared
		..11		The data set can be fully shared; with assistance supplied by VSAM.
		.... 1...		NOALLOCATION cluster
		1		DS not usable
		.... .xx.		Reserved
109	6D	1	OPENIND	Open indicator flag; if this byte contains X'80', the data set is open for output.
110	6E	4	BUFSIZE	Minimum buffer size
114	72	3	PRIMSPAC	Primary space allocated for the data set or index, specified when the data set or index was defined.
117	75	3	SCONSPAC	Secondary space allocation for the data set or index, specified when the data set or index was defined.

Figure 5.15 Data and index catalog record format (part 2 of 4)

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
120	78	1	SPACOPTN	Space options flags.
		10..		Track request, which indicates that space was allocated in terms of tracks.
		11..		Cylinder request, which indicates that space was allocated in terms of cylinders.
		...0		Secondary allocation class same as primary allocation class. (Cluster, AIX)
		...1		Use class-0 for secondary allocation
		.... .000		Data set in class-0 space.
		.... .001		Data set in class-1 space.
		.... .010		Data set in class-2 space
		.... .011		Data set in class-3 space
		.... .100		Data set in class-4 space
		.... .101		Data set in class-5 space
		.... .110		Data set in class-6 space
		.... .111		Data set in class-7 space.
		..x. x...		Reserved.
121	79	4	HURBADS	High used RBA of the data set or index.
125	7D	4	HARBADS	High allocated RBA of the data set or index.
129	81	4	LRECL	For a Data record, the logical record size of the data set described by this Data record. For an Index record, always X'FF's.
133	85	2	USERINFO	User information for the ISAM interface program (IIP).
135	87	8	EXCPEXIT	Exception exit
The following six-byte entry contains control information for the group occurrence pointers that follow it.[1]				
143	8F	5		Pointer to Extension record. If this record is not continued in an Extension record, this field contains zeros.
148	94	1		The number of group occurrence pointers that follow.[1]
			Group occurrence pointer (repetitive)	
Bit 0 and 1 of Byte 3 identify the group occurrence further:				
		00xx xxxx		Pointer to a group occurrence within the record **Byte** / **Meaning** 0 — Reserved 1-2 — Displacement of the group occurrence from the beginning of all group occurrences in this record. 3 — Bits 0 and 1 are set to zero. Bits 2 through 7 contain a code that describes the group occurrence pointed to. 4 — Sequence number of the group occurrence pointed to.

Figure 5.15 **Data and index catalog record format (part 3 of 4)**

Offset Dec	Hex	Bytes and Bit Pattern	Field Name	Description
		10xx xxxx		Pointer to a group occurrence contained in an Extension record. **Byte** **Meaning** 0-2 Control-interval number of the Extension record that contains this group occurrence. 3 Bits 0 and 1 are set to 1 and 0, respectively. Bits 2 through 7 contain a code that describes the group occurrence pointed to. 4 Sequence number of the group occurrence pointed to by code.
		01xx xxxx		Pointer to a group occurrence which has been deleted. The code in bits 2 through 7 of byte 3 and the sequence number in byte 4 have been retained however.

[1] For further information see section *Group occurrences in catalog records*.

Figure 5.15 **Data and index catalog record format (part 4 of 4)**

Extension Catalog Record

The Extension record contains overflow information from another catalog record. Figure 5.16 shows the format of an extension catalog record.

Offset Dec	Hex	Bytes and Bit Pattern	Field Name	Description
0	0	44		Key and ID area
0	0	1		Binary zeros
1	1	3	ENTIDNO	Control interval number of entry
4	4	1	RELIND	Release indicator; X'00' This record was created with a DOS/VS Release prior to Release 31 X'01' This record was created with DOS/VS Release 31 or later
5	5	6	CRAVOL	CRA volume serial number
11	B	3	CRAIDNO	CRA control interval number
14	E	4	CRADEVT	CRA device type
18	12	26		Binary zeros
44	2C	1	ENTYPE	Record type - 'E', for an extension of any other record except a volume catalog record where record type is 'W'.
45	2D	2		Record length

Figure 5.16 **Extension catalog record format (part 1 of 2)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
47	2F	1		Number of variable-length fields that precede the pointer to an Extension record. Always zero.
48	30	1		Length of the fixed-length fields in the header fields, excluding any fixed length fields following displacement 48 (30). This value is always equal to the displacement from the beginning of the record to the extension record's pointer.
The following six-byte entry contains control information for the group occurrence pointers that follow it.[1]				
49	31	5		Pointer to Extension record. If this record is not continued on an Extension record, this field contains zeros.
54	36	1		The number of group occurrence pointers that follow.
			Group occurrence pointer (repetitive)	
Bit 0 and 1 of Byte 3 identify the group occurrence further:				
		00xx xxxx		Pointer to a group occurrence within the record. **Byte** — **Meaning** 0 — Reserved 1-2 — Displacement of the group occurrence from the beginning of all group occurrences in this record. 3 — Bits 0 and 1 are set to zero. Bits 2 through 7 contain a code describing the group occurrence pointed to. 4 (4-5)[2] — Sequence number of the group occurrence pointed to be code.
		10xx xxxx		Pointer to a group occurrence contained in an Extension record. **Byte** — **Meaning** 0-2 — Control-interval number of the Extension record that contains this group occurrence. 3 — Bits 0 and 1 are set to 1 and 0, respectively. Bits 2 through 7 contain a code that describes the group occurrence pointed to. 4 (4-5)[2] — Sequence number of the group occurrence pointed to be code.
		01xx xxxx		Pointer to a group occurrence which has been deleted. The code in bits 2 through 7 of byte 3 and the sequence number in bytes 4 and 5 of 'W' type have been retained however.

1 For further information see section *Group occurrences in catalog records.*

2 For 'W' type records, this sequence number occupies bytes 4 and 5, and the group occurrence pointer is 6 bytes long.

Figure 5.16 Extension catalog record format (part 2 of 2)

Free Catalog Record

The Free record indicates that the control interval in which it resides is free and points to the next control interval that is free because of deletion. Note that the free space in the catalog that has never been assigned is not represented by a Free record; a Free record is used only to mark a record that was used and deleted. Figure 5.17 shows the format of the free record.

Offset Dec	Offset Hex	Bytes	Description	
0	0	44	Key.	
			Byte	**Meaning**
			0	Zeros.
			1-3	Control-interval number of this record
			4-43	Zeros
44	2C	1	Record type - 'F'.	
45	2D	3	Control-interval number of the next free control interval	

Figure 5.17 Free catalog record format

Alternate Index Catalog Record

The alternate index catalog record describes the alternate index as a key-sequenced data set and relates it to the base cluster it belongs to. Figure 5.18 shows the format if the alternate index catalog record.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	44		Key and ID area
0	0	1		Binary zeros
1	1	3	ENTIDNO	Control interval number of entry
4	4	1	RELIND	Release indicator; X'00' This record was created with a DOS/VS Release prior to Release 31 X'01' This record was created with DOS/VS Release 31 or later
5	5	6	CRAVOL	CRA volume serial number
11	B	3	CRAIDNO	CRA control interval number
14	E	4	CRADEVT	CRA device type
18	12	26		Binary zeros
44	2C	1	ENTYPE	Record type - 'G'
45	2D	2		Record length
47	2F	1		Number of variable-length fields that precede the pointer to an Extension record. Always zero.
48	30	1		Length of the fixed-length fields in this record, excluding any fixed-length fields that follow displacement 107 (6B). This value is always equal to the displacement from the beginning of the record to the pointer to an Extension record.
49	31	44	ENTNAME	Entry name (data set name)
93	5D	8	OWNERID	Owner of data set
101	65	3	DSETCRDT	Data set creation date (ydd)
104	69	3	DSETEXDT	Data set expiration date (ydd)
107	6B	1	RGATTR	AIX attribute, X'80' = upgrade (AIX)
The following six-byte entry contains control information for the group occurrence pointers that follow it.[1]				
108	6C	5		Pointer to Extension record. If this record is not continued on an Extension record, this field contains zeros.
		1		The number of group occurrence pointers that follow.

Figure 5.18 **Alternate index catalog record format (part 1 of 2)**

Offset Dec	Hex	Bytes and Bit Pattern	Field Name	Description
				Group occurrence pointer (repetitive)
Bit 0 and 1 of Byte 3 identify the group occurrence further:				
		00xx xxxx		Pointer to a group occurrence within the record. **Byte** **Meaning** 0 Reserved 1-2 Displacement of the group occurrence from the beginning of all group occurrences in this record. 3 Bits 0 and 1 are set to zero. Bits 2 through 7 contain a code that describes the group occurrence pointed to. 4 Sequence number of the group occurrence pointed to.
		10xx xxxx		Pointer to a group occurrence contained in an Extension record. **Byte** **Meaning** 0-2 Control-interval number of the Extension record that contains this group occurrence. 3 Bits 0 and 1 are set to 1 and 0, respectively. Bits 2 through 7 contain a code that describes the group occurrence pointed to. 4 Sequence number of the group occurrence pointed to by code.
		01xx xxxx		Pointer to a group occurrence which has been deleted. The code in bits 2 through 7 of byte 3 and the sequence number in byte 4 have been retained however.

[1] For further information see section *Group occurrences in catalog records*.

Figure 5.18 **Alternate index catalog record format (part 2 of 2)**

Index Catalog Record

For index catalog record format see data catalog record format. The record type (byte 44) is 'I'.

Catalog Control Record (CCR)

The catalog control record (CCR) is used by catalog management to control the allocation of control intervals in the low key range of the catalog, where catalog records, excluding the True Name records and the index, reside. The CCR also shows the catalog's high-used and high-allocated RBA values. The catalog control record is the fourth record (control interval) in the catalog.

For a request of one catalog record, catalog management tries to use a record that was freed because of deletion. This process is done before using unassigned space. If more than one catalog record is needed, catalog management tries to use contiguous unassigned space in the current extent; if sufficient unassigned space is not available, records that have been deleted are used. Figure 5.19 shows the CCR format.

Offset Dec	Offset Hex	Bytes	Description
0	0	44	Key **Byte** **Meaning** 0 Zeros. 1-3 Control-interval number of this record. 4-43 Zeros.
44	2C	1	Record type - 'L'.
45	2D	3	Number of the highest control interval within the current extent.[1]
48	30	3	Number of the next free control interval of those that have not been previously assigned.[1]
51	33	3	Count of deleted control intervals, that is, the count of control intervals that are free because of deletion.[2]
54	36	3	First deleted control interval in a chain of control intervals that are free because of deletion.[2]
The following fields are used to keep track of the RBA values that denote the current logical end of parts of the catalog:			
57	39	4	Data, low key range: high-key RBA
61	3D	4	Data, low key range: high-used RBA
65	41	4	Data, low key range: high-allocated RBA
69	45	4	Data, high key range: high-key RBA
73	49	4	Data, high key range: high-used RBA
77	4D	4	Data, high key range: high-allocated RBA
81	51	4	Index, high level: high-used RBA
85	55	4	Index, high level: high-allocated RBA
89	59	4	Index, low key range - sequence set: high-used RBA
93	5D	4	Index, low key range - sequence set: high-allocated RBA
97	61	4	Index, high key range - sequence set: high-used RBA
101	65	4	Index, high key range - sequence set: high-allocated RBA

1 This field is used to keep track of unassigned space within the current extent.

2 This field is used to keep track of previously-used records that are now available for use as another catalog record.

Figure 5.19 **Catalog control record format**

Path Catalog Record

The path catalog record describes a VSAM-path which is a logical connection between a base cluster and an alternate index. Figure 5.20 shows the format of the path catalog record.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	44		Key and ID area
0	0	1		Binary zeros
1	1	3	ENTIDNO	Control interval number of entry
4	4	1	RELIND	Release indicator; X'00' This record was created with a DOS/VS Release prior to Release 31 X'01' This record was created with DOS/VS Release 31 or later
5	5	6	CRAVOL	CRA volume serial number
11	B	3	CRAIDNO	CRA control interval number
14	E	4	CRADEVT	CRA device type
18	12	26		Binary zeros
44	2C	1	ENTYPE	Record type - 'R'
45	2D	2		Record length
47	2F	1		Number of variable-length fields that precede the pointer to an Extension record. Always zero.
48	30	1		Length of the fixed-length fields in this record, excluding any fixed-length fields following displacement 107 (6B). This value is always equal to the displacement from the beginning of the record to the pointer to an Extension record.
49	31	44	ENTNAME	Entry name (data set name)
93	5D	8	OWNERID	Owner of data set
101	65	3	DSETCRDT	Data set creation date (ydd)
104	69	3	DSETEXDT	Data set expiration date (ydd)
107	6B	1	RGATTR	Path attribute, X'80' = update (path)
The following six-byte entry contains control information for the group occurrence pointers that follow it.[1]				
108	6C	5		Pointer to Extension record. If this record is not continued on an Extension record, this field contains zeros.
		1		The number of group occurrence pointers that follow.

Figure 5.20 Path catalog record format (part 1 of 2)

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
				Group occurrence pointer (repetitive)
Bit 0 and 1 of Byte 3 identify the group occurrence further:				
		00xx xxxx		Pointer to a group occurrence within the record. **Byte** **Meaning** 0 Reserved 1-2 Displacement of the group occurrence from the beginning of all group occurrences in this record. 3 Bit 0 and 1 are set to zero. Bits 2 through 7 contain a code that describes the group occurrence pointed to. 4 Sequence number of the group occurrence pointed to.
		10xx xxxx		Pointer to a group occurrence contained in an Extension record **Byte** **Meaning** 0-2 Control-interval number of the Extension record that contains this group occurrence. 3 Bits 0 and 1 are set to 1 and 0, respectively. Bits 2 through 7 contain a code that describes the group occurrence pointed to. 4 Sequence number of the group occurrence pointed to by code.
		01xx xxxx		Pointer to a group occurrence which has been deleted. The code in bits 2 through 7 of byte 3 and the sequence number in byte 4 have been retained however.

[1] For further information see section *Group occurrences in catalog records*.

Figure 5.20 Path catalog record format (part 2 of 2)

User-Catalog Catalog Record

This record can only occur in the VSAM master catalog. It describes a user catalog. Figure 5.21 shows the format of the user-catalog catalog record.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	44		Key and ID area
0	0	1		Binary zeros
1	1	3	ENTIDNO	Control interval number of entry
4	4	1	RELIND	Release indicator; X'00' This record was created with a DOS/VS Release prior to Release 31 X'01' This record was created with DOS/VS Release 31 or later
5	5	6	CRAVOL	CRA volume serial number
11	B	3	CRAIDNO	CRA control interval number
14	E	4	CRADEVT	CRA device type
18	12	26		Binary zeros
44	2C	1	ENTYPE	Record type - 'U'
45	2D	2		Record length
47	2F	1		Number of variable-length fields that precede the pointer to an Extension record. Always zero.
48	30	1		Length of the fixed-length fields in this record, excluding any fixed-length fields following displacement 92 (5C). This value is always equal to the displacement from the beginning of the record to the pointer to an Extension record.
49	31	44	ENTNAME	Entry name (data set name)
The following six-byte entry contains control information for the group occurrence pointers that follow it.[1]				
93	5D	5		Pointer to Extension record. If this record is not continued on an Extension record, this field contains zeros.
98	62	1		The number of group occurrence pointers that follow.[1]
			Group occurrence pointer (repetitive)	
Bit 0 and 1 of Byte 3 identify the group occurrence further:				
		00xx xxxx		Pointer to a group occurrence within the record. **Byte** **Meaning** 0 Reserved 1-2 Displacement of the group occurrence from the beginning of all group occurrences in this record. 3 Bit 0 and 1 are set to zero. Bits 2 through 7 contain a code that describes the group occurrence pointed to. 4 Sequence number of the group occurrence pointed to.

Figure 5.21 **User-catalog catalog record format (part 1 of 2)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
		10xx xxxx		Pointer to a group occurrence contained in an Extension record. **Byte** **Meaning** 0-2 Control-interval number of the Extension record that contains this group occurrence. 3 Bits 0 and 1 are set to 1 and 0, respectively. Bits 2 through 7 contain a code that describes the group occurrence pointed to. 4 Sequence number of the group occurrence pointed to by code.
		01xx xxxx		Pointer to a group occurrence which has been deleted. The code in bits 2 through 7 of byte 3 and the sequence number in byte 4 have been retained however.

[1] For further information see section *Group occurrences in catalog records.*

Figure 5.21 **User-catalog catalog record format (part 2 of 2)**

Volume Catalog Record

The Volume record describes VSAM data spaces, their extents, and the data sets that reside in VSAM data spaces. Figure 5.22 shows the format of the volume record.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	44		Key and ID area
0	0	1		Binary zeros
1	1	3	ENTIDNO	Control interval number of entry
4	4	1	RELIND	Release indicator; X'00' This record was created with a DOS/VS Release prior to Release 31. X'01' This record was created with DOS/VS Release 31 or later.
5	5	6	CRAVOL	CRA volume serial number
11	B	3	CRAIDNO	CRA control interval number
14	E	4	CRADEVT	CRA device type
18	12	26		Binary zeros
44	2C	1	ENTYPE	Record type - 'V'
45	2D	2		Record length

Figure 5.22 **Volume catalog record format (part 1 of 3)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
47	2F	1		Number of variable-length fields that precede the pointer to an Extension record. Always zero.
48	30	1		Length of the fixed-length fields in this record, excluding any fixed-length fields following displacement 126 (7E). This value is always equal to the displacement from the beginning of the record to the pointer to an Extension record.
49	31	44	ENTNAME	Volume serial number, filled with binary zeros on the right, of the volume described by this record.
93	5D	8	VOLTSTMP	Volume time stamp, which indicates the time-of-day clock value when space was last added to, or deleted from, this volume.
101	65	20	VOLDVCHR	Device characteristics.
				Byte — **Meaning**
				0-3 — Volume device type.
				4-7 — Maximum device blocksize.
				8-9 — Number of cylinders on this volume.
				10-11 — Number of tracks per cylinder on this volume.
				12-13 — Number of bytes per track on this volume.
				14 — Number of bytes required for gaps and check bits for each keyed block other than the last block on a track for this volume.[1]
				15 — Number of bytes required for gaps and check bits for the last keyed block on a track for this volume.[1]
				16 — Number of bytes to be subtracted for a block that is not keyed.[1]
				17 — Flags:
		.... 1...		Block overhead for keyed records is a two-byte field (bytes 15 and 16.
		1		Use tolerance factor (bytes 18 and 19) on all blocks except the last block to calculate the effective length of a block.[1]
		xxxx .xx.		Reserved.
121	79	1	VOLRFLG	Volume record flags
122	7A	1	SYSEXTDS	Number of extents per suballocation-request allowed by the system.
123	7B	4		Space in the catalog record into which the following 4 bytes of derived information will be moved (in the buffer) if requested.

Figure 5.22 **Volume catalog record format (part 2 of 3)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
The following field names identify information that is not contained in the volume catalog record; the information is derived from the group occurrences in the volume catalog record and will be placed in the buffer if requested:				
123	7B	2	NODSPACE	Number of data spaces on the volume - a count of the Data Space Group occurrences.
125	7D	2	NODSET	Number of data sets on the volume - a count of the Data Set Directory Entry group occurrences.
The following six-byte entry contains control information for the group occurrence pointers that follow it.[1]				
127	7F	5		Pointer to Extension record. If this record is not continued on an Extension record, this field contains zeros.
132	84	1		Number of group occurrence pointers that follow.
			Group occurrence pointer (repetitive)	
Bit 0 and 1 of Byte 3 identify the group occurrence further:				
		00xx xxxx		Pointer to a group occurrence within the record. **Byte** — **Meaning** 0 — Reserved. 1-2 — Displacement of the group occurrence from the beginning of all group occurrences in this record. 3 — Bits 0 and 1 are set to zero. Bits 2 through 7 contain a code describing the group occurrence pointed to. 4-5 — Sequence number of the group occurrence pointed to.
		10xx xxxx		Pointer to a group occurrence contained in an Extension record. **Byte** — **Meaning** 0-2 — Control-interval number of the Extension record that contains this group occurrence. 3 — Bits 0 and 1 are set to 1 and 0, respectively. Bits 2 through 7 contain a code that describes the group occurrence pointed to. 4-5 — Sequence number of the group occurrence pointed to.

[1] For further information see section *Group occurrences in catalog records*.

Figure 5.22 Volume catalog record format (part 3 of 3)

Volume Extension Catalog Record

For this format see extension catalog record format. The record type (byte 44) is 'W'. Note that the Group occurrence pointers in a 'W' type record are 6 bytes long, similar to those in a 'V' type record.

Upgrade Set Catalog Record

An upgrade set catalog record describes an AIX in the upgrade set which is the group of alternate indexes belonging to a base cluster which are to be updated automatically. Figure 5.23 shows the format of the upgrade set catalog record.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	44		Key and ID area
0	0	1		Binary zeros
1	1	3	ENTIDNO	Control interval number of entry
4	4	1	RELIND	Release indicator; X'00' This record was created with a DOS/VS Release prior to Release 31 X'01' This record was created with DOS/VS Release 31 or later
5	5	6	CRAVOL	CRA volume serial number
11	B	3	CRAIDNO	CRA control interval number
14	E	4	CRADEVT	CRA device type
18	12	26		Binary zeros
44	2C	1	ENTYPE	Entry type - 'Y'
45	2D	2		Record length
47	2F	1		Number of variable-length fields that precede the pointer to an Extension record. Always zero.
48	30	1		Length of the fixed-length fields in this record, excluding any fixed-length fields following displacement 48 (30). This value is always equal to the displacement from the beginning of the record to the pointer to an Extension record.
The following six-byte entry contains control information for the group occurrence pointers that follow it.[1]				
49	31	5		Pointer to Extension record. If this record is not continued on an Extension record, this field contains zeros.
		1		The number of group occurrence pointers that follow.

Figure 5.23 Upgrade set catalog record format (part 1 of 2)

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
				Group occurrence pointer (repetitive)
Bit 0 and 1 of Byte 3 identify the group occurrence further:				
		00xx xxxx		Pointer to a group occurrence within the record.
				Byte **Meaning**
				0 Reserved.
				1-2 Displacement of the group occurrence from the beginning of all group occurrences in this record.
				3 Bits 0 and 1 are set to zero. Bits 2 through 7 contain a code that describes the group occurrence pointed to.
				4 Sequence number of the group occurrence pointed to.
		10xx xxxx		Pointer to a group occurrence contained in an Extension record.
				Byte **Meaning**
				0-2 Control-interval number of the Extension record that contains this group occurrence.
				3 Bits 0 and 1 are set to 1 and 0, respectively. Bits 2 through 7 contain a code that describes the group occurrence pointed to.
				4 Sequence number of the group occurrence pointed to by code.
		01xx xxxx		Pointer to a group occurrence which has been deleted.
				The code in bits 2 through 7 of byte 3 and the sequence number in byte 4 have been retained however.

[1] For further information see section *Group occurrences in catalog records.*

Figure 5.23 **Upgrade set catalog record format (part 2 of 2)**

Group Occurrences in Catalog Records

Group occurrences are fields of related information within a catalog record which are grouped together so that they can be treated as a unit. For example, all fields relating to a volume on which a data set resides are located in one group occurrence. There is one group occurrence for each volume concerned. Thus, if a data set resides on three volumes, there are three group occurrences, which are not necessarily contiguous.

The group occurrences are found by means of group occurrence pointers in the catalog record. These pointers also contain a code which identifies the type of group occurrence to which they point. The pointers are grouped in such a way that all pointers to a particular type of group occurrence are together. In the example above, the pointers to the three volume information group occurrences would be contiguous, even if the group occurrences are not.

Group Occurrences in Extension Records

This description is designed to show the steps involved in building the group of records shown in Figure 5.24. The numbers in parentheses in the description refer to the record numbers in Figure 5.24. Initially, when the base record (1) is created, the group occurrence pointers (GOPs) and group occurrences (GOs) are placed in the base record. As further GOs and GOPs are added, the available space becomes insufficient and the base record must be extended.

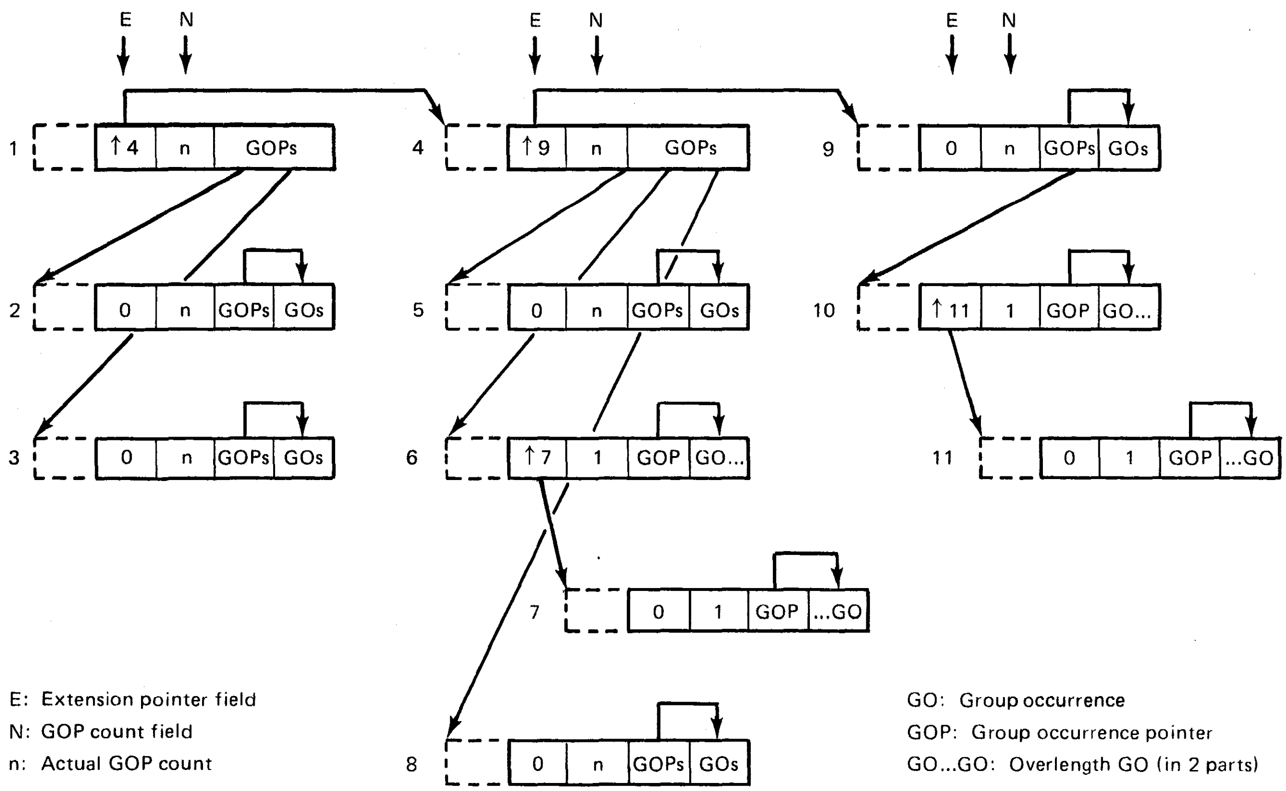
This is done by means of a **vertical chain** of extension records. GOs are moved out of the base record (1) into an extension record (2) to create space for GOPs. If necessary, further extension record(s) (3) can be added to this vertical chain. The GOs in the extension records are located by means of two GOPs: the first GOP, in the base record, points to the extension record, where a second GOP with the same group code and sequence number as the first, points to the associated GO within the extension record.

Later, the GOPs in the base record may require more space than is available, although all GOs have been moved to the vertical chain. In this case, the base record (1) is extended by a **horizontal chain**. The first record in the horizontal chain(4) is located by means of the extension pointer in the base record (1). This extension record (4) is now treated similarly to the original base record. It first contains both GOPs and GOs, then extended with records in a vertical chain (5,6,8), and finally extended horizontally (9).

If it becomes necessary to store a GO which will not fit into a single record, a **pseudo-horizontal chain** becomes necessary. If this is required within a vertical chain (6), the extension record (7) is simply chained with the aid of the extension pointer in the record containing the first part of the GO (6).

A pseudo-horizontal chain is however, not allowed within the horizontal chain (1,4,9). Thus, if an overlong GO is encountered at a time where the last record in the horizontal chain (9) still contains GOPs, GOs, and free space (which means that the next GO would normally be placed in this record) an artificial vertical chain (10) must first be built. This can the be extended in the form of a pseudo-horizontal chain (10,11) to hold the GO.

The first record in a pseudo-horizontal chain always contains an extension pointer to the second record and always has a GOP count of 1, because such a "spanned" GO may not share a record with any other GO.



Record 1: Base record, containing GOPs pointing to records 2 and 3 (vertical pointers) and an extension pointer to record 4 (horizontal pointer).

Record 2: Extension record, containing GOs from record 1 }
Record 3: Extension record, containing GOs from record 1 } Vertical chain

Record 4: Extension record in the horizontal chain, containing further GOPs pointing to records 5, 6, and 8 (vertical pointers) and an extension pointer to record 9 (horizontal pointer).

Record 5: Extension record containing GOs from record 4 }
Record 6: Extension record containing 1st part of long GO } Pseudo chain } Vertical chain
Record 7: Extension record containing 2nd part of long GO }
Record 8: Extension record containing GOs from record 4 }

Record 9: Extension record containing further GOPs and GOs (last record in the horizontal chain).

Record 10: Extension record containing 1st part of long GO } Artificial vertical chain
Record 11: Extension record containing 2nd part of long GO } with pseudo-horizontal chain

Figure 5.24 Group occurrences in extension records.

Types of Group Occurrences

The following list shows the various types of group occurrences, and the types of catalog records in which they can occur. Note that it is possible for one catalog record to contain many group occurrences.

- AMDSB (Access Method Data Set Statistics Block), which appears in Data and Index records. Only one copy of an AMDSB appears in a record. A pointer to AMDSB information contains a code of 1.
- Association information, which occurs in Data, Index, Cluster, Alternate Index, Path, and Upgrade Set records. For further details, see *Association Group Occurrences* later in this section. A pointer to association information contains a code of 2.
- Volume information, which appears in Data, Index, User Catalog, and non-VSAM records. This group occurrence describes all of the direct-access device space allocated to the data set (or index, etc.) on a particular volume. A separate set of volume information fields is used to describe the total space on each volume. If the data set's space on a volume is divided into key ranges, each key range is described in a separate set of volume-information fields. As many sets of volume-information fields as are required to describe allocated space can appear. A pointer to volume information contains a code of 3.
- Password information, which can appear in Data, Index, Path, Cluster and Alternate index records. This group occurrence contains the security information for a data set (or index, etc.). Only one set of password-information fields can appear. A pointer to password information contains a code of 4.

The following group occurrences occur only in Volume catalog records.

- Space map group occurrence. This group occurrence describes each track on the volume as allocated to a VSAM object or unallocated. Each volume record contains as many group occurrences as are necessary to reflect the total space on the volume. A pointer to track allocation information contains a code of 5.
- Data space group occurrence. This group occurrence describes a VSAM data space on the volume. One group occurrence is required to describe each data space and its extents on the volume. A pointer to data-space information contains a code of 6.
- Data set directory entry group occurrence. This group occurrence describes a data set that resides in a VSAM data space. One group occurrence is required for each data set. A pointer to data-set information contains a code of 8.

Association Group Occurrences

These group occurrences identify the relationships between the various records which describe a data set and its associated components.

Each Cluster, Data, Index, Alternate index, Path, or Upgrade Set catalog record can contain one or more association group occurrences, depending on the overall configuration of the data set.

Figure 5.25 shows a possible configuration for a data set which uses all possible association group occurrences. It consists of a key-sequenced data set (base cluster), an alternate index which belongs to the upgrade set, and two paths. Path 1 is defined over the base cluster and serves as an 'alias', while path 2 is defined over the AIX and is used to address the AIX/base cluster combination.

The arrows in the figure represent the 'direction' of the pointers in the group occurrences. For example, the base cluster record and the path 1 record contain pointers to each other. The path 1 record, however, points only to the base index and data records; there is no 'backward' pointer from these records to the path record.

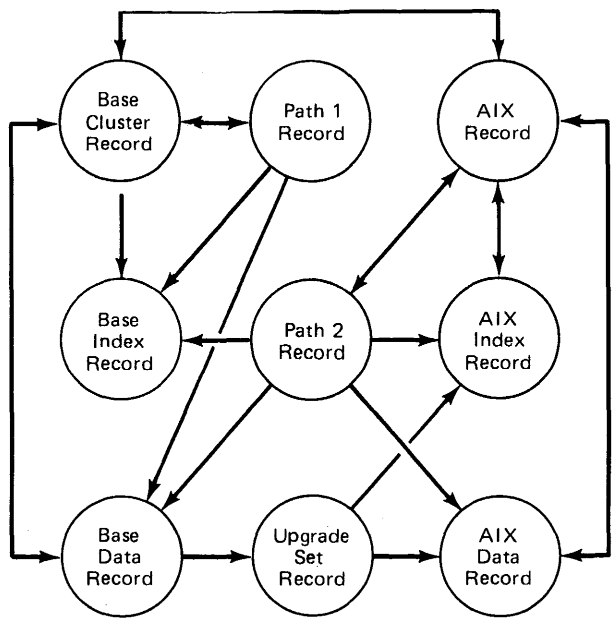


Figure 5.25 **Association group occurrences**

Figure 5.25 shows at least one of each type of association group occurrence. If, however, the data set configuration contains more alternate indexes, paths, etc. there will be correspondingly more association group occurrences which point to the additional components. The base cluster record, for example, will have an association group occurrence pointing to each alternate index record.

Group Occurrence Formats

The following section shows the format of each type of group occurrence.

AMDSB Group Occurrence

The AMDSB group occurrence contains a copy of the AMDSB control block, and is updated each time the data set is closed. This group occurrence is associated with a pointer that contains a group code of 1. Figure 5.26 shows the AMDSB format.

Field Size	Field Name	Description
2		Control information **Byte** **Meaning** 0 Count of the number of variable fields in this group occurrence 1 Hexadecimal displacement to the first variable-length field from the beginning of this group occurrence
96	AMDSBCAT	AMDSB. Only the first 96 bytes of the AMDSB are cataloged. The remainder are ignored. See description and format of AMDSB for detailed information, except for the following special case in the AMDSBs of an AIX. The field at offset 4 of the index AMDSB and of the group occurrence is referenced with the name AMD-NEST and contains the number of entries in an index section. In the data AMDSB group occurrence, this field is called AMDAXRKP and contains the relative key position (in the base cluster record) for this AIX. When the AMDSB group occurrences are transferred to the AMDSBs at OPEN time, the contents of AMDAXRKP are moved to field AMDAIRKP (at offset 118 decimal) and the field at offset 4 (now again called AMDNEST) is loaded with the contents of AMDNEST from the index AMDSB. Note that this movement of data is carried out by OPEN for all AMDSBs, even through the fields may be empty, or already contain the correct information.

Figure 5.26 AMDSB group occurrence format

Association Group Occurrence

The control interval number of a related record is contained in an association group occurrence. This group occurrence is associated with a pointer which contains a group code of 2. Figure 5.27 shows the format.

Field Size	Field Name	Description	
2		Control information	
		Byte	**Meaning**
		0	Count of the number of variable fields in this group occurrence
		1	Hexadecimal displacement to the first variable-length field from the beginning of this group occurrence
1	TYPE	'D'	if this entry describes a Data record
		'I'	if this entry describes an Index record
		'C'	if this entry describes a Cluster record
		'G'	if this entry describes an Alternate Index record
		'R'	if this entry describes a Path record
		'Y'	if this entry describes an Upgrade Set record
3	NAME	Control interval number of the record specified above	
The following fields are present only in an association group occurrence located in an Upgrade Set Record[1]			
1	TYPE2	'I'	This entry can describe only an Index record
3	NAME2		Control interval number of the Index record
1	Each association group occurrence in an Upgrade Set record describes the data component (TYPE, NAME) and the index component (TYPE2, NAME2) of an alternate index.		

Figure 5.27 **Association group occurrence format**

Volume Information Group Occurrence

All extents allocated to the data set, index, or data set's key range on a volume are described by a volume information group occurrence. This group occurrence is associated with a pointer that contains a group code of 3. Figure 5.28 shows its format.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description	
0	0	2	ENTVOL	Control information	
				Byte	**Meaning**
				0	Count of the number of variable fields in this group occurrence
				1	Hexadecimal displacement to the first variable-length field from the beginning of this group occurrence.
2	2	4	DEVTYP	Device type	
6	6	6	VOLSER	Volume serial number	
12	C	2	FILESEQ	File sequence number. (This field is used for nonVSAM data sets that reside on tape volumes.)	

Figure 5.28 **Volume information group occurrence format (part 1 of 2)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
14	E	1	VOLFLG	Volume flags, as follows:
		1...		Prime, which indicates that this volume was allocated when the data set was defined or that a data set that is not divided into parts according to key has been extended to this volume.
		.1..		Candidate, which indicates that this volume is available for use by the data set described by this record.
		..1.		Overflow, which indicates that this volume is being used by a data set that is divided into parts according to key, but this volume was not allocated when the data set was defined.
		...x xxxx		Reserved
The following fields are used only in data and index records				
15	F	1	NOEXTNT	Number of extents allocated in this set of extents on this volume for this data set.
16	10	4	HKRBA	RBA of the data control interval with the high key.
20	14	4	HURBA	High used RBA on this volume
24	18	4	HARBA	High allocated RBA on this volume
28	1C	4	PHYBLKSZ	Block size
32	20	2	NOBLKTRK	Number of blocks per track
34	22	2	NOTRKAU	Number of tracks per control area
36	24	1	ITYPEXT	Flags:
		1...		In an index record: the sequence set is with the data.
		.1..	ARDPRFMT	The extents are not preformatted
		..xx xxxx		Reserved
37	25	2	DSDIRSN	Data-set directory sequence number in the volume control
39	27	Variable	LOKEYV	Low key on the volume. This field can be a maximum of 64 bytes long; the first two bytes indicate the length of the field. For non-KSDS files, this field contains two bytes of zeros.
		Variable	HIKEYV	High key on the volume. This field can be a maximum of 64 bytes long; the first two bytes indicate the length of the field. For non-KSDS files, this field contains two bytes of zeros.
		Variable	EXTENT	The field contains a 2-byte length field, followed by a 20-byte field for each extent. The 20-byte field describes the start and end of the extent, in the form SSCCHHCCHHTTDDDDDDDD, where SS is the data-space sequence number, CCHHCCHH is the low and high cylinder and head, TT is the number of tracks, and DDDDDDDD is the low and high RBA of the extent.

Figure 5.28 **Volume information group occurrence format (part 2 of 2)**

Password Group Occurrence

Password information, if any, is contained in the password group occurrence. This group occurrence is associated with a pointer that contains a group code of 4. Figure 5.29 shows its format.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	2		Control information **Byte** **Meaning** 0 Count of the number of variable fields in this group occurrence 1 Hexadecimal displacement to the first variable-length field from the beginning of this group occurrence.
2	2	32	PASSWORD	Four eight-character passwords, in the following order: MASTER, CONTROL-INTERVAL, UPDATE, and READ.
34	22	8	PASSPRMT	Password prompting code name that allows the operator to provide the correct password without displaying the data-set name.
42	2A	2	PASSATMP	Maximum number of attempts allowed for the operator to provide correct password.
44	2C	8	USVRMDUL	Name of the user's security-verification module, if any.
52	34	Variable	USERAREC	User-authorization record. The first two bytes give the length of the user record (up to 256) which is contained in the following bytes.

Figure 5.29 **Password group occurrence format**

Space Map Group Occurrence

The tracks on a VSAM volume are allocated to a VSAM object, or are unallocated, as described by the Space Map group occurrence. Each bit position describes one track as allocated (bit =0) or unallocated (bit = 1). This group occurrence is associated with a pointer that contains a group code of 5. Figure 5.30 shows its format.

All tracks on the volume are mapped, starting at cylinder 0, track 0. Any tracks not owned by VSAM are marked 'allocated' in order to ensure that they are not used by VSAM data sets.

Field Size	Field Name	Description
2		Control information: **Byte** **Meaning** 0 Count of the number of variable-length fields in this group occurrence (X'01') 1 Hexadecimal displacement to the variable-length field from the beginning of the group occurrence (X'02').
2		Length of the field BITMAP
Variable	BITMAP	Portion of the volume bit map (1 to 440 bytes describing the allocated/unallocated status of 1 to 3520 tracks).

Figure 5.30 **Space map group occurrence format**

Data Space Group Occurrence

Each VSAM data space on the volume is described with a Data Space Group ocurrence. This group occurrence is associated with a pointer that contains a group code of 6. Figure 5.31 shows its format.

Bytes and Bit Pattern	Field Name	Description
2		Control information **Byte Meaning** 0 Count of the number of variable-length fields in this group occurrence (X'00'). 1 Hexadecimal displacement to the first variable-length field from the beginning of this group occurrence (X'55').
8	DSCBTS	Format-1 label timestamp, which indicates when the label was created. The timestamp is part of the name given to the format-1 label.
5	DSCBPTR	CCHHR of the format-1 label
1	SPHDFLG	Data-space flags
1...		Unique data space, that is, this data space contains all or part of only one VSAM object.
0...		Shared data space, that is, this data space contains all or part of two or more VSAM objects.
.1..		Automatically built data space, that is, this data space was built as a result of an end-of-volume request.
..1.		This data space contains a VSAM user catalog.
...1		This data space contains a VSAM master catalog.
.... 1...		This data space was built when the user issued an Access Method Services DEFINE catalog command.
.... .xxx		Reserved
1	NODSPEXT	Number of extents in this data space
1	DSPSOPT	Data-space allocation options
10..		Track request, which indicates that primary space was allocated in terms of tracks.
11..		Cylinder request, which indicates that primary space was allocated in terms of cylinders.
..xx x...		Reserved
.... .111		Class-7 data space.
.... .110		Class-6 data space.
.... .101		Class-5 data space.
.... .100		Class-4 data space.
.... .011		Class-3 data space.
.... .010		Class-2 data space.
.... .001		Class-1 data space.
.... .000		Class-0 data space.
3	DSPSSQ	Secondary space allocation quantity by which space is to be extended if required. This value is taken either from an Access Method Services DEFINE command or from the first data set on this volume that caused space to be used. This field is used only by OS/VS VSAM.
64	SPEXTENT	Sixteen 4-byte extent descriptors in the form TTNN: TT - starting track number of the extent (relative to the beginning of the volume). NN - number of tracks in the extent

Figure 5.31 Data space group occurrence format

Derived Data Space Information

The following field names identify information that is expected, but not contained in the data space group occurrence. The information is derived from fields in the volume catalog record. Figure 5.32 shows its format.

Bytes and Bit Pattern	Field Name	Description
1	SPHDFLG	Data space flags:
..1.		A VSAM user catalog is in this data space
...1		A VSAM master catalog is in this data space
xx.. xxxx		Reserved
2	NODSDSP	Number of data sets in the data space - this information is derived by searching each data and index catalog record (pointed to by Data Set Directory Entry group occurrences and Cluster catalog records) for a volume information group occurrence that contains the volume's serial number. Each group occurrence so identified is searched to determine if the data set or index has been allocated space in one of the data space's extents.
The following field names refer to information about an extent of the data space:		
2	TRKSUSED	Number of allocated tracks in the extent - the Space Map group occurrence is scanned to determine the number of allocated tracks, based on the extent's starting track number and total number of tracks (contained in SPEXTENT).
4	EXTSTART	Cylinder and track on which the extent begins - the extent's TT value (contained in SPEXTENT) is converted to a CCHH value.
2	NOTRKEXT	Number of tracks in the extent - the extent's value (contained in SPEXTENT).
2	SNSPHD	Sequence number of the group occurrence that describes the extent's data space - the sequence number of the Data Space Group occurrence.
Variable	SPACEMAP	A variable-length space map that defines the allocated and unallocated space in the extent - the Space map group occurrence is converted to the format of this variable-length field based on the extent's starting tracks number and total length (contained in SPEXTENT).

Figure 5.32 Derived data space information format

Data Set Directory Entry Group Occurrence

Each data set that resides in a VSAM data space on the volume is described with a Data Set Directory Entry group occurrence. This group occurrence is associated with a pointer that contains a group code of 8. Its format is shown in Figure 5.33.

Field Size	Field Name	Description
2		Control information **Byte** **Meaning** 0 Count of the number of variable fields in this group occurrence (X'00') 1 Hexadecimal displacement to the first variable-length field from the beginning of this group occurrence.
3	DSIDNO	Control-interval number of the Data or Index catalog record that describes this data set or index.

Figure 5.33 Data set directory entry group occurrence format

Derived Data Set Information

The following field names identify information that is expected, but not contained in, the Data Set Directory Entry group occurrence. The information is derived from fields in the volume catalog record. Figure 5.34 shows its format.

Bytes and Bit Pattern	Field Name	Description
1	DSDIRFLG	Flags:
1...		The Data or Index catalog record identifies the volume as a candidate volume -the Data or Index catalog record is searched; its volume information group occurrence VOLFLG field candidate volume indicator (B '.1.. ....' flag) is on.
Variable	DSSPSN	A variable-length collection of 3-byte fields that identify each data space within which the data set has extents allocated to it - this information is obtained by converting each volume information group occurrence's extent descriptor's (EXTENT) SS value (data space extent's sequence number) so that the resulting 3-byte field is: **Byte Meaning** 0-1 Sequence number of the Data Space Group occurrence 2 Number of extents (groups of contiguous tracks) assigned to the data set or index from the data space (limits: 1 - 123).

Figure 5.34 Derived data set information

Field Name Dictionary

The field name dictionary (defined in IGG0CLAY) is an internal data area that provides a map between field names and fields within catalog records, as well as derived information that is not contained in catalog records. The dictionary also allows the dictionary user to specify values (for example, the number of group occurrences to be processed) by associating them with a dictionary name. Using a field name specified in the CTGFL (field parameter list), the catalog management modules reference the field name dictionary for the location, length, and type of fields.

The field name dictionary is a series of 8-byte entries. The first four bytes contain a shortened field name consisting of the first, second, fifth, and sixth character of the field name in IKQCTGFL. Bytes 4-7 of the field name dictionary entry describe the field. When a caller makes a request in a CTGFL, dictionary information is moved from the dictionary to the CTGFL.

Figure 5.35 shows the field name dictionary entry format.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Description	
0		4	Shortened field name: the first, second, fifth, and sixth characters of the eight-character field name.	
4		1	Flags that describe the field.	
			Value	**Meaning**
		xxx.	000	The field is fixed-length and appears in the header portion of a record (before the Extension record pointer).
			001	A combination field name[1]
			010	The field is fixed-length and is part of a group occurrence that follows the Extension record pointer.
			100	The field is variable-length and appears in the header portion of a record (before Extension record pointer).
			110	The field is variable-length and is part of a group occurrence that follows the Extension record pointer.
			011	Special field[2]
			111	Special field[2]
		...x	0	Not a flag field, which means that a CLC (compare logical character) instruction can be used to test this field.
			1	Flag field, which means that a TM (test under mask) instruction can be used to test this field.
		.... x...	0	Not a fixed-length field within a variable-length field in a group occurrence
			1	A fixed-length field within a variable-length field in a group occurrence
		xxx		Reserved
5	5	1	Bytes that identify the location of the field:	
			Type of field name:	**Contents of this byte:**
			Fixed-length:	Displacement in bytes from the
			In the header:	beginning of the record
			In a group occurrence:	beginning of the group occurrence
			In a group of fixed-length fields within a variable-length field:	Length of the group fixed-length fields
			Variable-length:	Zero
			Combination:	Index value in the combination-name index

Figure 5.35 **Field name dictionary entry format (part 1 of 2)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Description	
6	6	1	Bytes that identify the location of the field (continued):	
			Type of field name:	**Contents of this byte:**
			Fixed-length:	Length of the field (in bytes)
			Variable-length:	Sequence number of the field
			Combination:	Number of fields identified by the combination name
			Value	**Meaning**
7	7	x...	0	Not a non-repeating volume entry field
			1	A non-repeating volume entry field (i.e., a header field)
		.xxx xxxx		Group code

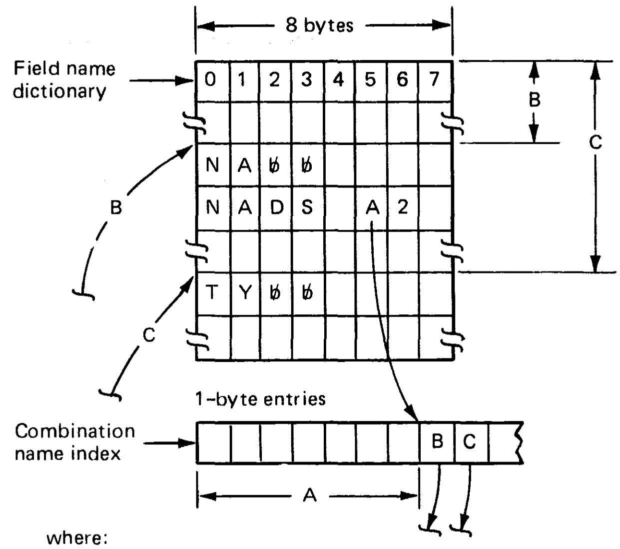
[1] *Combination field name* indicates that the name supplied is a name that allows catalog management to locate a group of related fields.

[2] The field-name dictionary permits catalog management to locate information that is not contained in catalog records.

Figure 5.35 Field name dictionary entry format (part 2 of 2)

Combination names are also included in the field name dictionary. Each combination name allows catalog management to locate more than one field at a time by means of a combination name index.

The following example shows the relationship of the field name dictionary to the combination name index. The three entries in this example are shortened field names for NAME, NAMEDS, and TYPE. Since NAME and TYPE are only four characters in length, the fifth and sixth characters are blanks. The shortened versions, NAbb and TYbb, are combinations of the first, second, fifth and sixth characters of the original field names. NAMEDS is a combination field name that includes fields NAME and TYPE. Thus, byte 5 contains an index value (A) that points to the displacement in the combination name index (B) which contains, in turn, the displacement of the first field name (NAME) in the field name dictionary. Because byte 6 contains a 2 (meaning that this combination name includes two field names), the displacement of the second field name (TYPE) in the field name dictionary is given in the byte C following the first displacement value in the combination name index.



where:

A, B, C	=	displacement values
NAb̸b̸	=	NAMEb̸b̸
NADS	=	NAMEDS
TYb̸b̸	=	TYPEb̸b̸

Figure 5.36 **Example of a combination name in field name dictionary**

Catalog Dictionary Entries

Figure 5.37 identifies the catalog dictionary entries by field name and briefly describes each field.

Field Name	Record Type[1]	Group Code[2]	Description
AMDSBCAT	DI	1	Combination name[4] that includes fields AMDS1, HILIRBA, SSRBA, AMDS2, TIMESTMP, CATSTAT
AMDS1	DI	1	Part of AMDSB
AMDS2	DI	1	Part of AMDSB
ATTR1	DI	-	Data set attributes
ATTR2	DI	-	Data set sharing attributes
BITMAP	V	5	Portion of volume bit map showing the allocated and unallocated tracks on a direct-access volume
BUFSIZE	DI	-	Minimum buffer size
CATACB[3]			Address of catalog ACB
CATSTAT	DI	1	Part of AMDSB
CATVOL	ADIU	3	Combination name[4] that includes fields RELREPNO, DEVTYP, VOLSER, FILESEQ, VOLFLG
CNTREPNO[3]			Maximum number of RELREPNOs to be processed
CRADEVT	ACDGIRUVY	-	CRA device type
CRAIDNO	ACDGIRUVY	-	CRA control interval number
CRAVOL	ACDGIRUVY	-	CRA volume serial number
DATASPAC	V	6	Combination name[4] that includes fields RELREPNO, SPHDFLG, DSCBPTR, DSCBTS, NODSPEXT, DSPSOPT, DSPSSQ, SPEXTENT
DEVTYP	ADIU	3	Device type
DEXTENTS	V	6	Combination name[4] that includes fields RELREPNO, SPHDFLG, NODSPEXT, DSPSOPT, SPEXTENT
DIRECTRY	V	8	Combination name[4] that includes fields RELREPNO, DSIDNO
DSATRO	DI	-	Combination name[4] that includes fields ATTR1, ATTR2, OPENIND
DSATTR	DI	-	Combination name[4] that includes fields ATTR1, ATTR2
DSCBPTR	V	6	CCHHR of F1 label
DSCBTS	V	6	F1 label timestamp
DSDIRECT	V	8	Combination name[4] that includes fields RELREPNO, NODSEXT, DSIDNO, DSDIRFLG, DSSPSN
DSDIRFLG[5]	V	8	Flags
DSDIRSN	DI	3	Data set directory sequence number in the extent descriptor of the volume information group occurrence

Figure 5.37 **Catalog dictionary entries (part 1 of 7)**

Field Name	Record Type[1]	Group Code[2]	Description
DSETCRDT	CDGIR	-	Data set creation date (YDD)
DSETEXDT	CDGIR	-	Data set expiration date (YDD)
DSIDNO	V	8	Control interval number of Data or Index catalog record that describes this data set or index
DSPDSCRP	V	7	Combination name[4] that includes fields RELREPNO, TRKSUSED, EXTSTART, NOTRKEXT, SNSPHD, SPACEMAP
DSPSOPT	V	6	Options for data space creation
DSPSSQ	V	6	Secondary data space allocation quantity
DSSPSN[5]	V	8	Sequence number and number of extents
DSTYPNAM	ACDGIRUVY	-	Combination name[4] that includes fields ENTYPE, ENTIDNO
ENTIDNO	ACDGIRUVY	-	Control interval number of this catalog record
ENTNAME	ACDGIRUVY	-	Data set name (Data, Index, or NonVSAM catalog record), cluster name (Cluster catalog record), volume serial number (Volume catalog record)
ENTVOL	DI	3	Combination name[4] that includes fields RELREPNO, DEVTYP, VOLSER, FILESEQ, VOLFLG, NOEXTNT, HKRBA, HURBA, HARBA, PHYBLKSZ, NOBLKTRK, NOTRKAU, ITYPEXT, DSDIRSN, LOKEYV, HIKEYV, EXTENT
ENTYPE	ACDGIRUVY	-	Catalog record type
EXCPEXIT	DI	-	Exception exit
EXTENT	DI	3	Extent descriptors
EXTSTART[5]	V	7	Cylinder and track on which extent begins
EXTVOL	DI	3	Combination name[4] that includes fields RELREPNO, DEVTYP, ITYPEXT, EXTENT
FATASPAC	V	6	Combination name[4] that includes RELREPNO, DSCBTS, DSCBPTR, SPHDFLG, NODSPEXT, FSPSOPT, FSPSSQ, FPEXTENT
FCONSPAC	DI	-	Fixed block secondary space allocation value
FEXTENTS	V	6	Combination name[4] that includes RELREPNO, SPHDFLG, NODSPEXT, FSPSOPT, FPEXTENT
FILESEQ	ADIU	3	File sequence number for a nonVSAM data set that resides on tape volume(s)

Figure 5.37 **Catalog dictionary entries (part 2 of 7)**

Field Name	Record Type[1]	Group Code[2]	Description
FNTVOL	DI	3	Combination name[4] that includes RELREPNO, DEVTYP, VOLSER, FILESEQ, VOLFLG, NOEXTNT, HKRBA, HURBA, HARBA, PHYBLKSZ, NOBLKTRK, NOTRKAU, ITYPEXT, DSDIRSN, LOKEYV, HIKEYV, FXTENT
FOLEXT	DI	3	Combination name[4] that includes RELREPNO, VOLSER, DSDIRSN, FXTENT
FOLPHY	DI	3	Combination name[4] that includes RELREPNO, VOLSER, VOLFLG, NOEXTNT, HKRBA, HURBA, HARBA, ITYPEXT, LOKEYV, HIKEYV, FXTENT
FOTRKEXT[5]	V	7	Fixed block devices: number of blocks represented by space descriptor CKD devices: number of tracks represented by space descriptor.
FPACEHDR[5]	V	6	Combination name[4] that includes RELREPNO, DSCBTS, DSCBPTR, SPHDFLG, NODSDSP, NODSPEXT, FSPSOPT, FSPSSQ
FPACOPTN	DI	-	Fixed block space options for data set
FPACPARM	DI	-	Combination name[4] that includes FRIMSPAC, FCONSPAC, FPACOPTN
FPEXTENT	V	6	Fixed block data space extent descriptors
FRENTVOL	D	3	Combination name [4] that includes RELREPNO, DEVTYP, VOLSER, FILESEQ, VOLFLG, NOEXTNT, HKRBA, HURBA, HARBA, PHYBLKSZ, NOBLKTRK, NOTRKAU, ITYPEXT, DSDIRSN, LOKEYV, HIKEYV, FXTENT
FRIMSPAC	DI	-	Fixed block primary space allocation value
FRKSUSED[5]	V	7	Fixed block devices: number of tracks used in space descriptor CKD devices: number of tracks allocated per extent
FSPDSCRP[5]	V	7	Combination name[4] that includes RELREPNO, FRKSUSED, FXTSTART, FOTRKEXT, SNSPHD, SPACEMAP
FSPSOPT	V	6	Fixed block space options for data set creation

Figure 5.37 **Catalog dictionary entries (part 3 of 7)**

Field Name	Record Type[1]	Group Code[2]	Description
FSPSSQ	V	6	Fixed block secondary data space allocation value
FXTENT	DI	3	Fixed block extent descriptors for volume group occurrence
FXTSTART[5]	V	7	Fixed block space descriptor for beginning of extent
FXTVOL	DI	3	Combination name[4] that includes RELREPNO, DEVTYP, ITYPEXT, FXTENT
GENDSP[3]			Generated data space name
HARBA	DI	3	High-allocated RBA on this volume for this data set
HARBADS	DI	-	High-allocated RBA of the data set or index (not always the same as HARBA if the data set resides on more than one volume)
HIKEYV	DI	3	Key-sequenced data set's high key value on a volume or, if the data set is divided into key ranges, the key range's high key value
HILIRBA	DI	1	AMDSB high-level index RBA
HKRBA	DI	3	RBA of the record containing the high key of a key-sequenced data set on a volume or, if the data set is divided into key ranges, the key range's high key value
HKURBA	DI	3	Combination name[4] that includes fields HKRBA, HURBA
HURBA	DI	3	High-used RBA on this volume for this data set
HURBADS	DI	-	High-used RBA for the data set or index (not always the same as HURBA if the data set resides on more than one volume)
ITYPEXT	DI	3	Flag
LOKEYV	DI	3	Key-sequenced data set's low key value on a volume, or if the data set is divided into key ranges, the key range's low key value
LRECL	DI	-	Logical record size (data set) or X'FF' (index)
MAPSPACE	V	5	Combination name[4] that includes fields RELREPNO, BITMAP
NAME	CDGIRY	2	Control interval number of associated catalog record
NAMEDS	CDGIRY	2	Combination name[4] that includes fields TYPE, NAME
NAME2	Y	2	Control interval number of associated catalog record
NOBLKTRK	DI	3	Number of blocks per track
NODSDSP[5]	V	6	Number of data sets in this data space
NODSET[5]	V	-	Number of data sets on this volume
NODSEXT[5]	V	8	Number of data set directory extents

Figure 5.37 **Catalog dictionary entries (Part 4 of 7)**

Field Name	Record Type[1]	Group Code[2]	Description
NODSPACE[5]	V	-	Number of data spaces on this volume
NODSPEXT	V	6	Number of extents in this data space
NOEXTNT	DI	3	Number of extents allocated in this set of extents on this volume for this data set
NOTRKAU	DI	3	Number of tracks per allocation unit
NOTRKEXT[5]	V	7	Number of tracks in extent
OPENCNT	DI	-	Open count
OPENIND	DI	-	Open indicator flag
OPNCALL1	DI	-	Combination name[4] that includes fields SPACOPTN, BUFSIZE, ENTNAME, HURBADS
OWNERID	CDGIR	-	Data set owner identification
PASSATMP	CDGIR	4	Maximum number of attempts allowed operator to provide correct password
PASSPRMT	CDGIR	4	Password prompting code name for security verification
PASSWALL	CDGIR	4	Combination name[4] that includes fields PASSWORD, PASSPRMT, PASSATMP, USVRMDUL, USERAREC
PASSWORD	CDGIR	4	Four 8-character password
PHYBLKSZ	DI	3	Physical block size
PRIMSPAC	DI	-	Primary space allocation
RELCRA	ACDGIRUVY	-	Combination name[4] that includes fields RELIND, CRAVOL, CRAIDNO, CRADEVT
RELIND	ACDGIRUVY	-	VSAM release indicator
RELREPNO[3]			Two-byte relative repetition number
REPNO[3]			Two-byte highest repetition number found
RGATTR	GR	-	Path/alternate index attributes
SCONSPAC	DI	-	Secondary space allocation requirement
SNSPHD[5]	V	7	Sequence number of the group occurrence that describes extent's data space
SPACEHDR[5]	V	6	Combination name[4] that includes fields RELREPNO, DSCBTS, DSCBPTR, SPHDFLG, NODSDSP, NODSPEXT, DSPSOPT, DSPSSQ
SPACEMAP[5]	V	7	Data space map that defines allocated and unallocated space in the extent
SPACOPTN	DI	-	Space options flags
SPACPARM	DI	-	Combination name[4] that includes fields PRIMSPAC, SCONSPAC, SPACOPTN
SPEXTENT	V	6	Extent descriptors

Figure 5.37 **Catalog dictionary entries (Part 5 of 7)**

Field Name	Record Type[1]	Group Code[2]	Description
SPHDFLG[5]	V	6	Data space flags
SRENTVOL	D	3	Combination name[4] that includes RELREPNO, DEVTYP, VOLSER, FILESEQ, VOLFLG, NOEXTNT, HKRBA, HURBA, HARBA, PHYBLKSZ, NOBLKTRK, NOTRKAU, ITYPEXT, DSDIRSN, LOKEYV, HIKEYV, EXTENT
SSRBA	DI	1	AMDSB sequence-set RBA
SYSEXTDS	V	-	Number of system-allowed extents per suballocation request
TIMESTMP	DI	1	AMDSB time stamp
TRBAEXT[3]	DI	3	This is a test RBA for EOV mount by RBA
TRKSUSED[5]	V	7	Number of allocated tracks per extent
TYPE	CDGIRY	2	Associated catalog record type
TYPE2	Y	2	Associated catalog record type
UPDVOL	DI	3	Combination name[4] that includes fields RELREPNO, DEVTYP, VOLSER, FILESEQ, VOLFLG, NOEXTNT, HKRBA, HURBA, HARBA, PHYBLKSZ, NOBLKTRK, NOTRKAU, ITYPEXT, DSDIRSN, LOKEYV, HIKEYV
UPGRADE	Y	2	Combination name[4] that includes fields TYPE, NAME, TYPE2, NAME2
USERAREC	CDIGR	4	User authorization record
USERINFO	DI	-	User information for ISAM Interface Program
USVRMDUL	CDIGR	4	User security verification module name
VOLDEV	DI	3	Combination name[4] that includes fields DEVTYP, PHYBLKSZ, NOBLKTRK, NOTRKAU
VOLDVCHR	V	-	Device characteristics
VOLEXT	DI	3	Combination name[4] that includes fields RELREPNO, VOLSER, DSDIRSN, EXTENT
VOLFLG	ADIU	3	Volume flags
VOLPHY	DI	3	Combination name[4] that includes fields RELREPNO, VOLSER, VOLFLG, NOEXTNT, HKRBA, HURBA, HARBA, ITYPEXT, LOKEYV, HIKEYV, EXTENT
VOLRFLG	V	-	Volume record flags
VOLSER	ADIU	3	Volume serial number
VOLTSTMP	V	-	Volume time stamp

Figure 5.37 **Catalog dictionary entries (Part 6 of 7)**

[1] Record type indicates which catalog record contains the field name:

- A - non-VSAM
- C - Cluster
- D - Data
- G - Alternate index
- I - Index
- R - Path
- U - User catalog
- V - Volume
- Y - Upgrade set

[2] Group code indicates which group occurrence contains the field:

- \- Field is in the header portion of the record, not associated with any group occurrence
- 1 AMDSB group occurrence
- 2 Association group occurrence
- 3 Volume information group occurrence
- 4 Password group occurrence
- 5 Space map group occurrence (contained only in a Volume catalog record)
- 6 Data space group occurrence (contained only in a Volume catalog record)
- 7 Space descriptor group occurrence (derived)
- 8 Data set directory entry group occurrence (contained only in a Volume catalog record)

[3] Special names not contained in the catalog; their values are derived from other sources, in most cases to locate repetitive data and control processing.

[4] Combination names allow catalog management to locate more than one field at a time

[5] These field names identify information that is expected, but not contained in, the data space or data set directory group occurrences. The information is derived from fields in the Volume catalog record.

Figure 5.37 Catalog dictionary entries (Part 7 of 7)

To clarify the use of the dictionary as a means of gaining access to catalog information, refer to the examples that follow.

Dictionary Example 1

The DSETCRDT (data set creation date) field appears in the dictionary, as follows:

C4E2C3D900650300

The first 00 is the fifth-byte value of the record; it indicates that DSETCRDT is (a) a fixed-length field, (b) not part of a group occurrence, and (c) not a flag field.

The 65 is the sixth-byte value of the record; it indicates that DSETCRDT is at displacement X'65' from the beginning of the record in which it appears.

The 03 is the seventh-byte value of the record; it indicates that DSETCRDT is three bytes long.

The 00 is the eighth-byte value of the record; it is zero because DSETCRDT is not part of a group occurrence and, therefore, is not associated with a group occurrence code.

Although the dictionary entries are 8 bytes long, only the first, second, fifth, and sixth characters of the field name appear. For DSETCRDT, only DSCR (C4E2C3D9) appears.

Dictionary Example 2

The DSPSOPT (data-space-creation space options) field appears in the dictionary, as follows:

C4E2D6D750130106

The 50 is the fifth-byte value of the record; it indicates, when converted to binary, that DSPSOPT is (a) a fixed-length field that is part of a group occurrence, (b) a flag field, and (c) not a repeating field within a variable-length field.

The 13 is the sixth-byte value of the record; it indicates that DSPSOPT is at displacement X'13' from the beginning of the record in which it appears.

The 01 is the seventh-byte value of the record; it indicates that DSPSOPT is one byte long.

The 06 is the eighth-byte value of the record; it indicates that DSPSOPT is part of a group occurrence associated with a code of 6, which means that it is part of a group occurrence that contains VSAM data-space information.

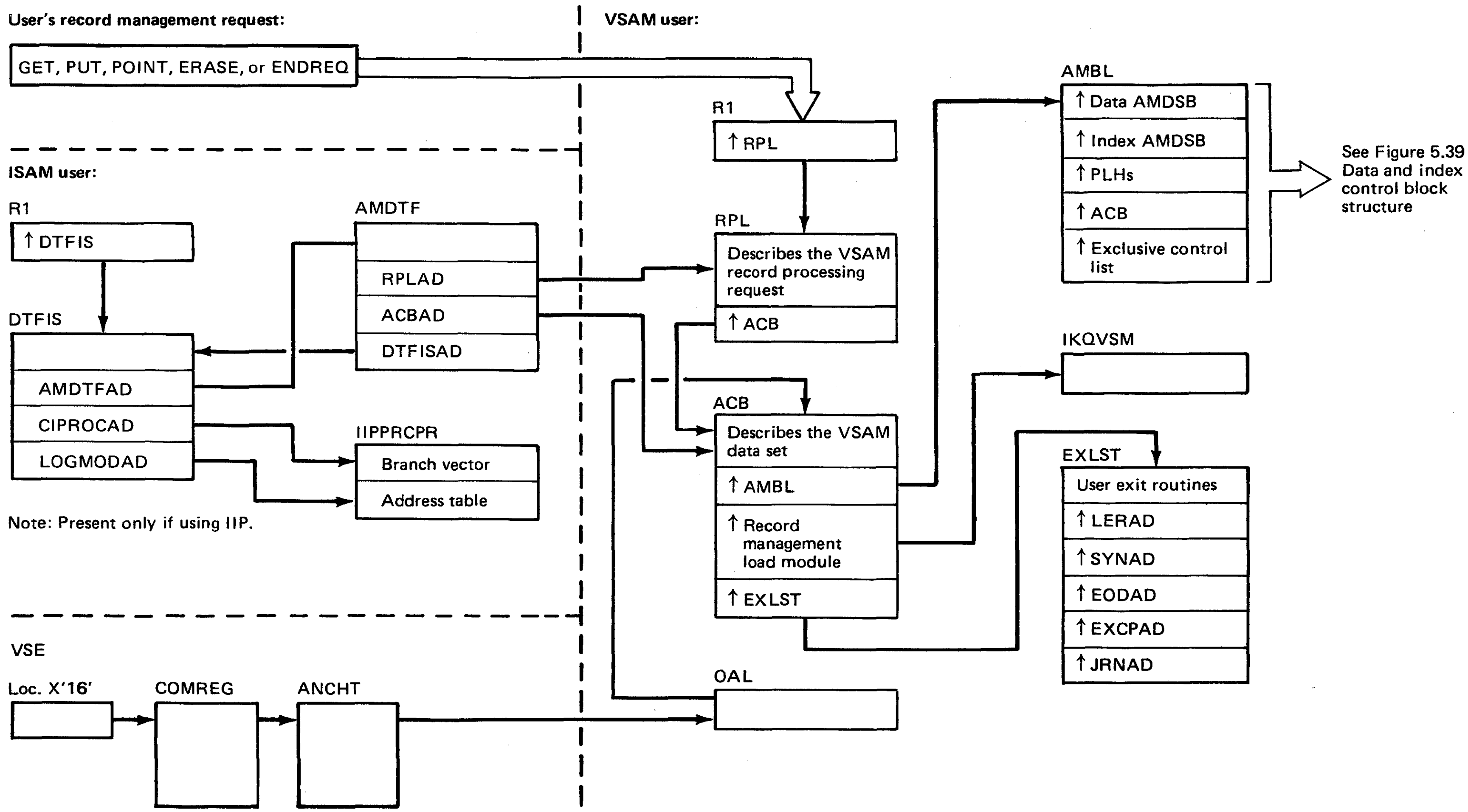


Figure 5.38 **VSAM control block structure for a key-sequenced data set**

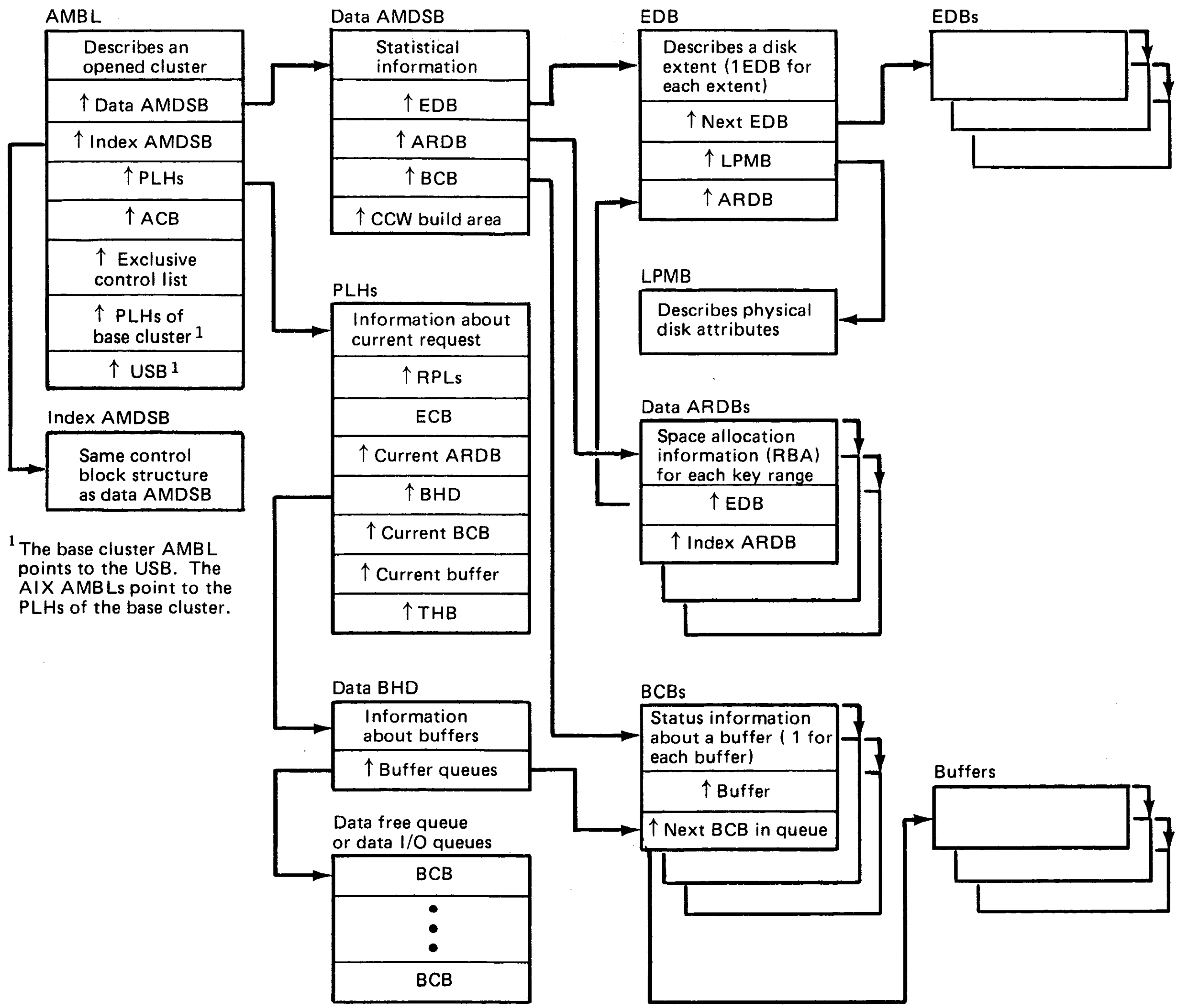
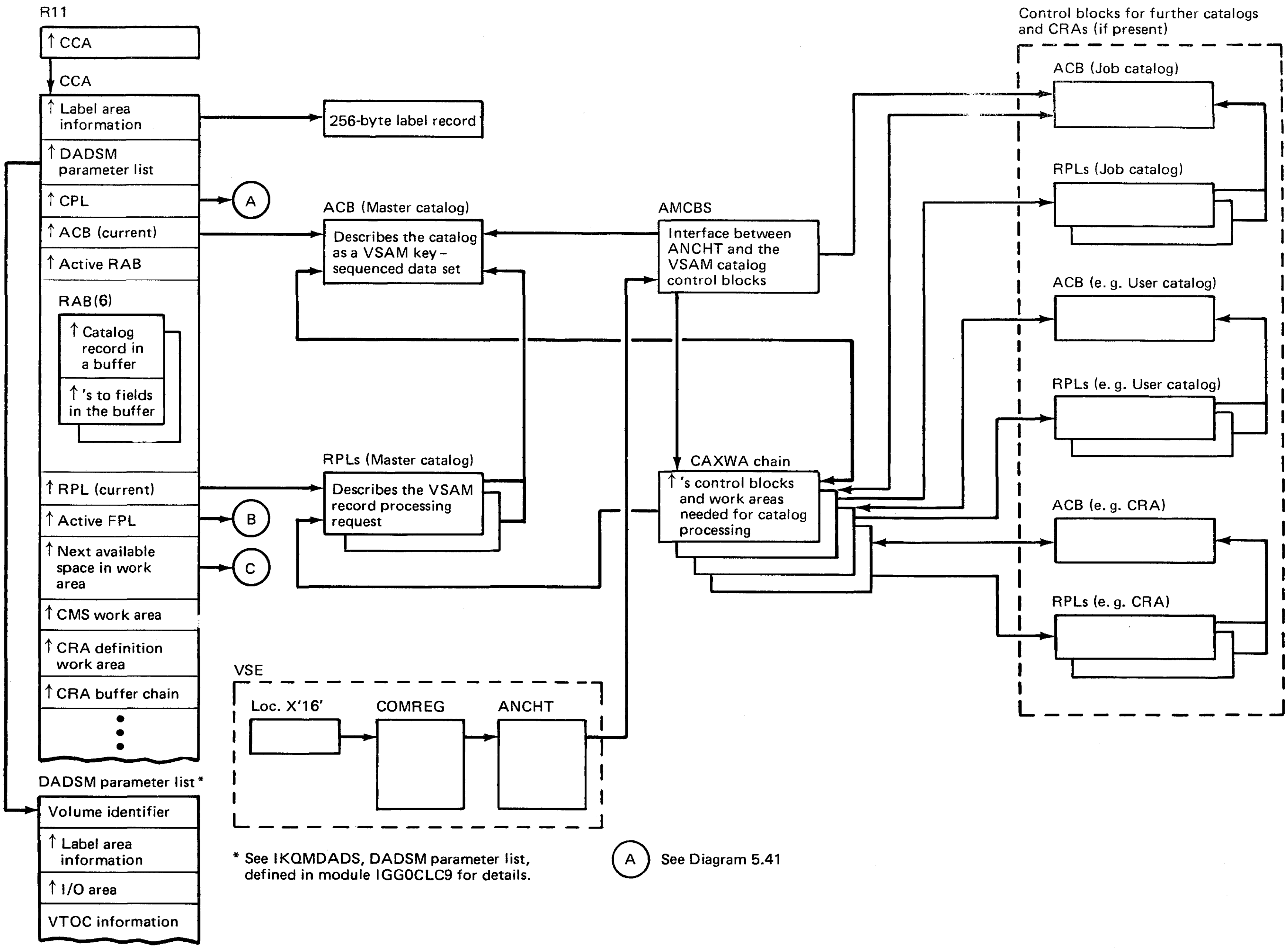


Figure 5.39 **Data and index control block structure**

Figure 5.40 Catalog management control blocks



* See IKQMDADS, DADSM parameter list, defined in module IGG0CLC9 for details.

A See Diagram 5.41

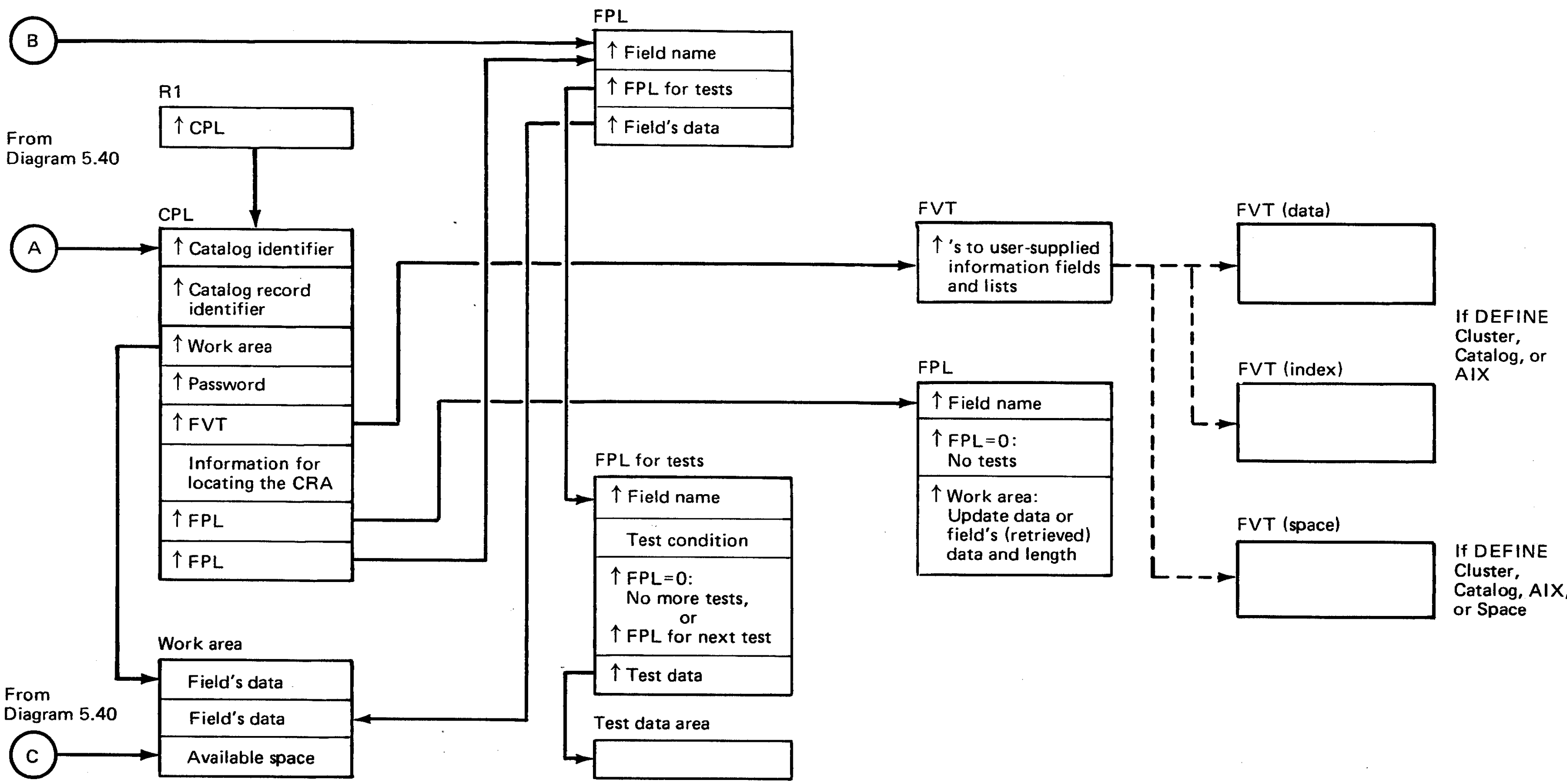


Figure 5.41 Caller-supplied control blocks for catalog management

Control Block Description and Format

Access Method Control Block (ACB)

The VSAM ACB describes a VSAM cluster. It is built by the user's program. The ACB points to the Exit List (EXLST). After the cluster is opened, the ACB is pointed to by the RPL (RPLDACB) that describes the user's record processing request. The ACB also describes the processing options, password, and I/O buffer space applicable to the user's program.

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
0	0	0	ACBST		
0	0	1	ACBID		ACB identifier = 'A0'
			ACBIDD	X'A0'	ACB equate
			ACBIDVAL	X'A0'	ACB equate
1	1	1	ACBSTYP		Release indicator
			ACBSDV1	X'00'	DOS/VS VSAM Release 1
			ACBSVSE1	X'10'	VSE/VSAM Release 1
				X'20'	VTAM
2	2	2	ACBLEN		Length of ACB in bytes
2	2	2	ACBLENG		Length of ACB in bytes[1]
4	4	4	ACBAMBL		Address of the AMBL
8	8	4	ACBAMO		Pointer to VSAM code
12	C	1	ACBACT		ACB active byte (X'FF'=ACB is active).
13	D	1	ACBINFLG		Catalog recovery flags
			ACBSCRA	X'80'	ACB is for a system-initiated OPEN of the CRA
			ACBUCRA	X'40'	ACB is for a user-initiated OPEN of the CRA
				X'20'	Reserved for CRA
				X'10'	Reserved for CRA
			ACBSTSKP	X'08'	Skip updating of statistics
				X'04'	Reserved for CMS
				X'02'	Reserved for CMS
				X'01'	Reserved for CMS
14	E	2	ACBDBUF		Number of data buffers
14	E	2	ACBBUFND		Number of data buffers
16	10	2	ACBIBUF		Number of index buffers
16	10	2	ACBBUFNI		Number of index buffers
18	12	2	ACBMACRF		MACRF
18	12	1	ACBMACR1		MACRF first byte
			ACBKEY	X'80'	Access data via index or relative record number
			ACBADD	X'40'	Access via RBA
			ACBADR	X'40'	Access via RBA
			ACBCNV	X'20'	Control interval processing
			ACBSEQ	X'10'	Sequential processing
			ACBDIR	X'08'	Direct processing
			ACBIN	X'04'	GET
			ACBOUT	X'02'	PUT
			ACBUBF	X'01'	User buffers

[1] If specified length is too small for a VSE/VSAM Release 2 ACB, a Release 1 ACB is built (X'00' in byte 1).

Figure 5.42 Access Method Control Block (ACB) description and format (part 1 of 6)

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
19	13	1	ACBMACR2		MACRF second byte
			ACBLSR	X'80'	Local shared resources
			ACBDFR	X'40'	Defer writing of buffers
			ACBSKP	X'20'	Skip sequential access
			ACBRST	X'10'	Reusable data set
			ACBAIX	X'08'	AIX processing only
			ACBJRACT	X'02'	JRNAD exit active
20	14	1	ACBDOSID		DOS DTF identifier
			ACBDTFID	X'28'	DTF type for VSAM
21	15	1	ACBOFLGS		Open/close flags
			ACBVOLMT	X'80'	Verify volume mounted
			ACBVMSG	X'40'	Message requested bit
			ACBEOV	X'20'	EOV detects completed
			ACBOPEN	X'10'	ACB is open
			ACBCAT	X'08'	ACB for VSAM catalog
			ACBEXFG	X'04'	User exit flag
			ACBSUB	X'02'	ACB is suballocated (is located in a control block allocation unit)
			ACBKEYOK	X'01'	Key processing all right for this ACB
22	16	1	ACBNST		Number of strings
22	16	1	ACBSTRNO		Number of strings
23	17	1	ACBERFLG		Error flags
				Open error return codes:	
			ACBOINCB	X'02'	Invalid control block structure
			ACBOALR	X'04'	This ACB is already open.
			ACBOLLUB	X'0E'	The symbolic unit in the DLBL statement is invalid.
			ACBONJIB	X'0F'	No job information blocks (JIBs) are available from the label information cylinder.
			ACBOLIGN	X'11'	The address in the ASSGN statement for the logical unit was IGN (assignment ignored).
			ACBOLUNA	X'12'	The address in the ASSGN statement for the logical unit was UA (logical unit unassigned).
			ACBOAASF	X'13'	Unable to automatically assign a logical unit number
			ACBOIDSP	X'20'	The OPEN disposition specified for the file conflicts with other file characteristics.
			ACBOCEXT	X'22'	The volume serial numbers specified in the EXTENT statement do not match those specified in the catalog entry.
			ACBONOAL	X'28'	No space available on any volume for primary allocation of a dynamic file
			ACBONANR	X'30'	An attempt was made to open a NOALLOCATION file which is not reusable.

Figure 5.42 **Access Method Control Block (ACB) description and format (part 2 of 6)**

Offset Dec	Hex	Bytes	Field Name	Hex. Digit	Description
			ACBOCDLD	X'32'	Unable to load VSAM modules via a CDLOAD macro instruction.
			ACBONCIF	X'40'	An attempt was made to open a NOCIFORMAT file using VSAM (ACB) access.
			ACBOSENS	X'41'	An attempt was made to open a SAM ESDS without the VSE/VSAM space Management for SAM Feature installed.
			ACBOIRCZ	X'42'	An attempt was made to open a DTF whose file characteristics do not match the file characteristics of the VSAM catalog.
			ACBOUEXP	X'43'	An attempt was made to open an unexpired file for output using a DTF.
			ACBODMOD	X'44'	The file to be opened has a name which begins with an invalid prefix.
			ACBONSDS	X'45'	An attempt was made to open a nonSAM ESDS file using a DTF.
			ACBOBNAM	X'46'	An invalid file-id was detected during implicit define or implicit delete.
			ACBORCSZ	X'47'	Allocation specifications for implicit define conflict with the file characteristics specified in the DTF and conversion to correct the conflict was unsuccessful.
			ACBONALC	X'48'	The file-id specified in your DLBL statement was not found in the catalog and insufficient allocation information is specified for an implicit define.
			ACBOIDEL	X'4E'	A catalog management error was detected during implicit delete.
			ACBOIDEF	X'4F'	A catalog management error was detected during implicit define.
			ACBONMNT	X'50'	Attempt to mount two volumes on the same drive when direct or keyed processing was specified. Or the operator failed to mount the volume.
			ACBONCRA	X'5C'	CRA volume not mounted
			ACBOIERR	X'60'	Unusable input data set
			ACBOUEMP	X'64'	Empty upgrade AIX
			ACBOTMST	X'68'	The time stamp of the volume on which a data set is stored doesn't match the system time stamp in the volume catalog entry.

Figure 5.42 **Access Method Control Block (ACB) description and format (part 3 of 6)**

Offset Dec	Hex	Bytes	Field Name	Hex. Digit	Description
			ACBOTIME	X'6C'	The system time stamps of a data set and its index do not match. This indicates that the data has been updated separately. This test is greater than or equal, i.e., no warning is given if the index time stamp is greater than the data time stamp.
			ACBOEMPT	X'6E'	Open empty data set for read only.
			ACBODSNC	X'74'	Data set was not closed the last time it was processed.
			ACBODEVT	X'75'	The symbolic unit specified in the EXTENT statements is not a valid VSAM device type, or invalid extents are specified.
			ACBONDLB	X'80'	The DLBL statement is missing or the filename in the DLBL doesn't match the ACB.
			ACBOIOER	X'84'	A permanent I/O error occurred while VSAM was reading label information from the label information cylinder.
			ACBONVRT	X'88'	Not enough virtual storage space is available in the partition for work areas, control blocks, or buffers.
			ACBOIOCA	X'90'	A permanent I/O error occurred while VSAM was reading or writing a catalog entry.
			ACBONCAT	X'94'	No entry was found in the catalog for this ACB.
			ACBOSECU	X'98'	Security verification failed; the password specified in the ACB for a specific level of access doesn't match the password in the catalog for that level of access.
			ACBOPARC	X'A0'	The operands specified in the ACB are inconsistent with each other or with the information in the catalog entry, for example, an open of an ESDS for keyed processing.
			ACBOKBUF	X'A1'	User-specified buffers with keyed access (user buffers can be specified only with CNV access).
			ACBOLIOE	X'A5'	A permanent I/O error was detected on the system lock file
			ACBOLTEX	X'A6'	The system lock table is not large enough to accommodate the concurrent requests.
			ACBOLFEX	X'A7'	The system lock file is not large enough to accommodate the concurrent requests.

Figure 5.42 **Access Method Control Block (ACB) description and format (part 4 of 6)**

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
			ACBONAVA	X'A8'	The data set is not available because it is being updated by (under the exclusive control of) another ACB or has been exported by Access Method Services.
			ACBONOCT	X'B4'	The VSAM catalog is not connected to the system on logical unit SYSCAT, or insufficient virtual storage available for OPEN.
			ACBOACT	X'BC'	ACB was active.
			ACBOOERR	X'C0'	Unusable output data set
			ACBOPEMP	X'C4'	Access via empty path
			ACBOLEMP	X'D4'	LSR is specified but the data set being opened is empty (which implies that it is to be loaded).
			ACBOLKEY	X'D8'	LSR is specified but the key length of the data set being opened is greater than the maximum key length specified for the resource.
			ACBOLBUF	X'DC'	LSR is specified but the CI size of the data set being opened is greater than the largest buffer size specified for the resource pool.
			ACBOLNRP	X'E4'	LSR is specified but there is no resource pool defined; may also be caused by problems while loading the resource table.
			ACBONRST	X'E8'	Non-reusable file is not empty.
			ACBOILAB	X'F8'	IKQLAB internal error
			ACBOLUNX	X'FE'	OPEN detected an unexpected return code from the Lock Manager
			ACBOCTER	X'FF'	Unexpected return from catalog locate function.
				Close error return codes	
			ACBOINCB	X'02'	Invalid control block, or ACB address not in OAL
			ACBCALR	X'04'	ACB already closed
			ACBCDSFL	X'4C'	CLOSE disposition failed
			ACBCNVRT	X'88'	Insufficient space available in user's partition for work area.
			ACBCIOCA	X'90'	Permanent I/O error occurred while VSAM was reading or writing a catalog entry.
			ACBCNCAT	X'94'	No catalog entry found
			ACBCIOER	X'B8'	Permanent I/O error occurred while VSAM was completing outstanding I/O requests.
			ACBCBUSY	X'BC'	ACB busy.

Figure 5.42 **Access Method Control Block (ACB) description and format (part 5 of 6)**

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
			ACBCDTFA	X'FC'	Automatic close of the DTF for a managed-SAM file failed
24	18	4	ACBAMBUF		Length of buffer pool
28	1C	8	ACBDDNM		DDname
36	24	4	ACBPRTCT		Pointer to password
40	28	4	ACBUAPTR		Pointer to user work area, or to CAXWA if ACB is for a catalog.
44	2C	4	ACBBFPL		Pointer to first data buffer in buffer pool
48	30	4	ACBEXLST		User exit list pointer
52	34	4	ACBBPLS		BAM parameter list pointer
56	38	1			Reserved
57	39	1	ACBOFLG2		Second OPEN/CLOSE flag byte
				X'80'	Reserved
				X'40'	Reserved
				X'20'	Reserved
				X'10'	Reserved
			ACBKEEP	X'08'	Close disposition is KEEP
			ACBDELET	X'04'	Close disposition is DELETE
			ACBDATE	X'02'	Close disposition is controlled by the expiration date
				X'01'	Reserved
58	3A	2	ACBMSGLN		Message area length
60	3C	4	ACBMSGAR		Message area
64	40	4	ACBNMPTR		Pointer to 44 character name
68	44	0	ACBEND		End of ACB

Figure 5.42 **Access Method Control Block (ACB) description and format (part 6 of 6)**

Access Method Block List (AMBL)

The AMBL describes a VSAM cluster and points to the cluster's data set and index AMDSBs. When the cluster is opened, an AMBL is built to describe the cluster. AMDSBs are built to describe the cluster's data set and, if the cluster is key-sequenced, to describe the index. The AMBL is pointed to by the cluster's ACB (ACBAMBL).

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
0	0		AMBLST		Beginning of AMBL
0	0	1	AMBLID	X'11'	AMBL identifier
1	1	1	AMBLACT		AMBL active byte (X'FF'=AMBL is active)
2	2	2	AMBLLEN		Length of control block
4	4	4	AMBLDTA		Pointer to data AMDSB
8	8	4	AMBLIX		Pointer to index AMDSB
12	C	4	AMBLPLHF		Pointer to first PLH*
16	10	4	AMBCHAIN		Reserved
20	14	4	AMBLACB		Pointer to ACB
24	18	2	AMBLPLHL		Length of PLH*
26	1A	1	AMBLPLHN		Total number of strings*
27	1B	1	AMBLFLAG		Flag byte
			AMBLPOST	X'80'	POST must be issued
28	1C	4	AMBAMBUF		Size of buffer space
32	20	2	AMBMACRF		Flags (copy of flags in ACBMACR1 and ACBMACR2)
32	20	1	AMBMACR1		*First byte*:
			AMBKEY	X'80'	Access data via index or relative record number
			AMBADD	X'40'	Access via RBA
			AMBADR	X'40'	Access via RBA
			AMBCNV	X'20'	Control interval processing
			AMBSEQ	X'10'	Sequential processing
			AMBDIR	X'08'	Direct processing
			AMBIN	X'04'	GET, READ processing
			AMBOUT	X'02'	PUT, WRITE processing
			AMBUBF	X'01'	User buffers

* When LSR is active, AMBLRPHD*0, and AMBLSR in AMBMACR2 is set on. In this case, the fields indicated by the asterisk refer to the PLH pool.

Figure 5.43 Access Method Block List (AMBL) description and format (part 1 of 3)

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
33	21	1	AMBMACR2		*Second byte:*
			AMBLSR	X'80'	Local shared resources
			AMBDFR	X'40'	Defer writing of buffers
			AMBSKP	X'20'	Skip sequential accessing
			AMBRST	X'10'	Reusable Data Set
			AMBAIX	X'08'	AIX processing
			AMBJRACT	X'02'	JRNAD exit enabled
			AMBOPEN	X'01'	Open is in process
34	22	2	AMBLTLEN		Length of GETVIS for close work area
36	24	2	AMBDBUF		Number of data buffers
38	26	2	AMBIBUF		Number of index buffers
40	28	4	AMBLOPWA		Pointer to open work area
				Split Control	
44	2C	4	AMBSECB		Split/pseudo-split ECB
			AMBSRCL	X'80'	IKQRCL00 set split lock
				X'40'	Reserved
				X'20'	Reserved
				X'10'	Reserved
				X'08'	Reserved
				X'04'	Reserved
				X'02'	Reserved
				X'01'	Reserved
45	2D	1			Reserved
46	2E	1	AMBSCOM		ECB post byte-split
			AMBSWAIT	X'80'	Wait bit-split
				X'40'	Reserved
				X'20'	Reserved
				X'10'	Reserved
				X'08'	Reserved
				X'04'	Reserved
				X'02'	Reserved
				X'01'	Reserved
47	2F	1	AMBSECBT		Test-and-set byte - split
48	30	4	AMBBECB		ECB for IKQRQC to ensure that only one string (at a time) extends the chain of base cluster RPLs during path processing in an LSR environment.
48	30	2			Reserved
50	32	1	AMBBCOM		ECB Post byte - buffer
			AMBBWAIT	X'80'	Wait bit-buffer manager
				X'40'	Reserved
				X'20'	Reserved
				X'01'	Reserved
				X'08'	Reserved
				X'04'	Reserved
				X'02'	Reserved
				X'01'	Reserved
51	33	1	AMBBECBT		Test-and-set byte - buffer
52	34	8	AMBRBAS		RBAs for split locking
52	34	4	AMBLORBA		Low RBA of control area being split
56	38	4	AMBHIRBA		High RBA of control area being split
60	3C	1	AMBSTRID		ID of string which holds control area.

Figure 5.43 **Access Method Block List (AMBL) description and format (part 2 of 3)**

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
61	3D	1	AMBOCSW		OPEN/CLOSE switches
			AMBLAUTO	X'80'	A dynamic volume list was built for this ACB
				X'40'	Reserved
				X'20'	Reserved
				X'10'	Reserved
				X'08'	Reserved
				X'04'	Reserved
				X'02'	Reserved
				X'01'	Reserved
62	3E	2			Reserved.
64	40	4	AMBPLH		Address of PLH in control
				Pointers	
68	44	4	AMBALIST		Executive control list address
72	48	4	AMBLRPLS		Address of RPL causing split
76	4C	4	AMBLCLWA		Pointer to close work area
80	50	4	AMBLCIWA		Pointer to CI split work area
84	54	4	AMBLBC		Pointer to base cluster PLH pool
88	58	4	AMBLUSB		Pointer to USB
92	5C	4	AMBBCACB		Pointer to base cluster ACB
96	60	4	AMBPEACB		Pointer to path entry ACB
100	64	4	AMBLRPHD		Pointer to resource pool header
104	68	4	AMBDECB		ECB for duplicate data recovery
104	68	2			Reserved
106	6A	1	AMBDCOM		ECB post byte - duplicate data recovery
			AMBDWAIT	X'80'	Traffic bit
				X'40'	Reserved
				X'20'	Reserved
				X'10'	Reserved
				X'08'	Reserved
				X'04'	Reserved
				X'02'	Reserved
				X'01'	Reserved
107	6B	1	AMBDECBT		Test-and-set byte - duplicate data recovery
108	6C	0	AMBLSHRW		Pointer to file sharing work area

Figure 5.43 **Access Method Block List (AMBL) description and format (part 3 of 3)**

Access Method Control Block Structure (AMCBS)

The AMCBS is pointed to by the Anchor Table (ANCHT) and, in turn, points to the VSAM catalog's ACB and CAXWA.

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
0	0	1	CBSID	X'00'	AMCBS identifier
1	1	1	CBSFLAGS		AMCBS flags
			CBSJCAT	X'80'	Job catalog not present
2	2	2	CBSSIZ		Length of the AMCBS
4	4	4	CBSCRACB		Pointer to CRA ACB
8	8	4	CBSACB		Pointer to ACB (Master)
12	C	4	CBSCRAPL		Pointer to AMS CRA parameter list.
16	10	4	CBSSYSUC		Pointer to job catalog ACB
20	14	4	CBSCAXCN		Pointer to CAXWA chain
24	18	4	CBSCBMM		Pointer to Control Block Manipulation Macro load module

Figure 5.44 **Access Method Control Block Structure (AMCBS) description and format**

Access Method Data Statistics Block (AMDSB)

The AMDSB contains statistical information about record processing in the data set. It also contains some of the data set's attributes and specifications. The AMDSB is built, using the data index or index catalog record's AMDSB group occurrence, when the cluster is opened. The AMBL (AMBLDTA/AMBLIX) points to the data and index AMDSBs.

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
				General	
0	0		AMDCOMM		Common part
0	0	1	AMDSBID	X'60'	AMDSB identifier
1	1	1	AMDATTR		Attributes of the data set
			AMDATTR1		Attributes (first byte):
			AMDDST	X'80'	Key-sequenced data set
			AMDWCK	X'40'	Check each record when it is written
			AMDSDT	X'20'	Sequence set is stored with the data
			AMDREPL	X'10'	Replication
			AMDORDER	X'08'	Use the volumes in the same order as in the volume list
			AMDRANGE	X'04'	The data set is divided into key ranges
			AMDRRDS	X'02'	Relative record data set
			AMDSPAN	X'01'	Spanned records
2	2	2	AMDLEN		Length of AMDSB in the catalog
4	4	2	AMDNEST		Number of entries in an index section (in all cases except AMDSB group occurrence in data record of AIX)[1]
4	4	2	AMDAXRKP		Relative key position in base record (only for AMDSB group occurrence in data record of a AIX)[1]
6	6	2	AMDRKP		Relative key position
8	8	2	AMDKEYLN		Key length
10	A	1	AMDPCTCA		Percentage of free control intervals in the control area
10	A	1	AMDRCFRM		SAM ESDS record format information
			AMDIMPDF	X'80'	File definition was by implicit define
			AMDNCIFT	X'20'	Non-control-interval format (processable by SAM only)
			AMDNCAFT	X'01'	Non-control-area format (This bit indicates the file is a SMA ESDS. If both this bit and AMDNCIFT are off, and file is a VSAM ESDS.)
			AMDSBLKD	X'04'	The SAM record format is blocked
			AMDSVAR	X'02'	The SAM record format is variable length records
			AMDSFIXD	X'01'	The SAM record format is fixed length records
11	B	1	AMDPCTCI		Percentage of free bytes in the control interval
12	C	2	AMDCIPCA		Number of control intervals in a control area
14	E	2	AMDFSCA		Number of free control intervals in a control area

[1]For more details of these fields, see the explanation of the AMDSB group occurrence.

Figure 5.45 **Access Method Data Statistics Block (AMDSB) description and format (part 1 of 4)**

Offset Dec	Hex	Bytes	Field Name	Hex. Digit	Description
16	10	4	AMDFSCI		Number of free bytes in a control interval
20	14	4	AMDCINV		Control interval size
24	18	4	AMDLRECL		Maximum record size. For a SAM ESDS, this is the maximum SAM logical blocksize
28	1C	4	AMDHLRBA		RBA of the high-level index record
28	1C	4	AMDNSLOT		Number of relative record slots
28	1C	4	AMDBLREC		SAM LRECL, for fixed-blocked SAM ESDS
32	20	4	AMDSSRBA		RBA of first sequence set record
32	20	4	AMDMAXRR		Max. relative record number
36	24	4	AMDPARDB		Pointer to first ARDB
40	28	1	AMDATTR3		Attributes
			AMDUNQ	X'80'	Non-unique keys in AIX
				X'00'	Unique keys in AIX
41	29	3			Reserved
44	2C	4			Reserved
				Statistics	
48	30		AMDSTAT		Statistics
48	30	8	AMDSTMST		System time stamp
48	30	8	AMDSTSP		System time stamp
56	38		AMDSTAT1		
56	38	2	AMDNIL		Number of index levels
58	3A	2	AMDNEDB		Number of EDBs
58	3A	2	AMDNEXT		Number of extents in the data set
60	3C	4	AMDNLR		Number of user-supplied (logical) records in the data set
64	40	4	AMDDELR		Number of deleted records
68	44	4	AMDIREC		Number of inserted records
72	48	4	AMDUPR		Number of updated records
76	4C	4	AMDRETR		Number of retrieved records
80	50	4	AMDASPA		Number of bytes of free space in the data set
84	54	4	AMDNCIS		Number of times a control interval was split
88	58	4	AMDNCAS		Number of times a control area was split
92	5C	4	AMDEXCP		Number of times EXCP was issued by VSAM I/O routines

Figure 5.45 **Access Method Data Statistics Block (AMDSB) description and format (part 2 of 4)**

Offset Dec	Hex	Bytes	Field Name	Hex. Digit	Description
				General Continue	
96	60	1	AMDSHOPT		Share option byte
			AMDSHR1	X'80'	Share option 1
			AMDSHR2	X'40'	Share option 2
			AMDSHR3	X'20'	Share option 3
			AMDSHR4	X'10'	Share option 4
97	61	4	AMDCDSN		Pointer to catalog ACB
101	65	3	AMDDSN		Catalog control interval number for data (index)
104	68	4	AMDHWRBA		High-water RBA for the data set
108	6C	1	AMDATTR2		Attributes (second byte):
			AMDREL	X'80'	Release unused space
			AMDLOAD	X'40'	Load mode
			AMDSPEED	X'20'	Speed option
			AMDINDX	X'10'	Index option
			AMDSHR	X'08'	Sharing
			AMDKR	X'04'	Key-range processing, duplicate of AMDRANGE
			AMDMXARC	X'02'	This component contains both fixed block and CKD files (set only when a mixed architecture index opens itself).
			AMDCAT	X'01'	AMDSB for catalog
109	6D	1	AMDACT		AMDSB test and set byte
110	6E	2	AMDFILT		User area (ISAM compatibility)
112	70	4	AMDPVOL		Pointer to volume list
116	74	1	AMDAMS		AMS flag byte
			AMDAIX	X'80'	Alternate index
			AMDPATH	X'40'	Access via path
			AMDBASE	X'20'	Access via base
117	75	1	AMDATTR4		Attributes (fourth byte):
			AMDARCH	X'80'	Data component: component resides on a fixed block device. Index component: high level index is on a fixed block device.
			AMDARCHS	X'40'	Sequence set resides on a fixed block device (index components only).
118	76	2	AMDAIRKP		Relative key position in base record (only in data AMDSB of AIX)[1]
				Local Statistics	
120	78		AMDLSTAT		Local statistics
120	78	2	AMDLNIL		Local number of index levels
122	7A	2	AMDLNEST		Local number of entries in the index section
124	7C	4	AMDLNLR		Local number of user-supplied (logical) records
128	80	4	AMDLDELR		Local number of deleted records
132	84	4	AMDLIREC		Local number of inserted records
136	88	4	AMDLUPR		Local number of updated records

[1]For more details of these fields, see the explanation of the AMDSB group occurrence.

Figure 5.45 **Access Method Data Statistics Block (AMDSB) description and format (part 3 of 4)**

Offset Dec	Hex	Bytes	Field Name	Hex. Digit	Description
140	8C	4	AMDLRETR		Local number of retrieved records
144	90	4	AMDLASPA		Local bytes of free space
148	94	4	AMDLNCIS		Local number of control interval splits
152	98	4	AMDLNCAS		Local number of control area splits
156	9C	4	AMDLEXCP		Local number of EXCPs issued by VSAM I/O routines
				Exception Exit	
160	A0	8	AMDEXEXT		Exception exit
				Buffer Mangement Information	
168	A8	2	AMDBCBNO		Number of buffers
170	AA	2	AMDBFREE		Number of unassigned buffers
172	AC	4	AMDFSBCB		Address of the first BCB (for LSR: address of the BSPH)
176	B0	4	AMDFFBCB		Address of the first free BCB
180	B4	4	AMDCCWA		Pointer to BKPHD, which is the first control block in the FCDB area. The rest of the FCDB area is divided into 64-byte FCDBs, which are suballocated as needed for CCB(s), CCW(s), FXL(s), and IOARG(s).
184	B8	4	AMDHERBA		High RBA of extent currently being processed, for SAM ESDS (Same value as EDBHIRBA)
188	BC	2	AMDCIMLT		CI multiplier, specifies the number of CIs that are to be considered as a CA in certain parts of Record Management processing. For a SAM ESDS, it has a value of one, otherwise, it has the same value as AMDCIPCA.
190	BE	1	AMDRCFM1		Same as AMDRCFRM; zero if not a SAM ESDS.
			AMDIMPD1	X'80'	Same as AMDIMPF
			AMDNCIFI	X'20'	Same as AMDNCIFT
			AMDNCAF1	X'10'	Same as AMDNCAFT
			AMDSBLK1	X'04'	Same as AMDSBLKD
			AMDSVAR1	X'02'	Same as AMDSVAR
			AMDSFIX1	X'01'	Same as AMDSFIXD
191	BF	1			Reserved
				EDB Header	
192	C0	4	AMDFSEDB		Address of first EDB
196	C4	2			Reserved
198	C6	2	AMDLEDB		Length of EDB

[1]For more details of these fields, see the explanation of the AMDSB group occurrence.

Figure 5.45 **Access Method Data Statistics Block (AMDSB) description and format (part 4 of 4)**

Access Method Define the File (AMDTF) Table

The AMDTF table, used by the ISAM interface program, is an extension to each ISAM DTF table which a VSAM data set is associated. It contains all the information necessary to process the VSAM data set that is not contained in the DTF table. The AMDTF table is contained in reformatted form in the core image library. It is loaded by IIPOPEN the first time an OPEN is issued for a file and is completed by IIPOPEN at this time.

Offset Dec	Hex	Bytes	Field Name	Hex. Digit	Description
0	0	72	SAVARPP		Used to store register contents of problem program
72	48	72	SAVARCI		ISAM interface program save area
144	90	4	ACBAD		Address of ACB
148	94	4			X'0A020000' (SVC2)
152	98	4	RPLAD		Address of RPL
156	9C		EREPL		Error exit parameter list (Valid only if ERREXT=YES is specified in DTFIS.)
156	9C	4	DTFISAD		Address of DTFIS
160	A0	4	EPLRECAD		Address of record in error (not supported by IIP)
164	A4	8	EPLDASDA		DASD address of record in error (not supported by IIP)
172	AC	1	EPLRECID		Record identification
			EPRLRECID	X'80'	Data record (VSAM data set)
			EPLXREC	X'40'	Index record (VSAM sequence set)
			EPLCXREC	X'20'	Cylinder index record (VSAM index set)
			EPLMXREC	X'10'	Master index record (VSAM index set)
			EPLREAD	X'02'	Read
			EPLWRITE	X'01'	Write
173	AD	1	EPLCMNDC	X'00'	Command code of failing CCW (not supported by IIP)
174	AE	2			Unused
176	B0		GENACB		GENCB information to generate the ACB
176	B0	4	GACBHAD		Address of header
180	B4	4	MACRFEAD		Address of MACRF element
184	B8	4	FILENEAD		Address of filename element
188	BC		GACBH		Header
188	BC	1	GACBBTC	X'A0'	Block-type code (ACB)
189	BD	1	GACBFTC	X'01'	Function-type code (GENCB)
190	BE	2	GACBNOC	X'0001'	Number of copies (1 copy)
192	C0	4	GACBWAAD		Address of work area set to 0; VSAM obtains space via GETVIS
196	C4	4	GACBWALN		Length of work area
200	C8		MACRFEL		MACRF element
200	C8	4	MACRFKTC	X'00120000'	Keyword-type code
204	CC	4	MACRFVAL		Value supplied by IIPOPEN

Figure 5.46 **Access Method Define the File (AMDTF) table description and format (part 1 of 3)**

Offset Dec	Hex	Bytes	Field Name	Hex. Digit	Description
208	D0		FNAMEEL		Filename (DDname) element
208	D0	4	FNAMEKTC	X'00090000'	Keyword-type code
212	D4	8	FNAMEACB		Filename (inserted by IIPOPEN)
220	DC		GENRPL		GENCB information to generate the RPL
220	DC	4	GRPLHAD		Address of header
224	E0	4	ARLNEAD		Address of AREALEN element
228	E4	4	ACBEAD		Address of ACB element
232	E8	4	KEYLNEAD		Address of KEYLEN element
236	EC	4	RECLNEAD		Address of RECLEN element
240	F0		GRPLH		Header
240	F0	1	GRPLBTC	X'C0'	Block-type code (RPL)
241	F1	1	GRPLFTC	X'01'	Function-type code (GENCB)
242	F2	2	GRPLNOC	X'0001'	Number of copies (1 copy)
244	F4	4	GRPLWAAD		Address of work area set to 0; VSAM obtains space via GETVIS
248	F8	4	GRPLWALN		Length of work area set to 0
252	FC		ARLNEL		AREALEN element
252	FC	4	ARLNKTC	X'002D0000'	Keyword-type code
256	100	4	ARLNVAL		Area length
260	104		ACBEL		ACB element
260	104	4	ACBKTC	X'002B0000'	Keyword-type code
264	108	4	ACBAD1		Address of ACB
268	10C		KEYLNEL		KEYLEN element
268	10C	4	KEYLNKTC	X'00300000'	Keyword-type code
272	110	4	KEYLNVAL		Key length
276	114		RECLNEL		RECORDLEN element
276	114	4	RECLNKTC	X'00350000'	Keyword-type code
280	118	4	RECLNVAL		Record length
284	11C		SHOWCB		Information to show ACB or RPL
284	11C	4	SHHAD		Address of header
288	120	4	SHEAD		Address of element
292	124		SHH		Header
292	124	1	SHBTC	X'00'	Block-type code
293	125	1	SHFTC	X'03'	Function-type code (SHOWCB)
294	126	2	SHOTC	X'0000'	Object-type code

Figure 5.46 **Access Method Define the File (AMDTF) table description and format (part 2 of 3)**

Offset Dec	Hex	Bytes	Field Name	Hex. Digit	Description
296	128	4	SHBAD		Address of block to be shown
300	12C	4	SHARAD		Address of area
304	130	4	SHARLN		Length of area
308	134	4	SHAR		Area where information is to be placed
312	138		SHEL		Element
312	138	4	SHKTC		Keyword-type code (set by IIP)
316	13C		MODRPL		MODCB information to modify the RPL
316	13C	4	MRPLHAD		Address of header
320	140	4	OPTCDEAD		Address of OPTCD element
324	144	4	AREAEAD		Address of AREA element
328	148	4	ARGEAD		Address of ARG element
332	14C		MRPLH		Header
332	14C	1	MRPLBTC	X'C0'	Block-type code (RPL)
333	14D	1	MRPLFTC	X'02'	Function-type code (MODCB)
334	14E	2			Unused
336	150	4	MRPLBAD		Address of block to be modified (supplied by IIPOPEN)
340	154		OPTCDEL		OPTCD element
340	154	4	OPTCDKTC	X'00340000'	Keyword-type code
344	158	4	OPTCDVAL		Bit pattern (supplied by IIP)
348	15C		AREAEL		AREA element
348	15C	4	AREAKTC	X'002C0000'	Keyword-type code
352	160	4	AREAAD		Address of area (supplied by IIP)
356	164		ARGEL		ARG element
356	164	4	ARGKTC	X'002E0000'	Keyword-type code
360	168	4	ARGAD		Address of ARG parameter (supplied by IIP)
364	16C		MSGOUT		Header
364	16C	16	MSCCB		CCB
384	180	8	MSCCW		CCW
392	188	6	ERRCDE		Error code of message
398	18E	5	ISAMCM		'ISAM'
403	193	9	ISCM		ISAM command area
412	19C	5	VSAMCM		'VSAM'
417	1A1	9	VCCM		VSAM command area
426	1AA	4	CRCM		'RC= '
430	1AE	5	CRC1		Return code area
435	1B3	20	SHOWCBF		Area if SHOWCB failed
455	1C7	5	CRC2		Return code from SHOWCB
460	1CC	4	CRSCM		'EC= '
464	1D0	4	CRSC		Error code area
468	1D4	1	BRKT		Closing bracket

Figure 5.46 **Access Method Define the File (AMDTF) table description and format (part 3 of 3)**

BLDVRP Parameter List (BVRPPL)

The BLDVRP parameter list contains all parameters needed by module IKQBRP to build the VSAM resource pool. The address of the parameter list is held in register 1.

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
0	0	4	VRPBFLST		Address of buffer list
4	4	1	VRPKEYLN		Maximum keylength in VSAM Resource Pool
5	5	1	VRPSTRNO		String number of VSAM Resource Pool
6	6	6			Reserved
		VRP Buffer List			
The following 8 bytes are repeated for each subpool specified.					
12	C	4	VRPBFSZE		Size of buffers in subpool
16	10	1	VRPBFIND		Indicator byte
			VRPBLEND	X'80'	End of buffer list
17	11	1			Reserved
18	12	2	VRPBFCNT		Number of buffers in subpool

Figure 5.47 BLDVRP Parameter List (BVRPPL) description and format

Catalog Auxiliary Work Area (CAXWA)

The CAXWA is built when the VSAM catalog is opened or is being created. The CAXWA is used to contain the addresses of control blocks and work areas needed when a catalog is being processed. The CAXWA also contains flags that indicate the type of processing being performed on the catalog and the component that invoked the processing. The CAXWA is pointed to by the ACB (ACBUAPTR). The AMCBS (CBSCAXCN) contains the address of the CAXWA chain.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	1	CAXID	Control Block identifier X'CA'
1	1	3		Reserved
4	4	4	CAXCHN	Address of the next CAXWA in the chain
8	8	1	CAXFLGS	Flags:
		1...	CAXBLD	Build request
		.1..	CAXOPN	The catalog is being opened
		..1.	CAXCLS	The catalog is being closed
		...1	CAXEOV	An end-of-volume rountine is in control
		.... 1...	CAXCMP	Open/Close/EOV processing is complete
		.... .x..	CAXMCT	1=Master catalog 0=User catalog
		.... ..1.	CAXCMR	Catalog management has been called by a catalog management routine
		x	CAXSCR	Reserved for OS
9	9	1	CAXFLG2	Flags:
		1...	CAXF2DT	The catalog has been deleted
		.1..	CAXF2NDD	No DLBL filename found
		..x.	CAXF2CCR	0 = CCR needs to be read 1 = CCR has been read
		...1	CAXF2CRA	CAXWA for CRA
		.... 1...	CAXF2REC	Recoverable catalog
		.... .1..	CAXF2EOV	End of volume flag
		.... ..x.		Reserved for OS
		1	CAXF2CA	Free CAXWA if error
10	A	1		Reserved
11	B	1	CAXACT	Catalog activity count
12	C	4	CAXATIOT	Reserved for OS
16	10	4	CAXSCHWA	Reserved for OS
20	14	4	CAXDRWP	Address of the catalog's VTOC label read work area

Figure 5.48 **Catalog Auxiliary Work Area (CAXWA) description and format (part 1 of 2)**

Offset Dec	Hex	Bytes and Bit Pattern	Field Name	Description
24	18	4	CAXACB	Address of the catalog's ACB
24	18	4	CAXCRACB	Address of CRA's ACB
28	1C	4	CAXUCB	Address of the COMREG
32	20	12	CAXCCR	Catalog control record information
32	20	3	CAXHACI	Catalog interval number of the highest allocated control interval in the catalog
35	23	3	CAXNFCI	Control interval number of the next free control interval in the catalog
38	26	3	CAXCDCI	Number of deleted control intervals
41	29	3	CAXFDCI	Control interval number of the first deleted control interval in the catalog
44	2C	2		Reserved
46	2E	2	CAXRPLCT	Number of RPLs associated with the CAXWA
48	30	4	CAXRPL	Address of the first RPL in the CAXWA's RPL chain
52	34	44	CAXCNAM	Catalog name
96	60	4	CAXOPLST	Open/Close parameter list:
96	60	1	COPTS	Option flags:
		1...	CENLST	End-of-list indicator
		.xxx xxxx		Reserved
97	61	3	COPACB	Address of the catalog's ACB
100	64	4	CAXOPEWA	Address of Open/Close/EOV work area
104	68	4	CAXCCA	Address of the CCA
108	6C	4	CAXJDE	Reserved for OS
112	70	4	CAXCAT	Address of the catalog's ACB associated with CRA
115	74	6	CAXVOLCR	Volume serial of CRA volume
112	7A	2	CAXSYSCR	SYS-number of CRA volume
124	7C	6	CAXVOLRM	Volume serial of volume containing CRA (at present not mounted)
130	82	2	CAXSYSRM	SYS-number of volume containing CRA (at present not mounted)
132	84	6	CAXOCPAR	O/C parameter list
132	84	4	CAXOCACB	ACB address
136	88	2	CAXOCEOL	End of list indicator (x'0A02')
138	8A	18	CAXASPL	Automatic assign parameter list
138	8A	6	CAXASVOL	Assigned volume serial
144	90	2	CAXASLUB	Assigned logical unit
146	92	1	CAXASFLG	Automatic assignment flags
		1...	CAXASGN	Assignment required
		.1..	CAXNOCUU	Don't recommend a cuu in mount message
		..xx xxxx		Reserved
147	93	2	CAXASCLT	Device type and class for recommending cuu
147	93	1	CAXASCL	Device class
148	94	1	CAXASTYP	Device type
149	95	7	CASFILNM	File name for mount message

Figure 5.48 **Catalog Auxiliary Work Area (CAXWA) description and format (part 2 of 2)**

Catalog Communications Area (CCA)

The CCA is built each time the CATLG macro instruction is issued to process a VSAM catalog record. The CCA contains information about the catalog being processed and about the catalog record and its extensions contained in each of the six buffers available to process the user's request. The CCA is used to pass information between catalog management procedures. Register 11 contains the address of the CCA. Figure 5.50 shows the CCA description and format.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	2	CCAID	Identifier - set to X'ACCA'
2	2	2	CCASZ	Size
4	4	1	CCACD1	Return code 1
5	5	1	CCACD2	Return code 2
6	6	1	CCAFLG1	Flag byte 1:
		1...	CCAF1LPS	Stop the loop
		.1..	CCAF1ARA	Assign RPL to auxilary record area
		..1.	CCAF1LRD	Catalog control record read into virtual storage
		...1	CCAF1KEY	Retrieve the catalog record based on a DSNAME value (GET)
		...0		Retrieve by CI number
		.... 1...	CCAF1KGE	Retrieve the next catalog record (next GET)
		.... .1..	CCAF1CR	A checkpoint of the CCR required
		.... ..1.	CCAF1UP	GET macro instruction issued for update
		1	CCAF1DK	When the caller is renaming a data set, this flag indicates that the data set's true-name record is to be deleted, but the data set's catalog record is not to be deleted.
7	7	1	CCAFLG2	Flag byte 2:
		1...	CCAF2SYS	Reserved for OS
		.1..	CCAF2NVC	No validity check on the caller's CTGFL or work area is required
		..1.	CCALIMCL	Cylinder allocation
		...1	CCAF2XEQ	Exclusive enqueue
		...0		Shared enqueue
		.... 1...	CCAF2RHS	When a catalog management routine calles the VSAM Open routines to open a newly created catalog, and the Open routines call VSAM Catalog Management routines to obtain information about the catalog to be opened, the situation is called a 'recursive call'. The catalog cannot be dequeued when the Catalog Management routines return to the caller (VSAM Open routines).
		.... .xx.	CCAF2COB	Combination of catalog open and build:
		.... .1..	CCAF2CO	Catalog is being opened
		.... ..1.	CCAF2CB	Catalog open during build
		1	CCAF2SMO	Reserved for OS

Figure 5.49 Catalog Communication Area (CCA) description and format (part 1 of 12)

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
8	8	1	CCAFLG3	Flag byte 3:
		1...	CCAEXGR1	Exit indicator
		.1..	CCAGC4	The catalog record contains a password group occurrence (identified by Group Code 4) (detected during IGGPSCNC processing)
		..1.	CCAGDSP	GENDSP
		...1	CCAEXGR2	Exit indicator
		.... 1...	CCANF	The group occurrence cannot be found
		.... .1..	CCAELC2	Exit indicator
		.... ..1.	CCALFT	First time
		1	CCAEGREC	Exit indicator
9	9	1	CCAFLG4	Flag byte 4:
		1...	CCAF4DRQ	The catalog must be dequeued after the request completes
		.1..	CCAF4BYS	Bypass the security verification
		..1.	CCAGVNC	The required variable-length field is not completely contained in the record currently in the buffer
		...1	CCAGVNF	The group occurrence identified by the caller-specified sequence number cannot be found
		.... 1...	CCAGVNBS	There is no buffer space available to contain an extension record
		.... .1..	CCAGVEX	Exit indicator
		.... ..1.	CCAGVNE	The field does not exist in the located group occurrence
		1	CCATCOMP	Test complete: all group occurrence pointers have been examined and all designated fields have been tested
10	A	1	CCAFLG5	Flag byte 5:
		1...	CCAMEX2	Exit indicator
		.1..	CCAMEX	Exit indicator
		..1.	CCAMEX1	Exit indicator
		...1	CCAMODPA	The catalog record's base record must be written (using IGGPPAD) into the catalog
		.... 1...	CCATHIT	Successful test: a group occurrence has been found that satisfies the test conditions
		.... .1..	CCATEX	Exit indicator
		.... ..1.	CCATEX1	Exit indicator
		1	CCATEX2	Exit indicator

Figure 5.49 **Catalog Communication Area (CCA) description and format (part 2 of 12)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
11	B	1	CCAFLG6	Flag byte 6:
		1...	CCAMCODR	The catalog must be dequeued when the request completes
		.1..	CCADELP	A deleted group occurrence pointer was found
		..1.	CCAMNOSP	The catalog record's free space isn't large enough to contain all the new catalog information during the group occurrence move operation
		...1	CCAINIT	Insert switch for variable-length field being retrieved
		.... 1...	CCASUPFD	Suppress password field information during field retrieval
		.... .1..	CCAREUSE	The contents of the caller's record areas (buffers) can be used by IGGPEXT and IGGPMOD
		.... ..1.	CCAEXT	Set when a catalog management routine calls the Extract routine (IGGPEXT)
		1	CCAMOD	Set when a catalog management routine calls the Modify routine (IGGPMOD)
12	C	4	CCALAB	Address of the label cylinder area
12	C	1	CCALBLEN	Count field in units of 128 bytes
13	D	3	CCALBCYL	Address field
16	10	4	CCARB CCADPL	Pointer to RB Address of DADSM parameter list
20	14	4	CCACPL	Address of the caller's CTGPL
24	18	4	CCAACB	Address of catalog's ACB
28	1C	4	CCANPCCB	Address of saved CAXWA
32	20	4	CCAURAB	Address of the record area block (RAB) currently in use
36	24	44	CCASRCH	Search argument (DSNAME of cluster, data, index, catalog, or nonVSAM data set, or a volume serial number)
36	24	3	CCASRID	Control interval number
36	24	3	CCASRCIN	Control interval number
39	27	41		Reserved (or continuation of CCASRCH)
80	50	20	CCARAB0	Record Area Block 0: Each record area block describes the catalog record contained in one of the six catalog management buffers available for the request. RABs 1 through 5 are identical in format to RAB 0. Note: 'x' in each field name is replaced by '0' through '5' to indicate a particular RAB's field.

Figure 5.49 **Catalog Communications Area (CCA) description and format (part 3 of 12)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
80	50	1	CCARxFLG	Flags: The first three flags are used by IGGPEXT and IGGPMOD:
		1...	CCARxUR	The RAB is in use. It cannot be used by IGGPEXT or IGGPMOD
		.1..	CCARxU1	The RAB is temporarily in use by IGGPEXT or IGGPMOD. It cannot be overlaid.
		..1.	CCARxU2	(Same as CCARxU1)
		...1	CCARxWR	The buffer must be written before another catalog record can be read into it.
		.... 1...	CCARxPA	The buffer contains a new catalog record; PUT ADD is required to add the record to the catalog
		.... .xx.		Reserved
		1	CCARxUPD	Update buffer not reused
81	51	1	CCARxRPL	Last assign, RPL index
82	52	2		Reserved
84	54	4	CCARxREC	Address of the record in the buffer
84	54	4	CCACPE1x	Address of the record in the buffer
88	58	12	CCARxSEG	Addresses of the segments
88	58	4	CCACPE2x	Address of the first byte after the fixed-length header fields
92	5C	4	CCACPE3x	Address of the first group occurrence
96	60	4	CCACPE4x	Address of the first free-space byte in the record
100	64	20	CCARAB1	Record Area Block 1 (See RAB 0 description)
120	78	20	CCARAB2	Record Area Block 2 (See RAB 0 description)
140	8C	20	CCARAB3	Record Area Block 3 (See RAB 0 description)
160	A0	20	CCARAB4	Record Area Block 4 (See RAB 0 description)
180	B4	20	CCARAB5	Record Area Block 5 (See RAB 0 description)
200	C8	1	CCARPLK	Assigned RPL count
201	C9	1	CCARPLF	Index to RPL found
202	CA	1	LPINDX	Loop indexing control (counter)
202	CA	1	CCARPLX	Work byte for catalog RPL (mult.-use)
202	CA	1	XIOOPT	I/O options
		1...	XIOGET	1=GET, 0=PUT
		.1..	XIOERS	ERASE
		..1.	XIOARA	1=auxiliary record area required 0=user record area required
		...1	XIOKEY	1=keyed required 0=address required
		.... 1...	XIONUP	No update required
		.... .1..	XIONCK	No error check required
		.... ..1.	XIOTNE	1=true name entry 0=normal entry
		1	XIOKGE	GET NEXT (GET)
		1	XIOSEQ	PUTSEQ (PUT)

Figure 5.49 **Catalog Communications Area (CCA) description and format (part 4 of 12)**

Offset Dec	Hex	Bytes and Bit Pattern	Field Name	Description
203	CB	1	CCARPLT	Work byte for catalog RPL (mult.-use)
204	CC	6	CCARPLAA	Indices to assigned RPLs
210	D2	2		Reserved
212	D4	4	CCARPL1	Address of the RPL in use
216	D8	132	CCADESA	Save area for the extent information returned by VSAM, DADSM and Catalog Management: Suballocate
216	D8	1	CCANDEXT	Number of extents
217	D9	1	CCAIXEXT	Extent index value
218	DA	2	CCASSVOL	Sequence number of the data set directory entry in the volume catalog record
220	DC	128	CCAEXTDE	Sixteen 8-byte extent descriptors:
			First extent descriptor	
220	DC	2	CCAEXTSS	Sequence number of the Data Space group occurrence that this extent's space is a part of
222	DE	4	CCAEXTAD	The extent's starting physical address:
222	DE	2	CCAEXTCC	Cylinder number CC
224	E0	2	CCAEXTHH	Head number HH
226	E2	2	CCAEXTTH	Number of tracks in the extent
228	E4	120		Space for remaining 15 extent descriptors
348	15C	1	CCAASCIK	Number of control intervals required to satisfy the caller's request
349	15D	1	CCACRRP	X'80' Build 'caller' chain for message 4223I
350	15E	1	CCAASCIX	Used by the ASSGN functions - points to the element in CCAASCI currently being processed
351	15F	1	CCASRPLX	Saved RPL flags
352	160	9	CCAASCI	Number of each assigned control interval
361	169	3	CCAUPGD	Control interval for UPG modification
364	16C	16	CCAEQDQ	Enqueue/Dequeue parameter list
364	16C	1	CCAEDXFF	End of parameter list, indicator byte =X'FF' (if list is empty)
365	16D	1	CCAEDRLN	Length of minor name
366	16E	1	CCAEDOPT	Enqueue/Dequeue Options
		x...	CCAEDSHR	1=Shared, 0=Exclusive
		.x..	CCARLSEB	Reserved
		..xx xxxx		Other options (set by macro)
367	16F	1	CCAEDRCD	Enqueue/Dequeue return code
368	170	4	CCARTSAV	Save area for CCAMLRET
372	174	4	CCACOMRG	COMRG pointer
376	178	4	CCAEDUCB	Work area
380	17C	4	CCAMLRET	Address of the caller's save area used by IGG0CLAG

Figure 5.49 Catalog Communications Area (CCA) description and format (part 5 of 12)

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
384	180	12	CCAMSSPL	GETVIS/FREEVIS parameter list area
384	180	4	CCAMNLEN	Number of bytes to process
388	184	4	CCAMNPTR	Address of the return address
392	188	1		Reserved for OS
393	189	1	CCAMNSPL	Reserved for OS
394	18A	2		Reserved for OS
396	18C	4	CCARPRM	Return parameters
400	190	8	CCACMS	Catalog Management Services work area
400	190	4	CCACMSWA	Address of the CMS calling routine's work area
404	194	4	CCAEXCMS	Address of a secondary CMS work area
The following fields are set and used by IGGPLOC, IGGPEXT, and IGGPTSTS, and catalog management subfunctions which these procedure call:				
408	198	0	CCALUME	
408	198	4	CCACPE5	Address of a selected group occurrence pointer
412	19C	4	CCACPE51	(Same as CCACPE5)
416	1A0	4	CCACPE52	(Same as CCACPE5)
420	1A4	4	CCACPE53	(Same as CCACPE5)
424	1A8	4	CCACPE6	Address of a selected group occurrence
428	1AC	4	CCACPE61	(Same as CCAPE6)
			CCARABSE	Save extract caller URAB
432	1B0	4	CCACPE7	Address of field value
			CCAIDPT	Insert data address
436	1B4	4	CCACPE71	Alternate address to field value
440	1B8	2	CCAGOPLN	Length of the group occurrence pointer
442	1BA	2	CCASL	Length of sequence number field (RELREPNO) in the group occurrence
444	1BC	4	CCAILNG	Length of the selected retrieved field
448	1C0	4	CCAFLPT CCATFLPT	Address of the requested-field CTGFL Address of the CTGFL-for-tests
452	1C4	4	CCARABPT	Address of the record area block
456	1C8	4	CCADICT	Dictionary information to describe the field, based on its field name
460	1CC	4	CCAXCPL CCAMCPL	Address of the CTGPL built when IGGPEXT and IGGPMOD are called, so that information in the caller's CTGPL is not altered
464	1D0	4	CCARABB	Address of the RAB that identifies the base catalog record
468	1D4	4	CCARABF	Address of the RAB that identifies the first record area (buffer) that can be used by IGGPEXT or IGGPMOD

Figure 5.49 **Catalog Communications Area (CCA) description and format (part 6 of 12)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
472	1D8	4	CCARABL	Address of the RAB that identifies the last record area (buffer) that can be used by IGGPEXT or IGGPMOD
476	1DC	3	CCACBASE	The control interval number of the base catalog record
479	1DF	1	CCAGC	Group code of the requested group occurrence
480	1E0	2	CCALREL CCALREL1	Relative repetition number of a selected group occurrence
482	1E2	2	CCASN CCASN1	Sequence number of a selected group occurrence
484	1E4	1	CCAFLG8	CRA flags
		1...	CCARPUT	Inhibit CRA PUT/UPDATE
		.1..	CCALSTC	Listcat request
		..1.	CCAEXTCR	Extend CRA in process
		...1	CCABLDCR	Open request for CRA build
		.... 1...	CCASPUCO	Special UCAT
		.... .x..	CCASCAX	1=CRA CAXWA search, 0=UCAT CAXWA search
		.... ..x.	CCAUPG	1=upgrade, 0=no upgrade
		1	CCABUF	Output buffering flag
485	1E5	1	CCAFLGA	More flags
		1...	CCAUPGRR	RAB1 to be restored by upgrade module
		.1..	CCARGET	Get record for compare before update CRA
		..1.	CCALBFVT	Multiple file parameter search at define
		...1	CCACRARD	Indicate CCR for CRA has been reached
		.... 1...	CCAF1LSV	Save indicator flag CCAF1LRD
		.... .1..	CCACANIN	Cancel INHIBIT
		xx		Reserved
486	1E6	2	CCAIXFPL	Index to the current CTGFL being processed
488	1E8	2	CCAIXREL	Index for CCATREL
490	1EA	2	CCATNREL	The sequence number of the next group occurrence to perform tests against if CCATREL is full or if there are no buffers available to contain the catalog record's next extension
492	1EC	2	CCATNUM	Number of successful relative repetition numbers (cannot exceed 16)
494	1EE	32	CCATREL	Successful relative repetition numbers
526	20E	2	CCATNO	Total number of successful relative repetition numbers (might exceed 16)
528	210	4	CCATEST	Address of the test CTGFL

Figure 5.49 Catalog Communications Area (CCA) description and format (part 7 of 12)

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
532	214	20	CCARBA	Work area for extent descriptors
532	214	2	CCASS	Sequence number of the Data Space group occurrence that contains the extent
534	216	4	CCACCHH1	Physical address - CCHH - of the extent's first track
538	21A	4	CCACCHH2	Physical address - CCHH - of the extent's last track
542	21E	2	CCATT	Number of tracks in the extent
544	220	4	CCARBA1	Low relative byte address (RBA)
548	224	4	CCARBA2	High relative byte address (RBA)
552	228	2	CCATLNG CCATLEN	Total length of the extent information that has been processed (CCATLNG); total length of the scanned field so far (CCATLEN)
554	22A	2	CCARBAL	RBA extent balance
556	22C	2	CCACNIX	Combination name index
558	22E	2	CCAREASN	Reason code
560	230	4	CCAIDPT2	Address of the available space in the caller's work area or of the caller-supplied update information
564	234	4	CCAIDPT3 CCARABSM	Address of the length-field of a variable length field in the user's return area
568	238	2	CCAGVCT	Number of group occurrence pointers processed so far
570	23A	2	CCANEVV	If the requested variable-length field is non-existent, this field is set to binary zero
572	23C	3	CCAGVEXT	Control interval number of the record's next extension record (not yet in a buffer)
575	23F	1	CCANEFV	If the requested fixed-length field is non-existent, this byte is set to X'FF'
576	240	1	CCADEF	Define flags
		1...	CCADED	Define with DEDICATE
		.xxx xxxx		Reserved
577	241	1	CCAGRGC	Group code of the requested group occurrence
578	242	2	CCAGRHI CCAGRHI1	High relative repetition number of the requested group occurrence
580	244	2	CCAIXTPL	Index to test FPL
582	246	2	CCADLEN	Number of bytes to be deleted from the catalog record
584	248	2	CCADIFF	The difference between the insert length and the delete length (can be a negative number)

Figure 5.49 **Catalog Communications Area (CCA) description and format (part 8 of 12)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
586	24A	2	CCAREPCT	Number of relative repetition numbers processed so far
588	24C	2	CCADISP	Displacement into variable-length field to the delete/insert location
590	24E	3	CCASVCI	Save area for the control interval number of the base catalog record
593	251	3	CCASVCI1	Save area for the control interval number
596	254	4	CCADTA	Address of the dictionary
600	258	4	CCACDTA	Address of the index combination table
604	25C	2	CCADTCT	Number of dictionary entries
606	25E	2	CCACDTCT	Number of index combination entries
608	260	4	CCACWAP	Controller work area
612	264	4	CCAMNADR	Address of the virtual storage obtained by a GETVIS request
616	268	4	CCAILNG3	Save area for the insertion length
620	26C	4	CCAILNG2	Length of the user-supplied insert data
624	270	4	CCAALPTR	Address of the space management work area
628	274	4	CCAGVLSV	Address of GVL work area
632	278	4	CCALCPL	Reserved for OS
636	27C	1	CCAFLG7	Flags:
		x...	CCALSP	Reserved for OS
		.1..	CCANRLSE	Release Control Bit
		..1.	CCACKDEL	Delete switch
		...1	CCASMFBR	Do GET for base record
		.... 1...	CCAONCE	Move only one occurrence
		.... .1..	CCAROREQ	Read only request
		.... ..1.	CCAFEOV	Force EOV
		1	CCAEQOPN	Enqueued on SYSOPEN
637	27D	3	CCARCI	CRA Record control interval number
640	280	4	CCALABSV	Saved address of IKQLAB area
644	284	4	CCARABSV	Saved address of RAB
648	288	2	CCAMODUL	Last two bytes of module name (IGG0CLxx)
650	28A	3	CCACHAIN	Control interval number save area
653	28D	3	CCACI1	Control interval number save area
656	290	3	CCACI2	(Same as CCACI1)
659	293	3	CCACI3	(Same as CCACI1)
662	296	2	CCAVARLN	Number of bytes to be inserted into the record
664	298	4	CCARRAB	Address of the RAB containing the group occurrence pointers where delete/insert processing is to begin
668	29C	4	CCARBASE	(Same as CCARRAB)

Figure 5.49 **Catalog Communications Area (CCA) description and format (part 9 of 12)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
672	2A0	4	CCAVARPT	Address of the information to be inserted into the record
676	2A4	2	CCADELN	Number of bytes to be deleted from the record
678	2A6	20	CCAVAR	Insert information save area
678	2A6		CCAPIN	MFG area for PIN
698	2BA	20	CCAVAR1	(Same as CCAVAR)
718	2CE	3	CCADEL1	The control interval number of the first record in a series of records to be deleted
721	2D1	3	CCADEL2	The control interval number of the last record in a series of records to be deleted
724	2D4	40	CCAXLATE	Translation work area
764	2FC	4	CCAR14S	Register 14 save area
768	300	0	CCABMINP	Input parameters to IGG0CLBR
768	300	2	CCABMTRK	Starting track
770	302	2	CCABMLIM	Check limit, nn for set
772	304	2	CCABMMIN	Conditional check minimum
774	306	1	CCABMFLG	State and function code
		x...	CCABMST	This bit can be 0 or 1, and is the state for which an extent is to be checked (if bit 1 is on) or the state to which a map bit is to be set (if bit 2 is on)
		.1..	CCABMCHK	ON - Perform check
		..1.	CCABMSET	ON - Perform set
		...1	CCABMCCK	ON - Perform condition check
		.... 1...	CCABMLST	ON - Last set request (write)
		xxx		Reserved
775	307	1		Reserved
776	308	0	CCABMOUT	Output parameters from IGG0CLBR
776	308	2	CCABMONN	Track number
778	30A	2	CCABMOTR	Starting track
780	30C	1	CCABMOFG	Output flags
		1...	CCABMOST	State of bits
		.xxx xxxx		Reserved
781	30D	6	CCAVOLCR	CRA volume identification
787	313	1	CCABMPAD	Padding character
788	314	4	CCABMGOP	Current bit mask GOP
792	318	4	CCABMPTR	Address of current bit mask byte
796	31C	4	CCABMEND	End of current bit mask
800	320	2	CCABMBT1	Bit count, first byte
802	322	2	CCABMBTL	Bit count, last byte
804	324	2	CCABMBYT	Number of full bytes
806	326	2	CCABMSTR	Current bit mask, start track
808	328	4	CCABMWK1	Work field
812	32C	4	CCABMWK2	Work field

Figure 5.49 **Catalog Communications Area (CCA) description and format (part 10 of 12)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
816	330	4	CCABMWK3	Work field
820	334	4	CCABMWK4	Work field
824	338	4	CCABMRB1	Address of first bit map RAB
828	33C	4	CCABMRB2	Address of second bit map RAB
832	340	4	CCACARWA	Address of CRA definition work area
836	344	4	CCACRABF	Address of CRA buffer
840	348	4	CCASACB	Address of saved CCAACB field
844	34C	4	CCAEXC	Save area for CCAACB
848	350	4	CCASRPL	Address of saved CCA, RPL field
852	354	4	CCAADBUF	Address of cluster record buffer (cluster record saved until CRA volume known)
856	358	4	CCASCAXS	Address of search argument for CAXWA chain search
860	35C	4	CCASCAXA	Address of found CAXWA
864	360	4	CCADEVT	CRA volume device type
868	364	8	CCANMF3	Save area for resource name
The following two fields are used by the no-upgrade/upgrade function, called by ALTER, DEFINE or DELETE.				
876	36C	3	CCAXDCI	AIX data control interval number
879	36F	3	CCAXICI	AIX index control interval number
882	372	1	CCACATIN	CLAH indicator
883	373	1	CCACPLSV	Catalog Parameter List options save area
884	374	4	CCACOPTR	CLCO work area
888	378	4	CCADEVA	Address of device attribute return area
892	37C	4	CCAFARE	Address of file identification
896	380	4	CCAAREA	Pointer to address of label record area
900	384	2	CCAMDSAV	Save area for CCA module
902	386	2	CCARSSAV	Save area for CCA
The following fields are used for converting block extents to track extents.				
904	388	4	CCALIMST	Extent starting address
908	38C	4	CCALIMED	Extent ending address
912	390	4	CCALIMBL	Extent size in blocks
916	394	4	CCALIMTR	Extent size in tracks
920	398	4	CCAFEXT	Size of first extent
924	39C	4	CCAFSUCB	UCB code save area
928	3A0	6	CCAFVLID	Vol-ID save area
934	3A6	1	CCAFBFLG	Fixed block flags:
		1...	CCAFBTER	Fixed block locate: device characteristics could not be retrieved
		.1..	CCAFBNRC	Don't repeat IGGPXVAL
		..1.	CCAFBIN1	FBA indicator
		...1	CCALIMEN	More than one extent
		.... 1...	CCALIMRC	Recoverable catalog
		xxx		Reserved

Figure 5.49 **Catalog Communications Area (CCA) description and format (part 11 of 12)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
935	3A7	1	CCAFBOPT	Space options save area:
		xx..	CCAFBSPO	Allocation unit
		..xx xxxx		Reserved
936	3A8	24	CCAFDEVC	Device characteristics save area:
936	3A8	4	CCAFBUCB	UCB code
936	3A8	1	CCAFUFG1	Operational characteristics
937	3A9	1	CCAFUFG2	Optional features
938	3AA	1	CCAFUCLS	Device class
939	3AB	1	CCAFUTYP	Device type
940	3AC	4	CCAFBBLK	Block size
944	3B0	2	CCAFBNCL	Number of cylinders
946	3B2	2	CCAFBNTK	Number of tracks per cylinder
948	3B4	2	CCAFBTSZ	Number of blocks per track
950	3B6	10		Reserved
960	3C0	12	CCARNM	Resource name
960	3C0	6	CCAVOLSV	Temporary VOLID save area
966	3C6	6		Reserved
972	3CC	4	CCADTL	DTL pointer
976	3D0	4	CCARNMP	Resource name printer
980	3D4	1	CCACNTL	Control parameter work area
981	3D5	3	CCACISAV	CI number save area
984	3D8	1	CCAFLG9	Flags
		1...	CCADQSYS	Unlock system resource
		.1..	CCADQVOL	Unlock volid resource
		..1.	CCADQDIC	Unlock ci# resource
		...1	CCADSOPN	A data set is opened and locked
		.... 1...	CCADQOPN	Dequeue system OPEN resource
		.... .1..	CCADQDIA	Dequeue data/index for AIX
		.... ..1.	CCADQNRQ	Dequeue not required
		1		Reserved
985	3D9	1	CCACD3	Return code save area
986	3DA	2		Reserved
988	3DC	40	CCATEMPS	Temporary area for PLS
1028	404	420	CCAREGS	Save area for registers
1028	404	4		Address of user save area
1032	408	8	CCAMODNM	Load module name
1040	410	401		Reserved
1448	5A8	4	CCAACBSV	ACB pointer save area
1452	5AC	3	CCACIND	ci# save area
1455	5AF	0	CCAEND	End CCA

Figure 5.49 **Catalog Communications Area (CCA) description and format (part 12 of 12)**

Catalog Field Parameter List (CTGFL)

The CTGFL is built before a user component issues the CATLG macro instruction to process a VSAM catalog record. The CTGFL defines one of the catalog record's fields or a group of logically related fields (a combination). The CTGFL is used in three situations:

- It identifies a catalog record field to retrieve or update. The CTGPL contains the address of each CTGFL used in this way.
- It identifies a catalog record field to compare against caller-supplied data. This is a 'test' CTGFL and is addressed by another CTGFL.
- For update-extend processing, one or three FPLs identify the volume information group occurrence(s) to be extended. The catalog record fields identified by the CTGFL(s) are not explicitly retrieved or updated for the caller.

When a catalog management routine is processing a CTGFL, the CTGFL's address is in the CCA (CCAFLPT or CCATEST).

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	1	CTGFLDNO	Number of entries in CTGFLDAT
1	1	1	CTGFLDCD	Test condition:
		X'00'		The FPL describes a field to be updated or retrieved. The FPL is pointed to by the caller's CTGPL (CTGFIELD entry).
		not X'00'		This FPL describes a test condition, and is pointed to by a request FPL. The codes for the test conditions are:
		X'80'		Equal
		X'60'		Not equal
		X'20'		Greater than
		X'40'		Less than
		X'A0'		Greater than or equal
		X'C0'		Less than or equal
		X'80'		Test under mask for zeros
		X'10'		Test under mask for ones
		X'40'		Test under mask for mixed
2	2	1	CTGFLDGC	Group code number
3	3	1	CTGFLDRE	Test results:
		0...	CTGFLFBA	CKD device
		1...		FBA device
		.xxx xxx.		Reserved
		0	CTGFLDTS	Successful test
		1		Test failed

Figure 5.50 **Catalog Field Parameter List (CTGFL) description and format (part 1 of 2)**

Offset Dec	Hex	Bytes and Bit Pattern	Field Name	Description
4	4	4	CTGFLDWA	Work area: contains information about the catalog record's field name from the dictionary
8	8	4	CTGFLDNM	Address of the field name
12	C	4	CTGFLCHN	Address of next field macro or zero
16	10	0	CTGFLDAT	Combined name for data length and address
16	10	4	CTGFLNG	Data length of:
20	14	4	CTGFLPT	Address (in caller's work area) of: • The field that was retrieved, if the request was LOCATE or CMS LIST-CAT, or • New data to replace or add to data in the catalog record, if the request was UPDATE, CMS DEFINE or CMS AL-TER, or • Data used to compare to catalog record fields, if the FPL is a FPL-for-tests.

Figure 5.50 **Catalog Field Parameter List (CTGFL) description and format (part 2 of 2)**

Catalog Field Vector Table (CTGFV)

The CTGFV is built by the Access Method Services utility programs and contains addresses of user-supplied information fields and lists. The CTGFV is built when the user issues a DEFINE or ALTER command. If the user is creating a cluster, a CTGFV is built for each catalog record that will be built to describe the cluster, i.e., Access Method Services builds a cluster CTGFV, a data CTGFV, and, if the cluster is key-sequenced, an index CTGFV. The CTGFV is pointed to by the CTGPL (CTGFVT). If Access Method Services builds more than one CTGFV, the cluster CTGFVs are pointed to by the CTGPL (CTGFVT) and the data and index CTGFVs are pointed to by the cluster CTGFV.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	1	CTGFVTYP	The CTGFV contains information used by the CMS Define routines to build a catalog record of the type:
		C'D'	CTGFVDTA	Data
		C'C'	CTGFVCL	Cluster
		C'I'	CTGFVIDX	Index
		C'V'	CTGFVVOL	Volume (Space)
		C'A'	CTGFVALN	Non-VSAM
		C'G'	CTGFVAIX	Alternate Index
		C'R'	CTGFVPTH	Path
1	1	1	CTGFVPRO	CMS processing option flags:
		1...	CTGFVAVL	ALTER: Add volumes
		.1..	CTGFVRVL	ALTER: Remove volumes
		..1.	CTGFVNDC	Device type converted switch
		..1	CTGFVIMP	Import request
		 xxxx		Reserved
2	2	1	CTGFVELM	Element number of CMSPCATR
3	3	1		Reserved
4	4	4	CTGFVDCH	Address of the cluster's data FVT
8	8	4	CTGFVICH	Address of the cluster's index FVT
12	C	4	CTGFVVCH	Address of the space vector table
16	10	4	CTGFVIND	Address of the associated DLBL statement
20	14	4	CTGFVENT	Address of the 44-byte entry name
24	18	4	CTGFVSTY	Address of the security information FPL (passwords, codewords, and number-of-tries)
28	1C	4	CTGFVOWN	Address of the owner identification FPL
32	20	4	CTGFVEXP	Address of the expiration data FPL
36	24	4	CTGFVCRE	Address of the creation date FPL
40	28	4	CTGFVVLT	Address of the volume serial number list
44	2C	4	CTGFVRNG	Address of the key range list
48	30	4	CTGFVDVT	Address of the device type FPL (for nonVSAM DEFINE only)
48	30	4	CTGFVSPR	Address of space return information
52	34	4	CTGFVSPC	Address of the space allocation information FPL
56	38	4	CTGFVAMD	Address of the AMDSB FPL (if VSAM DEFINE)
			CTGFVFSN	Address of the file sequence number (if NonVSAM DEFINE)
60	3C	4	CTGFVATR	Address of the data set attributes FPL
64	40	4	CTGFVBUF	Address of the buffer size FPL
68	44	4	CTGFVLRS	Address of the average record size FPL
72	48	4	CTGFVEXT	Address of exception exit
76	4C	4	CTGFVNAM	Address of related object
80	50	4	CTGFVUPG	Address of FPL for 'RGATTR'
84	54	4	CTGFVWKA	Address of CRA volume identification
88	58	4	CTGFVPWD	Relationship password

Figure 5.51 **Catalog Field Vector Table (CTGFV) description and format**

Catalog Parameter List (CTGPL)

The CTGPL is built before a VSE component issues the CATLG macro instruction to process a VSAM catalog record. The CTGPL defines the catalog management request and its options, the catalog record to be processed, and the VSAM catalog that contains the record. The CTGPL is pointed to by register 1. When the catalog management routines build a CCA to support the request, the address of the CTGPL is put into the CCA (CCACPL).

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	1	CTGOPTN1	First option indicator:
		1...	CTGBYPSS	Bypass the catalog management security verification processing
		.1..	CTGMAST	Check the master password
		..1.	CTGCI	Check the control interval password
		...1	CTGUPD	Check the update password
		.... 1...	CTGREAD	Check the read password
		.... .1..	CTGNAME	The CTGENT field contains the address of a 44-byte DSNAME, or a 6-byte volume serial number (padded with binary zeros)
		.... .0..		The CTGENT field contains the address of a 3-byte control interval number
		.... ..1.	CTGCNAME	The CTGCAT field contains the address of a 44-byte catalog DSNAME
		.... ..0.		The CTGCAT field contains the address of a VSAM catalog's ACB
		x		Reserved
1	1	1	CTGOPTN2	Second option indicator:
		1...	CTGEXT	Extend option (with UPDATE)
			CTGERASE	Erase option (with DELETE)
			GTFSMF	Write SMF
			GTGREL	Release (with UPDATE) Release secondary extents when data set is opened as a reusable data set
		..1. ...	CTGPURG	Purge option (with DELETE)
			CTGVMNT	Volume mount caller
		...1	CTGGTNXT	Get-next option (with LISTCAT)
		.... 1...	CTGDISC	Disconnect option (with DELETE)
		.... .1..	CTGOVRID	Erase override option (with DELETE)
		.... ..1.	CTGSCR	Scratch space option (with DELETE)
		1	CTGDEALL	Deallocate all (RELEASE)
2	2	1	CTGOPTN3	Third option indicator:
		xxx.	CTGFUNC	Specifies the caller-requested function:
		...1	CTGSUPLT	Super LOCATE
		.... x...		Reserved
		.... .x..	CTGSRH	Reserved
		.... ..x.	CTGNUM	Reserved
		1	CTGAMO	VSAM request
		0		Non-VSAM request

Figure 5.52 **Catalog Parameter List (CTGPL) description and format (part 1 of 2)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
3	3	1	CTGOPTN4	Reserved for OS
4	4	4	CTGENT	User entry address (address of volume in the case of OS)
			CTGFVT	Address of callers CTGFV (DEFINE, ALTER)
8	8	4	CTGCAT	Address of the catalogs DSNAME or ACB, as specified in CTGOPTN1
8	8	4	CTGCVOL	Catalog volume pointer - (Super LOCATE)
12	C	4	CTGWKA	Address of the callers work area
16	10	1	CTGDSORG	Data set origin - (Super LOCATE)
16	10	1	CTGOPTNS	CMS options:
		0000 1...	CTGDEFIN	DEFINE
		0001 0...	CTGALTER	ALTER
		0001 1...	CTGDELET	DELETE
		0010 0...	CTGLTCAT	LISTCAT
		xxx		Reserved
17	11	1	CTGCRFLG	CRA open flags
		1...	CTGLBCYL	Label area information is passed for CRA
		.1..	CTGCTRBL	Control blocks are passed for CRA
		..xx xxxx		Reserved
18	12	1	CTGTYPE	Type of catalog record
		C'D'	CTGTDATA	Data
		C'I'	CTGTINDX	Index
		C'A'	CTGTALIN	Non-VSAM
		C'U'	CTGTUCAT	User catalog
		C'V'	CTGTVOL	Volume
		C'C'	CTGTCL	Cluster
		C'M'	CTGTMCAT	Master catalog
		C'G'	CTGTAIX	Alternate index
		C'R'	CTGTPTH	Path
		C'Y'	CTGTUPG	Upgrade set
		C'F'	CTGTFREE	Free record
19	13	1	CTGNOFLD	Number of entries contained in CTGFIELD
20	14	4	CTGDDNM	Address of DD-name
			CTGNEWNM	Address of the new DSNAME, if the request is ALTER and the object's name is being changed
20	14	2	CTGFDBK	Super LOCATE feedback
22	16	2	CTGFBFLG	Super LOCATE flags
		1...	CTGPAR	Parallel mount - (Super LOCATE)
		.1..	CTGKEEP	Forced keep - (Super LOCATE)
		..xx xxxx		Reserved
23	17	1		Reserved
24	18	4	CTGJSCB	Reserved for OS
			CTGPSWD	Address of the caller supplied password
28	1C	4	CTGDDUC	Address of UCAT file name
32	20	4	CTGDDCR	Address of CRA file name
36	24	Variable	CTGFIELD	Field pointers

Figure 5.52 **Catalog Parameter List (CTGPL) description and format (part 2 of 2)**

Close Work Area

The Close Work Area is built when a VSAM data set is opened. It contains work area data for alternate index processing and save areas for close registers and catalog registers. It is pointed to by CLWAAD, displacement 112 (X'70'), in the Open Work Area during OPEN. After OPEN is complete, the AMBL field, AMBLCLWA, points to the close work area associated with its ACB.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	20	WAPASS	Used to pass information from OPEN to CLOSE
0	0	4	WAEYCAT	Eye catcher
4	4	1	DISP	Job control disposition
		1...	NEW	DISP=NEW
		.1..	OLD	DISP=OLD
		..xx		Reserved
		.... 1...	KEEP	DISP=(,KEEP)
		.... .1..	DELETE	DISP=(,DELETE)
		.... ..1.	DATE	(DISP=(,DATE)
		x		Reserved
5	5	1	DSATTRIB	Data set attributes
		1...	REUSE	Defined reusable
		.1..	NOALLOC	Defined NOALLOCATE
		..1.....	WAIFPDF	Defined implicitly
		...x xxxx		Reserved
6	6	3	EXPDATE	Expiration date
9	9	1		Reserved
10	A	2	WAFEATSW	Feature switches
			First Byte	
		1...	WAFDET	Features determined
		.xxx xxxx		Reserved
			Second Byte	
		xxxx xxx.		Reserved
		1	WAFSPMGT	Space management feature in system
12	C	8		Reserved
20	14		WACOMACB	Part common to all ACBs in close
			WAFLAG	Flag byte:
		1...	TCLOSE	Work area for TCLOSE
		.1..	CLOSE	Work area for Close
		..1.	OPEN	Work area for Open
		...1	OPAMDINX	Index AMDSB is being processed
		.... 1...	VOLFOUND	Volume serial number is in label cylinder record
		.... .1..	SSFLAG	Sequence set with data
		.... ..1.	RETRY	Catalog should be reupdated by Close
		1	FILEPROT	DOS Supervisor DASD file protect
21	15	1	WAERCODE	Error condition code
22	16	2	WALEN	Length of GETVIS area
24	18	4	WAWORK1	General work field
28	1C	4	WALISTP	Address of user ACB/DTF list
32	20	4	WACOMR	Address of partition COMREG
36	24	4	WADSAAD	Address of dynamic save area
40	28	2	WADSALN	Length of dynamic storage area (DSA)
42	2A	1	EDBCODE	Code to remember EDBs/GETVIS
43	28	1		Reserved

Figure 5.53 **Close Work Area description and format (part 1 of 3)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
44	2C	4	CATEXTPT	Pointer for extents
48	30	2	CATEXTLN	Length of total extents
50	32	2	EXTNUMB	Number of extents
52	34	80	USERSAVE	Room to save user jobname, PSW, and registers (from partition save area)
132	84	8		Reserved
140	8C	12	PARM	Parameter list for IKQOCSHR
140	8C	1	WAOPCODE	Operation code used for determining function
141	8D	7	DSID	Data set ID
141	8D	3	DSCI	CI number
144	90	4	CTACBPTR	Pointer to catalog ACB
148	94	1	SHAREOPT	Share option from catalog
		1...	WASHR1	Share option 1 file
		.1..	WASHR2	Share option 2 file
		..1.	WASHR3	Share option 3 file
		...1	WASHR4	Share option 4 file
		 xxxx		Reserved
149	95	1	WAPFLAG	Option flags
		1...	WAPOUT	1 - lock/unlock for output access
		0...		0 - lock/unlock for input access
		.xxx xxxx		Reserved
150	96	2	OUTCNT	Indicator of output user count
152	98	24	WAOCDTL	Area to contain DTL (must be large enough to hold MAPDTL)
176	B0	0	WACOMEND	End of common work area
176	B0	0	WORKAREA	Close work area
176	B0	8	TIME	Time used to update catalog
184	B8	4	CLWRET1	Return address of savearea 1
188	BC	4	CLWRET2	Return address of savearea 2

Figure 5.53 **Close Work Area description and format (part 2 of 3)**

Offset Dec	Hex	Bytes and Bit Pattern	Field Name	Description
192	C0	1	BITBANK	Close flags
		1...	ENQACT	USE macro was issued
		.1..	ENQOPN	Enqueue on SYSOPEN
		...x xxxx		Reserved
193	C1	1	SETOFLG	Byte of zero for resetting OPEN indicator
194	C2	2		Reserved
196	C4	0	CLAIXWA	AIX work area
196	C4	4	ACBREQ	Address of ACB for which CLOSE/TCLOSE is requested
200	C8	4	ACBBASE	Address of base cluster ACB
204	CC	4	USBADDR	Address of USB
208	D0	4	ACBCURR	Address of ACB currently being processed
212	D4	4	RPLCURR	Address of header RPL
216	D8	1	AIXATTR	AIX attribute of current ACB
		1...	AIXUPGRD	Member of upgrade set but not path entry
		.1..	AIXBASC	Base ACB
		..1.	AIXENTRY	Path entry ACB
		...x xxxx		Reserved
217	D9	1	AIXPROC	Processing status
		1...	UPGD2	First member of upgrade set has been processed
		.xxx xxxx		Reserved
218	DA	2	CLMSGFLG	IKQOCMSG flags
220	DC	8	FTAB	Resource name field for protection of file tab
228	E4	4	USBCURR	Address of current USB ACB
232	E8	4	SAVEPTR	Pointer to save area
236	EC	72	REGSAVE	Close register savearea
308	134	72	CATSAVE	Catalog register savearea
308	134	4	CLWTEMP	Temporary work area
380	17C	12	CATDATA	Catalog data
380	17C	4	CATLSTP	Address of catalog list
384	180	2	CATLSTSZ	Catalog list size
386	182	2	CATWASIZ	Catalog work area size
388	184	4	CATWAPTR	Address of catalog work area
392	188	808	CATWA	Catalog work area
392	188	52	DUMMYRPL	Dummy RPL
444	1BC	16	DUMMY234	Save area for registers 2 - 5
460	1CC	572	DUMMYPLH	Dummy PLH for LSR
1032	408	64	DUMLSRA	LSR savearea
1096	448	52	DUMDBHD	Dummy data buffer header
1148	47C	52	DUMIBHD	Dummy index buffer header

Figure 5.53 **Close Work Area description and format (part 3 of 3)**

DADSM Parameter List

The DADSM parameter list contains the information required by the DASD Space Management modules. Its address is held in register 1.

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
0	0	6	DADVOLID		Volume identifier
6	6	2	DADSYSLN		Number of system LUB
8	8	2	DADSFLAG		Processing flags
					First byte:
			DADRF1	X'80'	Read format 1 label
			DADFR4	X'40'	Read format 4 label
			DADRADDR	X'20'	Address for read
			DADBYPS	X'10'	Bypass volume 1 checking
			DADNEXT	X'08'	Get next label
			DADLKLBL	X'04'	IKQLAB has been called
			DADSPROT	X'02'	Scratch/rename protected files
			DADSHR	X'01'	Shared VTOC access
					Second byte:
			DADSVTOP	X'80'	VTOC open indicator
			DADSMSG	X'40'	Message flag
			DADSNOP	X'20'	Bypass VTOC open
			DADSFBA	X'10'	Fixed block device
			DADSCNCL	X'08'	Operator reply was cancel
				X'04'	Reserved
				X'02'	Reserved
				X'01'	Reserved
10	A	1	DADSRC		DADSM return code
11	B	1	DADCODEA		Return code save area
12	C	4	DADSLADD		Address of label record
16	10	4	DADSAREA		Address of I/O area
20	14	44	DADEXIST		Old data set name
20	14	44	DADSRDSN		Returned data set name
64	40	2	DADCOV1		Check for overlaps return area
66	42	44	DADCREAT		New data set name
110	6E	2	DADCOV2		Check for overlaps return area
112	70	4	DADSEXIT		DADSM exit address
116	74	16	DADSWORK		Work area
116	74	4	DADSWA		Work area
116	74	4	DADPARM1		Parameter 1
120	78	4	DADSAVE		Save area for scratch
120	78	4	DADPARM2		Parameter 2
124	7C	8	DADATE		Date
132	84	8	DADFLPTR		File pointer
140	8C	2	DADBLPTK		Number of blocks per track
142	8E	2	DADALU		Round allocation to this unit
144	90	24	DADCVHPL		CVH parameter list

Figure 5.54 **DADSM Parameter List description and format**

Define the File Indexed Sequential (DTFIS) Table

The DTFIS table is provided by the user program and contains all the information needed to process a specific ISAM file. Part of it is used by IIP when a VSAM data set is to be processed by an ISAM program. If this is the case, the DTFIS table is reformatted at OPEN time by IIPOPEN.

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
0	0	16	DTFCCB		
2	2	1	DTFCCBB2		
2	2		ERREXT	X'10'	Accept physical I/O error
16	10	1	FLAGBYTE		
			AMODTF	X'80'	VSAM bit (set to 1 if DTF belongs to a VSAM data set)
				X'20'	Assign 'ignore' bit
17	11	3	LOGMODAD		Address of logic module; if AMODTF is set to 1, then address of branch vector
20	14	1	FILETYPE		File type
			LOAD	X'24'	LOAD-type DTF
			ADD	X'25'	ADD-type DTF
			RETRVE	X'26'	RETRIEVE-type DTF
			ADDRTR	X'27'	ADD-RETRIEVE-type DTF
21	15	1	OPTIONS1		Options byte 1 (ISAM options)
			BLKDRECS	X'08'	Blocked records
22	16	7	FNAMEDTF		File name (DDname)
29	1D	1	OPTIONS2		Options byte 2 (not used by IIP)
30	1E	1	FNAMEC		Status byte
			LOAD files:		
			UNCIOERR	X'80'	Uncorrectable DASD I/O error
			WRGLEN	X'40'	Wrong length record (not used by IIP)
			PDARFULL	X'20'	No more VSAM data space available
			CYLXFULL	X'10'	No more VSAM data space available
			MASXFULL	X'08'	No more VSAM data space available
			DUPREC	X'04'	Duplicate record
			SEQCHECK	X'02'	Sequence check
			PDAROVFL	X'01'	Prime data area overflow (not used by IIP)

Figure 5.55 **Define the File Indexed Sequential (DTFIS) table description and format (part 1 of 3)**

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
			Non-LOAD files:		
			UNCIOERR	X'80'	Uncorrectable DASD I/O error
			WRGLEN	X'40'	Wrong length record (not used by IIP)
			EOF	X'20'	End of file
			NORECFND	X'10'	No record found
			ILLEGID	X'08'	Illegal identifier specified (not supported by IIP)
			DUPREC	X'04'	Duplicate record
			OFARFULL	X'02'	No more VSAM data space available
			OVFLREC	X'01'	Overflow record (RETRVE) (not used by IIP)
31	1F	12			Not used by VSAM
43	2B	1	RTRBYTE		RETRVE byte
			WORKR	X'80'	WORKR set to 1 if WORKR specified
			WORKS	X'40'	WORKS set to 1 if WORKS specified
44	2C	4	AMDTFAD		Address of AMDTF
48	30	4	CIPROCAD		Address of IIP processor
52	34	4	SAVERG		Save area for one register
56	38	4	PPRETAD		Return address to problem program if called from a $$B phase
60	3C	4	RECLOC		Address of record for LOAD IOREG
64	40	1	CISWITCH		IIP switches
			WNKA	X'80'	Write-new-key-add bit
			RKWK	X'40'	Read-key-write-key bit
			RK	X'20'	Read-key bit
			FIWRITE	X'08'	First write after SETFL
			FIWOK	X'04'	First write is all right
			LD	X'02'	LOAD
65	41	9			Not used by VSAM
74	4A	2	LRECLEN		Logical record length
76	4C	2	KEYLEN		Key length
78	4E	16			Not used by VSAM
94	5E	2	KEYLOC		Key location (not used by IIP)
96	60	4	KARGAD1		Address of KEYARG, moved from part 2 by IIPOPEN if RTR SEQ with KEY (POINT) or RTR RAN is specified
100	64	2	DSPLPRT2		Displacement of part 2 (ADD, RTR)
102	66	2	DSPLPRT3		Displacement of part 3 (ADD, RTR)

Figure 5.55 **Define the File Indexed Sequential (DTFIS) table description and format (part 2 of 3)**

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
104	68	4	LDIOREGS		For RTR SEQ: if WORKS=1, then NOP; if WORKS=0, then L IOREG, RECLOC
108	6C	4	LDIOREGR		For RTR RAN: If WORKR=1; then NOP; if WORKR=o, then L IOREG, RECLOC
112	70	4	WORKAD1		Address of WORKR moved from part 2
116	74	4	IOASAD1		Address of IOAREAS moved from part 2
120	78	64	SAVAR1		For LOAD-type DTF, save area for IIPOPEN
184	B8	4	IORLAD		Address of IOAREAL for LOAD
188	BC	4	DATIWLAD		Address of data in WORKL for LOAD
192	C0	4	KEYIWLAD		Address of key in WORKL for LOAD
196	C4	4			Not used by VSAM
200	C8	1	MIXEXTI		Extend indicator for LOAD
			CROREXT	X'10' X'00'	Extending file Creating file
201	C9	3			Not used by VSAM
204	CC	4	WORKLAD		Address of WORKL for ADD
208	D0	16			Not used by VSAM
224	E0	2	KLM1		KEYLEN-1 for LOAD
			Part 2 of DTF		
0	0	8			Not used by VSAM
8	8	4	IOASAD2		Address of IOAREAS
12	C	4	IOARAD		Address of IOAREAR
16	10	4	KARGAD2		Address of KEYARG
20	14	4	WORKRAD2		Address of WORKR
24	18	4	CURIOAAD		Address of current sequential I/O area
28	1C	4	LIOREGS		L IOREG, *-4 or NOP (RTR SEQ)
32	20	36			Not used by VSAM
68	44	2	NTAGRECS		Number of records tagged for deletion
70	46	2	LIOREGR		LR IOREG,) or NOP(RTR RAN)
			Part 3 of DTF		
0	0	64	SAVAR2		Save area for IIPOPEN, not LOAD type

Figure 5.55 **Define the File Indexed Sequential (DTFIS) table description and format (part 3 of 3)**

Extent Definition Block (EDB)

The EDB describes all extents of the space allocated to the cluster's data set. The EDB is built by the Open module from information in the data set's catalog record. The AMDSB contains the length of the EDB (AMDLEDB), the number of EDB entries (AMDNEDB) that follow the header, and the address of the first EDB (AMDFSEDB). Each EDB entry describes an extent, and contains the address of the associated LPMB.

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
0	0	4	EDBNEDB		Address of next EDB
4	4	2	EDBSYMU		Symbolic unit (for CCB)
4	4	1	EDBSUCLS		Symbolic unit class
5	5	1	EDBSUNUM		Symbolic unit number
6	6	2	EDBNUMTR		Number of tracks of extent
8	8	1	EDBFLGS		Flags
			EDBDWSS	X'80'	Data RBA with sequence set
			EDBSSWD	X'40'	Sequence set RBA with data
			EDBIXREP	X'20'	Index replication
			EDBMNT	X'10'	Volume mount flag
			EDBLGCC	X'08'	Device contains more than 256 cylinders
			EDBRPS	X'04'	Indicator for RPS device
			EDBARCH	X'02'	Extent is on a fixed block device
9	9	3	EDBMBB		Extent (M) and bin (BB) number
9	9	1	EDBM		Extent (M) number
10	A	2	EDBBB		Bin (BB) number
12	C	4	EDBLBBBB		Starting block of a fixed block extent
12	C	8	EDBXTNT		Combined name for low and high CCHH
12	C	4	EDBLCCHH		Low cylinder and head numbers
12	C	2	EDBLCC		Lowest cylinder
14	E	2	EDBLHH		Lowest head
16	10	4	EDBHBBBB		Ending block of a fixed block extent
16	10	4	EDBHCCHH		High cylinder and head numbers
16	10	2	EDBHCC		Highest cylinder

Figure 5.56 **Extent Definition Block (EDB) description and format (part 1 of 2)**

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
18	12	2	EDBHHH		Highest head
20	14	4	EDBLPMBA		Address of associated LPMB
24	18	4	EDBPARDB		Address of ARDB
28	1C	2	EDBVLSQ		Index to the VOLSER list
30	1E	2	EDBSTTRK		Relative track address of the start of the extent (zero for fixed block devices)
32	20	8	EDBRBAS		Combined name for low and high RBAs
32	20	4	EDBLORBA		Low RBA limit
36	24	4	EDBHIRBA		High RBA limit
40	28	4	EDBTLBCA		Total number of data blocks and sequence set blocks per CA, minus 1 (fixed block devices).
44	2C	4	EDBCASZ		Number of bytes per control area
48	30	4	EDBTRKCA		Number of tracks per control area
48	30	4	EDBTPBCA		Total number of data blocks, sequence set blocks, and wasted blocks per CA (fixed block devices).

Figure 5.56 Extent Definition Block (EDB) description and format (part 2 of 2)

Open ACB List (OAL)

The OAL is a list which contains all VSAM ACBs that have been opened. It is built by IKQOPN. The addresses of the ACBs are also entered by IKQOPN. OAL entries are deleted by IKQCLO00 and $$BACLOS. The field at displacement X'0C' of the Anchor Table contains the address of the OAL.

One OAL contains 16 bytes of "header" information plus 1 to 14 8-byte fields, each representing an open ACB. If there are more that 14 open ACBs, another OAL header is created, along with entries for the additional ACBs.

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
0	0	1	OALID		OAL identifier
			OALIDD	X'F0'	OAL identifier equate
1	1	1			Reserved
2	2	2	OALLEN		Length of this block (128)
4	4	4	OALNOAL		Pointer to next OAL (if more than 14 ACBs exit)
8	8	2	OALNOPN		No. of open data sets in the partition
10	A	2	OALNENT		No. of OAL entries (14)
12	C	4			Reserved
16	10	0	OALENTRY		Entry description (repetitive)
16	10	4	OALACB		Address of opened ACB
20	14	1	OALSVC		Delimiter (X'0A')
21	15	1	OALFLG		Flag byte
			OALACT	X'80'	ACB is open
22	16	2	OALCICHK		Value to check validity of control interval number of data set in catalog. Value is formed at open time by adding the third byte of AMDDSM to the first two bytes of AMDDSM.

Figure 5.57 **Open ACB List (OAL) description and format**

Open Work Area (OPNWA)

The Open Work Area is built when a VSAM data set is opened or is being created. It contains the addresses of control blocks and work areas needed when a data set is being opened. The Open Work Area also contains flags that indicate the type of processing being performed on the data set and the address of the SVC that invoked the processing. The Open Work Area is pointed to by the AMBL and register 4 (RWKA) during open.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	20	WAPASS	Used to pass information from OPEN to CLOSE
0	0	4	WAEYCAT	Eye catcher
4	4	1	DISP	Job control disposition
		1... ...	NEW	DISP=NEW
		.1..	OLD	DISP=OLD
		..xx		Reserved
		.... 1...	KEEP	DISP=(,KEEP)
		.... .1..	DELETE	DISP=(,DELETE)
		.... ..1.	DATE	DISP=(,DATE)
		x		Reserved
5	5	1	DSATTRIB	Data set attributes
		1...	REUSE	Defined reusable
		.1..	NOALLOC	Defined NOALLOCATE
		..1.	WAIFPDF	Defined implicitly
		...x xxxx		Reserved
6	6	3	EXPDATE	Expiration date
9	9	1		Reserved
10	A	2	WAFEATSW	Feature switches
			First Byte	
		1...	WAFDET	Features determined
		.xxx xxx		Reserved
			Second Byte	
		xxxx xxx.		Reserved
		1	WAFSPMGT	Space management feature in system
12	C	8		Reserved
20	14		WACOMACB	Part common to all ACBs in close
0	0	1	WAFLAG	Flag byte:
		1...	TCLOSE	Work Area for TCLOSE
		.1..	CLOSE	Work Area for Close
		..1.	OPEN	Work area for Open
		...1	OPAMDINX	Index AMDSB is being processed
		.... 1...	VOLFOUND	Volume serial number is in label area record
		.... .1..	SSFLAG	Sequence set with data
		.... ..1.	RETRY	Catalog should be reupdated by Close
		1	FILEPROT	DOS Supervisor DASD file protect
21	15	1	WAERCODE	Error condition code
22	16	2	WALEN	Length of GETVIS area
24	18	4	WAWORK1	General work field
28	1C	4	WALISTP	Address of user ACB/DTF list

Figure 5.58 Open Work Area (OPNWA) description and format (part 1 of 8)

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
32	20	4	WACOMR	Address of partition COMREG
36	24	4	WADSAAD	Address of dynamic save area
40	28	2	WADSALN	Length of dynamic storage area (DSA)
42	2A	1	EDBCODE	Code to remember EDBs/GETVIS
43	28	1		Reserved
44	2C	4	CATEXTPT	Pointer for extents
48	30	2	CATEXTLN	Length of total extents
50	32	2	EXTNUMB	Number of extents
52	34	80	USERSAVE	Room to save user jobname, PSW, and registers (from partition save area)
132	84	8		Reserved
140	8C	12	PARM	Parameter list for IKQOCSHR
140	8C	1	WAOPCODE	Operation code used for determining function
141	8D	7	DSID	Data set ID
141	8D	3	DSCI	CI number
144	90	4	CTACBPTR	Pointer to catalog ACB
148	94	1	SHAREOPT	Share option from catalog
		1...	WASHR1	Share option 1 file
		.1..	WASHR2	Share option 2 file
		..1.	WASHR3	Share option 3 file
		...1	WASHR4	Share option 4 file
		 xxxx		Reserved
149	94	1	WAPFLAG	Option flags
		1...	WAPOUT	1 - lock/unlock for output access
		0...		0 - lock/unlock for input access
		.xxx xxxx		Reserved
150	96	2	OUTCNT	Indicator of output user count
152	98	24	WAOCDTL	Area to contain DTL (must be large enough to hold MAPDTL)
176	B0	0	WACOMEND	End of common work area
176	B0	0	OWA	Partial map of work area obtained by GETVIS issued by $$BOVSAM
176	B0	4	WAVSLOD	Address of location where VSAM has been placed by CDLOAD (set by $$BOVSAM)
180	84	4	WAIKQLAB	Address of location where IKQLAB has been placed by CDLOAD (set by $$BOVSAM)
184	B8	4	CLWAAD	Close work area address saved
188	BC	1	LBLRCLEN	Length of work area pointed to by LABICPTR in multiple of 128
189	BD	3	LABICPTR	Pointer to work area reserved for label record
192	C0	4	SVCATACB	Pointer to catalog ACB
196	C4	4	CTGPLPTR	Pointer to catalog parameter list (CPL)
200	C8	4	CATWKPTR	Pointer to catalog work area (CTGWA) (contents moved to CPL)

Figure 5.58 **Open Work Area (OPNWA) description and format (part 2 of 8)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
204	CC	4	OLDEDB	Address of EDB
208	D0	4	NXTEDB	Address of next EDB
		Catalog Field List for AMDSB		
212	D4	0	FLAMDSB	Catalog field list work area (CTGFLDWA) for AMDSB
212	D4	0	SAVERET1	Pointer to contents of return register (R 14) if not catalog call
212	D4	4	RETREG1	Return address to save area 1
216	D8	0	SAVERET2	Return address to save area 2
216	D8	4	RETREG2	Return address to save area 2
220	DC	0	FLAMDSBN	Pointer to catalog field name AMDSBCAT
220	DC	4	RETREG3	Return address to save area 3
224	E0	4	*	Next CTGFLCHN number
228	E4	4	FLAMDSBL	Length of AMDSBCAT
232	E8	4	FLAMDSBA	Address of AMDSBCAT
		Catalog Field List for Volume Entry(ies)		
236	EC	8	FLENTVOL	
236	EC	2	KRNKEYS	No. of key ranges equals number of ARDBs
238	EE	2	KRNVOLS	Number of volumes for this key range
241	F4	4	FLVOLNTN	Volume entry name ENTVOL
248	F8	4	*	
248	F8	4	SVLENG	Length of ENTVOL
252	FC	4	VOLENTLN	Length of volume entry
256	100	4	VOLGPPTR	Address of ENTVOL data
		Catalog Field List for Data Set Attributes		
260	104	20	*	DSATTR field list
280	118	4	FLDSATRA	Base of DSATTR
		Catalog Field List for Open Indicator		
284	11C	8	FLOPNIND	Locate OPENIND and test for UPD
292	124	4	FLOPNINN	Open indicator field list
296	128	4	*	Chain
300	12C	4	FLOPNINL	Length of OPENIND
304	130	4	FLOPNINA	OPENIND address
		Catalog Field List for Minimum Buffer Size		
308	134	20	*	Flags, etc., for BUFSIZE
328	148	4	FLBUFSZA	Base of BUFSIZE

* Multi-use field

Figure 5.58 **Open Work Area (OPNWA) description and format (part 3 of 8)**

Offset Dec	Hex	Bytes and Bit Pattern	Field Name	Description
		Catalog Field List for High-Used RBA per Data Set		
332	14C	20	*	Miscellany for HURBADS
332	14C	20	NVOLLIST	No. of volumes per key range (Space for 10 two-byte entries)
352	160	4	FLHURDSA	Base for HURBADS
356	164	20	*	CATFILE field list
376	178	4	FLFILTA	Base for CATFILE
		Catalog Field List for Names of Related Data Sets		
380	17C	8	FLNAMEDS	Flags for NAMEDS
380	17C	8	PARMLIST	IKQVLAB parameter list
380	17C	4	PARM1	ACB DD name
384	180	1	PARMLEN	Length of work area for IKQLAB
385	181	3	PARM2	Pointer to 'LABICPTR'
388	184	4	FLNAMDSN	Pointer to 'NAMEDS'
392	188	4	*	
396	18C	4	FLNAMDSL	Length of associated names
400	190	4	FLNAMDSA	Address of NAMEDS groups
		Catalog Field List for Entry Type and Control Interval No.		
404	194	8	*	CTGFLDWA for this field list
412	19C	4	FLMISCLN	Pointer to 'DSTYPNAM'
416	1A0	4	*	Chain
420	1A4	4	FLMISCLL	Length of DSTYPNAM
424	1A8	4	FLMISCLA	Address of DSTYPNAM
		Catalog Field List to Find Catalog ACB Address		
428	1AC	20	FLCATACB	Field list #10 for catalog ACB
448	1C0	4	FLCTACBA	Pointer to catalog ACB pointer
452	1C4	20	FLEXPDT	Field list #11 for expiration date
472	1D8	4	FLEXPDTA	Pointer to expiration date from catalog
		Catalog Field List to Test for Write of Open Indicator		
476	1DC	8	FLWOPNND	Update OPENIND field list
476	1DC	4	TSTENTVL	Address of test ENTVOL (scan)
480	1E0	4	TSTENTLN	Address of end scan ENTVOL
484	1E4	4	FLWOPNNN	Pointer to 'OPENIND'
488	1E8	4	*	Chain
492	1EC	4	FLWOPNNL	Length of data
496	1F0	4	FLWOPNNA	Pointer to data
		Catalog Field List for Volume Time Stamp		
500	1F4	24	FLTMSTVF	VOLTSTMP field list
500	1F4	20	*	Greater part of field list
520	208	4	FLTMSTVA	Pointer to 'VOLTSTMP'

* Multi-use field

Figure 5.58 **Open Work Area (OPNWA) description and format (part 4 of 8)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
		End of Catalog Field List for Volume Time Stamp		
524	20C	1	WARNFLG	Used to save warning error code
525	20D	3	WAENTID	Entry ID (Catalog CI#)
528	210	2	LIMIT	Count of the ENTVOLS pointed to by VOL20PT
530	212	2	RELGP	Relative group number in the catalog
532	214	2	TEMP	Local calculations (on same listing page)
534	216	2	IARDB	Index for ARDB list
536	218	4	SAVDEV	Used to save device type
540	21C	4	SAVDEV2	Used to save sequence set device type
544	220	2	SAVTRKAU	Used to save number of tracks per allocation unit (control area) to help identify type of LPMB
546	222	2	SAVTRKA2	Same as SAVTRKAU but used only if sequence set with data
548	224	4	RLPMB2	Pointer to sequence set (index) LPMB
552	228	1	OWAFLAGS	Open flags and switches
		1...	OWFLAGZB	User did not specify buffer size in ACB
		.1..	OWFLAGBF	BCB building in process
		..1.	OWFLAGIB	Got buffer with AMBL for index
		...1	WARSOPEN	USE macro has been issued for SYSO-PEN (RELEASE macro must subsequent-ly be issued)
		.... 1...	DTACNT	Open count in look-aside table is bumped for data
		.... .1..	IDXCNT	Open count in look-aside table is bumped for index
		x		Reserved
553	229	3	INDEXSAV	Used to save index file name
556	22C	1	SAVTYPE	Used to save entry type when entry is not a cluster
557	22D	2		Reserved
559	22F	4	TESTSV1	Save a word for testing
563	233	1	SVOPNIN	Updated OPENIND for catalog
		1...	SVOPNINO	Flag open for output
		.xxx xxxx		Reserved
564	234	2	SVNEXTNT	Save number of EXTENT statements
566	236	2	SETNBUF	Count of buffers (used by SETADDR)
568	238	4	VOLSTPTR	Address of volume list
572	23C	4	VOLENTND	End of all ENTVOLs
576	240	2	VOLENTCT	Count of volume entries
578	242	2	IVOLS	Working index of ENTVOLs

Figure 5.58 **Open Work Area (OPNWA) description and format (part 5 of 8)**

Offset Dec	Hex	Bytes and Bit Pattern	Field Name	Description
580	244	4	VOL20PT	Pointer to volume entries to sort (address of VOLENT20 if less than 20)
584	248	80	VOLENT20	Volume entries to sort
664	238	4	VMTPTR	Pointer for right VOLSER
668	29C	4	REQBUFSP	Minimum buffer space required
672	2A0	4	CURBUFSP	Currently specified buffer space
676	2A4	4	CURBFSPD	Current buffer space specified for data
680	2A8	32	ADDAREA	Room to add without current specifications for index
680	2A8	4	CURBFSPI	Current buffer space specified for index
684	2AC	4	WRKCINV	Working value of CI
688	2B0	1	SVLUBPUB	Save index of PUB
689	2B1	1	NEXTJIB	Next JIB saved
690	2B2	10	SVPUB	LUBs for mounted volumes
700	2BC	2	IPUB	Index for SVPUB
702	2BE	8	OWAPRTCT	Room to build password
710	2C6	2		Unused
712	2C8	80	DUMCATPL	Room for catalog parameter list
792	318	512	OWACTWKA	Normal catalog work area
1304	518		CCWX	CCW definition
1304	518	1	CCWCODE	Write-to-console op code
1305	519	3	CCWDTA	Pointer to message buffer
1308	51C	2	*	
1310	51E	2	CCWCNT	Length of message buffer
1312	520	24	CCBX	CCB definition
1312	520	9	*	
1321	529	3	CCWPT	Pointer to channel program (CCWX)
1324	52C	12		Unused

* Multi-use field

Figure 5.58 **Open Work Area (OPNWA) description and format (part 6 of 8)**

Offset Dec	Hex	Bytes and Bit Pattern	Field Name	Description
1336	538	0	VMSG	Volume name is built and used as part of calling parameter when catalog is called to get the time stamp
1336	538	0	MSG	Volume time stamp built
1336	538	11	MSGID	Message identification
1347	543	8	MSGDSN	Data set name
1355	54B	46	MSGTXT	Message text
1401	579	3	*	
1404	57C	4	OWSTRTGV	Start of GETVIS
1408	580	4	OWAOAL	Address of OAL section
1412	584	4	UACBAD	User's ACB address
1416	588	4	AIXACBAD	AIX cluster ACB address
1420	58C	4	BCACBAD	Base cluster ACB address
1424	590	4	UPACBAD	ACB of upgrade member
1428	590	4	USBAD	Pointer to USB
1432	598	4	OWAUCPT	Pointer for IKQLAB
1436	59C	24	FLRGATTR	Copy of CTGFL for 'RGATTR'
1436	59C	16	*	
1452	5AC	4	*	Length of 'RGATTR' data
1456	5B0	4	FLRGATRA	Pointer to 'RGATTR'
1460	5B4	24	FLEXCPEX	Copy of exception exit CTGFL
1460	5B4	16	*	
1476	5C4	4	FLEXCEPL	Length of exception exit data
1480	5C8	4	FLEXCEPA	Address of EXCPEXIT
1484	5CC	80	INTCPL	Internal CPL
1564	61C	4	RPLPAD	RPL pool just handled
1568	620	4	PLHADDR	Address of first PLH
1572	624	4	AIXBUFAD	Upgrade buffer pool
1576	628	4	AIXBUFLN	Length of upgrade buffer pool
1580	62C	24	MSGPARMS	Parameter list for IKQOCMSG
1604	644	2	MSGFLGBT	Message flag byte
1606	646	2	NRPL	Number of user strings
1608	648	2	AIXBCLEN	GETVIS length for ACB/RPL
1610	64A	2	UPGRM	Number of members in upgrade set
1612	64C	2	UPGRCT	Upgrade set loop counter
1614	64E	2	AIXUPLEN	Length of upgrade set (RPL+PLH)

* Multi-use field

Figure 5.58 **Open Work Area (OPNWA) description and format (part 7 of 8)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
1616	650	1	AIXFLG	Alternate index flags
		1...	AIXUPGR	Upgrade set available
		.1..	AIXBASE	Base cluster handled
		..1.	AIXPE	Path entry handled
		...1	AIXPEU	Path entry of upgrade set
		.... 1...	AIXPATH	Path structure open
		.... .1..	AIXMUS	Member of upgrade set handled
		.... ..1.	AIXEUO	AIX as end-use object
		1	AIXUSERR	Upgrade set error
1617	651	1	AIXFLG2	Alternate index flags 2
		1...	AIXTHB	THB is for upgrade set
		.1..	AIXPEUBF	One additional index buffer already present
		..xx xxxx		Reserved
1618	652	1	PATHFLG	Path flags, first byte
		1...	PFLUPD	Update option
		.xxx xxxx		Reserved
1619	653	1	*	Path flags, second byte
		1...	RESETSW	Switch for reset of data set at open time
		.1..	ESDSERR	ESDS error flag
		..1.	OALEFND	OAL entry found
		...1	JRNACT	JRNAD active
		.... 1...	CATOPEN	Catalog open in progress
		.... .1..	REPETSW	Switch for allocate
		.... ..1.	VALCKSW	Validity check switch
		1	WARECLUS	Indicates a cluster refresh of catalog information is required
1620	654	1	SAVAIX	Save area for AIXFLG
1621	655	2	AIXUSAV	Save area for ACB option
1623	657	3	AIXYENTR	Internal address of Y entry
1626	65A	3	AIXDNAM	AIX data name
1629	65D	3	AIXINAM	AIX index name
1632	660	3	BCDNAM	Base cluster data name
1635	663	3	BCINAM	Base cluster index name
1638	666	3	CLUNAME	Cluster name save area
1641	669	5	INTWA	Internal catalog work area
1646	66E	2	LSRCNT	Number of data sets with LSR
1648	670	512	OWA2	Work area
1648	670	512	OWAUCAT	IKQCAT work area for UCAT
1648	670	512	USCTGWA	Catalog work area IKQOPNUS
1648	670	512	OWAMSGAR	Message work area
2160	870	4	AUROOT	Address of first control block in allocation unit
2164	874	4	CPAADR	Address of channel program area
2168	878	4	WAIKQSTM	Address of storage manager module

* Multi-use field

Figure 5.58 **Open Work Area (OPNWA) description and format (part 8 of 8)**

Request Parameter List (RPL)

The RPL contains user-request information and error feedback information. It also maintains information required by GET and PUT. The RPL is created by the user with the RPL macro instruction.

Offset Dec	Hex	Bytes	Field Name	Hex. Digit	Description
0	0	1	RPLID		Control block identifier = X'00'
0	0	1	RPLIDD	X'00'	RPL equate
1	1	1	RPLSTYP		Release indicator
			RPLSDV1	X'00'	DOS/VS VSAM Release 1
			RPLSVSE1	X'10'	VSE/VSAM Release 1
				X'20'	VTAM
2	2	2	RPLLEN		Length of RPL
4	4	4	RPLRBA		RBA of last record processed
4	4	4	RPLDDDD		DD field
8	8	4	RPLARG		Pointer to search argument
12	C	8	RPLRCD		Record description
12	C	4	RPLAREA		Address of the caller's work area
16	10	4	RPLRLEN		Length of record
20	14	4	RPLBUFL		User buffer size
24	18	4	RPLACB		Address of the caller's ACB
24	18	4	RPLDACB		Catalog compatibility
28	1C	1	RPLSTRID		RPL string identifier
29	1D	1	RPLREQ		Request type*
			RPLPOINT	X'00'	POINT request
			RPLGET	X'04'	GET request
			RPLERASE	X'08'	ERASE request
			RPLPUT	X'0C'	PUT request
			RPLUPDTE	X'0C'	Update request
			RPLINSRT	X'10'	Insert request
			RPLCHECK	X'14'	Check request
			RPLRCLSE	X'18'	RCLOSE request
			RPLENDRQ	X'1C'	ENDREQ request
			RPLFRCIO	X'1C'	FORCIO request
			RPLVERFY	X'20'	VERIFY request
			RPLPUTL	X'24'	PUT locate request
			RPLWRBFR	X'2C'	Write buffer request
30	1E	2	RPLKEYL		Key length
32	20	2	RPLOPTCD		Option codes

* This value may be altered internally by VSAM, for example, X'24' from application program is changed to X'0C' by IKQRQA

Figure 5.59 Request Parameter List (RPL) description and format (part 1 of 4)

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
32	20	1	RPLOPT1		First byte of options
			RPLKEY	X'80'	Keyed access
			RPLADR	X'40'	Addressed access
			RPLSEQ	X'20'	Sequential
			RPLDIR	X'10'	Direct processing
			RPLASY	X'08'	Asynchronous
			RPLSKP	X'04'	Skip sequential access
			RPLCNV	X'02'	CNV access (RBA)
			RPLUPD	X'01'	Update
33	21	1	RPLOPT2		Second byte of options
			RPLKGE	X'80'	Search key greater than or equal
			RPLGEN	X'40'	Generic key request
			RPLNSP	X'20'	Note string position
			RPLNUP	X'10'	No update
			RPLLOC	X'08'	Locate mode
			RPLUBF	X'04'	User buffers
			RPLBWD	X'02'	Backward processing
			RPLLRD	X'01'	Last record processing
34	22	1	RPLHLD2	X'FF'	Second test and set byte (RPL not available)
				X'00'	RPL available
35	23	1	RPLHLD	X'FF'	Test and set byte (RPL held - request not completed)
				X'00'	Request completed
36	24	1	RPLFLAG		Flag byte
			RPLECBPR	X'80'	CMS ECB indicator
37	25	3	RPLFDBK		Error feedback area
37	25	1	RPLFDB1		Error class (return) code
37	25	1	RPLRTNCD		Error class code
Error class codes (stored from Register 15)					
			RPLNOERR	X'00'	No error detected
			RPLNORPL	X'04'	RPL held by another request
			RPLLOGER	X'08'	Logical error
			RPLPHYER	X'0C'	Physical error
			RPLVABND	X'3C'	TP I/O prohibited
38	26	1	RPLFDB2		Function type code
38	26	1	RPLFTNCD		Function type code
			RPLFUPG	X'04'	Upgrade processing
			RPLFAIX	X'02'	AIX processing
			RPLFINC	X'01'	Upgrade set is incorrect
39	27	1	RPLFDB3		Error type code
39	27	1	RPLERRCD		Error type code
39	27	1	RPLERCD		Error type code
39	27	1	RPLFDBKC		Error type code
The following equates are for the various feedback returns that may be set for offset 39 (27). They fall into the three categories shown.					
Returns that are not errors (Register 15 = X'00')					
			RPLEOV	X'04'	EOV called during request
			RPLDPKEY	X'08'	Duplicate key (in AIX record)
			RPLNEWCA	X'10'	Index full - CA split required.
			RPLCIWNG	X'1C'	Possible duplicate records in this CI (address processing of KSDS).

Figure 5.59 **Request Parameter List (RPL) description and format (part 2 of 4)**

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
Logical errors (register 15 = X'08')					
			RPLEOFDS	X'04'	End of data set encountered
			RPLEODER	X'04'	End of data set encountered
			RPLDUPRC	X'08'	Duplicate record
			RPLDUP	X'08'	Duplicate record
			RPLSEQCK	X'0C'	Sequence error
			RPLNRFND	X'10'	No record found
			RPLNOREC	X'10'	No record found
			RPLEXCTL	X'14'	Data already in exclusive control
			RPLNVOLM	X'18'	Volume or extent unavailable
			RPLNRSPA	X'1C'	No DASD space available
			RPLNOEXT	X'1C'	No DASD space available
			RPLSPACE	X'1C'	No DASD space available
			RPLINRBA	X'20'	Invalid RBA specified
			RPLNKEYR	X'24'	No key range for new record
			RPLNOVIR	X'28'	Insufficient virtual storage
			RPLWRKAS	X'2C'	User's work area not large enough
			RPLCDLOD	X'30'	CDLOAD failure
			RPLVLERR	X'34'	Internal VSAM logic error
			RPLNOPLH	X'40'	PLH in use (no string available)
			RPLNOPEN	X'44'	Access type not requested at Open
			RPLKEYES	X'48'	Keyed request for ESDS
			RPLADRKS	X'4C'	ADR or CNV insert for KSDS
			RPLINERS	X'50'	Illegal ERASE request
			RPLINLOC	X'54'	Illegal locate mode specification
			RPLNOPOS	X'58'	Positioning error
			RPLNGUPD	X'5C	No valid GET UPD issued
			RPLUPDKC	X'60'	Key change during update
			RPLLENCN	X'64'	Length change for addressed update
			RPLCONOP	X'68'	Improper or conflicting RPL options
			RPLIMRCL	X'6C'	Improper RECLEN specified
			RPLIMGKL	X'70'	Improper generic key length specified
			RPLINLD	X'74'	Illegal request during data set load
			RPLCATLG	X'80'	Internal catalog call failure
			RPLSRLOC	X'84'	Illegal locate mode
			RPLSRADR	X'88'	Illegal request for spanned record
			RPLINCSR	X'8C'	Inconsistent spanned record
			RPLNOBAS	X'90'	No base record
			RPLMAXPT	X'94'	Maximum of pointers exceeded
			RPLNOBUF	X'98'	No buffers available (LSR only)
			RPLINCNV	X'9C'	Invalid CI, possibly duplicate data addressed using address mode for update.
			RPLINVRR	X'C0'	Invalid relative record number
			RPLRRADR	X'C4'	Illegal address requested (RRDS)
			RPLIPATH	X'C8'	Illegal path access
			RPLINBWD	X'CC'	Illegal backward mode requested

Figure 5.59 **Request Parameter List (RPL) description and format (part 3 of 4)**

Offset Dec	Hex	Bytes	Field Name	Hex. Digit	Description
Physical errors (register 15 = X'0C')					
			RPLRDERD	X'04'	Data read error
			RPLRDERI	X'08'	Index read error
			RPLRDERS	X'0C'	Sequence set read error
			RPLWTERD	X'10'	Data write error
			RPLWTERI	X'14'	Index write error
			RPLWTERS	X'18'	Sequence set write error
40	28	4	RPLCHAIN		Pointer to next RPL
44	2C	1	RPLAIXID		AIX information byte
			RPLAXPKP	X'01'	Prime key pointers are used (base cluster is a KSDS)
45	2D	1			Reserved
46	2E	2	RPLAIXPC		Number of base cluster pointers in the AIX record
48	30	1	RPLXID		Transaction ID
49	31	3			Reserved
52	34	0	RPLEND		End of RPL

Figure 5.59 **Request Parameter List (RPL) description and format (part 4 of 4)**

VSAM Shared Resource Table (VSRT)

The VSAM Shared Resource Table contains the addresses of the resource part of the resource pool (PLH pool, buffer pool, the RSCB, and CPA pool), together with the addresses of various control blocks built during processing of the BLDVRP macro.

The VSRT is contained in the phase IKQVRT and is loaded by means of CDLOAD when required.

Offset Dec	Hex	Bytes	Field Name	Hex. Digit	Description
0	0	1	VSRTBKID	X'15'	Control block identifier
1	1	1			Reserved
2	2	2	VSRTLEN		Length of VSRT
4	4	4	VSRTID		Control block name 'VSRT'
8	8	4			Reserved
12	C	1	VSRTFLG1		Flag byte 1
			LSR	X'40'	LSR pool indicator
13	D	1	VSRTFLG2		Flag byte 2
14	E	2	VSRTUNCT		Number of data sets opened with the LSR option
16	10	4	VSRTECB		ECB for the VSRT
18	12	1	VSRTCOM		Communication byte
			VSRTWAIT	X'80'	Wait flag
19	13	1	VSRTGATE		Exclusive control gate: X'00' = ECB is free X'FF' = ECB is in use

Figure 5.60 **VSAM Shared Resource Table (VSRT) description and format (part 1 of 2)**

Offset Dec	Offset Hex	Bytes	Field Name	Hex. Digit	Description
20	14	4	VSRTRP		Address of the control block part of the resource section of the VRP
24	18	4	VSRTRLEN		Length of the control block part
28	1C	4	VSRTPLHA		Address of the PLH pool in the resource section
32	20	1	VSRTSTRN		Total number of PLHs required for all data sets sharing the resource pool
33	21	1	VSRTKL		Maximum key length of the data sets sharing the resource pool
34	22	2	VSRTPLHL		Length of each PLH in the PLH pool
36	24	4	VSTRSCB		Address of the RSCB
40	28	4	VSRTBUFH		Address of buffer pool
44	2C	4	VSRT1STB		Address of first buffer
48	30	4	VSRTMBSZ		Maximum buffer size in buffer pool
52	34	2	VSRTSPNO		Number of subpools in buffer pool
54	36	2	VSRTBFNO		Number of buffers in buffer pool
56	38	4	VSRTBFLN		Length of the buffer part of the resource section
60	3C	4	VSRTCPAH		Address of the channel program area pool in the resource section
64	40	4	VSRTCPAL		Length of the channel program area pool
68	44	4	VSRTTHBA		Address of the THB pool
72	48	4	VSRTTHBL		Length of the THB pool

Figure 5.60 **VSAM Shared Resource Table (VSRT) description and format (part 2 of 2)**

Section 6. Diagnostic Aids

This chapter provides several aids that can be useful when trying to diagnose difficulties with VSAM modules. These aids include:

- A list of VSAM lock resource names (Figure 6.1) and their associated use by VSAM.
- A chart (Figure 6.2) showing the lock option/control for locking various types of files.
- A list of macro instructions (Figure 6.3) issued by VSAM users, modules or other macros and their use.
- Cross reference tables (Figures 6.4 and 6.5) showing the VSAM modules and the macros they issue.
- A description of the Catalog Communication Area, Register Save Area and a list of error codes (Figure 6.6), set in the CCA by catalog management modules, together with the reason codes belonging to each error code.
- A list of return codes (Figure 6.7) set in register 15 which indicate DADSM conditions when processing is completed.
- A list of error codes (Figure 6.8) showing control block manipulation modules and the error code(s) they issue.
- A list of error codes (Figure 6.9) which may be issued by OPEN modules.
- A list of error codes (Figure 6.10) which may be issued by CLOSE and TCLOSE modules.
- A list of error codes (Figure 6.11) which may be issued by the SHOWCAT module.
- A list of error codes (Figure 6.12) which may be issued by the BLDVRP and DLVRP modules.
- A list of calling procedures and calling sequence for catalog management procedures (Figure 6.13).
- A description of service aid phases and how to use them.

Additional Aids

Further aids can be found in other parts of the book and in the program listings. These include:

- Register contents on entry to a module, which are under *Input* in the module prologues.
- Use of registers and equated names for registers, which can be found under *Notes* in the module prologues.
- Error codes, which are under *Exit-Error* in the module prologues.
- A list, which is in the *Directory*, of modules, their component, their entry points, and their associated method of operation and program structure diagrams.
- A cross-reference list, which is in the *Directory*, of catalog external entry points and their associated modules.

VSAM Use of Locks

Figure 6.1 is a list of the lock resource names used by VSAM and their associated functions.

Resource Name	Function
V.addr.CAX.X'0000'	Serialize access on the C/M CAXWA chain during delete, update, or search operations on the chain.
V.OAL.X'0000000000000000'	Maintains integrity of OAL by serializing access through OPEN/CLOSE.
V.SYSMCO.X'0000000000'	Serialize master catalog define and open.
V.SYSOPEN.X'00000000	Serialize OPEN, CLOSE, DELETE, and DEFINE access to the catalog (e.g. OPEN indicator) and synchronize the catalog with share options locks.
V.volser.ci#.X'0000'[1]	File lock - Used to enforce SHAREOPTIONS protection for components of a file. The name volser.ci# uniquely identifies a component being protected. The volser is the serial number of the volume containing the catalog describing the component and ci# is the number of the control interval in the catalog where the component is being described.
V.volser.ci#.X'0001'	Outcount lock - Maintains a count of output users of the file denoted by volser.ci#. This lock is maintained for SHAREOPTIONS(3) and SHAREOPTIONS(4) files.
V.volser.ci#.X'0002'	Keyed access lock - Represents keyed access for output to a SHAREOPTIONS(4) file. It is used together with the address access lock to prevent concurrent keyed access and address access for output to a SHAREOPTIONS(4) file.
V.volser.ci#.X'0003'	Address access lock - Represents address or CNV access for output to a SHAREOPTIONS(4) file.
V.volser.ci#.X'0004'	Used by Record Management to serialize use of the Upgrade Set Block (USB) when the ACB has been opened with multiple strings.
V.volser.ci#.X'0005'	Used by Record Management to serialize allocation of control areas within an extent of a SHAREOPTIONS(4) file.
V.volser.X'0000000006'	Volume mount serialization - Used to synchronize mount requests for a given volume.
V.volser.ci#.X'nnnn'	Used for Record Management basic SHAREOPTIONS(4) locks on control areas, where nnnn is the CA number (CI number for index component) plus 1024.
V.volser.UPL.X'0000'	Serialize master and user catalog update and locate functions.

(Note: The period (.) as used in this list of lock resource names, represents concatenation only and is not part of the lock resource name.)

[1] The file lock is maintained by open/close using OWNER=PARTITION so that an ACB may be closed by a different task than the opening task. Figure 6.2 shows which lock option/control is used for locking various types of files.

Figure 6.1 VSAM Use of Locks

File being opened for:	File defined share option			
	1	2	3	4
INPUT	LOCKOPTION=1 CONTROL=SHARED	LOCKOPTION=2 CONTROL=SHARED	LOCKOPTION=3 CONTROL=SHARED	LOCKOPTION=4 CONTROL=SHARED
OUTPUT	LOCKOPTION=1 CONTROL=EXCLUSIVE	LOCKOPTION=2 CONTROL=EXCLUSIVE	LOCKOPTION=3 CONTROL=EXCLUSIVE	LOCKOPTION=4 CONTROL=EXCLUSIVE

Figure 6.2 Lock option/control for locking various types of files

Macro-to-Module Relationships

Figure 6.3 contains the macro instructions issued by VSAM users, modules, or other macros. Their types are identified as follows:

G – generating macro
SA – VSE action macro
M – mapping macro
I – internal (called by another macro)
A – VSAM action macro
S – copy source book macro

Macro	Type	SVC	Use
ACB	G		Generate an ACB
ASYSCOM	SA		Get address of systems communications region
AVRLIST	M		With DCTENTRY, map device characteristics
CANCEL	SA		Cancel a task
CATLG	A		Load address of catalog parameter list (CTGPL) into R1 and invoke catalog management
CCB	G		Build a CCB
CDLOAD	SA	65	Load module(s)
CLOSE	SA	2	Disconnect a user's program from a VSAM data set
COMRG	SA	33	Get communication region address
CVTOC	SA		CVH close VTOC
DCTENTRY	M		With AVRLIST, map device characteristics
DEQB	SA		Free B-transient
DTFCN	SA		SYSLOG DTF
ENDREQ	SA		Free a PLH and terminate processing on associated string
ENQB	SA		Hold B-transient
EOJ	SA		End of job
ERASE	SA		Delete a record
EXCP	SA	0	Execute channel program
EXLST	G		Generate EXLST
EXTRACT	SA	98	Get control block information from supervisor
FREEVIS	SA	62	Free virtual storage
GENCB	A		Generate a control block
GENDTL	SA		Generate a DTL block to be used as an interface to LOCK/UNLOCK
GET	SA		Retrieve a record
GETFLD	SA	107	Get specified field value
GETVCE	SA	99	Get device characteristics
GETVIS	SA	61	Get virtual storage
IDCDF60	M		Map Access Method Services catalog communication table (ACC), catalog CI number to CRA CI number translation table (CTT), and volume timestamp table (VTT).
IGGCAXWA	M		Map catalog auxiliary work area (CAXWA)
IGGCCA	M		Map catalog communications area (CCA)

Figure 6.3 **Macro types and uses (part 1 of 5)**

Macro	Type	SVC	Use
IGGMCDCL	M		Issue the following macros to define the commonly used declarations for VSAM catalog management modules: IGGCAXWA, IGGCCA, IGGMCTRC, IKQACB, IKQAMCBS, IKQCOMRG, IKQCTGFL, IKQCTGFV, IKQCTGPL, IKQVRGN
IGGMCMDM	M		Map the VSAM catalog management commonly used record structures
IGGMCMWA	M		Map the VSAM catalog management services work area
IGGMCTRC	M		Map catalog return codes
IGGMDLWA	M		Delete work area layout
IGGMDRWA	M		Map the VSAM catalog VTOC label read-in work area
IGGMDVCH	M		Map VSAM catalog management device characteristics
IGGMEND	G		Generate exit code at the end of catalog management modules
IGGMFDNM	M		VSAM catalog dictionary information for external field names
IGGMGVO	M		Map the volume information group occurrence
IGGMNAME	I		Generate catalog module name for error and reason codes
IGGMODUL	G		Generate header code for catalog modules
IGGMPROC	G		Generate header code for catalog internal procedures
IGGMSAWA	M		Map the VSAM catalog management suballocate work area
IGGMUPDE	M		Issue IGGMVEDC, IGGMCDCL, IGGMCMDM, IKQAMDSB, and IGGMSAWA to define the commonly used declarations for VSAM catalog management Update-Extend modules
IGGMVEDC	M		Map the volume catalog record
IIPAMDTF	G/M		Generate/map AMDTF table
IIPDTF	G/M		Generate/map DTF table
IIPPRAT	G/M		Generate/map address list
IJBLBRC	M		Map label area record
IJJHCPL	M		Map CVH Parameter List
IJJHDLST	M		Map CVH Volume Descriptor List
IJJHFMT1	M		Map format-1 VTOC label
IJJHFMT3	M		Map format-3 VTOC label
IJJHFMT4	M		Map format-4 VTOC label
IKQACB	M		Map ACB
IKQACBG	I		Generate ACB (called by IKQACB1)
IKQACB1	I		Generate ACB (called by ACB)
IKQAIR	M		Map alternate index record
IKQAMBL	M		Map AMBL
IKQAMCBS	M		Map AMCBS
IKQAMDSB	M		Map AMDSB
IKQARDB	M		Map ARDB
IKQAREX	M		Map EXLST argument entry
IKQARGH	M		Map argument header
IKQASGN	A		Invoke automatic assign function
IKQBHD	M		Map buffer header
IKQBKPHD	M		Map header for CCW area
IKQBLARD	G		Build an ARDB

Figure 6.3 **Macro types and uses (part 2 of 5)**

Macro	Type	SVC	Use
IKQBUFE	M		Map BCB
IKQCBMTB	G		Define table of constants for control block generation modules
IKQCB1	I		Transform operands for control block manipulation macro instructions GENCB, TESTCB, MODCB, SHOWCB, IKQCB2, and IKQERMAC
IKQCB2	I		Scan keywords and generate code for control block manipulation macro instructions GENCB, TESTCB, MODCB, SHOWCB, IKQCB1, and IKQERMAC
IKQCCB	M		Map IORB
IKQCCBCW	M		Map IORB
IKQCCW	M		Map CCW
IKQCGETC	S		Obtain storage in which to copy old ARDB
IKQCIW	M		Map control interval split work area
IKQCLCOR	S		Get address of space in which to build EDB(s)
IKQCLNUP	S		Disconnect ACB and AMBL
IKQCLRLS	S		Free storage obtained by Open and/or EOV
IKQCLWA	M		Close work area
IKQCOMB	G		Generate a combination name entry for the VSAM catalog dictionary
IKQCOMRG	M		Map communication region
IKQCTGFL	M		Map field parameter list (CTGFL)
IKQCTGFV	M		Map catalog field vector table (CTGFV)
IKQCTGPL	M		Map catalog parameter list (CTGPL)
IKQCWS	M		Map CCW skeletons
IKQDDR	M		Map duplicate data recovery Work Area
IKQDEVT	A	65	Read label area and/or determine the device type for the file-ID (IKQDEVT uses CDLOAD)
IKQECB	M		Map Event Control Block
IKQEDB	M		Map EDB
IKQEDBLD	G		Build EDB
IKQEQU	M		Map register equates
IKQERC	M		Internal error codes equate
IKQERMAC	I		Issue M-notes (assembler macro error messages) for control block manipulation macro instructions GENCB, TESTCB, MODCB, SHOWCB, IKQCB1, and IKQCB2
IKQEXLG	I		Generate EXLST (called by IKQEXL1)
IKQEXL1	I		Generate EXLST (called by EXLST)
IKQEXLST	M		Map EXLST
IKQEXP	M		Description of EXPAD parameter list
IKQFCDB	M		Map CCW blocks in CCW pool
IKQFNDLB	S		Find LUB (logical unit block) for symbolic unit
IKQFXL	M		Map Fix List (used with IORB)
IKQGCB	I		Generate a control block (called by GENCB)
IKQDTL	G		Generate a DTL for use by the Lock manager
IKQIOARG	M		Map DASD address
IKQIODRB	M		Map I/O driver block
IKQIORQU	M		Map register equates
IKQIOWKA	M		Map I/O work area in PLH
IKQIXHDR	M		Map index record header
IKQJIB	M		Map JIB (job information block)
IKQJRNDS	M		Parameter list for journalling
IKQKWTB	G		Define table of constants for control block manipulation modules

Figure 6.3 **Macro types and uses (part 3 of 5)**

Macro	Type	SVC	Use
IKQLOCK	A	63	Lock a system resource or file by means of the LOCK macro
IKQLPMB	M		Map LPMB
IKQLUB	M		Map Logical Unit Block
IKQMCB	I		Modify a control block (called by MODCB)
IKQMDADS	M		Map DADSM parameter list (interface block to DADSM)
IKQMSGPL	M		OPEN/CLOSE message primary list
IKQOAL	M		Open ACB list
IKQOCFSP	I		Free space for DSA
IKQOCGSP	I		Get space for DSA
IKQOCPRC	S		Connect dynamic storage area
IKQPARM	M		Map Buffer Manager Parameter List
IKQOPCLR	M		Register equates
IKQOPCLW	M		Map of common section of work area
IKQOPLCT	M		Map fields located by catalog
IKQOPNWA	M		Map Open work area
IKQPLH	M		Map PLH
IKQPUB	M		Map physical unit block
IKQRDF	M		Map RDF and CIDF fields
IKQRPL	M		Map RPL
IKQRPLG	I		Generate RPL (called by IKQRPL1)
IKQRPL1	I		Generate RPL (called by RPL)
IKQRQM	G		Generate modules IKQRQA and IKQRQB
IKQSCB	I		Display a control block (called by SHOWCB)
IKQSHRW	M		Map SHRW (File Sharing Work Area)
IKQTCB	I		Test a control block (called by TESTCB)
IKQTHB	M		Map THB
IKQUNLK	M		Unlock a system resource or file by means of the UNLOCK macro
IKQUSB	M		Upgrade set block
IKQVLST	M		Map list of volume unit, symbolic unit, and volume time stamp
IKQVOL1	M		Map volume-1 label
IKQVRGN	M		Map anchor table
IKQVRPPL	M		Map parameter list for BLDVRP function (IKQBRP)
IKQVSMDP	A		Map VSAM dump

Figure 6.3 **Macro types and uses (part 4 of 5)**

Macro	Type	SVC	Use
LABEL	SA		Interface macro to call symbolic label access
LOAD	A		Load a phase
LOCK	SA	110	Lock a system resource
LPLDCT	M		Map of label parameter list
MAPBDY	M		Map for partition boundries
MAPCOMR	M		Map partition COMREG layout
MAPPIB	M		Map program information block
MODCB	A		Modify a control block
MODDTL	SA		Modify a DTL
MODFLD	SA		Modify specified field value
OPEN	SA	2	Connect a user's program to a VSAM data set
OVTOC	SA		CVH open VTOC
POINT	SA		Position VSAM at a record
POST	SA		Post an ECB
PUT	SA		Store a new or updated record
PVTOC	SA		CVH process VTOC
RPL	G		Generate an RPL
SHOWCB	A		Display a control block
SYSCOM	M		Map system communication region layout
TCLOSE	SA		Purge buffer and update catalog (no disconnect)
TESTCB	A		Test a control block
UNLOCK	SA	110	Unlock a system resource
VERIFY	A		Build calling sequence for VSAM function VERIFY
WAIT	SA	7	Wait on a CCB or IORB for I/O to complete

Figure 6.3 Macro types and uses (part 5 of 5)

Module \ Macro	ASYSCOM	CANCEL	CCB	CDLOAD	COMRG	CVTOC	DUMP	ENDREQ	ERASE	EXCP	EXTRACT	FREEVIS	GENDTL	GET	GETVCE	GETVIS	IDCDF60	IGGCAXWA	IGGCCA	IGGMCDCL	IGGMCMDM	IGGMCMWA	IGGMCTRC	IGGMDLWA	IGGMDRWA	IGGMDVCH	IGGMEND	IGGMFDNM	IGGMGVO	IGGMNAME	IGGMODUL	IGGMPROC	IGGMSAWA	IGGMUPDE	IGGMVEDC	IJBLBRC	IJJHCPL	IJJHDLST	IJJHFMT1	IJJHFMT3	IJJHFMT4	IKQACB	IKQAMBL	IKQAMCBS	IKQAMDSB
IGG0CLAB								X					X					X	X	X			X			X	X			X	X	X										X		X	
IGG0CLAC																X		X	X	X			X				X			X	X	X										X		X	
IGG0CLAD																X		X	X	X			X				X			X	X	X										X		X	
IGG0CLAE				X								X						X	X	X	X	X	X				X			X	X	X	X		X	X	X				X	X		X	
IGG0CLAF				X														X	X	X	X	X	X				X			X	X	X			X	X	X				X	X		X	
IGG0CLAG																		X	X	X		X	X		X	X	X				X	X			X							X	X	X	X
IGG0CLAH				X												X	X	X	X	X			X			X	X			X	X	X				X						X		X	
IGG0CLAJ																X		X	X	X	X	X	X			X	X			X	X	X	X			X						X		X	X
IGG0CLAK																X		X	X	X	X	X	X			X	X			X	X	X	X		X							X		X	X
IGG0CLAL					X											X		X	X	X	X	X	X			X	X			X	X	X										X		X	X
IGG0CLAN																		X	X	X	X	X	X			X	X			X	X	X										X		X	X
IGG0CLAP				X														X	X	X	X	X	X				X			X	X	X										X		X	X
IGG0CLAQ				X								X				X		X	X	X		X	X			X	X			X	X	X			X	X	X		X		X	X		X	
IGG0CLAR																		X	X	X	X		X				X			X	X	X	X		X							X		X	
IGG0CLAS																X		X	X	X	X	X	X				X			X	X	X	X									X		X	X
IGG0CLAT											X	X				X		X	X	X	X	X	X			X	X			X	X	X										X		X	
IGG0CLAU								X										X	X	X			X				X			X	X	X	X		X							X		X	
IGG0CLAV																		X	X	X			X			X	X				X	X										X		X	
IGG0CLAW																		X	X	X			X			X	X			X	X	X										X		X	
IGG0CLAX																		X	X	X			X			X	X			X	X	X										X		X	
IGG0CLAY																		X	X	X			X			X	X			X	X	X										X		X	
IGG0CLAZ																		X	X	X			X			X	X			X	X	X			X							X		X	
IGG0CLA6																		X	X	X		X	X			X	X			X	X	X			X	X	X				X	X		X	
IGG0CLA7				X					X			X		X		X		X	X	X	X	X	X	X		X	X		X	X	X	X			X	X	X				X	X	X	X	X
IGG0CLA8												X						X	X	X	X	X	X			X	X				X	X										X		X	
IGG0CLBA																		X	X	X			X			X	X	X			X	X										X		X	
IGG0CLBB												X				X		X	X	X	X		X				X			X	X	X	X	X	X							X		X	X
IGG0CLBC																		X	X	X	X		X				X			X	X	X	X	X	X							X		X	X
IGG0CLBD												X				X		X	X	X	X	X	X			X	X			X	X	X	X			X	X					X		X	X
IGG0CLBE																X		X	X	X	X	X	X			X	X			X	X	X	X									X		X	X
IGG0CLBF																		X	X	X			X			X	X		X		X	X			X							X		X	
IGG0CLBG				X								X						X	X	X	X	X	X	X		X	X		X	X	X	X			X	X	X					X		X	
IGG0CLBH				X														X	X	X	X	X	X			X	X			X	X	X										X		X	
IGG0CLBL				X														X	X	X	X	X	X	X		X	X			X	X	X			X	X	X				X	X		X	
IGG0CLBM				X								X				X		X	X	X	X		X			X	X			X	X	X					X					X		X	X

Figure 6.4 **Macro-to-module relationships for catalog and DADSM components (part 1 of 4)**

Macro / Module	IKQARDR	IKQASGN	IKQBUFE	IKQCCBCW	IKQCOMB	IKQCOMRG	IKQCTGFL	IKQCTGFV	IKQCTGPL	IKQDEVT	IKQDICT	IKQEDB	IKQGEN	IKQLOCK	IKQLPMB	IKQLUB	IKQMDADS	IKQOPCLR	IKQOPCLW	IKQPLH	IKQRPL	IKQUNLK	IKQVLST	IKQVOL1	IKQVRGN	IKQVSMDP	LOAD	LOCK	MAPCOMR	OVTOC	PUT	PVTOC	SYSCOM	UNLOCK	VERIFY	WAIT
IGG0CLAB						X	X	X	X				X	X						X	X	X			X			X						X		
IGG0CLAC						X	X	X	X					X								X			X			X						X		
IGG0CLAD						X	X	X	X													X			X									X		
IGG0CLAE						X	X	X	X	X							X				X				X											
IGG0CLAF						X	X	X	X	X						X	X				X				X		X				X					
IGG0CLAG	X					X	X	X	X			X			X						X				X											
IGG0CLAH		X				X	X	X	X	X				X								X			X			X						X		
IGG0CLAJ						X	X	X	X																X											
IGG0CLAK						X	X	X	X																X											
IGG0CLAL						X	X	X	X																X											
IGG0CLAN						X	X	X	X																X											
IGG0CLAP						X	X	X	X	X															X											
IGG0CLAQ						X	X	X	X	X							X								X											
IGG0CLAR						X	X	X	X																X											
IGG0CLAS						X	X	X	X																X											
IGG0CLAT						X	X	X	X													X			X									X		
IGG0CLAU						X	X	X	X																X											
IGG0CLAV						X	X	X	X																X											
IGG0CLAW						X	X	X	X																X											
IGG0CLAX						X	X	X	X																X											
IGG0CLAY					X	X	X	X	X		X														X											
IGG0CLAZ						X	X	X	X																X											
IGG0CLA6						X	X	X	X								X								X											
IGG0CLA7		X				X	X	X	X	X							X				X				X						X					
IGG0CLA8						X	X	X	X																X											
IGG0CLBA						X	X	X	X																X											
IGG0CLBB						X	X	X	X												X				X											
IGG0CLBC						X	X	X	X																X											
IGG0CLBD						X	X	X	X					X			X					X			X											
IGG0CLBE						X	X	X	X																X											
IGG0CLBF						X	X	X	X																X											
IGG0CLBG						X	X	X	X	X							X				X				X											
IGG0CLBH						X	X	X	X																X											
IGG0CLBL						X	X	X	X	X							X								X											
IGG0CLBM						X	X	X	X								X								X		X									

Figure 6.4 **Macro-to-module relationships for catalog and DADSM components (part 2 of 4)**

Module	ASYSCOM	CANCEL	CCB	CDLOAD	COMRG	CVTOC	DUMP	ENDREQ	ERASE	EXCP	EXTRACT	FREEVIS	GENDTL	GET	GETVCE
IGG0CLBN				X											
IGG0CLBQ															
IGG0CLBR															
IGG0CLBS															
IGG0CLBT															
IGG0CLBU															
IGG0CLBW															
IGG0CLBX				X											
IGG0CLBY															
IGG0CLB8				X								X			
IGG0CLCA															
IGG0CLCB												X			
IGG0CLCD												X			
IGG0CLCG									X					X	
IGG0CLCL															
IGG0CLCO				X								X			
IGG0CLCP															
IGG0CLCR				X								X			
IGG0CLCS															
IGG0CLCX				X											
IGG0CLCY												X			
IGG0CLC9		X			X		X	X			X	X			
IGG0CLEG										X					
IGG0CLES															
IGG0CLET				X											
IGG0CLEX												X			
IGG0CLEZ															
IGG0CLFA											X	X			
IGG0CLFB												X	X		
IGG0CLFC												X			
IGG0CLFD												X			
IGG0CLFE															
IGG0CLFF															
IGG0CLFH					X										
IGG0CLFQ															
IKQALL00				X								X			
IKQPOPOO	X											X			
IKQRDS00												X			
IKQREN00												X			
IKQSCR00												X			
IKQVTC00						X						X			
IKQWDS00												X			
$$BCLCRA	X				X										X
$$BODADE												X			
$$BODADS			X		X										

Module	GETVIS	IDCDF60	IGGCAXWA	IGGCCA	IGGMCDCL	IGGMCMDM	IGGMCMWA	IGGMCTRC	IGGMDLWA	IGGMDRWA	IGGMDVCH	IGGMEND	IGGMFDNM	IGGMGVO	IGGMNAME
IGG0CLBN	X		X	X	X	X	X	X			X	X		X	X
IGG0CLBQ			X	X	X	X		X			X	X			
IGG0CLBR			X	X	X		X	X				X			
IGG0CLBS			X	X	X			X			X	X	X		
IGG0CLBT			X	X	X		X	X			X	X	X		
IGG0CLBU			X	X	X			X			X	X			
IGG0CLBW			X	X	X			X			X	X			
IGG0CLBX			X	X	X	X	X	X			X	X			X
IGG0CLBY			X	X	X	X	X	X				X			
IGG0CLB8	X		X	X	X	X	X	X	X		X	X		X	X
IGG0CLCA			X	X	X	X	X	X				X			X
IGG0CLCB	X		X	X	X	X		X			X	X			X
IGG0CLCD	X		X	X	X	X	X	X			X	X			X
IGG0CLCG		X	X	X	X		X	X		X	X	X			X
IGG0CLCL			X	X	X	X	X	X	X		X	X			X
IGG0CLCO	X		X	X	X			X				X			X
IGG0CLCP			X	X	X	X	X	X				X			X
IGG0CLCR	X		X	X	X	X	X	X				X			X
IGG0CLCS			X	X	X	X	X	X				X			X
IGG0CLCX			X	X	X	X	X	X	X		X	X		X	X
IGG0CLCY	X		X	X	X	X	X	X			X	X			X
IGG0CLC9	X		X	X	X			X		X	X				X
IGG0CLEG			X	X	X		X	X		X	X	X			X
IGG0CLES			X	X	X	X	X					X			
IGG0CLET			X	X	X	X	X	X			X	X			X
IGG0CLEX	X		X	X	X	X	X	X			X	X			X
IGG0CLEZ			X	X	X	X		X			X	X	X		
IGG0CLFA	X			X											
IGG0CLFB			X	X	X	X	X	X				X			X
IGG0CLFC	X			X											
IGG0CLFD	X		X	X		X	X	X							
IGG0CLFE			X	X		X	X	X			X				
IGG0CLFF			X	X		X	X	X			X				
IGG0CLFH			X	X	X			X		X	X	X			X
IGG0CLFQ			X	X	X		X	X			X				
IKQALL00	X														
IKQPOPOO	X														
IKQRDS00	X														
IKQREN00	X														
IKQSCR00	X														
IKQVTC00	X														
IKQWDS00	X														
$$BCLCRA															
$$BODADE															
$$BODADS															

Module	IGGMODUL	IGGMPROC	IGGMSAWA	IGGMUPDE	IGGMVEDC	IJBLBRC	IJJHCPL	IJJHDLST	IJJHFMT1	IJJHFMT3	IJJHFMT4	IKQACB	IKQAMBL	IKQAMCBS	IKQAMDSB
IGG0CLBN	X	X				X	X					X		X	
IGG0CLBQ	X	X			X							X		X	
IGG0CLBR	X	X			X							X		X	
IGG0CLBS	X	X			X							X		X	
IGG0CLBT	X	X			X							X		X	
IGG0CLBU	X	X					X					X		X	
IGG0CLBW	X	X										X		X	
IGG0CLBX	X	X			X	X						X		X	X
IGG0CLBY	X	X										X		X	X
IGG0CLB8	X	X			X	X	X					X		X	
IGG0CLCA	X	X										X		X	X
IGG0CLCB	X	X	X	X	X							X		X	X
IGG0CLCD	X	X	X									X		X	X
IGG0CLCG	X	X			X		X					X	X	X	X
IGG0CLCL	X	X			X	X	X		X		X	X		X	
IGG0CLCO	X	X				X	X					X	X	X	X
IGG0CLCP	X	X										X		X	X
IGG0CLCR	X	X	X			X	X				X	X	X	X	X
IGG0CLCS	X	X	X			X	X				X	X		X	X
IGG0CLCX	X	X			X							X		X	
IGG0CLCY	X	X	X		X							X		X	X
IGG0CLC9							X					X		X	
IGG0CLEG	X	X			X							X	X	X	X
IGG0CLES	X	X	X									X		X	X
IGG0CLET	X	X				X						X		X	
IGG0CLEX	X	X										X		X	X
IGG0CLEZ	X	X			X							X		X	
IGG0CLFA															
IGG0CLFB	X	X													
IGG0CLFC															
IGG0CLFD															
IGG0CLFE						X	X	X	X	X	X	X		X	X
IGG0CLFF						X	X	X	X	X	X	X		X	X
IGG0CLFH	X	X	X		X										
IGG0CLFQ					X	X	X		X		X	X		X	
IKQALL00						X	X				X				
IKQPOPOO						X	X		X	X					
IKQRDS00							X	X	X		X				
IKQREN00							X								
IKQSCR00							X								
IKQVTC00							X								
IKQWDS00							X				X				
$$BCLCRA															
$$BODADE												X			
$$BODADS															

Figure 6.4 Macro-to-module relationships for catalog and DADSM components (part 3 of 4)

Module \ Macro	IKQARDB	IKQASGN	IKQBUFE	IKQCCBCW	IKQCOMB	IKQCOMRG	IKQCTGFL	IKQCTGFV	IKQCTGPL	IKQDEVT	IKQDICT	IKQEDB	IKQGEN	IKQLOCK	IKQLPMB	IKQLUB	IKQMDADS	IKQOPCLR	IKQOPCLW	IKQPLH	IKQRPL	IKQUNLK	IKQVLST	IKQVOL1	IKQVRGN	IKQVSMDP	LOAD	LOCK	MAPCOMR	OVTOC	PUT	PVTOC	SYSCOM	UNLOCK	VERIFY	WAIT
IGG0CLBN						X	X	X	X	X							X								X											
IGG0CLBQ						X	X	X	X																X											
IGG0CLBR						X	X	X	X																X											
IGG0CLBS						X	X	X	X																X											
IGG0CLBT						X	X	X	X																X											
IGG0CLBU						X	X	X	X								X								X											
IGG0CLBW						X	X	X	X																X											
IGG0CLBX						X	X	X	X	X															X											
IGG0CLBY						X	X	X	X																X											
IGG0CLB8						X	X	X	X	X							X				X				X											
IGG0CLCA						X	X	X	X																X											
IGG0CLCB						X	X	X	X																X											
IGG0CLCD						X	X	X	X																X											
IGG0CLCG	X					X	X	X	X								X				X				X	X					X					
IGG0CLCL						X	X	X	X								X								X											
IGG0CLCO	X					X	X	X	X	X		X		X		X	X					X			X			X						X		
IGG0CLCP						X	X	X	X																X											
IGG0CLCR	X					X	X	X	X	X							X				X				X											
IGG0CLCS						X	X	X	X								X				X				X											
IGG0CLCX						X	X	X	X					X							X	X			X			X						X		
IGG0CLCY						X	X	X	X																X											
IGG0CLC9			X			X	X	X	X					X			X				X	X			X	X	X	X			X			X	X	
IGG0CLEG	X					X	X	X	X												X				X											X
IGG0CLES						X	X	X	X																X											
IGG0CLET						X	X	X	X	X															X											
IGG0CLEX						X	X	X	X																X											
IGG0CLEZ						X	X	X	X	X															X											
IGG0CLFA																																				
IGG0CLFB					X	X	X	X																	X											
IGG0CLFC																																				
IGG0CLFD					X	X	X	X	X								X																			
IGG0CLFE						X	X	X	X								X								X											
IGG0CLFF						X	X	X	X								X								X											
IGG0CLFH					X	X	X	X																	X											
IGG0CLFQ						X	X	X	X								X								X											
IKQALL00																	X																			
IKQPOP00																	X																X			
IKQRDS00																	X															X				
IKQREN00																	X															X				
IKQSCR00																	X															X				
IKQVTC00																	X										X			X						
IKQWDS00																	X															X				
$$BCLCRA														X								X						X	X				X	X	X	X
$$BODADE																			X																	
$$BODADS																																				

Figure 6.4 Macro-to-module relationships for catalog and DADSM components (part 4 of 4)

Module \ Macro	ASPL	ASSIGN	ASYSCOM	AVRLIST	CANCEL	CATLG	CCB	CDLOAD	COMRG	CVTOC	DCTENTRY	DEQB	ENQB	EOJ	EXCP	EXTENT	EXTRACT
IIPAMT00							X										
IIPBMR00					X		X								X		
IIPCLS00							X	X									
IIPOPN00							X	X									
IIPPRCMR					X		X					X	X		X		
IIPPRCPR							X	X									
IKQASNMT	X	X		X							X						
IKQBRP								X									
IKQCLCAT						X		X									
IKQCLEAN							X		X					X	X		
IKQCLIF								X									
IKQCLO00								X									
IKQCLOCL																	
IKQCLOVY																	
IKQCLRDD						X		X									
IKQDRP								X									
IKQDUMP							X	X	X						X		
IKQDUMPC									X								
IKQEDX						X		X	X								
IKQEOV					X			X	X								
IKQGEN																	
IKQJIBSM																X	
IKQLAB								X	X								X

Module \ Macro	FREEVIS	GENCB	GENDTL	GET	GETFLD	GETVCE	GETVIS	IGGCAXWA	IGGCCA	IGGMCMWA	IGGMDRWA	IIPAMDTF	IIPDTF	IIPRAT	IJBLBRC	IJJHCPL	IJJHDLST
IIPAMT00												X					
IIPBMR00																	
IIPCLS00												X	X	X			
IIPOPN00		X					X					X	X	X			
IIPPRCMR												X					
IIPPRCPR				X								X	X	X			
IKQASNMT	X		X			X	X										
IKQBRP	X						X										
IKQCLCAT	X		X				X	X									
IKQCLEAN	X						X										
IKQCLIF	X						X										
IKQCLO00	X		X														
IKQCLOCL																	
IKQCLOVY																	
IKQCLRDD	X						X										
IKQDRP	X																
IKQDUMP																	
IKQDUMPC								X	X	X	X						
IKQEDX	X						X										
IKQEOV	X						X										
IKQGEN	X						X										
IKQJIBSM	X						X										
IKQLAB	X						X								X		

Module \ Macro	IJJHFMT1	IJJHFMT3	IJJHFMT4	IKQACB	IKQAMBL	IKQAMCBS	IKQAMDSB	IKQARDB	IKQAREX	IKQARGH	IKQASMT	IKQAUHDR	IKQBHD	IKQBKPHD	IKQBLARD	IKQBSPH	IKQBUFE
IIPAMT00																	
IIPBMR00																	
IIPCLS00				X	X		X										
IIPOPN00				X	X		X										
IIPPRCMR																	
IIPPRCPR																	
IKQASNMT											X						
IKQBRP													X	X		X	X
IKQCLCAT				X	X		X	X					X				
IKQCLEAN	X		X														
IKQCLIF				X	X												
IKQCLO00				X	X		X	X						X			X
IKQCLOCL				X	X								X				
IKQCLOVY				X	X		X						X				X
IKQCLRDD				X	X		X						X				
IKQDRP														X			
IKQDUMP				X	X		X	X					X	X			X
IKQDUMPC				X		X											
IKQEDX					X		X	X							X		
IKQEOV					X		X	X									
IKQGEN				X	X		X		X	X		X					
IKQJIBSM					X		X	X									
IKQLAB																	

Figure 6.5 Macro-to-module relationships for IIP, control block manipulation, and open/close and service aids (part 1 of 6)

Macro / Module	IKQCBMTB	IKQCB1	IKQCB2	IKQCCB	IKQCCBCW	IKQCGETC	IKQCIW	IKQCLCOR	IKQCLRLS	IKQCLWA	IKQCOMRG	IKQCTGFL	IKQCTGFV	IKQCTGPL	IKQDTL	IKQEDB	IKQEDBLD	IKQEXLST	IKQFCDB	IKQFNDLB	IKQIOEQU	IKQJRNDS	IKQKWTB	IKQLPMB	IKQLUB	IKQMDADS	IKQMSGPL
IIPAMTT00																											
IIPBMR00																											
IIPCLS00		X	X																								
IIPOPN00		X	X																								
IIPPRCMR																											
IIPPRCPR		X	X																								
IKQASNMT															X												
IKQBRP																			X								
IKQCLCAT										X	X	X		X	X	X											
IKQCLEAN					X						X															X	
IKQCLIF											X																
IKQCLO00									X	X	X				X	X								X			X
IKQCLOCL										X	X																
IKQCLOVY										X	X																
IKQCLRDD										X	X	X		X													
IKQDRP																						X					
IKQDUMP							X									X					X			X			
IKQDUMPC											X	X	X	X												X	
IKOEDX						X		X			X	X		X		X	X			X				X	X		
IKQEOV											X					X				X					X		
IKQGEN	X																	X									
IKQJIBSM																X											
IKQLAB											X									X					X		

Figure 6.5 **Macro-to-module relationships for IIP, control block manipulation, and open/close and service aids (part 2 of 6)**

Module \ Macro	IKQLOCK	IKQMTPRM	IKQOAL	IKQOCFSP	IKQOCGSP	IKQOCPRC	IKQOPCLR	IKQOPCLW	IKQOPLCT	IKQOPNWA	IKQOPASMT	IKQPLH	IKQRPHD	IKQRPL	IKQRSCB	IKQSHRW	IKQSMBPL	IKQTHB	IKQUSB	IKQUNLK	IKQVLST	IKQVOL1	IKQVRGN	IKQVRPPL	IKQVSMDP	IKQVSRT
IIPAMT00																										
IIPBMR00																										
IIPCLS00																										
IIPOPN00																										
IIPPRCMR																										
IIPPRCPR																										
IKQASNMT	X	X					X				X									X						
IKQBRP												X	X		X									X		X
IKQCLCAT	X			X	X	X	X	X				X		X						X						
IKQCLEAN																						X				
IKQCLIF							X	X																		
IKQCLO00	X		X				X	X				X		X				X		X			X		X	X
IKQCLOCL							X	X				X		X					X							
IKQCLOVY				X	X	X	X	X				X		X					X							
IKQCLRDD				X	X	X	X	X				X		X			X									
IKQDRP												X														X
IKQDUMP								X				X		X				X	X		X					
IKQDUMPC														X									X			
IKQEDX							X	X													X					
IKQEOV							X	X													X					
IKQGEN												X		X												
IKQJIBSM																										
IKQLAB																										

Module \ Macro	LABEL	LOAD	LOCK	LPLDCT	MAPBDY	MAPCOMR	MAPDTL	MAPPIB	MAPSSID	MODCB	MODFLD	MODVCE	OVTOC	POINT	POST	PUT	PVTOC	SHOWCB	SUBSID	SYSCOM	UNLOCK	WAIT
IIPAMT00																						
IIPBMR00																						X
IIPCLS00																		X				
IIPOPN00		X																X				
IIPPRCMR																						X
IIPPRCPR										X				X		X		X				
IKQASNMT		X	X				X					X									X	
IKQBRP																						
IKQCLCAT			X				X														X	
IKQCLEAN																						X
IKQCLIF																						
IKQCLO00			X				X				X				X						X	X
IKQCLOCL																						
IKQCLOVY																						
IKQCLRDD																						
IKQDRP															X							
IKQDUMP						X									X							X
IKQDUMPC																						
IKQEDX																						
IKQEOV											X											
IKQGEN															X							X
IKQJIBSM																						
IKQLAB	X	X		X																		

Figure 6.5 **Macro-to-module relationships for IIP, control block manipulation, and open/close and service aids (part 3 of 6)**

Module \ Macro	ASPL	ASSIGN	ASYSCOM	AVRLIST	CANCEL	CATLG	CCB	CDLOAD	COMRG	CVTOC	DCTENTRY	DEQB	ENQB	EOJ	EXCP	EXTENT	EXTRACT	FREEVIS	GENCB	GENDTL	GET	GETFLD	GETVCE	GETVIS	IGGCAXWA	IGGCCA	IGGMCMWA	IGGMDRWA	IIPAMDTF	IIPDTF	IIPRAT	IJBLBRC	IJJHCPL	IJJHDLST	IJJHFMT1	IJJHFMT3	IJJHFMT4	IKQACB	IKQAMBL	IKQAMCBS	IKQAMDSB	IKQARDB	IKQAREX	IKQARGH	IKQASMT	IKQAUHDR	IKQBHD	IKQBKPHD	IKQBLARD	IKQBSPH	IKQBUFE
IKQMTMSG							X		X						X																																				
IKQNEX						X		X	X									X						X														X	X		X	X							X		
IKQOCMSG						X	X	X							X																							X													
IKQOCSHR																				X					X													X													
IKQOPN						X		X										X		X				X	X			X				X						X	X	X	X						X	X			X
IKQOPNAB																																X						X	X		X						X				X
IKQOPNAI						X		X																								X						X	X												
IKQOPNCT								X	X	X					X			X						X	X			X				X	X	X	X		X	X	X		X										
IKQOPNDO						X		X										X		X					X							X						X	X		X							X			X
IKQOPNDS						X		X										X						X								X						X			X										
IKQOPNHC						X		X										X		X				X	X							X						X			X										
IKQOPNNC																				X				X								X						X	X		X						X	X			X
IKQOPNOV					X	X		X										X							X							X						X	X		X	X									
IKQOPNPV																								X														X				X									
IKQOPNRD						X		X										X						X								X						X			X										
IKQOPNRP								X												X				X														X	X		X									X	
IKQOPNUC																																X						X	X												
IKQOPNUS						X		X										X		X				X	X													X	X												
IKQRBA						X		X										X						X															X		X	X									
IKQSTM																		X						X														X								X					
IKQTMSD																																						X	X		X										
IKQTMSF																																						X	X		X										
IKQVEDA			X				X		X					X	X																																				
$$BACLOS							X	X	X						X		X					X																X	X												
$$BCLCRA									X																																										
$$BCVSAM								X	X													X																X	X												
$$BCVS02																		X				X																X													
$$BCVS03									X									X						X																											
$$BCVS04															X																																				
$$BOVSAM							X	X	X	X					X		X	X				X		X										X				X													
$$BOVS01									X	X												X		X										X			X	X													
$$BTCLOS								X	X													X																X	X												

Figure 6.5 **Macro-to-module relationships for IIP, control block manipulation, and open/close and service aids (part 4 of 6)**

Module \ Macro	IKQCBMTB	IKQCB1	IKQCB2	IKQCCB	IKQCCBCW	IKQCGETC	IKQCIW	IKQCLCOR	IKQCLRLS	IKQCLWA	IKQCOMRG	IKQCTGFL	IKQCTGFV	IKQCTGPL	IKQDTL	IKQEDB	IKQEDBLD	IKQEXLST	IKQFCDB	IKQFNDLB	IKQFTMAP	IKQIOEQU	IKQJRNDS	IKQKWTB	IKQLPMB	IKQLUB	IKQMDADS	IKQMSGPL
IKQMTMSG											X																	
IKQNEX						X		X			X	X		X		X	X			X					X	X		
IKQOCMSG											X	X		X														X
IKQOCSHR															X													
IKQOPN										X	X			X	X	X		X	X		X				X	X		X
IKQOPNAB											X					X									X			
IKQOPNAI											X	X		X														
IKQOPNCT				X							X					X				X						X		
IKQOPNDO									X		X	X		X	X	X										X		
IKQOPNDS											X	X		X														
IKQOPNHC											X	X		X														X
IKQOPNNC											X				X				X									X
IKQOPNOV											X	X		X		X	X			X					X	X		
IKQOPNPV											X			X														
IKQOPNRD											X	X		X														
IKQOPNRP											X				X													
IKQOPNUC											X			X														
IKQOPNUS											X	X		X	X													
IKQRBA												X		X		X												
IKQSTM																												
IKQTMSD																		X						X				
IKQTMSF																		X						X				
IKQVEDA																												
$$BACLOS																												
$$BCLCRA																												
$$BCVSAM											X																	
$$BCVS02																												
$$BCVS03																												
$$BCVS04																												
$$BOVSAM																												
$$BOVS01																												
$$BTCLOS											X																	

Figure 6.5 Macro-to-module relationships for IIP, control block manipulation, and open/close and service aids (part 5 of 6)

Module \ Macro	IKQLOCK	IKQMTPRM	IKQOAL	IKQOCFSP	IKQOCGSP	IKQOCPRC	IKQOPCLR	IKQOPCLW	IKQOPLCT	IKQOPNWA	IKQOPASMT	IKQPLH	IKQRPHD	IKQRPL	IKQRSCB	IKQSHRW	IKQSMBPL	IKQTHB	IKQUSB	IKQUNLK	IKQVLST	IKQVOL1	IKQVRGN	IKQVRPPL
IKQMTMSG		X		X	X	X																		
IKQNEX							X	X													X			
IKQOCMSG							X	X																
IKQOCSHR	X			X	X	X														X				
IKQOPN	X		X	X	X	X	X	X	X	X		X			X	X	X	X	X	X	X		X	
IKQOPNAB				X	X	X	X	X	X	X		X					X							
IKQOPNAI				X	X	X	X	X	X	X														
IKQOPNCT				X	X	X	X																	
IKQOPNDO	X			X	X	X	X	X	X	X		X							X	X				
IKQOPNDS				X	X	X	X	X	X	X														
IKQOPNHC	X			X	X	X	X	X	X	X							X			X				
IKQOPNNC				X	X	X	X	X		X		X		X		X		X	X					
IKQOPNOV				X	X	X	X	X	X	X											X			
IKQOPNPV				X	X	X	X	X	X	X														
IKQOPNRD				X	X	X	X	X	X	X														
IKQOPNRP				X	X	X	X	X		X		X						X	X					
IKQOPNUC				X	X	X	X	X	X	X														
IKQOPNUS				X	X	X	X	X	X	X									X					
IKQRBA							X																	
IKQSTM																								
IKQTMSD													X	X										
IKQTMSF													X	X										
IKQVEDA																								
$$BACLOS			X					X				X												
$$BCLCRA	X																			X				
$$BCVSAM								X																
$$BCVS02								X									X							
$$BCVS03																								
$$BCVS04																								
$$BOVSAM								X		X														
$$BOVS01								X																
$$BTCLOS								X																

Module \ Macro	IKQVSMDP	IKQVSRT	LABEL	LOAD	LOCK	LPLDCT	MAPBDY	MAPCOMR	MAPDTL	MAPPIB	MAPSSID	MODCB	MODFLD	MODVCE	OVTOC	POINT	POST	PUT	PVTOC	SHOWCB	SUBSID	SYSCOM	UNLOCK	WAIT
IKQMTMSG								X																X
IKQNEX																								
IKQOCMSG																								
IKQOCSHR					X				X														X	
IKQOPN	X				X				X				X										X	
IKQOPNAB											X										X			
IKQOPNAI																								
IKQOPNCT				X											X				X					X
IKQOPNDO	X	X			X				X								X						X	X
IKQOPNDS																								
IKQOPNHC				X	X				X														X	
IKQOPNNC									X															
IKQOPNOV													X											
IKQOPNPV																								
IKQOPNRD																								
IKQOPNRP		X							X								X							X
IKQOPNUC																								
IKQOPNUS									X															
IKQRBA																								
IKQSTM																	X							X
IKQTMSD																								
IKQTMSF																								
IKQVEDA								X														X		X
$$BACLOS							X	X					X										X	X
$$BCLCRA					X			X														X	X	
$$BCVSAM																								
$$BCVS02																								
$$BCVS03																								
$$BCVS04																								X
$$BOVSAM				X			X	X							X									X
$$BOVS01				X				X							X				X					
$$BTCLOS																								

Figure 6.5 Macro-to-module relationships for for IIP, control block manipulation, and open/close and service aids (part 6 of 6)

Catalog Communication Area Register Save Area

A catalog communication area (CCA) is built for every call to catalog management. The CCA contains a register save area (CCAREGS) that allows the PSR (programming systems representative) to follow the flow of control from one catalog management procedure to another, through each procedure called to process the request.

The contents of the first three words in the CCA field named CCAREGS are as follows:

- the first word contains the contents of register 13, which is a pointer to the user's save area, and
- the second and third words are unused.

If a catalog management procedure is entered from another catalog management procedure, the fourth and subsequent words are used as three-word catalog save areas defined as follows:

- the first word contains the contents of register 12, which is the calling procedure's base address,
- the second word contains the contents of register 13, which is a pointer to the previous 12-byte entry in the register save area (CCAREGS), and
- the third word contains the contents of register 14, which is the return address (in the calling procedure).

Immediately after registers 12, 13, and 14 are saved (at register 13 + 12), register 12 is updated to contain the *called* procedure's base address. Register 13's value is increased by 12, so that it points to the latest entry in CCAREGS. While a catalog management procedure is processing, register 11 contains a pointer to the beginning of the CCA.

Note that backward movement is not recorded in the trace table. For example, if procedure B returns to procedure A, the return is not shown in the register save area.

Catalog Management Error Code-to-Module Relationship

The error codes issued by VSAM catalog modules are set in the CCACD1 field of the CCA and transferred to register 15 on exit from catalog managment. The reason codes are set in the CCAREASN field of the CCA. They are passed back to the user via AMS or Record Management.

Figure 6.6 contains the definitions of the error codes. To each error code belong one or several reason codes. Along with the reason code the module which detected the error is also shown. It is identified by the last two letters of its name. The PL/S names are those names to which the catalog management error codes are equated. To find the assembler instructions which set the error code, first reference the PL/S statement, then relate the PL/S statement to the appropriate assembler language statement.

PL/S Name for Code	Error Code		Reason Codes	Module Name	Description
	Dec	Hex			
RCCAT	4	4	2	AC, AD, AE, AH	Error when opening a catalog.
			4	AE, CS	Error when closing a catalog.
			8	AH	Internal error - an ACB was supplied to catalog management but its ID was not X'A0'. An IDUMP was issued. Can also be caused by a problem program overlaying storage it does not own.
			10	AH	The user catalog entry in the master catalog cannot be found. The user has either made a spelling error, has exported the user catalog, or has more than one master catalog and has executed IPL using the wrong one.
RCENT	8	8	2	AF	During catalog DELETE, the cluster record for the catalog cannot be found at its normal location (3rd self-describing record). An IDUMP was issued.
			4	EG	The catalog I/O routine read a record from the catalog and found it to be a free record. Since this never occurs during normal processing, an internal error is returned. An IDUMP was issued.
			6	CG, AN, BG, BN	A request to read a record resulted in a no record found error from VSAM.
			8	AL, AT, BD, CA, CG	A request to place a record by key into a catalog resulted in a duplicate key error from VSAM.
			12	CB	An internal error has occurred indicating that a record thought to be on the buffer chain is not present. An IDUMP was issued.
RCNOTCYL	16	10	0	ET, FF	The Access Mehtod Services CYLINDERS parameter was used, but the extents found on the corresponding DLBL and EXTENT statements do not start or end on a cylinder boundary. This error can only occur during DEFINE CATALOG, DEFINE SPACE, or DEFINE UNIQUE.
RCINSP	20	14		CG	Insufficent direct access space available to extend the catalog or CRA. (Reason codes are from Record Management - see Volume 2.) To correct this situation, a DEFINE SPACE is needed to provide more space for suballocation.
RCIOL	24	18	2	AG, AZ, CG	I/O error during a LOCATE.
			4	C9	I/O error during a CATALOG VERIFY operation.
RCIONL	28	1C	2	AG	I/O error but not during LOCATE processing.
			4	AG, CG, EG	I/O error during EXCP during catalog open (non-build case).
RCINCPL	32	20	2	BC	Internal error indicating that catalog management was unable to return the requested data in the catalog parameter list (CPL) for update extend. An IDUMP was issued.
RCDSNF	36	24	2	BD	Incorrect record type read.
			4	BN	Data set not found in the VTOC.
RCVLSZ	40	28	0	AL, BG, AZ, CX	Internal workarea provided was too small.
RCVLSM	44	2C	2	AF	It has been detected during DELETE CATALOG that the CTGWKA work area is too small. The user (Access Method Services) has to provide a larger area.

Figure 6.6 **Catalog Management error code-to-module relationship (Part 1 of 10)**

PL/S Name for Code	Error Code		Reason Codes	Module Name	Description
	Dec	Hex			
RCINFUNC	48	30	2	AB	An invalid CPL has been passed to the catalog management driver. An IDUMP was issued.
			4	ET	During DEFINE, an incorrect master catalog ACB was found. An IDUMP was issued.
			6	BD	Alter of nonVSAM data sets or NOCIFORMAT with ERASE is not allowed.
			8	BD	Alter of catalog name is not allowed.
			20	BL, CL	Forced delete space is not allowed on catalog volume.
			34	A7	ERASE was specified for DELETE of a NO-CI format ESDS.
RCIOU	52	34		A7, BD, BN, A6, AF, AQ, CL, FD	I/O error during VTOC processing. (Reason codes from DADSM - see Figure 6.6.)
RCSEC	56	38	2	BM	All attempts to provide a password via the system operator are used without a successful compare.
			6	BM	No prompt allowed and password not provided, or incorrect password provided via the catalog parameter list.
			8	BM	USVR requested stop.
			12	A7, AQ	Security violation from DADSM scratch.
RCINENT	60	36	0	BG	An attempt was made to delete a catalog without specifying an entry type.
			4	BG, BE	Invalid entry type 'C' or nonVSAM.
			6	BD	Invalid entry type 'C' for alter of attributes.
			8	BD	Invalid 'C' or 'I' entry type for alter buffer size.
			10	BD	'C' record invalid for alter of AMDSB.
			12	BE	No alter volume allowed on 'C'-type records.
			14	BE	Test CPL error during ADD volume. An IDUMP was issued.
			16	CA, CD	AIX G record association is not 'D'|'I'|'C' (AIX is not a KSDS).
			18	CA, CD	Upgrade set Y record association is not D|I.
			20	CA, CD	'Y' association in base cluster data record does not point to a 'Y' record.
			22	CA, CD	'D' association in 'C' record does not point to a 'D' record.
			24	BD	Upgrade or update for a record which is not a 'G' or 'R'.
			26	BD	Alter of exception exit but the record is not 'D' or 'I'.
			28	BD	Alter of average record size but the record is not 'D' or 'I'.
			30	BD	Alter of expiration date but the record is not 'C,' 'G,' 'R,' 'M,' or 'U.'
			32	FH	DEFINE, DELETE, ALTER of nonVSAM entry prohibited in recoverable catalog.
			34	BE	ALTER REMOVE VOLUMES is not allowed for a noallocate data set if the volume contains a CRA, or if the resulting volume list is null.
			36	BD	Alter of share options but the data set is open.

Figure 6.6 **Catalog Management error code-to-module relationship (Part 2 of 10)**

PL/S Name for Code	Error Code		Reason Codes	Module Name	Description
	Dec	Hex			
RCNAME	64	40	2	BL	Test field name not present in group space header.
			4	BE	Association names do not exist.
			6	EZ	Error on retrieval of fixed block device characteristics for 'D' or 'I' component during LOCATE of an F-field name. LOCATE issued a 0 return code, but failed to return request information. A system error.
RCNOSP	68	44	2	BB	Cannot extend a unique file.
			4	BB	No secondary extent value specified.
			6	BB	Not enough class-0 space available for suballocation (extend function).
			12	BB	More than 16 extents/volume for reusable file.
			16	AQ	On a data space allocation for a DEFINE SPACE command, rounding of extent(s) specified in the EXTENT statement(s) resulted in no data space being allocated.
			16	BX	On a DEFINE with the UNIQUE attribute, rounding of the first extent specified in the JCL resulted in no data space being allocated.
			18	AP	On a data space allocation for a DEFINE catalog, rounding of the space specified in the first (or only) EXTENT statement resulted in no space being allocated. The first extent must be large enough to contain the catalog.
			24	BB	Not enough space of the required class (nonzero) is available on any eligible volume to suballocate space for an existing file (extend function).
RCVOLFUL	70	46	2	FD	On a DEFINE SPACE or DEFINE MASTER CATALOG using the DEDICATE parameter, the volume was found to be full.
			4	FD	On a DEFINE SPACE or DEFINE CATALOG using the ORIGIN parameter, the volume was found to have space allocated within the limits of the requested area.
RCNMNTD	72	48	2	FB	All mounted volumes for the default model have been tried for suballocation and failed.
			4	A6, A7, BL, CG	No symbolic unit available, automatic assignment failed.
			8	A6, A7, BL, CG, CO, AH	Volume not mounted; invalid or unassigned symbolic unit.
			12	A6, A7, BL, CG	Volume not mounted, job canceled.
			16	A6, A7, BL, CG, CO, AH	The operator replied "NEWPAC" but the volume was not mounted.
			20	A6, A7, AH, BL, CG, CO	Automatic unassignment unsuccessful.
			24	A6, A7, AH, BL, CG, CO, ET, FD, FQ	During dynamic assignment, the lock table is full.
			28	A6, A7, AH, BL, CG, CO, ET, FD, FQ	Insufficient storage available for dynamic assignment or unassignment.

Figure 6.6 **Catalog Management error code-to-module relationship (Part 3 of 10)**

PL/S Name for Code	Error Code		Reason Codes	Module Name	Description
	Dec	Hex			
RCMAXEXT	74	4A	2	FE	For a DEFINE SPACE or DEFINE CATALOG with the DEDICATE parameter, the maximum of 16 extents was reached before the volume was completely dedicated.
RCRELOP	80	50	0	CA	Related object is reusable.
			2	CA	Related object is an RRDS.
			4	CA	Related object does not exist.
			6	CA, CP	AIX/PATH not allowed to be built over a catalog.
			8	CA	Name of AIX/PATH and related object are identical.
			10	CA, CP	No pointer to a related object of an AIX/PATH.
			12	CA, CP	AIX not being built over a base cluster, or related path object not a cluster or an AIX.
			14	CA, CP	AIX/PATH not allowed to be built over a SAM ESDS.
RCDATE	84	54	0	AF, BG	Expiration date not reached.
RCCRAOP	88	58	0	CO	Open of CRA failed.
			2	-	I/O error in CRA.
			4	CO	Internal call to CO for CRA open has conflicting parameters. An IDUMP was issued.
			6	A6	Space of less than one cylinder for CRA is not allowed.
RCDSEXT	92	5C	0	BB	Attempt was made to exceed the maximum number of extents.
RCSPANCK	96	60	0	BX, CA	Prime key as specified for spanned record is not completely contained in the first control interval of the record.
			4	CY	Maximum logical record size for spanned records exceeds CA size.
			6	CA	AIX key for spanned records is not completely contained within the first control interval of the base cluster record.
			8	CA	One of the following: The AIX key is not completely contained within the base cluster record, or the maximum record size for defining an alternate index is too small.
RCRECVOL	100	64	0	A6	Attempt was made to define a unique file on a volume owned by a recoverable catalog. You must define space before any unique files can be defined.
RCCATEX	104	68	0	AL	The master catalog is already open, but a request has been made for master catalog DEFINE.
RCINFNAM	108	6C	0	AY	Invalid field name. An IDUMP was issued.
RCINFPL	112	70	2	AY	Invalid group code in FPL. An IDUMP was issued.
			6	BT	A fixed block catalog dictionary field name has been used with UPDATE or MODIFY. Fixed block field names are valid only with LOCATE and only with non-test FPLs. An IDUMP was issued.
RCCATBAL	116	74	–		Catalog out of balance (not used by VSE).
RCSYSFLD	120	78	0	AX	Non-existent field being modified. An IDUMP was issued.

Figure 6.6 **Catalog Management error code-to-module relationship (Part 4 of 10)**

PL/S Name for Code	Error Code		Reason Codes	Module Name	Description
	Dec	Hex			
RCINCI	124	7C	2	AG, CG	Invalid RBA return code from record management.
			4	AG, EG	Catalog build open processing, but specified CI was greater than 9.
			6	AG, EG	CCR record ('L') read by error.
RCBLKVCK	128	80	-		Validity check on user block (not used by DOS/VSE).
RCINPTR	132	84	0	AL, FC	No pointer to VOLSER list and the appropriate default model was not found.
			2	BH	No pointer to VOLSER list.
			4	AL	No FPL to AMDSB of data.
			6	AL	No FVT from cluster level.
			8	AL	No pointer in FPL to data set attribute.
			10	AL	No FPL for volume space parameters.
			12	AL	No pointer to expiration date value.
			14	AL	No pointer to creation date in FPL.
			16	BH	No pointer to device type in FPL.
			18	BH	No FPL in FVT.
			20	AL	No pointer to the workarea.
			22	AL	No pointer to password data of related object.
			24	AL	No pointer to owner ID in FPL.
			26	EX	No pointer to cluster space parameter in FPL.
			28	EX	No pointer to data space parameter in FPL.
			30	EX	No pointer to index space parameter in FPL.
			32	AN	No buffersize FPL in data FVT.
			34	AN	No buffersize FPL in cluster FVT.
			36	AN	No buffersize FPL in index FVT.
			38	BX	No logical record size FPL in cluster or data FVT.
			40	BH	No pointer to data set file sequence number in volume list FPL.

Figure 6.6 **Catalog Management error code-to-module relationship (Part 5 of 10)**

PL/S Name for Code	Error Code		Reason Codes	Module Name	Description
	Dec	Hex			
RCMISPAR	136	88	2	AL	No length for volume serial list area.
			4	AL	Missing DNAME parameter with DEFINE UNIQUE FILE.
			6	AL	Missing cluster entry name.
			8	AL	Missing space parameter in space FVT.
			10	AL	Missing VOLSER list pointer in space FVT.
			12	AL	Missing DNAME pointer in space FVT.
			14	AL	No length in volume list from cluster FVT.
			16	EX	No space parameter on 'C' or 'D' FVT.
			18	BX	Average logical record size missing.
			20	AN	No key specified.
			24	BH	No entries in volume list.
			26	BH	No entries in device type list.
			28	CA	AIX name missing.
			30	CP	Path entry name missing.
RCINCNPM	140	8C	2	AL	Index FVT found for RRDS or ESDS. An IDUMP was issued.
			4	AL	Keyrange is invalid.
			6	AL	Keyranges found on both data and cluster FVT.
			8	AL	Workarea too small.
			10	EX	Space parameters found on both cluster and data FVT.
			12	AN	Buffersize specified more than once.
			14	BX	Average logical record size specified on index FVT.
			16	BX	Average logical record not valid for DEFINE CATALOG.
			18	BX	Average logical record size specified on cluster and data FVT.
			20	AN	Inconsistent keylength specified in 'D' and 'I' FVT.
			22	AP	Inconsistent VOLSER lists with different name in each list; or
				AQ	DLBL and volume list do not match.
			24	AP, ET, FE	Primary allocation for data space less than required for the catalog.
			26	AL	Invalid space request type for catalog DEFINE.
			28	BH	Unequal number of VOLSER and file sequence numbers in list entries.
			30	BH	More device type entries than VOLSERs.
			32	AN	Invalid key position specified on 'D' or 'I'.
			34	BX	Invalid space request type on DEFINE. An IDUMP was issued.

Figure 6.6 **Catalog Management error code-to-module relationship (Part 6 of 10)**

PL/S Name for Code	Error Code Dec	Hex	Reason Codes	Module Name	Description
			36	AJ, ET	The number of keyranges and the number of VOLSERs are not equal, or for a unique data component, the number of keyranges is greater than the number of VOLSERs.
			38	AL	Unique attribute is not allowed for catalog DEFINE.
			42	AL	RRDS has spanned attribute.
			44	AL	RRDS has a maximum record length that is not equal to the average record length.
			46	BD	An attempt was made to specify an exception exit for a data set which was originally defined with a DOS/VS Release earlier than Release 31.
			48	ET, EX, FB	Access Method Services issued a DEFINE with tracks or cylinders as the allocation unit for a fixed block device. If Generate Volume List was used, the unit indicated CKD device but there were no CKD devices on the Default model.
			50	ET, EX, FB	Access Method Services issued a DEFINE with blocks as the allocation unit for a CKD device. If Generate Volume List was used, the unit indicated FBA device but there were no FBA devices on the Default Model.
			52	FA	An error was returned from the EXTRACT macro. An IDUMP was issued.
RCINENTN	144	90	2	AL, CA	AIX/PATH name is invalid; first character must be alphabetic. An IDUMP was issued.
			4	AL	Unique name is invalid because it was Z999999, which is restricted.
			6	AL	Data and index names not allowed for a catalog. An IDUMP was issued.
			8	FA	A file-ID, greater than 27 characters was specified for a partition independent or processor independent file.
RCVOLOWN	148	94	0	A6	Volume already owned by another catalog.
RCDNECAT	152	98	0	AF	Non-empty catalog cannot be deleted.
RCNOSPSA	156	9C	0	AU	Not enough class-0 space available within the catalog for suballocation of a newly-defined file.
			2	CS	On a DEFINE catalog command, a class-0 data space was specified that was not large enough to allow suballocation of the catalog recovery area (CRA). There was sufficient space for the catalog.
			24	AU	There is insufficient space of the required class (nonzero) in the specified volume(s) to satisfy a request for suballocation of a newly-defined cluster or alternate index.
			26	CS	On a DEFINE catalog command, a nonzero class data space was specified that was not large enough to allow suballocation of the catalog recovery area (CRA). There was sufficent space for the catalog.
RCVNDSPD	160	A0	0	BL, BN	Volume record not deleted from the catalog because some file still owns space on the volume. All empty data spaces on this volume have been deleted.
RCINSSWA	164	A4	2	AB, AH, AQ, AT, BL, CB, CG, CO, CY, C9, EX, FA, FB, FC, FD, FQ	Catalog management GETVIS error: A continuous area of main storage of the requested size is not available.
			8	CG	Record management GETVIS error: As above, but the call came from record management.

Figure 6.6 **Catalog Management error code-to-module relationship (Part 7 of 10)**

PL/S Name for Code	Error Code		Reason Codes	Module Name	Description
	Dec	Hex			
RCINVDTY	168	A8	2	BX, AP	Device type not supported.
			4	BH	Invalid device name.
			8	BH	DEFINE NONVSAM has been specified for a fixed block device. Fixed block devices are not supported for NONVSAM.
RCDUPNVL	172	AC	4	BD	Duplicate name in VTOC.
RCNSPVTC	176	B0	0	AQ	No space available in the VTOC.
RCDSO	184	B8	2	CX	The catalog is in use by some partition. Until the number of users is zero, the catalog cannot be deleted.
			4	CX	The data or index component (or both) is (are) in use, and the file cannot be deleted.
RCCATUNA	188	BC	2	CG	No RPL available for processing, request ignored. An IDUMP was issued.
			4	AJ	Suballocate failure.
			6	AB	No RPL available for processing, request ignored.
RCMLRSZ	192	C0	0	BX	Maximum logical record size exceeds allowable maximum.
RCMCISZD	196	C4	0	BX	CI size too large for allowable maximum for data component.
RCMCISZI	200	C8	0	BX	CI size too large for index.
RCKEYINC	204	CC	0	BX	Key extends beyond the end of the maximum logical record.
RCBUFSIZ	208	D0	0	EX	Bufferspace too small for minimum number of CIs.
RCSIZCAL	212	D4	2	EX	Not enough parameters specified in DEFINE. An IDUMP was issued.
			4	CY	Specified and default values result in only one CI per CA for a KSDS or alternate index.
			6	CY	For a non-unique KSDS or alternate index, the index CI size is too small, and an attempt to reduce the number of data CIs failed.
			8	CY	For a unique KSDS or alternate index, the index CI size is too small. The number of data CIs cannot be reduced because their size is fixed at one cylinder.
			10	CY	Buffer space too small for a non-unique data set.
			12	CY	Buffer space too small for a unique data set.
			14	CY	The specified or default values result in less than one CI per CA for an ESDS or RRDS.
			16	AN	For a fixed block SAM ESDS, the SAM logical blocksize is not a multiple of the SAM logical record size.
RCEXTOVL	216	D8	2	AQ	Overlap on: a. VTOC b. Expired secure file c. Expired file d. Unexpired secure file e. Unexpired file f. New extents
RCINEXT	220	DC	-		Invalid EXTENT card.

Figure 6.6 **Catalog Management error code-to-module relationship (Part 8 of 10)**

PL/S Name for Code	Error Code		Reason Codes	Module Name	Description
	Dec	Hex			
RCMXGRP	224	E0	0	AW	Maximum number of group occurrence pointers have been processed.
			2	AW	More than 125 AIXs in the upgrade set.
RCLOCKER	228	E4	0	A7, AL, BL	Time of day clock hardware error.
RCRCDLDF	232	E8	16	BM, BH, CX, CG	CDLOAD error reason is: a. from CDLOAD b. 0 if no reason is available.
RCINDER	240	F0	4	AQ, AF, AH, AP, AE, A7, BX, BG, B8, BL, BN, CO, CS, FD	IKQVLAB error (see Figure 6.7).
			6	AQ, A7, BL, B8, ET	No extents found.
			8	A8, BL, ET, FE	Invalid device type, invalid extents specified, or IKQVLAB error during device type processing.
			10	AQ, ET	Too many extents, or duplicate volume serial numbers.
			12	ET	Number of tracks required by the Access Method Services command exceeds DLBL total per volume.
			22	AH	Catalog name in DLBL and CPL do not match.
			24	AH	Master catalog DLBL not found.
			26	AH	User or job catalog DLBL not found, or job catalog required but not found.
			28	AH	DLBL catalog name is missing.
			30	AH	Volume serial number in extent statement for job catalog does not match volume serial in the entry for this job catalog in master catalog.
			32	CO	Automatic assign for CRA failed due to no programmer logical unit.
			34	CO	Catalog recovery area OPEN failure because volume was not mounted.
			40	AH	Automatic assign for user catalog failed due to no programmer logical units.
			42	AH	User catalog OPEN failed because the catalog volume was not mounted.
RCEFRMPH	242	F2		A7	Physical I/O error occurred while trying to erase the data set being deleted. (Reason codes are from VSAM record management - see Volume 2.)
RCEF	244	F4	2	A7	This ACB could not be opened (OPEN failed while trying to erase). The address of the catalog ACB or the CI number may be wrong. (Reason codes from OPEN - see Figure 6.7.)

Figure 6.6 **Catalog Management error code-to-module relationship (Part 9 of 10)**

PL/S Name for Code	Error Code		Reason Codes	Module Name	Description
	Dec	Hex			
RCENQ	246	F6	0	AB, AH, C9, CO, AC, CX	Successful request.
			4	AB, AH, C9, CO, AC, CX	Resource is not available.
			8	AB, AH, C9, CO, AC, CX	Lock table space exhausted.
			12	AB, AH, C9, CO, AC, CX	Inconsistent request.
			16	AB, AH, C9, CO, AC, CX	A deadlock would have occurred.
			20	AB, AH, C9, CO, AC, CX	DTL format error.
			24	AB, AH, C9, CO, AC, CX	Resource is already exclusively owned.
			28	AB, AH, C9, CO, AC, CX	Lock file space exhausted.
			32	AB, AH, C9, CO, AC, CX	VOLID PARM used but the volume is not online.
			36	AB, AH, C9, CO, AC, CX	Irrecoverable error on the SYSLCK file.
RCVOLENT	248	F8	0	AR, AT, BX, BN, AK	Volume entry does not exist in this catalog.
RCEFRM	250	FA		A7	VSAM record management has found an error during ERASE. (Reason codes from VSAM record management - see Volume 2.)
RCEE	252	FC	0	FH	Early exit. (Internal indicator; if found in CCACD1, it does *not* indicate an error, but that the last O/C/EOV request for catalog open has completed.)

Figure 6.6 Catalog Management error code-to-module relationship (Part 10 of 10)

DADSM Error Code-to-Module Relationship

Module	Routine	Reason Code (Reg.15)	Dec.	Meaning
IKQWDS00*	-	X'00'	00	Write completed successfully
		X'40'	64	GETVIS failed
IKQRDS00	-	X'00'	00	Successful read
		X'40'	64	GETVIS failed
IKQREN00*	-	X'00'	00	Rename successful
		X'08'	08	Volume not mounted
		X'40'	64	GETVIS failed
IKQSCR00*	-	X'00'	00	Scratch successful
		X'08'	08	Volume not mounted
		X'40'	64	GETVIS failed
IKQALL00*	-	X'00'	00	Successful allocate
		X'08'	08	Volume not mounted
		X'28'	40	No extents given in EXTENT card
		X'40'	64	GETVIS failed
		X'48'	72	CDLOAD failed
IKQVTC00	-	X'00'	00	VTOC opened successfully
		X'40'	64	GETVIS failed
		X'68'	104	LOAD failed
IKQPOP00*	-	X'00'	00	Successful VTOC label build
		X'28'	40	No extents given
		X'40'	64	GETVIS failed
*Note: These modules also return codes issued by other internally called modules, including the common VTOC handler phase (IJJHCVH).				
IJJHCVH	-	X'04'	04	I/O error reading VOL label
		X'08'	08	Volume not mounted
		X'0C'	12	VTOC I/O error
		X'10'	16	A duplicate name was found during a write to the VTOC.
		X'14'	20	VTOC full
		X'1C'	28	Extents overlap unexpired file
		X'20'	32	Extents overlap protected, unexpired file
		X'24'	36	Extents overlap VTOC
		X'28'	44	A format-1, or the next label was not found on a VTOC read request.
		X'30'	48	Invalid label address
		X'38'	56	The specified extents overlap those of a protected expired file.
		X'40'	64	GETVIS failed
		X'44'	68	Security violation.
		X'4C'	76	Invalid VTOC share option.
		X'50'	80	Extents overlap each other
		X'54'	84	The caller supplied work area is too small.
		X'58'	88	Format-4 VTOC label not found
		X'5C'	92	VOL 1 label not found.
		X'60'	96	Extent block (JIB) processing failed
		X'6C'	108	Labels read are not format-1s or format-3s.
		X'70'	112	A lock request for the VTOC is inconsistent with a previous lock request.
		X'74'	116	A VTOC lock request would result in a deadlock condition.
		X'78'	120	An invalid control block was built by the common VTOC handler.
		X'7C'	124	An I/O error occured on the system lock file.

Figure 6.7 **DADSM error code-to-module relationships**

Control Block Manipulation Error Code-to-Module Relationship

Module	Return Code (Reg. 15)	Meaning	Error Code (Reg. 0)	Meaning
IKQGEN	X'00'	Successful completion	-	-
	X'04'	An error has been detected	X'01'	Invalid function type code
			X'02'	Invalid control block type-code
			X'03'	Invalid keyword type-code
			X'08'	Not enough storage available
			X'09'	User area too small
			X'0A'	Exit address is not specified in the input
			X'0E'	Inconsistent parameters
			X'0F'	The user area is not on a fullword boundary
	X'08'	Invalid use of the execute form of this macro instruction.	-	Since the return code is set by the macro expansion and not by IKQGEN, R0 contents do not indicate an error code.
	X'0C'	CDLOAD failure	-	The return code is set by the macro expansion, not by IKQGEN, and R0 contains the return code from CDLOAD.
IKQTMSD or IKQTMSF	X'00'	Successful completion	-	-
	X'04'	An error has been detected	X'01'	Invalid function type-code.
			X'02'	Invalid control block type code.
			X'03'	Invalid keyword type-code
			X'04'	Control block not of type specified
			X'05'	The ACB is closed; it must be open
			X'06'	The cluster is not key-sequenced (does not include an index)
			X'07'	The EXLST entry is not present
			X'09'	User area is too small (SHOWCB)
			X'0A'	Exit address is not specified in the input
			X'0B'	The RPL is active
			X'0C'	The ACB is open; it must be closed
			X'0D'	The exit address is not in the control block
			X'0E'	Inconsistent parameters
			X'0F'	The user area is not on a fullword boundary
			X'14'	The parameter 'STRMAX' was specified, but LSR is not active.
	X'08'	Same as for IKQGEN	-	Same as for IKQGEN
	X'0C'	Same as for IKQGEN	-	Same as for IKQGEN

Figure 6.8 **Control block manipulation error code-to-module relationships (Record Management)**

OPEN Error Code-to-Module Relationship

Error Code (ACBERFLG)	Return Code[1] (Reg.15)	Issuing Module	Meaning
X'02'	X'08'	IKQOPN	Invalid ACB ID
X'04'	X'08'	IKQOPN	This ACB is already open
X'0E'	X'08'	IKQOPN IKQOPNCT	The symbolic unit in the DLBL statement is invalid.
X'0F'	X'08'	IKQJIBSM	Extent block processing failed.
X'11'	X'08'	IKQLAB IKQOPN IKQOPNOV	The address in the ASSIGN statement for the logical unit was IGN (assignment ignored).
X'12'[2]	X'08'	IKQLAB IKQOPN IKQOPNOV	The address in the ASSIGN statement for the logical unit was UA (logical unit unassigned).
X'13'	X'08'	IKQOPNOV	Unable to automatically assign a logical unit number.
X'20'	X'08'	IKQOPN	The open disposition specified for the file conflicts with the other file characteristics.
X'22'[2]	X'08'	IKQOPNOV	The volume serial numbers in the EXTENT statement do not match those in the catalog entry.
X'28'	X'08'	IKQOPNDS	No space available on any volume for primary allocation of a dynamic file.
X'30'	X'08'	IKQOPNHC	An attempt was made to open a NOALLOCATION file which is not reusable.
X'32'	X'08'	$$BOVSAM IKQLAB IKQOPN IKQOPNHC IKQOPNOV IKQOPNRD	One or more VSAM processing modules cannot be loaded because the user's partition is too small.
X'40'	X'08'	IKQOPNHC	An attempt was made to open a NOCIFORMAT file using VSAM (ACB) access.
X'41'	X'08'	IKQOPNHC	An attempt was made to open a SAM ESDS without the VSE/VSAM Space Management for SAM Feature installed.
X'42'	X'08'	IKQOPNVC	An attempt was made to open a DTF whose file characteristics do not match the file characteristics of the VSAM catalog.
X'43'	X'08'	IKQOPNVC	An attempt was made to open an unexpired file for output using a DTF.
X'44'	X'08'	IKQOPN	The file to be opened has a name which begins with a restricted prefix.
X'45'	X'08'	IKQOPNVC	An attempt was made to open a nonSAM ESDS file using a DTF.
X'46'	X'08'	IKQOCIMR	An invalid file-id was detected during implicit define or implicit delete.
X'47'	X'08'	IKQOCIMR	Allocation specifications for implicit define conflict with the file characteristics specified in the DTF.
X'48'	X'08'	IKQOCIMR	The file-id specified in your DLBL statement was not found in the catalog and insufficient allocation information is specified for an implicit define.
X'4E'	X'08'	IKQOCIMR	A catalog management error was detected during implicit delete.
X'4F'	X'08'	IKQOCIMR	A catalog management error was detected during implicit define.

Figure 6.9 **OPEN error code-to-module relationships (part 1 of 4)**

Error Code (ACBERFLG)	Return Code[1] (Reg.15)	Issuing Module	Meaning
X'50'	X'08'	IKQOPNOV	An attempt was made to assign more than one volume of a multivolume data set to one unit when direct or keyed processing was specified in the ACB.
X'5C'	X'04'	IKQOPN	LSR was specified in the ACB macro, but no message area was specified. Ignore this error code if there is no ACB with the DFR option in the resource pool.
X'60'	X'04'	IKQOPNHC	A data set which is unusable (due to a failure in recovery) was opened for input.
X'64'	X'04'	IKQOPN	Open encountered an empty alternate index in the upgrade set.
X'68'	X'04'	IKQOPNOV	The time stamp of the data set's volume does not match the system time stamp in the volume catalog entry.
X'6C'	X'04'	IKQOPNAB IKQOPNHC	The system time stamps in the catalog entries for the data and index components of a data set do not match. This indicates that the data has been updated separately. **Note:** No warning is given if the index time stamp is greater than the data time stamp.
X'6E'	X'08'	IKQOPN	An attempt was made to open an empty data set for input only; or to open a data set which was not closed properly after initial loading.
X'74'	X'04'	IKQOPNHC	The data set was not closed the last time it was processed.
X'75'	X'08'	IKQLAB IKQOPNOV IKQOPNRD	The symbolic unit specified in the EXTENT statement is not a valid device type, or invalid extents are specified.
X'80'	X'08'	IKQLAB	The DLBL statement is missing or the filename in the DLBL does not match the ACB.
X'84'	X'08'	IKQLAB	A permanent I/O error occurred while VSAM was reading the label information area.
X'88'	X'08'	$$BOVSAM IKQOPN IKQOPNAB IKQOPNAI IKQOPNDS IKQOPNHC IKQOPNNC IKQOPNOV IKQOPNPV IKQOPNRD IKQOPNRP IKQOPNUS IKQLAB IKQSTM	VSAM could not obtain a continuous area of virtual storage of the size required for work areas, control blocks, and buffers needed by VSAM.

Figure 6.9 **OPEN error code-to-module relationships (part 2 of 4)**

Error Code (ACBERFLG)	Return Code[1] (Reg.15)	Issuing Module	Meaning
X'90'	X'08'	IKQOPNHC IKQOPNAI IKQOPNRD IKQOPNUS	A permanent I/O error occurred while VSAM was reading or writing a catalog entry.
X'94'	X'08'	IKQOPN IKQOPNHC IKQOPNOV IKQOPNAI	No valid entry was found in the catalog for this ACB or for the alternate index structure related to this ACB.
X'98'	X'08'	IKQOPNHC	Security verification failed: the password specified in the ACB, or supplied by the operator, for a specific level of access does not match the corresponding password in the catalog entry.
X'A0'	X'08'	IKQOPNAB IKQOPNAI IKQOPN IKQOPNHC	The operands specified in the ACB are inconsistent with each other or with information in the catalog entry, such as keyed processing specified for an ESDS or DFR specified when LSR is inactive.
X'A1'	X'08'	IKQOPN	User buffers were specified with keyed access (they may be specified only with CNV access) or with LSR.
X'A5'	X'08'	IKQOPN IKQOPNHC IKQOCSHR	A permanent I/O error was detected on the system lock file.
X'A6'	X'08'	IKQOPN IKQOPNHC IKQOCSHR	The system lock table is not large enough to accommodate the concurrent requests.
X'A7'	X'08'	IKQOPN IKQOPNHC IKQOCSHR	The system lock file is not large enough to accommodate the concurrent requests.
X'A8'	X'08'	IKQOPNAB IKQOPNHC IKQOCSHR	The data set is not available because it is being updated, loaded, or reset by (and under exclusive control of) another ACB or because it has been flagged *read only* by Access Method Services.
X'B4'	X'08'	IKQOPN IKQOPNCT IKQOPNHC IKQOPNAI	An error occurred while a catalog was being opened. Possibly an attempt to get additional storage failed, or an I/O error occurred while VSAM was reading the VTOC, a VTOC label, or a cluster.

Figure 6.9 OPEN error code-to-module relationships (part 3 of 4)

Error Code (ACBERFLG)	Return Code[1] (Reg.15)	Issuing Module	Meaning
X'BC'	X'08'	IKQOPN	ACB already active.
X'C0'	X'08'	IKQOPNHC	A data set which is unusable (because of a failure in recovery) was opened for output.
X'C4'	X'08'	IKQOPNAB	Access to data was requested via an empty alternate index.
X'D4'	X'08'	IKQOPNAB	LSR is specified, but the data set being opened is empty (which implies that it is to be loaded).
X'D8'	X'08'	IKQOPNRP	LSR is specified, but the keylength of the data set being opened is greater than the maximum keylength specified for the resource pool.
X'DC'	X'08'	IKQOPNRP	LSR is specified, but the CI size of the data set being opened is greater than the largest buffer size specified for the resource pool.
X'E4'	X'08'	IKQOPNRP	LSR is specified, but there is no resource pool defined; may also be caused by problems while loading the resource table.
X'E8'	X'08'	IKQOPNRD	Reset was specified for a non-reusable data set and the data set is not empty.
X'F8'	X'08'	IKQLAB	IKQLAB internal interface error with LABEL or EXTRACT macro.
X'FE'	X'08'	IKQOPN IKQOPNHC IKQOCSHR	Open detected an unexpected return code from the lock manager.
X'FF'	X'08'	IKQOPNAI IKQOPNDS IKQOPNHC IKQOPNRD IKQOPNUS	Unexpected return from catalog locate function.

[1] The contents of register 15 have the following meaning:

X'00' Open was successfully completed.
X'04' All ACBs were opened successfully, but one or more ACBs had a warning message.
X'08' One or more ACBs were not successfully opened.

Note that register 15 contains the worst return code encountered while opening a list of ACBs. This means that some of the ACBs in the list may have been opened successfully, even though register 15 contains X'04' or X'08'.

[2] This code is ignored by OPEN, but is meaningful to catalog and DADSM routines.

Figure 6.9 **OPEN error code-to-module relationships (part 4 of 4)**

CLOSE and TCLOSE Error Code-to-Module Relationship

Error Code (ACBERFLG)	Return Code (Reg.15)	Issuing Module	Meaning
X'02'	X'04'	IKQCLO00	Invalid control block ID or ACB address not in OAL.
X'04'	X'04'	IKQCLO00 $$BCVSAM $$BTCLOS	ACB is already closed or invalid control block structure.
X'4C'	X'04'	IKQCLCAT IKQCLRDD	Disposition processing failed during close.
X'88'	X'08'	IKQCLO00 IKQCLCAT	VSAM could not obtain a continous area of virtual storage large enough for the work area.
X'90'	X'08'	IKQCLCAT	A permanent I/O error occurred while VSAM was reading or writing a catalog entry.
X'A5'	X'04'	IKCLO00 IKQCLCAT	A permanent I/O error was detected on the system lock file.
X'A6'	X'04'	IKQCLO00 IKQCLCAT	The system lock table is not large enough to accommodate the concurrent requests.
X'A7'	X'04'	IKQCLO00 IKQCLCAT	The system lock file is not large enough to accommodate the concurrent requests.
X'B8'	X'04'	IKQCLO00	An internal error occurred while VSAM was completing outstanding I/O requests.
X'BC'	X'04'	IKQCLO00	The ACB is busy; it is being used, for example, by SHOWCB or TESTCB.
X'E4'	X'08'	IKQCLO00	Resource pool is invalid.
X'FC'	X'04'	IKQSMACL	Automatic close of the DTF for a managed-SAM file failed.
X'FE'	X'04'	IKQCLO00	Close detected an unexpected return code from the lock manager.

Figure 6.10 CLOSE and TCLOSE error code-to-module relationships

SHOWCAT Error Code-to-Module Relationship

Return Code (Reg. 15)	Issuing Module	Meaning
X'00'	IKQSCAT	VSAM successfully completed the request
X'04'	IKQSCAT	The specified work area is less than the minimum size (64 bytes) or too small to display all associated objects (in this case, as many objects as possible are displayed).
X'08'	IKQSCAT	Either the ACB address is invalid, or the VSAM master catalog does not exist or could not be opened.
X'0C'	IKQSCAT	CDLOAD failure while VSAM routines were being loaded
X'14'	IKQSCAT	The specified object or control interval does not exist.
X'18'	IKQSCAT	An I/O error occurred while accessing the catalog.
X'1C'	IKQSCAT	The specified CI number is invalid.
X'20'	IKQSCAT	The specified object is not a cluster, data set, index, alternate index, path, or upgrade set.
X'24'	IKQSCAT	The information in the catalog is at a different level than that in the CRA.
X'28'	IKQSCAT	The SHOWCAT module received an unexpected error code from catalog management.
X'2C'	IKQSCAT	Error while searching the label area for the data set name.
X'30'	IKQSCAT	EXTOPT field name is not valid for SHOWCAT.
X'34'	IKQSCAT	EXTOPT specified, but record type not D or I.

Note: In the case of return codes 0C, 14, 18, 1C, 24, 28, or 34, the SHOWCAT work area contains the return code and reason code issued by catalog management and the ID of the module in which the error was detected. In the case of return code X'2C', the work area contains the return code from IKQLAB, as shown in Figure 6.7. The format of the work area is then:

Offset	Length	Description
0	2	Length of work area
2	2	Catalog management or IKQLAB return code
4	2	Catalog management reason code
6	2	Catalog management module ID

Figure 6.11 SHOWCAT error code-to-module relationships

BLDVRP and DLVRP Error Code-to-Module Relationship

Macro	Return Code (Reg. 15)	Issuing Module	Description
BLDVRP	X'00'	IKQBRP	Request was successfully completed
	X'04'	IKQBRP	A resource pool already exists for the partition
	X'0C'	IKQBRP	CDLOAD failure
	X'14'	IKQBRP	STRNO is specified as less than one or more than 255
	X'18'	IKQBRP	The specified BUFFERS parameter is invalid
DELVRP	X'00'	IKQDRP	Request was successfully completed
	X'04'	IKQDRP	There is no resource pool to be deleted
	X'08'	IKQDRP	There is at least one open data set still using the resource pool
	X'0C'	IKQDRP	CDLOAD failure

Figure 6.12 Error codes for BLDVRP and DLVRP macros

Catalog Management Procedure Cross Reference

The following table lists, for each catalog management procedure, the names of the other procedure(s) that call it.

To Procedure: The called procedure.

To Module: The module to which the called procedure belongs. Only the last two characters of the module name are given; the first six characters are always IGG0CL.

From Procedure: The calling procedure.

From Module: The module to which the calling procedure belongs. Only the last two characters of the module name are given; the first six characters are always IGG0CL.

Sequence: The number indicates that this is the nth call that the calling procedure makes.

For example, procedure IGGPALEC (in module IGG0CLBE) can be called by either IGGPALSA (in module IGG0CLBE) or IGGPALVR (in module IGG0CLBN). When IGGPALSA calls IGGPALEC, it is IGGPALSA's 4th call to another procedure. When IGGPALVR calls IGGPALEC, it is IGGPALVR's 8th call to another procedure.

To Procedure	To Module	From Procedure	From Module
IGGPACDV	AB	IGG0CLC9	C9
IGGPADCI	AG	IGGPAOCI	AG
		IGGPAXCI	AG
		IGGPISCI	AG
IGGPADGO	AW	IGGPXDGO	BT
IGGPAPGO	AW	IGGPADGO	AW
IGGPAIX	CA	IGGPCDVR	AT
IGGPALAE	BE	IGGPALVA	BE
IGGPALBC	BE	IGGPALIX	BE
		IGGPALSA	BE
		IGGPALVA	BE
		IGGPALVL	BE
IGGPALBT	BD	IGGPALMD	BD
IGGPALEC	BE	IGGPALSA	BE
		IGGPALVR	BN
IGGPALF1	BD	IGGPALNM	BD
IGGPALGV	BD	IGGPALF1	BD
IGGPALIX	BE	IGGPALEC	BE
IGGPALMD	BD	IGGPALT	BD
IGGPALNM	BD	IGGPALT	BD
IGGPALPL	BN	IGGPALVE	BN
		IGGPALVR	BN
IGGPALSA	BE	IGGPALVA	BE
IGGPALSV	BN	IGGPALF1	BD
IGGPALT	BD	IGGPCDVR	AT
IGGPALT2	AX	IGGPXLT2	BT
IGGPALVA	BE	IGGPALIX	BE
		IGGPALVL	BE
IGGPALVE	BN	IGGPALVR	BN
IGGPALVL	BE	IGGPALT	BD
IGGPALVO	BN	IGGPALVR	BN
IGGPALVR	BN	IGGPALIX	BE
		IGGPALVL	BE
IGGPALY	CD	IGGPALT	BD
IGGPANCI	AG	IGGPAOCI	AG
		IGGPAXCI	AG
		IGGPISCI	AG
IGGPAOCI	AG	IGGPBAWP	CP
		IGGPCSMP	BT
		IGGPDAIN	BH
		IGGPDCCE	AN
		IGGPDCPC	AP
		IGGPNUPG	CD
IGGPASPP	CA	IGGPAIX	CA
IGGPASPT	AW	IGGPADGO	AW
		IGGPAGOP	AW
IGGPAXCI	AG	IGGPADGO	AW
		IGGPAGOP	AW
		IGGPDEFS	AQ
		IGGPDOIT	CG
		IGGPDOWN	BW
		IGGPMBGO	AX
		IGGPMVAR	AX
		IGGPRISE	BW
		IGGPSNK2	BW
IGGPBAMO	CP	IGGPPATH	CP
IGGPBASP	CP	IGGPPATH	CP
IGGPBAWP	CP	IGGPPATH	CP
IGGPBCAN	B8	IGGPDFBO	B8
IGGPBCRA	CR	IGGPCADR	CR
IGGPBDLT	FB	IGGPGVL	FB
		IGGPUPLT	FB
IGGPBDSH	B8	IGGPDFBO	B8
IGGPBDSS	B8	IGGPDFBO	B8
IGGPBMR	BR	IGGPEDS	AU
		IGGPMEBM	AE
		IGGPRELE	CB
		IGGPSALS	AU
		IGGPSSCR	BF
		IGGPXDGO	BT

To Procedure	To Module	From Procedure	From Module
		IGGPXEL2	BT
		IGGPXVAL	BS
IGGPBNUN	B8	IGGPDFBO	B8
IGGPBOUT	CA	IGGPDUND	AN
IGGPBSCR	B8	IGGPDFBO	B8
IGGPBSGT	BM	IGGPCKAU	BM
		IGGPCKCC	BM
		IGGPCLGT	BM
		IGGPPWVR	BM
IGGPCADR	CR	IGGPCCLN	AT
IGGPCCCR	EG	IGGPADCI	AG
		IGGPAOCI	AG
		IGGPAXCI	AG
		IGGPDEFC	AS
		IGGPDELC	AF
		IGGPDUND	AN
		IGGPISCI	AG
		IGGPRCU	C9
IGGPCCLN	AT	IGGPCDVR	AT
IGGPCDSD	A6	IGGPDFS2	A6
IGGPCDVR	AT	IGGPACDV	AB
IGGPCEXT	BB	IGGPSSWD	BB
		IGGPUPDE	BB
IGGPCGO	AX	IGGPMGO	AX
IGGPCHAC	CG	IGGPDOIT	CG
		IGGPFRE	AG
		IGGPUTE	AG
IGGPCHSP	FB	IGGPGVL	FB
IGGPCIND	AP	IGGPDCSP	AP
IGGPCKAU	BM	IGGPACDV	AB
		IGGPCDVR	AT
		IGGPCRAP	CO
		IGGPDEFS	AQ
		IGGPLSTC	BQ
		IGGPPRPW	CA
IGGPCKCC	BM	IGGPCKAU	BM
		IGGPPWVR	BM
IGGPCKEN	CA	IGGPDSEM	CA
		IGGPPASE	CP
IGGPCKEX	BM	IGGPCKAU	BM
		IGGPCKCC	BM
		IGGPCLGT	BM
		IGGPPWGT	BM
		IGGPPWVR	BM
IGGPCKLC	BA	IGGPLVAL	BA
IGGPCLBF	CB	IGGPDBDI	AJ
		IGGPDCMB	AK
		IGGPDUND	AN
IGGPCLGT	BM	IGGPPWVR	BM
IGGPCNBO	B8	IGGPDSPO	AJ
IGGPCNST	CP	IGGPBAWP	CP
IGGPCOBT	A6	IGGPDEFS	AQ
IGGPCRAO	CO	IGGPOPEN	CG
		IGGPPRDS	CS
IGGPCRAP	CO	IGGPACDV	AB
IGGPCRBO	CR	IGGPBCRA	CR
		IGGPCRVL	CS
		IGGPDEFS	AQ
IGGPCRDI	CR	IGGPBCRA	CR
IGGPCRTC	A6	IGGPDEFS	AQ
IGGPCRVL	CS	IGGPBCRA	CR
IGGPCSAL	BB	IGGPUPDE	BB
IGGPCSDG	A6	IGGPDFS2	A6
IGGPCSHG	A6	IGGPDFS2	A6
IGGPCSMP	BT	IGGPXDGO	BT
IGGPDAIN	BH	IGGPDEFA	BH
IGGPDAIX	CX	IGGPDLCL	BG
IGGPDALR	BX	IGGPAIX	CA
		IGGPDSCB	AN
IGGPDANL	BH	IGGPDAIN	BH

Figure 6.13 **Catalog management procedure cross reference (part 1 of 6)**

To Procedure	To Module	From Procedure	From Module
IGGPDATA	BY	IGGPDDSA	BY
		IGGPDRSP	BY
IGGPDAVO	BH	IGGPDEFA	BH
IGGPDBCV	AK	IGGPDBVO	AK
IGGPDBDI	AJ	IGGPAIX	CA
		IGGPDSCB	AN
IGGPDBSF	AN	IGGPAIX	CA
		IGGPDSCB	AN
IGGPDBVO	AK	IGGPDCMB	AK
IGGPDCAV	AL	IGGPAIX	CA
		IGGPDEF	AL
IGGPDCBE	AS	IGGPDEFC	AS
IGGPDCBO	AE	IGGPBCRA	CR
		IGGPDCOC	AE
IGGPDCCB	AE	IGGPDCOC	AE
IGGPDCCC	BX	IGGPDSPC	BX
IGGPDCCE	AN	IGGPDCPC	AP
		IGGPDRDA	AN
IGGPDCCI	CY	IGGPDCCO	CY
IGGPDCCO	CY	IGGPDSPC	BX
IGGPDCDA	AP	IGGPDSCB	AN
IGGPDCDE	AL	IGGPBASP	CP
		IGGPDDEP	AL
IGGPDCEB	AS	IGGPDCBE	AS
IGGPDCFL	ES	IGGPDCVO	ES
IGGPDCHD	ES	IGGPDCVO	ES
IGGPDCHK	ET	IGGPCDVR	AT
IGGPDCIM	BX	IGGPDCCC	BX
IGGPDCIS	BY	IGGPDRSP	BY
IGGPDCIX	ES	IGGPDCVO	ES
IGGPDCLD	ES	IGGPDCVO	ES
IGGPDCMB	AK	IGGPDBDI	AJ
IGGPDCME	AS	IGGPDCEB	AS
IGGPDCNV	AJ	IGGPDBDI	AJ
IGGPDCOC	AE	IGGPDEFC	AS
IGGPDCON	AP	IGGPDCSP	AP
IGGPDCPB	AS	IGGPDCBE	AS
IGGPDCPC	AP	IGGPDCSP	AP
IGGPDCPR	AE	IGGPDCOC	AE
IGGPDCPT	BX	IGGPDCCC	BX
IGGPDCRC	AS	IPPCRVL	CS
		IGGPDCHD	ES
		IGGPDCIX	ES
		IGGPDCLD	ES
		IGGPDCME	AS
IGGPDCSF	AL	IGGPDCAV	AL
IGGPDCSP	AP	IGGPDCVS	AP
IGGPDCSS	AS	IGGPDEFC	AS
IGGPDCVO	ES	IGGPDEFC	AS
IGGPDCVS	AP	IGGPDCDA	AP
IGGPDCWC	AL	IGGPDCAV	AL
IGGPDDCE	BX	IGGPDSPC	BX
IGGPDDEP	AL	IGGPAIX	CA
		IGGPDEF	AL
		IGGPGVSA	FB
IGGPDDNP	AL	IGGPDCAV	AL
IGGPDDRT	BY	IGGPDDSA	BY
		IGGPDRSP	BY
IGGPDDSA	BY	IGGPDRSP	BY
IGGPDDTC	BY	IGGPDDSA	BY
		IGGPDRSP	BY
IGGPDEDD	A7	IGGPVMSC	A7
IGGPDEDE	AL	IGGPDDEP	AL
IGGPDEF	AL	IGGPCDVR	AT
IGGPDEFA	BH	IGGPCDVR	AT
IGGPDEFC	AS	IGGPDSCB	AN
IGGPDEFS	AQ	IGGPASGN	FQ
		IGGPCDVR	AT
		IGGPCONV	FQ
		IGGPDCSP	AS
		IGGPDSPO	AJ
IGGPDEIN	BW	IGGPMVAR	AX
IGGPDEL	BG	IGGPCDVR	AT
IGGPDELA	BG	IGGPDEL	BG
IGGPDELC	AF	IGGPCDVR	AT
		IGGPDELU	CX
IGGPDELN	EX	IGGPDELC	AF
IGGPDELO	CX	IGGPDAIX	CX
		IGGPDELX	CX
		IGGPDLCL	BG
		IGGPDLSF	CL
IGGPDELP	CX	IGGPDEL	BG
IGGPDELS	BL	IGGPCDVR	AT
IGGPDELU	CX	IGGPDLDS	8G
		IGGPDELE	AF
		IGGPDLSF	FQ
IGGPDELX	CX	IGGPDEL	BG
IGGPDELY	CX	IGGPDLCL	BG
IGGPDEL2	AV	IGGPXEL2	BT
IGGPDEMV	A7	IGGPDELA	BG
		IGGPVMSC	A7
IGGPDESH	A7	IGGPVMSC	A7
IGGPDEUN	AN	IGGPDUND	AN
IGGPDEVG	A7	IGGPVMSC	A7
IGGPDEXA	BG	IGGPDAIX	CX
		IGGPDELP	CX
		IGGPDELX	CX
		IGGPDLCL	BG
		IGGPDLTC	BG
IGGPDEXB	CX	IGGPDAIX	CX
		IGGPDELP	CX
		IGGPDELX	CX
		IGGPDELY	CX
		IGGPDPTH	CX
IGGPDEXC	CX	IGGPDAIX	CX
		IGGPDPTH	CX
IGGPDEXD	AJ	IGGPDBCV	AK
		IGGPDBVO	AK
		IGGPDCMB	AK
		IGGPDSEX	AJ
		IGGPDSSP	AK
IGGPDEXP	BG	IGGPDELX	CX
		IGGPDLCL	BG
IGGPDEXT	FE	IGGPLBLS	FQ
		IGGPSPC	FQ
IGGPDFBI	B8	IGGPCNBO	B8
		IGGPDFBO	B8
IGGPDFBO	B8	IGGPDBDI	AJ
IGGPDFRE	A8	IGGPDFRS	A8
IGGPDFRS	A8	IGGPDBDI	AJ
IGGPDFSC	AL	IGGPDCAV	AL
IGGPDFS2	A6	IGGPDEFS	AQ
IGGPDGO	AX	IGGPDEL2	AV
		IGGPMGO	AX
IGGPDGOP	AX	IGGPDEL2	AV
		IGGPMGO	AX
IGGPDISA	BY	IGGPDRSP	BY
IGGPDJCL	BG	IGGPDLDS	BG
IGGPDLCD	BL	IGGPDELS	BL
IGGPDLCL	BG	IGGPDEL	BG
IGGPDLCR	BL	IGGPDELS	BL
IGGPDLDS	BG	IGGPDAIX	CX
		IGGPDELX	CX
		IGGPDELU	CX
		IGGPDLCL	BG
IGGPDLER	AF	IGGPDELC	AF
IGGPDLET	BL	IGGPDELS	BL
IGGPDLEX	BL	IGGPDELS	BL
IGGPDLSC	BL	IGGPDELS	BL
IGGPDLSF	CL	IGGPDELS	BL

Figure 6.13 Catalog management procedure cross reference (part 2 of 6)

To Procedure	To Module	From Procedure	From Module
		IGGPDELU	CX
IGGPDLSH	BL	IGGPDELS	BL
IGGPDLTC	BG	IGGPDEL	BG
IGGPDLTS	BL	IGGPDELS	BL
IGGPDLVC	BL	IGGPDELS	BL
IGGPDLVM	BL	IGGPDELS	BL
IGGPDLXT	BG	IGGPDAIX	CX
		IGGPDEL	BG
		IGGPDELA	BG
		IGGPDELO	CX
		IGGPDELP	CX
		IGGPDELX	CX
		IGGPDELY	CX
		IGGPDEXP	BG
		IGGDJCL	BG
		IGGPDLCL	BG
		IGGPDLDS	BG
		IGGPDPTH	CX
IGGPDMDS	CL	IGGPDLSF	CL
IGGPDMOP	CY	IGGPDCMB	AK
IGGPDOIT	CG	IGGPRIO	CG
IGGPDORG	FF	IGGPSPC	FQ
IGGPDOWN	BW	IGGPSNK2	BW
IGGPDPBI	CY	IGGPDCCC	BX
		IGGPDCCO	CY
IGGPDPTH	CX	IGGPDAIX	CX
		IGGPDELX	CX
		IGGPDLCL	BG
IGGPDRCA	AK	IGGPDBVO	AK
IGGPDRCS	AJ	IGGPDBDI	AJ
IGGPDRDA	AN	IGGPAIX	CA
		IGGPDSCB	AN
IGGPDRNG	AK	IGGPDBVO	AK
IGGPDRPG	AL	IGGPDCAV	AL
IGGPDRSP	BY	IGGPDRDA	AN
IGGPDSCB	AN	IGGPDEF	AL
		IGGPGVL	FB
IGGPDSEM	CA	IGGPAIX	CA
IGGPDSEX	AJ	IGGPDSPO	AJ
IGGPDSMD	A6	IGGPCDSD	A6
		IGGPCSDG	A6
		IGGPCSHG	A6
IGGPDSPC	BX	IGGPDRDA	AN
IGGPDSPF	EX	IGGPAIX	CA
		IGGPDSCB	AN
IGGPDSPO	AJ	IGGPDBDI	AJ
IGGPDSSP	AK	IGGPDBCV	AK
		IGGPDBVO	AK
IGGPDSTS	A7	IGGPVMSC	A7
IGGPDSTY	AL	IGGPBASP	CP
		IGGPDDEP	AL
IGGPDSXT	AQ	IGGPDEFS	AQ
IGGPDTIM	AL	IGGPDEDE	AL
		IGGPDEFS	AQ
IGGPDUND	AN	IGGPAIX	CA
		IGGPBAMO	CP
		IGGPDCBO	AE
		IGGPDEFA	BH
		IGGPDELA	BG
		IGGPDFBO	B8
		IGGPDLDS	BG
		IGGPDSCB	AN
IGGPDUSC	A7	IGGPDELA	BG
		IGGPVMSC	A7
IGGPDVMV	A7	IGGPDELA	BG
		IGGPVMSC	A7
IGGPDVSC	CL	IGGPDELS	BL
		IGGPDLSF	CL
IGGPDVV	FD	IGGPDEXT	FE
		IGGPDORG	FF

To Procedure	To Module	From Procedure	From Module
		IGGPRLAB	FD
IGGPDWAI	AL	IGGPBASP	CP
		IGGPDDEP	AL
IGGPDXDS	CL	IGGPDLSF	CL
IGGPEDS	AU	IGGPSALS	AU
IGGPEMIO	AF	IGGPIORA	CG
IGGPEMSG	AF	IGGPIORA	CG
IGGPERAS	A7	IGGPDLDS	BG
IGGPEVOL	FB	IGGPCHSP	FB
IGGPEXO	CG	IGGPDOIT	CG
		IGGPOPEN	CG
		IGGPRIO	CG
IGGPEXPD	AX	IGGPDOWN	BW
		IGGPMVAR	AX
		IGGPSINK	BW
IGGPEXT	AZ	IGGPALGV	BD
		IGGPALIX	BE
		IGGPALVA	BE
		IGGPALVL	BE
		IGGPALVR	BN
		IGGPALY	CD
		IGGPCKEX	BM
		IGGPDAIX	CX
		IGGPDEDD	A7
		IGGPDELC	AF
		IGGPDELP	CX
		IGGPDELS	BL
		IGGPDELX	CX
		IGGPDELY	CX
		IGGPDEMV	A7
		IGGPDESH	A7
		IGGPDEVG	A7
		IGGPDEXP	BG
		IGGPDLCD	BL
		IGGPDLCL	BG
		IGGPDLSF	CL
		IGGPDLTC	BG
		IGGPDPTH	CX
		IGGPDSEM	CA
		IGGPGVLM	FC
		IGGPINIT	BC
		IGGPLSTC	BQ
		IGGPPGAO	CP
		IGGPPUPG	CD
		IGGPRELE	CB
		IGGPSALL	AR
		IGGPSALS	AU
		IGGPSVOL	BC
		IGGPUPG	CD
IGGPFCHK	EX	IGGPDCSP	AP
IGGPFCON	EX	IGGPDCCC	BX
IGGPFGRW	CA	IGGPAIX	CA
IGGPFLIM	ET	IGGPDDCE	BX
		IGGPDEFS	AQ
		IGGPVCHK	ET
IGGPFRE	AG	IGGPANCI	AG
IGGPFREE	CO	IGGPCRAO	CO
IGGPFVXT	FC	IGGPAIX	CA
		IGGPBDLT	FB
		IGGPEVOL	FB
		IGGPGVL	FB
		IGGPGVLG	FB
		IGGPGVLM	FC
		IGGPUPLT	FB
IGGPF4PR	A6	IGGPDEFS	AQ
IGGPF4RD	BU	IGGPCRBO	CR
		IGGPCRVL	CS
		IGGPDCBO	AE
		IGGPDELC	AF
		IGGPDLET	BL

Figure 6.13 Catalog management procedure cross reference (part 3 of 6)

To Procedure	To Module	From Procedure	From Module
		IGGPDLTS	BL
		IGGPDSTS	A7
		IGGPF4PR	A6
IGGPF4WR	BU	IGGPCRBO	CR
		IGGPCRVL	CS
		IGGPDCBO	AE
		IGGPDELC	AF
		IGGPDLET	BL
		IGGPDLTS	BL
		IGGPDSTS	A7
		IGGPF4PR	A6
IGGPGDVC	CR	IGGPCRVL	CS
IGGPGET	EG	IGGPAIX	CA
		IGGPALEC	BE
		IGGPALIX	BE
		IGGPALNM	BD
		IGGPALT	BD
		IGGPALVE	BN
		IGGPALY	CD
		IGGPBAMO	CP
		IGGPBCAN	B8
		IGGPBCRA	CR
		IGGPBOUT	CA
		IGGPBSGT	BM
		IGGPCKCC	BM
		IGGPCKEN	CA
		IGGPCLGT	BM
		IGGPCNBO	B8
		IGGPCRAP	CO
		IGGPCRBO	CR
		IGGPCRVL	CS
		IGGPDAIN	BH
		IGGPDAIX	CX
		IGGPDCPC	AP
		IGGPDDCE	BX
		IGGPDEDE	AL
		IGGPDEFS	AQ
		IGGPDELC	AF
		IGGPDELP	CX
		IGGPDELX	CX
		IGGPDELY	CX
		IGGPDEUN	AN
		IGGPDFBO	B8
		IGGPDJCL	BG
		IGGPDLDS	BG
		IGGPDLET	BL
		IGGPDLSF	CL
		IGGPDLTC	BG
		IGGPDPTH	CX
		IGGPDSEM	CA
		IGGPDUND	AN
		IGGPFGRW	CA
		IGGPGETM	BR
		IGGPGREC	BA
		IGGPGVLM	FC
		IGGPINMD	BM
		IGGPISCI	AG
		IGGPLSTC	BQ
		IGGPPASE	CP
		IGGPPRDS	CS
		IGGPPRPW	CA
		IGGPPWGT	BM
		IGGPRELE	CB
		IGGPRUPG	CD
		IGGPSALL	AR
		IGGPSCA	AH
		IGGPSCAT	AH
		IGGPSPSC	BM
		IGGPSSWD	BB
		IGGPUPDE	BB

To Procedure	To Module	From Procedure	From Module
		IGGPUPG	CD
		IGGPVMSC	A7
		IGGPLTRN	EZ
IGGPGETM	BR	IGGPBMR	BR
IGGPGLUB	CO	IGGPCRAO	CO
		IGGPEMIO	AF
IGGPGNEX	AW	IGGPADGO	AW
		IGGPAGOP	AW
		IGGPDOWN	BW
		IGGPMBGO	AX
		IGGPMVAR	AX
IGGPGREC	BA	IGGPADGO	AW
		IGGPAGOP	AW
		IGGPDEL2	AV
		IGGPGREL	AW
		IGGPGREP	AZ
		IGGPGVAL	BA
		IGGPMBGO	AX
		IGGPMVAR	AX
		IGGPSCNF	AZ
		IGGPSFPL	AV
		IGGPSGOP	AV
		IGGPSNK2	BW
		IGGPTSTS	BA
		IGGPXDGO	BT
		IGGPXEL2	BT
		IGGPXEXS	BS
		IGGPXVAL	BS
IGGPGREL	AW	IGGPAGOP	AW
IGGPGREP	AZ	IGGPLOC2	AZ
IGGPGTBF	CB	IGGPDFBO	B8
		IGGPGREC	BA
IGGPGUPG	CD	IGGPALY	CD
		IGGPUPG	CD
IGGPGVAL	BA	IGGPALT2	AX
		IGGPDEIN	BW
		IGGPDOWN	BW
		IGGPLOC2	AZ
		IGGPMVAR	AX
		IGGPRISE	BW
		IGGPTSTS	BA
		IGGPXDGO	BT
		IGGPXEL2	BT
		IGGPXEXS	BS
		IGGPXVAL	BS
IGGPGVL	FB	IGGPAIX	CA
		IGGPDEF	AL
IGGPGVLG	FB	IGGPGVL	FB
IGGPGVLM	FC	IGGPDFSC	AL
IGGPGVSA	FB	IGGPAIX	CA
		IGGPGVL	FB
IGGPIDMP	AT	IGGPACDV	AB
		IGGPALVA	BE
		IGGPCKEN	CA
		IGGPCRAO	CO
		IGGPDCAV	AL
		IGGPDCCC	BX
		IGGPDCDE	AL
		IGGPDCHK	ET
		IGGPDEDE	AL
		IGGPDELC	AF
		IGGPGET	EG
		IGGPGTBF	CB
		IGGPITER	EX
		IGGPLAT2	AX
		IGGPOPEN	CG
		IGGPRPLM	AB
		IGGPSCA	AH
		IGGPSCNC	AY
		IGGPSVOL	BC

Figure 6.13 **Catalog management procedure cross reference (part 4 of 6)**

To Procedure	To Module	From Procedure	From Module
		IGGPXDGO	BT
		IGGPXEL2	BT
		IGGPXLT2	BT
IGGPIGOP	AW	IGGPADGO	AW
		IGGPAGOP	AW
		IGGPASPT	AW
		IGGPMGO	AX
IGGPINIT	BC	IGGPSSWD	BB
		IGGPUPDE	BB
IGGPINMD	BM	IGGPCKAU	BM
IGGPIORA	CG	IGGPADCI	AG
		IGGPFRE	AG
		IGGPISCI	AG
		IGGPUTE	AG
IGGPIOSI	FH	IGGPSCAT	FH
IGGPISCI	AG	IGGPDEFS	AQ
		IGGPSALL	AR
IGGPITER	EX	IGGPDCCC	BX
IGGPIVER	A6	IGGPDEFS	AQ
IGGPLOC	AZ	IGGPACDV	AB
IGGPLOC2	AZ	IGGPSCNF	AZ
IGGPLSTC	BQ	IGGPCDVR	AT
IGGPLTRN	EZ	IGGPLOC2	AZ
IGGPLVAL	BA	IGGPGVAL	BA
IGGPLVST	BM	IGGPCKAU	BM
IGGPMBGO	AX	IGGPMVAR	AX
IGGPMCO	AC	IGGPACDV	AB
IGGPMCO2	AD	IGGPDCOC	AE
		IGGPMCO	AC
IGGPMEBM	AE	IGGPDCSP	AS
IGGPMEXT	BB	IGGPSSWD	BB
		IGGPUPDE	BB
IGGPMGO	AX	IGGPAGOP	AW
		IGGPMBGO	AX
		IGGPMVAR	AX
IGGPMOD	AV	IGGPALMD	BD
		IGGPALSA	BE
		IGGPALVE	BN
		IGGPALVR	BN
		IGGPBAMO	CP
		IGGPBCAN	B8
		IGGPBDSH	B8
		IGGPBDSS	B8
		IGGPBOUT	CA
		IGGPCNBO	B8
		IGGPCRDI	CR
		IGGPCRVL	CS
		IGGPDAIX	CX
		IGGPDAVO	BH
		IGGPDCME	AS
		IGGPDEDD	A7
		IGGPDELP	CX
		IGGPDELX	CX
		IGGPDELY	CX
		IGGPDESH	A7
		IGGPDEUN	AN
		IGGPDLSH	BL
		IGGPDMDS	CL
		IGGPDMOP	CY
		IGGPDPTH	CX
		IGGPDSMD	A6
		IGGPMEXT	BB
		IGGPMODC	CA
		IGGPMVOL	BB
		IGGPNUPG	CD
		IGGPRELE	CB
		IGGPSALL	AR
		IGGPUPG	CD
IGGPMODC	CA	IGGPAIX	CA
IGGPMVAR	AX	IGGPALT2	AX
IGGPMVGO	AW	IGGPADGO	AW
IGGPMVOL	BB	IGGPSSWD	BB
		IGGPUPDE	BB
IGGPNFND	AF	IGGPEMSG	AF
IGGPNUPG	CD	IGGPUPG	CD
IGGPOPEN	CG	IGGPRIO	CG
IGGPPAD	EG	IGGPALNM	BD
		IGGPBAWP	CP
		IGGPCLBF	CB
		IGGPDCPC	AP
		IGGPDEFS	AQ
		IGGPFGRW	CA
		IGGPPREC	AW
IGGPPART	BR	IGGPBMR	BR
IGGPPASE	CP	IGGPPATH	CP
IGGPPATH	CP	IGGPCDVR	AT
IGGPPDE	EG	IGGPALNM	BD
		IGGPCRBO	CR
		IGGPDELC	AF
		IGGPDELP	CX
		IGGPDELY	CX
		IGGPDEL2	AV
		IGGPDEUN	AN
		IGGPDLET	BL
		IGGPDPTH	CX
		IGGPDUND	AN
		IGGPNUPG	CD
		IGGPRIO	CG
		IGGPSNK2	BW
IGGPPGAO	CP	IGGPPASE	CP
IGGPPIN	FA	IGG0CLC9	C9
IGGPPING	FA	IGGPPIN	FA
IGGPPRDS	CS	IGGPCRVL	CS
IGGPPREC	AW	IGGPDEL2	AV
		IGGPDOWN	BW
		IGGPGNEX	AW
		IGGPGREC	BA
		IGGPRISE	BW
		IGGPSFPL	AV
		IGGPSNK2	BW
IGGPPRPW	CA	IGGPAIX	CA
		IGGPBASP	CP
IGGPPTBF	CB	IGGPDCCE	AN
		IGGPPREC	AW
IGGPPUPC	EG	IGGPALT	BD
		IGGPALY	CD
		IGGPCSMP	BT
		IGGPDELS	BL
		IGGPDLSF	CL
		IGGPPRDS	CS
		IGGPPREC	AW
		IGGPPUTM	BR
IGGPPUTM	BR	IGGPBMR	BR
		IGGPGETM	BR
IGGPPWGT	BM	IGGPCKAU	BM
IGGPPWVR	BM	IGGPCKAU	BM
IGGPRAND	FB	IGGPGVSA	FB
IGGPRBAP	CG	IGGPCCCR	EG
		IGGPRCCR	EG
IGGPRCCR	EG	IGGPAXCI	AG
		IGGPISCI	AG
		IGGPOPEN	CG
		IGGPPAD	EG
		IGGPPDE	EG
		IGGPRCU	C9
IGGPRCU	C9	IGG0CLC9	C9
IGGPRDLB	CO	IGGPCRAO	CO
IGGPRELE	CB	IGGPUPD	AV
IGGPRIO	CG	IGGPXIO	CG
IGGPRISE	BW	IGGPDEIN	BW

Figure 6.13 Catalog management procedure cross reference (part 5 of 6)

To Procedure	To Module	From Procedure	From Module
IGGPRLAB	FD	IGGPBMAT	FD
IGGPRPLF	AB	IGGPDELC	AF
		IGGPRCU	C9
		IGGPSCA	AH
IGGPRPLM	AB	IGGPCRAP	CO
		IGGPSCA	AH
		IGGPSCAT	AH
IGGPRUPG	CD	IGGPALY	CD
		IGGPNUPG	CD
		IGGPUPG	CD
IGGPSALL	AR	IGGPALSA	BE
		IGGPCRVL	CS
		IGGPCSAL	BB
		IGGPDCNV	AJ
		IGGPDCSP	AS
		IGGPDSPO	AJ
IGGPSALS	AU	IGGPSALL	AR
IGGPSCA	AH	IGGPCDVR	AT
		IGGPDCOC	AE
		IGGPSCAT	FH
		IGGPVCAX	AH
IGGPSCAT	FH	IGGPACDV	AB
		IGGPCDVR	AT
		IGGPDEFS	AQ
		IGGPDELC	AF
IGGPSCAX	CO	IGGPBCRA	CR
		IGGPCRAO	CO
		IGGPDLCR	BL
		IGGPSCA	AH
		IGGPSDSP	AF
IGGPSCNC	AY	IGGPACDV	AB
		IGGPEXT	AZ
		IGGPMOD	AV
IGGPSCNF	AZ	IGGPEXT	AZ
		IGGPLOC	AZ
IGGPSDSP	AF	IGGPCRBO	CR
		IGGPDELC	AF
IGGPSET	A7	IGGPDEDD	A7
		IGGPDEMV	A7
		IGGPDESH	A7
		IGGPDEVG	A7
IGGPSFPL	AV	IGGPMOD	AV
		IGGPUPD	AV
IGGPSGOP	AV	IGGPADGO	AW
		IGGPDEL2	AV
		IGGPMGO	AX
		IGGPXVAL	BS
IGGPSHIN	AZ	IGGPLOC2	AZ
IGGPSHNK	AX	IGGPDOWN	BW
		IGGPMVAR	AX
		IGGPRISE	BW
		IGGPSINK	BW
IGGPSINK	BW	IGGPDEIN	BW
		IGGPRISE	BW
IGGPSNK2	BW	IGGPDEIN	BW
IGGPSPSC	BM	IGGPCKAU	BM
IGGPSSCR	BF	IGGPBNUN	B8
		IGGPVMSC	A7
IGGPSSWD	BB	IGGPUPDE	BB
IGGPSVOL	BC	IGGPSSWD	BB
		IGGPUPDE	BB
IGGPTCMP	BA	IGGPTSTS	BA
IGGPTRPL	CG	IGGPPDE	EG
		IGGPPUPC	EG
IGGPTSTS	BA	IGGPSCNF	AZ
		IGGPSFPL	AV
IGGPUPD	AV	IGGPACDV	AB
IGGPUPDE	BB	IGGPUPD	AV
IGGPUPG	CD	IGGPAIX	CA
		IGGPALY	CD

To Procedure	To Module	From Procedure	From Module
		IGGPDELX	CX
IGGPUPLT	FB	IGGPGVSA	FB
IGGPUPUP	BW	IGGPRISE	BW
IGGPUQRI	EX	IGGPDSPO	AJ
IGGPUTE	AG	IGGPANCI	AG
IGGPVCHK	ET	IGGPDCHK	ET
		IGGPDVV	FD
		IGGPFLIM	FQ
IGGPVMSC	A7	IGGPDFBO	B8
		IGGPDJCL	BG
		IGGPDLDS	BG
IGGPVMTV	A6	IGGPCOBT	A6
		IGGPCRVL	CS
		IGGPDEFS	AQ
		IGGPDELC	AF
IGGPXDGO	BT	IGGPSFPL	AV
IGGPXEL2	BT	IGGPSFPL	AV
IGGPXEXS	BS	IGGPXVOL	BS
IGGPXEXT	BS	IGGPEXT	AZ
IGGPXIO	CG	IGGPADCI	AG
		IGGPAOCI	AG
		IGGPAXCI	AG
		IGGPCCCR	EG
		IGGPDCPR	AE
		IGGPDOIT	CG
		IGGPFRE	AG
		IGGPGET	EG
		IGGPISCI	AG
		IGGPPAD	EG
		IGGPPDE	EG
		IGGPPRDS	CS
		IGGPPUPC	EG
		IGGPRCCR	EG
		IGGPRIO	CG
		IGGPTRPL	CG
		IGGPUTE	AG
IGGPXLT2	BT	IGGPSFPL	AV
IGGPXMOD	BT	IGGPMOD	AV
IGGPXVAL	BS	IGGPGVAL	BA
IGGPXVOL	BS	IGGPXVAL	BS
IKQALL00		IGGPDEFS	AQ
IKQRDS00		IGGPCOBT	A6
		IGGPCSDG	A6
		IGGPDVSC	CL
		IGGPF4RD	BU
IKQREN00		IGGPALSV	BN
IKQSCR00		IGGPBSCR	B8
		IGGPDCBO	AE
		IGGPDLSC	BL
		IGGPDUSC	A7
		IGGPDVSC	CL
		IGGPSDSP	AF
IKQVTC00		IGGPCOBT	A6
		IGGPCSDG	A6
		IGGPDVSC	CL
		IGGPF4RD	BU
		IGGPF4WR	BU
IKQWDS00		IGGPF4WR	BU

Figure 6.13 Catalog management procedure cross reference (part 6 of 6)

Service Aids

Service aid phases are available for:

- Enabling and disabling snap dumps within the VSAM component.
- Obtaining snap dumps of control blocks.
- Using UPSI to obtain diagnostic information for the VSAM catalog.
- Maintaining VTOC and VOL1 labels on DASD.
- Loading a VSAM phase or a program you have written.

The service aid phases IKQVDUMP and $$BCVS03 are included in the link-edit of VSAM. The other three phases, IKQVEDA, IKQVDU, and $$BCVS04 can be placed in the core image library by executing the following job.

```
// JOB        JOBNAME
// OPTION     CATAL
   INCLUDE    IKQCLNLK
/*
// EXEC       LNKEDT,REAL
/&
```

Enabling and Disabling Snap Dumps

The following snap points are available in VSAM. Each snap ID, if enabled with IKQVEDA, will produce the result indicated. If VSAM is running in the SVA, it must be reloaded from the core image library after the snap dump has been enabled in order to activate the snap, except for SNAP=0010 which takes effect immediately.

Snap number	Result of Enabling this Snap
0001	This snap allows Catalog Management diagnostic information to be obtained. (See section "Using UPSI to obtain Diagnostic Information for the VSAM Catalog" for details.) As snap 0001 uses the UPSI byte, it cannot be run when the user program in the partition also uses the UPSI byte.
0002	This snap enables the Buffer Manager trace, which provides the current usage of VSAM buffering.
0003	This snap enables the CLOSE control block dump at the beginning of CLOSE processing.
0004	This snap enables the VSAM I/O trace facility.
0005	This snap enables the I/O error trace.
0006	This snap enables the OPEN control block dump facility when open processing is complete.
0007	This snap enables the OPEN error trace. Control blocks are printed if an error occurs during open processing.
0008	This snap enables the Catalog Management I/O trace. All I/O operations done by catalog management are printed on SYSLST.
0009	This snap enables the VSAM Record Management error handler trace, allowing display of control blocks for any error detected by VSAM record management.
0010	This snap enables automatic close. VSAM is shipped with this snap enabled. To disable automatic close, disable this snap.

0011 This support enables the managed-SAM control block trace. Refer to *VSE/VSAM Space Management for SAM Feature Logic* for further information.

IKQVEDA is called by:

```
// EXEC IKQVEDA
```

The routine will print on SYSLOG:

```
ENTER FUNCTION ENABLE|DISABLE|END
```

You must enter either:

```
ENABLE SNAP=xxxx
                (where xxxx is one of the snap numbers)
```

or

```
DISABLE SNAP=xxxx
```

or

`END` (to terminate processing).

The program will look for a private core image library and print:

```
NO PRIVATE CORE IMAGE LIBRARY ASSIGNED
```

if it cannot be found and will then look in the core image library for the VSAM phase needed.

If the phase needed cannot be found in a library the program will inform you with the following message:

`phase NOT FOUND IN THE SYSTEM PRIVATE`
`CORE IMAGE LIBRARY` (where `phase` is the actual phase name)

Any error in input will result in the INVALID REPLY message and the ENTER FUNCTION message is reissued.

Entering ENABLE SNAP=0011 in a system without the VSE/VSAM Space Management for SAM Feature installed results in an INVALID REPLY message.

The program can only be ended by the END reply as noted earlier.

The following examples illustrate the use of IKQVEDA to enable and disable SNAP 0001:

```
// EXEC IKQVEDA
   ENTER FUNCTION ENABLE|DISABLE|END
   ENABLE SNAP=0001
   NO PRIVATE CORE IMAGE LIBRARY ASSIGNED
   SNAP 0001 ENABLED
   ENTER FUNCTION ENABLE|DISABLE|END
   DISABLE SNAP=0001
   NO PRIVATE CORE IMAGE LIBRARY ASSIGNED
   SNAP 0001 DISABLED
   ENTER FUNCTION ENABLE|DISABLE|END
   END
```

Obtaining Snap Dumps of Control Blocks

IKQVDUMP enables you to print out snap dumps of record management and catalog control blocks. Code is provided at certain points in VSAM modules which is nonoperational so far as normal execution of the modules is concerned. Refer to "Enabling and Disabling Snap Dumps," above.

IKQVDUMP is called by the following sequence of instructions (see also "Loading a VSAM phase or a Program You Have Written"):

```
          LA    1,PARMLIST
          SVC   2
          .
          .
          .
PARMLIST DC    CL8'$$BCVS03'    B transient
         DC     CL8'IKQVDUMP     phase that provides dump
                                  of control blocks
```

When the program has completed processing, $$BCVS03 returns the program to the instruction immediately following the SVC instruction.

Note that IKQVDUMP requires SYSLST to be assigned to a printer; assigment to disk or tape will result in an error.

Figure 6.14 shows the description and format of the parameter list that follows the two phase names in the above calling sequence.

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
0	0	1	PARMSW1	First byte of parameter list
		1...	PARMAMBL	Dump the AMBL
		.1..	PARMACB	Dump the ACB
		..1.	PARMAMDS	Dump the AMDSB
		...1	PARMARDB	Dump the ARDB
		.... 1...	PARMBCB	Dump the BCB
		.... .1..	PARMBUFE	Dump the buffer
		.... ..1.	PARMEDB	Dump the EDB
		1	PARMLPMB	Dump the LPMB
1	1	1	PARMSW2	Second byte of parameter list
		1...	PARMCCW	Dump the CCW
		.1..	PARMPLH	Dump the PLH
		..1.	PARMBHD	Dump the BHD
		...1	PARMRPL	Dump the RPL
		.... 1...	PARMEXCP	Dump the EXCPAD work area
		.... .1..	PARMCAT	Dump the catalog blocks
		.... ..1.	PARMDATA	Dump the non-catalog blocks
		1	PARMTHB	Dump the THB
2	2	1	PARMSW3	Third byte of parameter list
		1...	PARMOPEN	Dump the open work area
		.1..	PARMCLOS	Dump the close work area
		..1.	PARMCIW	Dump the control interval split area
		...1	PARMVLST	Dump the volume list
		.... 1...	PARMREGS	Dump the registers
		.... .1..	PARMCECL	Dump the control interval exclusive control list
		.... ..1.	PARMODLB	Dump the open DLBL
		1	PARMREQR	Dump the requester's registers
3	3	1	PARMSW4	Fourth byte of parameter list
		1...	PARMPAMB	1=Pointer to start dump is in parameter list (PARMAMBA) 0=Pointer to start dump is in register 11
		..1.	PARMCCAA	1=Pointer to CCA 0=Pointer to AMBL
		...1	PARMRTNA	Call the test routine
		.... 1...	PARMHDID	Dump the header ID
		.x.. .xxx		Available
4	4	4	PARMAMBA	Pointer to start dump
8	8	4	PARMID	Pointer to header
8	8	1	PARMIDLN	Length of the header
9	9	3	PARMIDAD	Address of the ID
12	C	1	PARMSW5	Fifth byte of parameter list
		1...	PARMCCA	Dump the CCA
		.1..	PARMCADL	Dump the CCA DLBL
		..1.	PARMCADP	Dump the CCA DADSM parameter list
		...1	PARMCARA	Dump the CCA record areas
		.... 1...	PARMCPL	Dump the catalog parameter list (CTGPL)
		.... .1..	PARMPLDN	Dump the CTGPL data set name
		.... ..1.	PARMPLNN	Dump the CTGPL new name
		1	PARMPLPW	Dump the CTGPL password

Figure 6.14 **IKQVDUMP parameter list description and format (part 1 of 2)**

Offset Dec	Offset Hex	Bytes and Bit Pattern	Field Name	Description
13	D	1	PARMSW6	Sixth byte of parameter list
		1...	PARMPLCN	Dump the CTGPL catalog name
		.1..	PARMPLCI	Dump the CTGPL control interval number
		..1.	PARMPLDL	Dump the CTGPL file CTGDDNM field
		...1	PARMPLWA	Dump the CTGPL work area
		.... 1...	PARMCFL	Dump the catalog field parameter list (CTGFL)
		.... .1..	PARMFLFD	Dump the CTGFL fields
		.... ..1.	PARMFLFN	Dump the CTGFL field name
		x		Available
14	D	1	PARMSW7	Seventh byte of the parameter list
		1...	PARMCFV	Dump the catalog field vector table (CTGFV)
		.1..	PARMFVDL	Dump the CTGFV file name
		..1.	PARMFVEN	Dump the CTGFV entry name
		...1	PARMFVKR	Dump the CTGFV key range list
		.... 1...	PARMFVVL	Dump the CTGFV volume serial list
		.... .1..	PARMDPDL	Dump the DADSM parameter list DLBL
		.... ..1.	PARMDPIO	Dump the DADSM parameter list I/O area
		1	PARMDPWA	Dump the DADSM parameter list work area
15	F	1	PARMSW8	Eighth byte of parameter list
		1...	PARMDPSV	Dump the DADSM parameter list save I/O area
		.1..	PARMCBS	Dump the AMCBS
		..1.	PARMCAXW	Dump the CAXWA
		...1	PARMCXRL	Dump the CAXWA RPL
		.... 1...	PARMCXDR	Dump the CAXWA VTOC label read-in work area (DRWA)
		.... .1..	PARMCMSW	Dump the CMS work area
		.... ..1.	PARMMSAM	Dump the managed-SAM control blocks.
		x		Available
16	10	8	PARMRTNN	Name of test routine

Figure 6.14 **IKQVDUMP parameter list description and format (part 2 of 2)**

Testing if a Dump is Required

IKQVDUMP allows a phase to be called before a dump is taken to see if a dump is desired. (The name of the test routine must be inserted into the parameter list at field name PARMRTNN.) The phase can use any logic to determine whether a dump is needed, and this logic will override a call for a dump if it is not needed. If a 0 is returned in register 15, the dump will be taken; if register 15 holds a nonzero return, the dump will not be taken.

The registers on entry to the test routine have the following contents:

R2 = Pointer to the parameter list
R11 = Caller's register 11
R13 = Pointer to 18-word save area
R14 = Return address of calling phase
R15 = Address of entry point

Using UPSI to Obtain Diagnostic Information for the VSAM Catalog

Manipulation of the UPSI job control statement enables you to screen catalog return codes and obtain a snap dump, cancel a job (which causes a full dump to be taken), or simply continue processing. You must first use IKQVEDA to enable Snap = 0001. Otherwise the UPSI statement will be inoperative. As snap 0001 uses the UPSI byte, it cannot be run when the user program in the partition also uses the UPSI byte.

The purpose of this service aid is to diagnose catalog errors that occur while running any program that causes the VSAM catalog to execute. Typically this would be an Access Method Services module or a record management program you have written.

The // UPSI nnnnnnnn job control statement must precede the // EXEC [progname] statement. If no UPSI statement is included, the default is // UPSI 00000000 (see type 3 request below).

On exit from catalog management after processing, a message will be printed out depending on the type of UPSI bit setting you have selected. Some messages require a reply from the operator. The return codes in the message are obtained from register 15. The format is:

```
** NNN,MN,RRR,FFFF,CCCCCCCCCCCCCCCC
```

where

NNN is the return code in decimal.

MN are the last two characters of the module name which issued the error. This is blank in case of error code 0.

RRR is the reason code in decimal.

FFFF is one of the following catalog management functions that had been processed:

DEFC	(define catalog)
DEFA	(define non-VSAM data set)
DEFS	(define space)
DEF	(define VSAM data set)
ALT	(alter)
DELC	(delete catalog)
DELS	(delete space)
DEL	(delete VSAM or non-VSAM data set)
LSTC	(list catalog)
UPD	(update or update-extend)
LOC	(locate)

C...C is either the control interval number in decimal or the first 16 characters of the data set name or volume serial number in EBCDIC.

If a reply is required from the system operator for certain types of requests, the operator must enter one of the following replies from the system console:

- Type in SNAP to get a snap dump by means of IKQVDUMP (see "IKQVEDA for enabling snap dumps"). The message will then be repeated and the operator should press the END key to continue processing.
- Type in CANCEL to cancel the job and obtain a full partition dump.
- Press the END key to resume processing.

The following paragraphs describe the four types of UPSI settings you can use to elicit a message and/or to determine the degree of return code screening done:

Type 1 UPSI Setting. If you want to obtain an operator message for all VSAM catalog return codes (including 0), you must include one of the following statements:

`// UPSI 11000000`	No reply is required from the operator
`// UPSI 01100000`	A reply is required from the operator

Type 2 UPSI Setting. An operator message is issued only if the return code is not 0 for the following statements:

`// UPSI 10000000`	No reply is required from the operator
`// UPSI 01000000`	A reply is required from the operator

Type 3 UPSI Setting. An operator message is not issued if one of the following conditions exists:

1. the Access Method Services command being processed was a LISTCAT and the return code is 8, or
2. the return code is 0, 40, 68, or 160

(These codes occur during normal processing and are, therefore, excluded).

If neither of these conditions exists, an operator message is issued for the following statements:

`// UPSI 00000000`	No reply is required from the operator
`// UPSI 01110000`	No reply is required from the operator

Type 4 UPSI Setting. If you want an operator message on a specific return code, you must include the following statements:

`// UPSI 00nnnnnn`	nnnnnn is set to the value, in binary, of the code divided by 4. A reply is required from the operator

Maintaining VTOC and VOL1 Labels on DASD

A VSAM DADSM service aid has been provided to assist the programmer and operator in maintaining the VTOC and VOL1 labels on DASD devices.

The following procedures should be followed to use IKQVDU at the system console for such maintenance. The key difference in the three procedures is the presence, or absence, of a // UPSI job control statement. Steps of the procedure in lower case letters are typed in at the console; steps in upper case letters are printed out.

Procedure 1	**Explanation**
`// assgn sys000,x'cuu'` (press END key)	*cuu* points at the volume you want to use.
`// upsi 1` (press END key)	This job control statement is optional. If it is included, the following events take place on the volume that was assigned to SYS000: • The VSAM volume ownership bit and CRA pointer in the F4 VTOC label are reset. • The entire VTOC is scratched, that is, empty VTOC labels are written over existing F1, F2, and F3 labels, with the exception of labels that have names starting with the characters "DOS." or "PAGE". • An operator authorization prompt is issued if the VTOC label to be scratched is security protected.
`// exec ikqvdu,size=auto` (press END key)	Start execution of the IKQVDU phase

Procedure 2	**Explanation**
`// assgn sys000,x'cuu'` (press END key)	*cuu* points at the volume you want to use.
`// upsi 11` (press END key)	This job control statement is optional. If it is included, the following events take place on the volume that was assigned to SYS000: • The VSAM volume ownership bit and CRA pointer in the F4 label are reset. • The entire VTOC is scratched, that is, F0 labels are written over existing F1, F2, and F3 labels, with the exception of labels that have names starting with the characters "DOS." or "PAGE".
`// exec ikqvdu,size=auto` (press END key)	Start execution of the IKQVDU phase.

Procedure 3	**Explanation**
`// assgn sys000,x'cuu'` (press END key)	*cuu* points at the volume you want to use.
`// exec ikqvdu,size=30k` (press END key)	Start execution of the IKQVDU phase.
`SPECIFY FUNCTION OR REPLY '?' FOR OPTIONS READY` `?` (press END key)	The character ? causes a list of the various functions that IKQVDU performs to be printed out at the system console.

```
TO SET THE VOLUME OWNERSHIP FLAG REPLY 'SET OWNERSHIP'
TO SET THE CRA POINTER REPLY 'SET OWNERSHIP'
TO RESET THE VOLUME OWNERSHIP FLAG AND CRA POINTER REPLY
    'RESET OWNERSHIP' OR 'RESET CRA'
TO SET THE SECURITY FLAG IN A F1 LABEL REPLY 'SET
     SECURITY'
TO RESET THE SECURITY FLAG IN A F1 LABEL REPLY 'RESET
     SECURITY'
TO REMOVE A LABEL FROM THE VTOC REPLY 'SCRATCH'
TO RENAME A LABEL REPLY 'RENAME'
TO ALLOCATE A LABEL REPLY 'ALLOCATE'
TO REINITIATE PROCESSING REPLY 'RESTART'
TO ALTER OR DISPLAY A DASD VOL1 LABEL
REPLY 'CLIP LABEL=SER=N..N' OR 'CLIP LABEL=DISPLAY'
TO TERMINATE PROCESSING REPLY 'END'
READY
```

You can avoid printing out this list of functions simply by specifying the function you wish as follows:

Procedure	**Explanation**
`set ownership` (press END key)	Causes the VSAM ownership bit to be set in the F4 VTOC label and optionally allows the user to set the CRA pointer.
`reset CRA or` `reset ownership`	Causes the VSAM ownership bit and CRA pointer to be reset in the F4 VTOC label.
`set security` (press END key)	Causes the security bit to be set in the F1 VTOC label. When the console responds with ENTER DSN, reply with the data set name of the VTOC label to be modified.
`reset security` (press END key)	Causes the security bit in the F1 label to be reset. When the console responds with ENTER DSN, reply with the data set name of the VTOC label to be modified.
`scratch dsn=dsname` (press END key)	Causes the VTOC label with the specified data set name to be scratched.

`scratch vtoc` (press END key)	Causes the entire VTOC to be scratched with the exception of data set names starting with the characters "DOS." and "PAGE". In addition, an operator-authorization prompt will be issued if the VTOC label is security-protected or describes a catalog.
`rename` (press END key)	Causes the DSNAME portion of the F1 VTOC label to be changed. When the console responds with ENTER OLD DSN, reply with the data set name of the VTOC label to be changed. Be sure to enter the correct OLD DSN. No error checking is performed in case an invalid name is specified. When the console responds with ENTER NEW DSN, reply with the new data set name.
`allocate` (press END key)	Causes a new label to be created and written in the VTOC. In order to utilize this function, a DLBL/EXTENT job control statement must be provided. When the console responds with ENTER FILENAME, reply with the same filename as that in the DLBL statement referred to above. When the console responds with ENTER NEW DSN, reply with the data set name of the data set to be created. When the console responds with DO YOU WISH TO SECURITY PROTECT THIS DATA SET? reply YES or NO. A reply of YES causes the data security bit to be set in the F1 VTOC label. A reply of NO causes the data security bit to be reset.
`restart` (press END key)	Causes processing to be reinitiated with a READY prompt. This keyword can be used as a response to any operator prompt.
`clip label=display` (press END key)	Causes the volume serial number to be displayed on the system console.
`clip label=ser=n..n` (press END key)	Causes the existing volume serial number to be changed to the one specified as n..n.

end
(press END key) Causes processing to terminate.

If an error occurs during execution of IKQVDU,

```
**ERROR** DADSM RETURN CODE IS nnn
```

prints out on the system console. The following shows the message code (nnn), the associated message, and the action required to correct the condition.

Example:

```
** ERROR** DADSM RETURN CODE IS 020 VTOC FULL
```

004 I/O ERROR WHILE READING VOLUME LABEL
Action: if the problem was not caused by a hardware error, restore the volume.

008 VOLUME NOT MOUNTED
Action: Mount the correct volume.

012 I/O ERROR ON VTOC
Action: If the problem was not caused by a hardware error, restore the volume.

016 DUPLICATE NAME ON VOLUME
Action: Choose another filename or scratch the original file from the volume. If duplication is due to key ranges, ensure each UNIQUE key range is on a separate volume.

020 VTOC FULL
Action: Delete any non-VSAM files or VSAM data spaces no longer needed from the volume to make additional Format 1 labels available, or reinitialize the volume with a larger VTOC.

024 EXTENT OVERLAPS EXPIRED FILE
Action: Examine the VTOC listing to determine where the overlap occurred. Correct the EXTENT statement causing the error. To delete the expired file, open a DTF using the same file-ID as that of the expired file, and instruct the operator to reply DELETE to message 4n33A when it is issued.

028 EXTENT OVERLAPS UNEXPIRED FILE
Action: Compare the high and low extent limits on the EXTENT statement or LSERV output with the file or data space limits on the VTOC display. If the extents overlap, correct the EXTENT statement in error.

032 EXTENT OVERLAPS PROTECTED UNEXPIRED FILE
Action: Examine the VTOC to determine where the overlap occurred. Correct the EXTENT statement causing the error. If necessary, use another volume.

036 EXTENT OVERLAPS VTOC
Action: Execute LVTOC. The Format 4 label (the first label in the VTOC display) contains the extent limits of the VTOC. If the program being executed uses a temporary label set and overlaps the VTOC, correct the EXTENT statements that overlap. If the job uses standard or partition standard labels, use the LSERV output to correct the extents of the overlapping file, VSAM data space, or UNIQUE VSAM file. Then rebuild the appropriate label tracks.

040 REQUIRED EXTENTS MISSING
Action: If temporary labels were used, match the extents on the incoming EXTENT card with the extents in the LVTOC output. If standard (permanent) labels were used, match the extents in the LSERV output with those in the LVTOC output.

044 LABEL NOT FOUND
Action: Use the LVTOC output to check for all file labels used in OPEN macros. If the file has been destroyed, it was probably due to deletion of overlapping extents on an unexpired file, and the file must be rebuilt.

048 INVALID LABEL ADDRESS
Action: Examine the VTOC for a label having an invalid forward chain pointer, and delete it. If no invalid labels are found, just rerun the job.

056 EXTENT OVERLAPS PROTECTED EXPIRED FILE
Action: Examine the VTOC listing to determine where the overlap occurred. Correct the EXTENT statement causing the error. If it is not necessary to save the expired file, open a DTF using the same file-ID as that of the expired file, and instruct the operator to reply DELETE to message 4n33A when it is issued.

064 GETVIS FAILURE ENCOUNTERED
Action: Allocate GETVIS area. If VSAM is running in the SVA, re-IPL and specify a new value for SET SVA. If VSAM is running in a partition, rerun the job in a larger partition.

072 CDLOAD FAILURE ENCOUNTERED
Action: Either the CDLOAD directory or the GETVIS area is full. Allocate more space.

080 OVERLAP AMONG NEW EXTENTS
Action: If DLBL and EXTENT statements are included in the program, determine the conflicting extents and correct them. If a standard label set is being used, use the LSERV output to locate and correct the conflicting file extents, and rebuild the standard label tracks.

088 FORMAT 4 LABEL NOT FOUND
Action: Reinitialize the VTOC to create a format-4 label.

092 VOL1 LABEL NOT FOUND
Action: Reinitialize the volume to create a VOL1 label.

096 JIB PROCESSING FAILURE
Action: Rerun the job when more JIBs are available.

Loading a VSAM Phase or a Program You Have Written

If you want to load and transfer control to and from a selected VSAM phase or a program you have written, you can use B-transient $$BCVS03 without destroying any registers in the following calling sequence:

```
            LA    1,PARMLIST
            SVC   2
             .
             .
             .
PARMLIST    DC    CL8'$$BCVS03'     B transient
RTNNAME     DC    CL8'XXXXXXXX'     Name of phase or program
                                      you have written
USERLIST    DC                       Parameter list for phase
             .                        'XXXXXXXX'
             .
             .
```

When control is received by 'XXXXXXXX', the registers have the following contents:

R0	=	Address of a work area (the size of the work area is specified by a halfword at offset 4 of 'XXXXXXXX' phase)
R1	=	Pointer to user's parameter list (USERLIST)
R2-13	=	Remain the same as they were when SVC 2 was issued
R14	=	Return address of calling module
R15	=	Address of entry point in 'XXXXXXXX'

Control is returned from 'XXXXXXXX' by a BR 14 instruction.

Definitions of Terms Used In This Book

Access Method Services. A multifunction service program that defines VSAM data sets and allocates space for them, converts indexed sequential data sets to key-sequenced data sets with indexes, modified data-set attributes in the catalog, recognizes data sets, facilitates data portability between operating systems, creates backup copies of data sets and indexes, helps make inaccessible data sets accessible, and lists data-set records and catalog entries.

address direct access. The retrieval or storage of a data record identified by its RBA (relative byte address) independent of the record's location relative to the previously retrieved or stored record. (See also keyed direct access, addressed sequential access, and keyed sequential access.)

addressed sequential access. The retrieval or storage of a data record in its entry (RBA) sequence relative to the previously retrieved or stored record. (See also keyed sequential access, addressed direct access, and keyed direct access.)

allocation chain (AC). All allocation units containing control blocks for the same ACB.

allocation unit (AU). One or more pages of virtual storage containing control blocks referenced by record management.

alternate index. A collection of index entries, related to a give base cluster and organized by a key other than the prime key of the associated base data records. Its function is to provide an alternate means of locating records in the data portion of the base cluster.

alternate index upgrade. The process of reflecting changes made to a base cluster in its associated alternate index(es). (See also upgrade set.)

alternate key. A key, other than the prime key, used to form an alternate index.

application. As used in this publication, the use to which an access method is put or the end result that it serves; contrasted to the internal operation of the access method.

backward processing. A variation of sequential processing, whereby the previous, rather than the next, record in the entry, key, or relative-record sequence is retrieved.

base catalog record. The first catalog record (control interval) that describes the VSAM object. This record contains the object's data set name, cluster name, or volume serial number in the ENTNAME field. This record also contains the header fields required for the object. The base catalog record can contain group occurrence pointers that point to group occurrences in the base catalog record, or that point to group occurrences in extension records (vertical extension). The base catalog record's extension pointer can point to a control interval that continues the information (group occurrence pointers) contained in the base catalog record (horizontal extension).

base cluster. A key-sequenced or entry-sequenced data set over which one or more alternate indexes are built.

candidate volume. A direct-access storage volume that has been defined in a VSAM catalog as a VSAM volume; VSAM can automatically allocate space on this volume, as needed. (See also overflow volume.)

catalog. (See VSAM catalog.)

cluster. A combination of related VSAM data sets, identified by one name in the VSAM catalog, that is, a key-sequenced data set and its index or an entry-sequenced data set alone.

collating sequence. An ordering assigned to a set of items, such that any two sets in that assigned order can be collated. As used in this publication, the order defined by the System/370 8-bit code for alphabetic, numeric, and special characters.

component. As used in this manual, a group of modules that perform a function, such as Open.

compression. (See key compression.)

control area. A group of control intervals used as a unit for formatting a data set before adding records to it. Also, in a key-sequenced data set, the set of control intervals pointed to by a sequence-set index record; used by VSAM for distributing free space and for placing a sequence-set index record adjacent to its data.

control-area split. The movement of the contents of some of the control intervals in a control area to a newly created control area, to facilitate the insertion or lengthening of a data record when there are no remaining free control intervals in the original control area.

control interval. A fixed-length area of direct-access storage in which VSAM stores records and distributes free space. It is the unit of information transmitted to or from direct-access storage by VSAM, independent of blocksize.

control interval access. The retrieval or storage of the contents of a control interval.

control-interval split. The movement of some of the stored records in a control interval to a free control interval, to facilitate the insertion or lengthening of a record that won't fit in the original control interval.

CRA. Catalog recovery area. An entry-sequenced data set which exists on each volume owned by a recoverable catalog, including the catalog volume itself. The CRA contains records which describe the volume and the data sets on the volume.

data integrity. Preservation of data or programs for their intended purpose. As used in this publication, the safety of data from inadvertent destruction or alteration.

data record. A collection of items of information from the standpoint of its use in an application and not from the standpoint of the manner in which it is stored (see also stored record).

data security. Prevention of access to or use of data or programs without authorization. As used in this publication, the safety of data from unauthorized use, theft, or purposeful destruction.

data set. The major unit of data storage and retrieval in the operating system, consisting of data in a prescribed arrangement and described by control information to which the system has access. As used in this publication, a collection of fixed- or variable-length records in direct-access storage, arranged by VSAM in key sequence or in entry sequence. (See also key-sequenced data set and entry-sequenced data set.)

data space. A storage area defined in the volume table of contents of a direct-access volume for the exclusive use of VSAM to store data sets, indexes, and catalogs.

direct access. The retrieval or storage of data by a reference to its location in a data set rather than relative to the previously retrieved or stored data. Direct access is equivalent to ISAM random access. (See also addressed direct access and keyed direct access.)

distributed free space. Space reserved within the control intervals of a key-sequenced data set for inserting new records into the data set in key sequence; also, whole control intervals reserved in a control area for the same purpose.

dynamic storage area (DSA). A block of storage set aside on entry to open/close which may be suballocated to provide for temporary storage requirements of individual modules.

entry-sequence. The order in which data records are physically arranged in direct-access storage, without respect to their contents. (Contrast to key sequence.)

entry-sequenced data set. A data set whose records are loaded without respect to their contents, and whose relative byte addresses cannot change. Records are retrieved and stored by addressed access, and new records are added at the end of the data set.

exclusive control. (See hold.)

extension record. The continuation of a catalog record that contains group occurrence pointers and their group occurrences. Group occurrence pointers in an extension record always point to group occurrences within the extension record. The extension record's extension pointer can point to a control interval that contains part of a group occurrence too large to fit in the extension record (horizontal extension).

extent. A continuous space allocated on a direct-access storage volume, reserved for a particular data space or data set.

external procedure. A procedure that can be called by any other VSAM procedure; a procedure whose name is in the module's (assembler listing) "external symbol dictionary."

field. In a record or a control block, a specified area used for a particular category of data or control information.

file. (See data set.)

fixed block architecture (FBA). A direct access storage device that supports a fixed, 512-byte physical record size. The counterpart, count-key-data (CKD) device, permits variable record sizes.

free space. (See distributed free space.)

free space percentage. (See packing factor.)

generic key. A high-order portion of a key, containing characters that identify those records that are significant for a certain application. For example, it might be desirable to retrieve all records whose keys begin with the generic key AB, regardless of the full key values.

group code. A code that identifies the type of group occurrence. (See Field Name Dictionary for a list of group codes.)

group occurrence. Related fields of information in catalog records. See "Group Occurrences in Catalog Records" in the "Data Areas" section for further details.

group occurrence pointer. A field used to identify and locate a group occurrence by its displacement from the beginning of the record's group occurrences (the group occurrence is in the same control interval as the group occurrence pointer) or by a control interval number (the group occurrence point is in the base catalog record or its extension and the group occurrence is in an extension record). Group occurrence pointers are grouped by type code and are in ascending sequence by sequence number.

high-used RBA. The next byte past the end of the last control interval containing significant data, for ESDA; otherwise, the RBA at which the last SEOF is written.

high-water RBA. The high-used RBA of a data set.

hold. Exclusive control exercised over data or index during an update, erase, or insert operation to prevent another request from making interim changes between initiation and completion of the original request.

horizontal extension. An extension record pointed to by a catalog record's extension field. (See also vertical extension.)

horizontal pointer. A pointer in a sequence set index record that gives the location of the next index record in the same sequence set; used for keyed sequential access.

index. As used in this publication, an ordered collection of pairs, each consisting of a key and a pointer, used by VSAM to sequence and locate the records of a key-sequenced data set; organized in levels of index records. (See also index level, index set and sequence set.)

index entry. A key and a pointer paired together, where the key is the highest key (in compressed form) entered in an index record in the next lower level or contained in a data record in a control interval, and the pointer gives the location of that index record or control interval.

index level. A set of index records that order and give the location of records in the next lower level or of control intervals in the data set that it controls.

index record. A collection of index entries that are retrieved and stored as a group. (Contrast to data record.)

index replication. The use of an entire track of direct-access storage to contain as many copies of a single index record as possible; reduces rotational delay.

index set. The set of index levels above the sequence set. The index set and the sequence set together comprise the index.

integrity. (See data integrity.)

internal procedure. A procedure that can be called only by another procedure within the module. (See also external procedure.)

I/O threshold. The maximum number of buffers that can be filled with data before I/O will be started.

ISAM interface. A set of routines that allow a processing program coded to use ISAM (Indexed Sequential Access Method) to gain access to a VSAM key-sequenced data set.

job catalog. A catalog made available for a job by means of the filename IJSYSUC in the corresponding DLBL job statement.

key. One or more characters within an item of data that are used to identify it or control its use. As used in this publication, one or more consecutive characters taken from a data record, used to identify the record and establish its order with respect to other records. (See also key field and generic key.)

key compression. The elimination of characters from the front and the back of a key that VSAM does not need to distinguish the key from the preceding or following key in an index record; reduces storage space for an index.

key-field. A field located in the same position in each record of a data set, whose contents are used for the key of the record.

key-sequence. The collating sequence of data records, determined by the value of the key field in each of the data records. May be the same as, or different from, the entry sequence of the records.

key-sequenced data set. A data set whose records are loaded in key sequence and controlled by an index. Records are retrieved and stored by keyed access or by addressed access, and new records are inserted in the data set in key sequence by means of distributed free space. Relative byte addresses of records can change.

keyed direct access. The retrieval or storage of a data record by use of either an index that related the record's key to its relative location in the data set, or a relative-record number, independent of the record's location relative to the previously retrieved or stored record. (See also addressed direct access, keyed sequential access, and addressed sequential access.)

keyed sequential access. The retrieval or storage of a data record in its key or relative-record sequence relative to the previously retrieved or stored record, as defined by the sequence set of an index. (See also addressed sequential access, keyed direct access, and addressed direct access.)

LOCK wait. A wait for the release of a named resource that is locked by another VSE task, using the LOCK macro.

mass sequential insertion. A technique VSAM uses for keyed sequential insertion of two or more records in sequence into a collating position in a data set; more efficient than inserting each record directly.

master catalog. The main VSAM catalog, that contains extensive data set and volume information required by VSAM to be able to locate data sets to allocate and deallocate storage space, to verify the authorization of a program or operator to gain access to a data set, and to accumulate usage statistics. (See also job catalog, user catalog.)

max-CA. A unit of allocation equivalent to the maximum control area size on a count-key-data or fixed block device. On a CKD device, the max-CA is equal to one cylinder.

min-CA. A unit of allocation equivalent to the minimum control area size on a count-key-data or fixed block device. On a CKD device, the min-CA is equal to one track.

module. As used in this manual, a program unit that is identifiable by means of a symbolic name starting with IGG0 or IKQ.

nonunique. Space for a nonunique data set or index must be a suballocation from existing data spaces.

object. As used in this manual, a cluster, a data set, an index, a catalog, or a data space.

overflow volume. When space on a candidate volume is allocated by VSAM, that volume is then termed an overflow volume. (See also candidate volume.)

overlapped operation. An operation in which processing continues without waiting for completion of input or output which had been initiated.

packing factor. Percentage of the data object's space allocation to be reserved during its initial loading and during subsequent reorganization. (See also distributed free space.)

password. A unique string of characters stored in a catalog that a program, a computer operator, or a terminal user must supply to meet security requirements before a program gains access to a data set.

physical record. The smallest readable or writable unit of data that is stored on a direct-access storage device. Records are separated from each other by interrecord gaps.

pointer. An address or other indication of location. For example, an RBA is a pointer that gives the relative location of a data record or a control interval in the data set to which it belongs. (See also horizontal pointer and vertical pointer.)

portability. The ability to use VSAM data sets with different operating systems. Volumes whose data sets are cataloged in a user catalog can be demounted from storage devices of one system, moved to another system, and mounted on storage devices of that system. Individual data sets can be transported between operating systems using Access Method Services.

prime index. The index of a key-sequenced data set which is a base cluster, and thus has one or more alternate indexes. (See also index, alternate index.)

prime key. The key which is used to form the prime index. (See also key, alternate key.)

procedure. A functional unit of VSAM code that is entered only at one entry point and exits at the end of the procedure (the last line of the procedure's code). The procedure can call (transfer control, with a return to the procedure expected) other procedures within the module (internal calls) and can call other procedures in other VSAM modules (external calls). (See also internal procedure and external procedure.)

pseudo hold. For SHAREOPTIONS(4), a second hold on the same control area by a single VSE task, under a single ACB (or both hold requests under the same Local Shared Resource pool). The control area is treated as held by the second request, as well as the first. If the second request encounters an actual confilct with the first request, then the second request will receive an "exclusive control conflict" return code (X'08' in register 15, X'14' in the RPL feedback).

random access. (See direct access.)

RBA. Relative byte address. The displacement of a data rcord or a control interval from the beginning of the data set to which it belongs; independent of the manner in which the data set is stored.

record. (See index record, data record, physical record, stored record.)

recoverable catalog. A catalog defined with the recoverable attribute. Duplicate catalog entries are stored in CRAs, and can be used to recover data in the event of a catalog failure. (See also CRA.)

recovery mode. A user option that causes the data object's initial allocation of space to be written throughout with special records, the last of which is set to 0 and is termed the SEOF (software end of file) record. This must be done if VERIFY is to be used. (See also speed mode.)

relative byte address. (See RBA.)

relative-record data set. A data set whose records are loaded into fixed-length slots and numbered by the relative numbers of the slots they occupy.

relative-record number. A number that identifies not only the slot, or data space, in a relative-record data set but also the record occupying the slot. Used as a key for keyed access to a relative-record data set.

relative repetition number. An integer representing the position of a particular field in a group of repeating fields. For example, in EOV, the relative repetition number (RELREPNO) of a particular volume in the catalog data record of a particular cluster is that number of its occurrence in the volume repeating group. EOV uses the RELREPNO to obtain information about a particular volume in order to update the ARDB and EDB.

replication. (See index replication.)

reusable data set. A VSAM data set which can be reused as a work data set, regardless of its old contents.

routine. As used in this manual, an ordered set of instructions that may have frequent use, generally internal usage within a module.

scratch (adj.). Used to describe the contents of a buffer that are no longer valid.

scratch (v.). In buffer management, used to indicate that a buffer contains nothing of significance; in DADSM, to remove a DSCB.

section. A subdivision of an index record used to expedite location of the place in an index record where an entry-by-entry key search can begin.

security. (See data security.)

SEOF. (See software end of file.)

sequence set. The lowest level of the index of a key-sequenced data set; it gives the locations of the control intervals in the data set and orders them by the key sequence of the data records they contain. The sequence set and the index set together comprise the index.

sequential access. The retrieval or storage of a data record in either its entry sequence, its key sequence, or its relative-record number sequence relative to the previously retrieved or stored record. (See also addressed sequential access and keyed sequential access.)

skip sequential access. Keyed sequential retrieval or storage of records in ascending, non-consecutive sequence (with skips); VSAM scans the sequence set to find a record or a collating position.

slot. A fixed-length, numbered space in a relative-record data set which accepts one data record. (See also relative-record data set, relative-record number.)

software end of file. A control interval with a CIDF of 0 that marks the end of preformatted records in a data object's initial allocation of space when the user specifies recovery mode of processing. (See also recovery mode.)

spanned record. A logical record whose length exceeds the control interval size and thus crosses, or spans, one or more control interval boundaries within a single control area.

stored record. A data record, together with its control information, as stored in a direct-access storage device.

string. A string is a single record or a sequentially ordered set of records in a data set. The maximum number of strings (STRNO) to be processed concurrently in a data set is established when a data set is opened. The number of active RPLs determines the number of concurrent strings being processed at any point in time.

string-set. Set of strings that are in communication with each other. For normal processing, this is the set of active RPLs referring to the same ACB. For Local Shared Resources, this is the set of all active RPLs using the Local Shared Resource pool.

unique. (1) A unique data space is occuplied by only one VSAM data set, and cannot be shared with other data sets. (2) A unique alternate key is one which occurs in only one data record in the base cluster. The alternate index record containing this key thus has only one pointer to the base cluster.

upgrade set. All the alternate indexes that VSAM has been instructed to update whenever there is a change to the data of the related base cluster.

user catalog. An optional catalog used in the same way as the master catalog and pointed to by the master catalog. It lessens the contention for the master catalog and facilitates volume portability.

vertical extension. An extension record pointed to by a group occurrence pointer in the object's base catalog record or its horizontal extension. (See also base catalog record and horizontal extension.)

vertical pointer. A pointer in an index record of a given level that gives the location of an index record in the next lower level or the location of a control interval in the data set controlled by the index.

VSAM catalog. A key-sequenced data set containing extensive data-set and volume information that VSAM requires to locate data sets, to allocate and deallocate storage space, to verify authorization of a program or operator to gain access to a data set, and to accumulate usage statistics for data sets. (See also master catalog, job catalog, user catalog.)

Index

This index lists the modules by their descriptive names, followed by their symbolic names. The symbolic names, together with further information about the modules, can be found in the module directory in the *Directory* chapter.

VSE/VSAM
VSAM Logic, Volume 1
LY24-5191-1

READER'S COMMENT FORM

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate. Comments may be written in your own language; use of English is not required.

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

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

	Yes	*No*
• Does the publication meet your needs?	☐	☐
• Did you find the material:		
Easy to read and understand?	☐	☐
Organized for convenient use?	☐	☐
Complete?	☐	☐
Well illustrated?	☐	☐
Written for your technical level?	☐	☐

• What is your occupation? ____________________

• How do you use this publication:

As an introduction to the subject?	☐	As an instructor in class?	☐
For advanced knowledge of the subject?	☐	As a student in class?	☐
To learn about operating procedures?	☐	As a reference manual?	☐

Your comments:

If you would like a reply, please supply your name and address on the reverse side of this form.

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

LY24-5191-1

Reader's Comment Form

Cut or Fold Along Line

Fold and Tape | Please Do Not Staple | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department G60
P. O. Box 6
Endicott, New York 13760

Fold | Fold

If you would like a reply, *please print:*

Your Name ____________________
Company Name ____________________ *Department* ____________
Street Address ____________________
City ____________________
State ____________________ *Zip Code* ____________
IBM Branch Office serving you ____________________

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N. Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N. Y., U. S. A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N. Y., U. S. A. 10601

VSE/VSAM VSAM Logic, Volume 1 (File No. S370-30) Printed in U.S.A. LY24-5191-1

VSE/VSAM VSAM Logic, Volume 1 (File No. S370-30) Printed in U.S.A. LY24-5191-1

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N. Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N. Y., U. S. A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N. Y., U. S. A. 10601