LY33-6034-0

# Customer Information Control System/Virtual Storage (CICS/VS) Version 1, Release 4.1

# Program Logic: Description (OS/VS)

Program Product

Program Number 5740-XX1 (CICS/OS/VS)

IBM

First Edition (July 1979)

This edition applies to Version 1, Release 4, Modification 1 (Version 1.4.1) of the IBM program product Customer Information Control System/Virtual Storage (CICS/VS), program number 5740-XX1.

This edition contains Sections 1,2, and 3 of the Version 1.4 CICS/VS Program Logic (OS/VS), LY33-6029-1. Changes from that edition are indicated by vertical lines to the left of the changes.

Information in this publication is subject to change. Any such changes will be published in new editions or technical newsletters. Before using this publication, consult the latest IBM System/370 Bibliography, GC20-0001, to learn which editions and technical newsletters are applicable and current.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM United Kingdom Laboratories Ltd., Technical Documentation Department, Hursley Park, Winchester, Hampshire, SO21 2JN, England. 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.

Publications are not stocked at the address given above. Requests for copies of IBM publications should be made to the IBM branch office serving your locality.

# PREFACE

The CICS/OS/VS Program Logic, LY33-6029 has been split into two separate manuals. This manual, the CICS/OS/VS Program Logic: Description contains three major sections from the CICS/OS/VS Program Logic. These sections are "Introduction", "Method of Operation", and "Program Organization". The other half of the split CICS/OS/VS Program Logic, named the CICS/OS/VS Program Logic: Data Areas, contains the "Directory" and "Data Areas" sections. The original section on "Diagnostic Aids" has been dropped as the information previously in this section is now fully documented in the CICS/VS Problem Determination Guide, SC33-0089.

This publication provides logic information for programming support representatives and system programmers who maintain the IBM Customer Information Control System/Virtual Storage in an OS/VS environment. It describes CICS/OS/VS. When used with CICS/OS/VS source-program listings and CICS/OS/VS Program Logic: Data Areas, this book enables you to understand the internal operation of CICS/OS/VS. The two program logic manuals do not replace the program listings; they supplement them and make the information in them more accessible.

The major parts of this book are:

- "Introduction," which contains general information about CICS/OS/VS services and overall operation
- "Method of Operation," which contains Hierarchy Input Process Output (HIPO) diagrams that describe CICS/OS/VS in terms of the functions it performs
- "Program Organization," which describes CICS/OS/VS in terms of its parts (which, in CICS/OS/VS, are source modules); an interface diagram is given for each of the main parts of CICS/OS/VS
- "Bibliography," which identifies other CICS/OS/VS publications and additional IBM publications of use to you when working with CICS/OS/VS

Note: In this publication, the term VTAM refers exclusively to the program product ACF/VTAM, program number 5735-RC2.

## TABLE OF CONTENTS

## FIGURES

The IBM Customer Information Control System/Virtual Storage (CICS/VS) is a data base/data communication system that provides an interface between a System/370 operating system and user-written application programs. Applicable in most online environments, CICS/VS provides many of the facilities for standard applications: message switching, inquiry, data collection, order entry, and conversational and batched data entry.

CICS/VS is available as two program products: CICS/DOS/VS, which executes under control of the Disk Operating System/Virtual Storage (DOS/VS), and CICS/OS/VS, which executes under Operating System/Virtual Storage (OS/VS1 or OS/VS2). Both program products consist of modular components from which a user installation can select and tailor a CICS/VS system to meet particular application processing needs. This publication, together with its companion publication CICS/OS/VS Program Logic: Data Areas, deals with CICS/OS/VS, and all subsequent references to CICS/VS in this publication refer to CICS/OS/VS.

The CICS/VS Introduction to Program Logic provides a technical overview of CICS/VS. Users are advised to read the CICS/VS Introduction to Program Logic before using this manual.

## USING THE PROGRAM LOGIC: DESCRIPTION AND PROGRAM LOGIC: DATA AREAS MANUALS

The two Program Logic Manuals, namely the CICS/OS/VS Program Logic: Description and the CICS/OS/VS Program Logic: Data Areas, provide the user with information that will help him to use his source listings and locate the causes of any problems that may occur.

Once the user has a technical overview of CICS/VS (by reading the CICS/VS Introduction to Program Logic, for example) and is aware of the general functions being performed, he can identify the specific modules involved in any function in which he is interested via the method of operation (HIPO) charts in Section 2 of this manual. Many of the modules interact extensively with other CICS/VS modules, or make frequent use of external control blocks. This interaction is described in Section 3 by means of interface diagrams supported by notes.

Certain control blocks provide vital system information. The format and content of these is detailed in Section 2 of the CICS/OS/VS Program Logic: Data Areas. Section 1 of the same publication provides an alphabetical list of all CICS/VS source modules, together with a brief description of each module.

## PURPOSE AND FUNCTIONS

CICS/VS is a data base/data communication interface between application programs and the host operating system. It provides most facilities needed in an online environment. In doing so, CICS/VS manages concurrent requests from:

- Terminals on a telecommunication network of mixed devices to activate transaction and execute a variety of user-written application programs
- Application programs for accesses to data sets and terminals, obtaining or freeing of storage, or use of other resources

The components of CICS/VS are:

- System Management – the supervisory functions: Task, Storage, Program, and Time Management, and the data management functions: Terminal, File, Transient Data, Temporary Storage, and Journal Management
- System Reliability – abnormal condition handling with possible recovery action including user-written error exits and error programs
- System Monitoring – the testing and debugging aids: Trace Management and Dump Management
- Application Services – the techniques for reducing application programming effort, including Basic Mapping Support, 2260 compatibility, and built-in functions
- System Services – the ready-to-use application programs for sign-on/sign-off procedures, master, supervisor, and operator terminal functions, displaying statistics, batched transaction processing, dynamic opening and closing of data sets, time-of-day control, terminal test, initiating snap dumps, and message switching
- System Support – the facilities needed to initiate and terminate CICS/VS sessions, to generate a CICS/VS system and to define the CICS/VS environment of the user's selection; the High-Level Language Preprocessor; and the CICS/VS Dump Utility programs

These components and their parts are summarized in the _CICS/VS Introduction to Program Logic_.

## ENVIRONMENTAL CHARACTERISTICS

CICS/VS operates as an OS/VS task within an OS/VS region in either a dedicated or multiprogramming environment. The selection of the environment is the user's responsibility, as is the selection of system options beyond those required for the operation of CICS/VS.

## SYSTEM CONFIGURATION

The system configuration required for CICS/VS is largely determined by the scope of the environment to be supported and the nature of the user's applications. A list of the terminals and control units supported by CICS/VS is given in the _CICS/VS General Information_.

## CICS/VS INTERFACES

The principal system interfaces of CICS/VS are as follows:

- Interfaces between CICS/VS and the host operating system (OS/VS1 or OS/VS2):
  - the CICS/VS data management functions, using CICS/VS system control tables, controlling the use of access methods, which are listed below
  - CICS/VS System Recovery program, for SPIE and STAE (OS/VS1 and OS/VS2 Release 1), and SPIE and ESTAE (OS/VS2 Release 2 onwards)
  - CICS/VS Task Management and Time Management, for the interval timer, to regain control after a WAIT releases control to OS/VS
  - CICS/VS System Initialization program, when obtaining or freeing storage, attaching tasks, opening files, or initially giving control to CICS/VS
- Interfaces between user-written application programs and CICS/VS:
  - Command-level interface using EXEC commands and the EXEC interface Block (EIB)
  - Macro-level interface using CICS/VS macro instructions, Task Control Area (TCA), Common System Area (CSA), and Terminal Control Table terminal entry.
- Interfaces between CICS/VS modules: system areas, work areas, system control and service tables, and allocation maps

A detailed listing of the required and optional components of OS/VS1 and OS/VS2 is given in the CICS/VS General Information.

The major interfaces of CICS/VS are the Common System Area (CSA) and the Task Control Area (TCA). The CSA contains the system constants, module addresses, control table addresses, task abnormal termination interface, time management storage, and work area for accumulating system statistics. On a CICS/VS system, the open list address pointers in the Common System Area point indirectly to the active interfaces of CICS/VS and the access methods of the operating system. The normal diagnostic and serviceability aids as provided by the operating system are utilized where applicable.

The TCA is the major vehicle for carrying parameter information from a task requesting services to the CICS/VS management module or service program that provides those services. It consists of a system control section, an application program communication section, and, optionally, a user Transaction Work Area (TWA).

## ACCESS METHODS

Standard access methods are supported by CICS/VS. These are:

- Basic Telecommunications Access Method (BTAM)

- Direct Access Method (DAM)
- Sequential Access Method (SAM)
- Indexed Sequential Access Method (ISAM)
- Telecommunications Access Method (TCAM)
- Virtual Storage Access Method (VSAM)
- Virtual Telecommunications Access Method (VTAM)

Either BTAM or VTAM, and DAM and SAM are essential to the running CICS/VS. Depending on the data base configuration, ISAM, VSAM or both may be required. Data sets created using any of these access methods can be operated on by application programs executing under the control of CICS/VS and by offline batch-processing programs.

In addition, CICS/VS users can use the Data Language/I (DL/I) facility of the program product Information Management System/Virtual Storage, (IMS/VS) (program number 5740-XX2)). DL/I is a general-purpose data management facility designed to satisfy diverse data base processing requirements in either a batch or an online environment. Through the CICS/VS-DL/I Interface CICS/VS application programs can access DL/I data bases using the standard DL/I CALL functions.

## PHYSICAL CHARACTERISTICS

CICS/VS is designed to run in a virtual storage environment. A system such as OS/VS1 or OS/VS2 provides CICS/VS with more address space than the real storage capacity of the computer on which it is run. This increase is accomplished by means of a paging technique, which simulates real storage on direct access storage. Real storage is organized into page frames which correspond to pages organized in a page data set on direct access storage. A description of paging is provided in the CICS/VS Introduction to Program Logic.

## CICS/VS ADDRESS SPACE

The CICS/VS main storage address space is structured as follows:

- CICS/VS nucleus, located in the high end of the CICS/VS address space, and composed of: CICS/VS modules, CICS/VS tables, and program storage for resident (fixed and non-fixed) application programs
- CICS/VS dynamic storage used for CICS/VS control areas, input/output areas, work areas, and program storage for nonresident loaded application programs
- OS/VS storage required for: OS/VS access methods and OS/VS work areas

In the CICS/VS nucleus, the high-usage CICS/VS modules (Task, Time, Terminal, and Storage Management) need not be fixed in real storage because activity causes them to be effectively fixed. Under OS/VS1, or under OS/VS2, the CICS/VS user has the option to tailor the placement of the nucleus modules in real storage through the use of the Nucleus Load Table (NLT). This table allows the user to structure the smallest working set for the CICS/VS nucleus and also, where necessary, specify modules as non-pageable. If a NLT is not specified at system initialization, a default load order is used to build the nucleus and modules will be paged in accordance with frequency of use.

The storage organization for CICS/VS is shown in Figure 1-1. Several aids for determining initial address space size are provided in the CICS/VS System Programmer's Reference Manual.

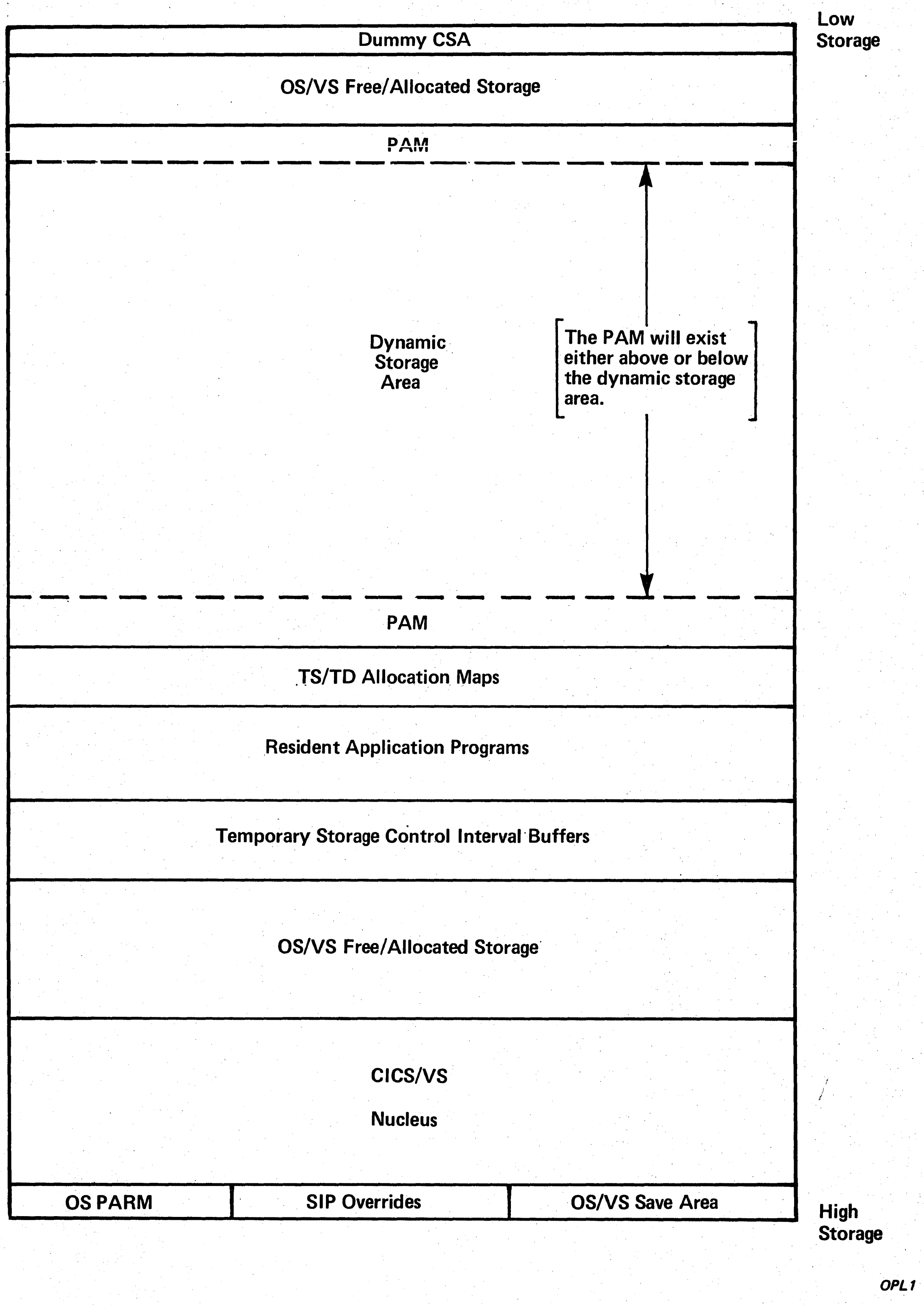

Low Storage
Dummy CSA
OS/VS Free/Allocated Storage
PAM
Dynamic Storage Area
The PAM will exist either above or below the dynamic storage area.
PAM
TS/TD Allocation Maps
Resident Application Programs
Temporary Storage Control Interval Buffers
OS/VS Free/Allocated Storage
CICS/VS Nucleus
OS PARM
SIP Overrides
OS/VS Save Area
High Storage
OPL1

Figure 1-1. Storage Organization Under CICS/OS/VS

### SUBPOOL ALLOCATION OF DYNAMIC STORAGE

The CICS/VS Storage Control program manages the CICS/VS dynamic storage space. It allocates storage, dependent on the type of request, from different subpools. Each subpool obtains and returns storage in page multiples from the dynamic storage area.

The subpools used by CICS/VS are:

- The program subpool containing dynamically loaded application programs
- The control subpool containing resource management storage
- The shared subpool containing temporary storage, BMS maps, DL/I areas, tables, etc.
- The teleprocessing subpool containing line and terminal input/output areas
- The mixed and isolated subpools containing user areas and file storage
- The RPL subpool containing parameter lists for VTAM

The effect of the use of subpools is to minimize the total number of page frames that are used for CICS/VS dynamic storage, while ensuring adequate performance

If insufficient real address space is available for the dynamic storage area, CICS/VS is usually able to operate, but with reduced performance, due to excessive virtual storage paging. An optimum ratio of virtual to real should be determined for each user installation, recognizing the processing requirements of that installation.

## DATA SETS

### CICS/VS DATA SETS

The access methods for use with CICS/VS data sets and the record format of these data sets are predefined within CICS/VS. Any required formatting of the data sets is performed by CICS/VS during initialization or by the CICS/VS maintenance functions. The CICS/VS data sets are:

- CICS/VS Program Library
- Restart data set
- Dump data set
- Intrapartition data set
- Temporary Storage data set
- System Log data set

- Automatic Statistics data set
- Auxiliary Trace data set
- Trace data set

CICS/VS data sets other than the Dump data set, the Trace data set, the System Log data set, and other data sets (for example, the Automatic Statistics data set) which may contain extrapartition transient data must be located on direct access storage. The Dump data set, Trace data set, System Log data set, and extrapartition transient data sets may be on either direct access storage or magnetic tape. Whether all of the data sets are required, and the amount of space needed for each, depends on the CICS/VS options selected at CICS/VS generation and the extent to which they are used. Information about these data sets is given in the CICS/VS System Programmer's Guide (OS/VS) and in the CICS/VS Introduction to Program Logic.

## USER DATA SETS

User data sets comprise those data sets that form the CICS/VS user data base. They may also include Extrapartition Transient Data data sets, Terminal Control sequential data sets, Data Language/I data sets, and journal data sets. Information about these data sets is given in the CICS/VS System Programmer's Guide (OS/VS) and in the CICS/VS Introduction to Program Logic.

## OPERATIONAL CONSIDERATIONS

CICS/VS is generated, initialized, and operated as outlined in the following paragraphs.

## SYSTEM GENERATION

The CICS/VS generation process provides the user with the means of setting up the specific CICS/VS management and service programs that will satisfy his requirements. CICS/VS generation procedures are given in the CICS/VS System Programmer's Guide (OS/VS) and the necessary macro instructions are described in detail in the CICS/VS System Programmer's Reference Manual.

## PREPARATION OF CICS/VS TABLES

CICS/VS is dependent on the user-created CICS/VS tables, which describe the user's data-base/data-communication environment and the treatment given to elements of that environment. Information regarding the user's terminals, data sets (permanent and temporary), programs, and transactions is contained in these tables. The CICS/VS System Programmer's Reference Manual contains full details of the macro instructions needed to generate the tables. The CICS/VS tables are as follows:

- Program Control Table (PCT)
- Processing Program Table (PPT)
- Terminal Control Table (TCT)
- System Recovery Table (SRT)
- File Control Table (FCT)
- Destination Control Table (DCT)
- Journal Control Table (JCT)

The system service tables include:

- Sign-On Table (SNT)
- Terminal List Tables (TLTs)
- Program List Tables (PLTs)
- Transaction List Tables (XLTs)

The contents of these tables are listed in Section 2 of the CICS/OS/VS Program Logic: Data Areas. Their use is summarized in the CICS/VS General Information.

## CICS/VS INITIALIZATION

CICS/VS may be dynamically configured under user control during CICS/VS initialization. The user selects (by using specified suffixes) the CICS/VS modules and tables desired for the current session. The System Initialization program is responsible for preparing CICS/VS for communication and inquiry by user-written application programs. It builds the CICS/VS nucleus, initializes data sets, opens system and user data sets, constructs and initializes tables, and builds the CICS/VS dynamic storage pool. This program is invoked as an OS/VS job by user-provided job control statements. (Details of the System Initialization program are given in section 3 of this manual).

## CICS/VS PROCESSING

To achieve its objective of providing fast response to terminal users, CICS/VS uses its own multitasking within one region of main storage. Such multitasking within one region is analogous to multiprogramming within the total OS/VS environment. Generally, tasks are initiated as a result of transactions entered at terminals. Whenever one task is forced to wait for completion of an I/O operation, availability of a resource, or some other reason, processing of another task within CICS/VS is initiated or continued. The interrelationships of, and services performed by, various CICS/VS System Management functions in the processing of a task are shown in Figure 1-2.

## ASYNCHRONOUS TRANSACTION PROCESSING

The optional Asynchronous Transaction Processing (ATP) facility allows transactions and the data associated with those transactions to be batched for asynchronous (concurrent) processing. This means that the transactions within a batch are not processed until the entire batch has been entered into CICS/VS. Then, while the batched transactions are being processed, the terminal operator may proceed to enter other transactions. When processing of the batched transactions is completed, the terminal operator may request that the resultant output be sent to the terminal that originated the batch or to some alternate terminal.

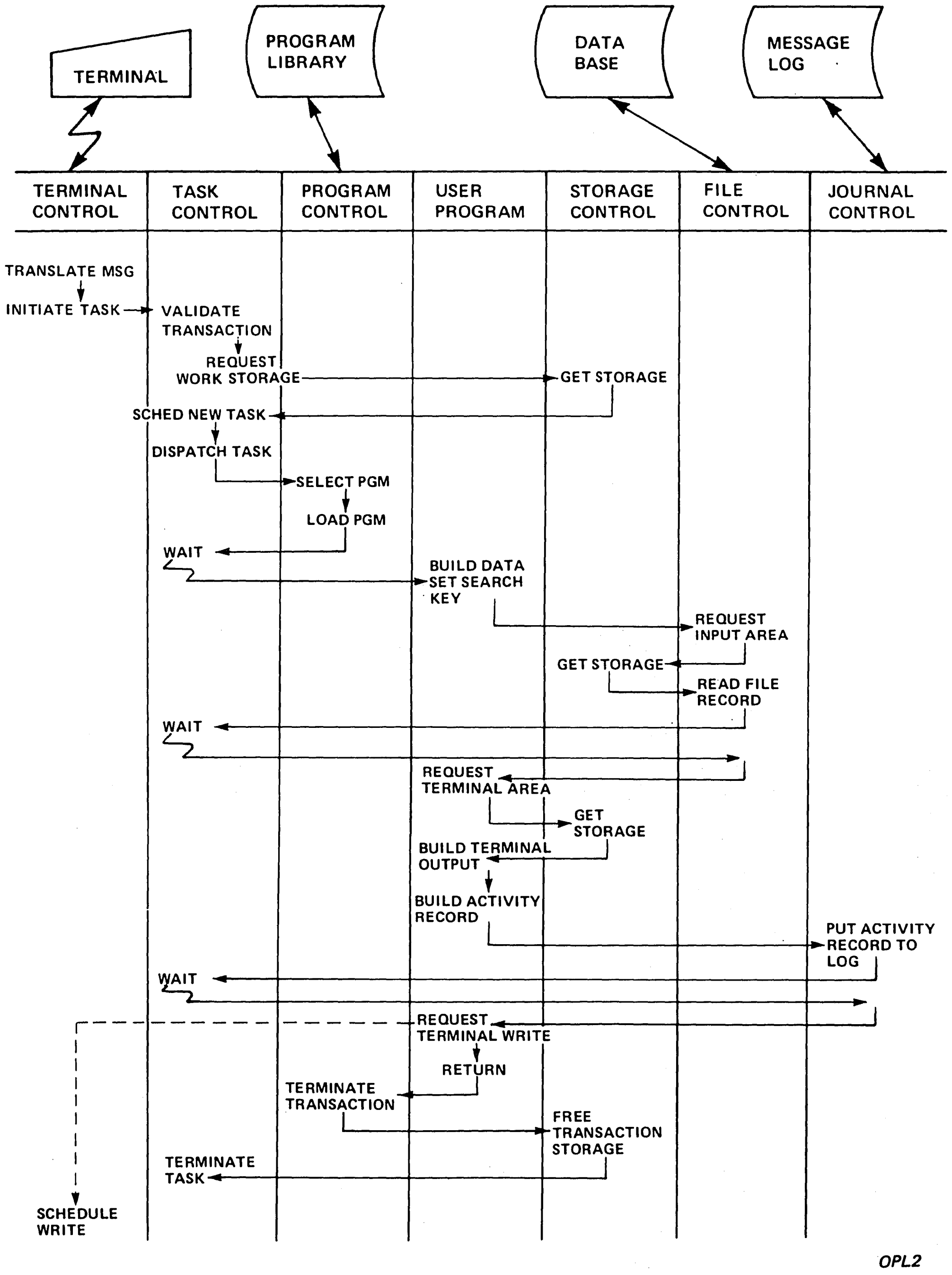


Figure 1-2. CICS/VS Transaction Flow

## MESSAGE SWITCHING

The Message Switching program provides the user with a general-purpose message switching capability during execution of CICS/VS. Messages can be routed to one or more terminals by specifying the transaction code CMSG, the message text, and optional parameters that identify receiving terminals. A message can be delivered immediately, or as soon as all required resources are available, or at a designated future time.

## CONTROL INFORMATION

System statistics are maintained during execution by the various CICS/VS management programs. These statistics can be written to a destination during the day in part or in their entirety on the request of any terminal operator whose security code allows the request of such information. They are gathered automatically when the system is normally terminated.

System statistics are transmitted to a user-specified destination (sequential output device) as variable-length, unblocked records (maximum block size is 136). The name of the destination is the control system system log (CSSL).

In addition, using the automatic statistics function, the user can have system statistics recorded on a data set on a regular (user-specified) interval and have them printed offline in an interval or summary format.

- Number of tasks in system for any period
- Number of tasks initiated
- Number of storage acquisitions
- Number of times storage cushion is released
- Number of times storage request is queued
- Number of times storage queue is established
- Maximum number of requests in storage queue
- Number of times a program is used
- Number of READ requests per data set
- Number of WRITE update requests per data set
- Number of WRITE add requests per data set
- Number of deletes from VSAM data set
- VSAM shared resources statistics
- Number of WRITE requests (per data set) to extrapartition data set
- Number of WRITE requests (per data set) to intrapartition data set

- Number of input messages per terminal
- Number of output messages per terminal
- Number of transmission errors per terminal
- Number of transactions
- Number of transaction errors
- Number of polls issued per line
- Maximum number of Temporary Storage Control Records used (for auxiliary storage)
- Maximum number of PUT requests to temporary storage, main storage, and auxiliary storage
- Maximum number of Temporary Storage unit tables used
- Maximum number of Temporary Storage group identifications (TSGIDs) used
- Temporary Storage main storage requirements
- Dynamic Transaction Backout statistics
- Transaction Restart statistics
- Number of records written per journal identification

These statistics are useful in determining proper data set allocation, identifying infrequently used programs that should not remain permanently resident, determining the activity from each terminal, and, in general, ensuring that the resources of the system are used effectively.

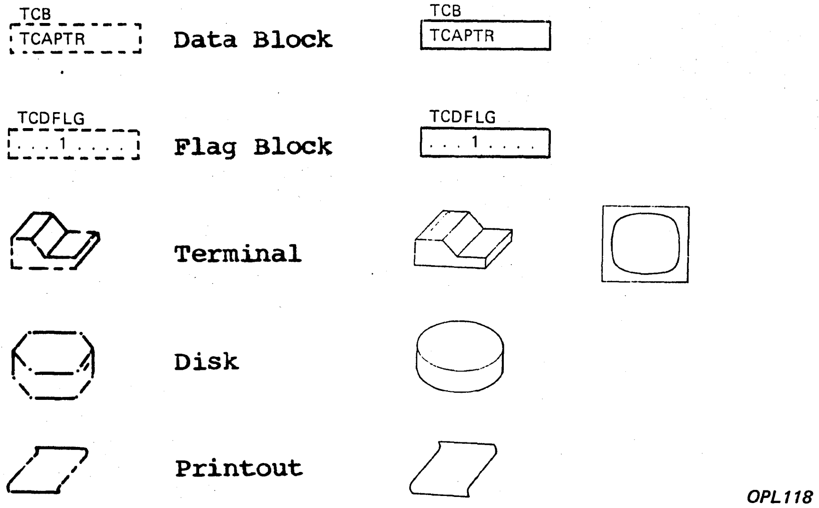
Although journaling for the File Control Program is performed automatically, the creation of an audit trail is largely the user's responsibility. Through the use of CICS/VS Journal Management facilities, the user can create and journal transaction data to sequential data sets. These data sets are called journal data sets. The user describes each journal data set when defining his system. During CICS/VS execution, an application program issues CICS/VS macro instructions that cause the Journal Control program to store transaction data in that data set when the transaction is completed. The same data set can also be used for other control and summary data.

If DL/I data bases are used, the DL/I logging function can be used to produce on the CICS/VS System Log data set (a unique journal data set), or on the DL/I System Log, which is not under the control of CICS/VS, an audit trail of changes made to DL/I data bases. The CICS/VS-DL/I Interface routine logs a scheduling or termination record to this data set when a transaction that alters a DL/I data base is scheduled or terminated. DL/I provides utilities that produce reports from the CICS/VS System Log.

CICS/VS is coded largely according to functional organization; specific modules perform specific functions. In this section, CICS/VS is divided into major functional areas. Within each area, information is divided into functions.

The information in this section is presented in the form of Hierarchy Input Process Output (HIPO) diagrams. Figure 2-1 shows the structure of a HIPO diagram. The HIPO diagrams contain three blocks of information. The left-hand block contains the input to the central block, which contains the processing steps involved in the function or subfunction represented by the diagram. The right-hand block contains the output (changes made to registers, control blocks etc.) from the processing steps. The processing steps are numbered and the numbers correspond to notes, if any, on the lower portion of the page. If notes are given, they include references to modules, routines, and labels involved.

The HIPO diagrams use a number of special symbols to convey information, and arrows to represent the flow through the diagrams. The special symbols are:



Arrows are used for either of two reasons:

- To connect blocks and shapes in the input and output areas to processing steps to which they are related
- To point from one processing step to a processing step other than that which follows it sequentially. The processing step pointed to can be on the same page, on a different page, or on a different diagram.

The six kinds of arrows are:

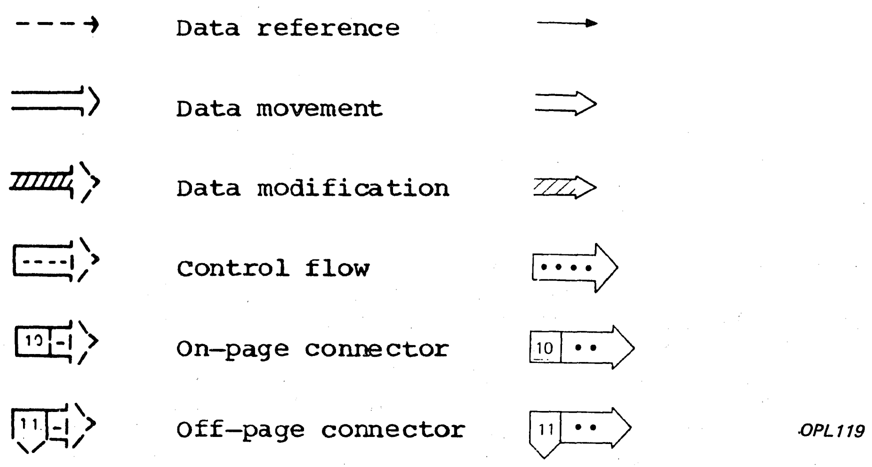
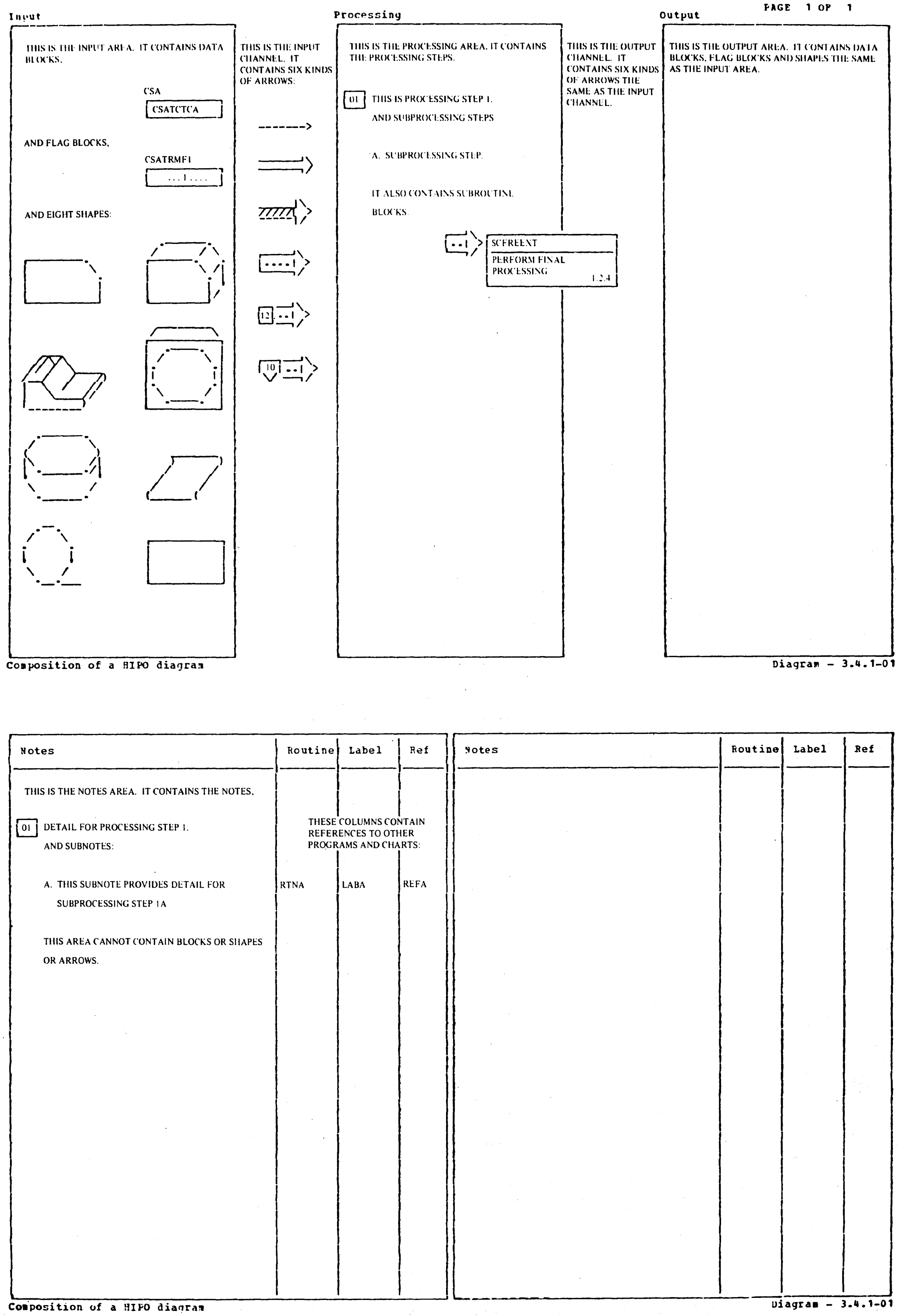


Figure 2–2 is a visual table of contents for the HIPO diagrams. The diagrams are numbered in a sequence that follows the pattern C.F.f where C is a major CICS/VS component, F is a functional service area, and f is a specific function. For example, diagram 1.1.5 is for component 1, System Management; functional area 1.1, Task Management; and function 1.1.5, Change Priority of a Task. Some functions are further subdivided as represented by the pattern C.F.f.s where s is a subfunction. The pages within each diagram are numbered sequentially beginning with 1.

More detailed visual tables of contents are provided at the C.F level for all functional service areas consisting of more than three diagrams. For example, diagram 1.1 is a visual table of contents for the Task Management functional service area.



Input

Processing

Output

PAGE 1 OF 1

THIS IS THE INPUT AREA. IT CONTAINS DATA BLOCKS.

CSA

CSATCTCA

AND FLAG BLOCKS,

CSATRMFI

...1....

AND EIGHT SHAPES:

THIS IS THE INPUT CHANNEL. IT CONTAINS SIX KINDS OF ARROWS:

THIS IS THE PROCESSING AREA. IT CONTAINS THE PROCESSING STEPS.

01 THIS IS PROCESSING STEP 1.

AND SUBPROCESSING STEPS

A. SUBPROCESSING STEP.

IT ALSO CONTAINS SUBROUTINE BLOCKS.

SCFREENT

PERFORM FINAL PROCESSING

1.2.4

THIS IS THE OUTPUT CHANNEL. IT CONTAINS SIX KINDS OF ARROWS THE SAME AS THE INPUT CHANNEL.

THIS IS THE OUTPUT AREA. IT CONTAINS DATA BLOCKS, FLAG BLOCKS AND SHAPES THE SAME AS THE INPUT AREA.

Composition of a HIPO diagram

Diagram - 3.4.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| THIS IS THE NOTES AREA. IT CONTAINS THE NOTES. | | | | | | | |
| 01 DETAIL FOR PROCESSING STEP 1.<br>AND SUBNOTES: | THESE COLUMNS CONTAIN REFERENCES TO OTHER PROGRAMS AND CHARTS: | | | | | | |
| A. THIS SUBNOTE PROVIDES DETAIL FOR SUBPROCESSING STEP 1A | RTNA | LABA | REFA | | | | |
| THIS AREA CANNOT CONTAIN BLOCKS OR SHAPES OR ARROWS. | | | | | | | |

Composition of a HIPO diagram

Diagram - 3.4.1-01

OPL4

Figure 2-1. Composition of a HIPO Diagram

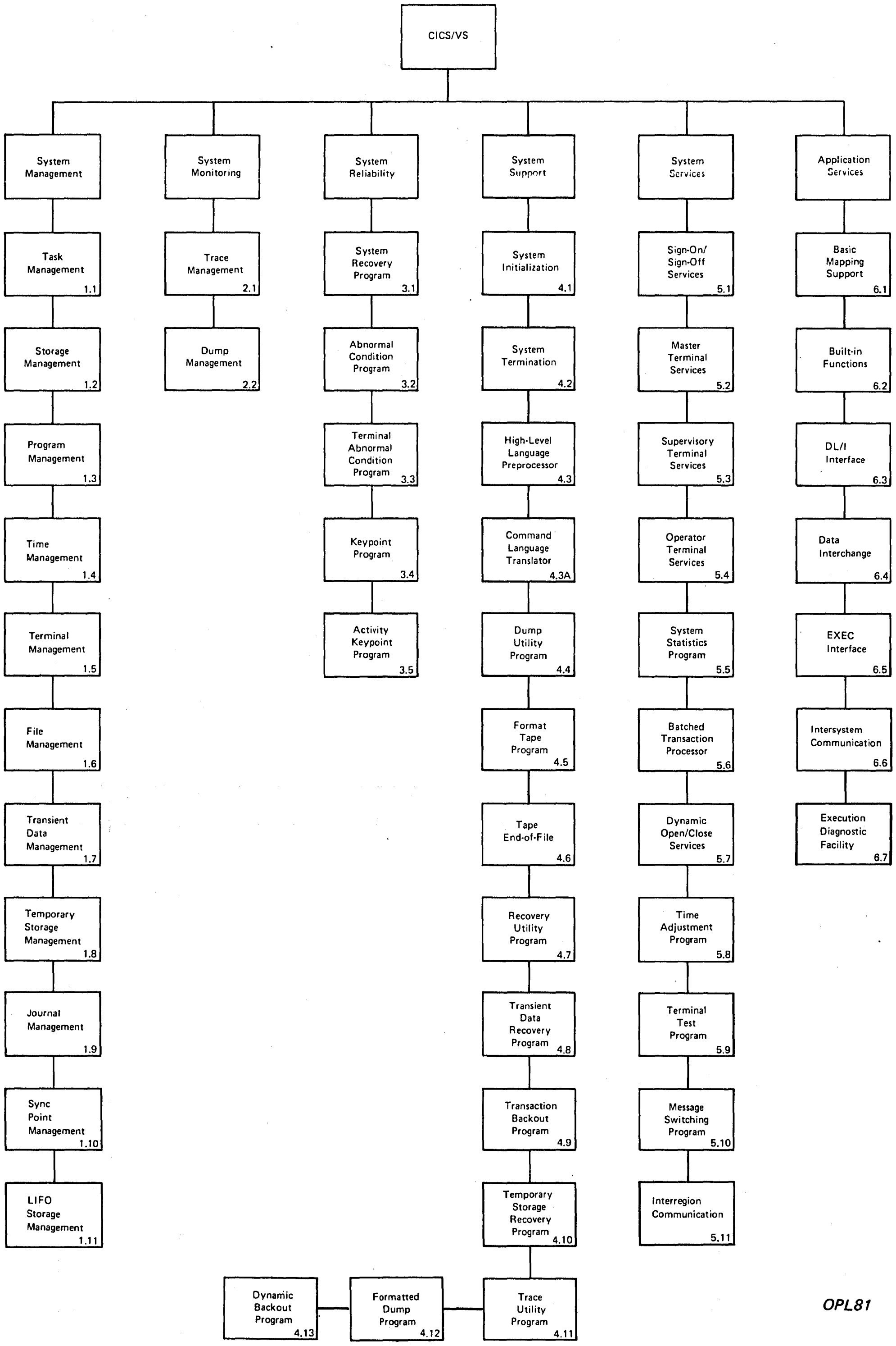

CICS/VS
System Management
Task Management 1.1
Storage Management 1.2
Program Management 1.3
Time Management 1.4
Terminal Management 1.5
File Management 1.6
Transient Data Management 1.7
Temporary Storage Management 1.8
Journal Management 1.9
Sync Point Management 1.10
LIFO Storage Management 1.11
System Monitoring
Trace Management 2.1
Dump Management 2.2
System Reliability
System Recovery Program 3.1
Abnormal Condition Program 3.2
Terminal Abnormal Condition Program 3.3
Keypoint Program 3.4
Activity Keypoint Program 3.5
System Support
System Initialization 4.1
System Termination 4.2
High-Level Language Preprocessor 4.3
Command Language Translator 4.3A
Dump Utility Program 4.4
Format Tape Program 4.5
Tape End-of-File 4.6
Recovery Utility Program 4.7
Transient Data Recovery Program 4.8
Transaction Backout Program 4.9
Temporary Storage Recovery Program 4.10
Trace Utility Program 4.11
Formatted Dump Program 4.12
Dynamic Backout Program 4.13
System Services
Sign-On/ Sign-Off Services 5.1
Master Terminal Services 5.2
Supervisory Terminal Services 5.3
Operator Terminal Services 5.4
System Statistics Program 5.5
Batched Transaction Processor 5.6
Dynamic Open/Close Services 5.7
Time Adjustment Program 5.8
Terminal Test Program 5.9
Message Switching Program 5.10
Interregion Communication 5.11
Application Services
Basic Mapping Support 6.1
Built-in Functions 6.2
DL/I Interface 6.3
Data Interchange 6.4
EXEC Interface 6.5
Intersystem Communication 6.6
Execution Diagnostic Facility 6.7
OPL81

Figure 2-2. Visual Table of Contents for CICS/OS/VS HIPO Diagrams

Task Management

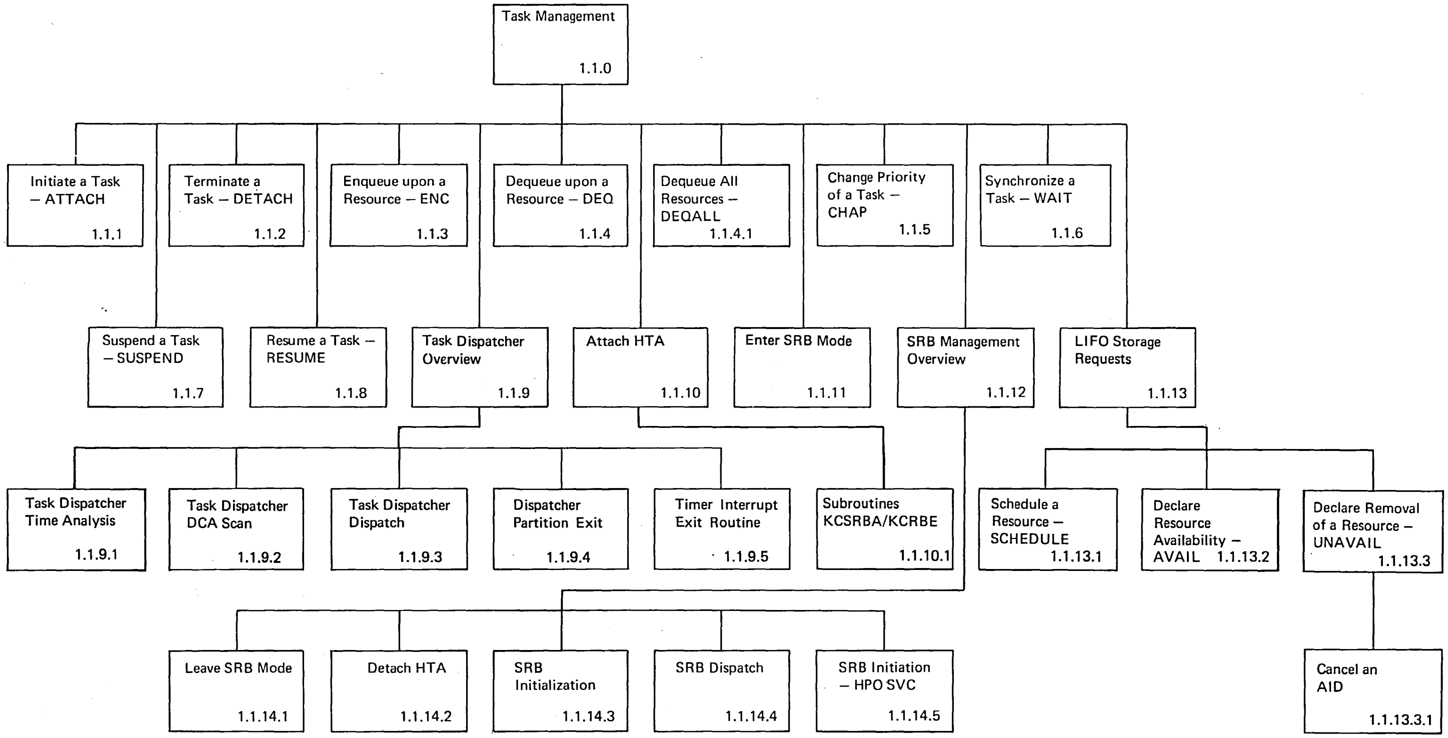
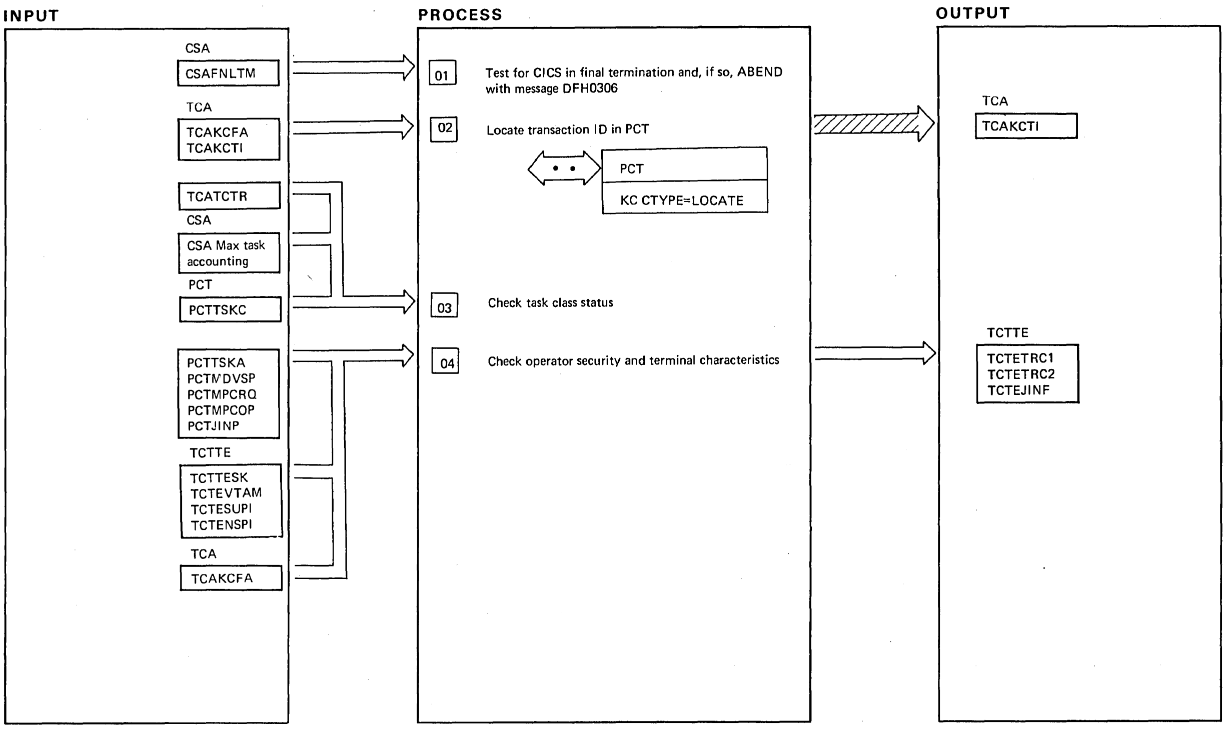

Task Management 1.1.0
Initiate a Task – ATTACH 1.1.1
Terminate a Task – DETACH 1.1.2
Enqueue upon a Resource – ENC 1.1.3
Dequeue upon a Resource – DEQ 1.1.4
Dequeue All Resources – DEQALL 1.1.4.1
Change Priority of a Task – CHAP 1.1.5
Synchronize a Task – WAIT 1.1.6
Suspend a Task – SUSPEND 1.1.7
Resume a Task – RESUME 1.1.8
Task Dispatcher Overview 1.1.9
Attach HTA 1.1.10
Enter SRB Mode 1.1.11
SRB Management Overview 1.1.12
LIFO Storage Requests 1.1.13
Task Dispatcher Time Analysis 1.1.9.1
Task Dispatcher DCA Scan 1.1.9.2
Task Dispatcher Dispatch 1.1.9.3
Dispatcher Partition Exit 1.1.9.4
Timer Interrupt Exit Routine 1.1.9.5
Subroutines KCSRBA/KCRBE 1.1.10.1
Schedule a Resource – SCHEDULE 1.1.13.1
Declare Resource Availability – AVAIL 1.1.13.2
Declare Removal of a Resource – UNAVAIL 1.1.13.3
Leave SRB Mode 1.1.14.1
Detach HTA 1.1.14.2
SRB Initialization 1.1.14.3
SRB Dispatch 1.1.14.4
SRB Initiation – HPO SVC 1.1.14.5
Cancel an AID 1.1.13.3.1

Diagram – 1.1-01

Page 1 of 3



01 Test for CICS in final termination and, if so, ABEND with message DFH0306

02 Locate transaction ID in PCT

03 Check task class status

04 Check operator security and terminal characteristics

Attach a Task

Diagram - 1.1.1 -01

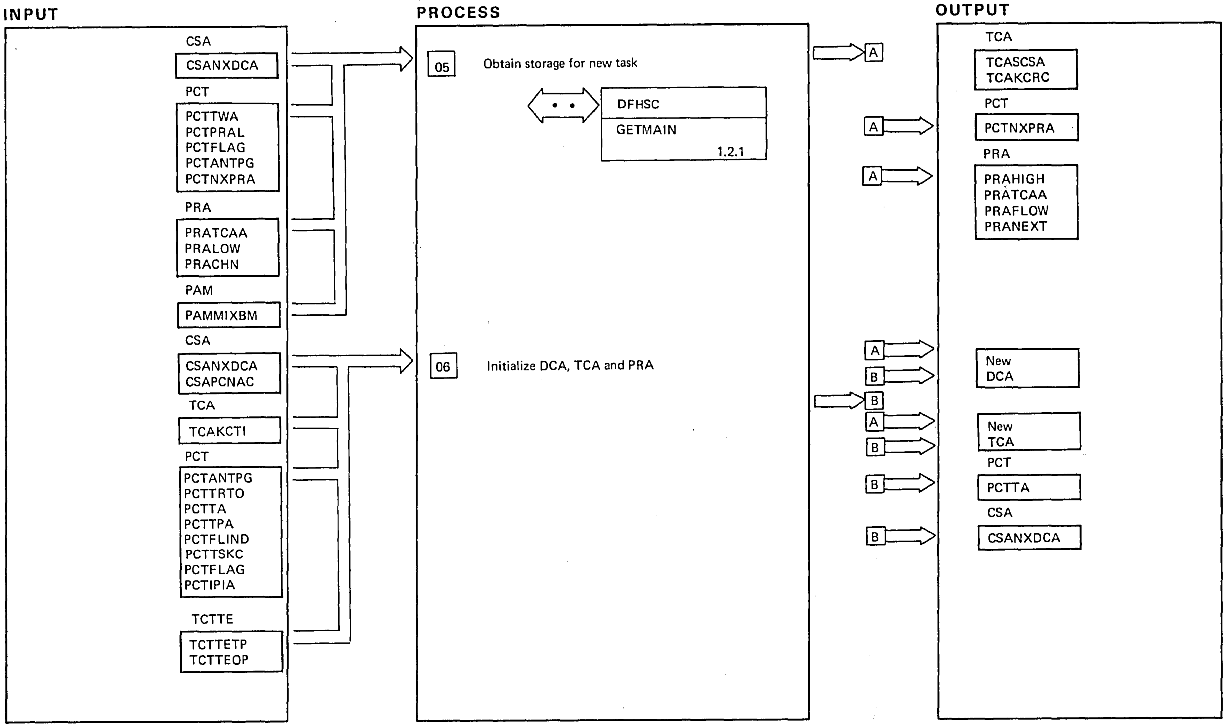
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 If the transaction ID cannot be found in the PCT 'CSAC' is substituted. | | KCTO | 3.2.2 |
| 03 If the ATTACH was conditional and the task is a member of a task class, then, if the number already attached in that class is greater than or equal to the limit the ATTACH is terminated. A 'negative response' code is set in TCAKCRC. | | KCTO10 | |
| 04 If a terminal is to be associated with the attached task (X'01' in first byte of TCAKCFA of caller) security and terminal characteristics checks are made. If any check fails 'CSAC' is substituted as transaction ID. | | KCTO20 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Attach a Task

Diagram - 1.1.1-01

Page 2 of 3



Attach a Task

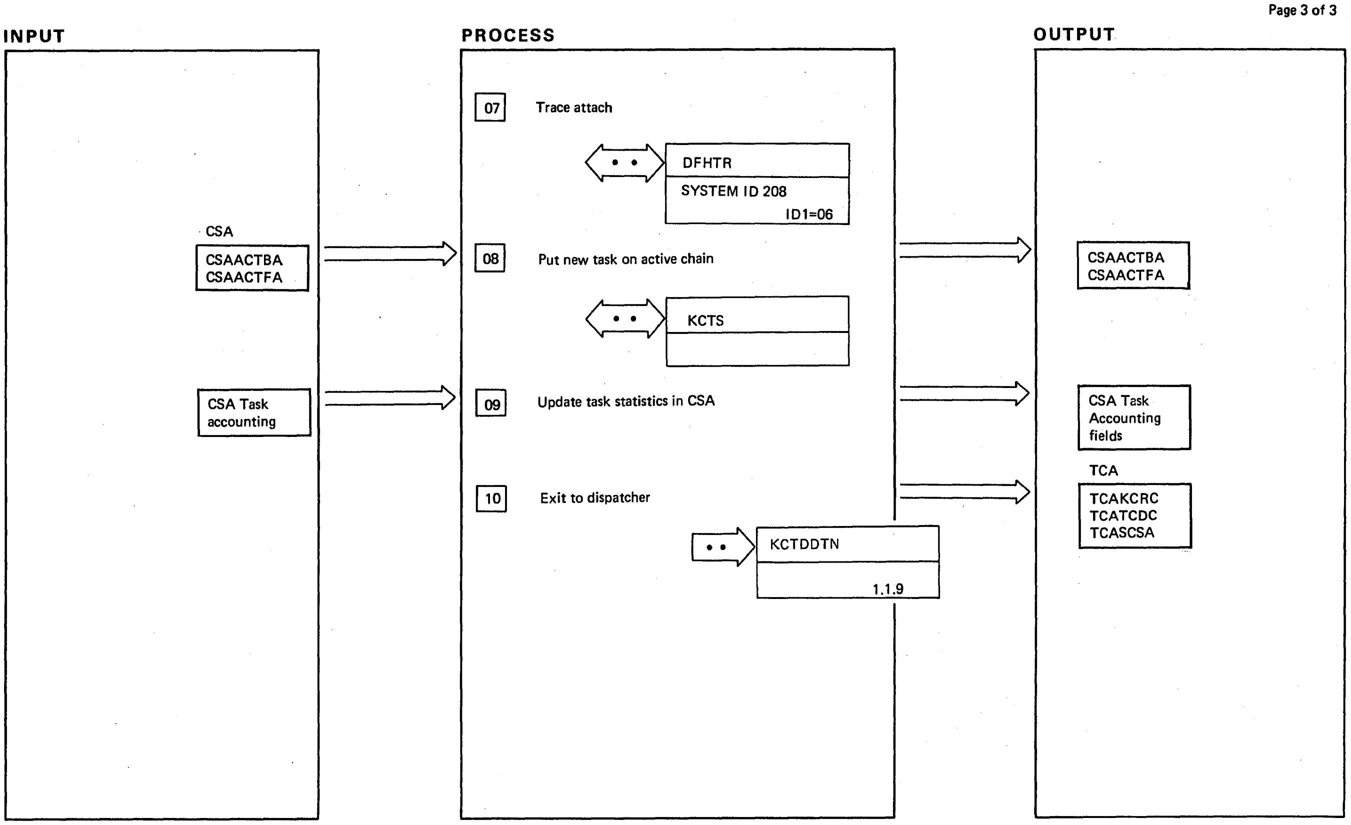
Diagram - 1.1.1 -02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 05 The DCA is acquired by GETMAIN or from the DCA quick-cell chain. The TCA is acquired by GETMAIN or, if the task is primed, is imbedded in a PRA which is in turn acquired either by GETMAIN, or from the PRA chain addressed by PCTNXPRA. If any GETMAIN fails then the attaching task is made non-dispatchable or if the ATTACH was conditional, the ATTACH is terminated with a negative response code | | KCT030 | |
| 06 The KC register save area in the new TCA is initialized so that, on first dispatch, the task will appear to be about to execute a PC TYPE=XCTL to the initial module of the task. The dispatch code is initially set to 'max task' or, if the task is in a class, 'max task in class' | | KCT060 | |

Attach a Task

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Diagram - 1.1.1 -02



Attach a Task

Diagram — 1.1.1-03

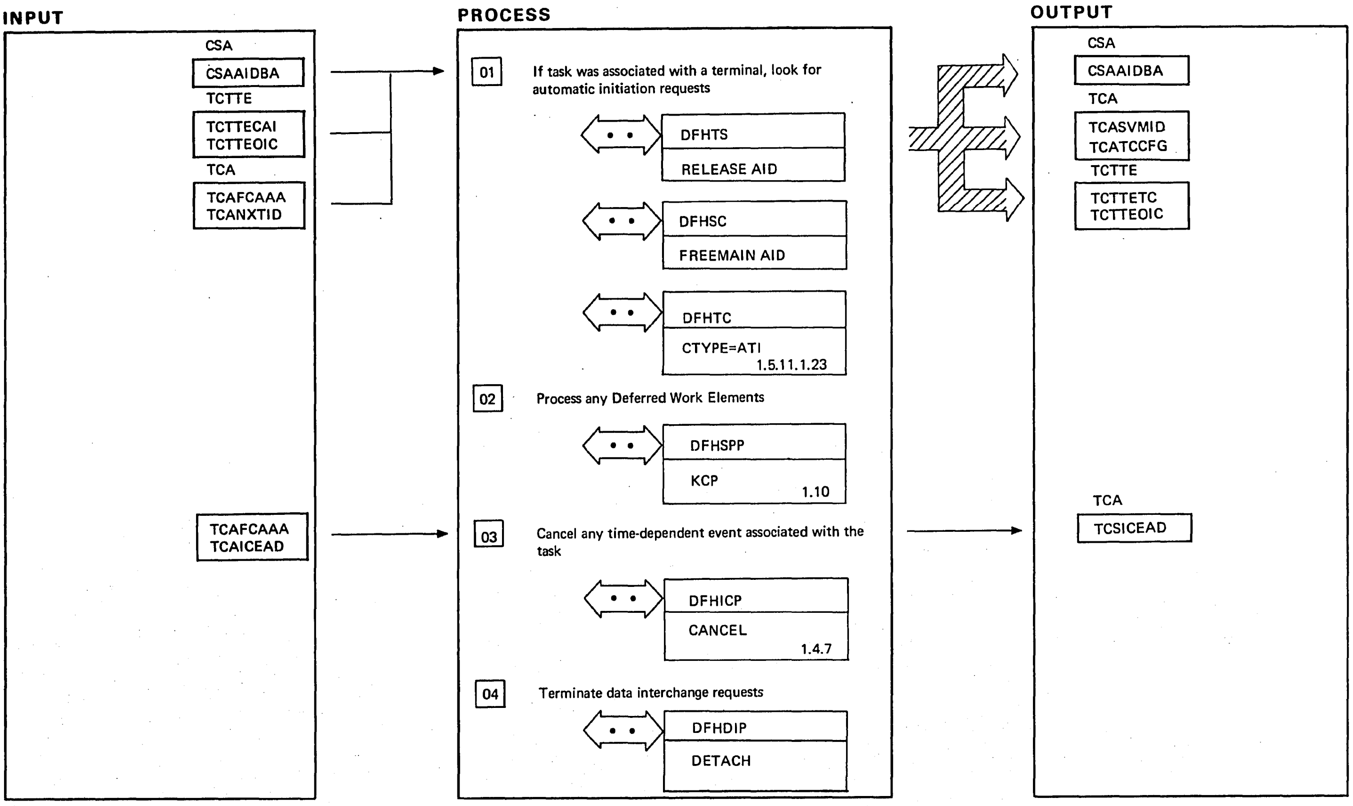
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 10 Normal exit is to KCTDDTN to redispatch the attaching task. If a GETMAIN fails on an unconditional ATTACH then exit is to KCTD00 to make attaching task non-dispatchable, save GETMAIN request, and do full dispatch. |  | 1.1.1 | 1.1.9 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
|  |  |  |  |

Attach a Task

Diagram — 1.1.1-03

Page 1 of 3



Detach a Task

Diagram - 1.1.2-01

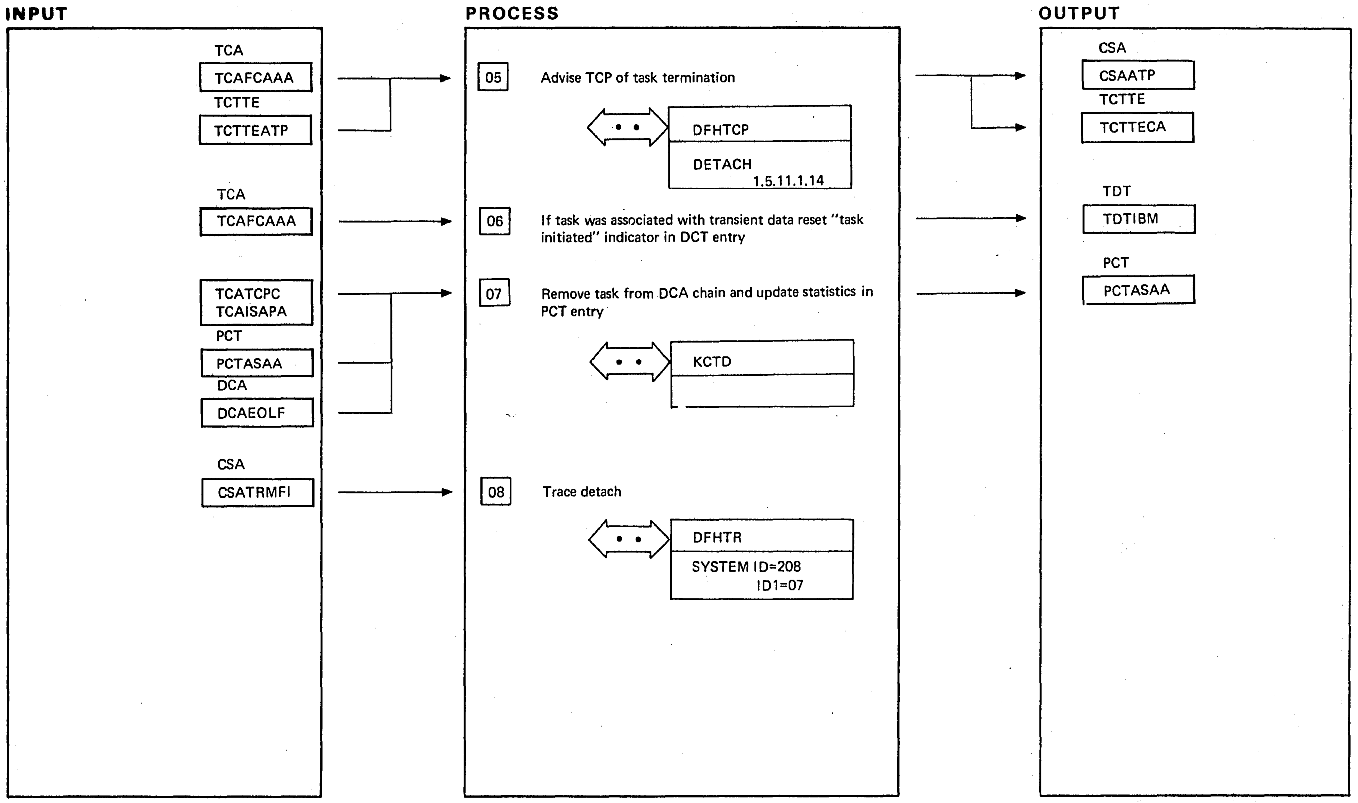
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The DETACH macro instruction is issued only by DFHPCP. The following terminal-dependent task clean-up functions are performed: | | | |
| Set next task's transaction ID, if given | | | |
| Scan AID chain to validate current setting of the automatic initiate indicator | | KCTTNIDB | |
| Go to TCP if another task is to be automatically initiated | | KCTTSETI | |
| 02 DFHSPP is entered to pı ocess any work that was deferred until task termination | | KCTTASBY | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Detach a Task

Diagram - 1.1.2-01

Page 2 of 3



05 Advise TCP of task termination

06 If task was associated with transient data reset "task initiated" indicator in DCT entry

07 Remove task from DCA chain and update statistics in PCT entry

08 Trace detach

Detach a Task

Diagram - 1.1.2-02

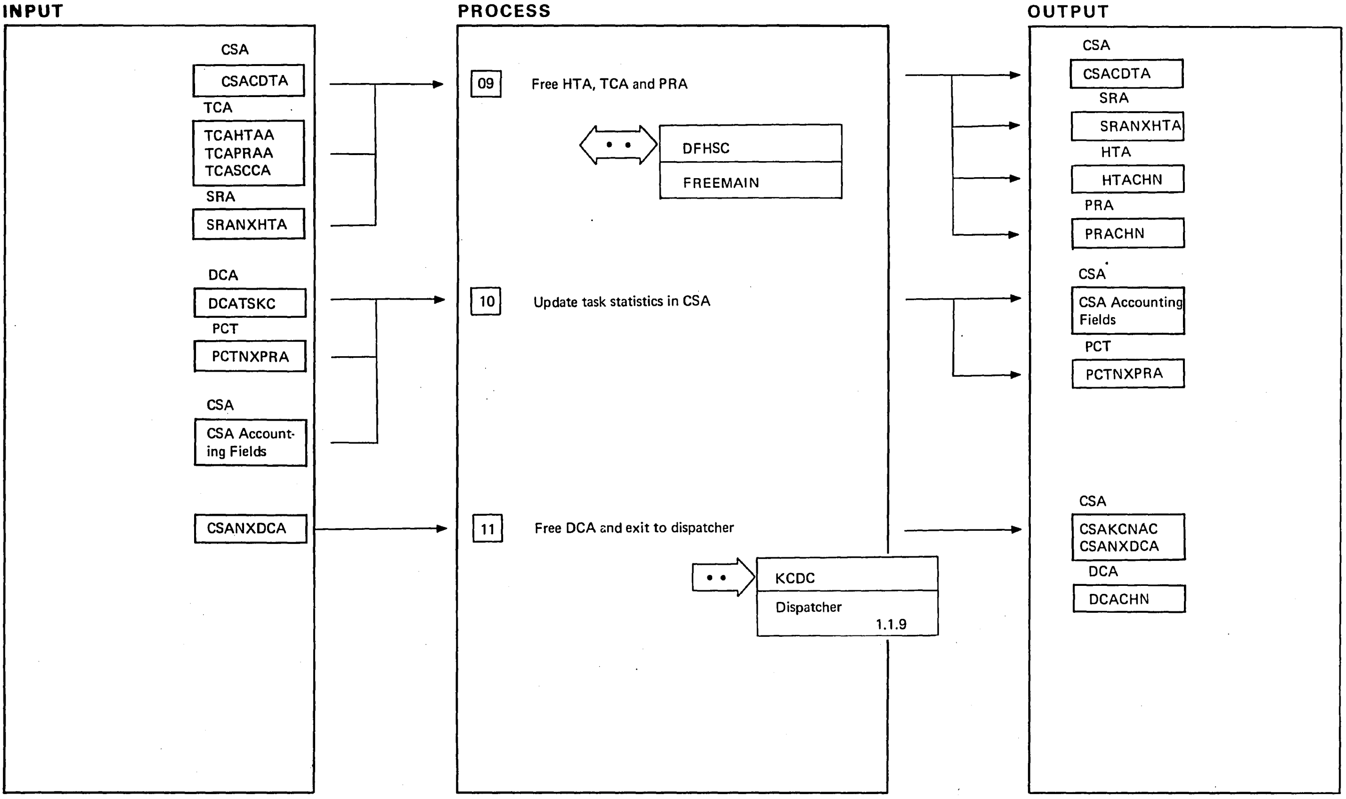
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 05 Clear task pointer for terminal. Post ATP, if applicable | | KCTTCNLB | |
| Advise TCP if a terminal dependent task terminating | | KCTITF | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Detach a Task

Diagram - 1.1.2-02

Page 3 of 3



Detach a Task

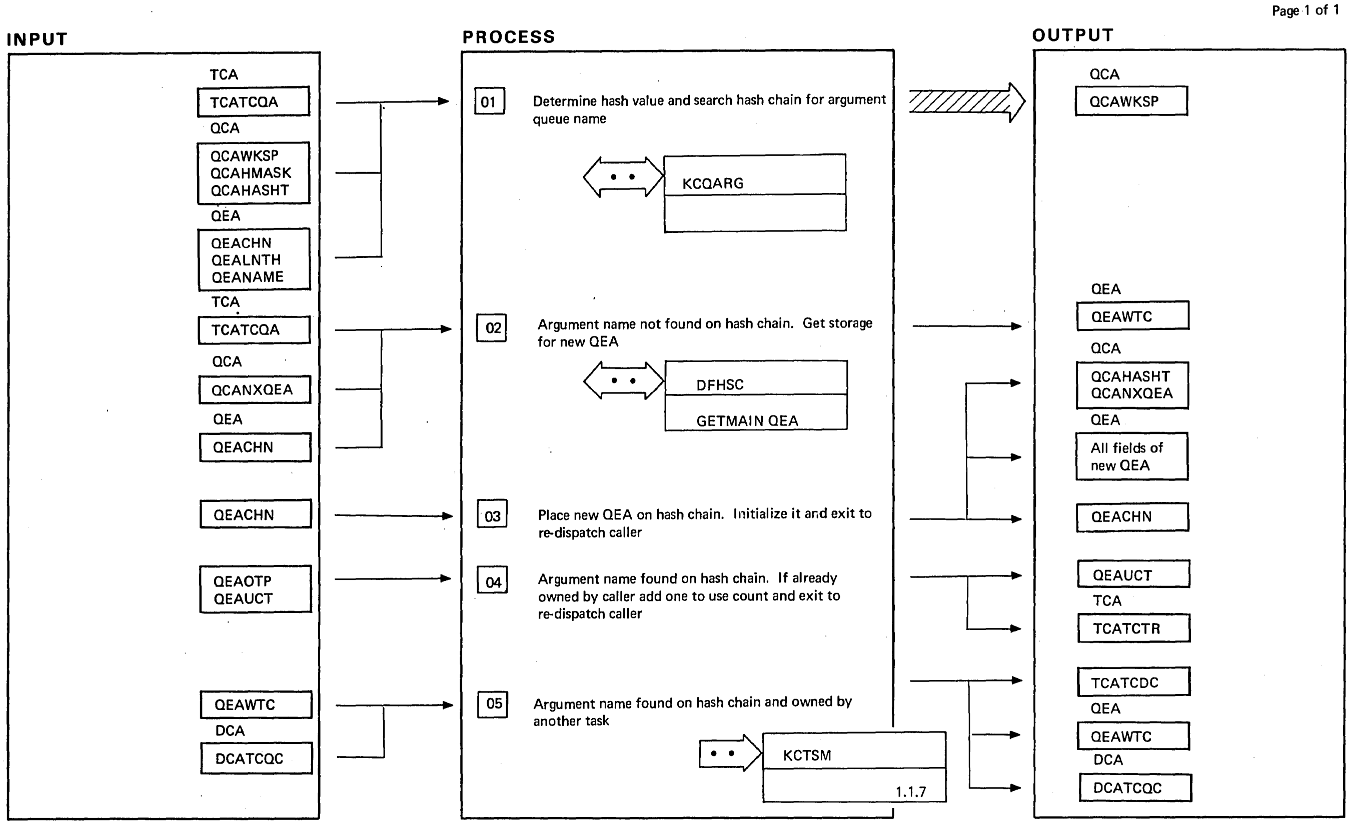
Diagram - 1.1.2-03

| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 09 | If the task has an HTA (HPO only) return the HTA to the quick call chain SRANXHTA. | | KCPNOTRM | |
| | If the task was not a primed storage task, call DFHSCP FREEMAIN TCA to free the TCA and any other storage owned by the task | | | |
| | If the task was a primed storage task FREEMAIN TCA is called only if there are outstanding allocations outside the primed area (TCASCCA≠TCA address). In this case, if FREEMAIN is called, it does not free the TCA. The PRA is placed on the free chain PCTNXPRA. | | KCTT70 | |
| 11 | The DCA is returned to the quick-cell chain CSANXDCA. CSAKCNAC is 'posted'. This word is used as a CICS system event by STP when waiting for system to quiesce. | | KCTT90 | 4.2.4.1 |
| | KCPMAXTR is increased by 1. This is an internal count of the number of tasks which can be activated without exceeding the 'active task' limit CSAMAXT. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Detach a Task

Diagram - 1.1.2-03


Page 1 of 1
INPUT
PROCESS
OUTPUT
TCA
TCATCQA
QCA
QCAWKSP
QCAHMASK
QCAHASHT
QEA
QEACHN
QEALNTH
QEANAME
TCA
TCATCQA
QCA
QCANXQEA
QEA
QEACHN
QEACHN
QEAOTP
QEAUCT
QEAWTC
DCA
DCATCQC
01 Determine hash value and search hash chain for argument queue name
KCQARG
02 Argument name not found on hash chain. Get storage for new QEA
DFHSC
GETMAIN QEA
03 Place new QEA on hash chain. Initialize it and exit to re-dispatch caller
04 Argument name found on hash chain. If already owned by caller add one to use count and exit to re-dispatch caller
05 Argument name found on hash chain and owned by another task
KCTSM
1.1.7
QCA
QCAWKSP
QEA
QEAWTC
QCA
QCAHASHT
QCANXQEA
QEA
All fields of new QEA
QEACHN
QEAUCT
TCA
TCATCTR
TCATCDC
QEA
QEAWTC
DCA
DCATCQC

Enqueue upon a Resource

Diagram - 1.1.3-01

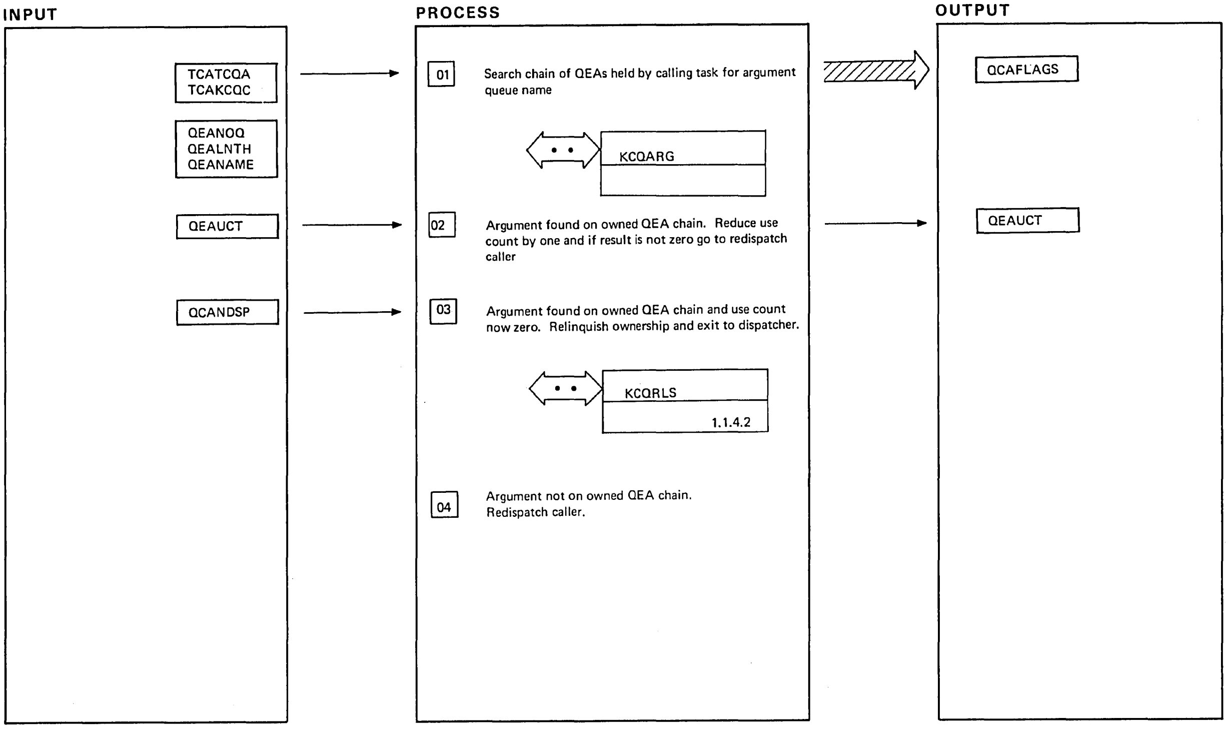
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The argument queue name is hashed to obtain an index to one of 64 QEA chain heads in the QCA. Each hash chain is ordered low to high by full queue name within queue name length. | | KCENQN | |
| 02 If the queue name is such that the QEA will be larger than 64 bytes a DFHSC GETMAIN QEA is issued. Otherwise a standard 64 byte QEA is used, obtained from the quick-cell chain QCANXQEA, or via GETMAIN if the chain is empty. | | KCENQ20 | |
| 03 The new QEA is inserted in the appropriate hash chain at the correct point to preserve ordering. It is also placed at the head of the chain of QEAs held by the task (TCAKCQC). The calling task is then redispatched | KCTDDTN | KCENQ40 | 1.1.9 |
| 04 A 'duplicate request' code is returned in TCATCTR to indicate that the caller was already enqueued on the given queue The caller is redispatched | KCTDDTN | KCENQ50 | 1.1.9 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 05 The caller's DCA is placed on the QEA wait chain. This is a loop through QEAWTC, the DCATCQC fields of the waiting DCAs and back to the QEA. The DCAs are in the order in which the ENQ requests were made. The caller is then suspended until the QEA is released by the present owner and other waiting tasks | KCTSN | KCENQ60 | 1.1.7 |

Enqueue upon a Resource

Diagram - 1.1.3-01

Page 1 of 1



01 Search chain of QEAs held by calling task for argument queue name

02 Argument found on owned QEA chain. Reduce use count by one and if result is not zero go to redispatch caller

03 Argument found on owned QEA chain and use count now zero. Relinquish ownership and exit to dispatcher.

04 Argument not on owned QEA chain. Redispatch caller.

Dequeue from a Resource

Diagram - 1.1.4-01

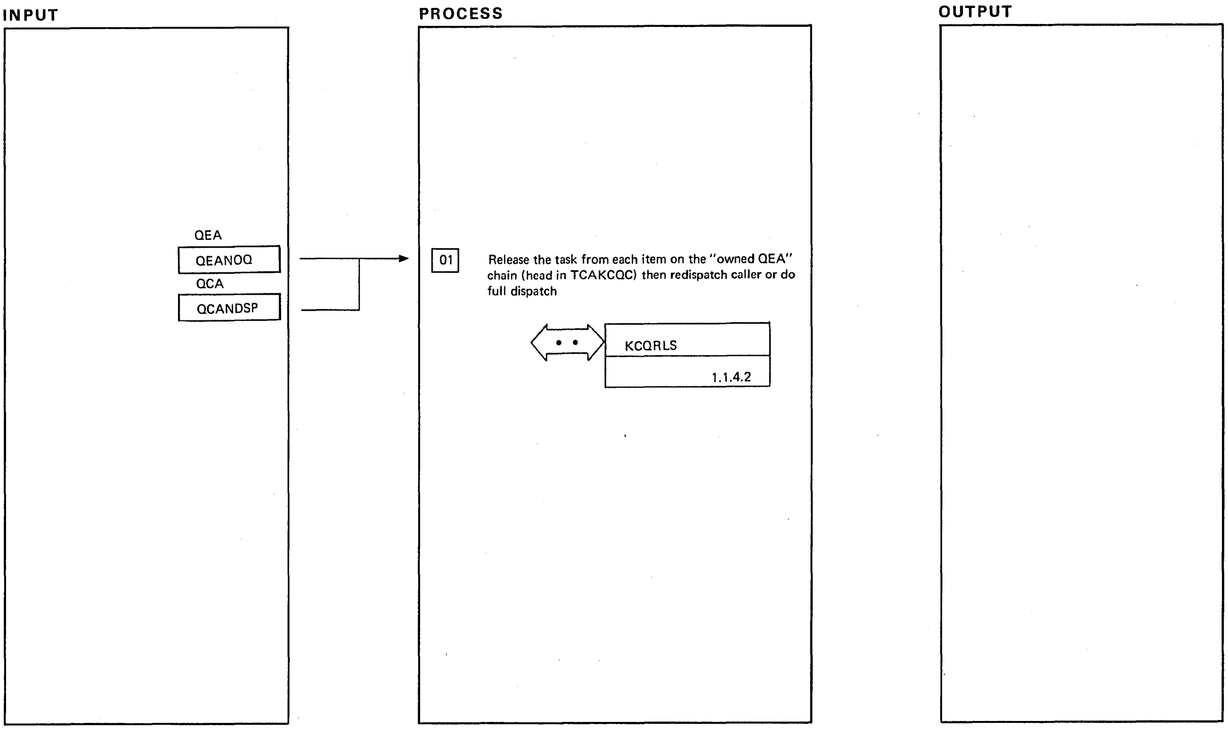
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The 'owned QEA' chain head is TCAKCQC. The chain, which is not ordered ends with a zero chain field. | | KCDEQN | |
| 03 The subroutine KCQRLS removes the QEA from the 'owned QEA' chain and resumes any waiting task. The subroutine KCQRLS sets the flag QCANDSP if any task is waiting. The exit to the dispatcher is to redispatch the caller, or if QCANDSP is set, to perform a full dispatch. | | | 1.1.4.2 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Dequeue from a Resource

Diagram - 1.1.4-01

Page 1 of 1



01 Release the task from each item on the "owned QEA" chain (head in TCAKCQC) then redispatch caller or do full dispatch

Dequeue All Resources

Diagram - 1.1.4.1-01

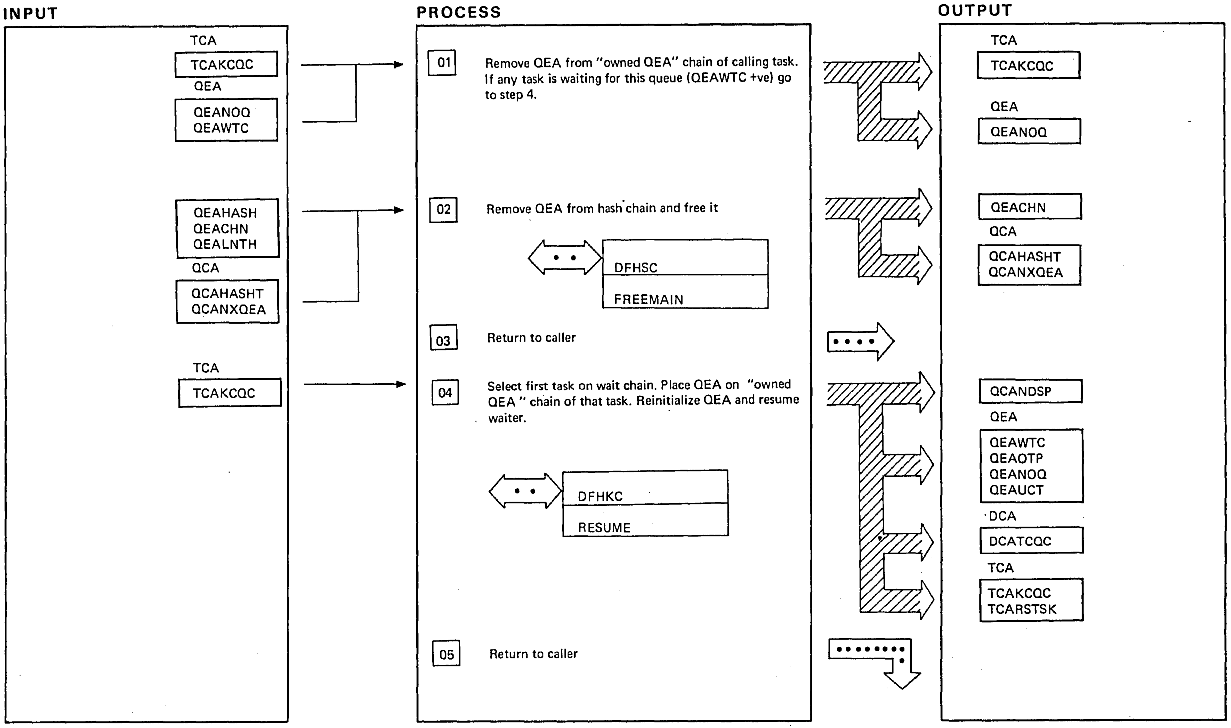
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 If any task was waiting for any of the queues released from the control of the calling task QCANDSP is set on, causing a full dispatch cycle rather than a simple redispatch of the caller. | KCDQALN | KCDQL20 | 1.1.4.2 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Dequeue All Resources

Diagram - 1.1.4.1-01

Page 1 of 1



01 Remove QEA from "owned QEA" chain of calling task. If any task is waiting for this queue (QEAWTC +ve) go to step 4.

02 Remove QEA from hash chain and free it

03 Return to caller

04 Select first task on wait chain. Place QEA on "owned QEA " chain of that task. Reinitialize QEA and resume waiter.

05 Return to caller

Subroutine KCQRLS

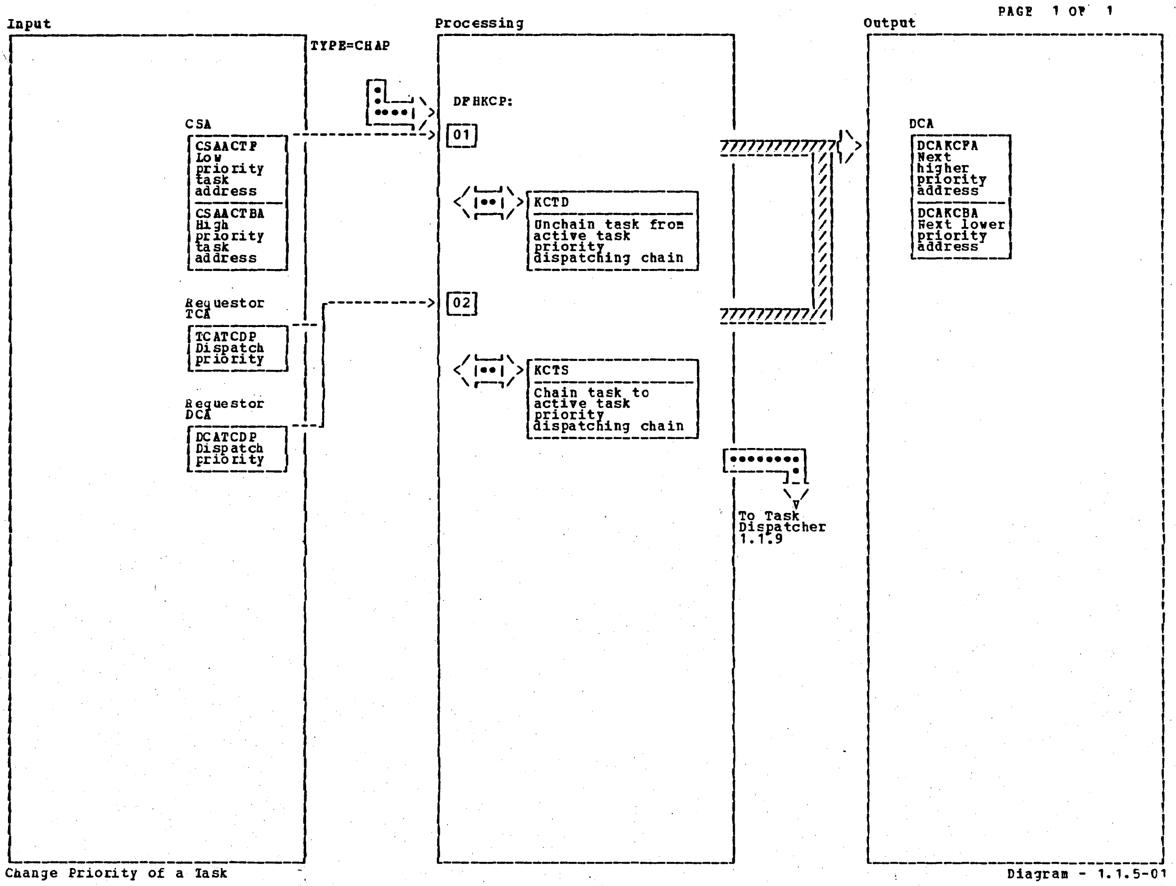
Diagram - 1.1.4.2-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The wait chain is a chain of DCA's in the form of a loop through QEAWTC and DCATCQC. The chain to the QEA is distinguished by a high order bit in the chain field. | | KCQRLS | |
| 02 If the QEA was for a short name the QEA is returned to the quick call chain headed by QCANXQEA. Otherwise it is freed by issuing a FREEMAIN. | | | |
| 04 The flag QCANDSP is set to indicate to the caller that a full dispatch is required since a task which was suspended has now become active again. | | KCQRLS5 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Subroutine KCQRLS

Diagram - 1.1.4.2-01


PAGE 1 OF 1
Input
Processing
Output
TYPE=CHAP
DFHKCP:
CSA
CSAACTF
Low priority task address
CSAACTBA
High priority task address
Requestor TCA
TCATCDP
Dispatch priority
Requestor DCA
DCATCDP
Dispatch priority
01
KCTD
Unchain task from active task priority dispatching chain
02
KCTS
Chain task to active task priority dispatching chain
To Task Dispatcher 1.1.9
DCA
DCAKCFA
Next higher priority address
DCAKCBA
Next lower priority address

Change Priority of a Task

Diagram - 1.1.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | KCTD | KCPC | | | | | |
| 02 | KCTS | | | | | | |
| 03 | KCDCA | | 1.1.9 | | | | |

Change Priority of a Task

Diagram - 1.1.5-01

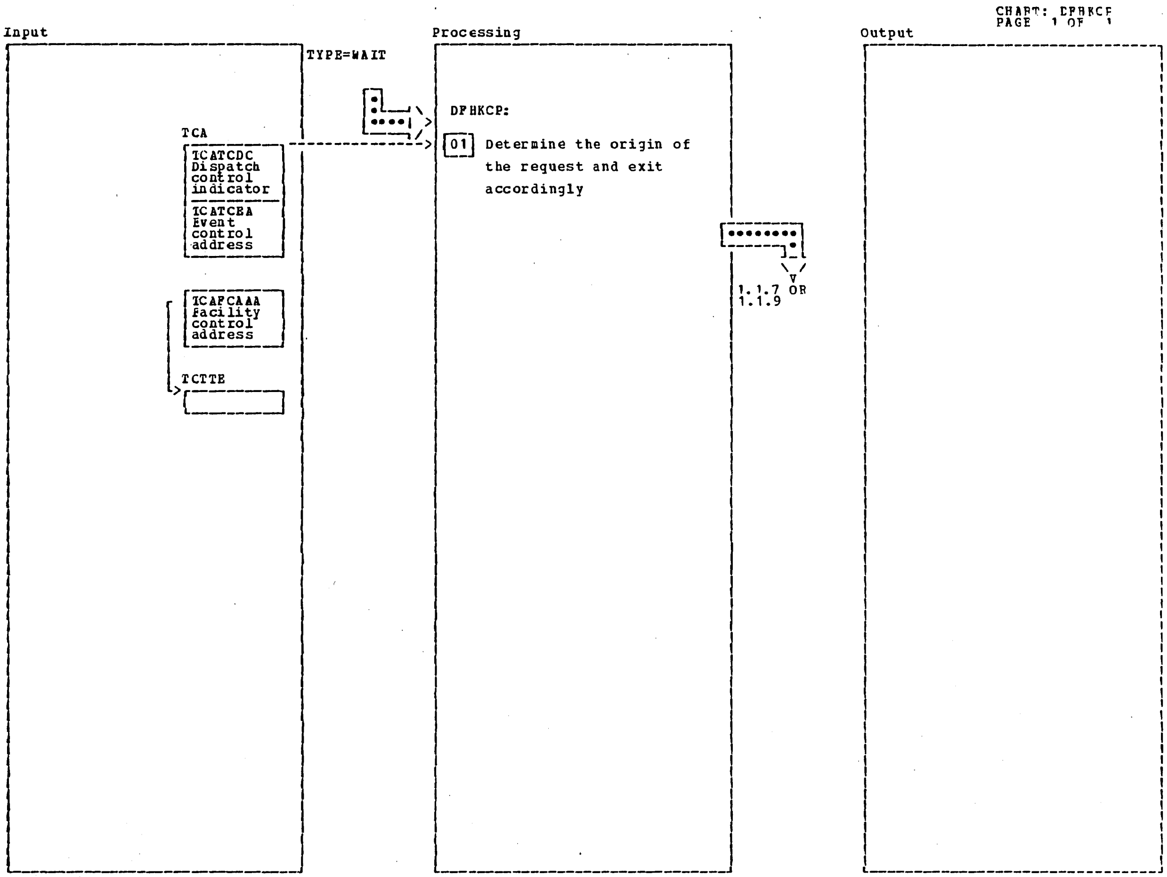


CHART: DFHKCP
PAGE 1 OF 1
Input
Processing
Output
TYPE=WAIT
TCA
TCATCDC Dispatch control indicator
TCATCEA Event control address
TCAFCAAA Facility control address
TCTTE
DFHKCP:
01 Determine the origin of the request and exit accordingly
1.1.7 OR 1.1.9

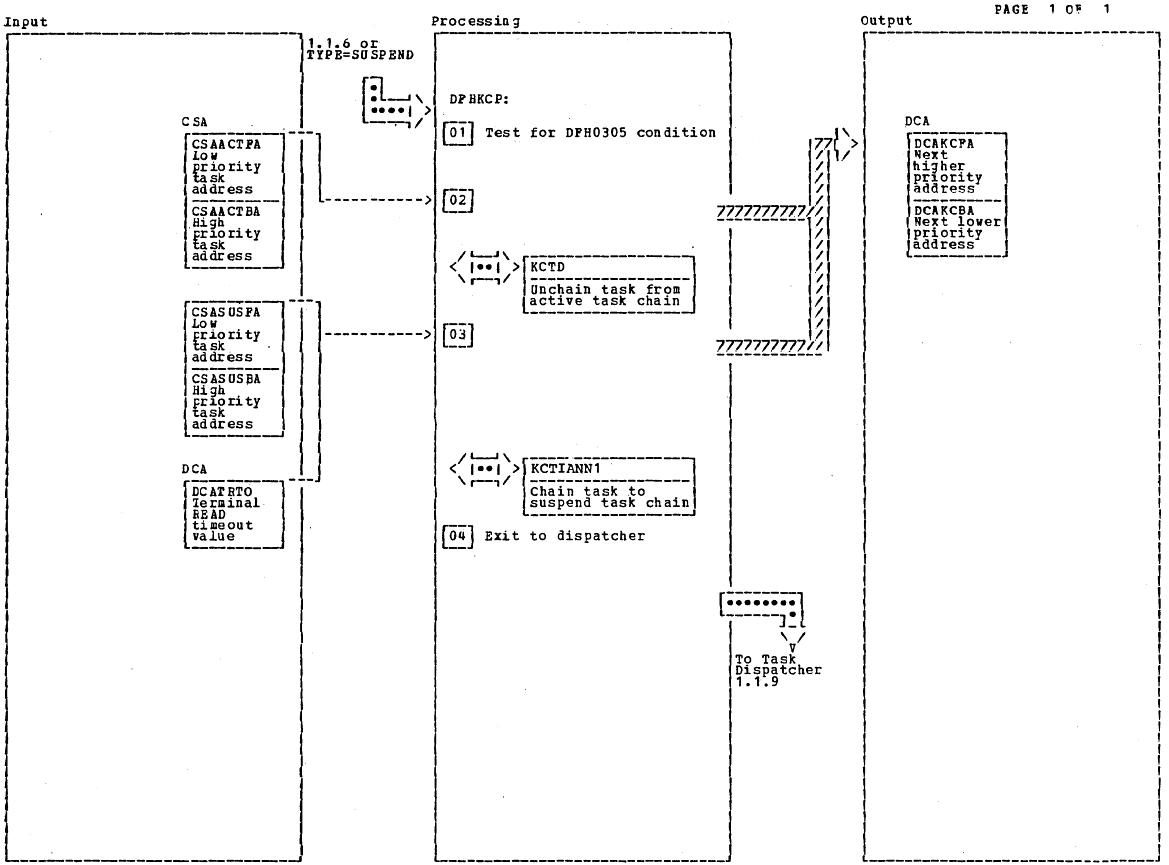
Synchronize a Task

Diagram - 1.1.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 A non-dispatchable WAIT request is assumed to be a Terminal Control WAIT. Further tests are made to distinguish an ATP task from one associated directly with a Terminal. | | KCWAIT | | | | | |
| For ATP tasks, ATP's ECB is posted, and control is transferred to the Task Dispatcher. | KCDCA | | Exit to 1.1.9 | | | | |
| For terminal-associated tasks, control is transferred to suspend the task. | KCTSN | | Exit to 1.1.7 | | | | |
| For tasks that require terminal READ timeout, the timeout value is determined and placed in the DCA (DCATRTO). | | | | | | | |
| For asynchronous paging tasks (anticipatory paging), a pageout will be started for the task's data areas, if running under the MVS or S1.6 variants of the operating system. | | | | | | | |

Synchronize a Task

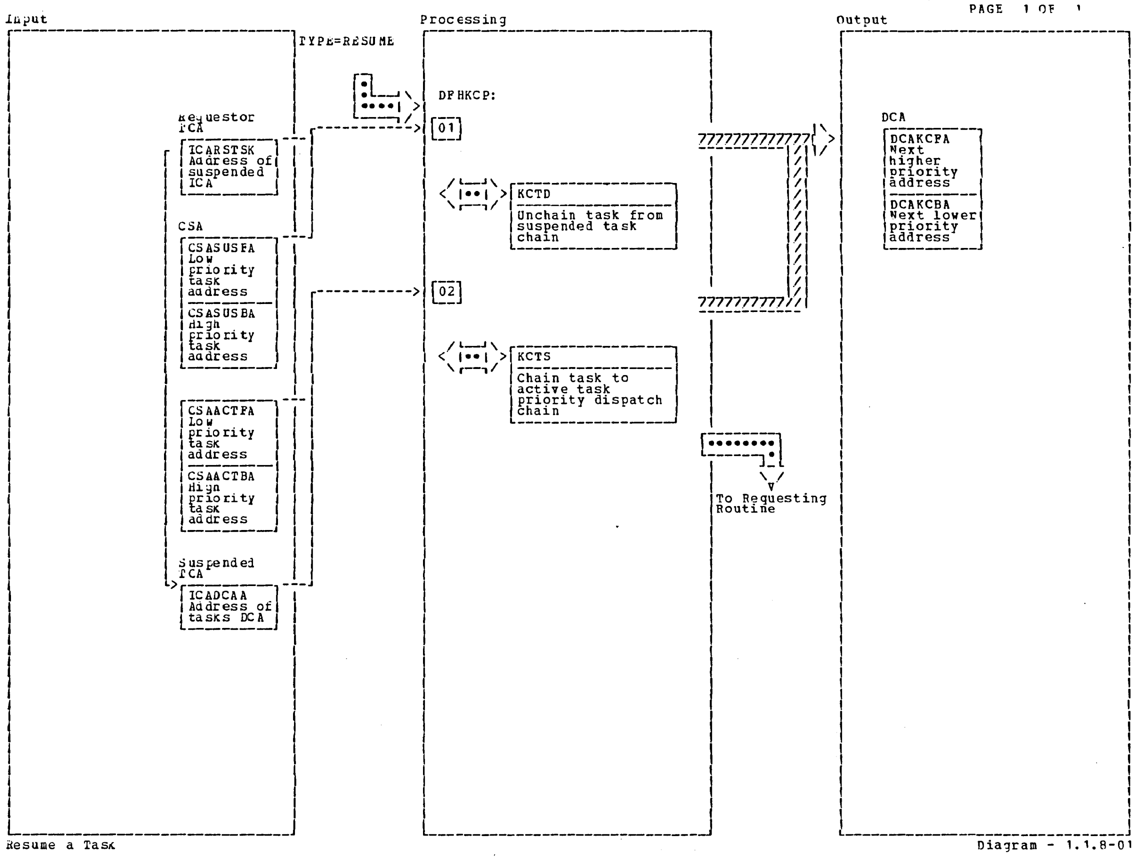
Diagram - 1.1.6-01


PAGE 1 OF 1
Input
Processing
Output
1.1.6 or
TYPE=SUSPEND
CSA
CSAACTPA
Low priority task address
CSAACTBA
High priority task address
CSASUSPA
Low priority task address
CSASUSBA
High priority task address
DCA
DCATRTO
Terminal READ timeout value
DFHKCP:
01 Test for DFH0305 condition
02
KCTD
Unchain task from active task chain
03
KCTIANN1
Chain task to suspend task chain
04 Exit to dispatcher
To Task Dispatcher 1.1.9
DCA
DCAKCPA
Next higher priority address
DCAKCBA
Next lower priority address

Suspend a Task Diagram - 1.1.7-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If the request is to suspend a CICS/VS system task (e.g., Terminal Control) CICS is abnormally terminated. | KCTD | KCTSN | | | | | |
| 02 | KCTIANN1 | | | | | | |
| 03 If the task requests timeout the task is inserted into the suspend chain in timeout value descending sequence. The tasks in the suspend chain are not in priority sequence. | | KCTSTOA | | | | | |
| 04 | KCDCA | | 1.1.9 | | | | |

Suspend a Task Diagram - 1.1.7-01


PAGE 1 OF 1
Input
Processing
Output
TYPE=RESUME
Requestor TCA
TCARSTSK Address of suspended TCA
CSA
CSASUSFA Low priority task address
CSASUSBA High priority task address
CSAACTFA Low priority task address
CSAACTBA High priority task address
Suspended TCA
TCADCAA Address of tasks DCA
DFHKCP:
01
KCTD
Unchain task from suspended task chain
02
KCTS
Chain task to active task priority dispatch chain
To Requesting Routine
DCA
DCAKCFA Next higher priority address
DCAKCBA Next lower priority address

Resume a Task

Diagram - 1.1.8-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The task's registers are saved in the CSA rather than in TCA. No switch occurs during processing. | KCTD | KCTRN | | | | | |
| 02 | KCTS | | | | | | |
| 03 | KCTRTNBY | | | | | | |

Resume a Task

Diagram - 1.1.8-01

Page 1 of 1

| INPUT | PROCESS | OUTPUT |
|---|---|---|
| | 01 Update clock values and perform any time-dependent functions which have become due<br>02 Select a task for dispatching<br>03 If a task is found dispatch it<br>04 If no task is found return to OS and wait for one or more events identified during step 2 | |

Task Dispatcher Overview

Diagram - 1.1.9-01

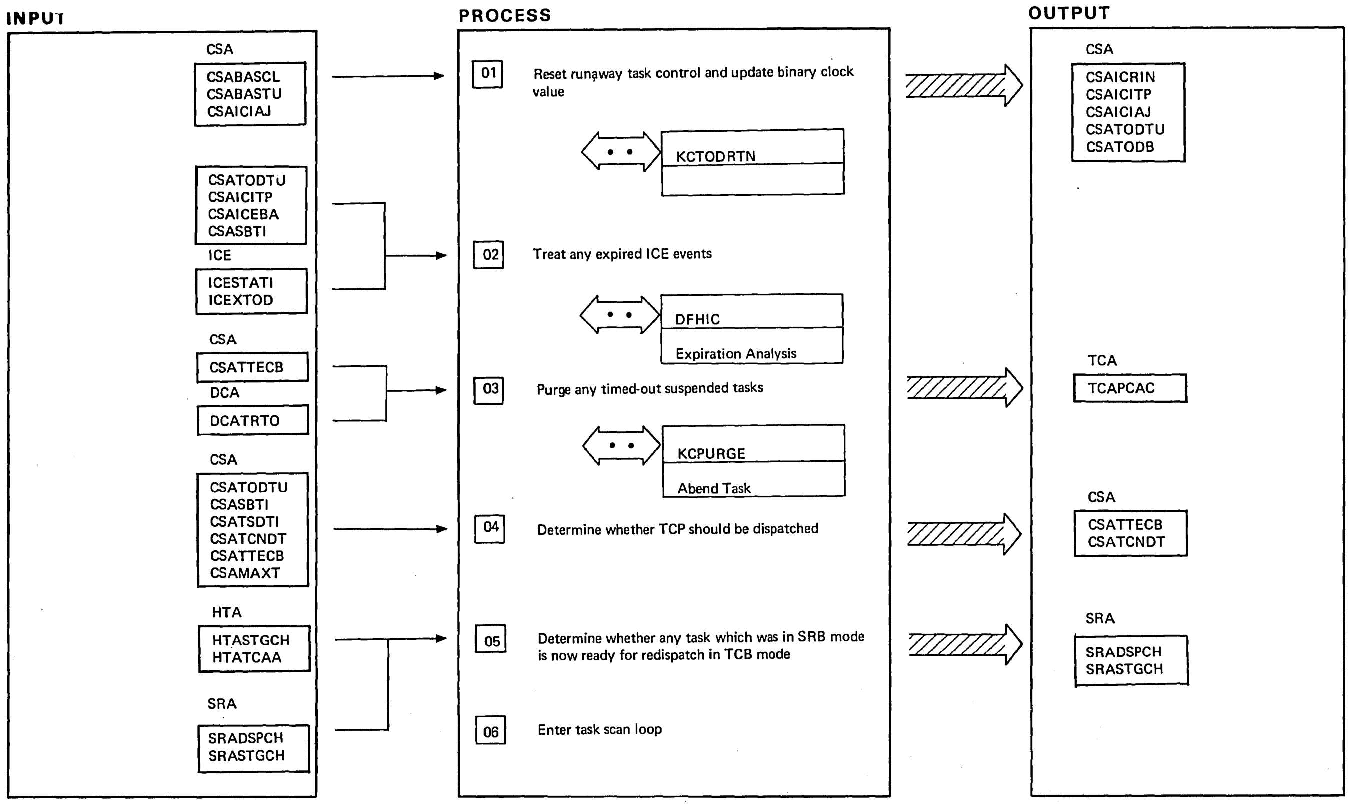
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 04 When control is regained after the WAIT the whole dispatch cycle is repeated after updating the 'packed' time-of-day CSATODP as well as binary values | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Task Dispatcher Overview

Diagram - 1.1.9-01

Page 1 of 1



Task Dispatcher Time Analysis

Diagram - 1.1.9.1-01

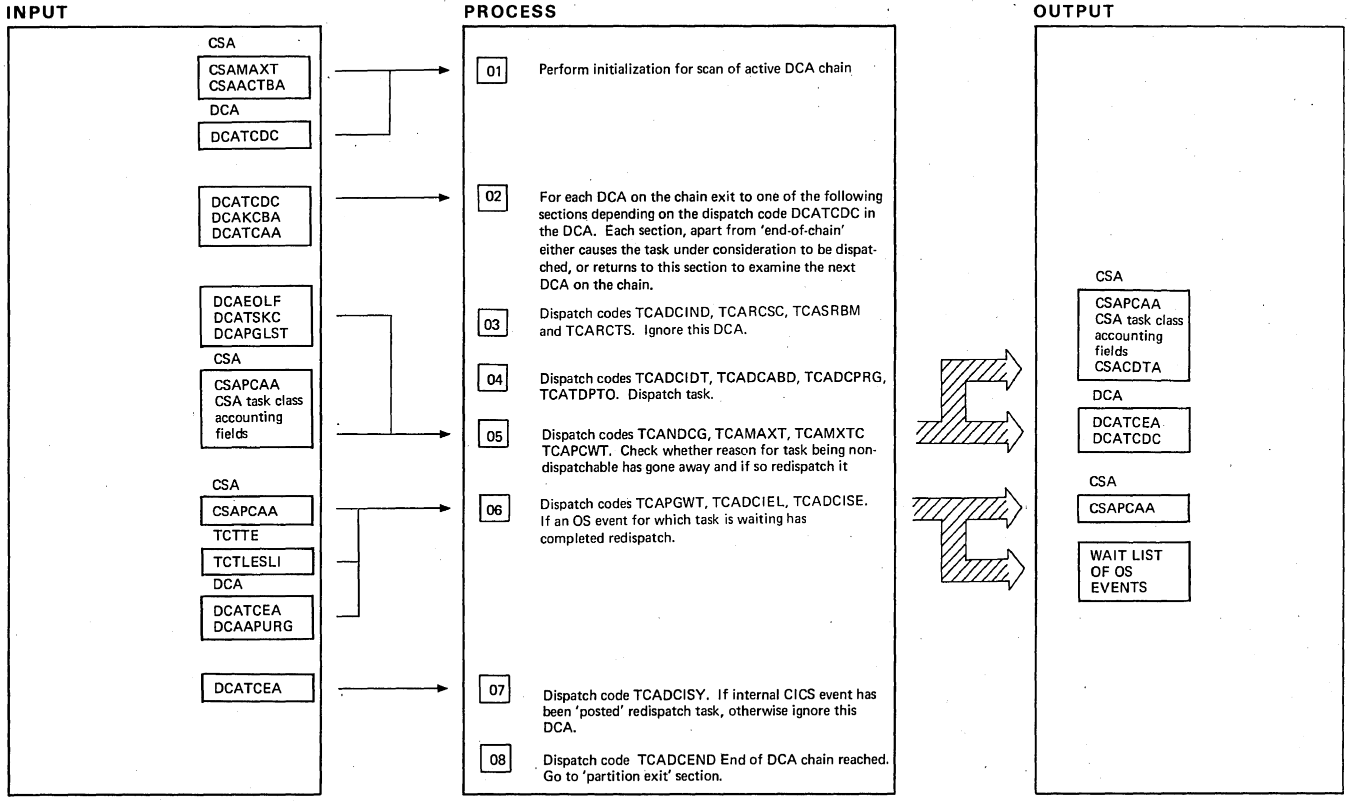
| | NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 01 | The time-of-day sub-routine recognizes the passing of 24.00 hours and sets CSAICIATP to cause DFHIC to be called in step 2. | KCTODRTN | KCDC | |
| 02 | The internal time interval KCTOTREM is set in this section to the time remaining to the first unexpired ICE, if any, otherwise to the system time interval (ICV value) | | KCXTAN | |
| 03 | If any task has been suspended waiting for a terminal read to be satisfied for longer than the 'time-out' interval specified in the PCT, it is purged with abend code 'AKCT'. The subroutine KCPURGE resumes the tasks and sets a 'timed-out' dispatch code TCADCRTO in the DCA. | | KCTEVXAB | |

| | NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 04 | If the time in CSATCNDT has expired, the timer ECB CSATTECB is set to X'40' to cause TCP task to be dispatched. If not expired, the remaining time is saved in KCTCREM.<br>If KCP is aware of any possible reason for activating the TCP task (KCTCPEVI set by task termination, WAIT non-dispatchable issued by some task etc.) the remaining time will be set to the scan delay interval CSASDTI and CSATCNDT set to correspond to that interval if this reduces the current value of CSATCNDT. | | KCCTRECB | |
| 05 | If the SRA dispatch queues are not empty, indicating that an SRB mode task is ready for redispatch, the first HTA is taken from the dispatch chain SRADSPCH (if this is empty the stage chain SRASTGCH is moved to SRADSPCH) and the associated task dispatch directly. SRASTGCH is fed by the 'LEAVE SRB MODE' service in KCSP, which also posts SRAECB. | | KCTD29 | |

Task Dispatcher Time Analysis

Diagram - 1.1.9.1-01

Page 1 of 1



INPUT

CSA: CSAMAXT, CSAACTBA
DCA: DCATCDC

DCATCDC, DCAKCBA, DCATCAA

DCAEOLF, DCATSKC, DCAPGLST
CSA: CSAPCAA, CSA task class accounting fields

CSA: CSAPCAA
TCTTE: TCTLESLI
DCA: DCATCEA, DCAAPURG

DCATCEA

PROCESS

01 Perform initialization for scan of active DCA chain

02 For each DCA on the chain exit to one of the following sections depending on the dispatch code DCATCDC in the DCA. Each section, apart from 'end-of-chain' either causes the task under consideration to be dispatched, or returns to this section to examine the next DCA on the chain.

03 Dispatch codes TCADCIND, TCARCSC, TCASRBM and TCARCTS. Ignore this DCA.

04 Dispatch codes TCADCIDT, TCADCABD, TCADCPRG, TCATDPTO. Dispatch task.

05 Dispatch codes TCANDCG, TCAMAXT, TCAMXTC TCAPCWT. Check whether reason for task being non-dispatchable has gone away and if so redispatch it

06 Dispatch codes TCAPGWT, TCADCIEL, TCADCISE. If an OS event for which task is waiting has completed redispatch.

07 Dispatch code TCADCISY. If internal CICS event has been 'posted' redispatch task, otherwise ignore this DCA.

08 Dispatch code TCADCEND End of DCA chain reached. Go to 'partition exit' section.

OUTPUT

CSA: CSAPCAA, CSA task class accounting fields, CSACDTA
DCA: DCATCEA, DCATCDC

CSA: CSAPCAA

WAIT LIST OF OS EVENTS

Task Dispatcher Active DCA Scan

Diagram - 1.1.9.2-01

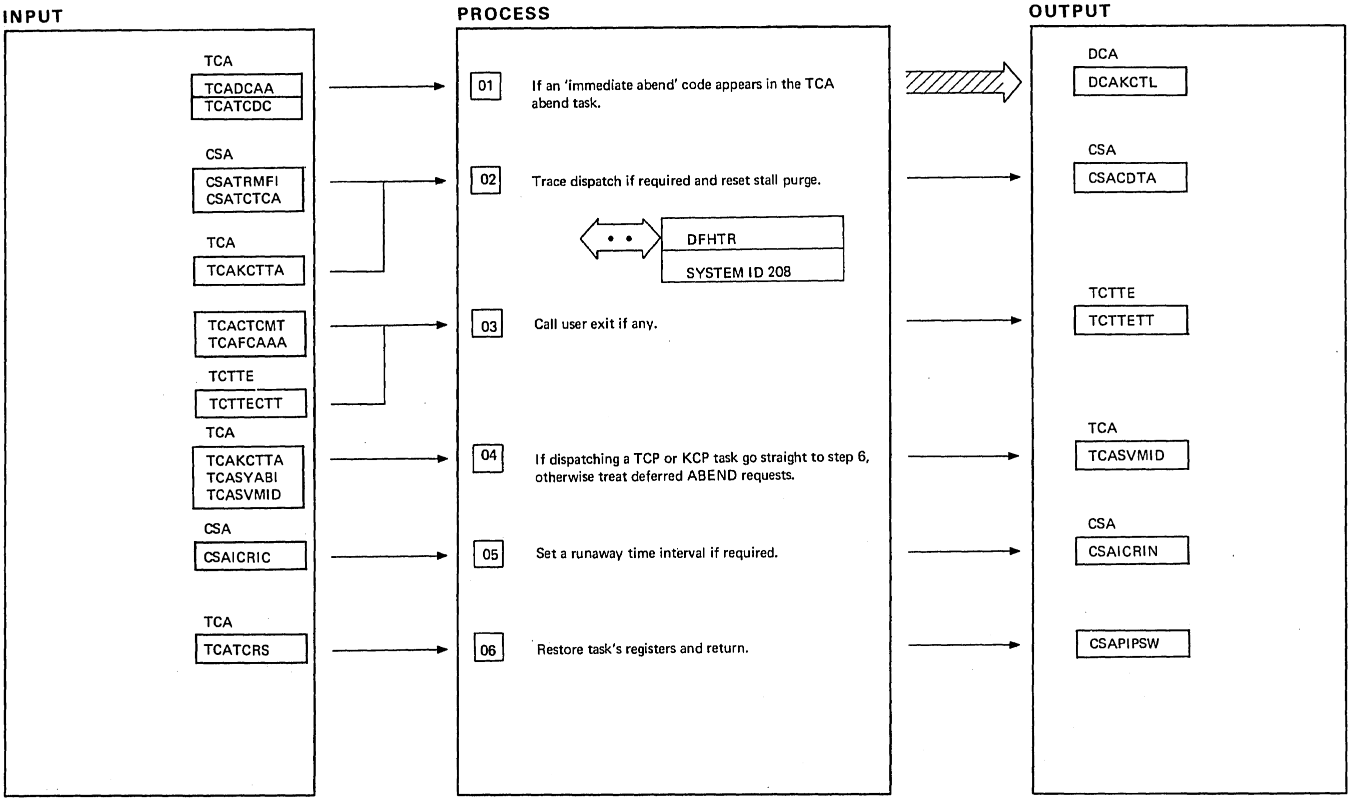
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The DCA scan is set to begin at either the first (TCP task) or the second DCA on the chain. The TCP task is skipped if the TCP dispatch time has not expired, avoiding examining all TCP ECBs on every dispatch scan. The internal switch KCTDTCP is set to show which of these courses was taken. | | KCTD40 | 1.1.9.1 |
| The address and length of storage in which to build an ECB list are loaded from the dispatcher TCA (placed there by SIFI). The I/O event counter is set to zero (used to determine the number of events to be waited for in O/S wait). | | | 1.1.9.4<br>1.1.9.4 |
| The active task counter is set to CSAMAXT | | | |
| 02 The scan may also end if the 'active task' counter falls to zero. This count is reduced by one for each 'non-system' task (i.e. not TCP or journalling) which is waiting for O/S events (see step 6). Control passes to partition exits as in step 8. | | | |
| 04 Tasks with codes other than TCACIDT are redispatched by calling PC ABEND on their behalf - this is an immediate ABEND as opposed to the deferred ABEND, recognized by the redispatch section, which waits until any CICS services have completed. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 05 • TCANDCG indicates that the task is waiting for storage to complete a KCP service. The storage request is retried. If successful the storage is freed again and the KCP service restarted. | | KCTDNDCG | |
| • TCAPCWT applies to SVS only and indicates that the task is waiting for a page fixed area in which to build a virtual subarea list for PGLOAD issued by DFHCSVC. (MVS and VS1 use register form of PGLOAD) | | KCTDPCW | |
| • TCAMAXT, TCAMAXTC are codes set when a task is inserted in the active DCA chain (the latter is for tasks belonging to a task class). Tasks are unable to proceed if active task limits would be exceeded. | | KCTDMAXT<br>KCTDMXTC | |
| 06 Tasks in these categories are waiting for OS events. If not yet posted the event(s) is (are) added to the list of ECBs for which an OS WAIT may eventually be issued. | | | |
| 08 TCADCEND is a special code assembled into the 'dummy' active DCA int the CSA at CSAACTOF. | | | |

Task Dispatcher DCA Scan

Diagram - 1.1.9.2-01

Page 1 of 1



01 If an 'immediate abend' code appears in the TCA abend task.

02 Trace dispatch if required and reset stall purge.

03 Call user exit if any.

04 If dispatching a TCP or KCP task go straight to step 6, otherwise treat deferred ABEND requests.

05 Set a runaway time interval if required.

06 Restore task's registers and return.

Task Dispatcher Dispatch

Diagram - 1.1.9.3-01

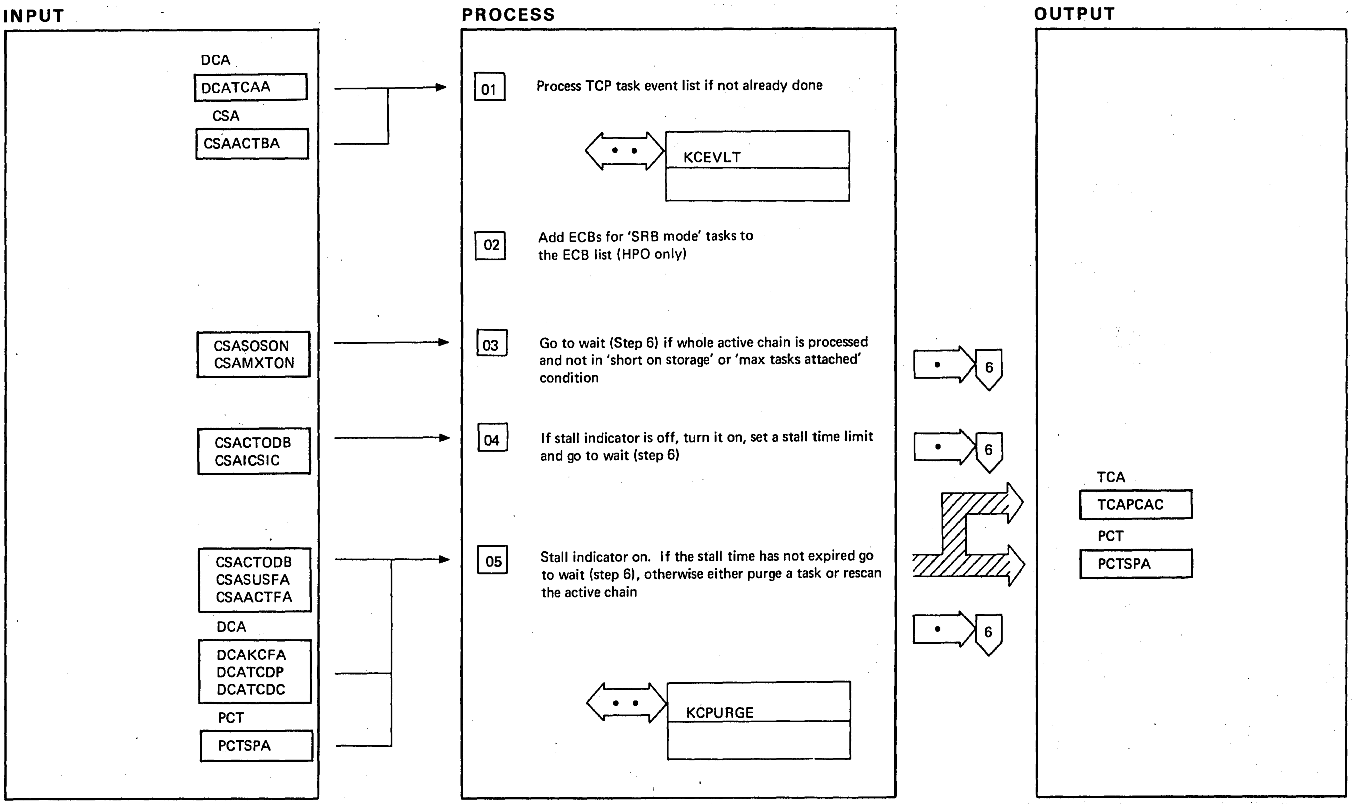
| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 02 | The stall purge indicator is turned on by partition exit when the system appears to have come to a halt. It is turned off here if a non-system task is to be dispatched. | | | 1.1.9.4 |
| 04 | The 'no flush' indicator is turned on in TCASVMID so that services invoked by the task after dispatch will not abend the task when returning.<br>For tasks other that KCP and TCP a test is made for deferred abend. In such cases the task is to.be abended when the present nest of CICS service call ends.<br>If DFHPC has already started the abend (TCAABIPM or TCAABRAM set) the dispatch continues normally. Otherwise the 'no flush' indicator is turned off and the flush linkage entered if no service calls are in progress (TCASVMID zero). | | | |

| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 05 | If CSAICRIC is zero, i.e. runaway task support is suppressed, the task is dispatched. | | | 1.1.9.5 |
| | Otherwise the interval timer is set to allow the task to be terminated if stile running when the interval expires. | | KCTD75 | |
| | To avoid frequent use of the STIMER O/S service the runaway interval is increased by 2 seconds. Successive dispatches use the same setting until the 2 seconds has elapsed, at which time a new STIMER must be issued. | | | |

Task Dispatcher Dispatch

Diagram - 1.1.9.3-01

Page 1 of 2



01 Process TCP task event list if not already done

02 Add ECBs for 'SRB mode' tasks to the ECB list (HPO only)

03 Go to wait (Step 6) if whole active chain is processed and not in 'short on storage' or 'max tasks attached' condition

04 If stall indicator is off, turn it on, set a stall time limit and go to wait (step 6)

05 Stall indicator on. If the stall time has not expired go to wait (step 6), otherwise either purge a task or rescan the active chain

Dispatcher - Partition Exit

Diagram - 1.1.9.4-01

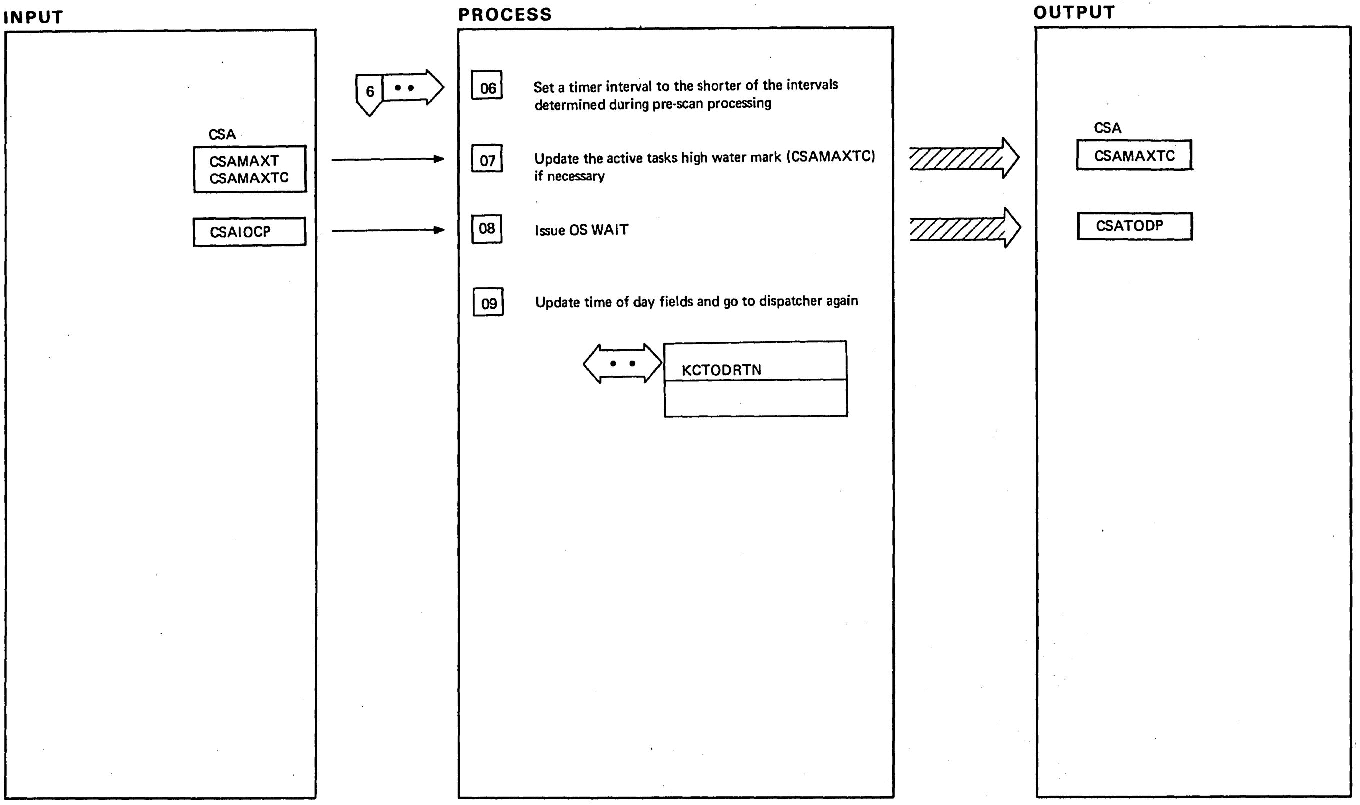
| | NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 01 | If the TCP task was skipped in the DCA active chain scan it must now be examined and its associated events added to the list for the OS WAIT.<br>If any event is in fact posted the TCP task will be dispatched immediately. | | KCTDPXN | 1.1.9.2 |
| 02 | The 'SRB mode' ECBs are posted by KCSP when a task running in SRB mode returns to TCB mode by issuing a 'LEAVE SRB MODE' request. | | KCTDPX20 | |
| 04 | The stall time limit is the present-time *CSAICSIC. If, after that time, no useful work has been done, tasks are purged from the system in the hope that freeing their resources will allow remaining tasks to proceed. | | KCSTTMTA | |
| 05 | Successive dispatches, separated in time by the system time interval (ICV) are attempted until the stall time has expired. If, when this occurs, the full active DCA chain has not been examined because of 'maximum active task' limits (CSAMAXT) a final dispatch cycle with no limit is attempted. If, and only if, this fails purging of tasks is begun. 'Purgeable' tasks are purged in order of increasing priorty from the suspended chain, then in ascending priority order from the active chain. | | KCSTTMTB<br>KCPATASK | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Dispatcher - Partition Exit

Diagram - 1.1.9.4-01

Page 2 of 2



06 Set a timer interval to the shorter of the intervals determined during pre-scan processing

07 Update the active tasks high water mark (CSAMAXTC) if necessary

08 Issue OS WAIT

09 Update time of day fields and go to dispatcher again

Dispatcher - Partition Exit

Diagram - 1.1.9.4-02

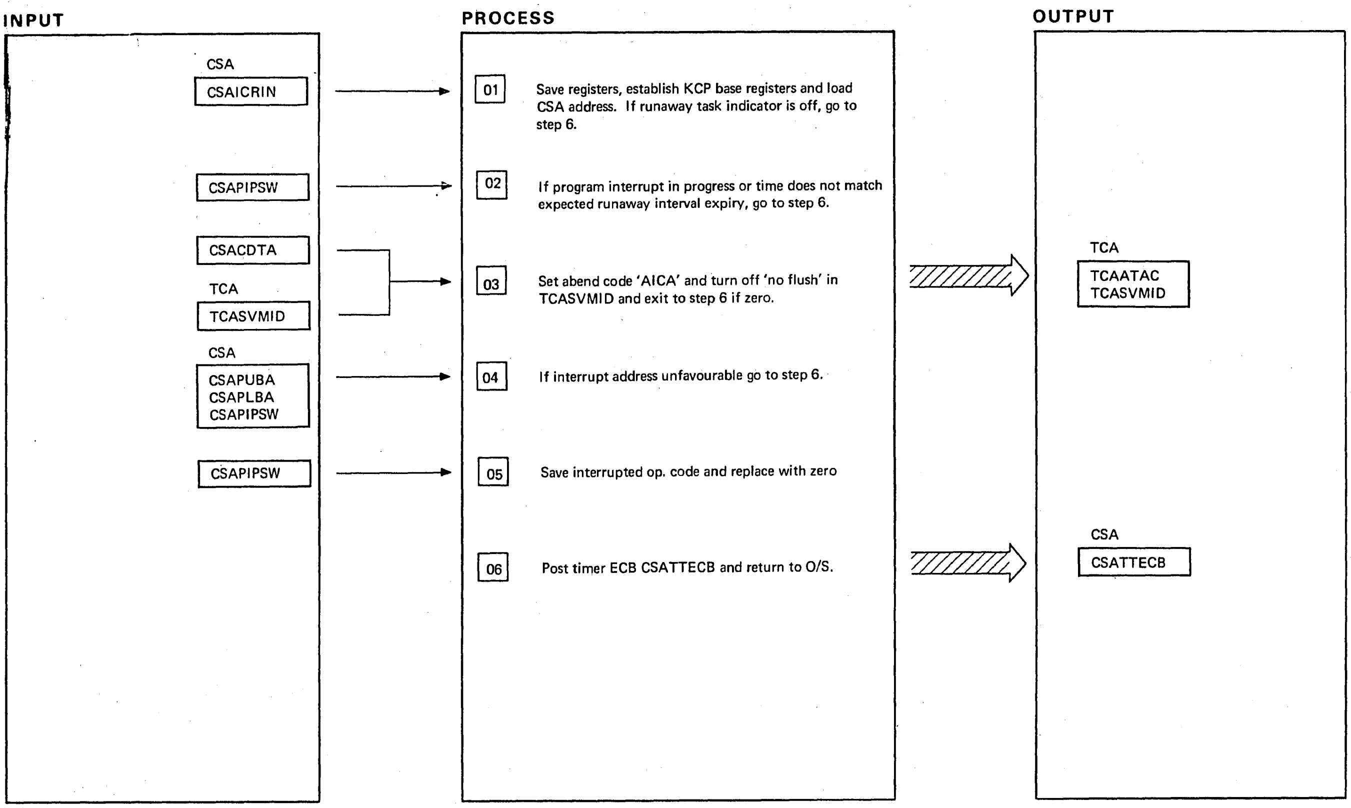
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 06 The interval is the time to the next timed event, or, if shorter, the system time interval (ICV) | | KCTDWT | 1.1.9.1 |
| 08 During the active DCA scan a count was maintained of I/O events ('single events' or page I/O events). A percentage (CSAIOCP) of this is used as count argument to the OS WAIT macro in an attempt, given a reasonably high percentage, to ensure that a reasonable amount of work will be on hand when control is returned after the wait. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Dispatcher - Partition Exit

Diagram - 1.1.9.4-02

Page 1 of 1



INPUT

CSA

CSAICRIN

CSAPIPSW

CSACDTA

TCA

TCASVMID

CSA

CSAPUBA
CSAPLBA
CSAPIPSW

CSAPIPSW

PROCESS

01 Save registers, establish KCP base registers and load CSA address. If runaway task indicator is off, go to step 6.

02 If program interrupt in progress or time does not match expected runaway interval expiry, go to step 6.

03 Set abend code 'AICA' and turn off 'no flush' in TCASVMID and exit to step 6 if zero.

04 If interrupt address unfavourable go to step 6.

05 Save interrupted op. code and replace with zero

06 Post timer ECB CSATTECB and return to O/S.

OUTPUT

TCA

TCAATAC
TCASVMID

CSA

CSATTECB

Timer Interrupt Exit Routine

Diagram - 1.1.9.5-01

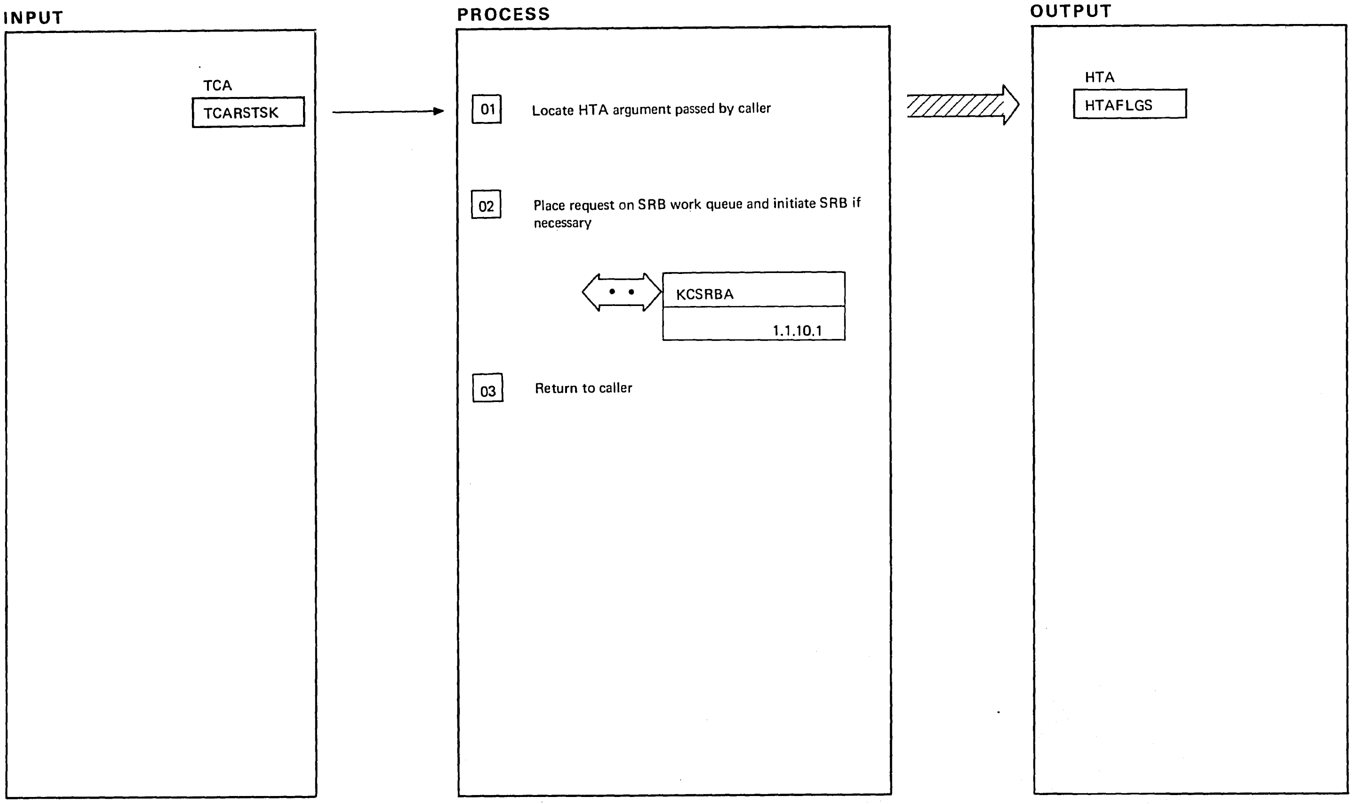
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 If runaway task indicator is off this represents the expiry of a 'partition exit' time interval. The CSA address is stored in KCP during initialization. | ICINTRPT |  | 1.1.9.4 |
| 02 In MVS it is possible for a timer interrupt to occur as a result of an STIMER issued before the current one. |  |  |  |
| 03 Timing off 'no flush' causes the task to be abended when control reaches the top of the nest of CICS service calls. |  | ICINT30 |  |
| 04 Interruption occuring outside CICS storage, in protected CICS modules, or in SPIE or ESTAE exits cannot be handled. |  |  |  |
| 05 The op. code is temporarily replaced by zero. When control returns to the point of interrupt a program check occurs which the SPIE routine, because of the saved op. code, recognizes as being in reality a timer interrupt. It restores the op. code and ABENDS the offending task. |  | ICNOPCHK |  |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Timer Interrupt Exit Routine

Diagram - 1.1.9.5-01

Page 1 of 1



01 Locate HTA argument passed by caller

02 Place request on SRB work queue and initiate SRB if necessary

03 Return to caller

Attach HTA

Diagram - 1.1.10-01

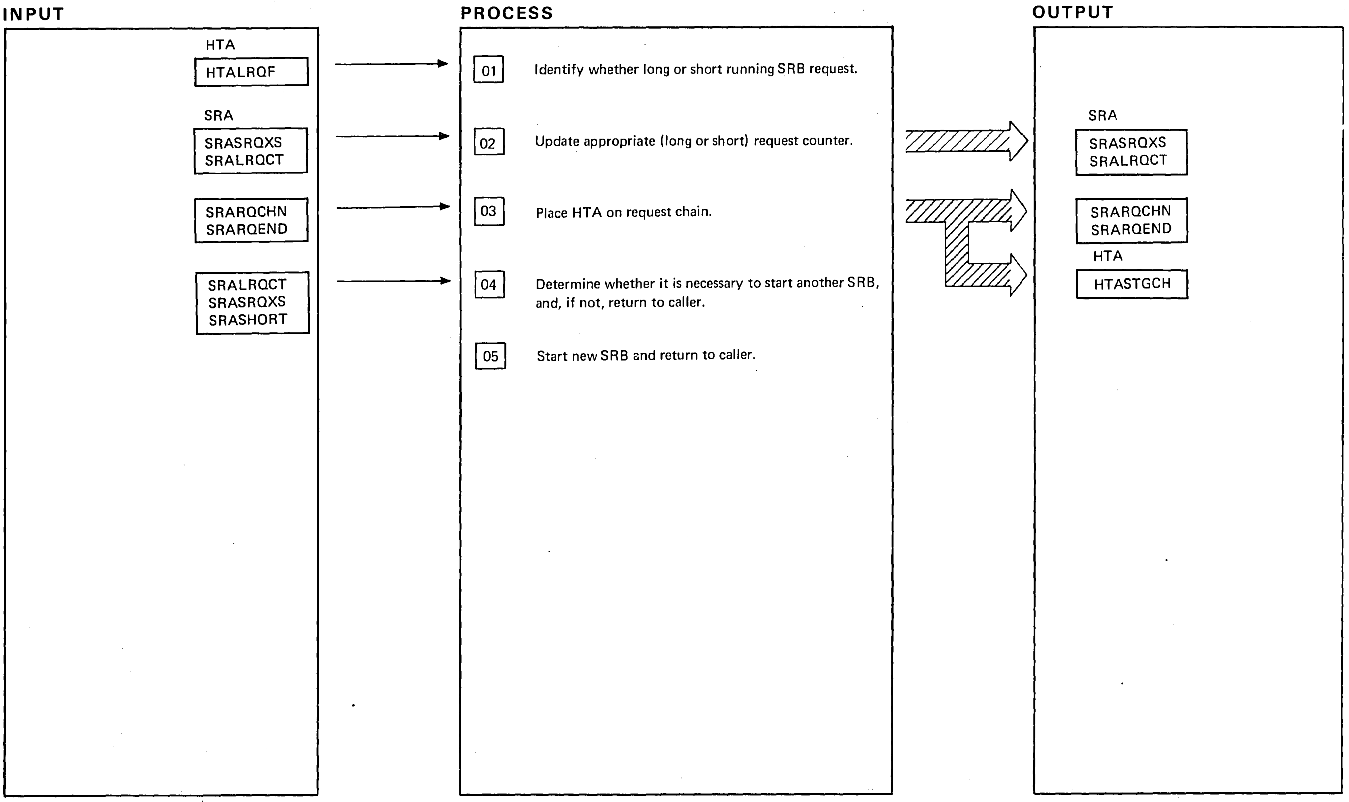
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The argument HTA is marked as being associated with an 'ATTACH' | | KCAH | |
| 03 The OS register save area in the CSA is used for this service | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Attach HTA

Diagram - 1.1.10-01

Page 1 of 1



01 Identify whether long or short running SRB request.

02 Update appropriate (long or short) request counter.

03 Place HTA on request chain.

04 Determine whether it is necessary to start another SRB, and, if not, return to caller.

05 Start new SRB and return to caller.

Subroutines KCSRBA/KCSRBE

Diagram - 1.1.10.1-01

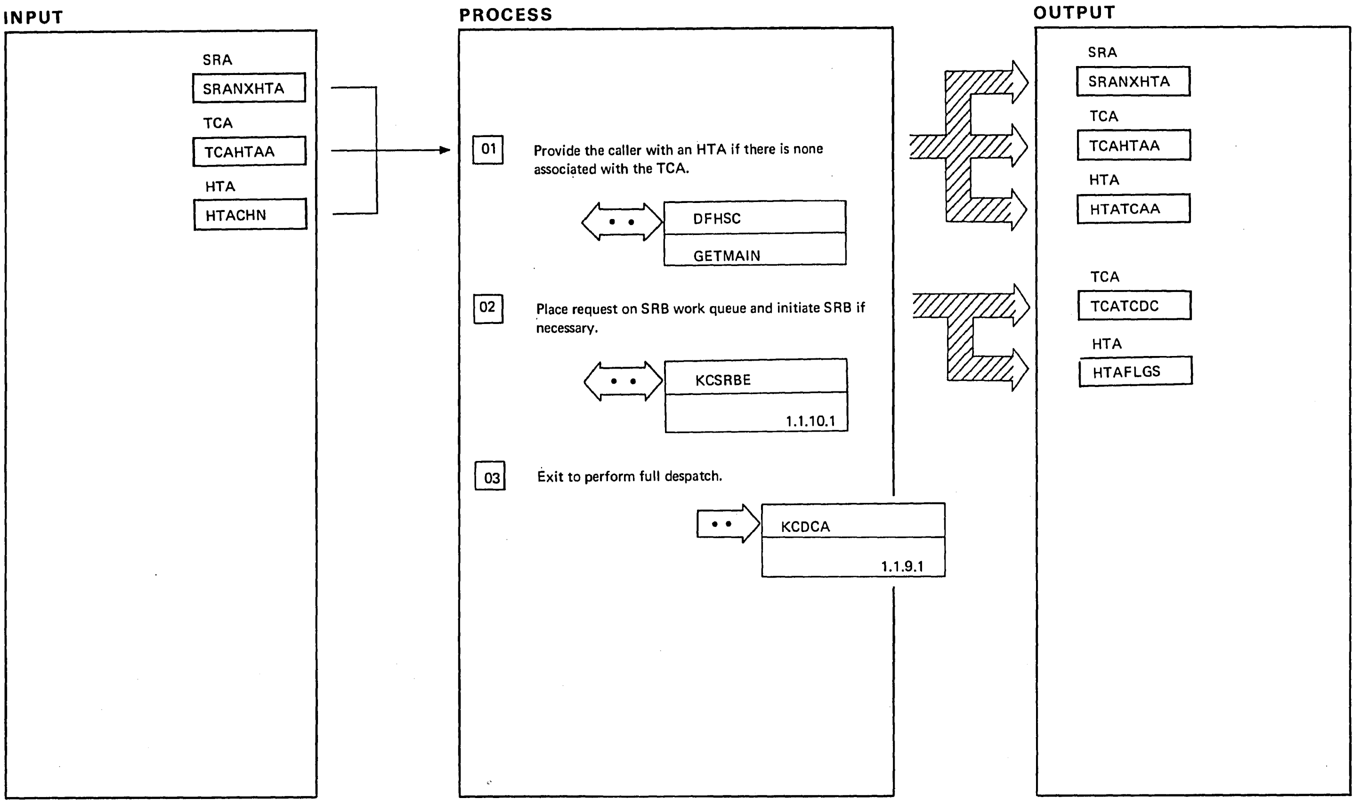
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Entry point KCSRBA is used by 'ATTACH HTA' and may request a long or short running SRB. Entry point KCSRBE is used by 'ENTER SRB MODE' and assumes a short run request. | | | 1.1.10<br>1.1.11 |
| 03 The HTA is placed at the end of the chain. Because SRB executions, possibly running concurrently, take items off this chain, some moderately devious compare and swap logic is required here. | | KCSRB20 | |
| 04 A new SRB is started if either there are very few 'short run' SRBs active, or the request queue has become larger than a threshold value. | | | |
| 05 A type 6 SVC is issued. This dispatches an SRB (with entry point in KCSP) and returns. | | KCSRB50 | 1.1.12 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Subroutines KCSRBA/KCSRBE

Diagram - 1.1.10.1-01

Page 1 of 1


INPUT
SRA
SRANXHTA
TCA
TCAHTAA
HTA
HTACHN
PROCESS
01
Provide the caller with an HTA if there is none associated with the TCA.
DFHSC
GETMAIN
02
Place request on SRB work queue and initiate SRB if necessary.
KCSRBE
1.1.10.1
03
Exit to perform full despatch.
KCDCA
1.1.9.1
OUTPUT
SRA
SRANXHTA
TCA
TCAHTAA
HTA
HTATCAA
TCA
TCATCDC
HTA
HTAFLGS

Enter SRB Mode

Diagram - 1.1.11-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 If the caller does not already have on HTA (TCAHTAA=0) one is obtained from the quick-cell chain SRANXHTA (using compare and swap since HTAs are returned by SRB mode process) or, if this is empty, by issuing a GETMAIN. If this fails the task is temporarily set aside with a 'non-dispatchable' code TCANDCG until storage becomes available. | | KCMS | |
| 03 From the point of view of normal TCB mode CICS, the task has now become non-dispatchable, and will remain so (with TCASRBM dispatch code) until, in SRB mode, it issues a 'LEAVE SRB MODE' service request to KCSP. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Enter SRB Mode

Diagram - 1.1.11-01

Page 1 of 1

**INPUT**

(See detail charts)

**PROCESS**

01 Initialize SRB mode execution.

02 Select item from request chain, check validity of request and provide environment exit to requestor.

03 On return from SRB mode execution via DETACH HTA or LEAVE SRB mode service, dispose of resources and go to step 2.

**OUTPUT**

(See detail charts)

SRB Management Overview

Diagram - 1.1.12-01

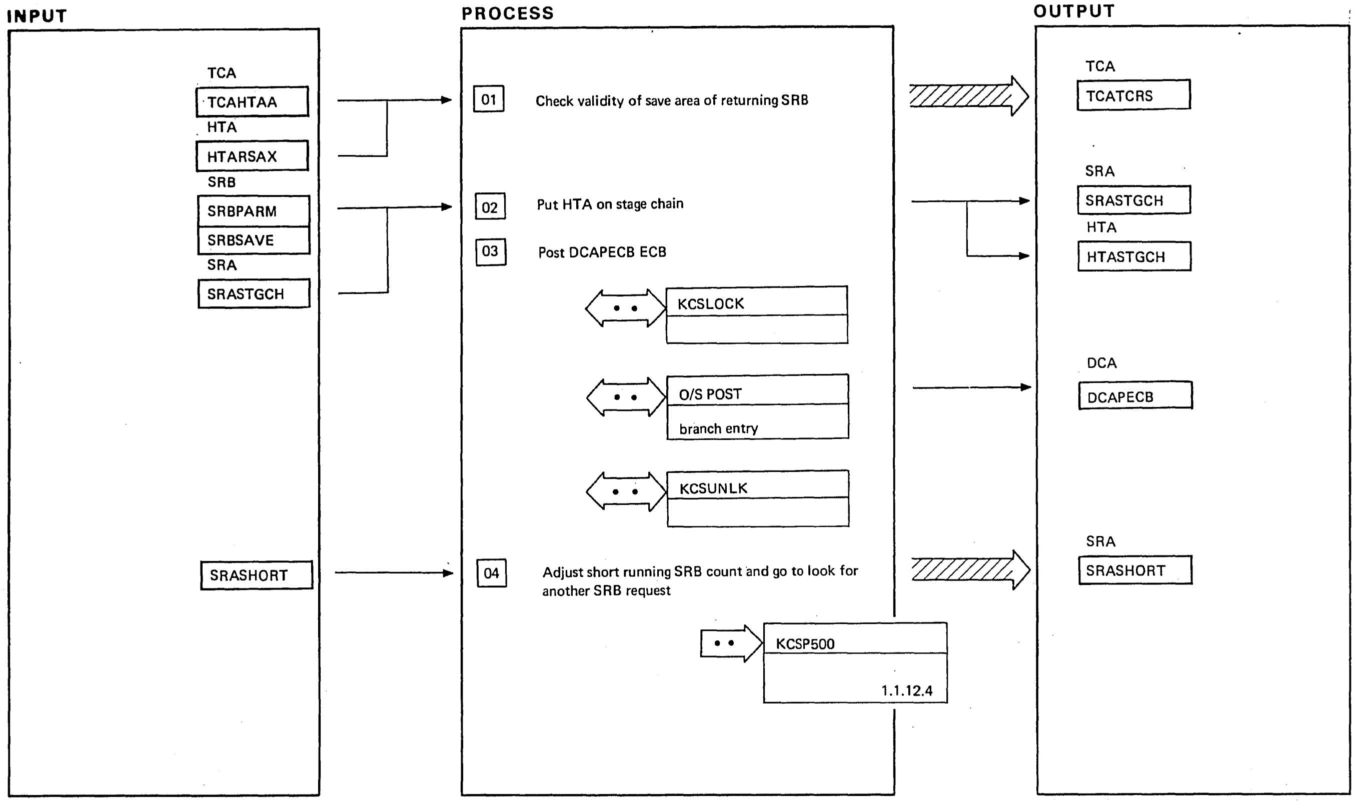
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 SRB mode execution is initiated by the issuing of the type 6 SVC SRBSVC by KCP. | | | 1.1.10.1<br>1.1.12.3 |
| 02 The request chain is built by KCP as a result of ATTACH HTA and ENTER SRB MODE service requests. If the request chain is empty the SRB terminates. | | | 1.1.10.1<br>1.1.12.4 |
| 03 These services and the above steps all reside in DFHKCSP, which is loaded in protected storage by DFHSIB1. | | | 1.1.12.1<br>1.1.12.4 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

SRB Management Overview

Diagram - 1.1.12-01

Page 1 of 1



01 Check validity of save area of returning SRB

02 Put HTA on stage chain

03 Post DCAPECB ECB

04 Adjust short running SRB count and go to look for another SRB request

Leave SRB Mode

Diagram - 1.1.12.1-01

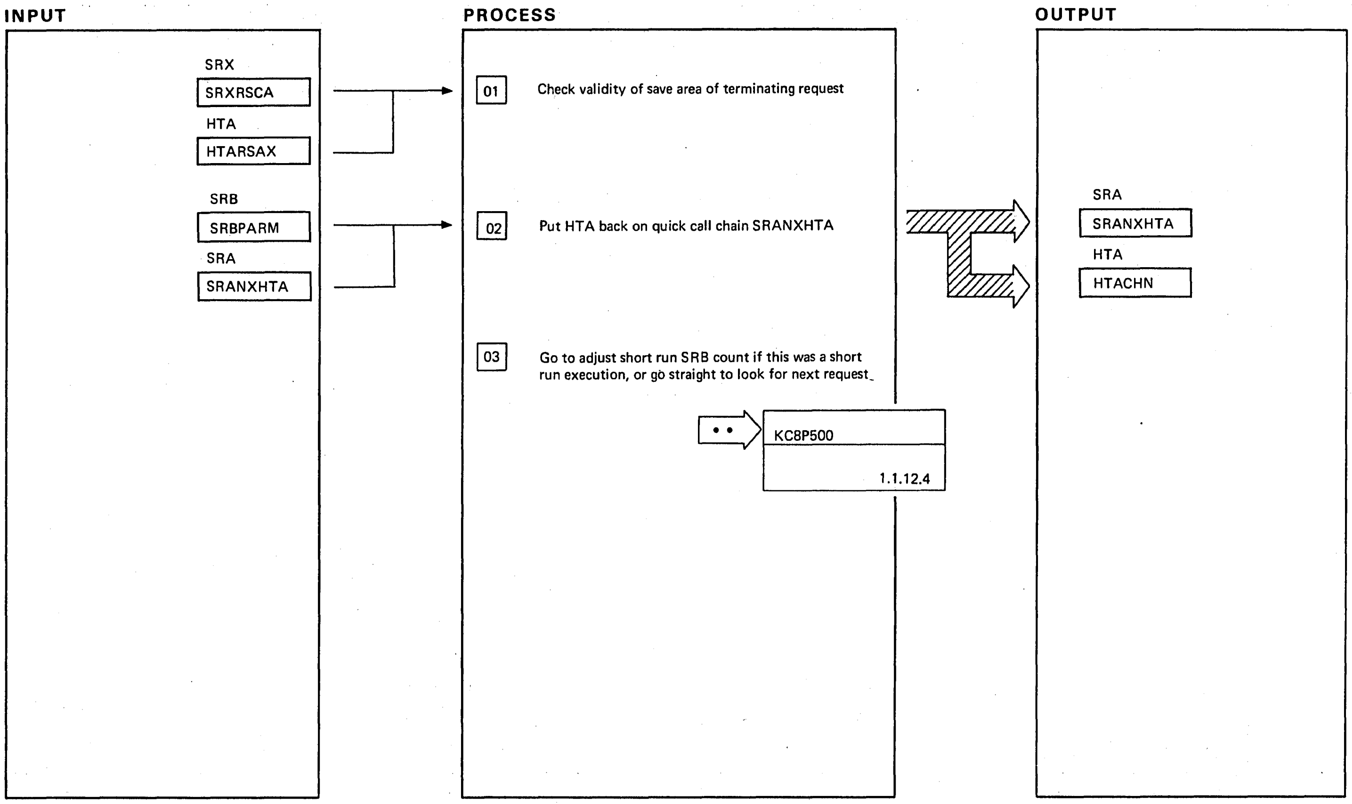
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 A check is made that the save area index which was associated with the SRB is still valid, and that the FRR parameter area is still the one established by CICS. The save area and FRR can be reused if so. | DFHKCSPL | | |
| 02 Placing the HTA on the stage chain makes the CICS task available for running in TCB mode again from the point at which 'LEAVE SRB MODE' was called. | | | |
| 03 DCAPECB is posted to indicated that a task is ready to run. If CICS is waiting on DCAPECB (among others) the local lock is acquired and POST invoked via branch entry. | | KCSP120 | |
| 04 The SRB execution has completed as far as the caller is concerned. If there is any outstanding SRB mode request this will now be satisfied. | | KCSP160 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Leave SRB Mode

Diagram - 1.1.12.1-01

Page 1 of 1



01 Check validity of save area of terminating request

02 Put HTA back on quick call chain SRANXHTA

03 Go to adjust short run SRB count if this was a short run execution, or go straight to look for next request

Detach HTA

Diagram - 1.1.12.2-01

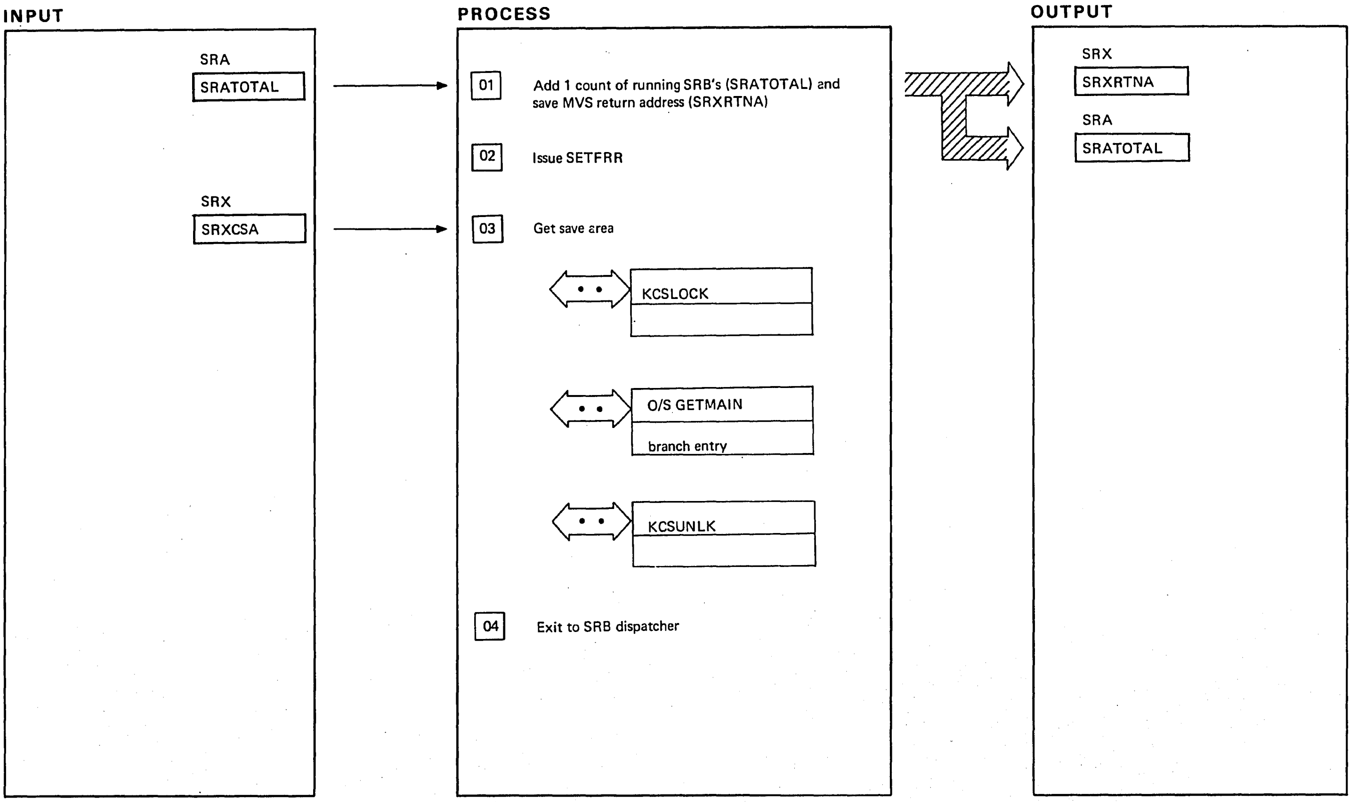
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Check is as for 'LEAVE SRB MODE'. | DCHKCSPD | | 1.1.12.1 |
| 03 If this was a short run SRB then the count must be reduced by 1. Control is passed to 1.1.12.1 step 4 in this case. | | | 1.1.12.1 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Detach HTA

Diagram - 1.1.12.2-01

Page 1 of 1



SRB Initialization .

Diagram - 1.1.12.3-01

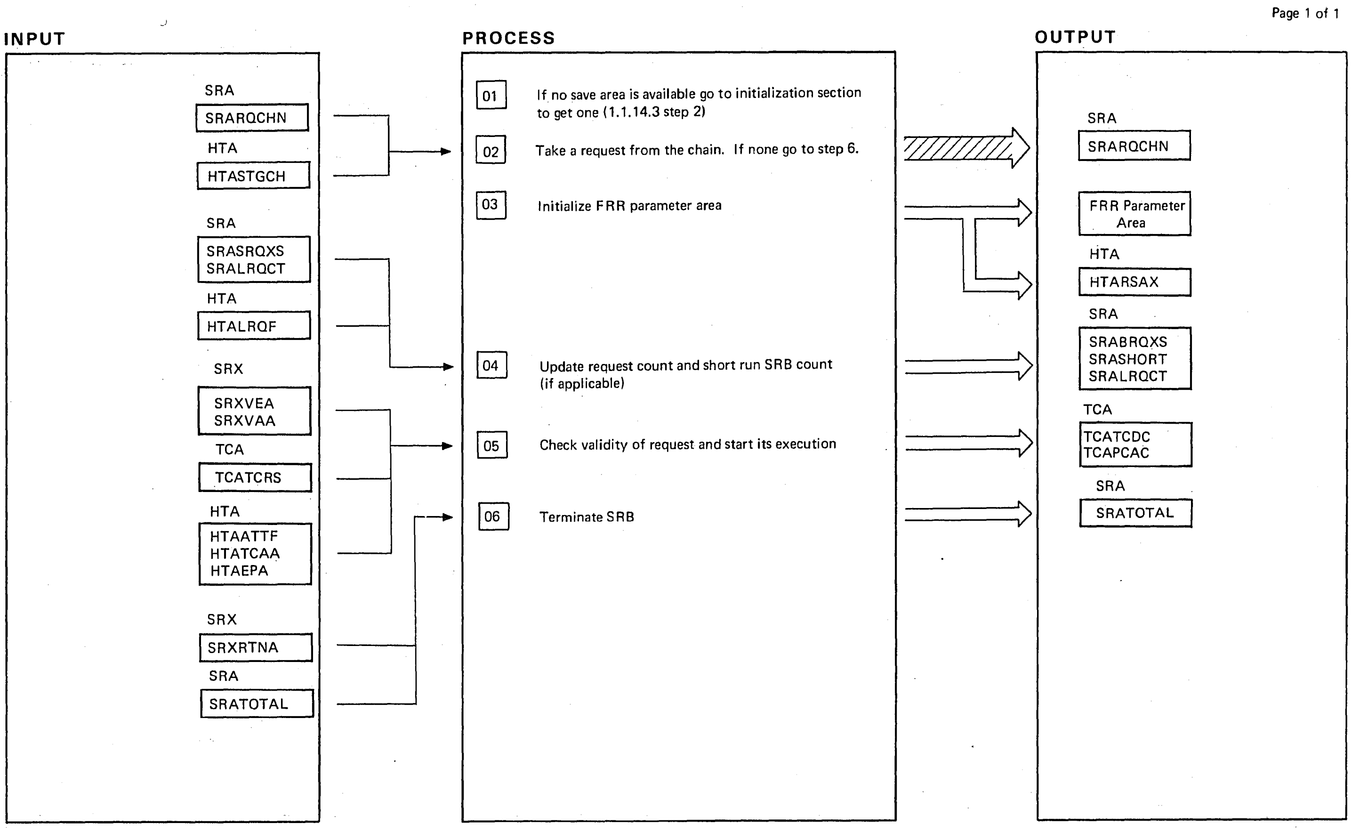
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The MVS return address is saved here. It is always saved in the same place, so SRBs always return to the same point in MVS. The SRB executions which return to MVS from KCSP are not normally initiated by CICS in any case. | DFHKCSPP | | |
| 02 The recovery routine nominated here simply ABENDS CICS. | | | KCSP310 |
| 03 An O/S save area must be supplied for use by System Services called by the SRB execution. The normal CICS O/S save area in the CSA cannot be used since the SRBs run in parallel.<br>The save areas are protected to prevent corruption during the comparatively long periods of use during I/O operations.<br>If the chain of free areas is empty a new area is obtained in subpool 252 using branch entry to GETMAIN (which requires that the local lock be held).<br>The SRB execution is terminated if the GETMAIN fails. | | | KCSP320 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

SRB Initialization

Diagram - 1.1.12.3-01

Page 1 of 1



01 If no save area is available go to initialization section to get one (1.1.14.3 step 2)

02 Take a request from the chain. If none go to step 6.

03 Initialize FRR parameter area

04 Update request count and short run SRB count (if applicable)

05 Check validity of request and start its execution

06 Terminate SRB

SRB Dispatcher

Diagram - 1.1.12.4-01

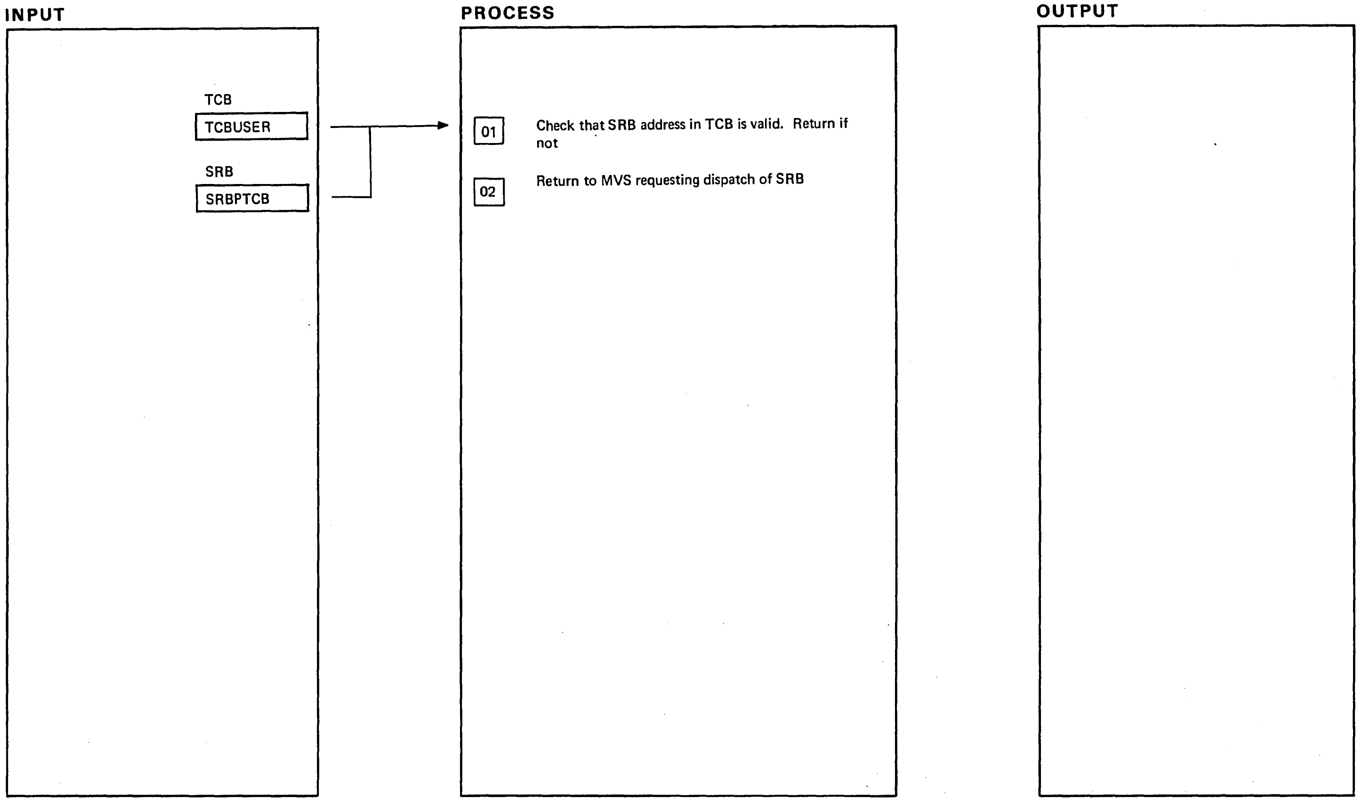
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 A save area may be unavailable if the check in 'LEAVE SRB MODE' or 'DETACH HTA' failed. | | KCSP500 | 1.1.12.1<br>1.1.12.2 |
| 02 The request chain SRARQCHN is built up by 'ENTER SRB MODE' and 'ATTACH HTA' services in KCP. | | KCSP510 | 1.1.10<br>1.1.11 |
| 03 The SRX address and the current HTA address are placed in the FRR parameter area. | | | |
| 05 A check is made to ensure that the address to which control will be passed is one of a protected list of valid addresses. | | KCSP540 | |
| 06 The save area is put back on the free chain and the total of running SRB executions reduced by one. Control is returned to MVS at the address saved in SRXRTNA | | KCSP580 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

SRB Dispatcher

Diagram - 1.1.12.4-01

Page 1 of 1



01 Check that SRB address in TCB is valid. Return if not

02 Return to MVS requesting dispatch of SRB

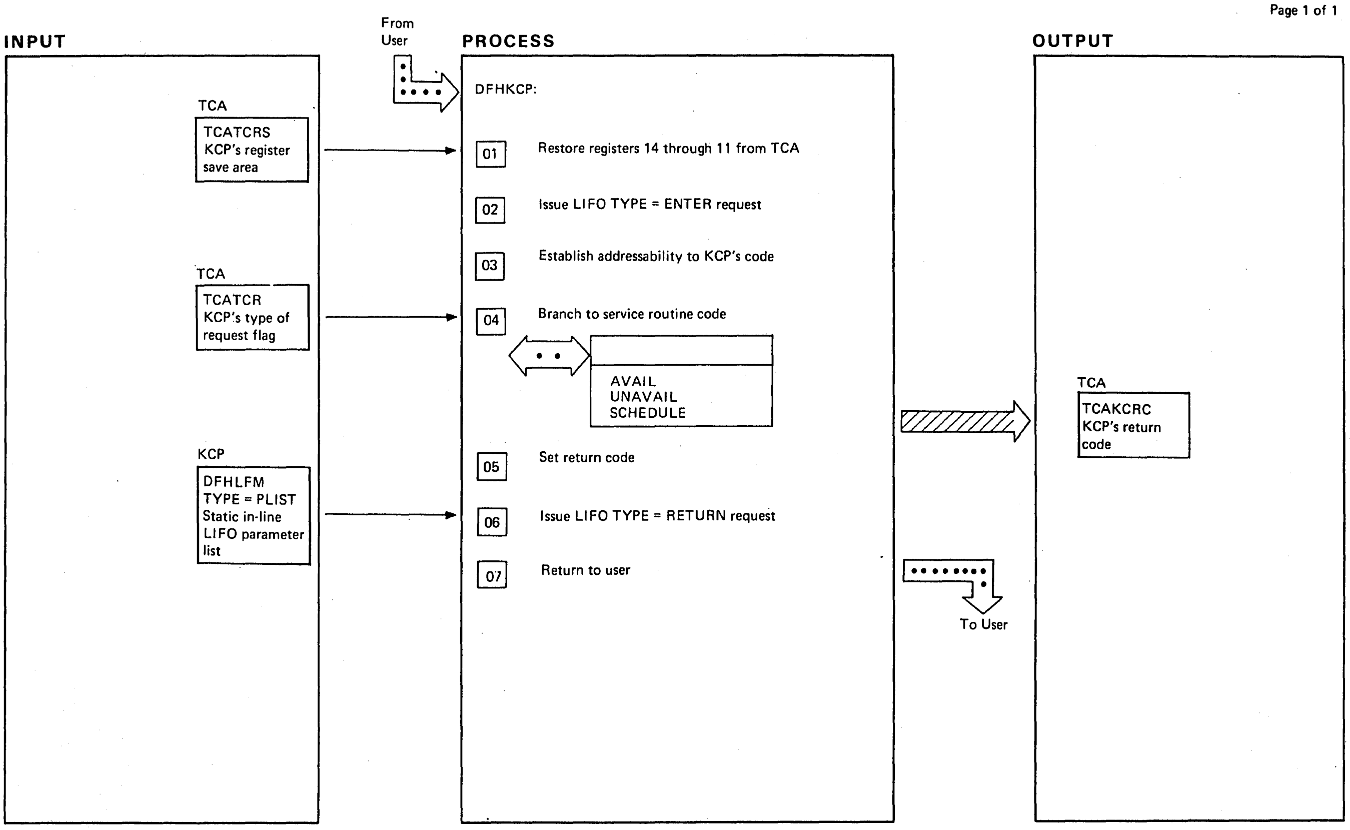
HPO SVC (DFHHPSVC)

Diagram - 1.1.12.5-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 This module is a type 6 SVC link edited into the MVS nucleus. It is invoked only by KCP in response to an 'ATTACH HTA' or 'ENTER SRB MODE' request issued by ZCP or FCP. Its sole function is to dispatch the SRB described by the SRB imbedded in the SRX allocated in SQA by DFHSIBI. | DFHHPSVC | | 1.1.9.2<br>1.1.9.3 |
| The user field of the callers TCB should address the SRX whose TCB pointer should point back to that TCB. | | | |
| 02 The program flow logically splits at this point, in SRB starting and the TCB continuing. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Page 1 of 1



DFHKCP:

01 Restore registers 14 through 11 from TCA

02 Issue LIFO TYPE = ENTER request

03 Establish addressability to KCP's code

04 Branch to service routine code

05 Set return code

06 Issue LIFO TYPE = RETURN request

07 Return to user

LIFO Storage Requests

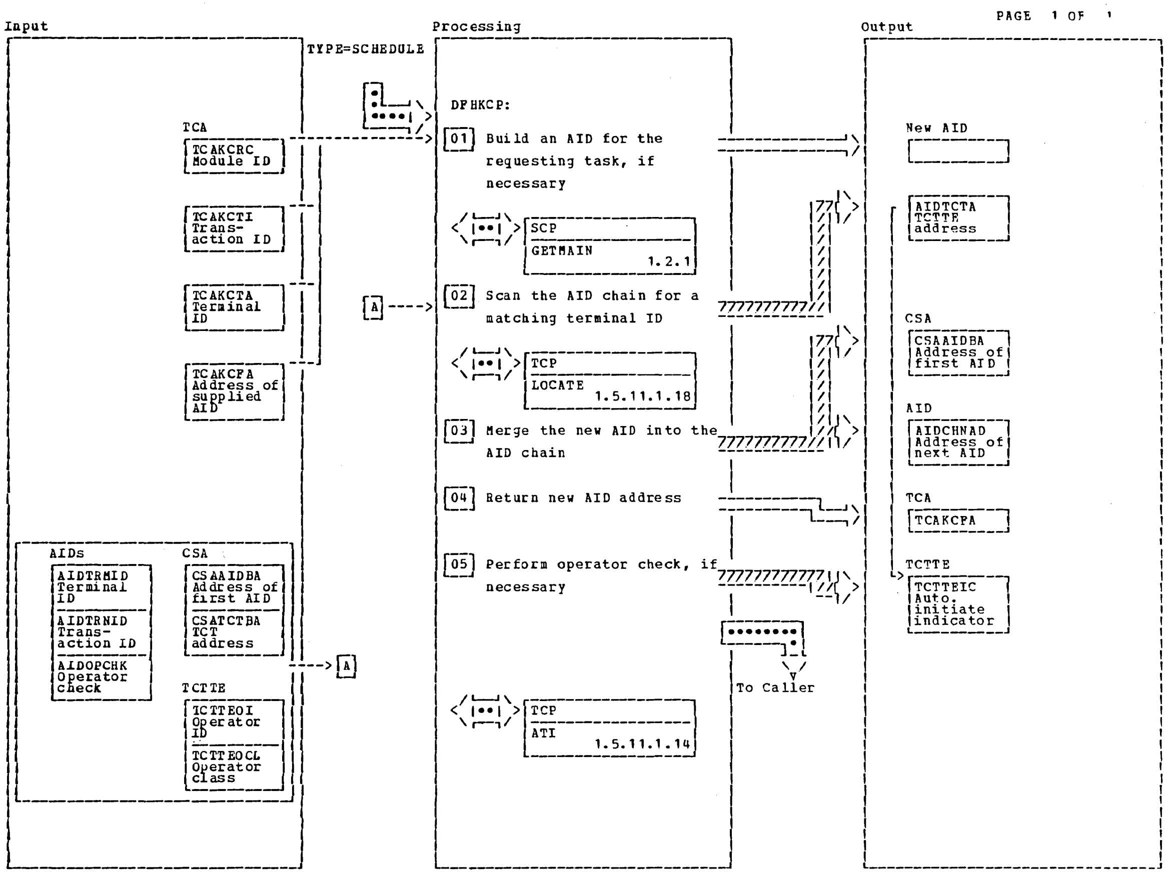
Diagram – 1.1.13-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The registers are restored at this point since return will be made from this request to the user, not to KCP. | | KCLFM | |
| 03 Base registers are set up to be compatible with KCP's registers in the non-LIFO case | | | |
| 04 Although the service routine is branched to, it always returns to set the return code | | | |
| 06 LIFO return is made to the requestor | | KCLFMRET | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

LIFO Storage Requests

Diagram – 1.1.13-01



Schedule a Resource

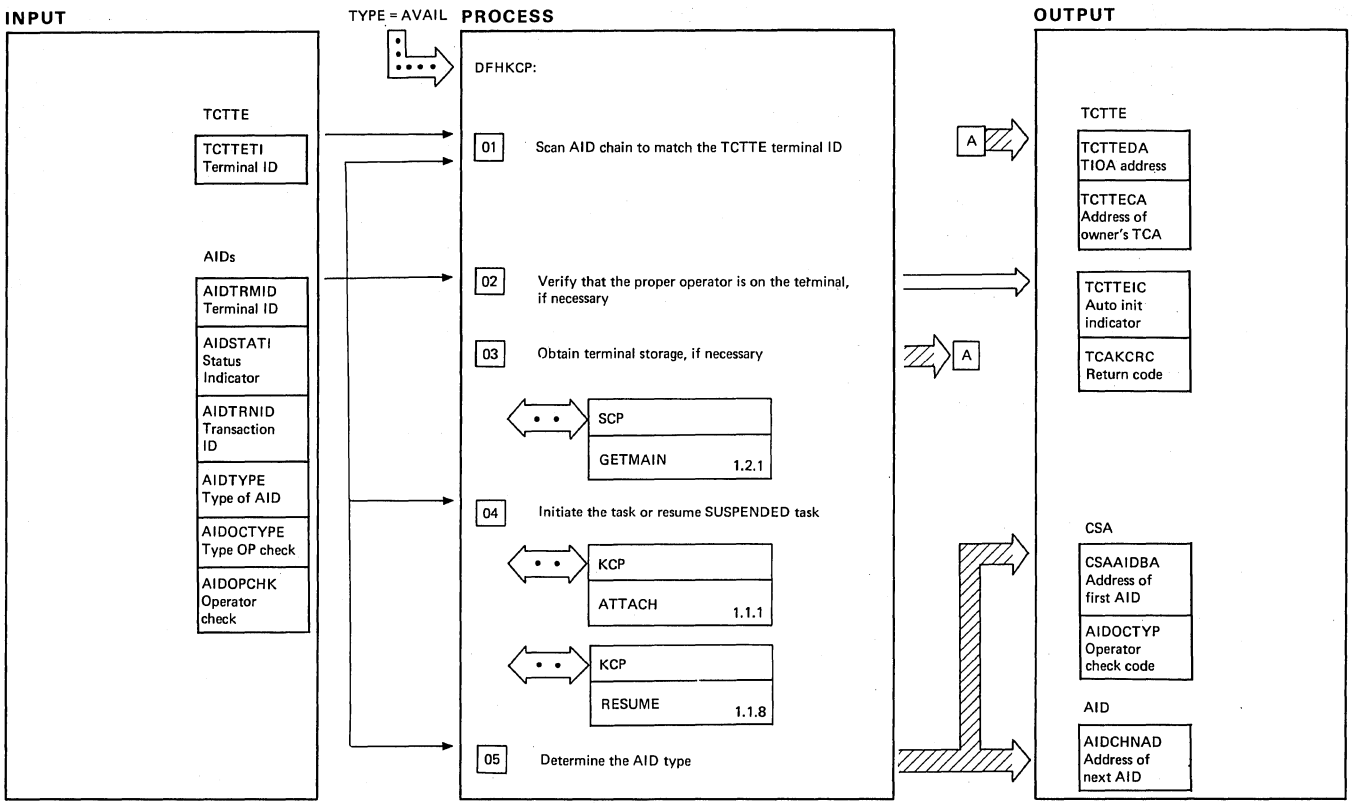
Diagram - 1.1.13.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 An AID is built if the requesting task does not supply one. If AID storage is not available, an appropriate response code is returned directly to the calling routine. | KCTDDTN | KCRSN | |
| 02 If an existing AID with a matching terminal ID is not found, the TCTTEs are searched, via the DFHTC LOCATE function, for one and the address of the matched TCTTE. If no matching TCTTE is found, an appropriate response code is returned to the calling routine. | | KCRSACN | |
| If a matching ID is found, the TCTTE address is put in the AID and the AID is then searched for matching transaction ID and operator check. | | | |
| 03 The new AID is matched against existing AIDs until the first no-match condition after a match of terminal IDs occurs. | | KCAIDMRG | |
| The new AID is merged where this condition occurs. | | | |
| If this condition does not occur, the new AID is merged at the end of the chain. | | | |
| 05 The new AID contains a code and an operator check value if a check is to be made. The check involves comparing the operator check field of the AID against one of two fields in the TCTTE. | | | |
| If equal, or if no check is required, the Automatic Initiate indicator is turned on in the TCTTE, via the DFHTC ATI function. If neither, the indicator is left unchanged. | | KCSETTCI | |
| Control is returned directly to the caller. | KCTDDTN | | |

Schedule a Resource

Diagram - 1.1.13.1-01

Page 1 of 2



Declare Resource Availability

Diagram — 1.1.13.2–01

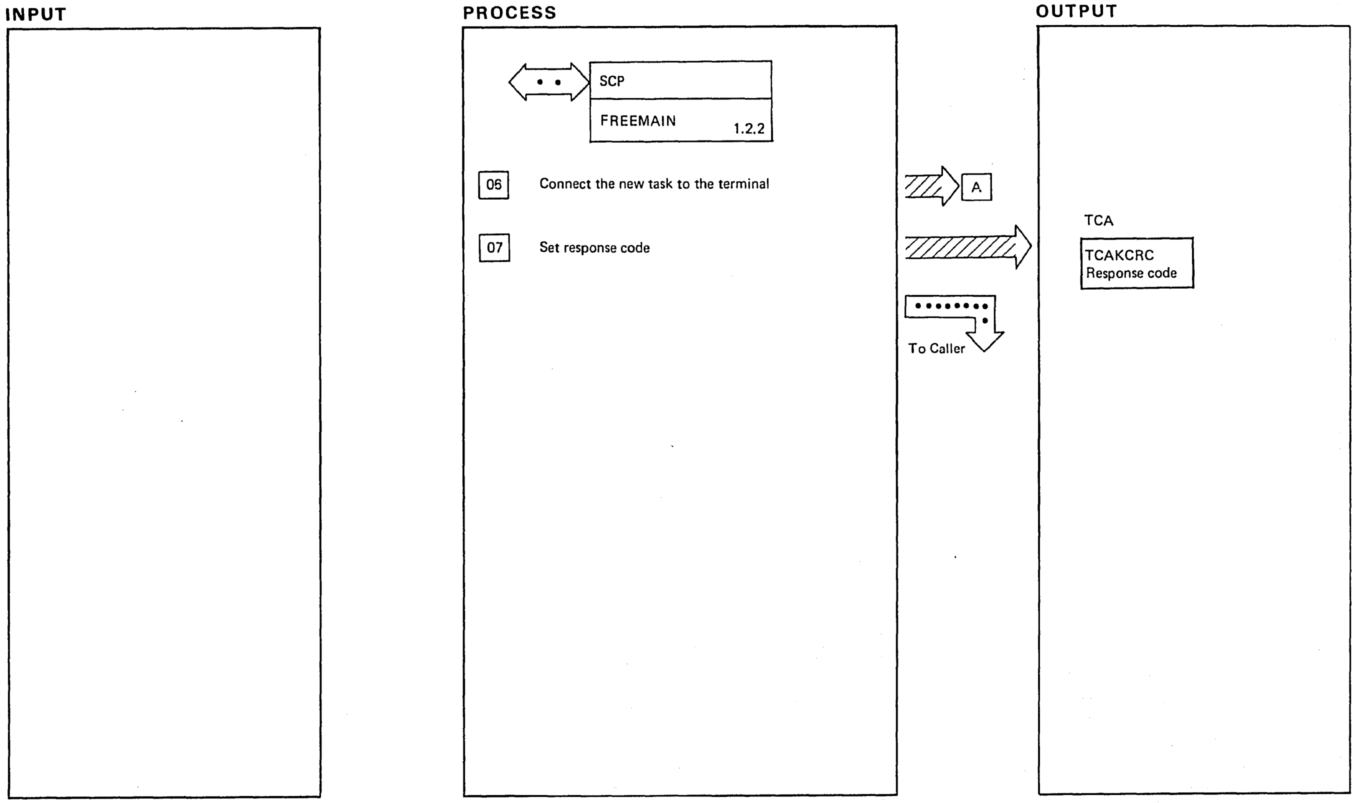
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 If a match is not found, a return code is set and control is returned to the caller. | | KCAVLSC | |
| 02 If the AID requires an operator check, the operator check field of the AID is compared with either TCTTEOI or TCTTEOCL, depending on the operator check code in the AID. If not equal, the Automatic Initiate indicator is reset in the TCTTE. If not equal, the indicator is left unchanged. | | | |
| 03 Terminal storage is obtained if none was previously acquired. If no storage is available, a negative response code is returned to the caller. | | KCAVLINT | |
| 04 If this AID represented a request by an existing task for a terminal, then that task is RESUMED, otherwise a Task Control ATTACH is issued. If unsuccessful, a negative response code is returned to caller. If successful, AID status is set to 'Task Initiated'. | | KCAVLGSB | 1.1.1 |
| 05 If the AID was not created by an Interval Control PUT request, the AID is unchained and its storage is released. For BMS-type AIDs, the data ID field is added to the new TCA. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Declare Resource Availability

Diagram — 1.1.13.2–01

Page 2 of 2



06 Connect the new task to the terminal

07 Set response code

Declare Resource Availability

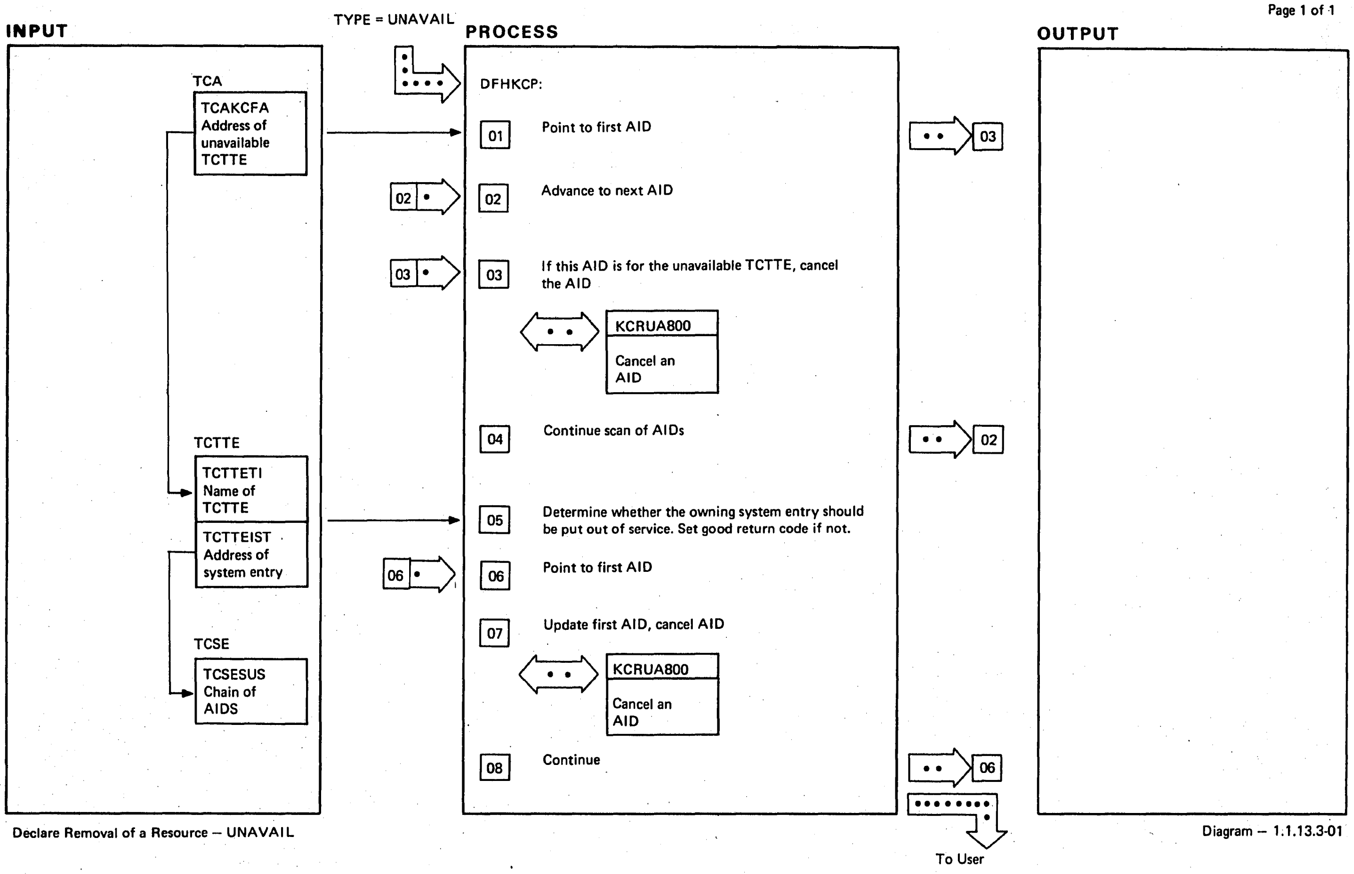
Diagram – 1.1.13.2–02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 06 The response indicates whether or not the task was initiated. Control is returned to the caller. | KCTDDTN | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Declare Resource Availability

Diagram – 1.1.13.2–02



Declare Removal of a Resource — UNAVAIL

Diagram — 1.1.13.3-01

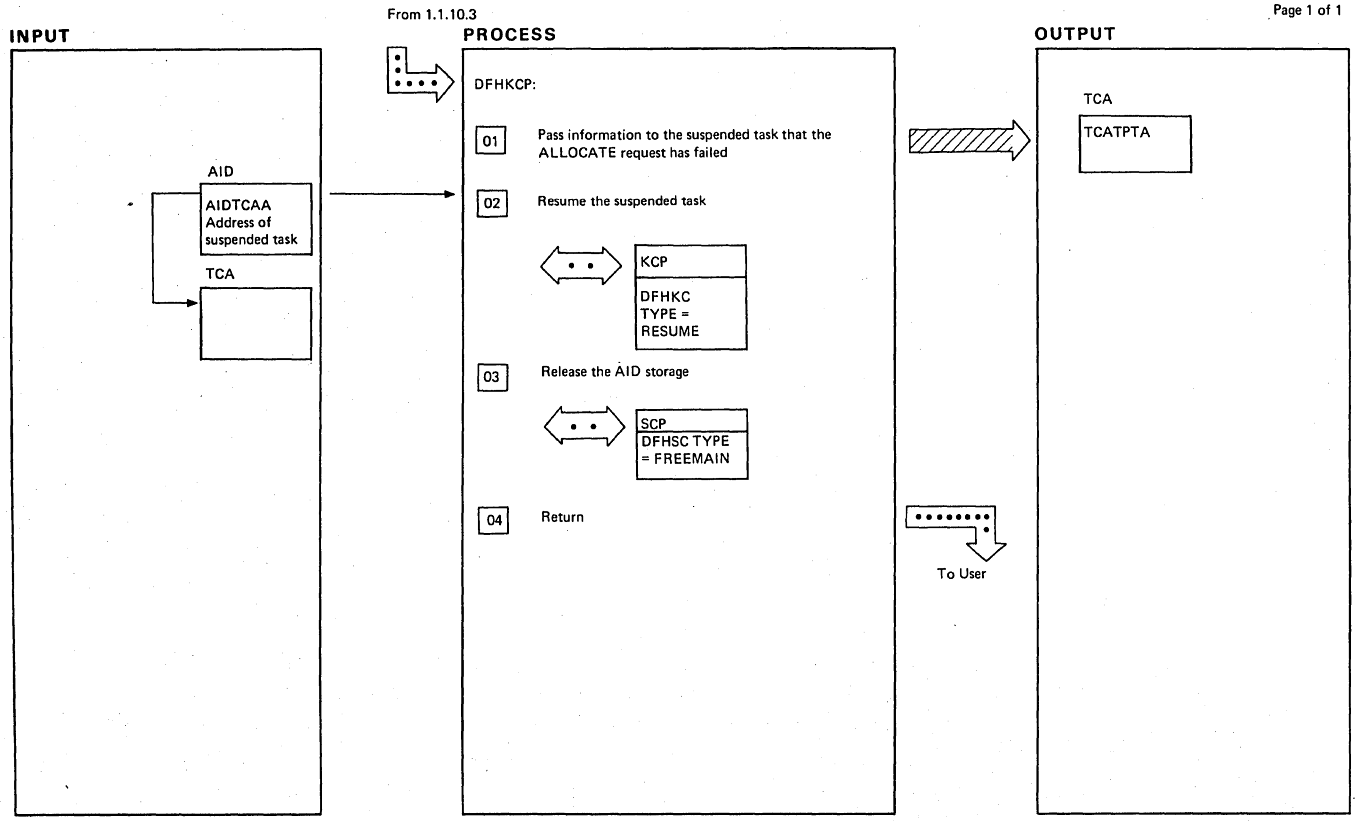
| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 01 | The TCTTE points to the owning system entry (TCSE). The system entry anchors the chain of AIDs | | KCRUAN | |
| 05 | If there are no available primary or secondary conversations, the system entry should be put out of service | | KCRUA040 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Declare Removal of a Resource — UNAVAIL

Diagram - 1.1.13.3-01

Page 1 of 1



01 Pass information to the suspended task that the ALLOCATE request has failed

02 Resume the suspended task

03 Release the AID storage

04 Return

Cancel on AID

Diagram – 1.1.13.3.1-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Field TCATPTA in the suspended tasks TCA is cleared. This indicates a failure to allocate a TCTTE | | KCRUA800 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Cancel on AID

Diagram – 1.1.13.3.1-01

Storage Management

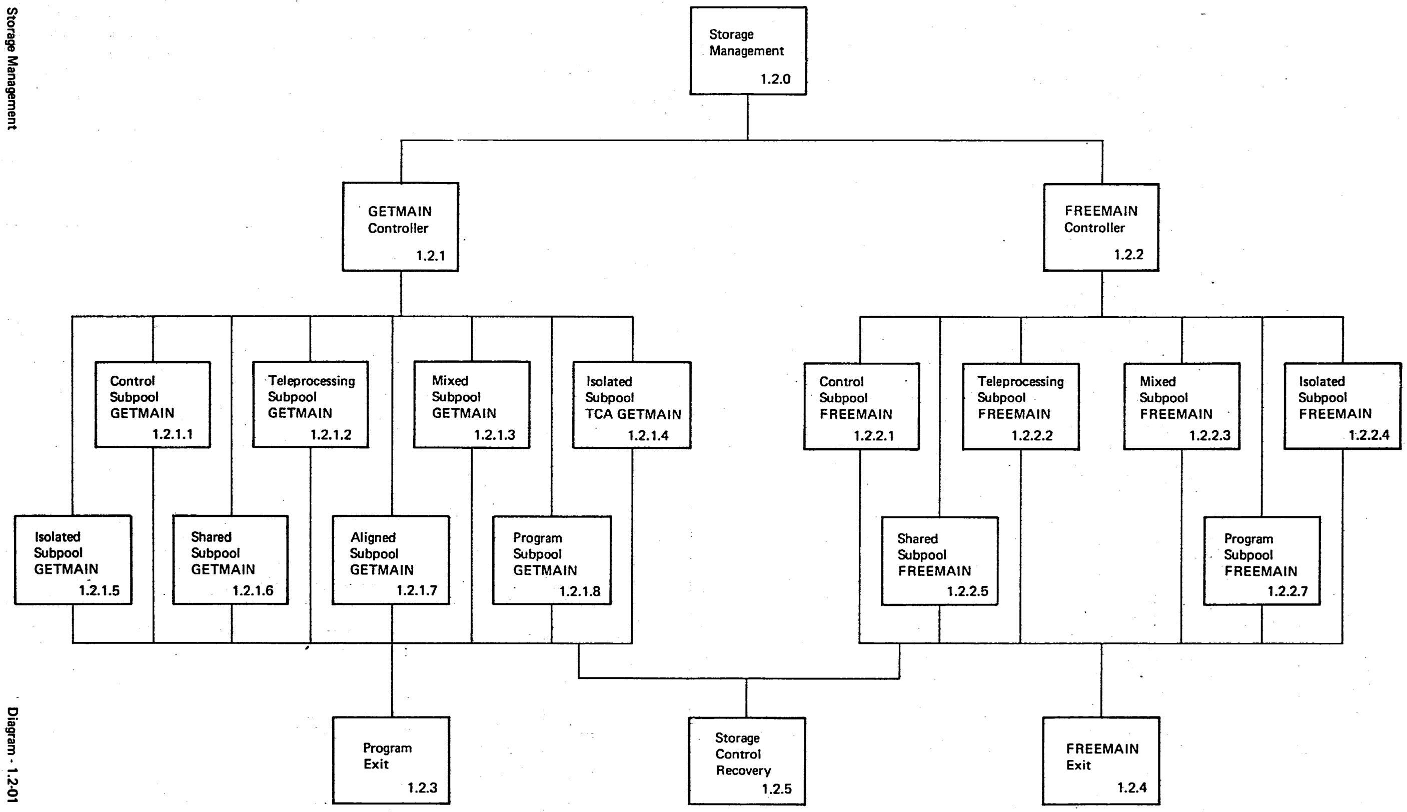
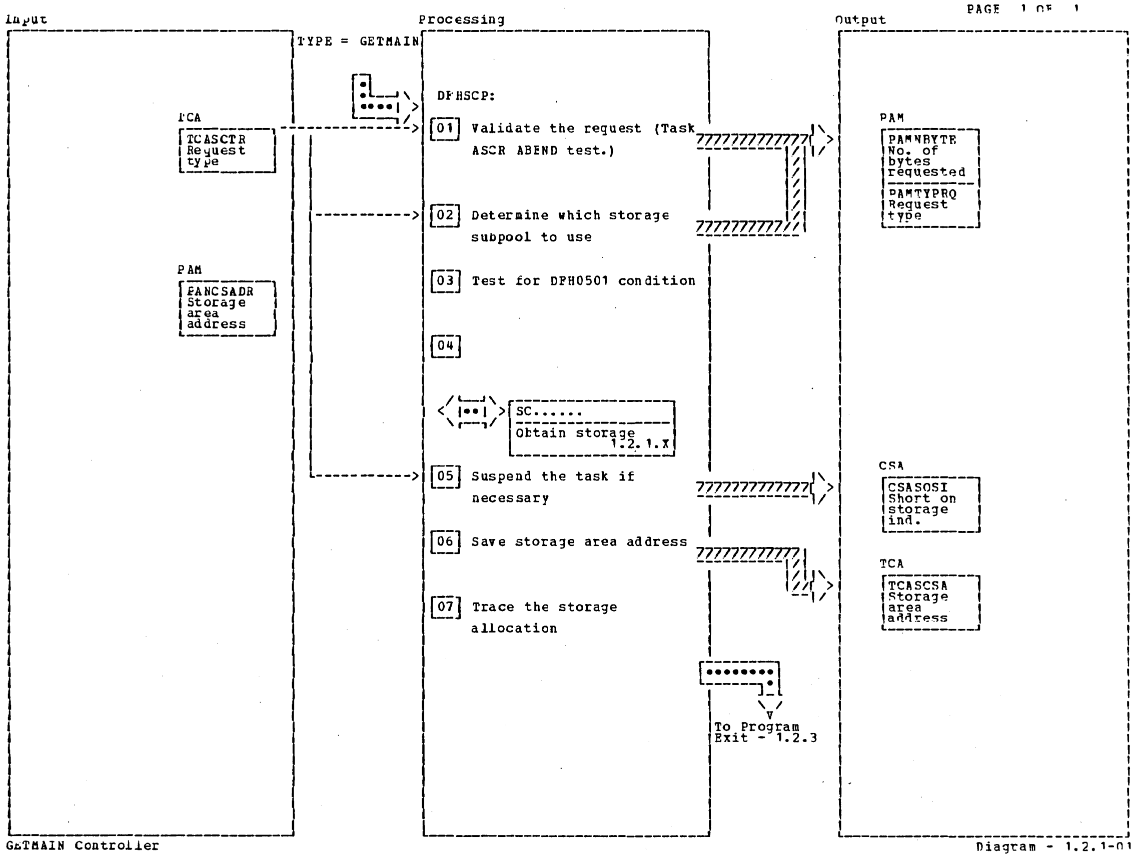

Storage Management 1.2.0
GETMAIN Controller 1.2.1
FREEMAIN Controller 1.2.2
Control Subpool GETMAIN 1.2.1.1
Teleprocessing Subpool GETMAIN 1.2.1.2
Mixed Subpool GETMAIN 1.2.1.3
Isolated Subpool TCA GETMAIN 1.2.1.4
Isolated Subpool GETMAIN 1.2.1.5
Shared Subpool GETMAIN 1.2.1.6
Aligned Subpool GETMAIN 1.2.1.7
Program Subpool GETMAIN 1.2.1.8
Control Subpool FREEMAIN 1.2.2.1
Teleprocessing Subpool FREEMAIN 1.2.2.2
Mixed Subpool FREEMAIN 1.2.2.3
Isolated Subpool FREEMAIN 1.2.2.4
Shared Subpool FREEMAIN 1.2.2.5
Program Subpool FREEMAIN 1.2.2.7
Program Exit 1.2.3
Storage Control Recovery 1.2.5
FREEMAIN Exit 1.2.4

Diagram - 1.2.01


PAGE 1 OF 1
Input
Processing
Output
TYPE = GETMAIN
TCA
TCASCTR Request type
PAM
PAMCSADR Storage area address
DFHSCP:
01 Validate the request (Task ASCR ABEND test.)
02 Determine which storage subpool to use
03 Test for DFH0501 condition
04
SC......
Obtain storage 1.2.1.X
05 Suspend the task if necessary
06 Save storage area address
07 Trace the storage allocation
To Program Exit - 1.2.3
PAM
PAMNBYTE No. of bytes requested
PAMTYPRQ Request type
CSA
CSASOSI Short on storage ind.
TCA
TCASCSA Storage area address

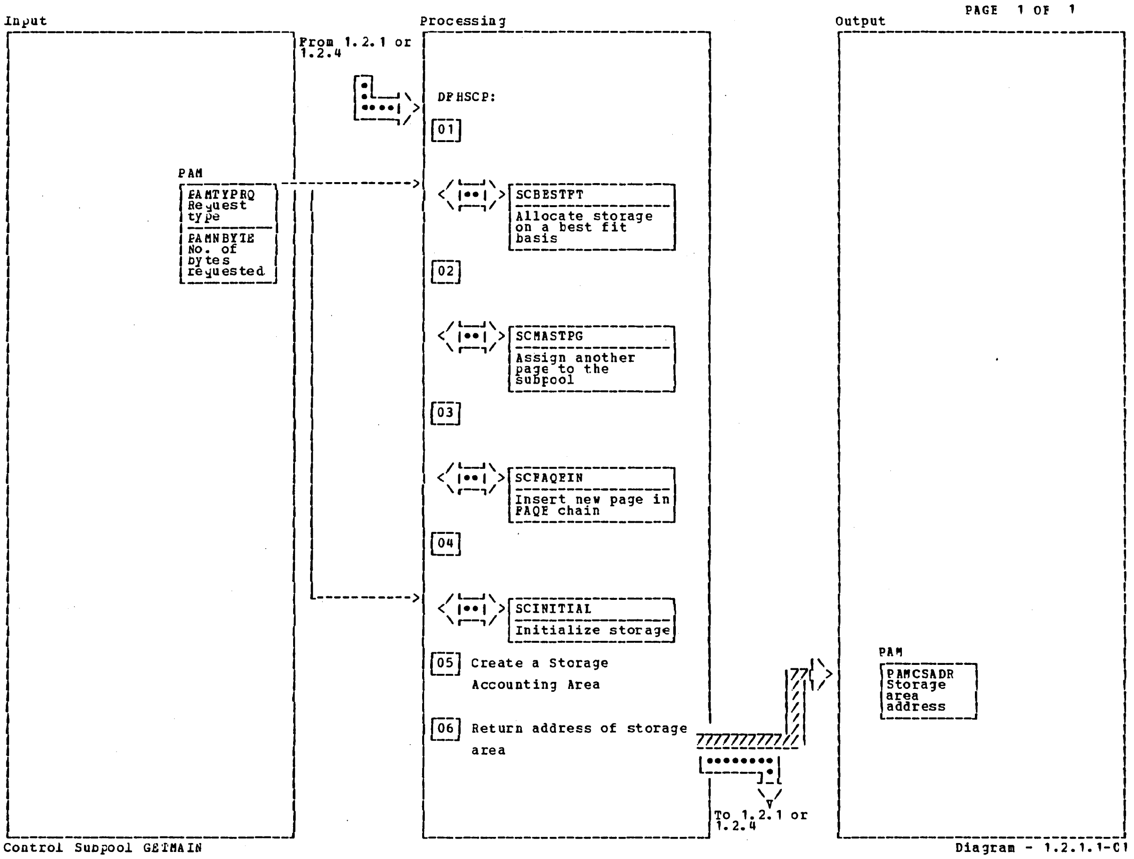
GETMAIN Controller

Diagram - 1.2.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If the number of bytes requested is zero or exceeds the maximum value, the task is abnormally terminated with a termination code of ASCR. | | SCSFGETM | | | | | |
| 02 The type of storage requested determines which subpool is used to obtain it. When the storage requested will be chained off the TCA chain, the storage is allocated from the same subpool as the TCA. | | | | | | | |
| 04 A branch is made to the required subpool GETMAIN routine to satisfy the request. | | | | | | | |
| 05 If the request was unconditional and could not be satisfied, the task is suspended. The FREEMAIN Exit routine will retry the request when some storage is freed. | | SCQUEUE | 1.1.7<br>1.2.4 | | | | |
| 07 The GETMAIN request is traced with a X'C8' trace entry. | | SCTRACEG | 2.1.4 | | | | |

GETMAIN Controller

Diagram - 1.2.1-01


PAGE 1 OF 1
Input
Processing
Output
From 1.2.1 or 1.2.4
DFHSCP:
01
PAM
PAMTYPRQ Request type
PAMNBYTE No. of bytes requested
SCBESTFT
Allocate storage on a best fit basis
02
SCMASTPG
Assign another page to the subpool
03
SCFAQEIN
Insert new page in FAQE chain
04
SCINITIAL
Initialize storage
05 Create a Storage Accounting Area
06 Return address of storage area
PAM
PAMCSADR Storage area address
To 1.2.1 or 1.2.4

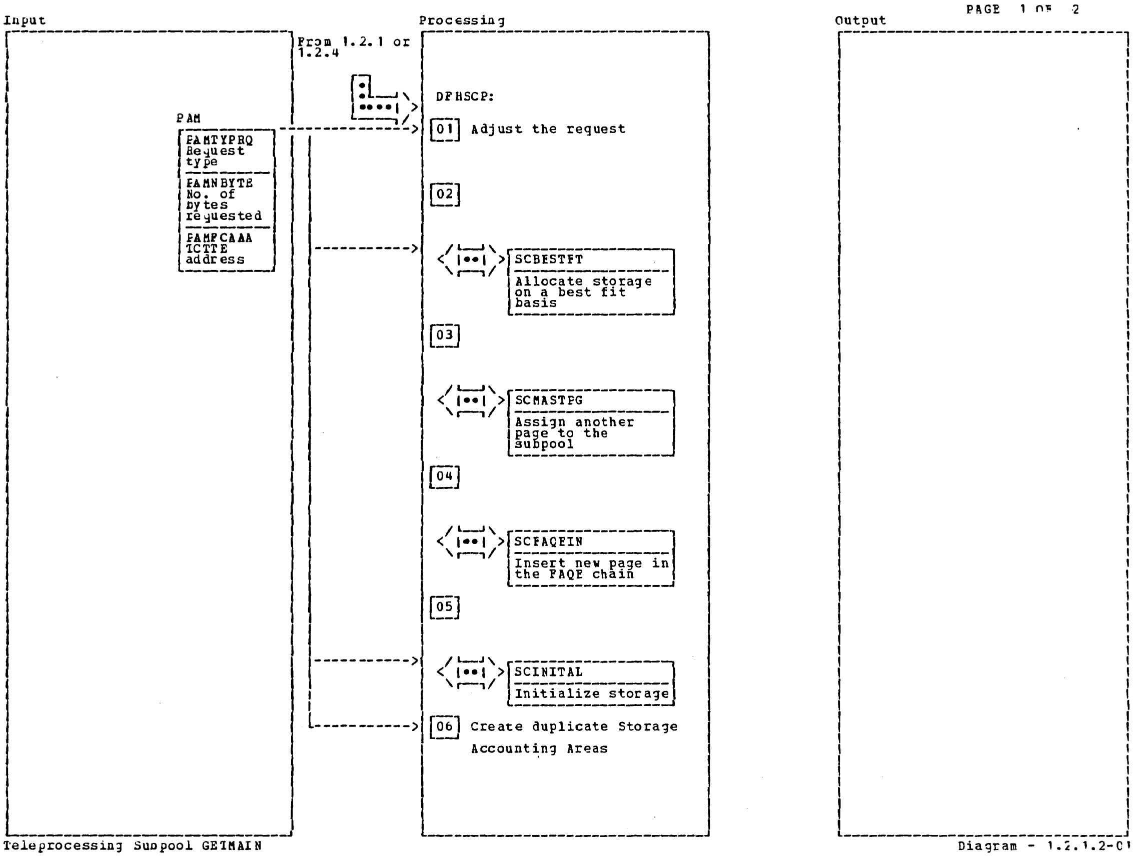
Control Subpool GETMAIN

Diagram - 1.2.1.1-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 An attempt is made to allocate storage from existing FAQEs on a best fit basis. | SCBESTFT | SCCTLGBF | | | | | |
| 02 Additional pages are assigned to the subpool only when the request cannot be satisfied from existing FAQEs. | SCMASTPG | | | | | | |
| 03 When new pages are assigned to the subpool, they are treated as a Free Area Queue Element (FAQE) and are inserted into the FAQE chain. The request for storage is then retried. | | SCFAQEIN | | | | | |
| 04 The area is initialized to the bit pattern given by the INITIMG= operand if the operand was specified. A four-byte Storage Accounting Area is built in the front of the area. | SCINITAL | SCCTLGOK | | | | | |
| 05 The address of the area allocated is returned to the calling routine. If no area could be allocated, an address of zeros is returned. | | SCCTLGEX | | | | | |

Control Subpool GETMAIN

Diagram - 1.2.1.1-01



Teleprocessing Subpool GETMAIN

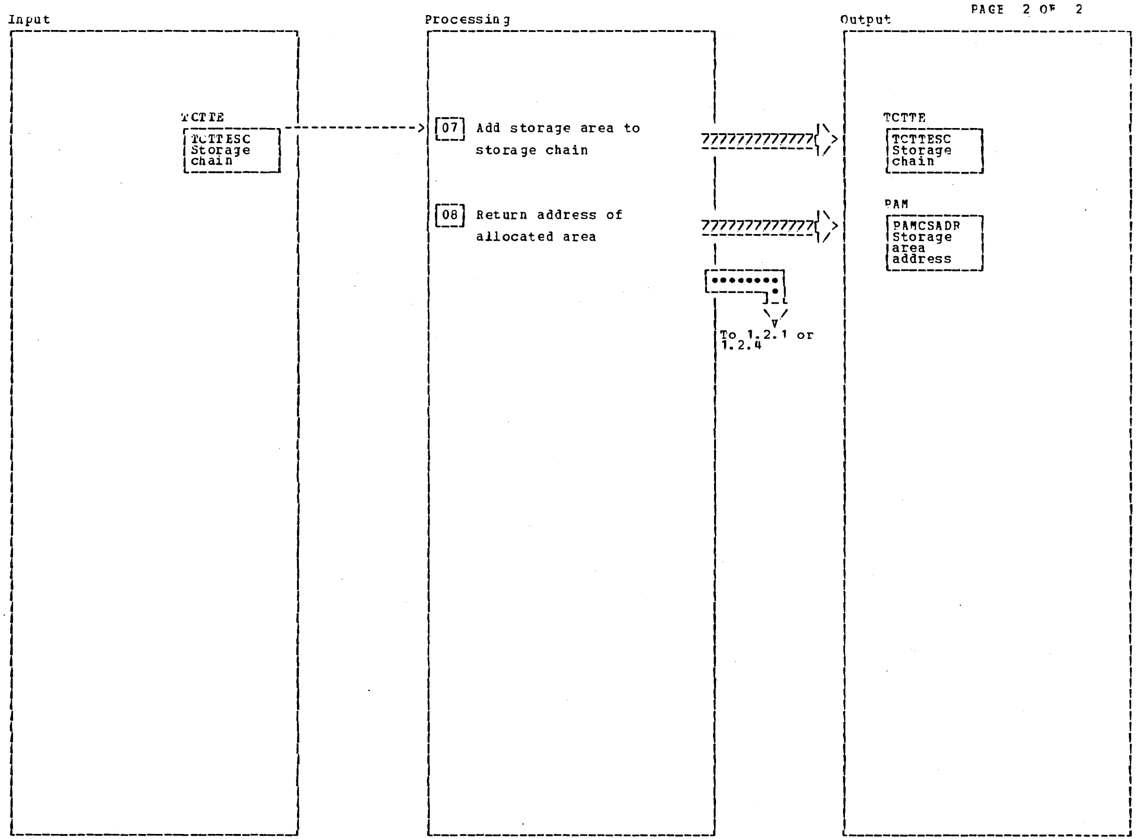
Diagram - 1.2.1.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The number of bytes requested is adjusted to include the duplicate Storage Accounting Areas.<br>A. For TIOA, adjustment is request + 13 rounded to double-doubleword.<br>B. For LIOA, adjustment is request + 8 rounded to double-doubleword. | | SCTELGET | |
| 02 An attempt is made to allocate storage from existing FAQEs on a best fit basis. | SCBESTFT | SCTELGBF | |
| 03 Additional pages are assigned to the subpool only when the request cannot be satisfied from existing FAQEs. | SCMASTPG | | |
| 04 When new pages are assigned to the subpool, they are treated as Free Area Queue Element (FAQE), and are inserted into the FAQE chain. The request for storage is then retried. | SCFAQEIN | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 05 The area is initialized to the bit pattern given by the INITIMG= operand if the operand was specified. | SCINITAL | SCTELGOK | |
| 06 Duplicate Storage Accounting Areas are built at the front and the back of the area. | | SCTELGE1 | |

Teleprocessing Subpool GETMAIN

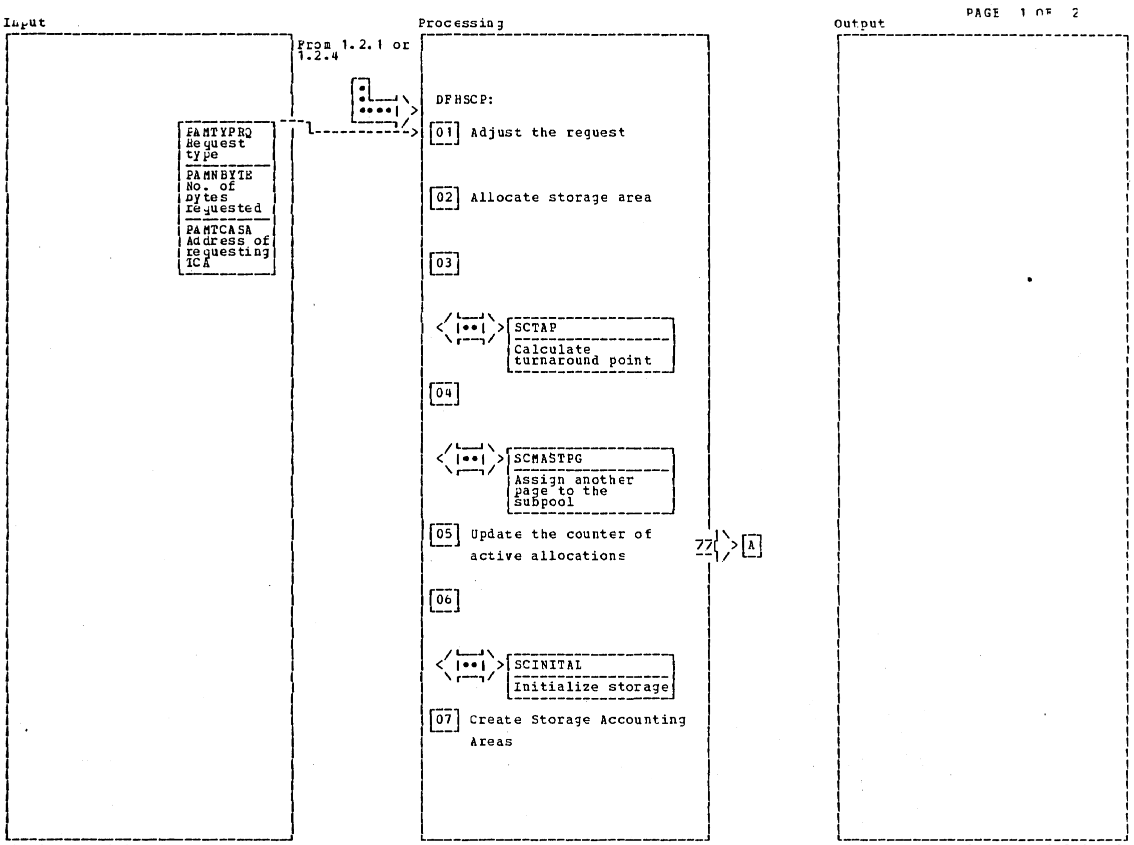
Diagram - 1.2.1.2-01



Teleprocessing Subpool GETMAIN

Diagram - 1.2.1.2-C2

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 07 If the area is a TIOA, it is added to the TCTTE storage chain. If the area is a LIOA, the chain points to itself. | | | | | | | |
| 08 The address of the area allocated is returned to the calling routine. If no area could be allocated, an address of zeros is returned. | | SCTELGEX | | | | | |


PAGE 1 OF 2
Input
Processing
Output
From 1.2.1 or 1.2.4
PAMTYPRQ Request type
PAMNBYTE No. of bytes requested
PAMTCASA Address of requesting TCA
DFHSCP:
01 Adjust the request
02 Allocate storage area
03
SCTAP
Calculate turnaround point
04
SCMASTPG
Assign another page to the subpool
05 Update the counter of active allocations
77
A
06
SCINITAL
Initialize storage
07 Create Storage Accounting Areas

Mixed Subpool GETMAIN

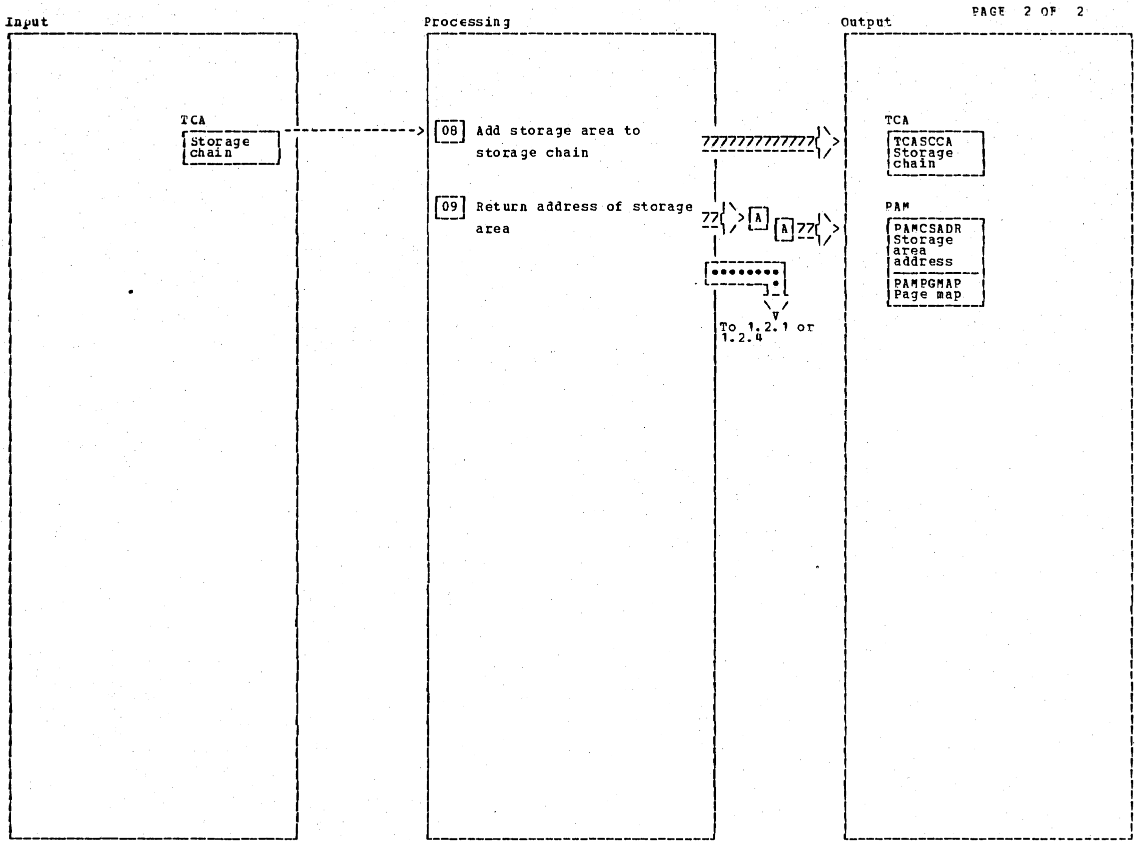
Diagram - 1.2.1.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The number of bytes requested is adjusted to include the duplicate Storage Accounting Areas.<br>A. For TCA, adjustment is request + 8 rounded to double-doubleword.<br>B. Non-TCA, adjustment is request + 16 rounded to double-doubleword. | | SCMIXTCA | |
| 02 The storage area is obtained from the next available location, and the pointer to the next available location is adjusted to point to the end of the area just allocated. | | SCMIXGRD | |
| 03 When additional pages are needed, the turnaround point is checked to determine whether the allocating of pages should continue in an upward direction or to turn around and start again from the low end of dynamic storage. | SCTAP | SCMIXGAP | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 04 New pages are allocated as needed, based on the turnaround point. | SCMASTPG | | |
| 05 A count of the number of active allccations is kept for each page assigned to the mixed subpool. This count is kept in the second half of the page map. | | SCMIXGF1 | |
| 06 The area is initialized to the bit pattern given by the INITIMG= operand if the operand was specified. | SCINITAL | SCMIXGL1 | |
| 07 Duplicate Storage Accounting Areas are built into the front and the back of the area. | | | |

Mixed Subpool GETMAIN

Diagram - 1.2.1.3-01



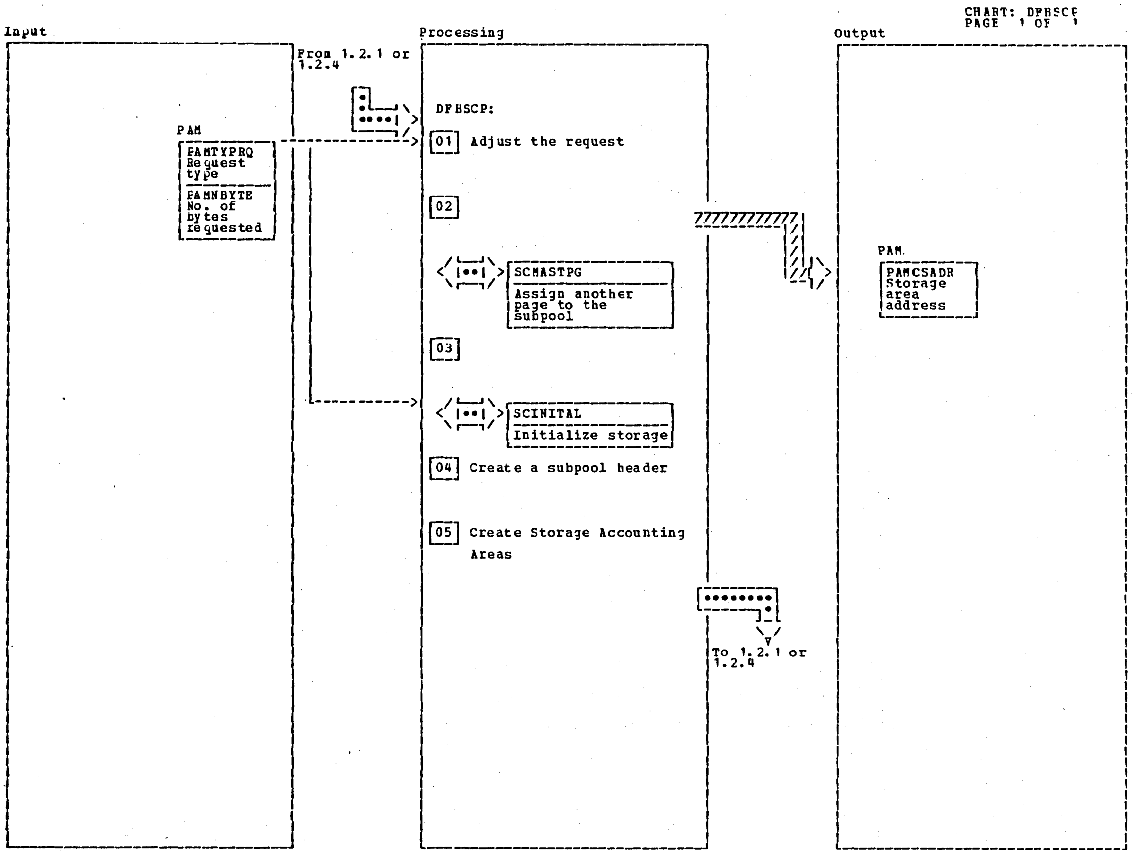
Mixed Subpool GETMAIN

Diagram - 1.2.1.3-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 08 The area is added to the TCA storage chain. (The chain is a push-down queue.) If a TCA is being allocated, the chain field is initialized to point to the front of the TCA. | | SCMIXGSC | | | | | |
| 09 The address of the area allocated is returned to the calling routine. If no area could be allocated, an address of zeros is returned. | | SCMIXGEX | | | | | |

Mixed Subpool GETMAIN

Diagram - 1.2.1.3-02



Isolated Subpool TCA GETMAIN

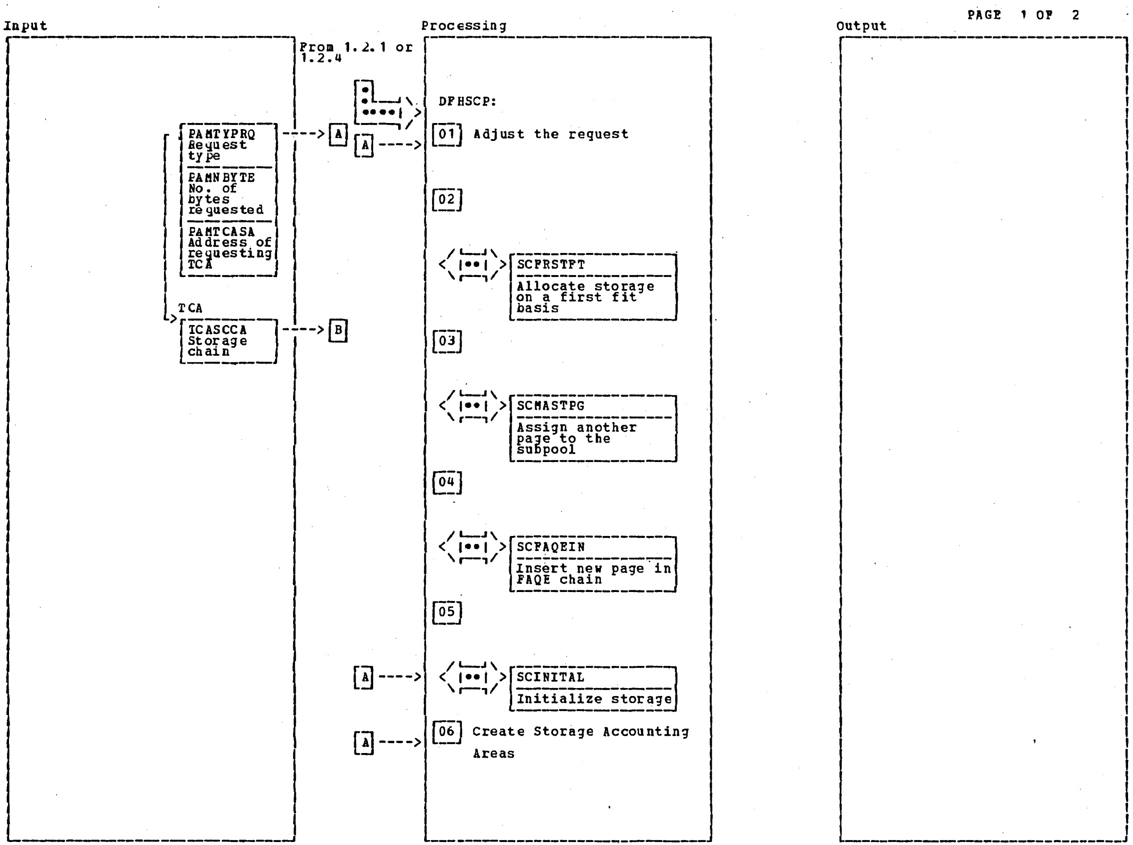
Diagram - 1.2.1.4-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Eight bytes are added to the length to allow for the duplicate Storage Accounting Areas. The request is converted to the number of pages required. This value is compared to the number of pages requested which is specified in the PCT. The higher number is stored as the number of pages required. | | SCISOTCA | |
| 02 Since isolated tasks have exclusive control of entire pages, new pages are assigned for the number of pages computed in step 1. | SCMASTPG | | |
| 03 The TCA is initialized to binary zeros. All pages allocated are marked in the PAM as part of the initial allocation. | SCINITAL | | |
| 04 A header containing forward and backward PAQE pointers is created in the TCA. The remainder of the area that is not a part of the TCA is made into a PAQE. | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 05 Duplicate Storage Accounting Areas are built at the front and the back of the TCA. The storage chain field is initialized to point to the front of the TCA. | | | |

Isolated Subpool TCA GETMAIN

Diagram - 1.2.1.4-01


Input
Processing
Output
PAGE 1 OF 2
From 1.2.1 or 1.2.4
PAMTYPRQ Request type
PAMNBYTE No. of bytes requested
PAMTCASA Address of requesting TCA
TCA
TCASCCA Storage chain
A
B
DFHSCP:
01 Adjust the request
02
SCFRSTFT
Allocate storage on a first fit basis
03
SCMASTPG
Assign another page to the subpool
04
SCFAQEIN
Insert new page in FAQE chain
05
SCINITAL
Initialize storage
06 Create Storage Accounting Areas

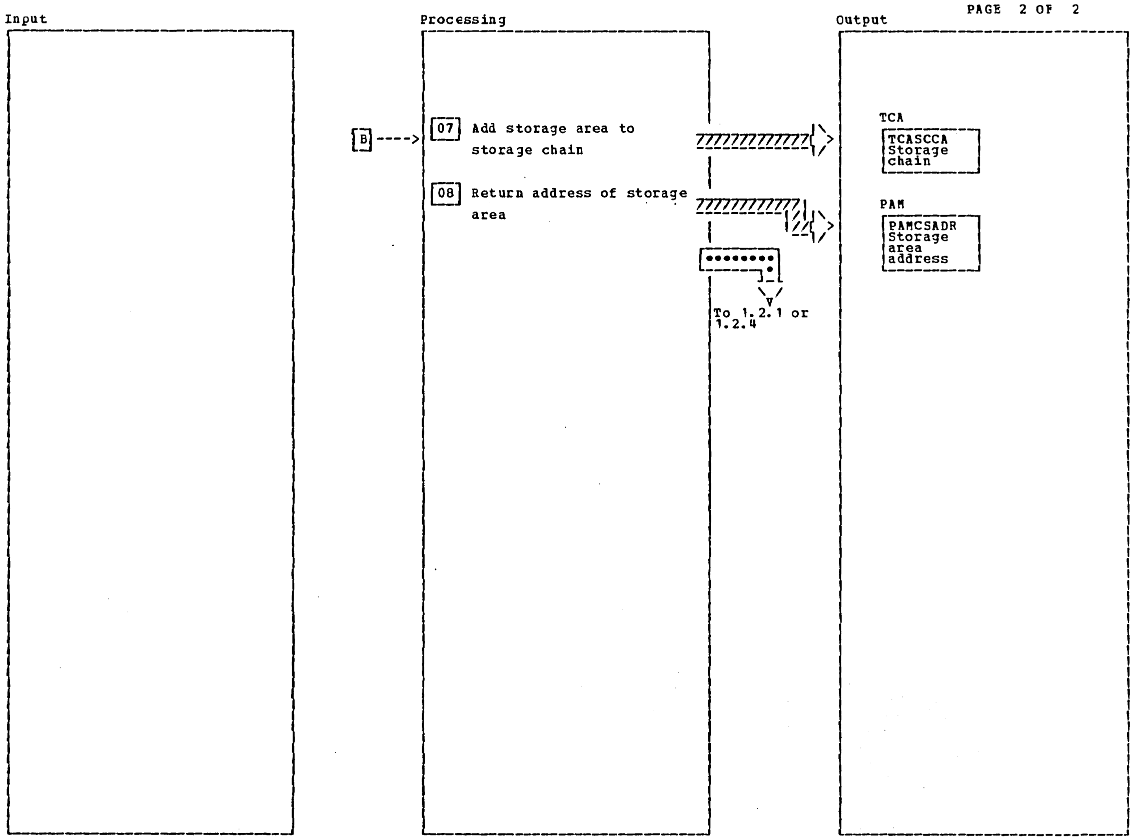
Isolated Subpool GETMAIN

Diagram - 1.2.1.5-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The number of bytes requested is adjusted to include the duplicate Storage Accounting Areas.<br>Adjustment is, request + 16 rounded to double - doubleword. | | SCISOGET | |
| 02 An attempt is made to allocate from existing FAQEs using first free area large enough to satisfy request. | SCFRSTFT | SCISOGFF | |
| 03 Additional pages are assigned to the subpool only if the request cannot be satisfied from existing FAQEs. | SCMASTPG | | |
| 04 When new pages are assigned to the subpool, they are treated as a Free Area Queue Element (FAQE) and are inserted in the FAQE chain. The request for storage is then retried. | SCFAQEIN | | |
| 05 The area is initialized to the bit pattern given by the INITIMG= operand if the operand was specified. | SCINITAL | SCTSOGOK | |
| 06 Duplicate Storage Accounting Areas are built at the front and the back of the storage area. | | SCISOGE1 | |

Isolated Subpool GETMAIN

Diagram - 1.2.1.5-01


PAGE 2 OF 2
Input
Processing
Output
B
07 Add storage area to storage chain
08 Return address of storage area
To 1.2.1 or 1.2.4
TCA
TCASCCA Storage chain
PAM
PAMCSADR Storage area address

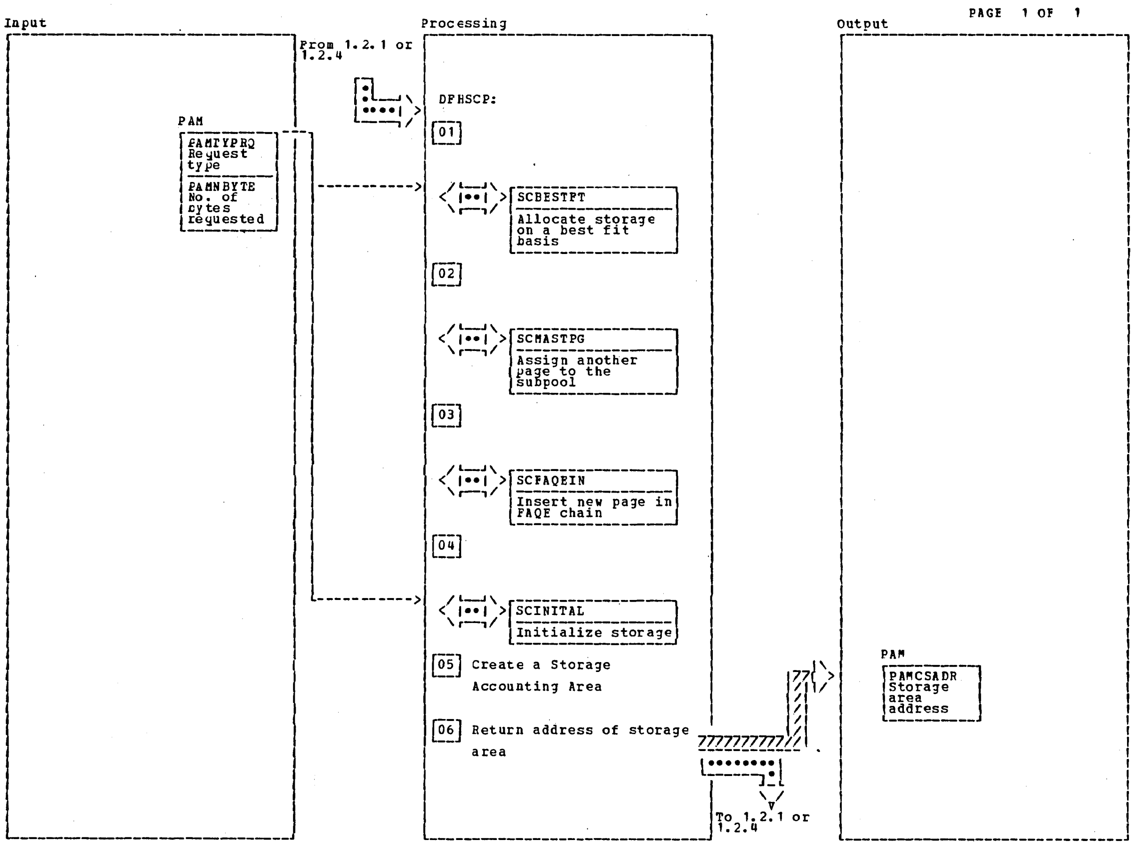
Isolated Subpool GETMAIN

Diagram - 1.2.1.5-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 07 The storage area is added to the TCA storage chain. ( The chain is a push-down queue.) | | | | | | | |
| 08 The address of the allocated storage area is returned to the calling routine. If no area could be allocated, an address of zeros is returned. | | SCISOGEX | | | | | |

Isolated Subpool GETMAIN

Diagram - 1.2.1.5-02


PAGE 1 OF 1
Input
Processing
Output
From 1.2.1 or 1.2.4
DFHSCP:
PAM
PAMTYPRQ Request type
PAMNBYTE No. of bytes requested
01
SCBESTFT
Allocate storage on a best fit basis
02
SCMASTPG
Assign another page to the subpool
03
SCFAQEIN
Insert new page in FAQE chain
04
SCINITAL
Initialize storage
05 Create a Storage Accounting Area
06 Return address of storage area
PAM
PAMCSADR Storage area address
To 1.2.1 or 1.2.4

Shared Subpool GETMAIN

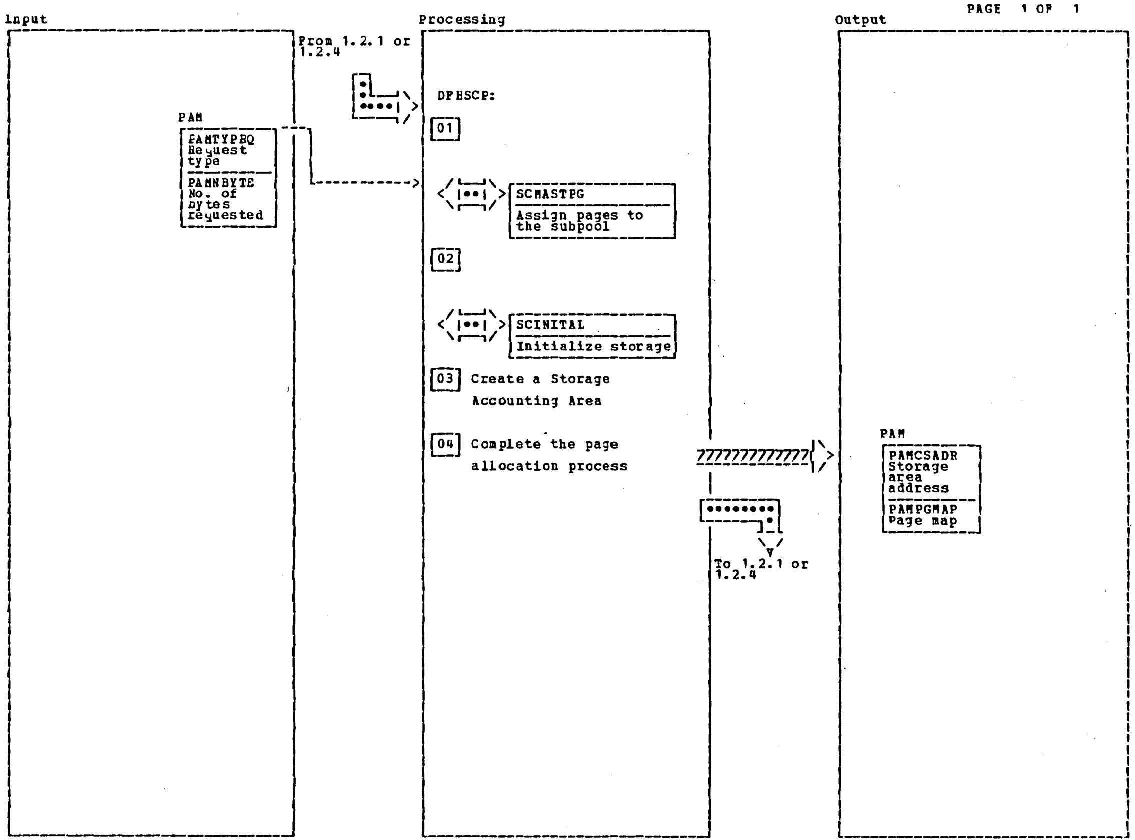
Diagram - 1.2.1.6-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 An attempt is made to allocate from existing FAQEs on a best fit basis. | SCBESTFT | SCSHRGBF | |
| 02 Additional pages are assigned to the subpool only when a request cannot be satisfied from existing FAQEs. | SCMASTPG | | |
| 03 When new pages are assigned to the subpool, they are each treated as a Free Area Queue Element (FAQE) and are inserted in the FAQE chain. The request for storage is then retried. | SCFAQEIN | | |
| 04 The area is initialized to the bit pattern given by the INITIMG= operand if the operand was specified. | SCINITAL | SCSHRGOK | |
| 05 A four-byte Storage Accounting Area is built in the front of the area. | | | |
| 06 The address of the allocated area is returned to the calling routine. If no area could be allocated, an address of zeros is returned. | | SCSHRGEX | |

Shared Subpool GETMAIN

Diagram - 1.2.1.6-01

PAGE 1 OF 1


Input
Processing
Output
From 1.2.1 or 1.2.4
PAM
PAMTYPRQ Request type
PAMNBYTE No. of bytes requested
DFBSCP:
01
SCMASTPG
Assign pages to the subpool
02
SCINITAL
Initialize storage
03 Create a Storage Accounting Area
04 Complete the page allocation process
PAM
PAMCSADR Storage area address
PAMPGMAP Page map
To 1.2.1 or 1.2.4

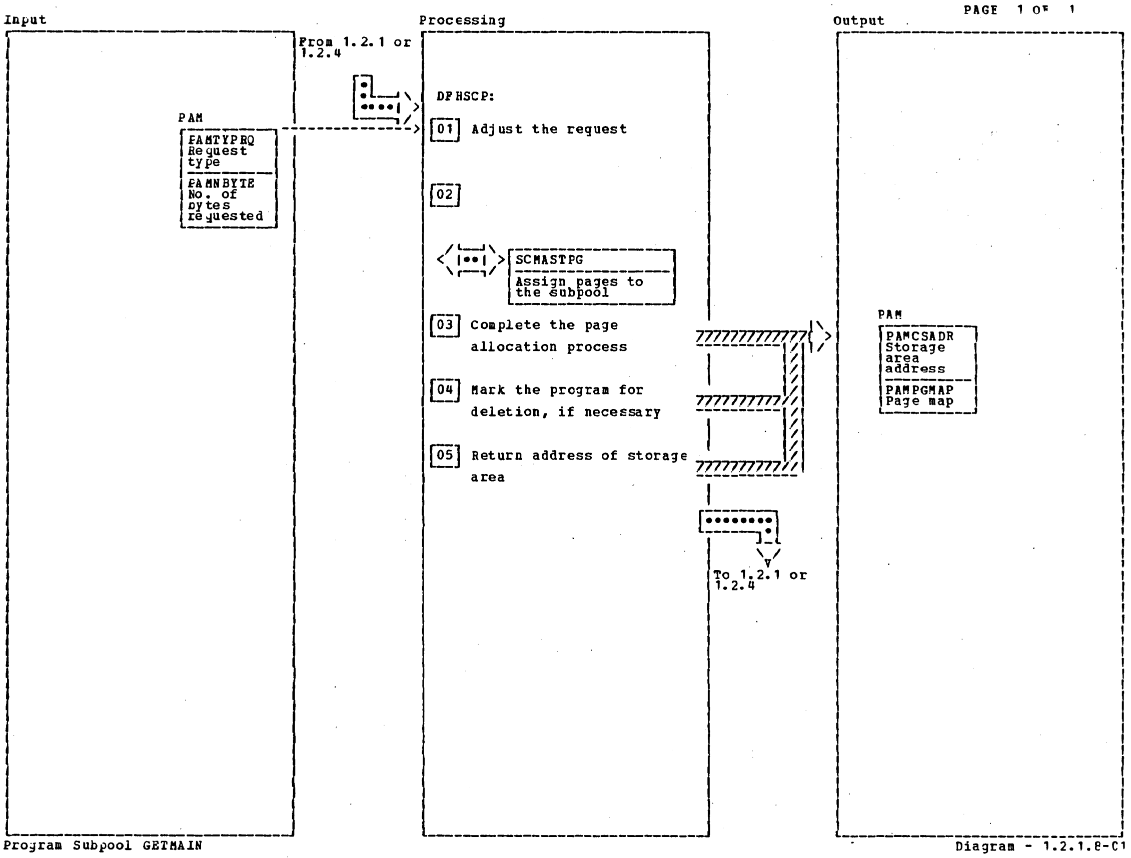
Aligned Subpool GETMAIN

Diagram - 1.2.1.7-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Since all areas in this subpool are allocated only in full pages, the requested number of pages are assigned to this subpool. | SCMASTPG | SCALGGET | | | | | |
| 02 An area is initialized to the bit pattern given by the INITIMG= operand if the operand was specified. | SCINITAL | | | | | | |
| 03 A four-byte Storage Accounting Area is built at the front of the area, even though it is not needed by Storage Control.<br>This is provided as a diagnostic aid for convenience in reading dump. | | | | | | | |
| 04 The page allocation process is completed by adjusting the second half of the page map to reflect the allocation. | | | | | | | |

Aligned Subpool GETMAIN

Diagram - 1.2.1.7-C1

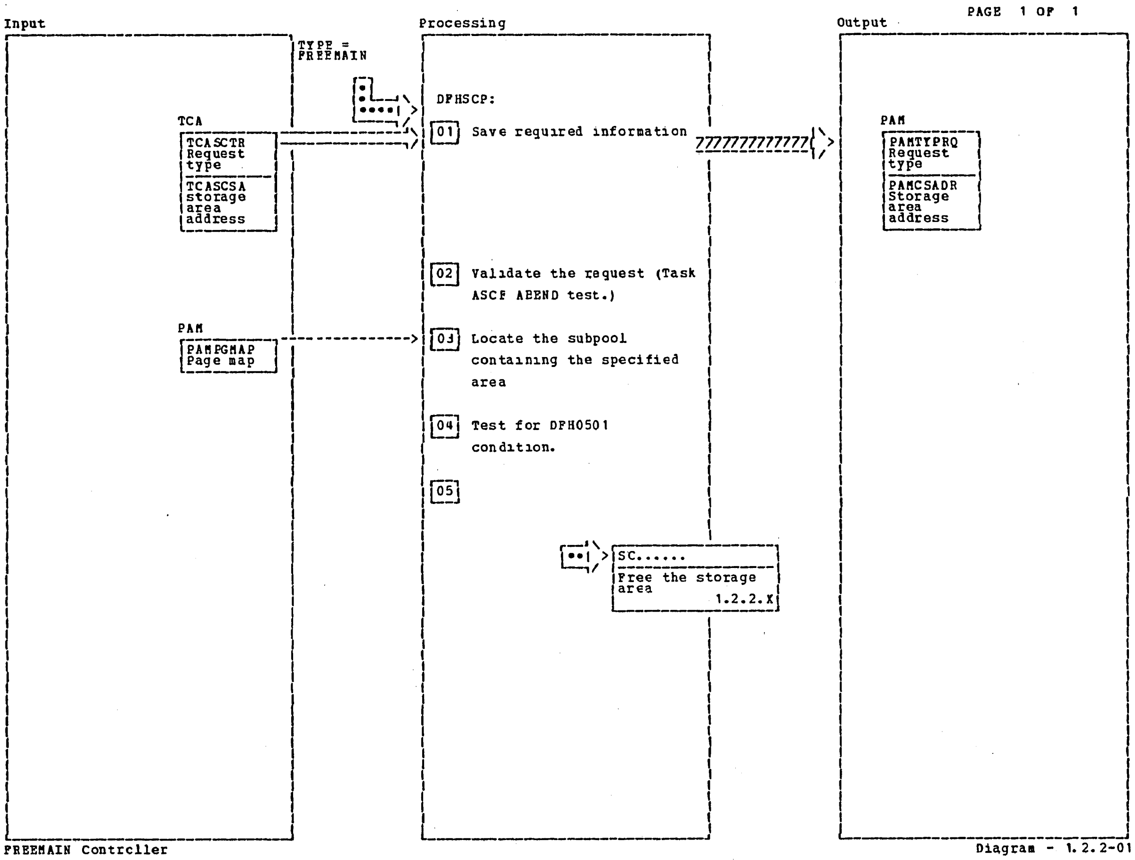

Input
Processing
Output
PAGE 1 OF 1
From 1.2.1 or 1.2.4
PAM
PAMTYPRQ Request type
PAMNBYTE No. of bytes requested
DFHSCP:
01 Adjust the request
02
SCMASTPG
Assign pages to the subpool
03 Complete the page allocation process
04 Mark the program for deletion, if necessary
05 Return address of storage area
To 1.2.1 or 1.2.4
PAM
PAMCSADR Storage area address
PAMPGMAP Page map
Program Subpool GETMAIN
Diagram - 1.2.1.8-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Program storage is requested in doublewords. The number of doublewords is converted to bytes. | | SCPGMGET | | | | | |
| 02 Program storage is allocated in contiguous full pages from the high end of dynamic storage. | SCMASTPG | | | | | | |
| 03 The number of pages associated with the allocation is kept in the second half of the PAM. | | | | | | | |
| 04 The program is marked for deletion if its allocated storage is so low in the dynamic storage that it is intermixed with pages assigned to other subpools. | | | | | | | |
| 05 The address of the allocated area is returned to the calling routine. If no area could be allocated, an address of zero is returned. | | | | | | | |

Program Subpool GETMAIN

Diagram - 1.2.1.8-C1

PAGE 1 OF 1



01 Save required information

02 Validate the request (Task ASCF ABEND test.)

03 Locate the subpool containing the specified area

04 Test for DFH0501 condition.

05

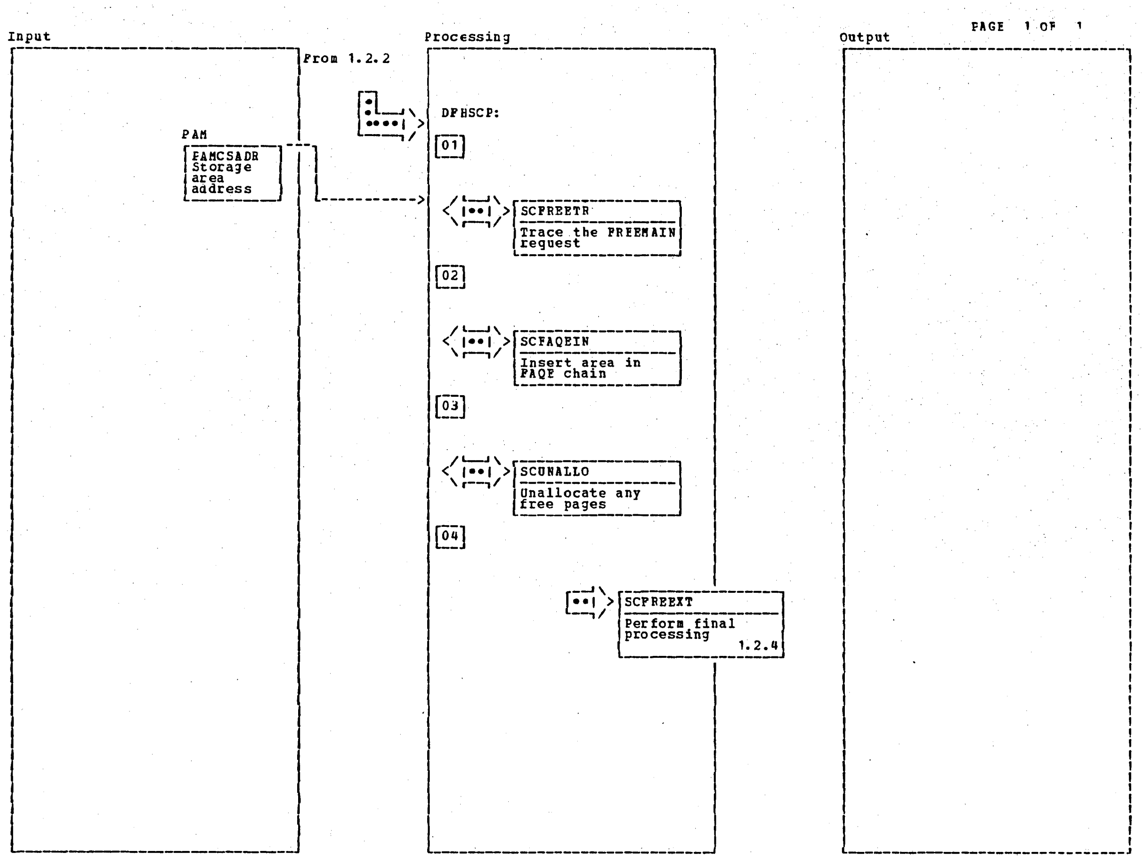
FREEMAIN Contrcller

Diagram - 1.2.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The request type and the address of the storage area to be freed are saved. | | | | | | | |
| 02 The address must be a double doubleword in the dynamic storage area. If the address is invalid, the task is abnormally terminated. | | SCSPFREE | | | | | |
| 03 The subpccl containing the specified area is determined from the page map in the PAM. | | | | | | | |
| 05 Control is passed to the appropriate subpool FREEMAIN routine to satisfy the request. | | | | | | | |

FREEMAIN Contrcller

Diagram - 1.2.2-01



Control Subpool FREEMAIN

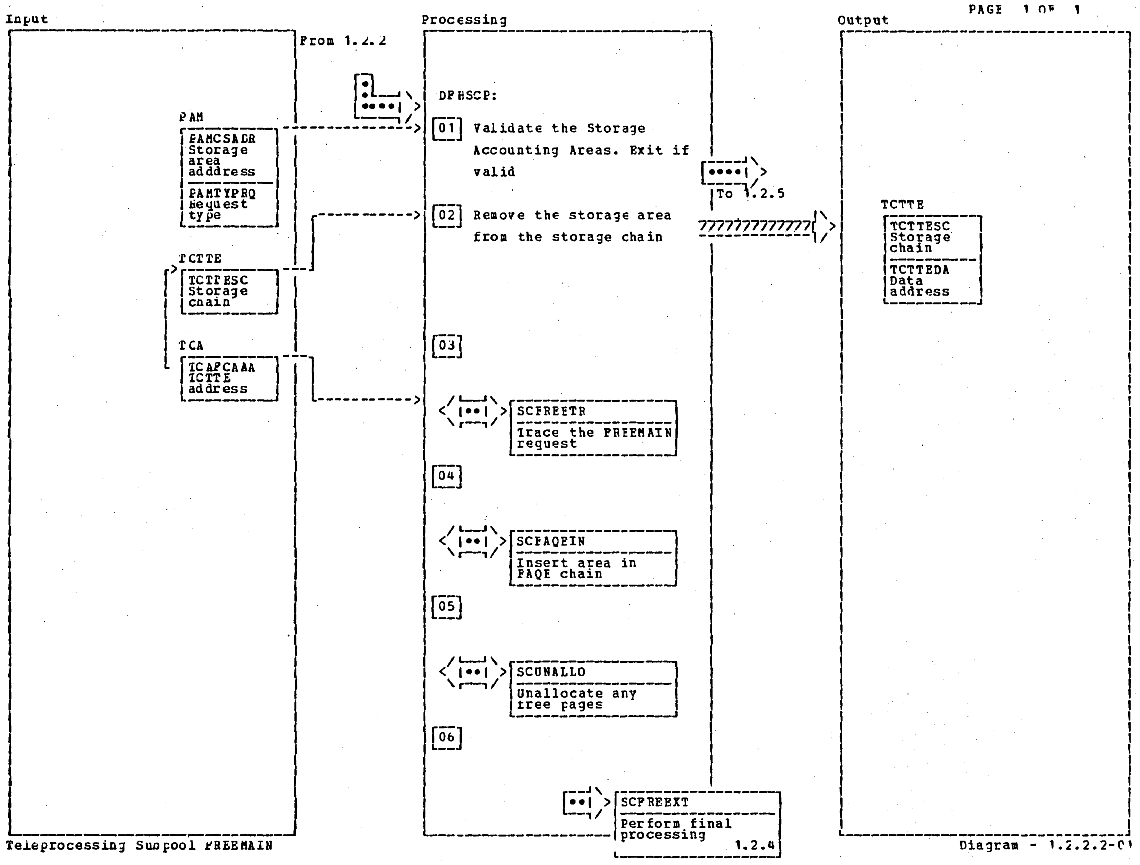
Diagram - 1.2.2.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The area to be freed is traced via a X'C9' trace entry. | SCFREETR | SCCILFRE | 2.1.1 |
| 02 The area to be freed is inserted in the Free Area Queue Element (FAQE) chain. | SCFAQEIN | | |
| 03 If the freed area had created an FAQE that spanned a full page, the page is removed from this subpool's page allocation and is now available to any subpool. | SCUNALLC | | |
| 04 The exit is to the common FREEMAIN exit. | SCFREEXT | | 1.2.4 |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Control Subpool FREEMAIN

Diagram - 1.2.2.1-01



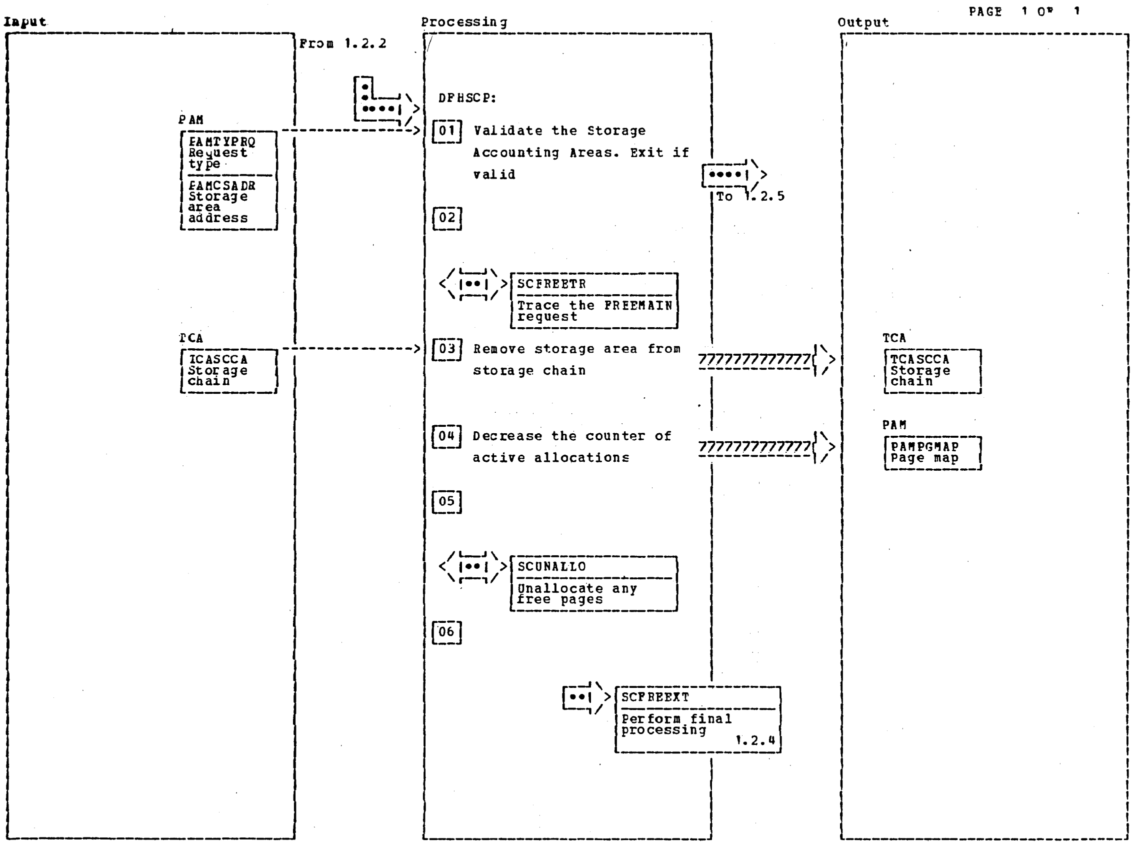
Teleprocessing Subpool FREEMAIN

Diagram - 1.2.2.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The duplicate Storage Accounting Areas are compared to see if either was destroyed. | | SCTELFBE | |
| 02 If the area being freed is a TIOA, it is removed from the terminal storage chain. | | | |
| 03 The area being freed is traced via a X'C9' trace entry. | SCFREETR | | 2.1.4 |
| 04 The area to be freed is inserted in the Free Area Queue Element (FAQE) chain. | SCFAQEIN | | |
| 05 If the freed area had created an FAQE that spanned a full page, the page is removed from this subpool's page allocation and is now available to any subpool. | SCUNALLO | | |
| 06 The exit is to the common FREEMAIN exit. | SCFREEXT | | 1.2.4 |
| NOTE: If all TIOAs are to be released (that is, RELEASE=ALL was specified), the above process is repeated for each area on the storage chain before exiting to the common FREEMAIN exit. | | | |

Teleprocessing Subpool FREEMAIN

Diagram - 1.2.2.2-01



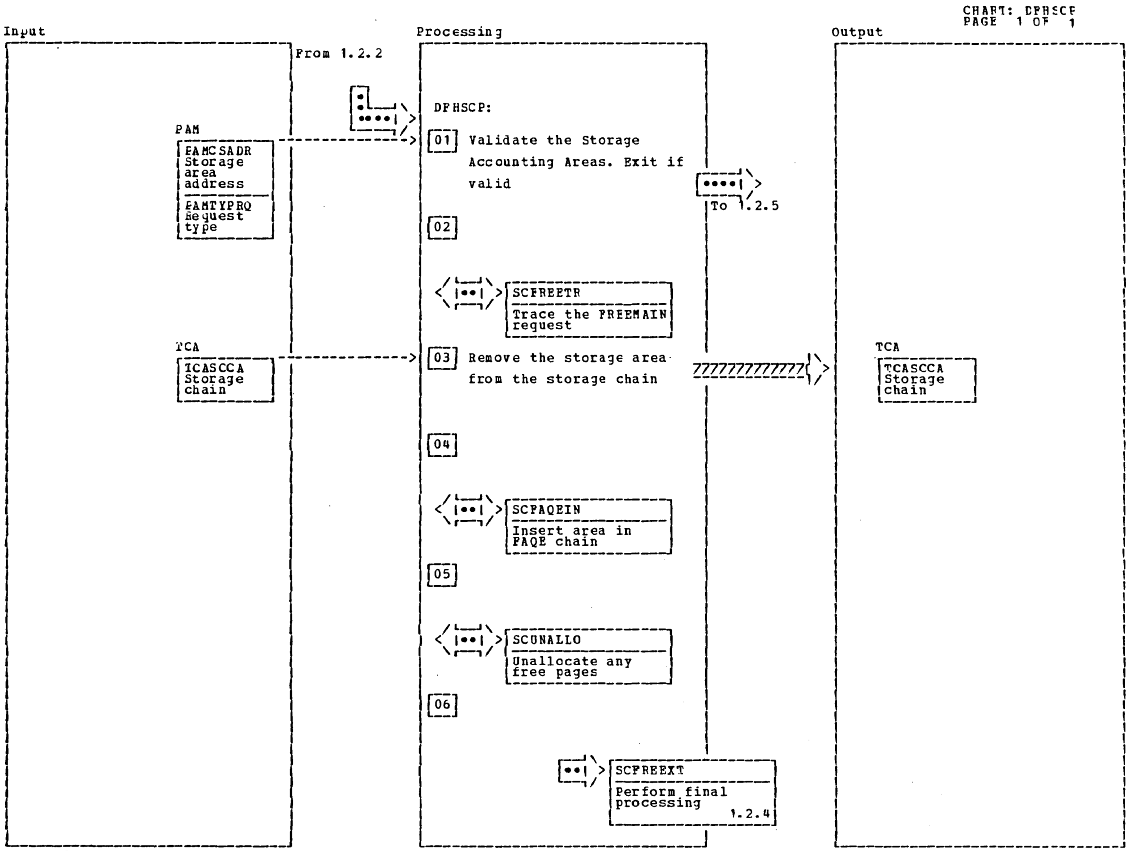
Mixed Subpool FREEMAIN

Diagram - 1.2.2.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The duplicate Storage Accounting Areas are compared to see if either was destroyed. | | SCMIXFRE | |
| 02 The area being freed is traced via a X'C9' trace entry. | SCFREETR | | 2.1.4 |
| 03 The storage chain is searched for the area being freed. When found, the area is removed from the chain. | | | |
| 04 The count of active allocations is decremented for every page touched by the area being freed. | | | |
| 05 When the count of active allocations reaches zero, the page is removed from the subpool's page allocation and is now available to any subpool. | SCUNALLO | | |
| 06 The exit is to the common FREEMAIN exit. | SCFREEXT | | 1.2.4 |
| NOTE: If the request is to free a TCA, the above process is repeated for each storage area on the storage chain before exiting to the common FREEMAIN exit. The TCA is the last area freed. | | | |

Mixed Subpool FREEMAIN

Diagram - 1.2.2.3-01



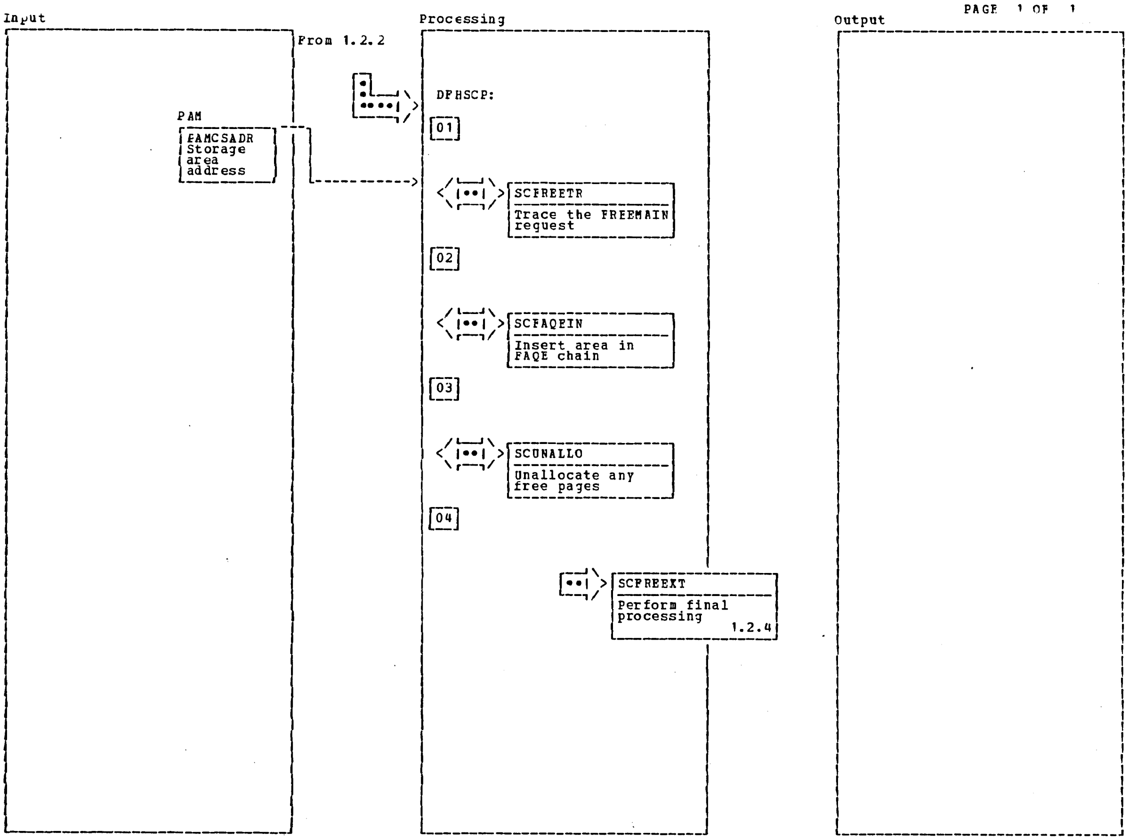
Isolated Subpool FREEMAIN

Diagram - 1.2.2.4-C1

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The duplicate Storage Accounting Areas are compared to see if either was destroyed. | | SCISOFRE | |
| 02 The area being freed is traced via a X'C9' trace entry. | SCFREETR | | 2.1.4 |
| 03 The storage chain is searched for the area being freed. When found, it is removed from the chain. | | | |
| 04 The area to be freed is inserted in the Free Area Queue Element (FAQE) chain. | SCFAQEIN | | |
| 05 Any full pages contained entirely in the new free area that are not part of the pages initially allocated for the TCA request are removed from the tasks page allocation and are now available to any subpool. | | | |
| 06 The exit is to the common FREEMAIN exit. | SCFREEXT | | 1.2.4 |
| NOTE: If the request is to free a TCA, the above process is repeated for each storage area on the storage chain before exiting to the common FREEMAIN exit. The TCA is the last area freed. | | | |

Isolated Subpool FREEMAIN

Diagram - 1.2.2.4-C1



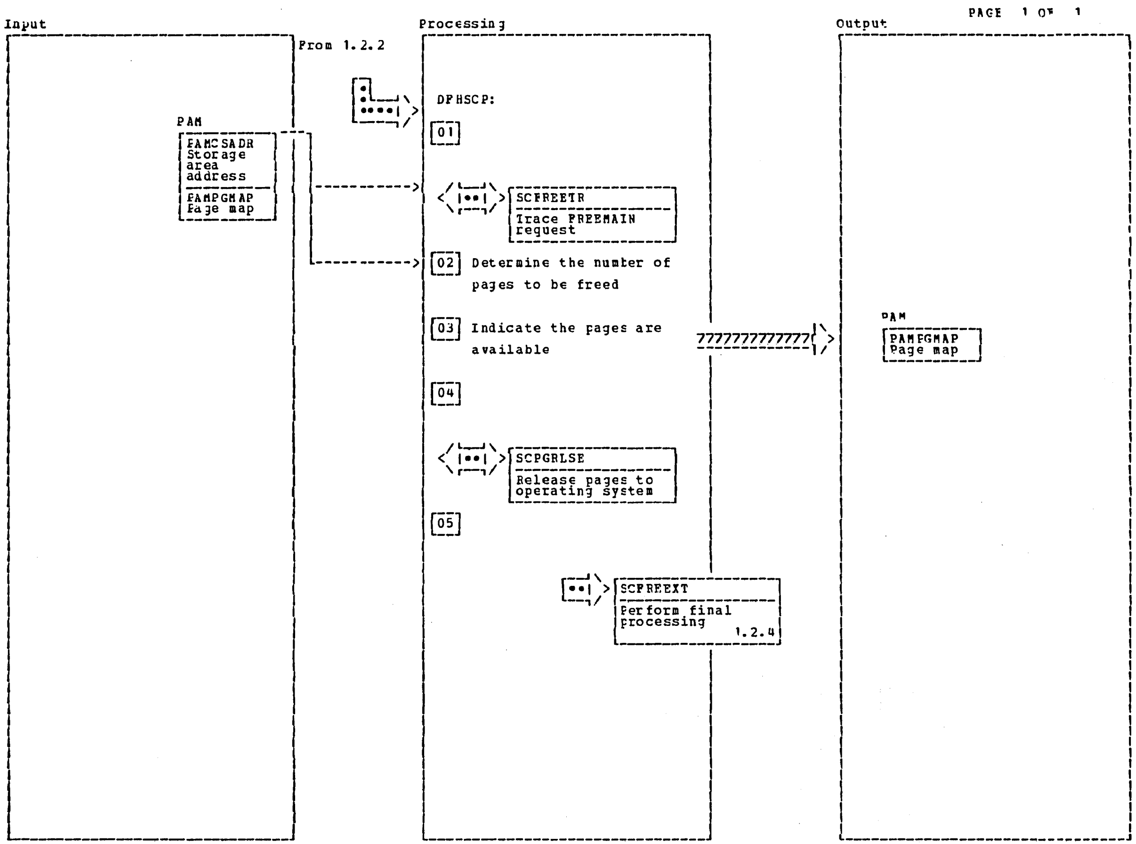
Shared Subpool FREEMAIN

Diagram - 1.2.2.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The area to be freed is traced via a X'C9' trace entry. | SCFREETR | SCSHRFRE | 2.1.4 | | | | |
| 02 The area to be freed is added to Free Area Queue Element (FAQE) chain. | SCFAQEIN | | | | | | |
| 03 If the free area had created an FAQE that spanned a full page, the page is removed from the subpool's page allocation and is now available to any subpool. | SCUNALLO | | | | | | |
| 04 The exit is to the common FREEMAIN exit. | SCFREEXT | | 1.2.4 | | | | |

Shared Subpool FREEMAIN

Diagram - 1.2.2.5-01



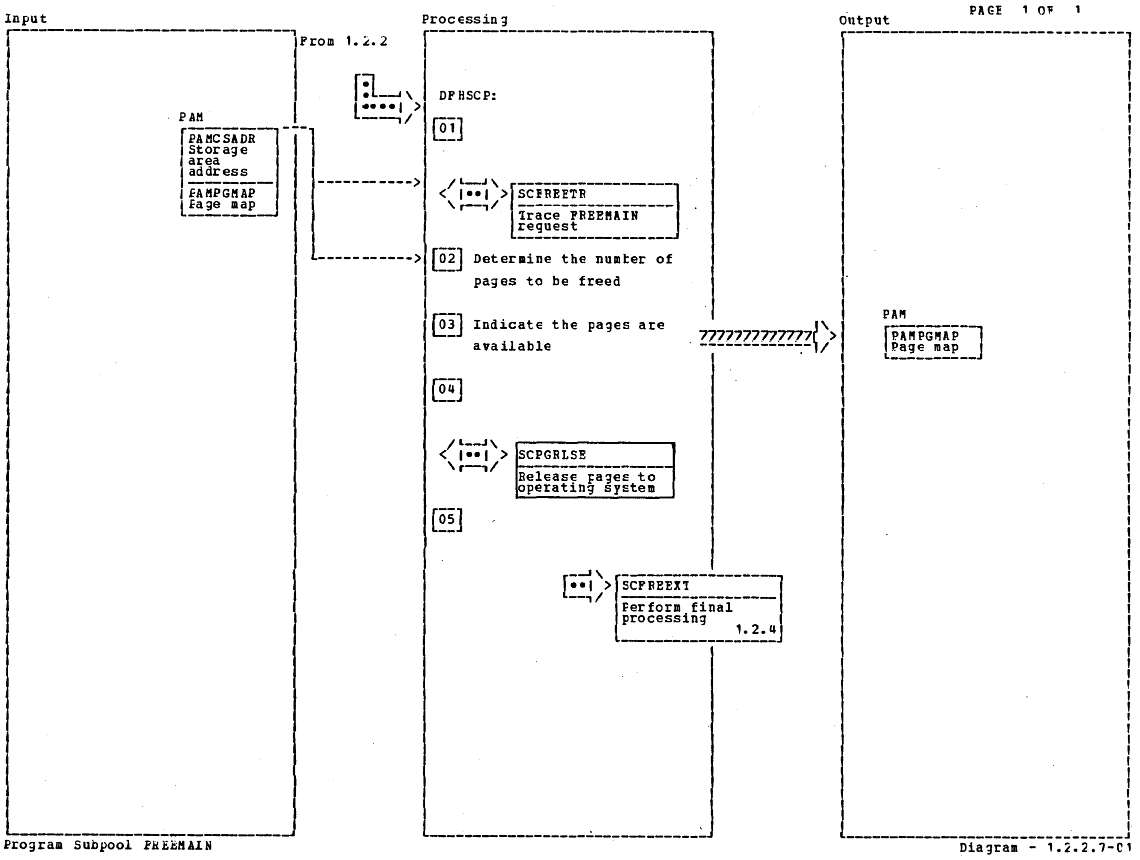
Aligned Subpool FREEMAIN

Diagram - 1.2.2.6-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The area to be freed is traced via a X'C9' trace entry. | SCFREETR | SCALGFRE | 2.1.4 | | | | |
| 02 The number of pages allocated to the area is determined from the second half of the PAM. | | | | | | | |
| 03 The page map in the PAM is changed to indicate that the pages which were allocated to the freed area are available to any subpool. | | | | | | | |
| 04 The freed pages are released to the operating system to prevent any unnecessary paging when the pages are again allocated. | SCPGRLSE | | | | | | |
| 05 The exit is to the common FREEMAIN exit. | SCFREEXT | | 1.2.4 | | | | |

Aligned Subpool FREEMAIN

Diagram - 1.2.2.6-C1



Program Subpool FREEMAIN

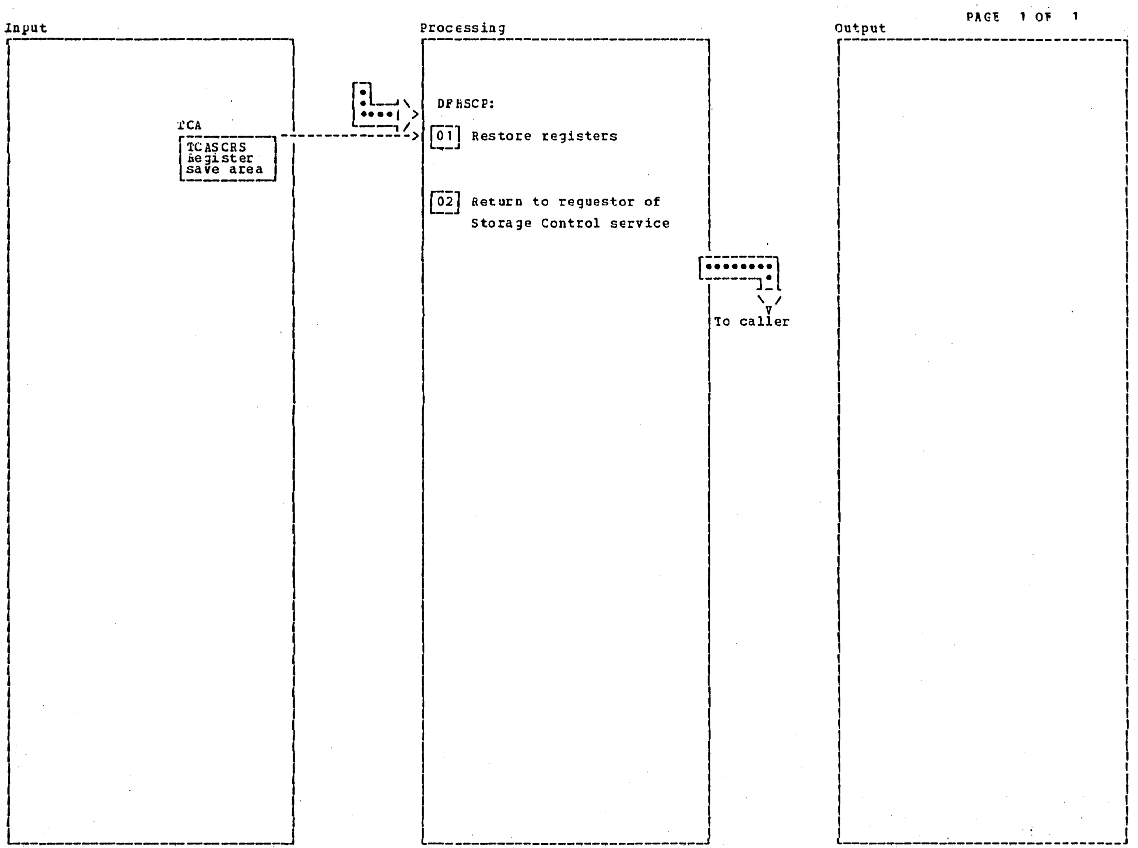
Diagram - 1.2.2.7-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The area to be freed is traced via a X'C9' trace entry. | SCFREETR | SCPGMFRE | 2.1.4 |
| 02 The number of pages allocated to the program is determined from the second half of the PAM. | | | |
| 03 The page map of the PAM is changed to indicate that the pages which were allocated to the freed area are available to any subpool. | | | |
| 04 The freed pages are released to the operating system to prevent any unnecessary paging when the pages are again allocated. | SCPGRLSE | | |
| 05 The exit is to the common FREEMAIN exit. | SCFREEXT | | 1.2.4 |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Program Subpool FREEMAIN

Diagram - 1.2.2.7-01


PAGE 1 OF 1
Input
Processing
Output
TCA
TCASCRS Register save area
DFHSCP:
01 Restore registers
02 Return to requestor of Storage Control service
To caller

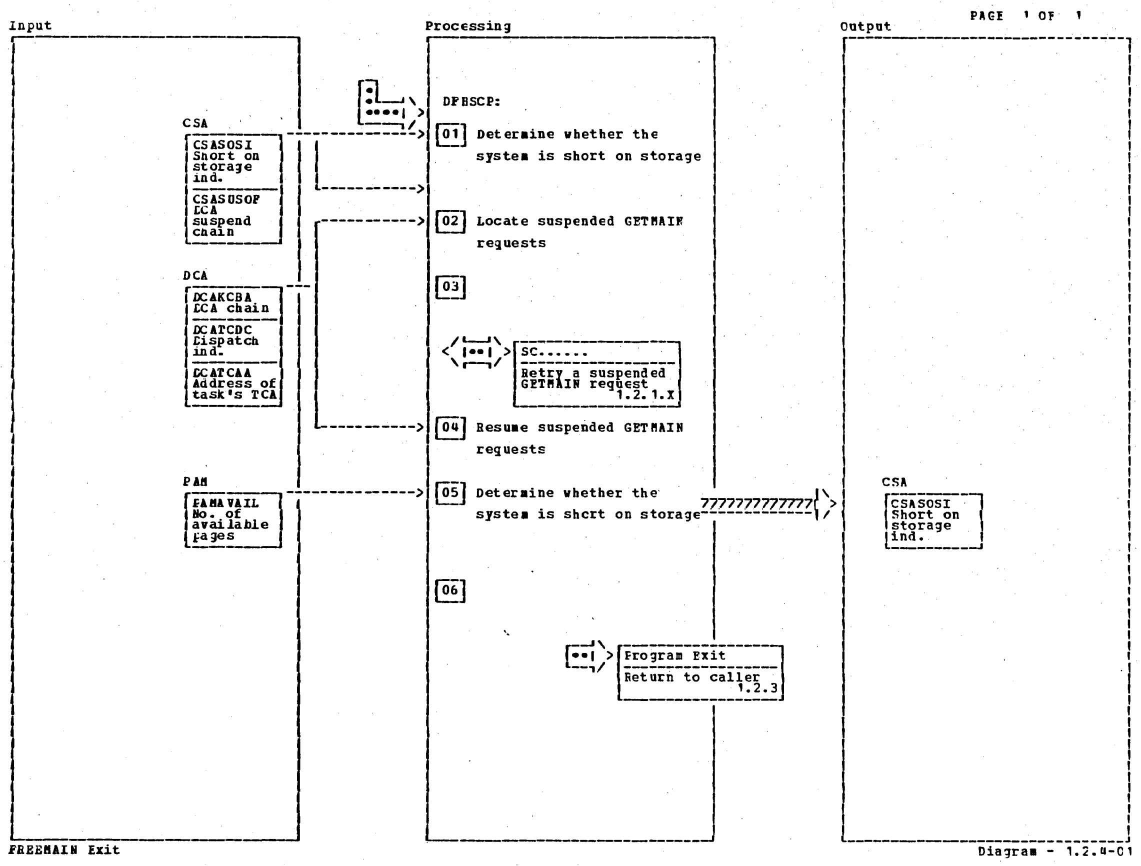
Program Exit

Diagram - 1.2.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 Control is returned to the requestor via the Runaway Task facility via an Interval Control EXIT macro instruction. | | | | | | | |

Program Exit

Diagram - 1.2.3-01

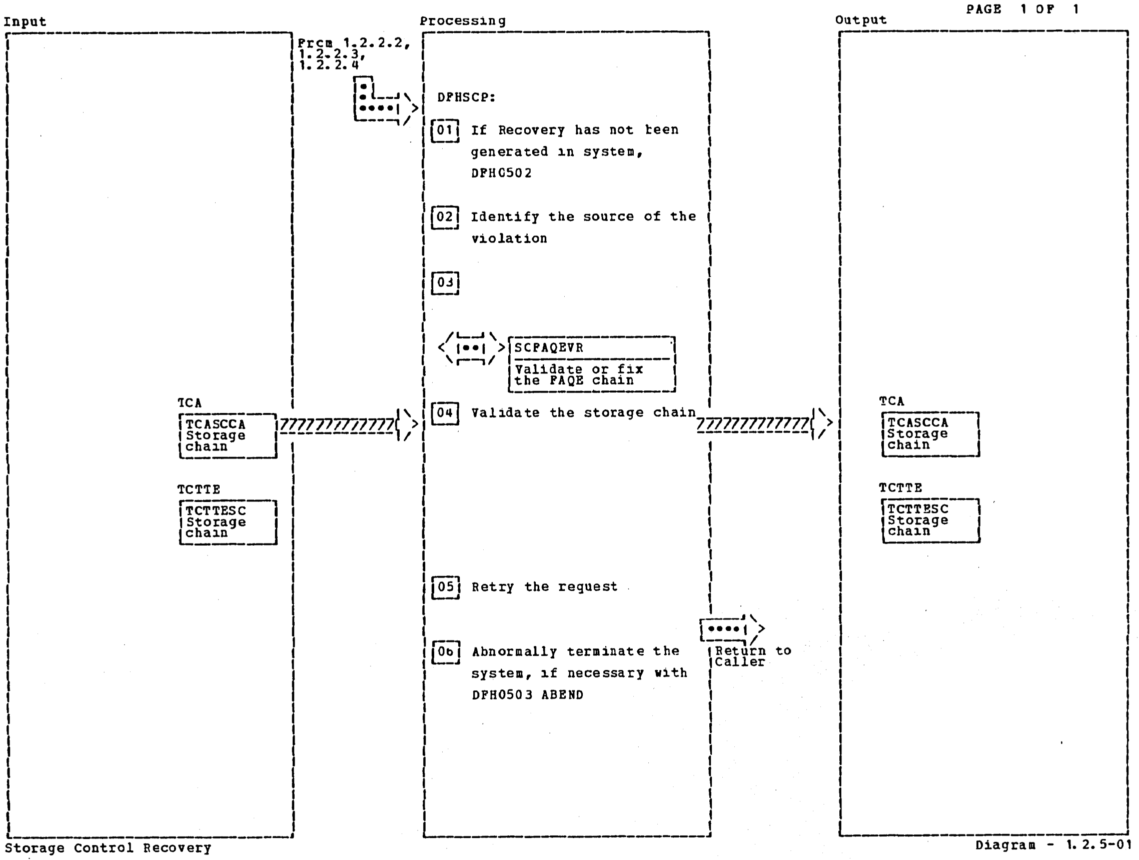

PAGE 1 OF 1
Input
Processing
Output
CSA
CSASOSI Short on storage ind.
CSASUSOF DCA suspend chain
DCA
DCAKCBA DCA chain
DCATCDC Dispatch ind.
DCATCAA Address of task's TCA
PAM
PAMAVAIL No. of available pages
DFHSCP:
01 Determine whether the system is short on storage
02 Locate suspended GETMAIN requests
03
SC......
Retry a suspended GETMAIN request 1.2.1.X
04 Resume suspended GETMAIN requests
05 Determine whether the system is short on storage
06
Program Exit
Return to caller 1.2.3
CSA
CSASOSI Short on storage ind.
FREEMAIN Exit
Diagram - 1.2.4-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| [02] The DCA suspend chain is searched for tasks suspended by Storage Control. | | | |
| [03] Each suspended request is retried by entering the required subpool GETMAIN routine. | | | |
| [04] When a suspended request is satisfied, the suspended task is resumed via a Task Control RESUME. | | SCFREXRE | 1.1.8 |
| [05] If the freed area relieved the short-on-storage condition, the short-on-storage indicator is turned off. | | SCFREXSS | |
| [06] The exit is to the program exit. | | | 1.2.3 |

| Notes | Routine | Label | Ref |
|---|---|---|---|

FREEMAIN Exit

Diagram - 1.2.4-01


PAGE 1 OF 1
Input
Processing
Output
Frcm 1.2.2.2, 1.2.2.3, 1.2.2.4
DFHSCP:
01 If Recovery has not been generated in system, DFHC502
02 Identify the source of the violation
03
SCFAQEVR
Validate or fix the FAQE chain
TCA
TCASCCA Storage chain
TCTTE
TCTTESC Storage chain
04 Validate the storage chain
05 Retry the request
06 Abnormally terminate the system, if necessary with DFH0503 ABEND
Return to Caller
TCA
TCASCCA Storage chain
TCTTE
TCTTESC Storage chain

Storage Control Recovery

Diagram - 1.2.5-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 02 The subpool that encountered the storage violation is determined by examining the current subpool header.<br>Violations are:<br>A. Nonmatched duplicate Storage Accounting Area.<br>B. Overlapping free areas. | | | |
| 03 If the subpool has an FAQE chain, each FAQE on the chain is checked for validity. If an error is found, the FAQE is rebuilt or a dummy is created in its place. | | | |
| 04 The TCA or TCTTE storage chain is verified or corrected using the duplicate Storage Accounting Areas. | | | |
| 05 If the problem was corrected, control is returned to the subpool routine that had contrcl when the problem was encountered. | | | |
| 06 If the problem could not be located or recovered, the system is abnormally terminated using operating system ABEND macro. | | | |

Storage Contrcl Recovery

Diagram - 1.2.5-01

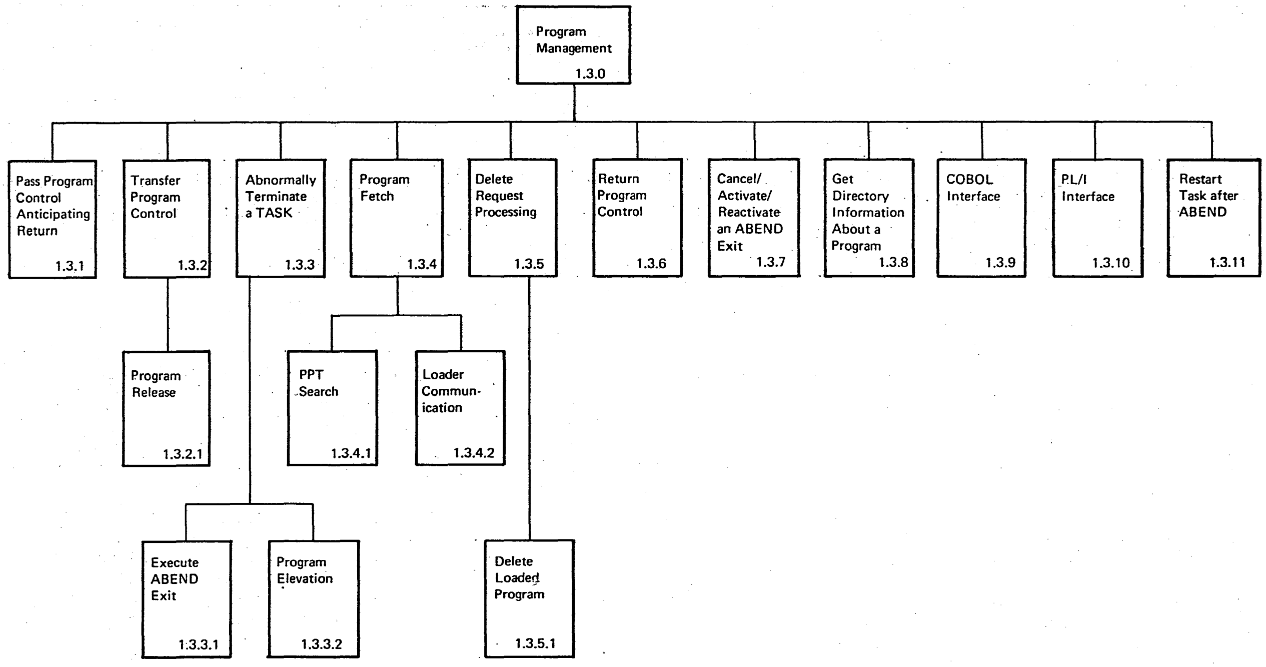
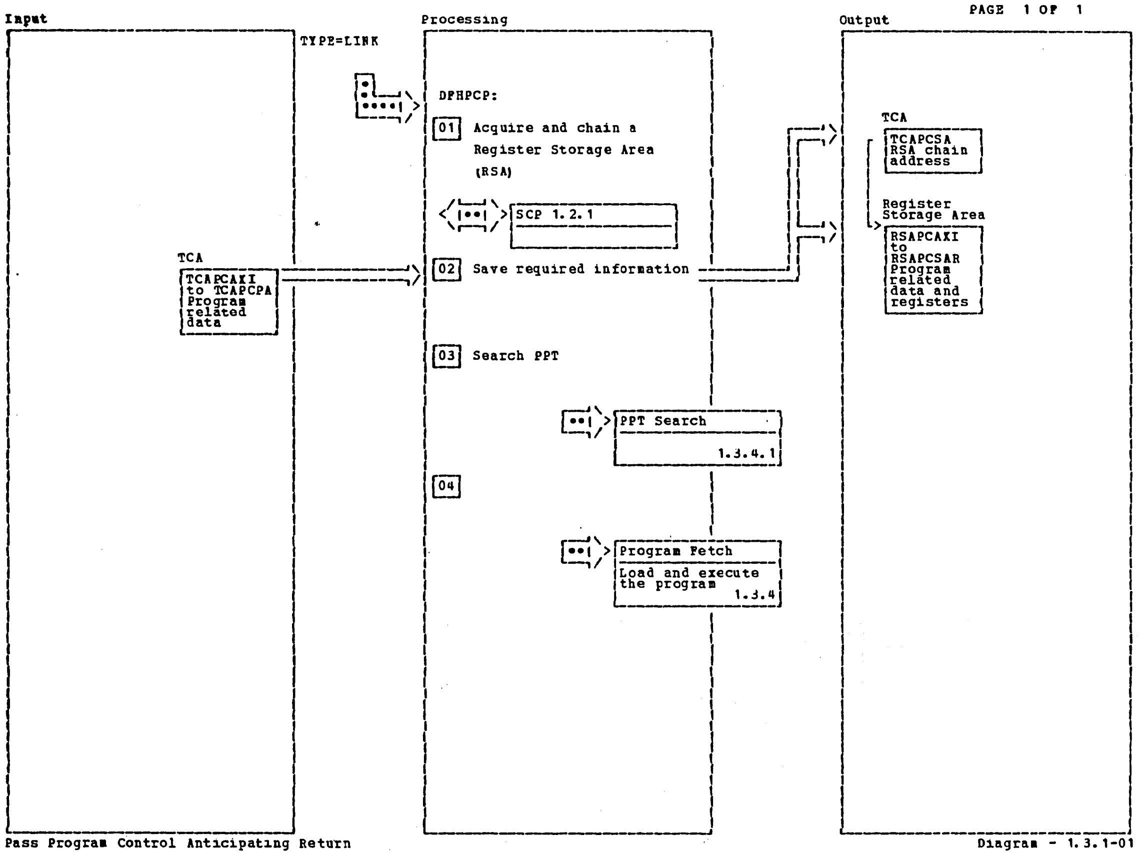

Program Management 1.3.0
Pass Program Control Anticipating Return 1.3.1
Transfer Program Control 1.3.2
Abnormally Terminate a TASK 1.3.3
Program Fetch 1.3.4
Delete Request Processing 1.3.5
Return Program Control 1.3.6
Cancel/ Activate/ Reactivate an ABEND Exit 1.3.7
Get Directory Information About a Program 1.3.8
COBOL Interface 1.3.9
PL/I Interface 1.3.10
Restart Task after ABEND 1.3.11
Program Release 1.3.2.1
PPT Search 1.3.4.1
Loader Commun- ication 1.3.4.2
Execute ABEND Exit 1.3.3.1
Program Elevation 1.3.3.2
Delete Loaded Program 1.3.5.1

Diagram – 1.3.0-01


PAGE 1 OF 1
Input
Processing
Output
TYPE=LINK
DFHPCP:
01 Acquire and chain a Register Storage Area (RSA)
SCP 1.2.1
TCA
TCAPCAXI to TCAPCPA Program related data
02 Save required information
03 Search PPT
PPT Search
1.3.4.1
04
Program Fetch
Load and execute the program
1.3.4
TCA
TCAPCSA RSA chain address
Register Storage Area
RSAPCAXI to RSAPCSAR Program related data and registers

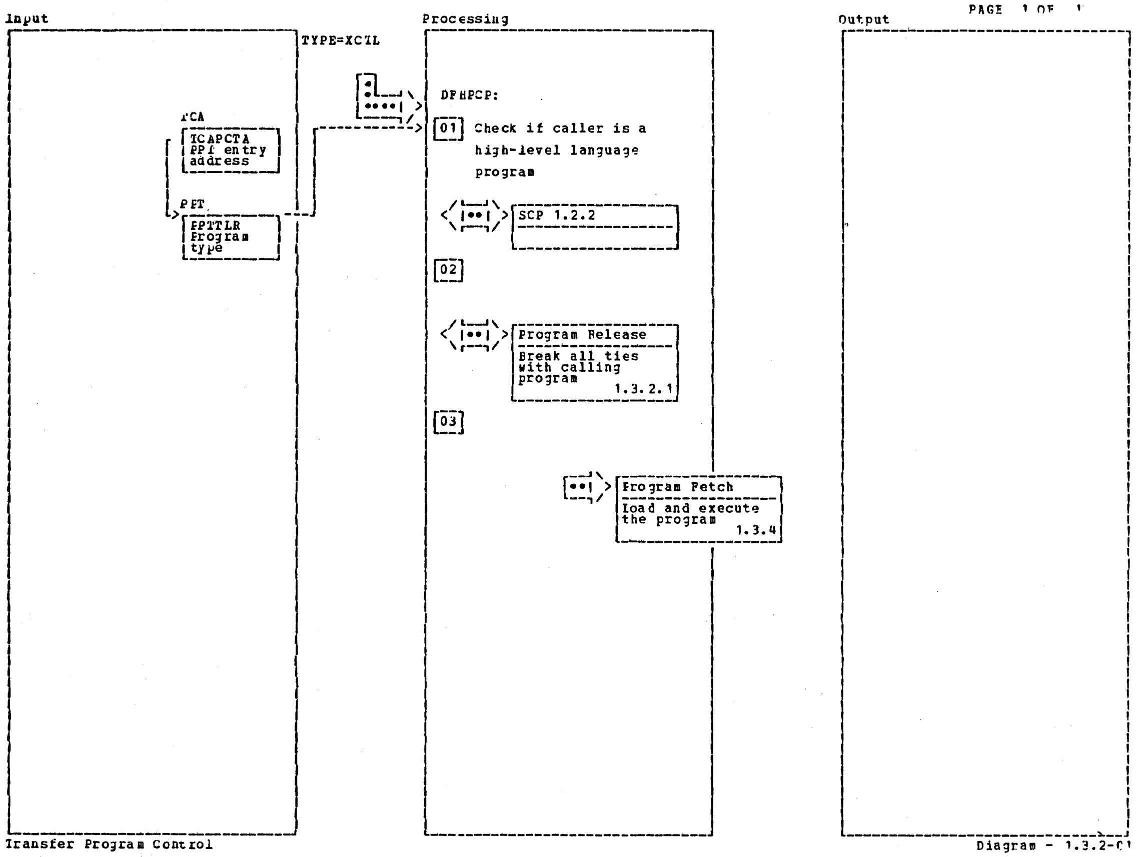
Pass Program Control Anticipating Return

Diagram - 1.3.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | PCLINK | | | | | |
| 02 The information saved includes the ABEND exit flags and addresses, the RSA chain address, the HLL save area address, the HLL storage chain address, and the return registers. | | | | | | | |

Pass Program Control Anticipating Return

Diagram - 1.3.1-01



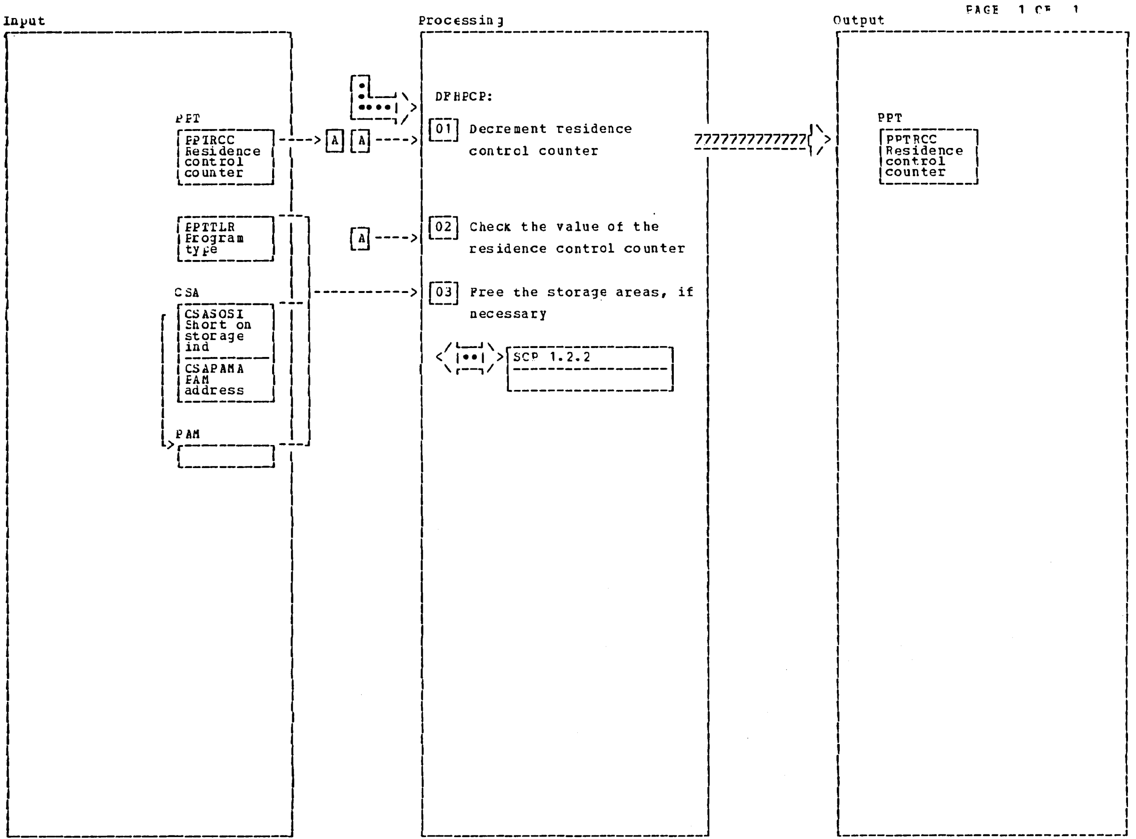
Transfer Program Control

Diagram - 1.3.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If COBOL is used, free the storage area acquired for TGT.<br>If PL/I is used, free the storage areas acquired by PL/I for this program. |  | PCXCTL |  |  |  |  |  |

Transfer Program Control

Diagram - 1.3.2-01


PAGE 1 OF 1
Input
Processing
Output
PPT
PPTRCC Residence control counter
A
A
PPTTLR Program type
A
CSA
CSASOSI Short on storage ind
CSAPAMA PAM address
PAM
DFHPCP:
01 Decrement residence control counter
02 Check the value of the residence control counter
03 Free the storage areas, if necessary
SCP 1.2.2
PPT
PPTRCC Residence control counter

Program Release

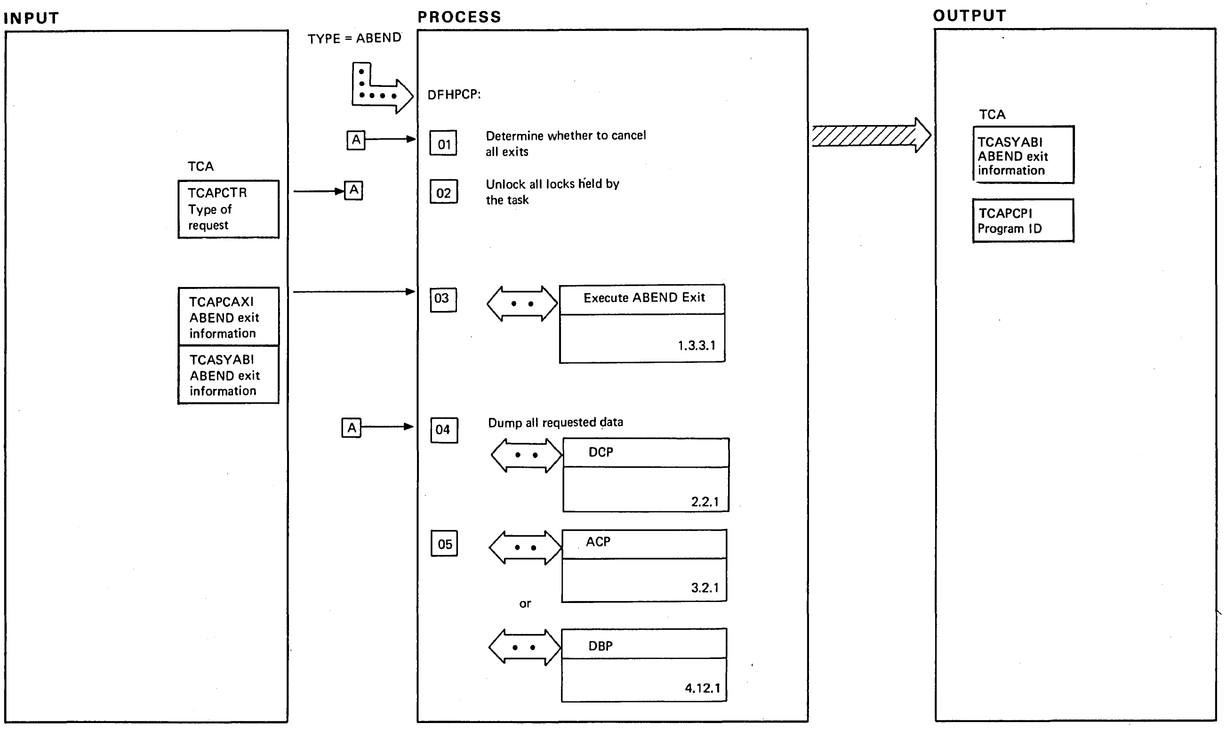
Diagram - 1.3.2.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | PCCPRRN | | | | | |
| 02 If the counter is less than zero, the task is abnormally terminated with an APCN ABEND code. | | | | | | | |
| If the counter is greater than zero, control is returned to the caller. | | | | | | | |
| 03 The storage areas are released if either: | | | | | | | |
| The module is a map. | | PCPRRLE | | | | | |
| The short on storage indicator is on. | | PCPRFB | | | | | |
| The program is to be deleted. | | | | | | | |

Program Release

Diagram - 1.3.2.1-01

Page 1 to 1



Abnormally Terminate a Task

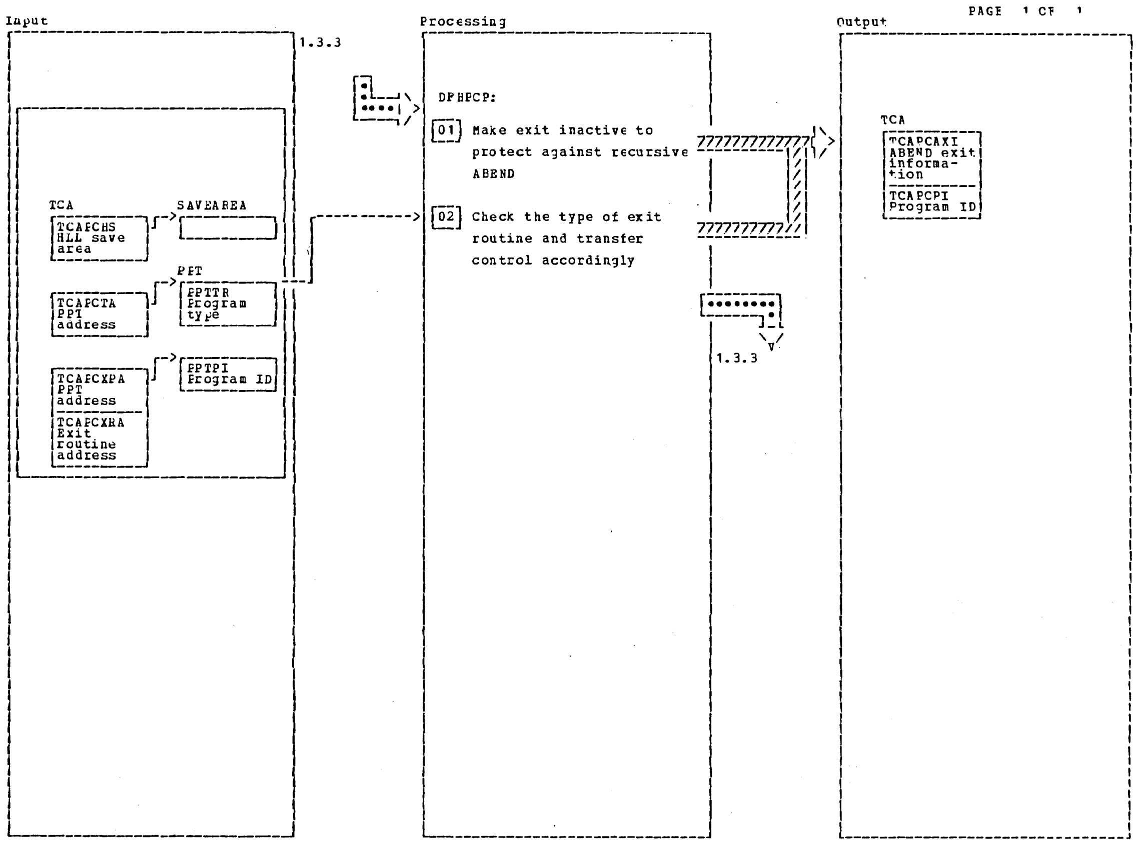
Diagram - 1.3.3-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 A cancel all exits request is specified by the CANCEL=YES operand. | | PCABEND | |
| 02 | | PCSTXTB | |
| 03 | | PCABCEB | |
| 04 A dump is requested with the ABCODE=xxxx operand. | | PCUNLKB | |
| 05 XCTL to DFHACP, or DFHDBP if Dynamic Transaction Backout is required. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Abnormally Terminate a Task

Diagram - 1.3.3-01


PAGE 1 OF 1
Input
1.3.3
TCA
TCAFCHS HLL save area
SAVEAREA
PPT
TCAFCTA PPT address
PPTTR Program type
TCAFCXPA PPT address
TCAFCXRA Exit routine address
PPTPI Program ID
Processing
DFHPCP:
01 Make exit inactive to protect against recursive ABEND
02 Check the type of exit routine and transfer control accordingly
1.3.3
Output
TCA
TCAPCAXI ABEND exit information
TCAPCPI Program ID

Execute ABEND Exit

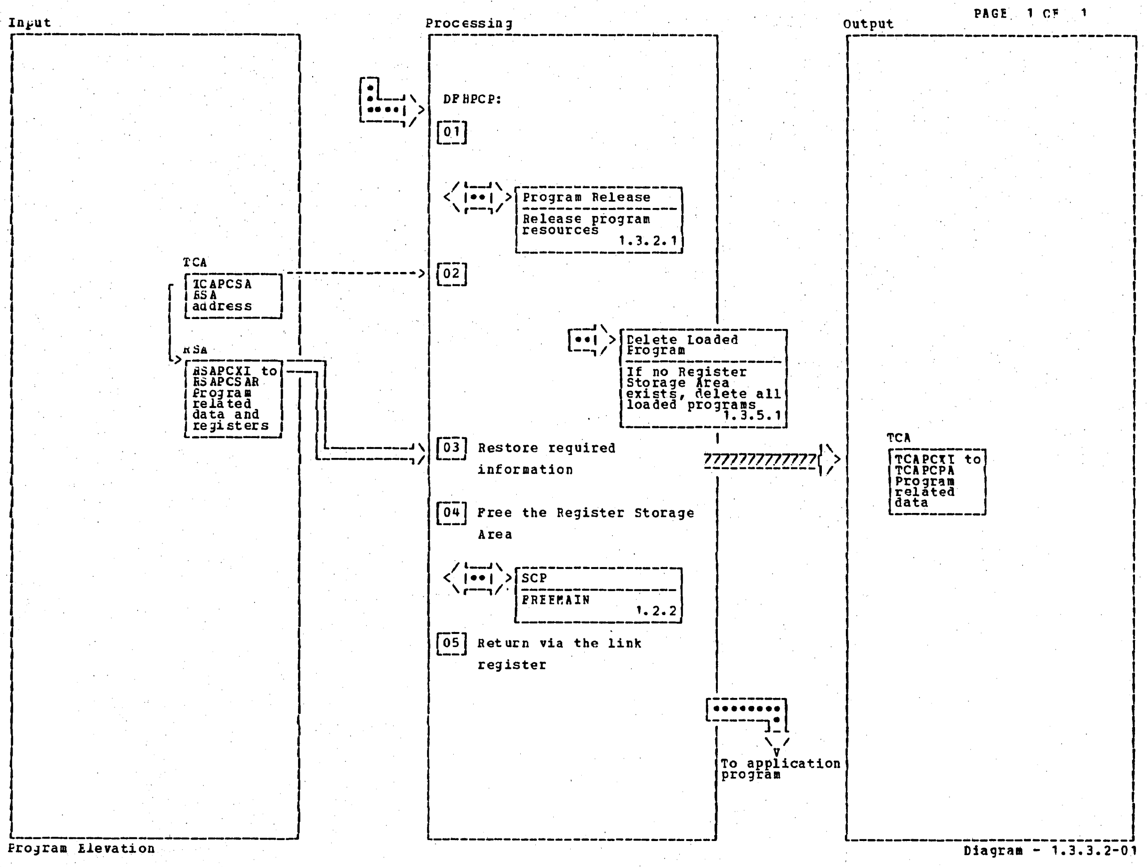
Diagram - 1.3.3.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 02 If the exit is a program, link to it. | | | |
| If the exit is an Assembler language routine, branch to it. | | PCSTXTR | |
| If the exit is a HLL routine, restore the registers and branch to it. | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Execute ABEND Exit

Diagram - 1.3.3.1-01



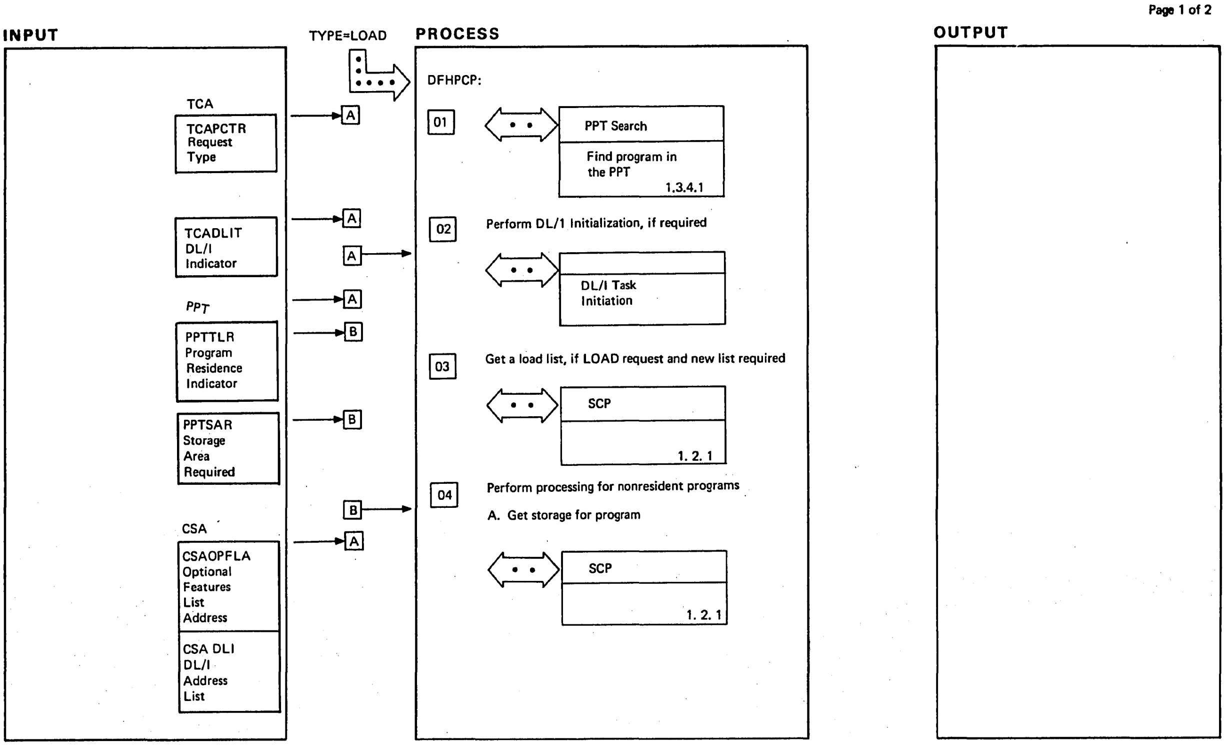
Program Elevation

Diagram - 1.3.3.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | PCCPERN | | | | | |
| 03 If the released program was linked by TYPE=LINK, the Register Storage Area is restored into the TCA. | | | | | | | |

Program Elevation

Diagram - 1.3.3.2-01


Page 1 of 2
INPUT
TYPE=LOAD
PROCESS
OUTPUT
TCA
TCAPCTR Request Type
TCADLIT DL/I Indicator
PPT
PPTTLR Program Residence Indicator
PPTSAR Storage Area Required
CSA
CSAOPFLA Optional Features List Address
CSA DLI DL/I Address List
A
B
DFHPCP:
01
PPT Search
Find program in the PPT
1.3.4.1
02
Perform DL/1 Initialization, if required
DL/I Task Initiation
03
Get a load list, if LOAD request and new list required
SCP
1. 2. 1
04
Perform processing for nonresident programs
A. Get storage for program
SCP
1. 2. 1

Program Fetch

Diagram - 1.3.4-01

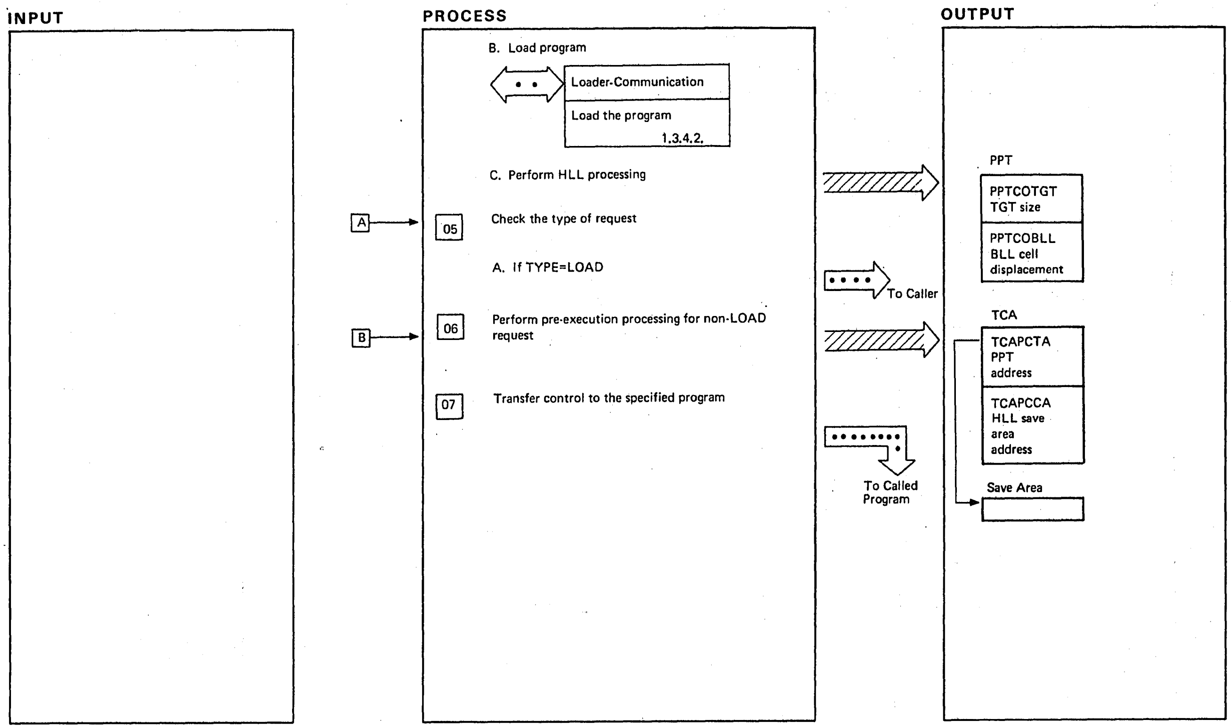
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 | | RCPF | |
| 04 In addition, all permanently resident HLL programs must be initialized the first time they are referenced. | | | |
| A. The Storage required (in number or doublewords) is kept in the PPT. | | PCGPS | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Program Fetch

Diagram - 1.3.4-01

Page 2 of 2



Program Fetch

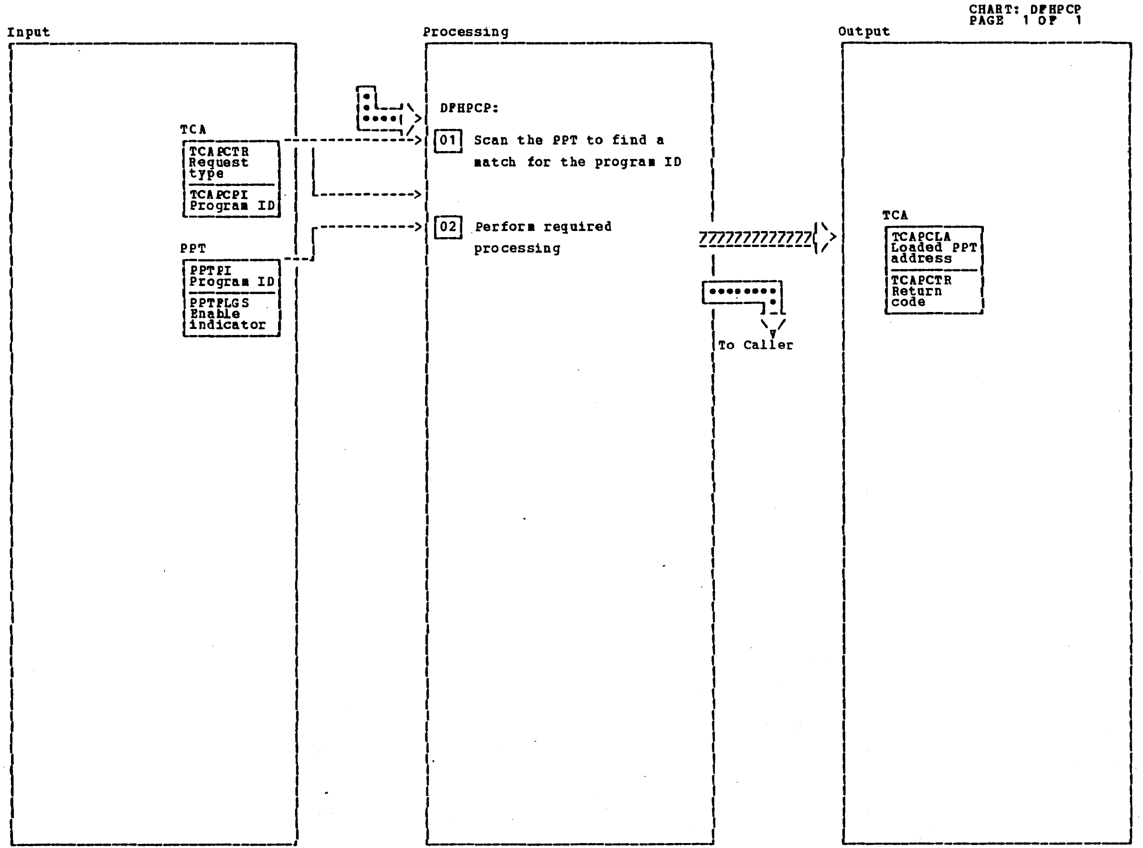
Diagram — 1.3.4-02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| B. Program load | | PCCLOAD | |
| C. The address of the PL/I or COBOL macro interface routine is placed in the PL/I or COBOL program. | | PCPEXIT | |
| 05 If a program load was requested, control is returned to the caller. | | PCPFRCUB | |
| 06 The PPT address is placed in the TCA. | | DOCPCP10 | |
| For HLL request, task-related storage is obtained. | | | |
| For an application using the command level interface, LOAD DFHEIP if it has not already been loaded. | | DFHPCEI | 6.5.1 |
| For COBOL applications the first two BLL cells, BLLDS and CSA are initialized | | | |
| 07 | | PCPE | |

Program Fetch

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Diagram — 1.3.4-02



01 Scan the PPT to find a match for the program ID

02 Perform required processing

PPT Search

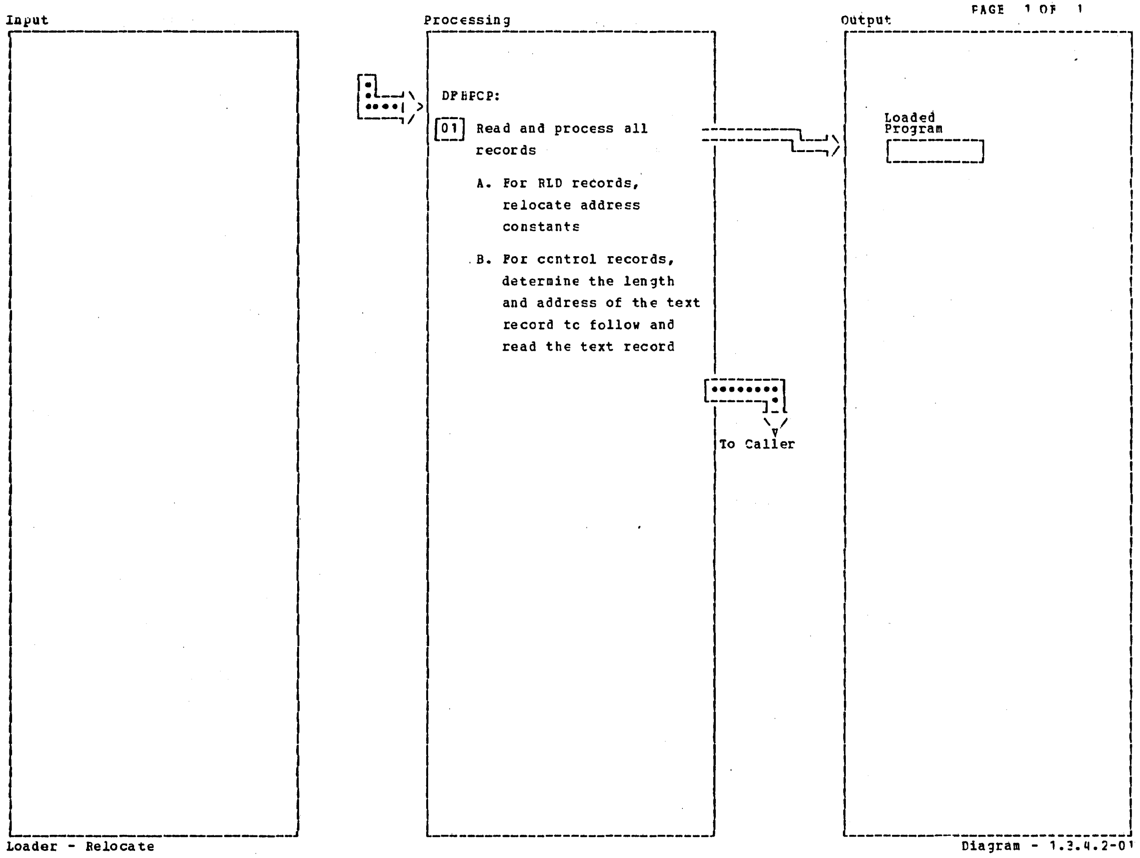
Diagram - 1.3.4.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | PCPPTSN | | | | | |
| 02 If the PPT entry was disabled, the request is treated as an error. | | PCPPTLB | | | | | |
| If an error occurs, the task is abnormally terminated (ABEND code APCT) unless the request was conditional or SETXIT was specified. (For SETXIT or conditional requests, the error indicator is returned as the message.) | | PCPPTEOK | | | | | |
| If LOCATE was requested, the PPT address is placed in the TCA, and control is returned to the calling program. | | PCPPTHF | | | | | |
| If not LOCATE, return. | | | | | | | |
| The PPT is searched, using the DFHPC=LOCATE macro instruction. | | | | | | | |

PPT Search

Diagram - 1.3.4.1-01

PAGE 1 OF 1



DFHPCP:

01 Read and process all records

A. For RLD records, relocate address constants

B. For ccntrol records, determine the length and address of the text record tc follow and read the text record

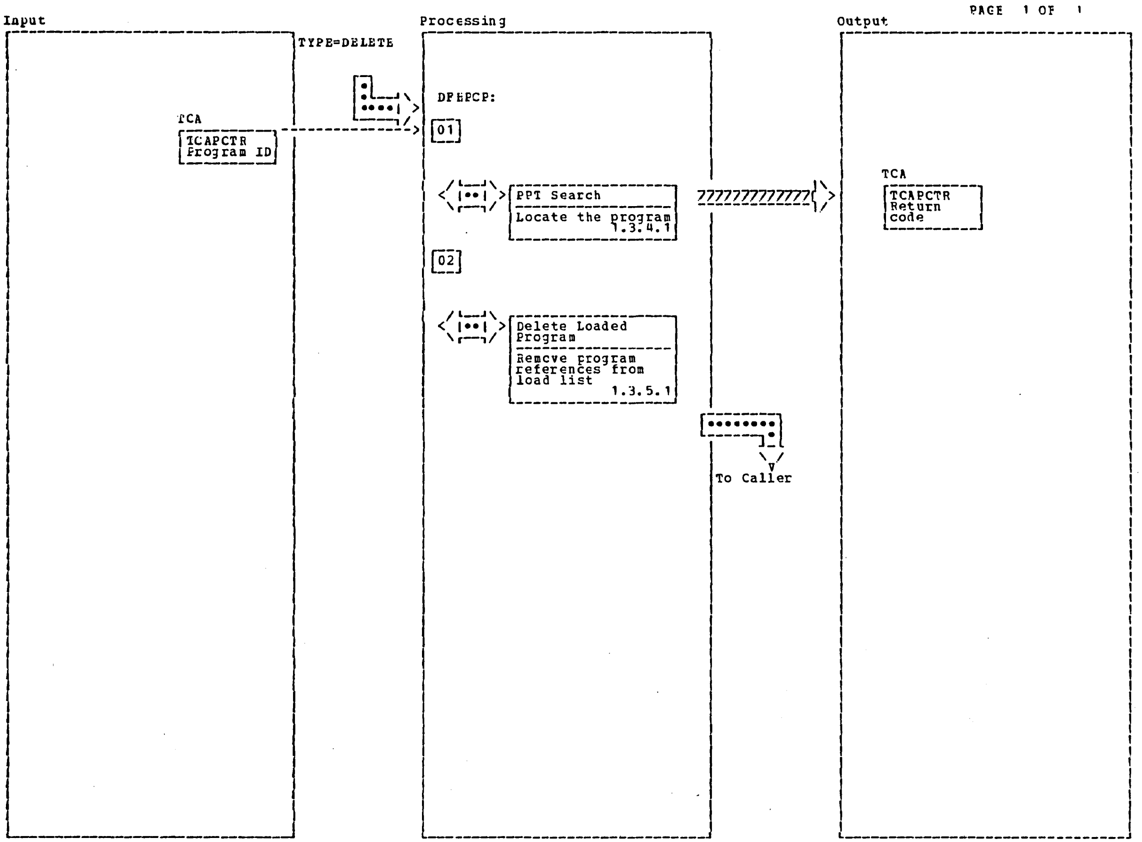
Loader - Relocate

Diagram - 1.3.4.2-01

| Notes | Routine | Latel | Ref | Notes | Routine | Latel | Ref |
|---|---|---|---|---|---|---|---|
| 01 Control is returned to caller either when the last RLD record is processed or wnen the last text record is read. |  | PCLCADRT |  |  |  |  |  |

Loader - Relocate

Diagram - 1.3.4.2-01



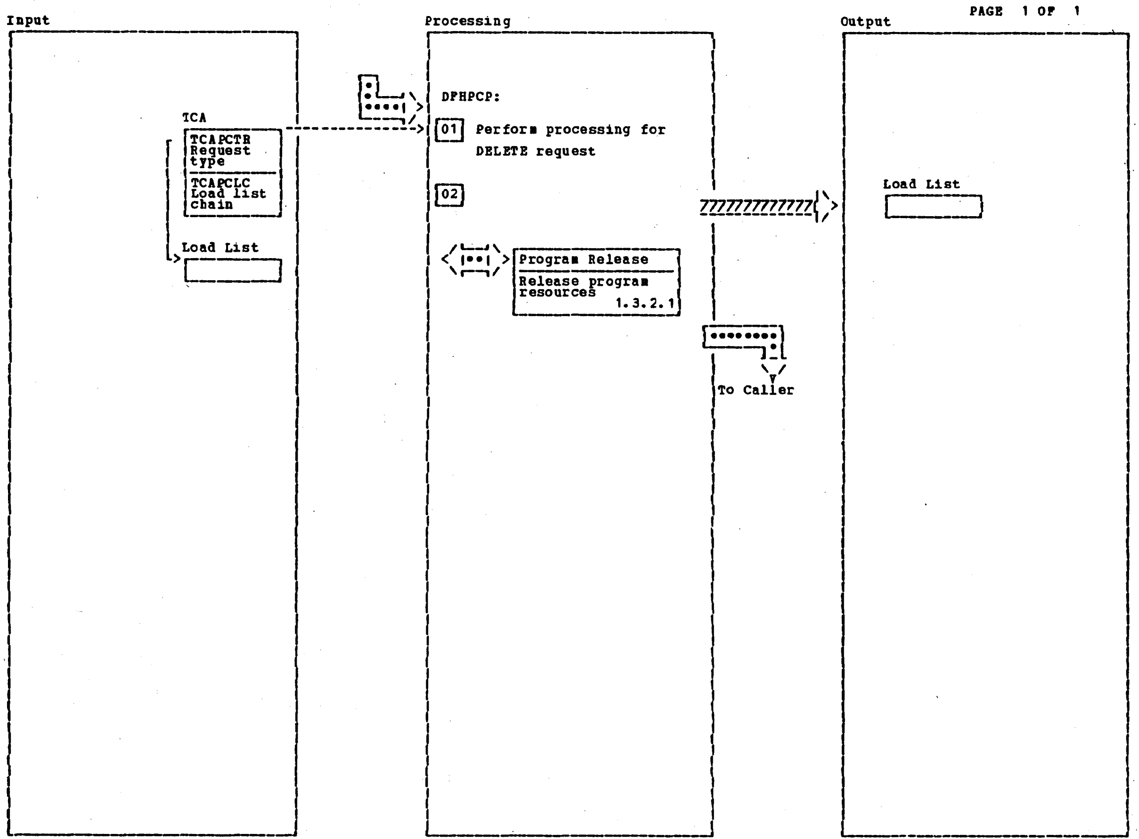
DFHPC TYPE=DELETE Request Processing

Diagram - 1.3.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | PCDELETE | | | | | |

DFHPC TYPE=DELETE Request Processing

Diagram - 1.3.5-01



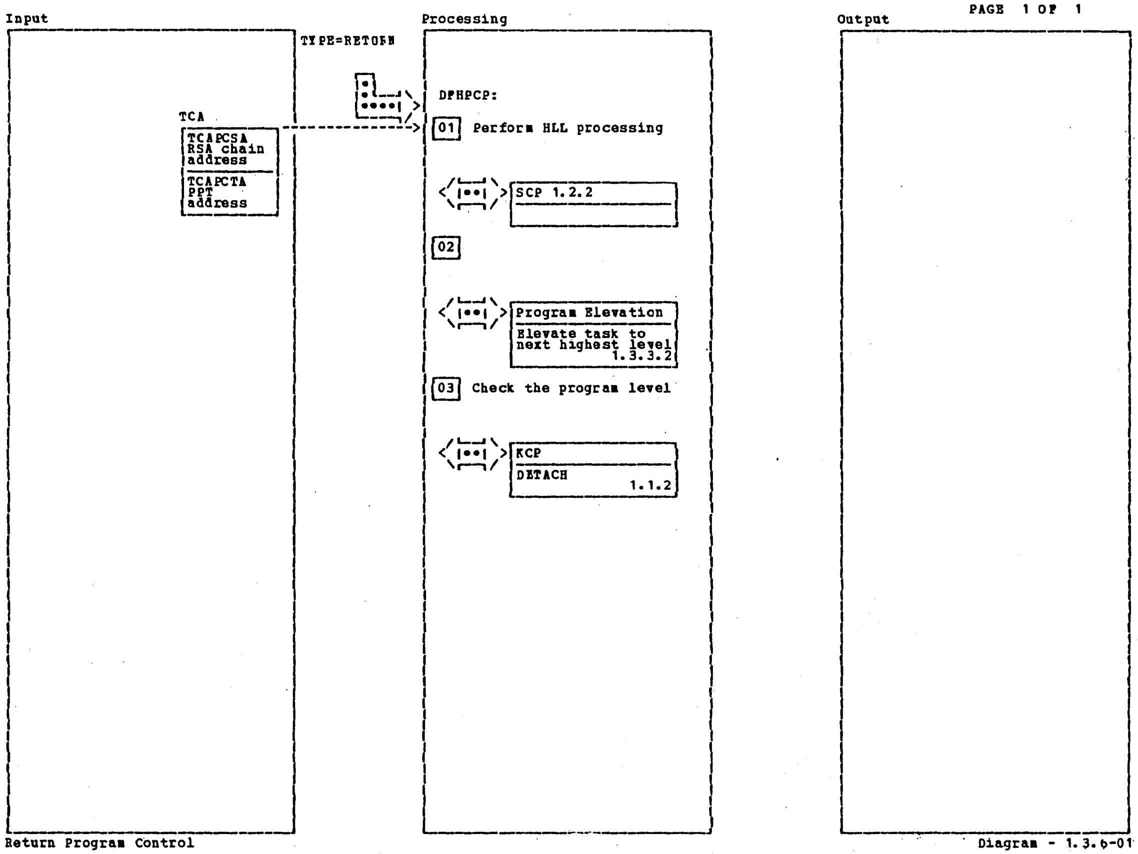
Delete Loaded Program

Diagram - 1.3.5.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 For DELETE request, program is removed from the load list.<br>If a matching ID is not found, control is returned to the caller. |  | PCCDLPN |  |  |  |  |  |

Delete Loaded Program

Diagram - 1.3.5.1-01



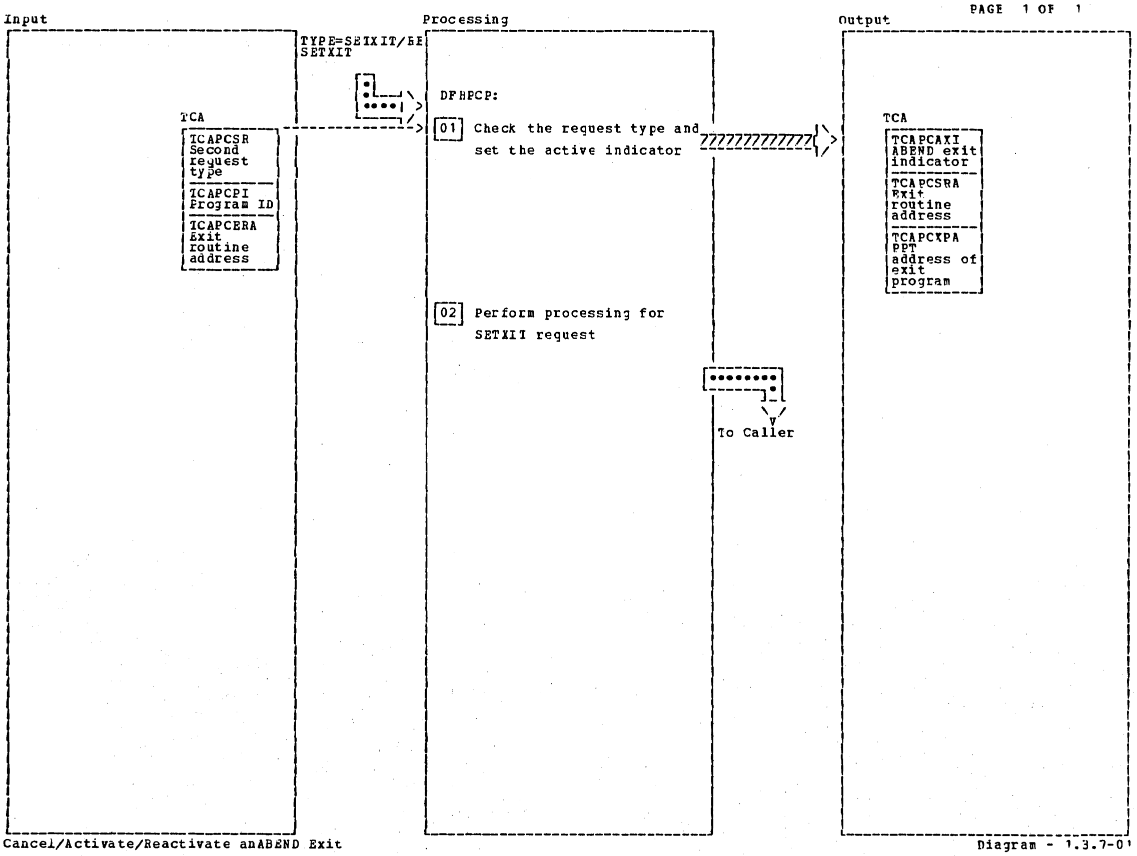
Return Program Control

Diagram - 1.3.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 All HLL storage areas are released. | | PCRETURN | | | | | |
| 03 If the program is at the highest level, a Task Control DETACH is issued to terminate the task.<br>If not at the highest level, control is transferred to the specified program. | | | | | | | |

Return Program Control

Diagram - 1.3.6-01


PAGE 1 OF 1
Input
Processing
Output
TYPE=SETXIT/RE
SETXIT
TCA
TCAPCSR Second request type
TCAPCPI Program ID
TCAPCERA Exit routine address
DFHPCP:
01 Check the request type and set the active indicator
02 Perform processing for SETXIT request
To Caller
TCA
TCAPCAXI ABEND exit indicator
TCAPCSRA Exit routine address
TCAPCXPA PPT address of exit program

Cancel/Activate/Reactivate anABEND Exit

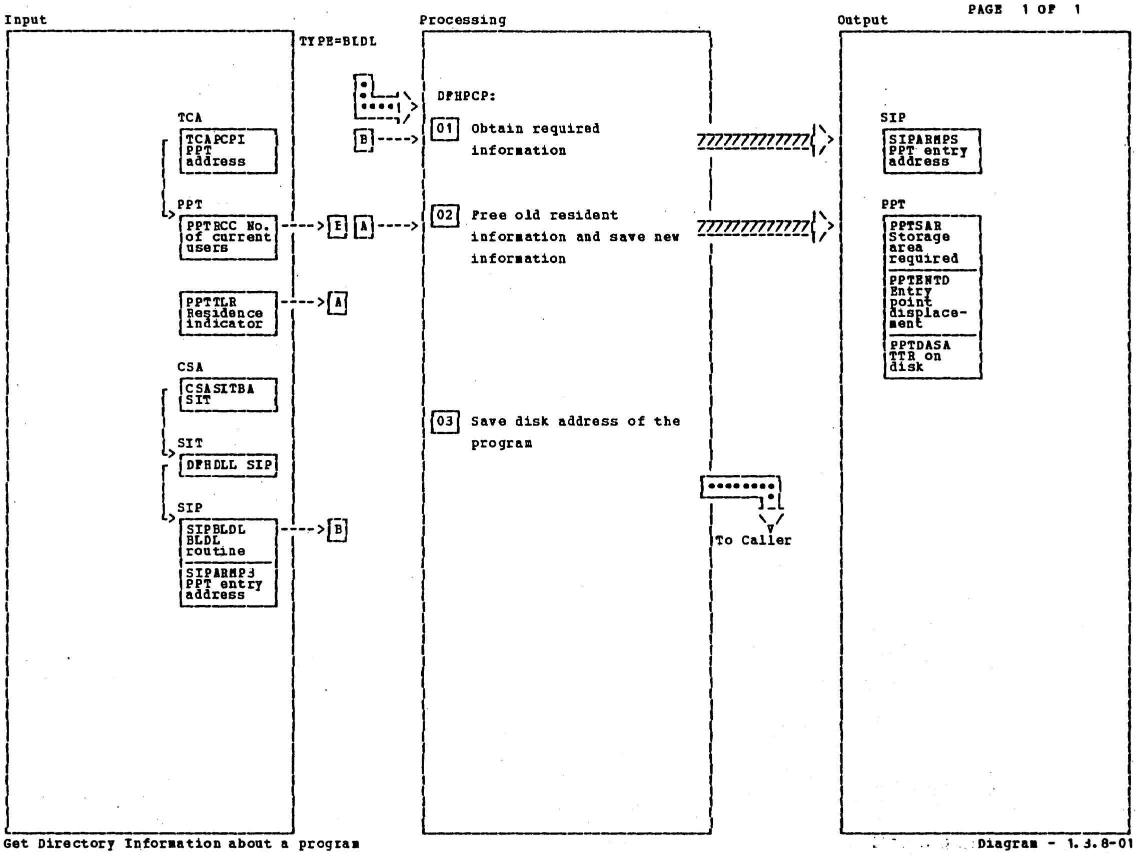
Diagram - 1.3.7-01

| Notes | Routine | Latel | Ref | Notes | Routine | Latel | Fef |
|---|---|---|---|---|---|---|---|
| 01 For activate/reactivate requests, tne active indicator is turned on.<br>For cancel requests, the active indicator is turned off. |  | PCSETXIT |  |  |  |  |  |
| 02 The routine or PPT entry address is set up in tne TCA. (The PPT is searchea to determine tne PPT entry address.) | 1.3.4.1 |  |  |  |  |  |  |

Cancel/Activate/Reactivate anABEND Exit

Diagram - 1.3.7-01

PAGE 1 OF 1


Input
Processing
Output
TYPE=BLDL
DFHPCP:
01 Obtain required information
02 Free old resident information and save new information
03 Save disk address of the program
To Caller
TCA
TCAPCPI PPT address
PPT
PPTRCC No. of current users
E
A
PPTTLR Residence indicator
A
CSA
CSASITBA SIT
SIT
DFHDLL SIP
SIP
SIPBLDL BLDL routine
B
SIPARMP3 PPT entry address
B
SIP
SIPARMPS PPT entry address
PPT
PPTSAR Storage area required
PPTENTD Entry point displacement
PPTDASA TTR on disk

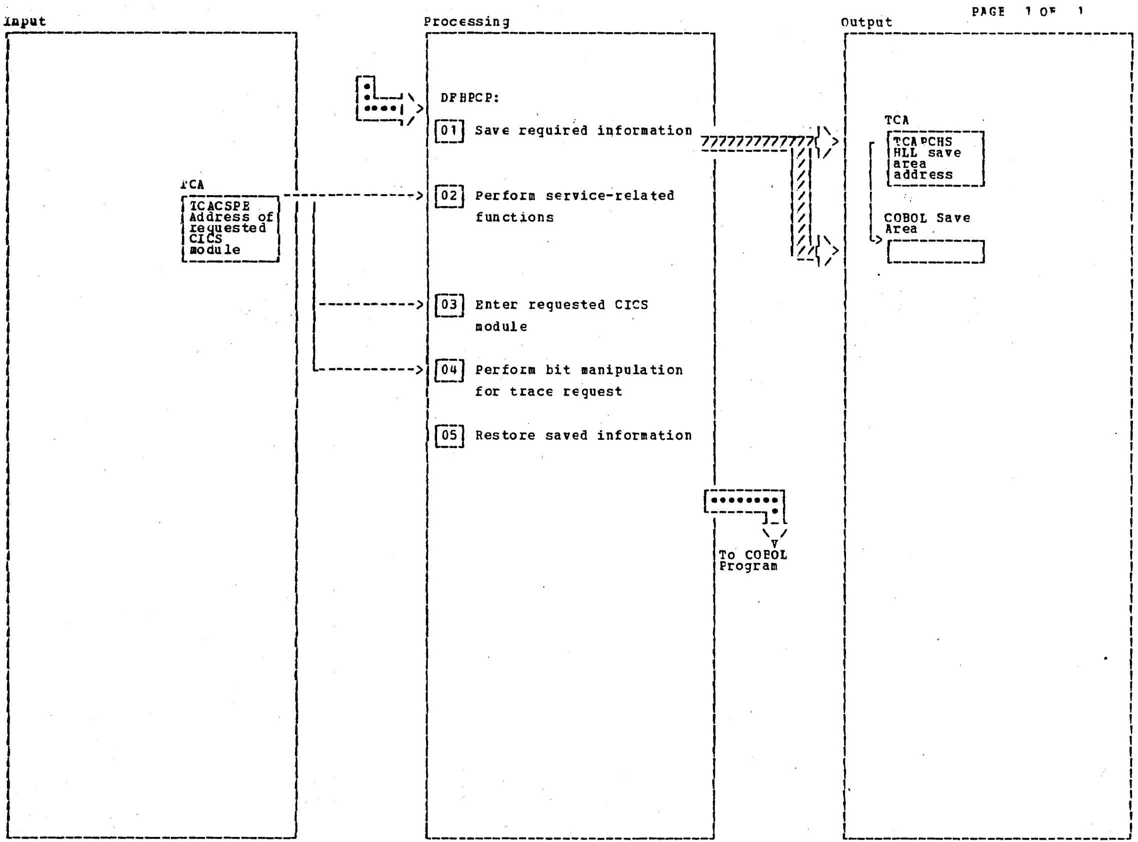
Get Directory Information about a program

Diagram - 1.3.8-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 A DOS/VS LOAD macro instruction is issued to obtain the program size, its entry point displacement, and its disk location. The LOAD macro instruction is issued with a TXT=NO parameter to cause a directory search only. | | PCBLDL | | | | | |
| 02 | | PCBLDLOK | | | | | |
| 03 | | PCBLDFB | | | | | |

Get Directory Information about a program

Diagram - 1.3.8-01


PAGE 1 OF 1
Input
Processing
Output
TCA
TCACSPE Address of requested CICS module
DFHPCP:
01 Save required information
02 Perform service-related functions
03 Enter requested CICS module
04 Perform bit manipulation for trace request
05 Restore saved information
To COBOL Program
TCA
TCAPCHS HLL save area address
COBOL Save Area

COBOL Interface

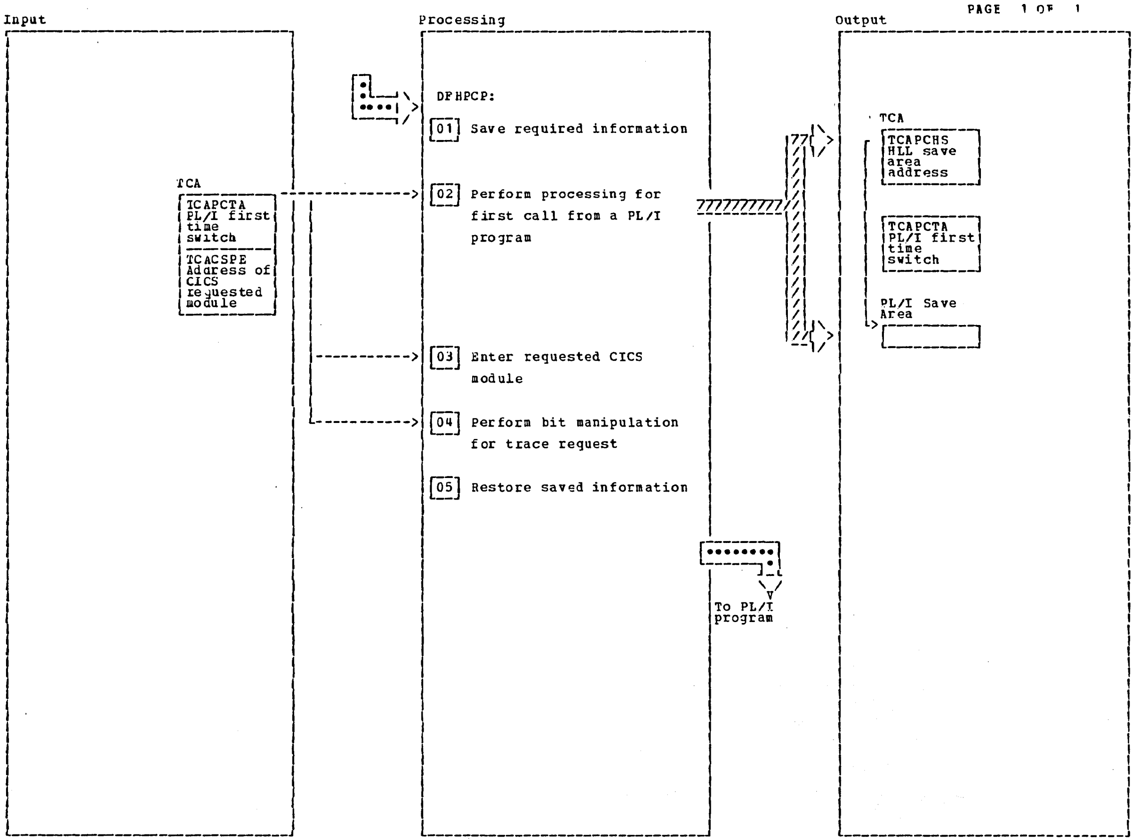
Diagram - 1.3.9-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The contents of the registers are saved. | | PCCBLIN | | | | | |
| 02 Move information previously moved by the macro request expansion. | | | | | | | |
| 03 | | PCKCPPB | | | | | |
| 04 | | PCHILTR | | | | | |
| 05 | | PCCOBRTN | | | | | |

COBOL Interface

Diagram - 1.3.9-01

PAGE 1 OF 1


Input
Processing
Output
TCA
TCAPCTA PL/I first time switch
TCACSPE Address of CICS requested module
DFHPCP:
01 Save required information
02 Perform processing for first call from a PL/I program
03 Enter requested CICS module
04 Perform bit manipulation for trace request
05 Restore saved information
To PL/I program
TCA
TCAPCHS HLL save area address
TCAPCTA PL/I first time switch
PL/I Save Area

PL/I Interface

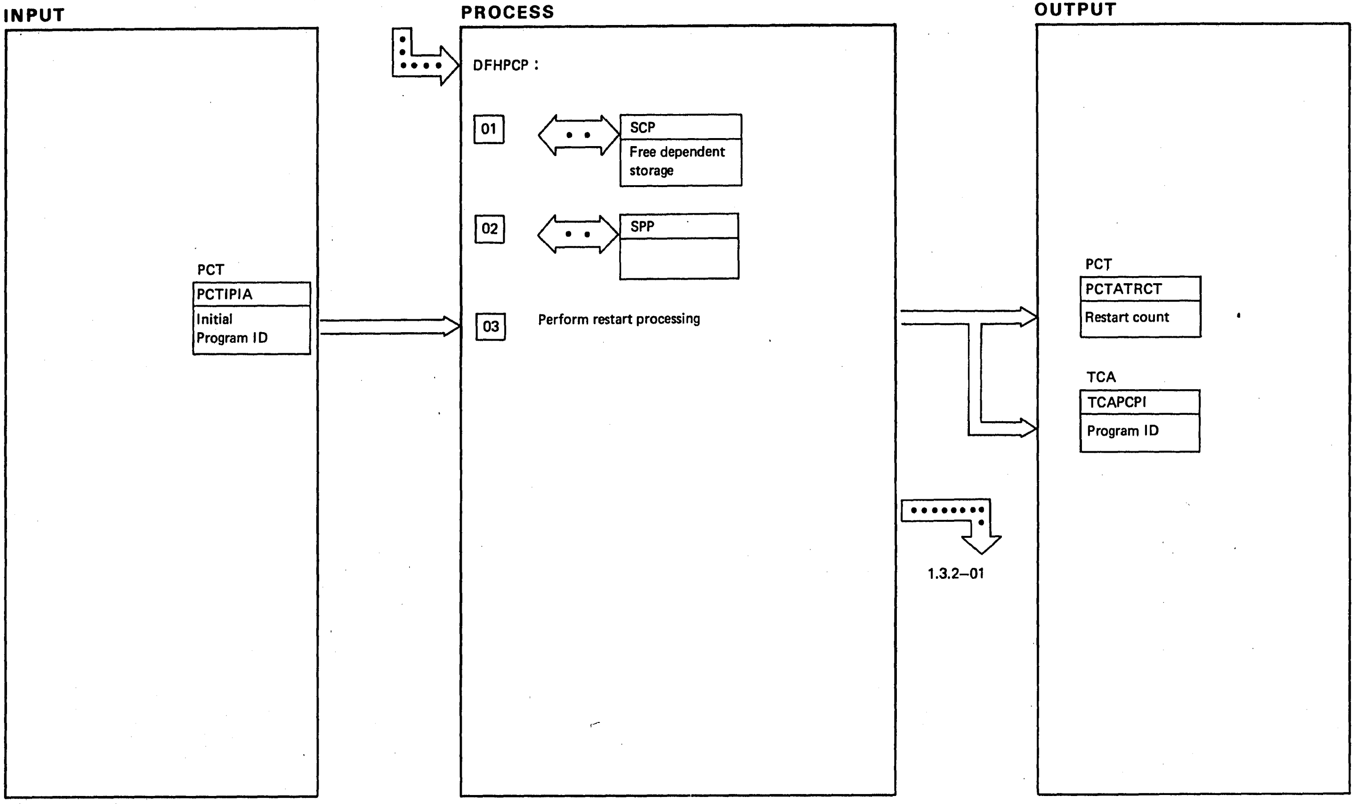
Diagram - 1.3.10-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The contents of the registers are saved. | | PCPL1IN | | | | | |
| 02 The CSA address is saved in the location specified by the PL/I call statement. | | | | | | | |
| 03 | | PCBCICS | | | | | |
| 04 | | PCHLLTR | | | | | |
| 05 | | PCPL1XN | | | | | |

PL/I Interface

Diagram - 1.3.10-01

Page 1 of 1



Restart task after ABEND

Diagram 1.3.11–01

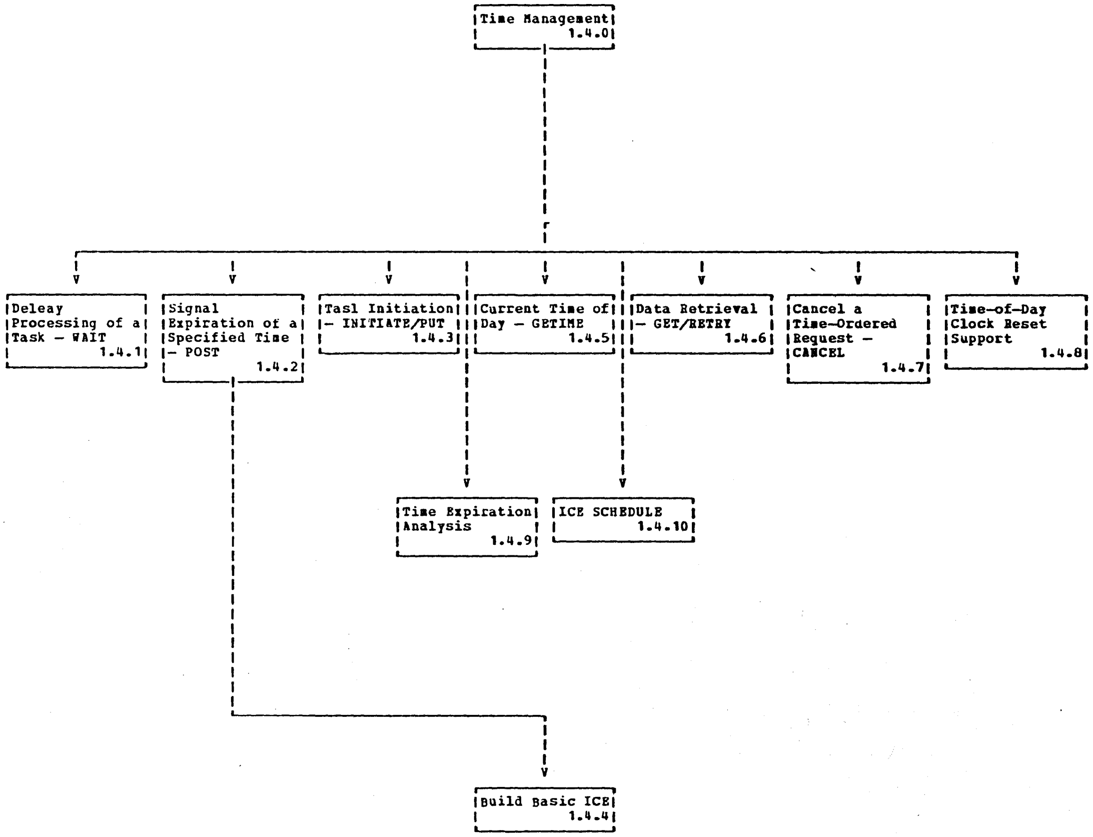
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Free user class storage, not TCA. | | PCRETRY | |
| 02 Re-initialize Dynamic Log. | | | |
| 03 Set 'restarted' flag, clear ABEND code, and 'synchpointed' flag. Move initial program ID into PCXCTL. | | | |

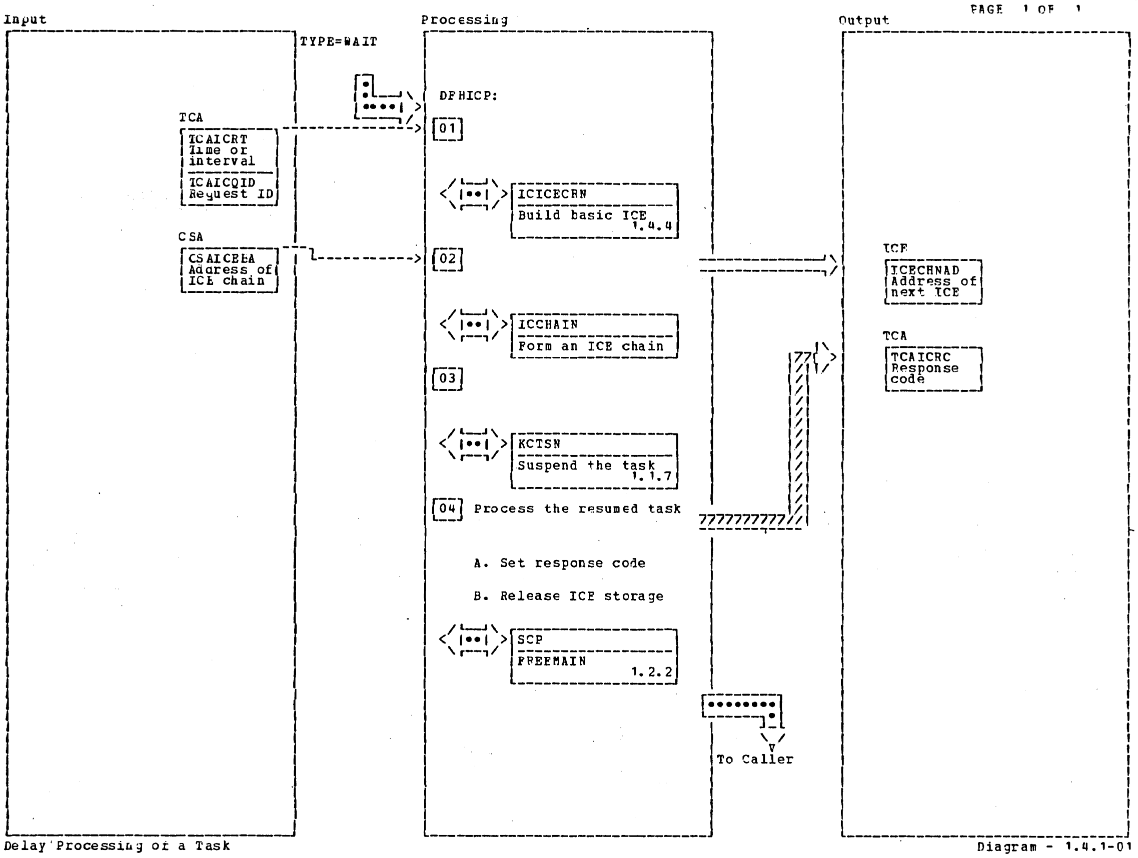
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Restart task after ABEND

Diagram 1.3.11–01

CHART: ICPVT
PAGE 1 OF 1


Time Management 1.4.0
Deleay Processing of a Task — WAIT 1.4.1
Signal Expiration of a Specified Time — POST 1.4.2
Tasl Initiation — INITIATE/PUT 1.4.3
Current Time of Day — GETIME 1.4.5
Data Retrieval — GET/RETRY 1.4.6
Cancel a Time-Ordered Request — CANCEL 1.4.7
Time-of-Day Clock Reset Support 1.4.8
Time Expiration Analysis 1.4.9
ICE SCHEDULE 1.4.10
Build Basic ICE 1.4.4



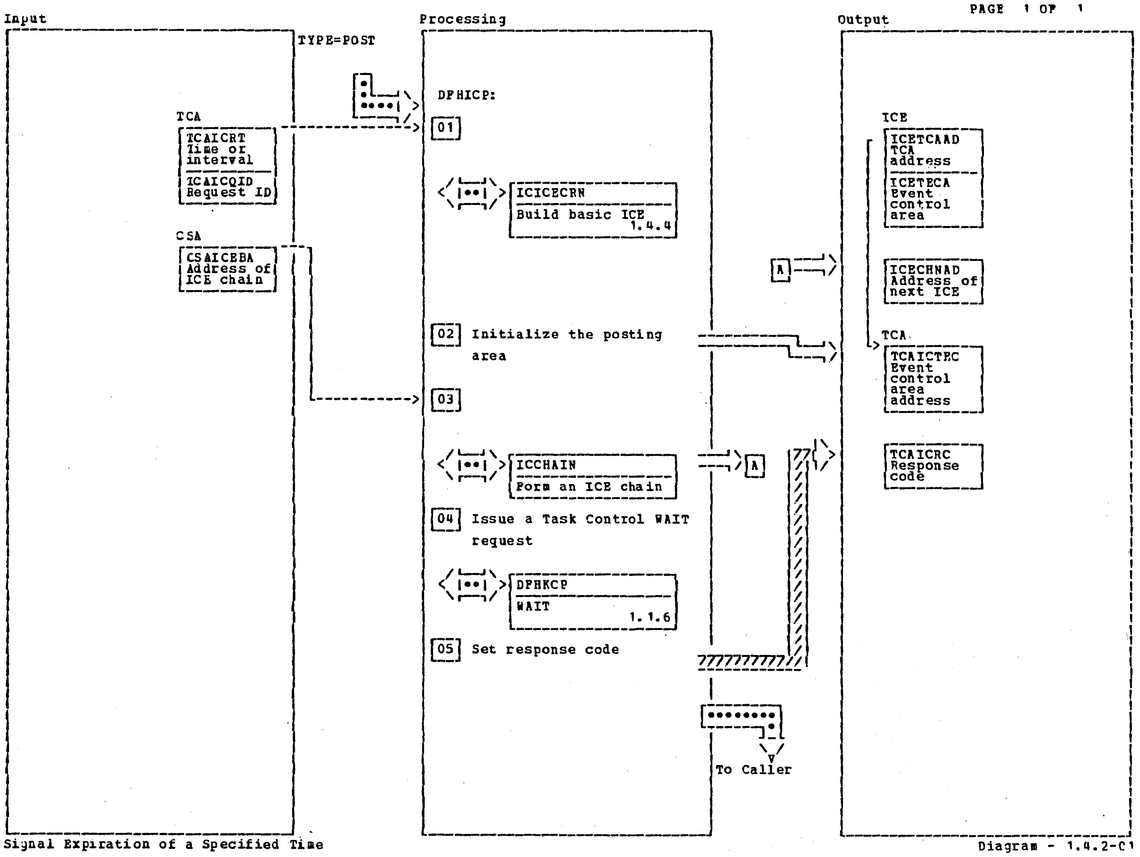
Delay Processing of a Task

Diagram - 1.4.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | ICWAITN | 1.4.1 | | | | |
| 02 The chain sequence is formed according to expiration times. | ICCHAIN | | | | | | |
| 04 | | | 1.1.2 | | | | |
| A. The suspended task is resumed upon expiration of the interval or time requested. When the suspended task is resumed, process begins immediately. If the expiration time had elapsed when the original request was made, an appropriate response is returned. | | | 1.4.9 | | | | |
| B. A Storage Control FREEMAIN is issued. | | ICWTRICE | 1.2.2 | | | | |

Delay Processing of a Task

Diagram - 1.4.1-C1



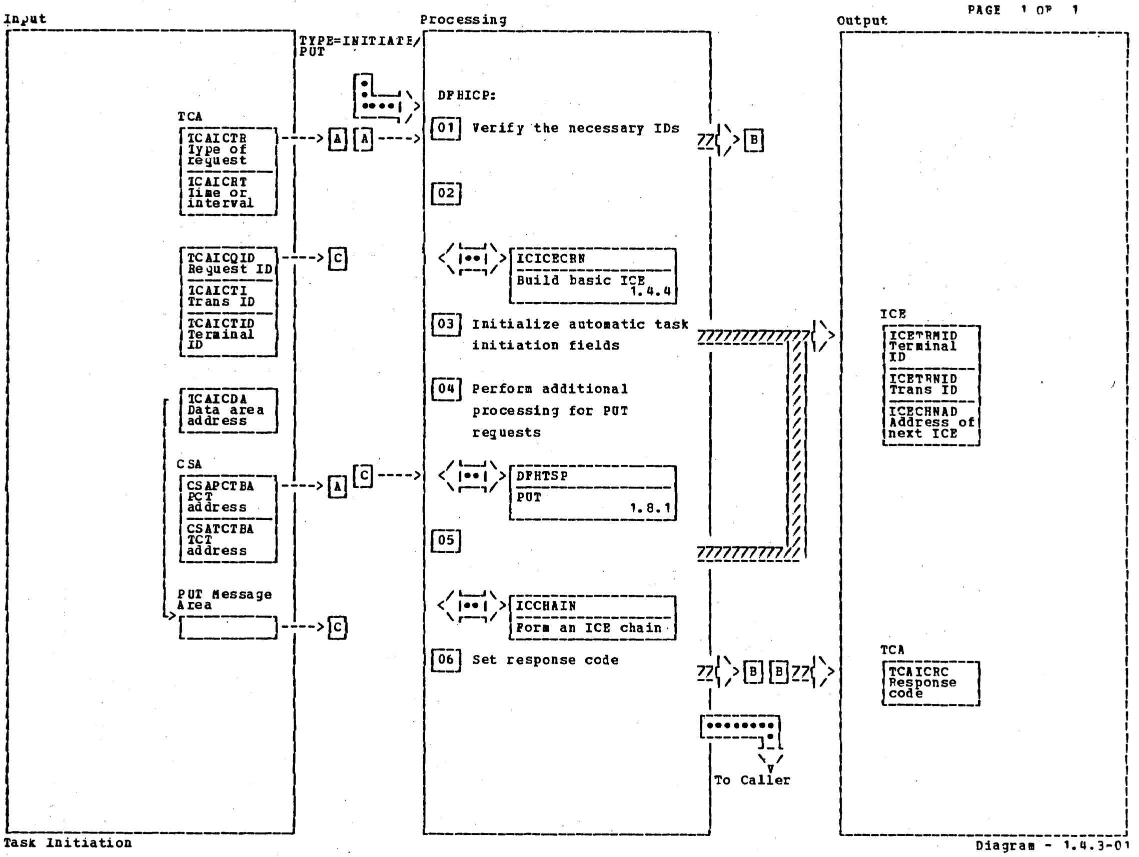
Signal Expiration of a Specified Time

Diagram - 1.4.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | ICPOSTN | | | | | |
| 02 The event control area that will be posted at the requested time or interval is part of the ICE and is posted upon expiration of the requested time of interval. | | | 1.4.9 | | | | |
| 03 The chain sequence is formed according to expiration time. | | | | | | | |
| 04 Expiration of time-dependent event is detected by Task Control Task Dispatcher. The Task Control WAIT is issued to cause immediate posting if the expiration time had elapsed when the original request was made. | | | | | | | |

Signal Expiration of a Specified Time

Diagram - 1.4.2-01



Task Initiation

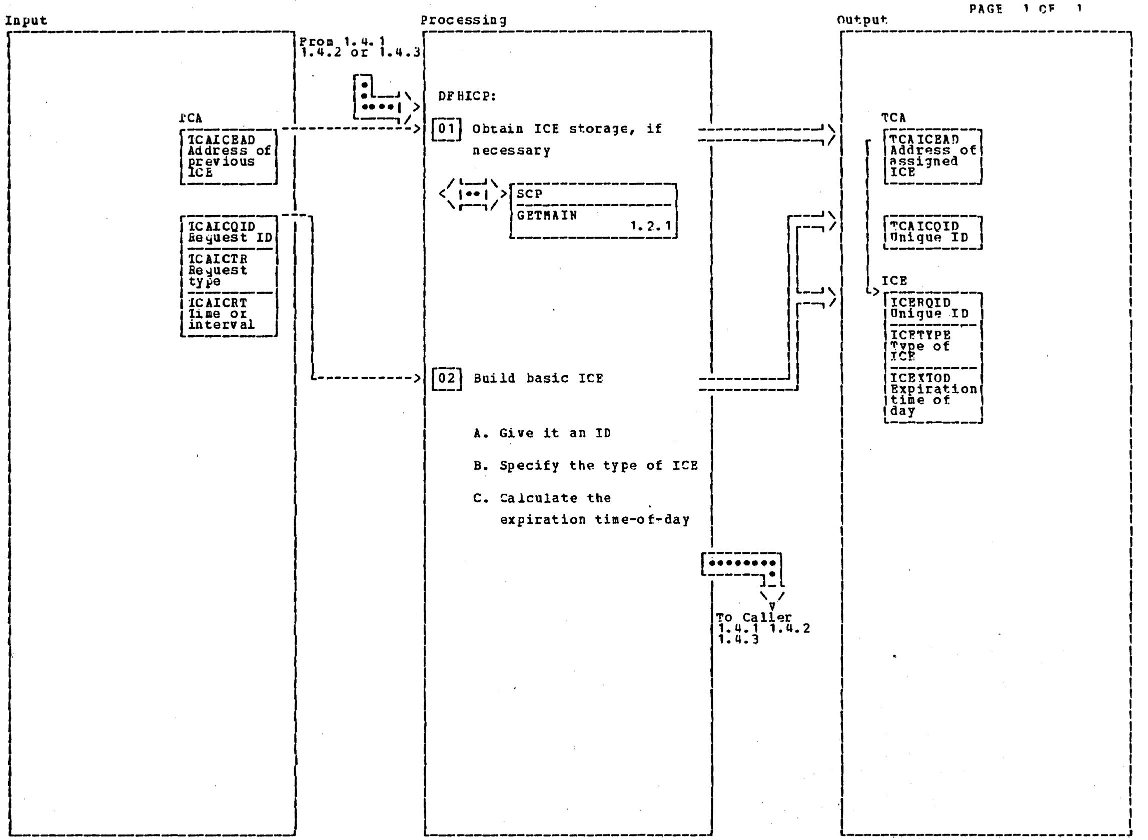
Diagram - 1.4.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The transaction ID is verified by scanning the PCT for an entry that matches the ID of the original request. If no match, set return code and return to caller. | | ICPCTSN | | | | | |
| A. For terminal-dependent tasks, if a terminal ID was specified in the original request, it is verified by scanning the TCTTE for a matching entry. If no match, set the appropriate return code and return to caller. | | ICTSTERM | | | | | |
| 04 All time-ordered messages are stored using Temporary Storage facilities. The request ID is used as data ID for the stored messages. | | | | | | | |
| 05 The chain sequence is formed according to expiration times. | | | | | | | |

Task Initiation

Diagram - 1.4.3-01

PAGE 1 OF 1



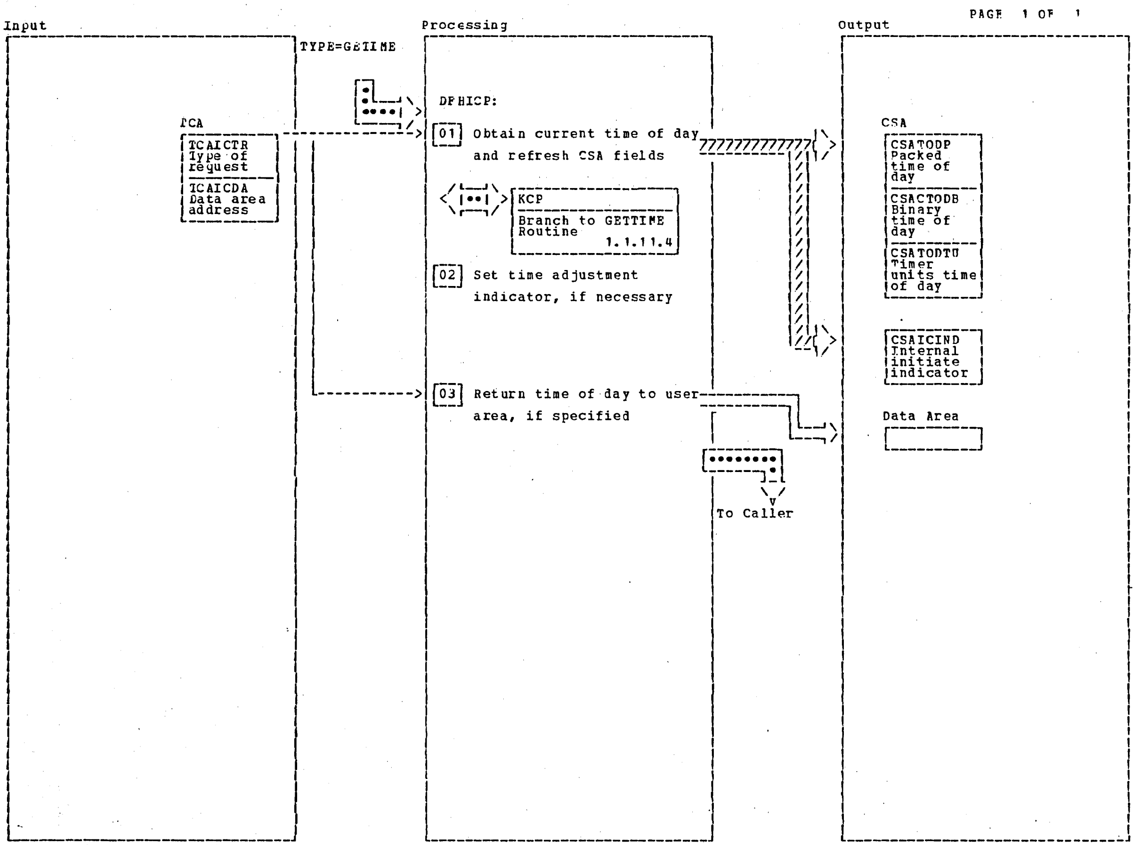
Build Basic ICE

Diagram - 1.4.4-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 If the task had previously issued a POST request, an ICE would remain assigned to the task and it is reused for any Interval Control service requiring an ICE. | | ICICECRN | |
| 02 | | ICICECRX | |
| A. If no ID was supplied with the request, a unique ID is developed using the counter in CSAUNQID. The CICS component code, DFH, prefixes the system-generated ID. | ICREQIDN | | |
| B. Types of ICEs and corresponding codes are shown in the Interval Control Element data area in Chapter 5. | | | |
| C. The user-supplied value is converted to 300ths of a second in binary form. If an interval value was specified, it is added to the current time of day to produce an expiration time. If a time value was specified, it is used alone as the expiration time. | ICCNVRTN | | |

Build Basic ICE

Diagram - 1.4.4-01


PAGE 1 OF 1
Input
Processing
Output
TYPE=GETIME
TCA
TCAICTR Type of request
TCAICDA Data area address
DFHICP:
01 Obtain current time of day and refresh CSA fields
KCP
Branch to GETTIME Routine 1.1.11.4
02 Set time adjustment indicator, if necessary
03 Return time of day to user area, if specified
To Caller
CSA
CSATODP Packed time of day
CSACTODB Binary time of day
CSATODTU Timer units time of day
CSAICIND Internal initiate indicator
Data Area

Current Time of Day

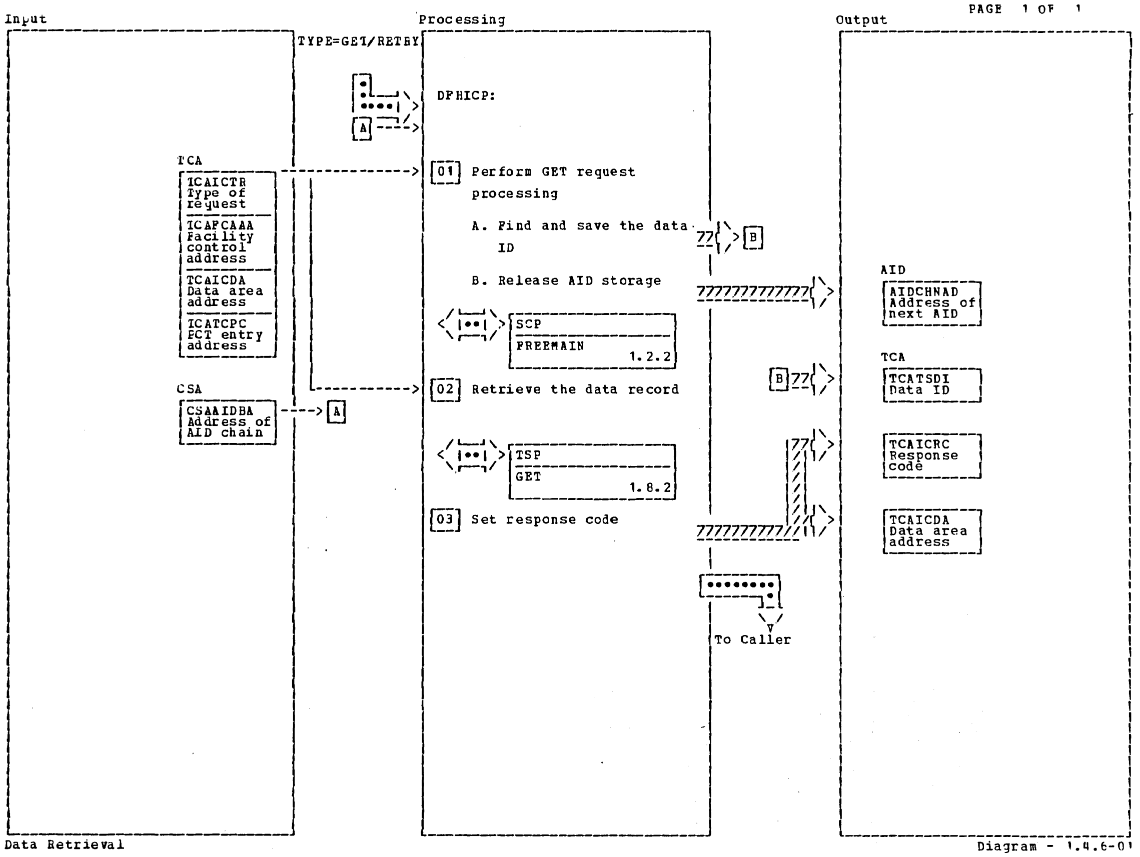
Diagram - 1.4.5-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The requested form of the time of day is obtained through Task Control in: | | ICGTIMEN | |
| Binary | ICTODBIN | | |
| Packed decimal | ICTODPAC | | |
| 02 Within the above subroutines, a test is made to see if the current time of day obtained from the operating system is a value smaller than a previously obtained value. If so, it indicates that the operating system clock has been reset (at midnight, for example). An indicator in the CSA is then set to cause the Time Adjustment program to be initiated in the Time Expiration Analysis. | | | 1.4.9 |
| 03 | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Current Time of Day

Diagram - 1.4.5-01


PAGE 1 OF 1
Input
Processing
Output
TYPE=GET/RETRY
DFHICP:
TCA
TCAICTR Type of request
TCAFCAAA Facility control address
TCAICDA Data area address
TCATCPC PCT entry address
CSA
CSAAIDBA Address of AID chain
A
01 Perform GET request processing
A. Find and save the data ID
B. Release AID storage
SCP
FREEMAIN
1.2.2
02 Retrieve the data record
TSP
GET
1.8.2
03 Set response code
B
To Caller
AID
AIDCHNAD Address of next AID
TCA
TCATSDI Data ID
TCAICRC Response code
TCAICDA Data area address

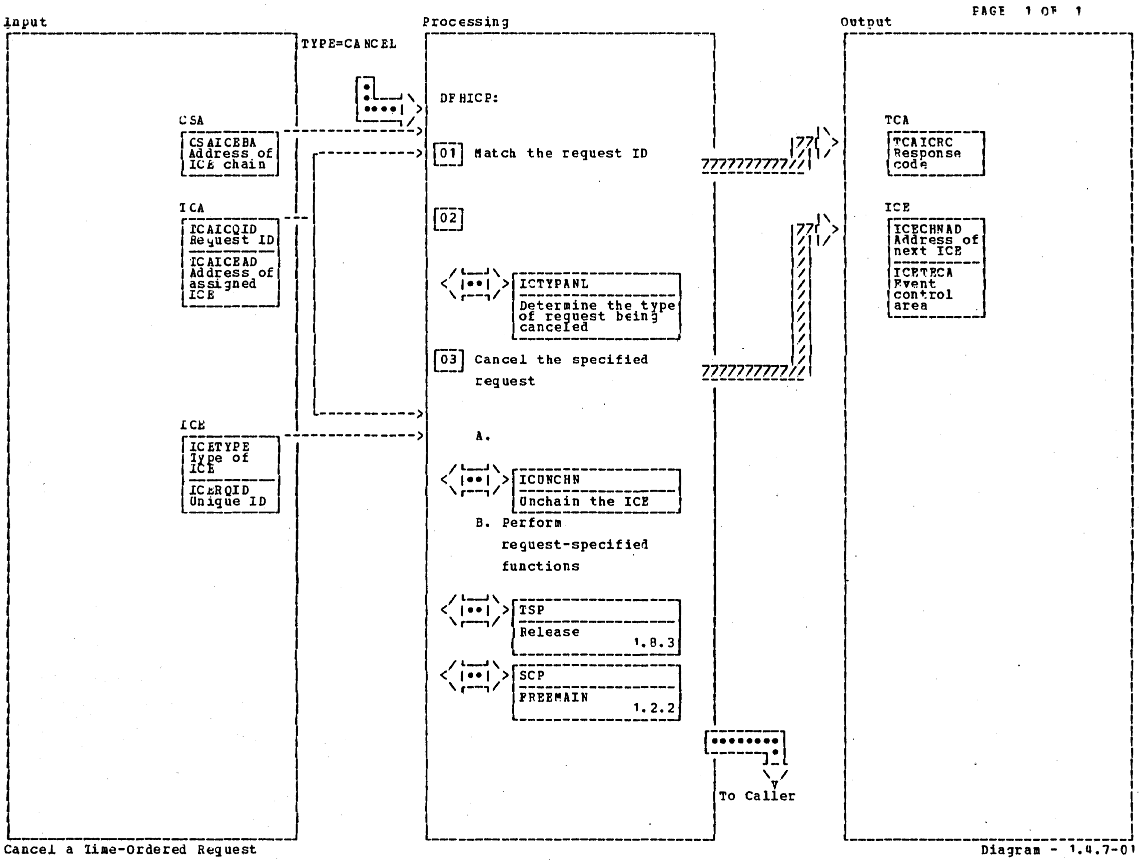
Data Retrieval

Diagram - 1.4.6-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | | |
| A. If the task is associated with a terminal, the AID chain is scanned for an AID created by a PUT request that has the same terminal ID as in the task's associated TCTTE and the same transaction ID as in the task's associated PCT. If found, the data ID is stored in the TCA. If no matching AID is found, an end-of-file code condition is returned to the caller. If the task is not associated with a terminal, the data ID is in the AID pointed to by the facility control address in the TCA. | | ICGTTTDM | |
| 02 The data associated with the request is retrieved from Temporary Storage. If an I/O error is encountered in Temporary Storage, or if a dummy Temporary Storage module exists, an appropriate return code is set. | | ICREADAT | |
| 03 If no data area was supplied with the request, the data area address is returned with the response code to the requesting routine. | | | |

Data Retrieval

Diagram - 1.4.6-01



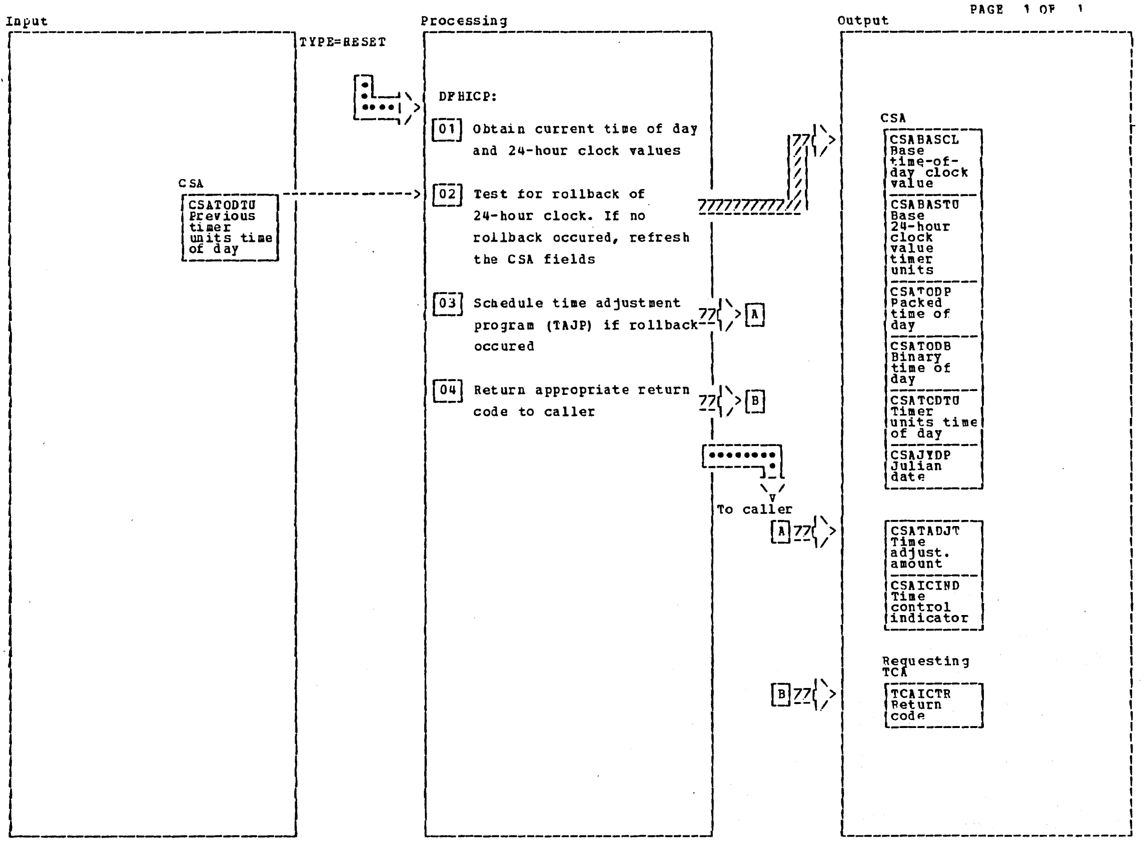
Cancel a Time-Ordered Request

Diagram - 1.4.7-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The ICE chain is scanned for an ICE with a matching request ID. | | ICCANCLN | |
| If no request ID was given with the cancelation request, the ICE previously assigned to the task is unchained via a subroutine, and its storage is released through Storage Control. | ICUNCHN | | |
| 02 This subroutine also directs the processing to the appropriate cancelation processing. | | | |
| 03 | | | |
| B. A WAIT request is canceled by resuming the waiting task via a Task Control RESUME. | | ICWTXPON | 1.4.1<br>1.1.8 |
| A POST request is canceled by turning on the event completion bits compatible with both OS/VS ECBs and DOS/VS CCBs in the event control area assigned to task. | | ICPSXPDN | 1.4.2 |
| A PUT request is canceled by releasing the data stored in Temporary Storage and by releasing the ICE storage. | | ICINCNLN | 1.4.3<br>1.8.3 |
| C. An INITIATE request is canceled by releasing the ICE storage. | | | 1.4.3<br>1.2.2 |

Cancel a Time-Ordered Request

Diagram - 1.4.7-01


PAGE 1 OF 1
Input
Processing
Output
TYPE=RESET
DFHICP:
01 Obtain current time of day and 24-hour clock values
02 Test for rollback of 24-hour clock. If no rollback occured, refresh the CSA fields
03 Schedule time adjustment program (TAJP) if rollback occured
04 Return appropriate return code to caller
A
B
To caller
CSA
CSATODTU Previous timer units time of day
CSA
CSABASCL Base time-of-day clock value
CSABASTU Base 24-hour clock value timer units
CSATODP Packed time of day
CSATODB Binary time of day
CSATCDTU Timer units time of day
CSAJYDP Julian date
CSATADJT Time adjust. amount
CSAICIND Time control indicator
Requesting TCA
TCAICTR Return code

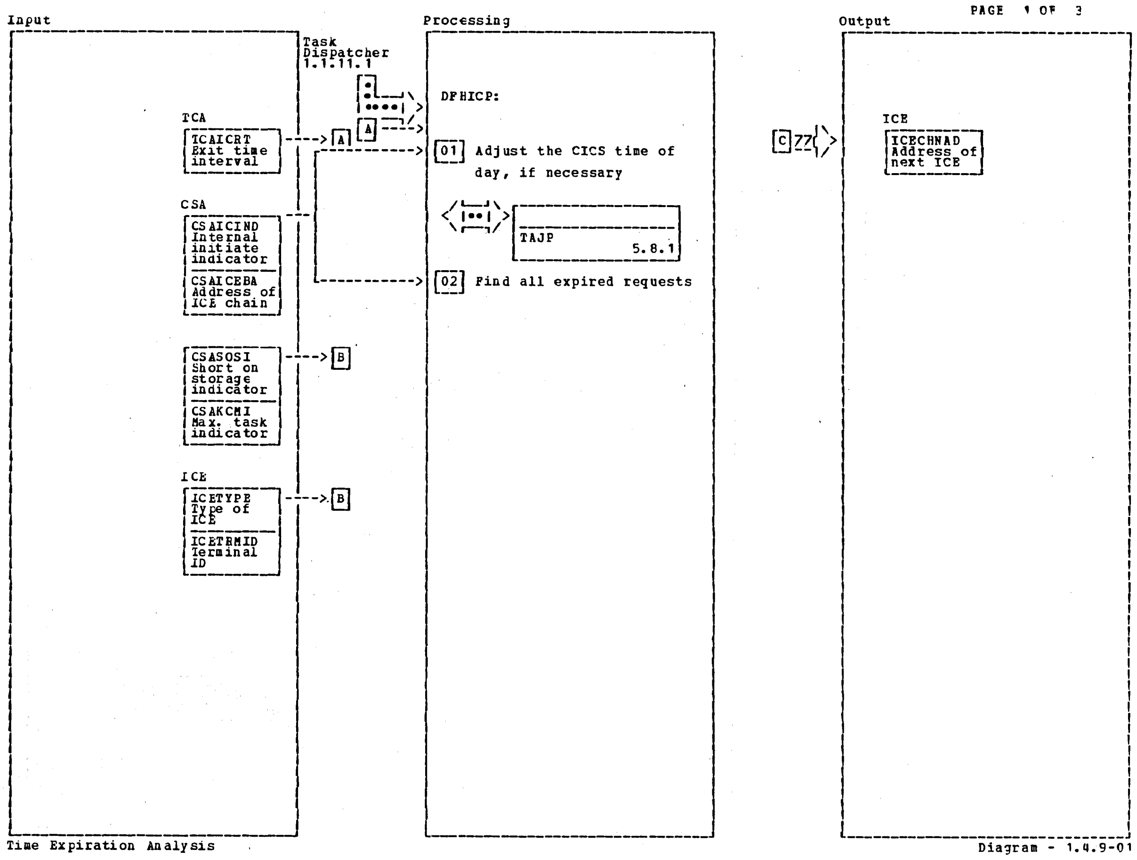
Time-of-Day Reset Routine

Diagram - 1.4.8-C1

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Current time-of-day clock is obtained via a Store Clock instruction. If tne condition code indicates a clock failure return a 'clock inoperative' return code.<br>Obtain the 24-hour clock (timer units) value from the operating system. | ICRSCLNG | ICRESETN | |
| 02 A clock rollback condition exists if the current timer units value is smaller than the previous timer units value. The base clock and timer units values and Julian date are extablished and the other time-of-day values in the CSA are obtained through task control (KCP). A normal return code is returned. | ICDIRXII | KCTODRTN | |
| 03 If a clock rollback is detected, the amount of change is saved in the CSA for TAJP. If TAJP is not scheduled it will be scheduled by setting the appropriate indicators in the CSA. The time adjustment task is initiated by the task dispatcher. A 'time adjustment is scheduled' return code is returned. | | ICRSTAJB | 1.4.9 |

Time-of-Day Reset Routine

Diagram - 1.4.8-C1


PAGE 1 OF 3
Input
Processing
Output
Task Dispatcher 1.1.11.1
TCA
ICAICRT Exit time interval
A
CSA
CSAICIND Internal initiate indicator
CSAICEBA Address of ICE chain
CSASOSI Short on storage indicator
CSAKCMI Max. task indicator
B
ICE
ICETYPE Type of ICE
ICETRMID Terminal ID
B
DFHICP:
A
01 Adjust the CICS time of day, if necessary
TAJP 5.8.1
02 Find all expired requests
C
ICE
ICECHNAD Address of next ICE

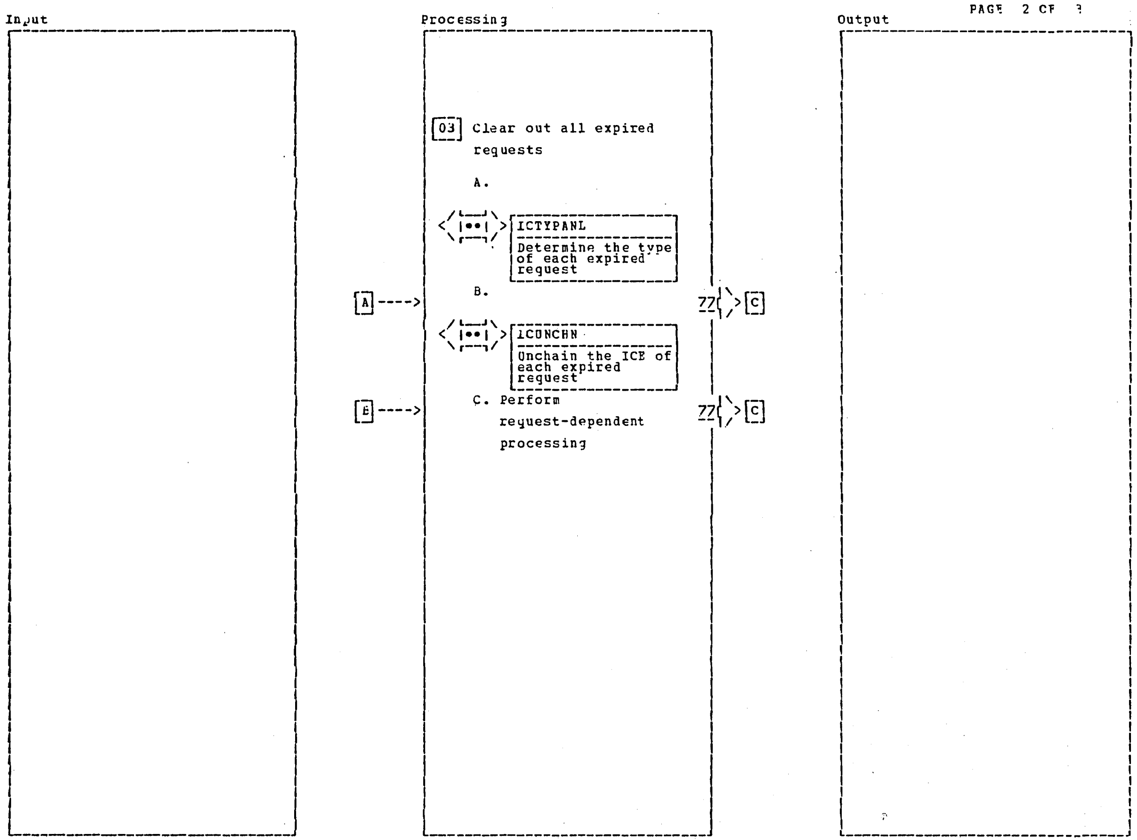
Time Expiration Analysis

Diagram - 1.4.9-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 If tne operating system clock has been reset (at midnight,for example), a Task Control ATTACH is issued for the Time Adjustment program. One of the indicators which were tested to check for the reset condition is then turned off to indicate that the condition is being handled. | | ICEXPANL | 1.1.1<br>5.8.1 |
| 02 Each successive ICE of the ICE chain is tested to see if its expiration time has passed. | | ICEXPTEN | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| | | | |

PAGE 2 OF 3



03 Clear out all expired requests

A.

ICTYPANL
Determine the type of each expired request

B.

ICUNCHN
Unchain the ICE of each expired request

C. Perform request-dependent processing

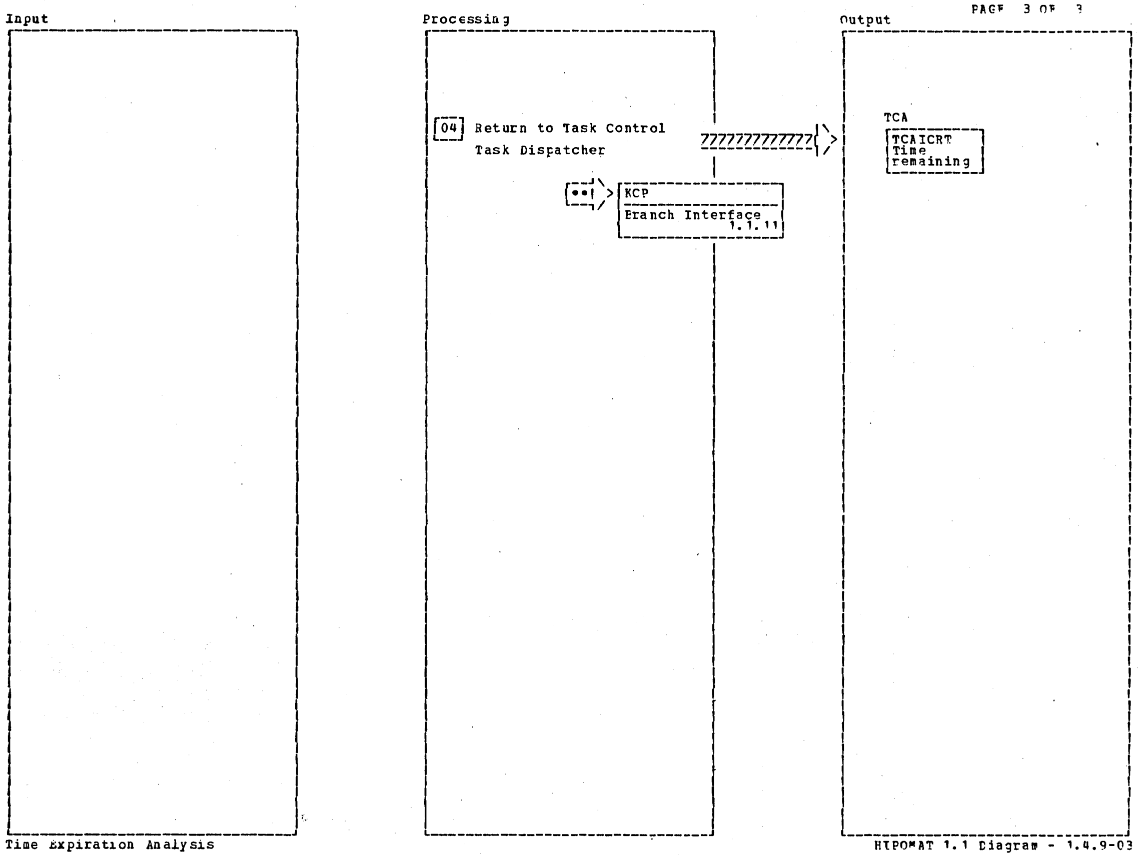
Time Expiration Analysis

Diagram - 1.4.9-02

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 03 | | | |
| A. This subroutine also directs the processing to the appropriate expiration function. | | | |
| C. For expired WAIT requests, the waiting task is resumed via a Task Control RESUME. | | ICWIXPDN | 1.1.8 |
| For expired POST requests, the event completion bits compatible with both OS/VS ECBs and DOS/VS CCBs are turned on in the event control area assigned to the task. | | ICPSXPDN | |
| For expired INITIATE or PUT requests that are not terminal-dependent,the task is immediately initiated via a Task Control ATTACH, provided that the system is not under stress. If the system is under stress, the expired ICE remains on the ICE chain and an attempt will be made to initiate the task the next time through the Task Dispatcher. | | ICINXPDN | 1.1.1 |
| For PUT requests, the ICE address is used as the facility control address. The task can then retrieve the data via an Interval Control GET request. | | | |
| For expired terminal-dependent INITIATE or PUT requests, the task is scheduled for initiation via a Task Control SCHEDULE, to synchronize the initiation with the availability of the terminal. | | | 1.1.9 |
| The ICE address is passed to the Task Control SCHEDULE facilities as an AID address. | | ICSCAIN | |
| For each expired request, the user-specified exit is taken if one was given. | | | |

Time Expiration Analysis

Diagram - 1.4.9-02



Time Expiration Analysis

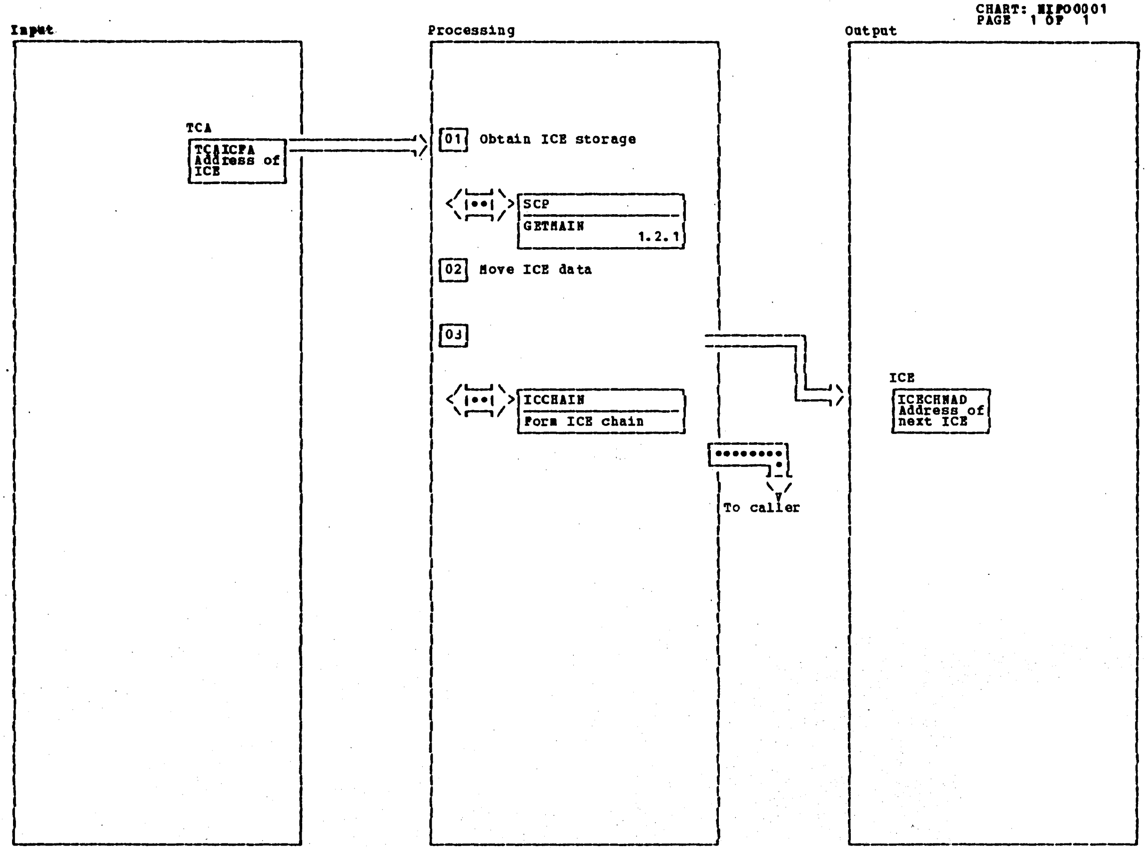
HIPOMAT 1.1 Diagram - 1.4.9-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 When the first unexpired ICE is encountered during the scan for expired requests, the time remaining until this ICE expires is returned to the requesting routine (the Task Dispatcher). | | ICEXPTEN | | | | | |

Time Expiration Analysis

Diagram - 1.4.9-03

CHART: HIPO0001
PAGE 1 OF 1

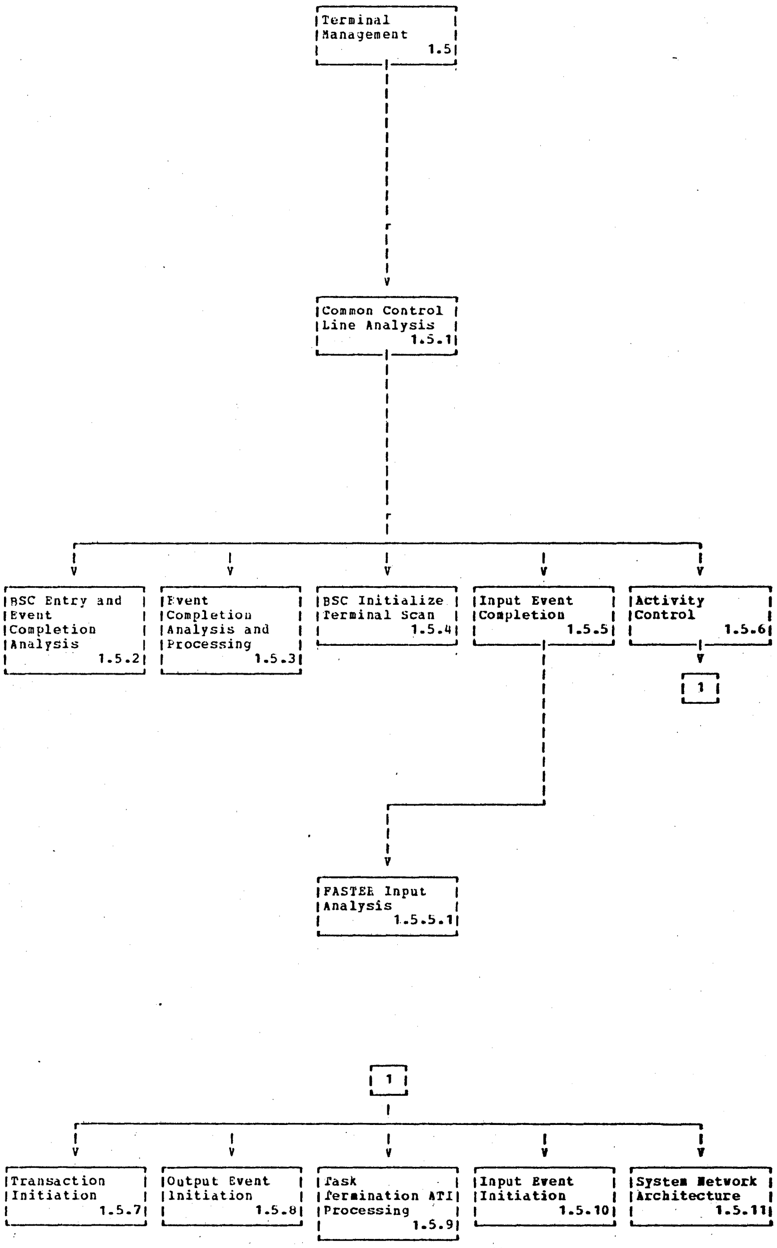


ICSHEDN - ICE Schedule    Diagram - 1.4.10-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | ICSCHEDN | | | | | |
| 02 Move ICE data to new area. | | | | | | | |
| 03 Merge ICE on chain by expiration time. | | | | | | | |

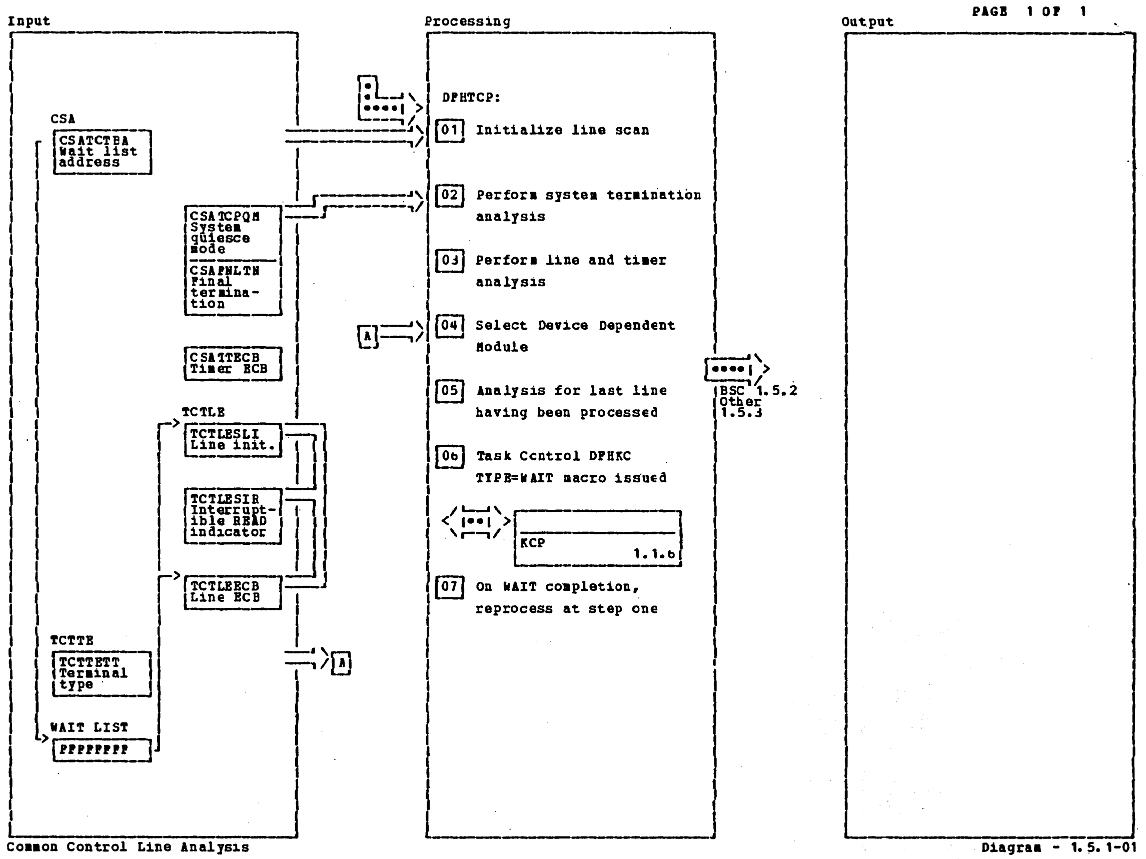
ICSHEDN - ICE Schedule    Diagram - 1.4.10-01

CHART: HIPO0001
PAGE 1 OF 1


Terminal Management 1.5
Common Control Line Analysis 1.5.1
BSC Entry and Event Completion Analysis 1.5.2
Event Completion Analysis and Processing 1.5.3
BSC Initialize Terminal Scan 1.5.4
Input Event Completion 1.5.5
Activity Control 1.5.6
1
FASTER Input Analysis 1.5.5.1
1
Transaction Initiation 1.5.7
Output Event Initiation 1.5.8
Task Termination ATI Processing 1.5.9
Input Event Initiation 1.5.10
System Network Architecture 1.5.11

Visual Table of Contents

Diagram - 1.5-01


PAGE 1 OF 1
Input
Processing
Output
CSA
CSATCTBA Wait list address
CSATCPQM System quiesce mode
CSAFNLTM Final termina-tion
CSATTECB Timer ECB
TCTLE
TCTLESLI Line init.
TCTLESIR Interrupt-ible READ indicator
TCTLEECB Line ECB
TCTTE
TCTTETT Terminal type
WAIT LIST
FFFFFFFF
A
DFHTCP:
01 Initialize line scan
02 Perform system termination analysis
03 Perform line and timer analysis
04 Select Device Dependent Module
05 Analysis for last line having been processed
06 Task Control DFHKC TYPE=WAIT macro issued
KCP 1.1.6
07 On WAIT completion, reprocess at step one
BSC 1.5.2
Other 1.5.3

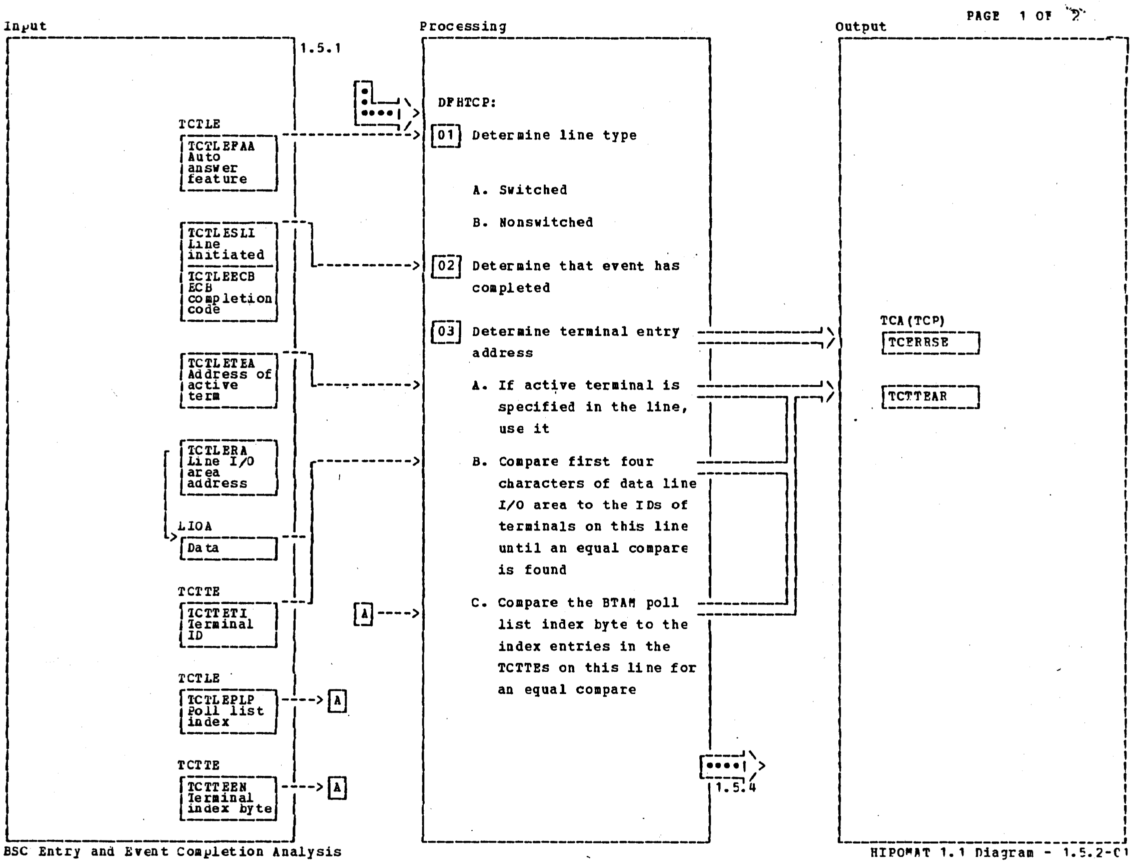
Common Control Line Analysis

Diagram - 1.5.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The wait list which contains the addresses of each line entry is initialized. | | TCCSIR | |
| 02 If system is in quiesce mode, only writes are performed. Quiesce mcde occurs when only the system termination task remains. | | TCCSIN | |
| However, all other tasks may be gone with a WRITE remaining on the terminal. If final termination is posted a Task Control WAIT is issued. | | | STEP 6 |
| 03 If timer has elapsed, all lines will be processed if poseted complete, if interruptible or not initiated. | | TCCCTECB | |
| If timer has not been posted, then only lines that are posted complete are processed. Otherwise advance to next line. | | | |
| 04 Perform Device Dependent Module selection. | | TCCCLTMA | |
| For BSC lines | | | 1.5.2 |
| All others | | | 1.5.3 |
| 05 When the end of wait list is reached, then all lines have been processed. | | TCCCTMR | |
| 06 | | TCCCANAEA | |

Common Contrcl Line Analysis

Diagram - 1.5.1-01


Input
Processing
Output
PAGE 1 OF 2
1.5.1
TCTLE
TCTLEPAA Auto answer feature
TCTLESLI Line initiated
TCTLEECB ECB completion code
TCTLETEA Address of active term
TCTLERA Line I/O area address
LIOA
Data
TCTTE
TCTTETI Terminal ID
TCTLE
TCTLEPLP Poll list index
A
TCTTE
TCTTEEN Terminal index byte
A
A
DFHTCP:
01 Determine line type
A. Switched
B. Nonswitched
02 Determine that event has completed
03 Determine terminal entry address
A. If active terminal is specified in the line, use it
B. Compare first four characters of data line I/O area to the IDs of terminals on this line until an equal compare is found
C. Compare the BTAM poll list index byte to the index entries in the TCTTEs on this line for an equal compare
1.5.4
TCA(TCP)
TCERRSE
TCTTEAR

BSC Entry and Event Completion Analysis

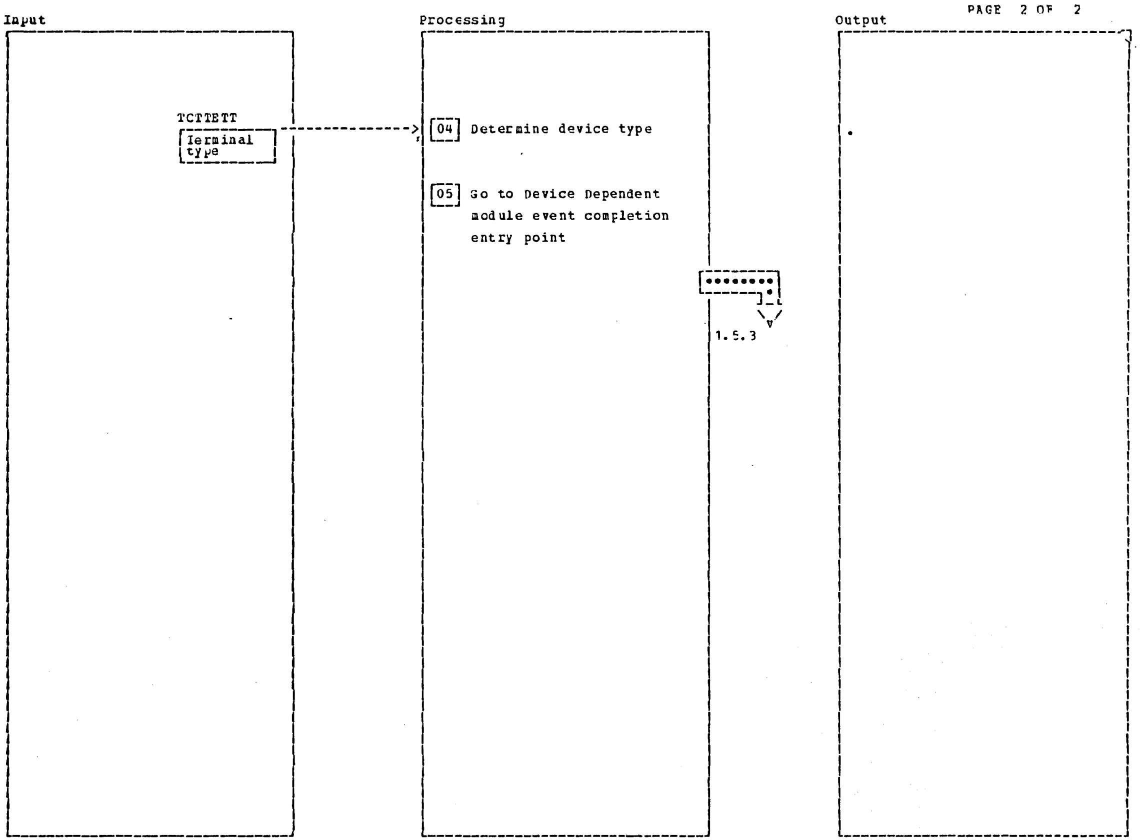
HIPOMAT 1.1 Diagram - 1.5.2-C1

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 BSC lines may have multiple device types. Because of this, Terminal Control must determine which terminal entry is associated with an event completion before processing the completion. | | DFHBSEA | |
| A. Switched line | | DFHBSDEA | |
| B. Nonswitched line | | DFHBNDEA | |
| 02 If either the line initiated indicator is not set or the ECB is not posted complete, exit this routine to initialize Terminal Scan. | | | |
| 03 Switched | | TCBSDTAS | |
| Nonswitched | | TCBSTES | |
| A. This will be the means by which the correct terminal entry will be determined once initial contact is established. On a point-to-point nonswitched line, the TCTEAR will be loaded pointing to the first (and only) entry without further testing. | | | |
| B. This applies to switched lines only. If the transmission code is ASCII, the data will be translated to EBCDIC before doing the comparison. | | | |
| C. If all terminals on a line are searched without finding an equal compare, the TCT search error will be set up and TACP attached. If a WRITE type operation has completed, go to initialize Terminal Scan. | | | 1.5.4 |

BSC Entry and Event Completion Analysis

Diagram - 1.5.2-C1

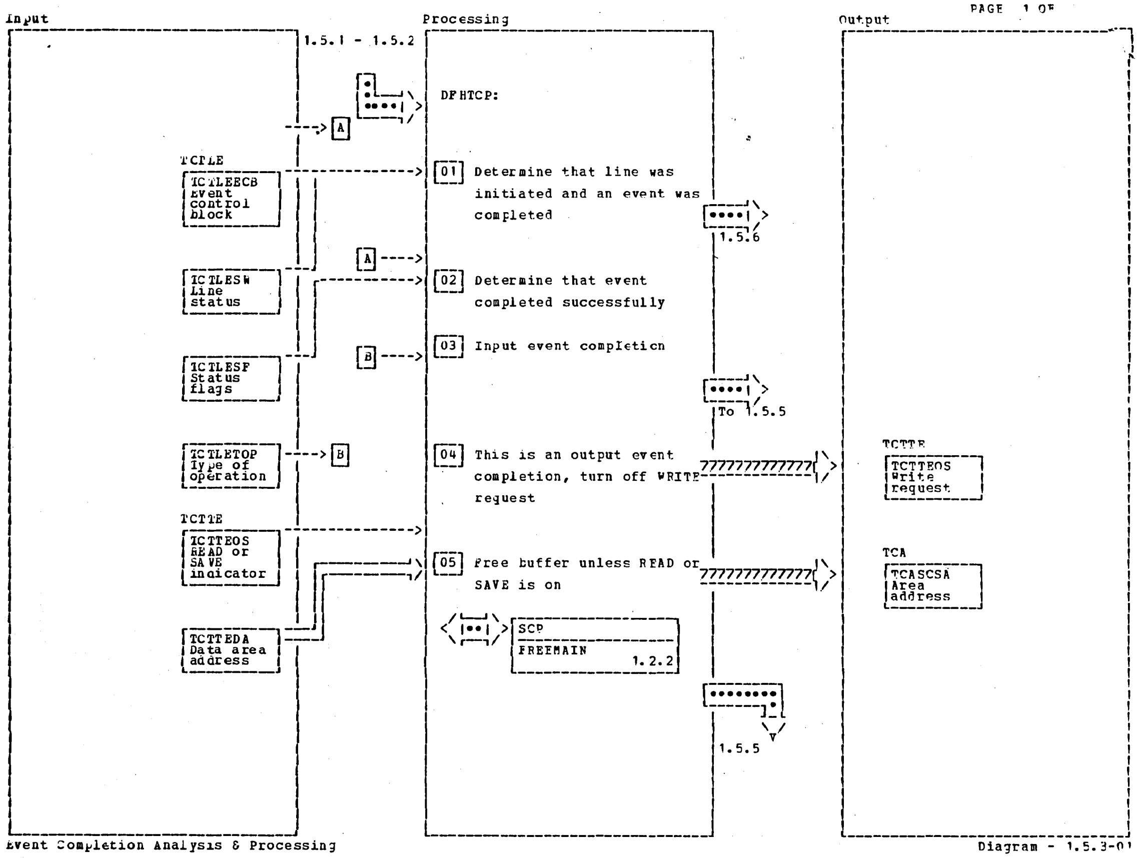
PAGE 2 OF 2



BSC Entry and Event Completion Analysis

Diagram - 1.5.2-C2

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| D. If a WRITE type command has completed, go to Initialize Terminal Scan. | | | 1.5.4 | | | | |
| A failure to find an equal compare after reaching the last terminal will result in setting up a Terminal Table search error and attaching TACP. | | | | | | | |
| 04 Switched | | TCBSDDEC | | | | | |
| Nonswitched | | TCBSNDEC | | | | | |


PAGE 1 OF
Input
Processing
Output
1.5.1 - 1.5.2
DFHTCP:
A
TCTLE
TCTLEECB Event control block
01 Determine that line was initiated and an event was completed
1.5.6
TCTLESW Line status
A
02 Determine that event completed successfully
TCTLESF Status flags
B
03 Input event completion
To 1.5.5
TCTLETOP Type of operation
B
04 This is an output event completion, turn off WRITE request
TCTTE
TCTTEOS Write request
TCTTE
TCTTEOS READ or SAVE indicator
05 Free buffer unless READ or SAVE is on
TCA
TCASCSA Area address
TCTTEDA Data area address
SCP
FREEMAIN
1.2.2
1.5.5

Event Completion Analysis & Processing

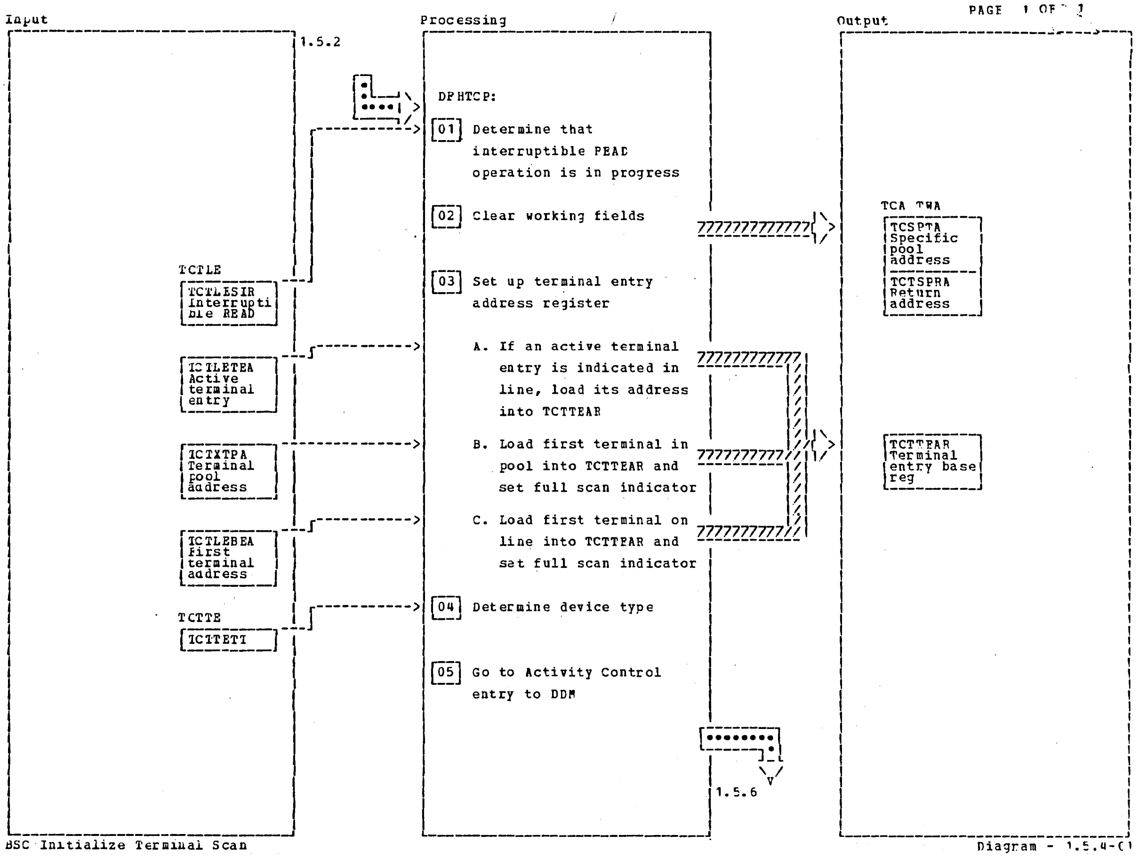
Diagram - 1.5.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| Start/Stop lines will enter from 1.5.1. BSC line will enter from 1.5.2. | | | |
| 01 If either the line was not initiated or the ECB is not posted, go to activity Control. Start/stop terminals 2741 and 7770 are handled in terminal dependent module. | | TCASSEA | 1.5.6 |
| 02 Determine that the request did not complete with a negative response and that the completion code was '7F'. Binary synchronous terminals, except 2770 and 2780 are handled in terminal dependent modules. | | TCBSDECA | |
| 03 EXITO - If operation completing is a READ type request. | | | 1.5.5 |
| 05 If either the READ or SAVE bits are on, retain the storage for use by the transaction. | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| | | | |

Event Completion Analysis & Processing

Diagram - 1.5.3-01


PAGE 1 OF 1
Input
Processing
Output
1.5.2
DFHTCP:
01 Determine that interruptible READ operation is in progress
02 Clear working fields
03 Set up terminal entry address register
A. If an active terminal entry is indicated in line, load its address into TCTTEAR
B. Load first terminal in pool into TCTTEAR and set full scan indicator
C. Load first terminal on line into TCTTEAR and set full scan indicator
04 Determine device type
05 Go to Activity Control entry to DDM
1.5.6
TCTLE
TCTLESIR Interruptible READ
TCTLETEA Active terminal entry
TCTXTPA Terminal pool address
TCTLEBEA First terminal address
TCTTE
TCTTETT
TCA TWA
TCSPTA Specific pool address
TCTSPRA Return address
TCTTEAR Terminal entry base reg

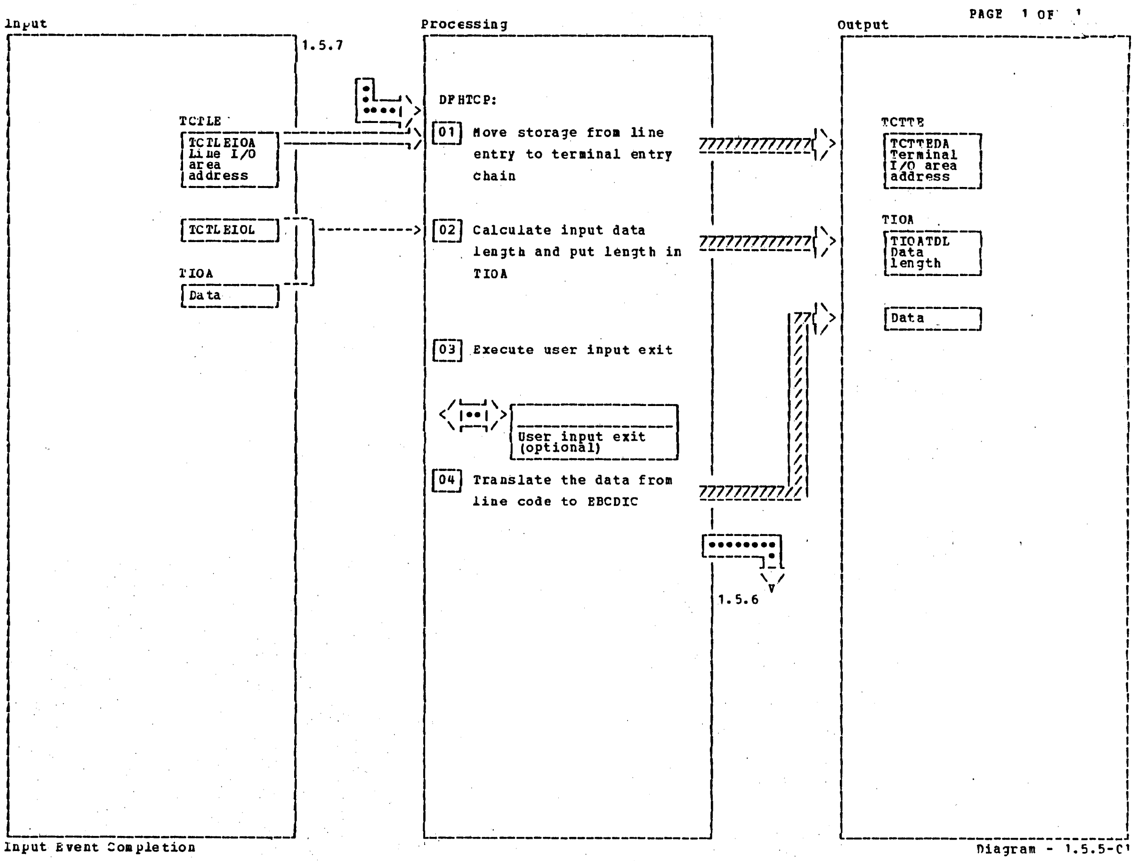
BSC Initialize Terminal Scan

Diagram - 1.5.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Switched | | TCBSDATS | 1.5.1 STEP 3 | | | | |
| Nonswitched | | TCBNDATS | | | | | |
| If the operation in progress is not an interruptible READ, go to Line Advance. | | | | | | | |
| 02 If the line is out of service, perform a RESET POLL and go to Line Advance. | | | 1.5.1 STEP 3 | | | | |
| 03 | | | | | | | |
| A. Switched and nonswitched lines. | | TCBSDITZ | | | | | |
| B. Switched lines only. | | | | | | | |
| C. Nonswitched lines only. | | | | | | | |
| 04 Switched | | TCBSDAC | | | | | |
| Nonswitched | | TCBSNDAC | | | | | |

BSC Initialize Terminal Scan

Diagram - 1.5.4-01


PAGE 1 OF 1
Input
Processing
Output
1.5.7
TCTLE
TCTLEIOA Line I/O area address
TCTLEIOL
TIOA
Data
DFHTCP:
01 Move storage from line entry to terminal entry chain
02 Calculate input data length and put length in TIOA
03 Execute user input exit
User input exit (optional)
04 Translate the data from line code to EBCDIC
1.5.6
TCTTE
TCTTEDA Terminal I/O area address
TIOA
TIOATDL Data length
Data

Input Event Completion

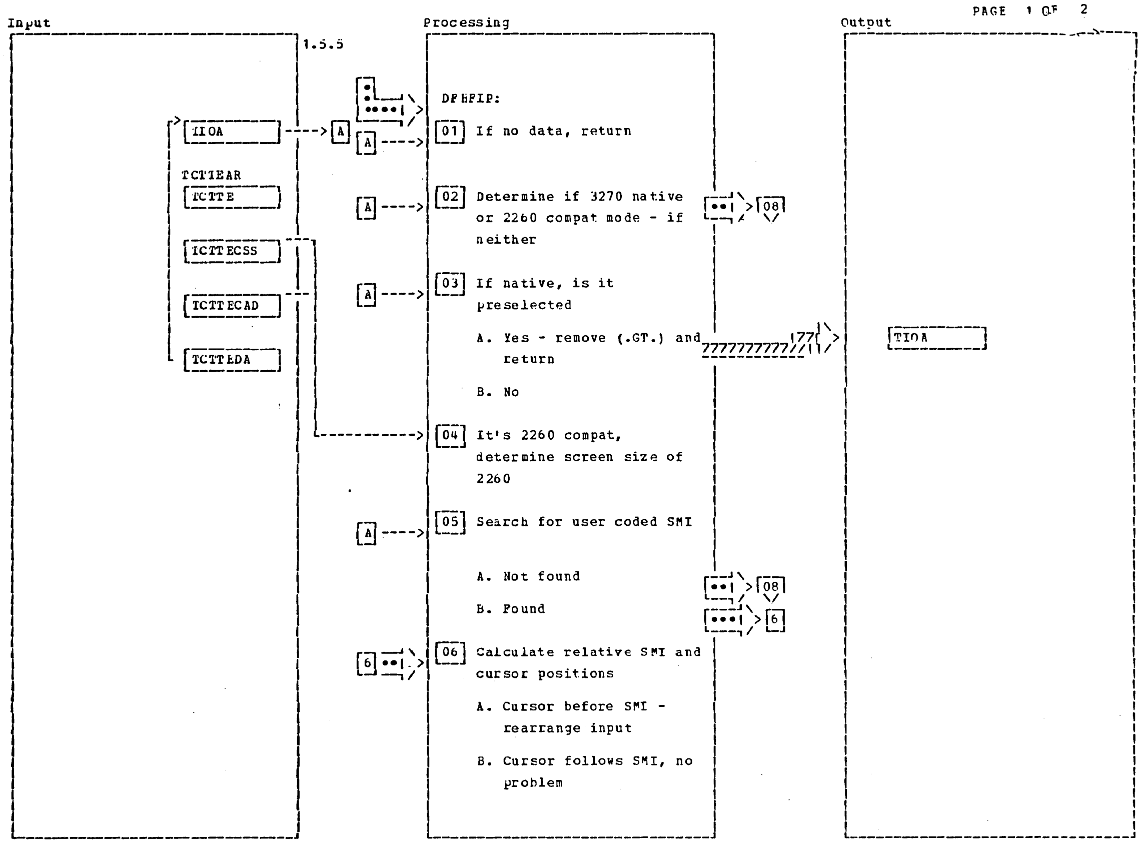
Diagram - 1.5.5-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 This is done in the device-dependent modules. | | | | | | | |
| 02 Scan data for ending character. If the end of the LIOA is reached before finding the data, set up a message to log error. | | TCIDLCR | | | | | |
| 03 A user exit may be included at TCP generation. | | | | | | | |
| 04 The translation of input data is bypassed for a device on a BSC line using EBCDIC transmission code. | | TCIOTRAN | | | | | |

Input Event Completion

Diagram - 1.5.5-C1

PAGE 1 OF 2



DFHFIP:

01 If no data, return

02 Determine if 3270 native or 2260 compat mode - if neither

03 If native, is it preselected

A. Yes - remove (.GT.) and return

B. No

04 It's 2260 compat, determine screen size of 2260

05 Search for user coded SMI

A. Not found

B. Found

06 Calculate relative SMI and cursor positions

A. Cursor before SMI - rearrange input

B. Cursor follows SMI, no problem

FASTER Input Analysis

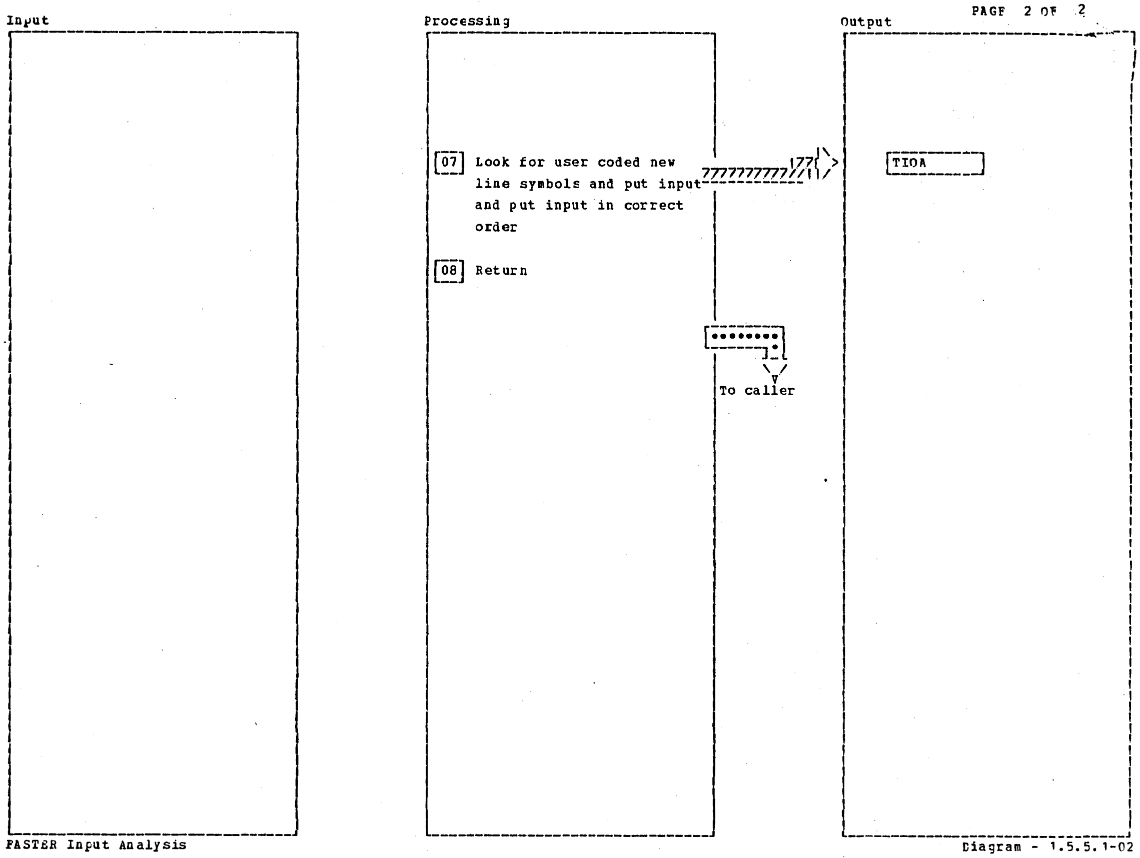
Diagram - 1.5.5.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Determine if input data exists in stream ( not PA key, PF key, or ENTER key with no data). If no data, return. | | DFHIPNA | |
| 02 If terminal is not in either 2260 compat mode or 3270 native (fields in input). If neither, return. | | ITSAMSG | |
| 03 If native mode, remove preselected character (.GT.) if one exists, and return. | | NATV | |
| 04 Determine 2260 screen size being simulated ( 240, 480,960). | | PGWSET | |
| 05 Search for user coded SMI in input stream, if none found, return. | | CLRTAB | |
| 06 When SMI is found, calculate its relative scree address and determine if it is before or after the cursor. | | SOMFND | |
| A. If cursor is before SMI, | | ROLLFLDS | |
| rearrange input data (SMI, DATA, CURSOR). | | | |

FASTER Input analysis

Diagram - 1.5.5.1-01

PAGE 2 OF 2



07 Look for user coded new line symbols and put input and put input in correct order

08 Return

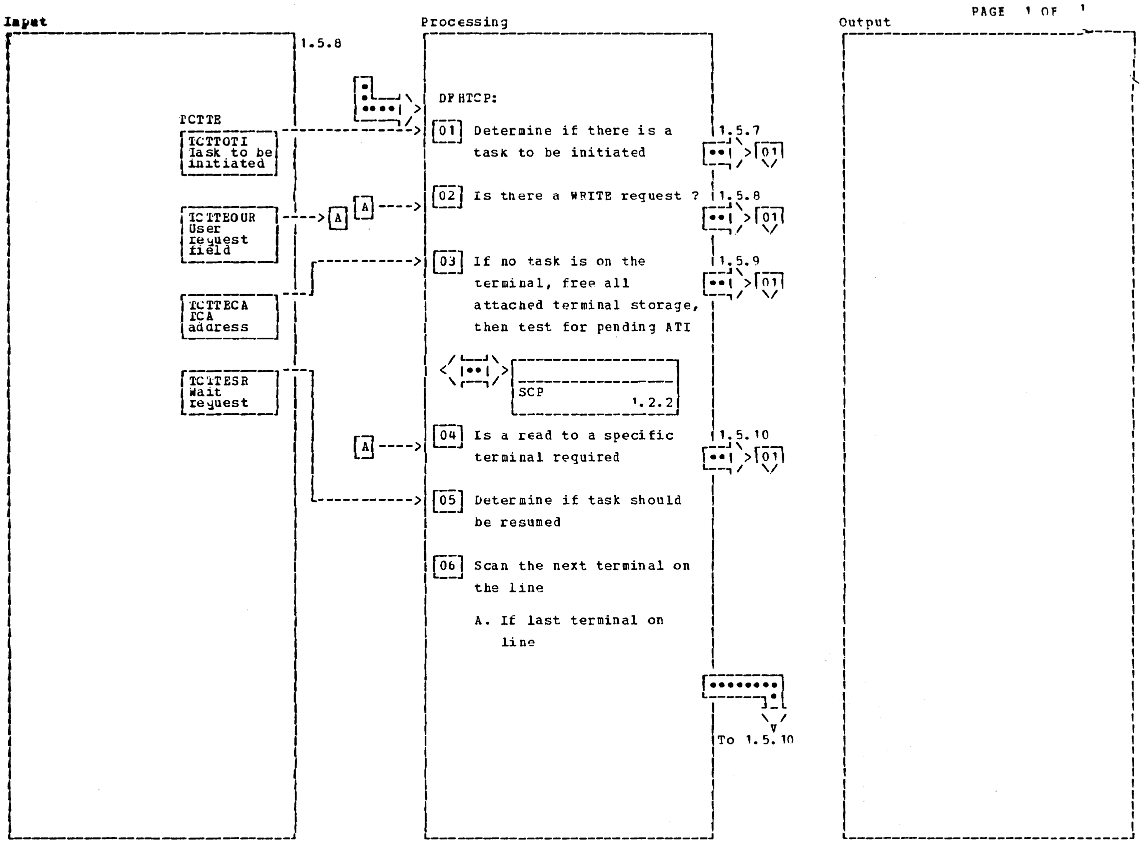
FASTER Input Analysis

Diagram - 1.5.5.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 07 Search for user coded new line symbols, and rearrange data stream to eliminate data positions between new line and the next line of input. | | SETNL<br>MOVLINE1<br>MOVELINE | | | | | |
| 08 Return to TCP. | | | | | | | |

FASTER Input Analysis

Diagram - 1.5.5.1-02



DFHTCP:

01 Determine if there is a task to be initiated

02 Is there a WRITE request ?

03 If no task is on the terminal, free all attached terminal storage, then test for pending ATI

04 Is a read to a specific terminal required

05 Determine if task should be resumed

06 Scan the next terminal on the line

A. If last terminal on line

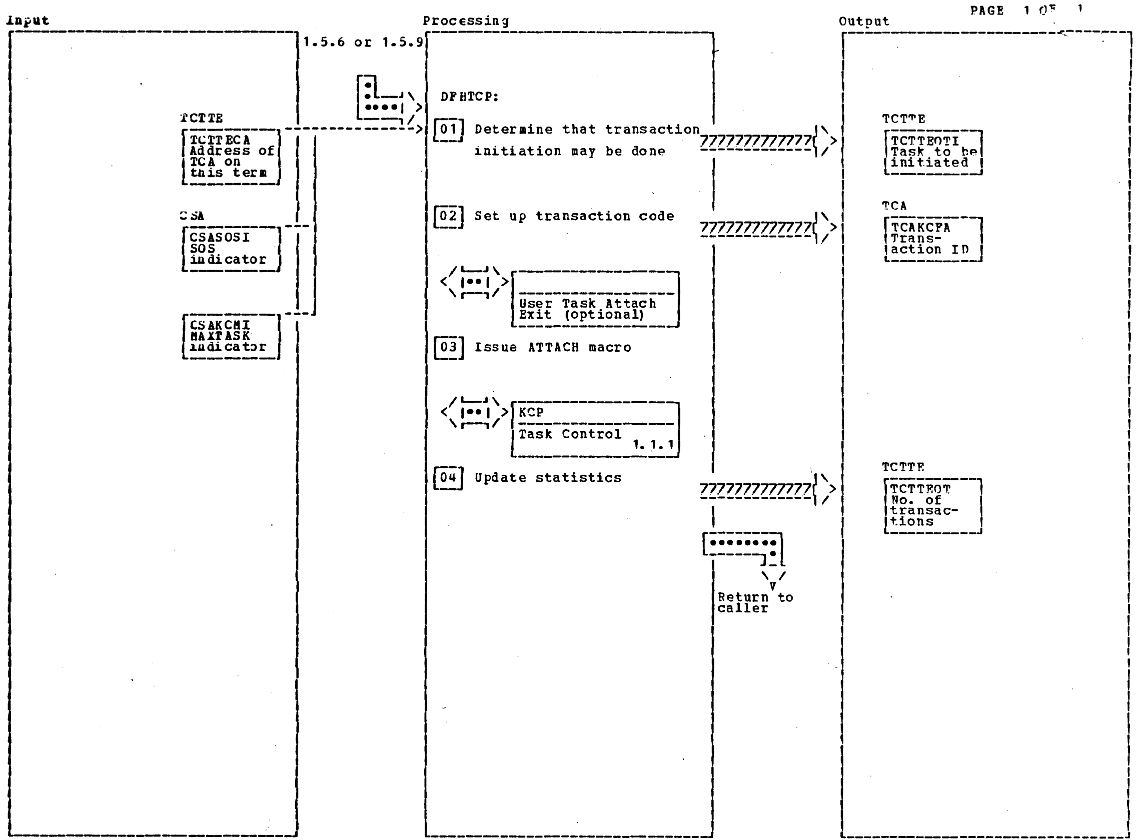
Activity Control

Diagram - 1.5.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 This routine scans each terminal on a line to determine what requests are pending.<br>If, on a previous scan, Terminal Control was not able to initiate a task because the system was at MAXTASK or SOS, the task to be initiated indicator will have been left on.<br>Detection of a task initiation will cause a return to the caller. | DFHTCCS<br>DFHTCCBS | TCSSNAS<br>TCBSAS | | | | | |
| 02 This will cause an entry to output event preparation. | DFHTCCSS<br>DFHTCCBS | TCSSNITB | | | | | |
| 03 This routine will clean up any storage chained to terminal entry after a task has been terminated. | DFHTCCSS<br>DFHTCCBS | | | | | | |
| 04 The detection of a READ request will cause a return to the caller + 8. The Skip Flag logic routine will be entered. | DFHTCCSS<br>DFHTCCBS | | | | | | |

Activity Control

Diagram - 1.5.6-01


PAGE 1 OF 1
Input
Processing
Output
1.5.6 or 1.5.9
DFHTCP:
TCTTE
TCTTECA Address of TCA on this term
CSA
CSASOSI SOS indicator
CSAKCMI MAXTASK indicator
01 Determine that transaction initiation may be done
02 Set up transaction code
User Task Attach Exit (optional)
03 Issue ATTACH macro
KCP
Task Control 1.1.1
04 Update statistics
Return to caller
TCTTE
TCTTEOTI Task to be initiated
TCA
TCAKCPA Trans-action ID
TCTTE
TCTTEOT No. of transac-tions

Transaction Initiation

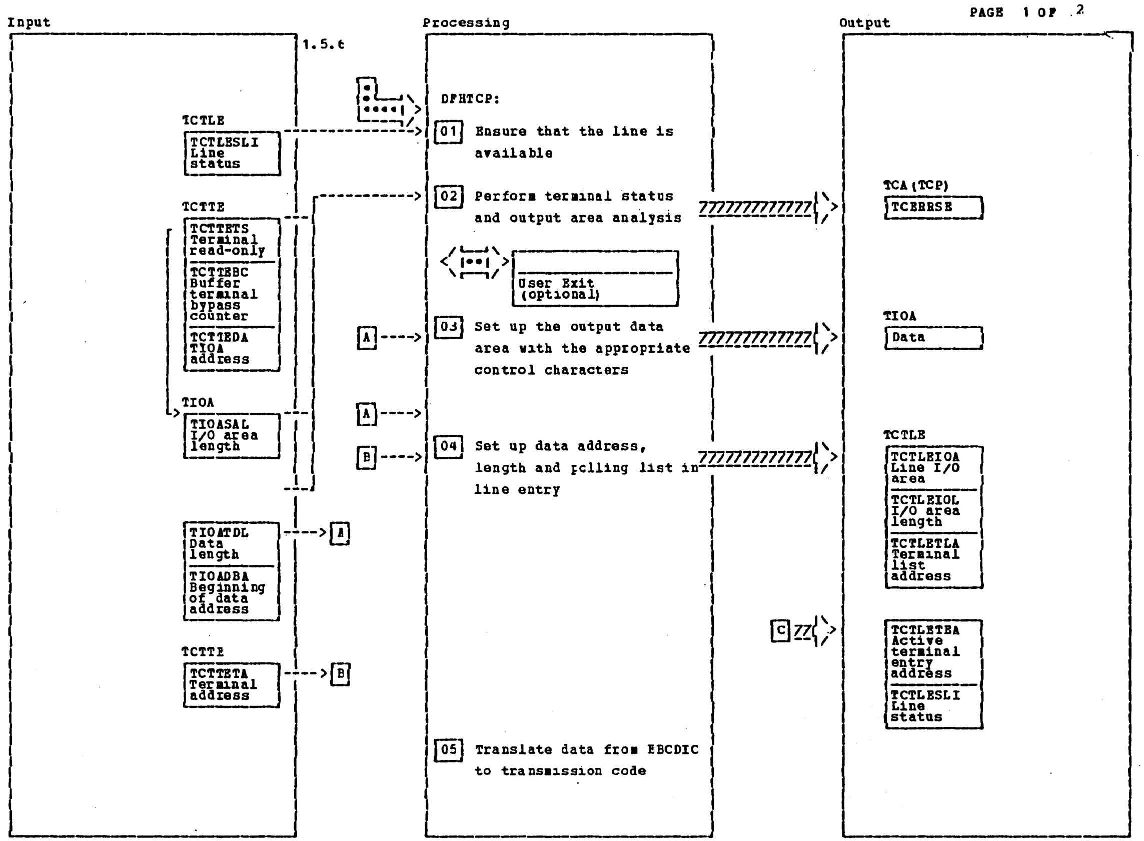
Diagram - 1.5.7-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If a task is already on the terminal, return to the caller before setting the task to be initiated bit. If either SOS or MAXTASK indicators are on, return to the caller, leaving the task to be initiated bit set. This will cause activity control to BAL to this routine the next time this line is scanned. | | | | | | | |
| 02 The transaction ID may be obtained from the input data, the TCTTETC. If paging is active, cause CSPG to be invoked or be generated as a result of the depression of a PA or PF key on a 3270. | | DFHTCTII | | | | | |
| 03 If the transaction ID passed to Task Control does not have a corresponding entry in the PCT, an abnormal return will be passed to TCP. TCP will set up an error code for ACP. | | TCPCTLK | | | | | |

Transaction Initiation

Diagram - 1.5.7-C1

PAGE 1 OF 2


Input
Processing
Output
1.5.t
TCTLE
TCTLESLI Line status
TCTTE
TCTTETS Terminal read-only
TCTTEBC Buffer terminal bypass counter
TCTTEDA TIOA address
TIOA
TIOASAL I/O area length
TIOATDL Data length
TIOADBA Beginning of data address
TCTTE
TCTTETA Terminal address
A
B
C
DFHTCP:
01 Ensure that the line is available
02 Perform terminal status and output area analysis
User Exit (optional)
03 Set up the output data area with the appropriate control characters
04 Set up data address, length and pclling list in line entry
05 Translate data from EBCDIC to transmission code
TCA (TCP)
TCERRSE
TIOA
Data
TCTLE
TCTLEIOA Line I/O area
TCTLEIOL I/O area length
TCTLETLA Terminal list address
TCTLETEA Active terminal entry address
TCTLESLI Line status

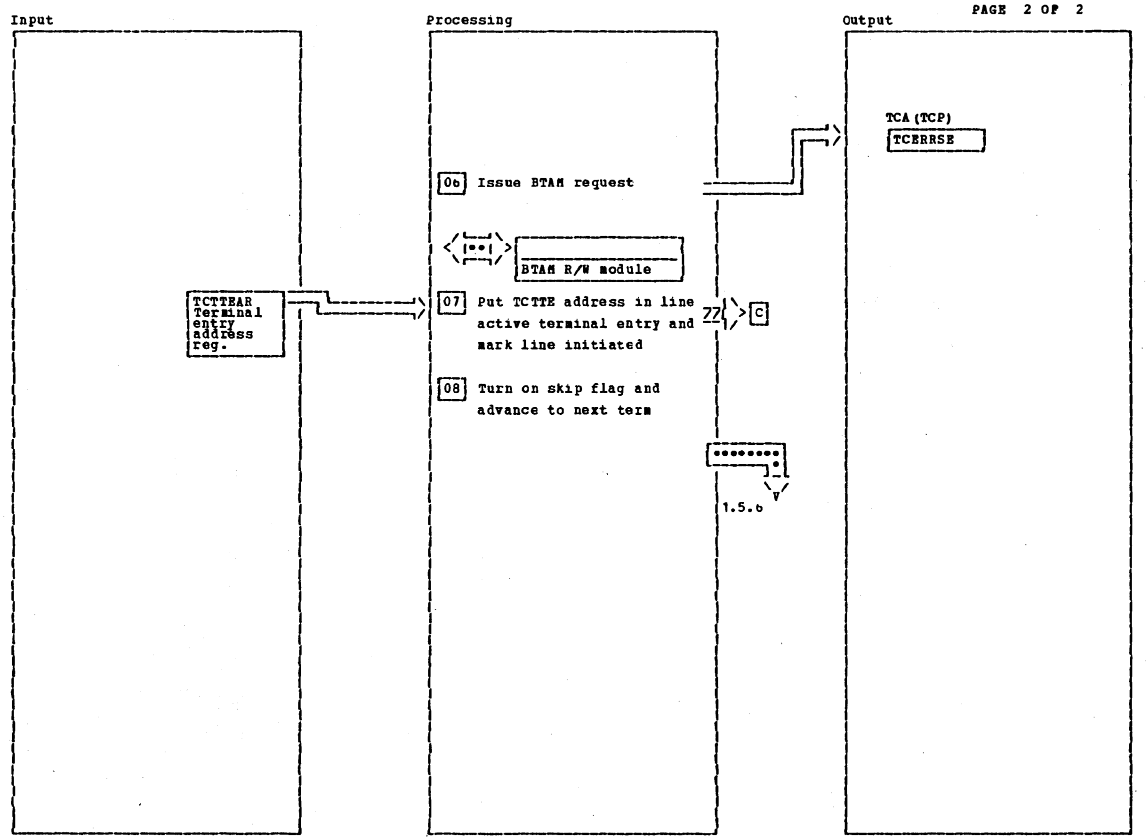
Output Event Initiation

Diagram - 1.5.8-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Perform a RESET POLL, then test that the line is nct initiated. If it is initiated, set the SKIPO flag on. | | | |
| 02 Three error conditions are tested in this routine.<br>A. The applications is attempting to write to a read-only device.<br>B. The user has failed to place the output area address in TCTTEDA.<br>C. The user has set up a data stream (TIOATDL) longer than the area (TIOASAI) he acquired.<br>D. If error found, set error indicator and attach TACP.<br>The user exit will be executed after the device status test but before the analysis of the output area. | | TCOAA | |
| 03 This will be performed by the Device Dependent Module (DDM). | | | |
| 04 This is done in the DDM. The data length will be adjusted for the line control charaters added in step 3. | | | |
| 05 A BSC line with EBCDIC transmission code will bypass this step. | | TCIOTRAN | |

Output Event Initiation

Diagram - 1.5.8-01


PAGE 2 OF 2
Input
Processing
Output
TCA (TCP)
TCERRSE
06 Issue BTAM request
BTAM R/W module
TCTTEAR Terminal entry address reg.
07 Put TCTTE address in line active terminal entry and mark line initiated
77
C
08 Turn on skip flag and advance to next term
1.5.6

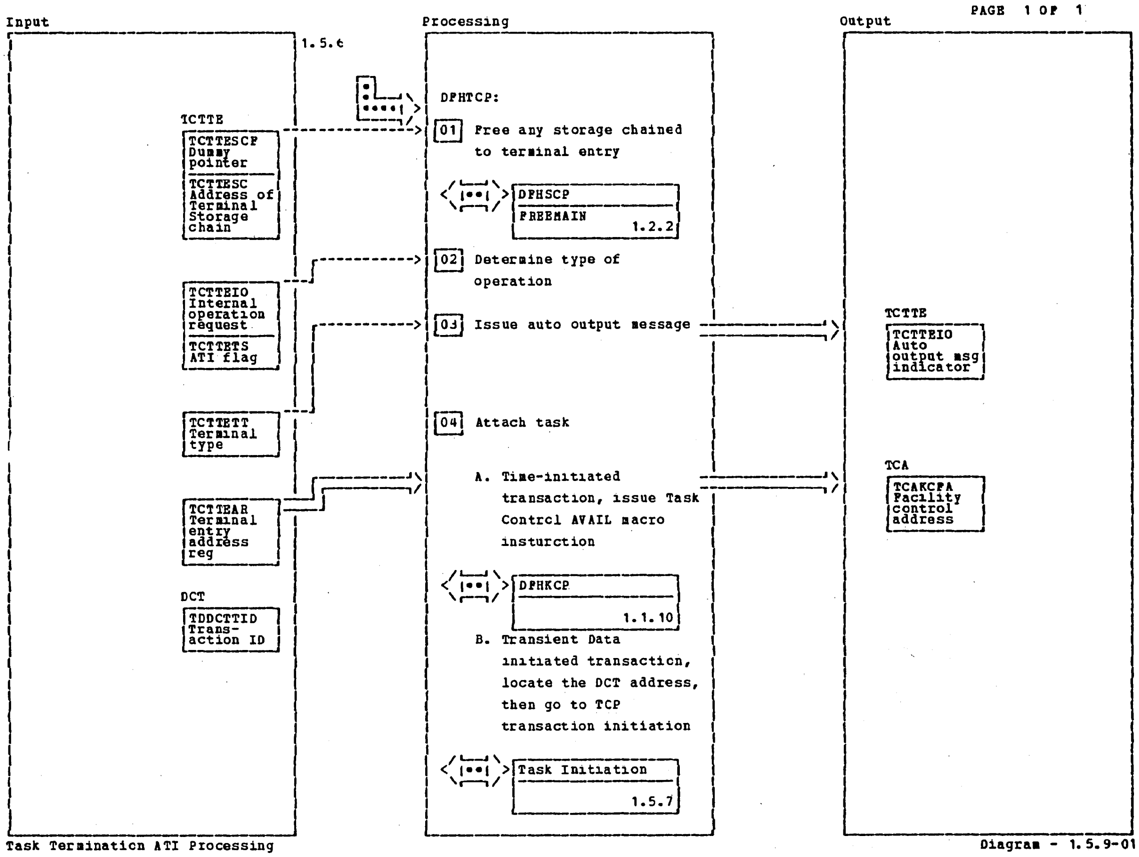
Output Event Initiation

Diagram - 1.5.8-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 06 Upon return from the access method, test the return code in register 15 to ensure the request was accepted. If the return code is nonzero, set up output event rejected and attach TACP. | | | | | | | |

Output Event Initiation

Diagram - 1.5.8-02

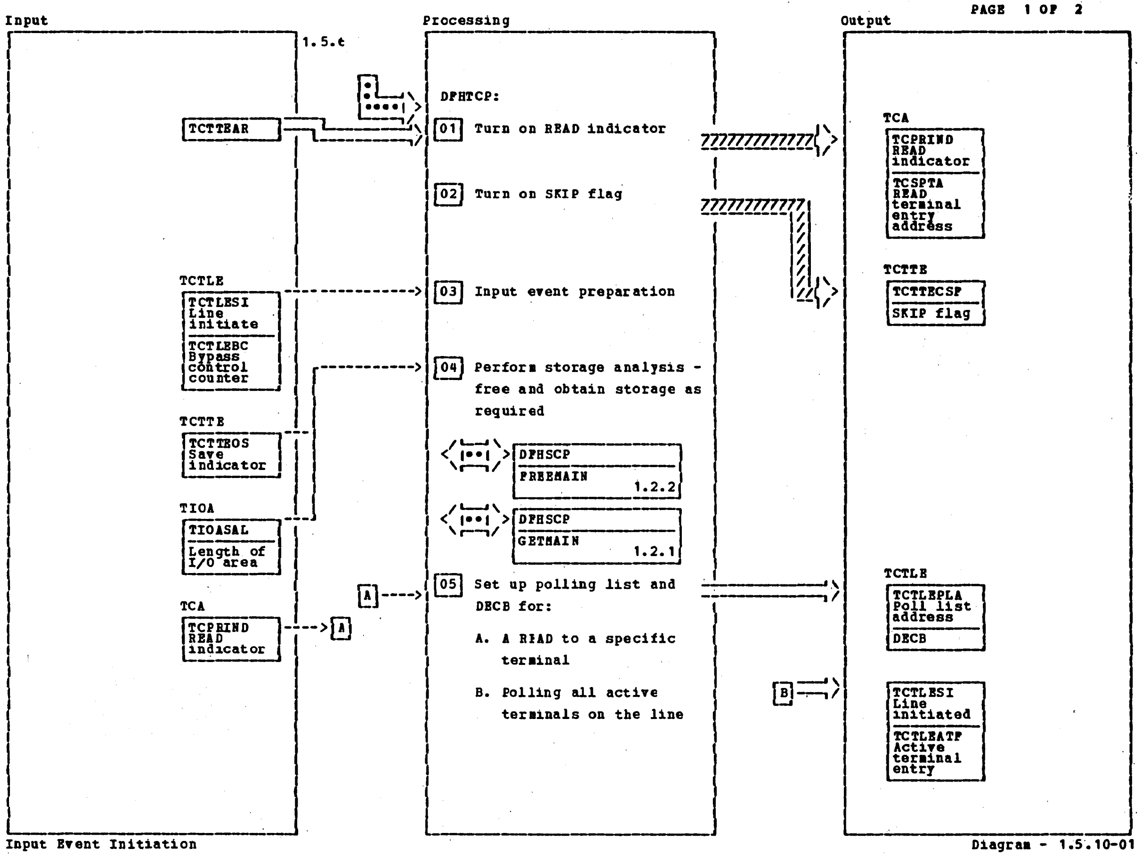

PAGE 1 OF 1
Input
Processing
Output
1.5.6
TCTTE
TCTTESCF Dummy pointer
TCTTESC Address of Terminal Storage chain
TCTTEIO Internal operation request
TCTTETS ATI flag
TCTTETT Terminal type
TCTTEAR Terminal entry address reg
DCT
TDDCTTID Trans-action ID
DFHTCP:
01 Free any storage chained to terminal entry
DFHSCP
FREEMAIN
1.2.2
02 Determine type of operation
03 Issue auto output message
04 Attach task
A. Time-initiated transaction, issue Task Contrcl AVAIL macro insturction
DFHKCP
1.1.10
B. Transient Data initiated transaction, locate the DCT address, then go to TCP transaction initiation
Task Initiation
1.5.7
TCTTE
TCTTEIO Auto output msg indicator
TCA
TCAKCFA Facility control address

Task Terminaticn ATI Processing

Diagram - 1.5.9-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Entry to this routine is from the Activity Scan routine.<br>If the contents of TCTTESC point to TCTTESCE, there is no storage on this terminal, bypass the FREEMAIN and go to ATI analysis.<br>If they are not equal, perform the FREEMAIN of all attached storage. | DFHTCORS | TCTDTI | | | | | |
| 02 The setting of the Internal Request field and the ATI flag will allow Terminal Control to determine if there is a task to be initiated and the source of the request: time initiation or Transient Data. | | TCCFATSB | | | | | |

PAGE 1 OF 2


Input
Processing
Output
1.5.6
TCTTEAR
DFHTCP:
01 Turn on READ indicator
02 Turn on SKIP flag
TCTLE
TCTLESI Line initiate
TCTLEBC Bypass control counter
03 Input event preparation
04 Perform storage analysis - free and obtain storage as required
TCTTE
TCTTEOS Save indicator
DFHSCP
FREEMAIN
1.2.2
DFHSCP
GETMAIN
1.2.1
TIOA
TIOASAL
Length of I/O area
A
05 Set up polling list and DECB for:
A. A READ to a specific terminal
B. Polling all active terminals on the line
TCA
TCPRIND READ indicator
A
TCA
TCPRIND READ indicator
TCSPTA READ terminal entry address
TCTTE
TCTTECSF
SKIP flag
TCTLE
TCTLEPLA Poll list address
DECB
B
TCTLESI Line initiated
TCTLEATF Active terminal entry

Input Event Initiation

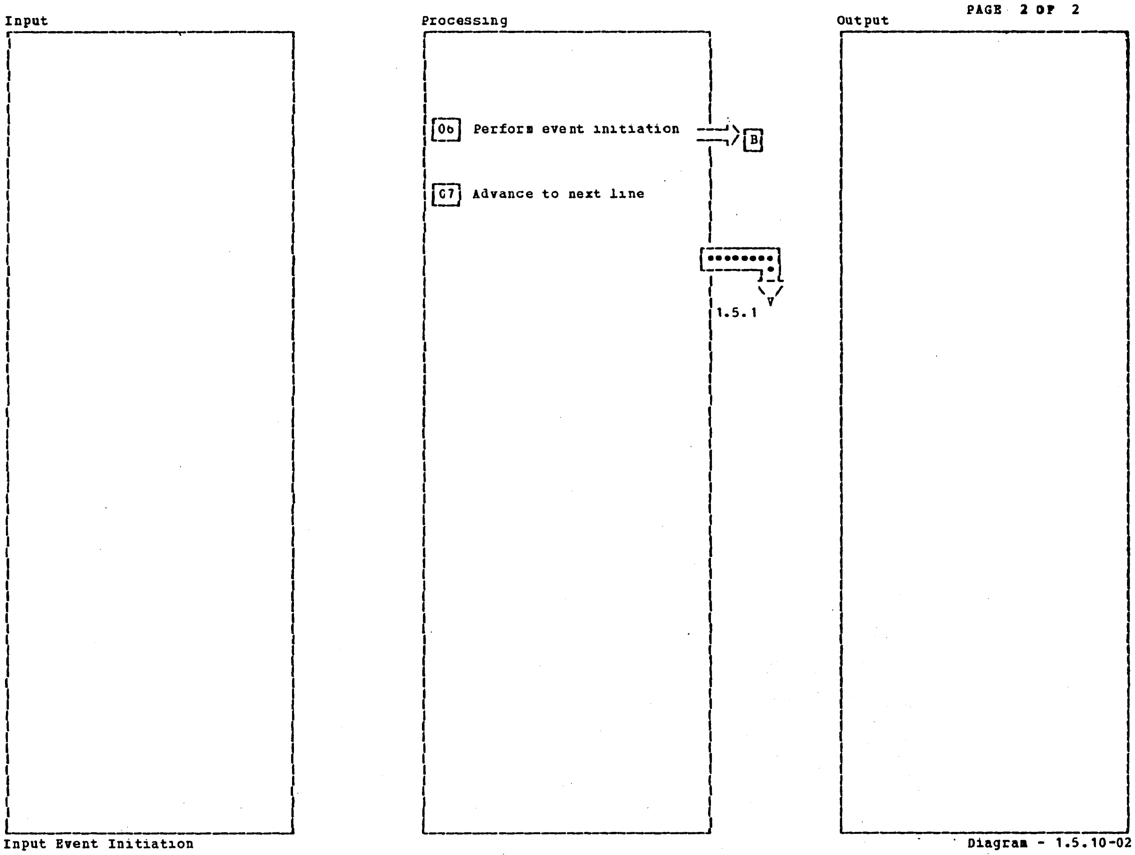
Diagram - 1.5.10-01

| Notes | Rottine | Label | Ref |
|---|---|---|---|
| 01 This function is performed in the DDM. For a dial-up or point to point terminal, go directly to step 3. | | | |
| For a terminal on a multipoint line, scanning of all terminals on the line continues until Terminal Scan Termination detects the last terminal on the line. | | | |
| At this time, control will be passed to Input Event Preparation. | | | |
| 02 Performed in individual DDMs. | | | |
| 03 Input Event Preparation will ensure that the line is not already activated. | | | |
| If activated, I/O will be delayed until processing on same line. | | | |
| 04 Test will be made to determine that the user has provided storage, that it is as large or larger than the minimum line area size, and that the user has not specified 'save' on his buffer. | | | |
| If any of the above conditions exist, a GETMAIN will be done to acquire a new area to read into. In addition if the system is SOS, all line buffers will be freed and processing will advance to next line. | | | |
| 05 The source of the information for the polling list depends on the type of list the user defined at CICS/VS generation. | | | |

Input Event Initiation

Diagram - 1.5.10-01

PAGE 2 OF 2



Input Event Initiation

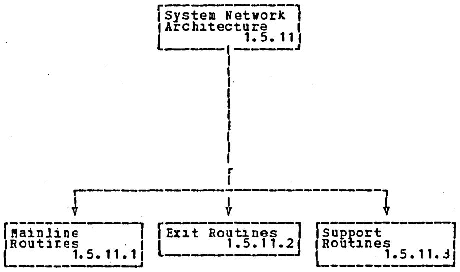
Diagram - 1.5.10-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 06 Issue BTAM macro instruction. Ensure that it was accepted. | | | | | | | |

Input Event Initiation

Diagram - 1.5.10-02

PAGE 1 OF 1


System Network Architecture 1.5.11
Mainline Routires 1.5.11.1
Exit Routines 1.5.11.2
Support Routines 1.5.11.3

PAGE 1 OF 2

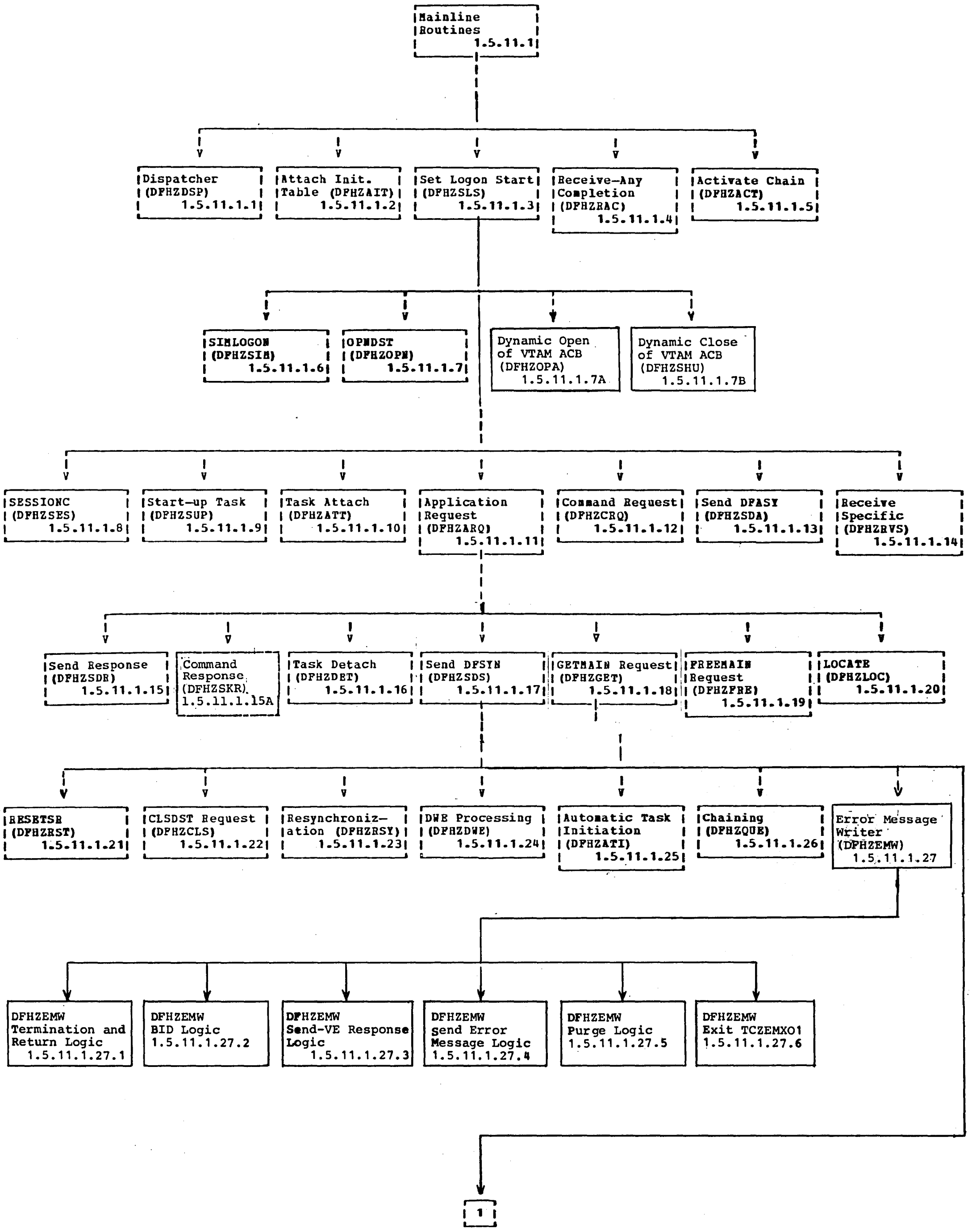
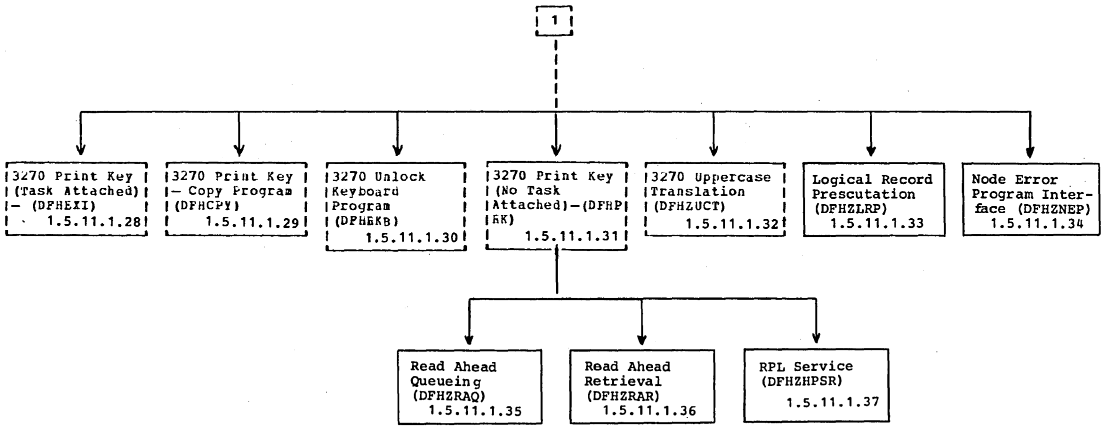
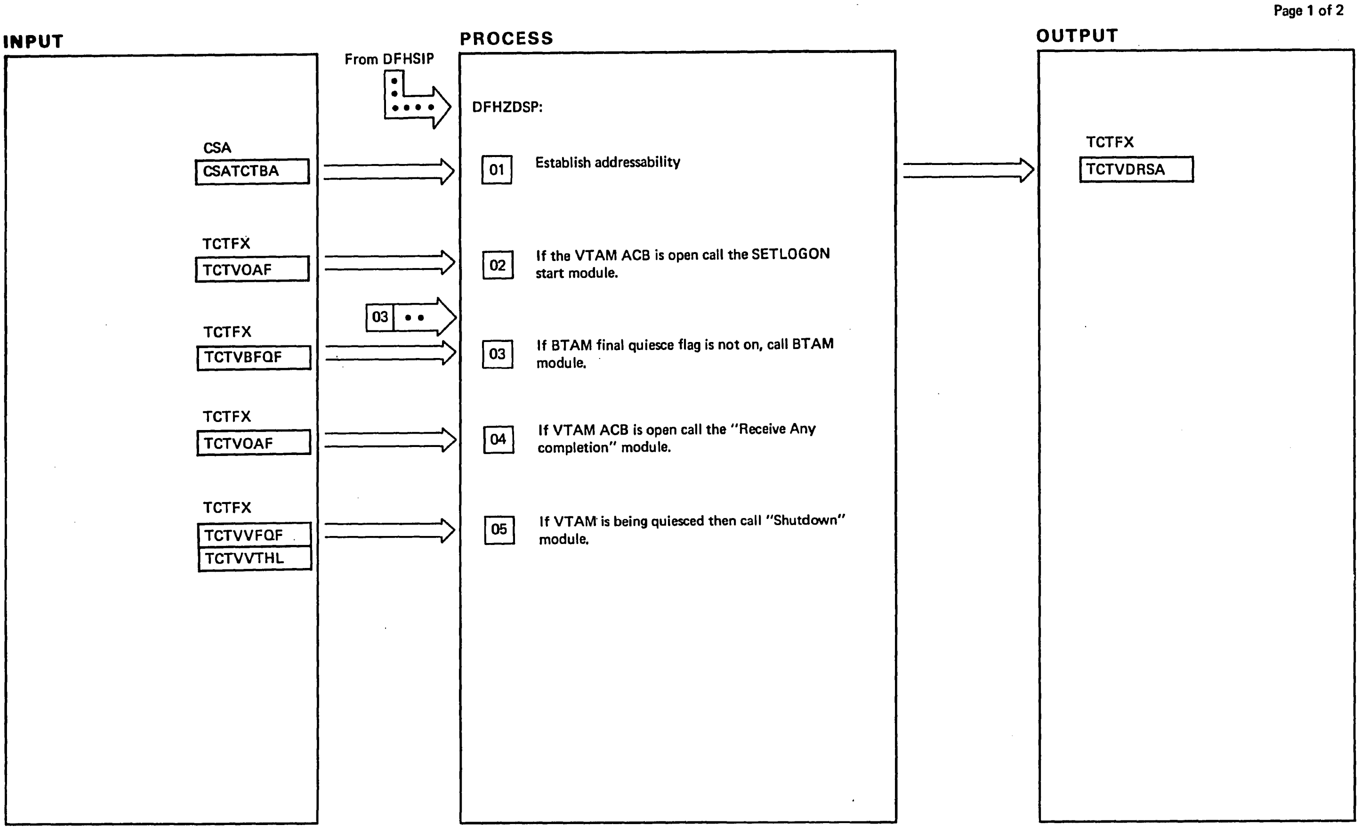

Mainline Routines 1.5.11.1
Dispatcher (DFHZDSP) 1.5.11.1.1
Attach Init. Table (DFHZAIT) 1.5.11.1.2
Set Logon Start (DFHZSLS) 1.5.11.1.3
Receive-Any Completion (DFHZRAC) 1.5.11.1.4
Activate Chain (DFHZACT) 1.5.11.1.5
SIMLOGON (DFHZSIM) 1.5.11.1.6
OPNDST (DFHZOPN) 1.5.11.1.7
Dynamic Open of VTAM ACB (DFHZOPA) 1.5.11.1.7A
Dynamic Close of VTAM ACB (DFHZSHU) 1.5.11.1.7B
SESSIONC (DFHZSES) 1.5.11.1.8
Start-up Task (DFHZSUP) 1.5.11.1.9
Task Attach (DFHZATT) 1.5.11.1.10
Application Request (DFHZARQ) 1.5.11.1.11
Command Request (DFHZCRQ) 1.5.11.1.12
Send DFASY (DFHZSDA) 1.5.11.1.13
Receive Specific (DFHZRVS) 1.5.11.1.14
Send Response (DFHZSDR) 1.5.11.1.15
Command Response (DFHZSKR) 1.5.11.1.15A
Task Detach (DFHZDET) 1.5.11.1.16
Send DFSYN (DFHZSDS) 1.5.11.1.17
GETMAIN Request (DFHZGET) 1.5.11.1.18
FREEMAIN Request (DFHZFRE) 1.5.11.1.19
LOCATE (DFHZLOC) 1.5.11.1.20
RESETSR (DFHZRST) 1.5.11.1.21
CLSDST Request (DFHZCLS) 1.5.11.1.22
Resynchronization (DFHZRSY) 1.5.11.1.23
DWE Processing (DFHZDWE) 1.5.11.1.24
Automatic Task Initiation (DFHZATI) 1.5.11.1.25
Chaining (DFHZQUE) 1.5.11.1.26
Error Message Writer (DFHZEMW) 1.5.11.1.27
DFHZEMW Termination and Return Logic 1.5.11.1.27.1
DFHZEMW BID Logic 1.5.11.1.27.2
DFHZEMW Send-VE Response Logic 1.5.11.1.27.3
DFHZEMW Send Error Message Logic 1.5.11.1.27.4
DFHZEMW Purge Logic 1.5.11.1.27.5
DFHZEMW Exit TCZEMXO1 1.5.11.1.27.6
1

CHART: DTCPSNA2
PAGE 2 OF 2


1
3270 Print Key (Task Attached) — (DFHEXI) 1.5.11.1.28
3270 Print Key — Copy Program (DFHCPY) 1.5.11.1.29
3270 Unlock Keyboard Program (DFHEKB) 1.5.11.1.30
3270 Print Key (No Task Attached)—(DFHPRK) 1.5.11.1.31
3270 Uppercase Translation (DFHZUCT) 1.5.11.1.32
Logical Record Prescutation (DFHZLRP) 1.5.11.1.33
Node Error Program Interface (DFHZNEP) 1.5.11.1.34
Read Ahead Queueing (DFHZRAQ) 1.5.11.1.35
Read Ahead Retrieval (DFHZRAR) 1.5.11.1.36
RPL Service (DFHZHPSR) 1.5.11.1.37



DFHZDSP:

01 Establish addressability

02 If the VTAM ACB is open call the SETLOGON start module.

03 If BTAM final quiesce flag is not on, call BTAM module.

04 If VTAM ACB is open call the "Receive Any completion" module.

05 If VTAM is being quiesced then call "Shutdown" module.

Dispatcher (DFHZDSP)

Diagram 1.5.11.1.1-01

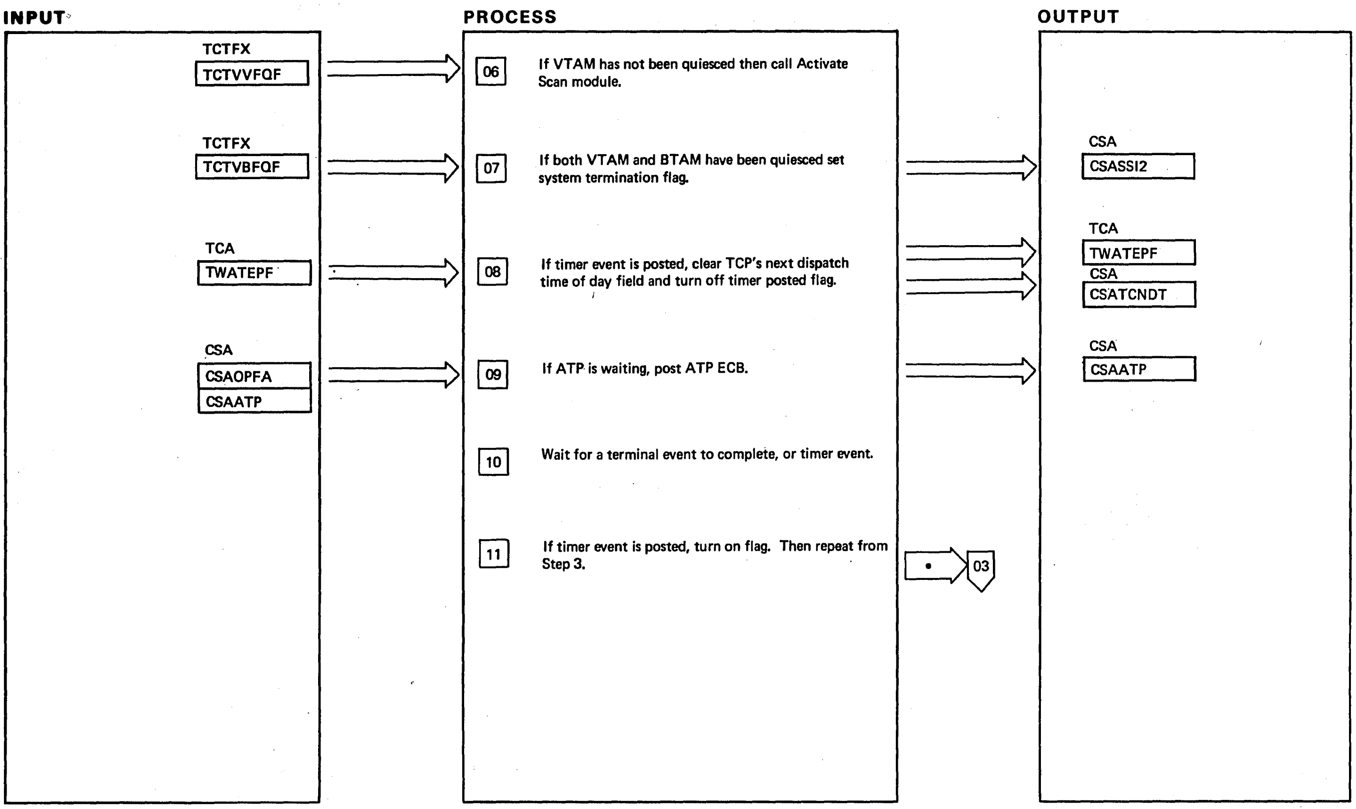
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 CICS may be generated with or without VTAM. SETLOGON start module. | DFHZSLS | | |
| 03 If BTAM is included, it turns on BTAM final quiesce flag after executing its quiesce logic; which is invoked by CICS terminating, without abending transactions. | | | |
| 04 Receive Any Completion module. | DFHZRAC | | |
| 05 VTAM may be being quiesced because<br>a) CICS is being terminated.<br>b) VTAM network is being closed.<br>c) CICS Master Terminal operator wants to uncouple CICS and VTAM. | DFHZSHU | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Dispatcher (DFHZDSP)

Diagram 1.5.11.1.1-01

Page 2 of 2



06 If VTAM has not been quiesced then call Activate Scan module.

07 If both VTAM and BTAM have been quiesced set system termination flag.

08 If timer event is posted, clear TCP's next dispatch time of day field and turn off timer posted flag.

09 If ATP is waiting, post ATP ECB.

10 Wait for a terminal event to complete, or timer event.

11 If timer event is posted, turn on flag. Then repeat from Step 3.

Dispatcher (DFHZDSP)

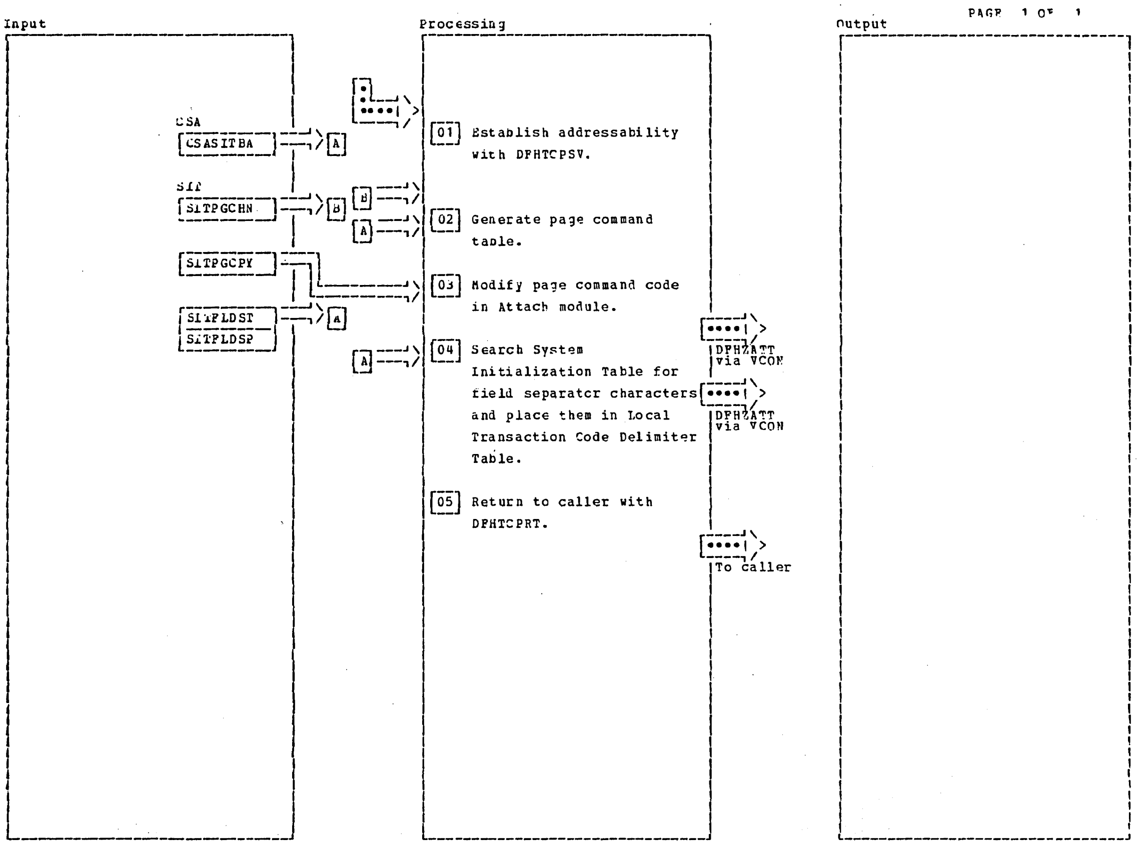
Diagram 1.5.11.1.1-02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 06 Activate scan module deals with outstanding VTAM communication requests for a node. | DFHZACT | | |
| 07 BTAM will only be quiesced when CICS is terminating via DFHSTP module, which issues a KC WAIT on ECB CSASSI2. When this is posted all terminal activity is finished and CICS can terminate. | | | |
| 10 DFHKC TYPE=WAIT,<br>DCI=TCP | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Dispatcher (DFHZDSP)

Diagram 1.5.11.1.1-02



01 Establish addressability with DFHTCPSV.

02 Generate page command table.

03 Modify page command code in Attach module.

04 Search System Initialization Table for field separator characters and place them in Local Transaction Code Delimiter Table.

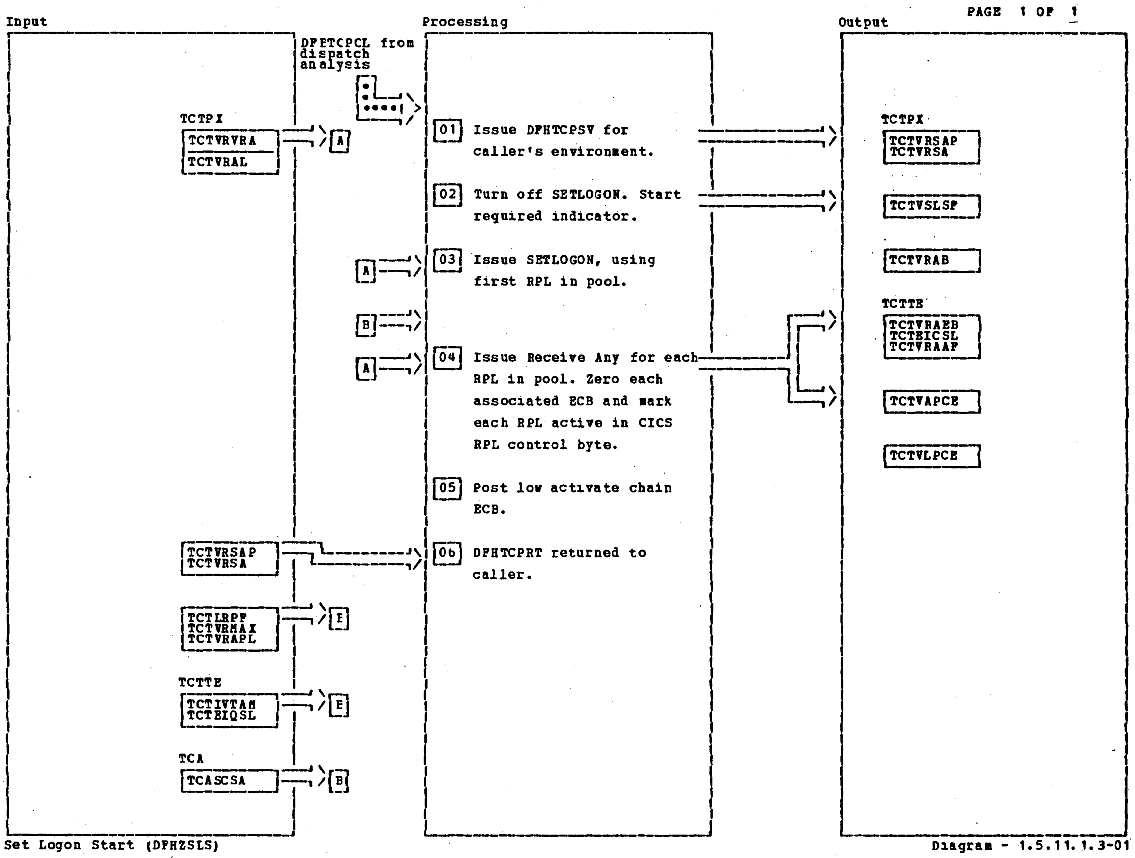
05 Return to caller with DFHTCPRT.

Attach Initialization Tables (DFHZAIT) Diagram - 1.5.11.1.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|

| Notes | Routine | Label | Ref |
|---|---|---|---|

Attach Initialization Tables (DFHZAIT) Diagram - 1.5.11.1.2-01



01 Issue DFHTCPSV for caller's environment.

02 Turn off SETLOGON. Start required indicator.

03 Issue SETLOGON, using first RPL in pool.

04 Issue Receive Any for each RPL in pool. Zero each associated ECB and mark each RPL active in CICS RPL control byte.

05 Post low activate chain ECB.

06 DFHTCPRT returned to caller.

Set Logon Start (DFHZSLS)

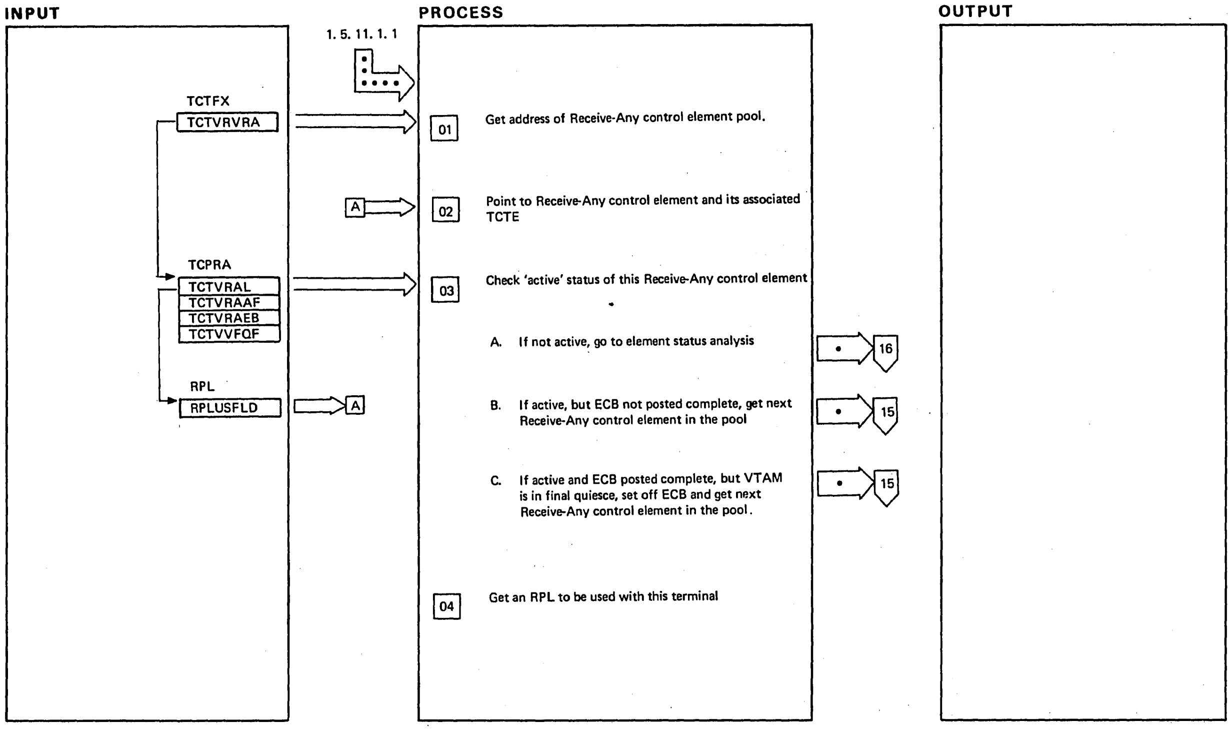
Diagram - 1.5.11.1.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 First time, control is given to this routine by Dispatch Entry Analysis when the TCTVSLSF flag is set on. Thereafter, Receive Any is issued as a completion is processed. | | | | | | | |
| 04 Allocate an input area of size given in TCTVRMAX for each Receive Any issued. | | | | | | | |
| 05 The ECB is posted to cause an initial scan for work it Any as been placed on the process chain by emergency restart recovery routines. | | | | | | | |

Set Logon Start (DFHZSLS)

Diagram - 1.5.11.1.3-01

Page 1 of 4



01 Get address of Receive-Any control element pool.

02 Point to Receive-Any control element and its associated TCTE

03 Check 'active' status of this Receive-Any control element

A. If not active, go to element status analysis

B. If active, but ECB not posted complete, get next Receive-Any control element in the pool

C. If active and ECB posted complete, but VTAM is in final quiesce, set off ECB and get next Receive-Any control element in the pool.

04 Get an RPL to be used with this terminal

Receive-Any (DFHRAC)

Diagram - 1.5.11.1.4-01

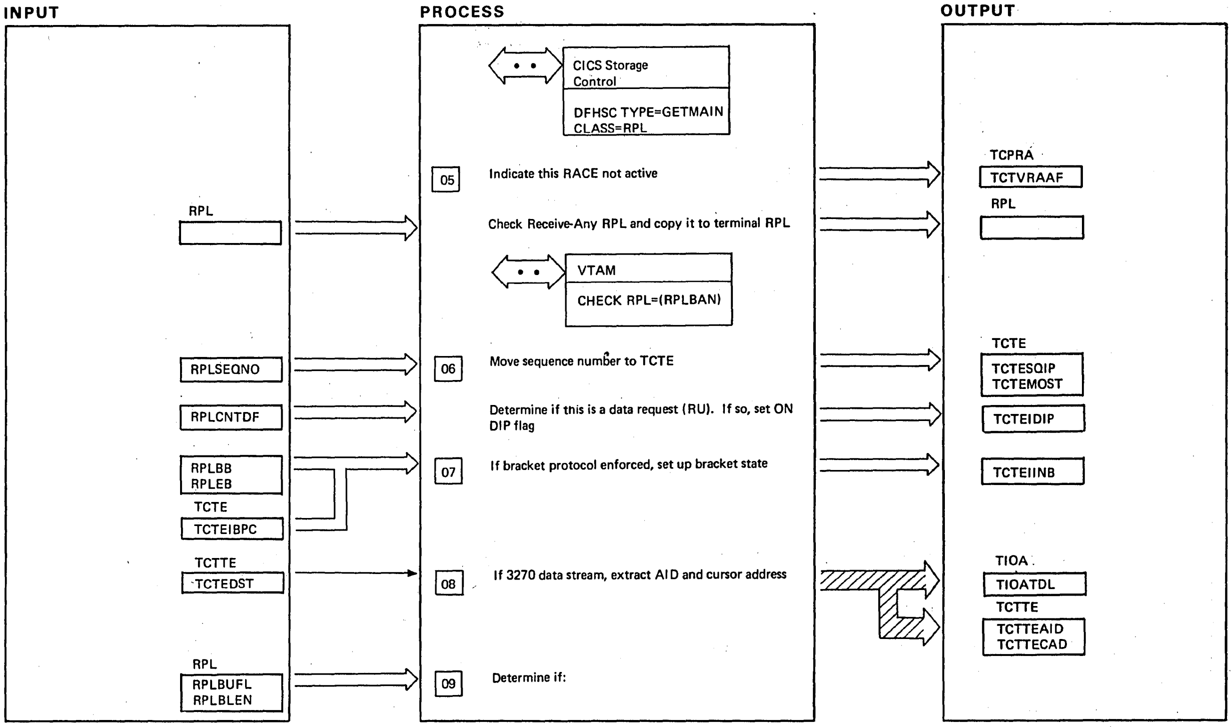
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Registers are saved and RPLBAR and NIOABAR are set up | | DFHZRAC | |
| 03 Getmain is conditional - if it fails, get the next Receive-Any control element | | TCZRACA2 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Receive-Any (DFHRAC)

Diagram - 1.5.11.1.4-01

Page 2 of 4



05 Indicate this RACE not active

Check Receive-Any RPL and copy it to terminal RPL

06 Move sequence number to TCTE

Determine if this is a data request (RU). If so, set ON DIP flag

07 If bracket protocol enforced, set up bracket state

08 If 3270 data stream, extract AID and cursor address

09 Determine if:

Receive-Any (DFHRAC)

Diagram - 1.5.11.1.4-02

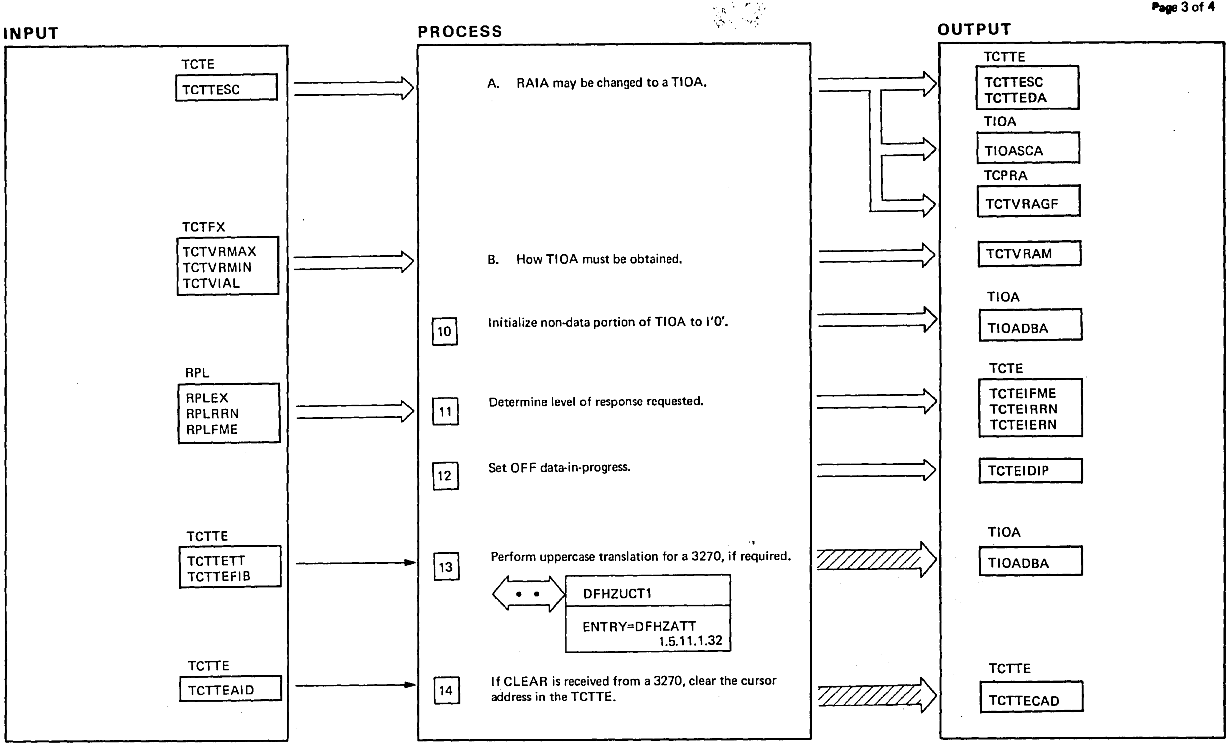
| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 05 | The check is contained in the DFHZHPCH macro, which expands into a VTAM check unless authorized path SRB support is being used; in which case the expansion also checks for an 'SRB Mode Run' RPL and calls LERAD/SYNAD routine if necessary.<br>If an error is detected on the check, a test is made to determine if the user field of the RPL points to a valid TCTE. If not, TCTTEAR is loaded with the address of a dummy TCTE. The terminal is then chained to NACP for error processing | | | |
| 06 | If a data flow synchronous command is received, see Receive-Any command processing | | TCZRACA9 | |
| 07 | If bracket protocol is enforced and the RPL indicates no begin bracket and no end bracket, a bracket RACE error is returned to the LU. If an end bracket but no begin bracket is indicated, the error is passed to NACP | | | |
| 09 | If a new TIOA must be acquired, a storage control GETMAIN is performed. If it fails, TCTVRAM flag will be left on. If successful, TCTVRAM is turned off and the data moved to the new TIOA | | TC2RACD1 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Receive-Any (DFHRAC)

Diagram - 1.5.11.1.4-02

Page 3 of 4


INPUT
PROCESS
OUTPUT
TCTE
TCTTESC
TCTFX
TCTVRMAX
TCTVRMIN
TCTVIAL
RPL
RPLEX
RPLRRN
RPLFME
TCTTE
TCTTETT
TCTTEFIB
TCTTE
TCTTEAID
A. RAIA may be changed to a TIOA.
B. How TIOA must be obtained.
10 Initialize non-data portion of TIOA to I'0'.
11 Determine level of response requested.
12 Set OFF data-in-progress.
13 Perform uppercase translation for a 3270, if required.
DFHZUCT1
ENTRY=DFHZATT
1.5.11.1.32
14 If CLEAR is received from a 3270, clear the cursor address in the TCTTE.
TCTTE
TCTTESC
TCTTEDA
TIOA
TIOASCA
TCPRA
TCTVRAGF
TCTVRAM
TIOA
TIOADBA
TCTE
TCTEIFME
TCTEIRRN
TCTEIERN
TCTEIDIP
TIOA
TIOADBA
TCTTE
TCTTECAD

Receive-Any (DFHRAC)

Diagram - 1.5.11.1.4 - 03

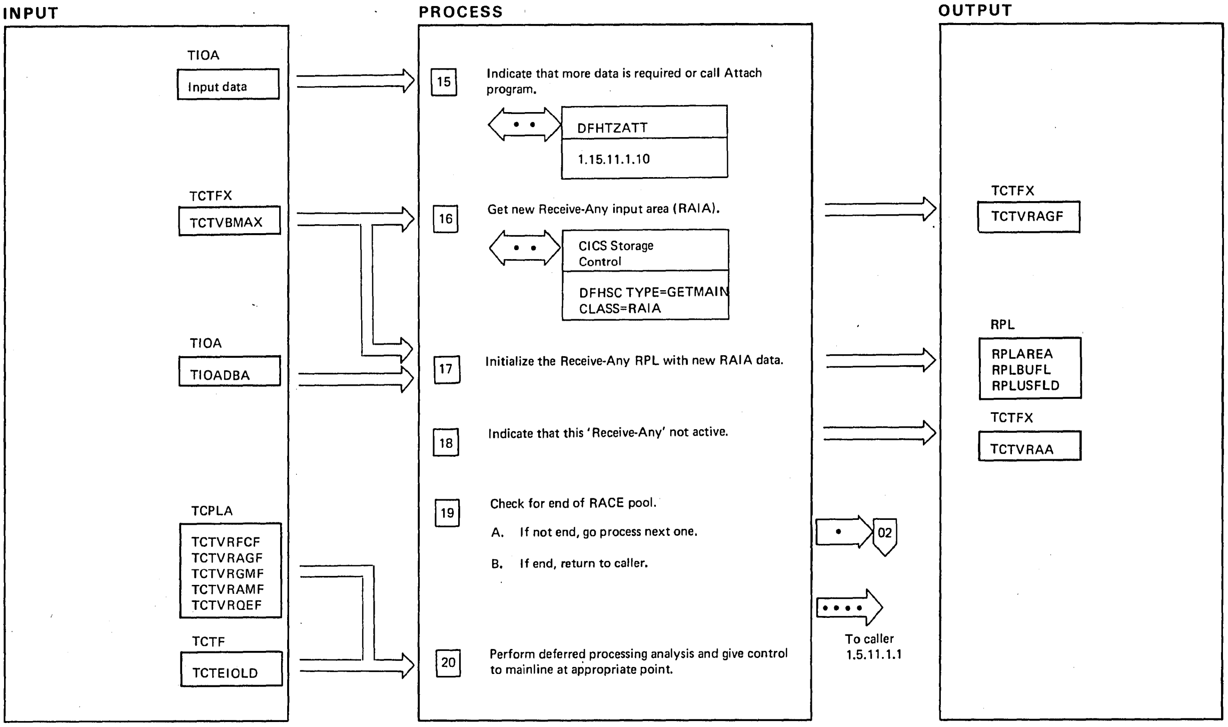
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 11 If no response is requested, an error is set up and the terminal entry is chained for processing by NACP, unless the terminal is a 3270. | | TCZBACB2 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Receive-Any (DFHRAC)

Diagram - 1.5.11.1.4 - 03

Page 4 of 4


INPUT
PROCESS
OUTPUT
TIOA
Input data
TCTFX
TCTVBMAX
TIOA
TIOADBA
TCPLA
TCTVRFCF
TCTVRAGF
TCTVRGMF
TCTVRAMF
TCTVRQEF
TCTF
TCTEIOLD
15
Indicate that more data is required or call Attach program.
DFHTZATT
1.15.11.1.10
16
Get new Receive-Any input area (RAIA).
CICS Storage Control
DFHSC TYPE=GETMAIN CLASS=RAIA
17
Initialize the Receive-Any RPL with new RAIA data.
18
Indicate that this 'Receive-Any' not active.
19
Check for end of RACE pool.
A. If not end, go process next one.
B. If end, return to caller.
02
To caller
1.5.11.1.1
20
Perform deferred processing analysis and give control to mainline at appropriate point.
TCTFX
TCTVRAGF
RPL
RPLAREA
RPLBUFL
RPLUSFLD
TCTFX
TCTVRAA

Receive-Any (DFHRAC)

Diagram - 1.5.11.1.4 - 04

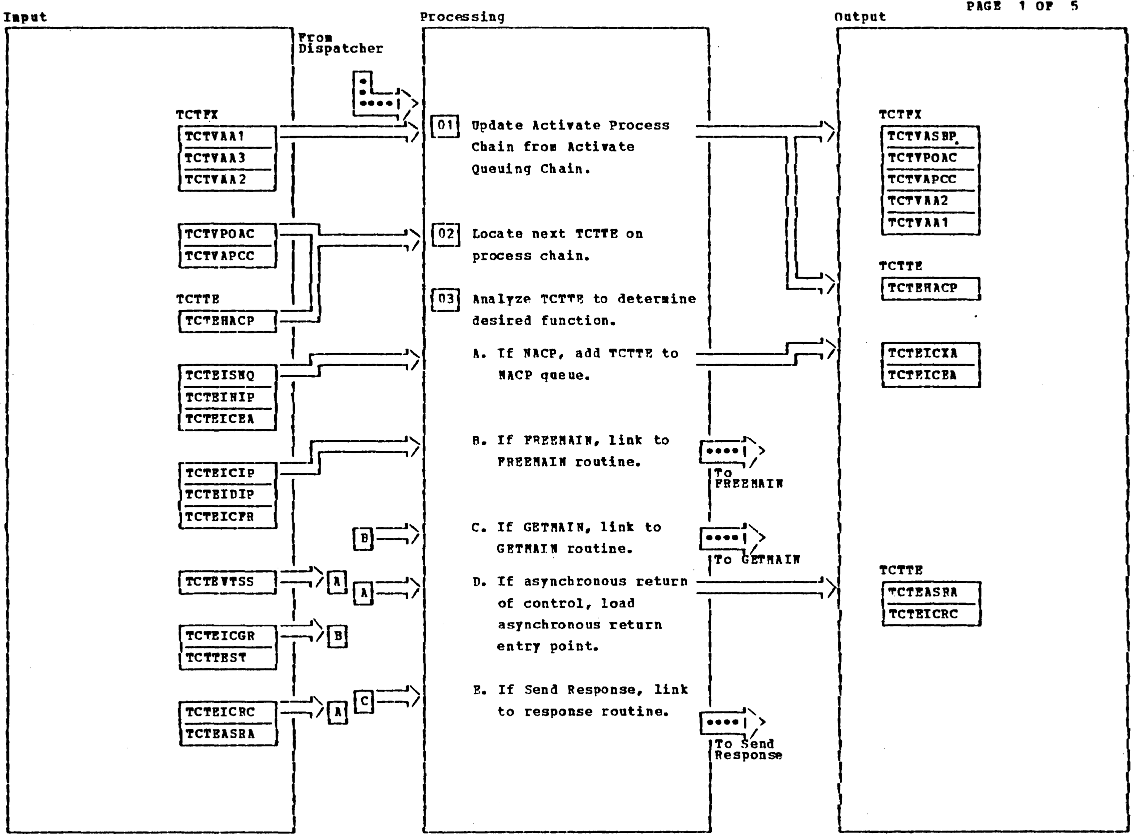
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 15 If a stand-alone FMR is received, indicator TCTRIDHC is set and no task is attached. | | | |
| 16 If the system is at SOS, a new RAIA will not be obtained. The TCTVEAG will remain on. | | TCZBACBG | |
| 17 If the system is at MAXTSK, the Receive-Any will be left inactive.<br>If authorized path SRB support is being used, the RPL service routine is invoked to reissue the request to VTAM. | DFHZHPSR | TCZBRCAP<br>1.5.11.1.37 | |
| 20 This routine provides a redrive mechanism. If, on a previous pass, a function could not be performed because of resource contraints, an indicator in the RACE shows where control should be returned in the mainline. | | TCZBACJ1 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Receive-Any (DFHRAC)

Diagram - 1.5.11.1.4 - 04

CHART: HIPO0001
PAGE 1 OF 5



01 Update Activate Process Chain from Activate Queuing Chain.

02 Locate next TCTTE on process chain.

03 Analyze TCTTE to determine desired function.

A. If NACP, add TCTTE to NACP queue.

B. If FREEMAIN, link to FREEMAIN routine.

C. If GETMAIN, link to GETMAIN routine.

D. If asynchronous return of control, load asynchronous return entry point.

E. If Send Response, link to response routine.

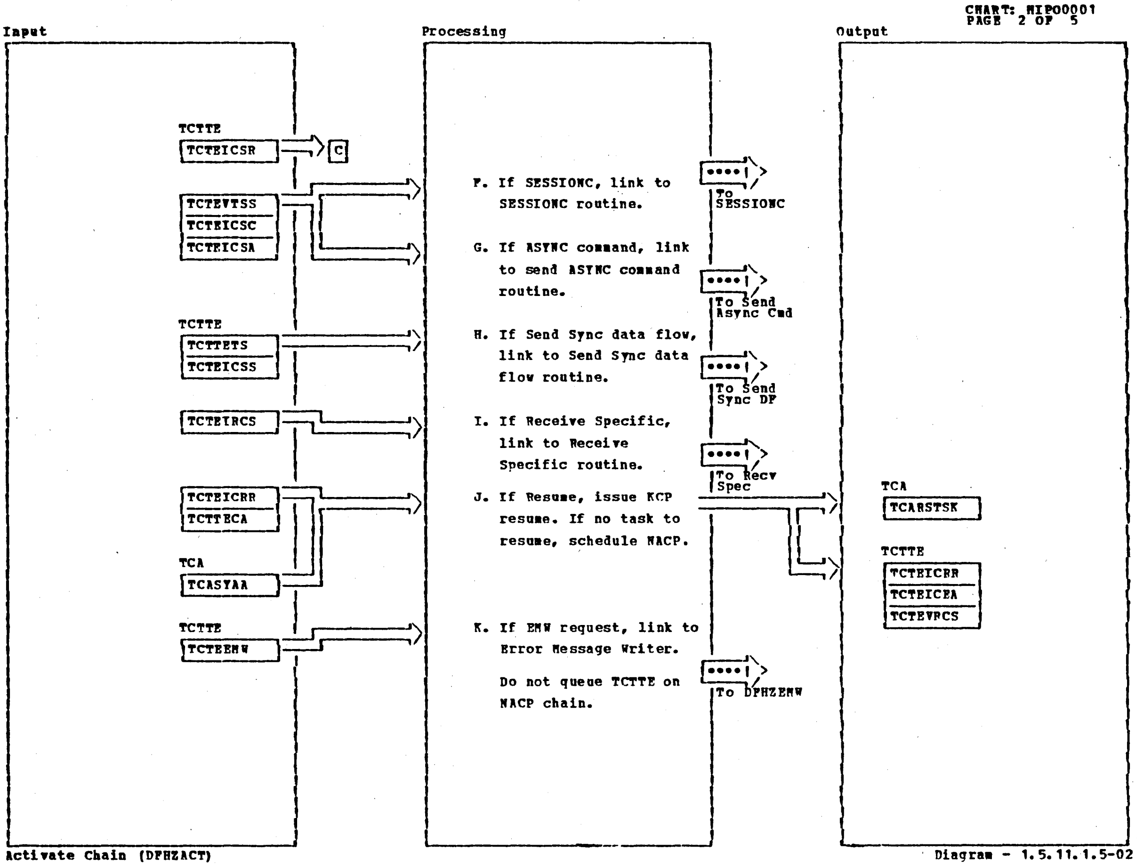
Activate Chain (DFHZACT) Diagram - 1.5.11.1.5-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 03 TCTTE is analyzed in the following order: | | | |
| NACP.............QUE | | | |
| FREEMAIN.........LINK | DFHZFRE | | |
| GETMAIN..........LINK | DFHZGET | | |
| ASYNC RETURN.....IN LINE | | | |
| SEND RESP........LINK | DFHZSDR | | |
| SESSIONC.........LINK | DFHZSES | | |
| SEND ASYNC.......LINK | DFHZSDA | | |
| SEND SYNC........LINK | DFHZSDS | | |
| RECEIVE SPEC.....LINK | DFHZRVS | | |
| RESUME...........MACRO | | | |
| ENW..............LINK | DFHZENW | | |
| DETACH...........LINK | DFHZDET | | |
| ATTACH...........LINK | DFHZATT | | |
| LOG RESPONSE.....QUE | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| RESETSR..........LINK | DFHZRST | | |
| CLSDST...........LINK | DFHZCLS | | |
| SIMLOGON.........LINK | DFHZSIM | | |
| OPNDST...........LINK | DFHZOPN | | |
| RESYNC...........LINK | DFHZRSY | | |
| ATI..............LINK | DFHZATI | | |

Activate Chain (DFHZACT) Diagram - 1.5.11.1.5-01

CHART: HIPO0001
PAGE 2 OF 5



F. If SESSIONC, link to SESSIONC routine.

G. If ASYNC command, link to send ASYNC command routine.

H. If Send Sync data flow, link to Send Sync data flow routine.

I. If Receive Specific, link to Receive Specific routine.

J. If Resume, issue KCP resume. If no task to resume, schedule NACP.

K. If EMW request, link to Error Message Writer.

Do not queue TCTTE on NACP chain.

Activate Chain (DFHZACT)

Diagram - 1.5.11.1.5-02

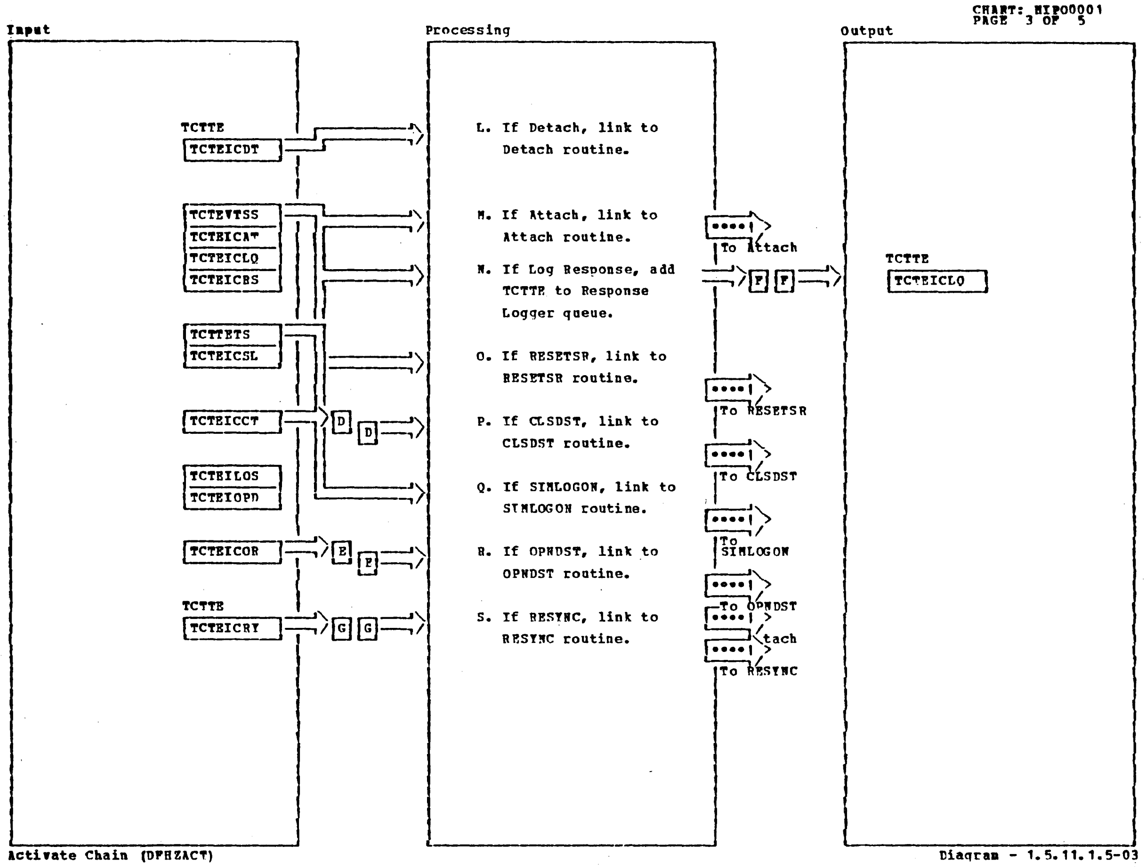
| Notes | Routine | Label | Ref |
|---|---|---|---|

| Notes | Routine | Label | Ref |
|---|---|---|---|

Activate Chain (DFHZACT)

Diagram - 1.5.11.1.5-02

CHART: HIP00001
PAGE 3 OF 5



L. If Detach, link to Detach routine.

M. If Attach, link to Attach routine.

N. If Log Response, add TCTTE to Response Logger queue.

O. If RESETSR, link to RESETSR routine.

P. If CLSDST, link to CLSDST routine.

Q. If SIMLOGON, link to SIMLOGON routine.

R. If OPNDST, link to OPNDST routine.

S. If RESYNC, link to RESYNC routine.

Activate Chain (DFHZACT)

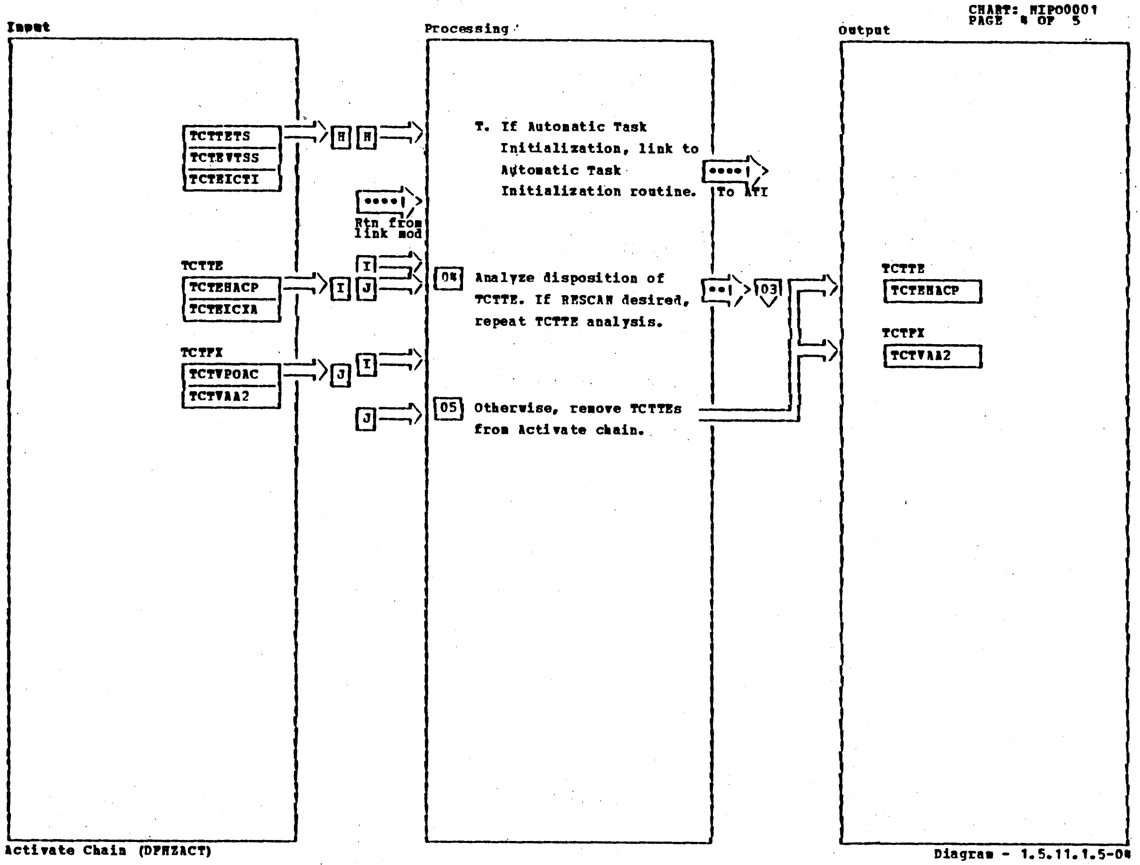
Diagram - 1.5.11.1.5-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Activate Chain (DFHZACT)

Diagram - 1.5.11.1.5-03

CHART: MIP00001
PAGE 4 OF 5



T. If Automatic Task Initialization, link to Automatic Task Initialization routine.

04 Analyze disposition of TCTTE. If RESCAN desired, repeat TCTTE analysis.

05 Otherwise, remove TCTTEs from Activate chain.

Activate Chain (DFHZACT)

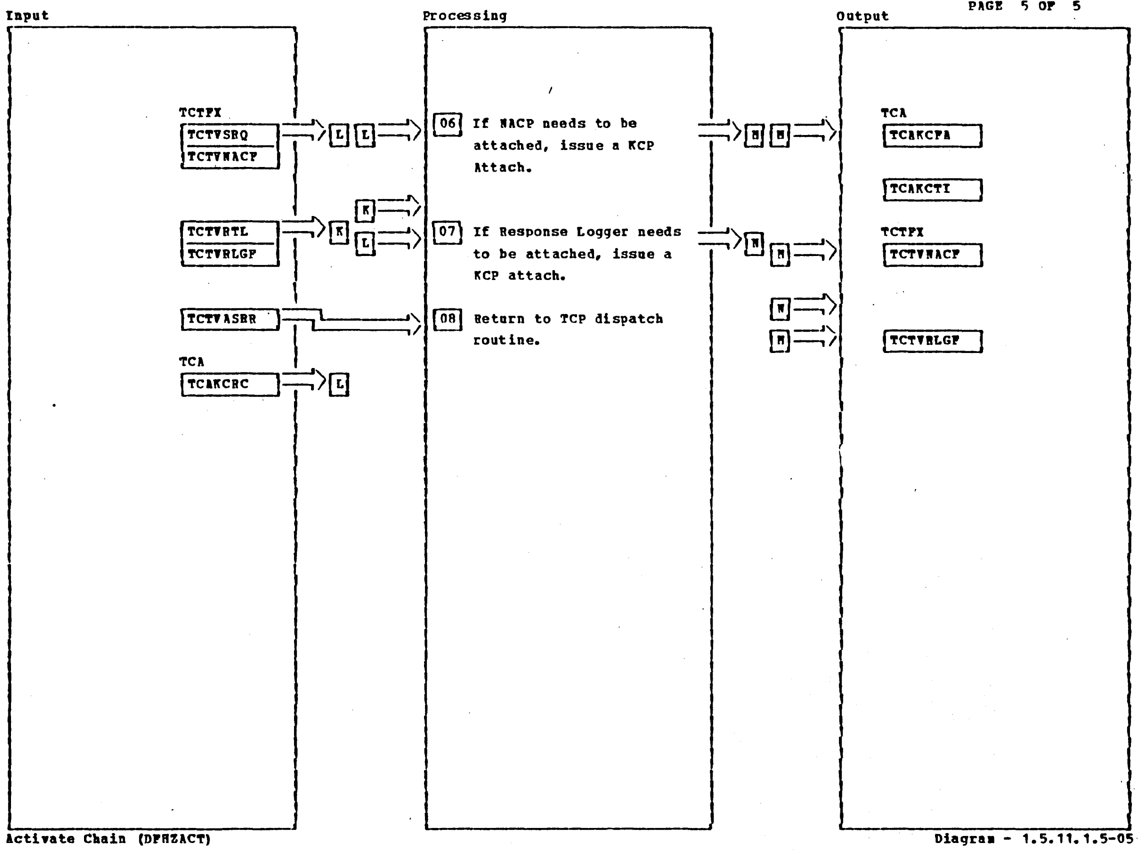
Diagram - 1.5.11.1.5-04

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

Activate Chain (DFHZACT)

Diagram - 1.5.11.1.5-04

CHART: HIPO0001
PAGE 5 OF 5



06 If NACP needs to be attached, issue a KCP Attach.

07 If Response Logger needs to be attached, issue a KCP attach.

08 Return to TCP dispatch routine.

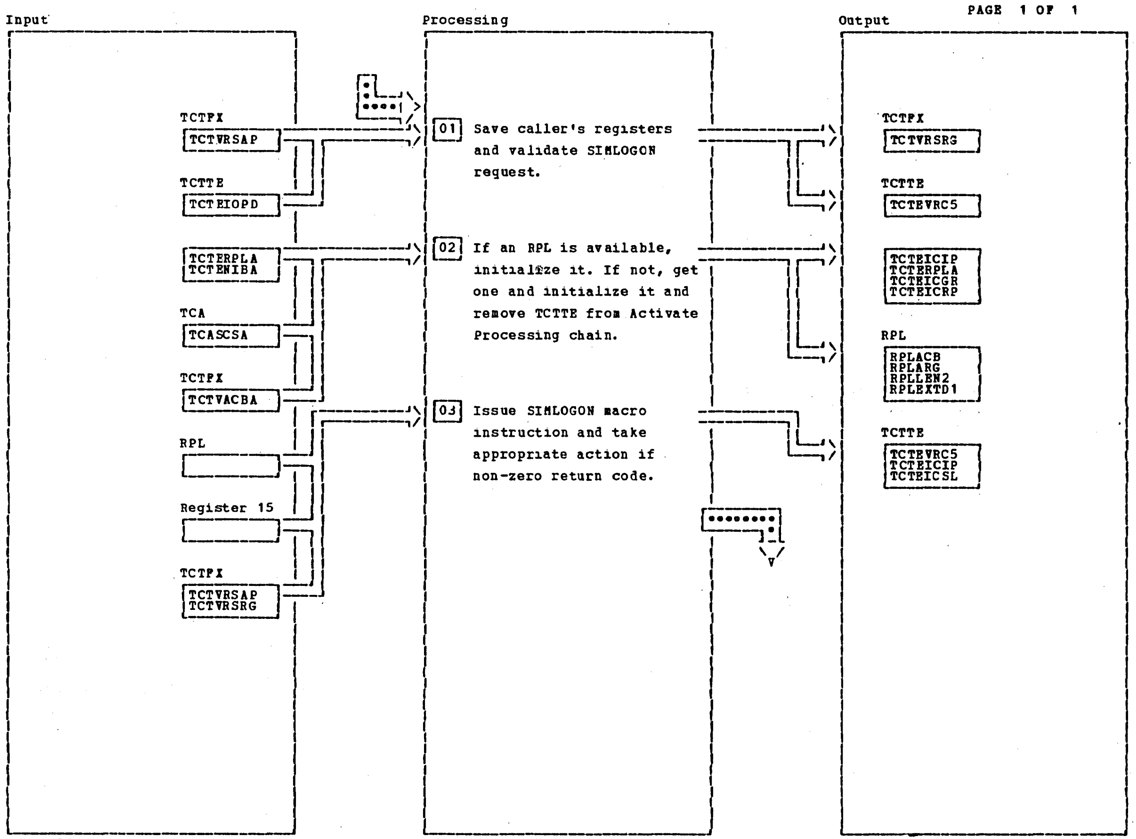
Activate Chain (DFHZACT)

Diagram - 1.5.11.1.5-05

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

Activate Chain (DFHZACT)

Diagram - 1.5.11.1.5-05



01 Save caller's registers and validate SIMLOGON request.

02 If an RPL is available, initialize it. If not, get one and initialize it and remove TCTTE from Activate Processing chain.

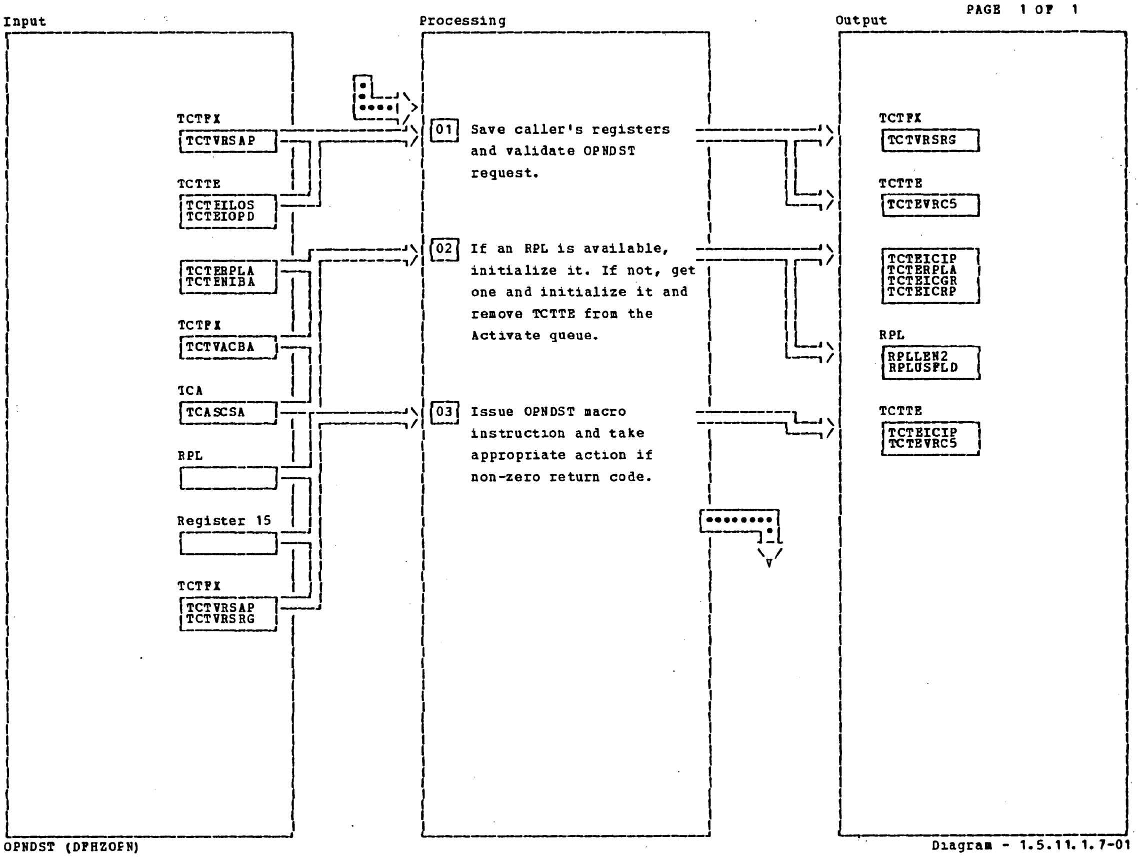
03 Issue SIMLOGON macro instruction and take appropriate action if non-zero return code.

SIMLOGON (DFHZSIM) Diagram - 1.5.11.1.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Check Activate Scan request bits to insure this is a valid SIMLOGON request. | | | | | | | |
| 02 A storage control GETMAIN is used to acquire RPL storage. If it should fail, an RPL GETMAIN is queued for that TCTTE.<br>The DFHZXRPL macro instruction is issued to insure integrity of an existing RPL. | | | | | | | |
| 03 If register 15 is negative, LERAD or SYNAD have done the error processing. Return.<br>If register 15 is X'F0' the TCTTE is placed back on the Activate queue for another attempt at SIMLOGON. X'F0' means VTAM has experienced a temporary storage problem.<br>If register 15 is none of the above, the TCTTE is queued to NACP for error processing. | | | | | | | |

SIMLOGON (DFHZSIM) Diagram - 1.5.11.1.6-01

PAGE 1 OF 1



01 Save caller's registers and validate OPNDST request.

02 If an RPL is available, initialize it. If not, get one and initialize it and remove TCTTE from the Activate queue.

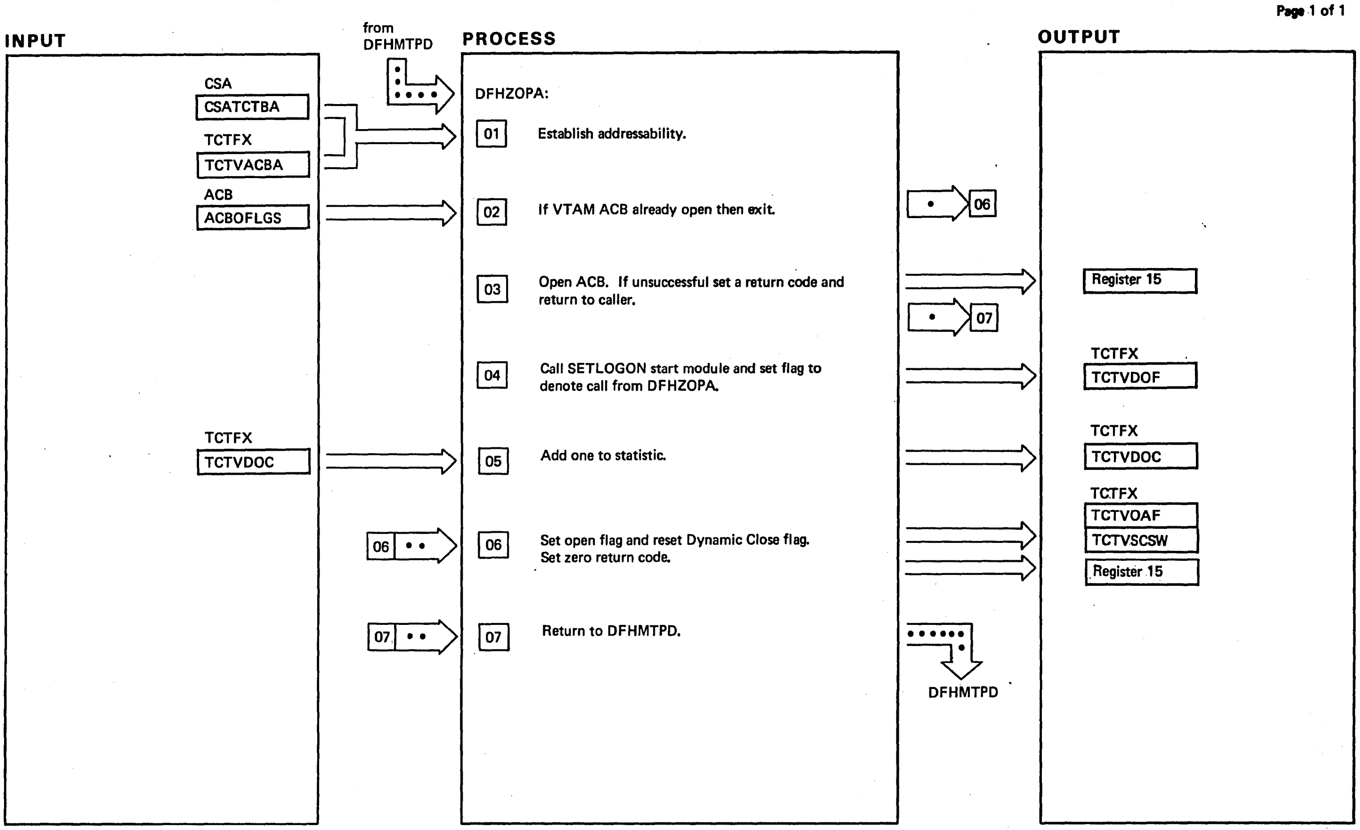
03 Issue OPNDST macro instruction and take appropriate action if non-zero return code.

OPNDST (DFHZOPN) Diagram - 1.5.11.1.7-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Check Activate Request bit to insure OPNDST should be done. | | | | | | | |
| 02 A storage control GETMAIN is used to acquire RPL storage. If it should fail, an RPL GETMAIN is queued for that TCTTE.<br>The DFHZXRPL macro instruction is issued to insure the integrity of an existing RPL. | | | | | | | |
| 03 If register 15 is negative, LERAD or SYNAD have done the error processing. Return.<br>If register 15 is X'F0', the TCTTE is placed back on the Activate Processing queue for another attempt at OPNDST. X'F0' means VTAM has experienced a temporary storage problem.<br>If register 15 is none of the above, queue the TCTTE to NACP. | | | | | | | |

OPNDST (DFHZOPN) Diagram - 1.5.11.1.7-01

Page 1 of 1



DFHZOPA:

01 Establish addressability.

02 If VTAM ACB already open then exit.

03 Open ACB. If unsuccessful set a return code and return to caller.

04 Call SETLOGON start module and set flag to denote call from DFHZOPA.

05 Add one to statistic.

06 Set open flag and reset Dynamic Close flag. Set zero return code.

07 Return to DFHMTPD.

Dynamic Open of VTAM ACB (DFHZOPA)

Diagram 1.5.11.1.7A-01

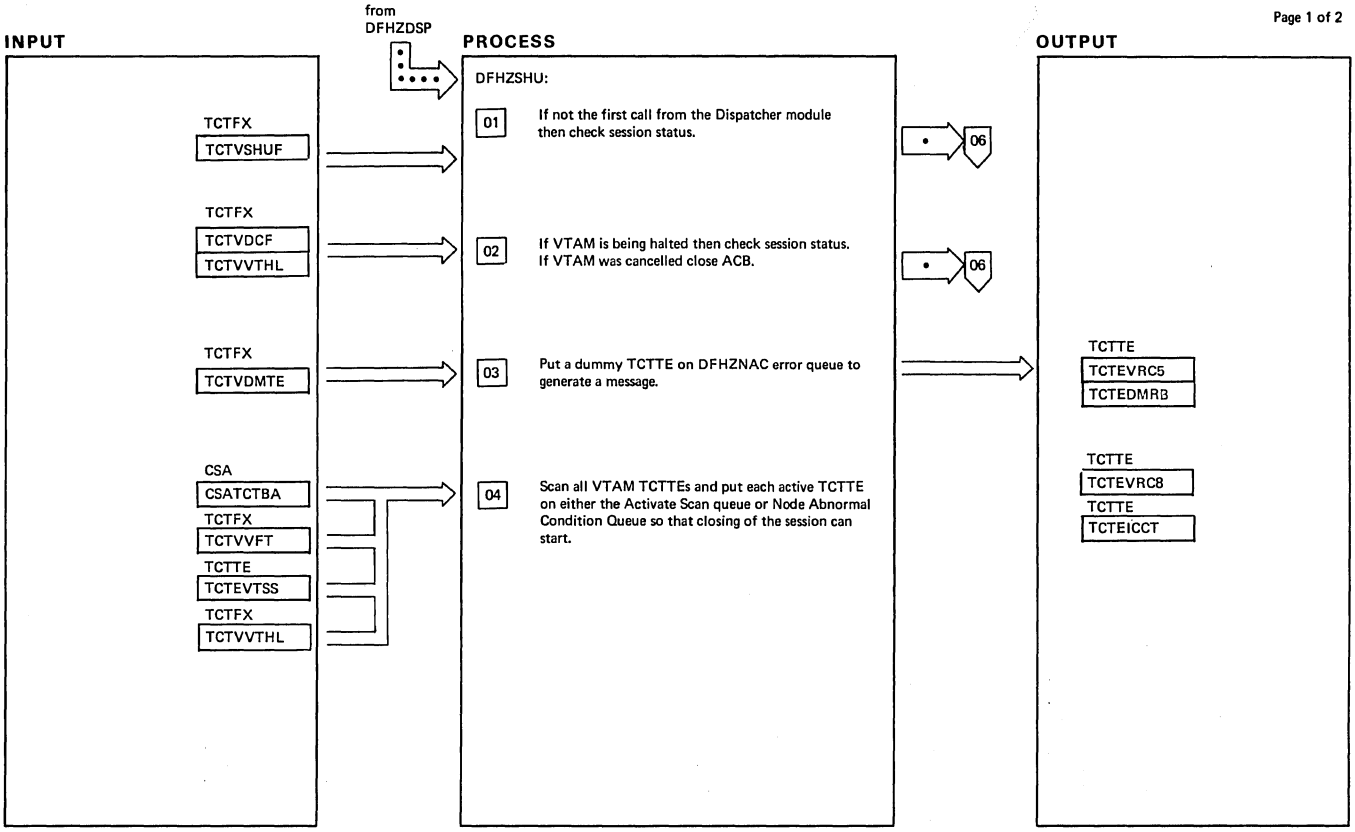
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 04 SETLOGON start module. | DFHZSLS | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Dynamic Open of VTAM ACB (DFHZOPA)

Diagram 1.5.11.1.7A-01

Page 1 of 2



DFHZSHU:

01 If not the first call from the Dispatcher module then check session status.

02 If VTAM is being halted then check session status. If VTAM was cancelled close ACB.

03 Put a dummy TCTTE on DFHZNAC error queue to generate a message.

04 Scan all VTAM TCTTEs and put each active TCTTE on either the Activate Scan queue or Node Abnormal Condition Queue so that closing of the session can start.

Dynamic Close of VTAM ACB (DFHZSHU)

Diagram 1.5.11.1.7B-01

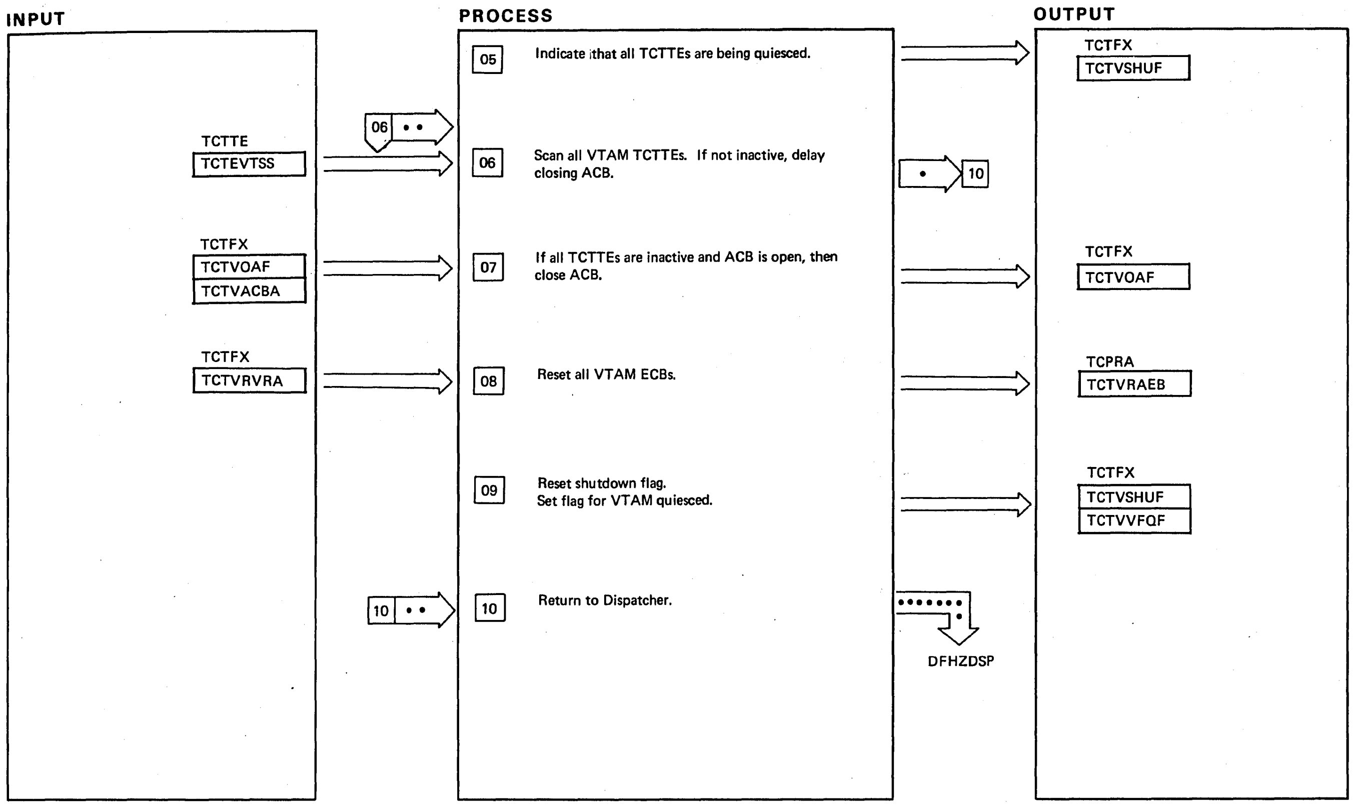
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Dynamic Close of VTAM ACB (DFHZSHU)

Diagram 1.5.11.1.7B-01

Page 2 of 2



05 Indicate ithat all TCTTEs are being quiesced.

06 Scan all VTAM TCTTEs. If not inactive, delay closing ACB.

07 If all TCTTEs are inactive and ACB is open, then close ACB.

08 Reset all VTAM ECBs.

09 Reset shutdown flag.
Set flag for VTAM quiesced.

10 Return to Dispatcher.

Dynamic Close of VTAM ACB (DFHZSHU)

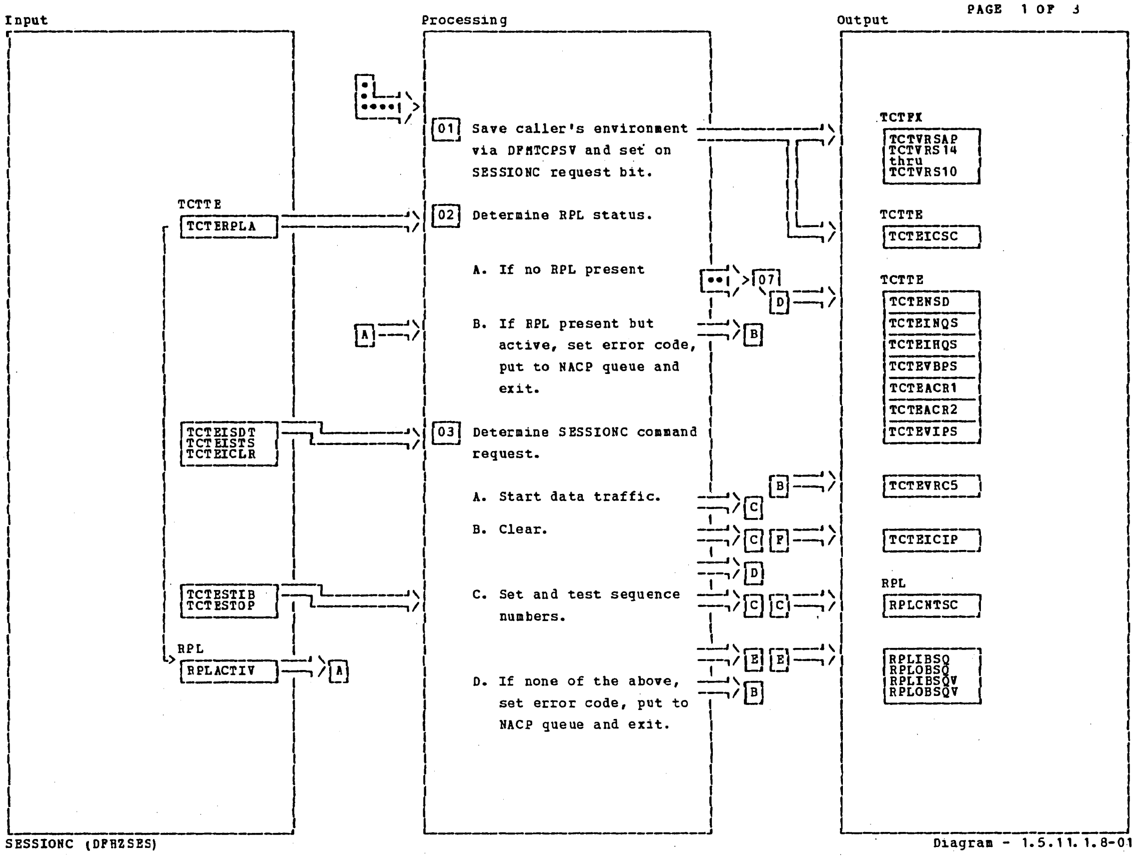
Diagram 1.5.11.1.7B-02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Dynamic Close of VTAM ACB (DFHZSHU)

Diagram 1.5.11.1.7B-02



01 Save caller's environment via DFHTCPSV and set on SESSIONC request bit.

02 Determine RPL status.

A. If no RPL present

B. If RPL present but active, set error code, put to NACP queue and exit.

03 Determine SESSIONC command request.

A. Start data traffic.

B. Clear.

C. Set and test sequence numbers.

D. If none of the above, set error code, put to NACP queue and exit.

SESSIONC (DFHZSES)

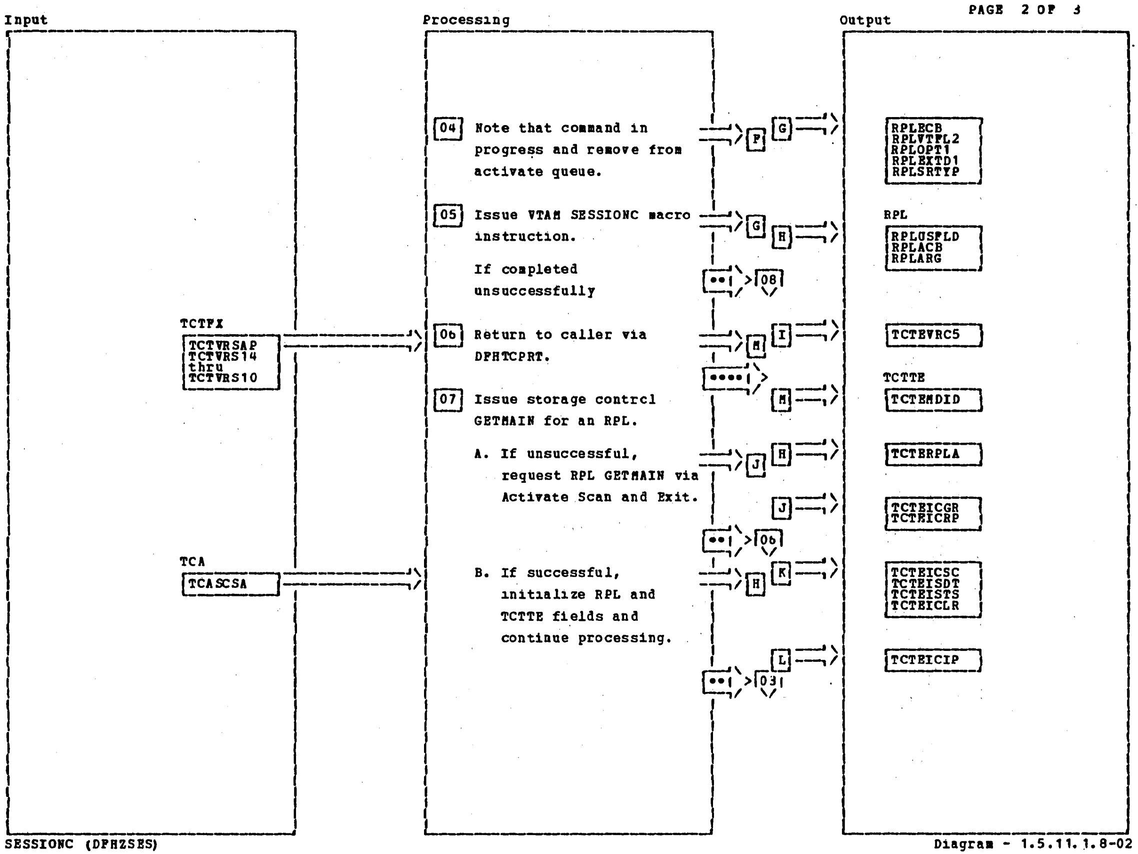
Diagram - 1.5.11.1.8-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

SESSIONC (DFHZSES)

Diagram - 1.5.11.1.8-01

PAGE 2 OF 3



Input

Processing

Output

04 Note that command in progress and remove from activate queue.

05 Issue VTAM SESSIONC macro instruction.

If completed unsuccessfully

06 Return to caller via DFHTCPRT.

07 Issue storage control GETMAIN for an RPL.

A. If unsuccessful, request RPL GETMAIN via Activate Scan and Exit.

B. If successful, initialize RPL and TCTTE fields and continue processing.

SESSIONC (DFHZSES)

Diagram - 1.5.11.1.8-02

| Notes | Routine | Label | Ref |
|---|---|---|---|
| | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| | | | |

SESSIONC (DFHZSES)

Diagram - 1.5.11.1.8-02

PAGE 3 OF 3

Input

Processing

08 Error analysis of unsuccessful VTAM SESSIONC. → L

A. LERAD/SYNAD handled error, clean up and exit. → K

B. Handle own error, queue to Activate chain and exit. → 06

→ 06

C. Assume VTAM is dead, put to NACP queue and exit. → I

→ 06

Output

SESSIONC (DFHZSES)

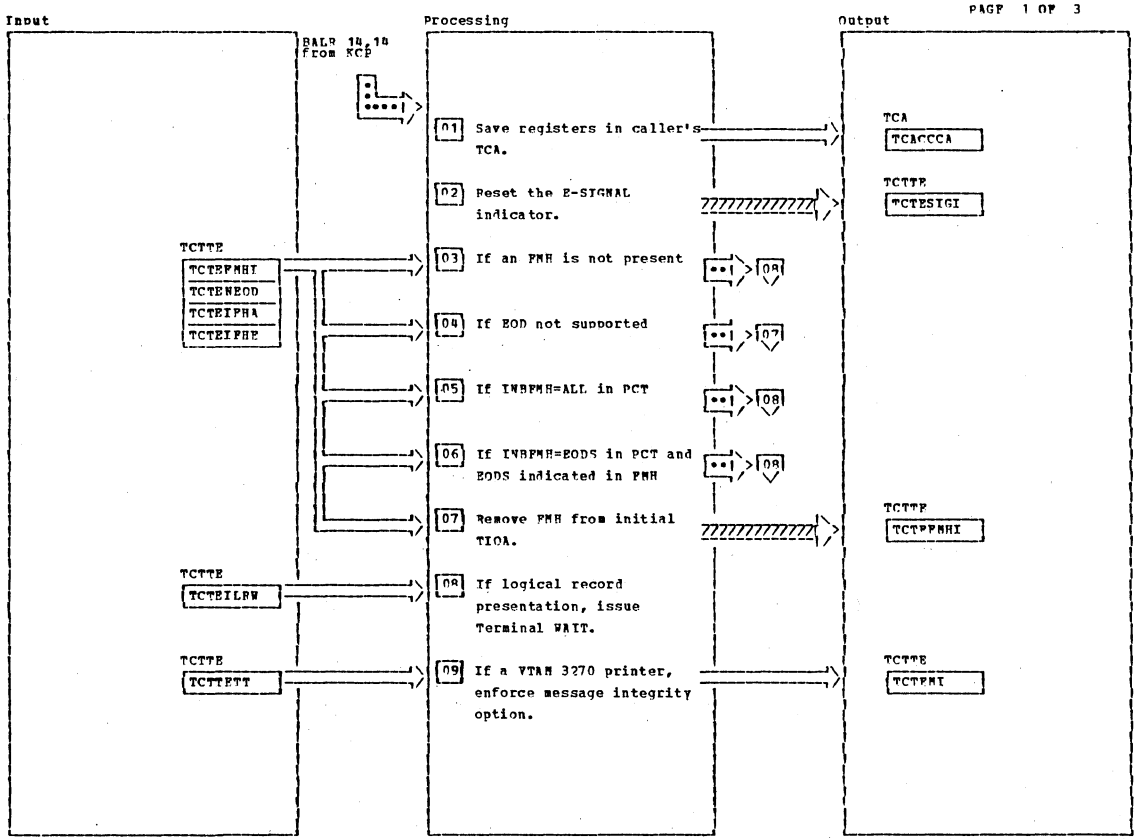
Diagram - 1.5.11.1.8-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 08 VTAM return code (in register 15) will be 'FF', 'F0', or '20' meaning, respectively:<br>- error condition has already been handled by LERAD/SYNAD<br>- handle your own error<br>- VTAM is dead | | | | | | | |

SESSIONC (DFHZSES)

Diagram - 1.5.11.1.8-03

PAGE 1 OF 3



01 Save registers in caller's TCA.

02 Reset the E-SIGNAL indicator.

03 If an FMH is not present

04 If EOD not supported

05 If INBFMH=ALL in PCT

06 If INBFMH=EODS in PCT and EODS indicated in FMH

07 Remove FMH from initial TIOA.

08 If logical record presentation, issue Terminal WAIT.

09 If a VTAM 3270 printer, enforce message integrity option.

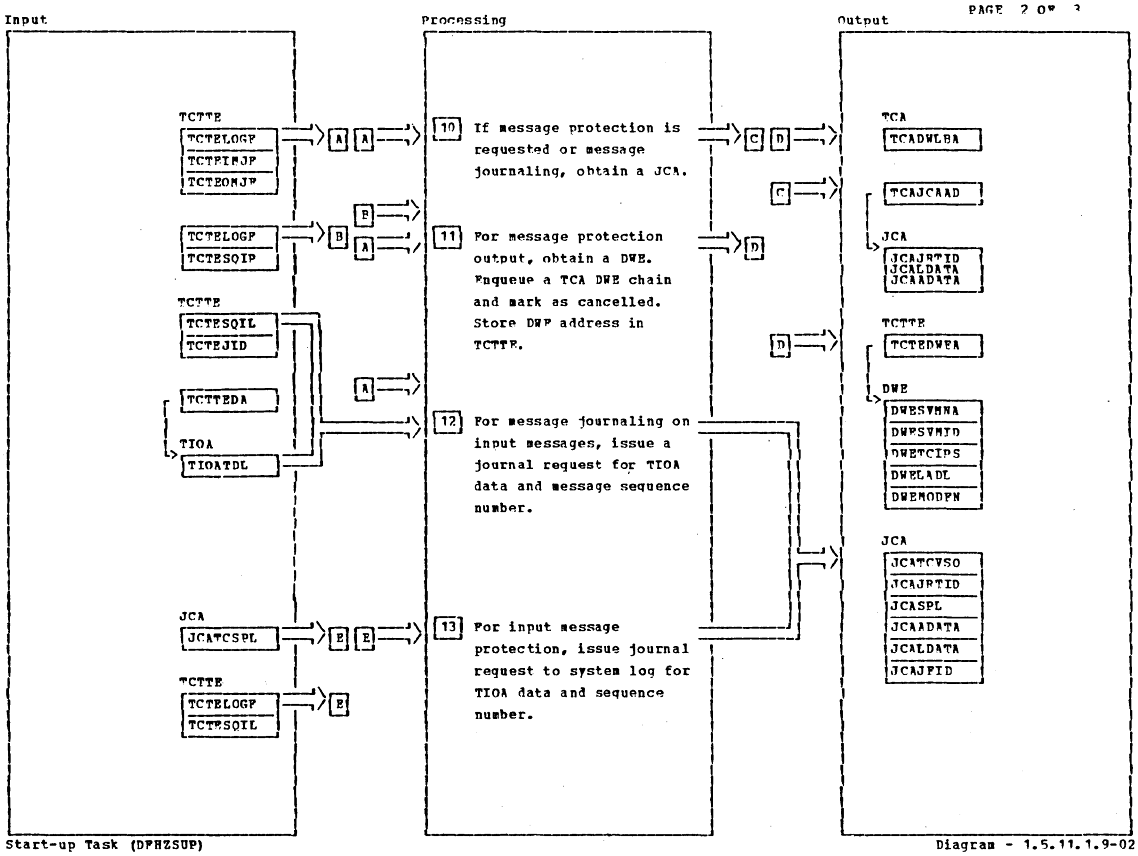
Start-up Task (DFHZSUP)

Diagram - 1.5.11.1.9-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Control is received directly by a BALR 14,14 the first time the task is dispatched, following its creation by ATTACH. This subroutine operates under the user's TCA. | | | | | | | |
| 08 KC Wait is non-dispatchable. The task is resumed when deblocking has completed. | | | | | | | |
| 09 VTAM 3270 printers require to be run in definite response mode. | | | | | | | |

Start-up Task (DFHZSUP)

Diagram - 1.5.11.1.9-01



10 If message protection is requested or message journaling, obtain a JCA.

11 For message protection output, obtain a DWE. Enqueue a TCA DWE chain and mark as cancelled. Store DWE address in TCTTE.

12 For message journaling on input messages, issue a journal request for TIOA data and message sequence number.

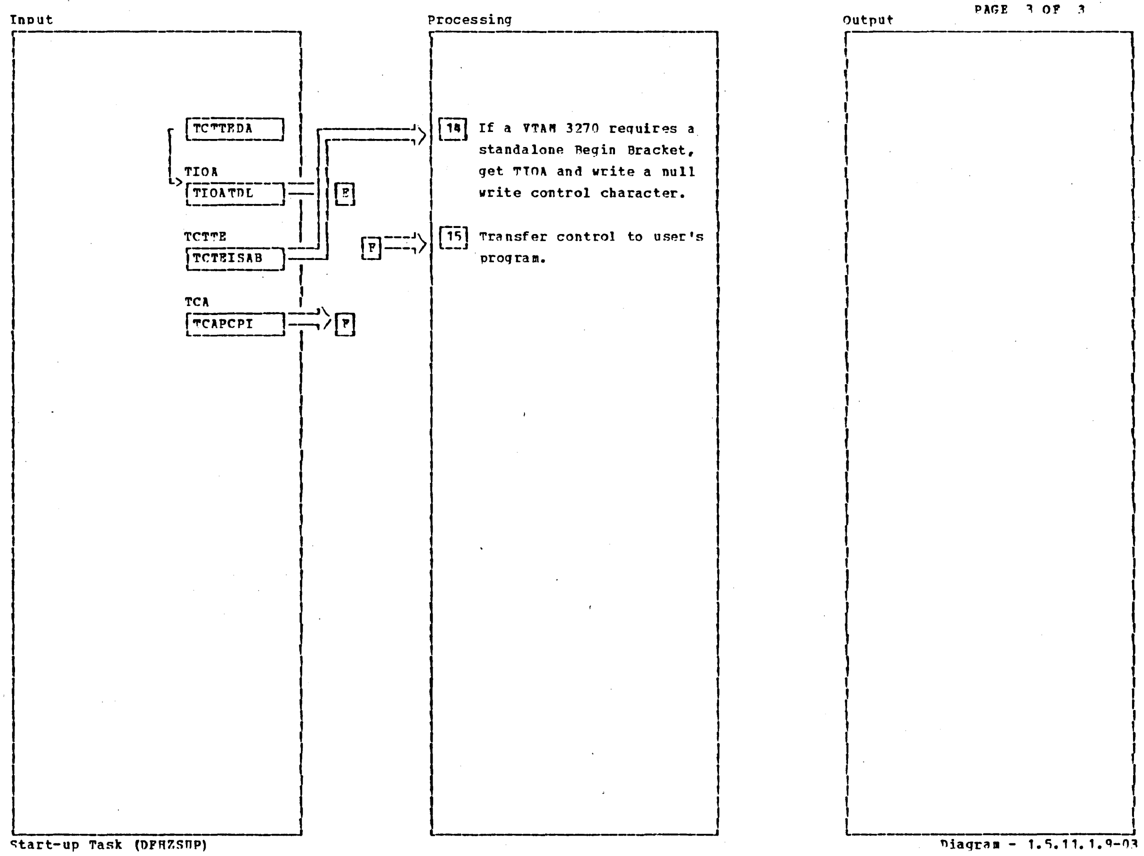
13 For input message protection, issue journal request to system log for TIOA data and sequence number.

Start-up Task (DFHZSUP) Diagram - 1.5.11.1.9-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 10 Any class of message protection or journaling will require a JCA potentially so it is always allocated. | | | | | | | |
| 11 Output message protection potentially requires a DWE so it is always allocated. | | | | | | | |
| 12 Automatic journaling of input messages is performed here for the first task originating message. | | | | | | | |
| 13 Input message protection requires that the first message and its sequence number be put to the system log for availability to emergency restart-process routines in case of catastrophic system failure. | | | | | | | |

Start-up Task (DFHZSUP) Diagram - 1.5.11.1.9-02

PAGE 3 OF 3



14 If a VTAM 3270 requires a standalone Begin Bracket, get TIOA and write a null write control character.

15 Transfer control to user's program.

Start-up Task (DFHZSUP)

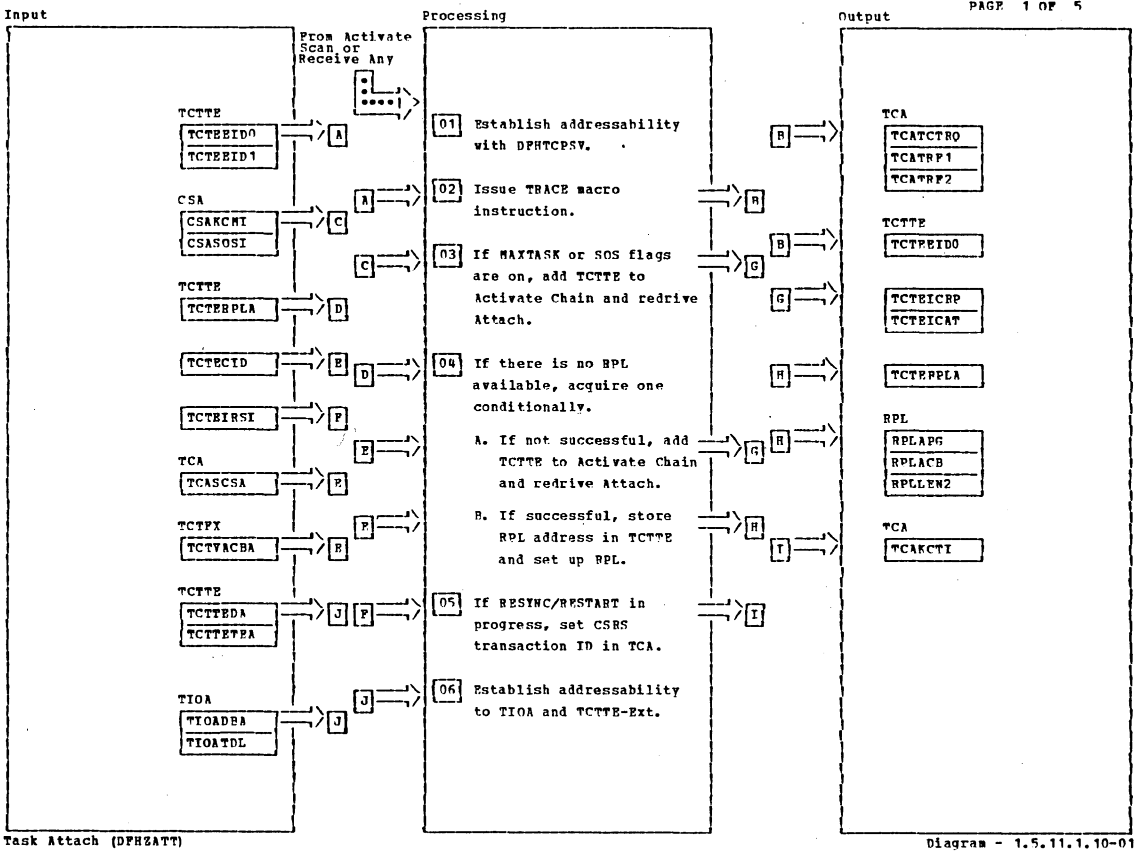
Diagram - 1.5.11.1.9-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 14 For a VTAM 3270 terminal, if no data has flowed since the session started, a standalone Begin Bracket is sent with definite response via a DFHTC TYPE=(WRITE,WAIT) macro. | | | | | | | |
| 15 DFHZSUP invokes program control to transfer control to the user's program, whose name was in the TCA when DFHZSUP got control. | | | | | | | |

Start-up Task (DFHZSUP)

Diagram - 1.5.11.1.9-03

CHART: HIPO0001
PAGE 1 OF 5



01 Establish addressability with DFHTCPSV.

02 Issue TRACE macro instruction.

03 If MAXTASK or SOS flags are on, add TCTTE to Activate Chain and redrive Attach.

04 If there is no RPL available, acquire one conditionally.

A. If not successful, add TCTTE to Activate Chain and redrive Attach.

B. If successful, store RPL address in TCTTE and set up RPL.

05 If RESYNC/RESTART in progress, set CSRS transaction ID in TCA.

06 Establish addressability to TIOA and TCTTE-Ext.

Task Attach (DFHZATT)

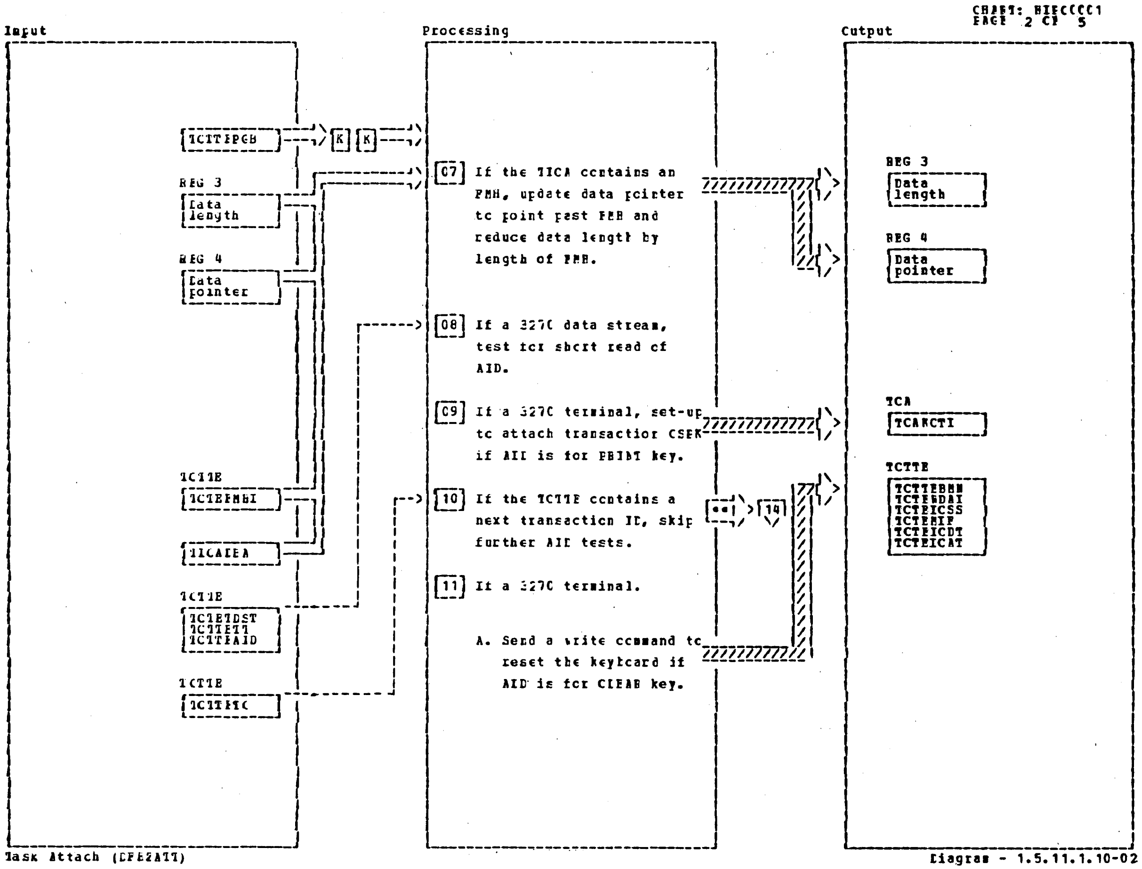
Diagram - 1.5.11.1.10-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 | | TCZATT94 | | | | | |
| 04 DFHSC TYPE=GETMAIN CLASS=RPL<br>Register 15 is set to 8 for Activate Scan if TCTTE is added to chain. | | | | | | | |

Task Attach (DFHZATT)

Diagram - 1.5.11.1.10-01

CHART: HIECCCC1
PAGE 2 OF 5



07 If the TIOA contains an FMH, update data pointer to point past FMH and reduce data length by length of FMH.

08 If a 3270 data stream, test for short read of AID.

09 If a 3270 terminal, set-up to attach transaction CSPK if AID is for PRINT key.

10 If the TCTTE contains a next transaction ID, skip further AID tests.

11 If a 3270 terminal.

A. Send a write command to reset the keyboard if AID is for CLEAR key.

Task Attach (DFHZATT)

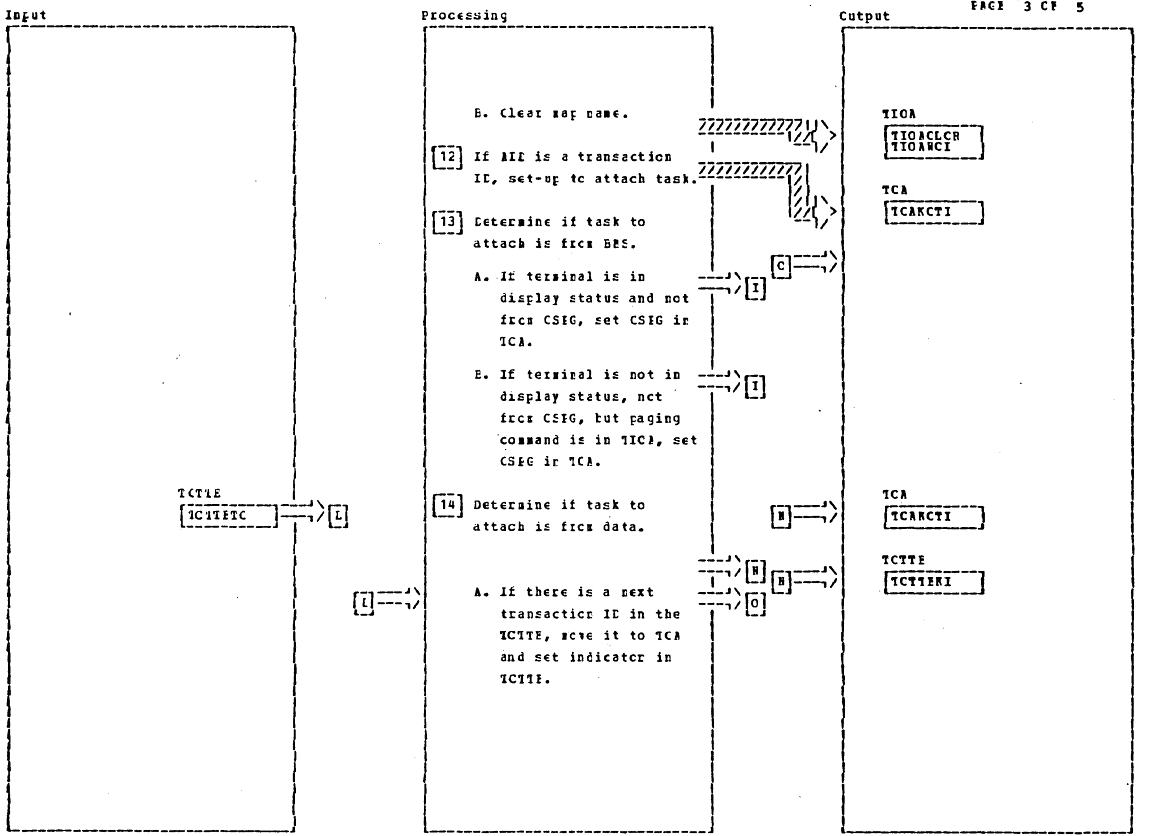
Diagram - 1.5.11.1.10-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Task Attach (DFHZATT)

Diagram - 1.5.11.1.10-02

CHART: HIFCCCC1
PAGE 3 OF 5



B. Clear map name.

12 If AID is a transaction ID, set-up tc attach task.

13 Determine if task to attach is frcm BMS.

A. If terminal is in display status and not frcm CSPG, set CSPG in TCA.

B. If terminal is not in display status, nct frcm CSPG, but paging command is in TIOA, set CSPG in TCA.

14 Determine if task to attach is frcm data.

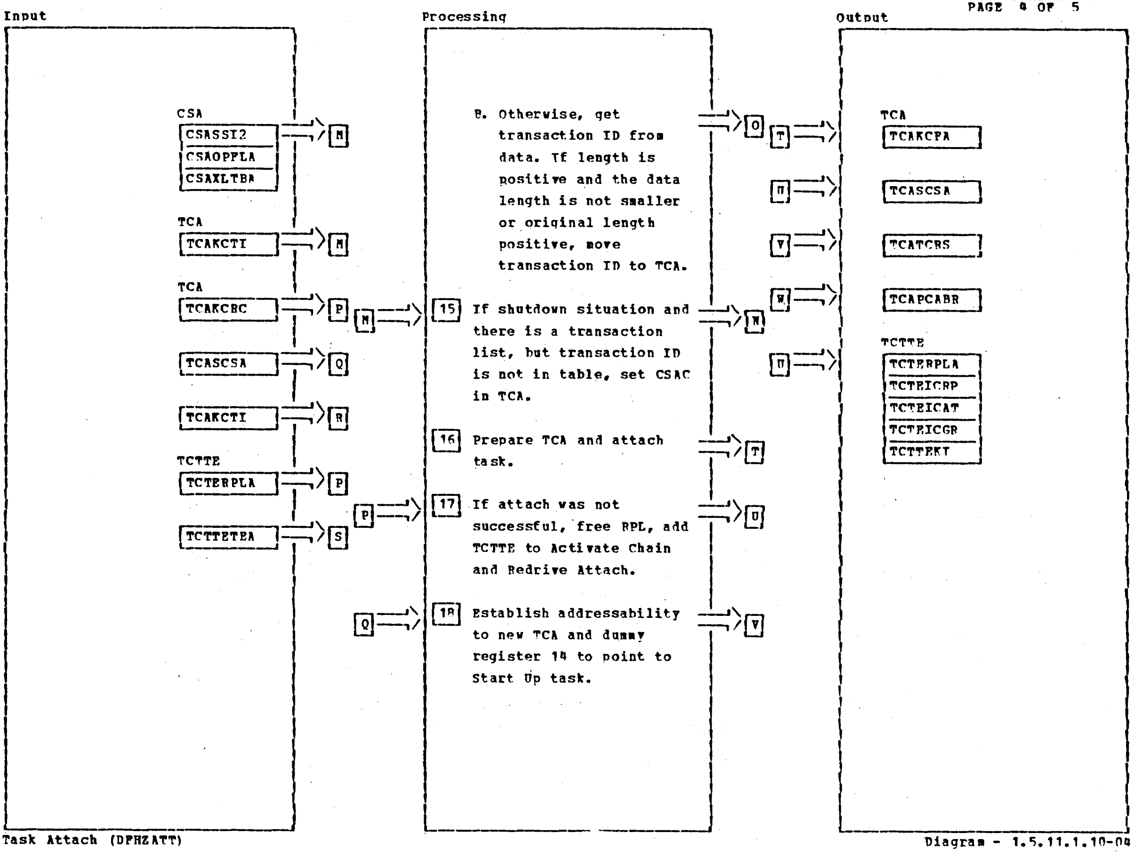
A. If there is a next transaction ID in the TCTTE, move it to TCA and set indicator in TCTTE.

Task Attach (DFH2A11)    Diagram - 1.5.11.1.10-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Task Attach (DFH2A11)    Diagram - 1.5.11.1.10-03

CHART: HIPO0001
PAGE 4 OF 5



B. Otherwise, get transaction ID from data. If length is positive and the data length is not smaller or original length positive, move transaction ID to TCA.

15 If shutdown situation and there is a transaction list, but transaction ID is not in table, set CSAC in TCA.

16 Prepare TCA and attach task.

17 If attach was not successful, free RPL, add TCTTE to Activate Chain and Redrive Attach.

18 Establish addressability to new TCA and dummy register 14 to point to Start Up task.

Task Attach (DFHZATT)

Diagram - 1.5.11.1.10-04

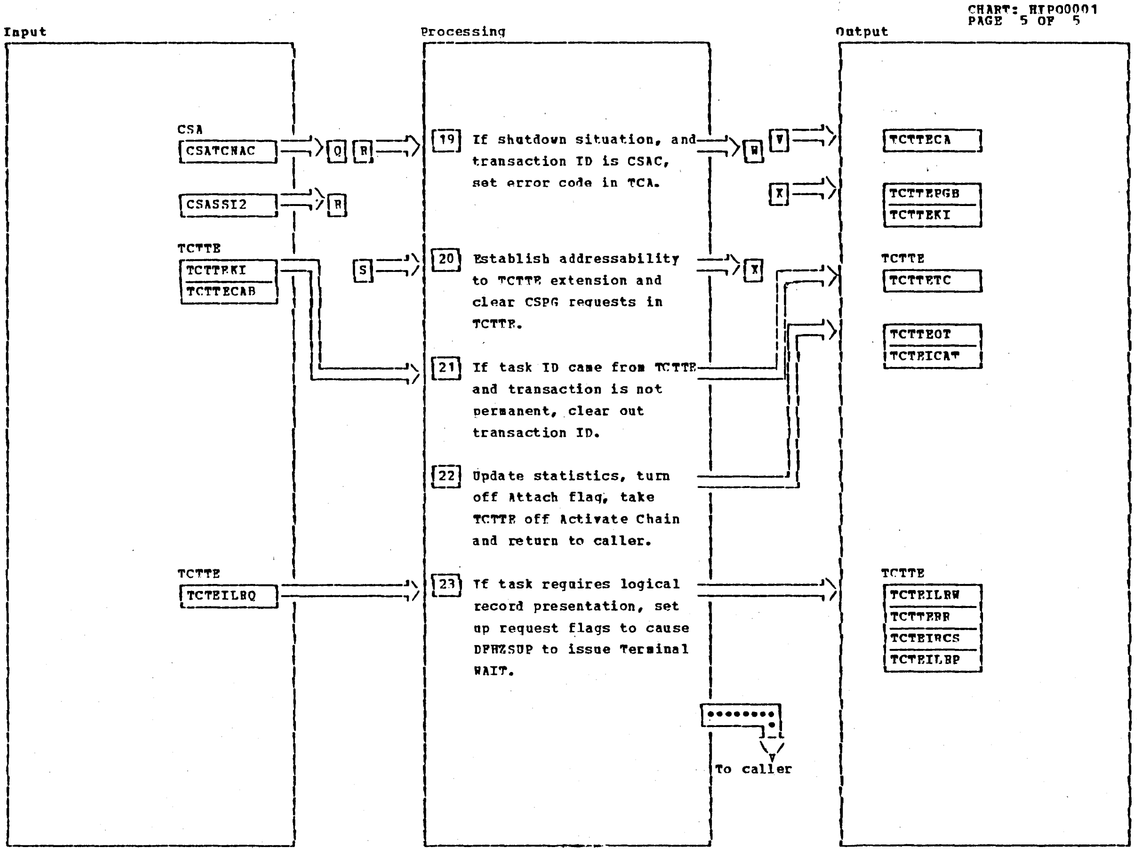
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 11 The SEND SYNCH DATA FLOW in DFHZCP is called. | DFHZSDS | TCZATT15 | 1.5.11 .1.17 |
| 16 Link to user exit before Attach, if requested. | | | |
| 17 DFHSC FREEMAIN<br>Register 15 is set to 8 for Activate Scan. | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| | | | |

Task Attach (DFHZATT)

Diagram - 1.5.11.1.10-04

CHART: HIPO0001
PAGE 5 OF 5



19 If shutdown situation, and transaction ID is CSAC, set error code in TCA.

20 Establish addressability to TCTTE extension and clear CSPG requests in TCTTE.

21 If task ID came from TCTTE and transaction is not permanent, clear out transaction ID.

22 Update statistics, turn off Attach flag, take TCTTE off Activate Chain and return to caller.

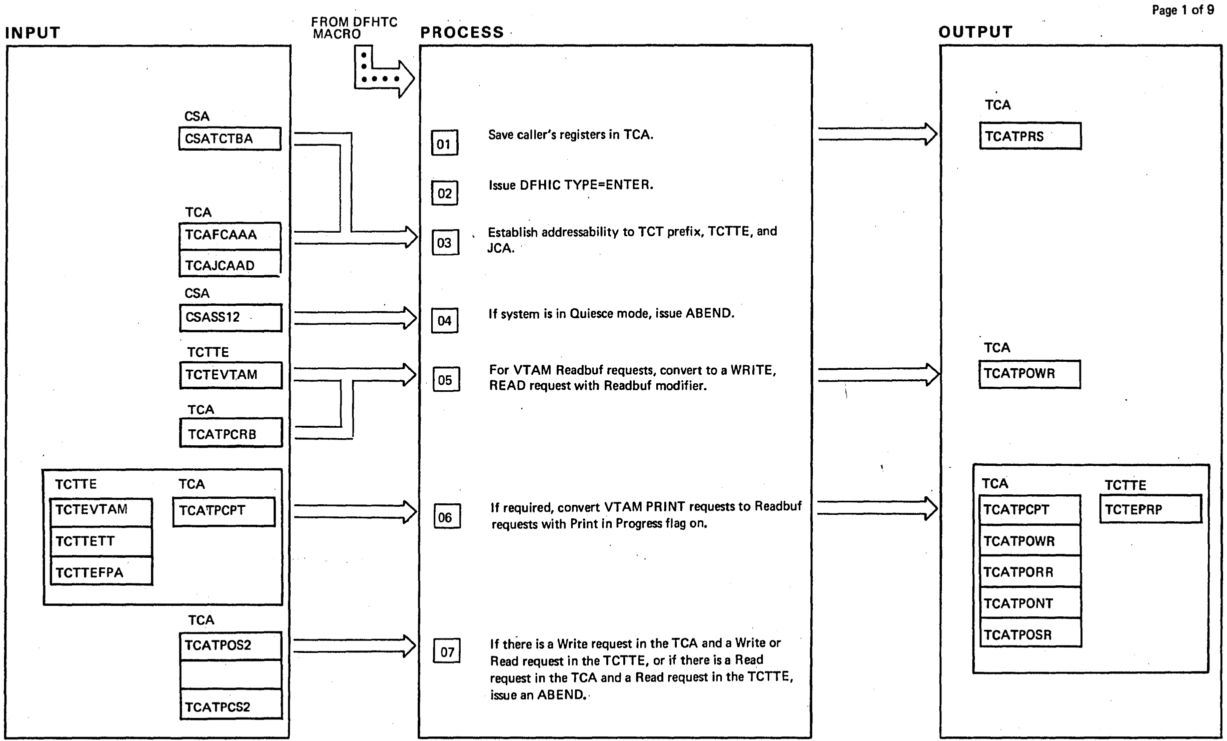
23 If task requires logical record presentation, set up request flags to cause DFHZSUP to issue Terminal WAIT.

Task Attach (DFHZATT) Diagram - 1.5.11.1.10-05

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 22 DFHTCQUE QUE=ACTIVATE CHAIN=OFF | | | | | | | |

Task Attach (DFHZATT) Diagram - 1.5.11.1.10-05

Page 1 of 9



01 Save caller's registers in TCA.

02 Issue DFHIC TYPE=ENTER.

03 Establish addressability to TCT prefix, TCTTE, and JCA.

04 If system is in Quiesce mode, issue ABEND.

05 For VTAM Readbuf requests, convert to a WRITE, READ request with Readbuf modifier.

06 If required, convert VTAM PRINT requests to Readbuf requests with Print in Progress flag on.

07 If there is a Write request in the TCA and a Write or Read request in the TCTTE, or if there is a Read request in the TCA and a Read request in the TCTTE, issue an ABEND.

Application Request

Diagram – 1.5.11.1.11–01

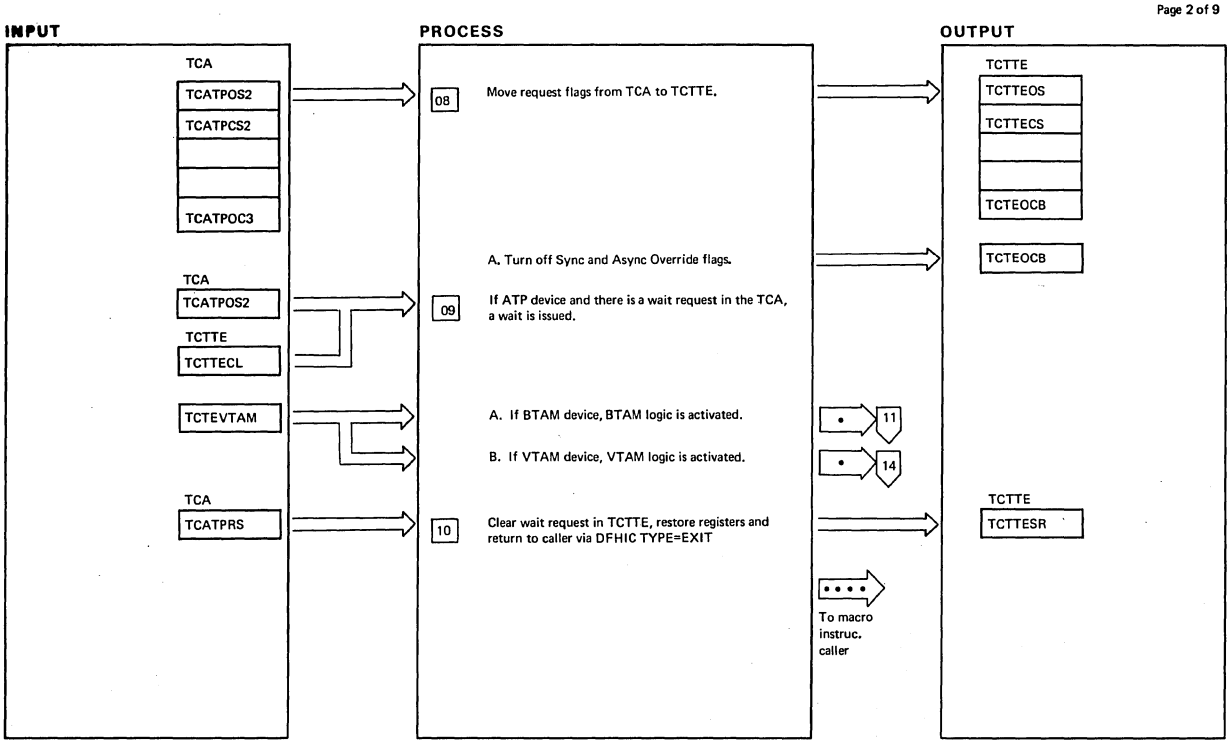
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 Prevent TCP from being abnormally ended as a runaway task. | | | |
| 04 DFHPC TYPE=ABEND, ABCODE= ATCA | | | |
| 05 For VTAM 3270, a Readbuf command must be sent to the lodgical unit. | | | |
| 06 VTAM PRINT requests require a READ BUF to be done unless the terminal is<br>A. A3650 Host convertsational 3275<br>B. A 3275 without the PRINT feature<br>C. A 3270 Compatibility Mode logical unit. | | | |
| 07 DFHPC TYPE=ABEND,ABCODE=ATCA | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Application Request

Diagram – 1.5.11.1.11–01

Page 2 of 9



08 Move request flags from TCA to TCTTE.

A. Turn off Sync and Async Override flags.

09 If ATP device and there is a wait request in the TCA, a wait is issued.

A. If BTAM device, BTAM logic is activated.

B. If VTAM device, VTAM logic is activated.

10 Clear wait request in TCTTE, restore registers and return to caller via DFHIC TYPE=EXIT

Application Request

Diagram — 1.5.11.1.11 — 02

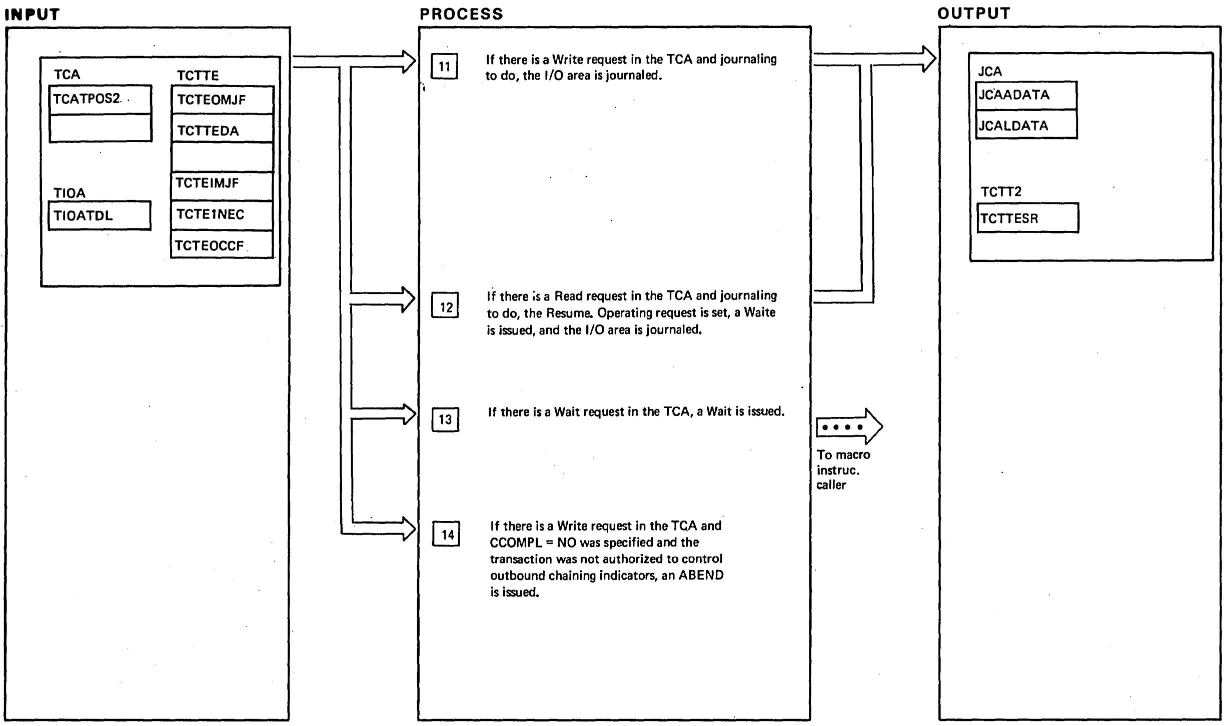
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 08 A. The Sync or Async override flags may have been set by the previous TC macro instruction. | | | |
| 09 DFHKC TYPE=WAIT, DCI=TERMINAL | | | |
| 10 | | TCZARQ92 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Application Request

Diagram — 1.5.11.1.11 — 02

Page 3 of 9



11 If there is a Write request in the TCA and journaling to do, the I/O area is journaled.

12 If there is a Read request in the TCA and journaling to do, the Resume. Operating request is set, a Waite is issued, and the I/O area is journaled.

13 If there is a Wait request in the TCA, a Wait is issued.

14 If there is a Write request in the TCA and CCOMPL = NO was specified and the transaction was not authorized to control outbound chaining indicators, an ABEND is issued.

Application Request

Diagram — 1.5.11.1.11 — 03

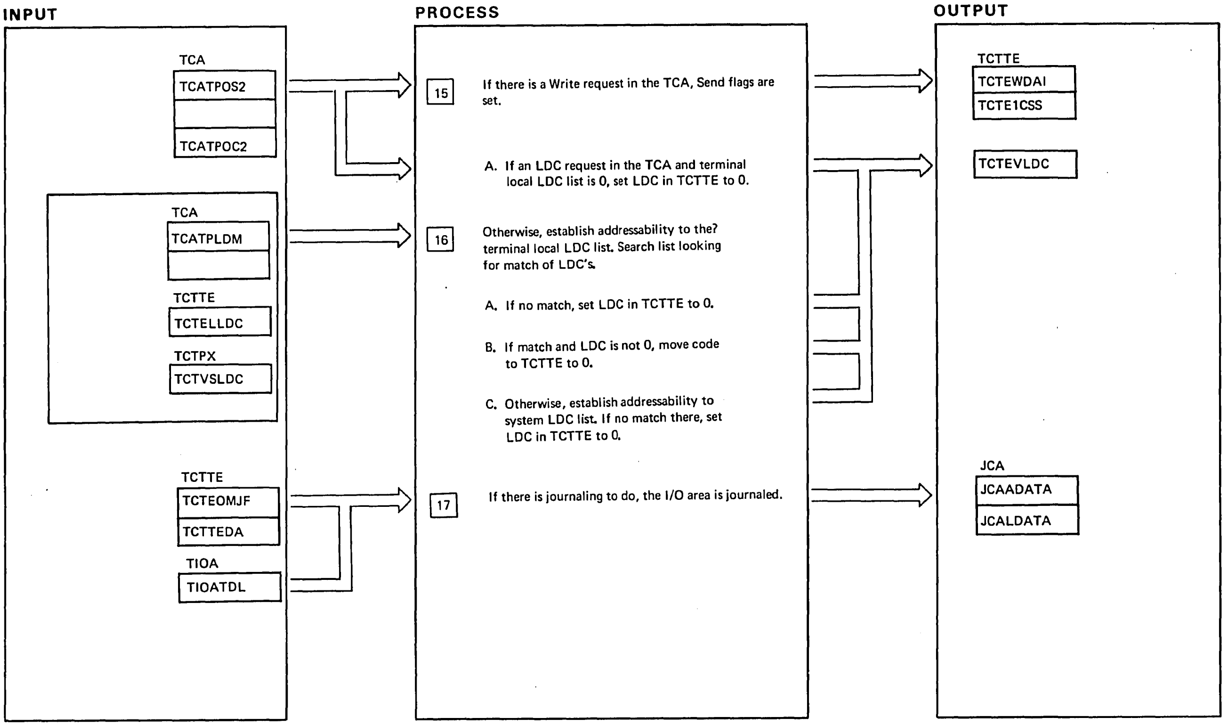
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 11 | TCZARQ20 | | |
| 12 | TCZARQ22 | | |
| 13 DFHKC TYPE=WAIT, DCI=TERMINAL<br>Return to macro instruction caller. | TCZARQ92 | | |
| 14 DFHPC TYPE = ABEND, ABCUDE = ATCG | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Application Request

Diagram — 1.5.11.1.11 — 03

Page 4 of 9



15 If there is a Write request in the TCA, Send flags are set.

A. If an LDC request in the TCA and terminal local LDC list is 0, set LDC in TCTTE to 0.

16 Otherwise, establish addressability to the? terminal local LDC list. Search list looking for match of LDC's.

A. If no match, set LDC in TCTTE to 0.

B. If match and LDC is not 0, move code to TCTTE to 0.

C. Otherwise, establish addressability to system LDC list. If no match there, set LDC in TCTTE to 0.

17 If there is journaling to do, the I/O area is journaled.

Application Request

Diagram – 1.5.11.1.11 – 04

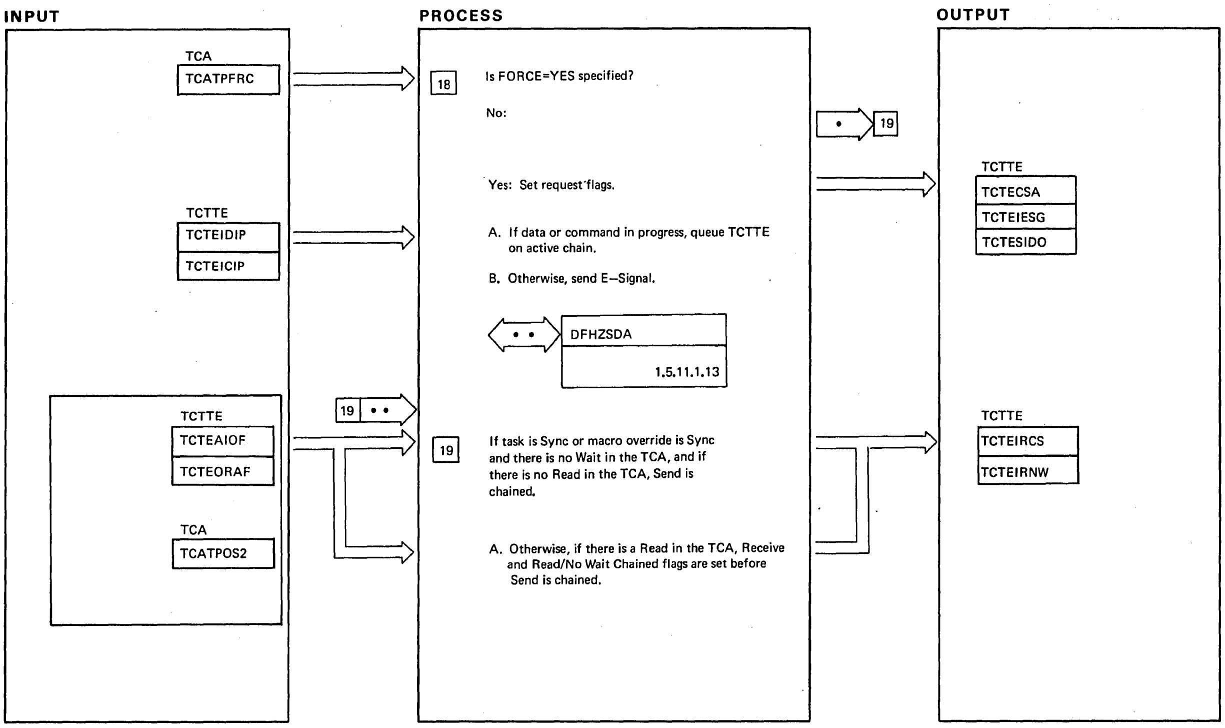
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 15 A. Process as normal Write. | | TCZARQ40 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Application Request

Diagram – 1.5.11.1.11 – 04

Page 5 of 9



18 Is FORCE=YES specified?

No:

Yes: Set request flags.

A. If data or command in progress, queue TCTTE on active chain.

B. Otherwise, send E–Signal.

19 If task is Sync or macro override is Sync and there is no Wait in the TCA, and if there is no Read in the TCA, Send is chained.

A. Otherwise, if there is a Read in the TCA, Receive and Read/No Wait Chained flags are set before Send is chained.

Application Request

Diagram – 1.5.11.1.11 – 05

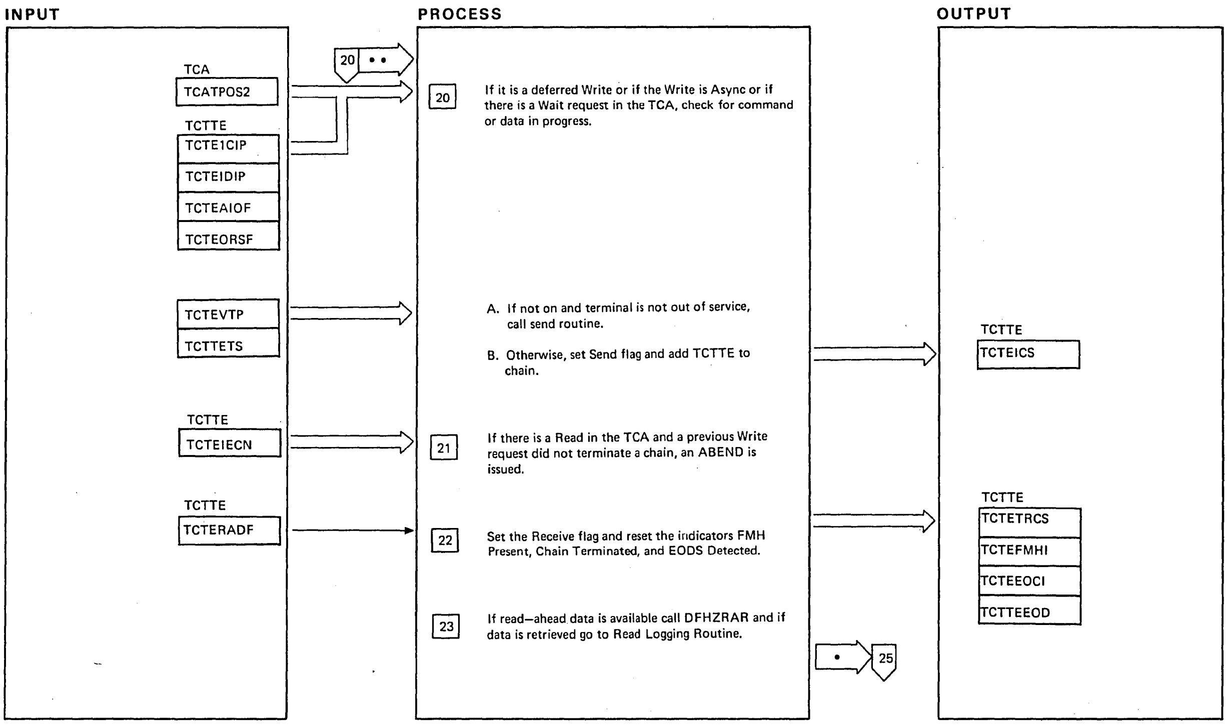
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 19 The TCTTE is added to the Activate chain before exciting this module. | | TCZARQ90 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Application Request

Diagram – 1.5.11.1.11 – 05

Page 6 of 9



20 If it is a deferred Write or if the Write is Async or if there is a Wait request in the TCA, check for command or data in progress.

A. If not on and terminal is not out of service, call send routine.

B. Otherwise, set Send flag and add TCTTE to chain.

21 If there is a Read in the TCA and a previous Write request did not terminate a chain, an ABEND is issued.

22 Set the Receive flag and reset the indicators FMH Present, Chain Terminated, and EODS Detected.

23 If read–ahead data is available call DFHZRAR and if data is retrieved go to Read Logging Routine.

Application Request

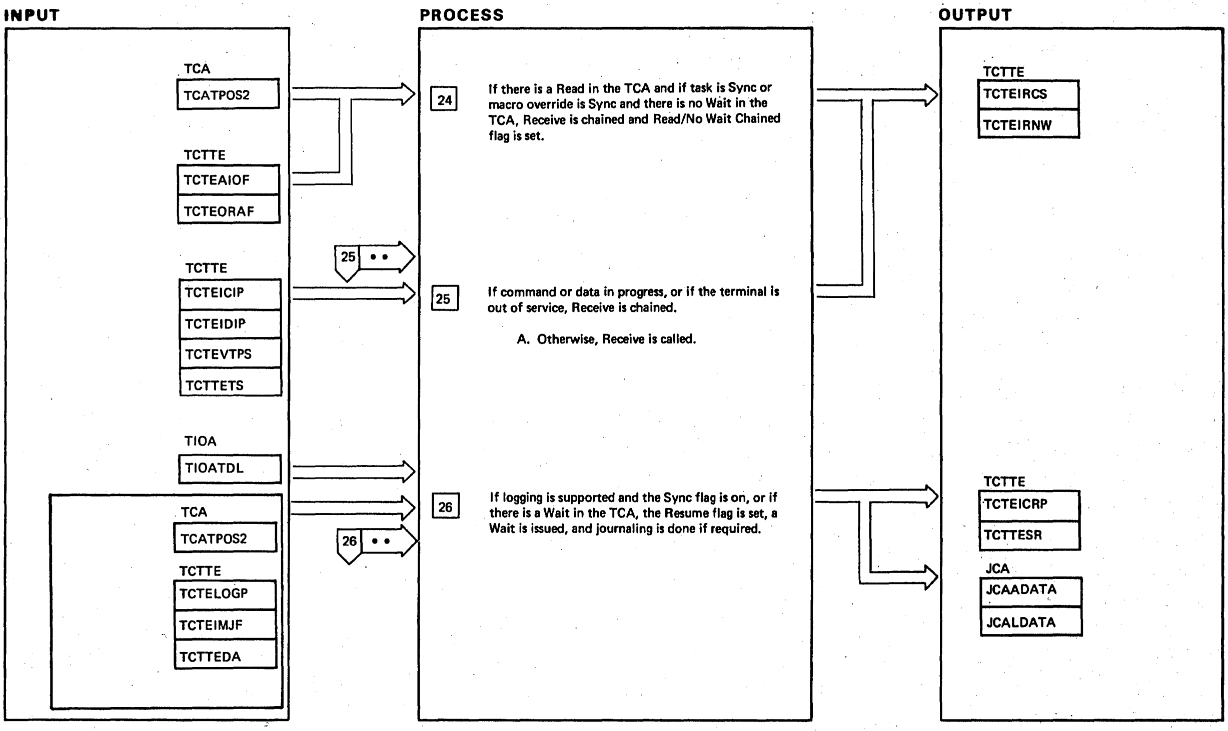
Diagram – 1.5.11.1.11 – 06

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 20 Send Sync data flow. | DFHZSDS | | 1.5.11.1.17 |
| 21 DFHPC TYPE=ABEND, ABCODE= ATCH | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Application Request

Diagram – 1.5.11.1.11 – 06



24 If there is a Read in the TCA and if task is Sync or macro override is Sync and there is no Wait in the TCA, Receive is chained and Read/No Wait Chained flag is set.

25 If command or data in progress, or if the terminal is out of service, Receive is chained.

A. Otherwise, Receive is called.

26 If logging is supported and the Sync flag is on, or if there is a Wait in the TCA, the Resume flag is set, a Wait is issued, and journaling is done if required.

Application Request

Diagram — 1.5.11.1.11 — 07

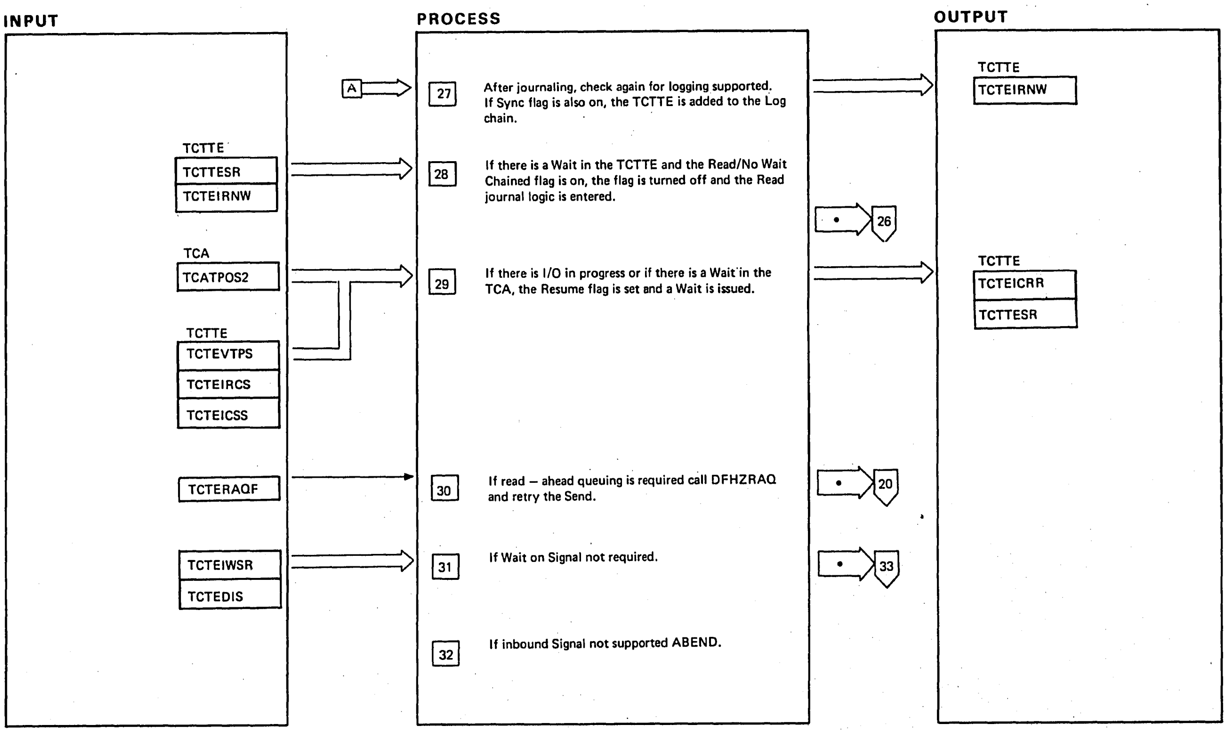
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 24 TCTTE is added to Activate chain. | | TCZARQ90 | |
| 25 A. Receive Module. | DFHZRVS | 1.5.11.1.14 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Application Request

Diagram — 1.5.11.1.11 — 07

Page 8 of 9



27 After journaling, check again for logging supported. If Sync flag is also on, the TCTTE is added to the Log chain.

28 If there is a Wait in the TCTTE and the Read/No Wait Chained flag is on, the flag is turned off and the Read journal logic is entered.

29 If there is I/O in progress or if there is a Wait in the TCA, the Resume flag is set and a Wait is issued.

30 If read — ahead queuing is required call DFHZRAQ and retry the Send.

31 If Wait on Signal not required.

32 If inbound Signal not supported ABEND.

Application Request

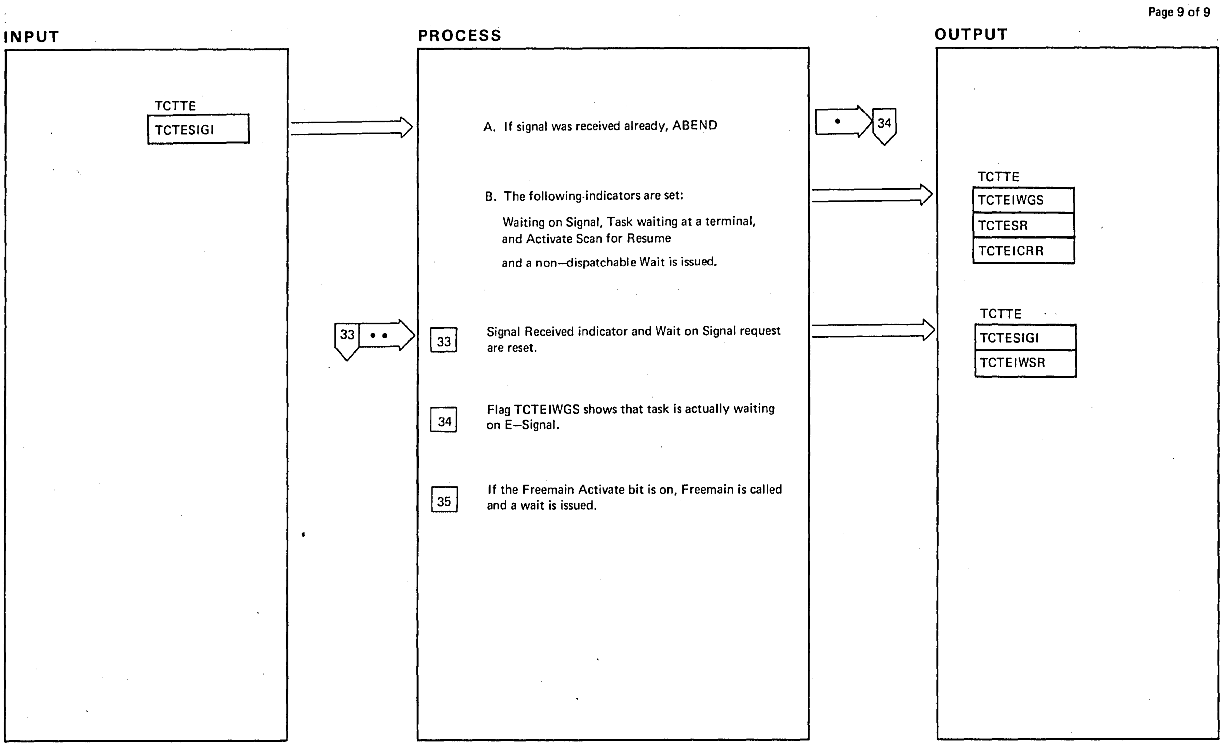
Diagram — 1.5.11.1.11 — 08

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 27 DFHTCQUE QUE=LOG | | | |
| 28 | | TCZARQ60 | |
| 29 DFHKC TYPE=WAIT, DCI=TERMINAL | | TCZARQ78 | |
| 30 Read — ahead queueing will be required if the Send failed because the receiver is in transmit status. DFHZSDS will set bit TCTERAQ to indicate this. | | TCZARQ8A | |
| 32 DFHPC TYPE=ABEND, ABCODE= ATCO | | TCZARQ8C | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Application Request

Diagram — 1.5.11.1.11 — 08



A. If signal was received already, ABEND

B. The following indicators are set:

Waiting on Signal, Task waiting at a terminal, and Activate Scan for Resume

and a non–dispatchable Wait is issued.

33 Signal Received indicator and Wait on Signal request are reset.

34 Flag TCTEIWGS shows that task is actually waiting on E–Signal.

35 If the Freemain Activate bit is on, Freemain is called and a wait is issued.

Application Request

Diagram – 1.5.11.1.11 – 09

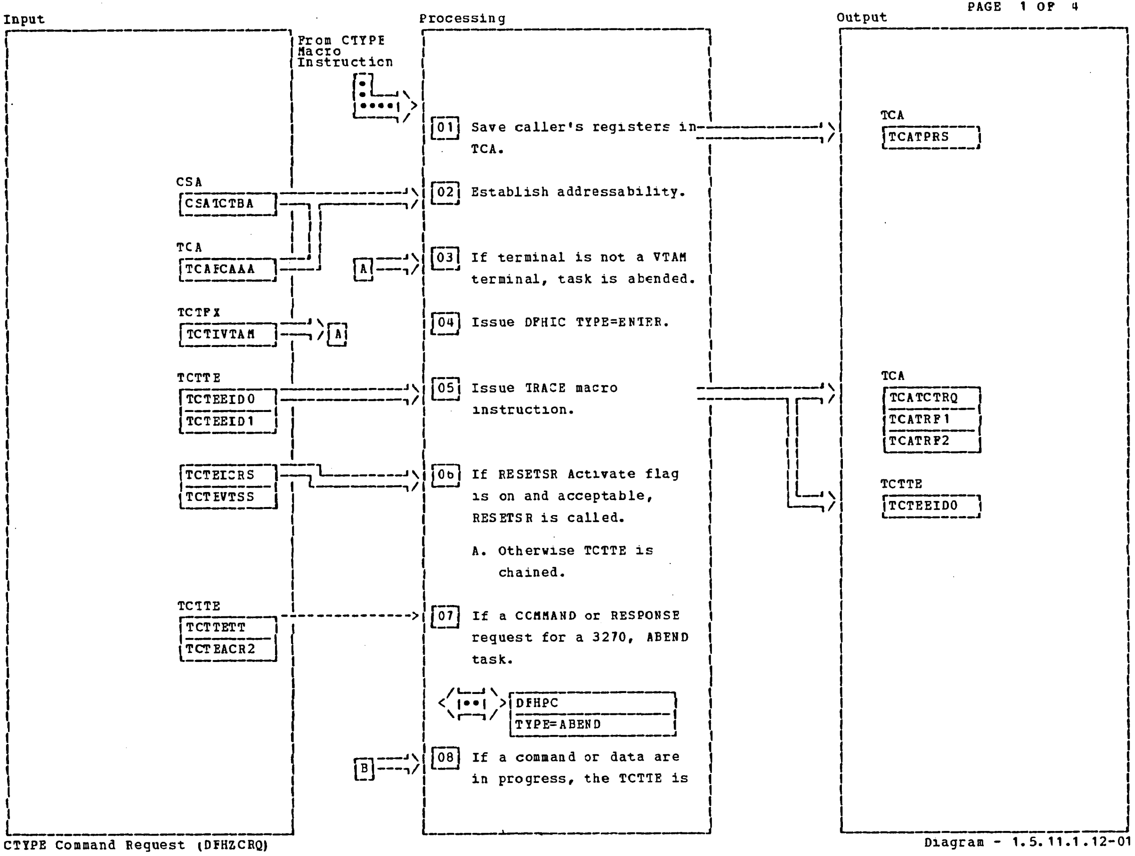
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| A. DFHPC TYPE=ABEND, ABCODE= ATCO | | TCZARQ8C | |
| B. DFHKC TYPE=WAIT, DCI= TERMINAL | | | |
| 33 FREEMAIN | DFHZFRE | | |
| DFHKC TYPE = WAIT, DCI = TERMINAL | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Application Request

Diagram – 1.5.11.1.11 – 09

CHART: HIPO0001
PAGE 1 OF 4



01 Save caller's registers in TCA.

02 Establish addressability.

03 If terminal is not a VTAM terminal, task is abended.

04 Issue DFHIC TYPE=ENTER.

05 Issue TRACE macro instruction.

06 If RESETSR Activate flag is on and acceptable, RESETSR is called.

A. Otherwise TCTTE is chained.

07 If a COMMAND or RESPONSE request for a 3270, ABEND task.

08 If a command or data are in progress, the TCTTE is

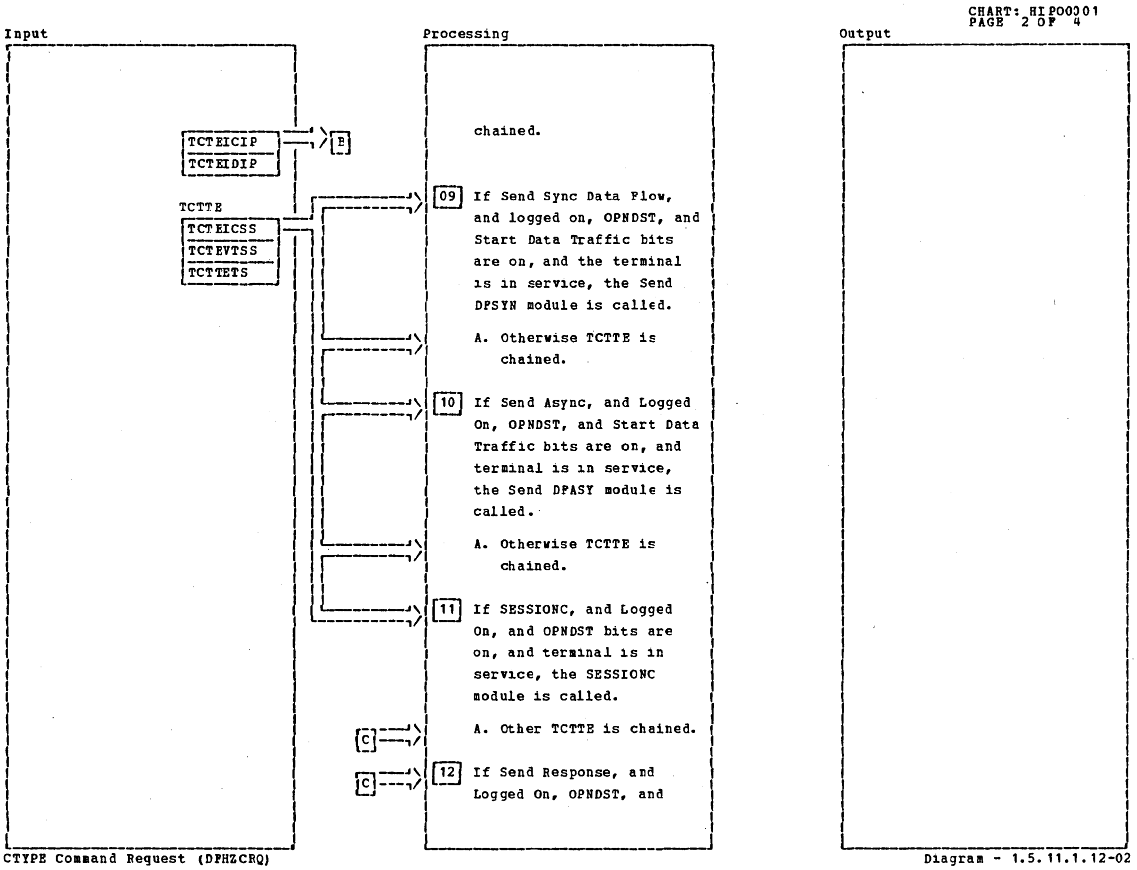
CTYPE Command Request (DFHZCRQ)

Diagram - 1.5.11.1.12-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 DFHPC TYPE=ABEND ABCODE=ATCP | | | | | | | |
| 04 Prevent TCP from being abnormally ended as a runaway task. | | | | | | | |
| 06 RESETSR | DFHZRST | | 1.5.11<br>.1.21 | | | | |
| 07 ABCODE=ATCP | | TCZCRQ10 | | | | | |

CTYPE Command Request (DFHZCRQ)

Diagram - 1.5.11.1.12-01



chained.

09 If Send Sync Data Flow, and logged on, OPNDST, and Start Data Traffic bits are on, and the terminal is in service, the Send DFSYN module is called.

A. Otherwise TCTTE is chained.

10 If Send Async, and Logged On, OPNDST, and Start Data Traffic bits are on, and terminal is in service, the Send DFASY module is called.

A. Otherwise TCTTE is chained.

11 If SESSIONC, and Logged On, and OPNDST bits are on, and terminal is in service, the SESSIONC module is called.

A. Other TCTTE is chained.

12 If Send Response, and Logged On, OPNDST, and

CTYPE Command Request (DFHZCRQ)

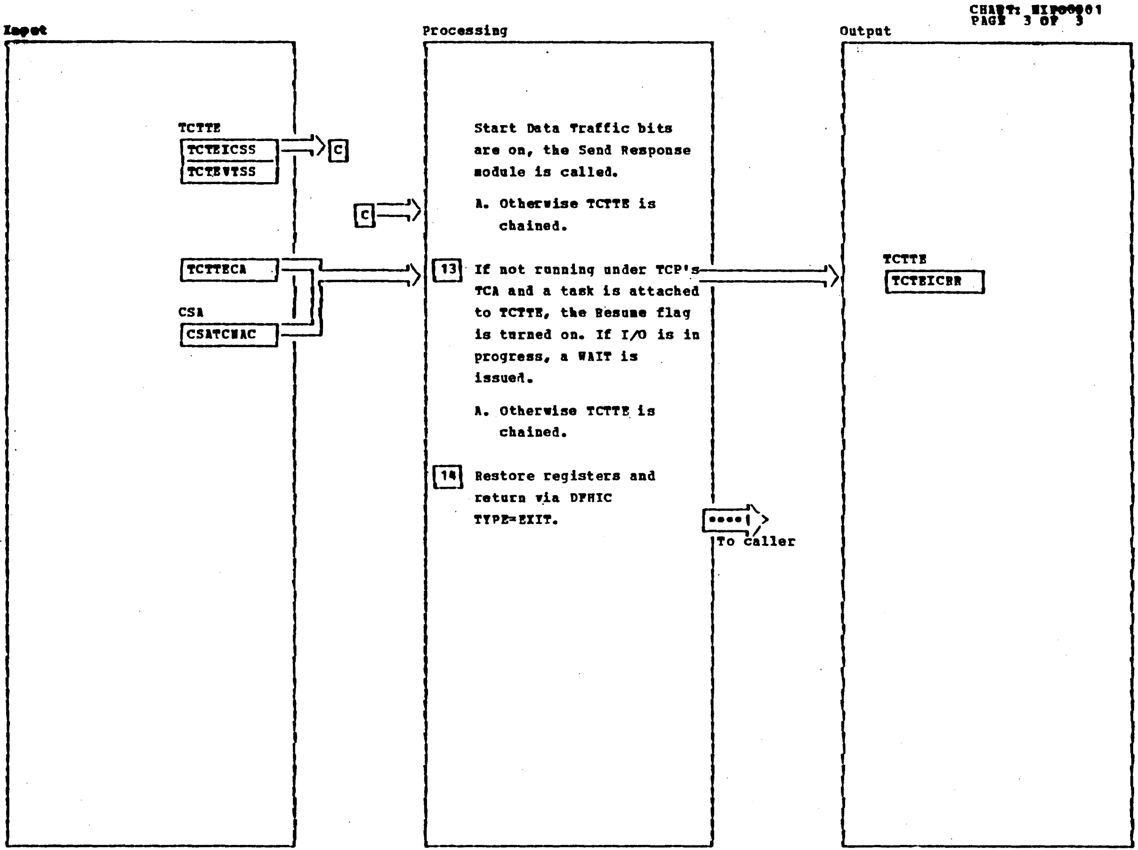
Diagram - 1.5.11.1.12-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 09 Send Sync DFSYN | DFHZSDS | | 1.5.11 .1.17 | | | | |
| A. TCTTE chained. | | TCZCRQ90 | | | | | |
| 10 Send Async | DFHZSDA | | | | | | |
| A. TCTTE chained. | | TCZCRQ90 | 1.5.11 .1.13 | | | | |
| 11 SESSIONC | DFHZSES | | 1.5.11 .1.8 | | | | |
| A. TCTTE chained. | | TCZCRQ90 | | | | | |

CTYPE Command Request (DFHZCRQ)

Diagram - 1.5.11.1.12-02

CHART: HIP00001
PAGE 3 OF 3



Start Data Traffic bits are on, the Send Response module is called.

A. Otherwise TCTTE is chained.

13 If not running under TCP's TCA and a task is attached to TCTTE, the Resume flag is turned on. If I/O is in progress, a WAIT is issued.

A. Otherwise TCTTE is chained.

14 Restore registers and return via DFHIC TYPE=EXIT.

CTYPE Command Request (DFHZCRQ)

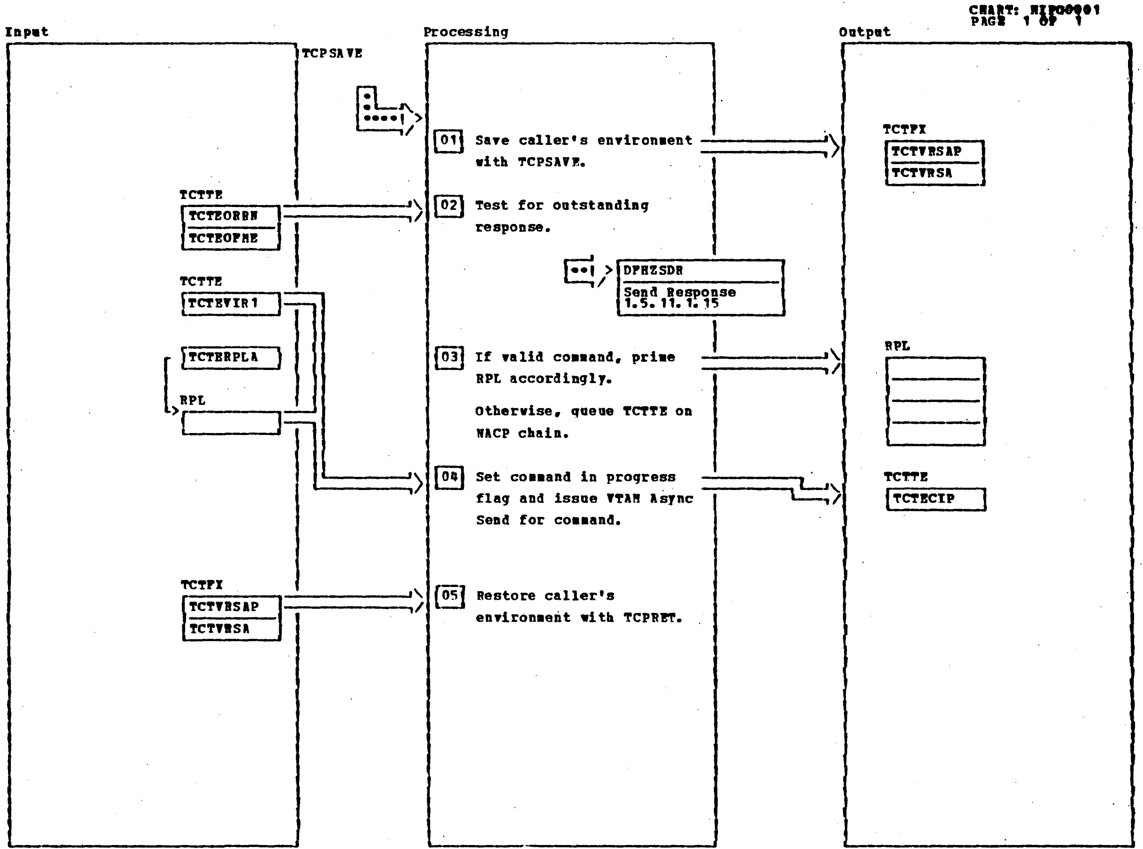
Diagram - 1.5.11.1.12-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 12 Send Response | DFHZSDR | | 1.5.11.1.15 | | | | |
| A. TCTTE chained. | | TCZCAQ90 | | | | | |
| 13 DFHKC TYPE=WAIT is only issued after a module has been called. | | | | | | | |

CTYPE Command Request (DFHZCRQ)

Diagram - 1.5.11.1.12-03

CHART: HIPO0001
PAGE 1 OF 1



01 Save caller's environment with TCPSAVE.

02 Test for outstanding response.

03 If valid command, prime RPL accordingly.

Otherwise, queue TCTTE on WACP chain.

04 Set command in progress flag and issue VTAM Async Send for command.

05 Restore caller's environment with TCPRET.

Send Asynchronous Command (DFHZSDA)

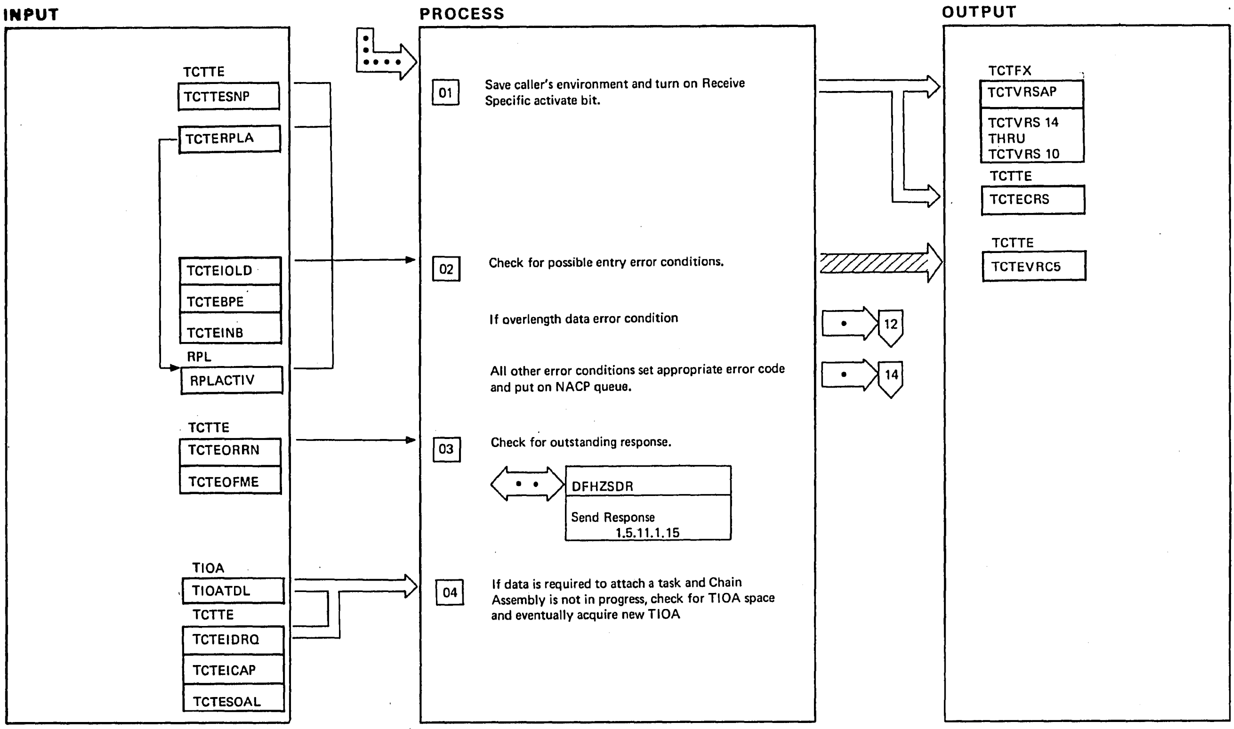
Diagram - 1.5.11.1.13-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 Event completion will be checked when the list is scheduled. The requestor of this routine is responsible for issuing any task control Wait or Asynchronous Resume request that may be required. | | | | | | | |

Send Asynchronous Command (DFHZSDA)

Diagram - 1.5.11.1.13-01

Page 1 of 4



01 Save caller's environment and turn on Receive Specific activate bit.

02 Check for possible entry error conditions.

If overlength data error condition

All other error conditions set appropriate error code and put on NACP queue.

03 Check for outstanding response.

04 If data is required to attach a task and Chain Assembly is not in progress, check for TIOA space and eventually acquire new TIOA

Receive Specific (DFHZRVS)

Diagram - 1.5.11.1.14-01

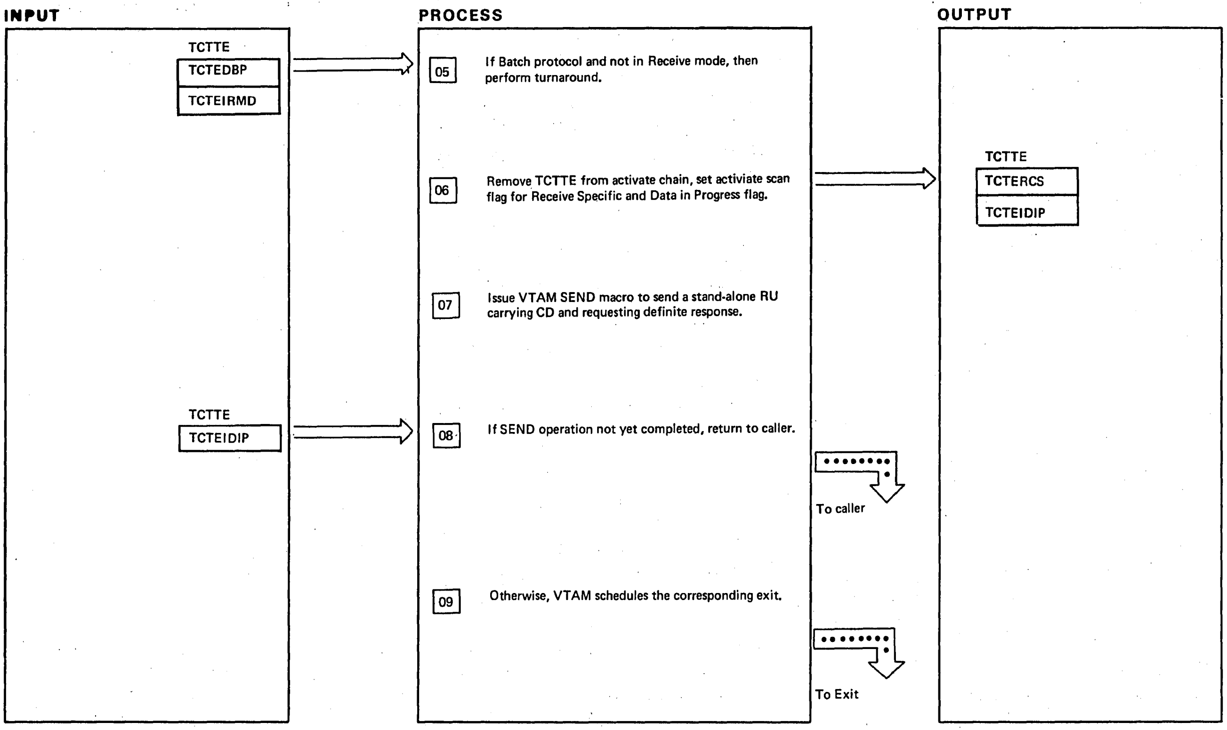
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 Error conditions checked for are: | | | |
| NIB disabled | | TCZRVSQS | |
| CLSDST scheduled | | TCZRVSQS | |
| RPL missing | | TCZRVSQA | |
| RPL busy | | TCZRVSQC | |
| Overlength data | | TCZRVSD2 | |
| Read to TCTTE in Receive status | | TCZRVSQE | |
| Task bracket protocol violation | | TCZRVSQG | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Receive Specific (DFHZRVS)

Diagram - 1.5.11.1.14-01

Page 2 of 4



05 If Batch protocol and not in Receive mode, then perform turnaround.

06 Remove TCTTE from activate chain, set activiate scan flag for Receive Specific and Data in Progress flag.

07 Issue VTAM SEND macro to send a stand-alone RU carrying CD and requesting definite response.

08 If SEND operation not yet completed, return to caller.

09 Otherwise, VTAM schedules the corresponding exit.

Receive Specific (DFHZRVS)

Diagram - 1.5.11.1. 14-02

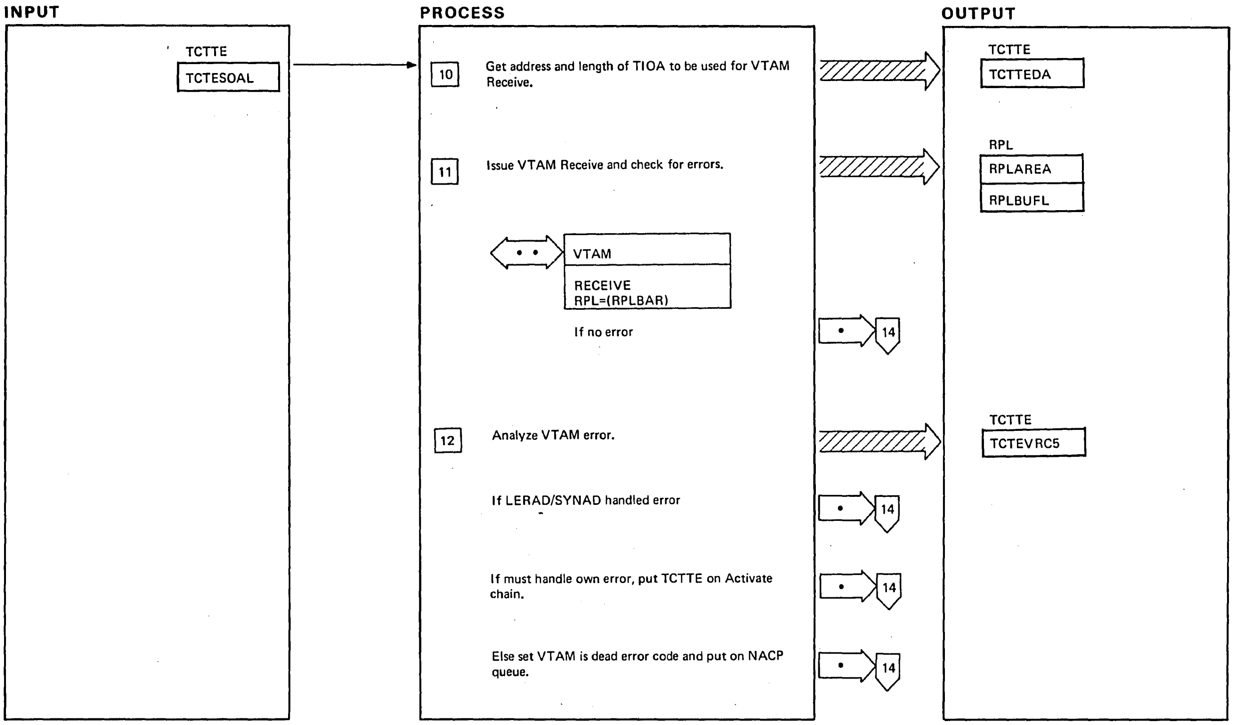
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 05 Turnaround is performed by sending a stand-alone RU in definite response mode carrying CD. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Receive Specific (DFHZRVS)

Diagram - 1.5.11.1. 14-02

Page 3 of 4



INPUT

TCTTE
TCTESOAL

PROCESS

10 Get address and length of TIOA to be used for VTAM Receive.

11 Issue VTAM Receive and check for errors.

VTAM
RECEIVE RPL=(RPLBAR)

If no error → 14

12 Analyze VTAM error.

If LERAD/SYNAD handled error → 14

If must handle own error, put TCTTE on Activate chain. → 14

Else set VTAM is dead error code and put on NACP queue. → 14

OUTPUT

TCTTE
TCTTEDA

RPL
RPLAREA
RPLBUFL

TCTTE
TCTEVRC5

Receive Specific (DFHZRVS)

Diagram - 1.5.11.1. 14-03

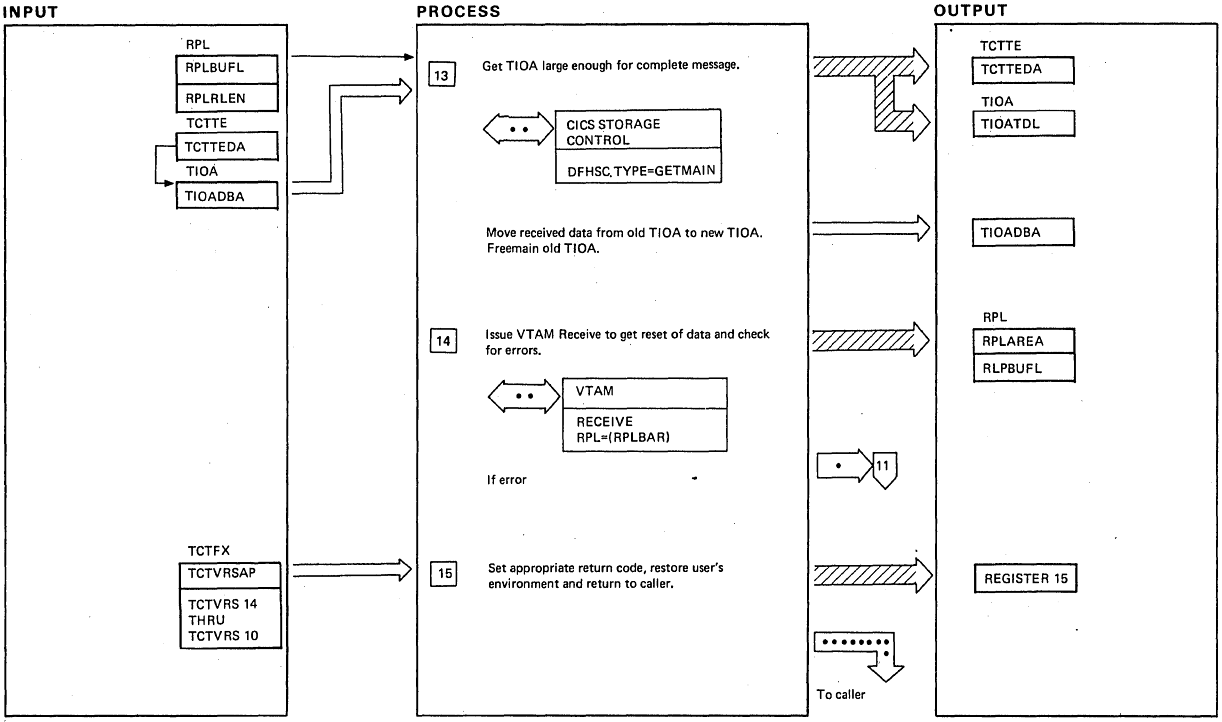
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 11 If authorized path SRB support is being used, this invokes the RPL service routine to interface to VTAM. | DFHZHPSR | | 1.5.11.1.37 |
| 12 Return code in register 15 after Receive macro instruction has following meanings: | TCZRVSQ2 | | |
| R15 = 0 – No abnormal condition | TCZRVSP2 | | |
| R15 = –1 – Error handled by LERAD/SYNAD routines | TCZRVSQ8 | | |
| R15 = X'F0' – An error occurred that the module must handle | TCZRVSQ2 | | |
| R15 = X'20' – VTAM has terminated abnormally | TCZRVSQ4 | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Receive Specific (DFHZRVS)

Diagram - 1.5.11.1.14 - 03

Page 4 of 4



13 Get TIOA large enough for complete message.

Move received data from old TIOA to new TIOA. Freemain old TIOA.

14 Issue VTAM Receive to get reset of data and check for errors.

If error

15 Set appropriate return code, restore user's environment and return to caller.

Receive Specific (DFHZRVS)

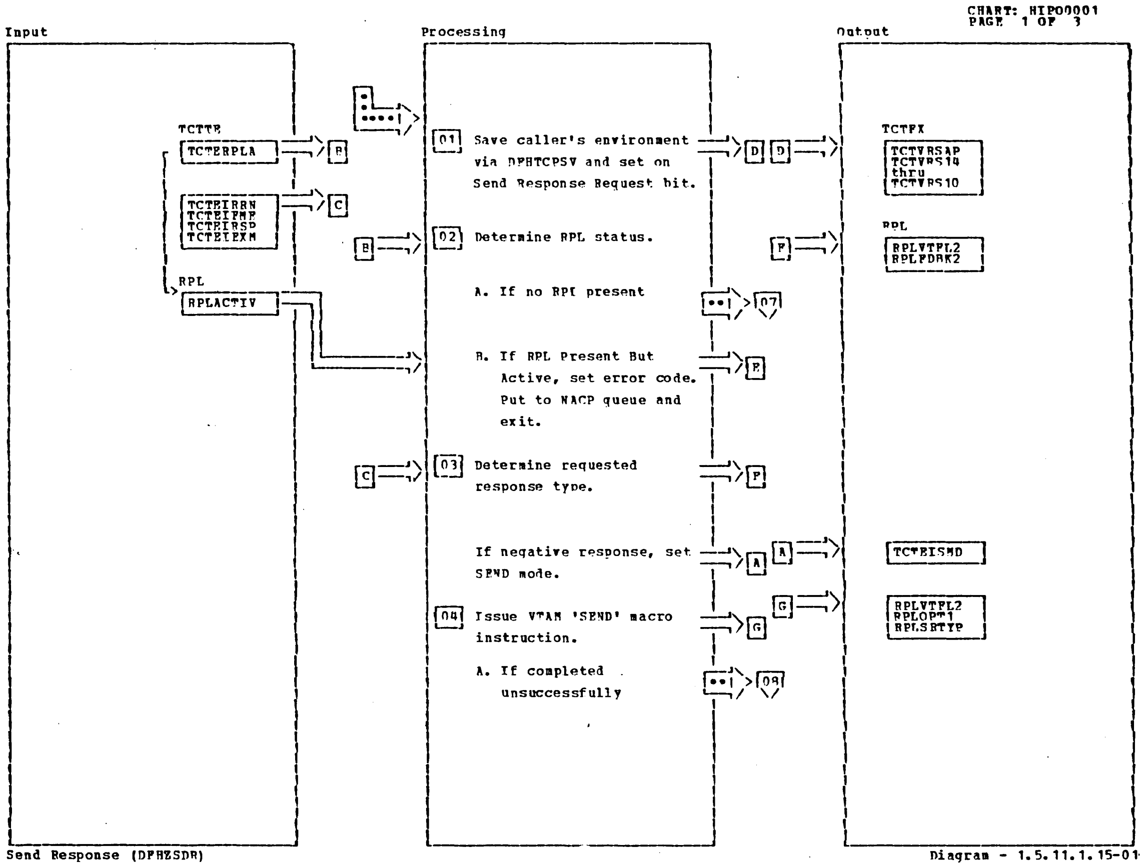
Diagram - 1.5.11.1.14 - 04

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Receive Specific (DFHZRVS)

Diagram - 1.5.11.1.14 - 04



01 Save caller's environment via DFHTCPSV and set on Send Response Request bit.

02 Determine RPL status.

A. If no RPL present

B. If RPL Present But Active, set error code. Put to NACP queue and exit.

03 Determine requested response type.

If negative response, set SEND mode.

04 Issue VTAM 'SEND' macro instruction.

A. If completed unsuccessfully

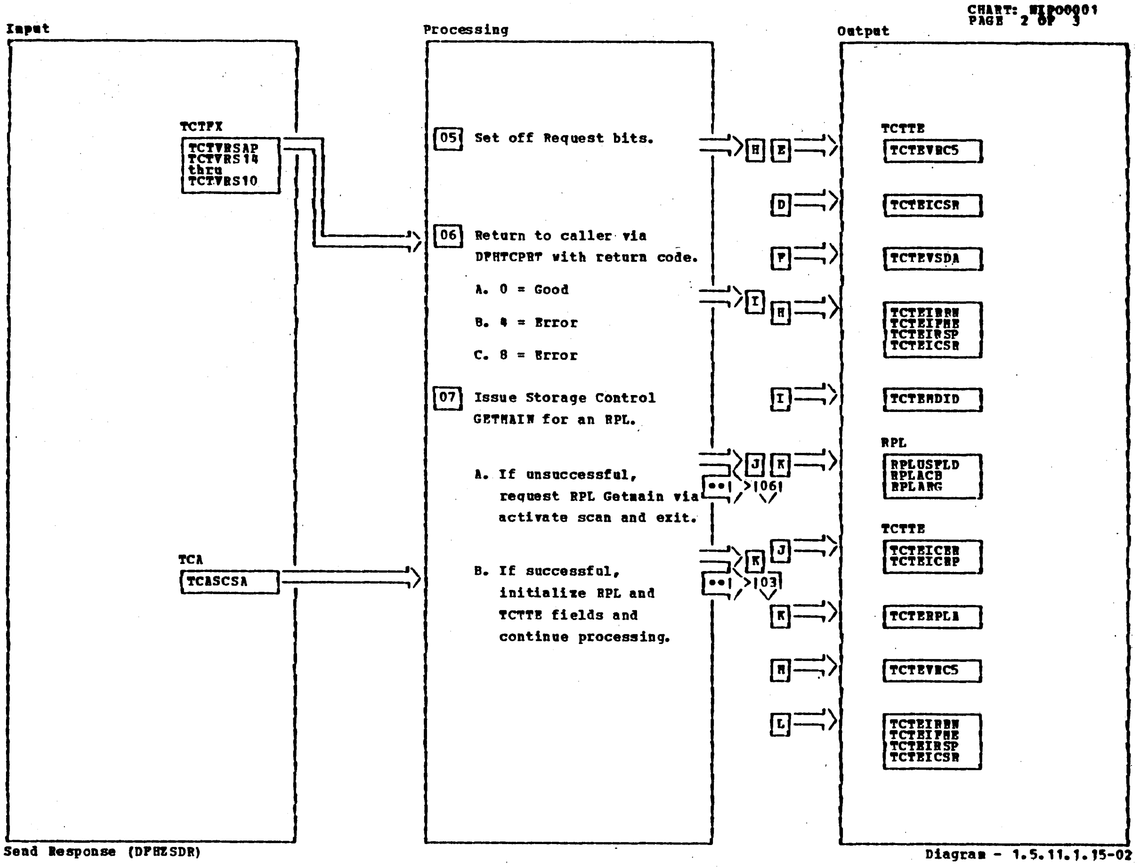
Send Response (DFHZSDR) Diagram - 1.5.11.1.15-01

| Notes | Routine | Label | Ref |
|---|---|---|---|

| Notes | Routine | Label | Ref |
|---|---|---|---|

Send Response (DFHZSDR) Diagram - 1.5.11.1.15-01

CHART: NIP00001
PAGE 2 OF 3



05 Set off Request bits.

06 Return to caller via DFHTCPRT with return code.

A. 0 = Good

B. 4 = Error

C. 8 = Error

07 Issue Storage Control GETMAIN for an RPL.

A. If unsuccessful, request RPL Getmain via activate scan and exit.

B. If successful, initialize RPL and TCTTE fields and continue processing.

Send Response (DFHZSDR)

Diagram - 1.5.11.1.15-02

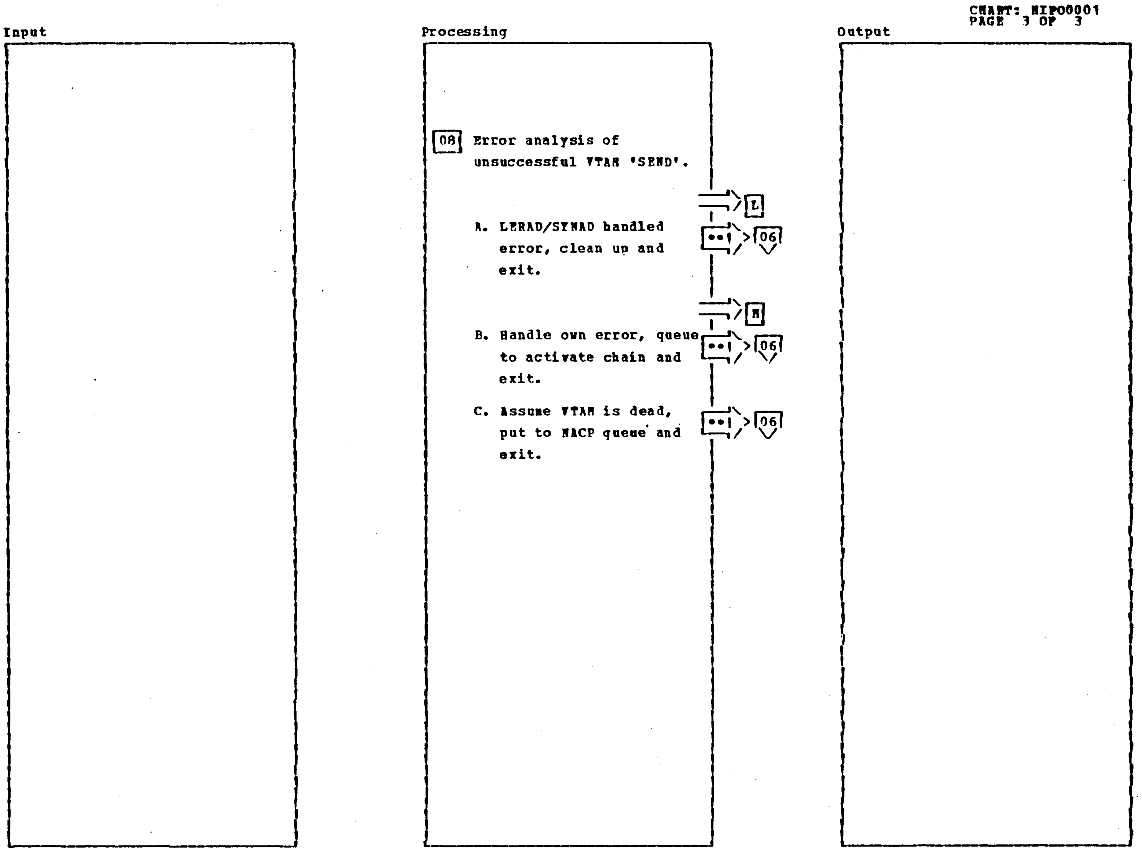
| Notes | Routine | Label | Ref |
|---|---|---|---|

| Notes | Routine | Label | Ref |
|---|---|---|---|

Send Response (DFHZSDR)

Diagram - 1.5.11.1.15-02

CHART: HIPO0001
PAGE 3 OF 3



08 Error analysis of unsuccessful VTAM 'SEND'.

A. LERAD/SYNAD handled error, clean up and exit.

B. Handle own error, queue to activate chain and exit.

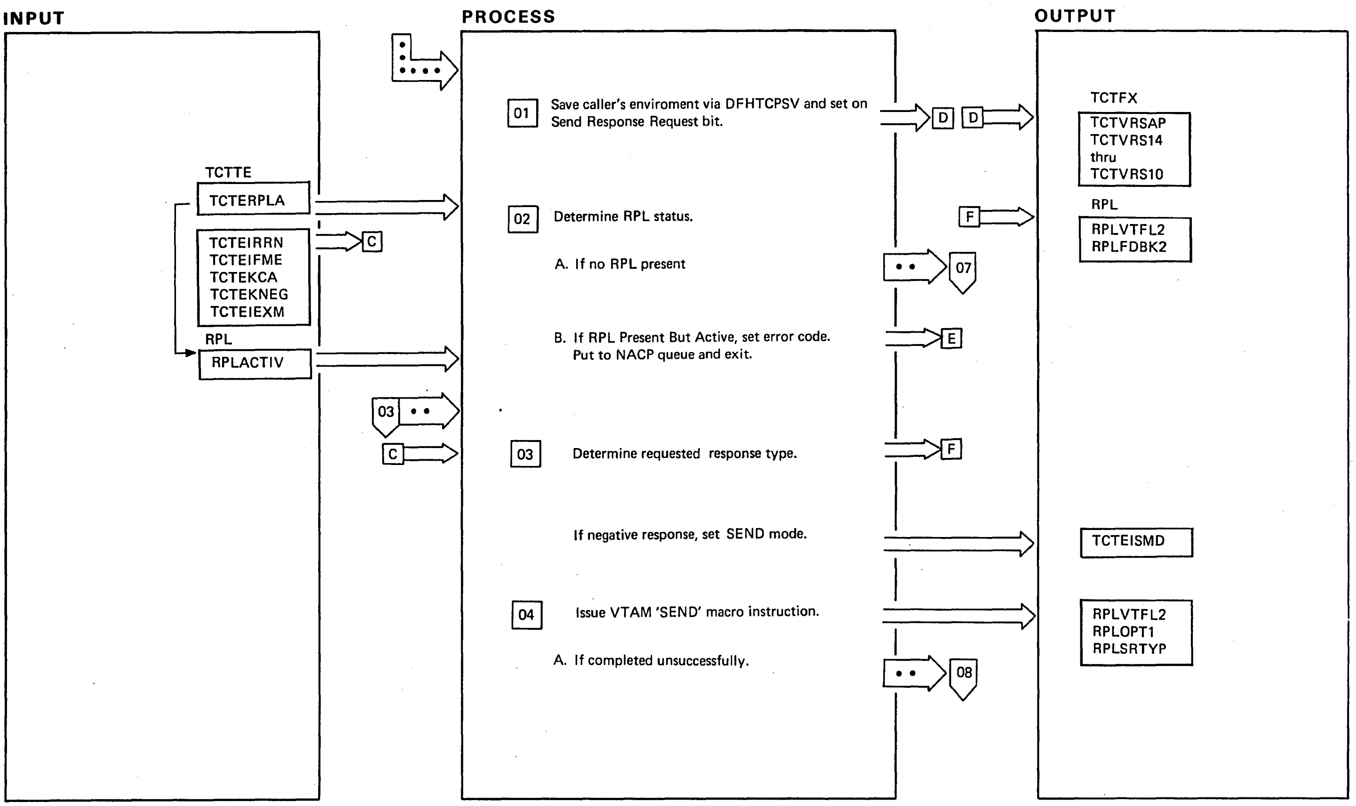
C. Assume VTAM is dead, put to NACP queue and exit.

Send Response (DFHZSDR)  Diagram - 1.5.11.1.15-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 08 VTAM return code (in register 15) will be 'FF', 'F0', or '20' meaning respectively:<br>A. Error condition has already been handled by LERAD/SYNAD.<br>B. Handle your own error.<br>C. VTAM is dead. | | | | | | | |

Send Response (DFHZSDR)  Diagram - 1.5.11.1.15-03

PAGE 1 OF 3



01 Save caller's enviroment via DFHTCPSV and set on Send Response Request bit.

02 Determine RPL status.

A. If no RPL present

B. If RPL Present But Active, set error code. Put to NACP queue and exit.

03 Determine requested response type.

If negative response, set SEND mode.

04 Issue VTAM 'SEND' macro instruction.

A. If completed unsuccessfully.

Command Response (DFHZSKR)

Diagram - 1.5.11.1.15A-01

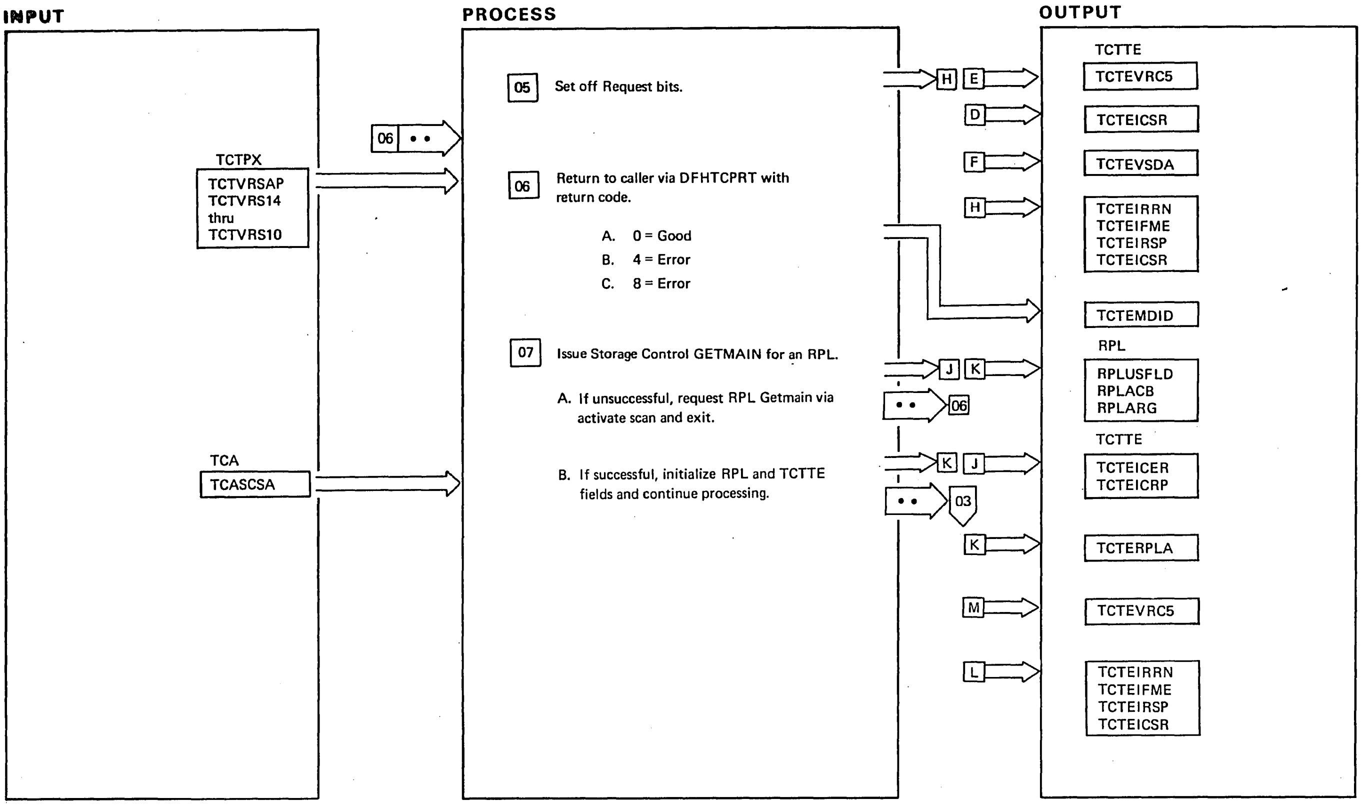
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Command Response (DFHZSKR)

,Diagram - 1.5.11.1.15A-01

PAGE 2 OF 3



05 Set off Request bits.

06 Return to caller via DFHTCPRT with return code.

A. 0 = Good
B. 4 = Error
C. 8 = Error

07 Issue Storage Control GETMAIN for an RPL.

A. If unsuccessful, request RPL Getmain via activate scan and exit.

B. If successful, initialize RPL and TCTTE fields and continue processing.

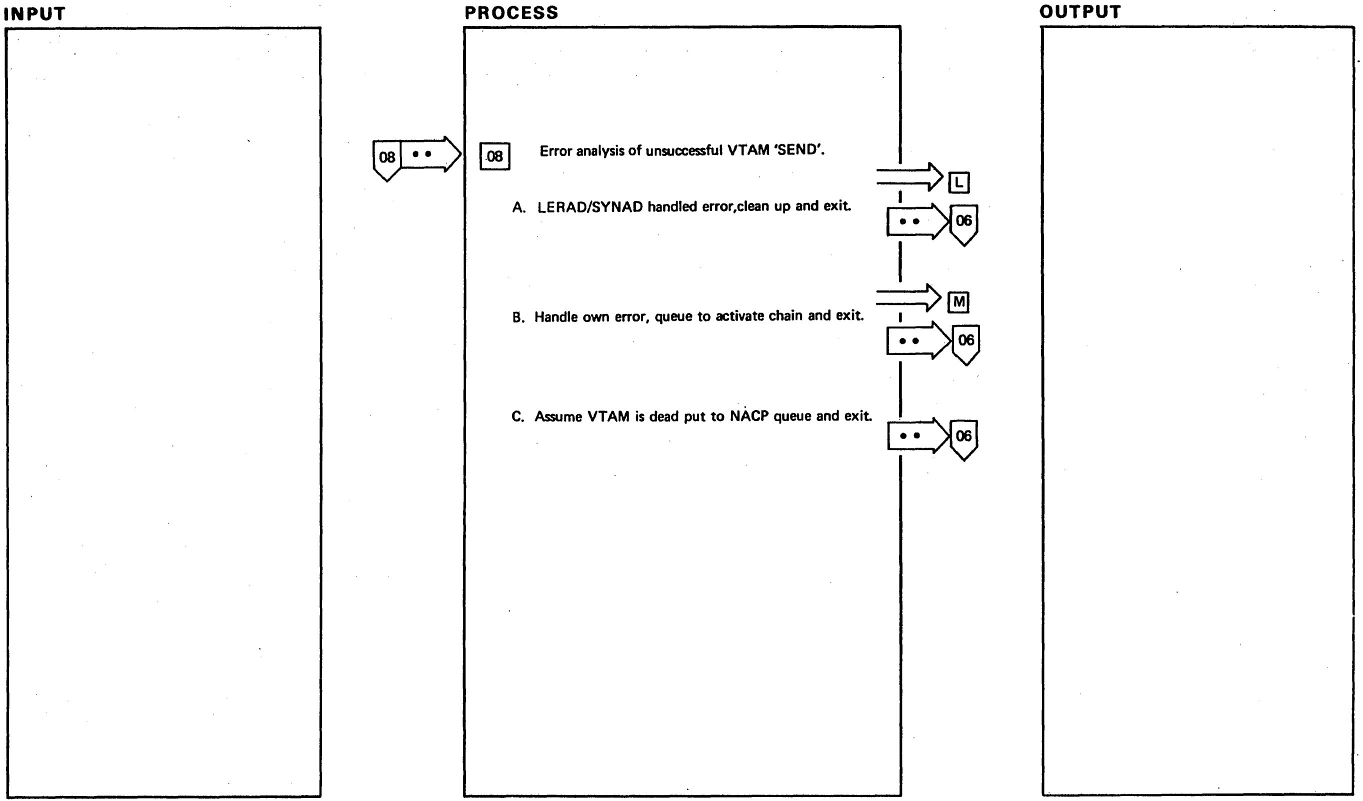
Command Response (DFHZSKR)

Diagram - 1.5.11.1.15A-02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

PAGE 3 OF 3



08 Error analysis of unsuccessful VTAM 'SEND'.

A. LERAD/SYNAD handled error,clean up and exit.

B. Handle own error, queue to activate chain and exit.

C. Assume VTAM is dead put to NACP queue and exit.

Command Response (DFHZSKR)

Diagram - 1.5.11.1.15A-03

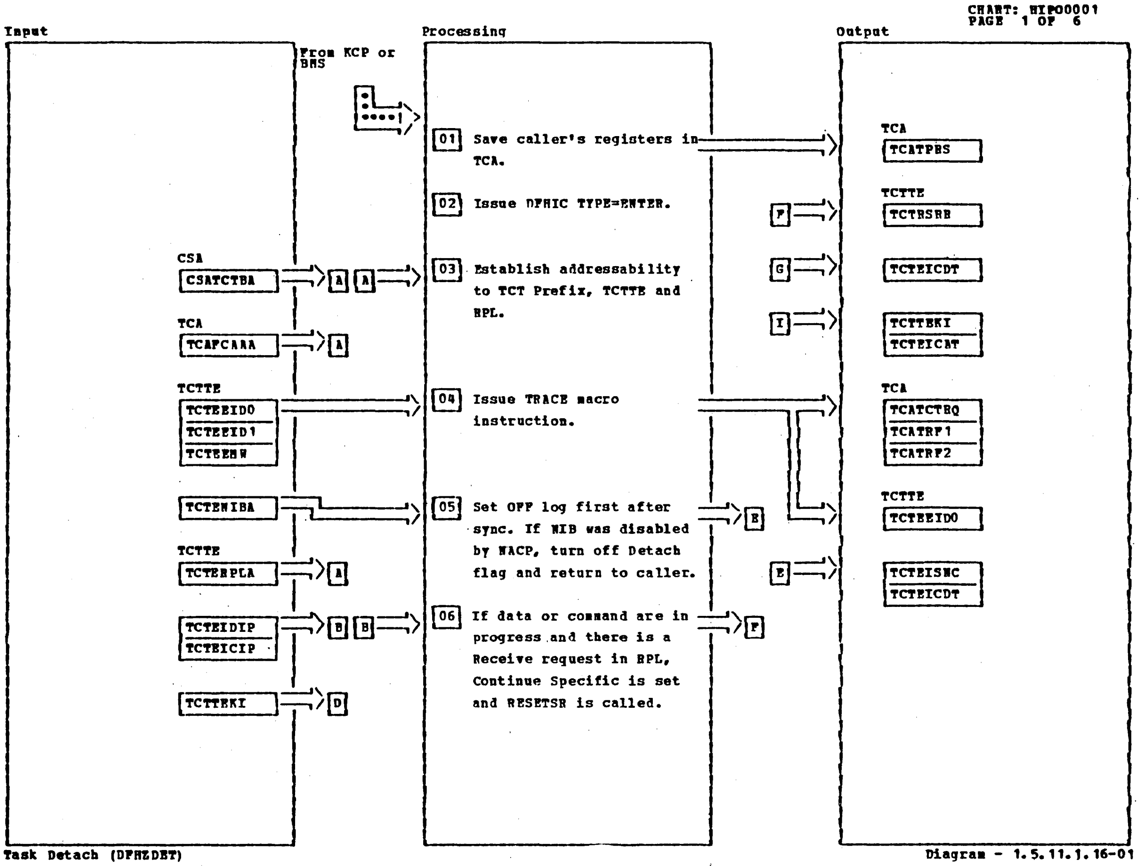
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 08 VTAM return code (in register 15) will be 'FF', 'F0', 20 meaning respectively:<br>A. Error condition has already been handled by LERAD/SYNAD.<br>B. Handle your own error.<br>C. VTAM is dead. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Command Response (DFHZSKR)

Diagram - 1.5.11.1.15A-03

CHART: HIP00001
PAGE 1 OF 6



01 Save caller's registers in TCA.

02 Issue DFHIC TYPE=ENTER.

03 Establish addressability to TCT Prefix, TCTTE and RPL.

04 Issue TRACE macro instruction.

05 Set OFF log first after sync. If NIB was disabled by NACP, turn off Detach flag and return to caller.

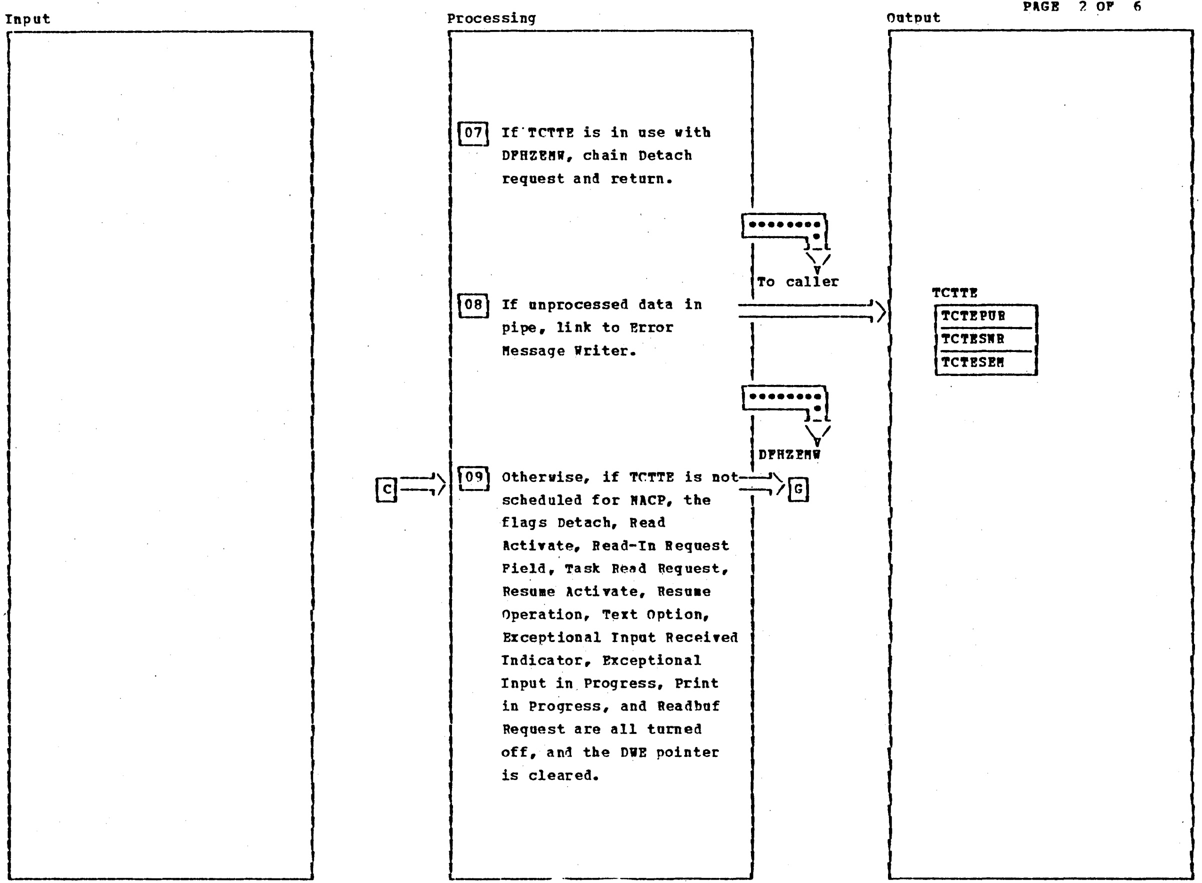
06 If data or command are in progress and there is a Receive request in RPL, Continue Specific is set and RESETSR is called.

Task Detach (DFHZDET) Diagram - 1.5.11.1.16-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 Prevent TCP from being abnormally ended as a runaway task. | | | | | | | |
| 06 RESETSR module. | DFHZRST | | 1.5.11.1.21 | | | | |
| The TCTTE will be chained to redrive detach. | | | | | | | |

Task Detach (DFHZDET) Diagram - 1.5.11.1.16-01

CHART: HIPO0001
PAGE 2 OF 6



07 If TCTTE is in use with DFHZEMW, chain Detach request and return.

08 If unprocessed data in pipe, link to Error Message Writer.

09 Otherwise, if TCTTE is not scheduled for NACP, the flags Detach, Read Activate, Read-In Request Field, Task Read Request, Resume Activate, Resume Operation, Text Option, Exceptional Input Received Indicator, Exceptional Input in Progress, Print in Progress, and Readbuf Request are all turned off, and the DWE pointer is cleared.

Task Detach (DFHZDET)

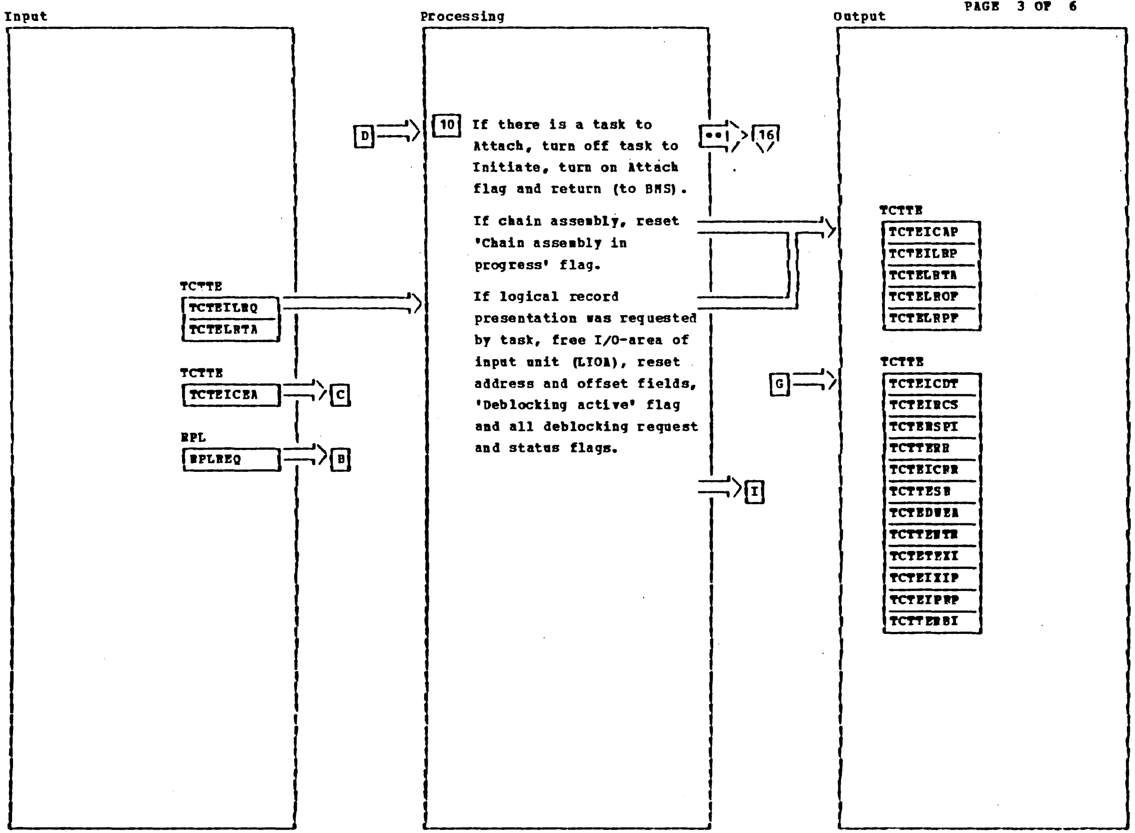
Diagram - 1.5.11.1.16-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

Task Detach (DFHZDET)

Diagram - 1.5.11.1.16-02

CHART: HIPO0001
PAGE 3 OF 6



10 If there is a task to Attach, turn off task to Initiate, turn on Attach flag and return (to BMS).

If chain assembly, reset 'Chain assembly in progress' flag.

If logical record presentation was requested by task, free I/O-area of input unit (LIOA), reset address and offset fields, 'Deblocking active' flag and all deblocking request and status flags.

Task Detach (DFHZDET)

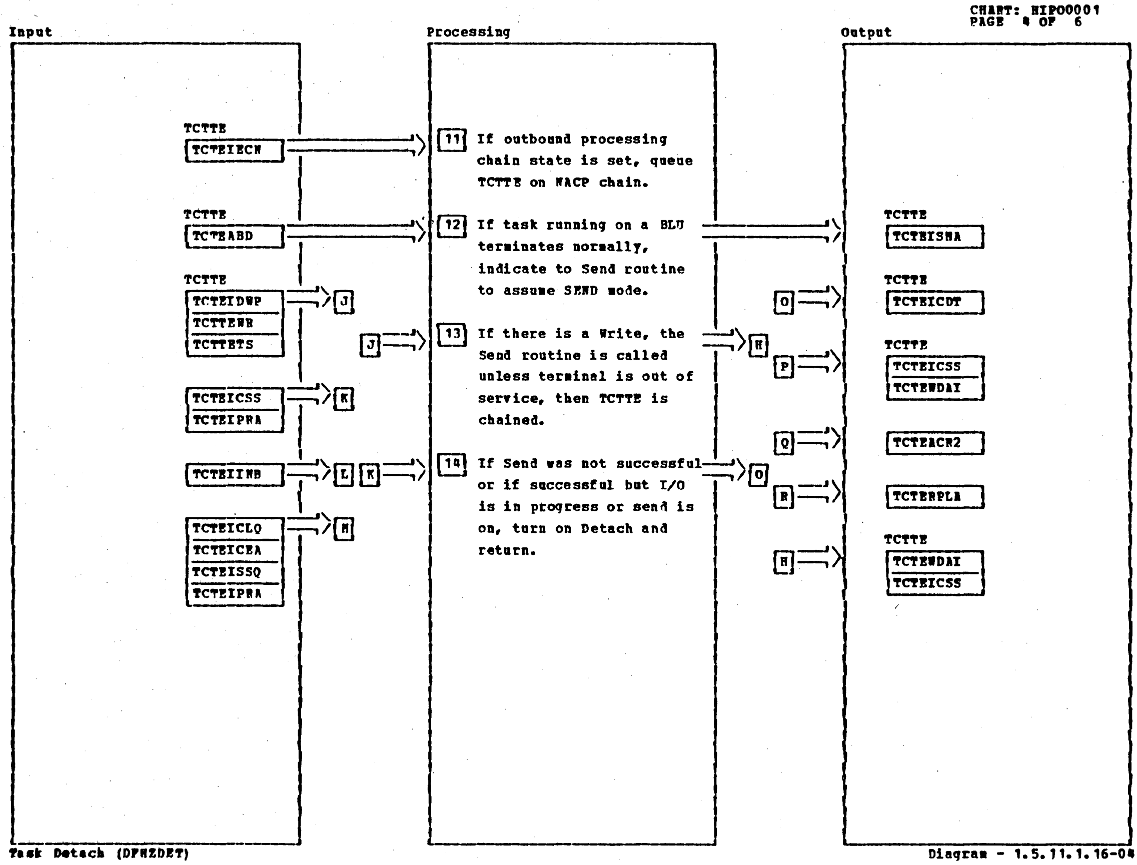
Diagram - 1.5.11.1.16-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 10 The TCTTE will be chained to drive the Attach module for BMS. | DFHZATT | TCZDET90 | 1.5.11<br>.1.10 | | | | |

Task Detach (DFHZDET)

Diagram - 1.5.11.1.16-03

CHART: HIPO0001
PAGE 4 OF 6



11 If outbound processing chain state is set, queue TCTTE on NACP chain.

12 If task running on a BLU terminates normally, indicate to Send routine to assume SEND mode.

13 If there is a Write, the Send routine is called unless terminal is out of service, then TCTTE is chained.

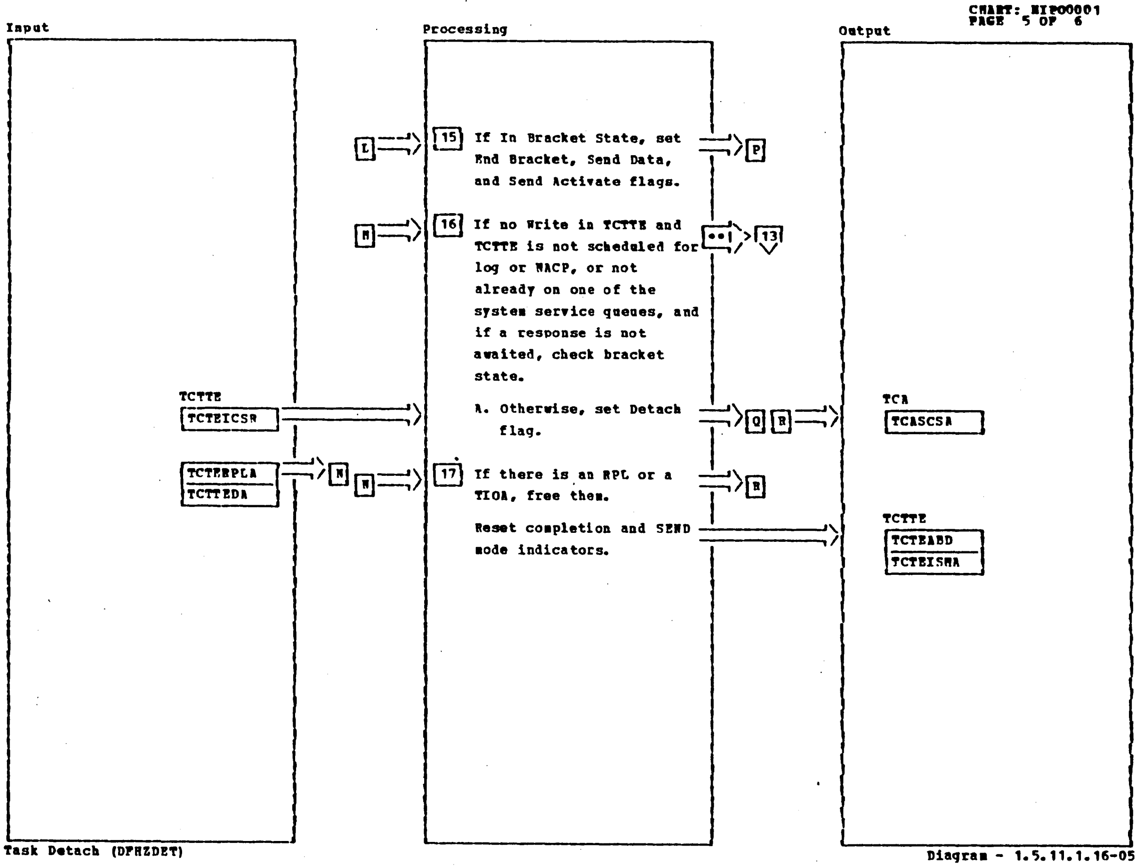
14 If Send was not successful or if successful but I/O is in progress or send is on, turn on Detach and return.

Task Detach (DFHZDET)

Diagram - 1.5.11.1.16-04

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 11 NACP error code X'9C'. | | | | | | | |
| 12 Normal termination of a task running on a BLU is indicated by TCTEABD being set OFF. | | | | | | | |
| 13 Send Sync DFSYN. | DFHZSDS | | 1.5.11<br>.1.7 | | | | |
| 14 The TCTTE will be chained. | | TCZDET90 | | | | | |

CHART: HIP00001
PAGE 5 OF 6



15 If In Bracket State, set End Bracket, Send Data, and Send Activate flags.

16 If no Write in TCTTE and TCTTE is not scheduled for log or NACP, or not already on one of the system service queues, and if a response is not awaited, check bracket state.

A. Otherwise, set Detach flag.

17 If there is an RPL or a TIOA, free them.

Reset completion and SEND mode indicators.

Task Detach (DFHZDET)

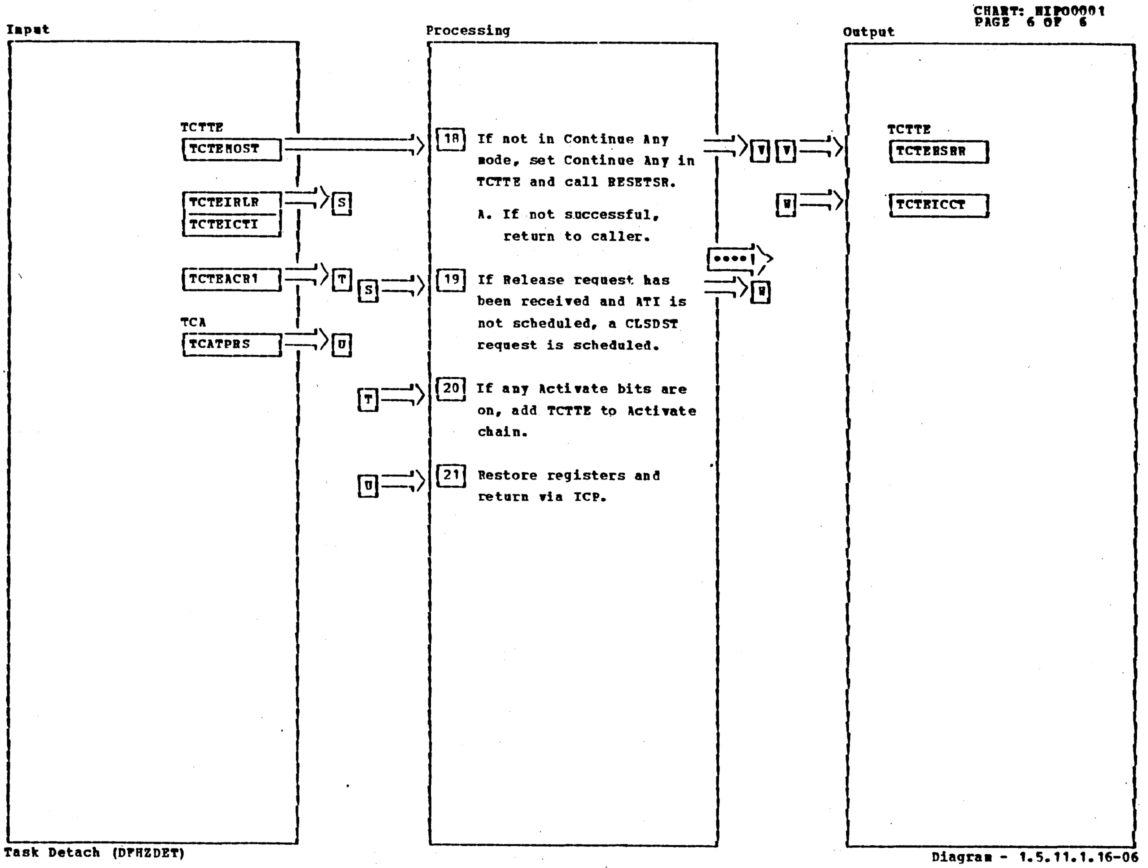
Diagram - 1.5.11.1.16-05

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Task Detach (DFHZDET)

Diagram - 1.5.11.1.16-05

CHART: HIP00001
PAGE 6 OF 6



18 If not in Continue Any mode, set Continue Any in TCTTE and call RESETSR.

A. If not successful, return to caller.

19 If Release request has been received and ATI is not scheduled, a CLSDST request is scheduled.

20 If any Activate bits are on, add TCTTE to Activate chain.

21 Restore registers and return via ICP.

Task Detach (DFHZDET)

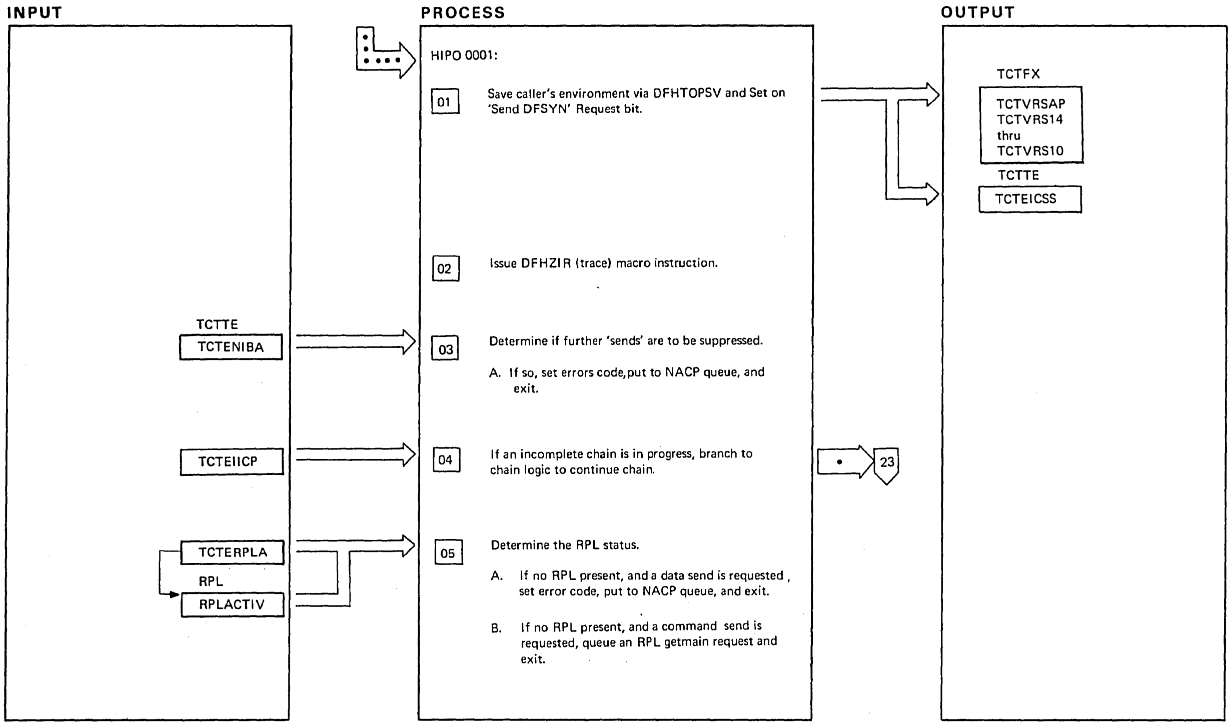
Diagram - 1.5.11.1.16-06

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 18 RESETSR module.<br>Register 15 will contain zeros if reset was successful. | DFHZRST | | 1.5.11<br>.1.21 | | | | |
| 21 DFHICP TYPE=EXIT | | | | | | | |

Task Detach (DFHZDET)

Diagram - 1.5.11.1.16-06

Page 1 of 12



HIPO 0001:

01 Save caller's environment via DFHTOPSV and Set on 'Send DFSYN' Request bit.

02 Issue DFHZIR (trace) macro instruction.

03 Determine if further 'sends' are to be suppressed.

A. If so, set errors code,put to NACP queue, and exit.

04 If an incomplete chain is in progress, branch to chain logic to continue chain.

05 Determine the RPL status.

A. If no RPL present, and a data send is requested, set error code, put to NACP queue, and exit.

B. If no RPL present, and a command send is requested, queue an RPL getmain request and exit.

Send DFSYN (DFHZSDS)

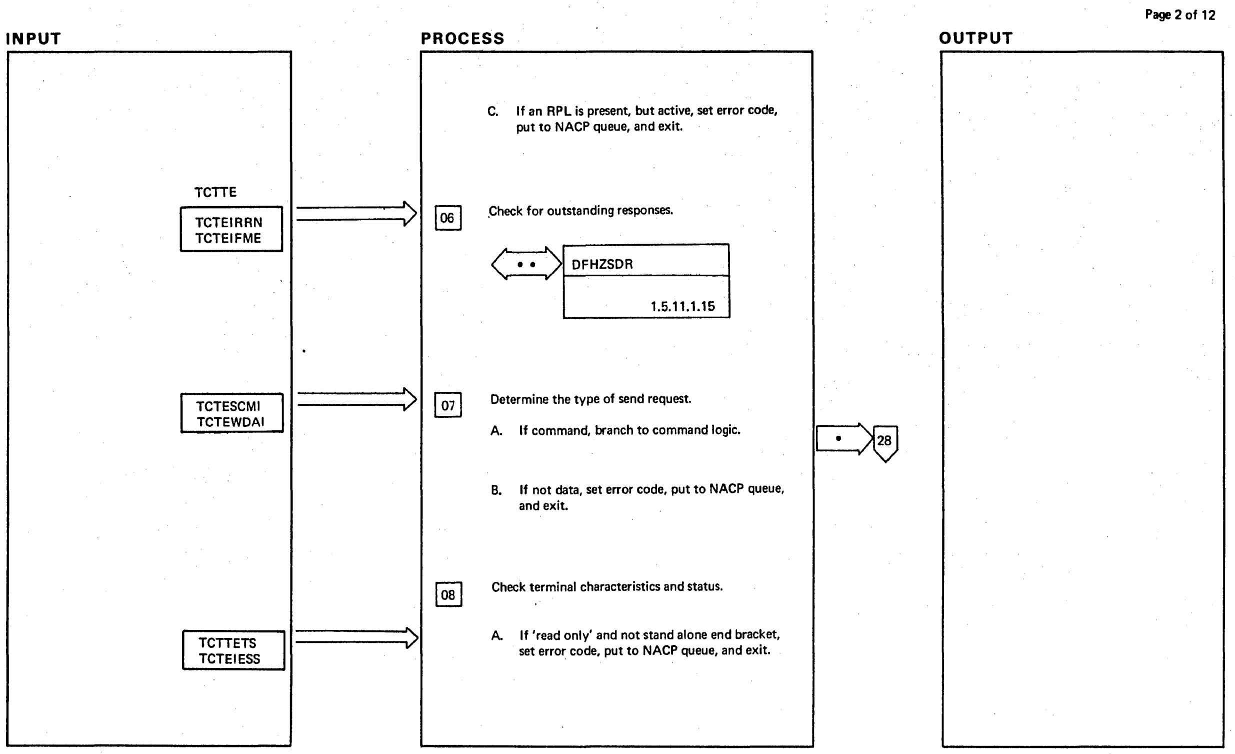
Diagram - 1.5.11.1.17-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 04 If an incomplete chain is in progress, we must finish the sending of that message before allowing any other task to gain control. | | TCZSDS02 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-01



C. If an RPL is present, but active, set error code, put to NACP queue, and exit.

06 Check for outstanding responses.

07 Determine the type of send request.

A. If command, branch to command logic.

B. If not data, set error code, put to NACP queue, and exit.

08 Check terminal characteristics and status.

A. If 'read only' and not stand alone end bracket, set error code, put to NACP queue, and exit.

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-02

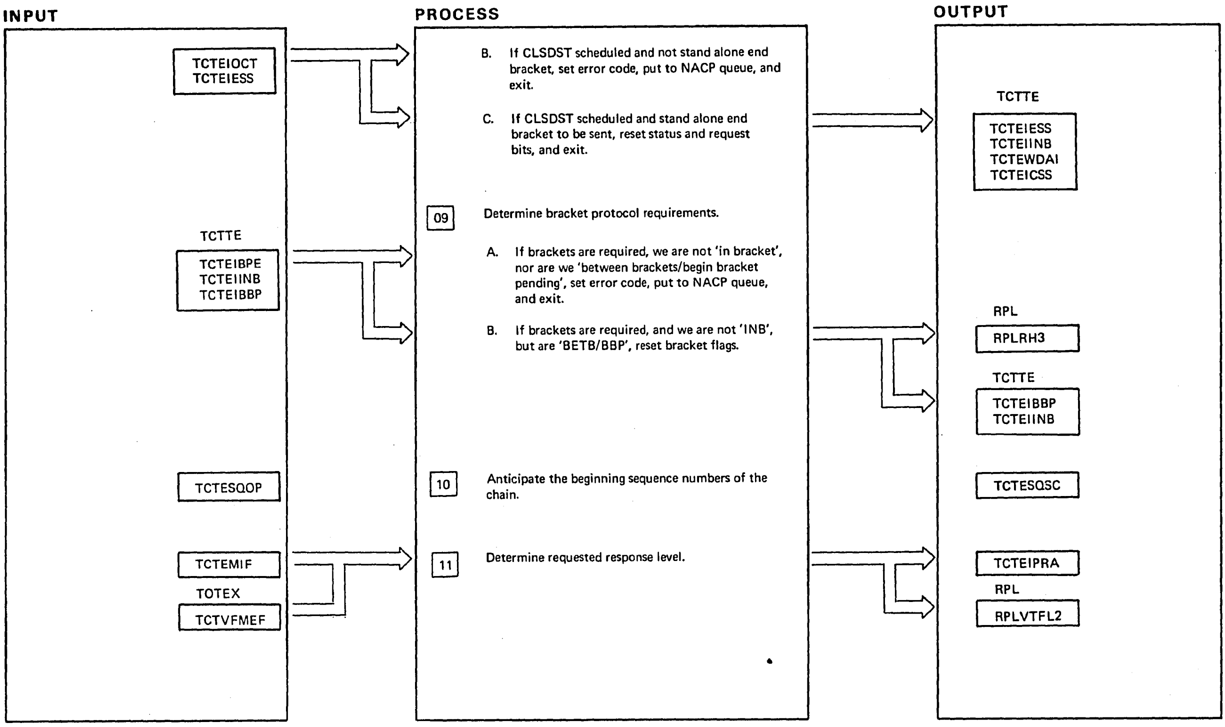
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 06 Upon return from DFHZSDR, the completion code is checked to see if the 'send' was successful. If it was not, DFHZSDS will be exited. The failure will have been noted in DFHZSDR. | | TCZSDS06 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-02

Page 3 of 12



B. If CLSDST scheduled and not stand alone end bracket, set error code, put to NACP queue, and exit.

C. If CLSDST scheduled and stand alone end bracket to be sent, reset status and request bits, and exit.

09 Determine bracket protocol requirements.

A. If brackets are required, we are not 'in bracket', nor are we 'between brackets/begin bracket pending', set error code, put to NACP queue, and exit.

B. If brackets are required, and we are not 'INB', but are 'BETB/BBP', reset bracket flags.

10 Anticipate the beginning sequence numbers of the chain.

11 Determine requested response level.

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-03

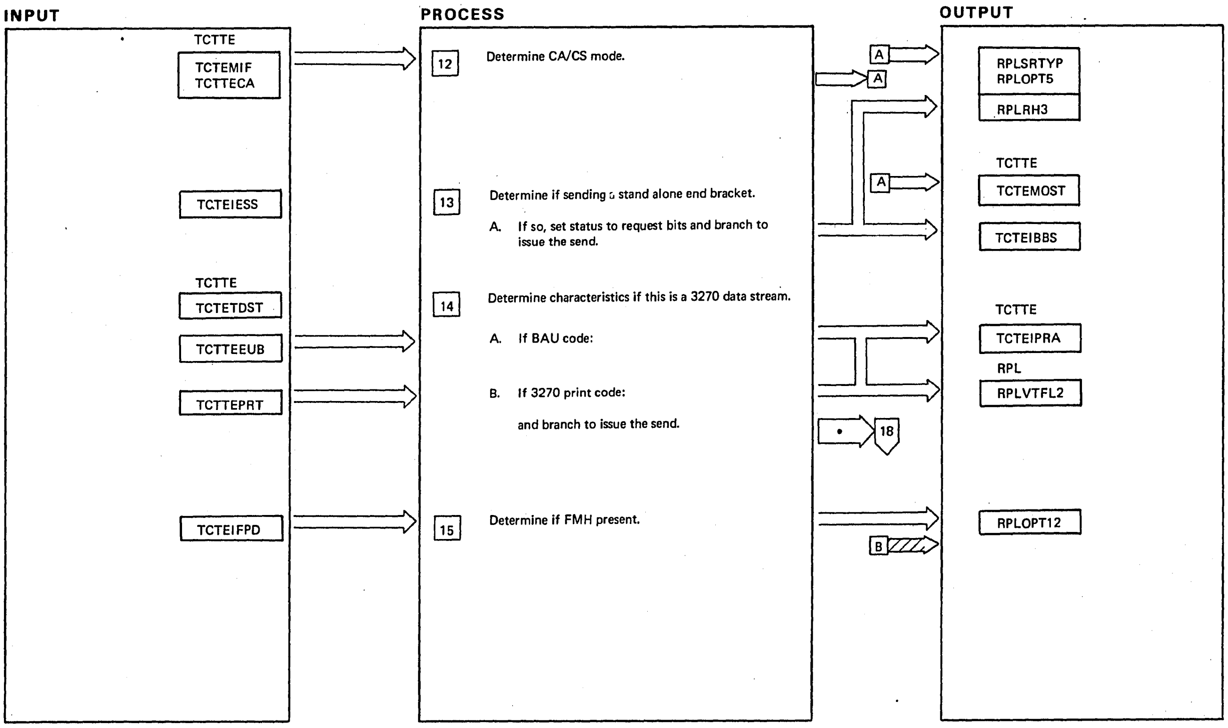
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 11 Response level will be one of the following four combinations:<br>DBF/FME<br>DCF/RRN<br>EX/FMC<br>EX/RRN | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-03

Page 4 of 12



12 Determine CA/CS mode.

13 Determine if sending a stand alone end bracket.

A. If so, set status to request bits and branch to issue the send.

14 Determine characteristics if this is a 3270 data stream.

A. If BAU code:

B. If 3270 print code:

and branch to issue the send.

15 Determine if FMH present.

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-04

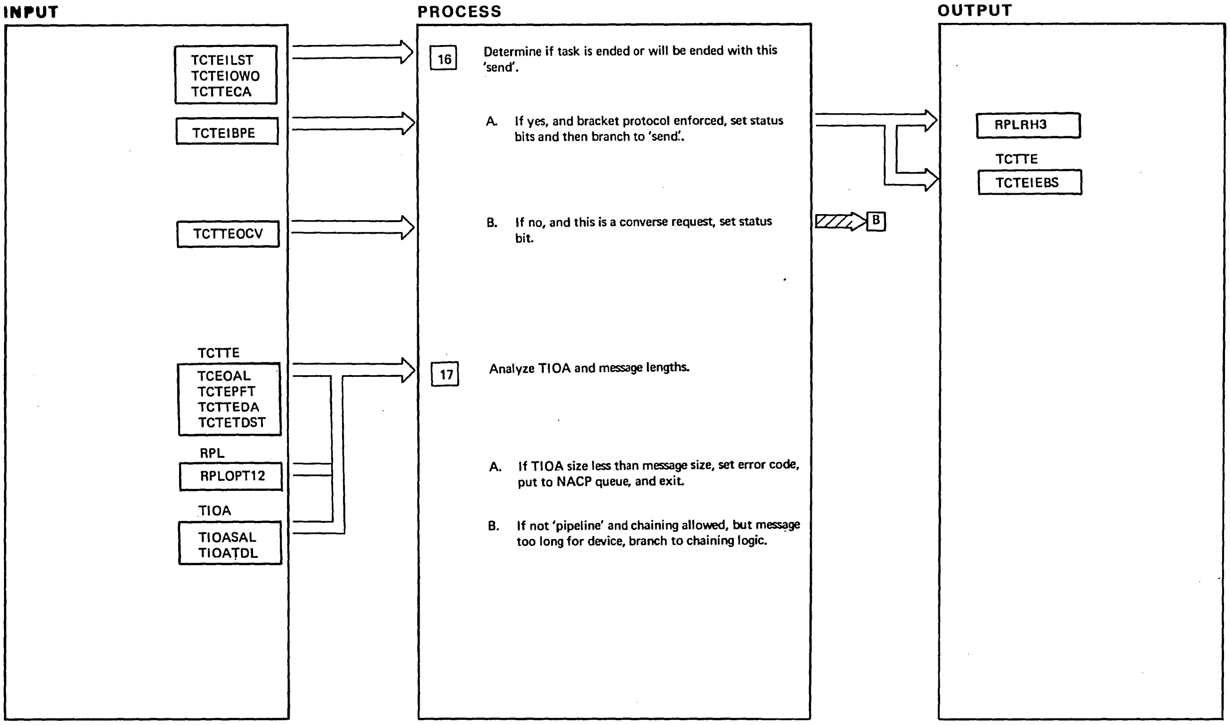
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-04

Page 5 of 12



16 Determine if task is ended or will be ended with this 'send'.

A. If yes, and bracket protocol enforced, set status bits and then branch to 'send'.

B. If no, and this is a converse request, set status bit.

17 Analyze TIOA and message lengths.

A. If TIOA size less than message size, set error code, put to NACP queue, and exit.

B. If not 'pipeline' and chaining allowed, but message too long for device, branch to chaining logic.

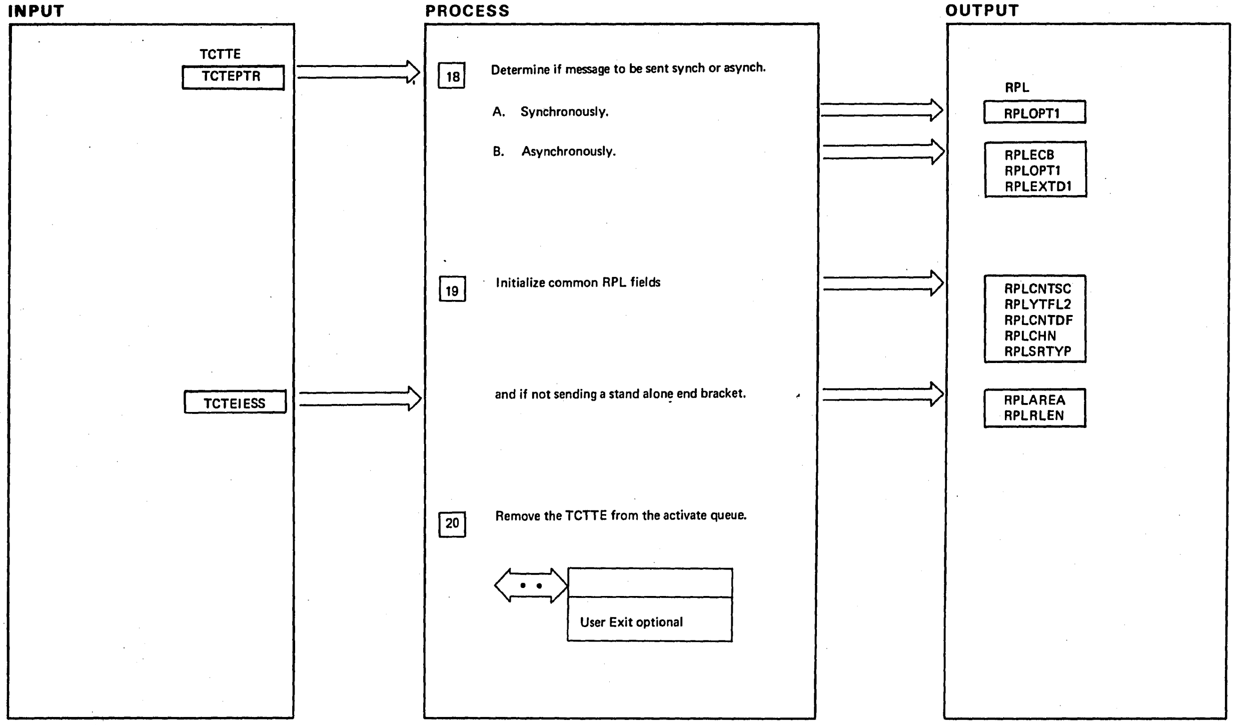
Send DFSYN (DFHZSDS) Diagram - 1.5.11.1.17-05

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 17 If chaining not allowed by the user (TCT parm BUFFER=0), the user accepts the responsibility for ensuring that the message will fit the device to which it will be sent. |  | TCZSDS2B |  |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Send DFSYN (DFHZSDS) Diagram - 1.5.11.1.17-05

Page 6 of 12



18 Determine if message to be sent synch or asynch.

A. Synchronously.

B. Asynchronously.

19 Initialize common RPL fields

and if not sending a stand alone end bracket.

20 Remove the TCTTE from the activate queue.

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-06

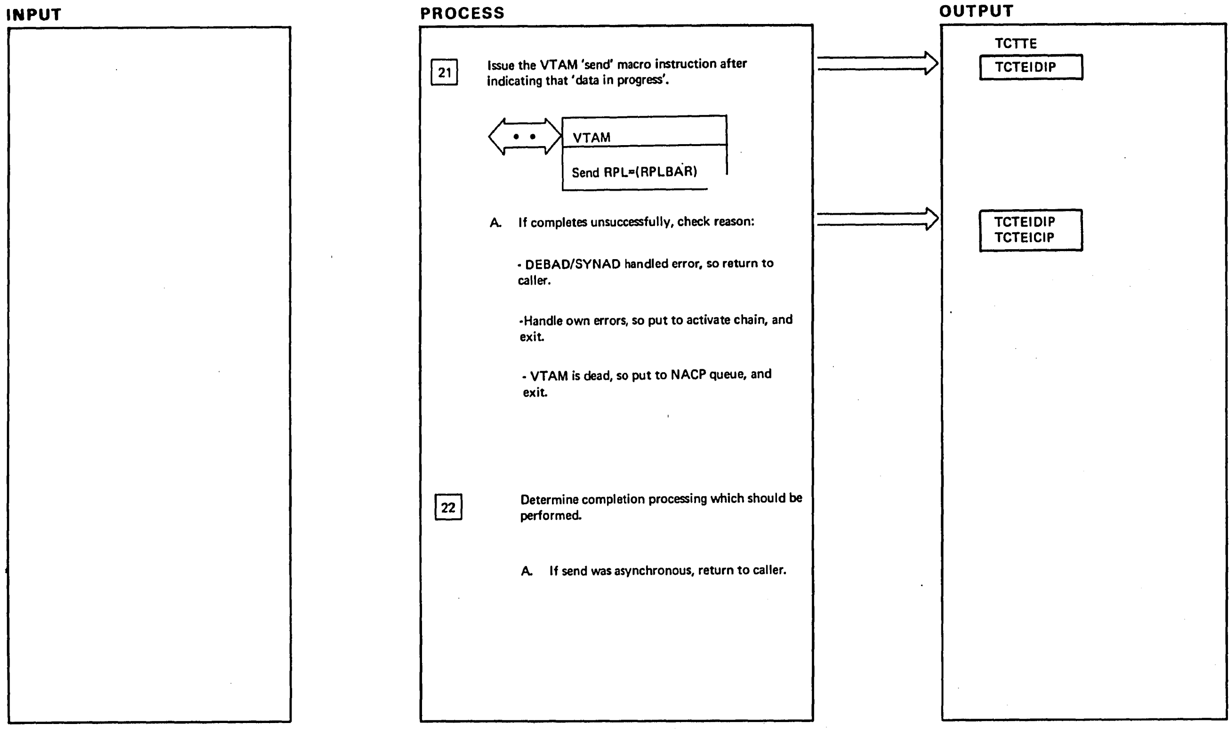
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-06

Page 7 of 12



INPUT

PROCESS

OUTPUT

21 Issue the VTAM 'send' macro instruction after indicating that 'data in progress'.

VTAM

Send RPL=(RPLBAR)

A. If completes unsuccessfully, check reason:

- DEBAD/SYNAD handled error, so return to caller.

-Handle own errors, so put to activate chain, and exit.

- VTAM is dead, so put to NACP queue, and exit.

22 Determine completion processing which should be performed.

A. If send was asynchronous, return to caller.

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-07

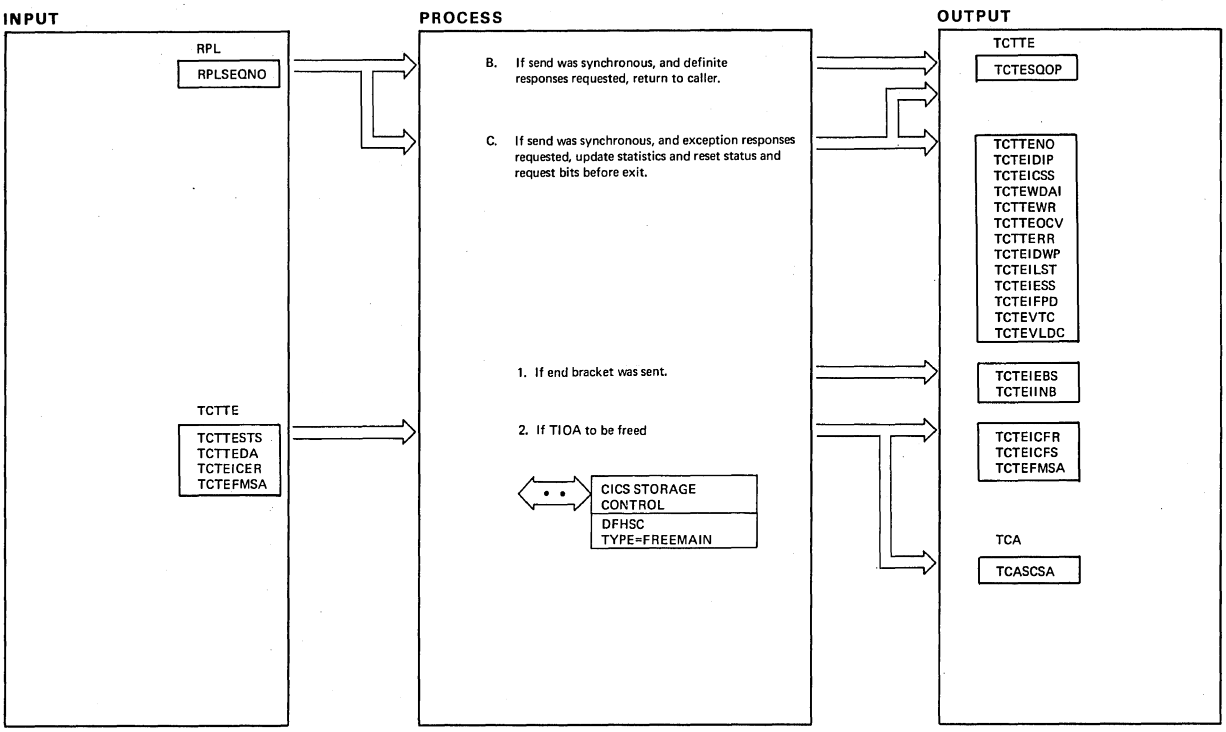
| | NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 21 | If the send was made synchronously, control returns immediately after VTAM accepts the request. If asynchronously, control returns after activation of the RPL exit (DFHZSDX). | | TCZSIDS28 | |
| | If authorized path SRB support is being used, this invokes the RPL service routine to interface to VTAM synchronously. | DFHZHPSR | | 1.5.11.1.37 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-07

Page 8 of 12



B. If send was synchronous, and definite responses requested, return to caller.

C. If send was synchronous, and exception responses requested, update statistics and reset status and request bits before exit.

1. If end bracket was sent.

2. If TIOA to be freed

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-08

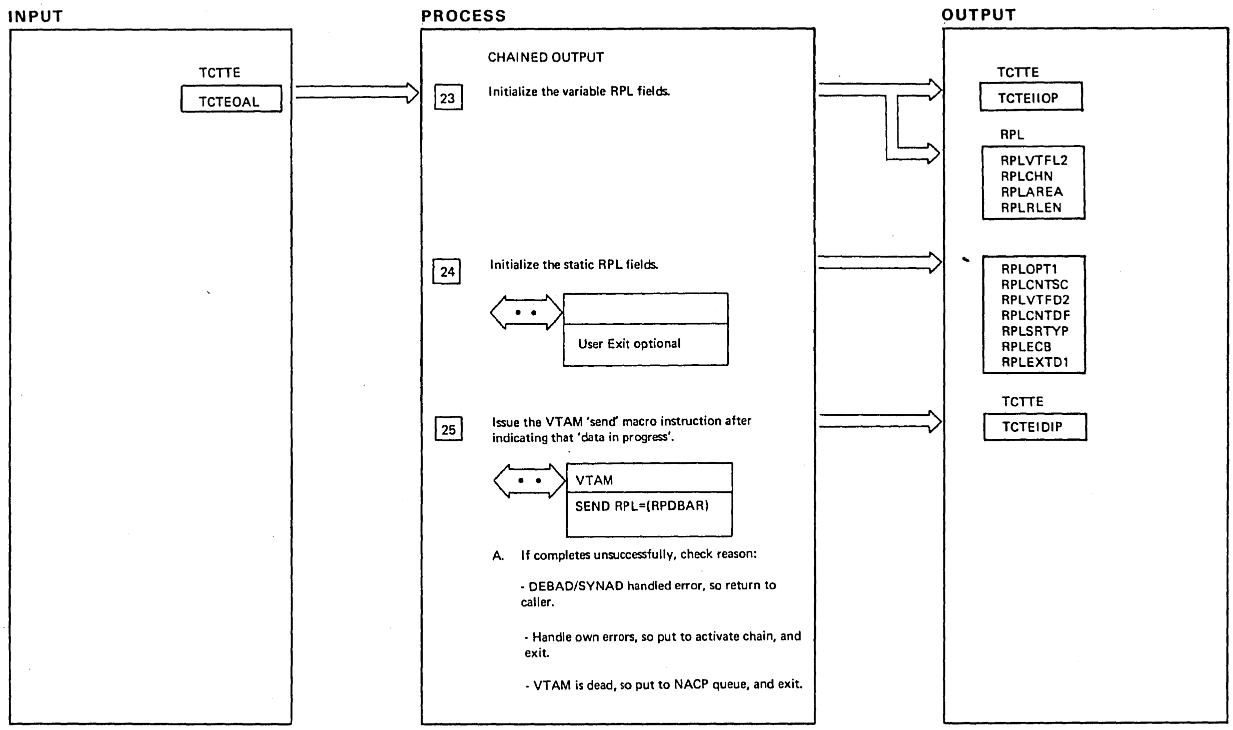
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 22 If send was asynchronous, completion processing will be determined in the RPL-exit (DFHZSDX). If send was synchronous and definite responses requested, completion will be determined in the Response Exit (DFHZRPX). | | TCZSDS30 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-08

Page 9 of 12



INPUT

TCTTE

TCTEOAL

PROCESS

CHAINED OUTPUT

23 Initialize the variable RPL fields.

24 Initialize the static RPL fields.

User Exit optional

25 Issue the VTAM 'send' macro instruction after indicating that 'data in progress'.

VTAM

SEND RPL=(RPDBAR)

A. If completes unsuccessfully, check reason:

- DEBAD/SYNAD handled error, so return to caller.

- Handle own errors, so put to activate chain, and exit.

- VTAM is dead, so put to NACP queue, and exit.

OUTPUT

TCTTE

TCTEIIOP

RPL

RPLVTFL2
RPLCHN
RPLAREA
RPLRLEN

RPLOPT1
RPLCNTSC
RPLVTFD2
RPLCNTDF
RPLSRTYP
RPLECB
RPLEXTD1

TCTTE

TCTEIDIP

Send DFSYN (DFHZSDS)

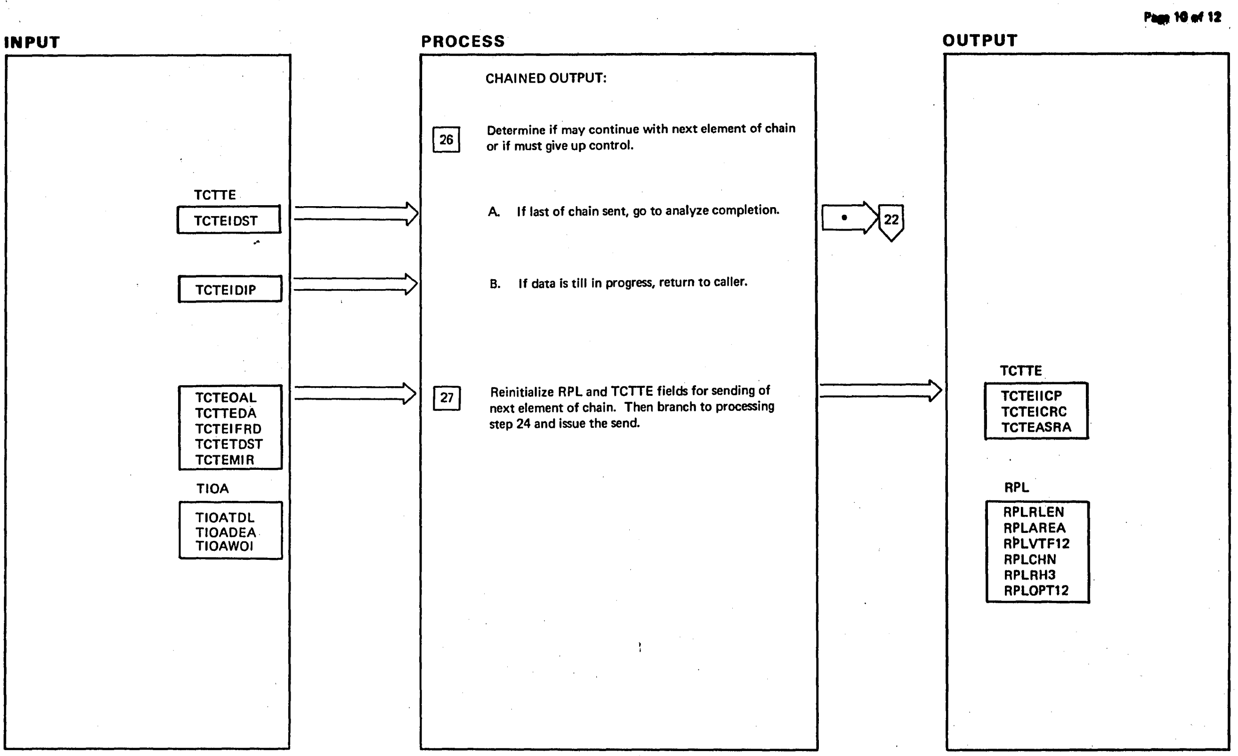
Diagram - 1.5.11.1.17-09

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 25 If authorized path SRB support is being used this invokes the RPL service routine to interface to VTAM synchrouously. | DFHZHPSR | | 1.5.11.1.37 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-09



CHAINED OUTPUT:

26 Determine if may continue with next element of chain or if must give up control.

A. If last of chain sent, go to analyze completion.

B. If data is till in progress, return to caller.

27 Reinitialize RPL and TCTTE fields for sending of next element of chain. Then branch to processing step 24 and issue the send.

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-10

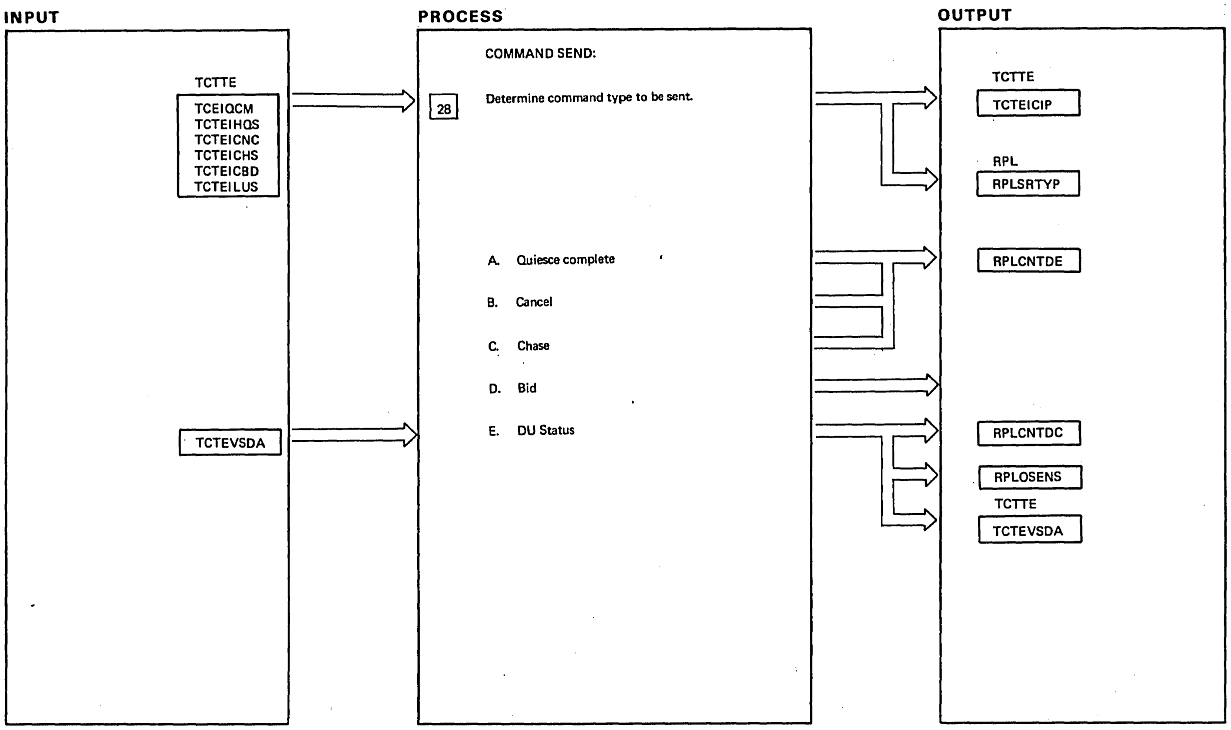
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 26 A. If data is still in progress, the previous 'send' has not yet completed. Control will be returned so that other work unaffected by 'data in progress' may be done. The RPL exit (DFHZSDX) will set up for an asynch return of control to finish sending the chain. | TCZSDS46 | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-10

Page 11 of 12



Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-11

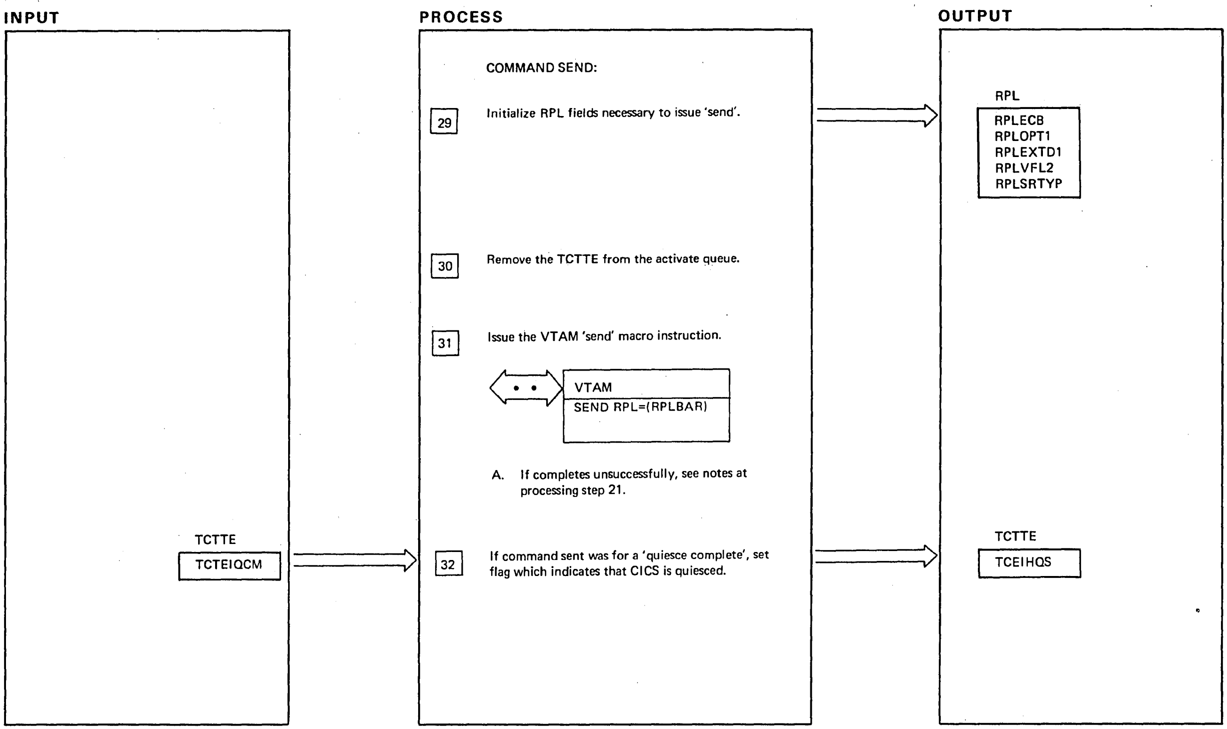
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-11

Page 12 of 12



COMMAND SEND:

29 Initialize RPL fields necessary to issue 'send'.

30 Remove the TCTTE from the activate queue.

31 Issue the VTAM 'send' macro instruction.

A. If completes unsuccessfully, see notes at processing step 21.

32 If command sent was for a 'quiesce complete', set flag which indicates that CICS is quiesced.

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-12

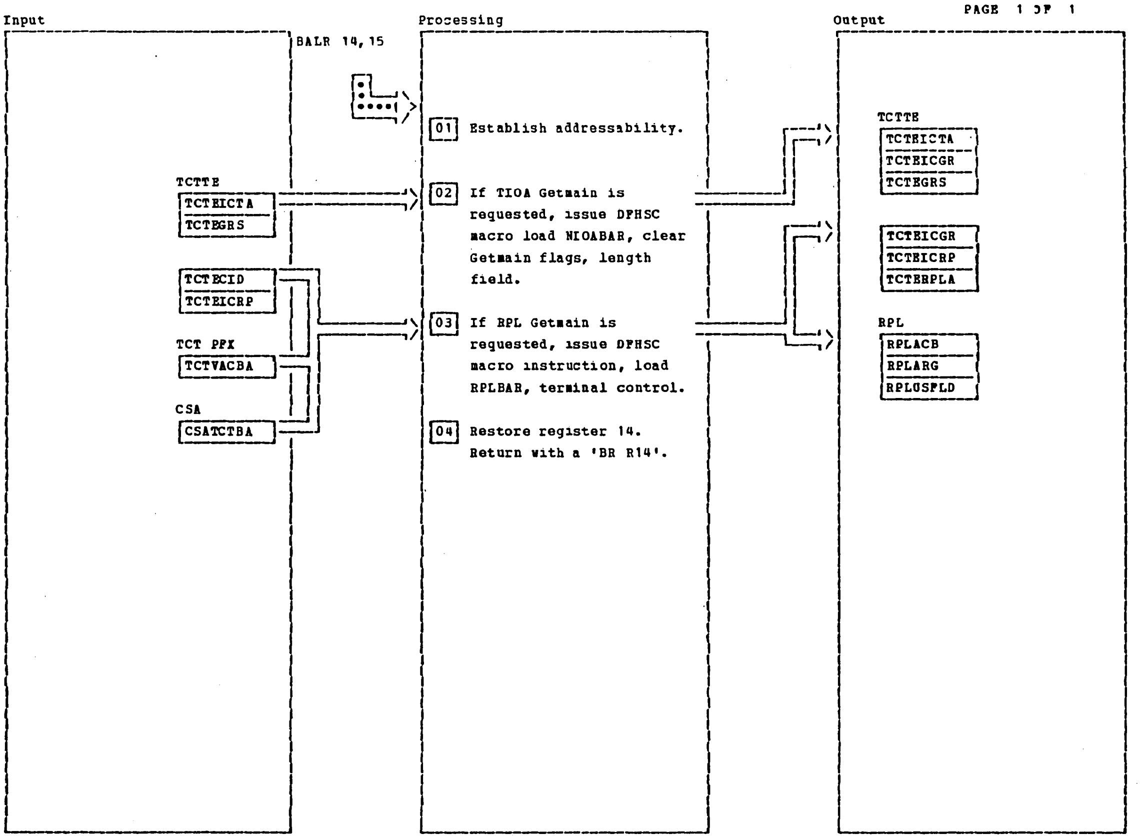
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Send DFSYN (DFHZSDS)

Diagram - 1.5.11.1.17-12

PAGE 1 OF 1



01 Establish addressability.

02 If TIOA Getmain is requested, issue DFHSC macro load NIOABAR, clear Getmain flags, length field.

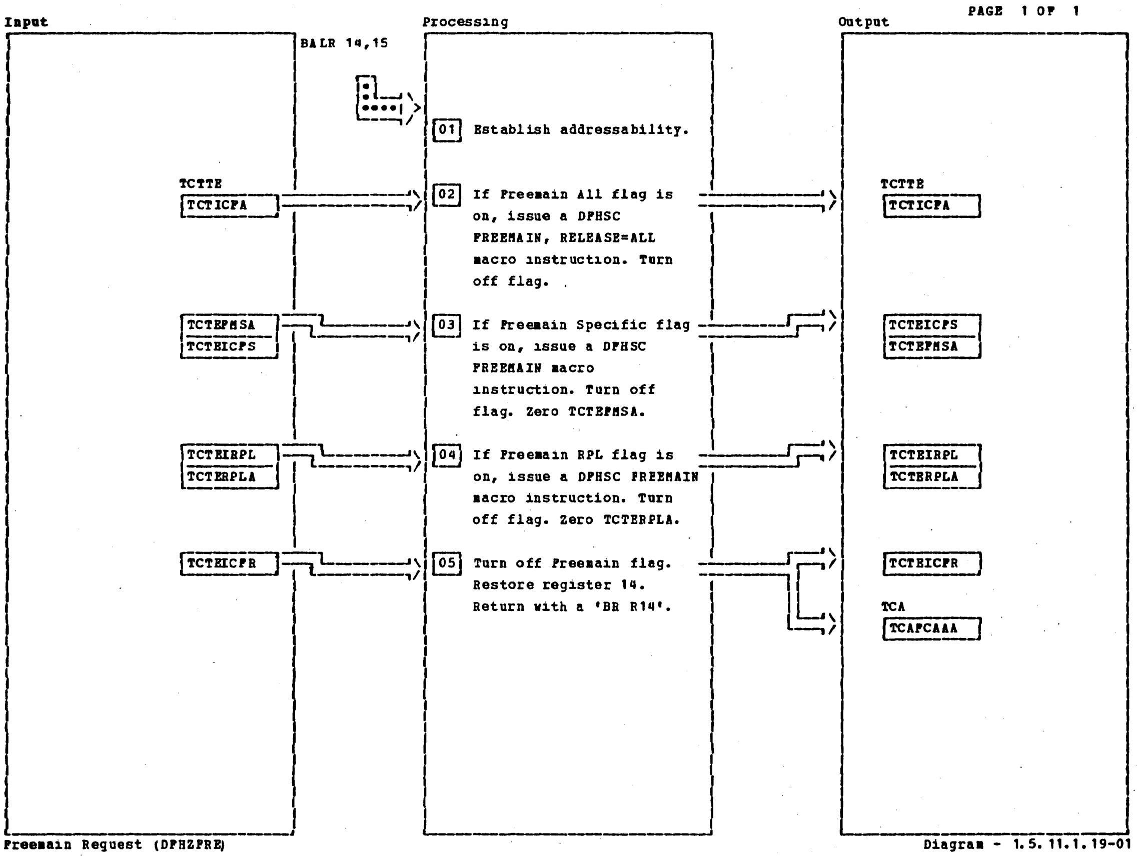
03 If RPL Getmain is requested, issue DFHSC macro instruction, load RPLBAR, terminal control.

04 Restore register 14. Return with a 'BR R14'.

GETMAIN Request (DFHZGET) Diagram - 1.5.11.1.18-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Control is received from various VTAM TCP modules. DFHZGET is treated as a subroutine, to Getmain storage for TIOA or RPL. Register 15 is used as the base register. Register 0 is used to save register 14. Register 14 is used as a work register. No other registers are used, none are saved. | | | | | | | |
| 02 Test TCTEICTA. | | | | | | | |
| 03 If TCTEICTA (GETMFLG) is NOT on, it is assumed TCTEICRP IS on (RPL Getmain). | | | | | | | |
| 04 Return to caller. | | | | | | | |

GETMAIN Request (DFHZGET) Diagram - 1.5.11.1.18-01



01 Establish addressability.

02 If Freemain All flag is on, issue a DFHSC FREEMAIN, RELEASE=ALL macro instruction. Turn off flag.

03 If Freemain Specific flag is on, issue a DFHSC FREEMAIN macro instruction. Turn off flag. Zero TCTEFMSA.

04 If Freemain RPL flag is on, issue a DFHSC FREEMAIN macro instruction. Turn off flag. Zero TCTERPLA.

05 Turn off Freemain flag. Restore register 14. Return with a 'BR R14'.

Freemain Request (DFHZFRE)

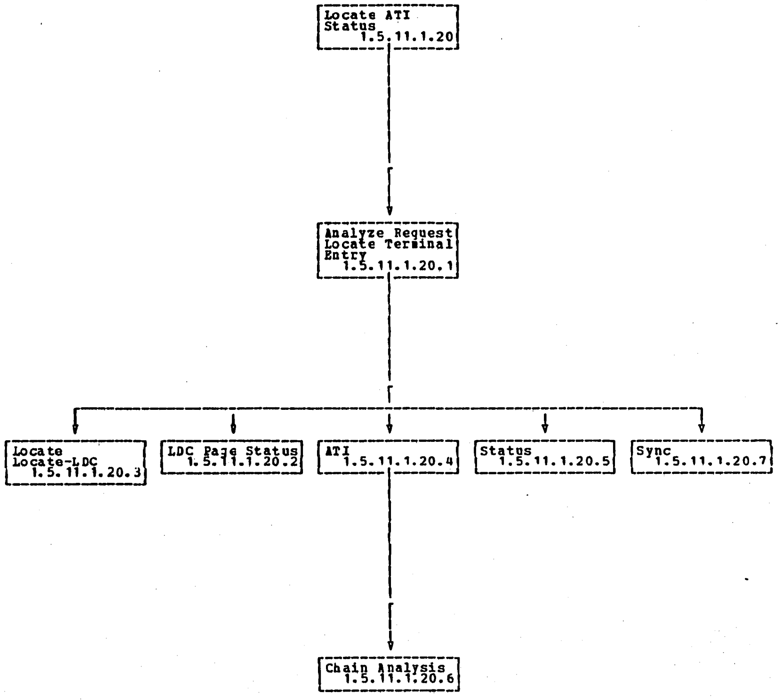
Diagram - 1.5.11.1.19-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Use register 15 as base register. Save register 14 in register 0. Use register 14 as work register. No other registers are used, none are saved.<br>Control is received from various VTAM TCP modules. DFHZFRE is treated as a subroutine to free storage. | | | | | | | |
| 02 After Freemain All flag is processed, continue with step 3. | | | | | | | |
| 03 After Freemain Specific flag is processed, continue with step 4. | | | | | | | |
| 04 After Freemain RPL is processed, continue with step 5. | | | | | | | |
| 05 Return to caller. | | | | | | | |

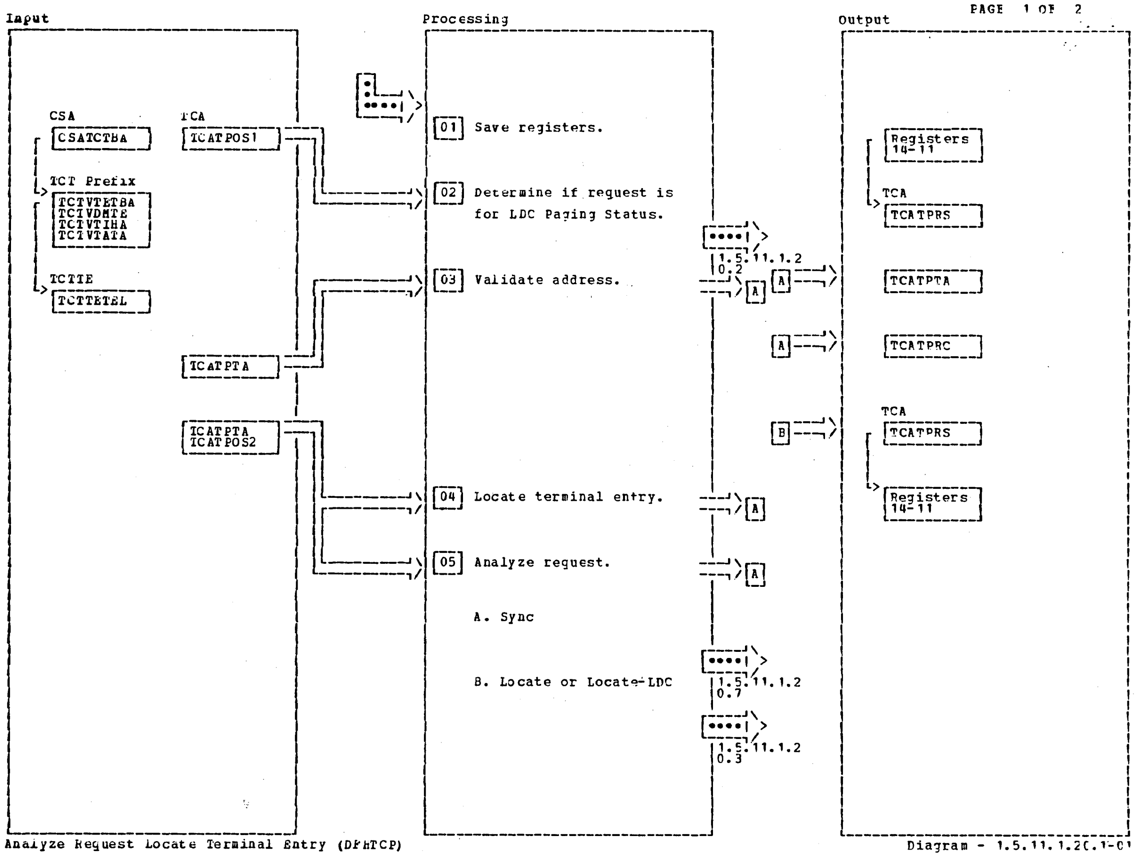
Freemain Request (DFHZFRE)

Diagram - 1.5.11.1.19-01

PAGE 1 OF 1


Locate ATI Status 1.5.11.1.20
Analyze Request Locate Terminal Entry 1.5.11.1.20.1
Locate Locate-LDC 1.5.11.1.20.3
LDC Page Status 1.5.11.1.20.2
ATI 1.5.11.1.20.4
Status 1.5.11.1.20.5
Sync 1.5.11.1.20.7
Chain Analysis 1.5.11.1.20.6

PAGE 1 OF 2



Analyze Request Locate Terminal Entry (DFHTCP)

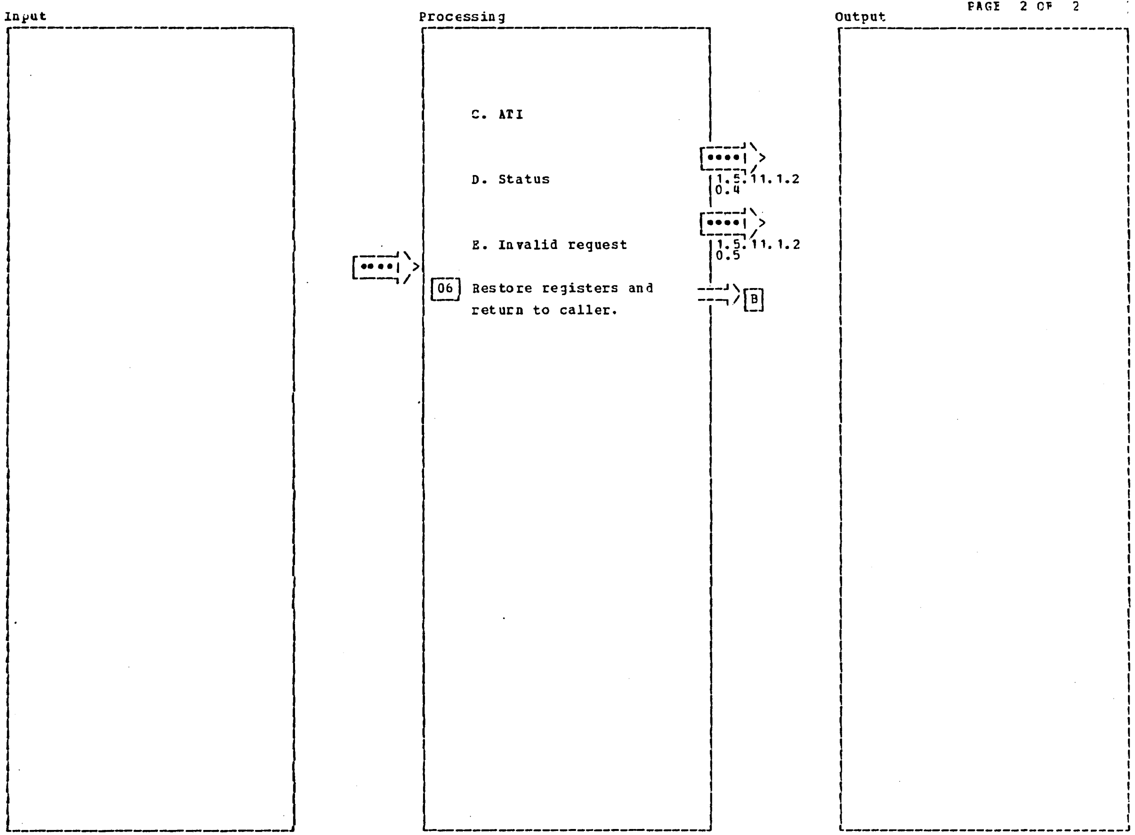
Diagram - 1.5.11.1.2C.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Entered from application program by DFHTC macro instruction for Locate, ATI, Status or Sync. | DFHZLOC | DFHZLOC1 | | | | | |
| 03 If this terminal request is First or ID the address is validated. | | TCZLOCFR | | | | | |
| 04 If the terminal request is for ID, the TCTTE is found using the DFHHASH macro instruction. | | TCZLOCLL | | | | | |
| 05 If the request is invalid, the invalid request bit is turned on in the TCA. | | TCZLOCAR | | | | | |

Analyze Request Locate Terminal Entry (DFHTCP)

Diagram - 1.5.11.1.2C.1-01

PAGE 2 OF 2



Analyze Request Locate Terminal Entry (DFHTCP) Diagram - 1.5.11.1.20.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 06 Input for this step is from charts 1.5.11.1.20.7, 1.5.11.1.20.2, 1.5.11.1.20.3, 1.5.11.1.20.5, 1.5.11.1.20.6 |  | TCZLOCRT |  |  |  |  |  |

Analyze Request Locate Terminal Entry (DFHTCP) Diagram - 1.5.11.1.20.1-02

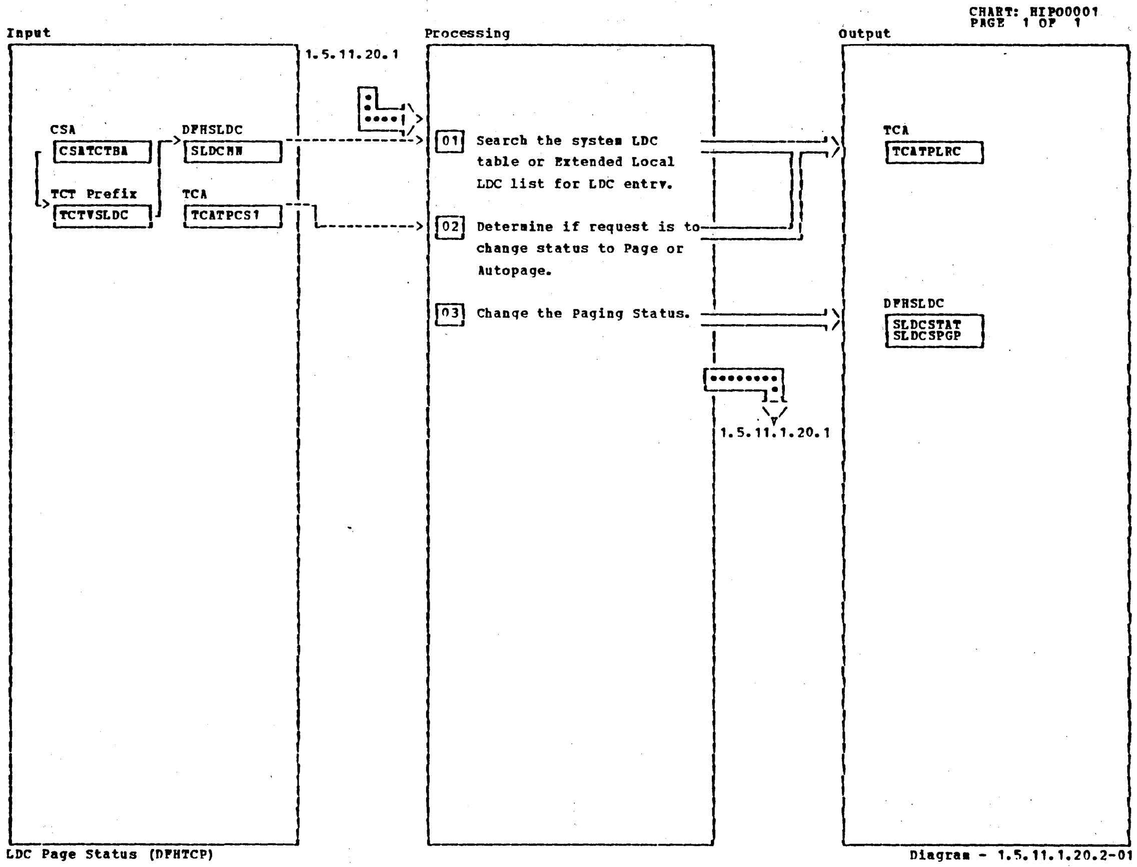


CHART: HIPO0001
PAGE 1 OF 1
Input
Processing
Output
1.5.11.20.1
CSA
CSATCTBA
DFHSLDC
SLDCNN
TCT Prefix
TCTVSLDC
TCA
TCATPCS1
01 Search the system LDC table or Extended Local LDC list for LDC entry.
02 Determine if request is to change status to Page or Autopage.
03 Change the Paging Status.
TCA
TCATPLRC
DFHSLDC
SLDCSTAT
SLDCSPGP
1.5.11.1.20.1

LDC Page Status (DFHTCP)

Diagram - 1.5.11.1.20.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 If LDC is not found in the table, return with invalid LDC return code. | DFHZLOC | TCZLOCDC<br>TCZLOCDL | |
| 02 If Status is not Page or Autopage return with invalid request error code. | | TCZLOCDF | |
| 03 Set LDC Status to Page or Autopage. | | TCZLOCDF<br>TCZLOCDA | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

LDC Page Status (DFHTCP)

Diagram - 1.5.11.1.20.2-01

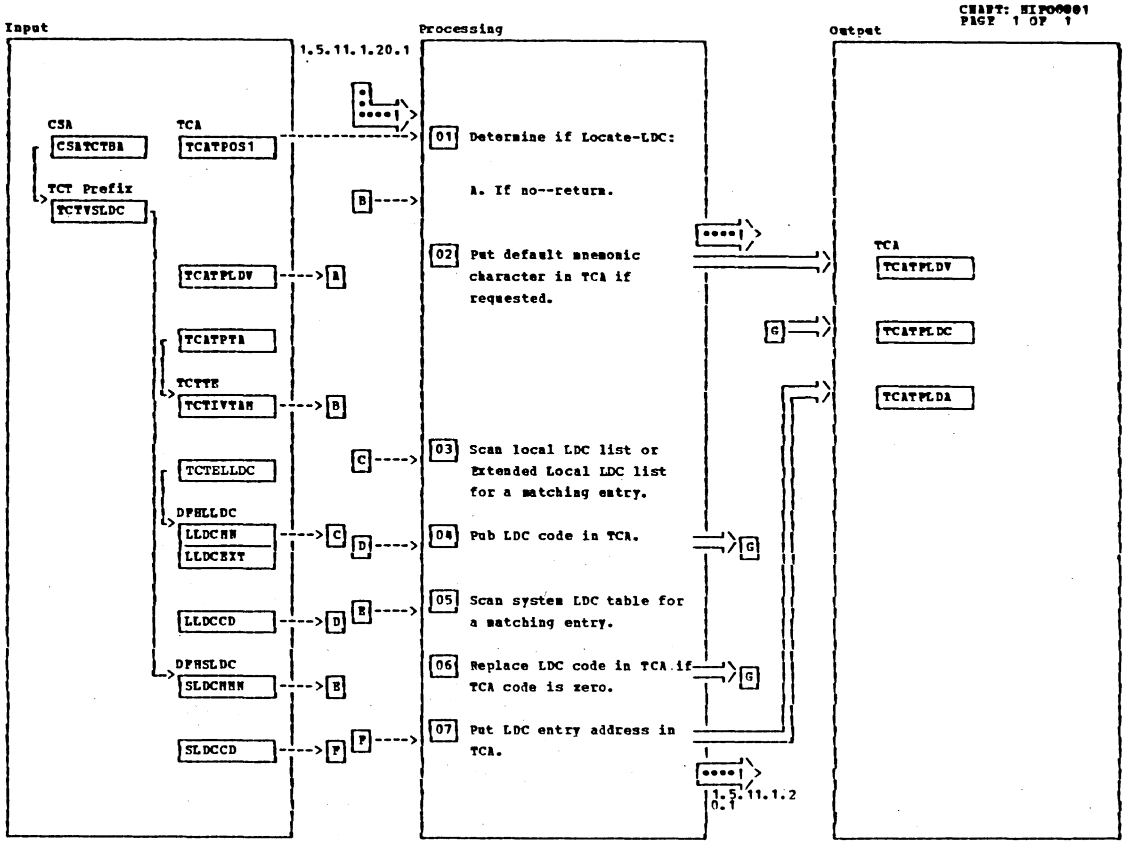


CHART: HIPO0001
PAGE 1 OF 1
Input
Processing
Output
1.5.11.1.20.1
CSA
CSATCTBA
TCT Prefix
TCTVSLDC
TCA
TCATPOS1
TCATPLDV
TCATPTA
TCTTE
TCTIVTAM
TCTELLDC
DFHLLDC
LLDCMN
LLDCEXT
LLDCCD
DFHSLDC
SLDCMMN
SLDCCD
A
B
C
D
E
F
G
01 Determine if Locate-LDC:
A. If no--return.
02 Put default mnemonic character in TCA if requested.
03 Scan local LDC list or Extended Local LDC list for a matching entry.
04 Pub LDC code in TCA.
05 Scan system LDC table for a matching entry.
06 Replace LDC code in TCA if TCA code is zero.
07 Put LDC entry address in TCA.
1.5.11.1.20.1
TCA
TCATPLDV
TCATPLDC
TCATPLDA

Locate and Locate-LDC (DFHTCP)

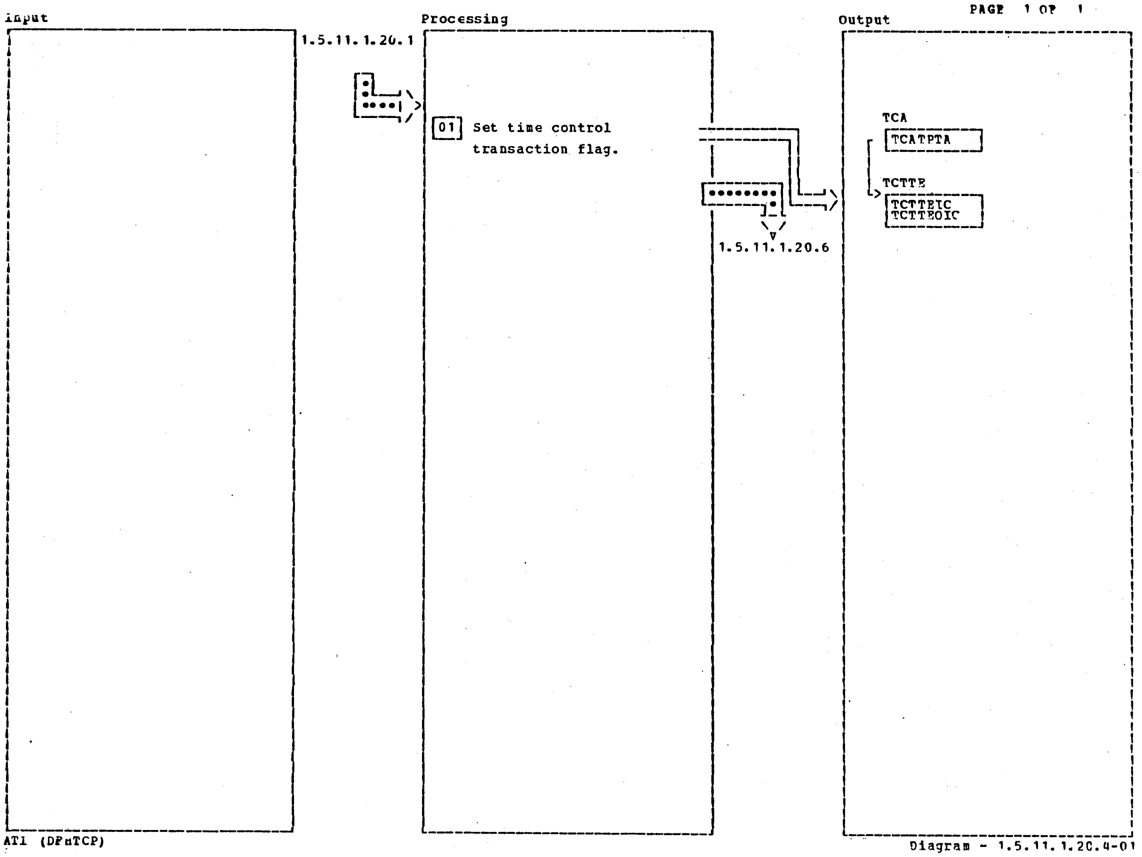
Diagram - 1.5.11.1.20.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If not an LDC request or not a VTAM terminal clear the associated TCA fields and return. | DFHZLOC | TCZLOCLD | | | | | |
| 02 If LDCs are not supported by the terminal, return. | | | | | | | |
| 03 If no match is found, return. | | TCZLOCLL | | | | | |
| 04 | | TCZLOCLS | | | | | |
| 05 If no match, return. | | TCZLOCSL | | | | | |
| 06 | | TCZLOCLX | | | | | |
| 07 | | TCZLOCS | | | | | |

Locate and Locate-LDC (DFHTCP)

Diagram - 1.5.11.1.20.3-01

PAGE 1 OF 1



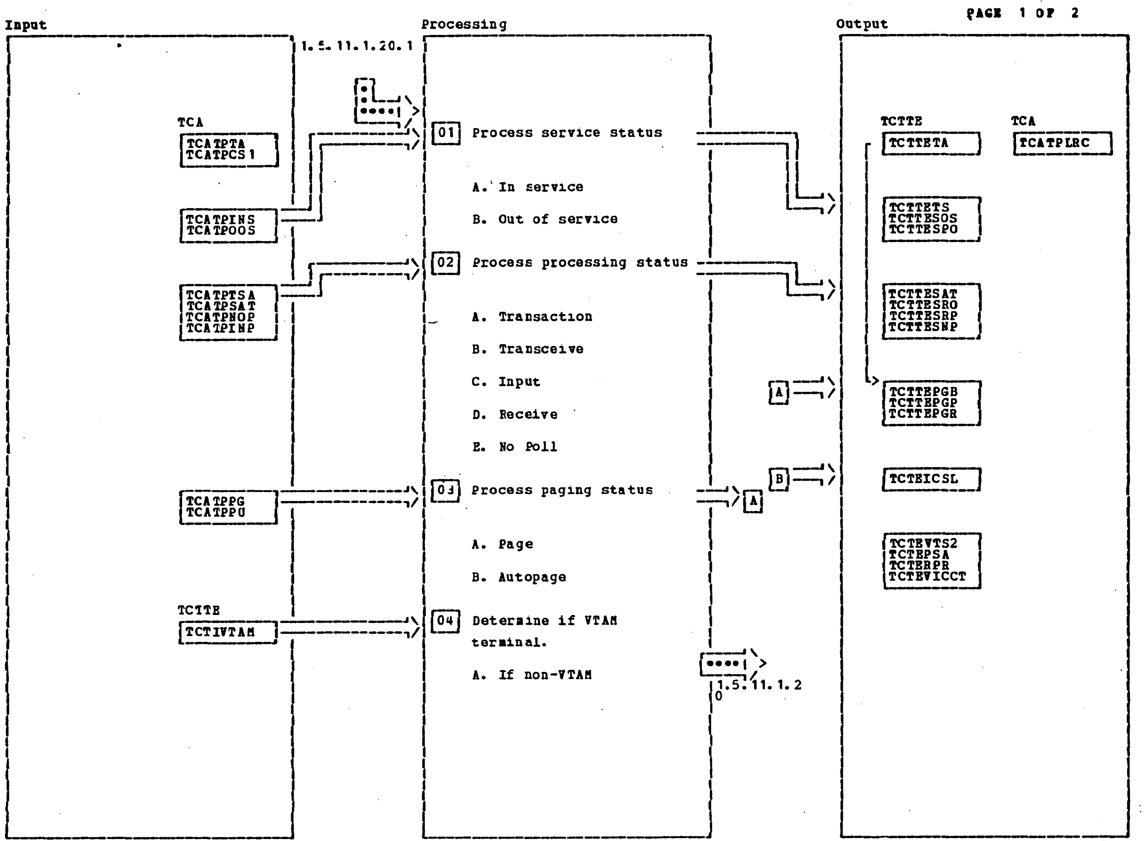
ATI (DFHTCP)

Diagram - 1.5.11.1.20.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | DFHZLOC | TCZLOCSR | | | | | |

ATI (DFHTCP)

Diagram - 1.5.11.1.20.4-01

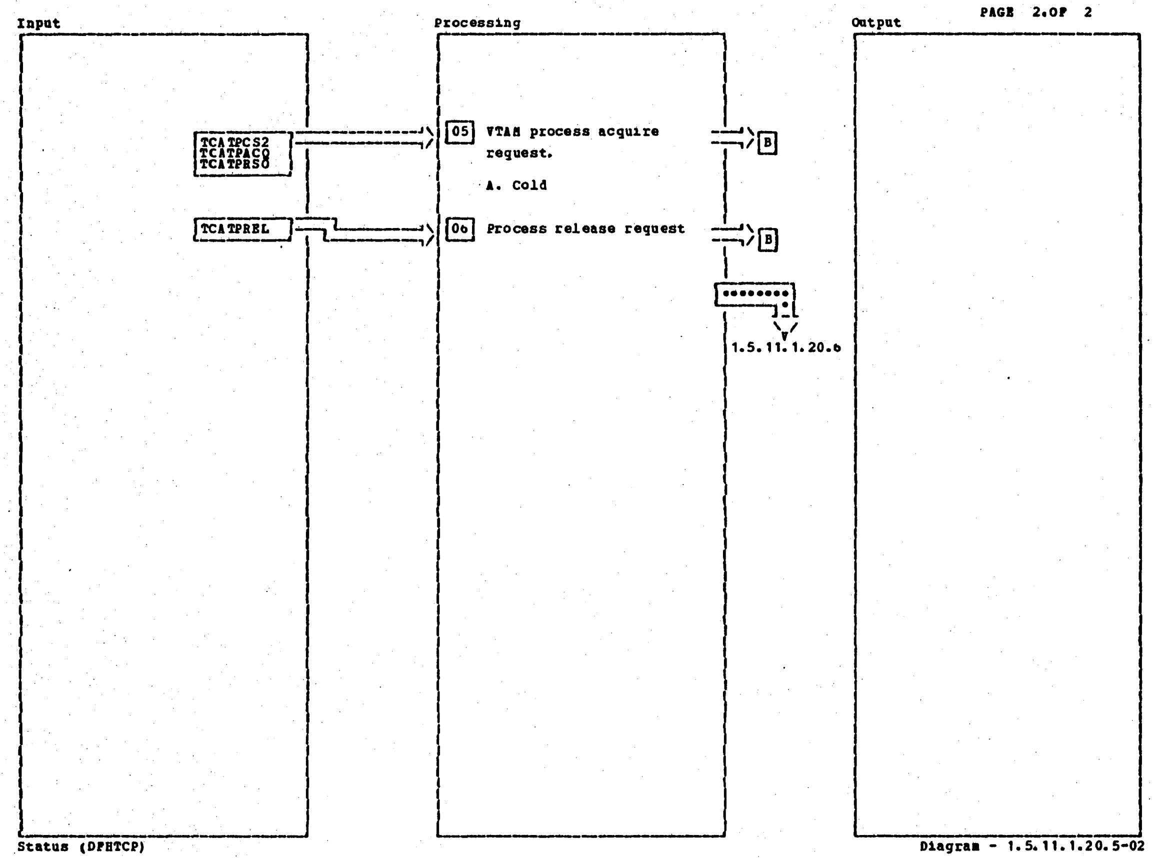


Status (DFHTCP)  Diagram - 1.5.11.1.20.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | DFHZLOC | TCZLOCSR | | | | | |
| 02 | | TCZLOCSM | | | | | |
| 03 If the paging status is changed the reverse status bit is turned off. | | TCZLOCAP | | | | | |
| 04 Control returns if the terminal is not a VTAM terminal. | | TCZLOCVA | | | | | |

Status (DFHTCP)  Diagram - 1.5.11.1.20.5-01

PAGE 2 OF 2



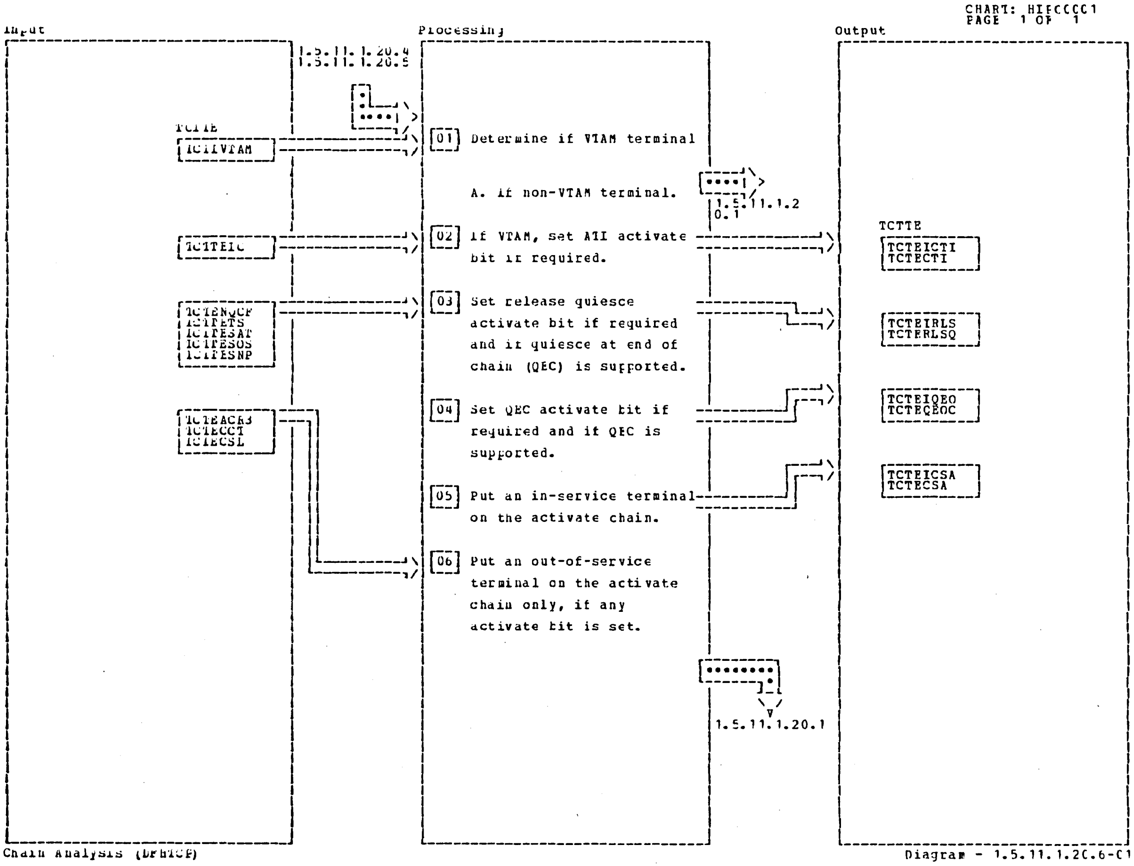
Status (DFHTCP)

Diagram - 1.5.11.1.20.5-02

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 05 The terminal is scheduled for a SIMLOGCS.<br>If Cold the previous ABEND and Resync required bits are reset. | | | |
| 06 The terminal is scheduled for a CLSDST, and any task attached is scheduled to be abnormally terminated.<br>If more than one function within a category is detected (such as transceive, input) control returns to the caller with an invalid request indicated in the return code. | | TCZLOCRL | |

Status (DFHTCP)

Diagram - 1.5.11.1.20.5-02



01 Determine if VTAM terminal

A. If non-VTAM terminal.

02 If VTAM, set ATI activate bit if required.

03 Set release quiesce activate bit if required and if quiesce at end of chain (QEC) is supported.

04 Set QEC activate bit if required and if QEC is supported.

05 Put an in-service terminal on the activate chain.

06 Put an out-of-service terminal on the activate chain only, if any activate bit is set.

Chain Analysis (DFHTCP)

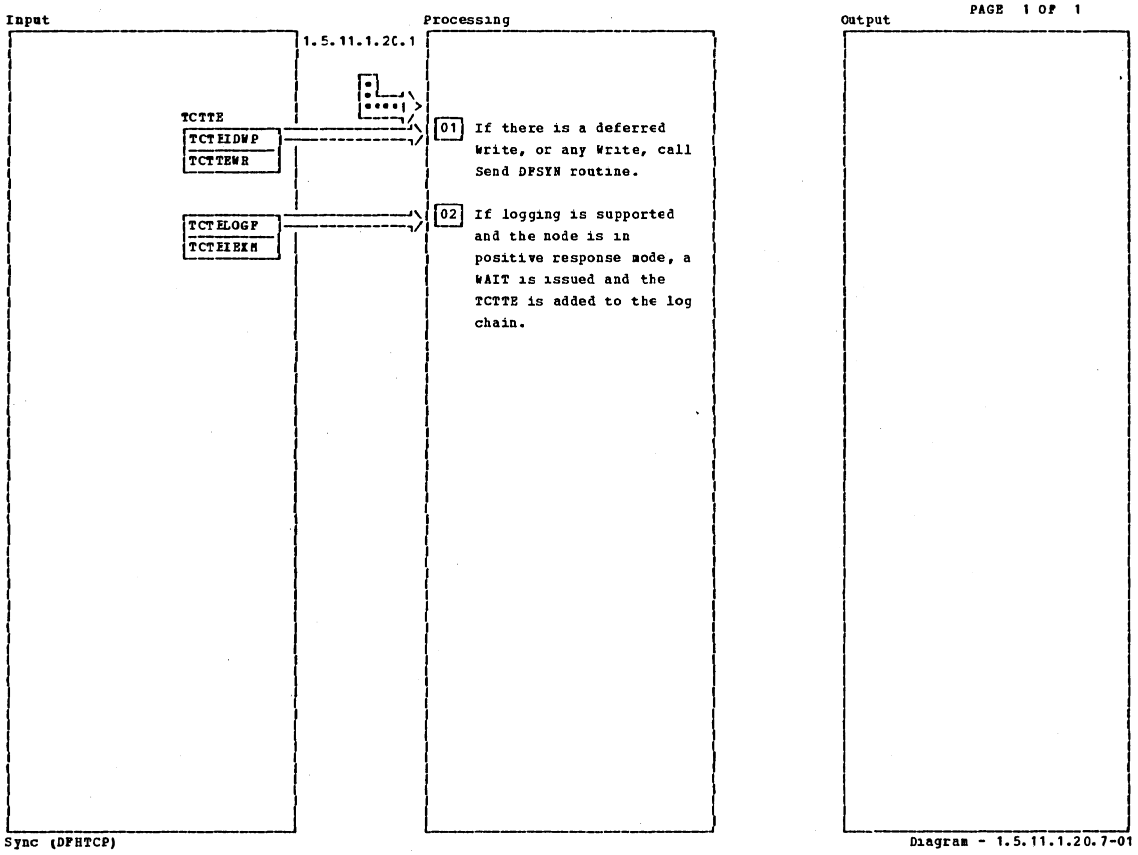
Diagram - 1.5.11.1.20.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If terminal is not a VTAM terminal control returns to the caller. | DFHZLOC | TCZLOCCH<br>TCZLOCVA | 1.5.16.1 | | | | |
| 02 Terminal in service and in a receive mode. | | TCZLOCPQ<br>TCZLOCRS | | | | | |
| 03 Terminal quiesced, in service and not in receive-only status. | | TCZLOCPQ | | | | | |
| 04 Terminal Out of Service or in Receive-Only Status and not quiesced by CICS. | | TCZLOCRQ<br>TCZLOCQO | | | | | |
| 05 | | TCZLOCQU | | | | | |
| 06 LOGON or CLSDST bits set.<br>Control then returns to the caller. | | TCZLOCQO<br>TCZLOCAQ | 1.5.16.1 | | | | |

Chain Analysis (DFHTCP)

Diagram - 1.5.11.1.20.6-01

PAGE 1 OF 1



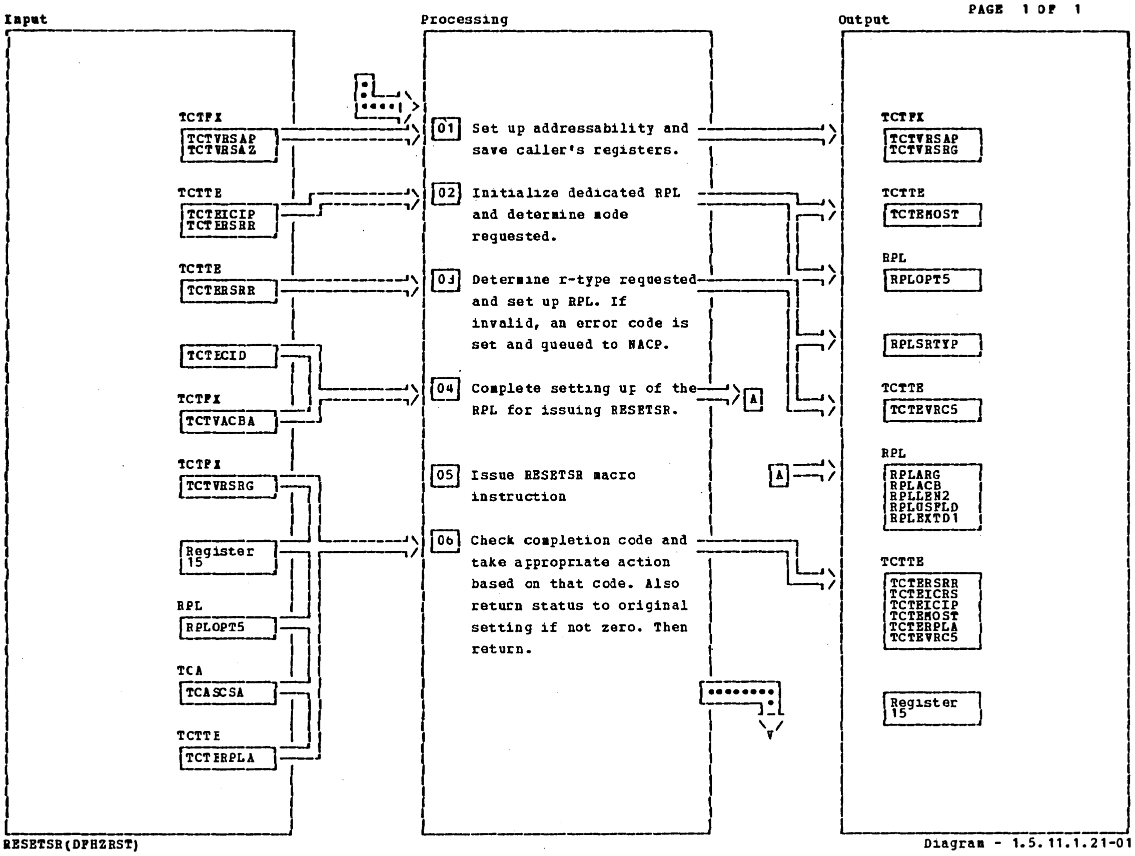
01 If there is a deferred Write, or any Write, call Send DFSYN routine.

02 If logging is supported and the node is in positive response mode, a WAIT is issued and the TCTTE is added to the log chain.

Sync (DFHTCP) Diagram - 1.5.11.1.20.7-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Send DFSYN module | DFHZSDS | | HL.05 | | | | |

Sync (DFHTCP) Diagram - 1.5.11.1.20.7-01



01 Set up addressability and save caller's registers.

02 Initialize dedicated RPL and determine mode requested.

03 Determine r-type requested and set up RPL. If invalid, an error code is set and queued to NACP.

04 Complete setting up of the RPL for issuing RESETSR.

05 Issue RESETSR macro instruction

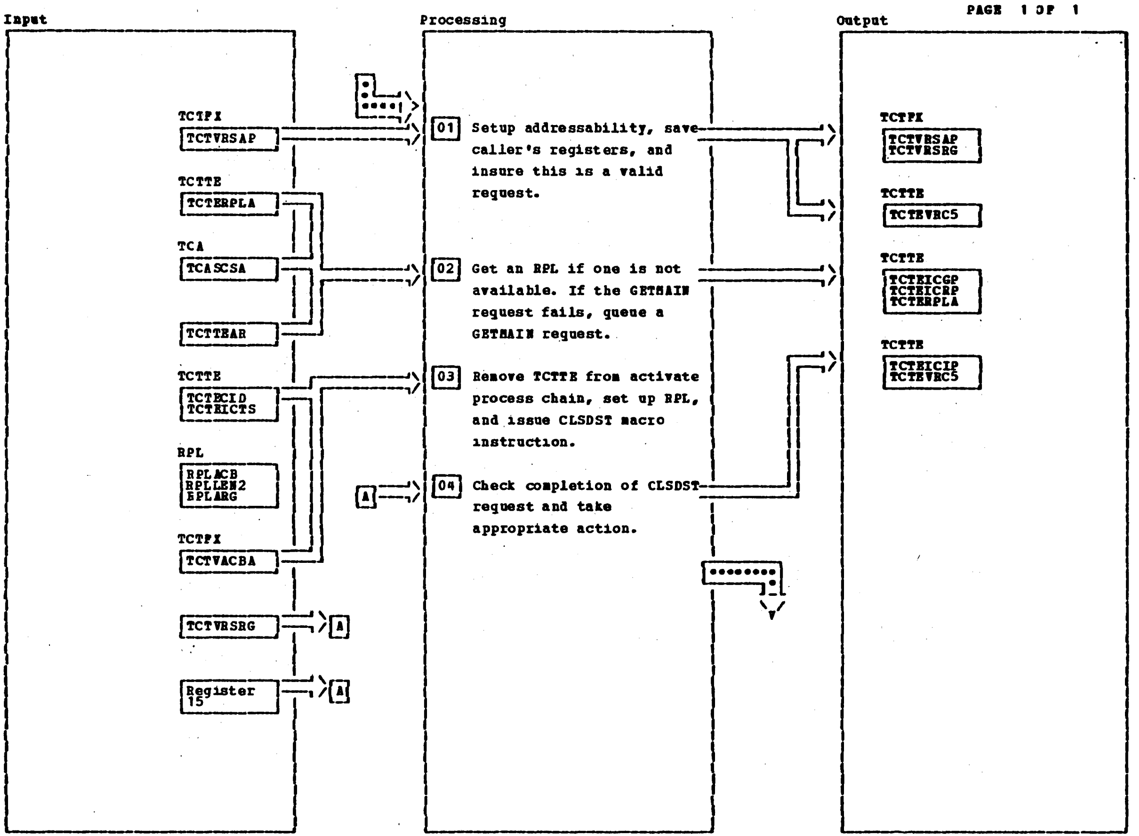
06 Check completion code and take appropriate action based on that code. Also return status to original setting if not zero. Then return.

RESETSR(DFHZRST) Diagram - 1.5.11.1.21-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 TCT prefix contains pointers and RSA. | | | | | | | |
| 02 Set RPL to zeroes. | | | | | | | |
| 03 Issue DFHTCQUE CHAIN=NACP macro instruction to place TCTTE on NACP chain. (Error queue.) | | | | | | | |
| 04 Fill in VTAM-required fields. | | | | | | | |
| 05 RESETSR changes mode status (CA-CS) and also purges any outstanding receives. | | | | | | | |
| 06 VTAM returns a completion code in register 15 indicating the success or failure of the request.<br>LERAD or SYNAD may also send a completion code.<br>Since RESETSR can be linked to, a completion code is inserted in register 15 on exit. | | | | | | | |

RESETSR(DFHZRST) Diagram - 1.5.11.1.21-01

PAGE 1 OF 1



01 Setup addressability, save caller's registers, and insure this is a valid request.

02 Get an RPL if one is not available. If the GETMAIN request fails, queue a GETMAIN request.

03 Remove TCTTE from activate process chain, set up RPL, and issue CLSDST macro instruction.

04 Check completion of CLSDST request and take appropriate action.

CLSDST Request (DFHZCLS)

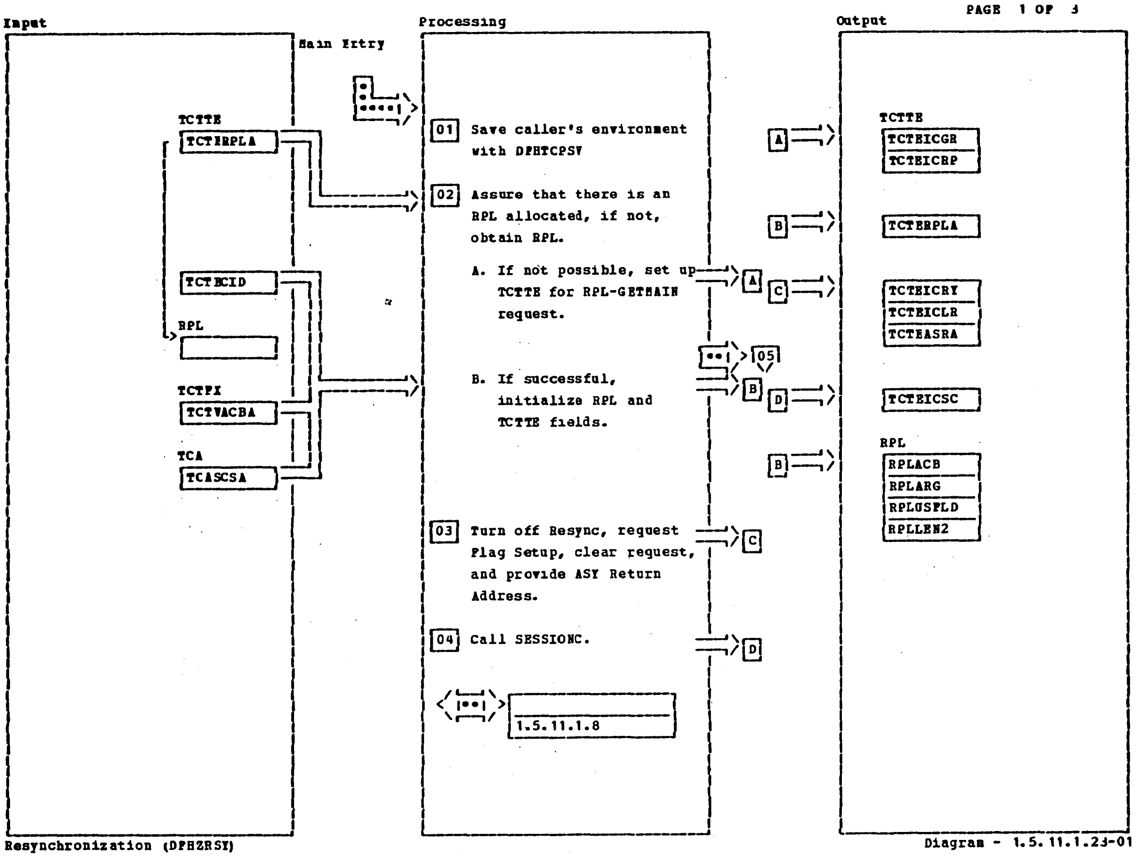
Diagram - 1.5.11.1.22-01

| Notes | Rottine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Save caller's registers and validate CLSDST request. | | | | | | | |
| 03 With issurance of the DFHTCQUE CHAIN=OFF macro instruction, the TCTTE is removed from the activate process chain. Prior to issuing the above macro instruction, the RPL is initialized to zeroes by issuing the DFHZIRPL macro instruction. | | | | | | | |
| 04 VTAM passes a completion code back in register 15. Normal completion is when register 15 = zero. If it is negative, LERAD or SYNAD was entered due to an error but has been taken care of. If it equals X'F0', VTAM experienced a temporary storage problem and the TCTTE is chained up for another attempt at CLSDST of the node. If none of the above are true, the TCTTE is passed to NACP for error processing. | | | | | | | |

CLSDST Request (DFHZCLS)

Diagram - 1.5.11.1.22-01

PAGE 1 OF 3


Input
Processing
Output
Main Entry
TCTTE
TCTERPLA
TCTECID
RPL
TCTFX
TCTVACBA
TCA
TCASCSA
01 Save caller's environment with DFHTCPSV
02 Assure that there is an RPL allocated, if not, obtain RPL.
A. If not possible, set up TCTTE for RPL-GETMAIN request.
B. If successful, initialize RPL and TCTTE fields.
03 Turn off Resync, request Flag Setup, clear request, and provide ASY Return Address.
04 Call SESSIONC.
1.5.11.1.8
05
A
B
C
D
TCTTE
TCTEICGR
TCTEICRP
TCTERPLA
TCTEICRY
TCTEICLR
TCTEASRA
TCTEICSC
RPL
RPLACB
RPLARG
RPLUSFLD
RPLLEN2

Resynchronization (DFHZRSY)

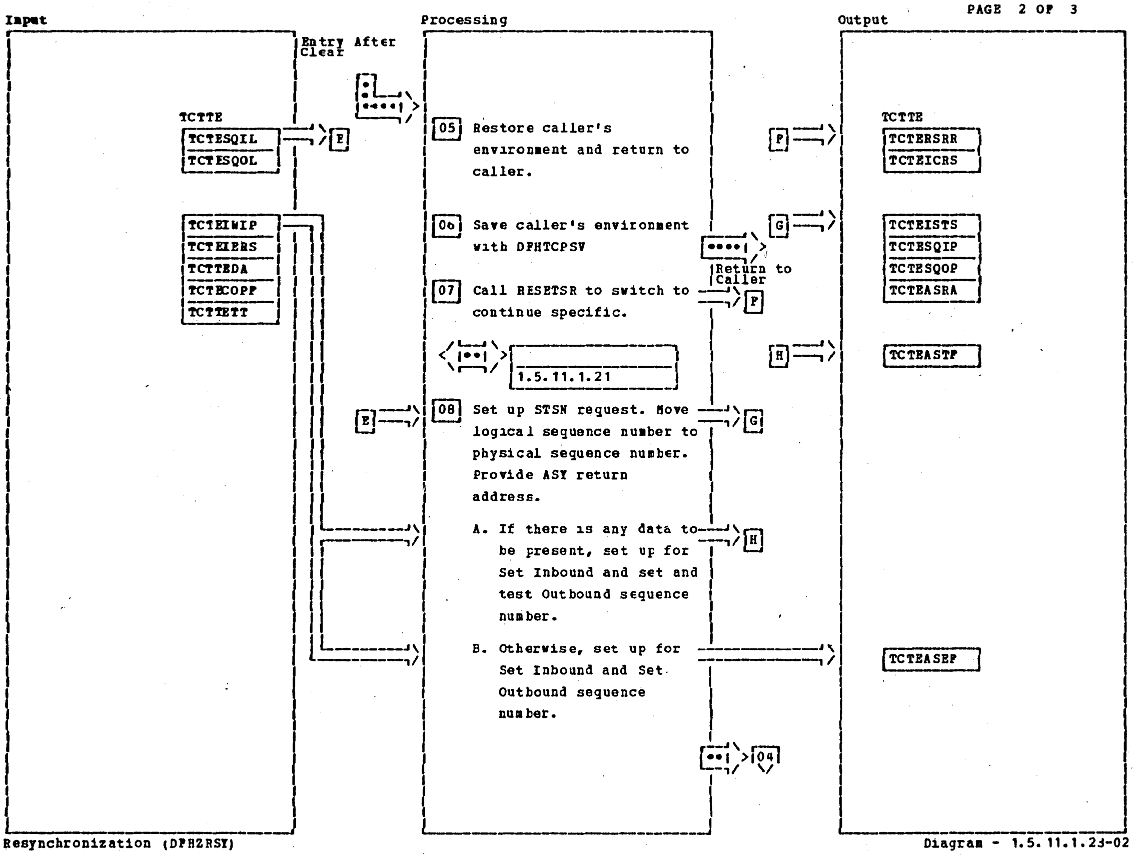
Diagram - 1.5.11.1.23-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Save caller's registers in the TCT prefix area. | | | | | | | |
| 02 An RPL must be on hand for the several VTAM requests issued in DFHZRSY. If one is not available RPL storage is required. Providing the GETMAIN is successful the RPL is formatted for the first VTAM request. If it fails, an RPL GETMAIN is queued. | | TCZRSY10<br>TCZRSY14 | | | | | |
| 03 CLEAR is the first request issued, thus that request bit is turned on in the TCTTE and the function performed. | | TCZRSY12 | | | | | |
| 04 SESSIONC is called because it issues the Clear request. | | TCZRSYSC | | | | | |

Resynchronization (DFHZRSY)

Diagram - 1.5.11.1.23-01

PAGE 2 OF 3



05 Restore caller's environment and return to caller.

06 Save caller's environment with DFHTCPSV

07 Call RESETSR to switch to continue specific.

08 Set up STSN request. Move logical sequence number to physical sequence number. Provide ASY return address.

A. If there is any data to be present, set up for Set Inbound and set and test Outbound sequence number.

B. Otherwise, set up for Set Inbound and Set Outbound sequence number.

Resynchronization (DFHZRSY)

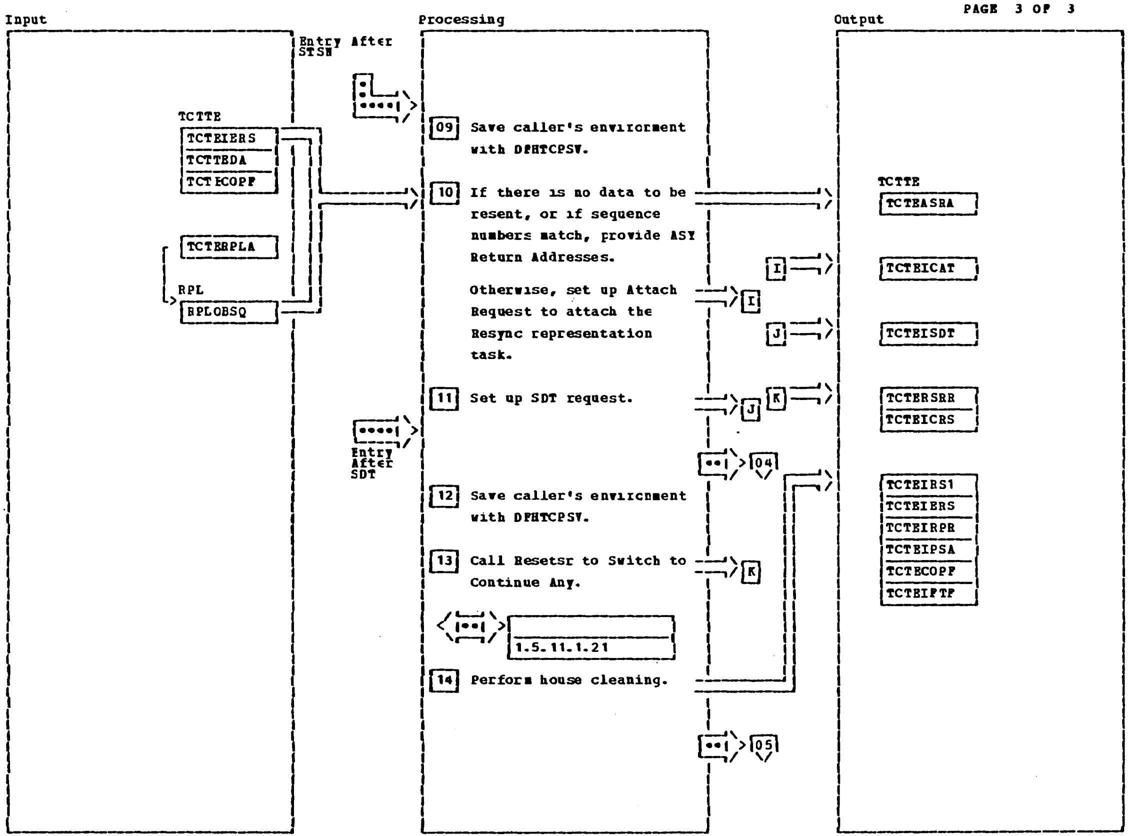
Diagram - 1.5.11.1.23-02

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 05 Upon return from issuing the Clear request DFHZRSY registers are reinstated and processing continues. | | | |
| 06 Due to asynchronous point of control implementation registers must be saved in the TCT prefix before issuing the next request. | | DFHZRSY2 | |
| 07 Resetsr (DFHZRST) must be executed to insure the node is in Continue Specific mode. | | TCZRSY20 | |
| 08 The next VTAM request is set and test sequence numbers (STSN). CICS/VS sequence numbers are set up in such a fashion as to allow the Logical Unit to compare and reply positively or negatively to these presented to him. | | TCZRSY26 | |

Resynchronization (DFHZRSY)

Diagram - 1.5.11.1.23-02

PAGE 3 OF 3



09 Save caller's enviroment with DFHTCPSV.

10 If there is no data to be resent, or if sequence numbers match, provide ASY Return Addresses.

Otherwise, set up Attach Request to attach the Resync representation task.

11 Set up SDT request.

12 Save caller's environment with DFHTCPSV.

13 Call Resetsr to Switch to Continue Any.

14 Perform house cleaning.

Resynchronization (DFHZRSY)

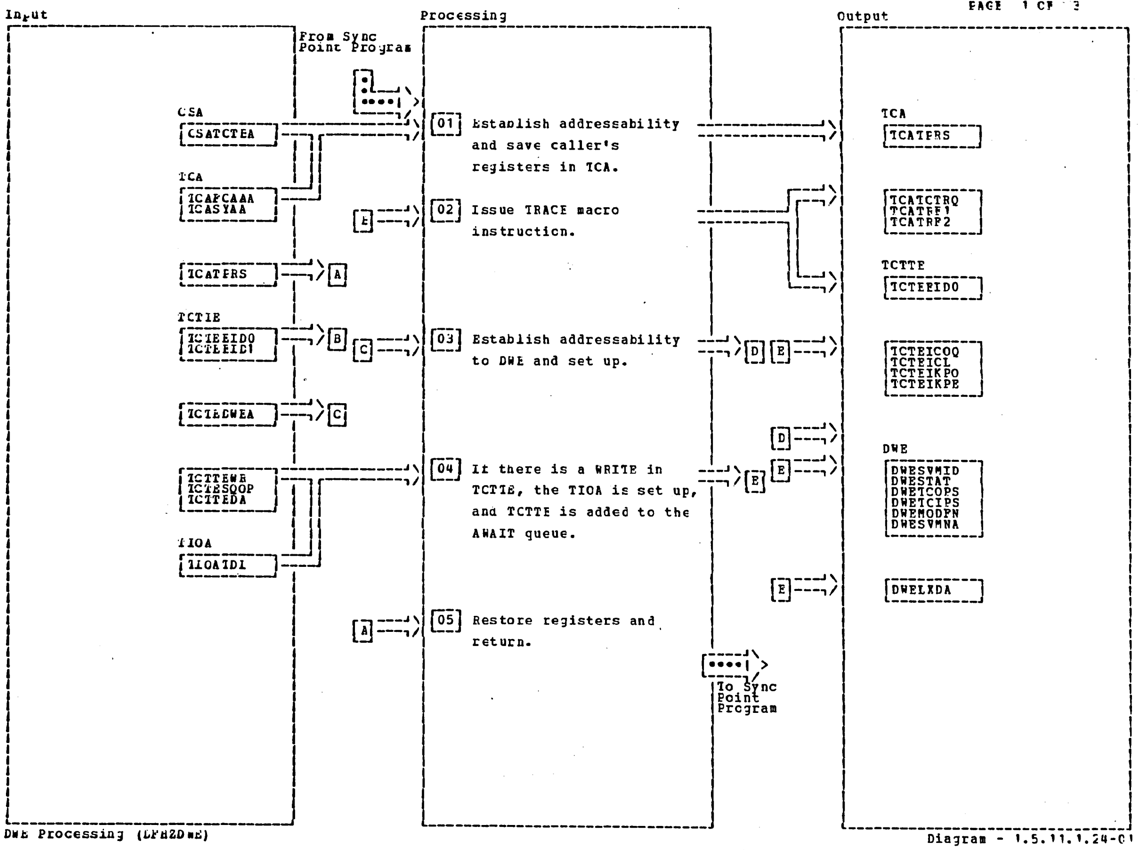
Diagram - 1.5.11.1.23-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 09 Registers are stored in the TCT prefix. | | DFHZRSY3 | | | | | |
| 10 If a TIOA is present at resynchrcnization it is represented to the Logical Unit if not a start data traffic request (SDT) is set up for the Logical Unit and the session is fully established again. | | TCZRSY30 | | | | | |
| 11 Same as note 10. | | | | | | | |
| 12 Registers are saved prior to the Resetsr request. | | DFHZRSY4 | | | | | |
| 13 To allow the Logical Unit to input data or commands the node must but switch to Continue Any (CA) Resetsr is called with a CA request indicator on. | | | | | | | |
| 14 All Bit indicators relating to resynchronization are turned off at this point because resynchronization is complete. | | TCZRSY40 | | | | | |

Resynchronization (DFHZRSY)

Diagram - 1.5.11.1.23-03

CHART: HIPCCC01
PAGE 1 OF 3



01 Establish addressability and save caller's registers in TCA.

02 Issue TRACE macro instruction.

03 Establish addressability to DWE and set up.

04 If there is a WRITE in TCTTE, the TIOA is set up, and TCTTE is added to the AWAIT queue.

05 Restore registers and return.

DWE Processing (DFHZDWE)

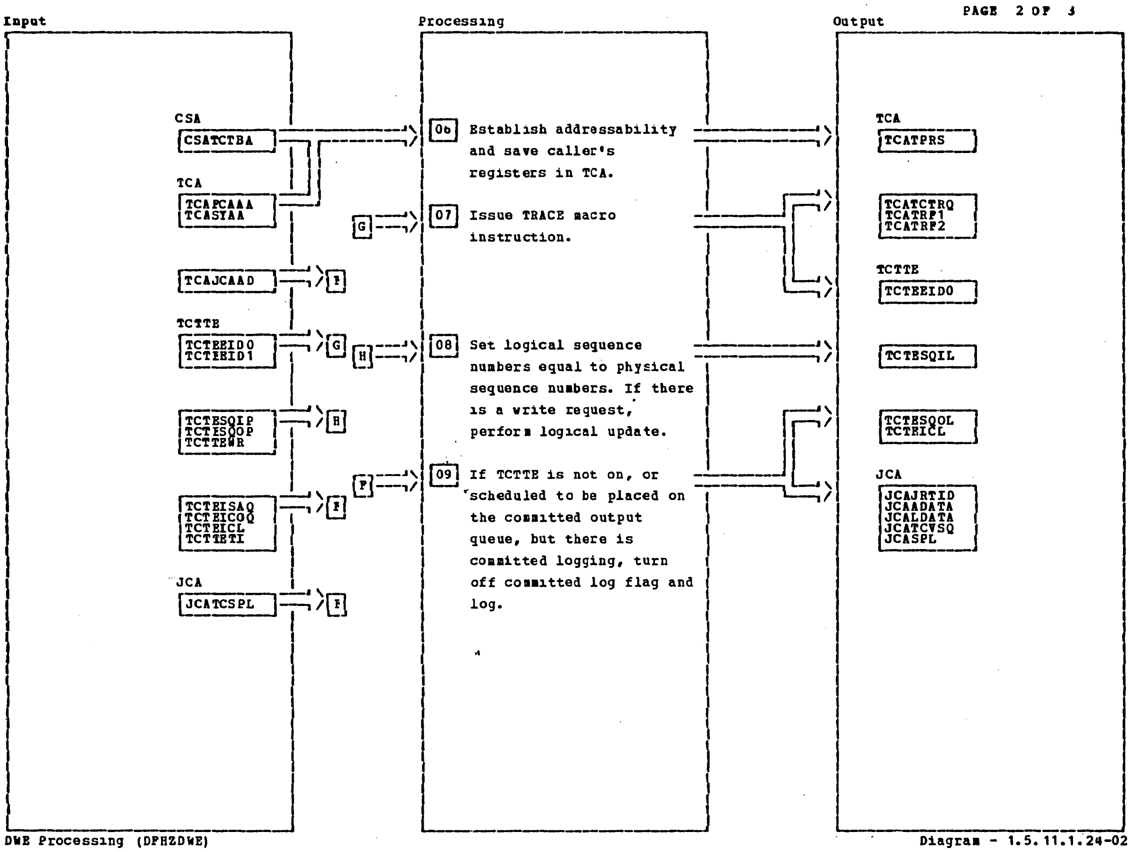
Diagram - 1.5.11.1.24-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

DWE Processing (DFHZDWE)

Diagram - 1.5.11.1.24-01

PAGE 2 OF 3



06 Establish addressability and save caller's registers in TCA.

07 Issue TRACE macro instruction.

08 Set logical sequence numbers equal to physical sequence numbers. If there is a write request, perform logical update.

09 If TCTTE is not on, or scheduled to be placed on the committed output queue, but there is committed logging, turn off committed log flag and log.

DWE Processing (DFHZDWE)

Diagram - 1.5.11.1.24-02

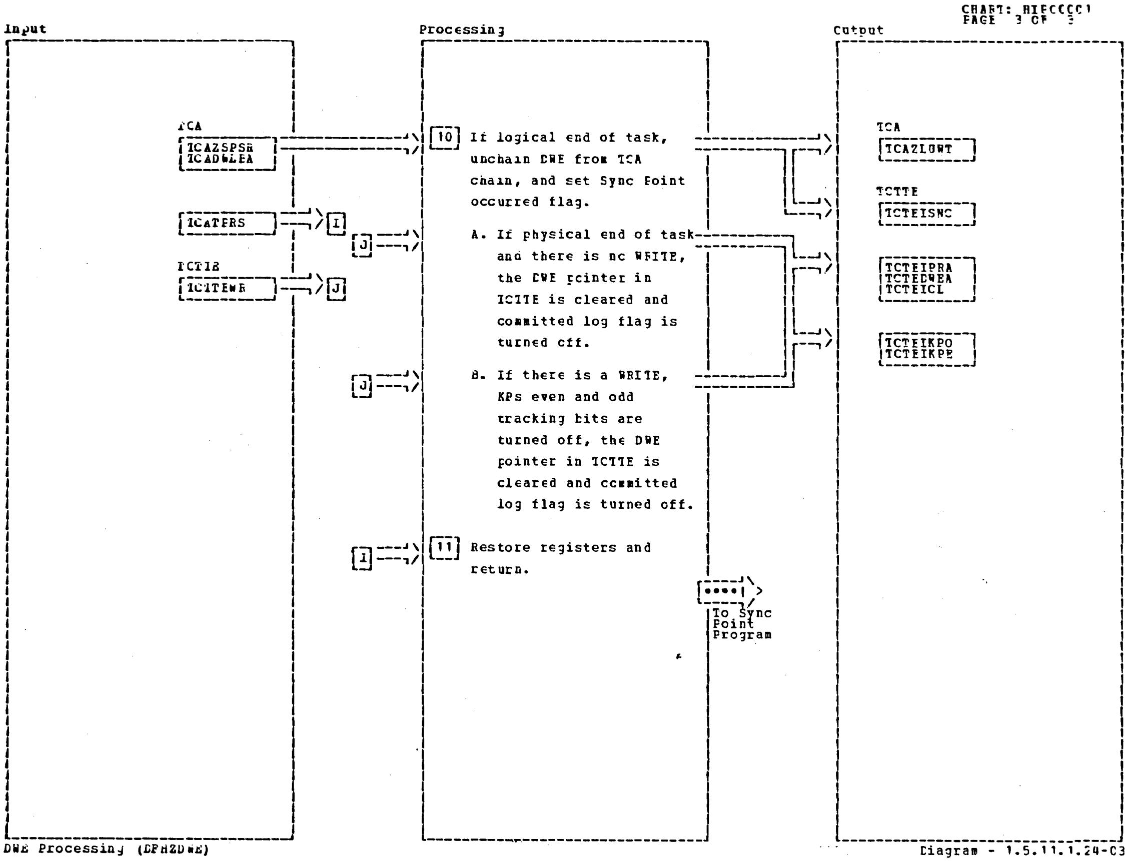
| Notes | Routine | Label | Ref |
|---|---|---|---|

| Notes | Routine | Label | Ref |
|---|---|---|---|

DWE Processing (DFHZDWE)

Diagram - 1.5.11.1.24-02

CHART: HIECCCC1
PAGE 3 OF 3



10 If logical end of task, unchain DWE from TCA chain, and set Sync Point occurred flag.

A. If physical end of task and there is no WRITE, the DWE pointer in TCTTE is cleared and committed log flag is turned off.

B. If there is a WRITE, KPs even and odd tracking bits are turned off, the DWE pointer in TCTTE is cleared and committed log flag is turned off.

11 Restore registers and return.

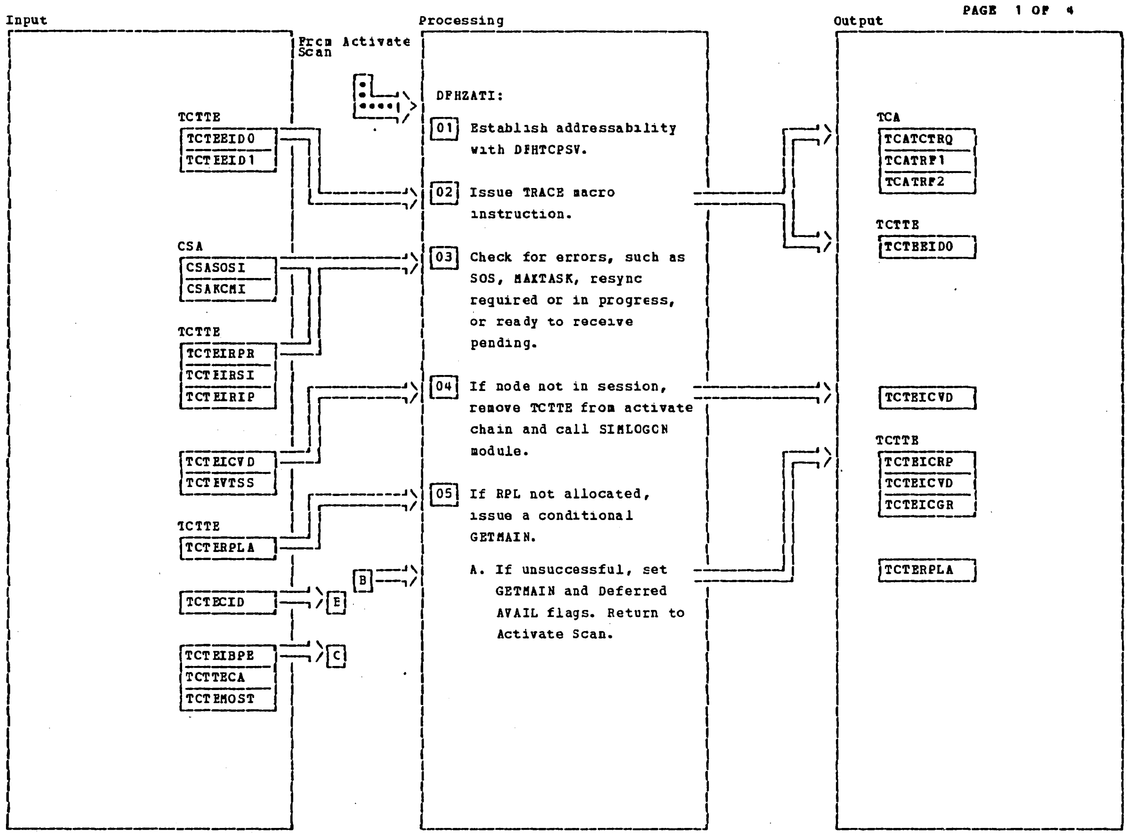
DWE Processing (DFHZDWE)

Diagram - 1.5.11.1.24-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

DWE Processing (DFHZDWE)

Diagram - 1.5.11.1.24-03



DFHZATI:

01 Establish addressability with DFHTCPSV.

02 Issue TRACE macro instruction.

03 Check for errors, such as SOS, MAXTASK, resync required or in progress, or ready to receive pending.

04 If node not in session, remove TCTTE from activate chain and call SIMLOGON module.

05 If RPL not allocated, issue a conditional GETMAIN.

A. If unsuccessful, set GETMAIN and Deferred AVAIL flags. Return to Activate Scan.

Automatic Task Initialization(DFHZATI)

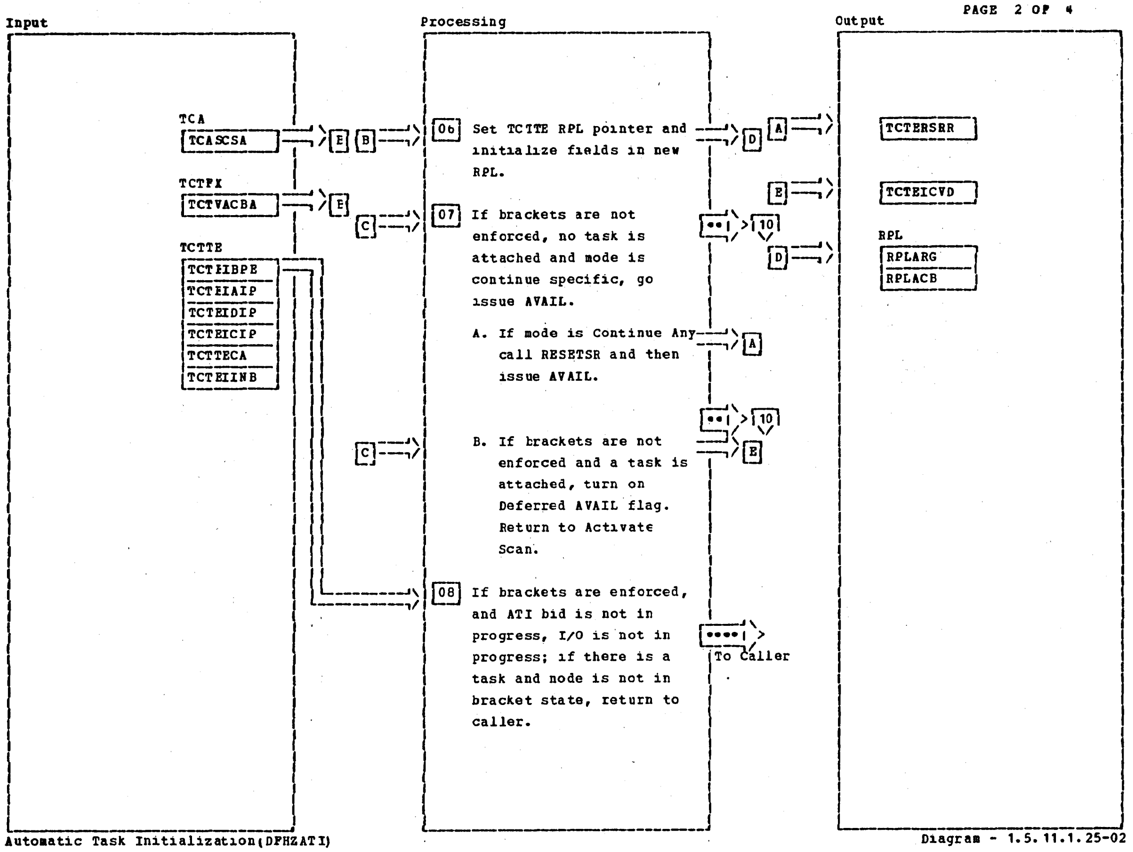
Diagram - 1.5.11.1.25-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 If SOS or MAXTASK Indicators are on, or if resync is required or in progress, return code 8 is set and control is returned to Activate Scan. If ready to receive is pendirg, TCTTE is taken off chain. |  | TCZATI92 | 1.5.11.1.5 |  |  |  |  |
| 04 | DFHZSIM |  | 1.5.11.1.6 |  |  |  |  |

Automatic Task Initialization(DFHZATI)

Diagram - 1.5.11.1.25-01

PAGE 2 OF 4



06 Set TCTTE RPL pointer and initialize fields in new RPL.

07 If brackets are not enforced, no task is attached and mode is continue specific, go issue AVAIL.

A. If mode is Continue Any call RESETSR and then issue AVAIL.

B. If brackets are not enforced and a task is attached, turn on Deferred AVAIL flag. Return to Activate Scan.

08 If brackets are enforced, and ATI bid is not in progress, I/O is not in progress; if there is a task and node is not in bracket state, return to caller.

Automatic Task Initialization(DFHZATI)

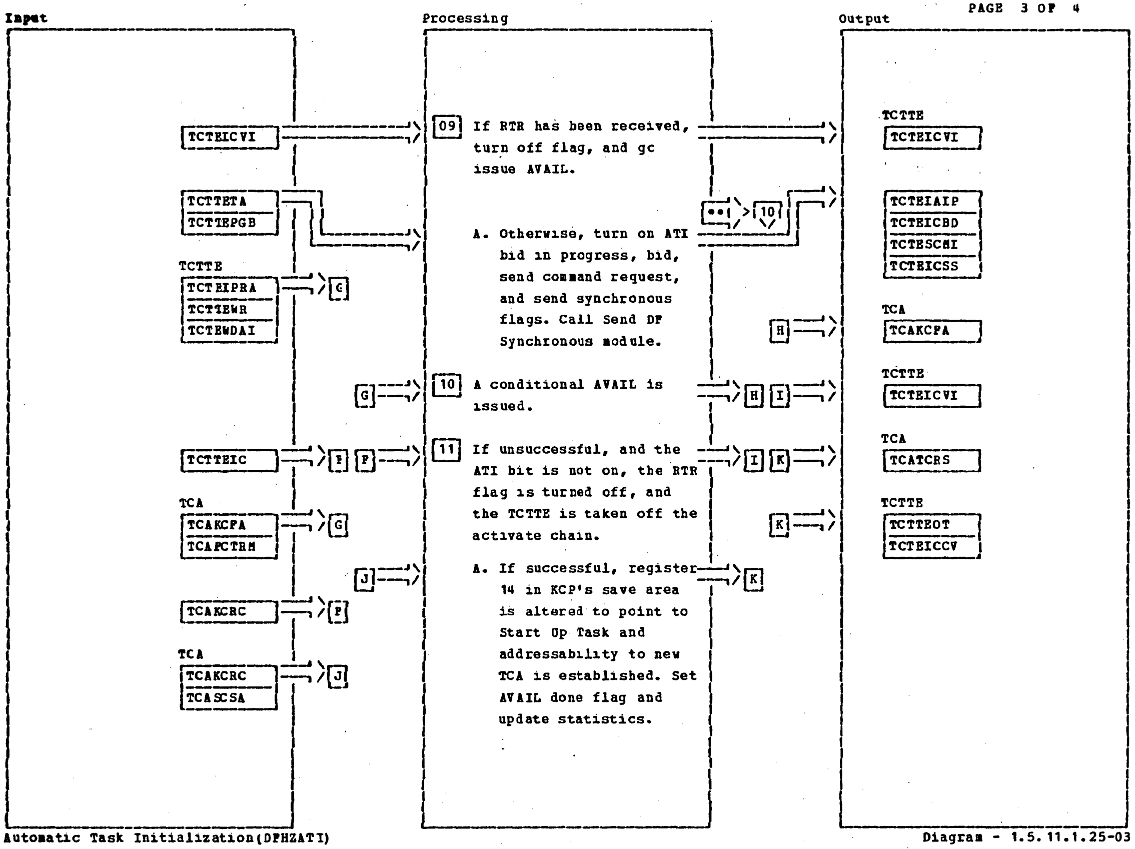
Diagram - 1.5.11.1.25-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 06 If RPL available, go on to step 7. | | | | | | | |
| 07 Send DF Sync module. | DFHZSDS | | 1.5.11.1.17 | | | | |

Automatic Task Initialization(DFHZATI)

Diagram - 1.5.11.1.25-02

PAGE 3 OF 4



09 If RTR has been received, turn off flag, and gc issue AVAIL.

A. Otherwise, turn on ATI bid in progress, bid, send command request, and send synchronous flags. Call Send DF Synchronous module.

10 A conditional AVAIL is issued.

11 If unsuccessful, and the ATI bit is not on, the RTR flag is turned off, and the TCTTE is taken off the activate chain.

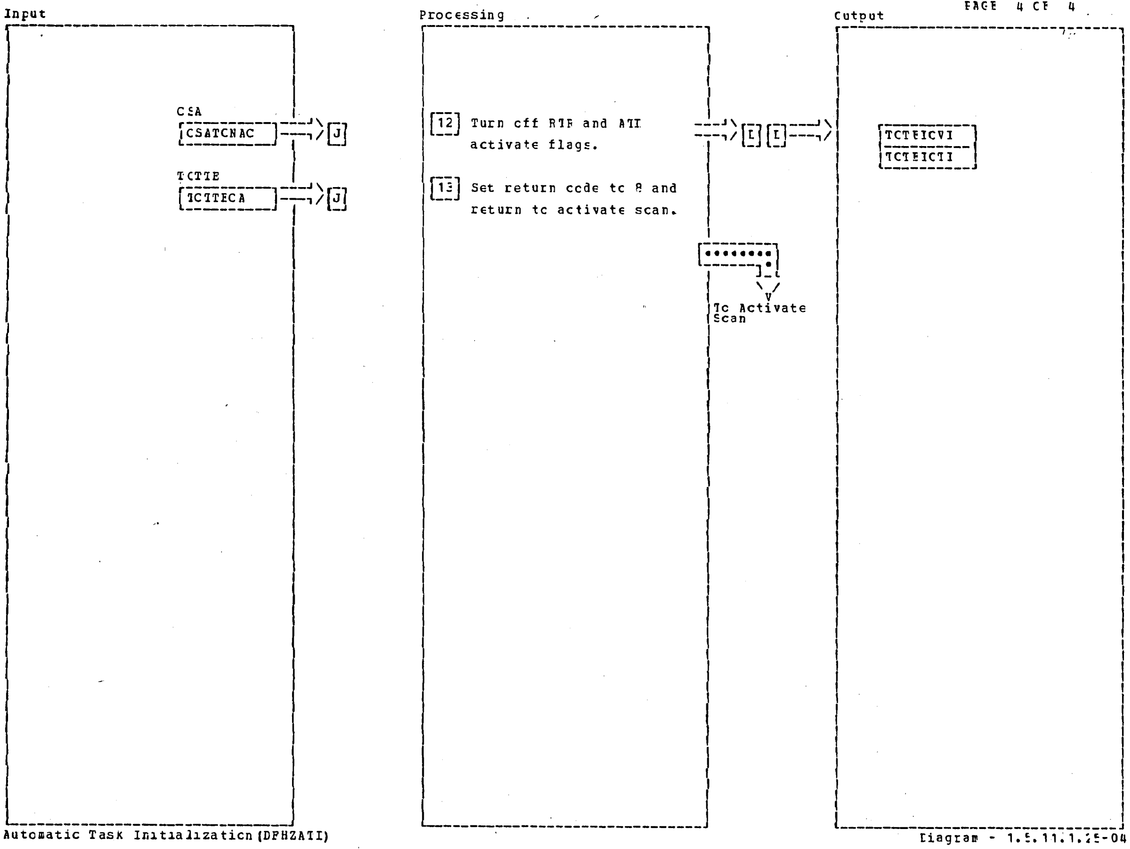
A. If successful, register 14 in KCP's save area is altered to point to Start Up Task and addressability to new TCA is established. Set AVAIL done flag and update statistics.

Automatic Task Initialization(DFHZATI) Diagram - 1.5.11.1.25-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 11 If ATI bit is cn, return code 8 is set and control is returned to Activate Scan. | | TCZATI92 | 1.5.11 .1.5 | | | | |
| A. Startup Task | DFHZSUP | | | | | | |

Automatic Task Initialization(DFHZATI) Diagram - 1.5.11.1.25-03

CHART: DFHZATI
PAGE 4 OF 4



12 Turn off RTR and ATI activate flags.

13 Set return code to 8 and return to activate scan.

Automatic Task Initialization (DFHZATI)

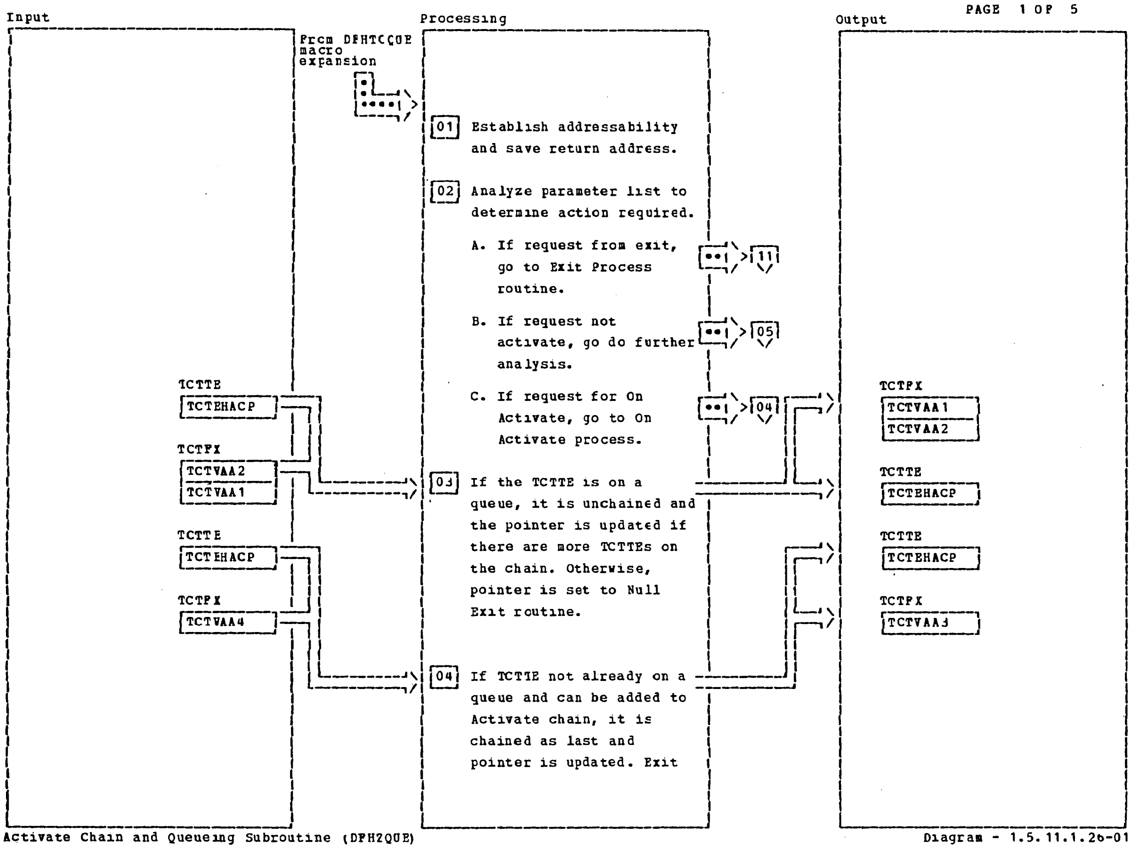
Diagram - 1.5.11.1.25-04

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Automatic Task Initialization (DFHZATI)

Diagram - 1.5.11.1.25-04

PAGE 1 OF 5



01 Establish addressability and save return address.

02 Analyze parameter list to determine action required.

A. If request from exit, go to Exit Process routine.

B. If request not activate, go do further analysis.

C. If request for On Activate, go to On Activate process.

03 If the TCTTE is on a queue, it is unchained and the pointer is updated if there are more TCTTEs on the chain. Otherwise, pointer is set to Null Exit routine.

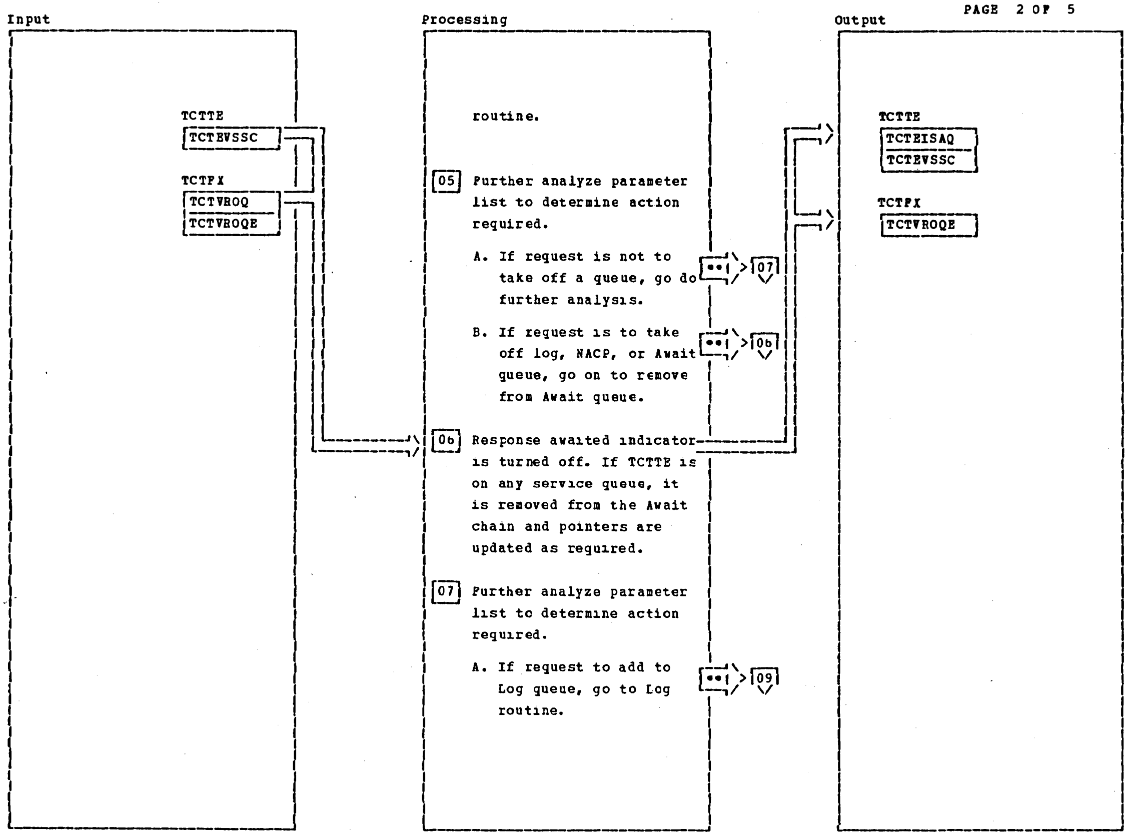
04 If TCTTE not already on a queue and can be added to Activate chain, it is chained as last and pointer is updated. Exit

Activate Chain and Queueing Subroutine (DFHZQUE) Diagram - 1.5.11.1.26-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 The expansion of the DFHTCQUE macro instruction creates a one-word in-line parameter list that contains a bit string to control the functions in this module. | | | | | | | |

Activate Chain and Queueing Subroutine (DFHZQUE) Diagram - 1.5.11.1.26-01

PAGE 2 OF 5



routine.

05 Further analyze parameter list to determine action required.

A. If request is not to take off a queue, go do further analysis.

B. If request is to take off log, NACP, or Await queue, go on to remove from Await queue.

06 Response awaited indicator is turned off. If TCTTE is on any service queue, it is removed from the Await chain and pointers are updated as required.

07 Further analyze parameter list to determine action required.

A. If request to add to Log queue, go to Log routine.

Activate Chain and Queueing Subroutine (DFHZQUE)

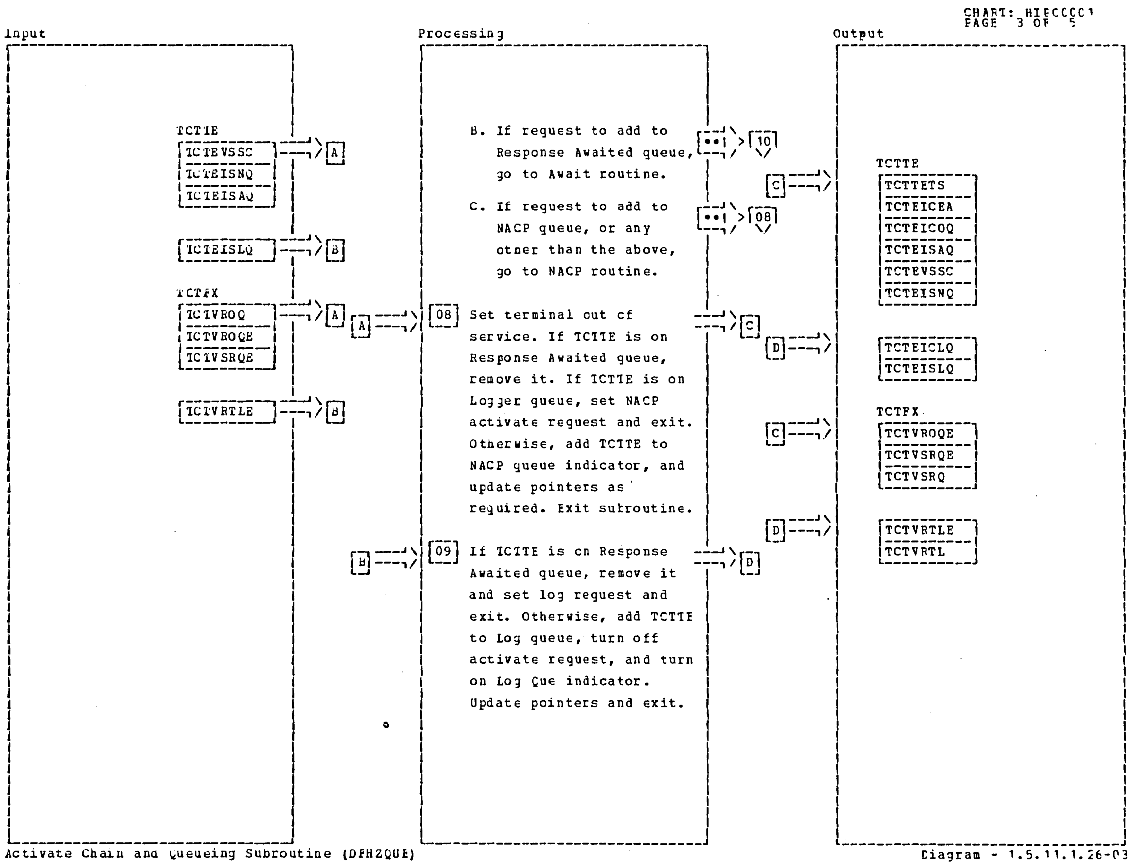
Diagram - 1.5.11.1.26-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 Only Await queue is currently implemented. | | | | | | | |

Activate Chain and Queueing Subroutine (DFHZQUE)

Diagram - 1.5.11.1.26-02

CHART: HIECCCC1
PAGE 3 OF 5



B. If request to add to Response Awaited queue, go to Await routine.

C. If request to add to NACP queue, or any other than the above, go to NACP routine.

08 Set terminal out of service. If TCTTE is on Response Awaited queue, remove it. If TCTTE is on Logger queue, set NACP activate request and exit. Otherwise, add TCTTE to NACP queue indicator, and update pointers as required. Exit subroutine.

09 If TCTTE is on Response Awaited queue, remove it and set log request and exit. Otherwise, add TCTTE to Log queue, turn off activate request, and turn on Log Que indicator. Update pointers and exit.

Activate Chain and Queueing Subroutine (DFHZQUE)

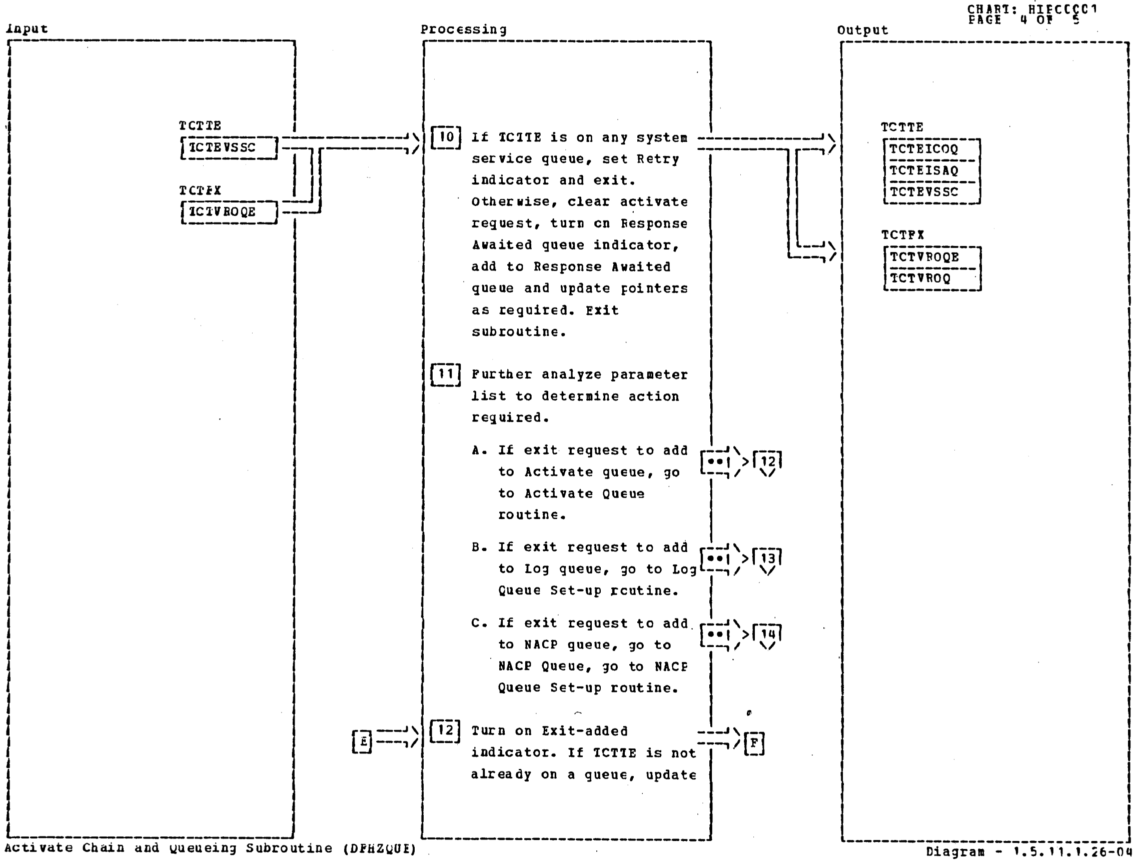
Diagram - 1.5.11.1.26-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Activate Chain and Queueing Subroutine (DFHZQUE)

Diagram - 1.5.11.1.26-03

CHART: HIECCCC1
PAGE 4 OF 5



10 If TCTTE is on any system service queue, set Retry indicator and exit. Otherwise, clear activate request, turn cn Response Awaited queue indicator, add to Response Awaited queue and update pointers as required. Exit subroutine.

11 Further analyze parameter list to determine action required.

A. If exit request to add to Activate queue, go to Activate Queue routine.

B. If exit request to add to Log queue, go to Log Queue Set-up routine.

C. If exit request to add to NACP queue, go to NACP Queue Set-up routine.

12 Turn on Exit-added indicator. If TCTTE is not already on a queue, update

Activate Chain and Queueing Subroutine (DFHZQUE)

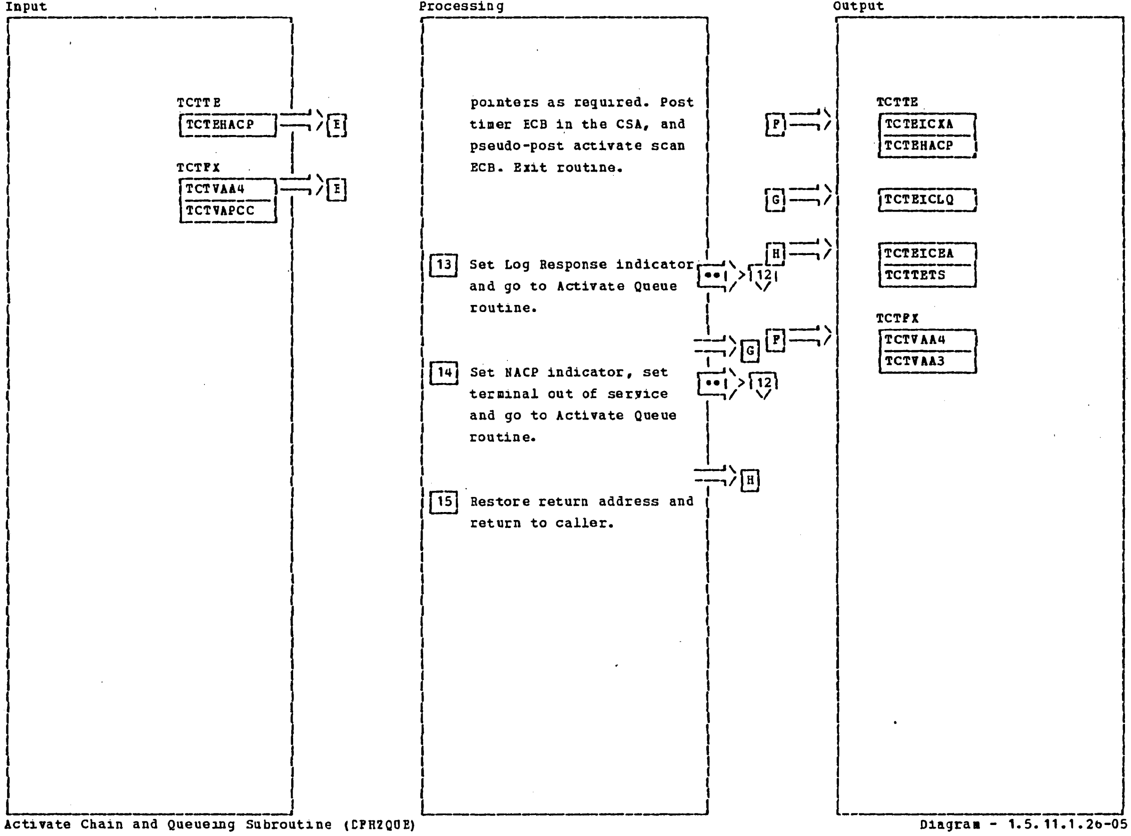
Diagram - 1.5.11.1.26-04

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

Activate Chain and Queueing Subroutine (DFHZQUE)

Diagram - 1.5.11.1.26-04

PAGE 5 OF 5



pointers as required. Post timer ECB in the CSA, and pseudo-post activate scan ECB. Exit routine.

13 Set Log Response indicator and go to Activate Queue routine.

14 Set NACP indicator, set terminal out of service and go to Activate Queue routine.

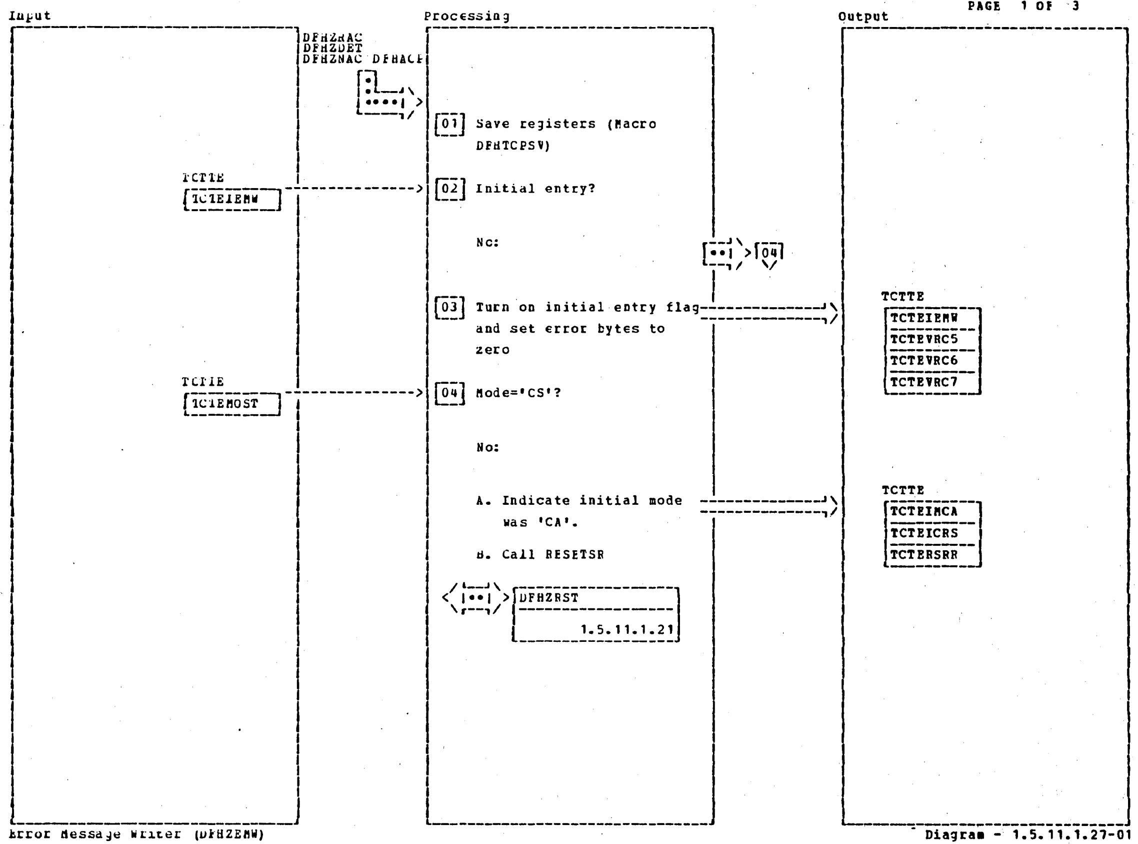
15 Restore return address and return to caller.

Activate Chain and Queueing Subroutine (DFHZQUE) Diagram - 1.5.11.1.26-05

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 15 If ECB was posted (step 12), bypass reload of register 14 from register 0, which is volatile across system post macro instruction. | | | | | | | |

Activate Chain and Queueing Subroutine (DFHZQUE) Diagram - 1.5.11.1.26-05

CHART: HIECC001
PAGE 1 OF 3



01 Save registers (Macro DFHTCPSV)

02 Initial entry?

No:

03 Turn on initial entry flag and set error bytes to zero

04 Mode='CS'?

No:

A. Indicate initial mode was 'CA'.

B. Call RESETSR

Error Message Writer (DFHZEMW)

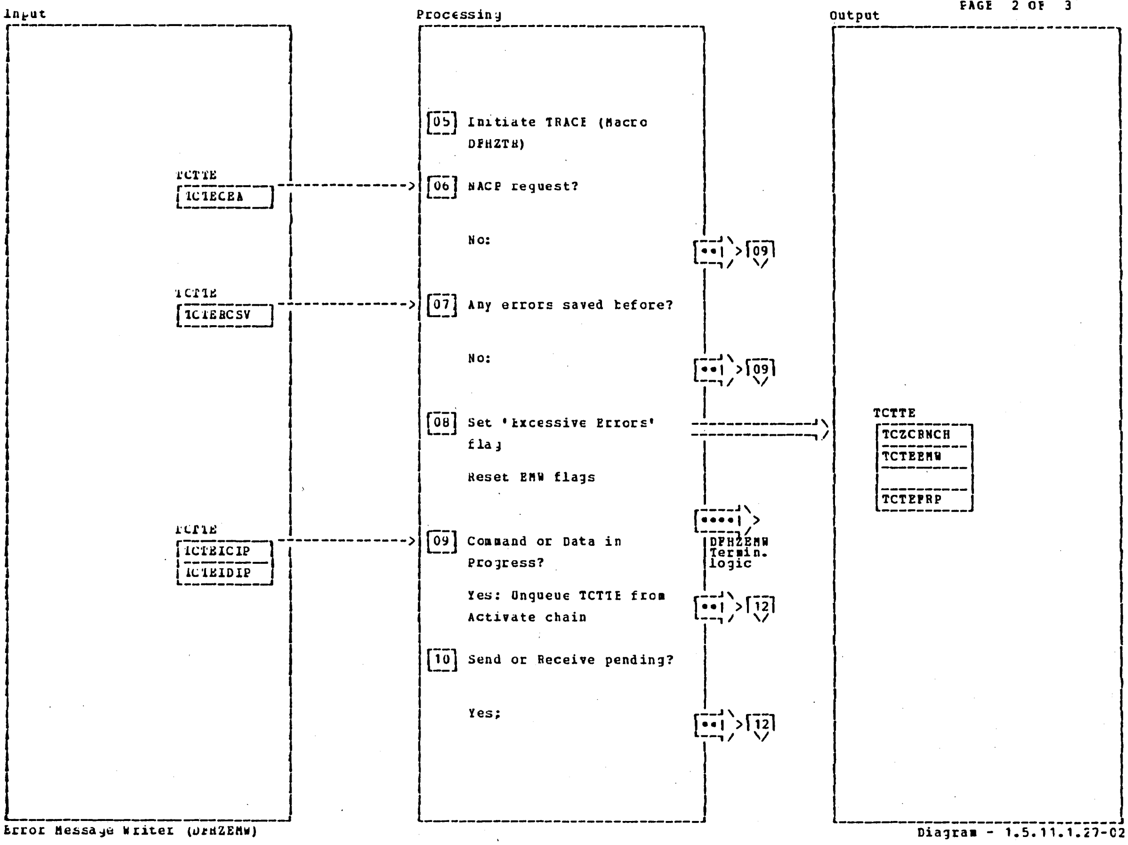
Diagram - 1.5.11.1.27-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

Error Message Writer (DFHZEMW)

Diagram - 1.5.11.1.27-01

CHART: HIECCOC1
PAGE 2 OF 3


Input
Processing
Output
TCTTE
TCTECEA
TCTTE
TCTEECSV
TCTTE
TCTEICIP
TCTEIDIP
05 Initiate TRACE (Macro DFHZTR)
06 NACP request?
No:
•• 09
07 Any errors saved before?
No:
•• 09
08 Set 'Excessive Errors' flag
Reset EMW flags
•••• DFHZEMW Termin. logic
09 Command or Data in Progress?
Yes: Unqueue TCTTE from Activate chain
•• 12
10 Send or Receive pending?
Yes:
•• 12
TCTTE
TCZCBNCH
TCTEEMW
TCTEFRP

Error Message Writer (DFHZEMW)

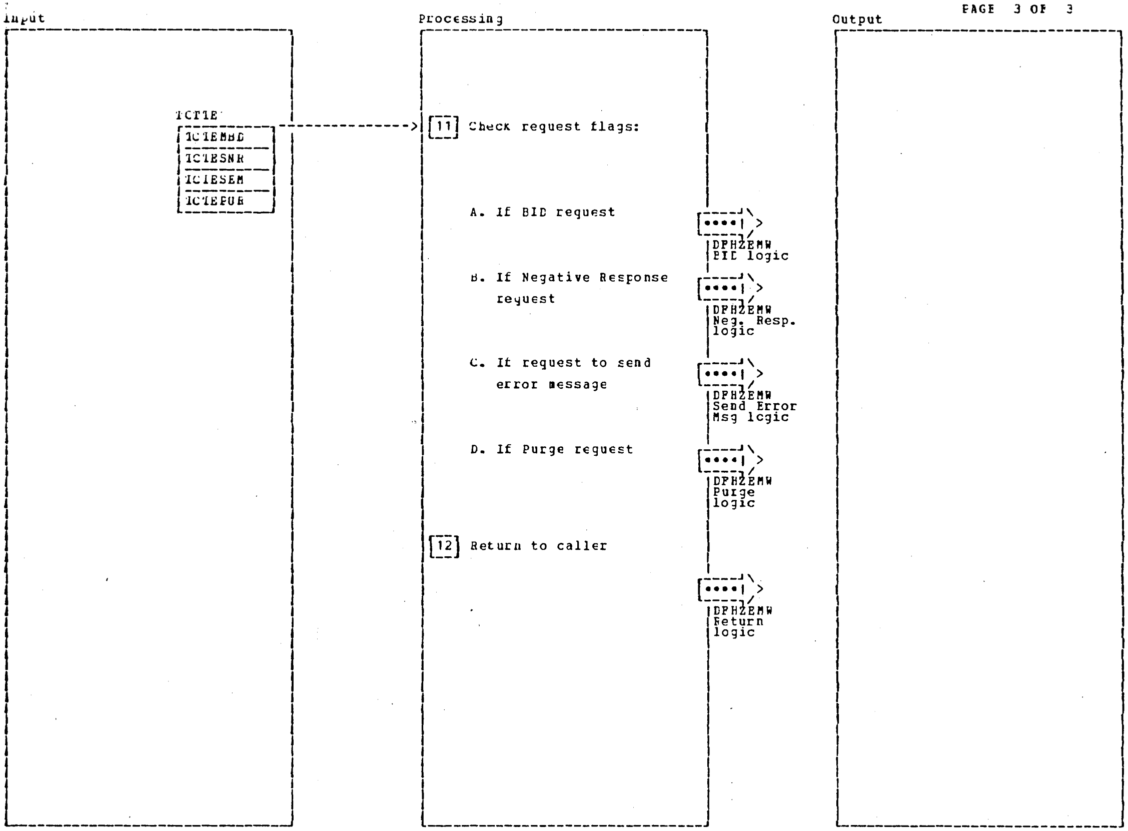
Diagram - 1.5.11.1.27-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Error Message Writer (DFHZEMW)

Diagram - 1.5.11.1.27-02

PAGE 3 OF 3



[11] Check request flags:

A. If BID request

B. If Negative Response request

C. If request to send error message

D. If Purge request

[12] Return to caller

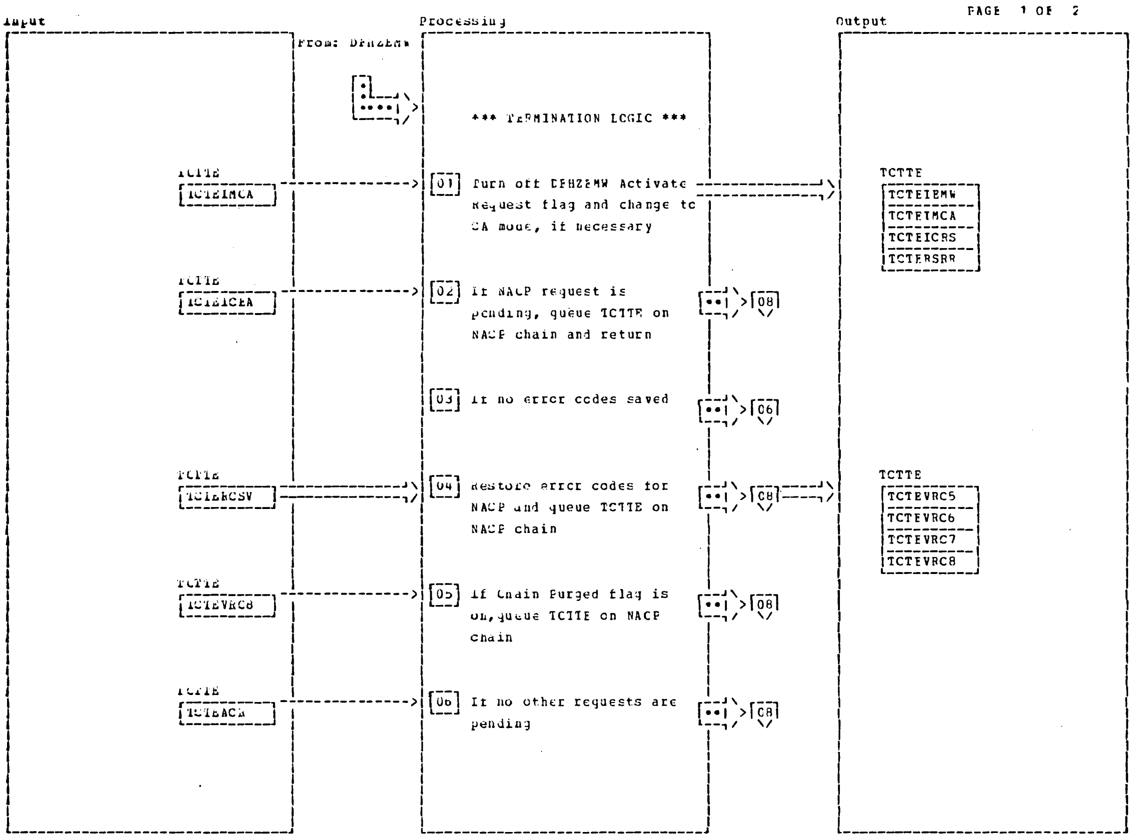
Error Message Writer (DFHZEMW)

Diagram - 1.5.11.1.27-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Error Message Writer (DFHZEMW)

Diagram - 1.5.11.1.27-03



*** TERMINATION LOGIC ***

01 Turn off DFHZEMW Activate Request flag and change to CA mode, if necessary

02 If NACP request is pending, queue TCTTE on NACP chain and return

03 If no error codes saved

04 Restore error codes for NACP and queue TCTTE on NACP chain

05 If Chain Purged flag is on, queue TCTTE on NACP chain

06 If no other requests are pending

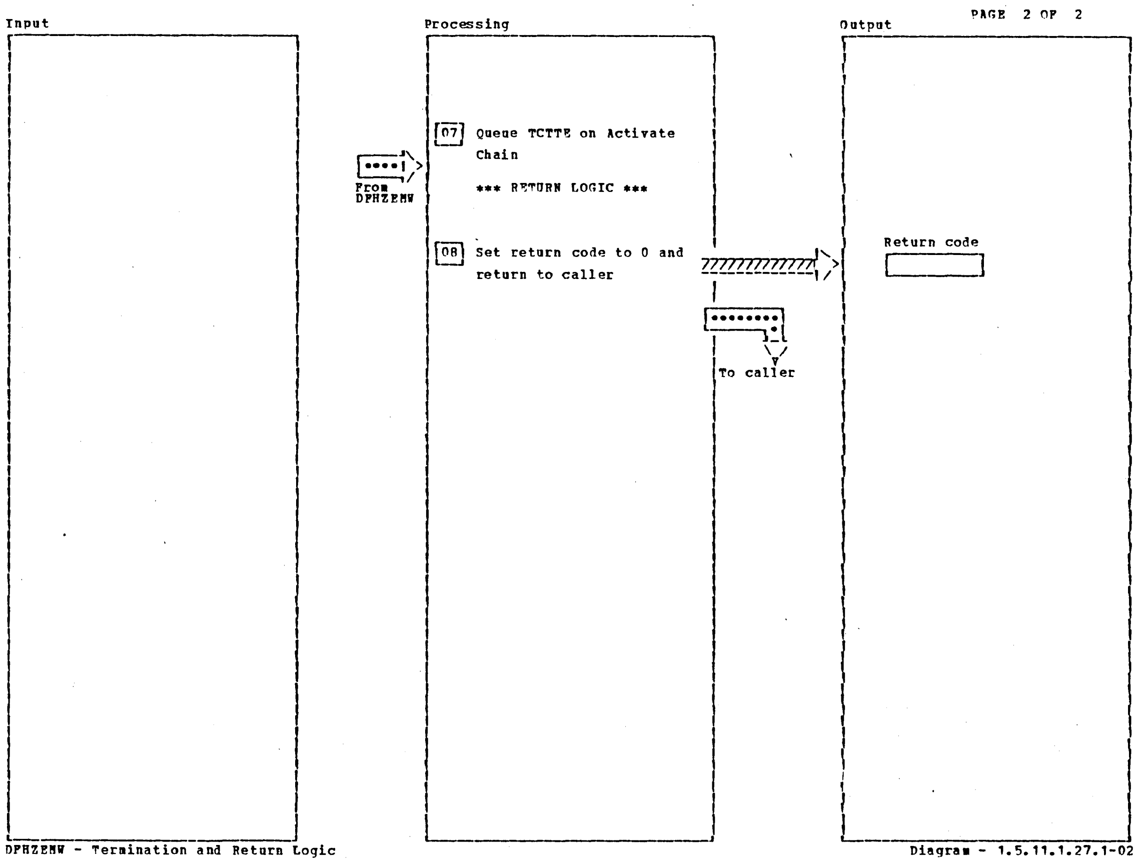
DFHZEMW - Termination and Return Logic

Diagram - 1.5.11.1.27.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

DFHZEMW - Termination and Return Logic

Diagram - 1.5.11.1.27.1-01



07 Queue TCTTE on Activate Chain

*** RETURN LOGIC ***

08 Set return code to 0 and return to caller

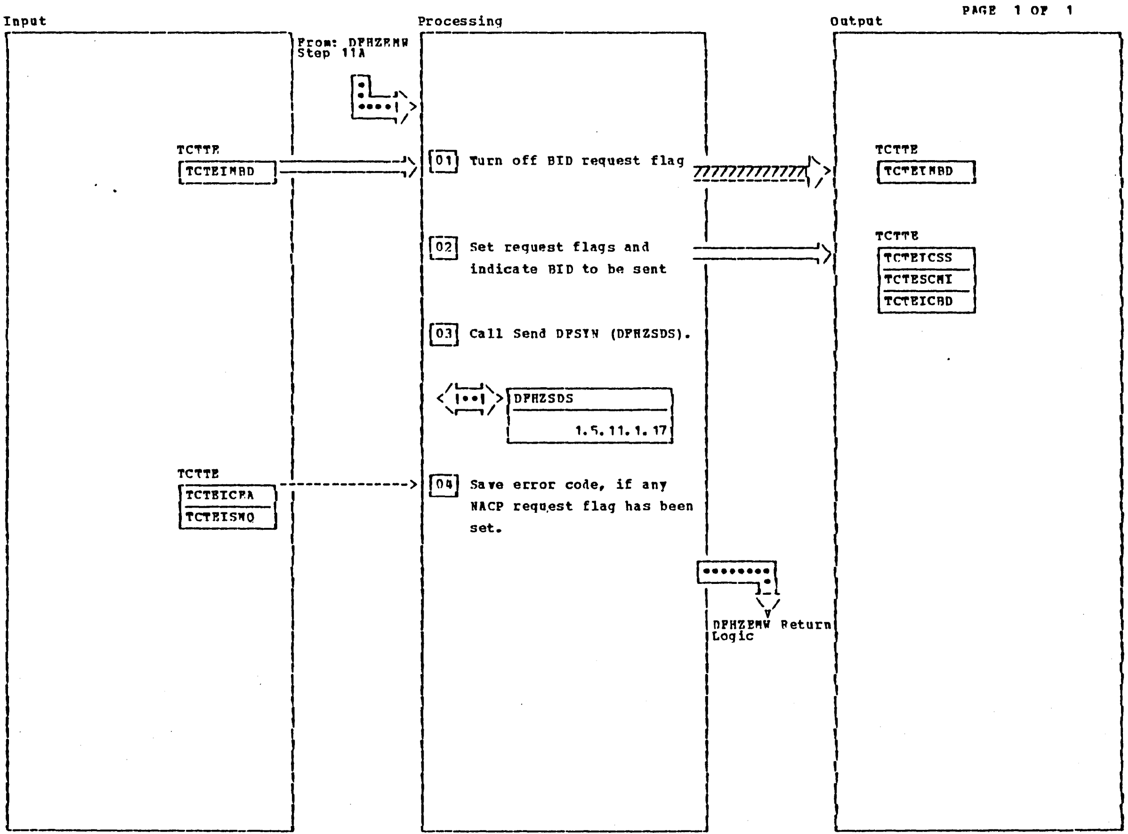
DFHZEMW - Termination and Return Logic

Diagram - 1.5.11.1.27.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

DFHZEMW - Termination and Return Logic

Diagram - 1.5.11.1.27.1-02



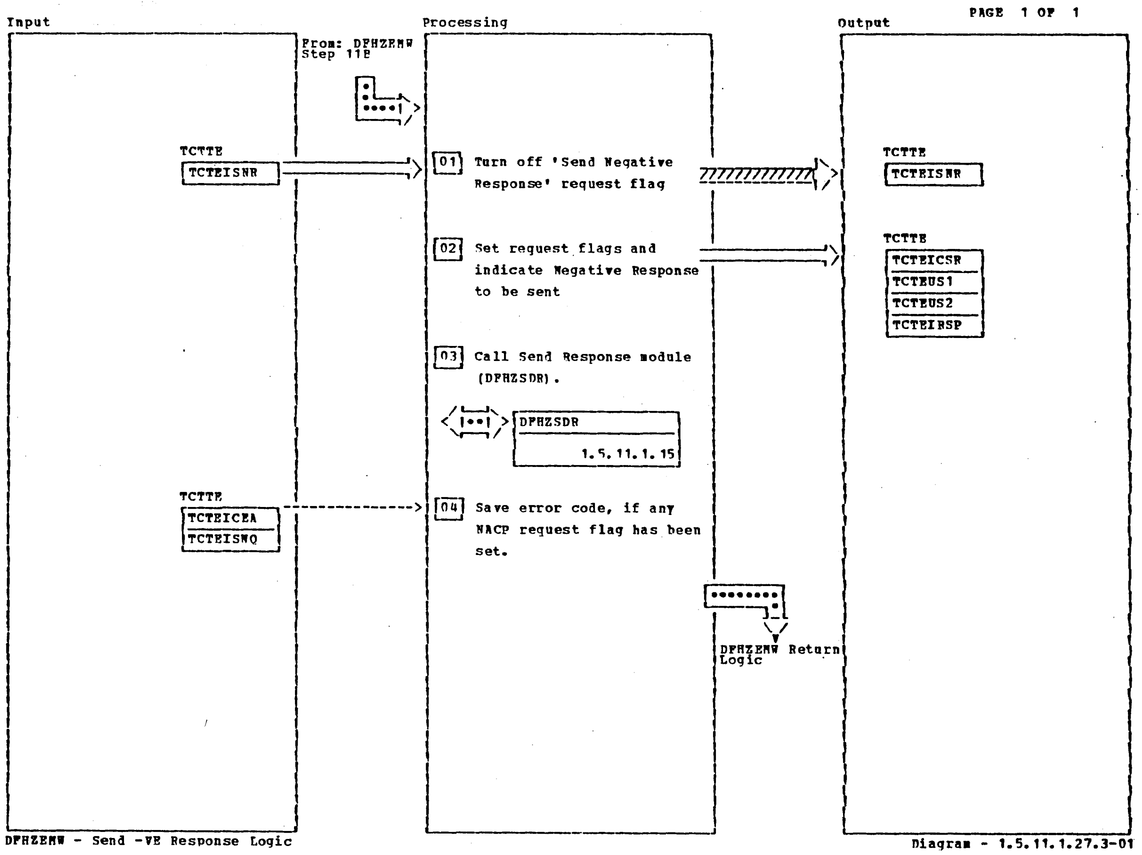
DFHZEMW - BID logic

Diagram - 1.5.11.1.27.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

DFHZEMW - BID logic

Diagram - 1.5.11.1.27.2-01


PAGE 1 OF 1
Input
Processing
Output
From: DFHZENW
Step 11E
TCTTE
TCTEISNR
01 Turn off 'Send Negative Response' request flag
TCTTE
TCTEISNR
02 Set request flags and indicate Negative Response to be sent
TCTTE
TCTEICSR
TCTEUS1
TCTEUS2
TCTEIRSP
03 Call Send Response module (DFHZSDR).
DFHZSDR
1.5.11.1.15
TCTTE
TCTEICEA
TCTEISNQ
04 Save error code, if any NACP request flag has been set.
DFHZENW Return Logic

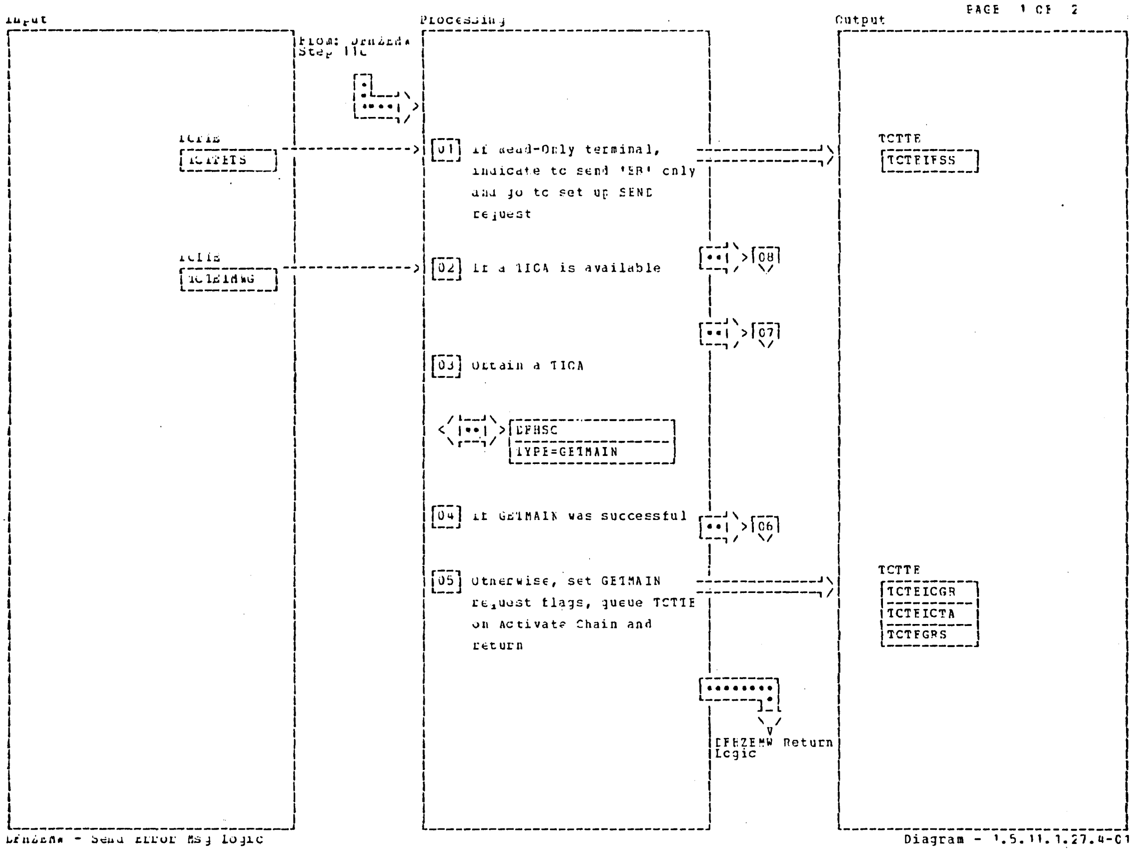
DFHZENW - Send -VE Response Logic

Diagram - 1.5.11.1.27.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |

DFHZENW - Send -VE Response Logic

Diagram - 1.5.11.1.27.3-01



01 If Read-Only terminal, indicate to send 'EB' only and go to set up SEND request

02 If a TIOA is available

03 Obtain a TIOA

04 If GETMAIN was successful

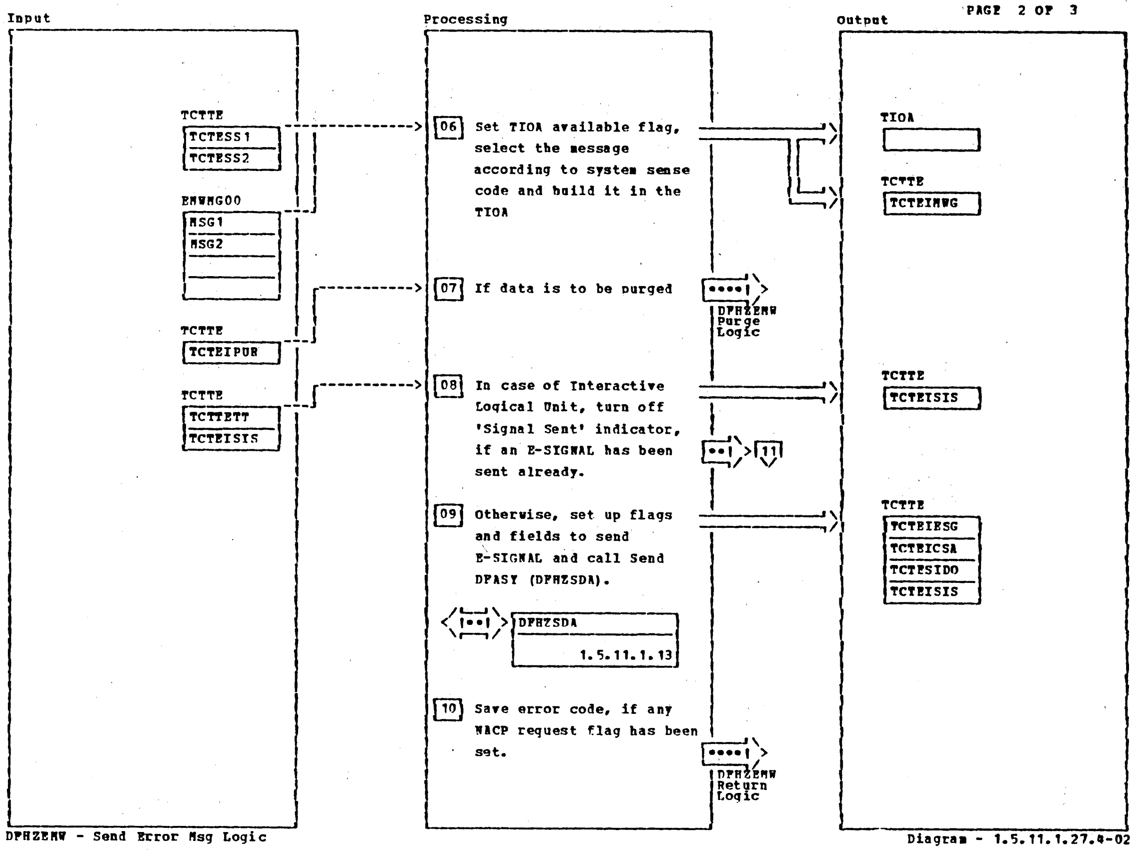
05 Otherwise, set GETMAIN request flags, queue TCTTE on Activate Chain and return

DFHZEMW - Send Error Msg Logic

Diagram - 1.5.11.1.27.4-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

PAGE 2 OF 3



06 Set TIOA available flag, select the message according to system sense code and build it in the TIOA

07 If data is to be purged

08 In case of Interactive Logical Unit, turn off 'Signal Sent' indicator, if an E-SIGNAL has been sent already.

09 Otherwise, set up flags and fields to send E-SIGNAL and call Send DFASY (DFHZSDA).

10 Save error code, if any WACP request flag has been set.

DFHZEMW - Send Error Msg Logic

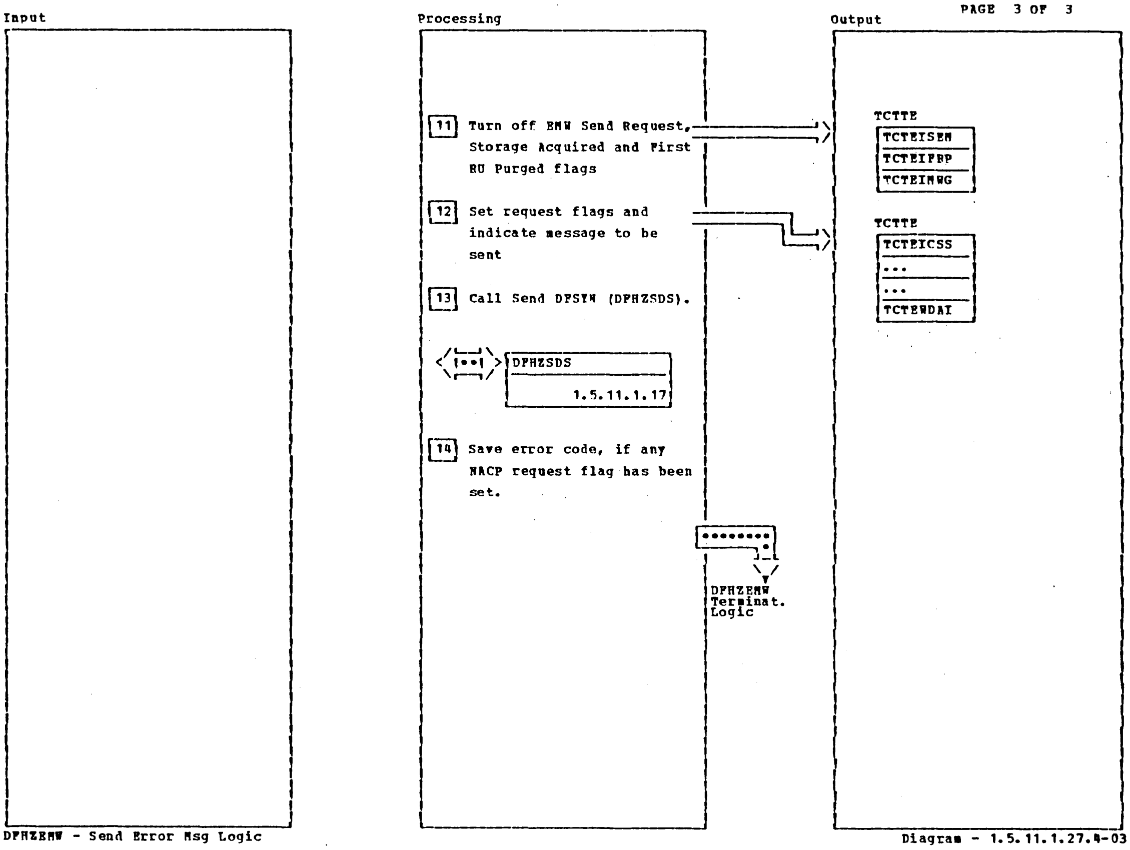
Diagram - 1.5.11.1.27.4-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

DFHZEMW - Send Error Msg Logic

Diagram - 1.5.11.1.27.4-02

PAGE 3 OF 3



11 Turn off BMW Send Request, Storage Acquired and First RU Purged flags

12 Set request flags and indicate message to be sent

13 Call Send DFSYN (DFHZSDS).

14 Save error code, if any NACP request flag has been set.

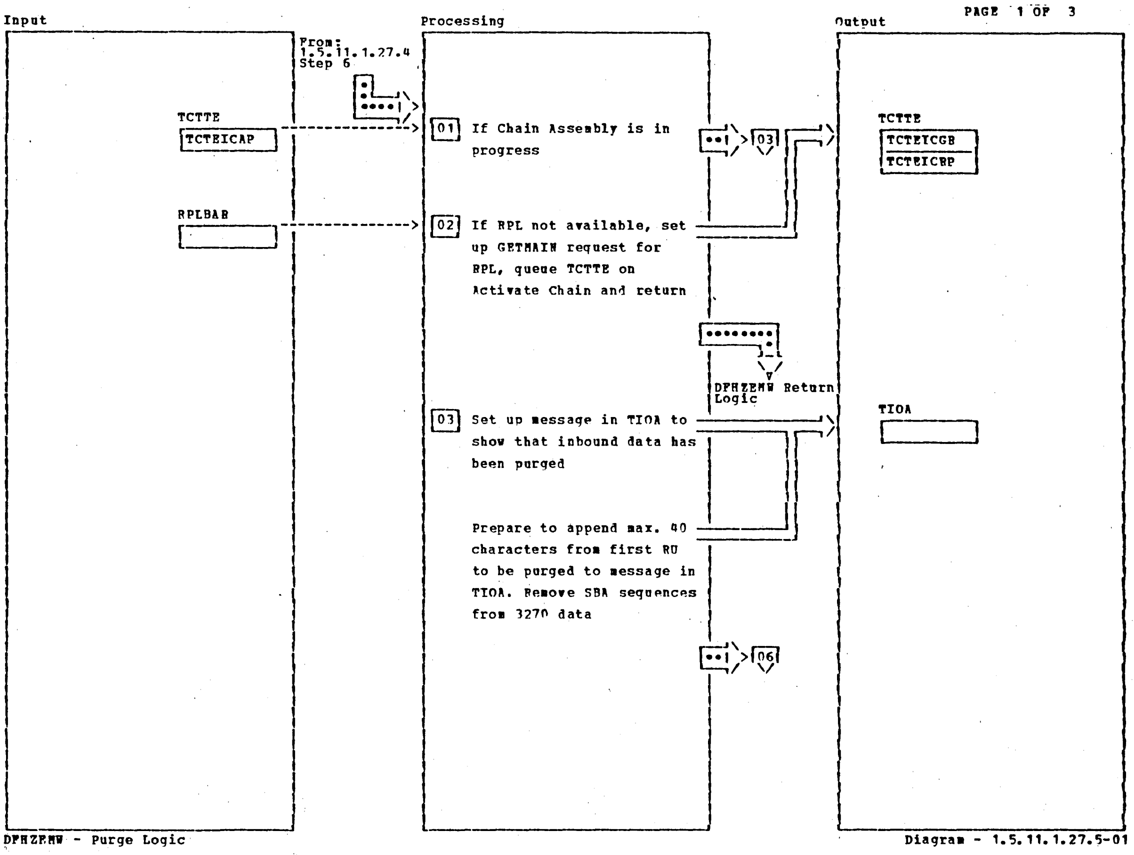
DFHZEMW - Send Error Msg Logic

Diagram - 1.5.11.1.27.4-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 12 Remove any New Line characters from messages to be sent to a 3270 Data Stream terminal. | | | | | | | |

DFHZEMW - Send Error Msg Logic

Diagram - 1.5.11.1.27.4-03



01 If Chain Assembly is in progress

02 If RPL not available, set up GETMAIN request for RPL, queue TCTTE on Activate Chain and return

03 Set up message in TIOA to show that inbound data has been purged

Prepare to append max. 40 characters from first RU to be purged to message in TIOA. Remove SBA sequences from 3270 data

DFHZEMW - Purge Logic

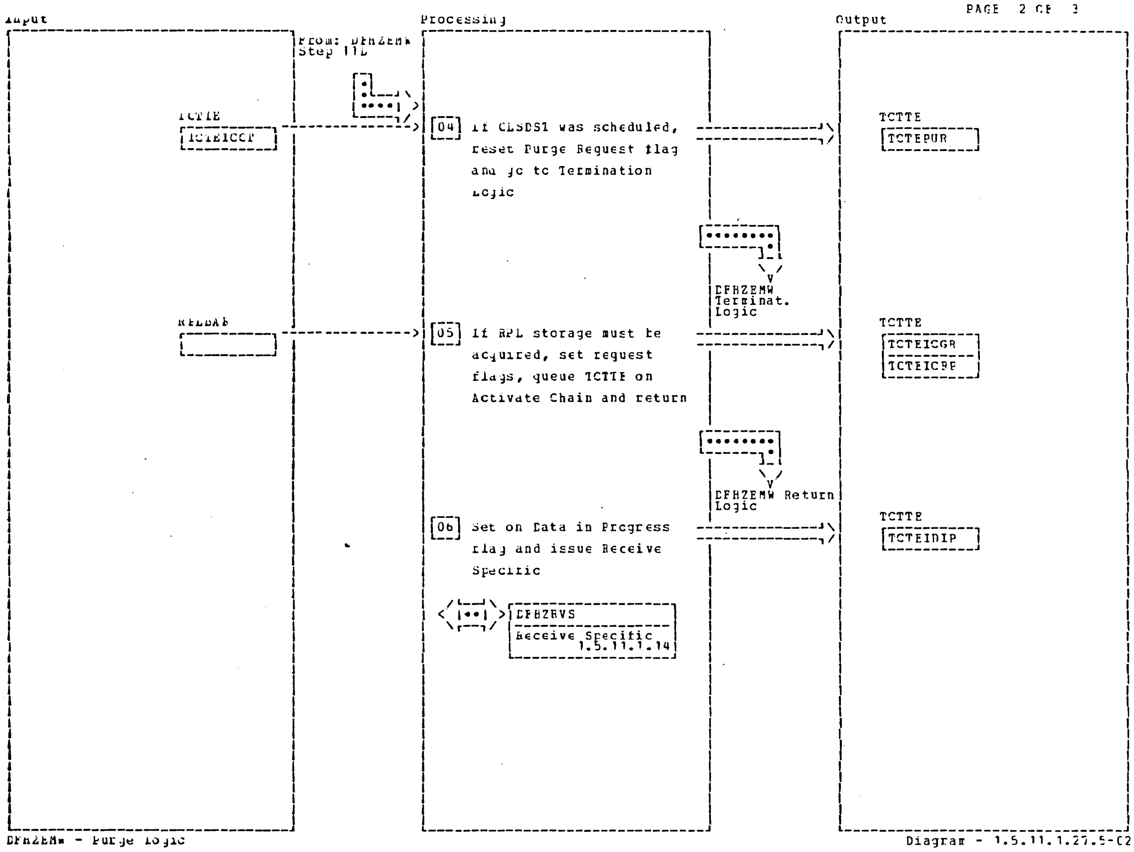
Diagram - 1.5.11.1.27.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

DFHZEMW - Purge Logic

Diagram - 1.5.11.1.27.5-01

PAGE 2 OF 3



04 If CLSDST was scheduled, reset Purge Request flag and go to Termination Logic

05 If RPL storage must be acquired, set request flags, queue TCTTE on Activate Chain and return

06 Set on Data in Progress flag and issue Receive Specific

DFHZEMW - Purge Logic

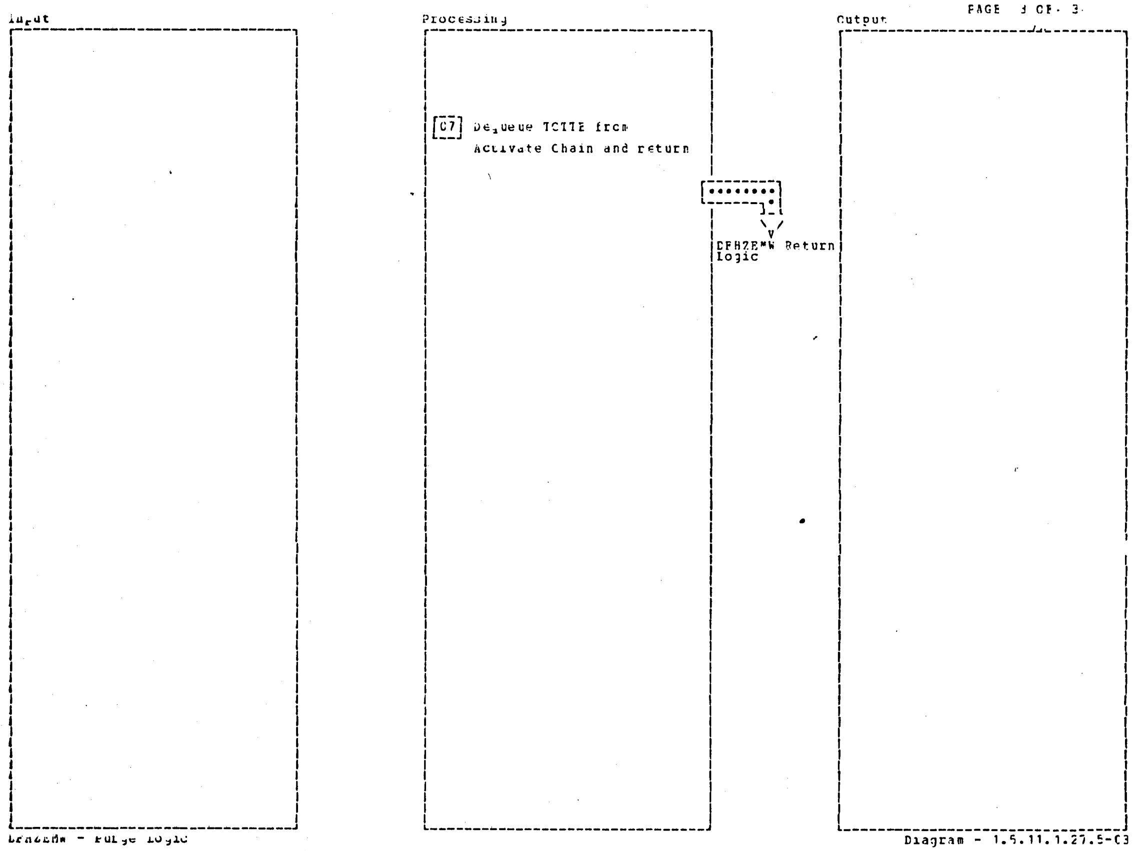
Diagram - 1.5.11.1.27.5-C2

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 06 On completion, DFHZRVS will schedule EMW-Exit TCZEMX01 (see Chart 1.5.11.1.27.6). | TCZEMX01 | | | | | | |

DFHZEMW - Purge Logic

Diagram - 1.5.11.1.27.5-C2

PAGE 3 OF 3



C7 Dequeue TCTTE from Activate Chain and return

DFHZEMW - Purge Logic

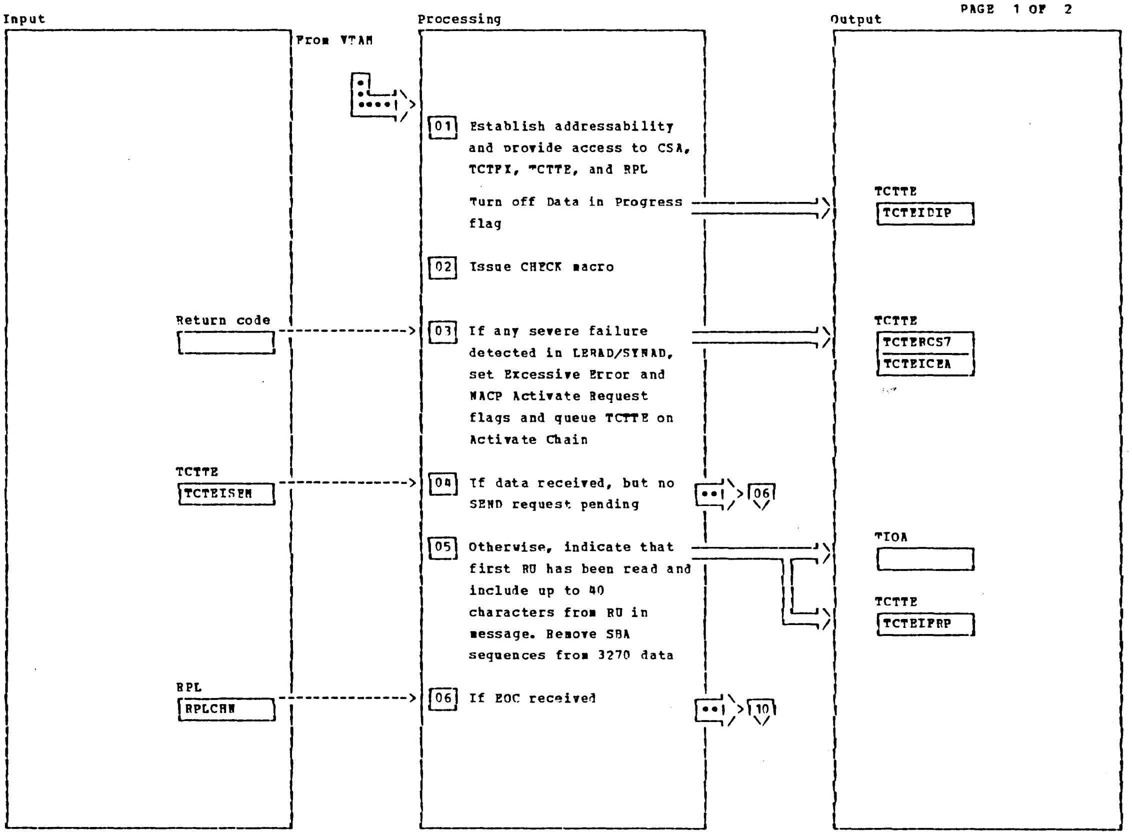
Diagram - 1.5.11.1.27.5-C3

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

DFHZEMW - Purge Logic

Diagram - 1.5.11.1.27.5-C3

PAGE 1 OF 2



01 Establish addressability and provide access to CSA, TCTFX, TCTTE, and RPL

Turn off Data in Progress flag

02 Issue CHECK macro

03 If any severe failure detected in LERAD/SYNAD, set Excessive Error and NACP Activate Request flags and queue TCTTE on Activate Chain

04 If data received, but no SEND request pending

05 Otherwise, indicate that first RU has been read and include up to 40 characters from RU in message. Remove SBA sequences from 3270 data

06 If EOC received

DFHZEMW - Exit TCZEMX01

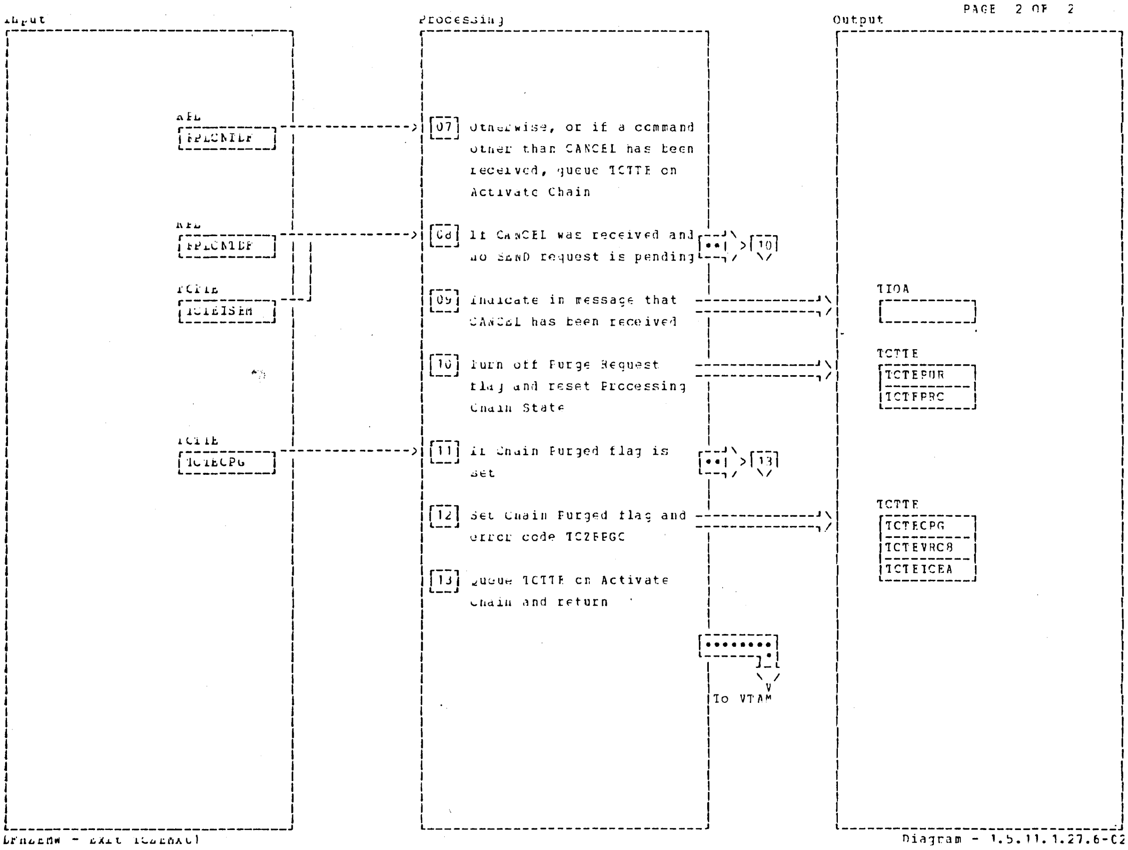
Diagram - 1.5.11.1.27.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 DFHZEMW will perform the complete set-up for Excessive Error. | | | | | | | |

DFHZEMW - Exit TCZEMX01

Diagram - 1.5.11.1.27.6-01

PAGE 2 OF 2



Input

RPL
RPLCNTDF

RPL
RPLCNTDF

TCTTE
TCTEISEM

TCTTE
TCTECPG

Processing

07 Otherwise, or if a command other than CANCEL has been received, queue TCTTE on Activate Chain

08 If CANCEL was received and no SEND request is pending → 10

09 Indicate in message that CANCEL has been received

10 Turn off Purge Request flag and reset Processing Chain State

11 If Chain Purged flag is set → 13

12 Set Chain Purged flag and error code TC2PPGC

13 Queue TCTTE on Activate Chain and return

To VTAM

Output

TIOA

TCTTE
TCTEPUR
TCTEPRC

TCTTE
TCTECPG
TCTEVRC8
TCTEICEA

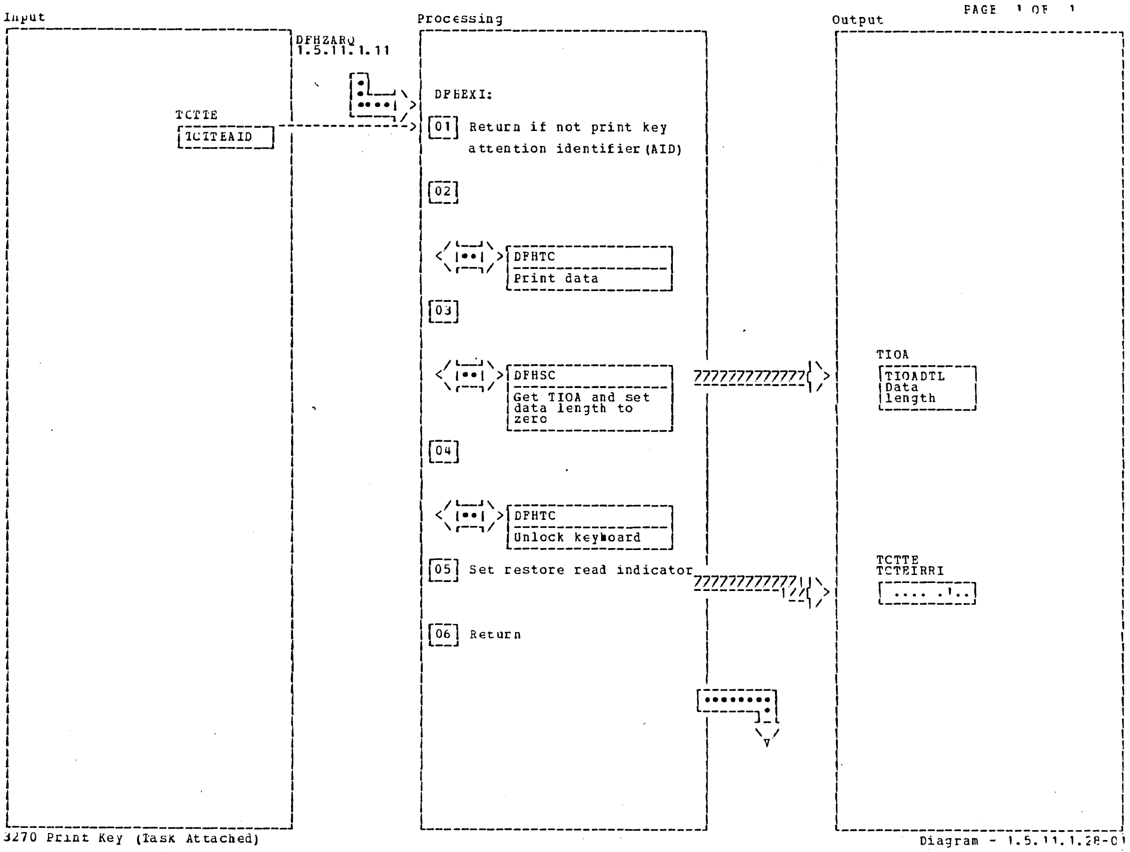
DFHZEMW - Exit ICZEMXC

Diagram - 1.5.11.1.27.6-C2

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 12 This causes NACP to write 'Purged inbound chain' message to CSMT. | | | | | | | |

DFHZEMW - Exit ICZEMXC

Diagram - 1.5.11.1.27.6-C2



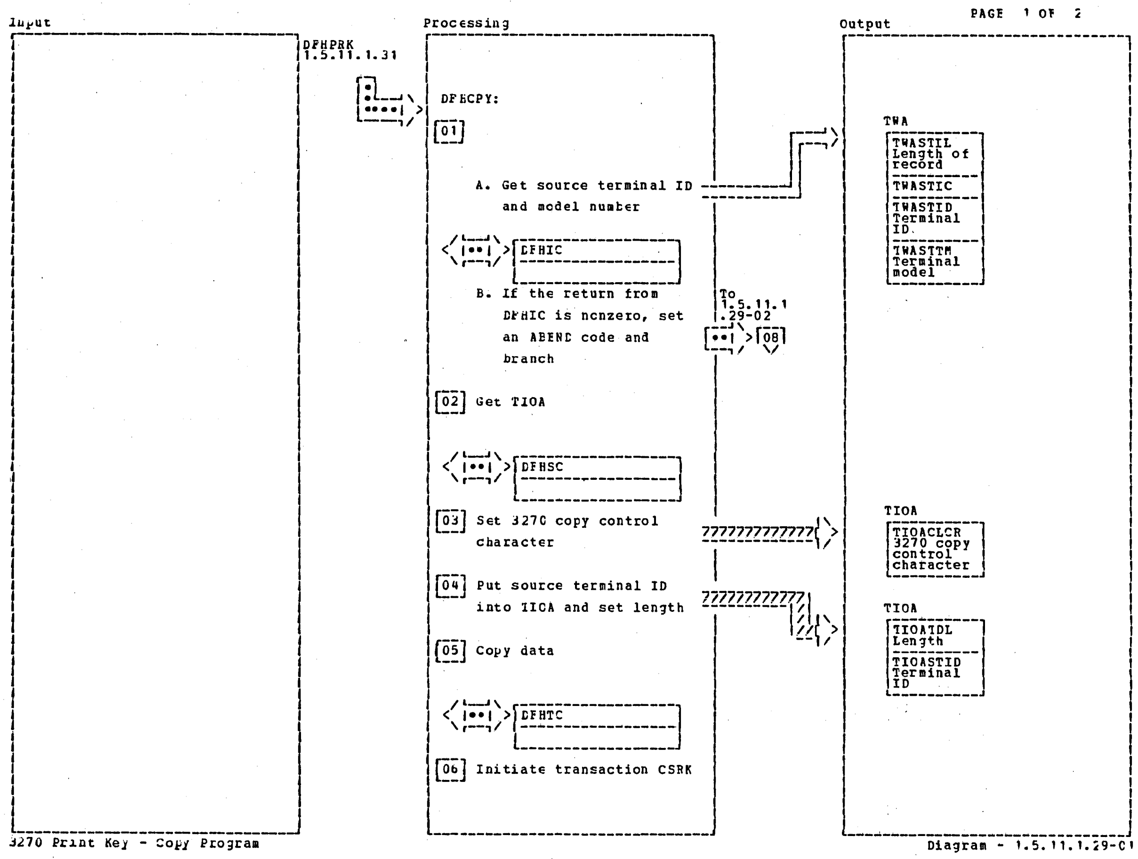
3270 Print Key (Task Attached)

Diagram - 1.5.11.1.28-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The AID defines the 3270 program application key that has been specified as the print key during CICS generation of the TCTTE. | | EXIST | | | | | |
| 02 DFHTC TYPE=PRINT macrc. | | | | | | | |
| 04 C3(hex) specified as the 3270 write control character in the CTLCHAR operand of a DFHTC TYPE=WRITE macro unlocks the keyboard. | | | | | | | |
| 06 | | EXIRET | | | | | |

3270 Print Key (Task Attached)

Diagram - 1.5.11.1.28-01



01

A. Get source terminal ID and model number

B. If the return from DFHIC is nonzero, set an ABEND code and branch

02 Get TIOA

03 Set 3270 copy control character

04 Put source terminal ID into TIOA and set length

05 Copy data

06 Initiate transaction CSRK

3270 Print Key - Copy Program

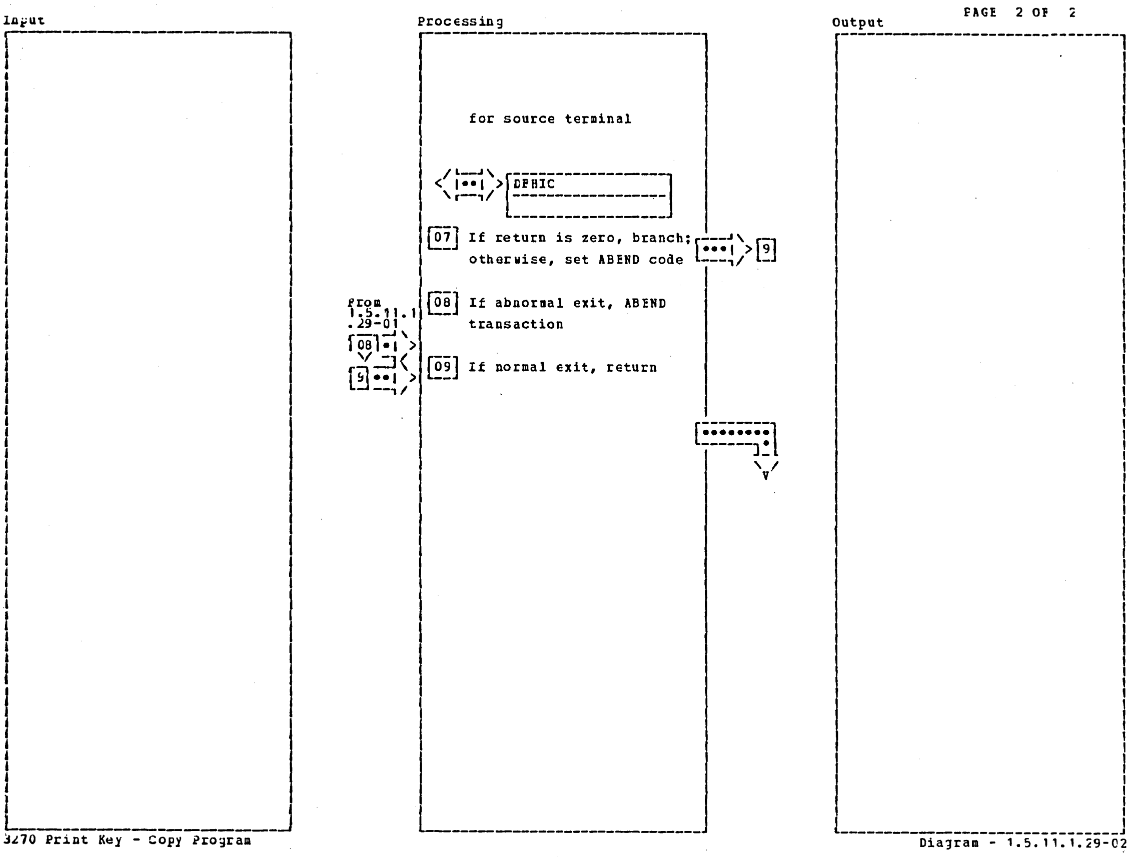
Diagram - 1.5.11.1.29-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 DFHCPY has been invoked by a DFHIC TYPE=PUT macro with data. The data is recovered by way of the DFHIC TYPE=GET macro.<br>ABEND code = ACF1. | | CPYST | | | | | |
| 02 | | CPYGET | | | | | |
| 03 The 3270 copy control character must be provided to start printing from the buffer. The character used depends on the model of the source screen. For example, Model 1 requires a character of 5B (hex), whereas Model 2 requires 7B (hex). The model number of the source screen is available in TWASTTM in character form. | | | | | | | |
| 04 | | CPYSTID | | | | | |
| 05 The DFHTC TYPE=COPY macro will copy from the terminal whose ID is specified in the TIOA. | | | | | | | |

3270 Print Key - Copy Program

Diagram - 1.5.11.1.29-01

PAGE 2 OF 2



for source terminal

07 If return is zero, branch; otherwise, set ABEND code

08 If abnormal exit, ABEND transaction

09 If normal exit, return

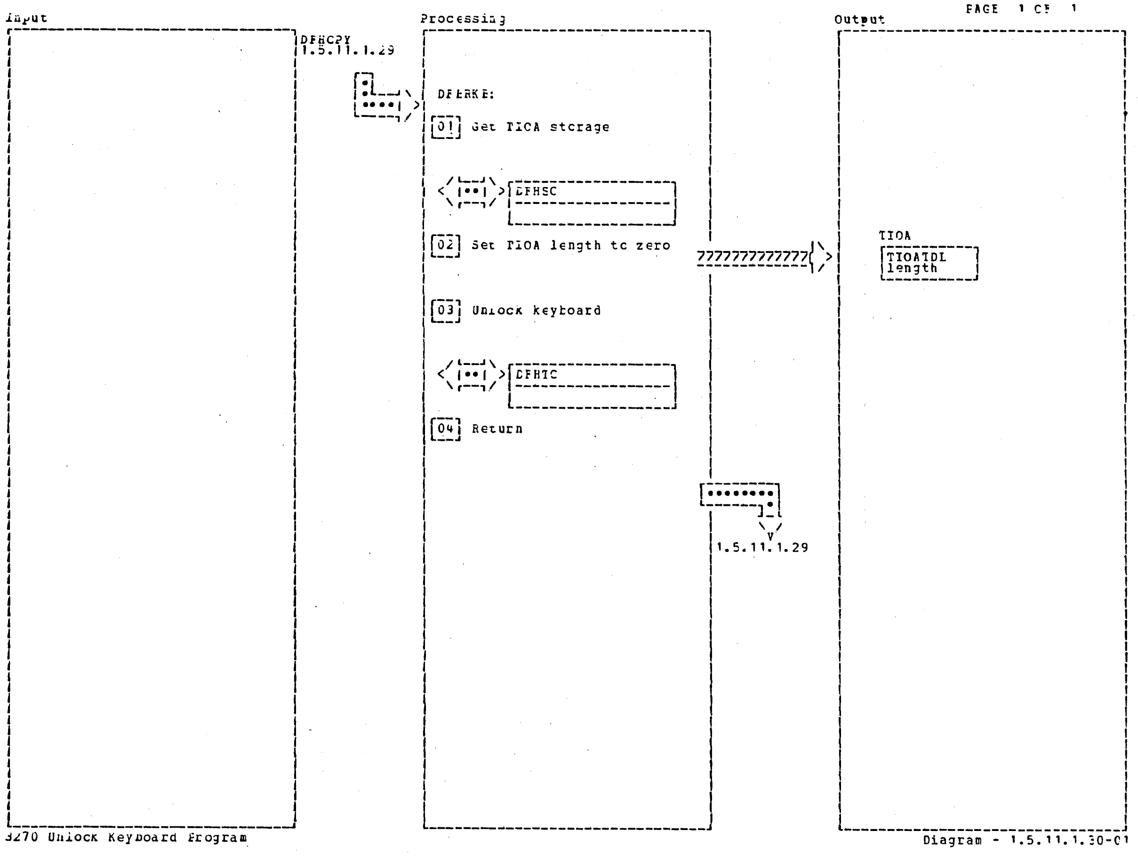
3270 Print Key - Copy Program

Diagram - 1.5.11.1.29-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 06 Transaction CSHK will unlock the source terminal keyboard. | | | | | | | |
| 07 ABEND code = ACE2. | | | | | | | |
| 08 | | CPYABND | | | | | |
| 09 | | CPYRET | | | | | |

3270 Print Key - Copy Program

Diagram - 1.5.11.1.29-02



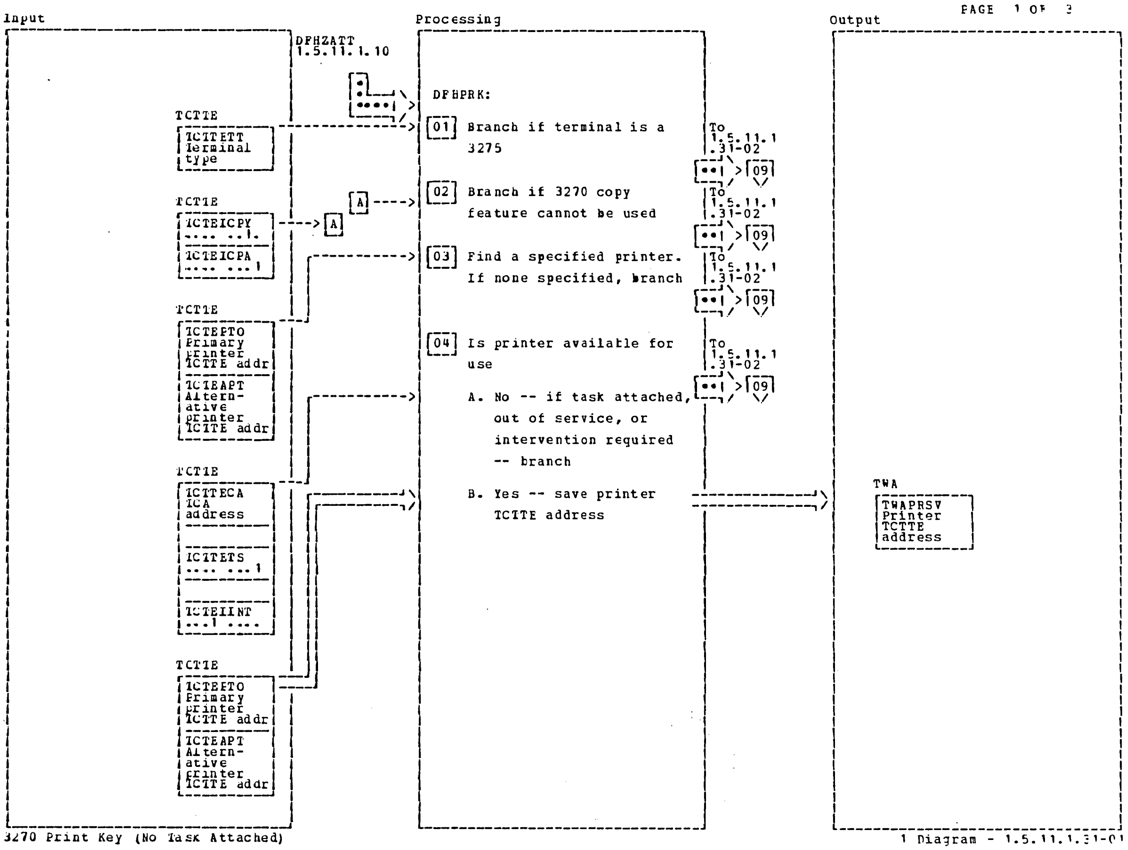
3270 Unlock Keyboard Program

Diagram - 1.5.11.1.30-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 C3(hex) specified as the 3270 write control character in the CTLCHAR operand of a DFHTC TYPE=WRITE macro unlocks the keyboard. | | | | | | | |

3270 Unlock Keyboard Program

Diagram - 1.5.11.1.30-01



3270 Print Key (No Task Attached)

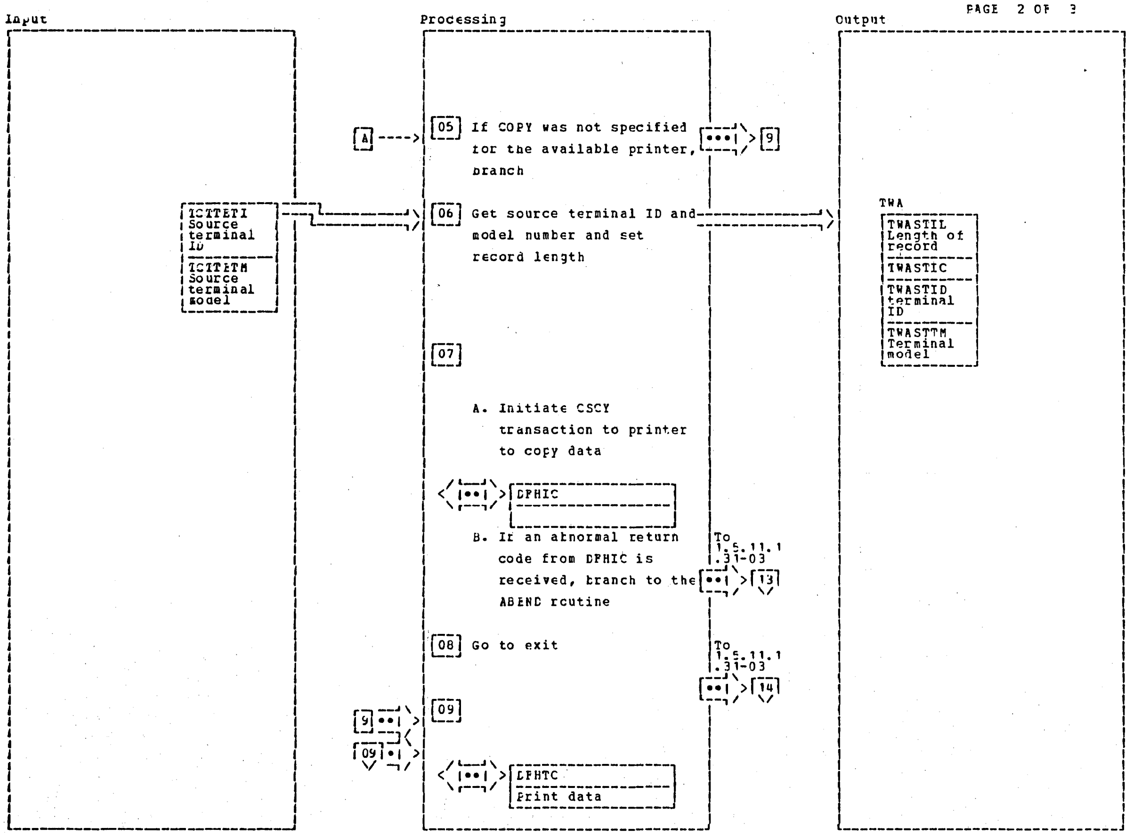
1 Diagram - 1.5.11.1.31-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Hardware copy feature not available for 3275. | | PRKST | | | | | |
| 02 The COPY option must have been specified in the PRINTTO or ALTPRT operands of the DFHTCT macro. | | | | | | | |
| 03 If no printer is available, terminal control will diagnose this condition when the DFHTC TYPE=PRINT macro is invoked. | | PRKFDPT | | | | | |
| 04 See also Note 03. | | PRKAVL | | | | | |

3270 Print Key (No Task Attached)

Diagram - 1.5.11.1.31-01

PAGE 2 OF 3


Input
Processing
Output
A
05 If COPY was not specified for the available printer, branch
9
TCTTETI Source terminal ID
TCTTETM Source terminal model
06 Get source terminal ID and model number and set record length
TWA
TWASTIL Length of record
TWASTIC
TWASTID terminal ID
TWASTTM Terminal model
07
A. Initiate CSCY transaction to printer to copy data
DFHIC
B. If an abnormal return code from DFHIC is received, branch to the ABEND routine
To 1.5.11.1.31-03
13
08 Go to exit
To 1.5.11.1.31-03
14
9
09
09
DFHTC
Print data

3270 Print Key (No Task Attached)

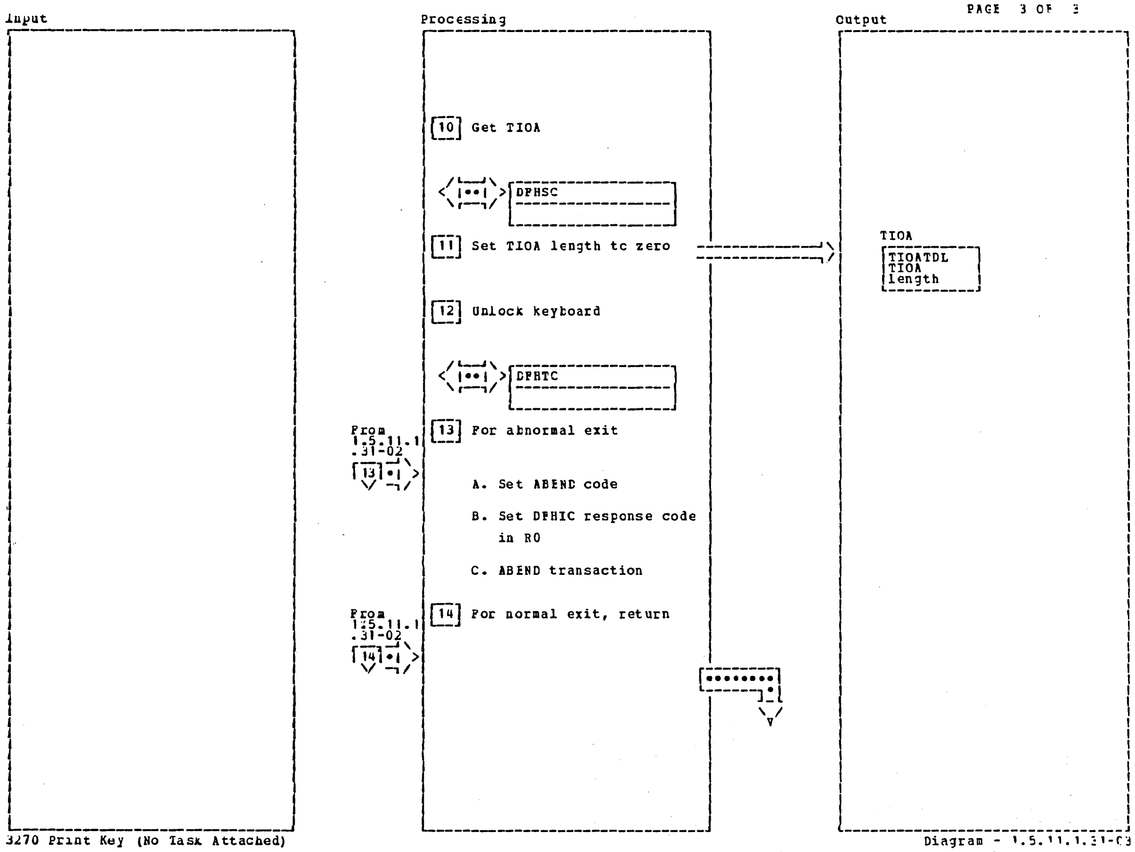
Diagram - 1.5.11.1.31-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

3270 Print Key (No Task Attached)

Diagram - 1.5.11.1.31-02

PAGE 3 OF 3



3270 Print Key (No Task Attached)

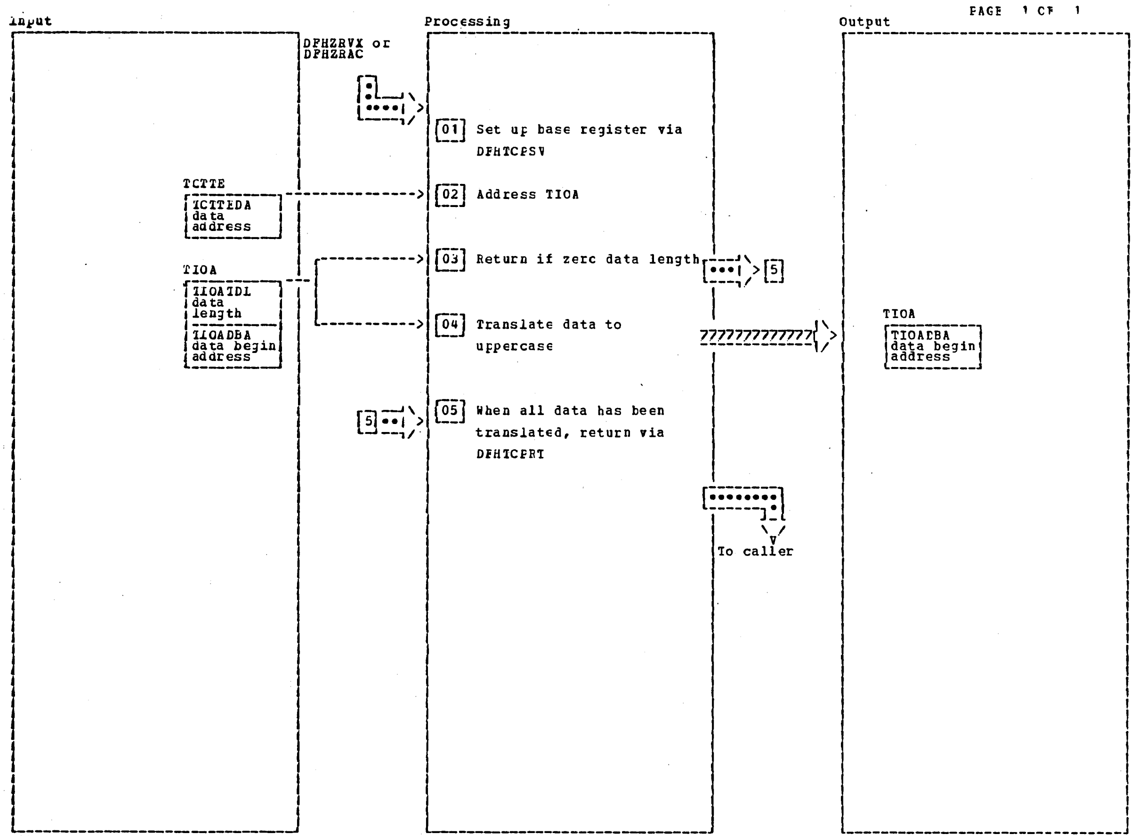
Diagram - 1.5.11.1.31-03

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 05 If required, the COPY option will have been specified in the PRINTTO (primary printer) or ALTPRT (alternative printer) operands of the DFHTCT macro. See also Note 03. | | | |
| 06 | | PRKCPY | |
| 07 The CSCY transaction is initiated by a DFHIC TYPE=PUT macro. Data is provided in V-format, starting at TWASTIL in the TWA, and contains the source terminal ID and terminal model number. | | | |
| 09 If the 3270 copy feature cannot be used, copy is performed by the DFHTC TYPE=PRINT macro. | | PRKPRT | |
| 12 C3(hex) specified as the 3270 write control character in the CTLCHAR operand of a DFHTC TYPE=WRITE macro unlocks the keyboard. | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 13 ABEND code = APR1. | | PRKABND | |
| 14 | | PRKRET | |

3270 Print Key (No Task Attached)

Diagram - 1.5.11.1.31-03


PAGE 1 OF 1
Input
Processing
Output
DFHZRVX or
DFHZRAC
TCTTE
TCTTEDA data address
TIOA
TIOATDL data length
TIOADBA data begin address
01 Set up base register via DFHTCPSV
02 Address TIOA
03 Return if zero data length
5
04 Translate data to uppercase
TIOA
TIOADBA data begin address
5
05 When all data has been translated, return via DFHTCPRT
To caller

3270 Uppercase Translation (DFHZUCT)

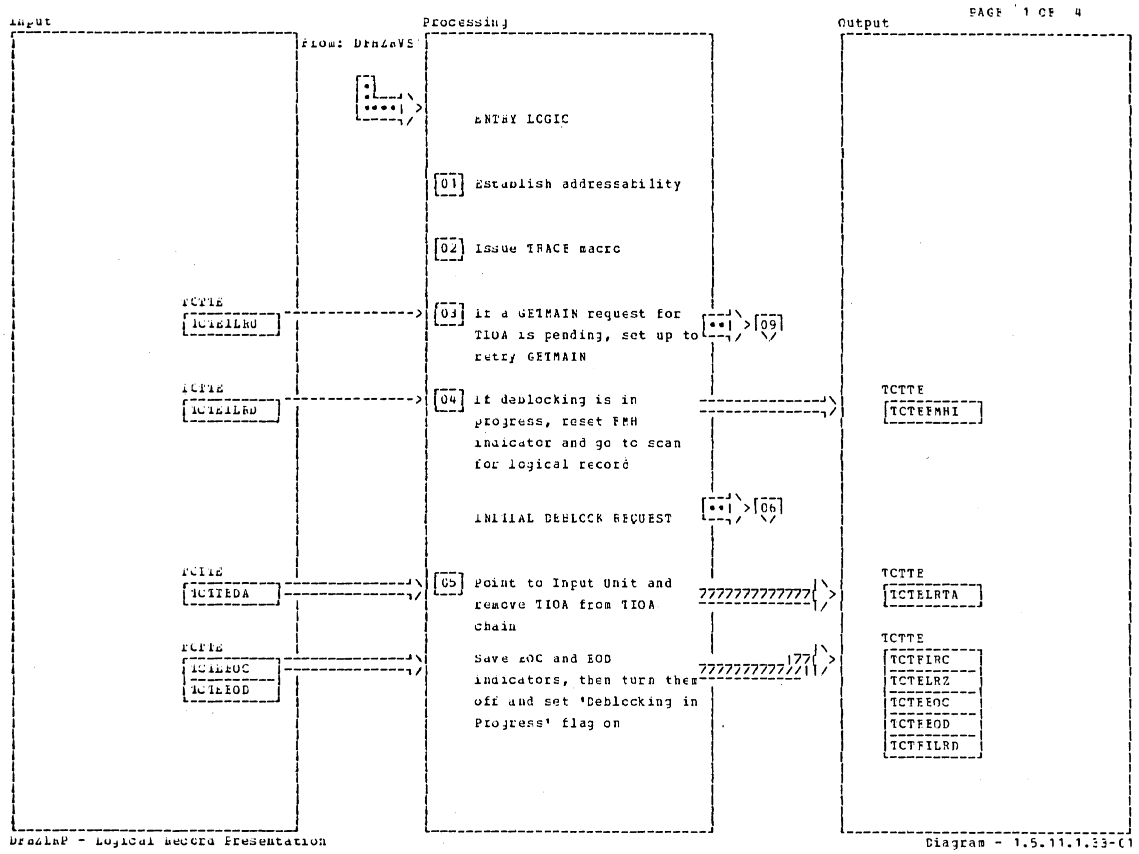
Diagram - 1.5.11.1.32-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 No registers are saved. |  | DFHZUCT1 |  |  |  |  |  |
| 03 TIOATDL is tested for zero. |  | TCZUCT1 |  |  |  |  |  |
| 04 The data is translated 256 bytes at a time. |  | TCZUCT1 |  |  |  |  |  |
| 05 |  | TCZUCT2 |  |  |  |  |  |

3270 Uppercase Translation (DFHZUCT)

Diagram - 1.5.11.1.32-01

PAGE 1 OF 4



ENTRY LOGIC

01 Establish addressability

02 Issue TRACE macro

03 If a GETMAIN request for TIOA is pending, set up to retry GETMAIN

04 If deblocking is in progress, reset FMH indicator and go to scan for logical record

INITIAL DEBLOCK REQUEST

05 Point to Input Unit and remove TIOA from TIOA chain

Save EOC and EOD indicators, then turn them off and set 'Deblocking in Progress' flag on

DFHZLRP - Logical Record Presentation

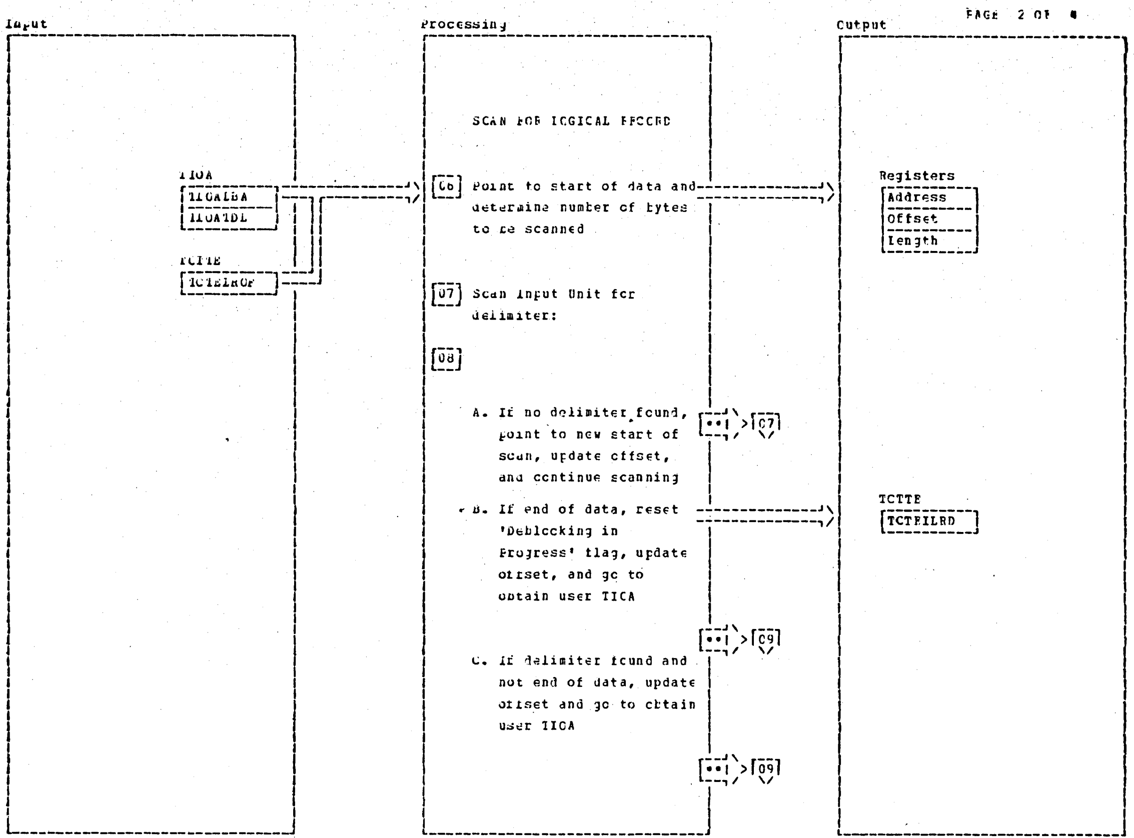
Diagram - 1.5.11.1.33-(1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05<br>DFHSC TYPE=DECHAIN | | | | | | | |

DFHZLRP - Logical Record Presentation

Diagram - 1.5.11.1.33-(1

PAGE 2 OF 4



SCAN FOR LOGICAL RECORD

06 Point to start of data and determine number of bytes to be scanned

07 Scan Input Unit for delimiter:

08

A. If no delimiter found, point to new start of scan, update offset, and continue scanning

B. If end of data, reset 'Deblocking in Progress' flag, update offset, and go to obtain user TIOA

C. If delimiter found and not end of data, update offset and go to obtain user TIOA

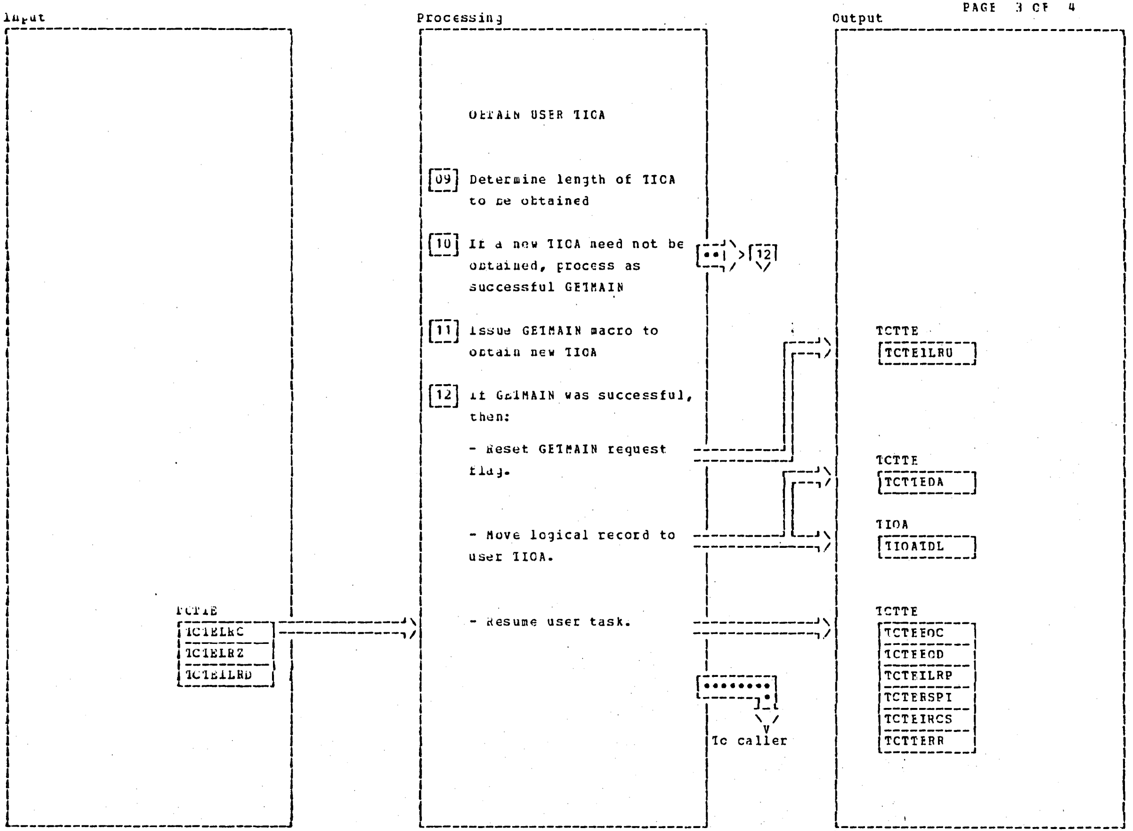
DFHZLRP - Logical Record Presentation

Diagram - 1.5.11.1.33-C2

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 06 If an FMH is present (TCTEFMHI is on), data starts behind FMH. | | | | | | | |
| 08 | | | | | | | |
| A. | | | | | | | |
| B. The end of an Input Unit is considered to be the end of a logical record, regardless whether or not a delimiter was encountered. | | | | | | | |

DFHZLRP - Logical Record Presentation

Diagram - 1.5.11.1.33-C2


PAGE 3 OF 4
Input
Processing
Output
OBTAIN USER TICA
09 Determine length of TICA to be obtained
10 If a new TICA need not be obtained, process as successful GETMAIN
12
11 Issue GETMAIN macro to obtain new TICA
12 If GETMAIN was successful, then:
- Reset GETMAIN request flag.
- Move logical record to user TIOA.
- Resume user task.
To caller
TCTTE
TCTEILRC
TCTEILRZ
TCTEILRU
TCTTE
TCTEILRU
TCTTE
TCTTEDA
TIOA
TIOATDL
TCTTE
TCTEEOC
TCTEEOD
TCTEILRP
TCTERSPI
TCTEIRCS
TCTTERR

DFHZLRP - Logical Record Presentation

Diagram - 1.5.11.1.33-C3

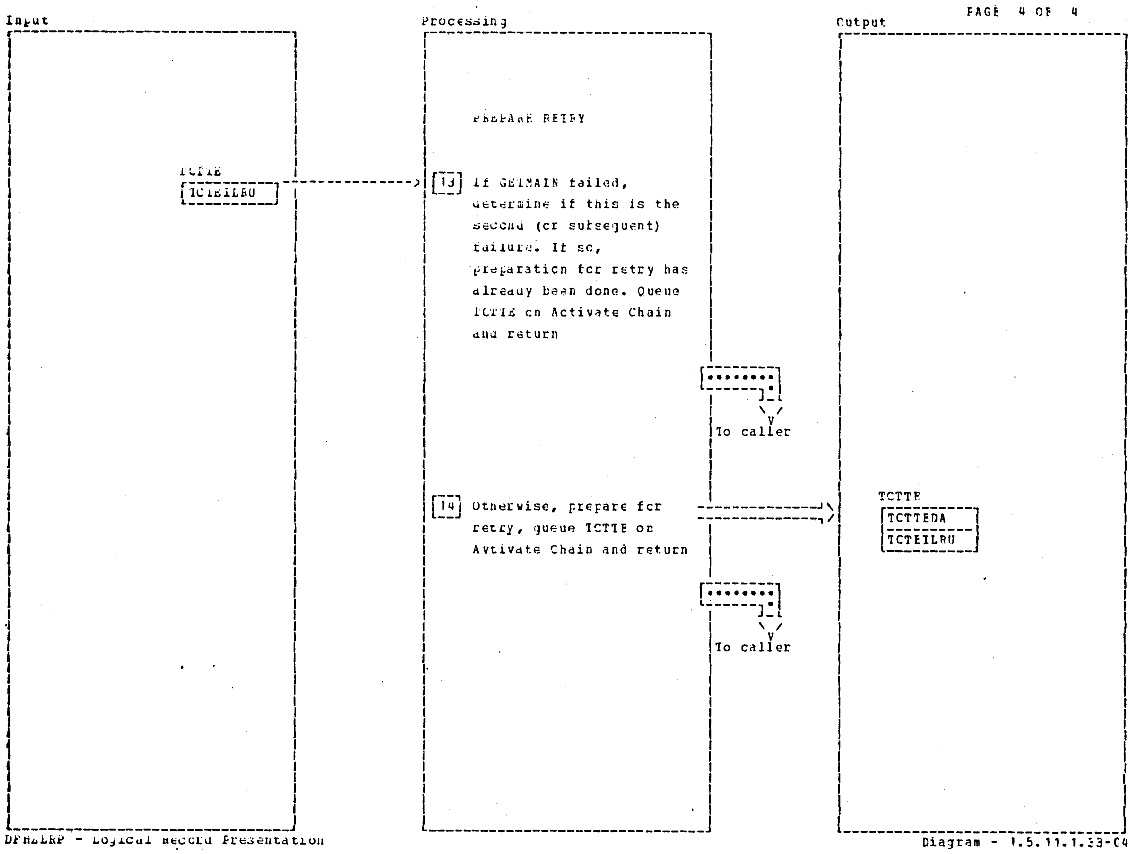
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 10 A new TIOA must be obtained if one of the following conditions is fulfilled:<br>- No user TICA is available.<br>- There is a SAVE request.<br>- The existing TIOA is not large enough. In this case, the existing TIOA is freed. | | | |
| 11 DFHSC TYPE=GETMAIN<br>for terminal storage | | | |
| 12 Moving the logical record includes:<br>- Pointing behind FMH if there exists one.<br>- Establishing pointers to TIOA and next logical record.<br>- Checking for delimiters: NL characters are moved, TRN and IRS characters are not. | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| Resuming the user task includes checking if RU is exhausted. If yes, TCTELFF is reset to cause DFHZRVS to read the next RU. This is done in addition to resetting the flags TCTERSPI, TCTEIRCS, and TCTTERR, which is done regardless whether the RU is exhausted or not. | | | |

DFHZLRP - Logical Record Presentation

Diagram - 1.5.11.1.33-C3

PAGE 4 OF 4



13 If GETMAIN failed, determine if this is the second (or subsequent) failure. If so, preparation for retry has already been done. Queue TCTTE on Activate Chain and return

14 Otherwise, prepare for retry, queue TCTTE on Avtivate Chain and return

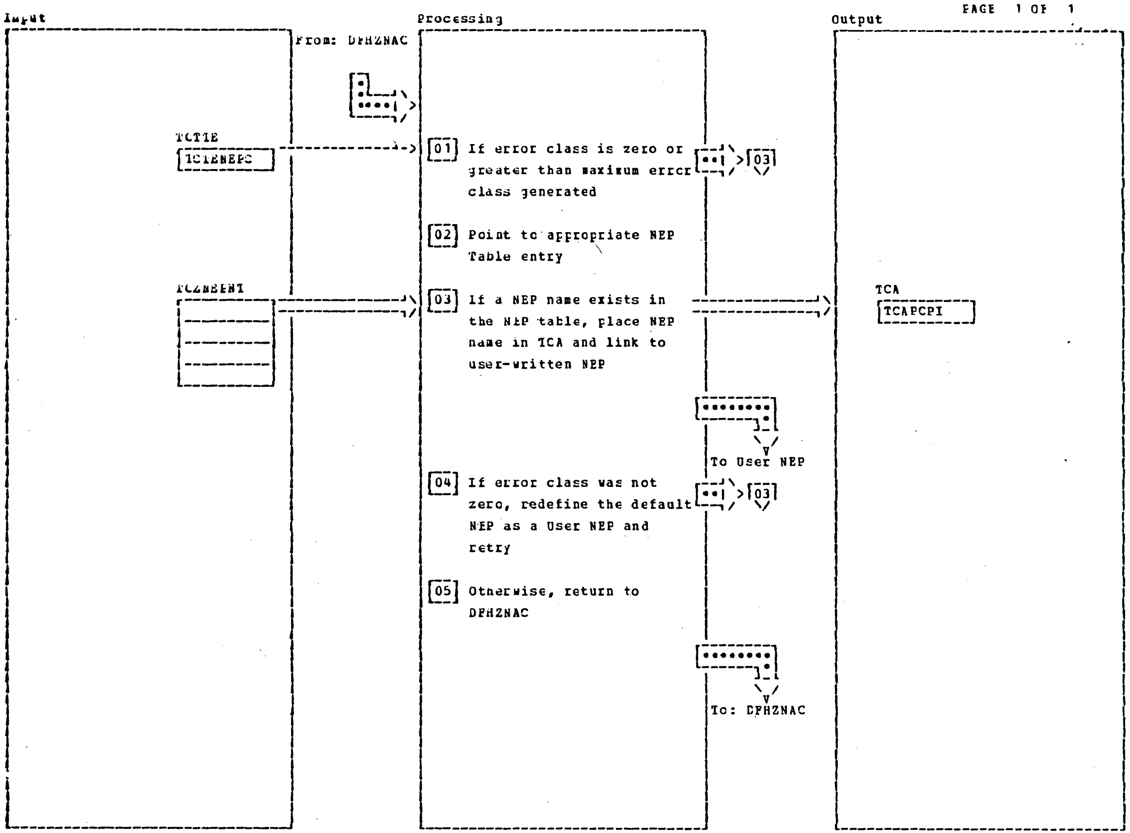
DFHZLRP - Logical Record Presentation

Diagram - 1.5.11.1.33-C4

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 14 The pointer containing the End-of-Data of delimiter address is saved in TCTTEDA in order to maintain the old offset value. | | | | | | | |

DFHZLRP - Logical Record Presentation

Diagram - 1.5.11.1.33-C4



01 If error class is zero or greater than maximum error class generated

02 Point to appropriate NEP Table entry

03 If a NEP name exists in the NEP table, place NEP name in TCA and link to user-written NEP

04 If error class was not zero, redefine the default NEP as a User NEP and retry

05 Otherwise, return to DFHZNAC

DFHZNEP - Node Error Program Interface

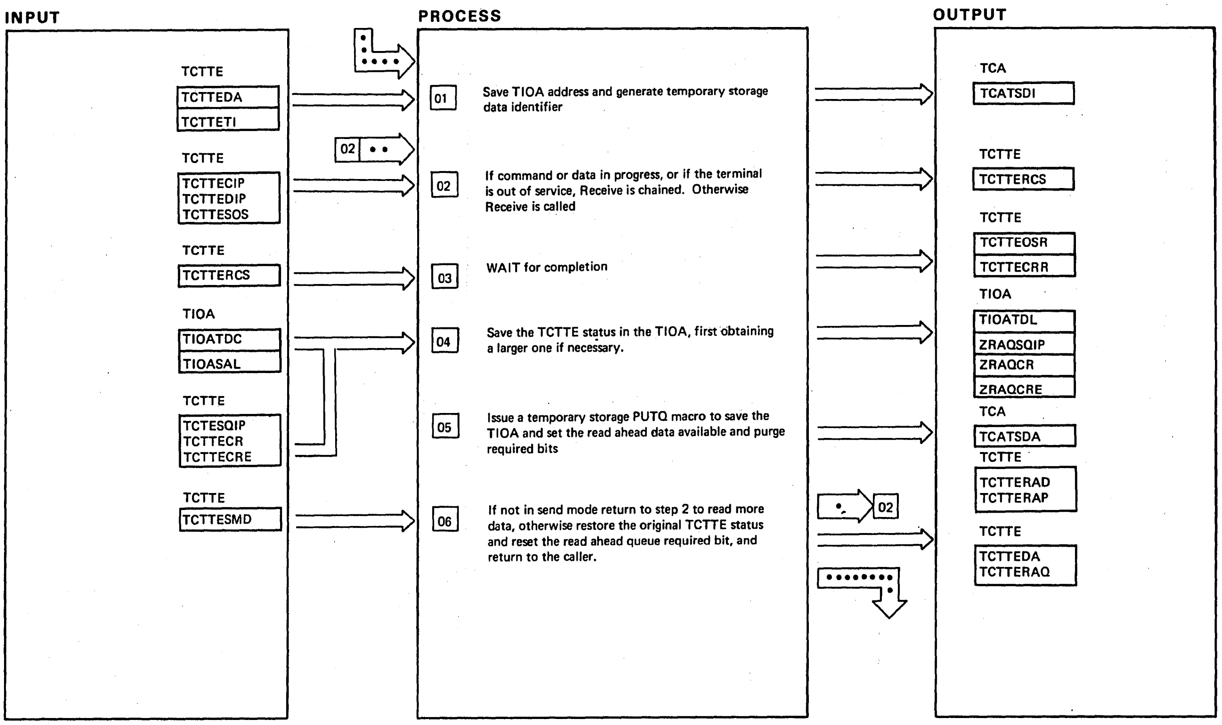
Diagram - 1.5.11.1.34-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 DFHPC TYPE=LINK,COND=YES |  |  |  |  |  |  |  |
| 04 The pointer to the NEP table entry is reset to point to the first table entry. |  |  |  |  |  |  |  |
| 05 DFHPC TYPE=RETURN |  |  |  |  |  |  |  |

DFHZNEP - Node Error Program Interface

Diagram - 1.5.11.1.34-C1

Page 1 of 1



01 Save TIOA address and generate temporary storage data identifier

02 If command or data in progress, or if the terminal is out of service, Receive is chained. Otherwise Receive is called

03 WAIT for completion

04 Save the TCTTE status in the TIOA, first obtaining a larger one if necessary.

05 Issue a temporary storage PUTQ macro to save the TIOA and set the read ahead data available and purge required bits

06 If not in send mode return to step 2 to read more data, otherwise restore the original TCTTE status and reset the read ahead queue required bit, and return to the caller.

DFHZRAQ — Read ahead Queueing

Diagram 1.5.11.1.35–01

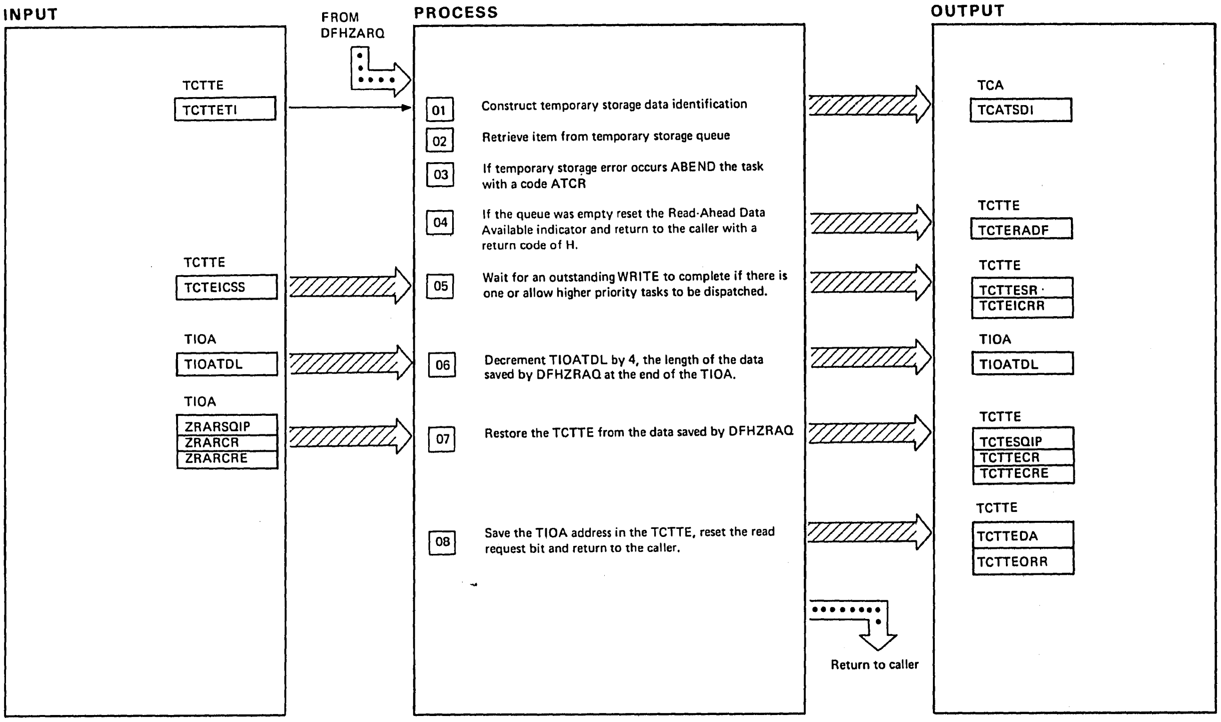
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 Receive module | DFHZRVS | ZRAQ10 | 1.5.11.1.14 |
| 03 DFHKC TYPE=WAIT<br>DCI=TERMINAL<br>or DFHKC TYPE=WAIT, DCI=DISP | | ZRAQ30 | |
| 05 DFHTS TYPE = PUTQ | | ZRAQ60 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

DFHZRAQ — Read ahead Queueing

Diagram 1.5.11.1.35–01

Page 1 of 1



01 Construct temporary storage data identification

02 Retrieve item from temporary storage queue

03 If temporary storage error occurs ABEND the task with a code ATCR

04 If the queue was empty reset the Read-Ahead Data Available indicator and return to the caller with a return code of H.

05 Wait for an outstanding WRITE to complete if there is one or allow higher priority tasks to be dispatched.

06 Decrement TIOATDL by 4, the length of the data saved by DFHZRAQ at the end of the TIOA.

07 Restore the TCTTE from the data saved by DFHZRAQ

08 Save the TIOA address in the TCTTE, reset the read request bit and return to the caller.

DFHZRAR – Read Ahead Retrieval

Diagram 1.5.11.1.36–01

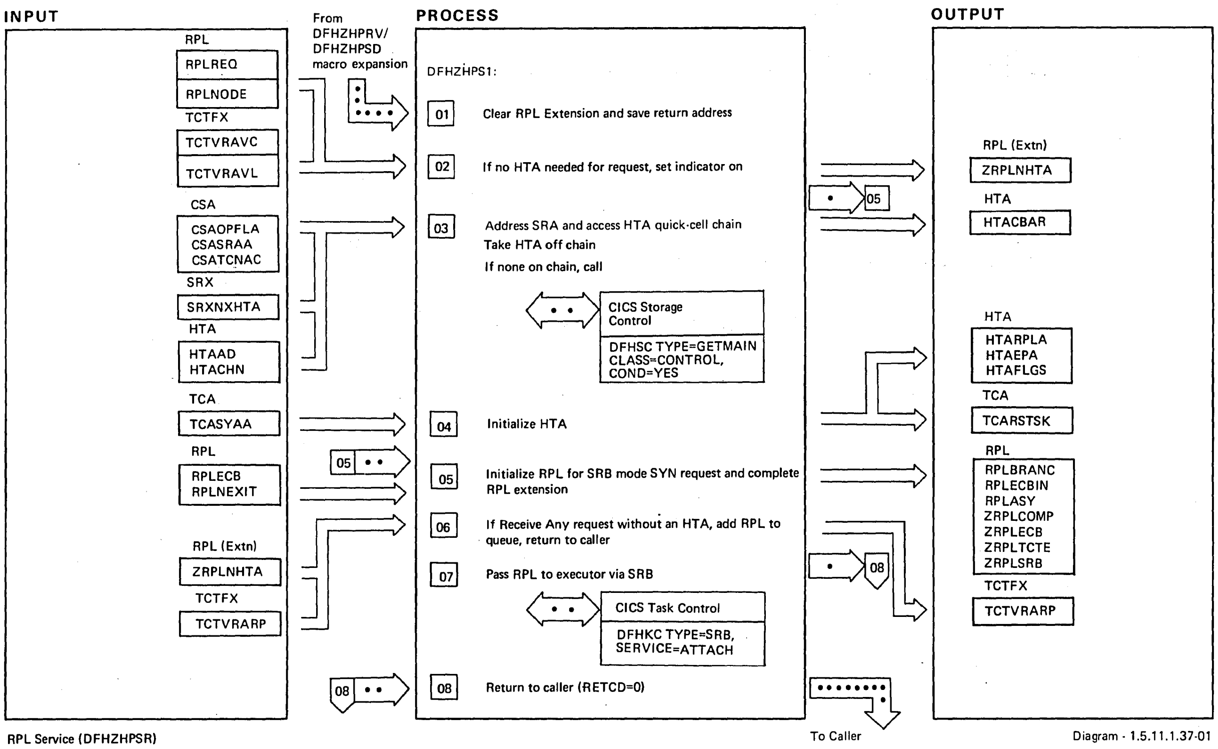
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 03 | | ZRAR90 | |
| 04 | | ZRAR80 | |
| 06 | | ZRAR20 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

DFHZRAR – Read Ahead Retrieval

Diagram 1.5.11.1.36–01

Page 1 of 1



RPL Service (DFHZHPSR)

Diagram - 1.5.11.1.37-01

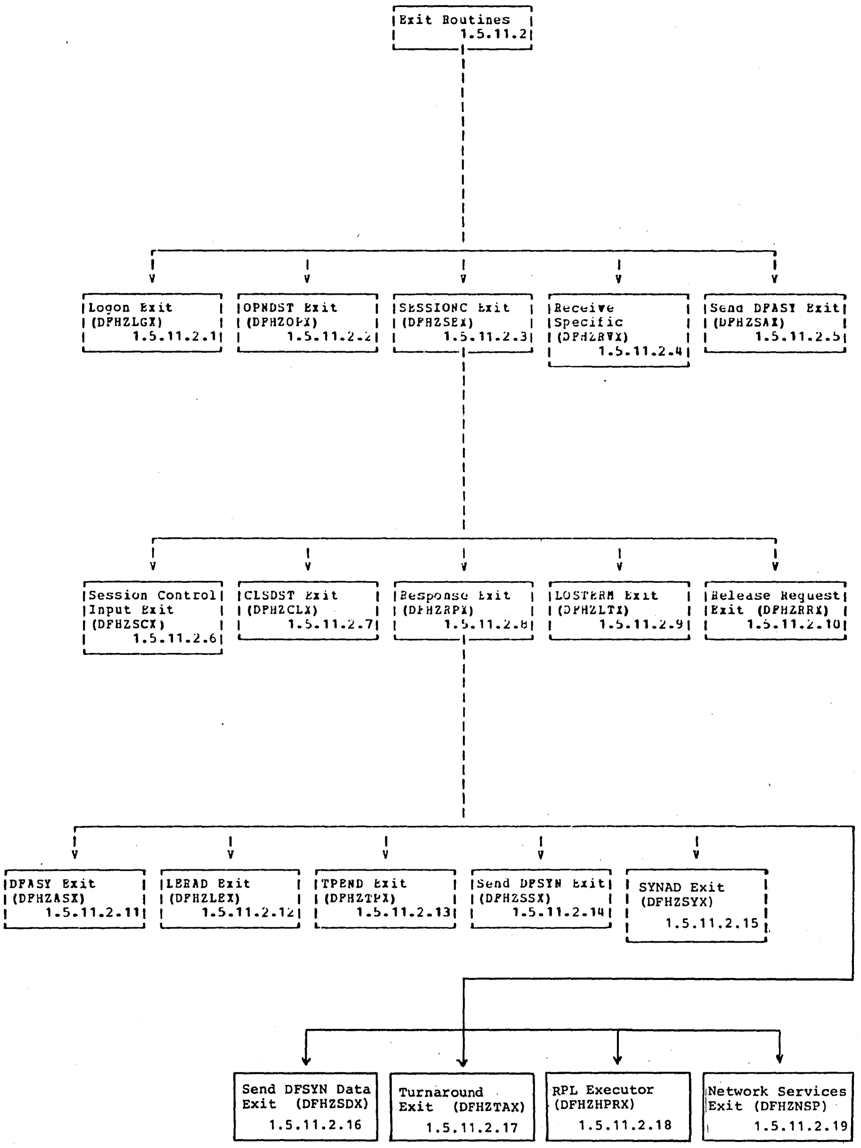
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Return address saved in RPL extension while HTA obtained, then in Register 0. | | | |
| 02 HTA not needed for Receive Any requests unless there are too few active SRB made Receive Anys, always obtain HTA for SEND or RECEIVE SPECIFIC for SRB attach. | | | |
| 07 Control is given to the RPL executor in SRB mode, via Type 6 SVC, issued from SRB management services in task control. | DFHZHPRX | 1.5.11.2.18 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

RPL Service (DFHZHPSR)

Diagram - 1.5.11.1.37-01

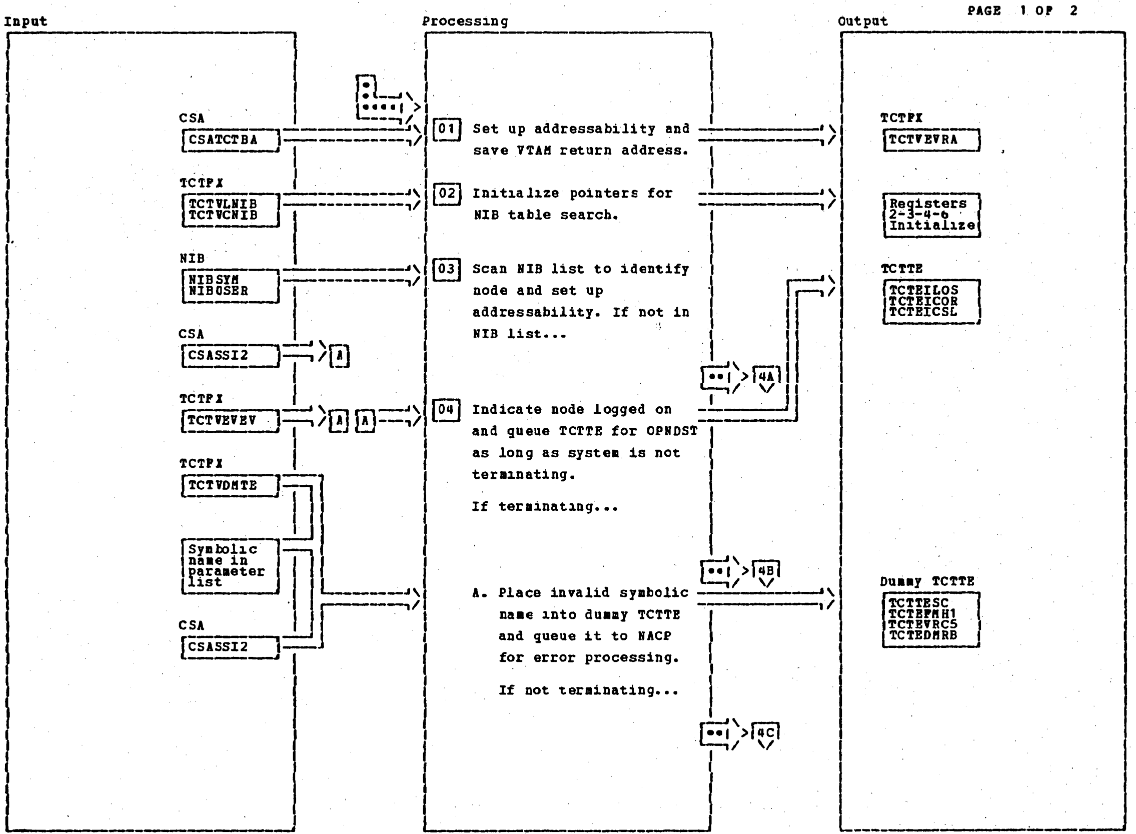
PAGE 1 OF 1


Exit Routines 1.5.11.2
Logon Exit (DFHZLGX) 1.5.11.2.1
OPNDST Exit (DFHZOPX) 1.5.11.2.2
SESSIONC Exit (DFHZSEX) 1.5.11.2.3
Receive Specific (DFHZRVX) 1.5.11.2.4
Send DFASY Exit (DFHZSAX) 1.5.11.2.5
Session Control Input Exit (DFHZSCX) 1.5.11.2.6
CLSDST Exit (DFHZCLX) 1.5.11.2.7
Response Exit (DFHZRPX) 1.5.11.2.8
LOSTERM Exit (DFHZLTX) 1.5.11.2.9
Release Request Exit (DFHZRRX) 1.5.11.2.10
DFASY Exit (DFHZASX) 1.5.11.2.11
LERAD Exit (DFHZLEX) 1.5.11.2.12
TPEND Exit (DFHZTPX) 1.5.11.2.13
Send DFSYN Exit (DFHZSSX) 1.5.11.2.14
SYNAD Exit (DFHZSYX) 1.5.11.2.15
Send DFSYN Data Exit (DFHZSDX) 1.5.11.2.16
Turnaround Exit (DFHZTAX) 1.5.11.2.17
RPL Executor (DFHZHPRX) 1.5.11.2.18
Network Services Exit (DFHZNSP) 1.5.11.2.19

Visual Table of Contents

Diagram — 1.5.11.2-01

PAGE 1 OF 2



01 Set up addressability and save VTAM return address.

02 Initialize pointers for NIB table search.

03 Scan NIB list to identify node and set up addressability. If not in NIB list...

04 Indicate node logged on and queue TCTTE for OPNDST as long as system is not terminating.

If terminating...

A. Place invalid symbolic name into dummy TCTTE and queue it to NACP for error processing.

If not terminating...

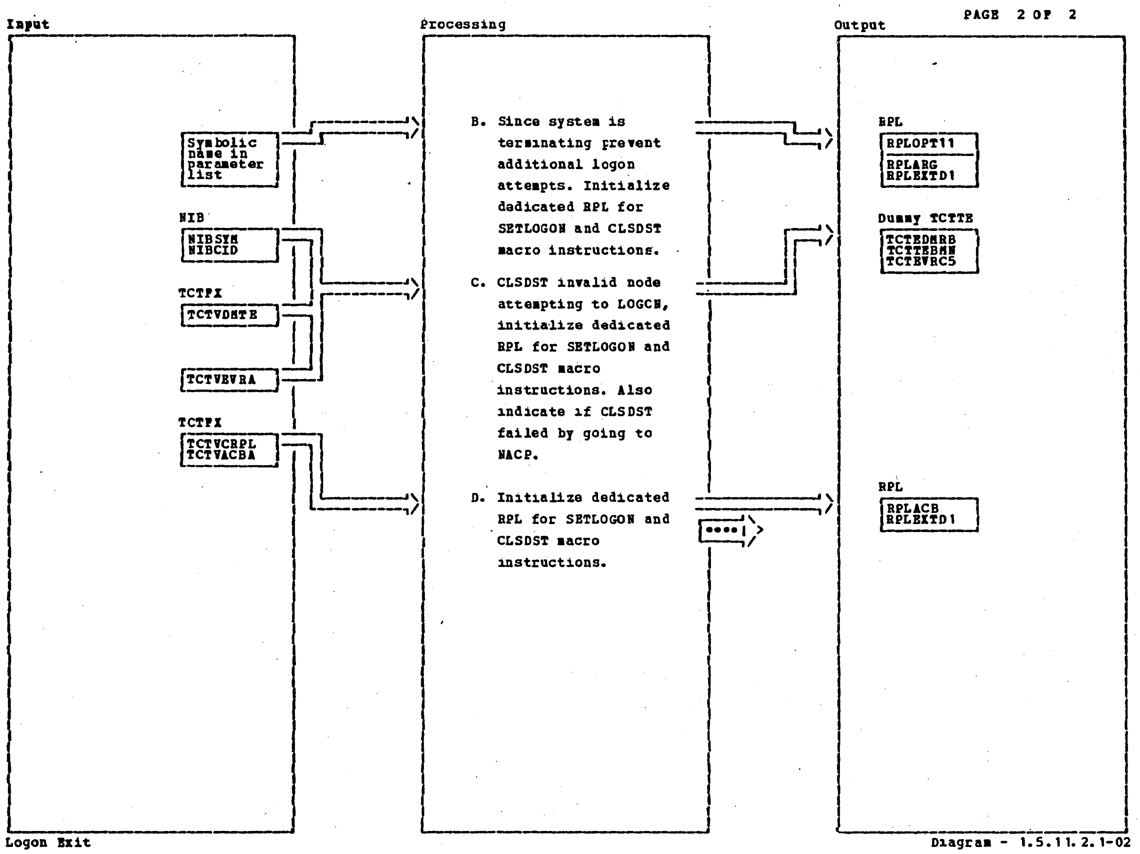
Logon Exit

Diagram - 1.5.11.2.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Register 14 must be saved for it is the return address to VTAM. | | | | | | | |
| 04 Due to the speed at which the logon exit is driven by VTAM, it is very possible that the TCTTE could wind up in SIMLOGON again before the SIMLOGON request bit is turned off. Thus, Logon Exit turns it off also.<br>Note: This hipo indicates the normal flow.<br>A. In order to log the symbolic name of the invalid node, a dummy TCTTE is needed. | | | | | | | |

Logon Exit

Diagram - 1.5.11.2.1-01



B. Since system is terminating prevent additional logon attempts. Initialize dedicated RPL for SETLOGON and CLSDST macro instructions.

C. CLSDST invalid node attempting to LOGON, initialize dedicated RPL for SETLOGON and CLSDST macro instructions. Also indicate if CLSDST failed by going to NACP.

D. Initialize dedicated RPL for SETLOGON and CLSDST macro instructions.

Logon Exit

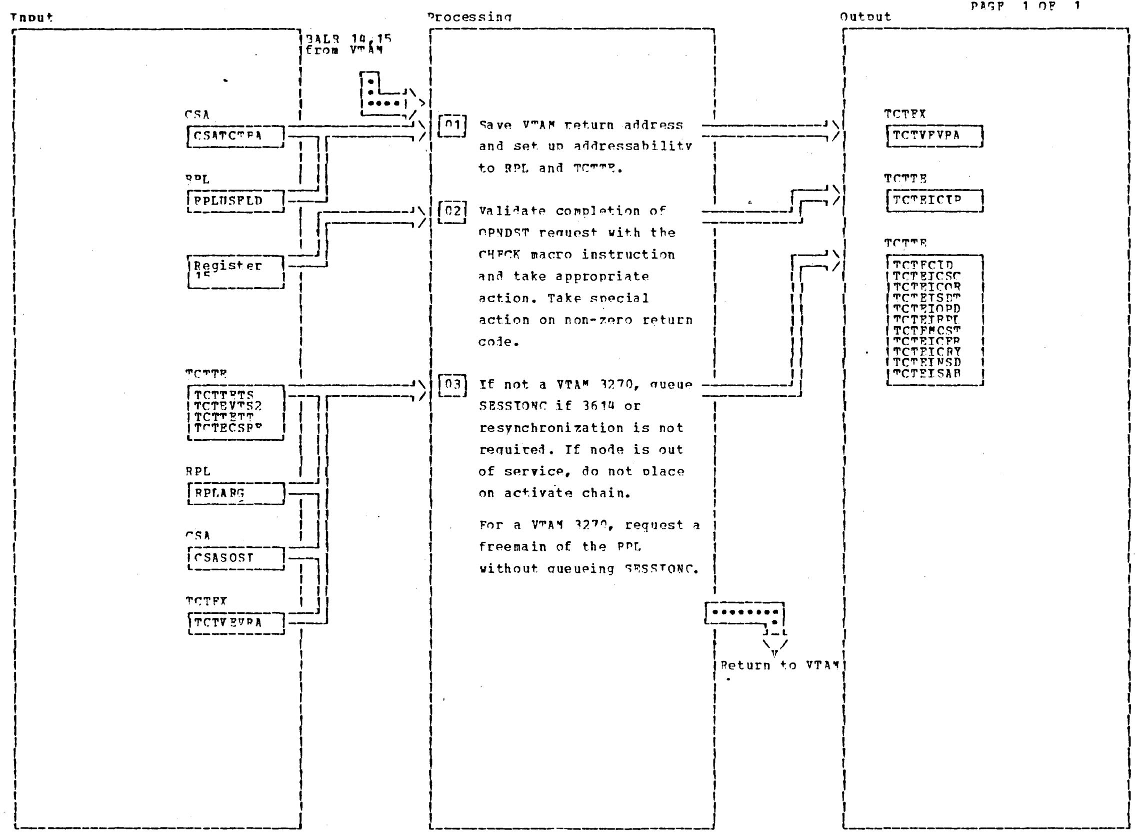
Diagram - 1.5.11.2.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| B. By issuing SETLOGON Quiesce, all subsequent LOGONs will be rejected by VTAM. A common RPL set up routine is linked to with a BALR. | | | | | | | |
| C. Whether the system is terminating or not, a CLSDST of the invalid node is done. A BAL is made to the common RPL setup routine. | | | | | | | |
| D. Common RPL set-up routine is linked (BALR) to by the SETLOGON and CLSDST routine. | | | | | | | |

Logon Exit

Diagram - 1.5.11.2.1-02

CHART: HIP00001
PAGE 1 OF 1



01 Save VTAM return address and set up addressability to RPL and TCTTE.

02 Validate completion of OPNDST request with the CHECK macro instruction and take appropriate action. Take special action on non-zero return code.

03 If not a VTAM 3270, queue SESSIONC if 3614 or resynchronization is not required. If node is out of service, do not place on activate chain.

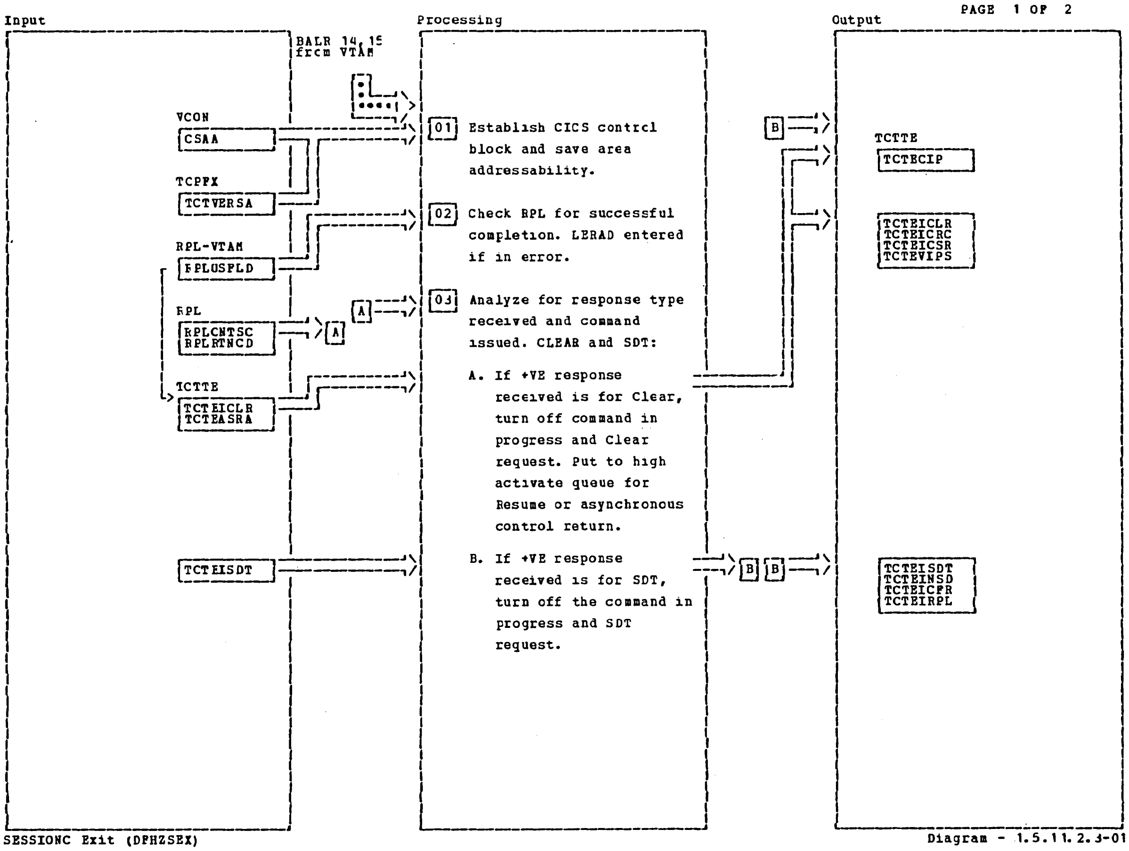
For a VTAM 3270, request a freemain of the RPL without queueing SESSIONC.

OPNDST Exit (DFHZOPX) Diagram - 1.5.11.2.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 TCTTEAR is loaded with the contents of RPLUSFLD. | | | | | | | |
| 02 VTAM inserts a return code in register 15 at completion of the CHECK macro instruction processing indicating the status of the OPNDST request.<br>If register 15 is negative LERAD or SYNAD have done the error processing so we just return. If register 15 has any other code in it, the TCTTE is queued to NACP. | | | | | | | |
| 03 If the previous session was abnormally terminated, the TCTTE is queued for resynchronization processing. If the node is to be brought up out of service, the TCTTE will not be placed on the activate processing chain. Also, if the system is SOS, the RPL will be freed.<br>Resynchronization is not performed for a cold start terminal. | | | | | | | |

OPNDST Exit (DFHZOPX) Diagram - 1.5.11.2.2-01

PAGE 1 OF 2



01 Establish CICS control block and save area addressability.

02 Check RPL for successful completion. LERAD entered if in error.

03 Analyze for response type received and command issued. CLEAR and SDT:

A. If +VE response received is for Clear, turn off command in progress and Clear request. Put to high activate queue for Resume or asynchronous control return.

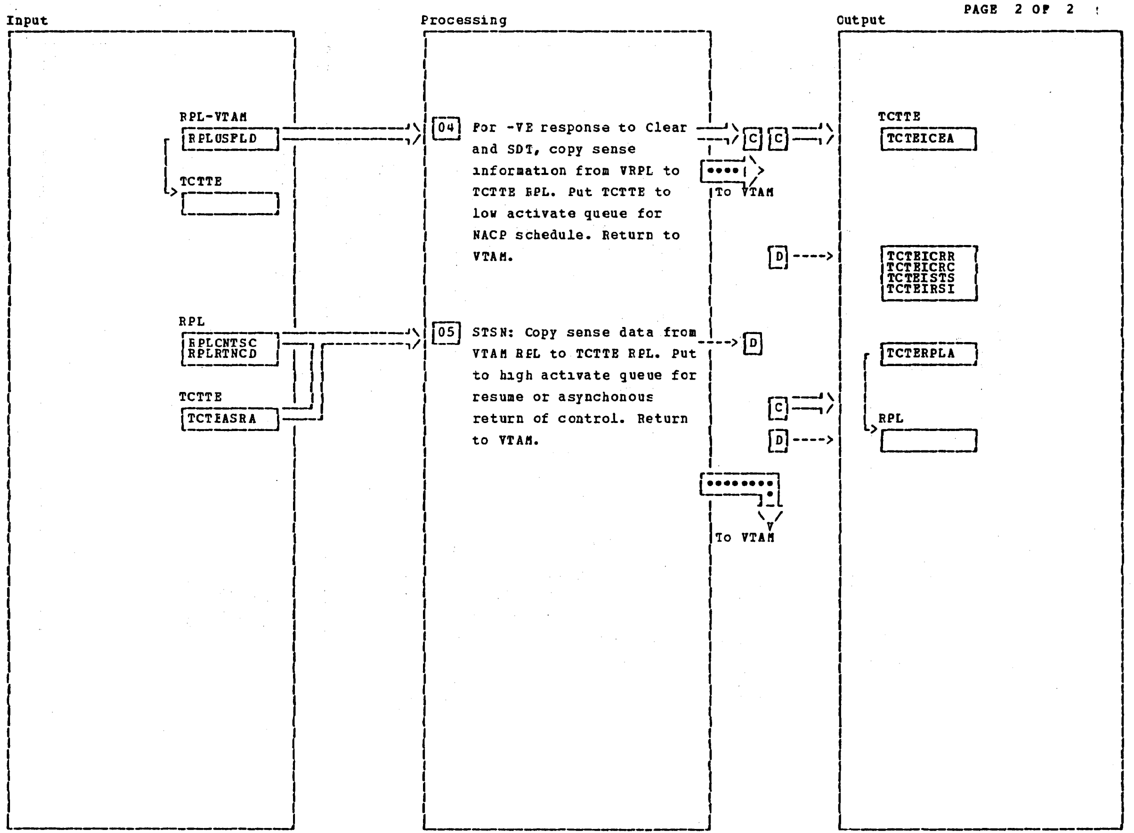
B. If +VE response received is for SDT, turn off the command in progress and SDT request.

SESSIONC Exit (DFHZSEX)  Diagram - 1.5.11.2.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 Completicn of Clear will cause return of control to original requestor of Clear command service. Ccmpletion of SDT makes TCTTE available to CICS for processing. There is no requirement to return to the requestor of SDT. | | | | | | | |

SESSIONC Exit (DFHZSEX)  Diagram - 1.5.11.2.3-01

PAGE 2 OF 2



04 For -VE response to Clear and SDT, copy sense information from VRPL to TCTTE RPL. Put TCTTE to low activate queue for NACP schedule. Return to VTAM.

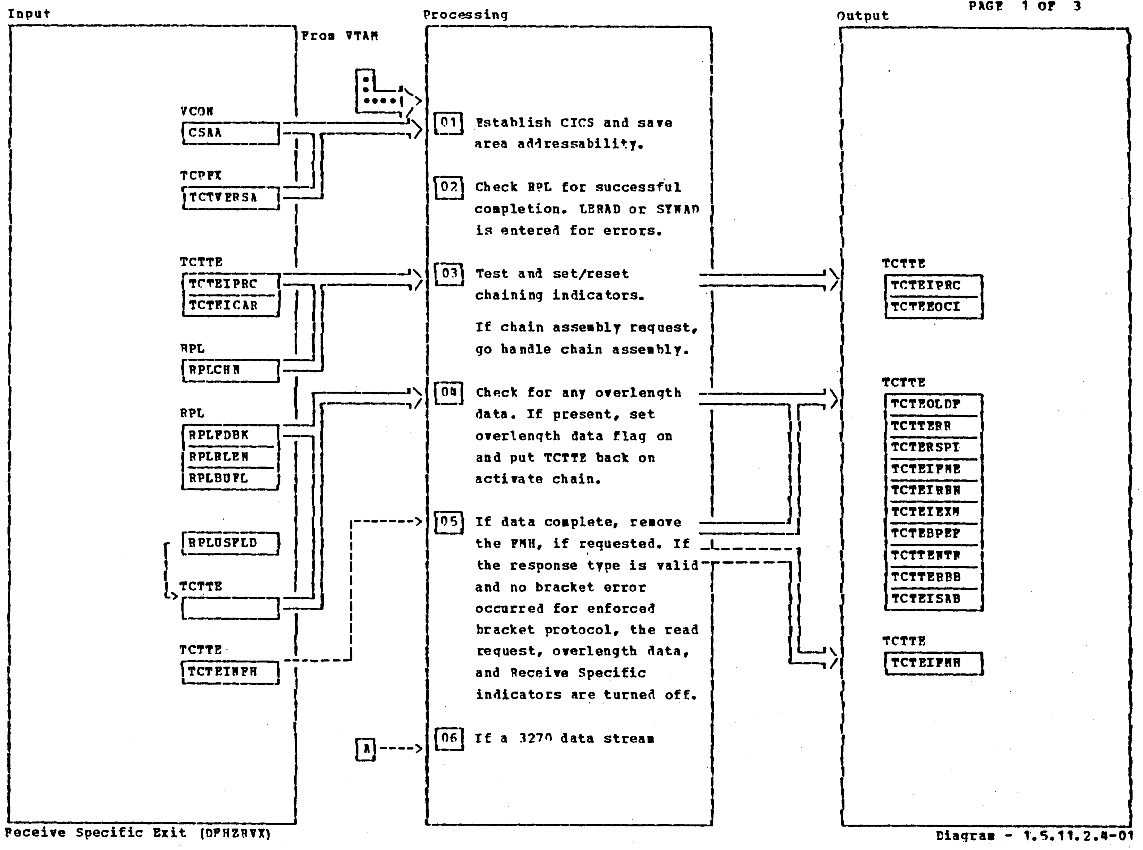
05 STSN: Copy sense data from VTAM RPL to TCTTE RPL. Put to high activate queue for resume or asynchonous return of control. Return to VTAM.

SESSIONC Exit (DFHZSEX) Diagram - 1.5.11.2.3-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 For -VE response, NACP is scheduled to analyze and resolve the errcr condition. The sense information is copied from the VTAM read-only RPL to the RPL attached to the TCTTE. This avoids an extra VTAM Receive merely to obtain the -VE response associated sense informaticn.<br>When ccntrol is returned to VTAM, the VTAM RPL is no longer retained by VTAM. This is why the case data must be copied. | | | | | | | |
| 05 Control will always be returned to the routine requesting the STSN operation, after copying the sense data. Resynchronization routine should be the only requestor of STSN in the current TCP design but future support may require device dependent resynchcnizaticn modules so the STSN service is treated as any other general command service request. | | | | | | | |

SESSIONC Exit (DFHZSEX) Diagram - 1.5.11.2.3-02

CHART: HIPO0001
PAGE 1 OF 3



01 Establish CICS and save area addressability.

02 Check RPL for successful completion. LERAD or SYNAD is entered for errors.

03 Test and set/reset chaining indicators.

If chain assembly request, go handle chain assembly.

04 Check for any overlength data. If present, set overlength data flag on and put TCTTE back on activate chain.

05 If data complete, remove the FMH, if requested. If the response type is valid and no bracket error occurred for enforced bracket protocol, the read request, overlength data, and Receive Specific indicators are turned off.

06 If a 3270 data stream

Peceive Specific Exit (DFHZRVX)

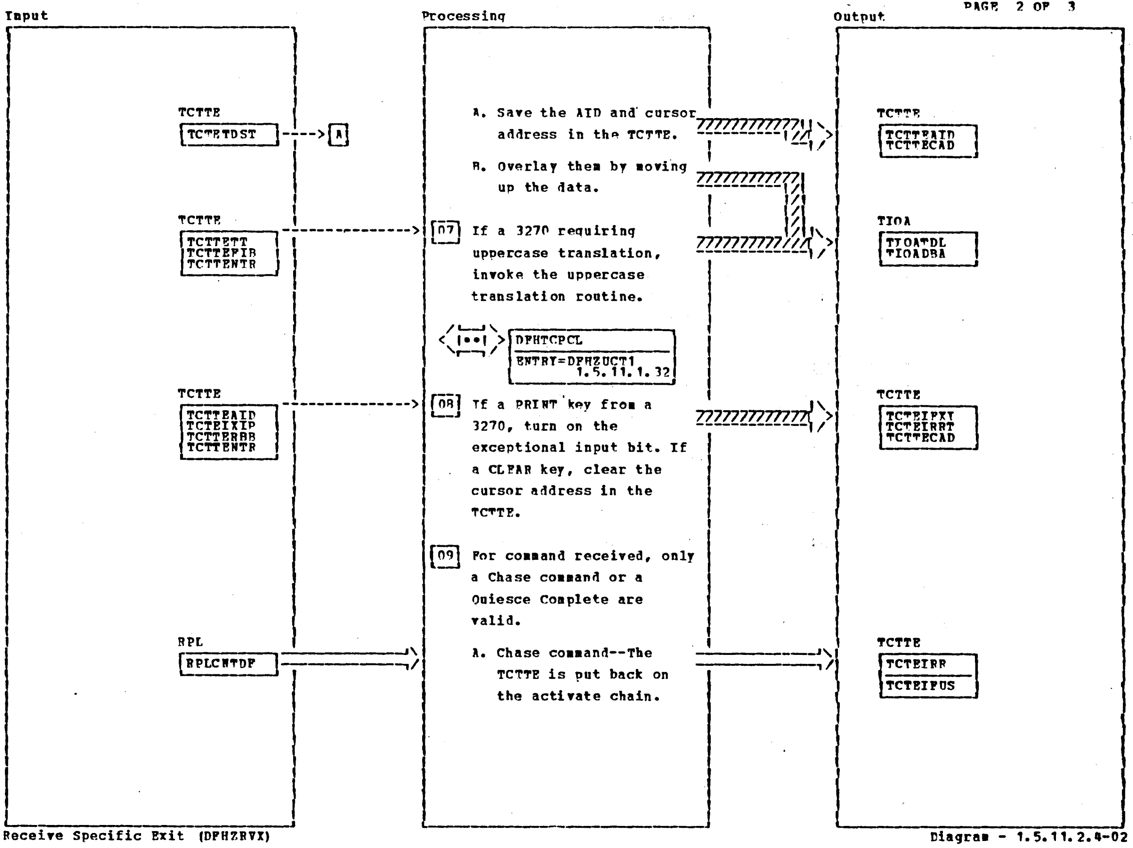
Diagram - 1.5.11.2.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 The FMH is removed according to the specification of INBFMH in the appropriate PCT entry. | | | | | | | |

Receive Specific Exit (DFHZRVX)

Diagram - 1.5.11.2.4-01

CHART: HIPO0001
PAGE 2 OF 3



A. Save the AID and cursor address in the TCTTE.

B. Overlay them by moving up the data.

07 If a 3270 requiring uppercase translation, invoke the uppercase translation routine.

08 If a PRINT key from a 3270, turn on the exceptional input bit. If a CLEAR key, clear the cursor address in the TCTTE.

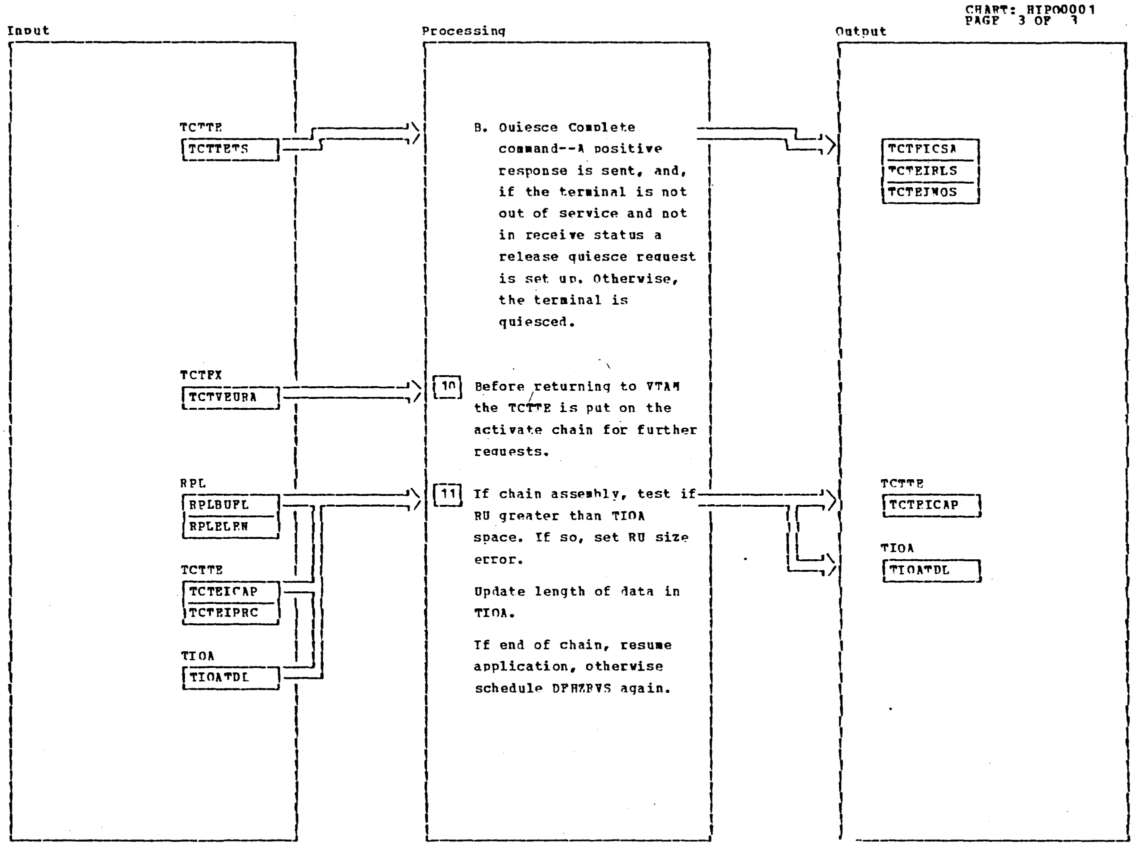
09 For command received, only a Chase command or a Quiesce Complete are valid.

A. Chase command--The TCTTE is put back on the activate chain.

Receive Specific Exit (DFHZRVX) Diagram - 1.5.11.2.4-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 09 | | TCZRVX20 | 1.5.11<br>.1.32 | | | | |

Receive Specific Exit (DFHZRVX) Diagram - 1.5.11.2.4-02



B. Ouiesce Complete command--A positive response is sent, and, if the terminal is not out of service and not in receive status a release quiesce request is set up. Otherwise, the terminal is quiesced.

10 Before returning to VTAM the TCTTE is put on the activate chain for further requests.

11 If chain assembly, test if RU greater than TIOA space. If so, set RU size error.

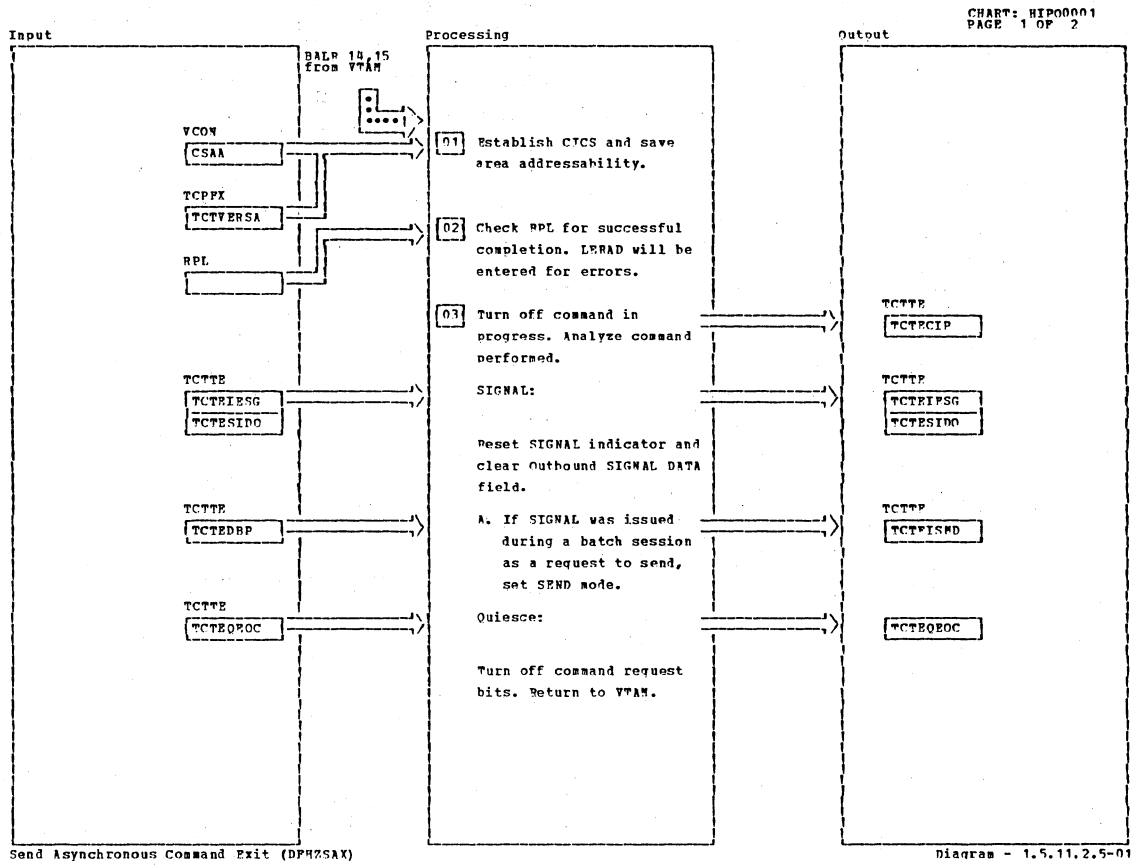
Update length of data in TIOA.

If end of chain, resume application, otherwise schedule DFHZRVS again.

Receive Specific Exit (DFHZRVX) Diagram - 1.5.11.2.4-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 11 Error code for RU size error is TCZRARS set in TCTEVRC7. | | | | | | | |

Receive Specific Exit (DFHZRVX) Diagram - 1.5.11.2.4-03



01 Establish CICS and save area addressability.

02 Check RPL for successful completion. LERAD will be entered for errors.

03 Turn off command in progress. Analyze command performed.

SIGNAL:

Reset SIGNAL indicator and clear Outbound SIGNAL DATA field.

A. If SIGNAL was issued during a batch session as a request to send, set SEND mode.

Quiesce:

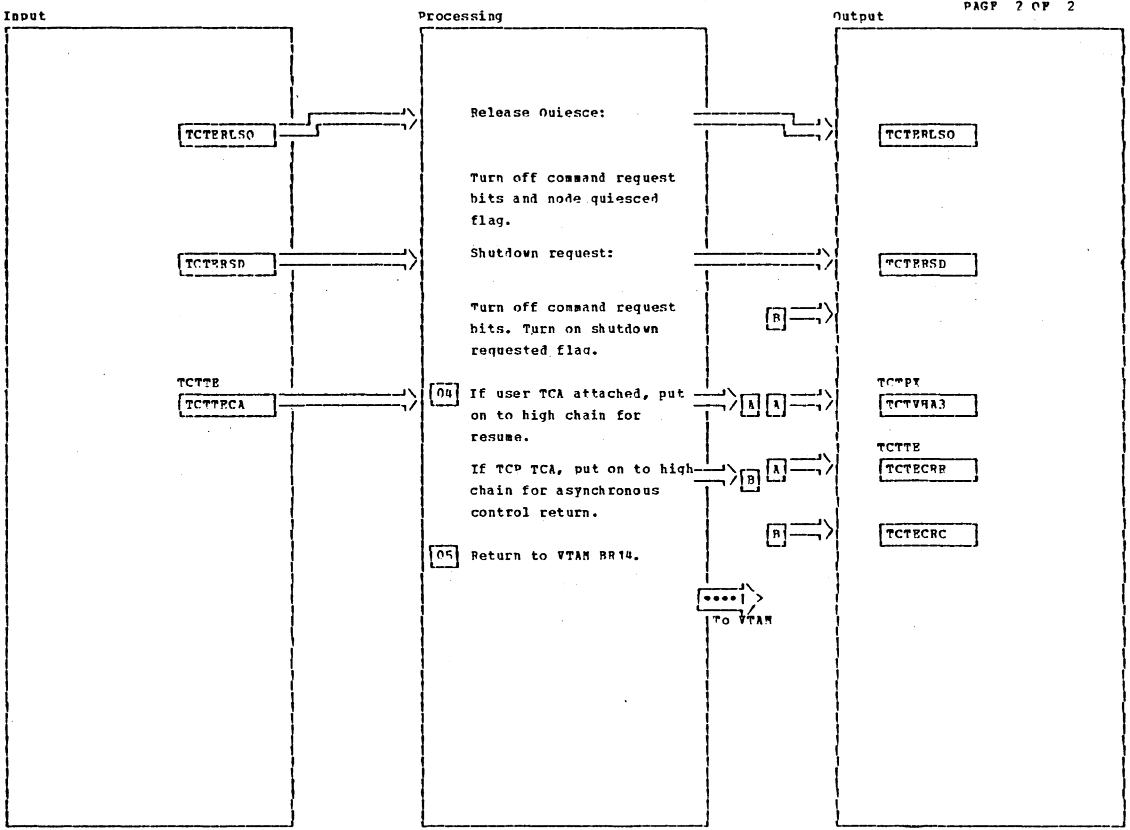
Turn off command request bits. Return to VTAM.

Send Asynchronous Command Exit (DFHZSAX) Diagram - 1.5.11.2.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

Send Asynchronous Command Exit (DFHZSAX) Diagram - 1.5.11.2.5-01

CHART: HIPO0001
PAGE 2 OF 2



Release Quiesce:

Turn off command request bits and node quiesced flag.

Shutdown request:

Turn off command request bits. Turn on shutdown requested flag.

04 If user TCA attached, put on to high chain for resume.

If TCP TCA, put on to high chain for asynchronous control return.

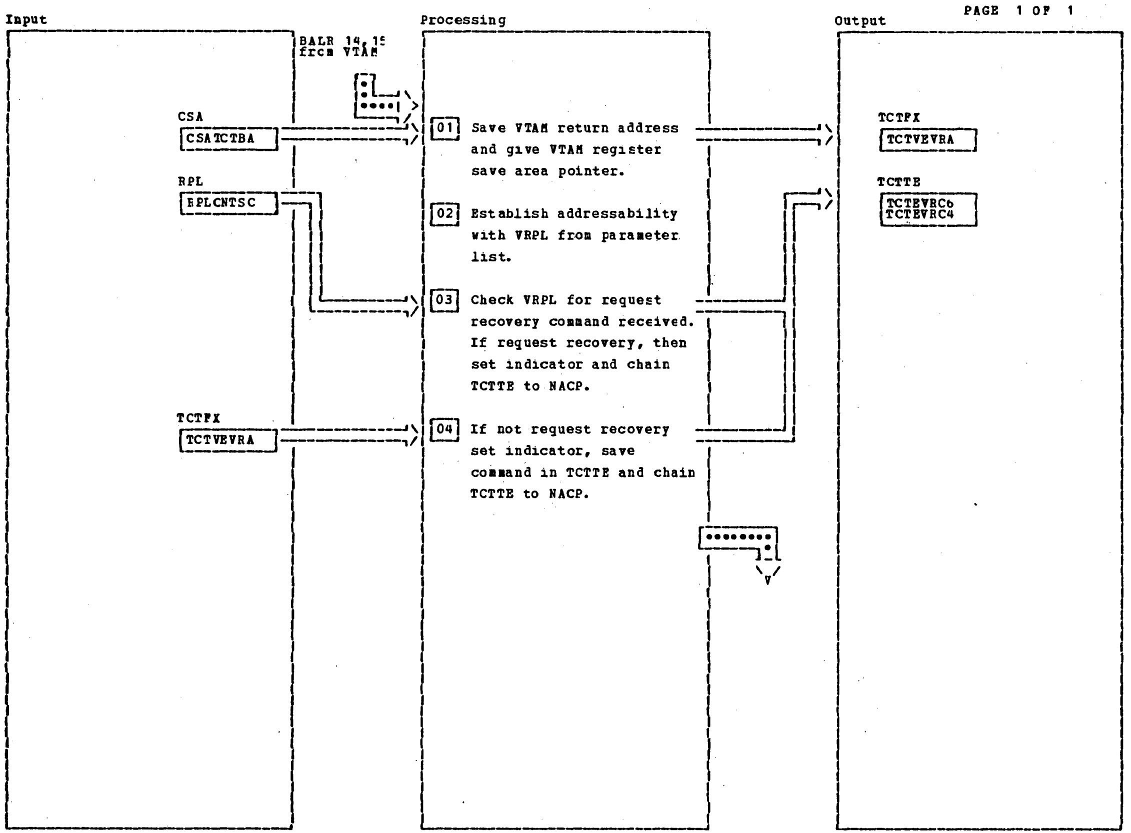
05 Return to VTAM BR14.

Send Asynchronous Command Exit (DFHZSAX) Diagram - 1.5.11.2.5-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Send Asynchronous Command Exit (DFHZSAX) Diagram - 1.5.11.2.5-02

PAGE 1 OF 1



01 Save VTAM return address and give VTAM register save area pointer.

02 Establish addressability with VRPL from parameter list.

03 Check VRPL for request recovery command received. If request recovery, then set indicator and chain TCTTE to NACP.

04 If not request recovery set indicator, save command in TCTTE and chain TCTTE to NACP.

SESSION Control Input Exit (DFHZSCX)

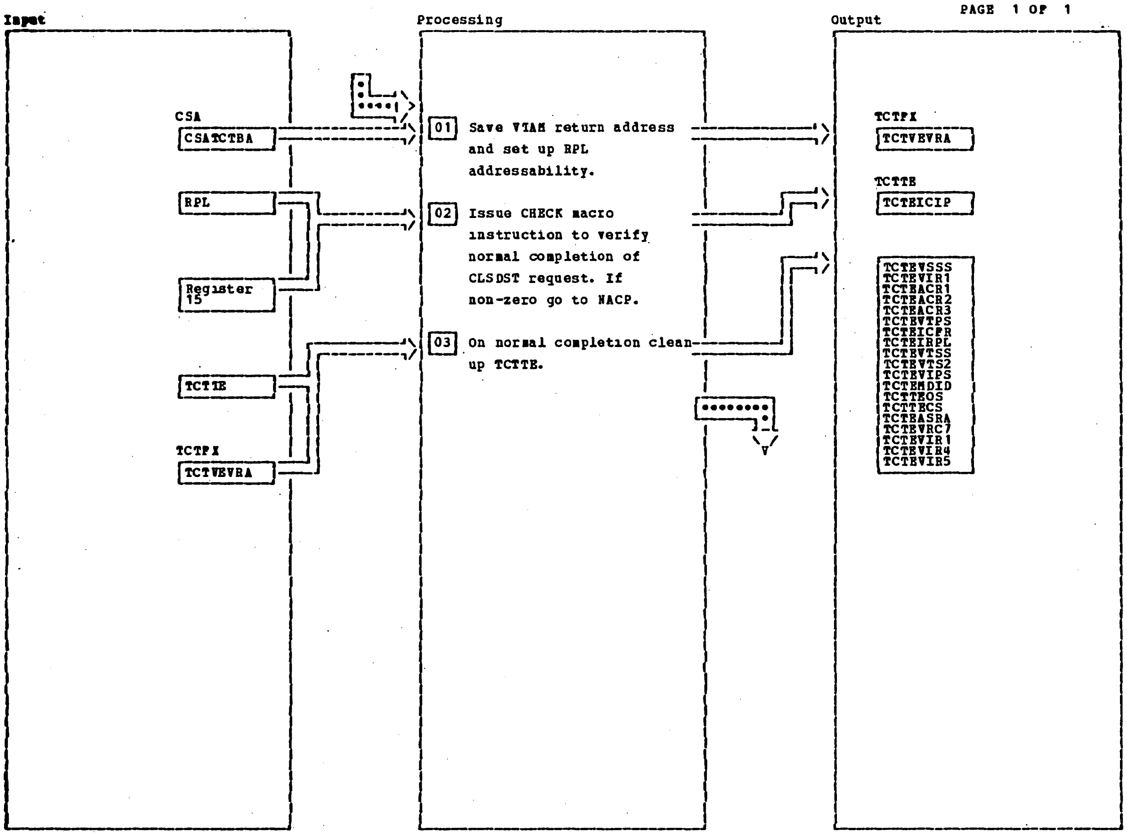
Diagram - 1.5.11.2.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 To chain to NACP, issue DFHTCQUE QUE=NACP macro instruction. | | | | | | | |
| 04 To chain to NACP, issue DFHTCQUE QUE=NACP macro instruction. | | | | | | | |

SESSION Control Input Exit (DFHZSCX)

Diagram - 1.5.11.2.6-01

PAGE 1 OF 1



01 Save VTAM return address and set up RPL addressability.

02 Issue CHECK macro instruction to verify normal completion of CLSDST request. If non-zero go to NACP.

03 On normal completion clean-up TCTTE.

CLSDST Exit (DFHZCLX)

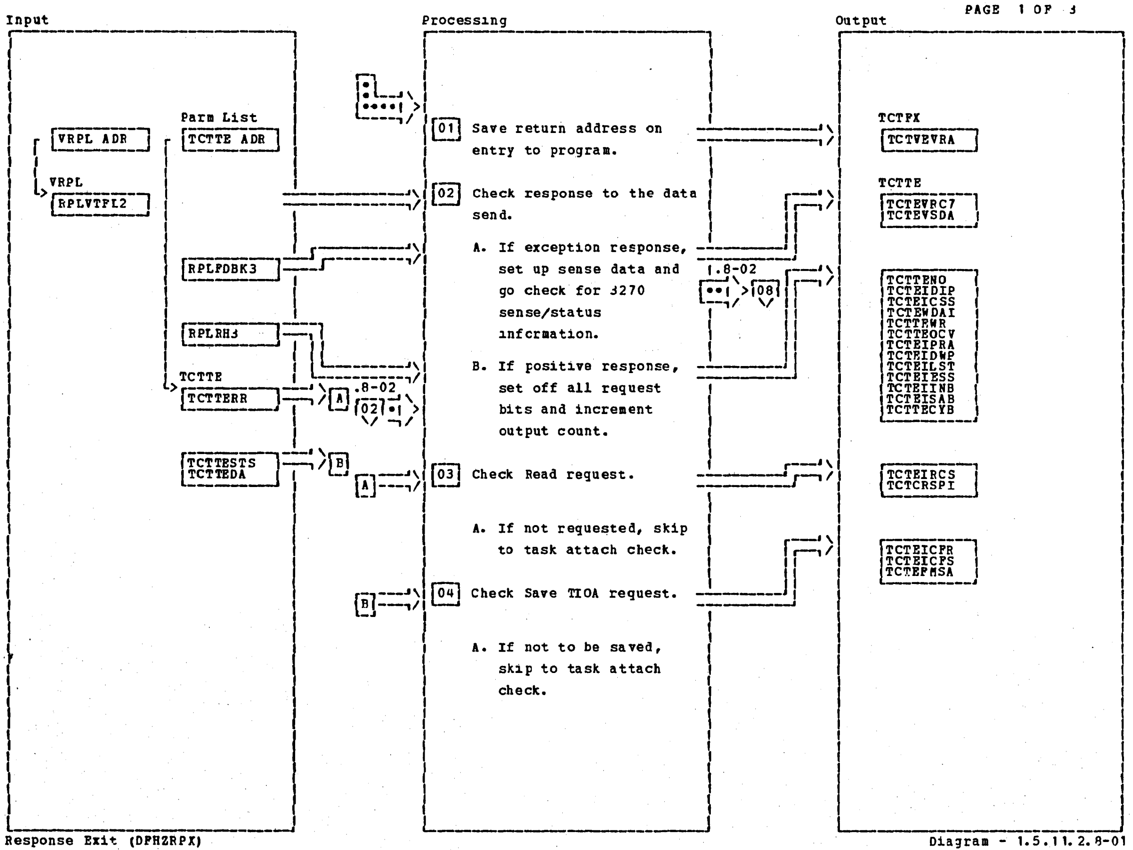
Diagram - 1.5.11.2.7-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 VTAM passes the address of the RPL that the CLSDST was issued against. | | | | | | | |
| 02 In order to verify good completion of the CLSDST, request the CHECK macro instruction is issued. If a non-zero return code is passed back in register 15, the TCTTE is passed to NACP for error processing. | | | | | | | |
| 03 On zero completion of the CHECK macro instruction, turn off all non-generated TCTTE bits. | | | | | | | |

CLSDST Exit (DFHZCLX)

Diagram - 1.5.11.2.7-01

PAGE 1 OF 3



01 Save return address on entry to program.

02 Check response to the data send.

A. If exception response, set up sense data and go check for 3270 sense/status infcrmation.

B. If positive response, set off all request bits and increment output count.

03 Check Read request.

A. If not requested, skip to task attach check.

04 Check Save TIOA request.

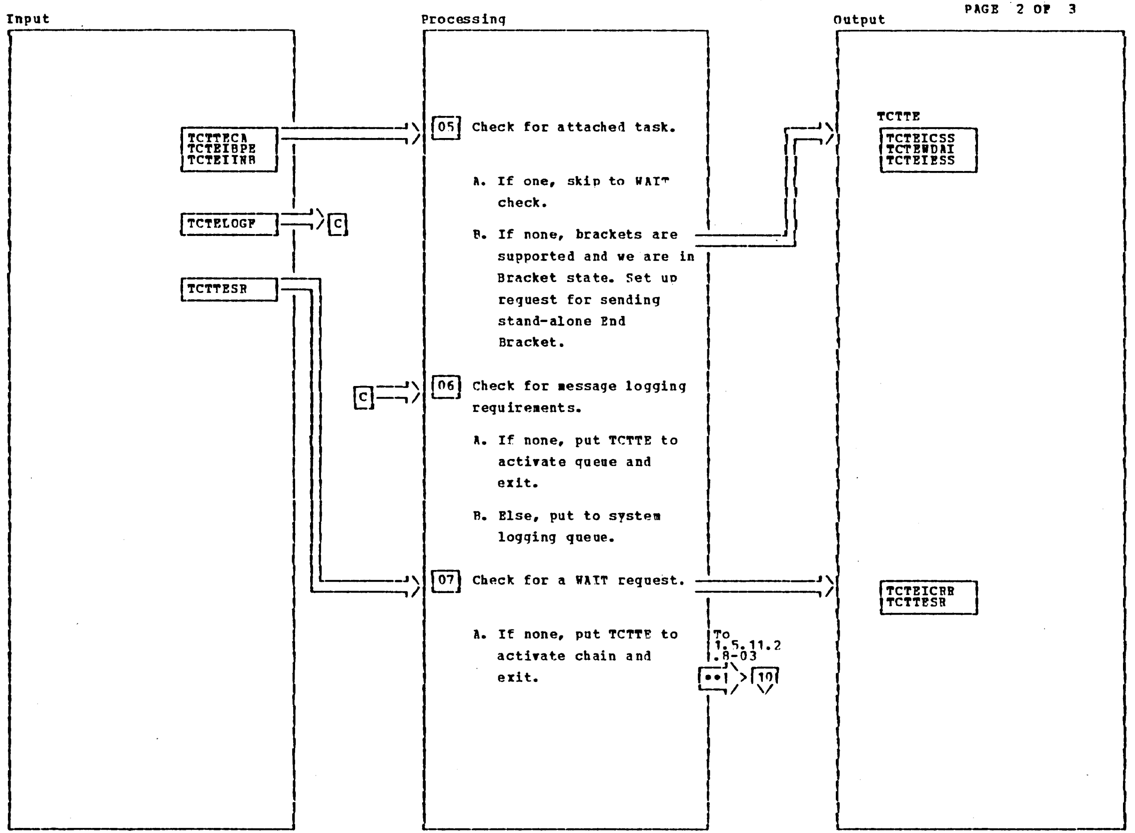
A. If not to be saved, skip to task attach check.

Response Exit (DFHZRPX) Diagram - 1.5.11.2.8-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Response Exit (DFHZRPX) Diagram - 1.5.11.2.8-01

PAGE 2 OF 3



05 Check for attached task.

A. If one, skip to WAIT check.

B. If none, brackets are supported and we are in Bracket state. Set up request for sending stand-alone End Bracket.

06 Check for message logging requirements.

A. If none, put TCTTE to activate queue and exit.

B. Else, put to system logging queue.

07 Check for a WAIT request.

A. If none, put TCTTE to activate chain and exit.

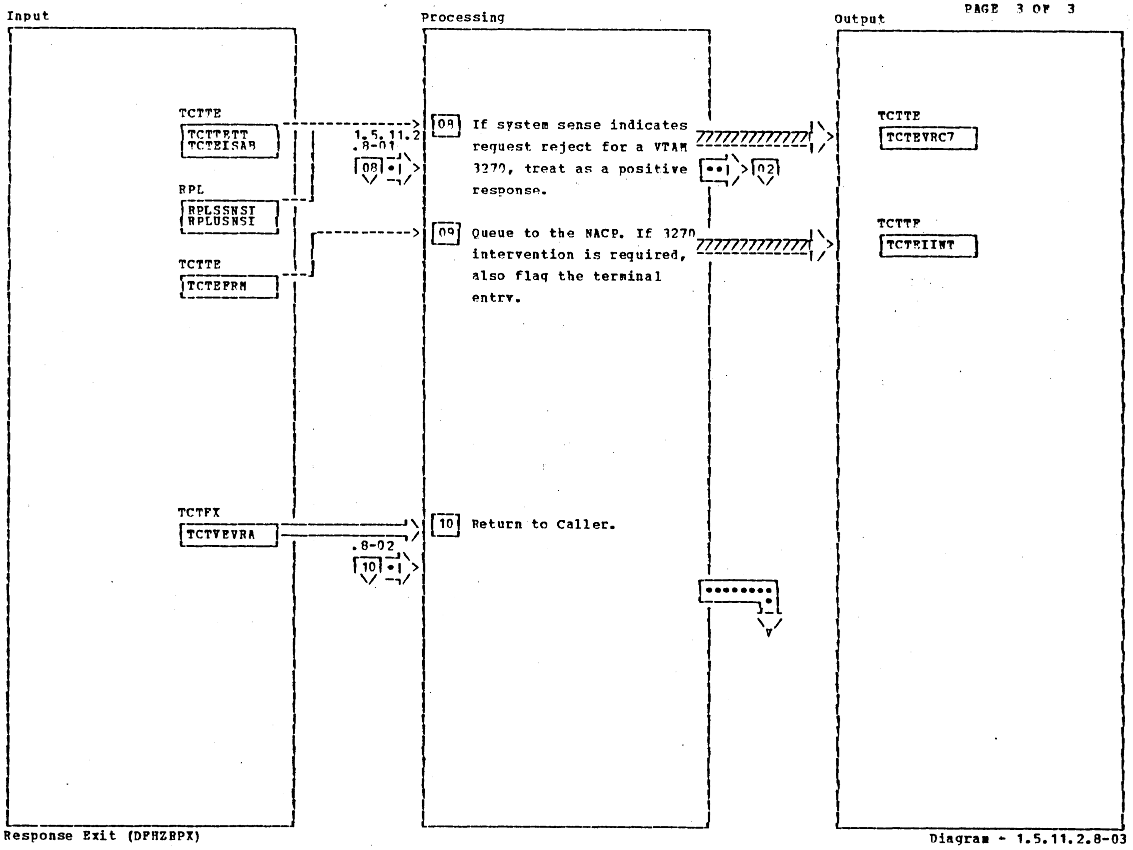
Response Exit (DFHZRPX) Diagram - 1.5.11.2.8-02

| Notes | Routine | Label | Ref |
|---|---|---|---|

| Notes | Routine | Label | Ref |
|---|---|---|---|

Response Exit (DFHZRPX) Diagram - 1.5.11.2.8-02

PAGE 3 OF 3



08 If system sense indicates request reject for a VTAM 3270, treat as a positive response.

09 Queue to the NACP. If 3270 intervention is required, also flag the terminal entry.

10 Return to Caller.

Response Exit (DFHZRPX)

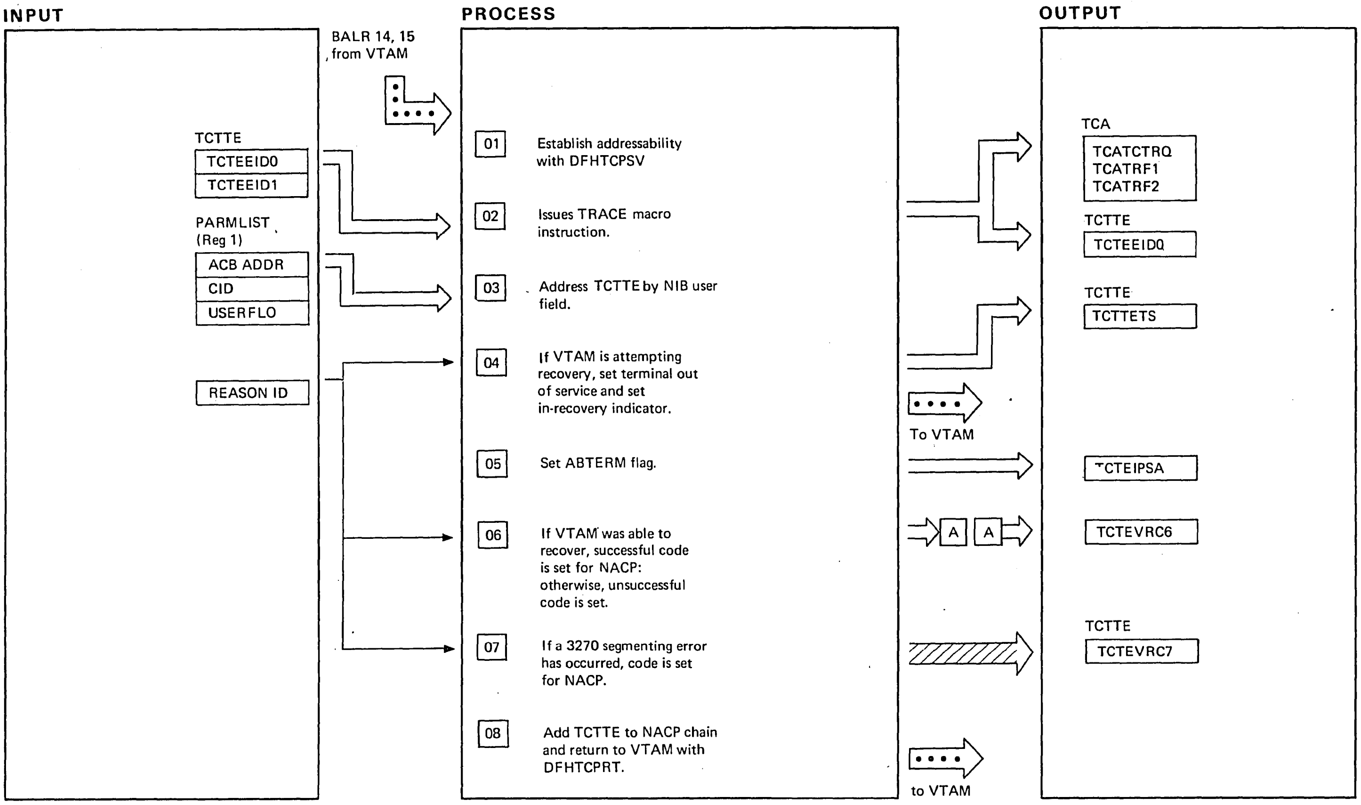
Diagram - 1.5.11.2.8-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Response Exit (DFHZRPX)

Diagram - 1.5.11.2.8-03

PAGE 1 OF 2



01 Establish addressability with DFHTCPSV

02 Issues TRACE macro instruction.

03 Address TCTTE by NIB user field.

04 If VTAM is attempting recovery, set terminal out of service and set in-recovery indicator.

05 Set ABTERM flag.

06 If VTAM was able to recover, successful code is set for NACP: otherwise, unsuccessful code is set.

07 If a 3270 segmenting error has occurred, code is set for NACP.

08 Add TCTTE to NACP chain and return to VTAM with DFHTCPRT.

LOSTERM Exit DFHZLTX

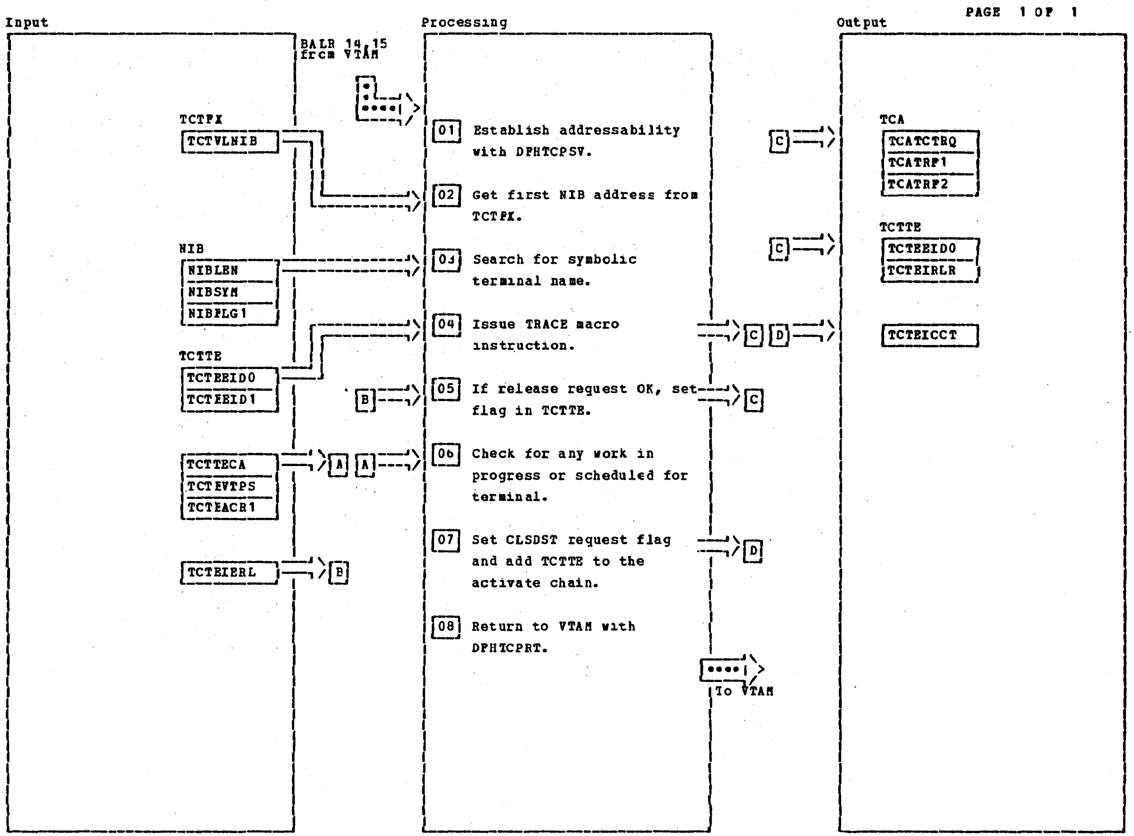
Diagram — 1.5.11.2.9-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

LOSTERM Exit DFHZLTX

Diagram — 1.5.11.2.9-01



01 Establish addressability with DFHTCPSV.

02 Get first NIB address from TCTFX.

03 Search for symbolic terminal name.

04 Issue TRACE macro instruction.

05 If release request OK, set flag in TCTTE.

06 Check for any work in progress or scheduled for terminal.

07 Set CLSDST request flag and add TCTTE to the activate chain.

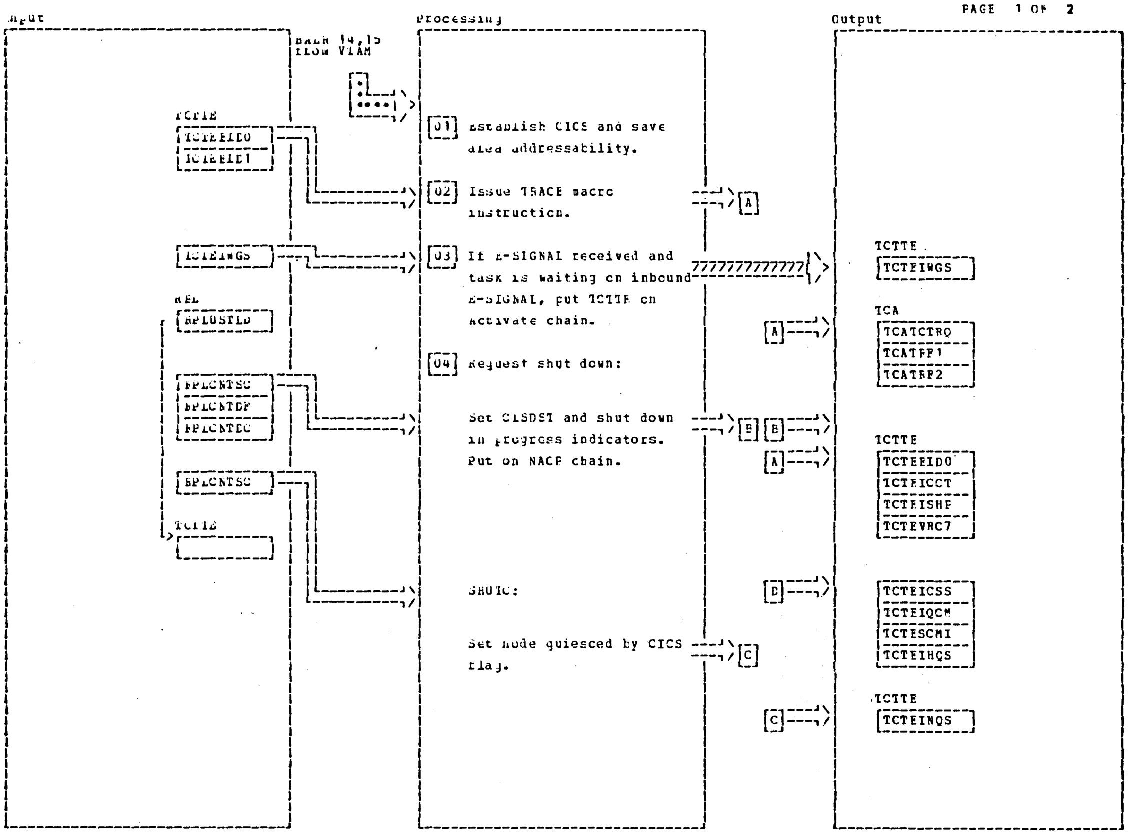
08 Return to VTAM with DFHTCPRT.

Release Request Exit (DFHZRRX) Diagram - 1.5.11.2.10-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 If no match is found, control is returned to VTAM with no further processing. | | | | | | | |
| 06 If any work is in progress or scheduled for terminal, control is returned to VTAM. | | | | | | | |

Release Request Exit (DFHZRRX) Diagram - 1.5.11.2.10-01

PAGE 1 OF 2



01 Establish CICS and save area addressability.

02 Issue TRACE macro instruction.

03 If E-SIGNAL received and task is waiting on inbound E-SIGNAL, put TCTTE on Activate chain.

04 Request shut down:

Set CLSDST and shut down in progress indicators. Put on NACP chain.

SHUTC:

Set node quiesced by CICS flag.

DFASY Exit (DFHZASX)

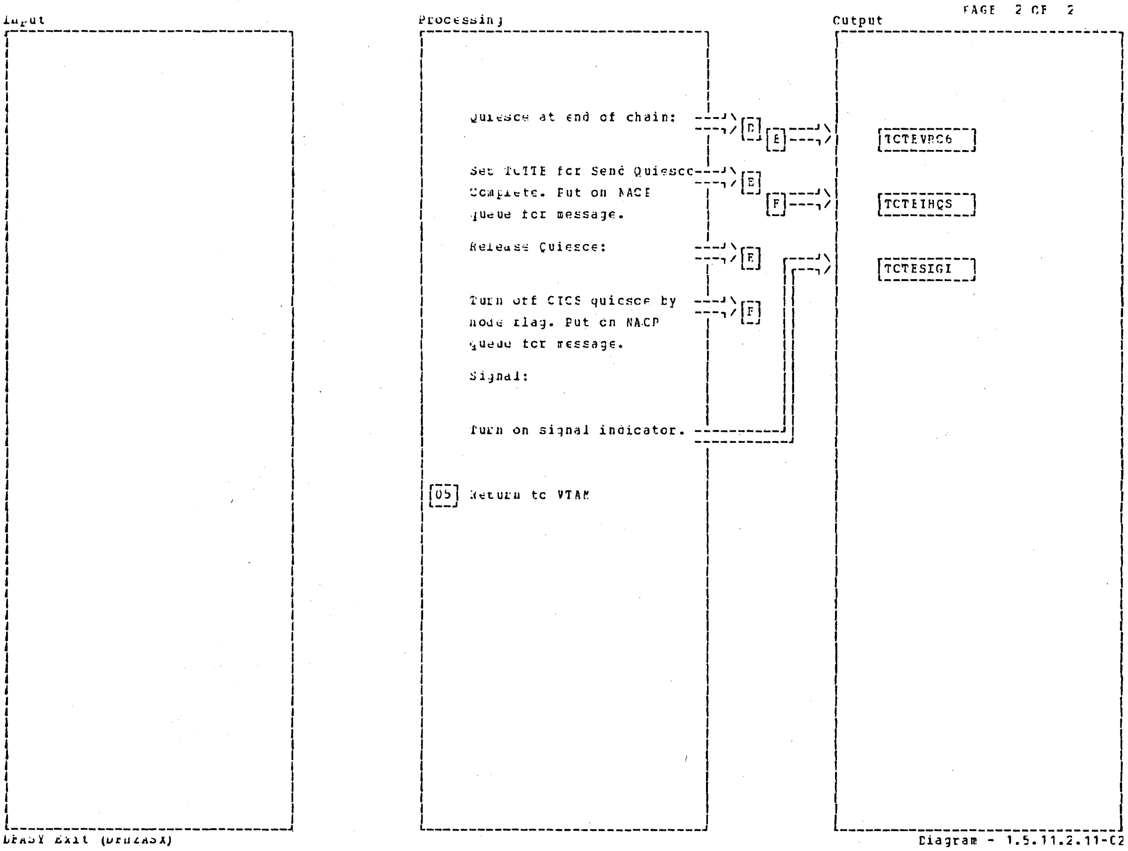
Diagram - 1.5.11.2.11-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

DFASY Exit (DFHZASX)

Diagram - 1.5.11.2.11-C1

PAGE 2 OF 2



Quiesce at end of chain:

Set TCTTE for Send Quiesce Complete. Put on NACP queue for message.

Release Quiesce:

Turn off CICS quiesce by node flag. Put on NACP queue for message.

Signal:

Turn on signal indicator.

05 Return to VTAM

DFASY Exit (DFHZASX)

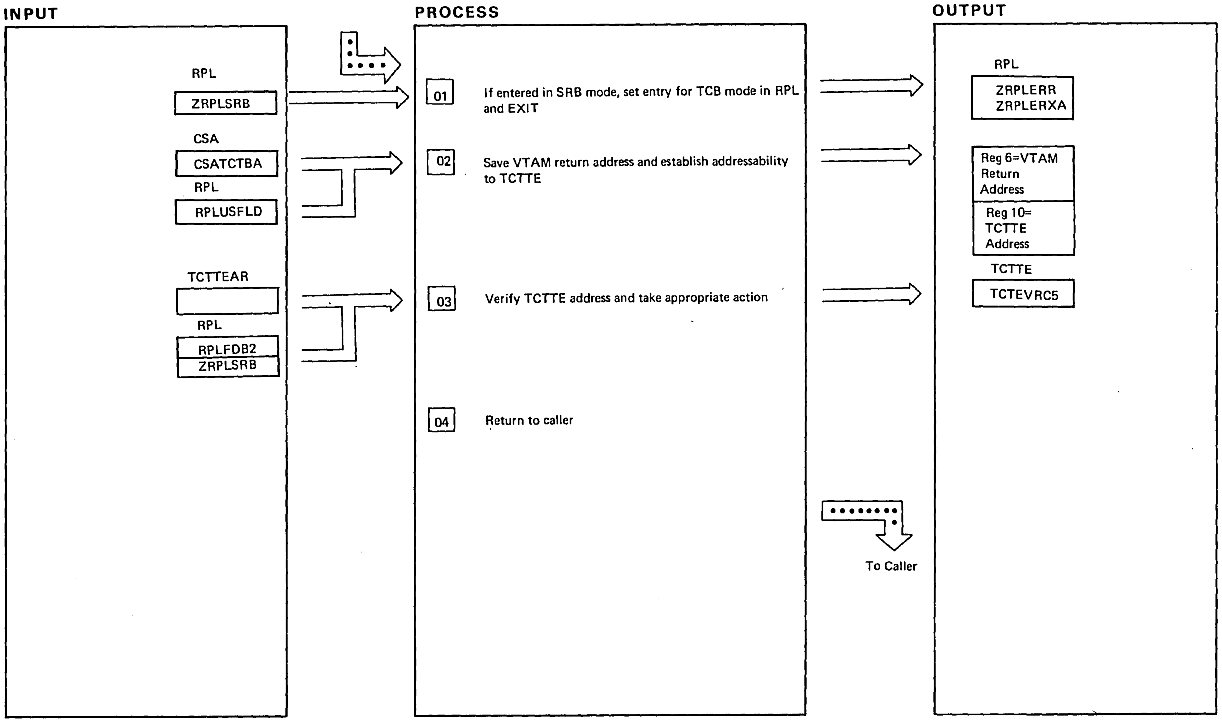
Diagram - 1.5.11.2.11-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

DFASY Exit (DFHZASX)

Diagram - 1.5.11.2.11-02

Page 1 of 1



01 If entered in SRB mode, set entry for TCB mode in RPL and EXIT

02 Save VTAM return address and establish addressability to TCTTE

03 Verify TCTTE address and take appropriate action

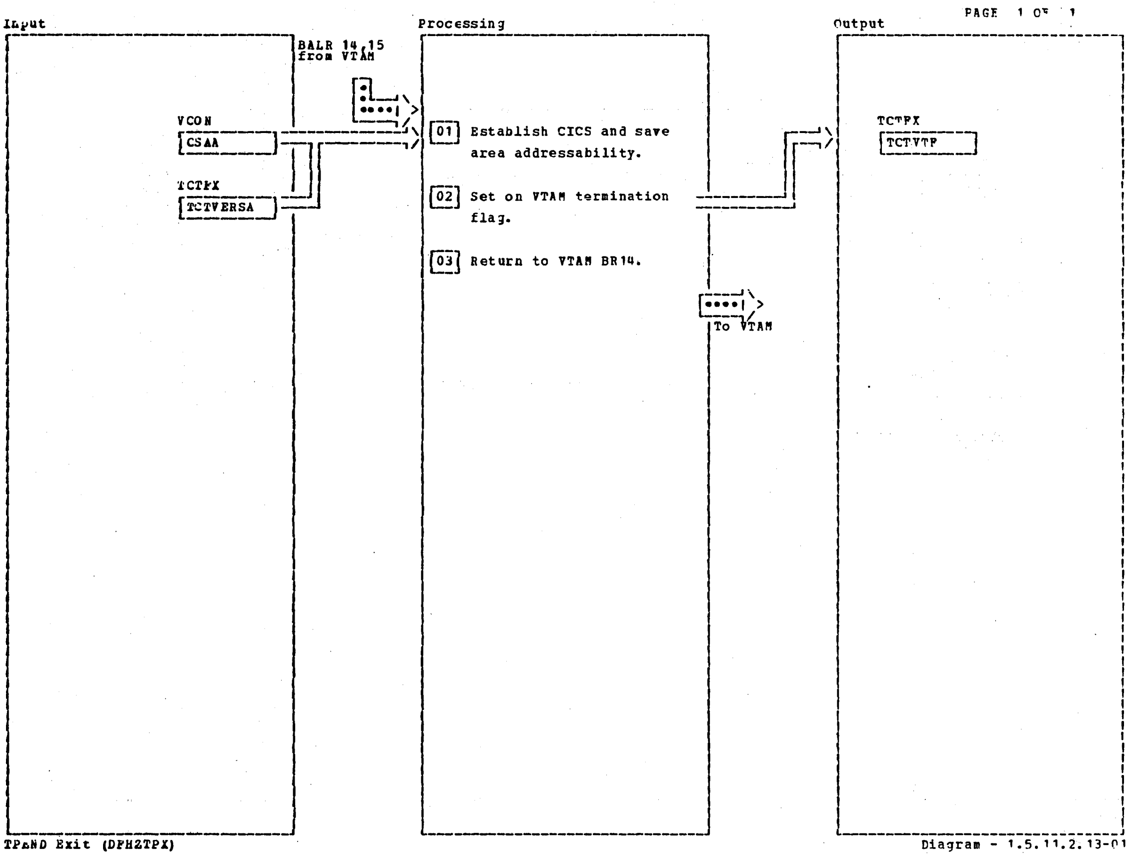
04 Return to caller

LERAD Exit (DFHZLEX) 1.5.11.2.12-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Save VTAM return address in a register. LERAD must be reentrant. | | | |
| 02 If TCTTEAR is zero, VTAM did not correctly fill in the RPLUSFLD that will cause a dump to be initiated | | TCZIEXD | |
| If TCTTEEAR has an address in it and feedback information is available, the TCTTE is passed toNACP. | | TCZLEXNO | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

LERAD Exit (DFHZLEX) 1.5.11.2.12-01



01 Establish CICS and save area addressability.

02 Set on VTAM termination flag.

03 Return to VTAM BR14.

TPEND Exit (DFHZTPX)

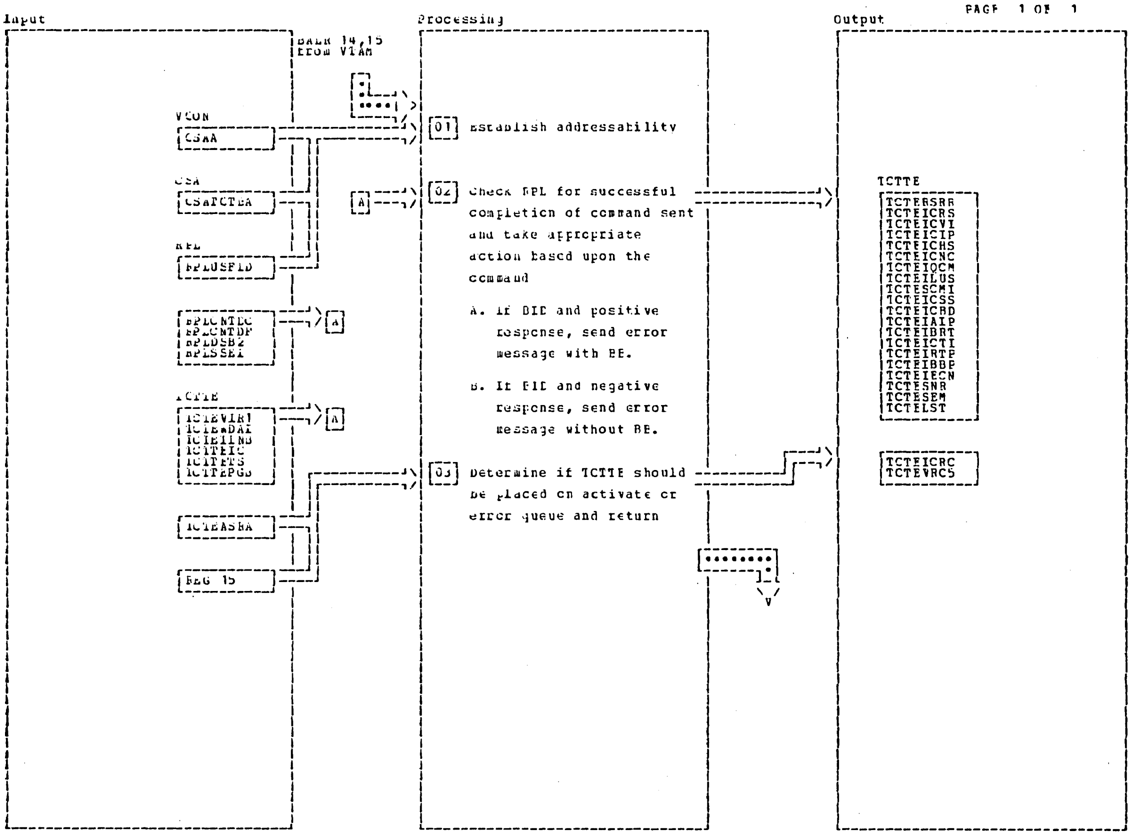
Diagram - 1.5.11.2.13-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

TPEND Exit (DFHZTPX)

Diagram - 1.5.11.2.13-01

PAGE 1 OF 1



01 Establish addressability

02 Check RPL for successful completion of command sent and take appropriate action based upon the command

A. If BID and positive response, send error message with BE.

B. If BID and negative response, send error message without BE.

03 Determine if TCTTE should be placed on activate or error queue and return

Send Synchronous Data Flow Exit (DFHZSSX)

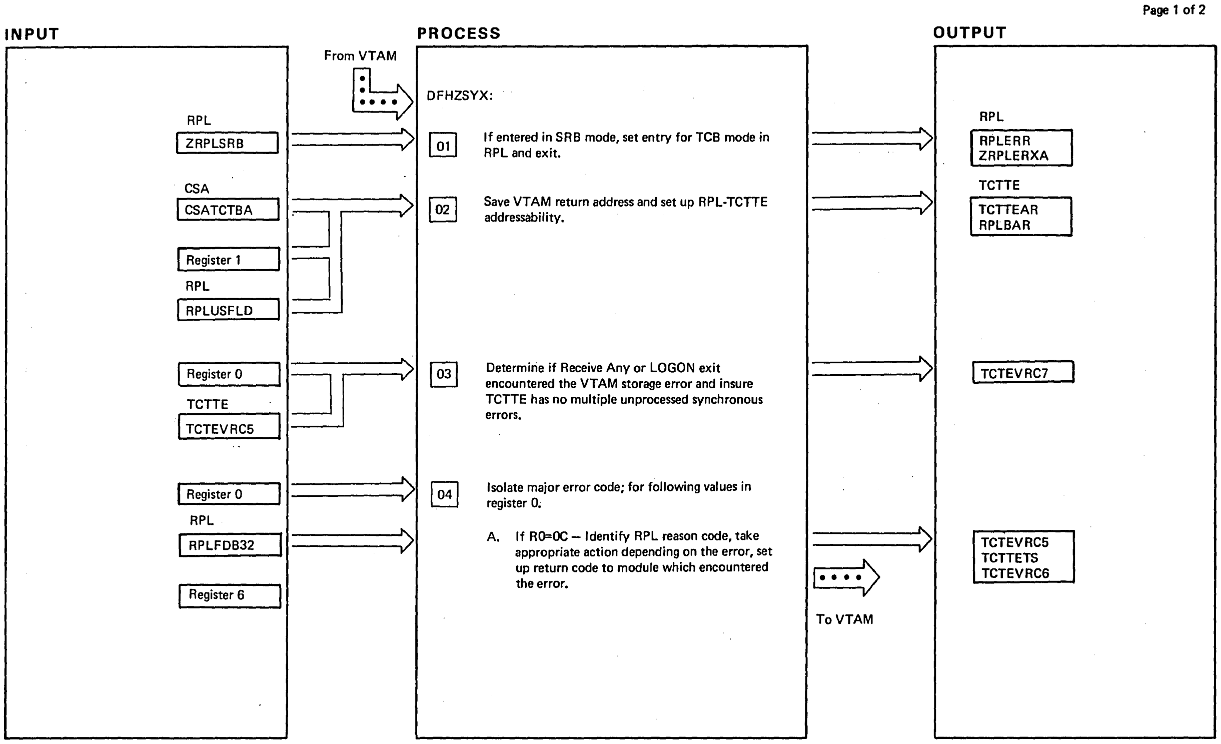
Diagram - 1.5.11.2.14-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Addressability with the TCT prefix and the TCTTE must be provided. |  | TCZSSXER |  |  |  |  |  |
| 02 The VTAM CHECK macro is issued on behalf of the RPL in question. With a normal completion (R15=0) it must be determined which command was sent. |  | TCZSSX10<br>TCZSSX20 |  |  |  |  |  |
| If BID was sent the RPL sense codes are checked to see if the logical unit sent ready to receive (RTR). If so the TCTTE is set up for transaction initiation. If not an error is indicated. |  | TCZSSX50 |  |  |  |  |  |
| Upon receipt of CHASE, CANCEL, QUIESCE COMPLETE, or LU status the request bit is turned off in the TCTTE. |  | TCZSSX22<br>TCZSSX24<br>TCZSSX26 |  |  |  |  |  |
| 03 If any further processing is required, the TCTTE is placed on the activate queue. In case of an error, the TCTTE is placed on the system error queue. |  | TCZSSX30<br>TCZSSX32<br>TCZSSX59 |  |  |  |  |  |

Send Synchronous Data Flow Exit (DFHZSSX)

Diagram - 1.5.11.2.14-01

Page 1 of 2



DFHZSYX:

01 If entered in SRB mode, set entry for TCB mode in RPL and exit.

02 Save VTAM return address and set up RPL-TCTTE addressability.

03 Determine if Receive Any or LOGON exit encountered the VTAM storage error and insure TCTTE has no multiple unprocessed synchronous errors.

04 Isolate major error code; for following values in register 0.

A. If R0=0C — Identify RPL reason code, take appropriate action depending on the error, set up return code to module which encountered the error.

SYNAD Exit (DFHZSYX)

Diagram - 1.5.11.2. 15-01

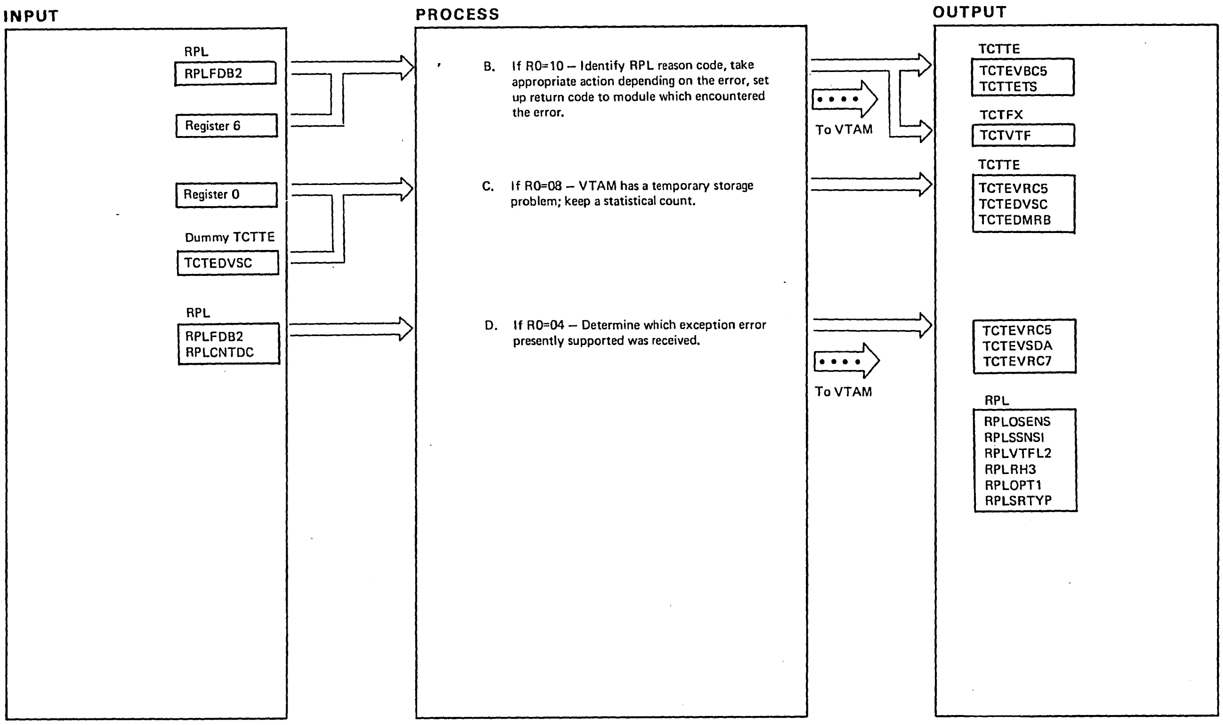
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 Since SYNAD must be reentrant, the VTAM return address must be saved in a register. | | DFHZSYZ1 | |
| RPLUSFLD contains its corresponding TCTTE address. | | | |
| 03 If RPLUSFLD is zero and register 0 = 08 (VTAM storage problem) SYNAD assumes Receive Any encountered the error. | | DFHZSYX1 | |
| If RPLUSFLD is X'FF's, SYNAD assumes LOGON exit encountered an error. | | TCZSYX00 | |
| Most important is when TCTEVRC5 already contains an other error code. This is an indication that a previously encountered error has not been processed, thus indicating catastrophic errors. | | TCZAYXA0 | |
| 04 The major error code is isolated and unique major code routines are branched to, depending on that error. | | TCZSYXA1 | |
| A. A reson code which identifies the error, each anticipated reason code has an associated routine which passes a return code with the terminal entry to NACP as well as to the module which encountered the error in register 15. | | TCZSYXA1 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

SYNAD Exit (DFHZSYX)

Diagram - 1.5.11.2. 15-01

Page 2 of 2



B. If R0=10 – Identify RPL reason code, take appropriate action depending on the error, set up return code to module which encountered the error.

C. If R0=08 – VTAM has a temporary storage problem; keep a statistical count.

D. If R0=04 – Determine which exception error presently supported was received.

SYNAD Exit (DFHZSYX)

Diagram - 1.5.11.2. 15-02

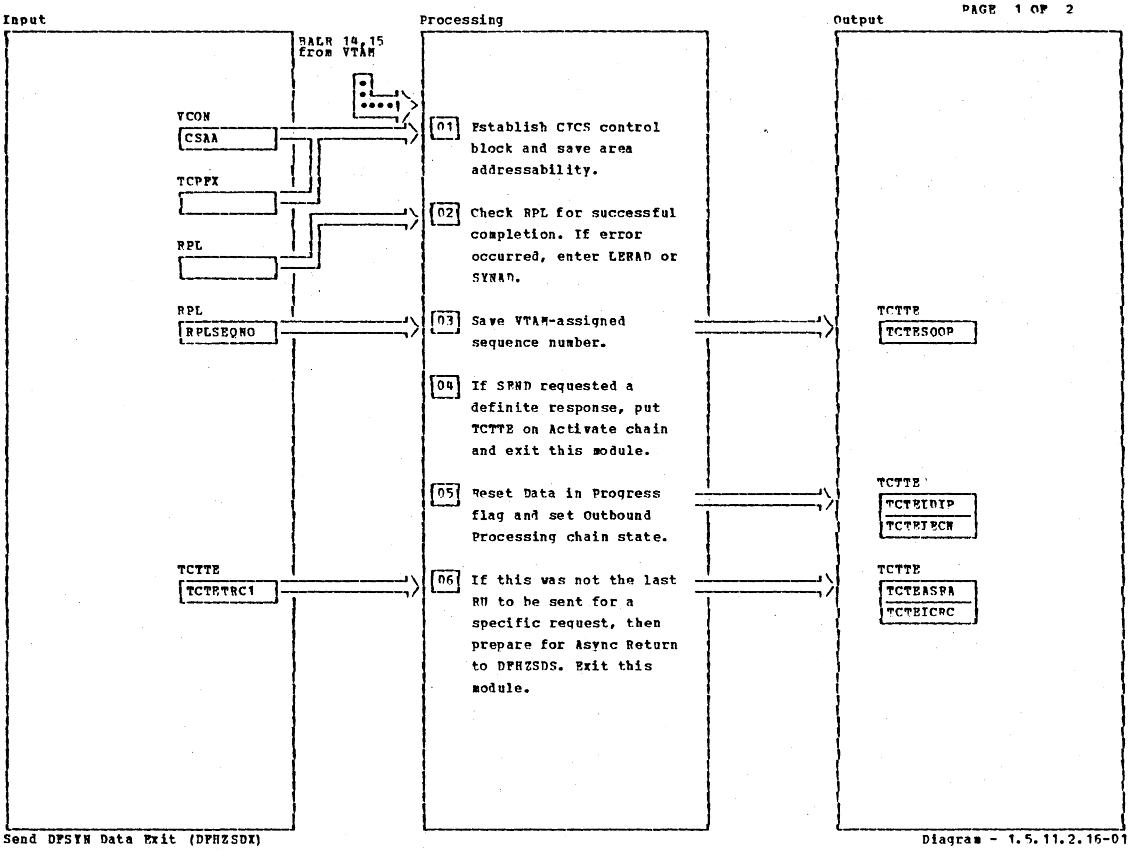
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| B. A reason code which identifies the error, each anticipated reason code has an associated routine which passes a return code with the terminal entry to NACP as well as to the module which encountered the error in register 15. | | TCZSYX00 | |
| C. This is not an error condition but a way for VTAM to tell the application program (CICS) that temporarily it does not have enough storage to honor a request.<br>CICS keeps track of how many times this condition is encountered in the dummy TCTTE. | | | |
| D. If an exception request required indication was received, SYNAD exit will send an exception response with the sense codes received in the original request. | | TCZSYXF3 | |
| However, if an exception-response-received indication is received and the exception is to a previous BID request, control is returned to DFHZSSX, which will handle its own error recovery. If it is other than a BID, an error code is set and the TCTTE is placed on the system error queue. | | TCZSYXFC | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| If an exception request containing 3270 sense/status information is received, the information is checked for the presence of the intervention-required and device-end conditions. If intervention-required is present, the terminal entry is flagged before it is queued to the NACP. If device-end is present, the foregoing intervention-required flag is turned off and the caller is returned-to with a special return code. | | | |

SYNAD Exit (DFHZSYX)

Diagram - 1.5.11.2. 15-02

PAGE 1 OF 2



01 Establish CICS control block and save area addressability.

02 Check RPL for successful completion. If error occurred, enter LERAD or SYNAD.

03 Save VTAM-assigned sequence number.

04 If SEND requested a definite response, put TCTTE on Activate chain and exit this module.

05 Reset Data in Progress flag and set Outbound Processing chain state.

06 If this was not the last RU to be sent for a specific request, then prepare for Async Return to DFHZSDS. Exit this module.

Send DFSYN Data Exit (DFHZSDX)

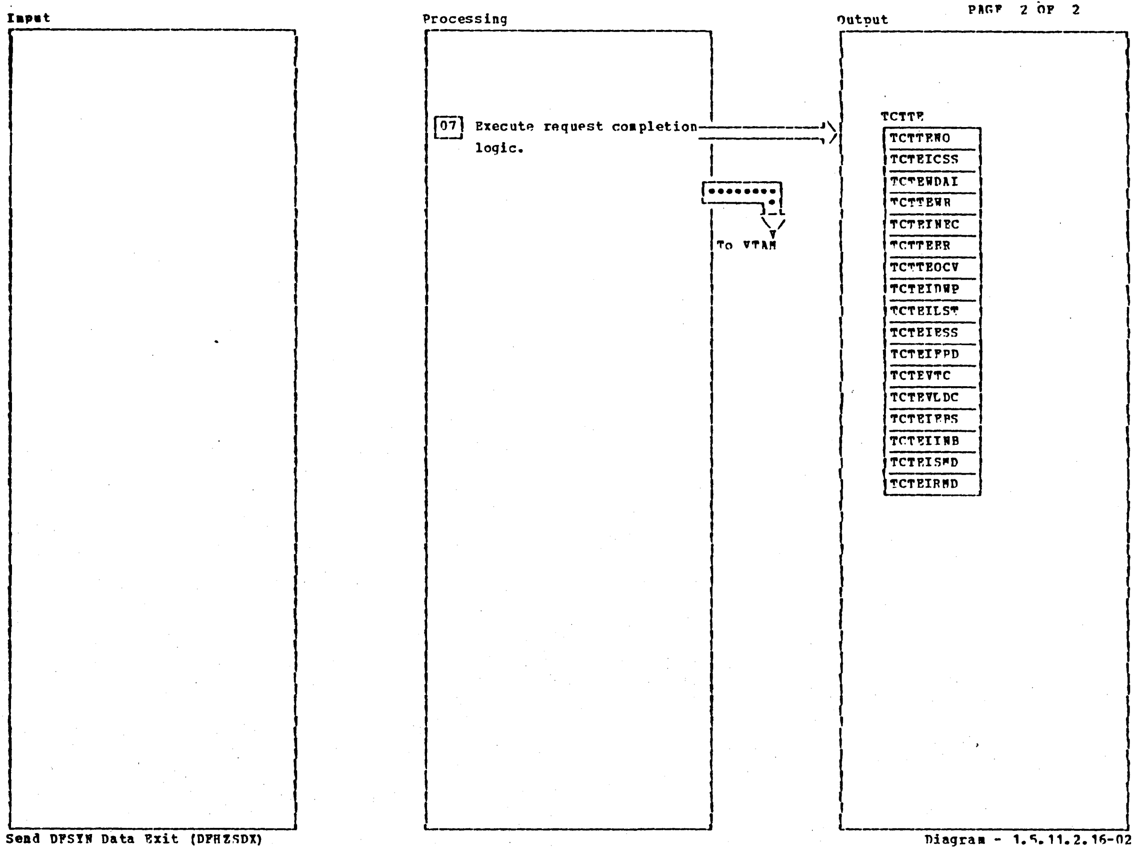
Diagram - 1.5.11.2.16-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 If definite response was requested, the completion logic will eventually be executed in the Response Exit (DFHZRPX). | | | | | | | |
| 06 This can occur if the TIOA associated with the DFHTC TYPE=WRITE request is longer than the maximum allowable output. In this case, DFHZSDS must issue several SEND requests. | | | | | | | |

Send DFSYN Data Exit (DFHZSDX)

Diagram - 1.5.11.2.16-01

PAGE 2 OF 2



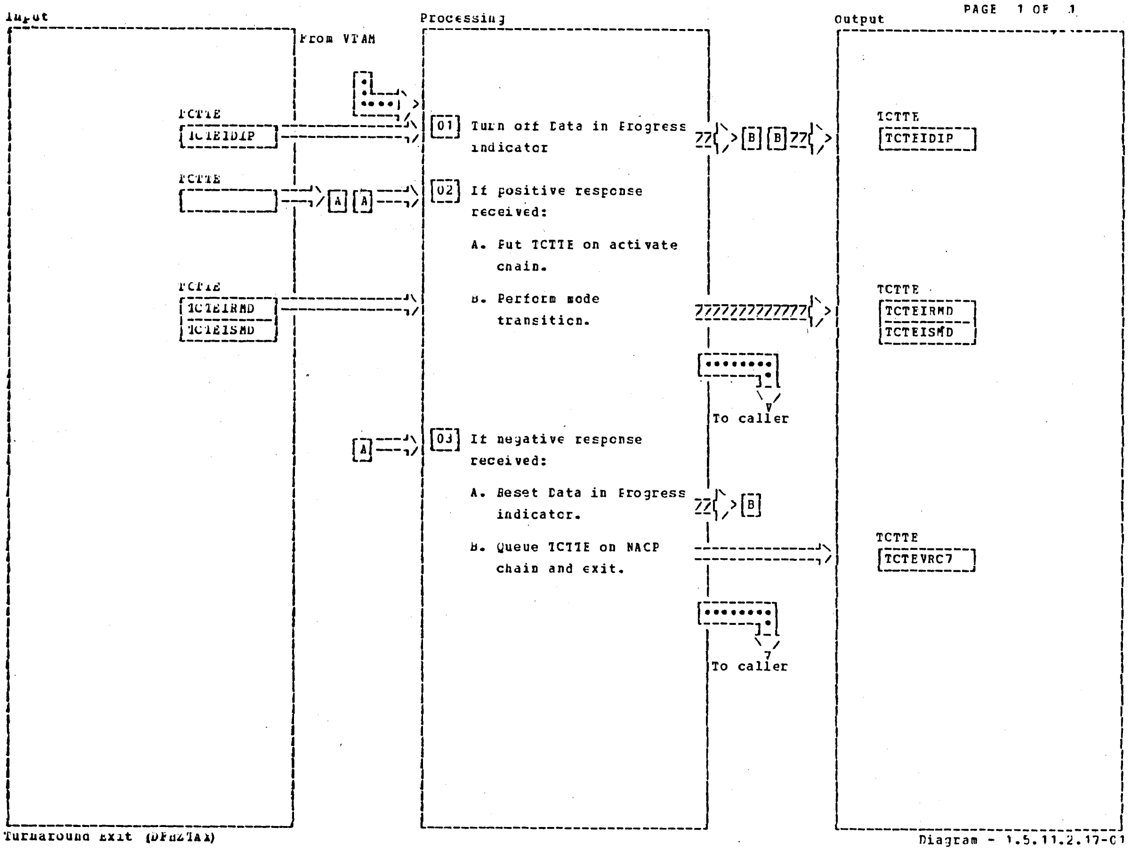
Send DFSYN Data Exit (DFHZSDX)

Diagram - 1.5.11.2.16-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 07 Certain indicators are reset and fields are cleared. Transition of Send mode and Receive mode is performed if End Bracket was sent in Batch Protocol. A FREEMAIN request for the TIOA is set up if SAVE was not specified in the DFHTC request. | | | | | | | |

Send DFSYN Data Exit (DFHZSDX)

Diagram - 1.5.11.2.16-02



01 Turn off Data in Progress indicator

02 If positive response received:

A. Put TCTTE on activate chain.

B. Perform mode transition.

03 If negative response received:

A. Reset Data in Progress indicator.

B. Queue TCTTE on NACP chain and exit.

Turnaround Exit (DFHZTAX)

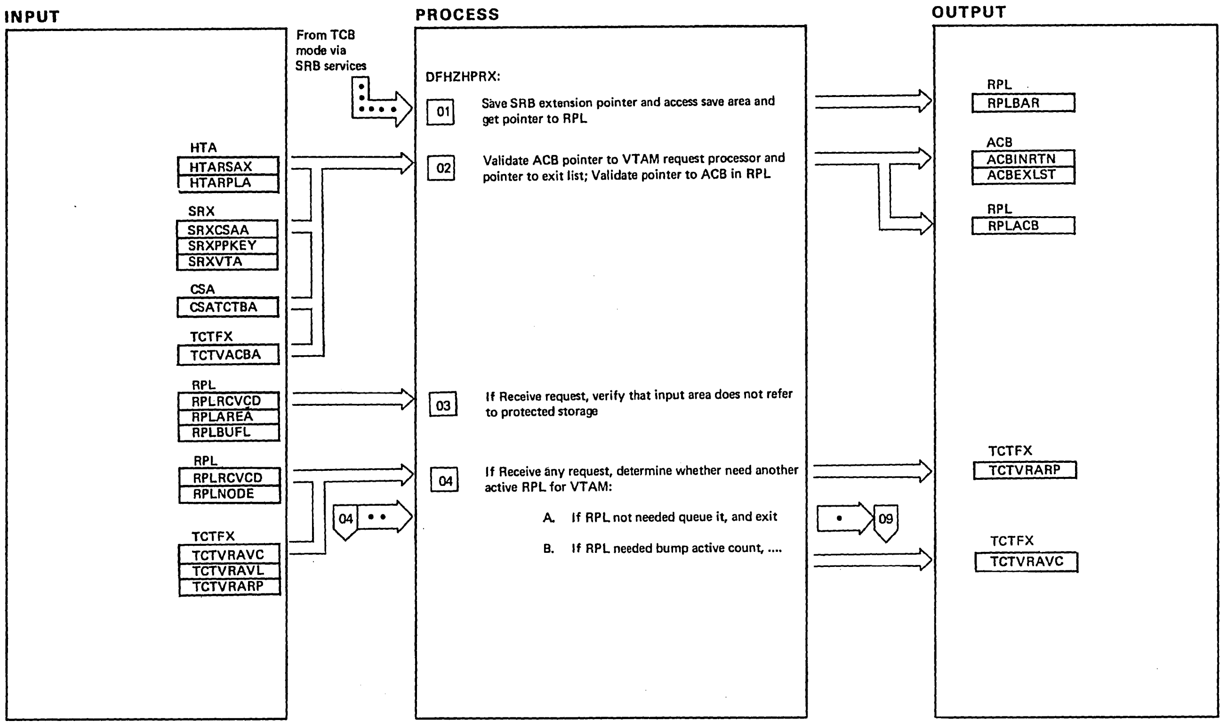
Diagram - 1.5.11.2.17-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Turnaround Exit (DFHZTAX)

Diagram - 1.5.11.2.17-01

Page 1 of 3



DFHZHPRX:

01 Save SRB extension pointer and access save area and get pointer to RPL

02 Validate ACB pointer to VTAM request processor and pointer to exit list; Validate pointer to ACB in RPL

03 If Receive request, verify that input area does not refer to protected storage

04 If Receive any request, determine whether need another active RPL for VTAM:

A. If RPL not needed queue it, and exit

B. If RPL needed bump active count, ....

RPL Executor (DFHZHPRX)

Diagram - 1.5.11.2.18-01

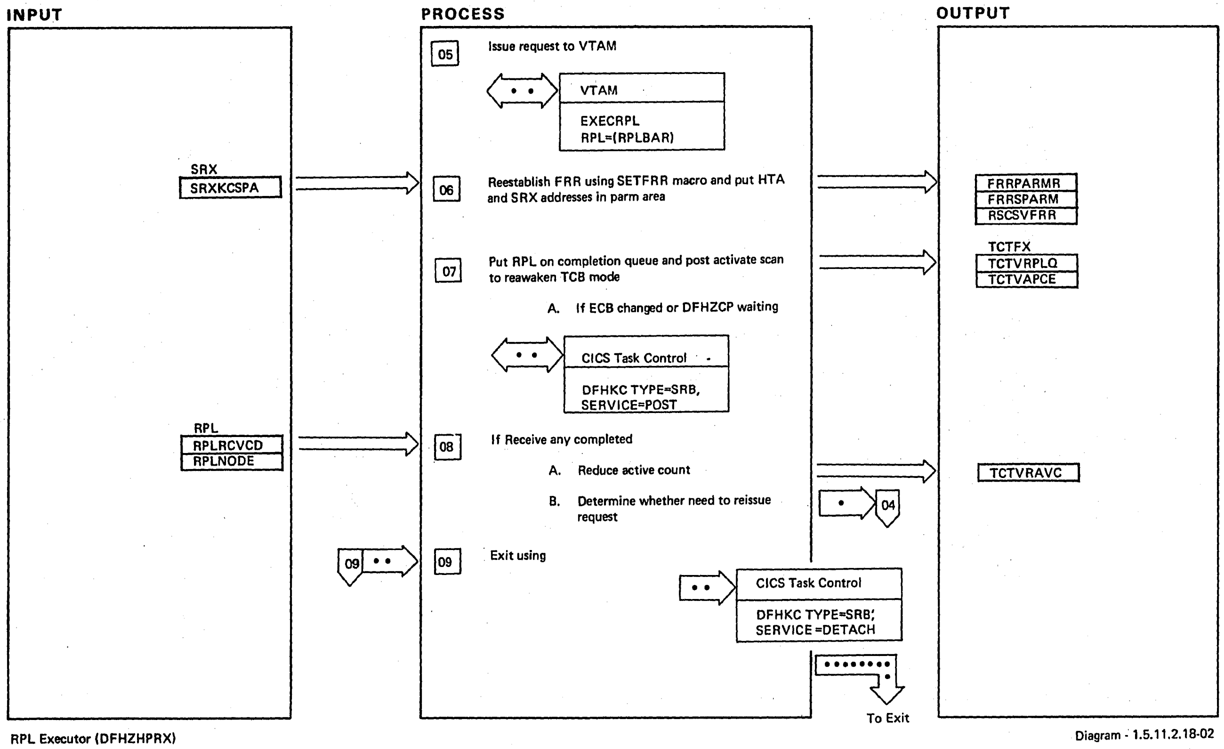
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 FRR already set up by SRB management services. Save SRB extension (SRX) pointer in a register to allow register 13 to be set to save area; save area is in protected storage, pointer set in HTA by SRB services. | | | |
| 02 Data area should have application program key. | | | |
| 03 When number of active SRB mode Receive Any requests matches threshold, RPL is queued, so that request can be reissued to VTAM, as soon as Receive any is completed, without going back into TCB mode. Receive any RPLs are also added to this queue in TCB mode | DFHZHPSR | | 1.5.11.1.37 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

RPL Executor (DFHZHPRX).

.Diagram - 1.5.11.2.18-01

Page 2 of 3



INPUT
PROCESS
OUTPUT
SRX
SRXKCSPA
RPL
RPLRCVCD
RPLNODE
09
05 Issue request to VTAM
VTAM
EXECRPL
RPL=(RPLBAR)
06 Reestablish FRR using SETFRR macro and put HTA and SRX addresses in parm area
07 Put RPL on completion queue and post activate scan to reawaken TCB mode
A. If ECB changed or DFHZCP waiting
CICS Task Control
DFHKC TYPE=SRB, SERVICE=POST
08 If Receive any completed
A. Reduce active count
B. Determine whether need to reissue request
04
09 Exit using
CICS Task Control
DFHKC TYPE=SRB, SERVICE=DETACH
To Exit
FRRPARMR
FRRSPARM
RSCSVFRR
TCTFX
TCTVRPLQ
TCTVAPCE
TCTVRAVC

RPL Executor (DFHZHPRX)

Diagram - 1.5.11.2.18-02

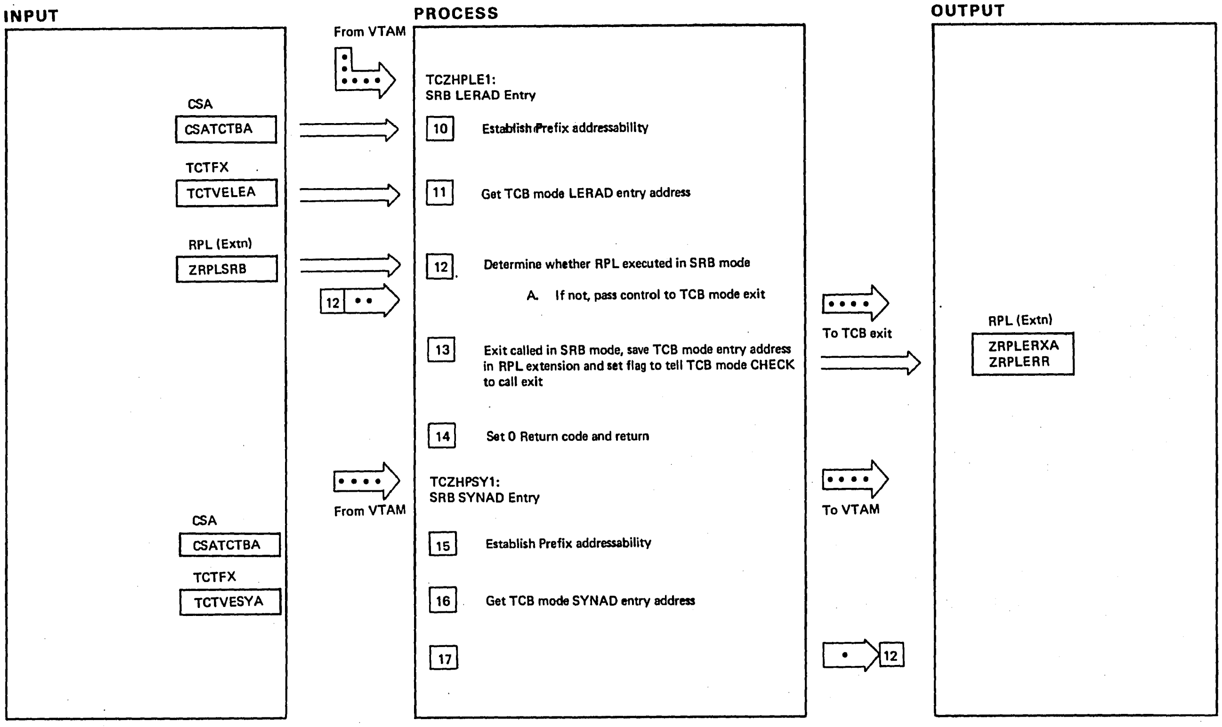
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 05 Type of request was set in RPLREQ in TCB mode. SRB will be suspended by VTAM, and 'reawakened' when the request completes. | DFHZRAC<br>DFHZRVS<br>DFHZSDS | | 1.5.11.1.4<br>1.5.11.1.14<br>1.5.11.1.17 |
| 06 FRR address in SRB management services is passed | | | |
| 07 If a full POST is needed, the SRB management services will use Branch entry to POST and Get/Free the local lock | | | |
| 08 By trying to reissue Receive any requests in SRB mode, the number of SRB to TCB mode switches is reduced. If no RPL is available, the SRB is allowed to terminate | | | |
| 09 Terminates SRB, unless the SRB management services find more work for it, e.g. another HTA request | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

RPL Executor (DFHZHPRX)

Diagram - 1.5.11.2.18-02

Page 3 of 3


INPUT
PROCESS
OUTPUT
From VTAM
TCZHPLE1:
SRB LERAD Entry
CSA
CSATCTBA
10
Establish Prefix addressability
TCTFX
TCTVELEA
11
Get TCB mode LERAD entry address
RPL (Extn)
ZRPLSRB
12
Determine whether RPL executed in SRB mode
A. If not, pass control to TCB mode exit
12
To TCB exit
13
Exit called in SRB mode, save TCB mode entry address in RPL extension and set flag to tell TCB mode CHECK to call exit
RPL (Extn)
ZRPLERXA
ZRPLERR
14
Set 0 Return code and return
From VTAM
TCZHPSY1:
SRB SYNAD Entry
To VTAM
CSA
CSATCTBA
15
Establish Prefix addressability
TCTFX
TCTVESYA
16
Get TCB mode SYNAD entry address
17
12

RPL Executor (DFHZHPRX)

Diagram - 1.5.11.2.18-03

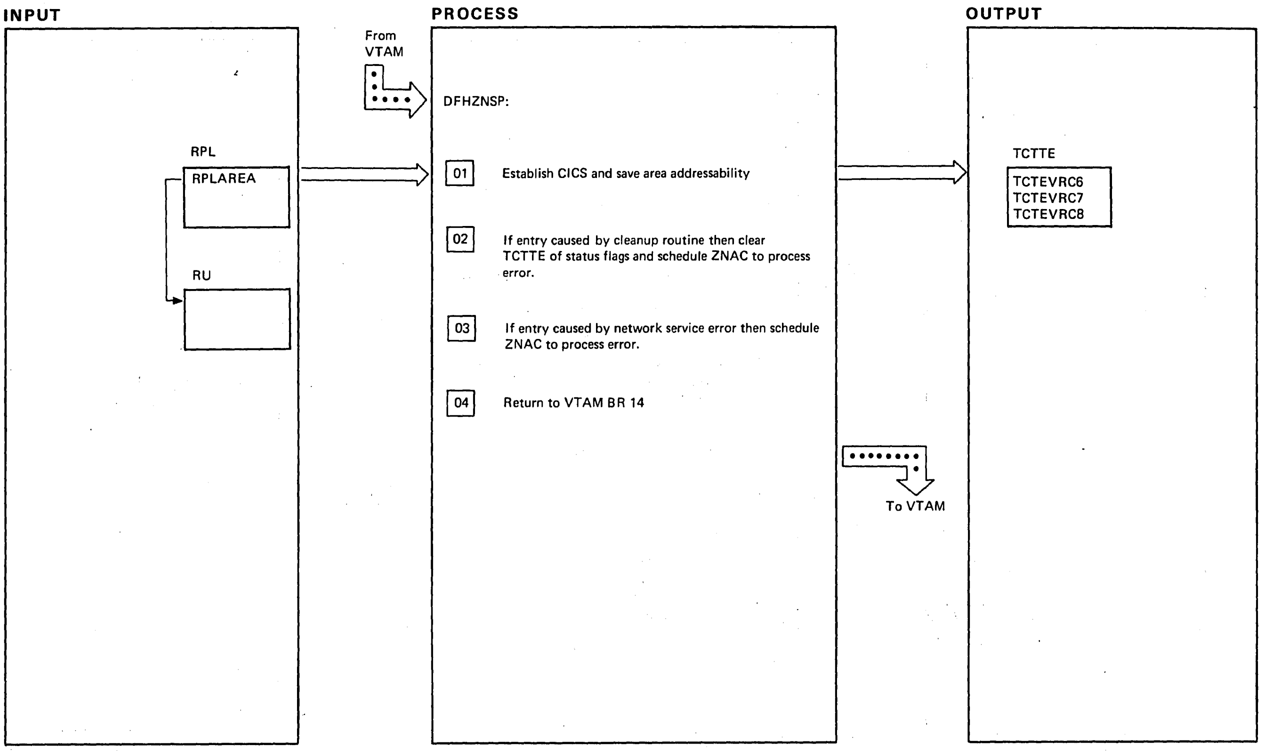
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

RPL Executor (DFHZHPRX)

Diagram - 1.5.11.2.18-03

Page 1 of 1



DFHZNSP:

01 Establish CICS and save area addressability

02 If entry caused by cleanup routine then clear TCTTE of status flags and schedule ZNAC to process error.

03 If entry caused by network service error then schedule ZNAC to process error.

04 Return to VTAM BR 14

Network Services Exit (DFHZNSP)

Diagram – 1.5.11.2.19-01

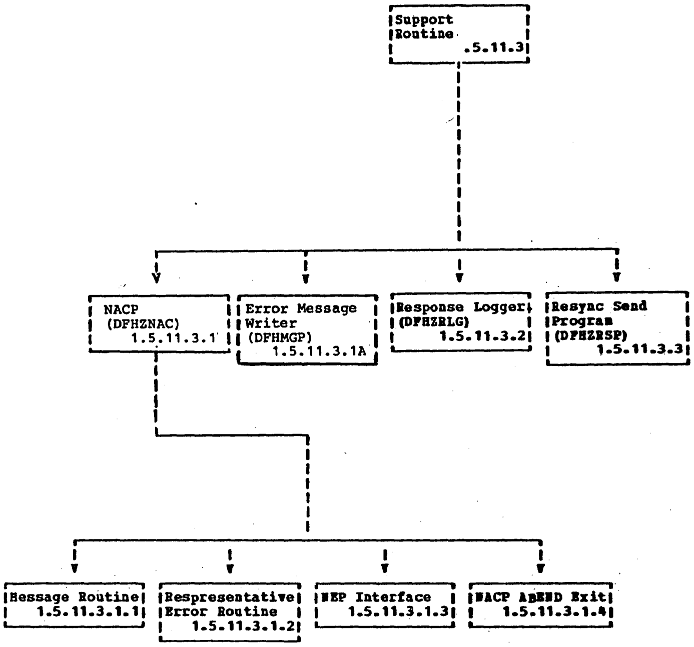
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Network Services Exit (DFHZNSP)

Diagram – 1.5.11.2.19-01

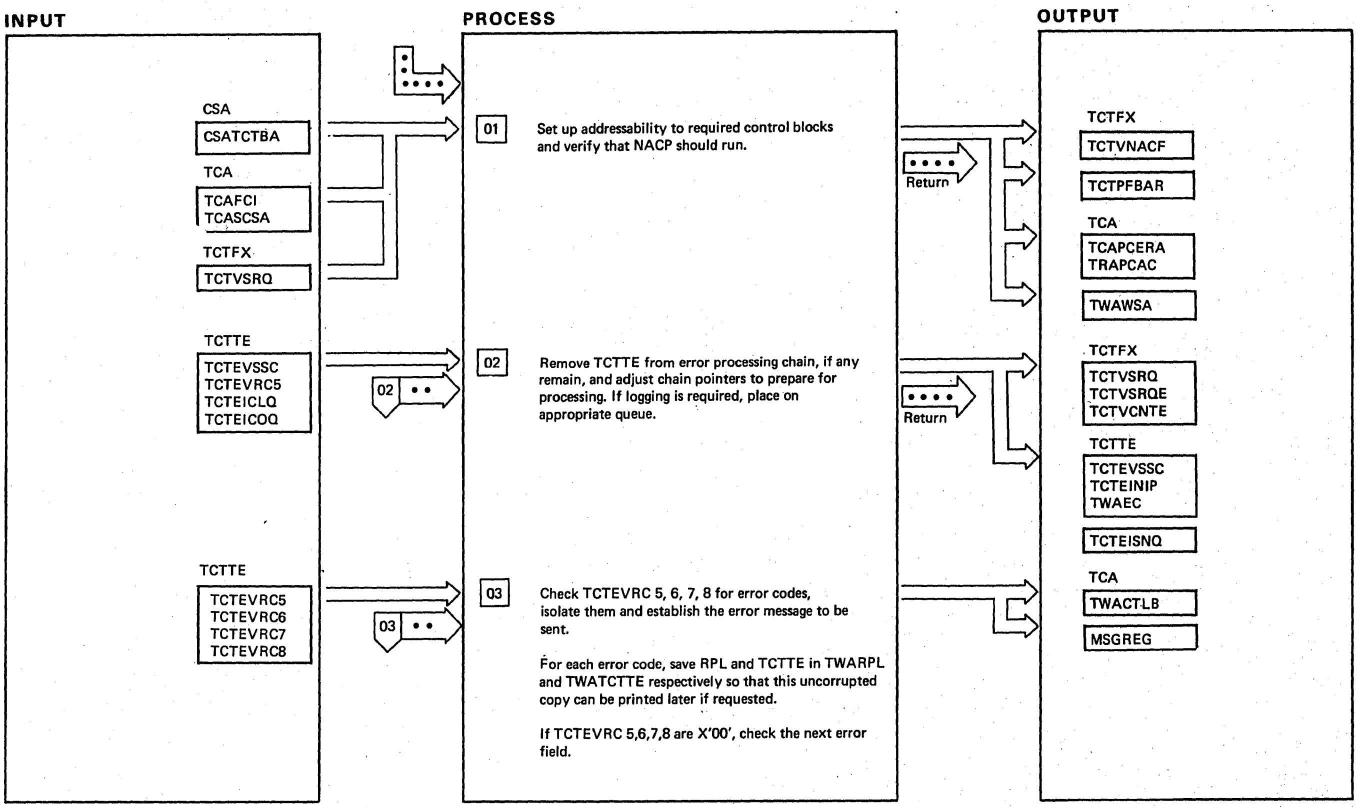
PAGE 1 OF 1


Support Routine .5.11.3
NACP (DFHZNAC) 1.5.11.3.1
Error Message Writer (DFHMGP) 1.5.11.3.1A
Response Logger (DFHZRLG) 1.5.11.3.2
Resync Send Program (DFHZRSP) 1.5.11.3.3
Message Routine 1.5.11.3.1.1
Respresentative Error Routine 1.5.11.3.1.2
NEP Interface 1.5.11.3.1.3
NACP ABEND Exit 1.5.11.3.1.4

Visual Table of Contents

Diagram — 1.5.11.3-01

Page 1 of 3



01 Set up addressability to required control blocks and verify that NACP should run.

02 Remove TCTTE from error processing chain, if any remain, and adjust chain pointers to prepare for processing. If logging is required, place on appropriate queue.

03 Check TCTEVRC 5, 6, 7, 8 for error codes, isolate them and establish the error message to be sent.

For each error code, save RPL and TCTTE in TWARPL and TWATCTTE respectively so that this uncorrupted copy can be printed later if requested.

If TCTEVRC 5,6,7,8 are X'00', check the next error field.

Node Abnormal Condition Program (DFHZNAC)

Diagram — 1.5.11.3.1-01

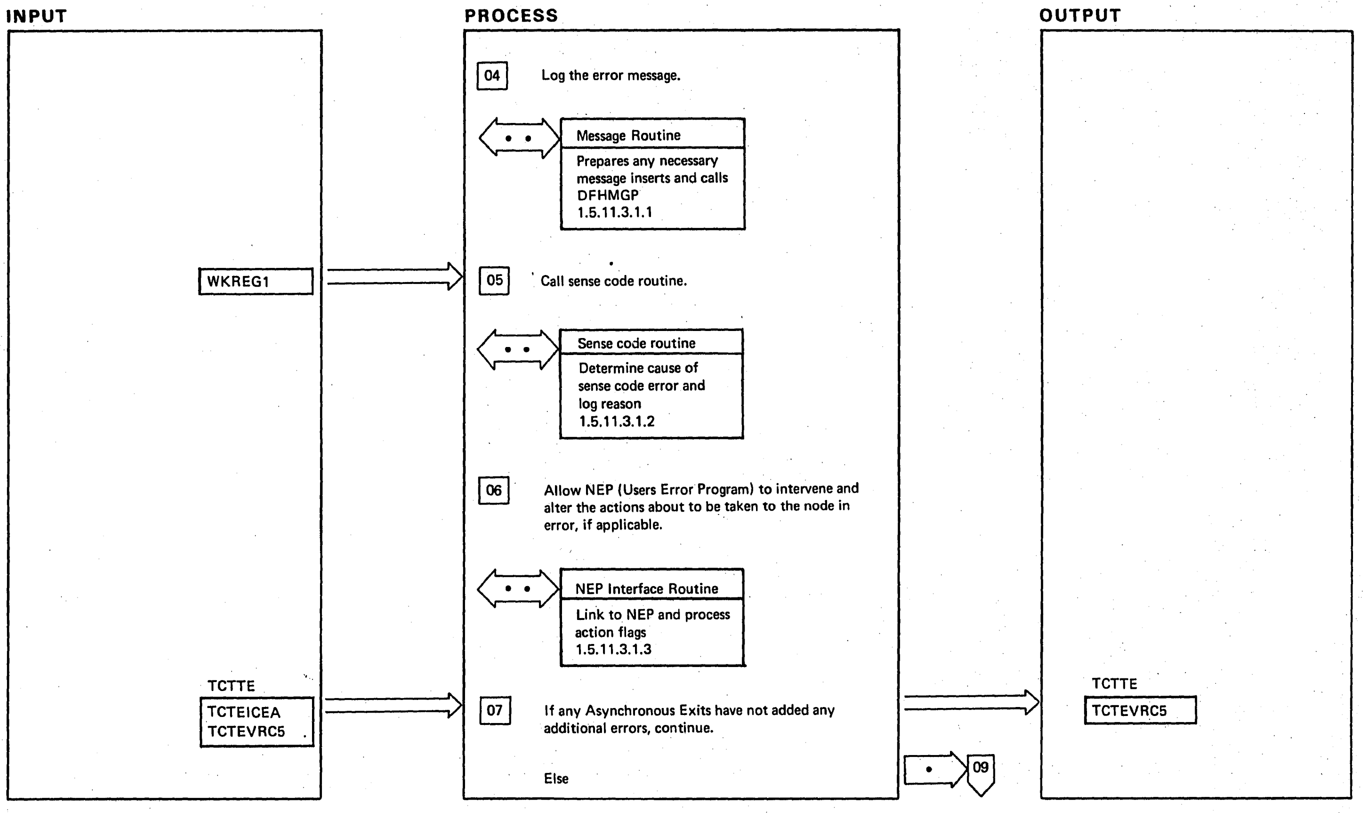
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Addressability to the TCT prefix is required:<br>Since NACP is a transaction which is attached, it must be verified that it was not initiated from a node and, if so, NACP will ABEND. (DMPCODE = ATNA)<br>NACP must also acquire working storage in which messages are built, control blocks formatted and information temporarily saved.<br>The NACP ABEND Exit is armed at this time. | | | |
| 02 A pointer in the TCT prefix contains the address of the TCTTEs in error. Each time NACP removes an entry the pointers must be changed. When the chain is empty X'FF's replace the chain pointers.<br>Before processing the TCTTE, the TWA must be cleared of any residual indicators. | | | |
| 03 If TCTEVRC5 is X'00', indicating no error code, check the next error field.<br>The error code is used to access an index into a table of message entries for each error. The message entry contains the number of the message to be written, the actions to be taken for the error, and an offset to the error routines to be taken. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Node Abnormal Condition Program (DFHZNAC)

Diagram -- 1.5.11.3.1-01

Page 2 of 3



04 Log the error message.

Message Routine
Prepares any necessary message inserts and calls DFHMGP
1.5.11.3.1.1

05 Call sense code routine.

Sense code routine
Determine cause of sense code error and log reason
1.5.11.3.1.2

06 Allow NEP (Users Error Program) to intervene and alter the actions about to be taken to the node in error, if applicable.

NEP Interface Routine
Link to NEP and process action flags
1.5.11.3.1.3

07 If any Asynchronous Exits have not added any additional errors, continue.

Else

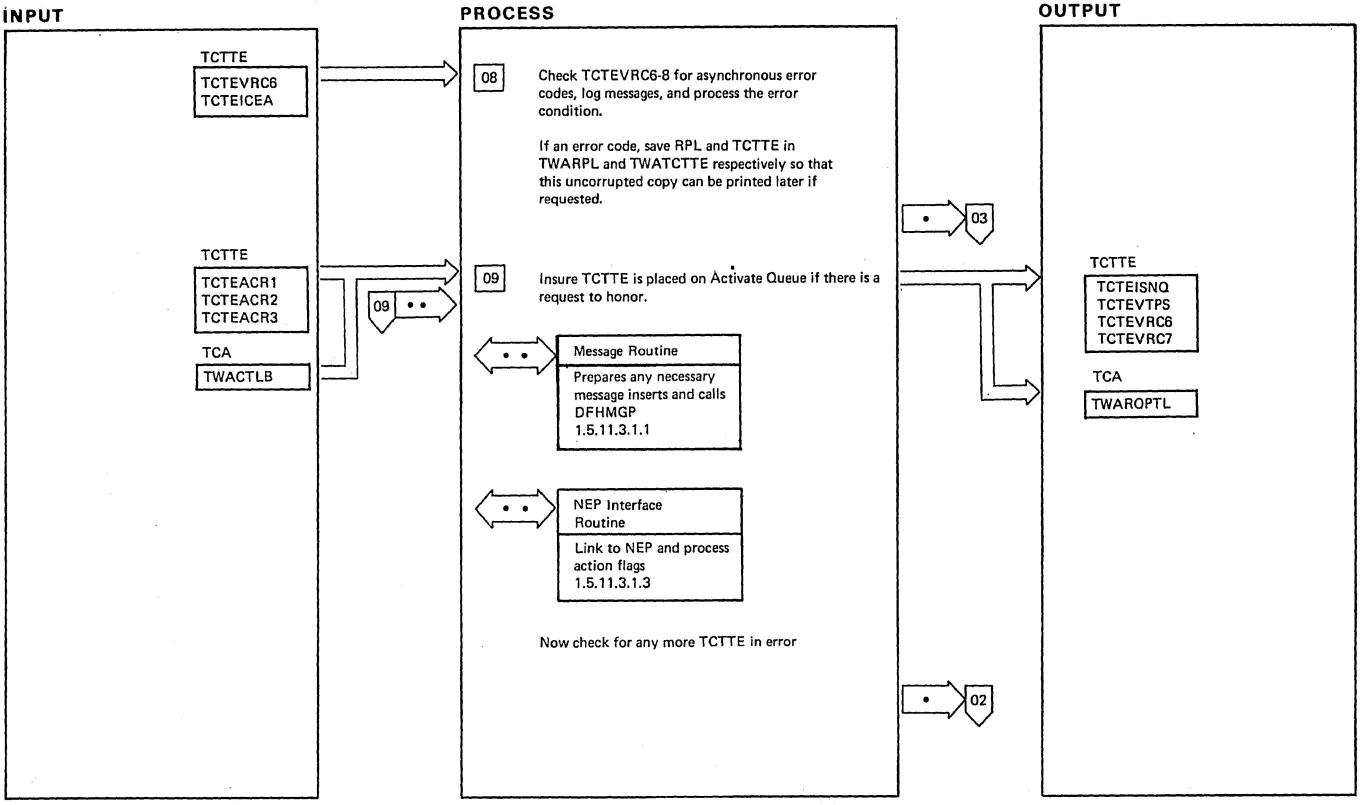
Node Abnormal Condition Program (DFHZNAC)

Diagram — 1.5.11.3.1-02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 04 A BALR is made to the message routine which calls DFHMGP to log the message. | | | |
| 05 If the message entry indicates that sense code analysis is required for this error condition, call the sense code routine | | | |
| 06 BALR to the NEP interface routine. Note that the NACP does not call the NEP for all conditions. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Page 3 of 3



08 Check TCTEVRC6-8 for asynchronous error codes, log messages, and process the error condition.

If an error code, save RPL and TCTTE in TWARPL and TWATCTTE respectively so that this uncorrupted copy can be printed later if requested.

09 Insure TCTTE is placed on Activate Queue if there is a request to honor.

Message Routine

Prepares any necessary message inserts and calls DFHMGP 1.5.11.3.1.1

NEP Interface Routine

Link to NEP and process action flags 1.5.11.3.1.3

Now check for any more TCTTE in error

Node Abnormal Condition Program (DFHZNAC)

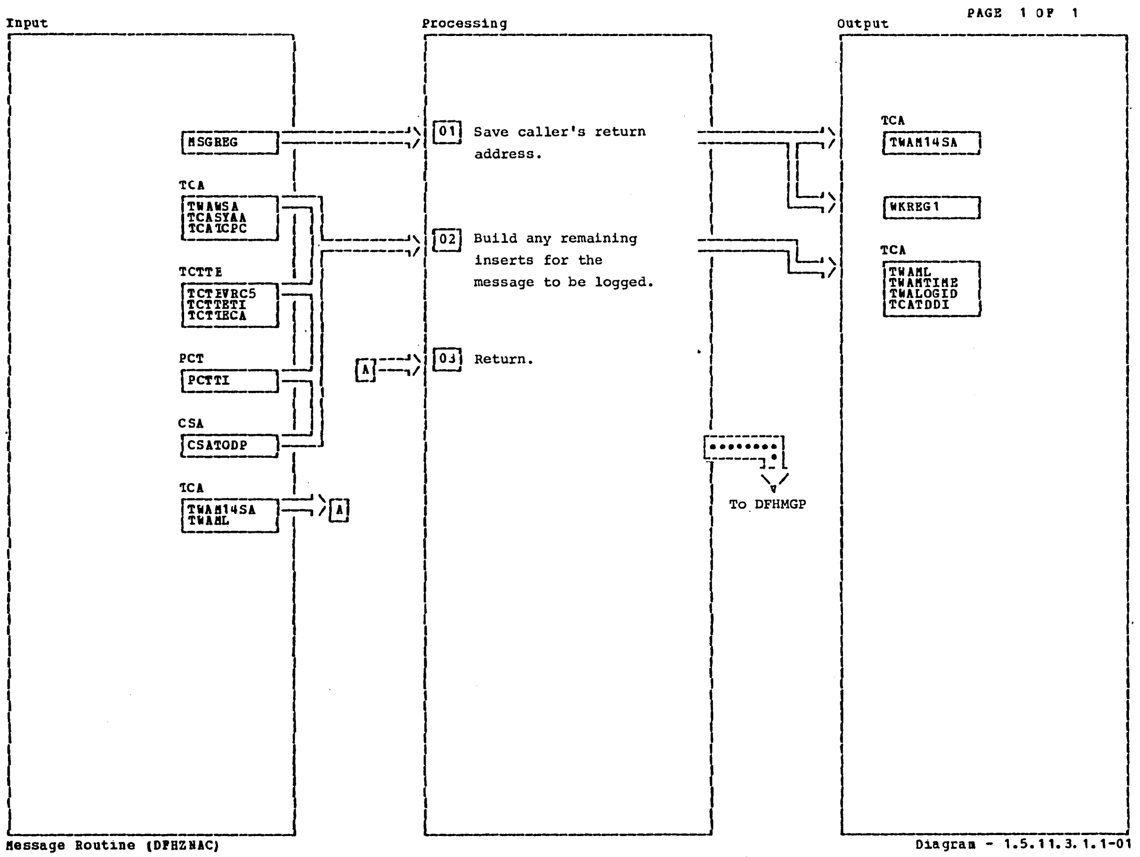
Diagram — 1.5.11.3.1-03

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 09 If there is a request outstanding, the TCTTE will be placed on the Activate Queue.<br>At this point the error queue is scanned again. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

de Abnormal Condition Program (DFHZNAC)

Diagram — 1.5.11.3.1-03



01 Save caller's return address.

02 Build any remaining inserts for the message to be logged.

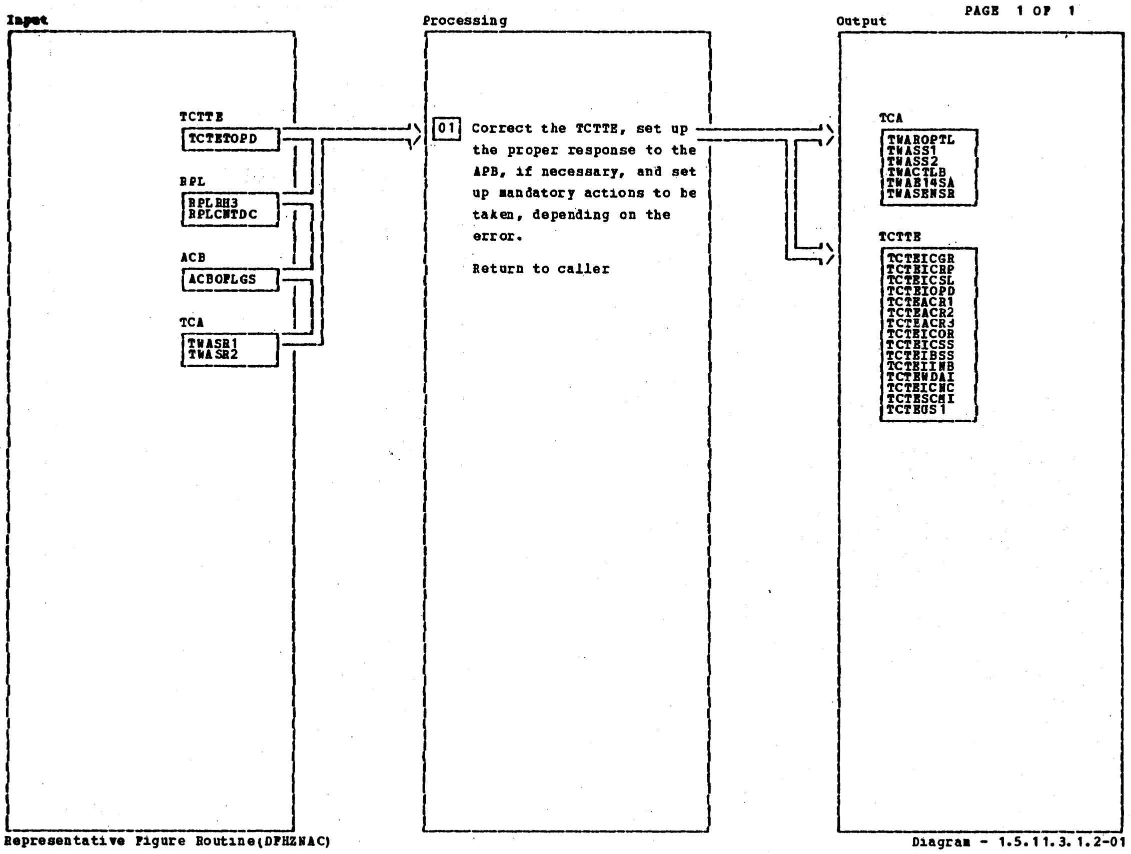
03 Return.

Message Routine (DFHZNAC) Diagram - 1.5.11.3.1.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Insure integrity of caller's return address. | | | | | | | |
| 02 Each message put to the message logs contain time, terminal id, task id, and the message OK control block (i.e., RPL, TCTTE). This message format can be altered simply by changing the message prototype. | | | | | | | |
| 03 After the entire message has been logged, the caller's return address is restored and returned to. | | | | | | | |

Message Routine (DFHZNAC) Diagram - 1.5.11.3.1.1-01

PAGE 1 OF 1



01 Correct the TCTTE, set up the proper response to the APB, if necessary, and set up mandatory actions to be taken, depending on the error.

Return to caller

Representative Figure Routine(DFHZNAC)

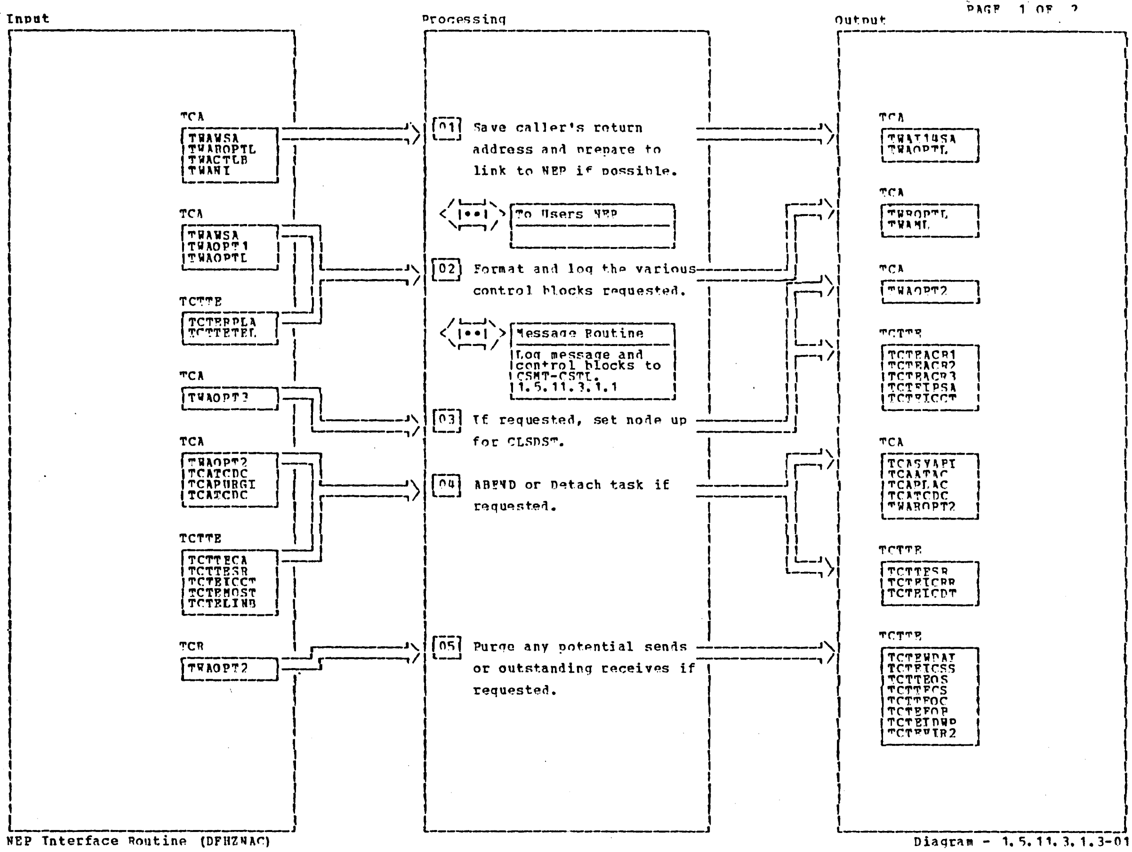
Diagram - 1.5.11.3.1.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 This hipo represents more than fifty individual unique error routines in NACP.<br>Many bits are examined in the TCTTE, as well as other control blocks, to isolate the error and correct it. | | | | | | | |

Representative Figure Routine(DFHZNAC)

Diagram - 1.5.11.3.1.2-01

PAGE 1 OF 2



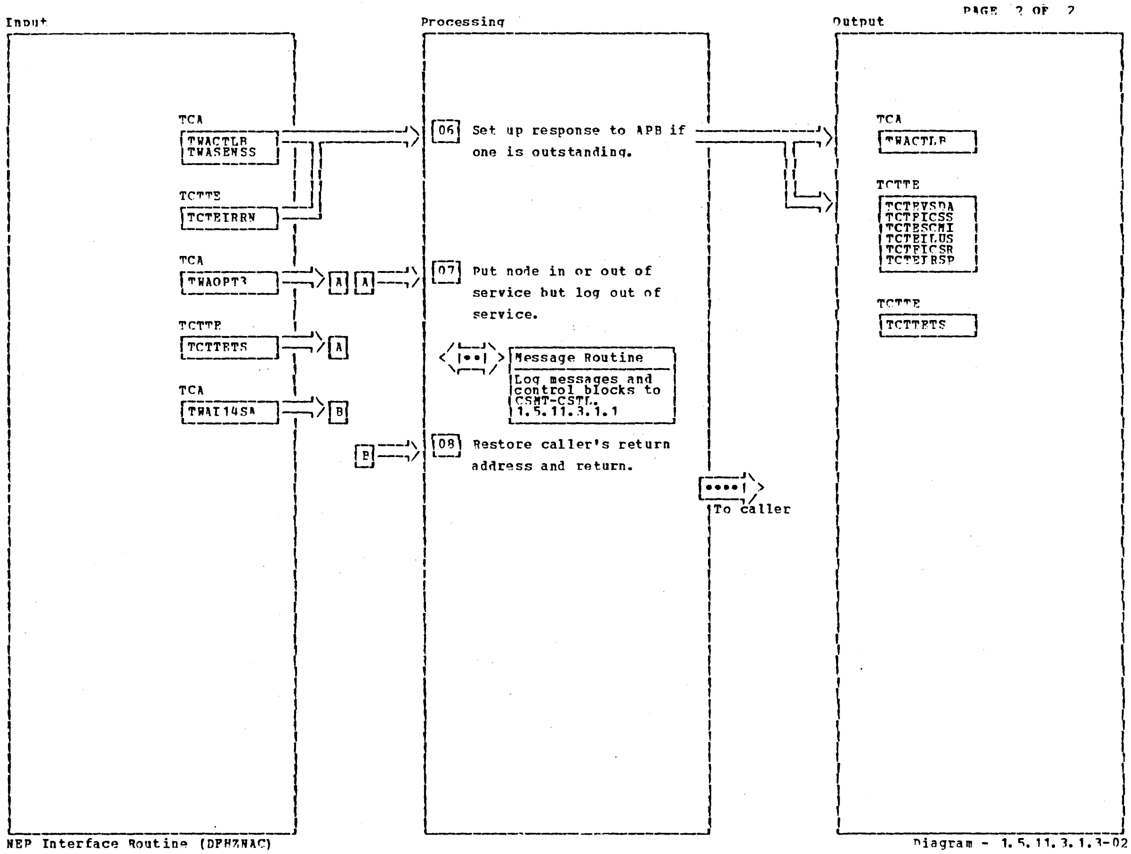
NEP Interface Routine (DFHZNAC)

Diagram - 1.5.11.3.1.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Insure integrity of caller's return address by placing it in the TWA RSA.<br>If NEP has been disabled or has been abnormally terminated previously, do not link to it.<br>If linking to NEP activates the NEP ABEND, exit for the safety of NACP. | | | |
| 02 If the error routine or NEP indicate the TCTTE, PPT, or action flags should be printed, this is where that is done.<br>A BALR is made to the message routine that actually logs them.<br>If the unavailable-printer condition is detected, and the NEP provides a printer address, the data is put to the printer. | | | |
| 03 If the error routine or NEP indicates the error was of the magnitude to warrant a CLSDST, this routine insures that it is set up. | | | |
| 04 If the error routine or NEP indicates that it is necessary to ABEND or detach the task, this routine performs that function.<br>When CLSDST has been effected, the task is abnormally terminated. | | | |
| 05 If the error routine or NEP indicates that it is necessary to purge Sends and/or Receives, that function is performed by this routine.<br>When a task is abnormally terminated, purging of Sends and Receives is automatic. | | | |

NEP Interface Routine (DFHZNAC)

Diagram - 1.5.11.3.1.3-01



NEP Interface Routine (DFHZNAC)

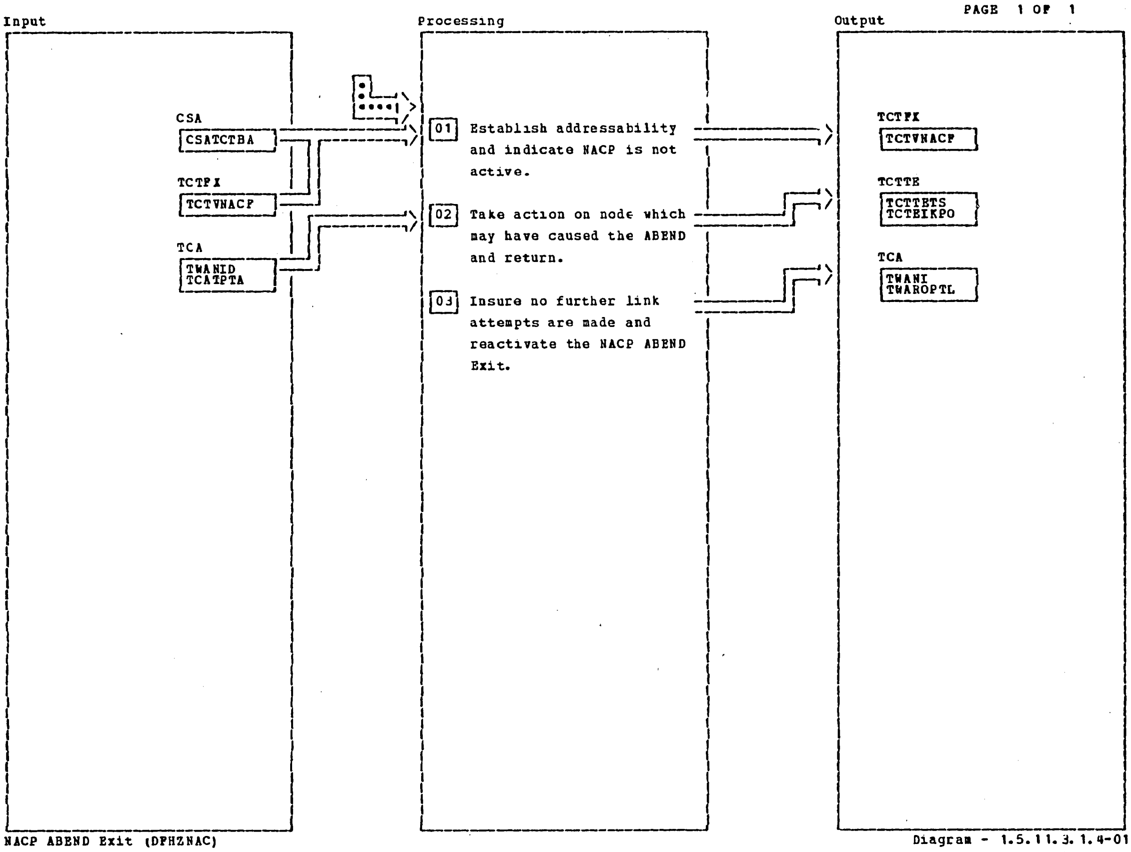
Diagram - 1.5.11.3.1.3-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 06 If N response is outstanding to the APB or if the error routine decides a response is required, this routine performs that function. No response is sent to a VTAM 3270 device, however. | | | | | | | |
| 07 When a node is placed out of service, either because of NEP or the error routine, an out-of-service message is sent to the Master Terminal Log. | | | | | | | |

NEP Interface Routine (DFHZNAC)

Diagram - 1.5.11.3.1.3-02

PAGE 1 OF 1



01 Establish addressability and indicate NACP is not active.

02 Take action on node which may have caused the ABEND and return.

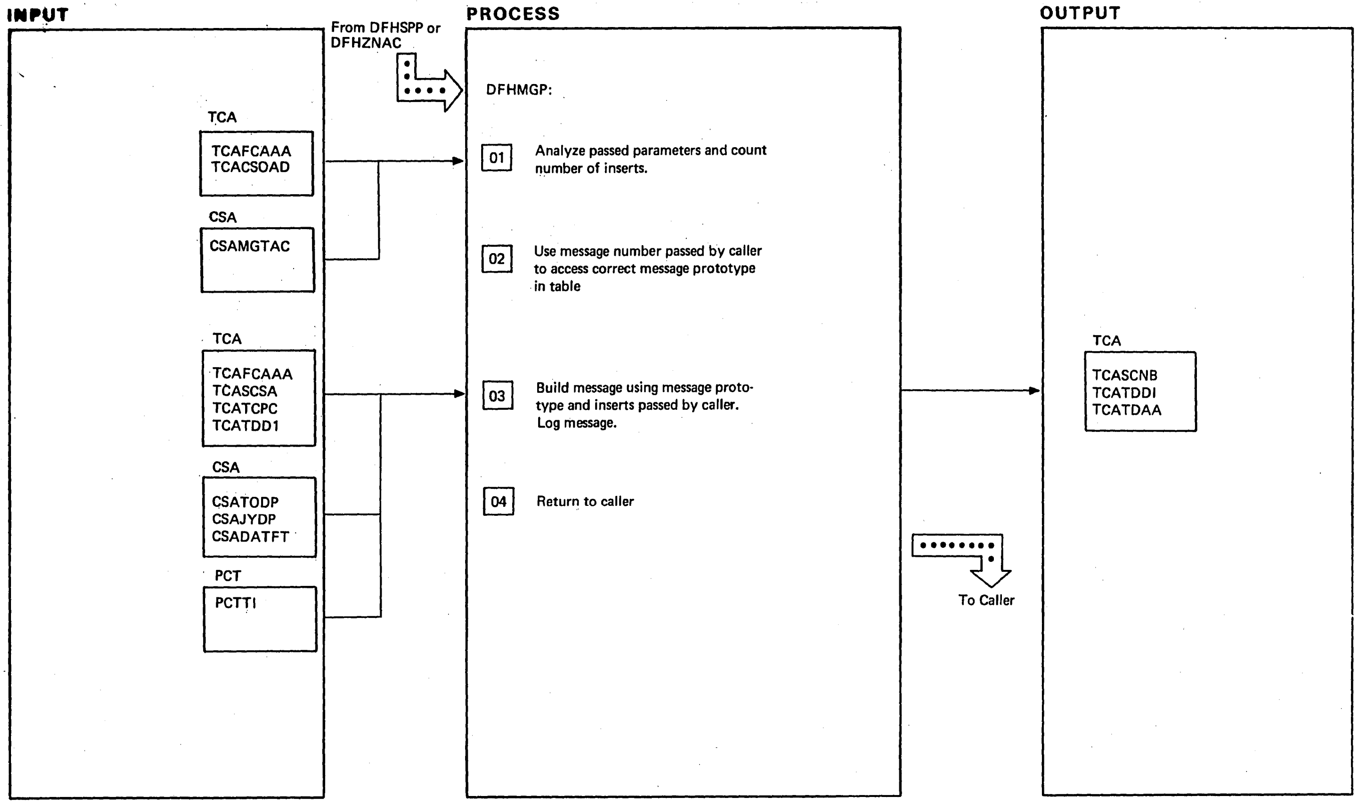
03 Insure no further link attempts are made and reactivate the NACP ABEND Exit.

NACP ABEND Exit (DFHZNAC) Diagram - 1.5.11.3.1.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Addressability to various areas must be confirmed.<br>The 'NACP Is Processing' flag must be turned off to enable NACP to be reattached and continue processing. | | | | | | | |
| 02 Assuming the cause of NACP ABENDing was invalid or incorrect data in the TCTTE, that TCTTE must again be located and put permanently out of service to insure it does not cause another ABEND. | | | | | | | |
| 03 If an NEP ABEND occurs or if NEP is disabled, NACP continues processing without attempting future links to NEP. | | | | | | | |

NACP ABEND Exit (DFHZNAC) Diagram - 1.5.11.3.1.4-01

Page 1 of 1



DFHMGP:

01 Analyze passed parameters and count number of inserts.

02 Use message number passed by caller to access correct message prototype in table

03 Build message using message prototype and inserts passed by caller. Log message.

04 Return to caller

Error Message Writer (DFHMGP)

Diagram – 1.5.11.3.1A-01

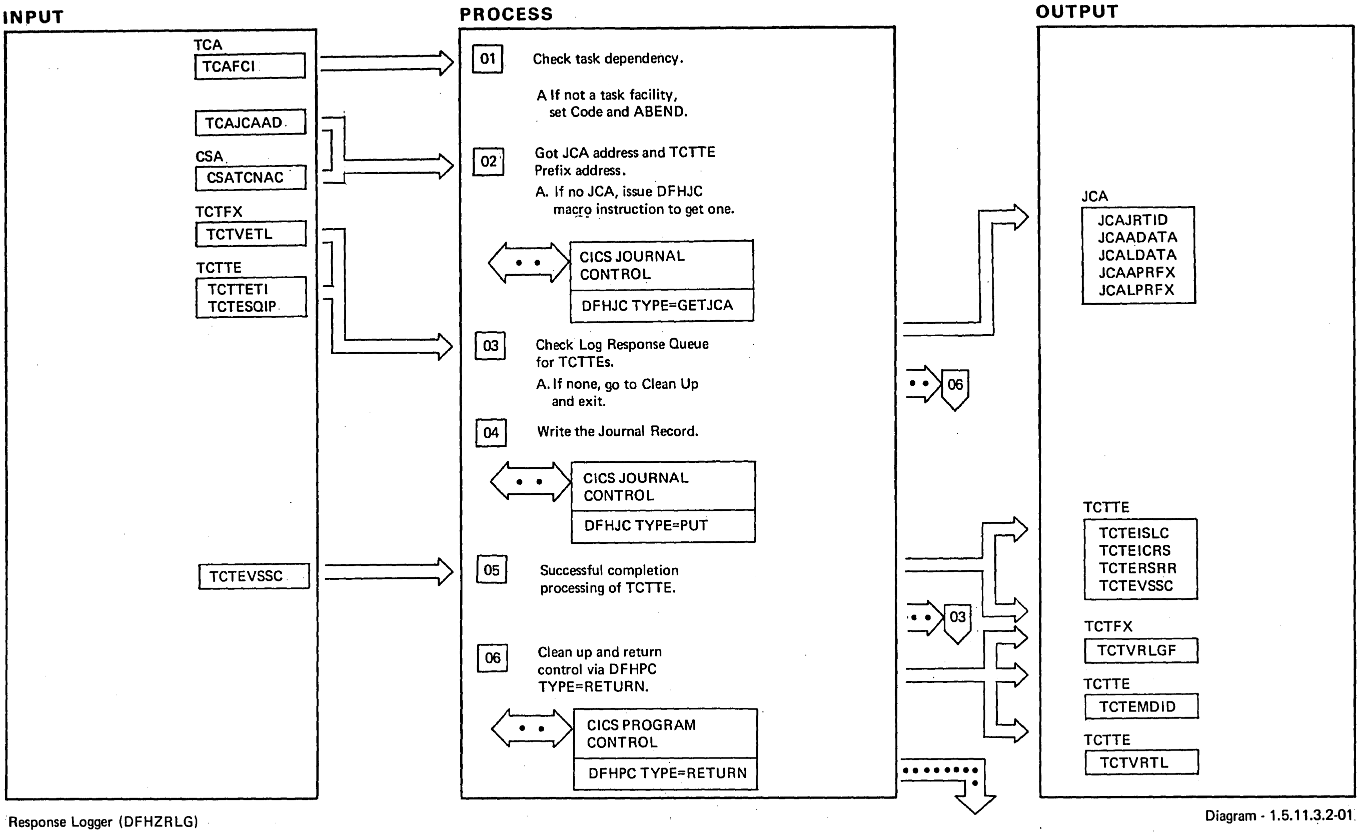
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The passed parameter list contains a series of pointers; the first points to a control record containing the message number and the destination of the message, the second and subsequent point to individual inserts for the message. | | | |
| 02 Message prototypes are held in a special module DFHMGT. A single prototype contains the constants for one message together with control bytes denoting the location and type of any inserts. Certain inserts, e.g., the time stamp, can be provided automatically by DFHMGP and do not have to be passed by the caller.<br>A simple index is created in the module DFHMGT to allow rapid location of the required prototype, using the provided message number. | | | |
| 03 Using the inserts passed, and the prototype, the full message can now be constructed and logged. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Error Message Writer (DFHMGP)

Diagram – 1.5.11.3.1A-01

PAGE 1 OF 1


INPUT
TCA
TCAFCI
TCAJCAAD
CSA
CSATCNAC
TCTFX
TCTVETL
TCTTE
TCTTETI
TCTESQIP
TCTEVSSC
PROCESS
01 Check task dependency.
A If not a task facility, set Code and ABEND.
02 Got JCA address and TCTTE Prefix address.
A. If no JCA, issue DFHJC macro instruction to get one.
CICS JOURNAL CONTROL
DFHJC TYPE=GETJCA
03 Check Log Response Queue for TCTTEs.
A. If none, go to Clean Up and exit.
06
04 Write the Journal Record.
CICS JOURNAL CONTROL
DFHJC TYPE=PUT
05 Successful completion processing of TCTTE.
03
06 Clean up and return control via DFHPC TYPE=RETURN.
CICS PROGRAM CONTROL
DFHPC TYPE=RETURN
OUTPUT
JCA
JCAJRTID
JCAADATA
JCALDATA
JCAAPRFX
JCALPRFX
TCTTE
TCTEISLC
TCTEICRS
TCTERSRR
TCTEVSSC
TCTFX
TCTVRLGF
TCTTE
TCTEMDID
TCTTE
TCTVRTL

Response Logger (DFHZRLG)

Diagram - 1.5.11.3.2-01

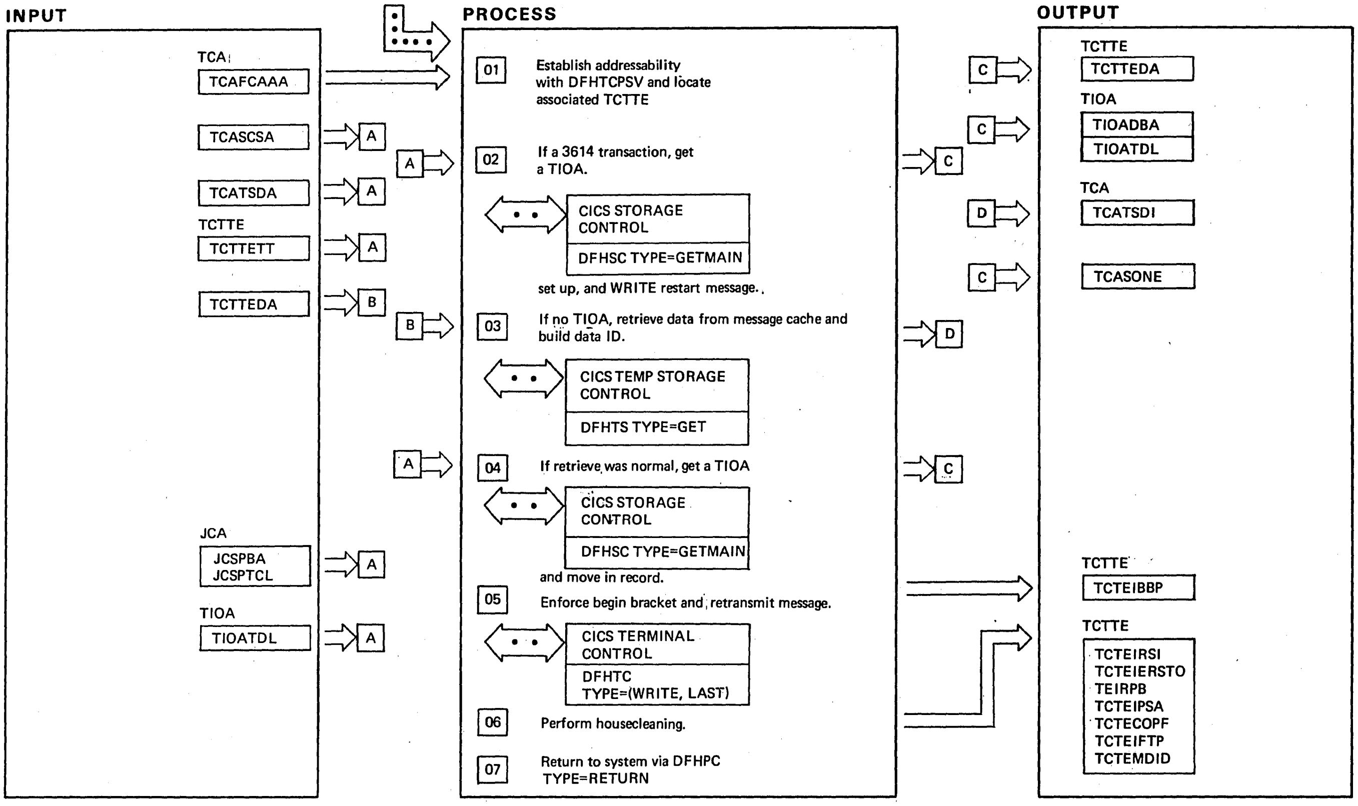
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The program may be activated only by the activate scan routine of TCP. Any other facility code causes the program to ABEND. | | TCZFLG10 | |
| 04 Invokes the Journal Control Program to write the record to the system log. | | TCZFLG30 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Response Logger DFHZRLG.

Diagram - 1.5.11.3.2-01

PAGE 1 OF 1



01 Establish addressability with DFHTCPSV and locate associated TCTTE

02 If a 3614 transaction, get a TIOA.

CICS STORAGE CONTROL
DFHSC TYPE=GETMAIN

set up, and WRITE restart message.

03 If no TIOA, retrieve data from message cache and build data ID.

CICS TEMP STORAGE CONTROL
DFHTS TYPE=GET

04 If retrieve was normal, get a TIOA

CICS STORAGE CONTROL
DFHSC TYPE=GETMAIN

and move in record.

05 Enforce begin bracket and retransmit message.

CICS TERMINAL CONTROL
DFHTC TYPE=(WRITE, LAST)

06 Perform housecleaning.

07 Return to system via DFHPC TYPE=RETURN

Resync Send Program (DFHZRSP)

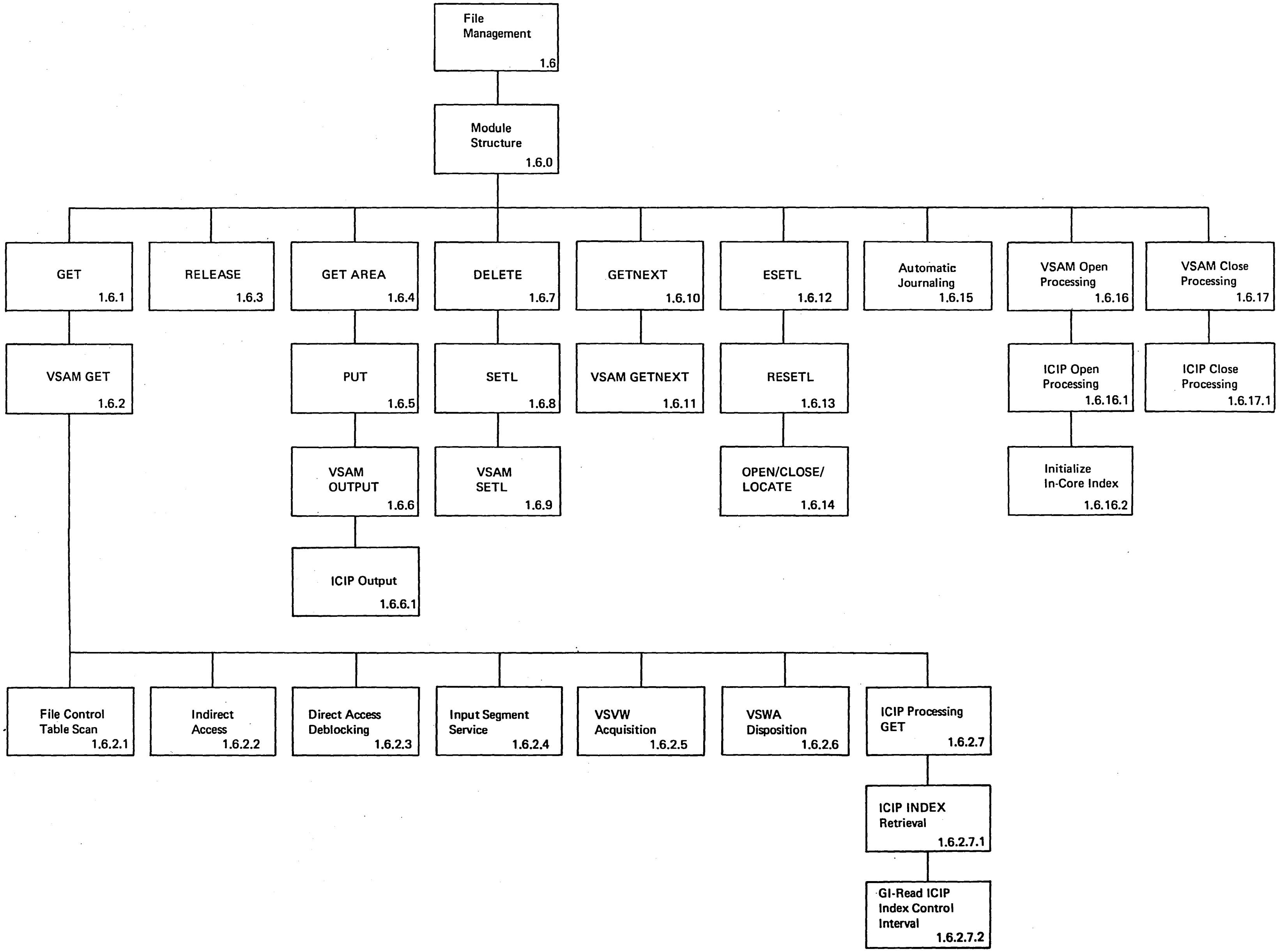
Diagram - 1.5.11.3.3-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

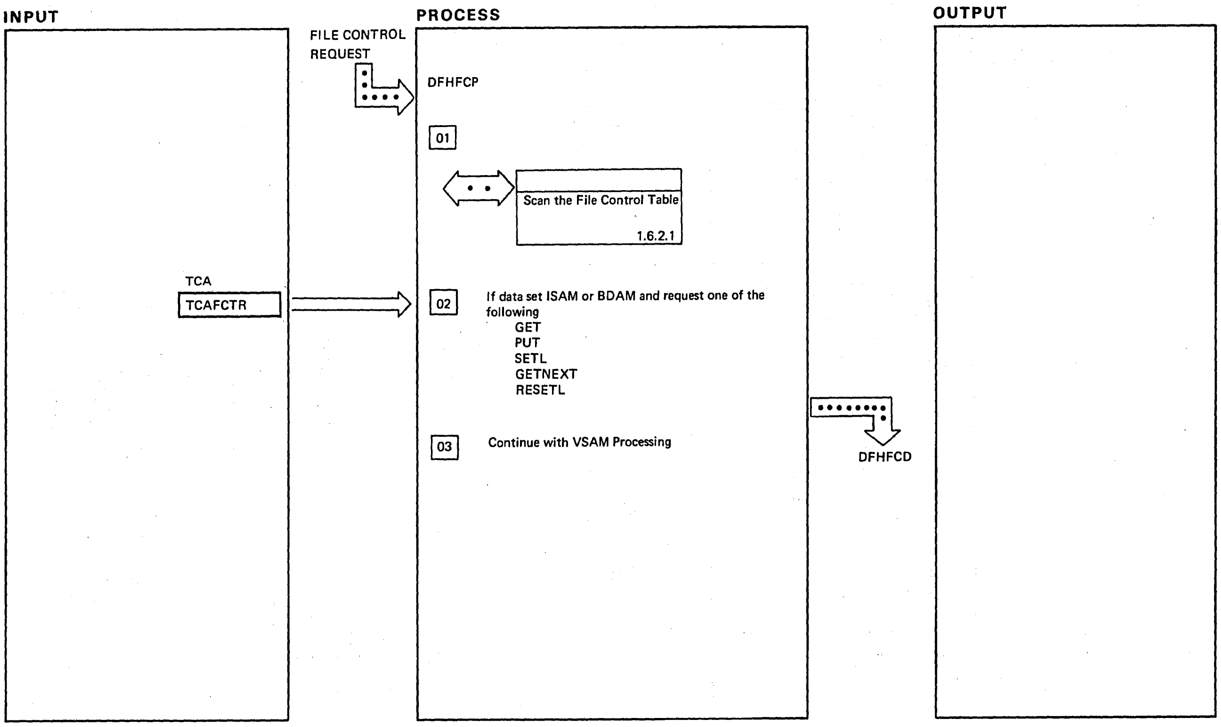
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Resync Send Program (DFHZRSP)

Diagram - 1.5.11.3.3-01


File Management 1.6
Module Structure 1.6.0
GET 1.6.1
RELEASE 1.6.3
GET AREA 1.6.4
DELETE 1.6.7
GETNEXT 1.6.10
ESETL 1.6.12
Automatic Journaling 1.6.15
VSAM Open Processing 1.6.16
VSAM Close Processing 1.6.17
VSAM GET 1.6.2
PUT 1.6.5
SETL 1.6.8
VSAM GETNEXT 1.6.11
RESETL 1.6.13
ICIP Open Processing 1.6.16.1
ICIP Close Processing 1.6.17.1
VSAM OUTPUT 1.6.6
VSAM SETL 1.6.9
OPEN/CLOSE/ LOCATE 1.6.14
Initialize In-Core Index 1.6.16.2
ICIP Output 1.6.6.1
File Control Table Scan 1.6.2.1
Indirect Access 1.6.2.2
Direct Access Deblocking 1.6.2.3
Input Segment Service 1.6.2.4
VSVW Acquisition 1.6.2.5
VSWA Disposition 1.6.2.6
ICIP Processing GET 1.6.2.7
ICIP INDEX Retrieval 1.6.2.7.1
GI-Read ICIP Index Control Interval 1.6.2.7.2

Page 1 of 1



01 Scan the File Control Table 1.6.2.1

02 If data set ISAM or BDAM and request one of the following
GET
PUT
SETL
GETNEXT
RESETL

03 Continue with VSAM Processing

Module Structure

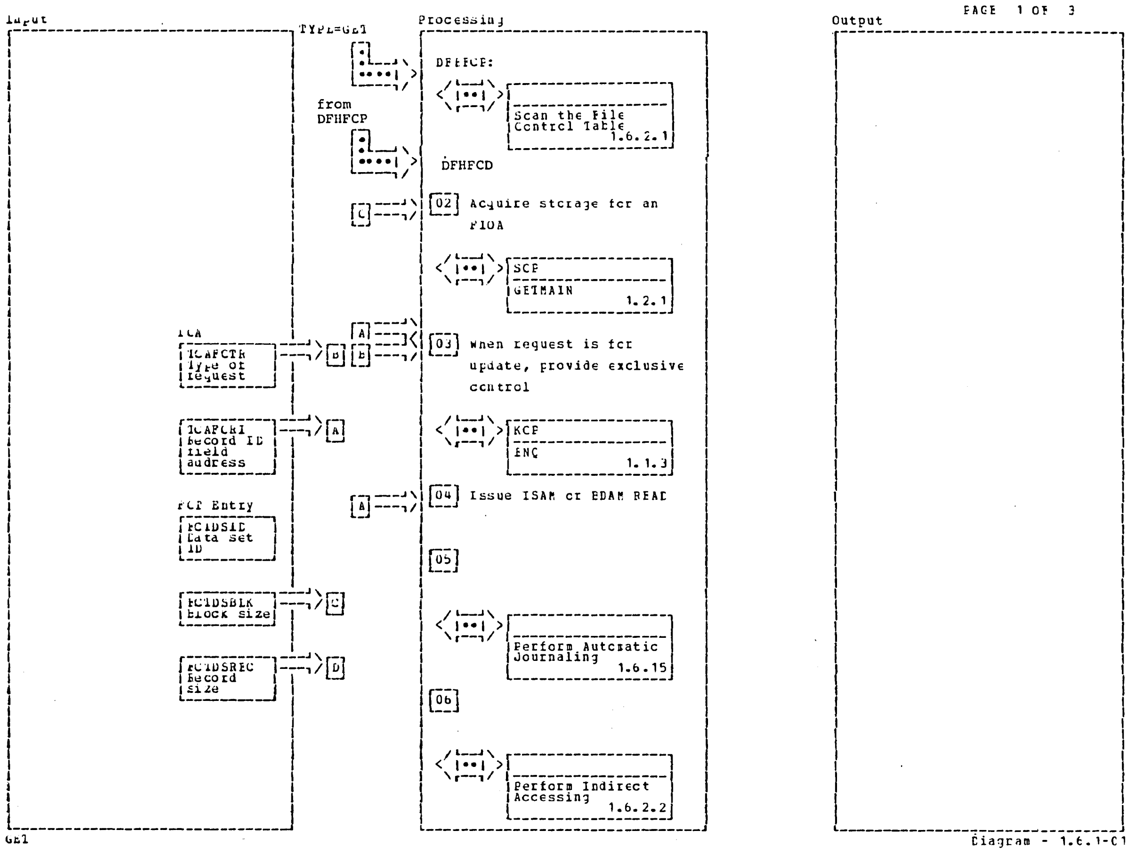
Diagram 1.6.0-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Note that in some cases (eg PUT UPDATE) it is not necessary to scan the File Control Table. The FCT address has been stored on the previous related request (GET UPDATE) and can be picked up from the stored location. | | | |
| 02 Note that not all ISAM and BDAM requests are processed in DFHFCD. For the following ISAM or BDAM requests the processing is carried out completely in DFHFCP:<br>OPEN<br>CLOSE<br>LOCATE<br>ESETL<br>RELEASE<br>GETAREA<br>Linkages back to DFHFCP are made from DFHFCD in order to use the common routines in DFHFCP. Exit for these requests is always made at the appropriate exit in DFHFCP. The labels which follow are those in DFHFCP which are referenced by DFHFCD for linkage purposes. | FCDSINVR<br>FCXSENDF<br>FCXSRNFR<br>FCXSIOER<br>FCXSNSR<br>FCXIPUX<br>FCXSIRR<br>FCXDUPR | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | FCFSAN<br>FCFSANB<br>FCFSANC<br>FCFSAND<br>FCFSANE<br>FCRFAN<br>FCAJRNN<br>FCJOURNL<br>FCDWEACN<br>FCECNA<br>FCOUTSEG<br>FCISSNB<br>FCISSNA<br>FCISSNC<br>FCRFSAN<br>FCRFFIOA<br>FCIXUUT<br>FCIACNA<br>FCFRIFLN<br>FCCVTDBN<br>FCLOCKYN<br>FCIRE | |

Module Structure

Diagram 1.6.0-01



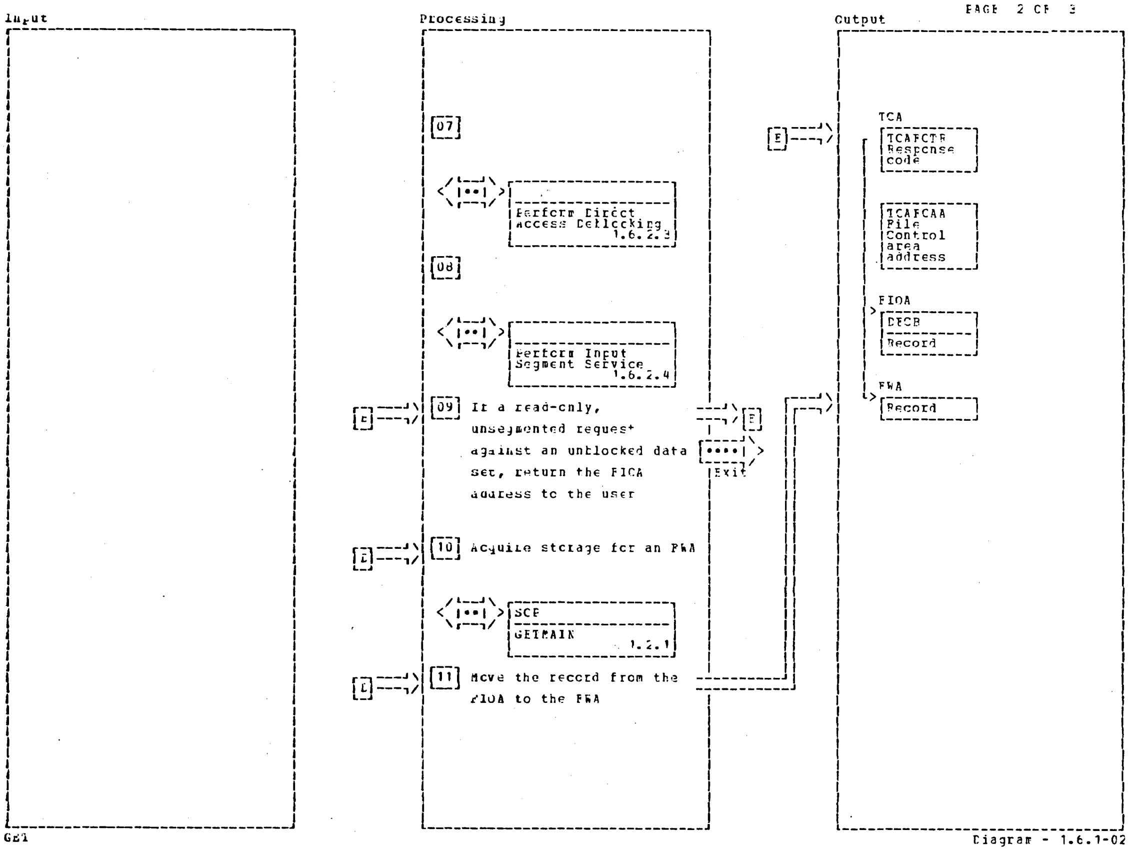
02 Acquire storage for an FIOA

03 When request is for update, provide exclusive control

04 Issue ISAM or BDAM READ

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 The storage required is the FIOA length + blocksize. If the request is for update, the length also includes the record identification field length + 8 for two journal event control number save areas. | FCPSAN | FCPSANC | | | | | |
| 03 Exclusive control for ISAM is provided by enqueuing on the concatenation of data set name and key. If the data set is protected (LOG=YES) the enqueue is maintained until sync point or end of task. For BDAM, exclusive control is provided via a READ EXCLUSIVE request to BDAM. If the dataset is protected, an enqueue is also issued and maintained until sync point or end of task. This enqueue is on the concatenation of data set name, block reference, and key, if any. | GET | | | | | | |

GE1 Diagram - 1.6.1-C1



09 If a read-only, unsegmented request against an unblocked data set, return the FICA address to the user

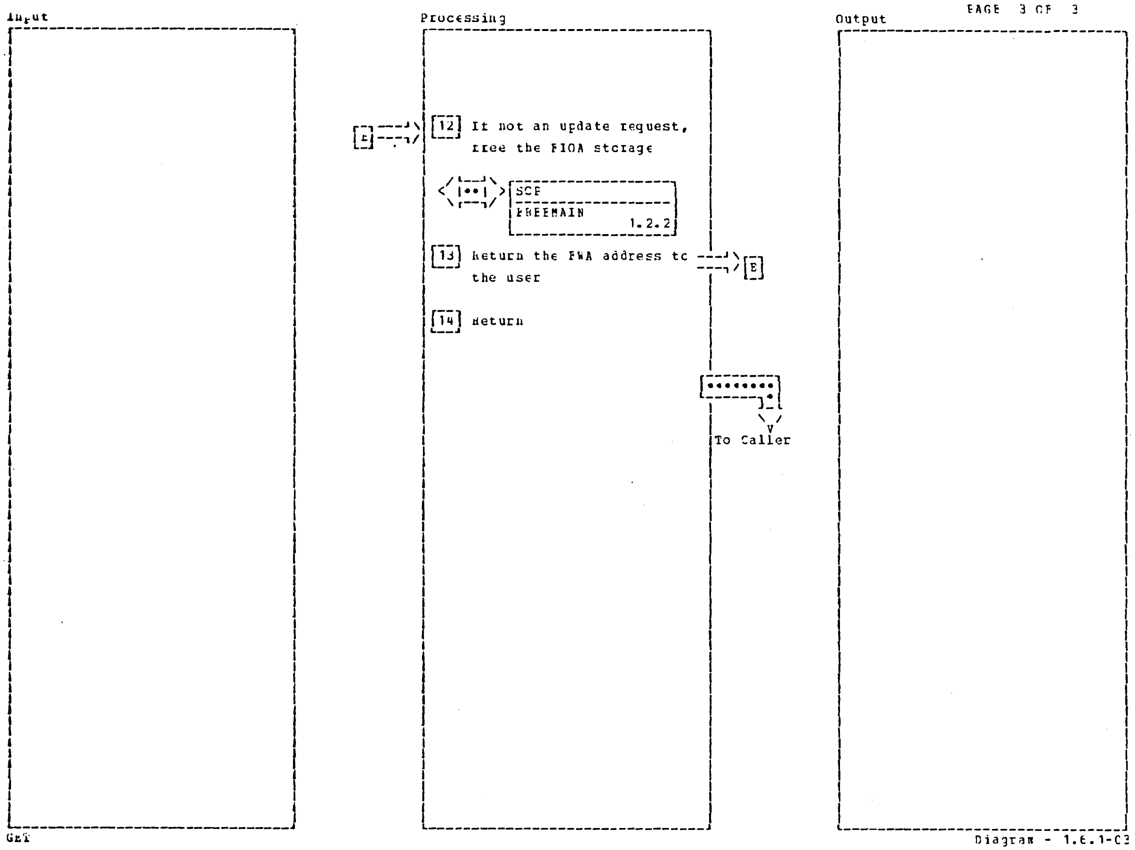
10 Acquire storage for an FWA

11 Move the record from the FIOA to the FWA

Diagram - 1.6.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 10 The length of the storage acquired is the FWA length + record size. | FCFSAN | FCFSANE | | | | | |

G&T



12 If not an update request, free the FIOA storage

13 Return the FWA address to the user

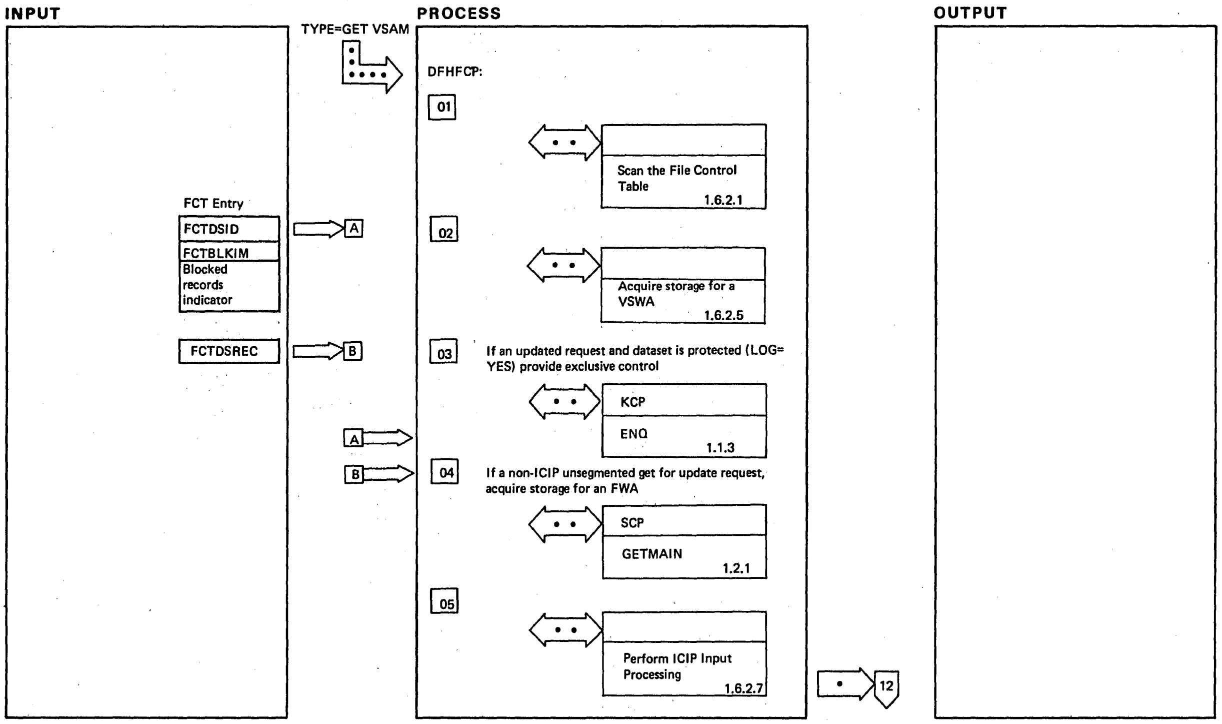
14 Return

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

GET

Diagram - 1.6.1-03

Page 1 of 3


INPUT
PROCESS
OUTPUT
TYPE=GET VSAM
DFHFCP:
01
Scan the File Control Table
1.6.2.1
FCT Entry
FCTDSID
FCTBLKIM
Blocked records indicator
FCTDSREC
A
B
02
Acquire storage for a VSWA
1.6.2.5
03
If an updated request and dataset is protected (LOG= YES) provide exclusive control
KCP
ENQ
1.1.3
04
If a non-ICIP unsegmented get for update request, acquire storage for an FWA
SCP
GETMAIN
1.2.1
05
Perform ICIP Input Processing
1.6.2.7
12

VSAM GET

Diagram - 1.6.2-01

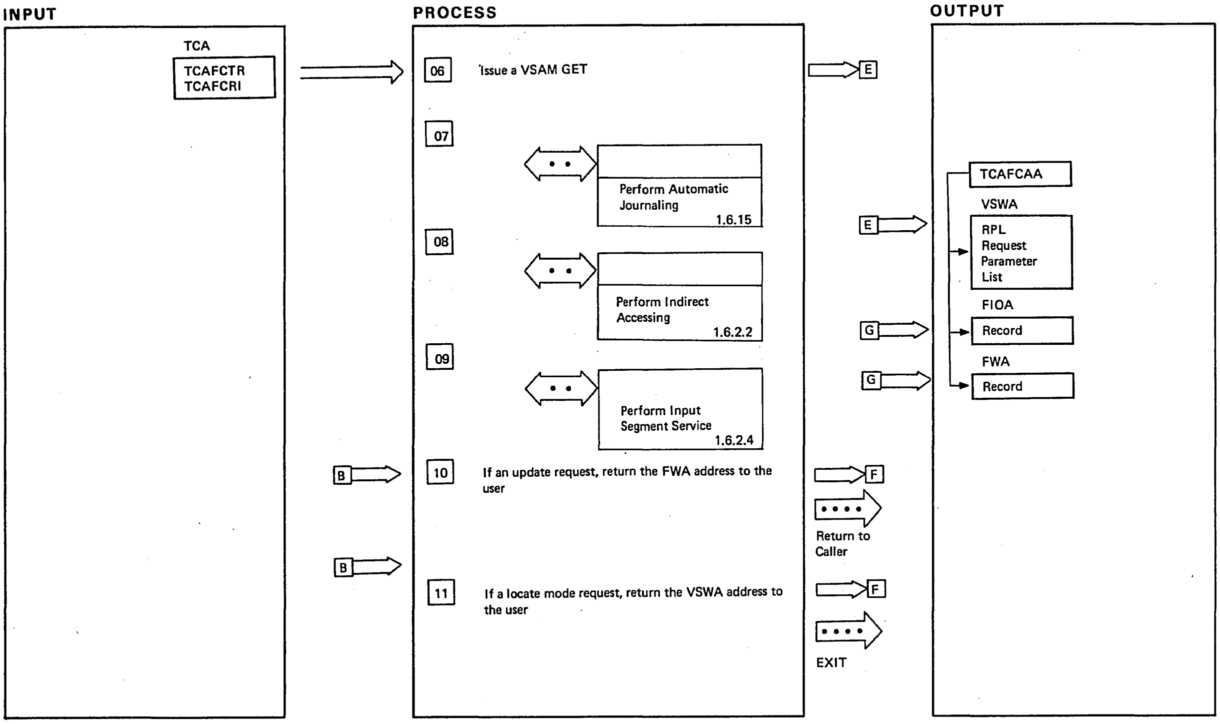
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 03 If dataset is a KSDS, the search must be full key equal. The enqueue argument is the concatenation of data set name and key or RBA | | | |
| 04 The storage length is the FWA length and record size + 4 if variable-length for the LLbb field | FCFSAN | FCFSANB | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

VSAM GET

Diagram - 1.6.2-01

Page 2 of 3



06 Issue a VSAM GET

10 If an update request, return the FWA address to the user

11 If a locate mode request, return the VSWA address to the user

VSAM GET

Diagram - 1.6.2-02

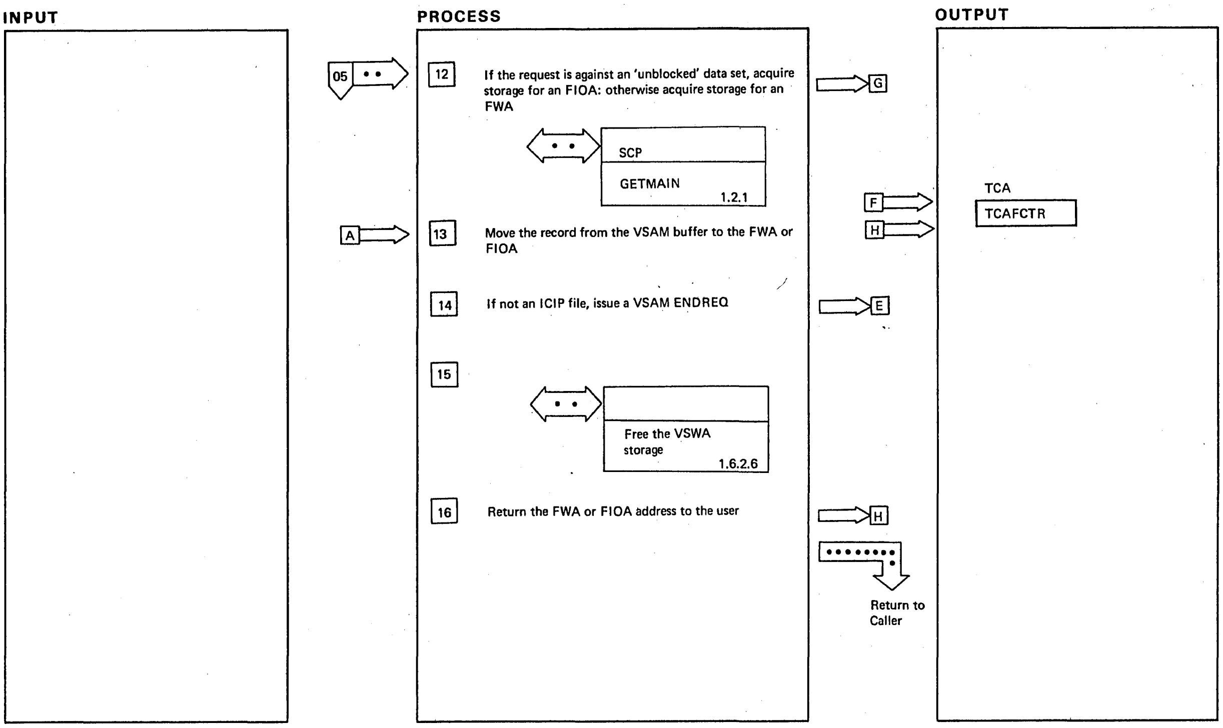
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 06 Unless an update request, will be locate mode | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

VSAM GET

Diagram - 1.6.2-02

Page 3 of 3



INPUT

PROCESS

OUTPUT

12 If the request is against an 'unblocked' data set, acquire storage for an FIOA: otherwise acquire storage for an FWA

SCP GETMAIN 1.2.1

13 Move the record from the VSAM buffer to the FWA or FIOA

14 If not an ICIP file, issue a VSAM ENDREQ

15 Free the VSWA storage 1.6.2.6

16 Return the FWA or FIOA address to the user

Return to Caller

VSAM GET

Diagram - 1.6.2-03

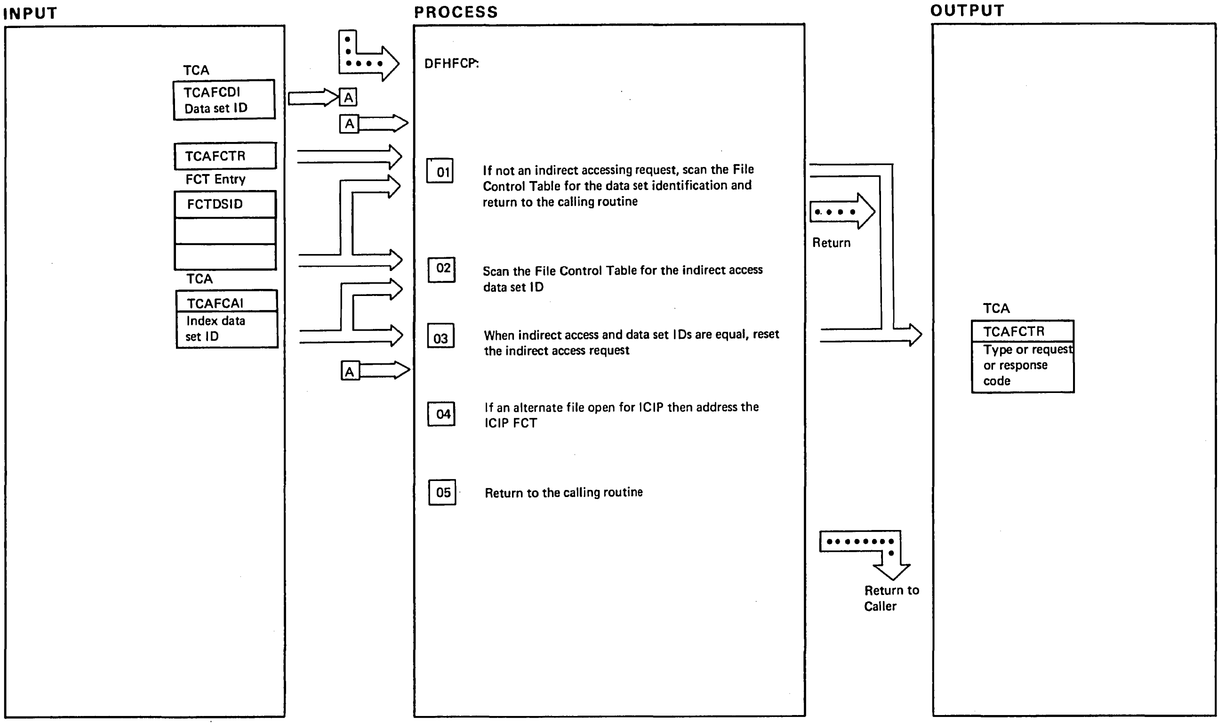
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 12 Since VSAM data sets are neither blocked nor unblocked, an unblocked specification indicates ISAM compatibility | | | |
| 13 If variable-length record, an LLbb field is created | | | |
| 14 For ICIP files the ENDREQ is accomplished by in-line code referencing VSAM control blocks. See 1.6.7-01, process step 08 | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

VSAM GET

Diagram - 1.6.2-03

Page 1 of 1



01 If not an indirect accessing request, scan the File Control Table for the data set identification and return to the calling routine

02 Scan the File Control Table for the indirect access data set ID

03 When indirect access and data set IDs are equal, reset the indirect access request

04 If an alternate file open for ICIP then address the ICIP FCT

05 Return to the calling routine

File Control Table Scan

Diagram - 1.6.2.1-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 If no match, set return code and exit to caller | | | |
| 02 The FCT is scanned by issuing the DFHFC CTYPE=LOCATE macro instruction | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

File Control Table Scan

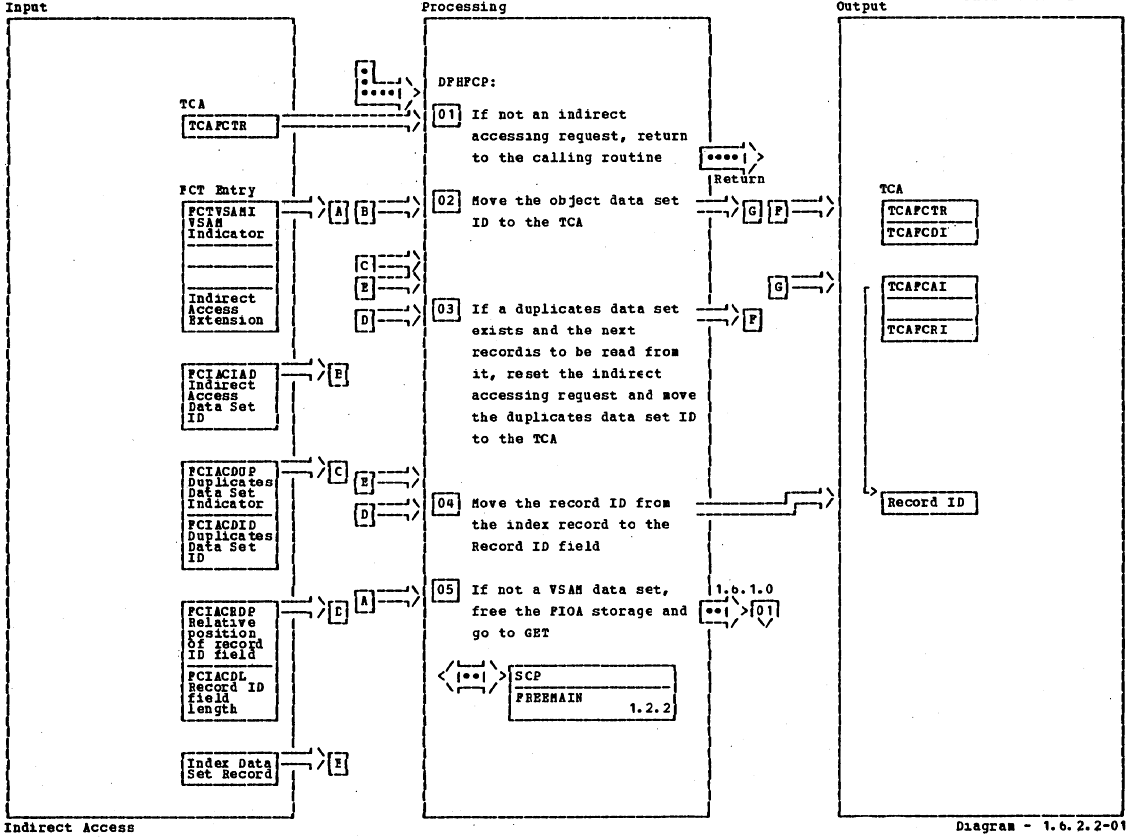
Diagram - 1.6.2.1-01

PAGE 1 OF 2

Input

Processing

Output


TCA
TCAFCTR
FCT Entry
FCTVSAMI VSAM Indicator
Indirect Access Extension
FCIACIAD Indirect Access Data Set ID
FCIACDUP Duplicates Data Set Indicator
FCIACDID Duplicates Data Set ID
FCIACRDP Relative position of record ID field
FCIACDL Record ID field length
Index Data Set Record
A
B
C
D
E
F
G
Return
1.6.1.0
01
SCP
FREEMAIN
1.2.2
TCA
TCAFCTR
TCAFCDI
TCAFCAI
TCAFCRI
Record ID

DFHFCP:

01 If not an indirect accessing request, return to the calling routine

02 Move the object data set ID to the TCA

03 If a duplicates data set exists and the next recordis to be read from it, reset the indirect accessing request and move the duplicates data set ID to the TCA

04 Move the record ID from the index record to the Record ID field

05 If not a VSAM data set, free the FIOA storage and go to GET

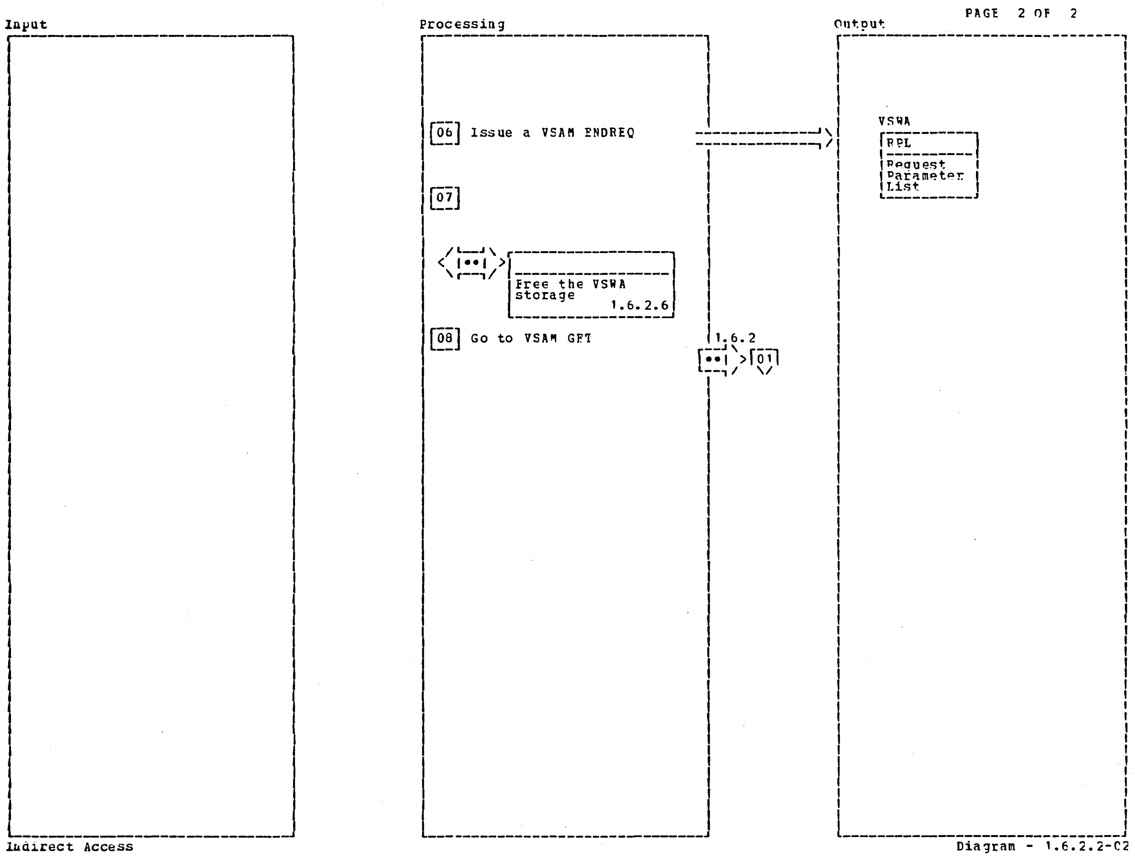
Indirect Access

Diagram - 1.6.2.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 The next record is to be read from the duplicates data set if the first byte of the Record Identification contains the duplicates indicator. | | | | | | | |

Indirect Access

Diagram - 1.6.2.2-01

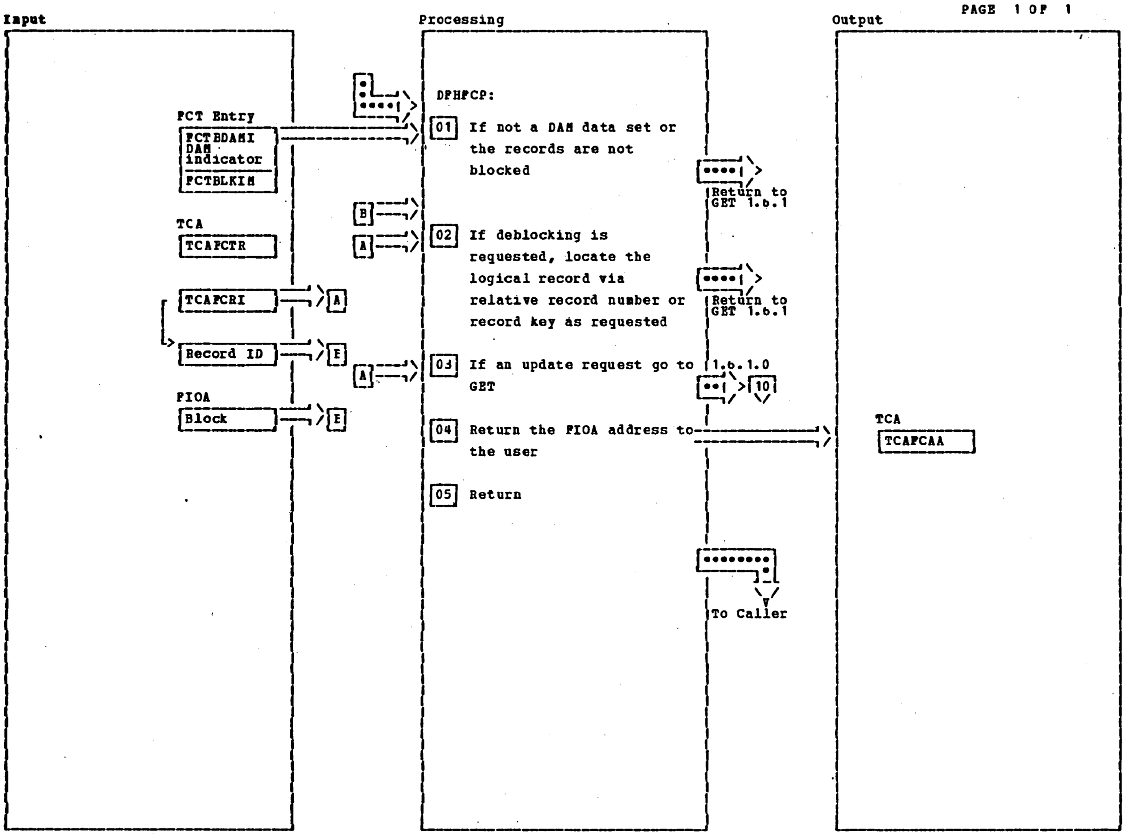


| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |

Indirect Access

Diagram - 1.6.2.2-C2

PAGE 1 OF 1



DFHFCP:

01 If not a DAM data set or the records are not blocked

02 If deblocking is requested, locate the logical record via relative record number or record key as requested

03 If an update request go to GET

04 Return the FIOA address to the user

05 Return

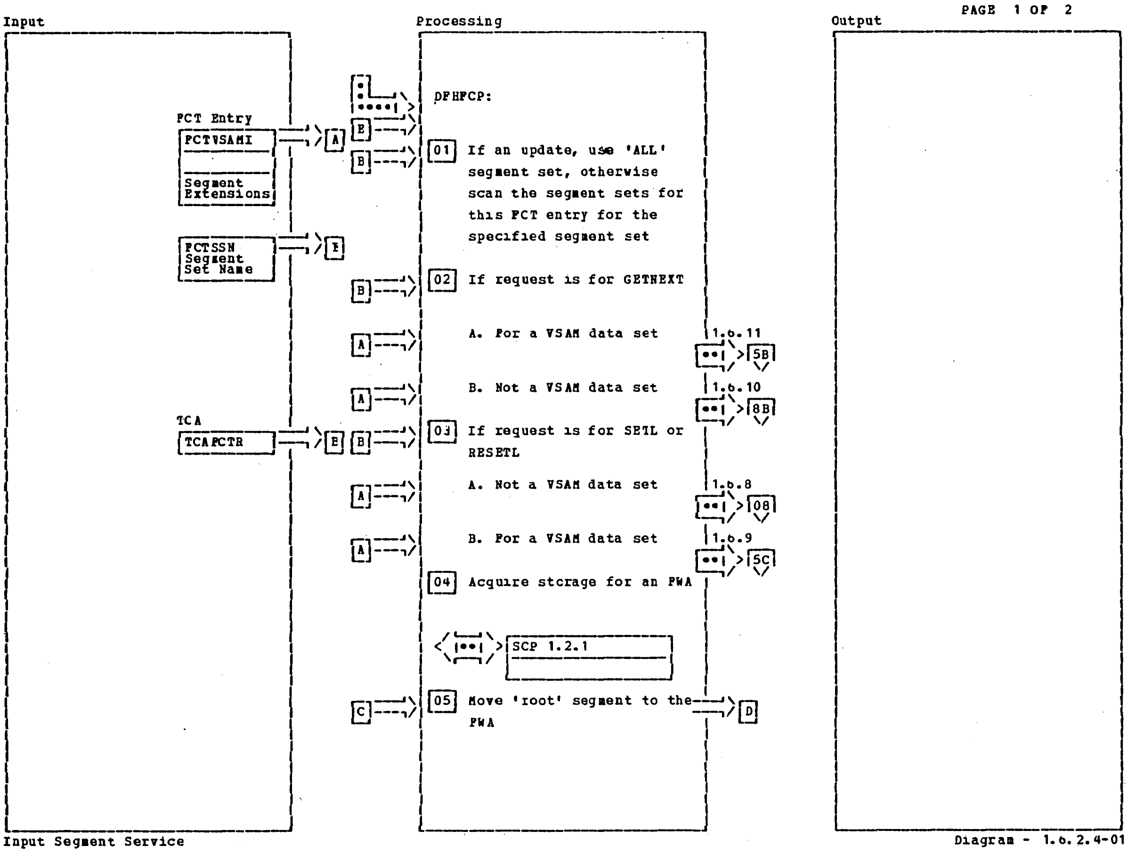
Direct Access Deblocking

Diagram - 1.6.2.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

Direct Access Deblocking

Diagram - 1.6.2.3-01



DFHFCP:

01 If an update, use 'ALL' segment set, otherwise scan the segment sets for this FCT entry for the specified segment set

02 If request is for GETNEXT

A. For a VSAM data set

B. Not a VSAM data set

03 If request is for SETL or RESETL

A. Not a VSAM data set

B. For a VSAM data set

04 Acquire storage for an FWA

05 Move 'root' segment to the FWA

Input Segment Service

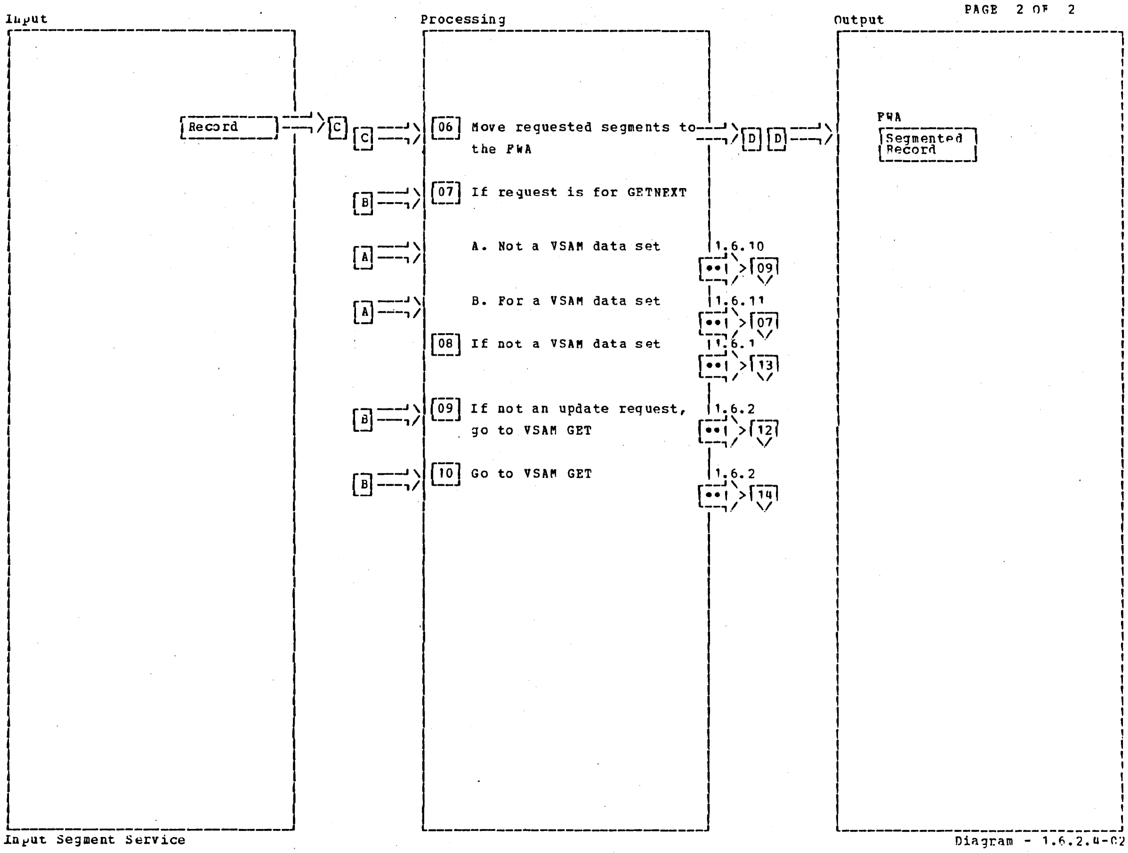
Diagram - 1.6.2.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 Storage acquired will be enough to hold the segment set. | FCFSAN | FCFSANE | | | | | |

Input Segment Service

Diagram - 1.6.2.4-01

PAGE 2 OF 2



06 Move requested segments to the FWA

07 If request is for GETNEXT

A. Not a VSAM data set

B. For a VSAM data set

08 If not a VSAM data set

09 If not an update request, go to VSAM GET

10 Go to VSAM GET

Input Segment Service

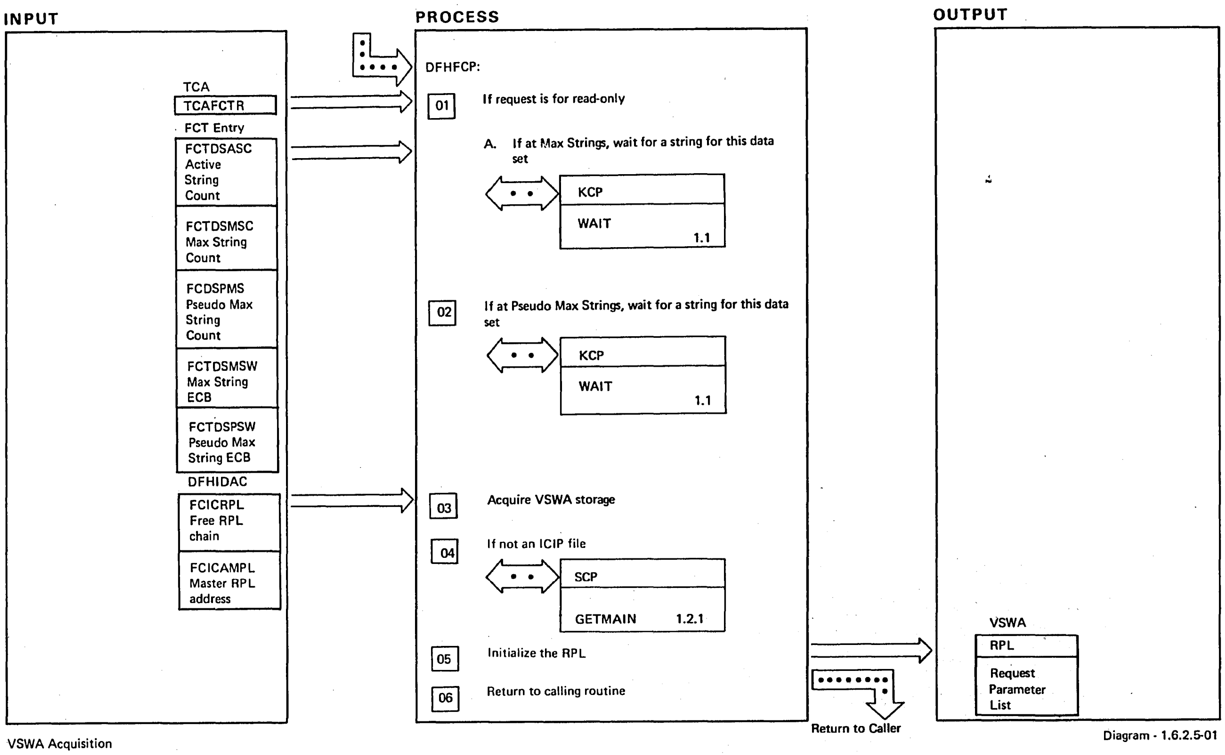
Diagram - 1.6.2.4-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 06 If a segment is present in the record but not part of the segment set, the segment is not moved to the FWA. If a segment is not present in the record but is part of the segment set, space will be left for the maximum length of that segment. | | | | | | | |

Input Segment Service

Diagram - 1.6.2.4-02

Page 1 of 1


INPUT
TCA
TCAFCTR
FCT Entry
FCTDSASC Active String Count
FCTDSMSC Max String Count
FCDSPMS Pseudo Max String Count
FCTDSMSW Max String ECB
FCTDSPSW Pseudo Max String ECB
DFHIDAC
FCICRPL Free RPL chain
FCICAMPL Master RPL address
PROCESS
DFHFCP:
01 If request is for read-only
A. If at Max Strings, wait for a string for this data set
KCP
WAIT 1.1
02 If at Pseudo Max Strings, wait for a string for this data set
KCP
WAIT 1.1
03 Acquire VSWA storage
04 If not an ICIP file
SCP
GETMAIN 1.2.1
05 Initialize the RPL
06 Return to calling routine
Return to Caller
OUTPUT
VSWA
RPL
Request Parameter List

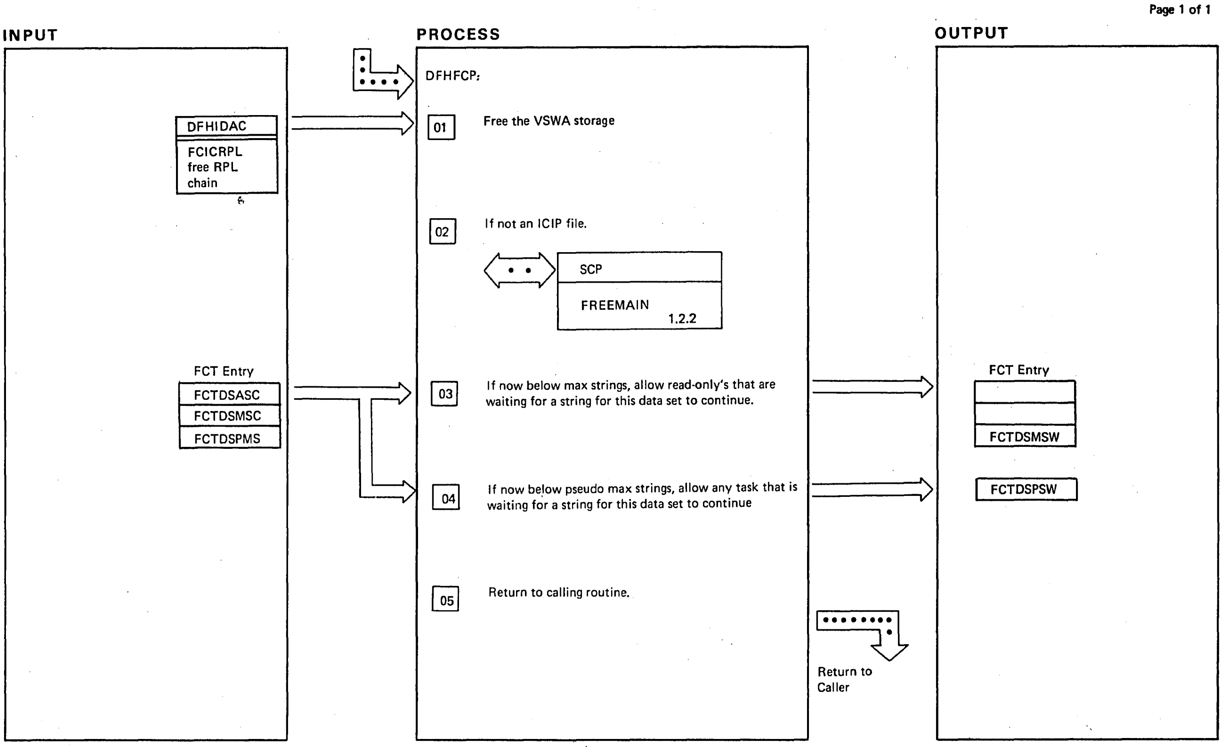
VSWA Acquisition

Diagram - 1.6.2.5-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Read-only requests in this case include segmented requests. | | | |
| A. Wait on maximum number of (Max Strings) ECB (FCTDSMSW) | | | |
| 02 Pseudo Max Strings is approximately 80% of Max Strings so that updates, adds, and browes don't lock out read-only's. Wait on Pseudo Max Strings ECB (FCTDSPSW) | | | |
| 03 Storage length will be as follows: | | | |
| For SETL, the length of a browse VSWA + key length. | | | |
| For DELETE, the length of basic VSWA + record length + 8 for journal ECN save area. | | | |
| If not a GET=UPDATE, the length of a basic VSWA. | | | |
| If a nonsegmented update, the length of a basic VSWA + 8 for journal ECN save areas. | | | |
| If a segmented update, the length of a basic VSWA + record length + 8 for journal ECN save area. | | | |
| For ICIP files the VSWA is obtained from a chain of free VSWAs anchored in IDA. (SCP is not invoked) | | | |

VSWA Acquistion

Diagram - 1.6.2.5-01



01 Free the VSWA storage

02 If not an ICIP file.

03 If now below max strings, allow read-only's that are waiting for a string for this data set to continue.

04 If now below pseudo max strings, allow any task that is waiting for a string for this data set to continue

05 Return to calling routine.

VSWA Disposition

Diagram - 1.6.2.6-01

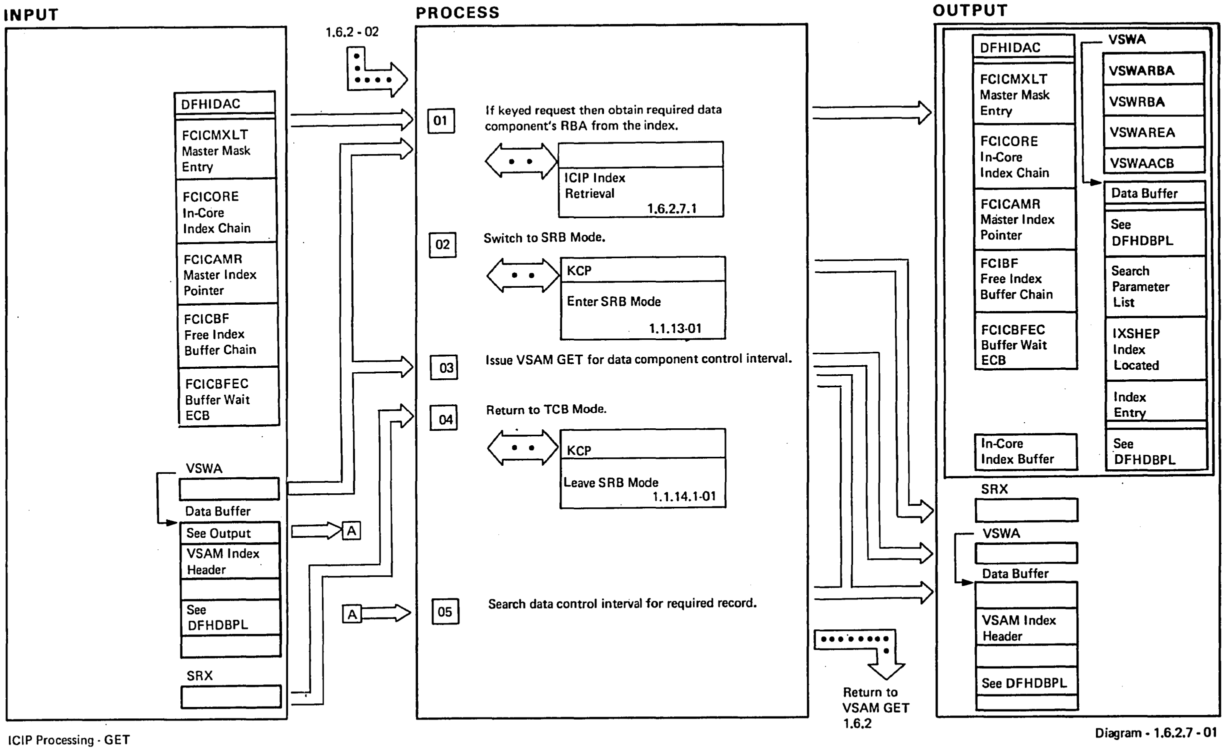
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 For ICIP files the VSWA is returned to the free VSWA chain anchored in IDA (SCP is not invoked) | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

VSWA Disposition

Diagram - 1.6.2.6-01

Page 1 of 1



01 If keyed request then obtain required data component's RBA from the index.

02 Switch to SRB Mode.

03 Issue VSAM GET for data component control interval.

04 Return to TCB Mode.

05 Search data control interval for required record.

ICIP Processing - GET

Diagram - 1.6.2.7 - 01

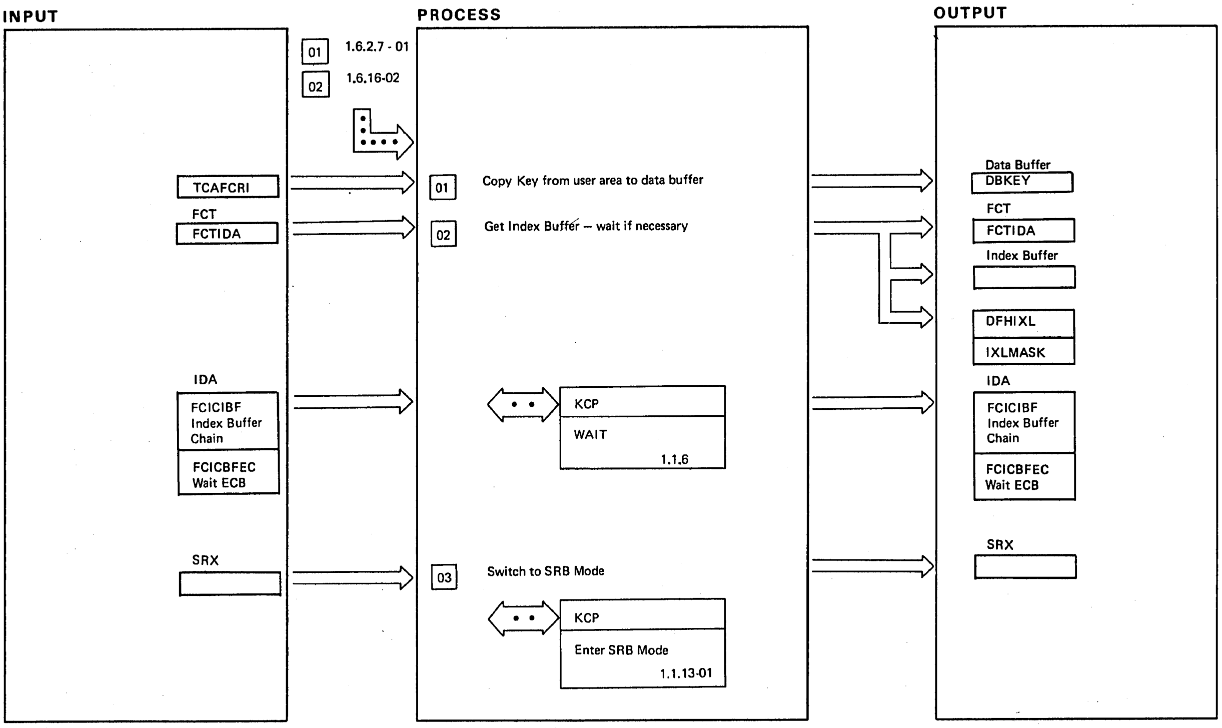
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 This is not necessary for KSDS files if the search index code is already in SRB Mode. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

ICIP Processing - GET

Diagram - 1.6.2.7 - 01

Page 1 of 2



ICIP INDEX Retrieval

Diagram - 1.6.2.7.1 - 01

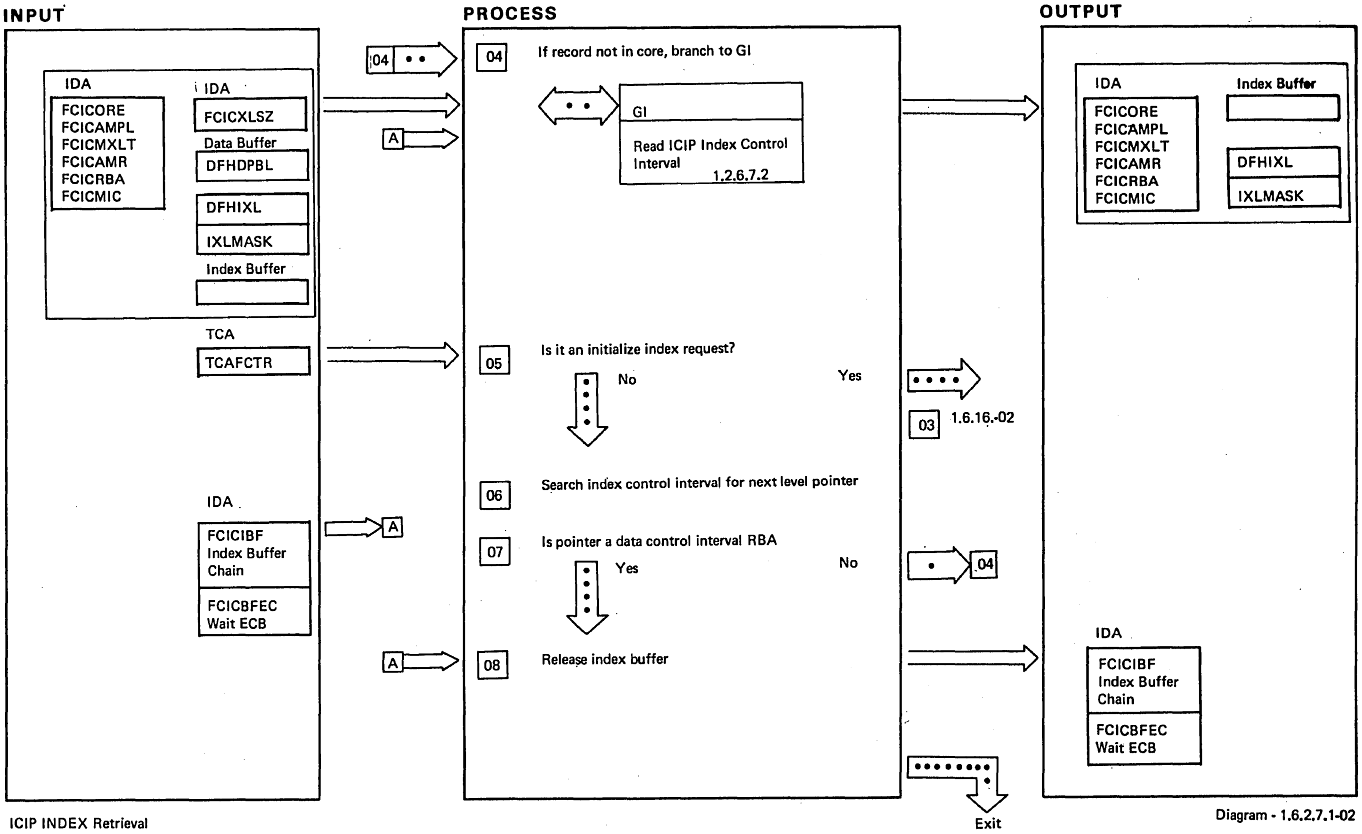
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

ICIP INDEX Retrieval

Diagram - 1.6.2.7.1 - 01

Page 2 of 2



04 If record not in core, branch to GI

05 Is it an initialize index request?

06 Search index control interval for next level pointer

07 Is pointer a data control interval RBA

08 Release index buffer

ICIP INDEX Retrieval

Diagram - 1.6.2.7.1-02

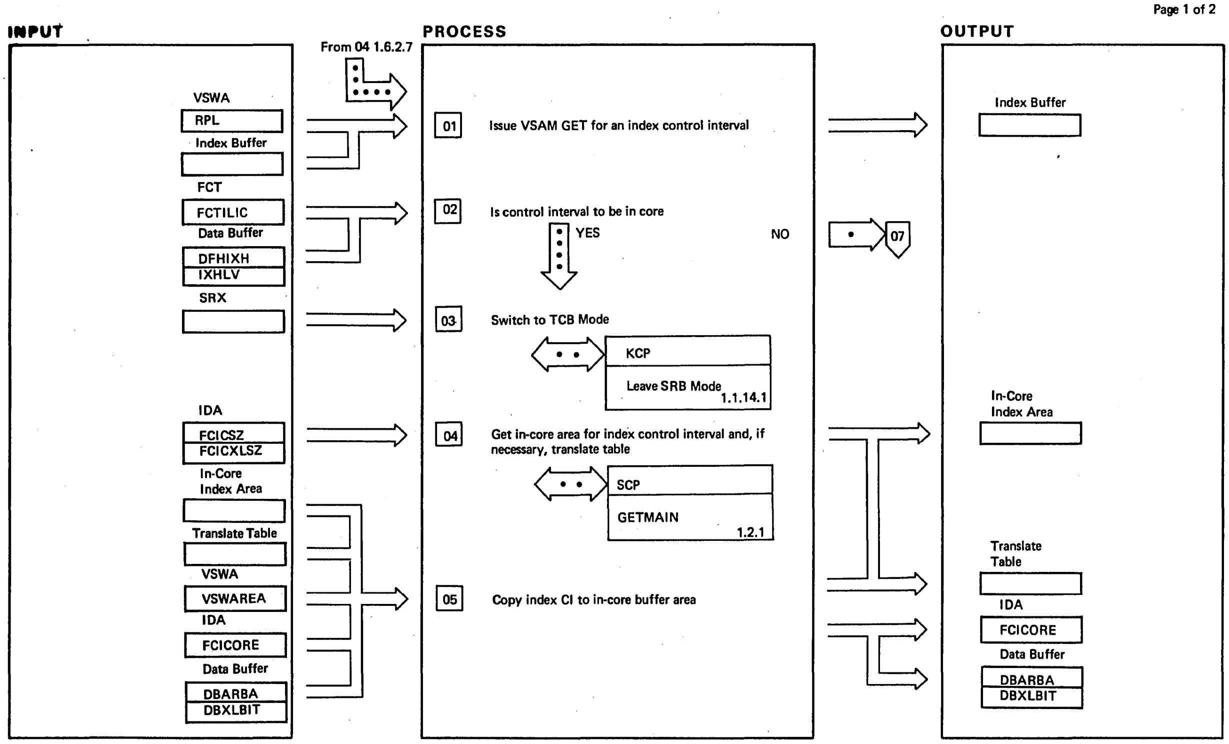
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 05 This code is used by DFHOCP to read-in levels of in-core index when an ICIP file is opened with the INIT parameter. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

ICIP INDEX Retrieval

Diagram - 1.6.2.7.1-02

Page 1 of 2



GI - Read ICIP Index Control Interval

Diagram - 1.6.2.7.2-01

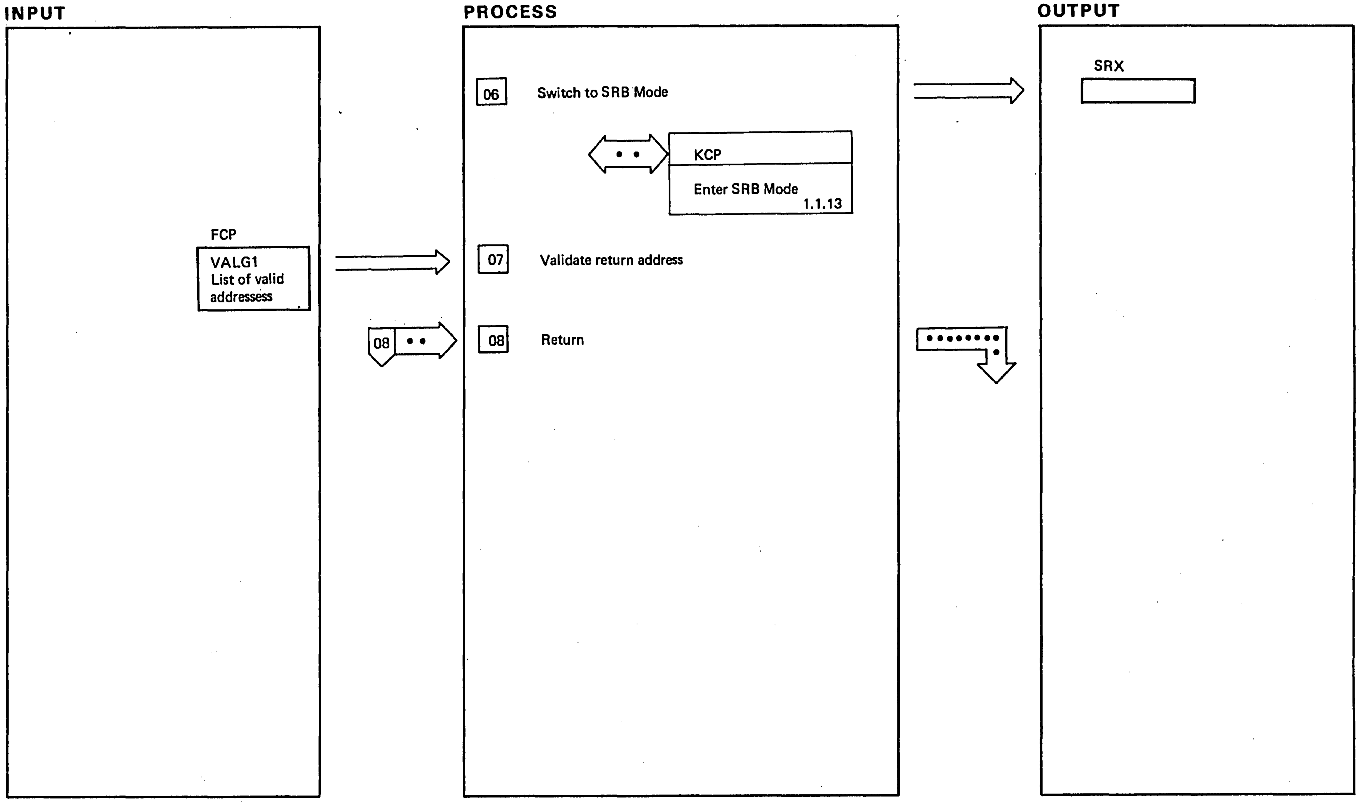
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

GI - Read ICIP Index Control Interval

Diagram - 1.6.2.7.2-01

Page 2 of 2



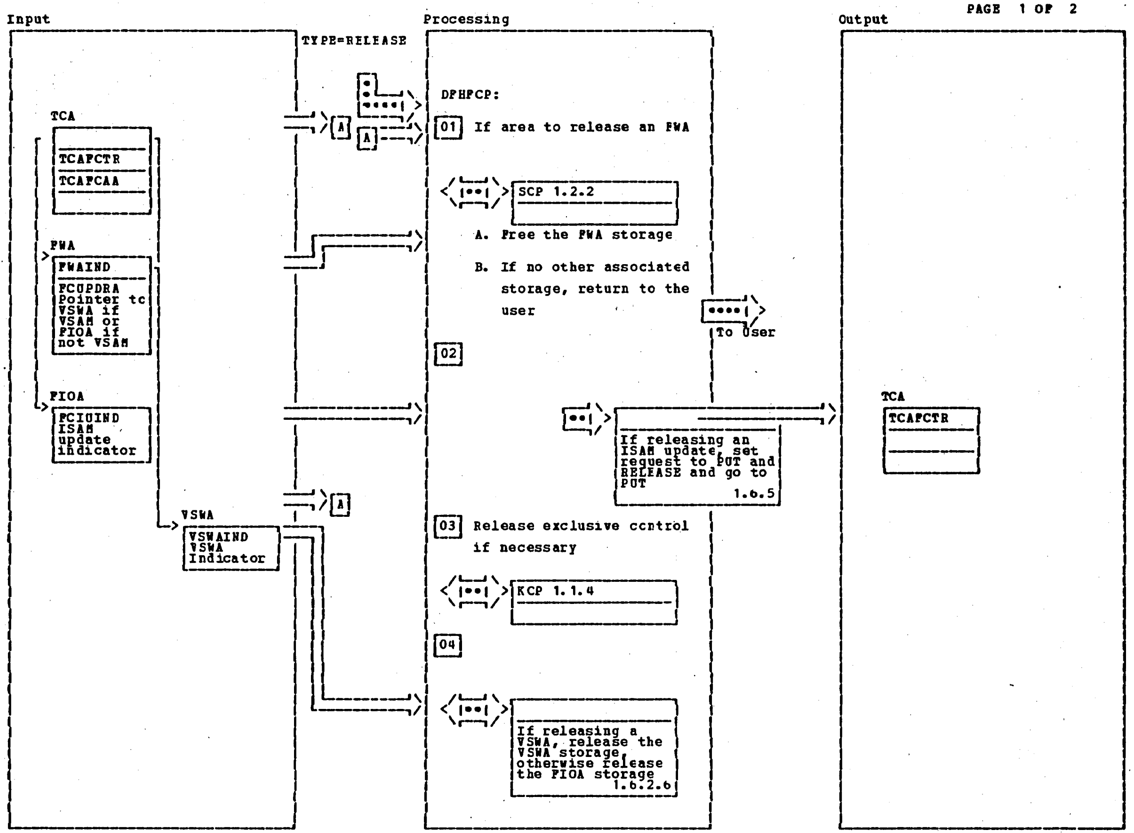
GI - Read ICIP Index Control Interval

Diagram - 1.6.2.7.2-02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 07 Carried out to protect against illegal entry into SRB mode | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

PAGE 1 OF 2



DFHFCP:

01 If area to release an FWA

A. Free the FWA storage

B. If no other associated storage, return to the user

02

If releasing an ISAM update, set request to PUT and RELEASE and go to PUT 1.6.5

03 Release exclusive ccntrol if necessary

04

If releasing a VSWA, release the VSWA storage, otherwise release the FIOA storage 1.6.2.6

RELEASE

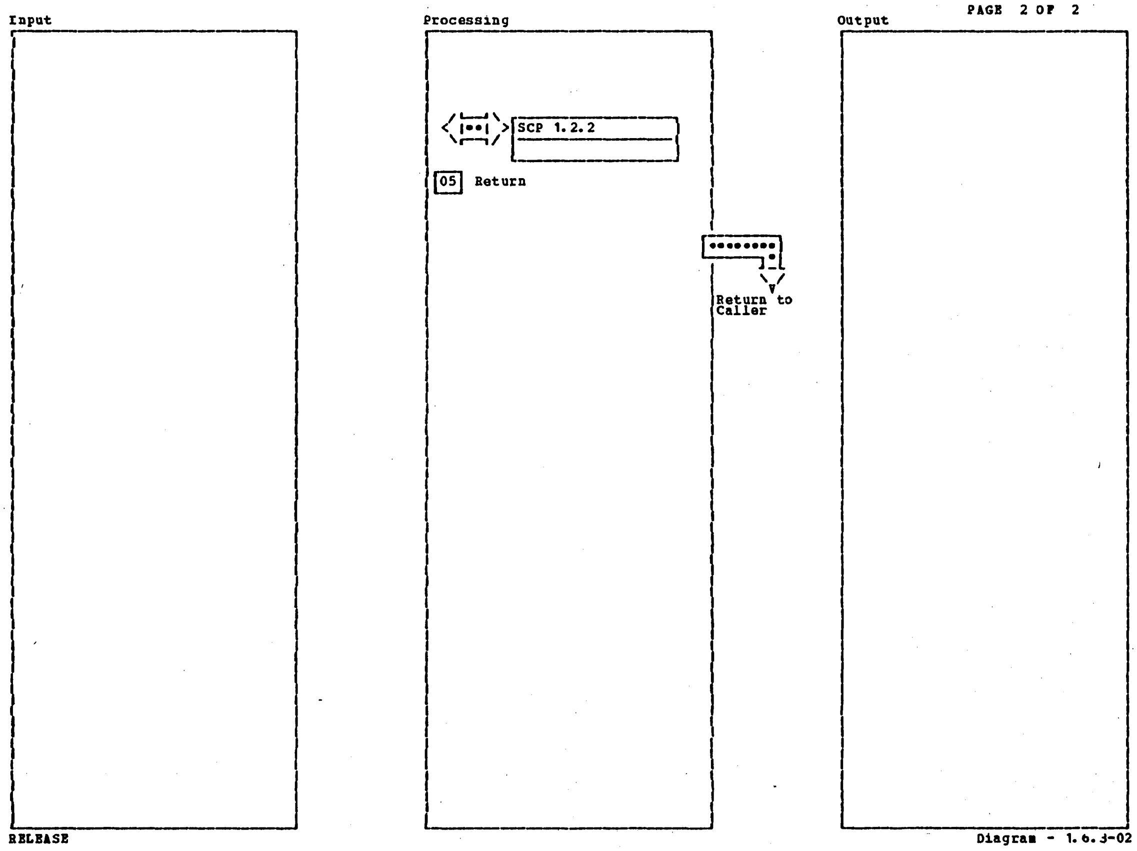
Diagram - 1.6.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 If data set modifications are being logged (see Automatic Journaling Chart), exclusive control is maintained until end of task. Otherwise exclusive control is released by dequeuing on concatenated data set name and key. | | | 1.6.15 | | | | |

RELEASE

Diagram - 1.6.3-01

PAGE 2 OF 2



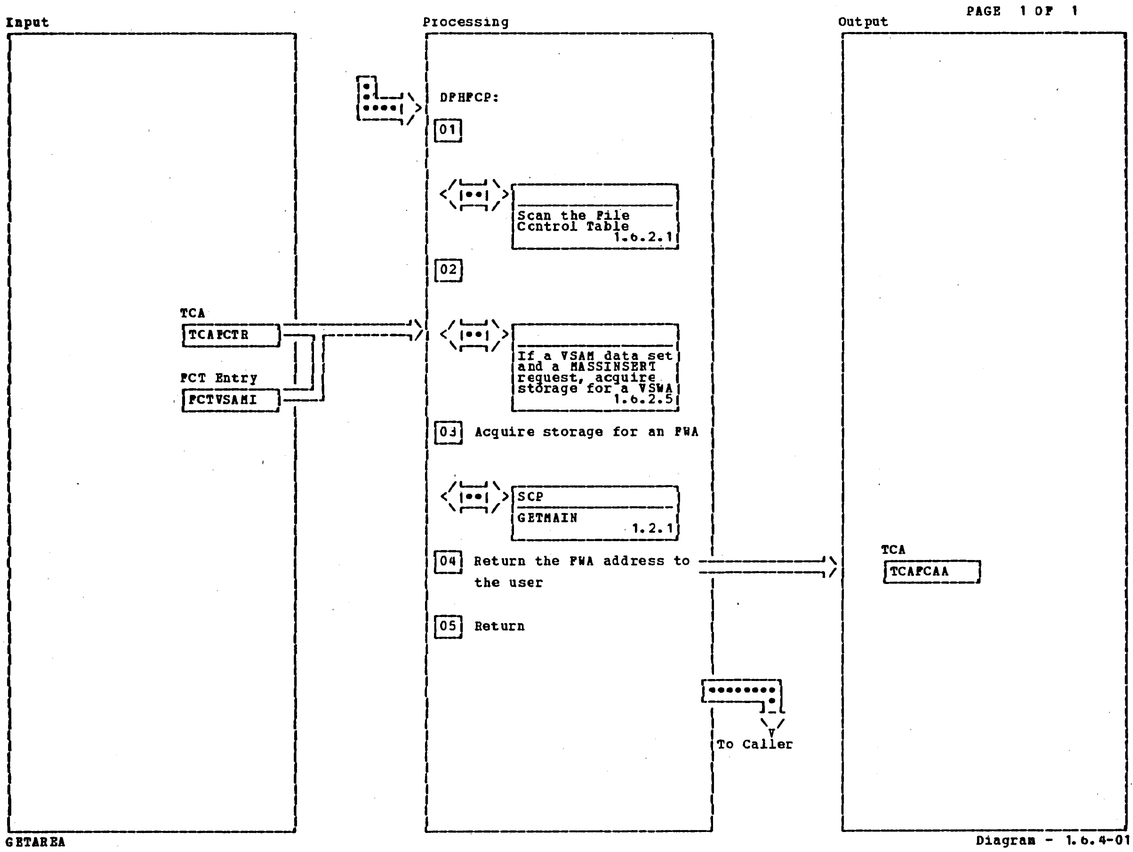
RELEASE

Diagram - 1.6.3-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

RELEASE

Diagram - 1.6.3-02


Input
Processing
Output
PAGE 1 OF 1
DFHFCP:
01
Scan the File Control Table 1.6.2.1
02
TCA
TCAFCTR
FCT Entry
FCTVSAMI
If a VSAM data set and a MASSINSERT request, acquire storage for a VSWA 1.6.2.5
03 Acquire storage for an FWA
SCP
GETMAIN 1.2.1
04 Return the FWA address to the user
TCA
TCAFCAA
05 Return
To Caller

GETAREA

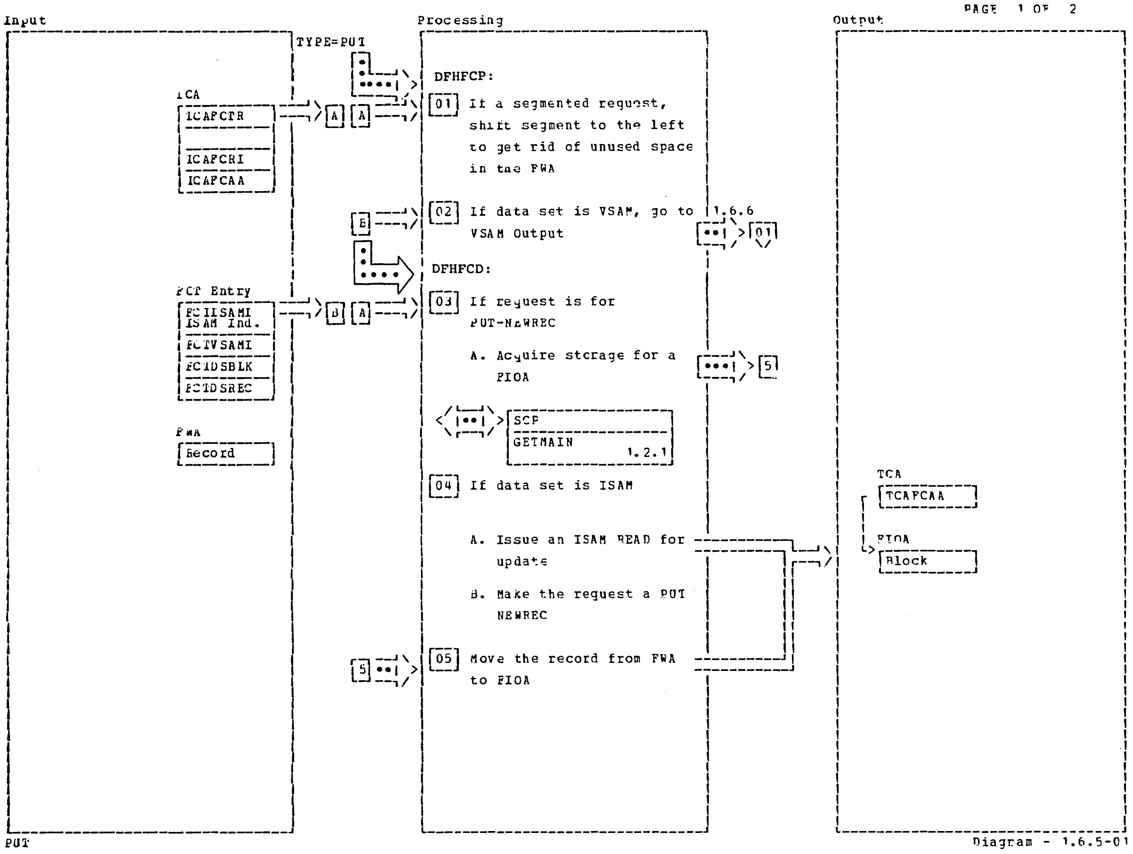
Diagram - 1.6.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

GETAREA

Diagram - 1.6.4-01

PAGE 1 OF 2



DFHFCP:

01 If a segmented request, shift segment to the left to get rid of unused space in the FWA

02 If data set is VSAM, go to VSAM Output

DFHFCD:

03 If request is for PUT-NEWREC

A. Acquire storage for a FIOA

04 If data set is ISAM

A. Issue an ISAM READ for update

B. Make the request a PUT NEWREC

05 Move the record from FWA to FIOA

PUT

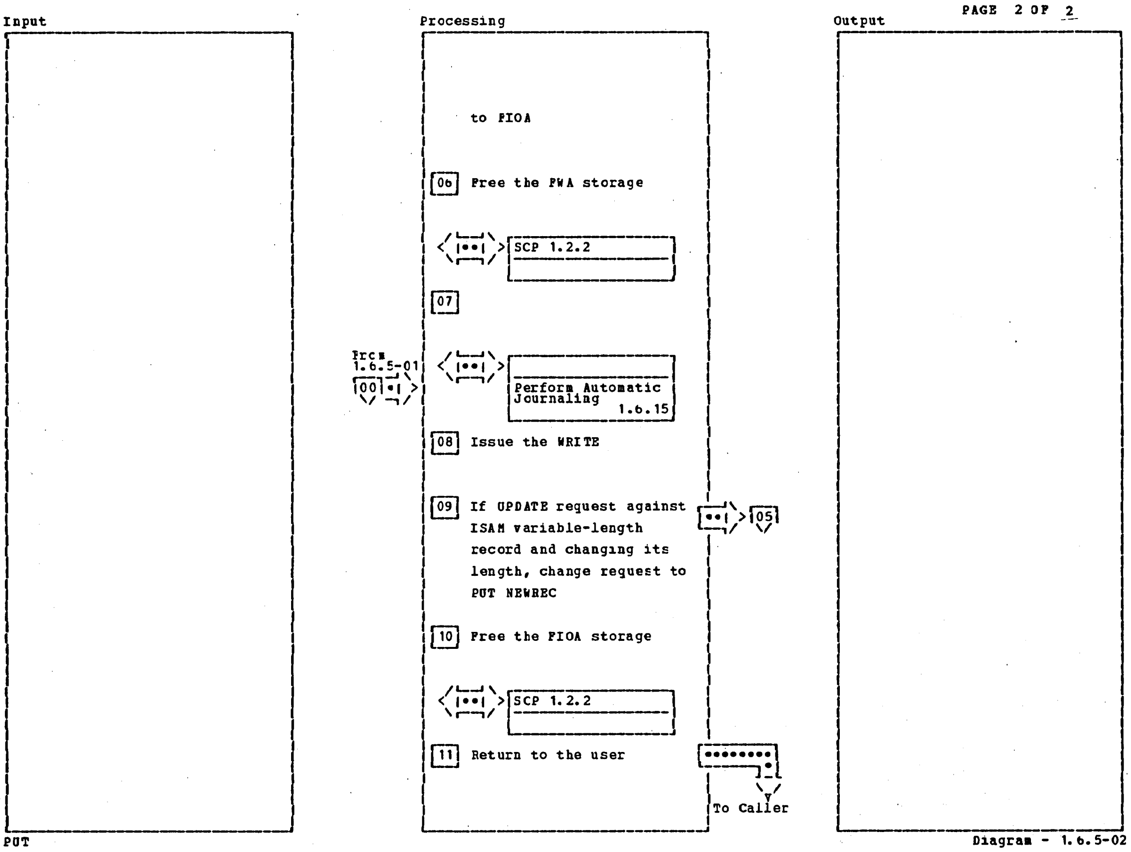
Diagram - 1.6.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 To avoid lockouts against the data set within the ISAM modules, the GET for update is deferred until the PUT is requested. | | | | | | | |

PUT

Diagram - 1.6.5-01

PAGE 2 OF 2



Input

Processing

Output

to FIOA

06 Free the FWA storage

07

08 Issue the WRITE

09 If UPDATE request against ISAM variable-length record and changing its length, change request to PUT NEWREC

10 Free the FIOA storage

11 Return to the user

PUT

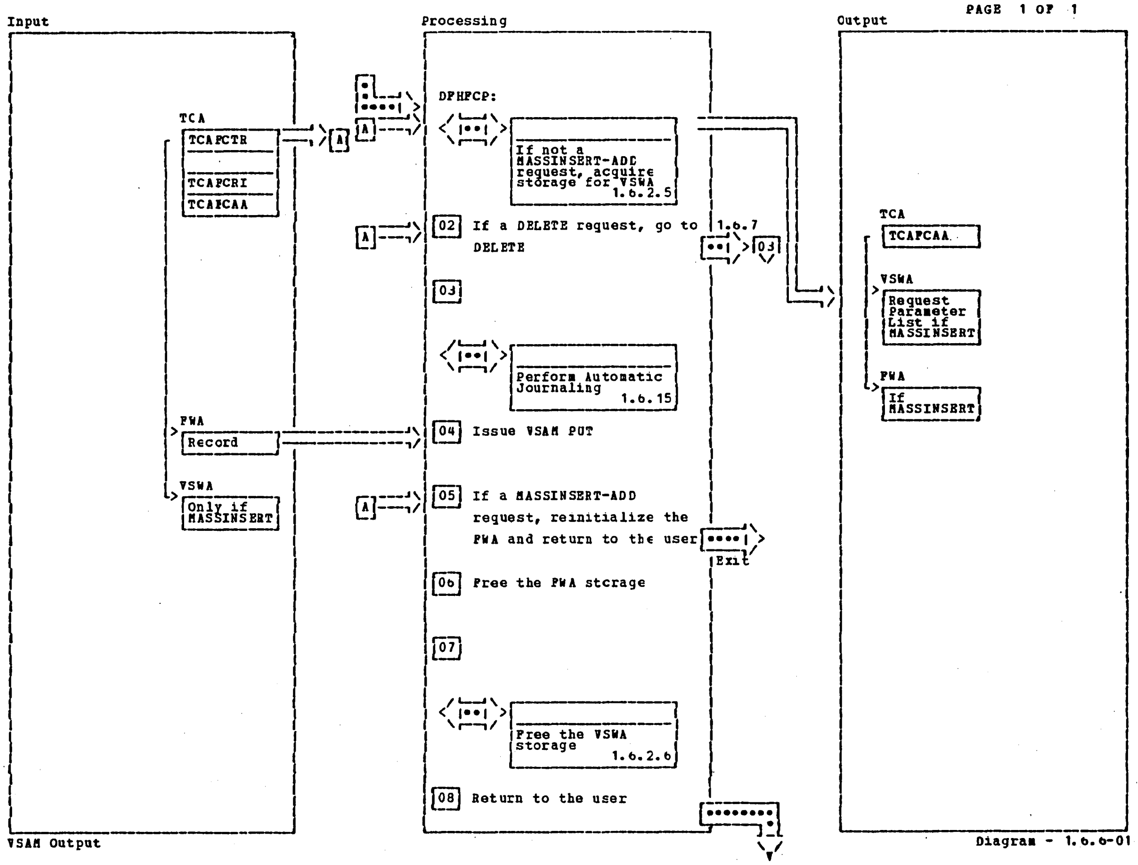
Diagram - 1.6.5-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

PUT

Diagram - 1.6.5-02

PAGE 1 OF 1


Input
Processing
Output
TCA
TCAFCTR
TCAFCRI
TCAFCAA
FWA
Record
VSWA
Only if MASSINSERT
A
DFHFCP:
If not a MASSINSERT-ADD request, acquire storage for VSWA 1.6.2.5
02 If a DELETE request, go to DELETE
1.6.7
03
03
Perform Automatic Journaling 1.6.15
04 Issue VSAM PUT
05 If a MASSINSERT-ADD request, reinitialize the FWA and return to the user
Exit
06 Free the FWA storage
07
Free the VSWA storage 1.6.2.6
08 Return to the user
TCA
TCAFCAA
VSWA
Request Parameter List if MASSINSERT
FWA
If MASSINSERT

VSAM Output

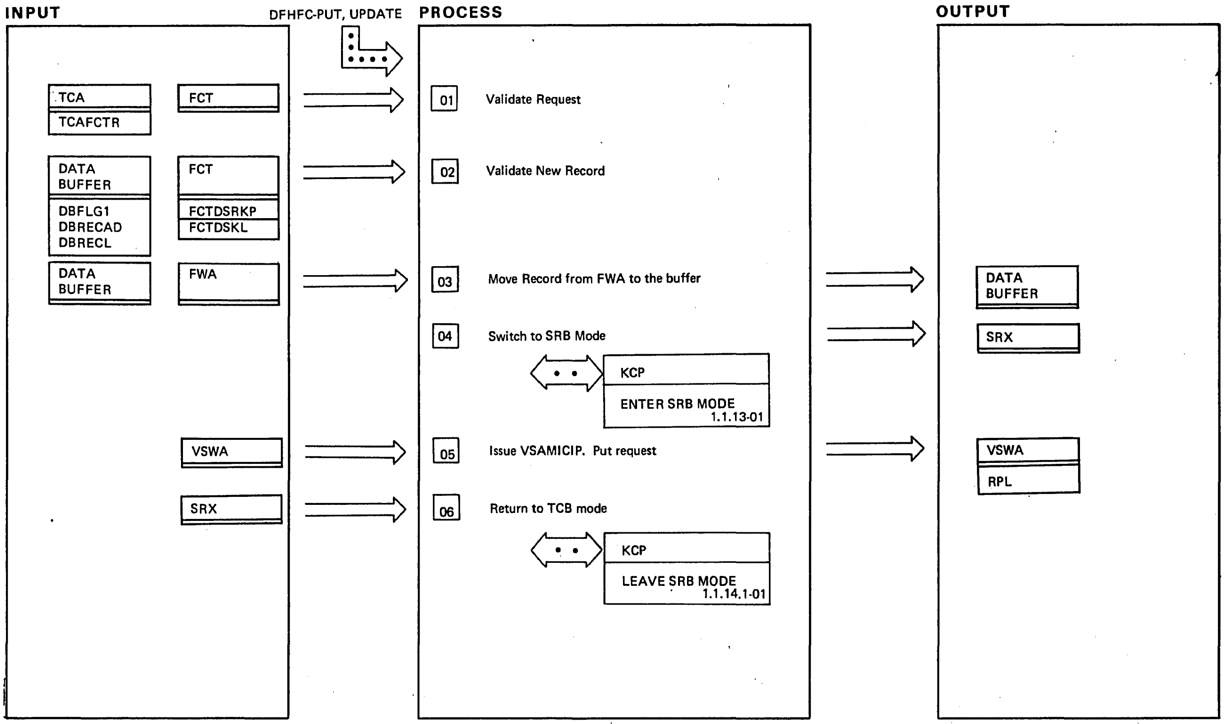
Diagram - 1.6.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

VSAM Output

Diagram - 1.6.6-01

Page 1 of 2



ICIP Output

Diagram - 1.6.6.1-01

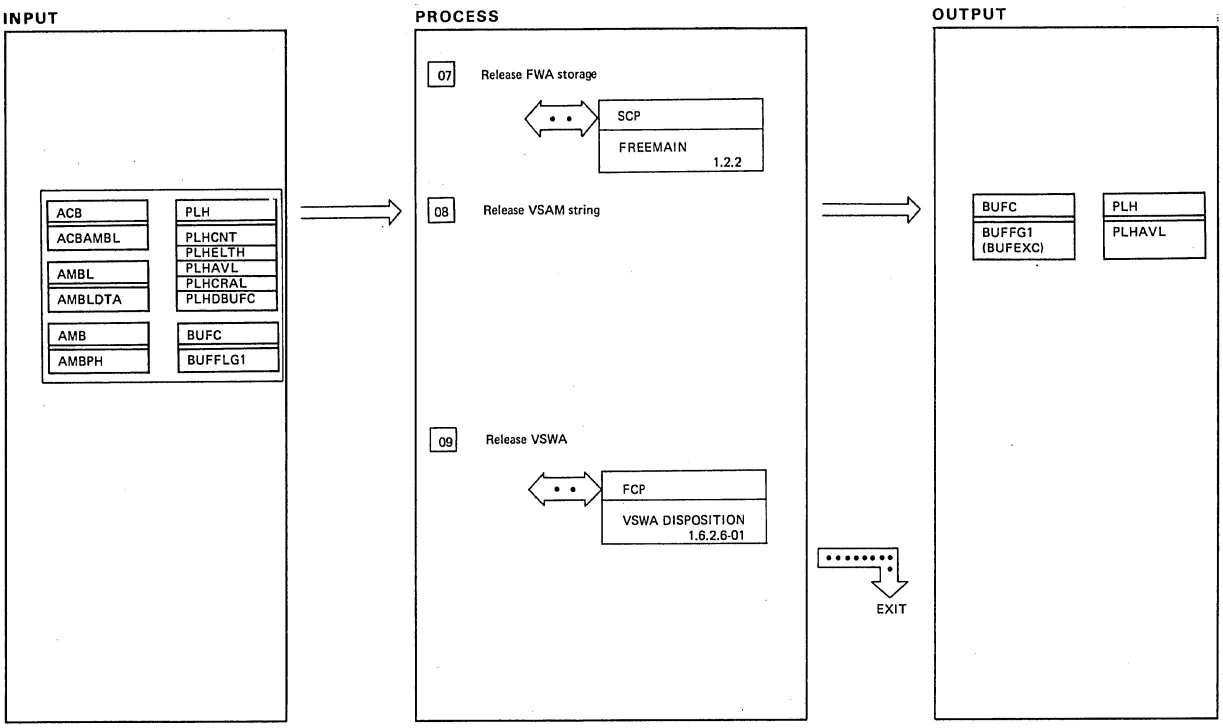
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 Checks for change in record length or change of the key value these are not allowed for ICIP files | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

ICIP Output

Diagram - 1.6.6.1-01

Page 2 of 2



ICIP Output

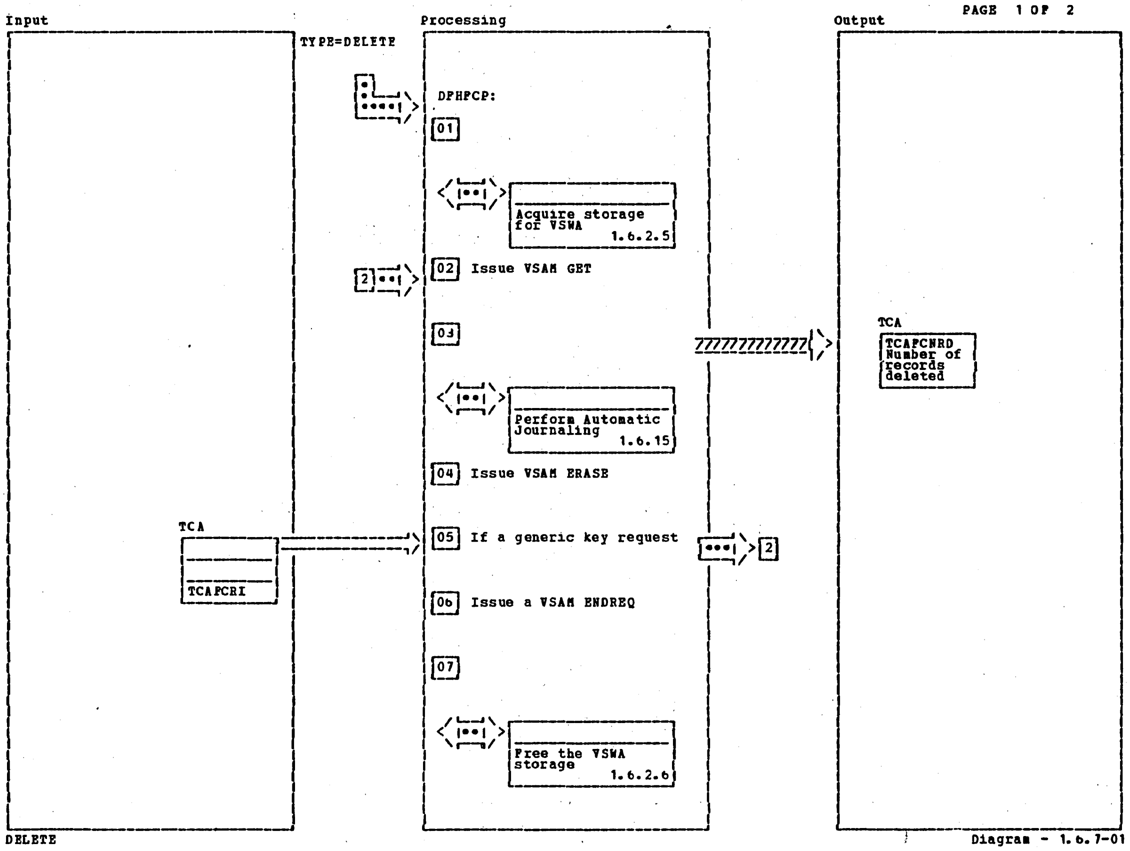
Diagram - 1.6.6.1-02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

ICIP Output

Diagram - 1.6.6.1-02

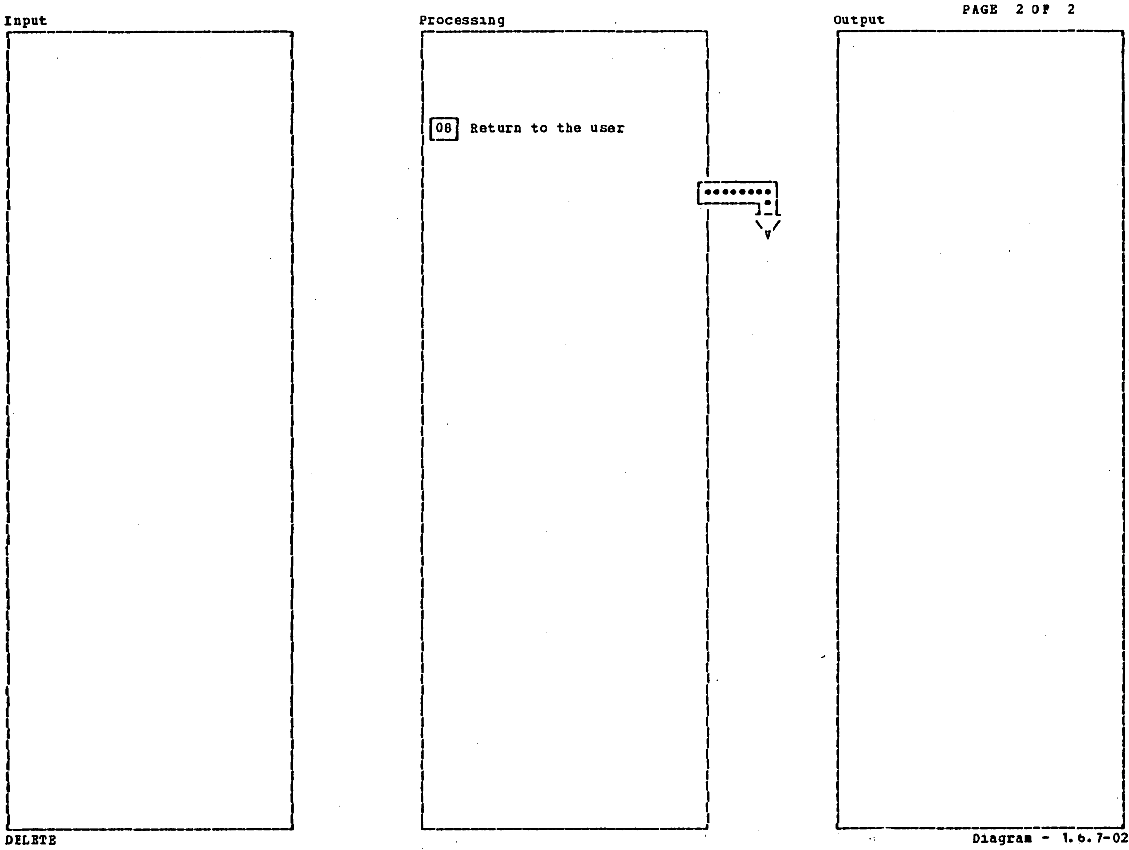

PAGE 1 OF 2
Input
Processing
Output
TYPE=DELETE
DFHFCP:
01
Acquire storage for VSWA 1.6.2.5
2
02 Issue VSAM GET
03
TCA
TCAFCNRD Number of records deleted
Perform Automatic Journaling 1.6.15
04 Issue VSAM ERASE
TCA
TCAFCRI
05 If a generic key request
2
06 Issue a VSAM ENDREQ
07
Free the VSWA storage 1.6.2.6

DELETE Diagram - 1.6.7-01

| Notes | Routine | Label | Ref | Notes | ıe | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 Steps 2 thru 5 will be repeated for a generic key until the GET results in a 'no record found' condition. The number of records deleted is being maintained in the TCA at TCAFCNRD. | | | | | | | |

DELETE Diagram - 1.6.7-01

PAGE 2 OF 2



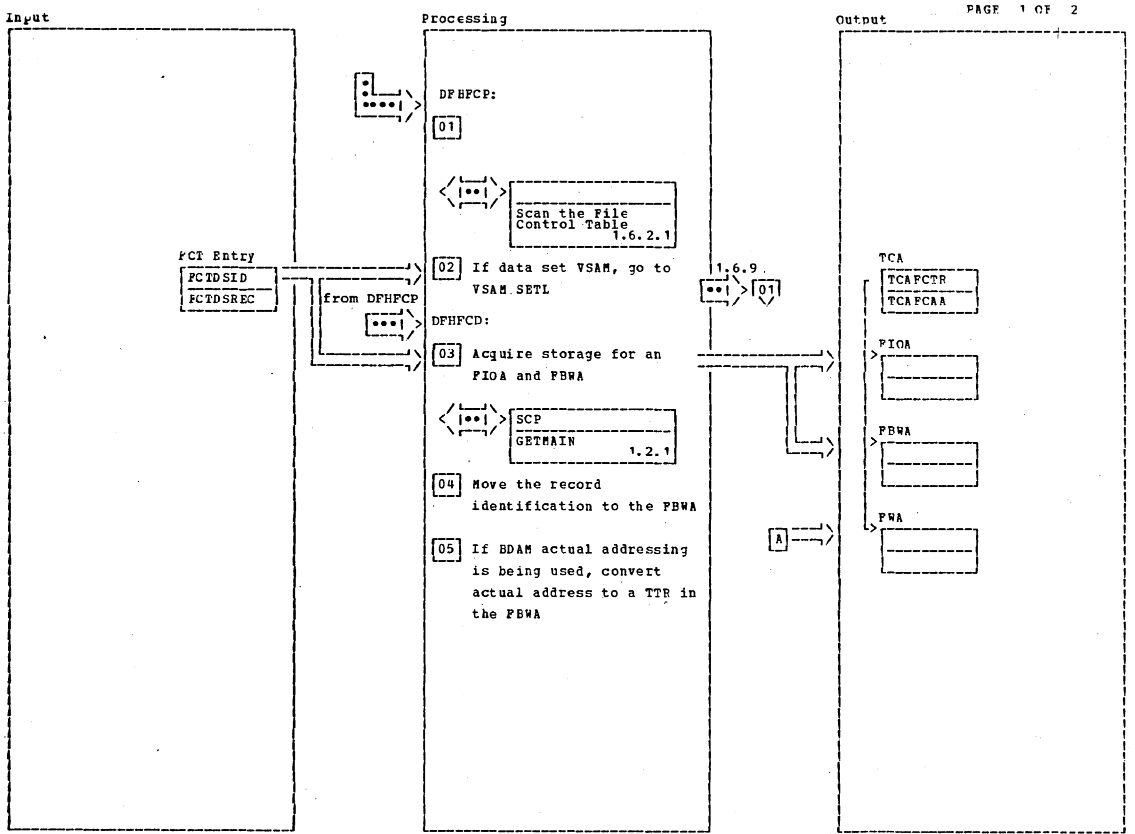
DELETE

Diagram - 1.6.7-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |

DELETE

Diagram - 1.6.7-02



02 If data set VSAM, go to VSAM SETL

03 Acquire storage for an FIOA and FBWA

04 Move the record identification to the FBWA

05 If BDAM actual addressing is being used, convert actual address to a TTR in the FBWA

SETL

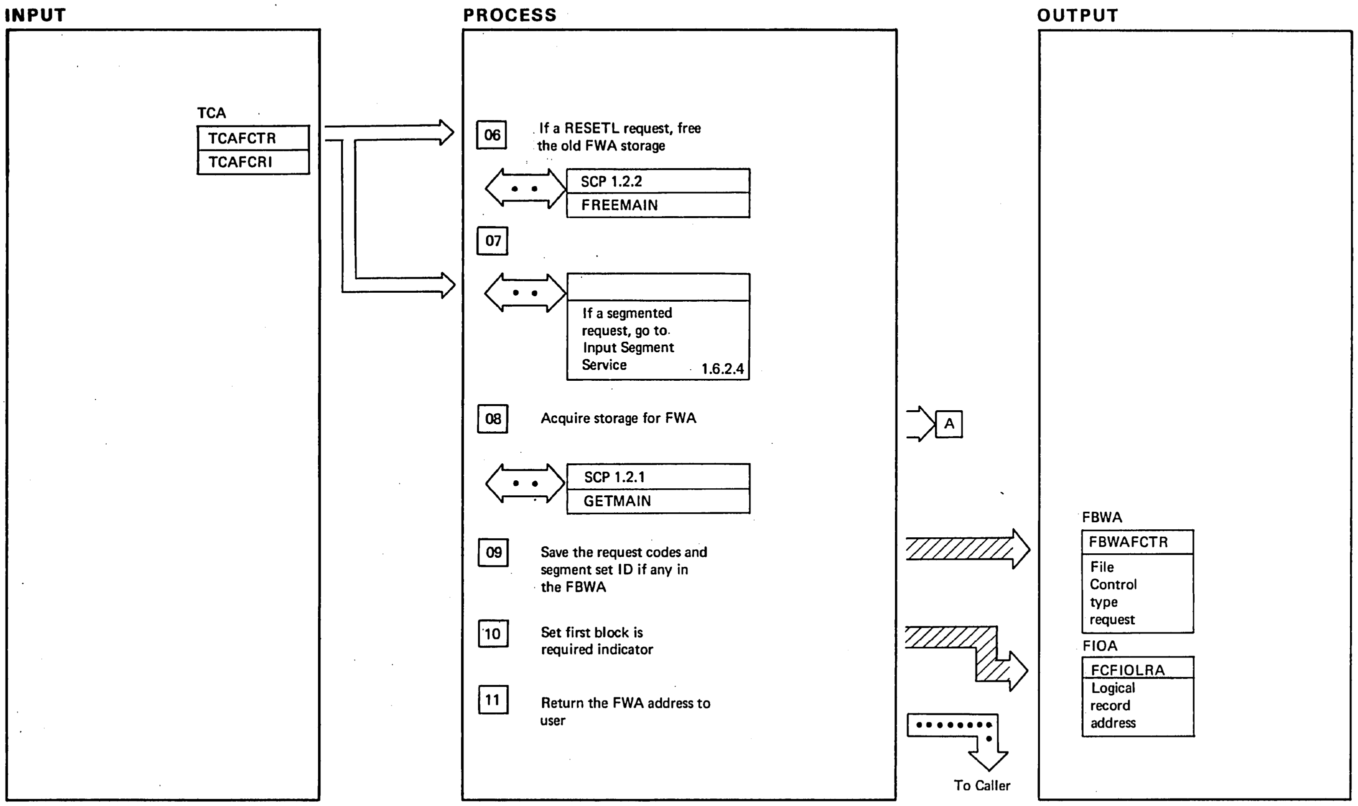
Diagram - 1.6.8-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 This is done using BDAM's address conversion routine. | | | | | | | |

SETL

Diagram - 1.6.8-01

PAGE 2 OF 2



06 If a RESETL request, free the old FWA storage

07 If a segmented request, go to Input Segment Service 1.6.2.4

08 Acquire storage for FWA

09 Save the request codes and segment set ID if any in the FBWA

10 Set first block is required indicator

11 Return the FWA address to user

SETL

Diagram - 1.6.8-02

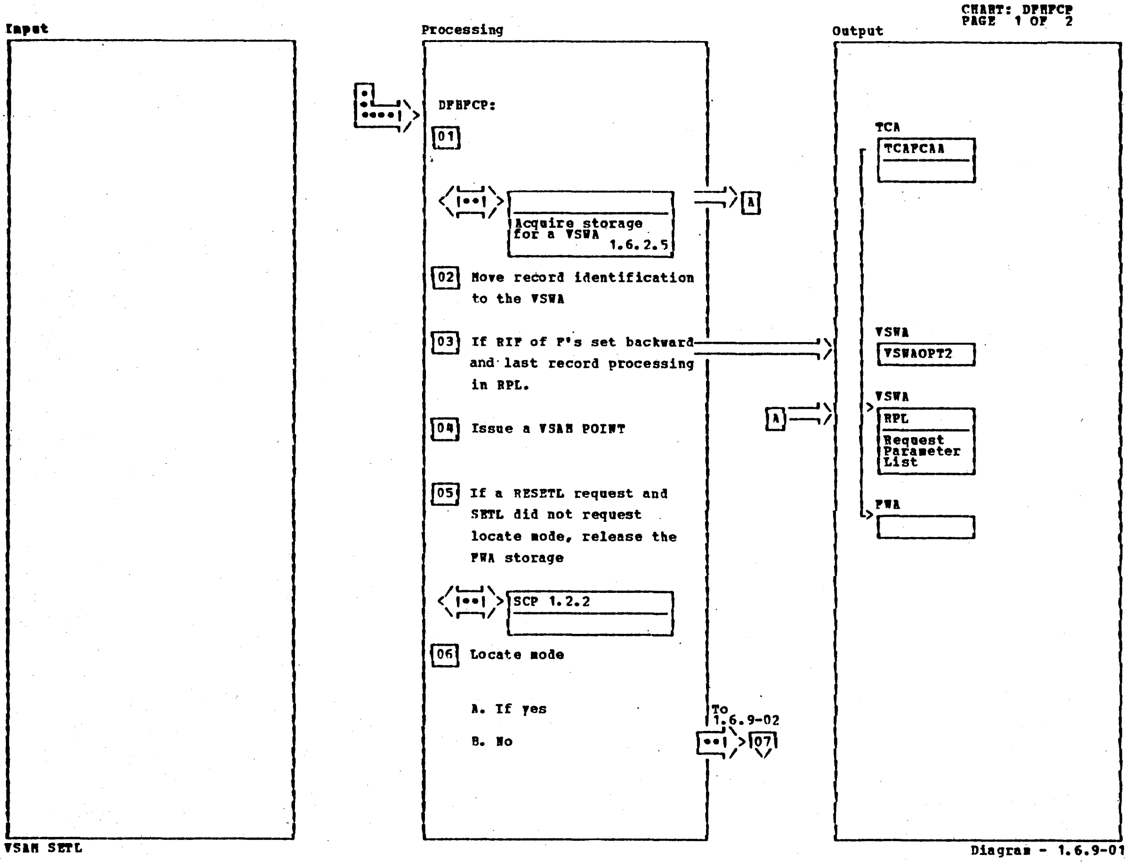
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

SETL

Diagram - 1.6.8-02

CHART: DFHFCP
PAGE 1 OF 2



01

02 Move record identification to the VSWA

03 If RIF of F's set backward and last record processing in RPL.

04 Issue a VSAM POINT

05 If a RESETL request and SETL did not request locate mode, release the FWA storage

06 Locate mode

A. If yes

B. No

VSAM SETL

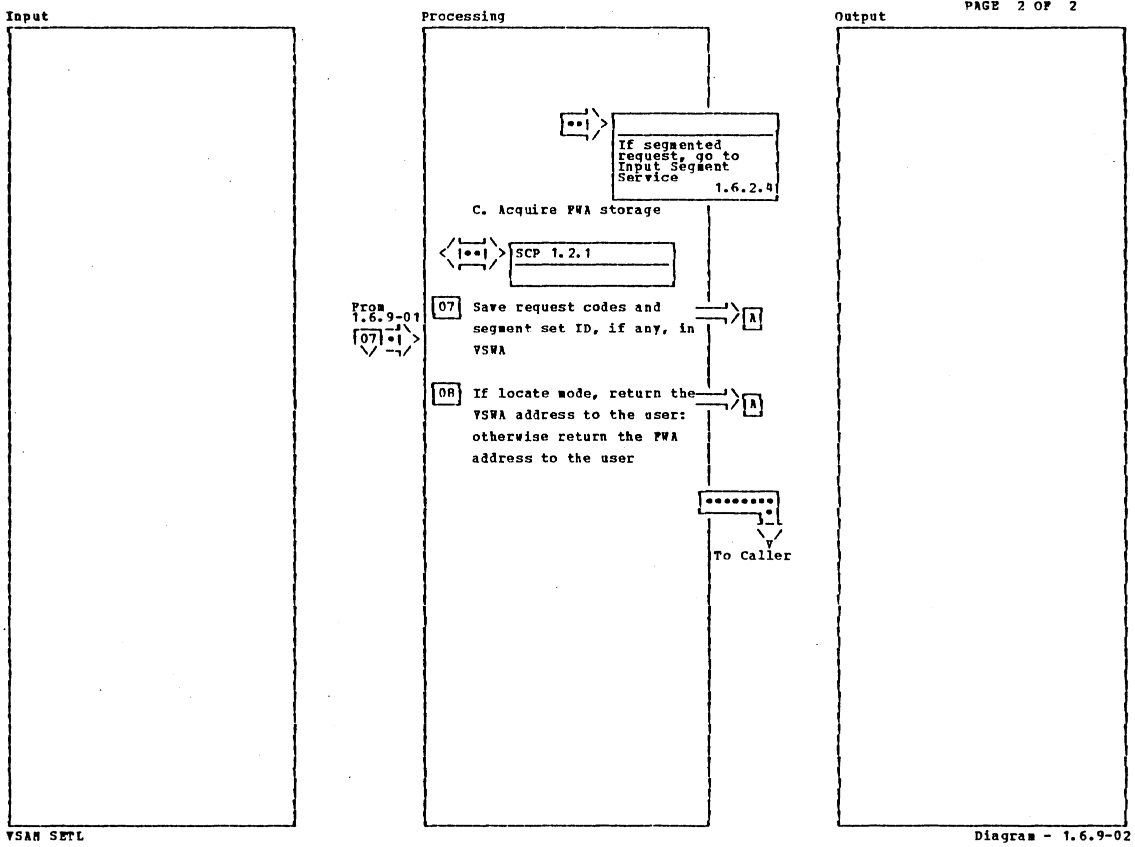
Diagram - 1.6.9-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

VSAM SETL

Diagram - 1.6.9-01

CHART: DFHFCP
PAGE 2 OF 2



If segmented request, go to Input Segment Service 1.6.2.4

C. Acquire FWA storage

07 Save request codes and segment set ID, if any, in VSWA

08 If locate mode, return the VSWA address to the user: otherwise return the FWA address to the user

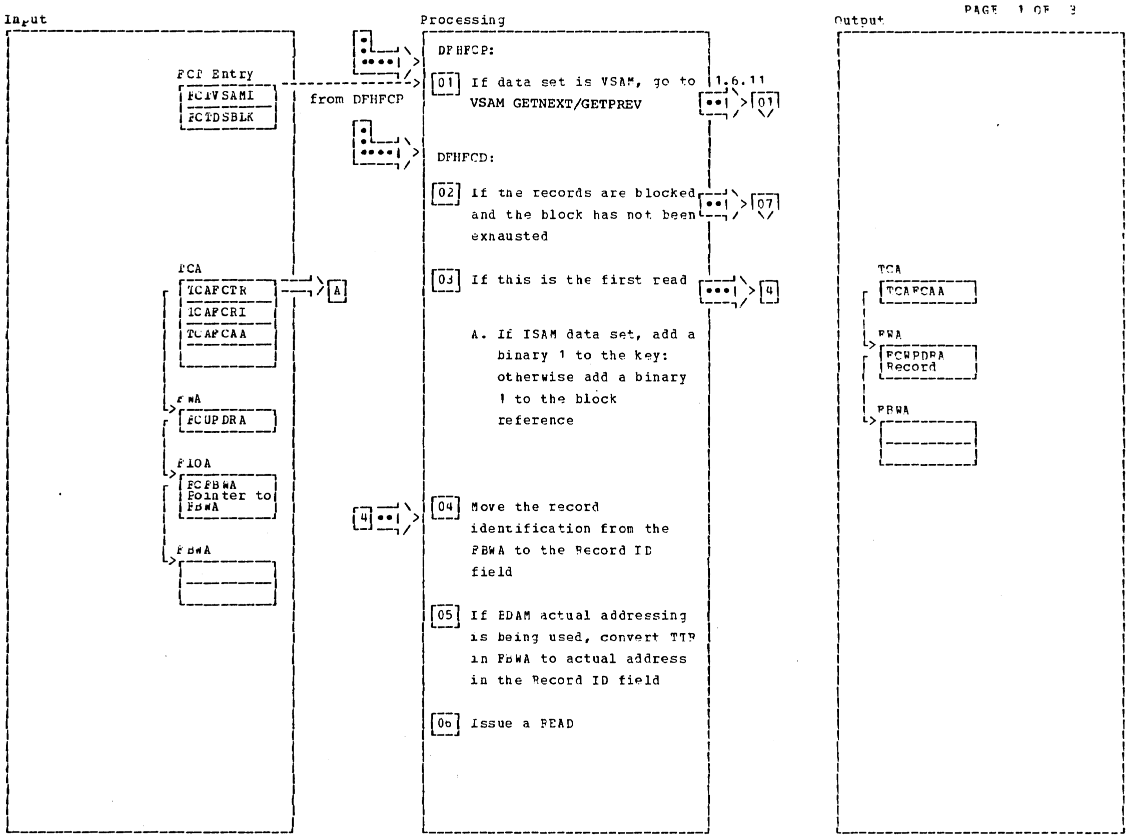
VSAM SETL

Diagram - 1.6.9-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

VSAM SETL

Diagram - 1.6.9-02



DFHFCP:

01 If data set is VSAM, go to VSAM GETNEXT/GETPREV

DFHFCD:

02 If the records are blocked and the block has not been exhausted

03 If this is the first read

A. If ISAM data set, add a binary 1 to the key: otherwise add a binary 1 to the block reference

04 Move the record identification from the FBWA to the Record ID field

05 If BDAM actual addressing is being used, convert TTR in FBWA to actual address in the Record ID field

06 Issue a READ

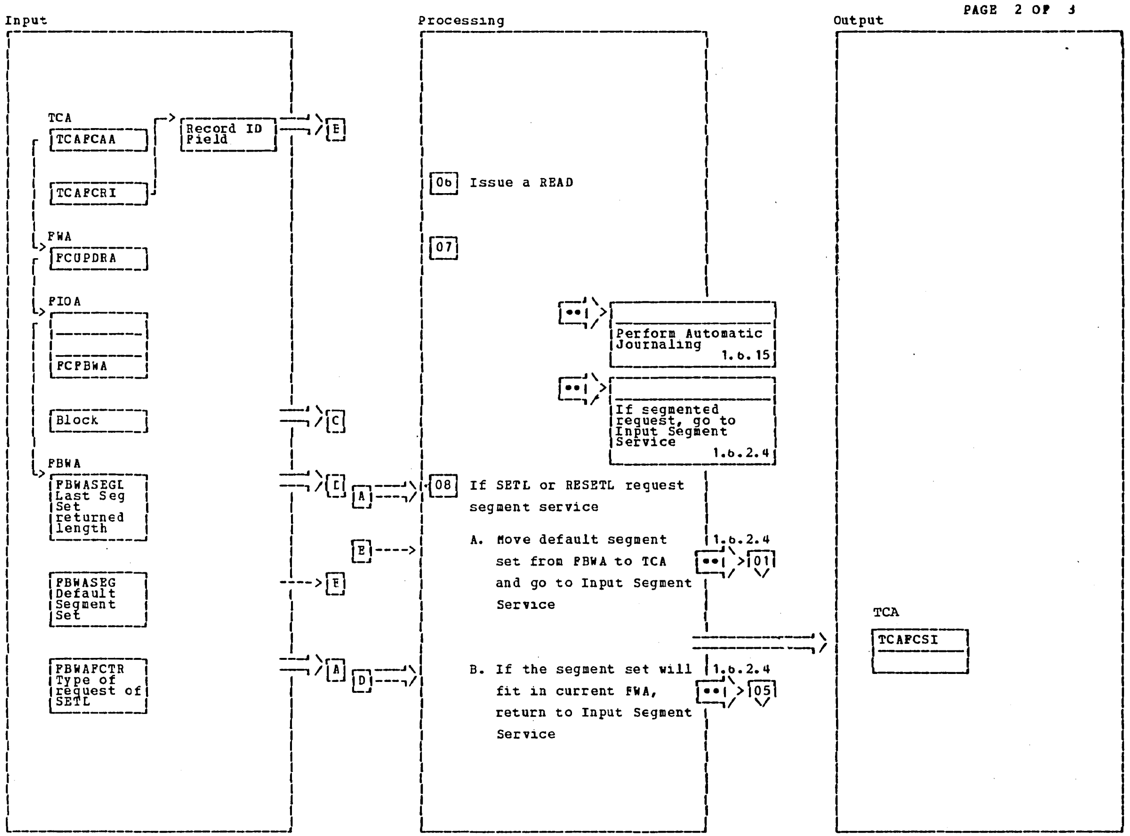
GETNEXT/GETPREV

Diagram - 1.6.10-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 This is done using BDAM's address conversion routine. | | | | | | | |
| 06 No record found conditions are intercepted. For BDAM, the TT of the TTR is incremented by a binary 1 and the R is set to 1. For ISAM, data sets are always defined as blocked when they are to be browsed so that data is transferred into storage even on a no record found. | | | | | | | |

GETNEXT/GETPREV

Diagram - 1.6.10-01

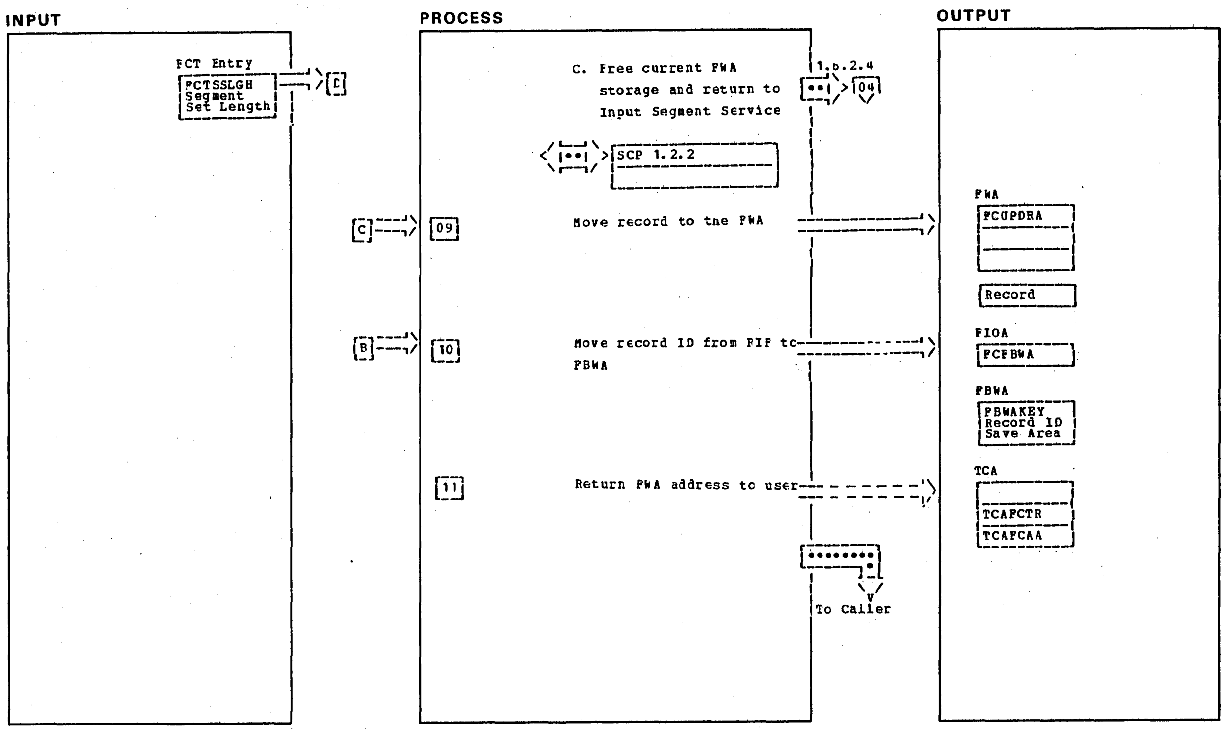

PAGE 2 OF 3
Input
Processing
Output
TCA
TCAFCAA
TCAFCRI
Record ID Field
E
FWA
FCUPDRA
FIOA
FCFBWA
Block
C
FBWA
FBWASEGL Last Seg Set returned length
C
A
FBWASEG Default Segment Set
E
E
FBWAFCTR Type of request of SETL
A
D
06 Issue a READ
07
Perform Automatic Journaling 1.6.15
If segmented request, go to Input Segment Service 1.6.2.4
08 If SETL or RESETL request segment service
A. Move default segment set from FBWA to TCA and go to Input Segment Service
1.6.2.4
01
B. If the segment set will fit in current FWA, return to Input Segment Service
1.6.2.4
05
TCA
TCAFCSI

GETNEXT/GETPREV Diagram - 1.6.10-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 06 No record found conditions are intercepted. For DAM, the TT of the TTR is incremented by a binary 1 and the R is set to 1. For ISAM, data sets are always defined as blocked when they are to be browsed so that data is transferred into storage even cn a no record found. | | | | | | | |

GETNEXT/GETPREV Diagram - 1.6.10-02

PAGE 3 OF 3



C. Free current FWA storage and return to Input Segment Service

09 Move record to the FWA

10 Move record ID from FIF to FBWA

11 Return FWA address to user

GETNEXT/GETPREV

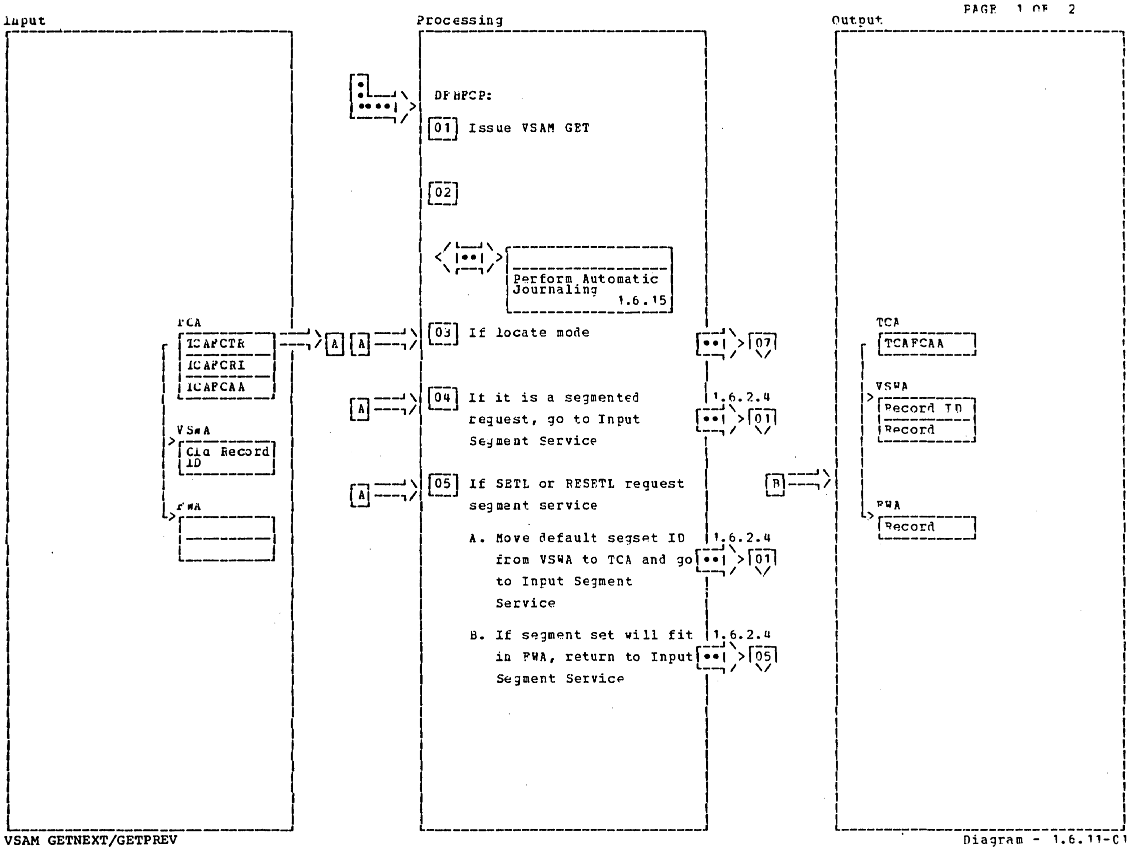
Diagram - 1.6.10-03

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

GETNEXT/GETPREV

Diagram - 1.6.10-03


PAGE 1 OF 2
Input
Processing
Output
DFHFCP:
01 Issue VSAM GET
02
Perform Automatic Journaling 1.6.15
FCA
TCAFCTR
TCAFCRI
TCAFCAA
A
03 If locate mode
07
04 If it is a segmented request, go to Input Segment Service
1.6.2.4
01
VSWA
Cla Record ID
05 If SETL or RESETL request segment service
A. Move default segset ID from VSWA to TCA and go to Input Segment Service
1.6.2.4
01
B. If segment set will fit in FWA, return to Input Segment Service
1.6.2.4
05
B
FWA
TCA
TCAFCAA
VSWA
Record ID
Record
FWA
Record

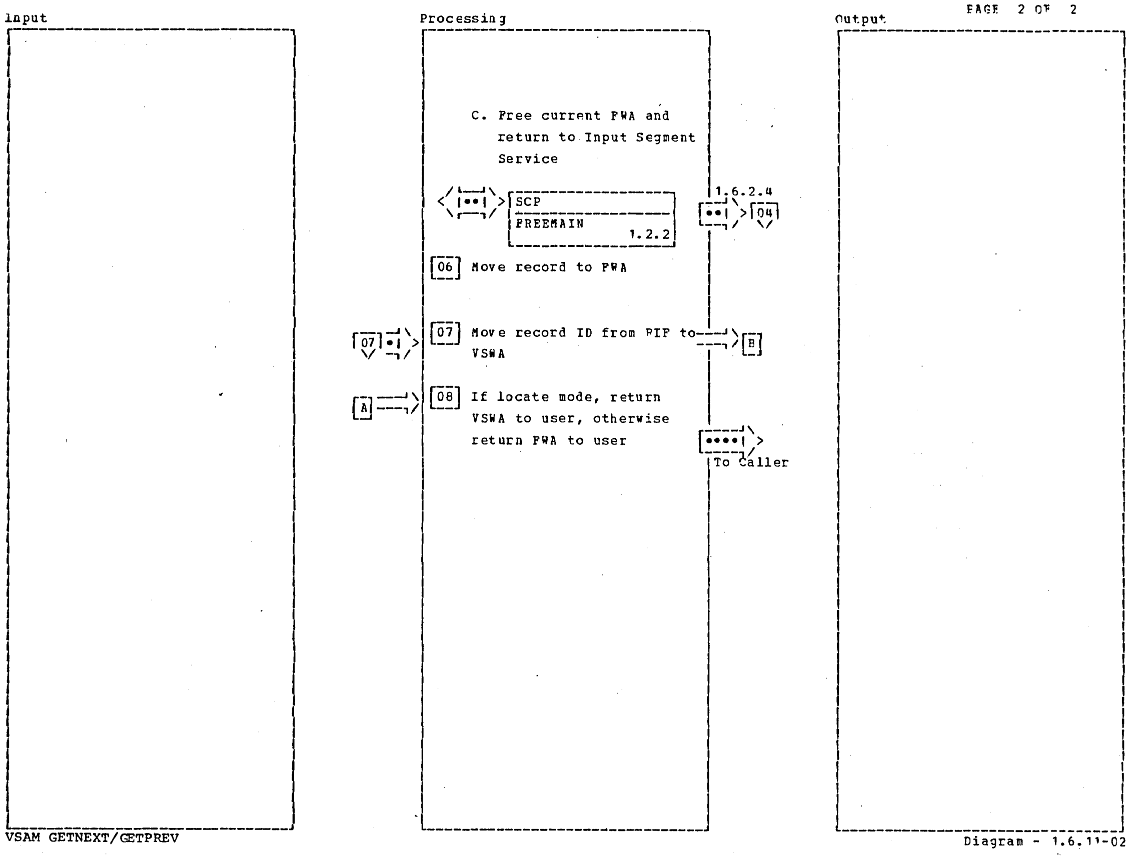
VSAM GETNEXT/GETPREV

Diagram - 1.6.11-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

VSAM GETNEXT/GETPREV

Diagram - 1.6.11-01



C. Free current PWA and return to Input Segment Service

06 Move record to PWA

07 Move record ID from PIF to VSWA

08 If locate mode, return VSWA to user, otherwise return PWA to user

VSAM GETNEXT/GETPREV

Diagram - 1.6.11-02

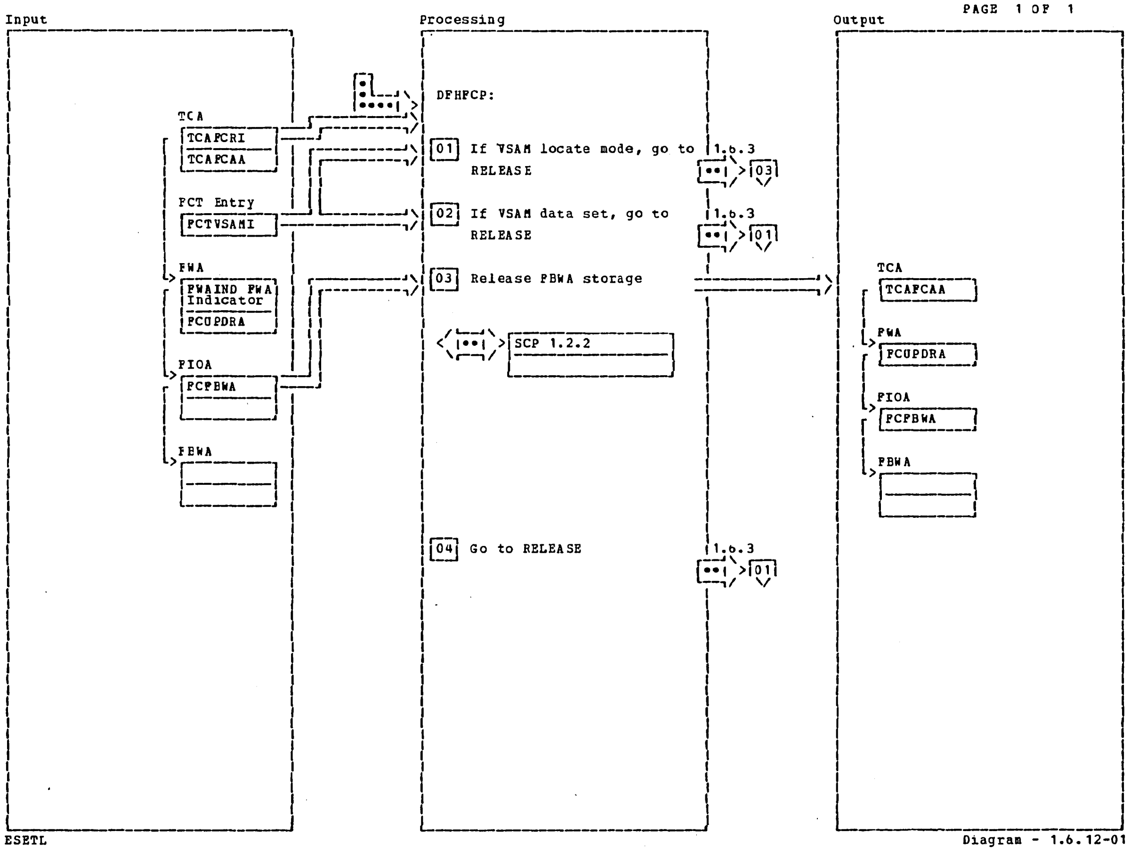
| Notes | Routine | Label | Ref |
|---|---|---|---|

| Notes | Routine | Label | Ref |
|---|---|---|---|

VSAM GETNEXT/GETPREV

Diagram - 1.6.11-02

PAGE 1 OF 1



01 If VSAM locate mode, go to RELEASE

02 If VSAM data set, go to RELEASE

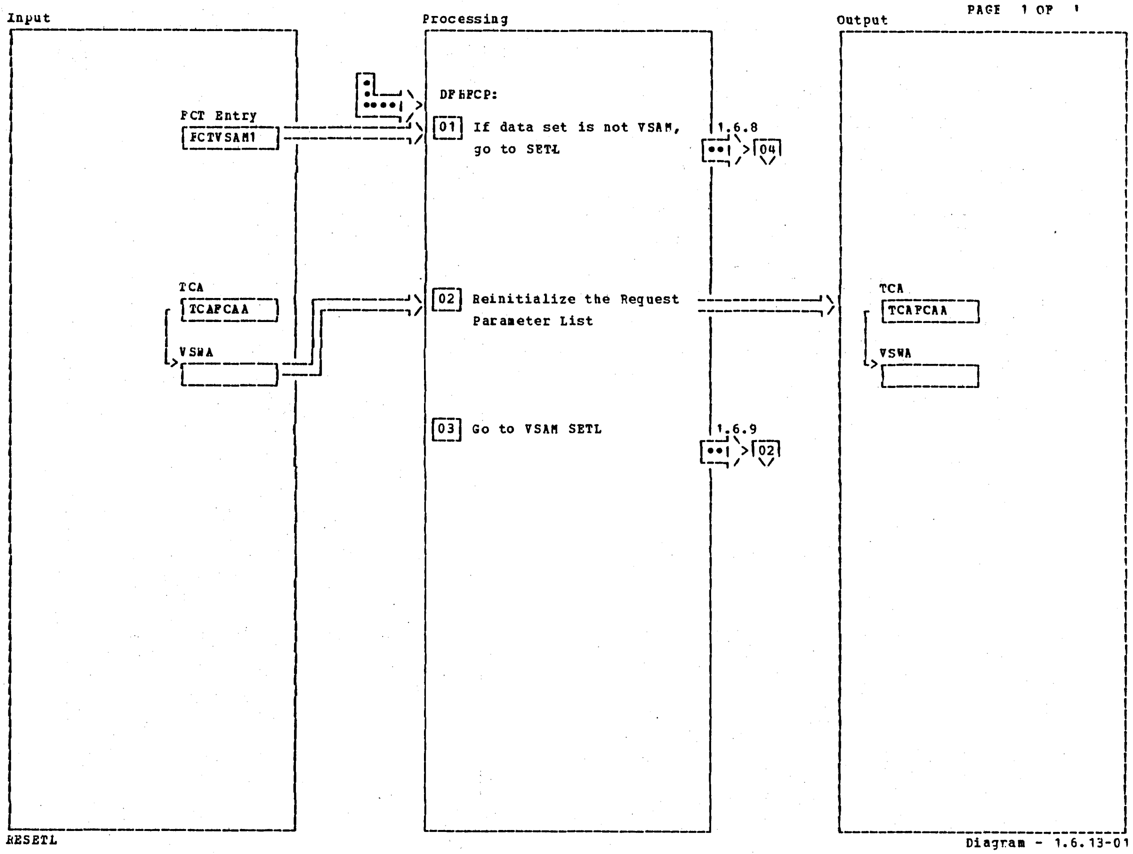
03 Release FBWA storage

04 Go to RELEASE

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

ESETL

Diagram - 1.6.12-01



PAGE 1 OF 1

Input

FCT Entry

FCTVSAM1

TCA

TCAFCAA

VSWA

Processing

DFHFCP:

01 If data set is not VSAM, go to SETL

1.6.8 04

02 Reinitialize the Request Parameter List

03 Go to VSAM SETL

1.6.9 02

Output

TCA

TCAFCAA

VSWA

RESETL

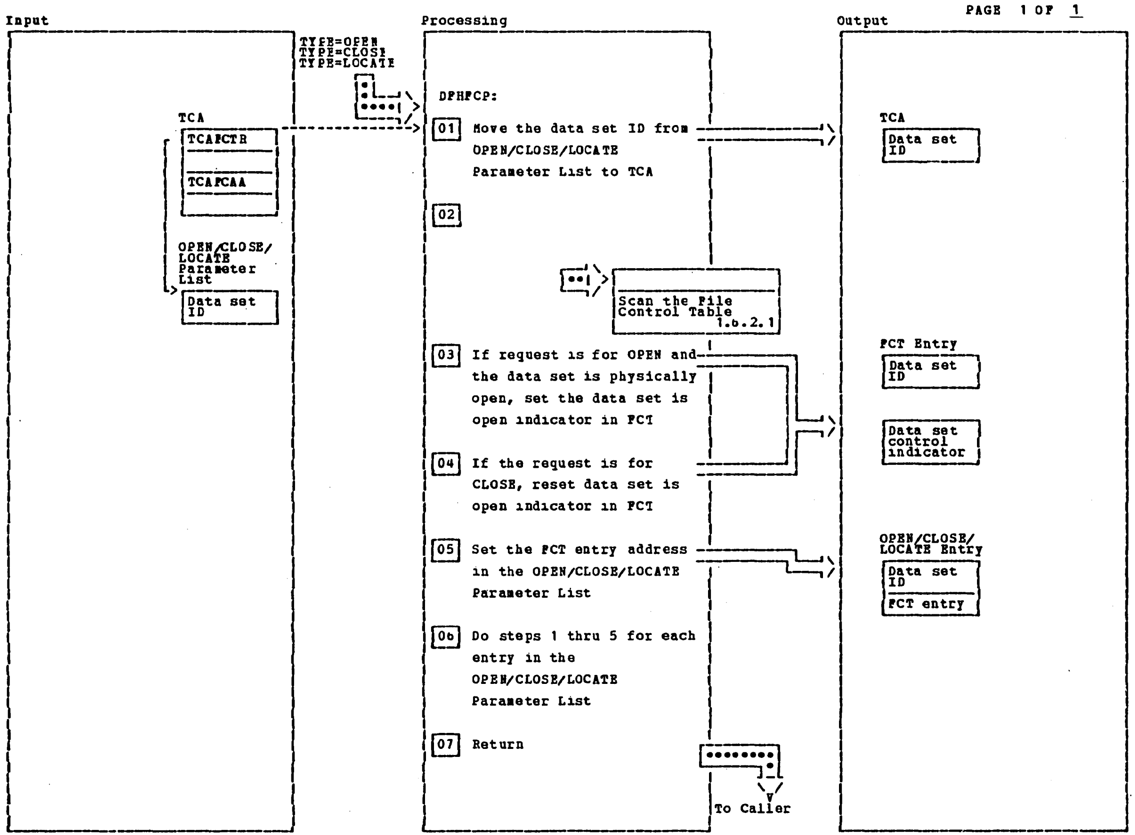
Diagram - 1.6.13-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

RESETL

Diagram - 1.6.13-01

PAGE 1 OF 1



01 Move the data set ID from OPEN/CLOSE/LOCATE Parameter List to TCA

02

03 If request is for OPEN and the data set is physically open, set the data set is open indicator in FCT

04 If the request is for CLOSE, reset data set is open indicator in FCT

05 Set the FCT entry address in the OPEN/CLOSE/LOCATE Parameter List

06 Do steps 1 thru 5 for each entry in the OPEN/CLOSE/LOCATE Parameter List

07 Return

OPEN/CLOSE/LOCATE

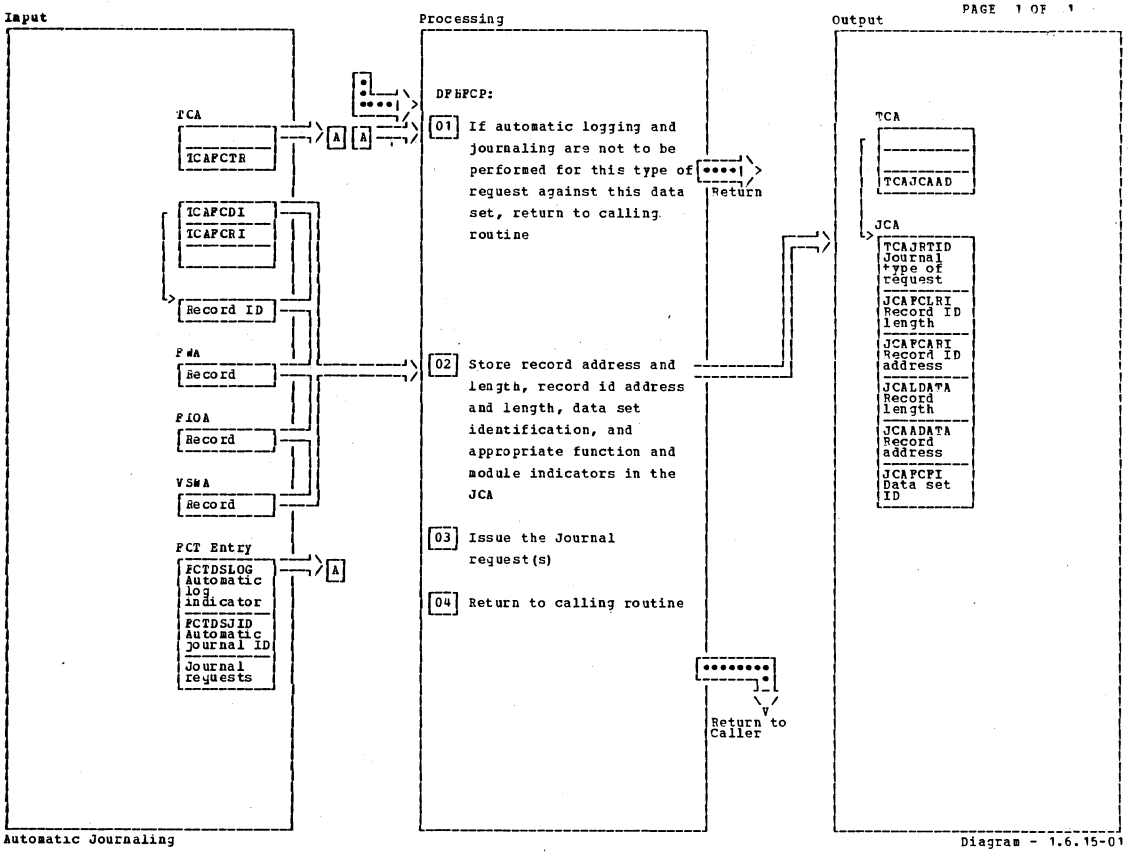
Diagram - 1.6.14-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The TYPE=LOCATE is used to find the address of a specific File Control Table entry. | | | | | | | |
| 02 If data set identification cannot be found in the FCT, the FCT entry address is set to zero at step 5. | | | | | | | |

OPEN/CLOSE/LOCATE

Diagram - 1.6.14-01

PAGE 1 OF 1



DFHFCP:

01 If automatic logging and journaling are not to be performed for this type of request against this data set, return to calling routine

02 Store record address and length, record id address and length, data set identification, and appropriate function and module indicators in the JCA

03 Issue the Journal request(s)

04 Return to calling routine

Automatic Journaling

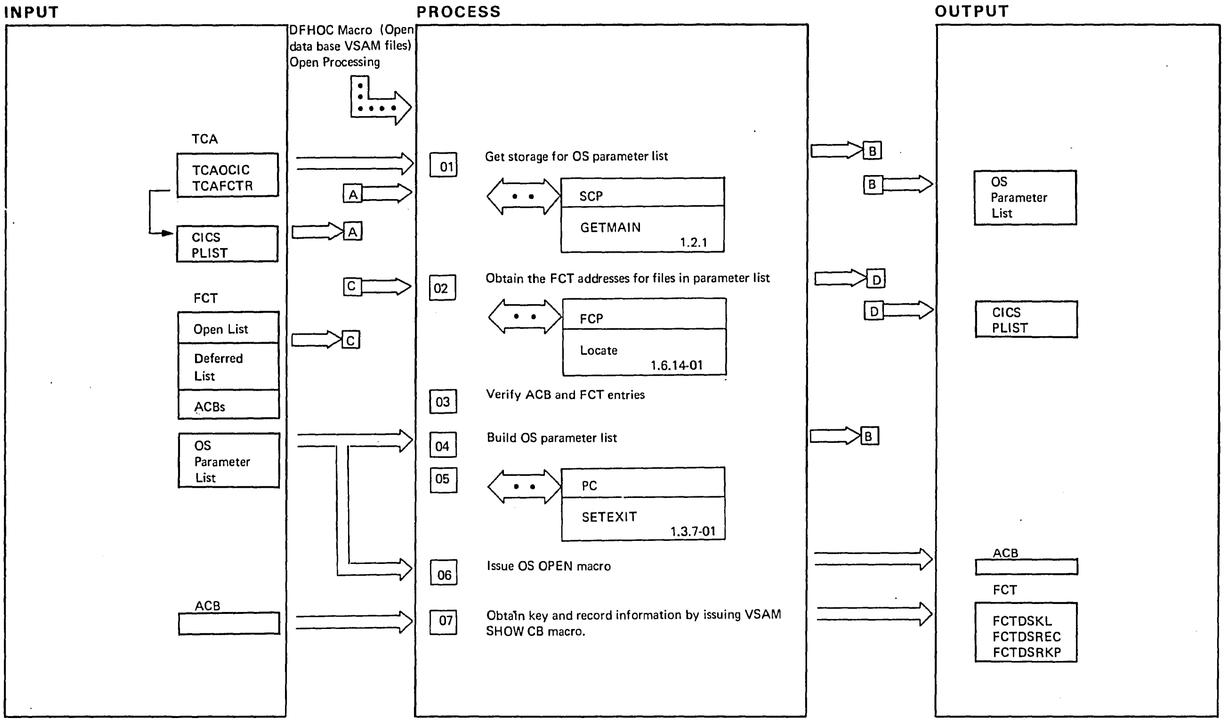
Diagram - 1.6.15-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Reads for update are automatically logged asynchronously, and synchronized on the writes for update. Write-adds are automatically logged syncnronously. What is automatically journaled is entirely up to the user. |  |  |  |  |  |  |  |
| 02 A DFHJC TYPE=GETJCA macro instruction is always issued to ensure tnere is a JCA for this task. |  |  |  |  |  |  |  |

Automatic Journaling

Diagram - 1.6.15-01

Page 1 of 2



VSAM Open Processing

Diagram — 1.6.16 - 01

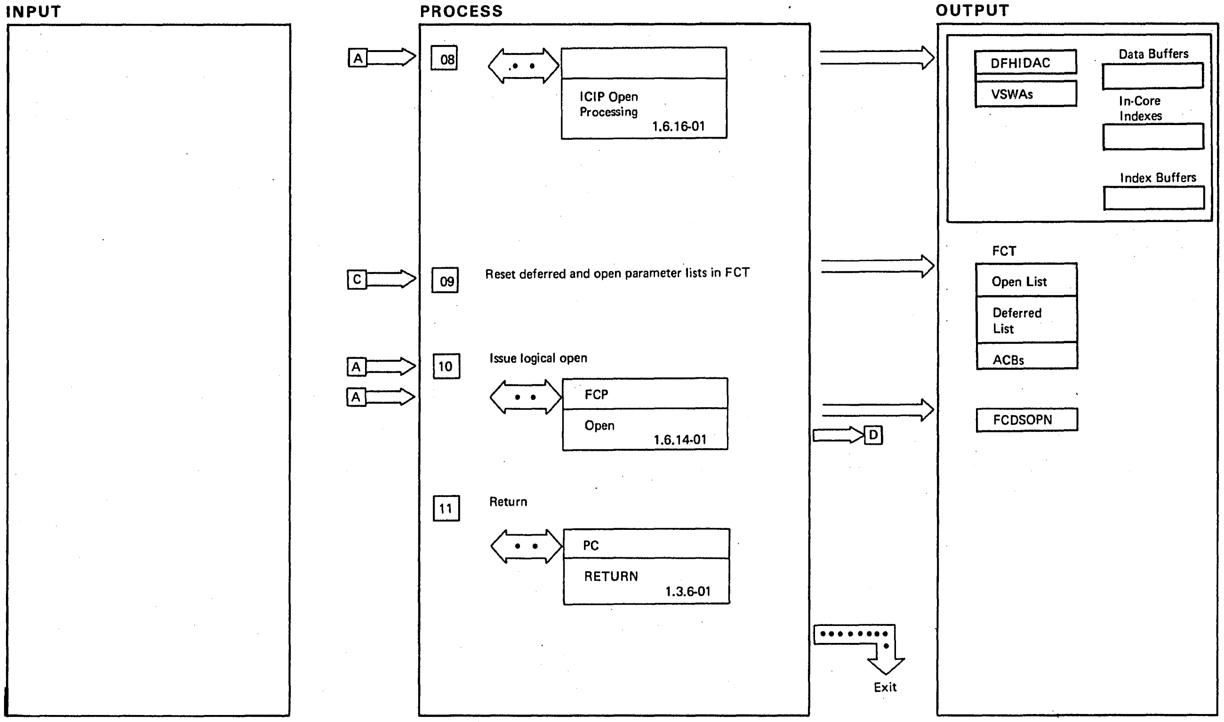
| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 06 | If the ICIP option (& VSICIP) has been sysgened then the OS OPEN macro is issued in authorized mode using the CICS SVC (DFHCSVC). | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

VSAM Open Processing

Diagram — 1.6.16 - 01

Page 2 of 2



VSAM Open Processing

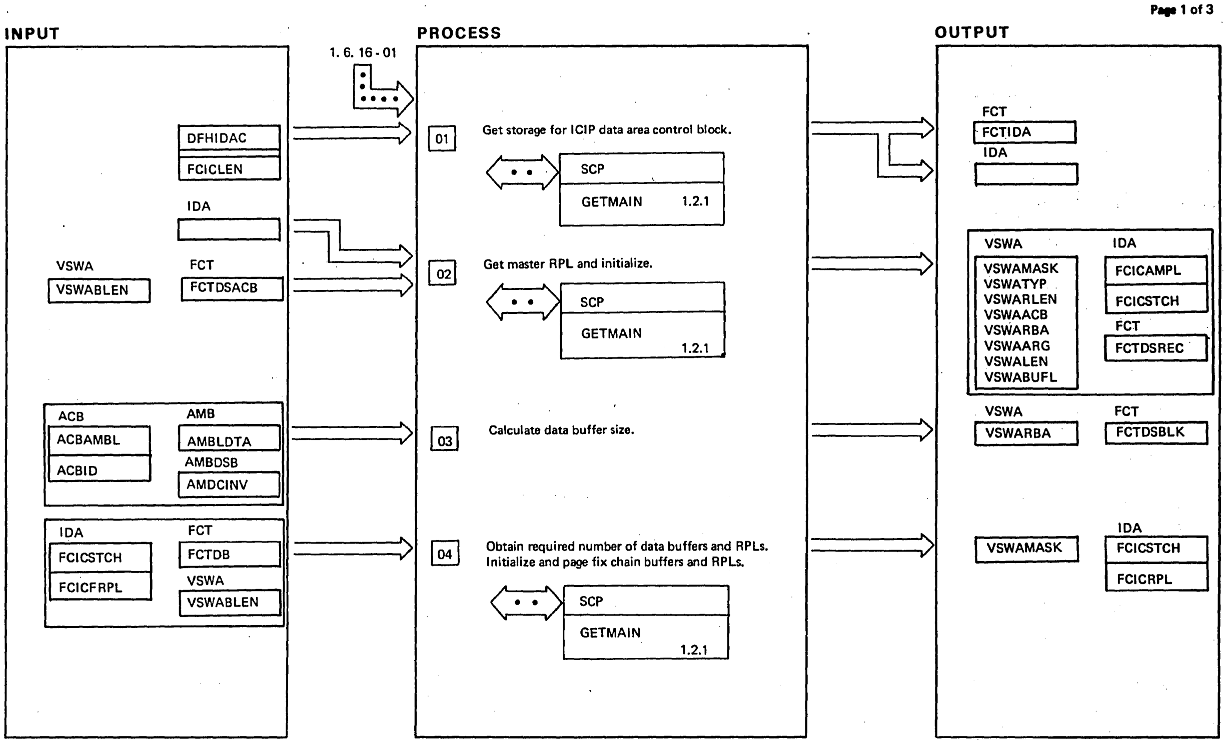
Diagram – 1.6.16-02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

VSAM Open Processing

Diagram – 1.6.16-02



ICIP Open Processing

Diagram - 1. 6. 16. 1-01

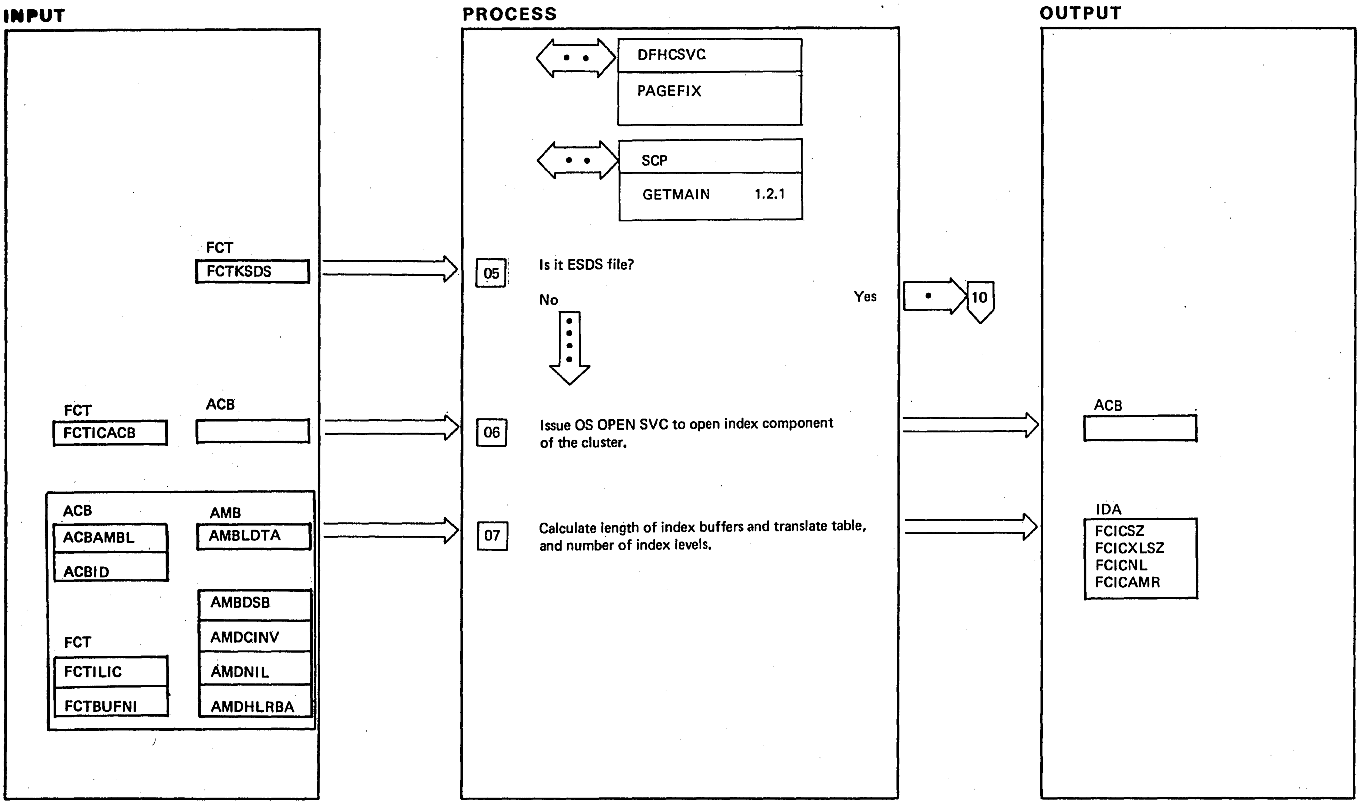
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 02 03 08 09<br>Should there be insufficient storage for the CICS GETMAIN or should the page-fix fail, then the following messages will be output.<br>DFHI521 - Insufficient file storage for an ICIP file.<br>DFHI522 - Pagefix of buffers for an ICIP file has failed.<br>The files will be closed and any storage already obtained will be released. | | | |
| 02 04 08<br>This storage is all chained using the third word of the storage obtained (the first two being used by CICS storage management). | | | |
| 04 The data buffer is chained from the VSWA using the first word of the VSWA. i.e. 8 bytes before the start of the RPL. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

ICIP Open Processing

Diagram - 1.6.16.1-01

Page 2 of 3



ICIP Open Processing

Diagram - 1.6.16.1-02

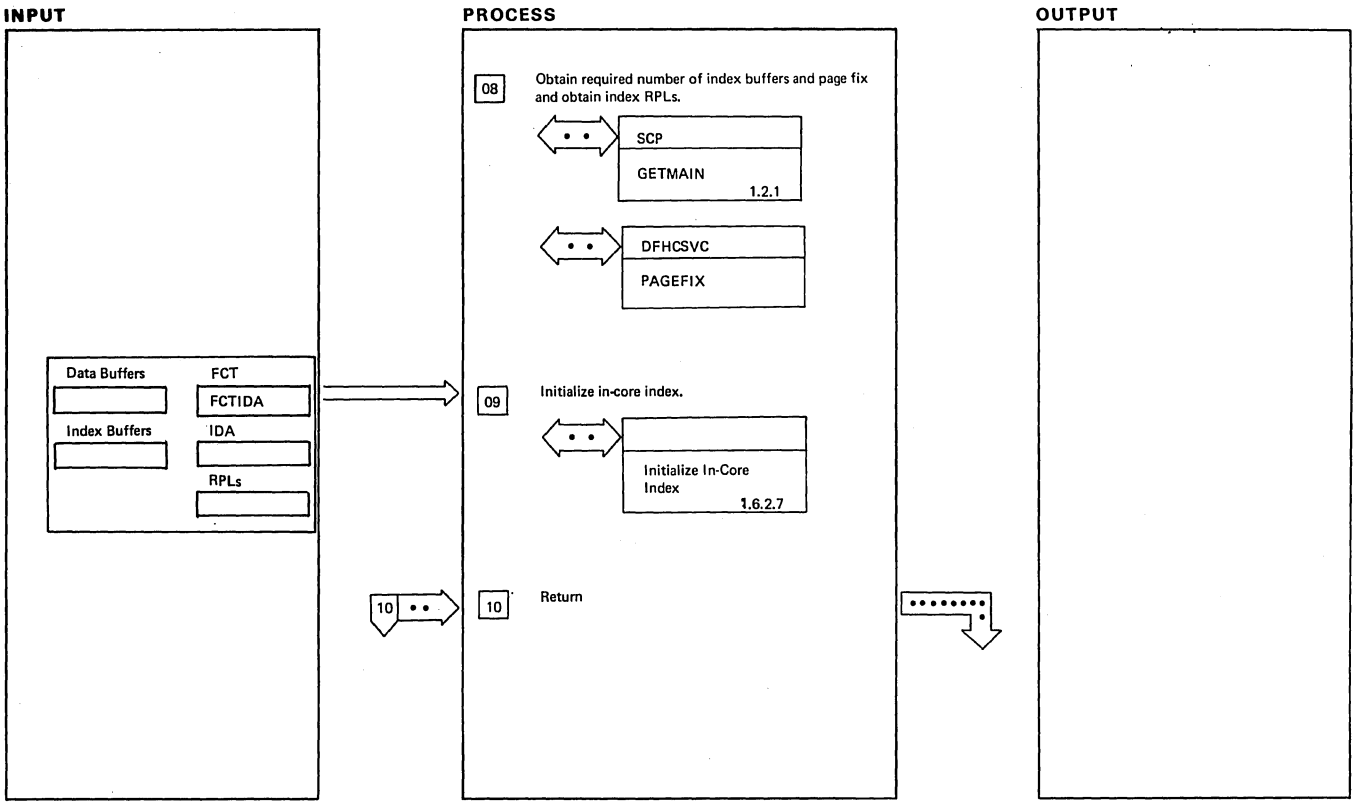
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 06 The OS open macro is issued in authorized mode using CICS SVC (DFHCSVC). | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

ICIP Open Processing

Diagram - 1.6.16.1-02

Page 3 of 3



08 Obtain required number of index buffers and page fix and obtain index RPLs.

09 Initialize in-core index.

10 Return

ICIP Open Processing

Diagram - 1. 6. 16. 1-03

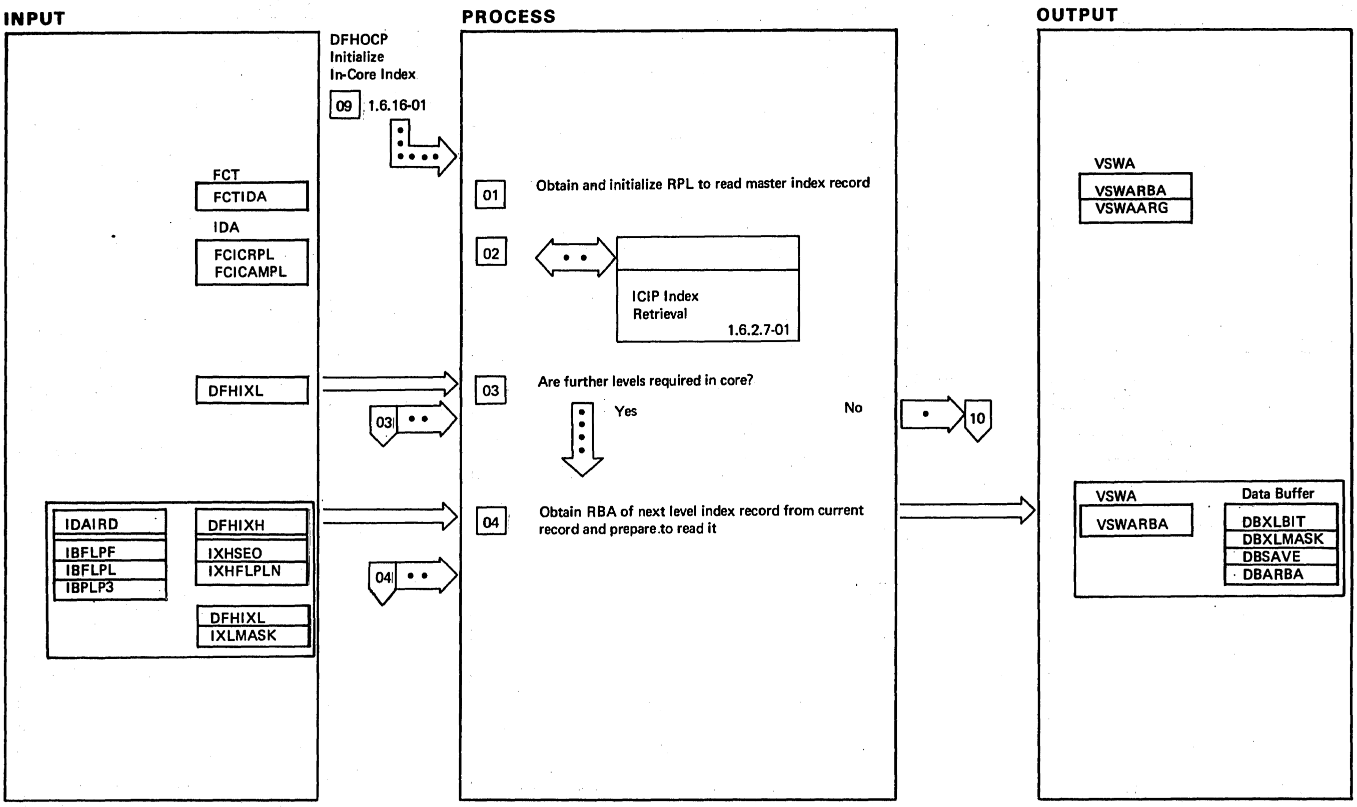
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 08 See notes on page - 01. | | | |
| 09 See notes on page - 01. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

ICIP Open Processing

Diagram - 1. 6. 16. 1-03

Page 1 of 3



01 Obtain and initialize RPL to read master index record

02

03 Are further levels required in core?

04 Obtain RBA of next level index record from current record and prepare to read it

Initialize In-Core Index

Diagram - 1.6.16.2-01

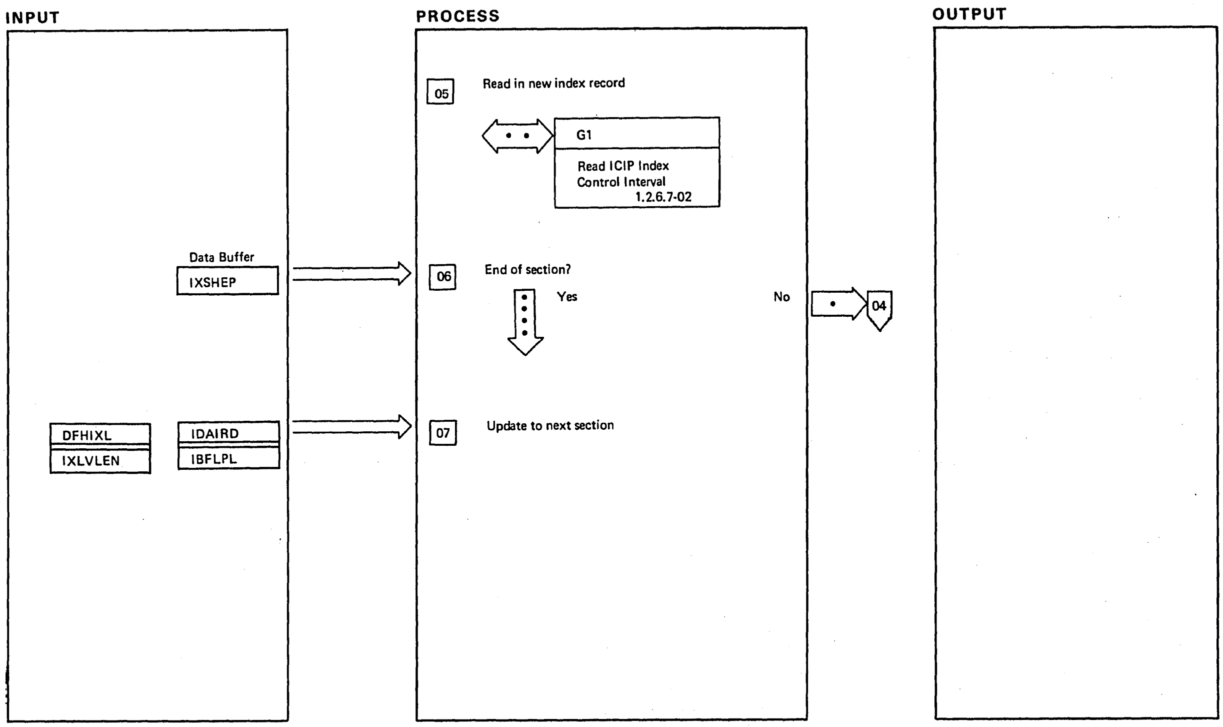
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 An RPL is obtained, but not dechained from the free RPL queue in IDA | | | |
| 02 The code is that used to initialize the in-core index dynamically. | | | |
| 03 04 If 1 level of in-core index is specified, the master index record is read in. If further levels are required in core at open time then there will be a translate table associated with the master record and the RBA of each index entry in the master record will be used to read in the next level of index.<br>Subsequent required levels are built up in a similar manner | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Initialize In-Core Index

Diagram - 1.6.16.2-01

Page 2 of 3



Initiallze In-Core Index

Diagram - 1. 6. 16. 2-02

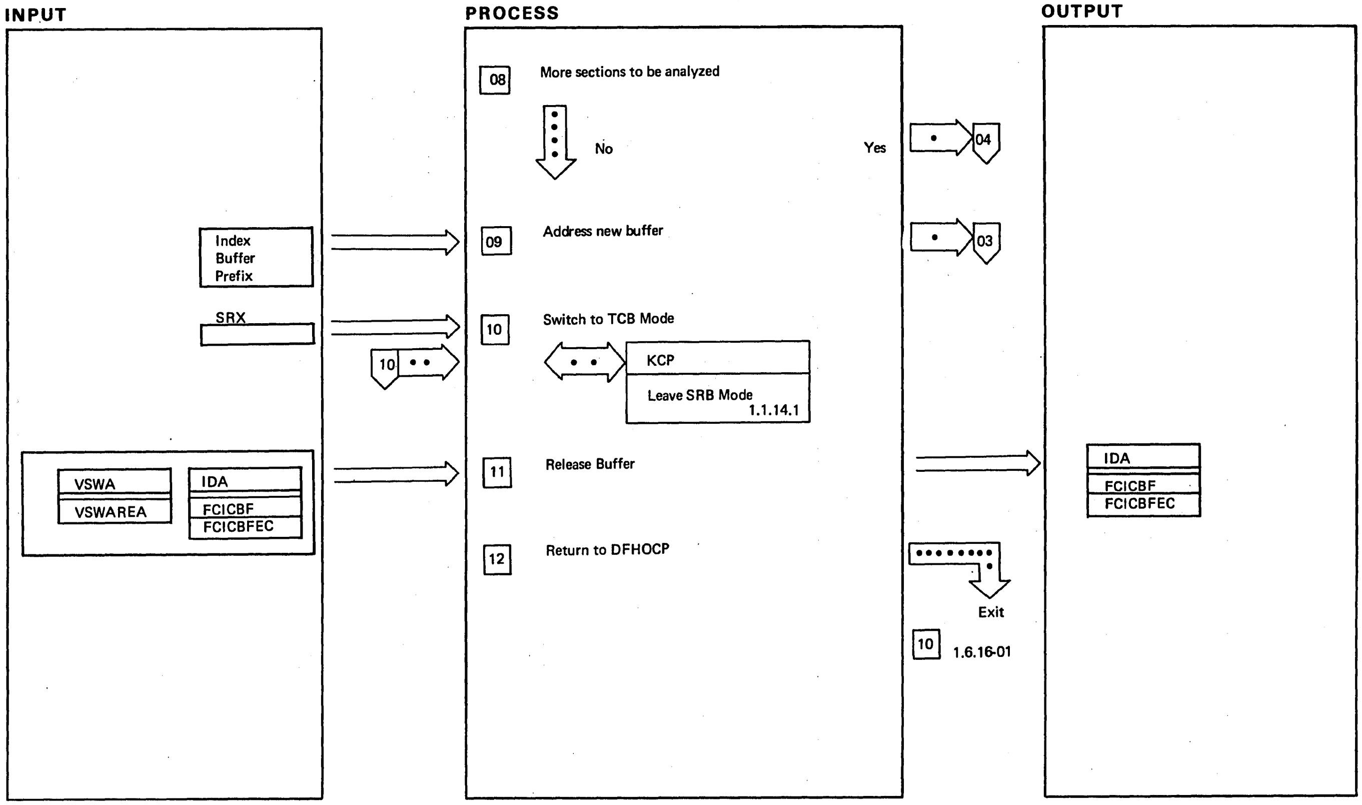
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 06 07<br>Each VSAM index CI is split into seperate sections | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Initialize In-Core Index

Diagram - 1. 6. 16. 2-02

Page 3 of 3



Initialize In-Core Index

Diagram – 1.6.16.2 – 03

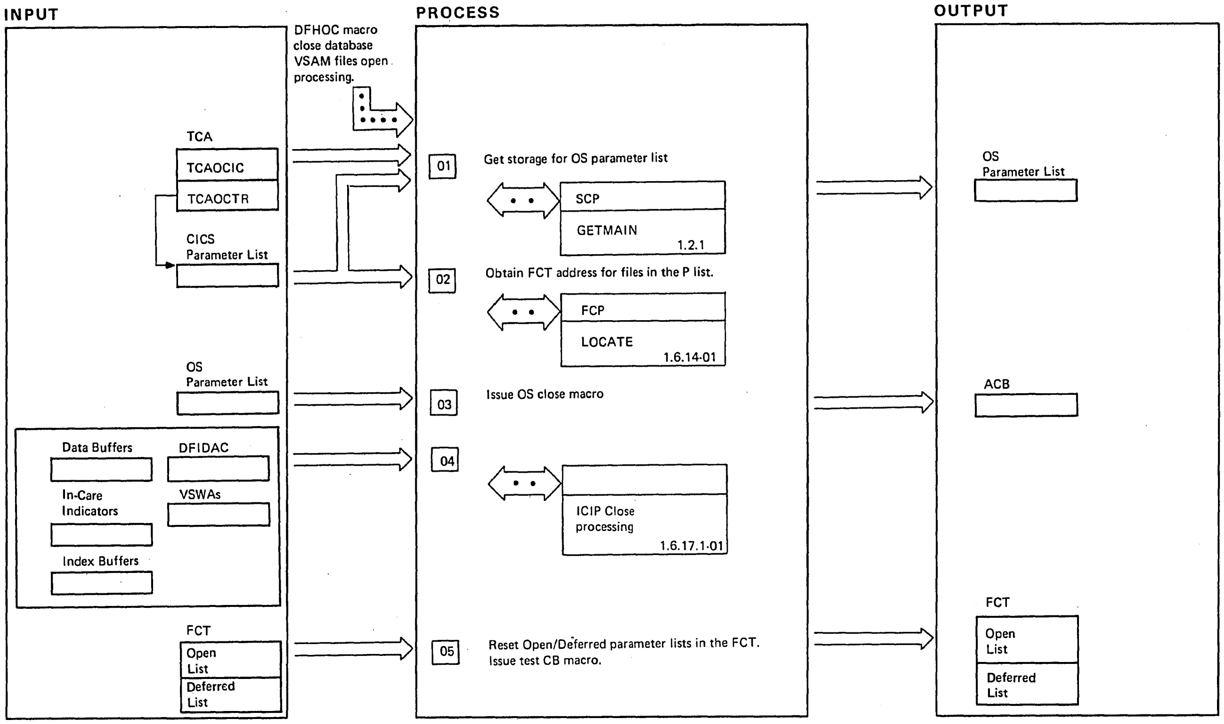
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Initialize In-Core Index

Diagram – 1.6.16.2 – 03

Page 1 of 2



VSAM Close Processing

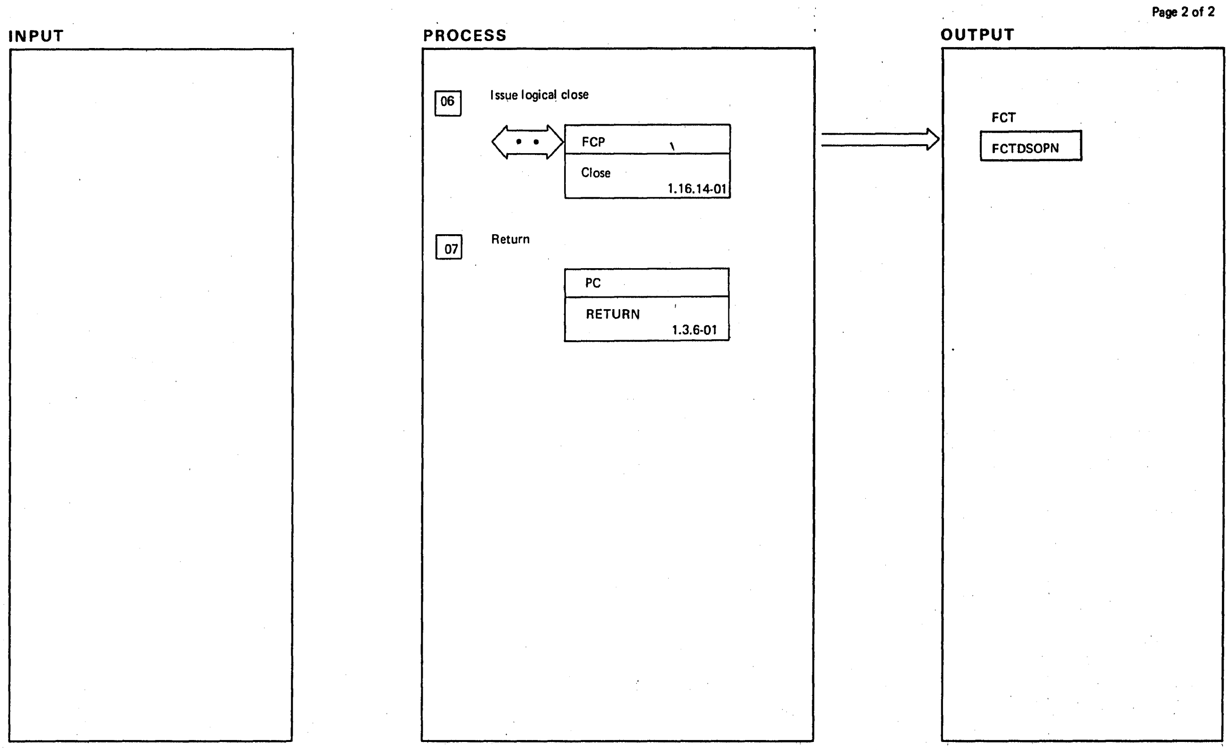
Diagram - 1.6.17 - 01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 03 If the ICIP sysgen option (& VSICIP) has been used there the OS close macro is issued in authorized mode using the CICS SVC. (DFHCSVC) | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

VSAM Close Processing

Diagram - 1.6.17 - 01



VSAM Close Processing

Diagram - 1.6.17 - 02

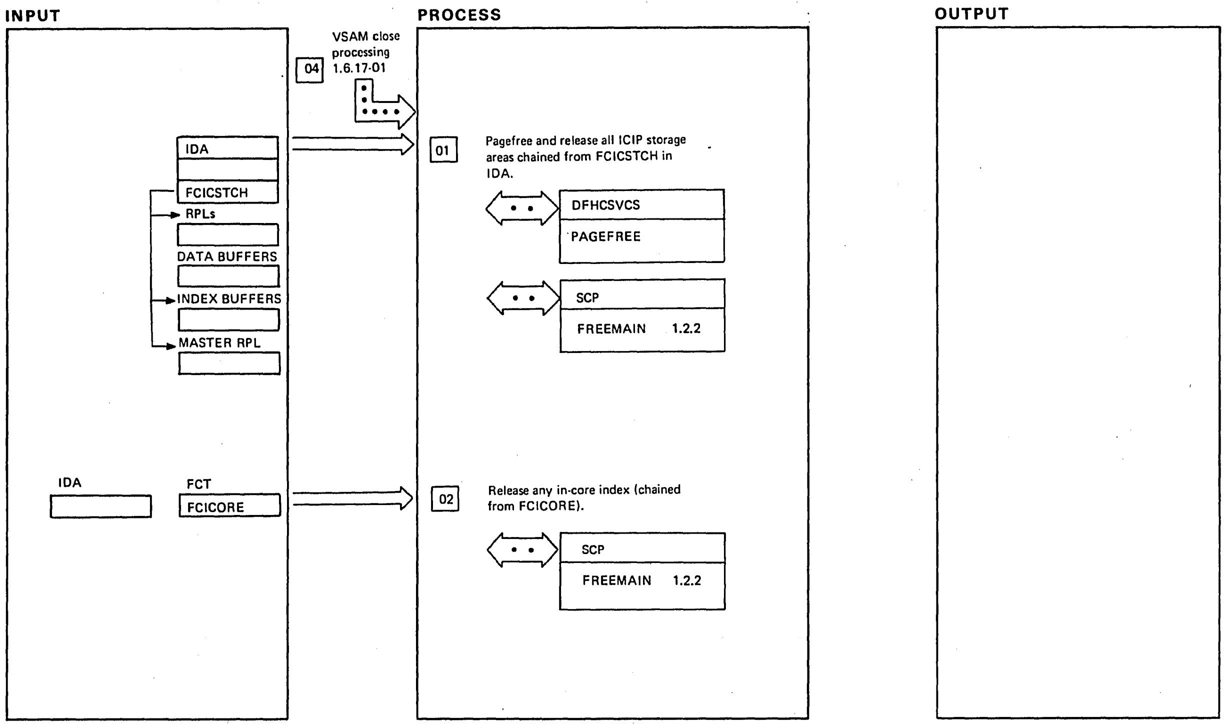
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

VSAM Close Processing

Diagram - 1.6.17 - 02

Page 1 of 2


INPUT
PROCESS
OUTPUT
04
VSAM close processing 1.6.17-01
IDA
FCICSTCH
RPLs
DATA BUFFERS
INDEX BUFFERS
MASTER RPL
01
Pagefree and release all ICIP storage areas chained from FCICSTCH in IDA.
DFHCSVCS
PAGEFREE
SCP
FREEMAIN 1.2.2
IDA
FCT
FCICORE
02
Release any in-core index (chained from FCICORE).
SCP
FREEMAIN 1.2.2

ICIP Close Processing

Diagram - 1.6.17.1-01

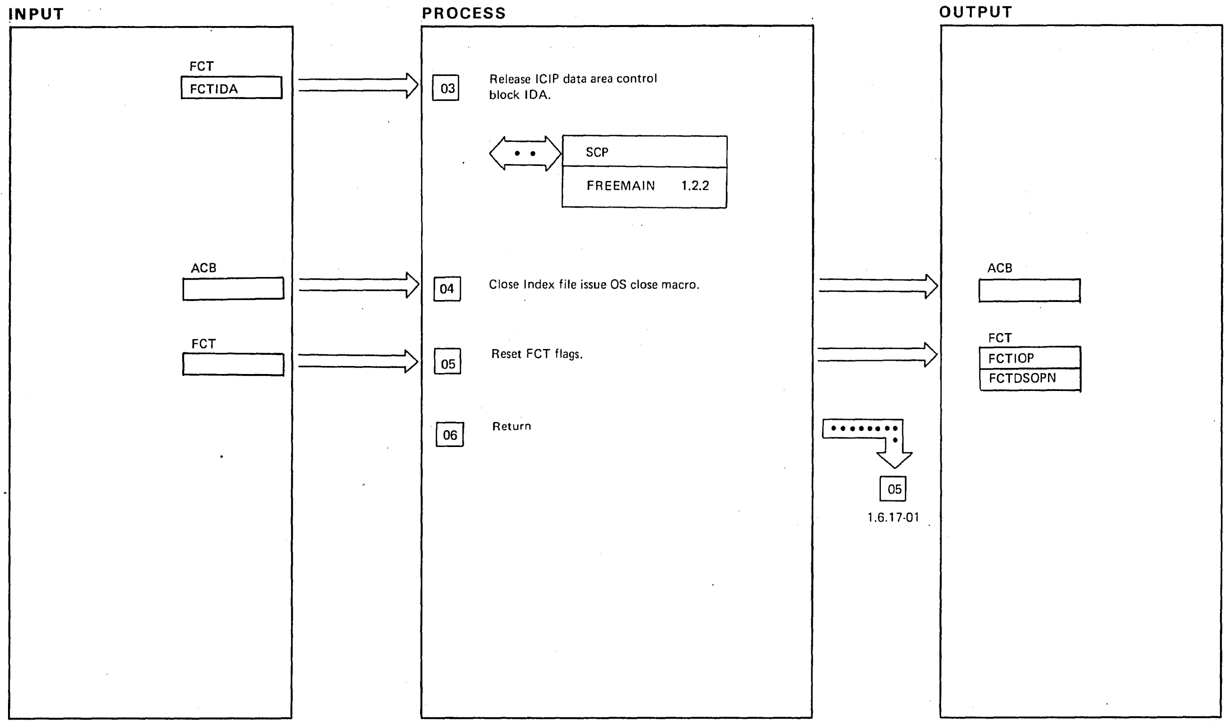
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

ICIP Close Processing

Diagram - 1.6.17.1-01

Page 2 of 2



ICIP Close Processing

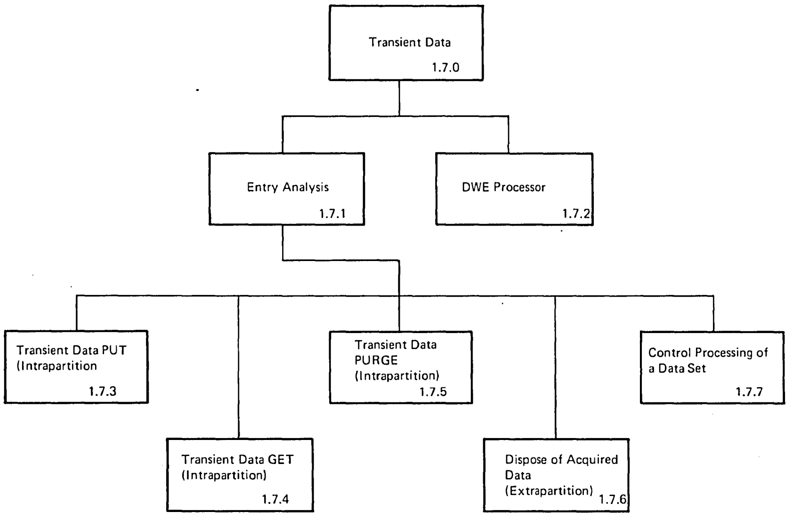
Diagram - 1.6.17.1-02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

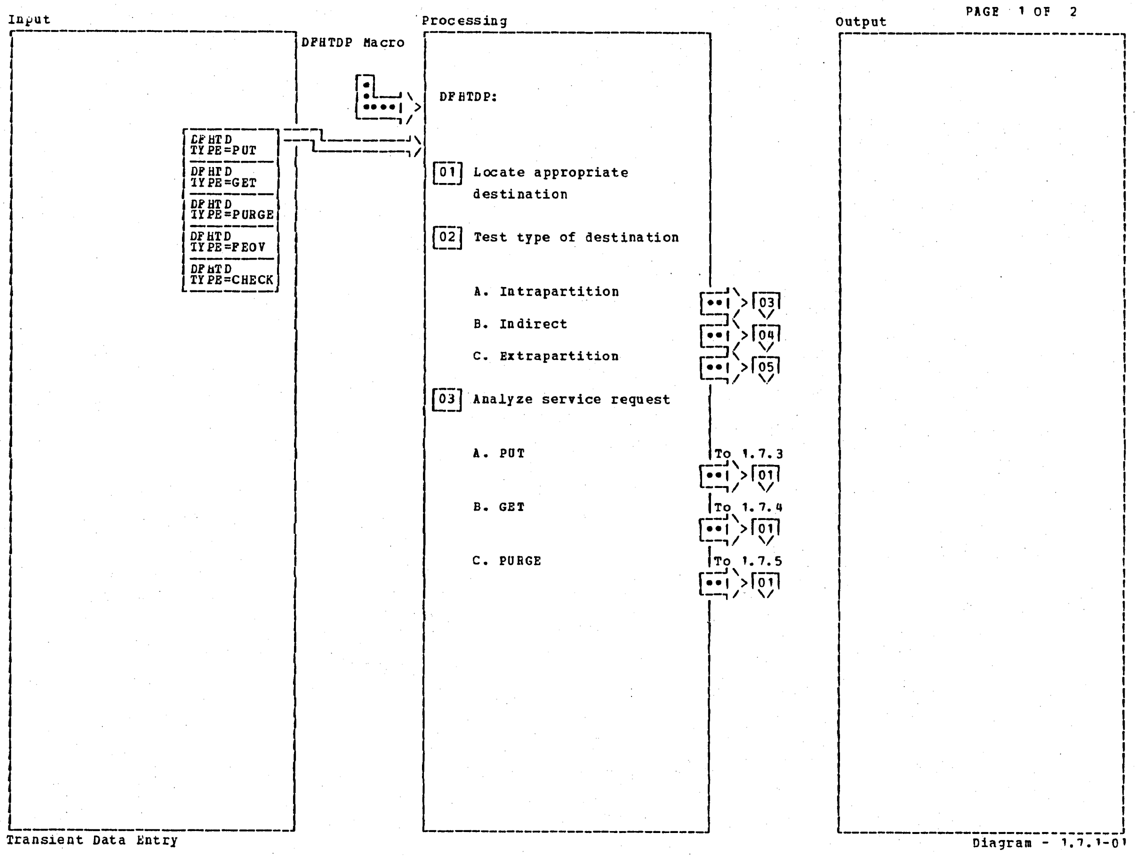
ICIP Close Processing

Diagram - 1.6.17.1-02



Transient Data

Diagram – 1.7.0-01



DFHTDP:

01 Locate appropriate destination

02 Test type of destination

A. Intrapartition

B. Indirect

C. Extrapartition

03 Analyze service request

A. PUT

B. GET

C. PURGE

Transient Data Entry

Diagram - 1.7.1-01

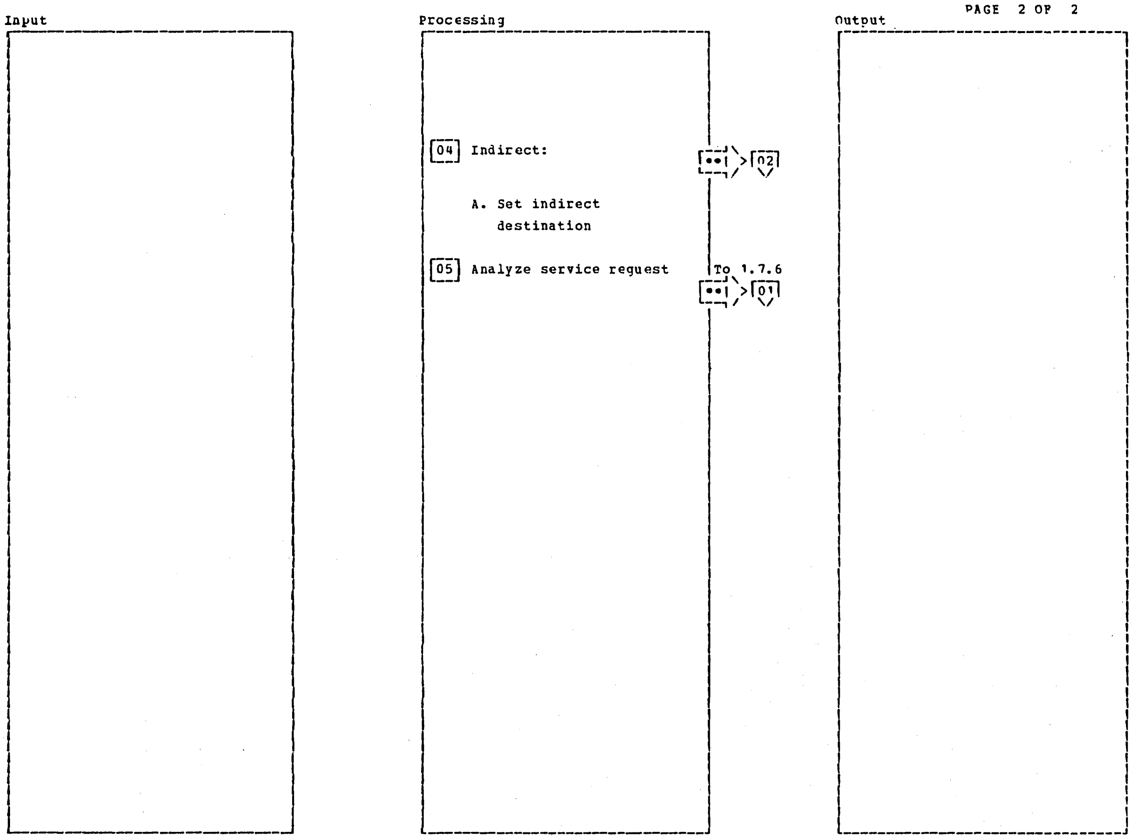
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Check for ID error. If error, issue X'02' return code to caller. | ENTRY ANALYSIS | TDCEA | |
| 02 If not A, B, or C, ABEND=ATDI. | | TDEATR | |
| 03 Intrapartition: Go to proper service routine for: | | TDEATSAN | |
| A. PUT | | TDWRTRTN | |
| B. GET | | TDIARTN | |
| C. PURGE | | TDEAPPRG | |
| The DFHTD CTYPE=LOCATE macro instruction is used to scan the Destination Control Table. | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Transient Data Entry

Diagram - 1.7.1-01

PAGE 2 OF 2



Transient Data Entry

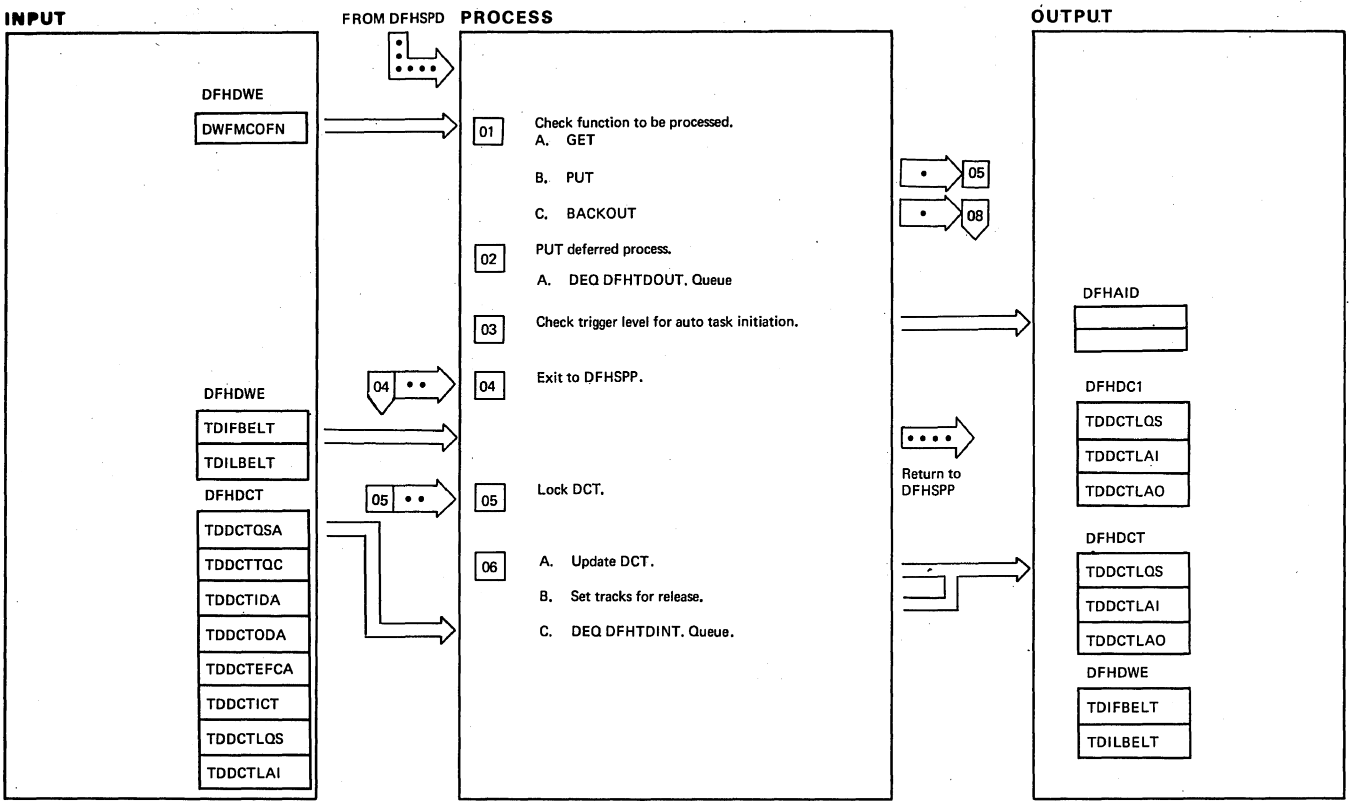
Diagram - 1.7.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 Indirect: An indirect DCT entry contains the address of the appropriate DCT entry | | TDEAIDR | | | | | |
| 05 Extrapartition: check for FEOV request. | extra,<br>par,<br>dest | TDEXTPAR | | | | | |

Transient Data Entry

Diagram - 1.7.1-02

Page 1 of 3



01 Check function to be processed.
A. GET
B. PUT
C. BACKOUT

02 PUT deferred process.
A. DEQ DFHTDOUT. Queue

03 Check trigger level for auto task initiation.

04 Exit to DFHSPP.

05 Lock DCT.

06 A. Update DCT.
B. Set tracks for release.
C. DEQ DFHTDINT. Queue.

Transient Data DWE Processor

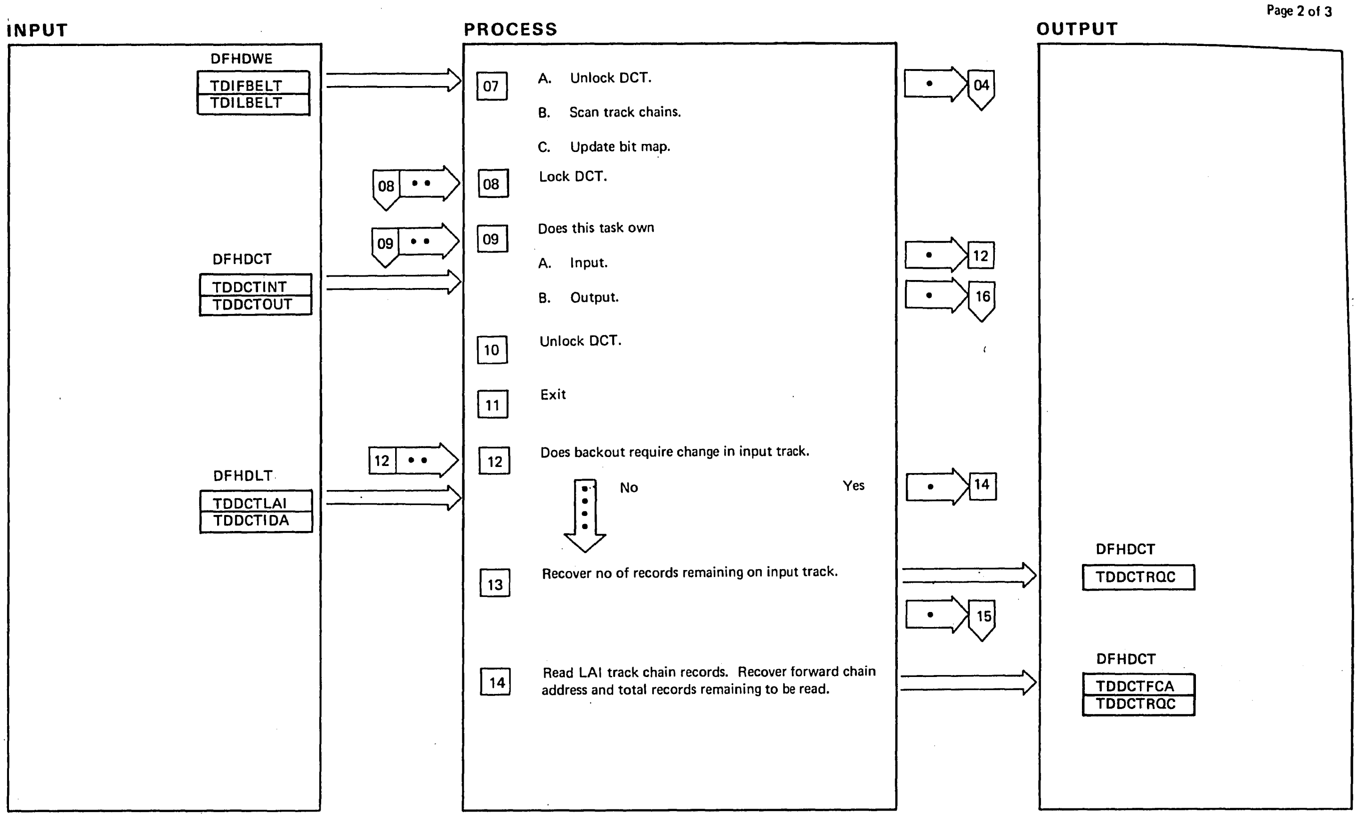
Diagram 1.7.2-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Control is passed to the DWE Processor from DFHSPP at end of LUW (logical unit of work) or from DFHDBP when dynamic transaction backout is required. In this case, DWEMODFN will have the backout bit set. | DWE PROCESSOR | TDDWEPR | |
| 03 For nonterminal destinations, tasks are attached. For terminal destinations, an AID is built. | | | |
| 05 Lock DCT to prevent extraneous modification. | | | |
| 06 This is the deferred processing done at end of LUW (logical unit of work). | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Transient Data DWE Processor

Diagram 1.7.2-01



07 A. Unlock DCT.

B. Scan track chains.

C. Update bit map.

08 Lock DCT.

09 Does this task own

A. Input.

B. Output.

10 Unlock DCT.

11 Exit

12 Does backout require change in input track.

13 Recover no of records remaining on input track.

14 Read LAI track chain records. Recover forward chain address and total records remaining to be read.

Transient Data DWE Processor

Diagram 1.7.2-02

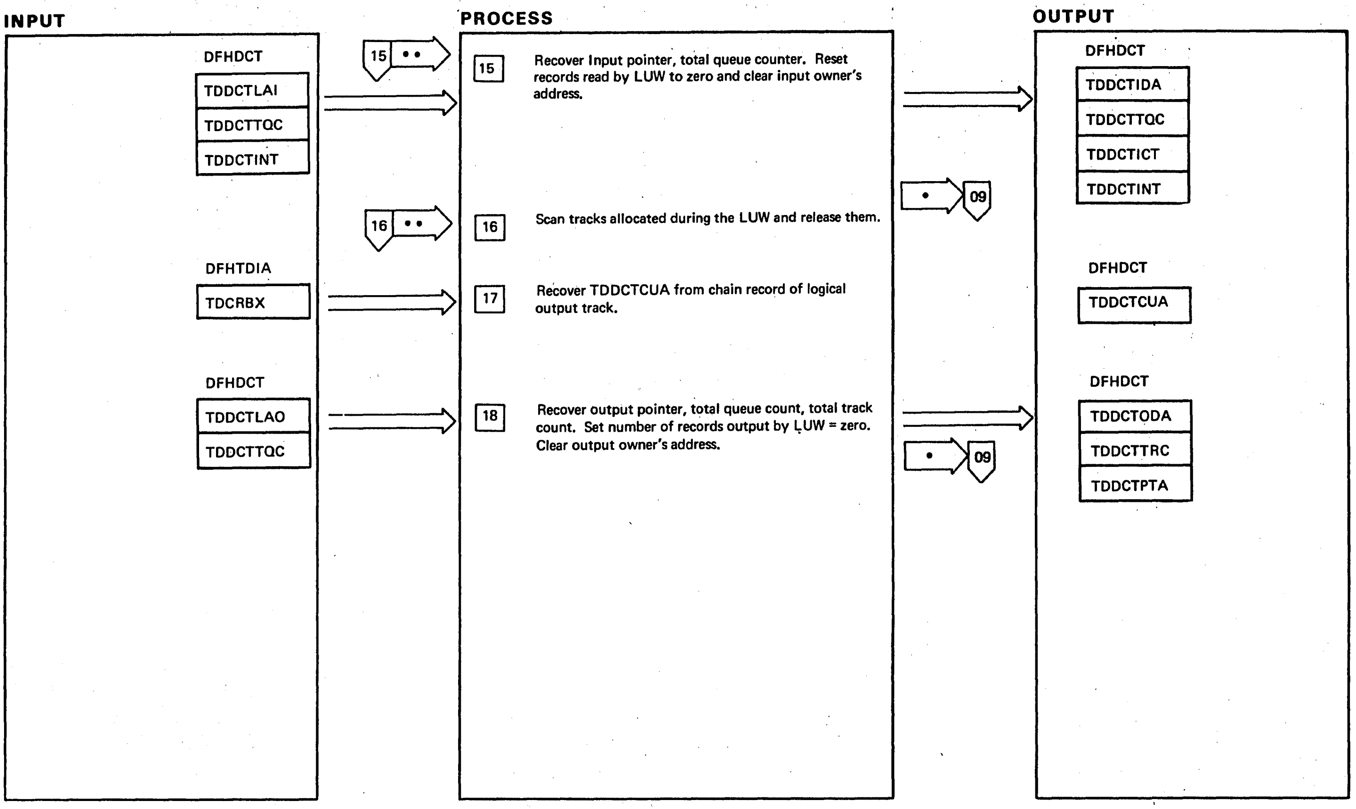
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 07 For nonreusable queues, the data is left intact. For reusable queues, the tracks are made available for other users and will be formatted upon allocation. | | | |
| 09 Regardless of whether this is an input or output DWE, input must be backed out first. The existence of input activity for the task is indicated by its, 'owning' the input for the destination. | | | |
| 13 New number of records remaining on input track = old less difference between logical and physical input pointers. | | | |
| 14 Number of records remaining to be read = total records on the track less those passed by logical input pointer. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Transient Data DWE Processor

Diagram 1.7.2-02

Page 3 of 3



15 Recover Input pointer, total queue counter. Reset records read by LUW to zero and clear input owner's address.

16 Scan tracks allocated during the LUW and release them.

17 Recover TDDCTCUA from chain record of logical output track.

18 Recover output pointer, total queue count, total track count. Set number of records output by LUW = zero. Clear output owner's address.

Transient Data DWE Processor

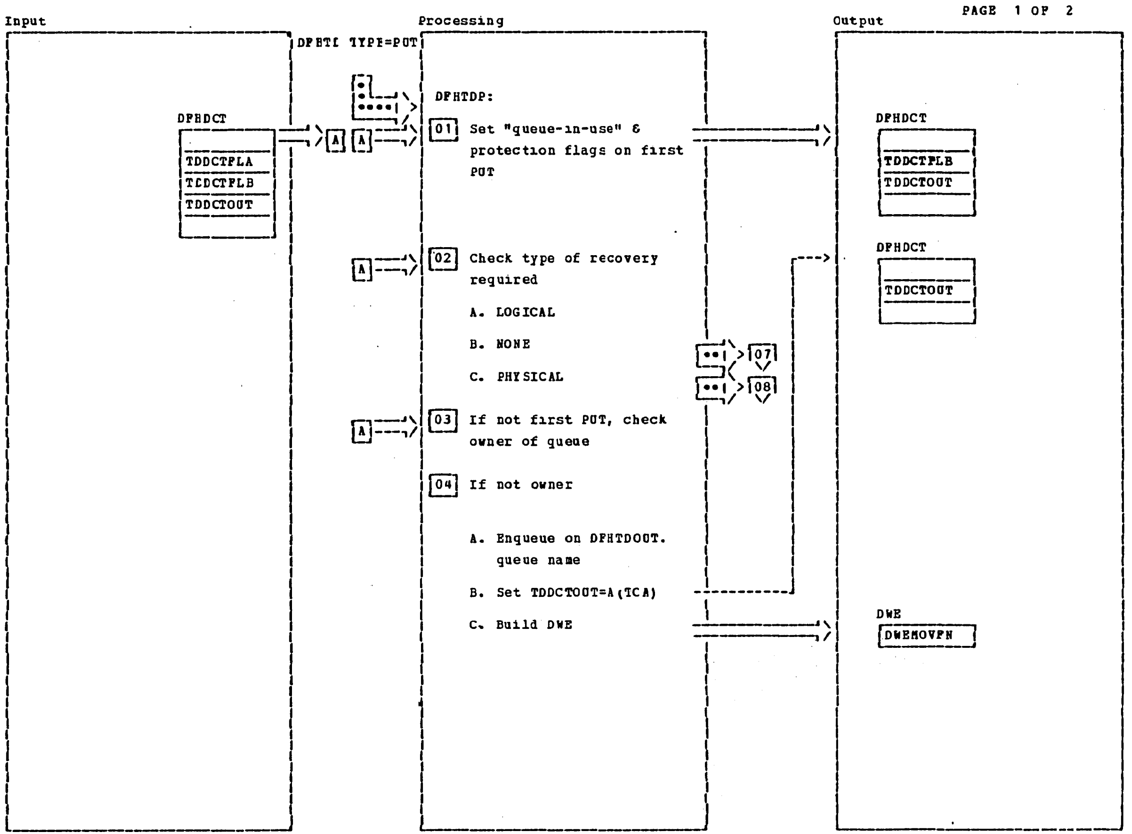
Diagram 1.7.2-03

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Transient Data DWE Processor

Diagram 1.7.2-03



Transient Data PUT

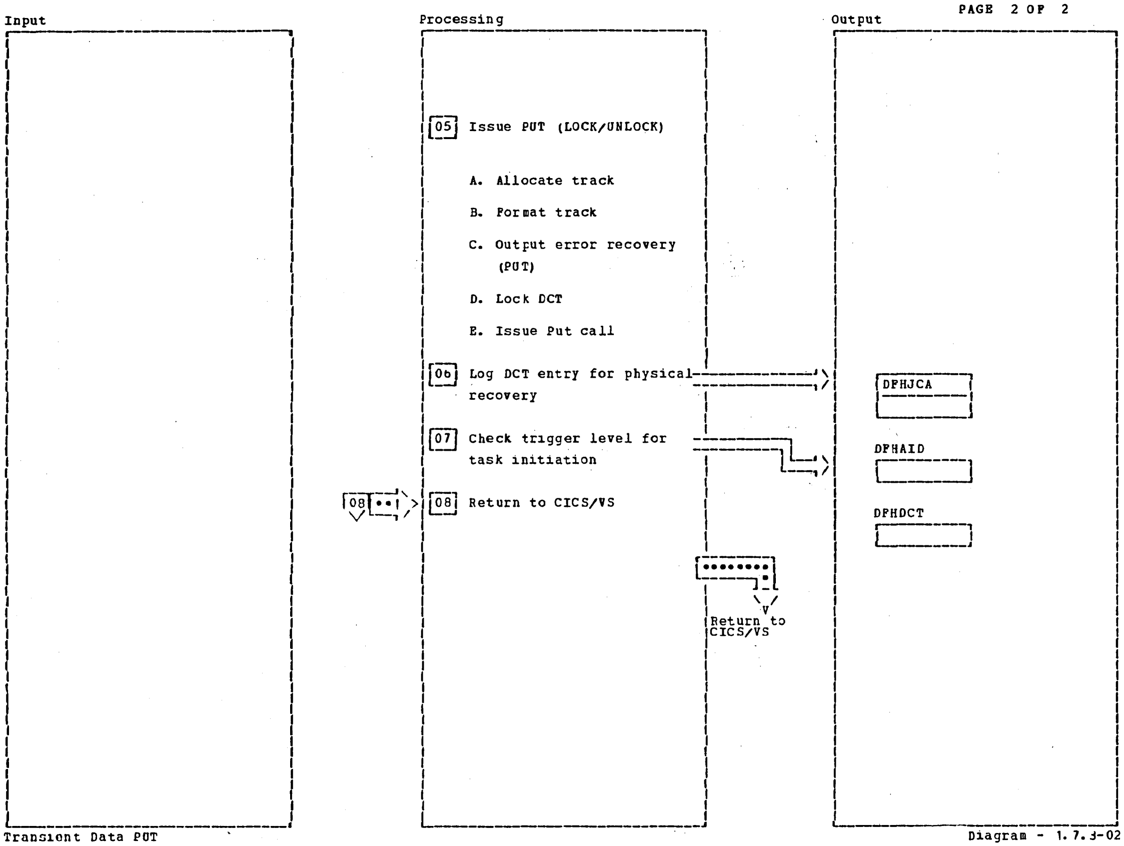
Diagram - 1.7.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Set TDDCTFLB using - TDDCTUSE = queue-in-use - TDDCTFLA moved to TDDCTFLB. | Recovery analysis | TRRANAL | | | | | |
| 02 TDDCTDPR = physical recovery & TDDCTDLR = logical recovery | | | | | | | |

Transient Data PUT

Diagram - 1.7.3-01

PAGE 2 OF 2



05 Issue PUT (LOCK/UNLOCK)

A. Allocate track

B. Format track

C. Output error recovery (PUT)

D. Lock DCT

E. Issue Put call

06 Log DCT entry for physical recovery

07 Check trigger level for task initiation

08 Return to CICS/VS

Transient Data PUT

Diagram - 1.7.3-02

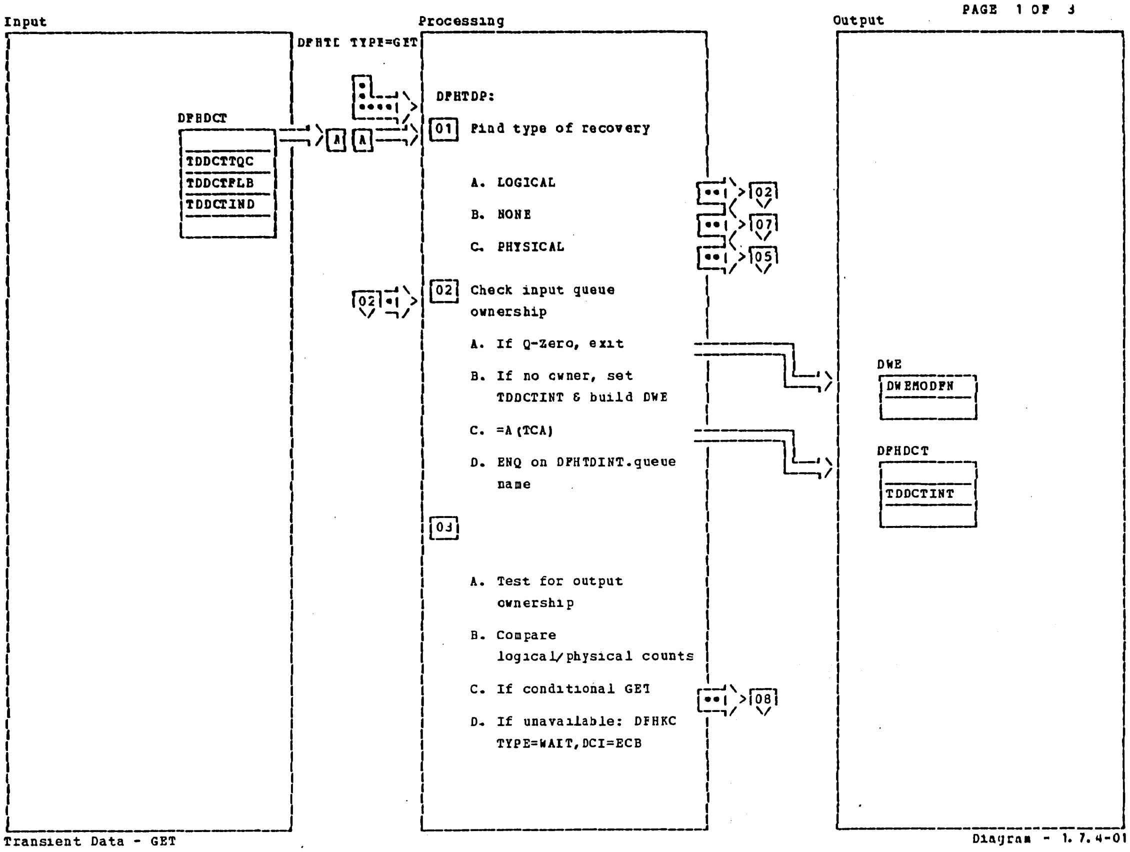
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 05 | | | |
| A. Format routine is single threaded. | | TDSPMGRS | |
| B. Track is formatted when allocated. | Acquir-ing a track | | |
| C. Output error recovery is used as needed. | Write error process-ing | TDWERR | |
| D. The DCT is locked prior to the PUT to prevent extraneous modification (DFHKC TYPE=LOCK). | | | |
| E. Issue Put call. | | | |
| 06 On the first Put, if the destination is physically recoverable the DCT entry is logged. | | TDPLRAJ | |
| 07 If the trigger level is met, an AID is scheduled if the destination is a terminal, or the specified task is attached if the destination is not a terminal. | | TDTINT | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 08 Set return code in TCATDTR and return to caller. | | | |

Transient Data PUT

Diagram - 1.7.3-02

PAGE 1 OF 3



Transient Data - GET

Diagram - 1.7.4-01

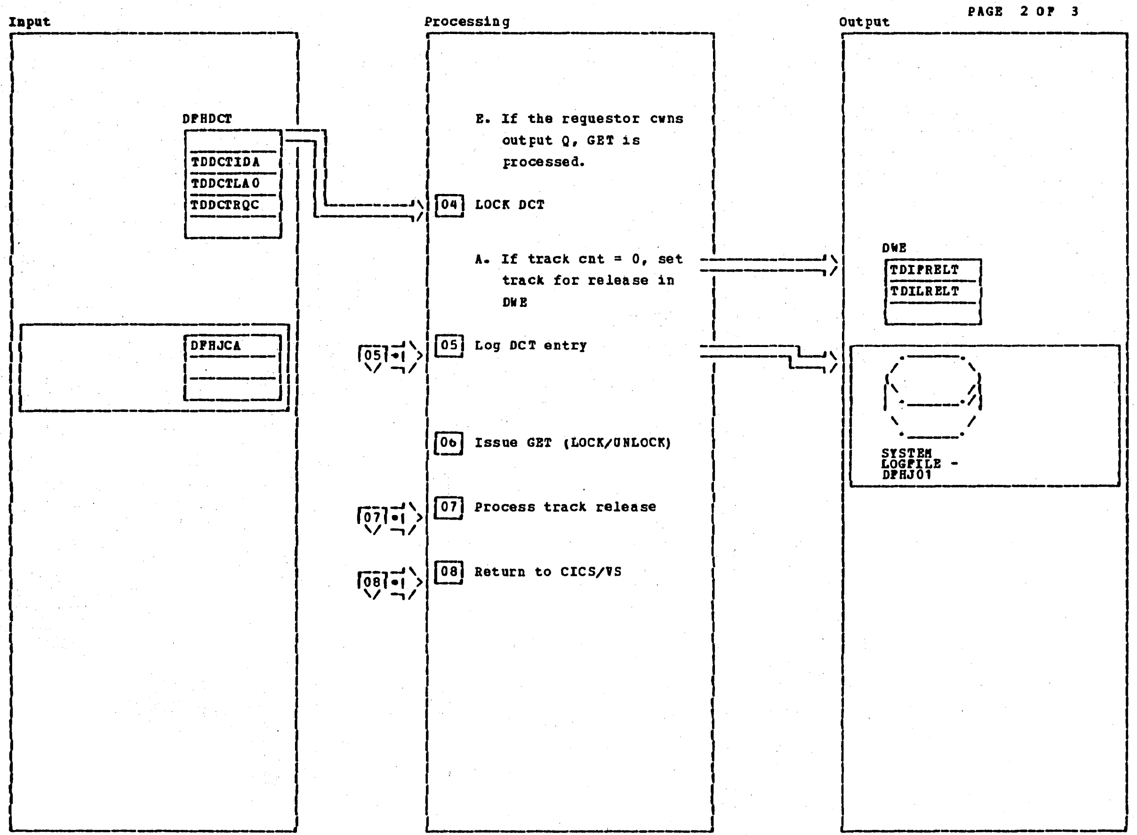
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 TDDCTFLB is tested for | Recovery Analysis | TDRANAL | |
| A. TCDCTDFR - physical recovery | | | |
| B. TDDCTDLR - logical recovery | | | |
| 02 TDDCTINT is tested | TDGIBAA | | |
| A. A check is made for a previous owner: if none, the DCT is reset. | | | |
| 03 | | | |
| A. If the request was conditicnal, control is returned to the user with a QUEBUSY return. | | | |
| B. Logical/Physical ccmpare is made to see if data is available to requestor. | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Transient Data - GET

Diagram - 1.7.4-01

PAGE 2 OF 3



E. If the requestor cwns output Q, GET is processed.

04 LOCK DCT

A. If track cnt = 0, set track for release in DWE

05 Log DCT entry

06 Issue GET (LOCK/UNLOCK)

07 Process track release

08 Return to CICS/VS

Transient Data - GET

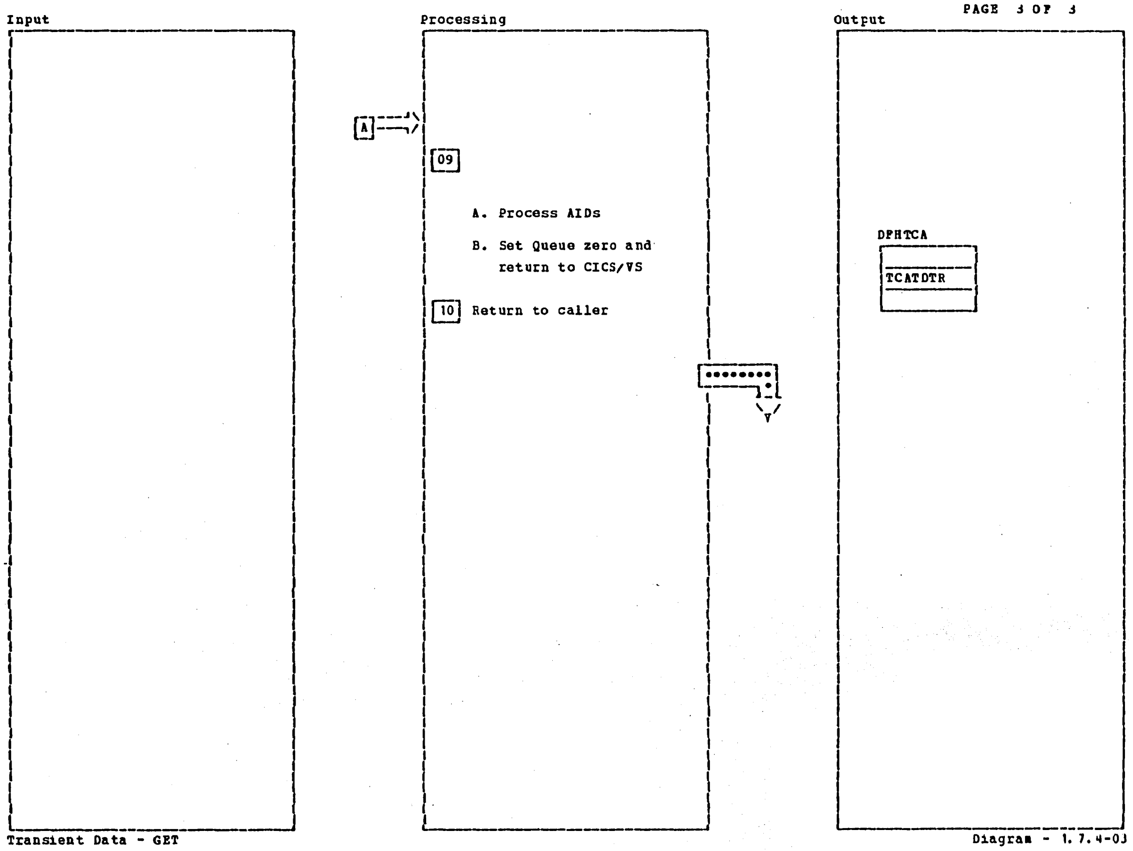
Diagram - 1.7.4-02

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 04 | | | |
| A. DCT is locked to prevent extraneous modification. | | | |
| B. If TCCCTRQC is Q, TDDCTIOA is updated to point to new track. | | | |
| C. If TDDCTRQC is 0, track is to be released. Update the DWE with TDIFRELT, TDILRELT. See DWE DSECT for field descriptions. The release is processed by DWE Processor at end of task. | | | |
| 05 For physical recovery, the DCT entry is logged with Journal Control macro, then the actual track release routine is entered prior to the GET. | | TDPIFIA | |
| 06 GET is processed. | | | |
| 07 If the destination is not recoverable but is reusable and all records of a track have been processed, the track is released. | | | |
| 08 Set response code in TCATDTR and return to caller. | | TDRTE | |

Transient Data - GET

Diagram - 1.7.4-02

PAGE 3 OF 3



09

A. Process AIDs

B. Set Queue zero and return to CICS/VS

10 Return to caller

Transient Data - GET

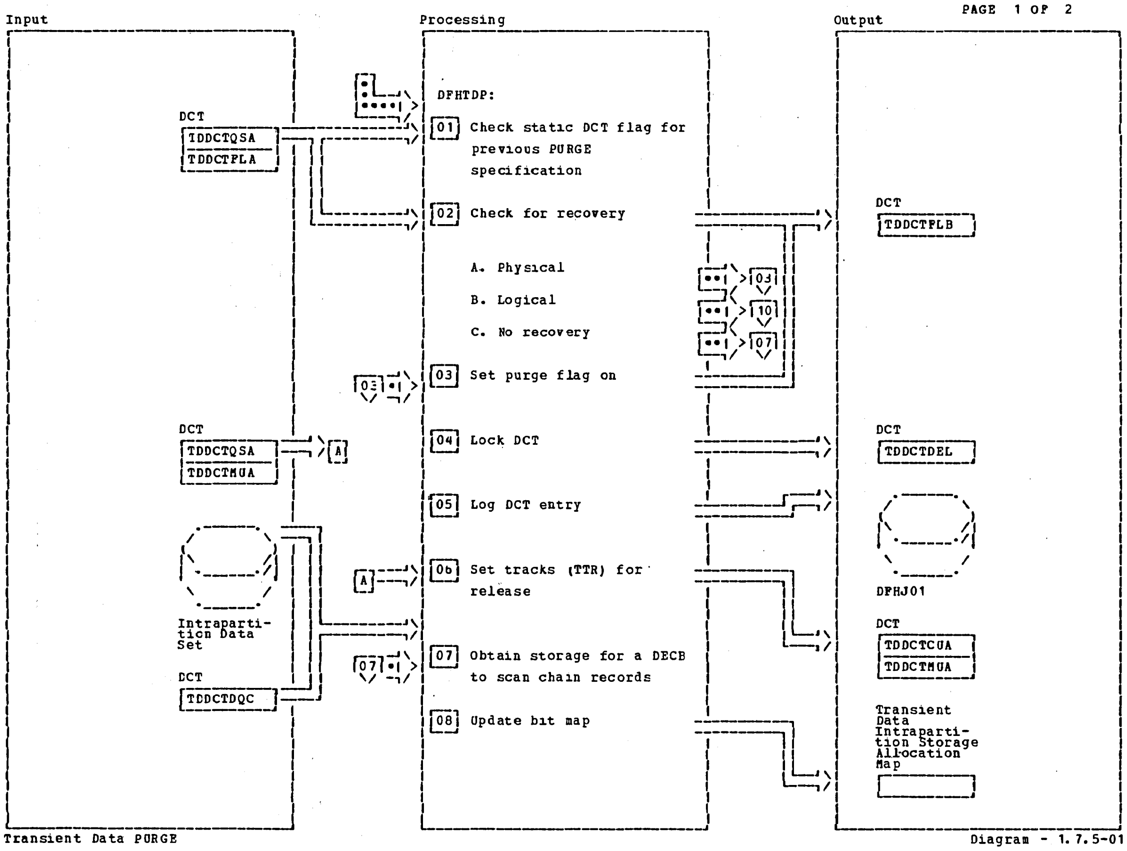
Diagram - 1.7.4-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 09 Cancel existing AIDs, reset DCT. | | | | | | | |
| 10 Set response code in TCATDTR, unlock DCT entry, and return to caller. | | | | | | | |

Transient Data - GET

Diagram - 1.7.4-03

PAGE 1 OF 2



DFHTDP:

01 Check static DCT flag for previous PURGE specification

02 Check for recovery

A. Physical

B. Logical

C. No recovery

03 Set purge flag on

04 Lock DCT

05 Log DCT entry

06 Set tracks (TTR) for release

07 Obtain storage for a DECB to scan chain records

08 Update bit map

Transient Data PURGE

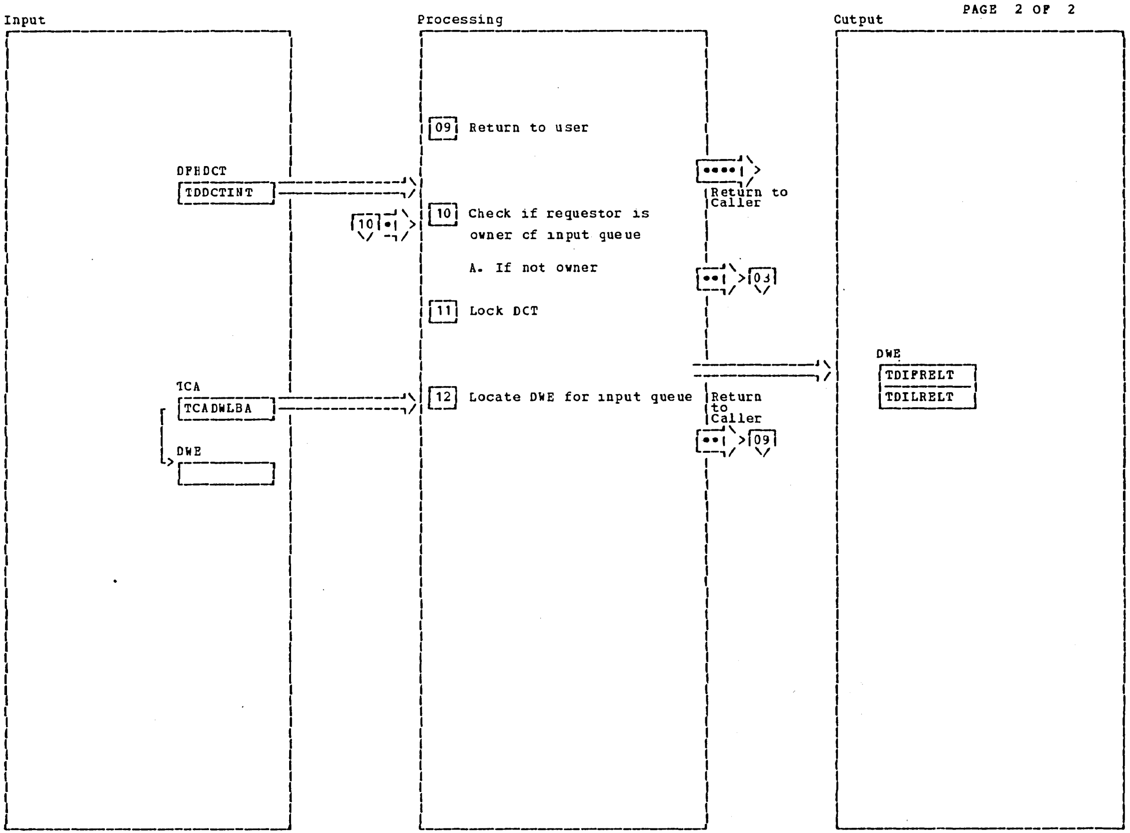
Diagram - 1.7.5-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 If TDDCTQSA = 0, then previously purged. Issue good return code to user. | | | |
| 02 If destination is LOGICAL or PHYSICAL, get Journal Control Area. Set dynamic flag in DCT. | | | |
| 03 Purge flag is used during recovery. | PHYSICAL RECOVERY PURGE | TDPGRAE | |
| 04 DCT is locked to prevent extraneous modification. | | | |
| 05 DCT is written to the System Log. | | | |
| 06 Reset DCT-set tracks for release<br>first track - TDDCTCUA<br>last track - TDDCTMUA | | | |
| 07 Allow PURGE on a Q zero condition for nonreusable Q. | COMMON PURGE PROCESS | TDPGRAD | |
| 08 Tracks are unallocated: data is left intact. | SPACE MANAGE-M ENT ROUTINE | TDSPMPRG | |

Transient Data PURGE

Diagram - 1.7.5-01

PAGE 2 OF 2



09 Return to user

10 Check if requestor is owner cf input queue

A. If not owner

11 Lock DCT

12 Locate DWE for input queue

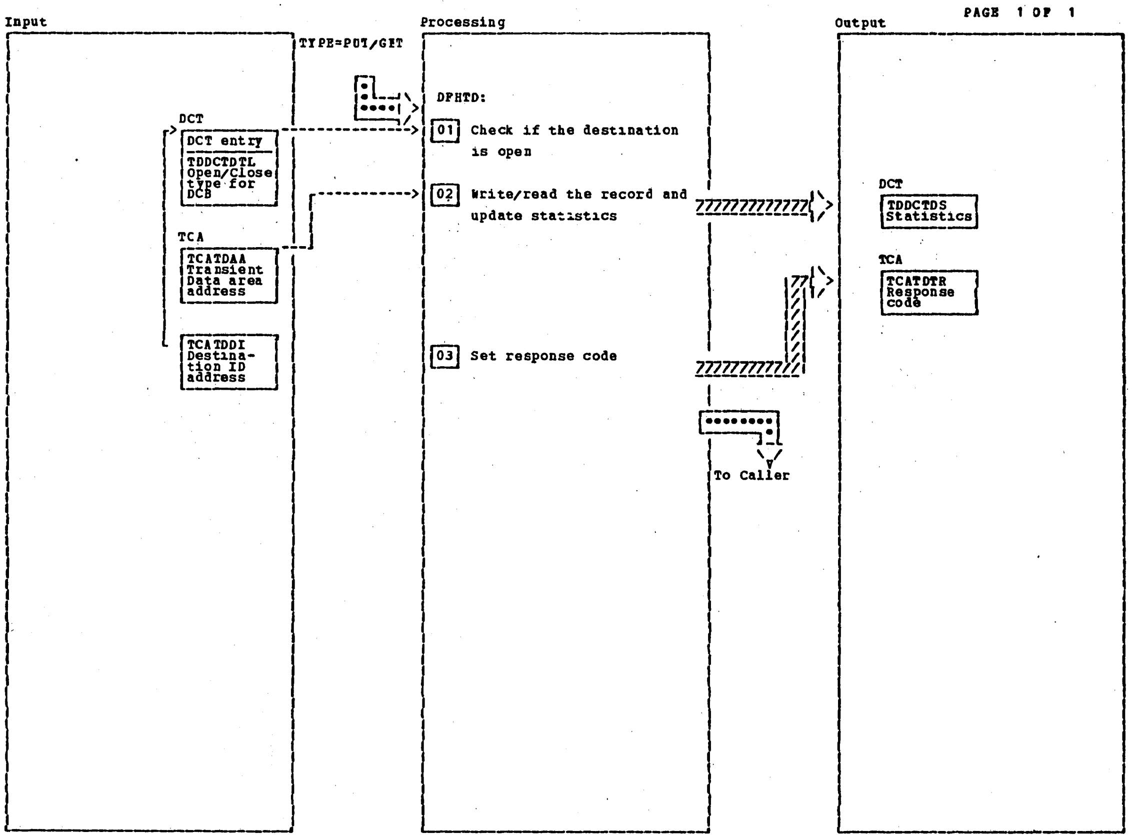
Transient Data PURGE

Diagram - 1.7.5-02

| Notes | Rcutine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 09 Return code is set. | | TDEXA1 | | | | | |
| 10 If requestor is new process same as physical - purge is not deferred. | LOGICAL RECOVERY PURGE | TDPGRAA | | | | | |
| 11 DCT is locked to prevent extraneous modification. | | | | | | | |
| 12 First and last tracks are set for deferred release. | | | | | | | |

Transient Data PURGE

Diagram - 1.7.5-02


PAGE 1 OF 1
Input
Processing
Output
TYPE=PUT/GET
DFHTD:
01 Check if the destination is open
02 Write/read the record and update statistics
03 Set response code
To Caller
DCT
DCT entry
TDDCTDTL Open/Close type for DCB
TCA
TCATDAA Transient Data area address
TCATDDI Destination ID address
DCT
TDDCTDS Statistics
TCA
TCATDTR Response code

Dispose of Acquired Data (Extrapartiticn)

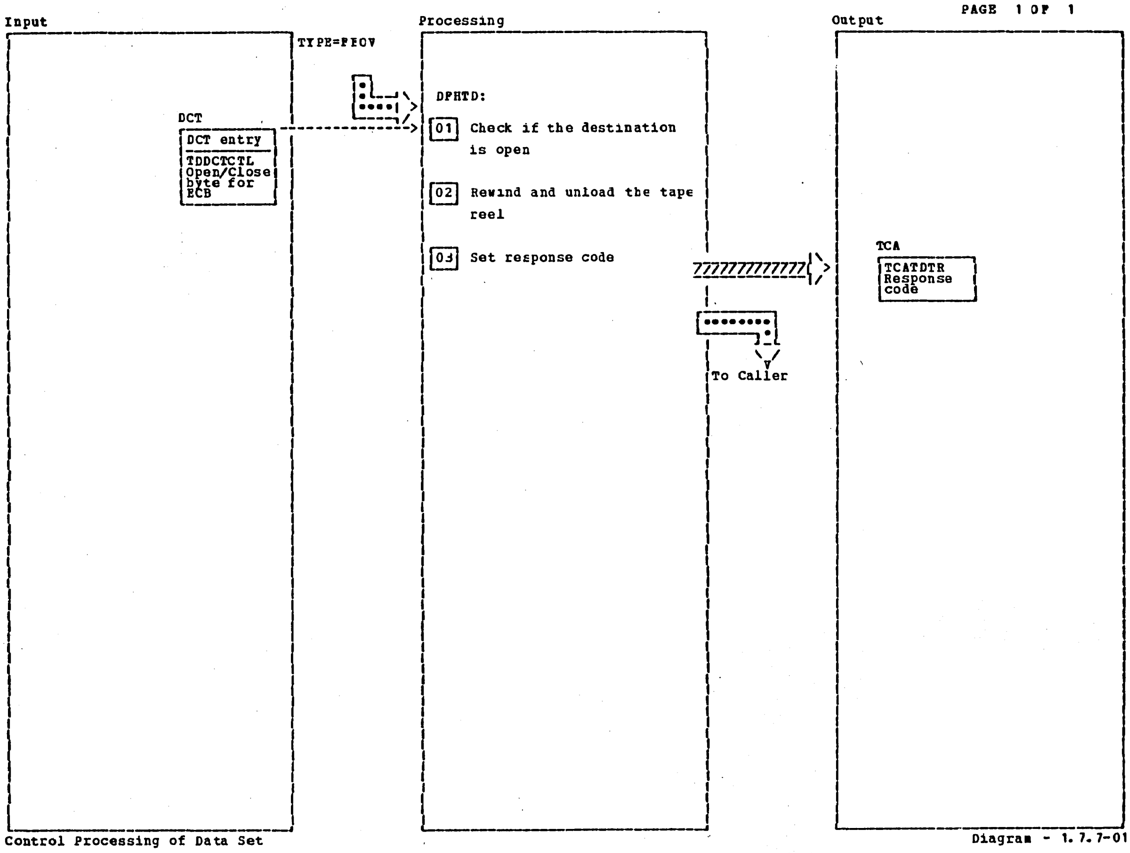
Diagram - 1.7.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If the DCB is closed, task is terminated with ABEND code ATDD. | | DOCTDP08 | | | | | |
| 02 A link is made to the QSAM GET/PUT routine. (The same routine is entered for both GET and PUT requests.) The appropriate statistics are then updated. | | DOCTDP11 | | | | | |
| 03 Either the normal response code or the queue zero response (GET requests only) is issued. | | | | | | | |

Dispose of Acquired Data (Extrapartition)

Diagram - 1.7.6-01

PAGE 1 OF 1


Input
Processing
Output
TYPE=FEOV
DCT
DCT entry
TDDCTCTL Open/Close byte for ECB
DFHTD:
01 Check if the destination is open
02 Rewind and unload the tape reel
03 Set response code
TCA
TCATDTR Response code
To Caller

Control Processing of Data Set

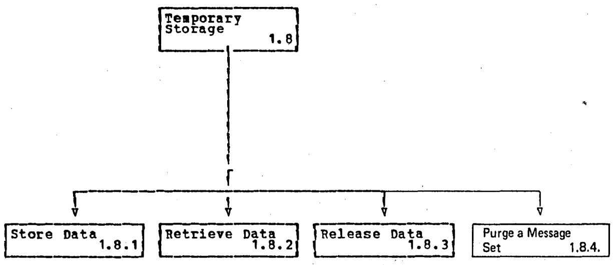
Diagram - 1.7.7-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If the DCB is closed, an error response is set and control is returned to the caller. | | DOCTDP08 | | | | | |
| 02 An operating system supervisor call is issued. | | DOCTDP09 | | | | | |
| 03 Only the normal response can be issued. | | DOCTDP17 | | | | | |

Control Processing of Data Set

Diagram - 1.7.7-01

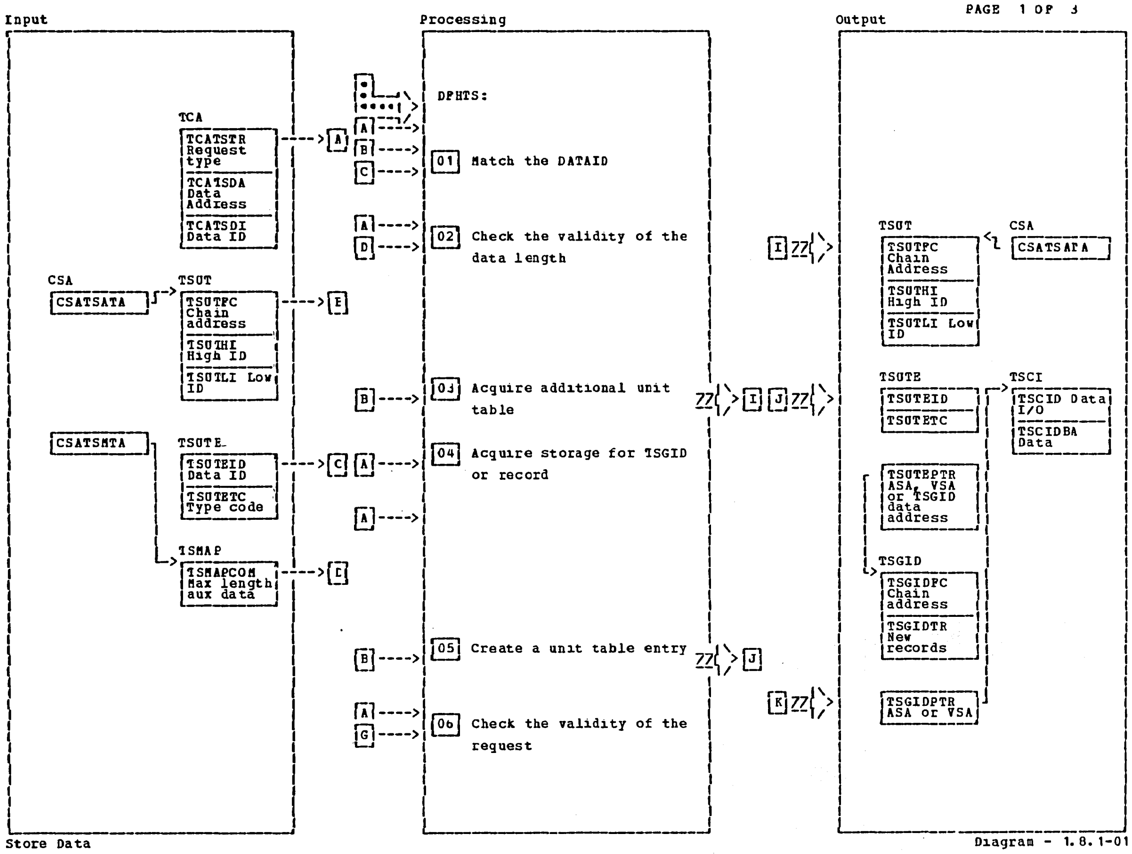
PAGE 1 OF 1



Temporary Storage

Diagram - 1.8-01

PAGE 1 OF 3



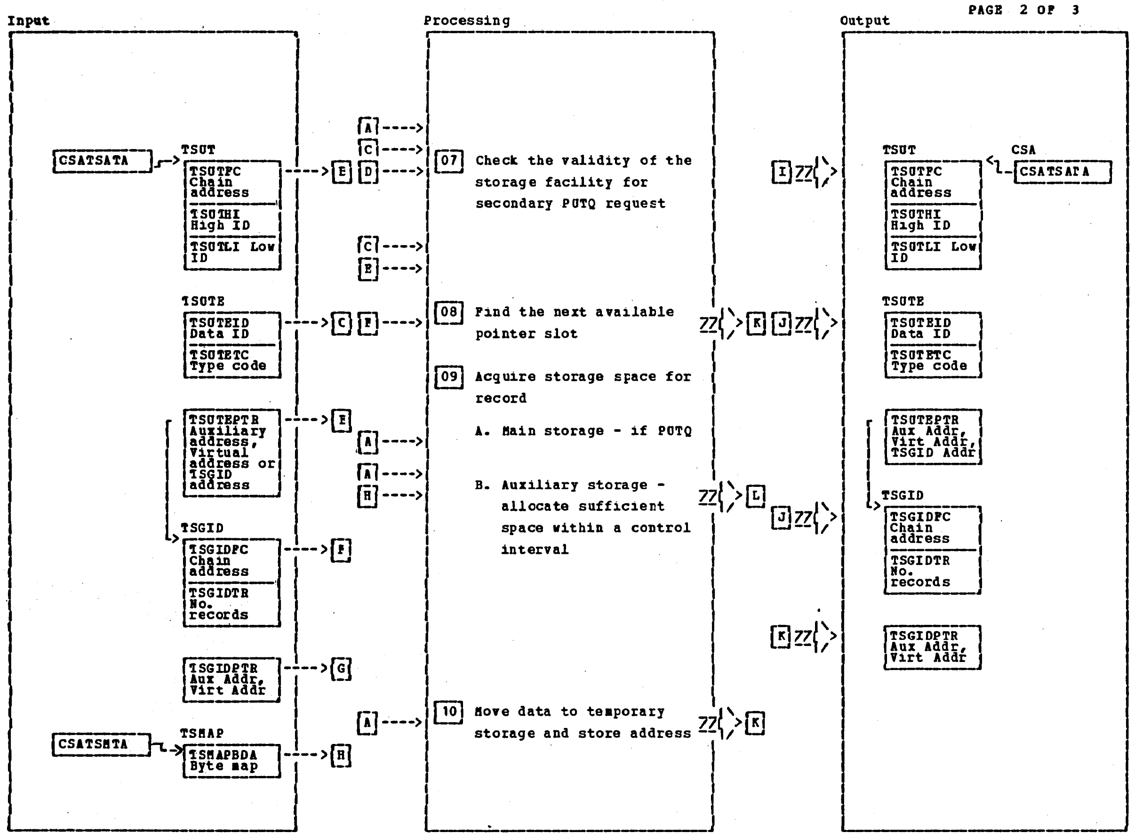
Store Data

Diagram - 1.8.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The unit tables are scanned to determine if the DATAID was entered by a previous PUT or PUTQ. | | TSP020 | |
| 02 The length of a user data record must be greater than four bytes and, for auxiliary storage requests, less than the maximum length that can be contained within a control interval. | | TSP040 | |
| 03 If an entry already exists in the unit table for the specified DATAID, go to step 6. Else, if no space exists in the unit table for creating an entry, acquire an additional unit table. | | | |
| 04 For a PUT to main storage, acquire storage for the record. | TSF220 | TSP041<br>TSP044 | |
| For a PUTQ, acquire storage for a TSGID. | TSPC95 | TSP043 | |
| 05 Create a unit table entry in proper collating sequence with other entries in the table and fill in control information. Go to step C8. | | TSP066 | |
| 06 Message sets created by PUTQ request are restricted to the PUTQ, GETQ, and PURGE request set. Entries created by a PUT request are restricted to the GET, PUT-REPLACE, and RELEASE request set. Any violation of these rules results in an invalid request response. With no recovery, a PUT request to an existing entry results in a TSGID being created. With recovery, a PUT request to an existing entry results in an ATSP abend (multiple requests must be PUTQ requests). | | TSP100 | |

Store Data

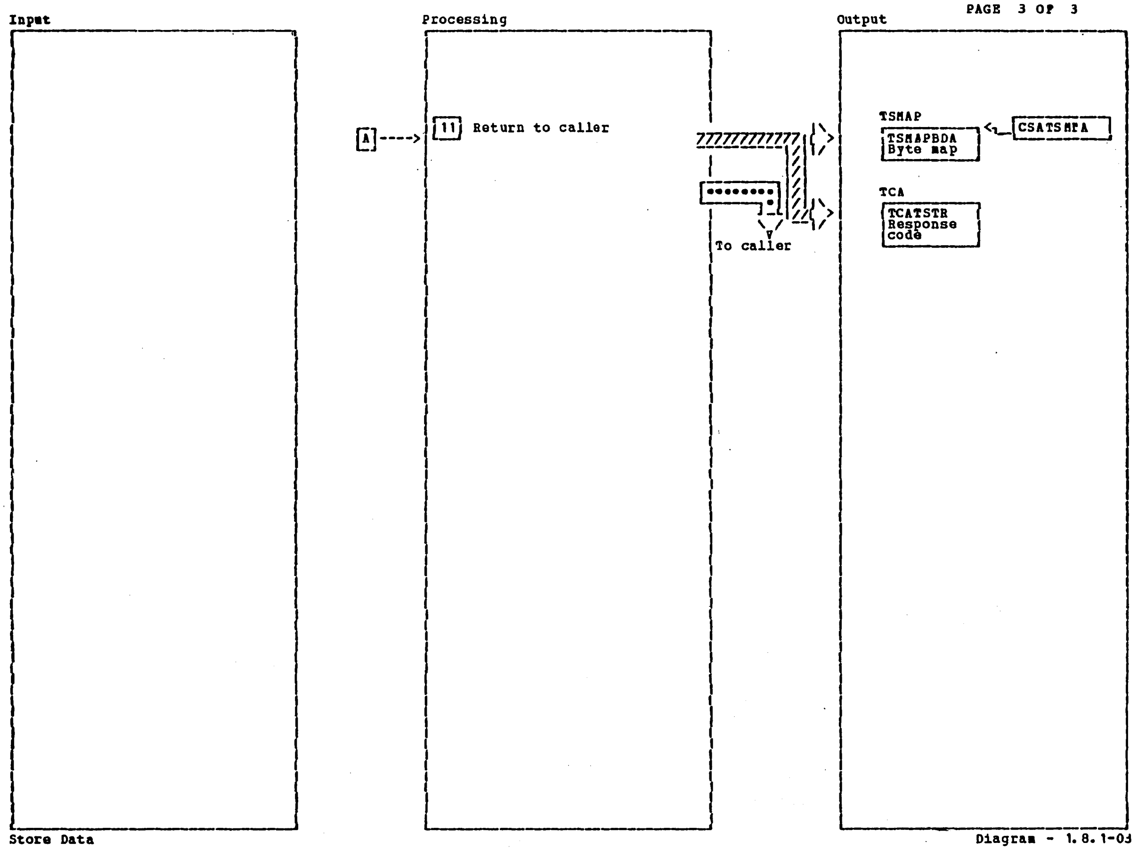
Diagram - 1.8.1-01


PAGE 2 OF 3
Input
Processing
Output
CSATSATA
TSUT
TSUTFC Chain address
TSUTHI High ID
TSUTLI Low ID
TSUTE
TSUTEID Data ID
TSUTETC Type code
TSUTEPTR Auxiliary address, Virtual address or TSGID address
TSGID
TSGIDFC Chain address
TSGIDTR No. records
TSGIDPTR Aux Addr, Virt Addr
CSATSMTA
TSMAP
TSMAPBDA Byte map
A
C
D
E
F
G
H
07 Check the validity of the storage facility for secondary PUTQ request
08 Find the next available pointer slot
09 Acquire storage space for record
A. Main storage - if PUTQ
B. Auxiliary storage - allocate sufficient space within a control interval
10 Move data to temporary storage and store address
I
J
K
L
TSUT
CSA
CSATSATA
TSUTFC Chain address
TSUTHI High ID
TSUTLI Low ID
TSUTE
TSUTEID Data ID
TSUTETC Type code
TSUTEPTR Aux Addr, Virt Addr, TSGID Addr
TSGID
TSGIDFC Chain address
TSGIDTR No. records
TSGIDPTR Aux Addr, Virt Addr

Store Data    Diagram - 1.8.1-02

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 07 The initial PUTQ request determines the storage facility of the data. A subsequent PUTQ request forces the storage facility of the request to conform to the facility of the original request. | | TSP120 | |
| 08 The next available pointer slot is found if a message set exists. This will contain either the virtual storage address (VSA) or the auxiliary storage address (ASA) of the record. Acquire an additional TSGID if necessary. If not a message set, the pointer slot is in the unit table entry. | TSP600 | TSP200 | |
| 09 | | | |
| A. For main storage request, acquire storage for record if a PUTQ request. | TSP220 | | |
| B. For auxiliary storage request, scan the byte map to find a CI that has space available to contain the record. If space is not available, the task is suspended until space becomes available, unless the request was conditional. | TSP400 | TSP250 | |
| 10 Move the user's data to the acquired area. For an auxiliary storage request, update the byte map. If the CI is full, write it. Store the record address in the pointer slot. | | TSP300 | |

Store Data    Diagram - 1.8.1-02



Store Data

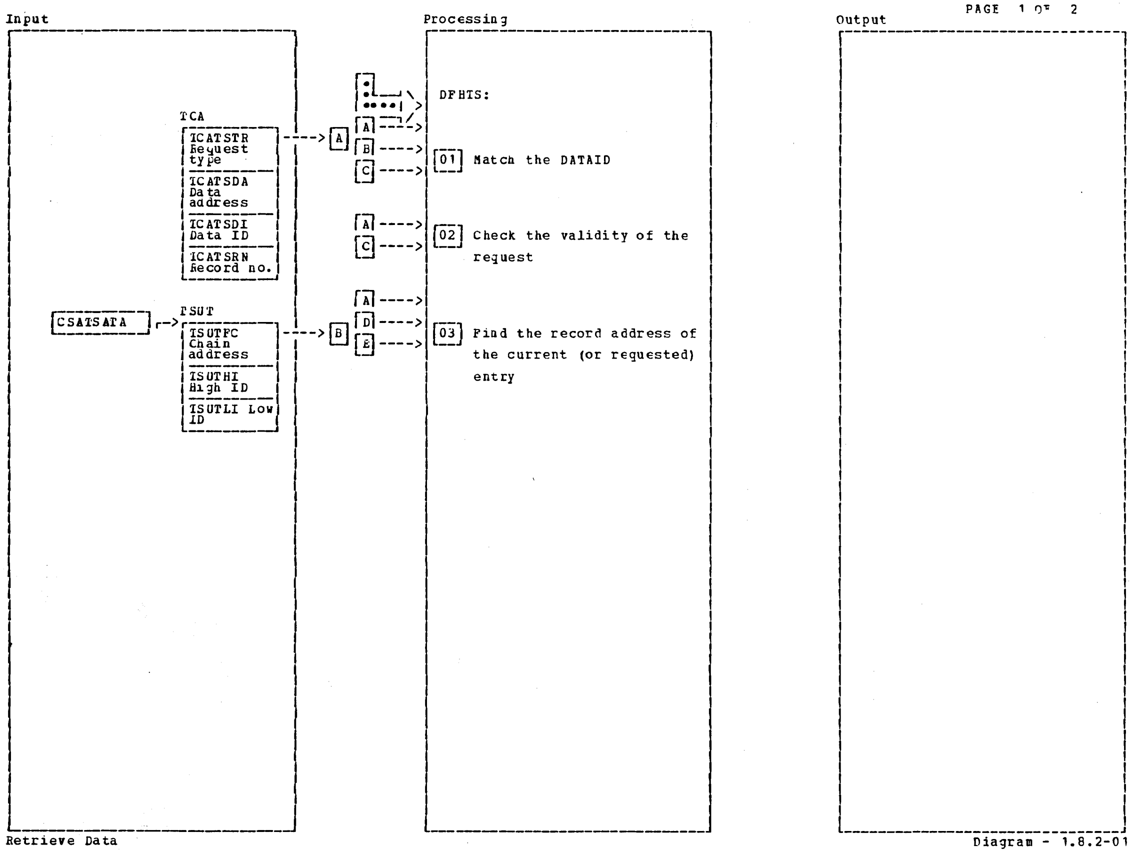
Diagram - 1.8.1-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 11 Set response code in the TCA and return to caller. | | TSNXIT | | | | | |

Store Data

Diagram - 1.8.1-03

PAGE 1 OF 2



DFHTS:

01 Match the DATAID

02 Check the validity of the request

03 Find the record address of the current (or requested) entry

Retrieve Data

Diagram - 1.8.2-01

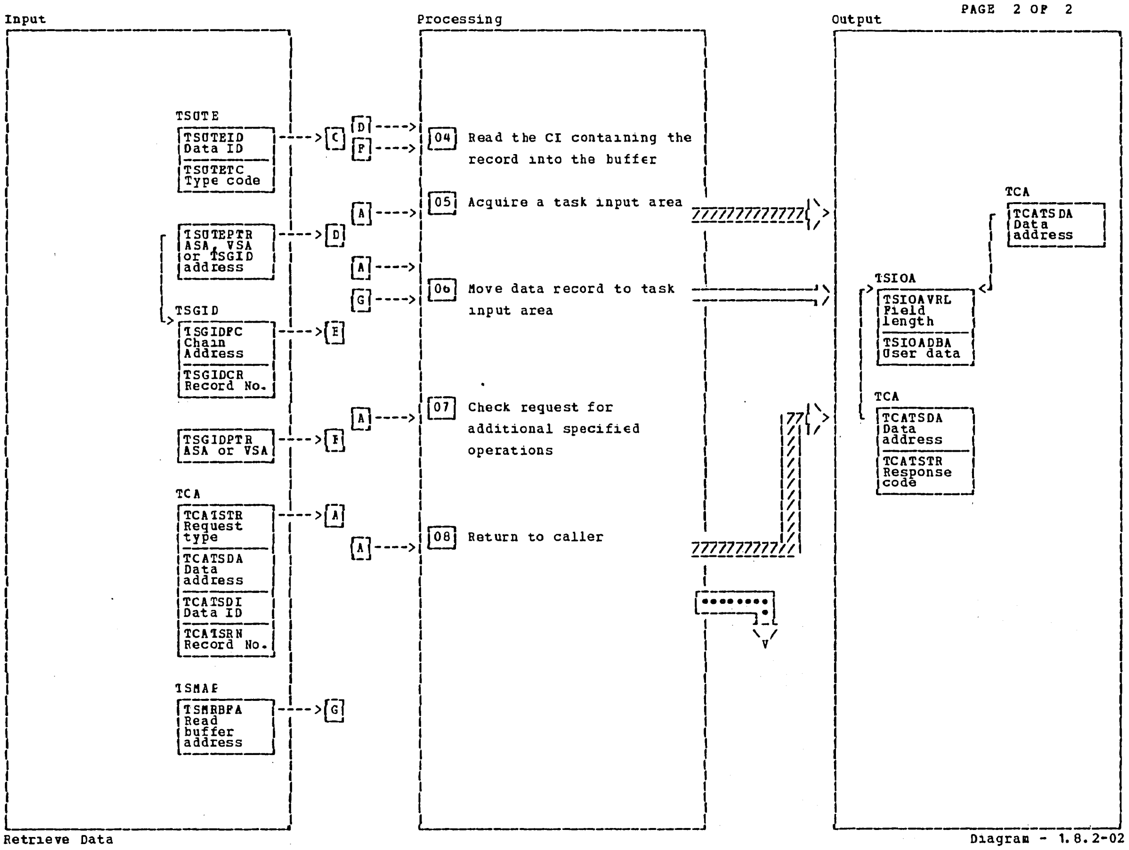
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The unit tables are scanned to determine if the DATAID was entered by a previous PUT or PUTQ. If a match is not found, the user receives an ID error message. | | TSP020 | |
| 02 A GET request is valid only for an entry created by a PUT. A GETQ request is valid only for a message set created by a PUTQ. | | TSP100 | |
| 03 If the data to be retrieved is an entry of a message set, the pointer slot is found. If the entry does not exist, the user receives an entry error response. | TSP600 | TSP500 | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Retrieve Data

Diagram - 1.8.2-01

PAGE 2 OF 2



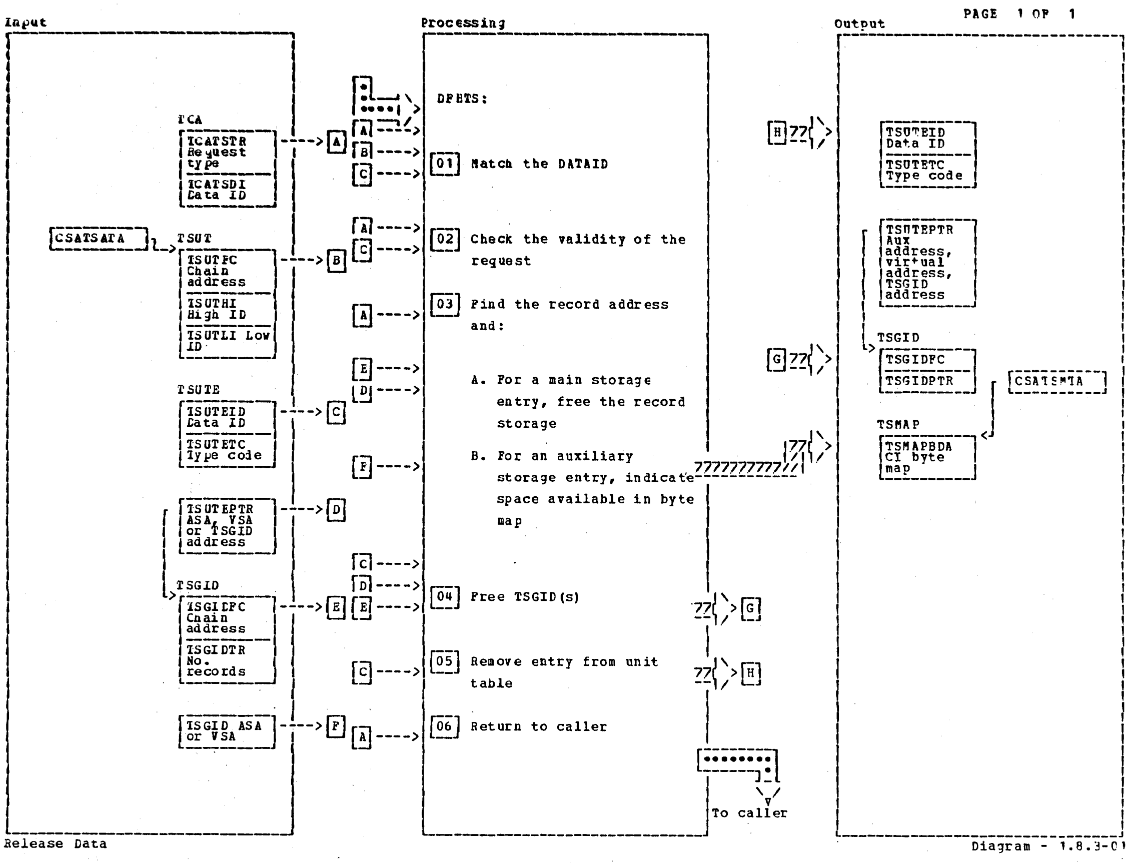
Retrieve Data

Diagram - 1.8.2-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 If the record is in auxiliary storage and is not currently in either the read of write buffer, the CI containing the record is read into the read buffer. | | TSP540 | | | | | |
| 05 If the task has not provided an input area, temporary storage dynamically acquires one. | | TSP570 | | | | | |
| 06 Move data record to task input area. | | TSP590 | | | | | |
| 07 Release may be specified in conjunction with GET. | TSPRUL | TSP595 | 1.8.3 | | | | |
| 08 Set response code in the TCA and return to caller | | TSNXIT | | | | | |

Retrieve Data

Diagram - 1.8.2-02



Release Data

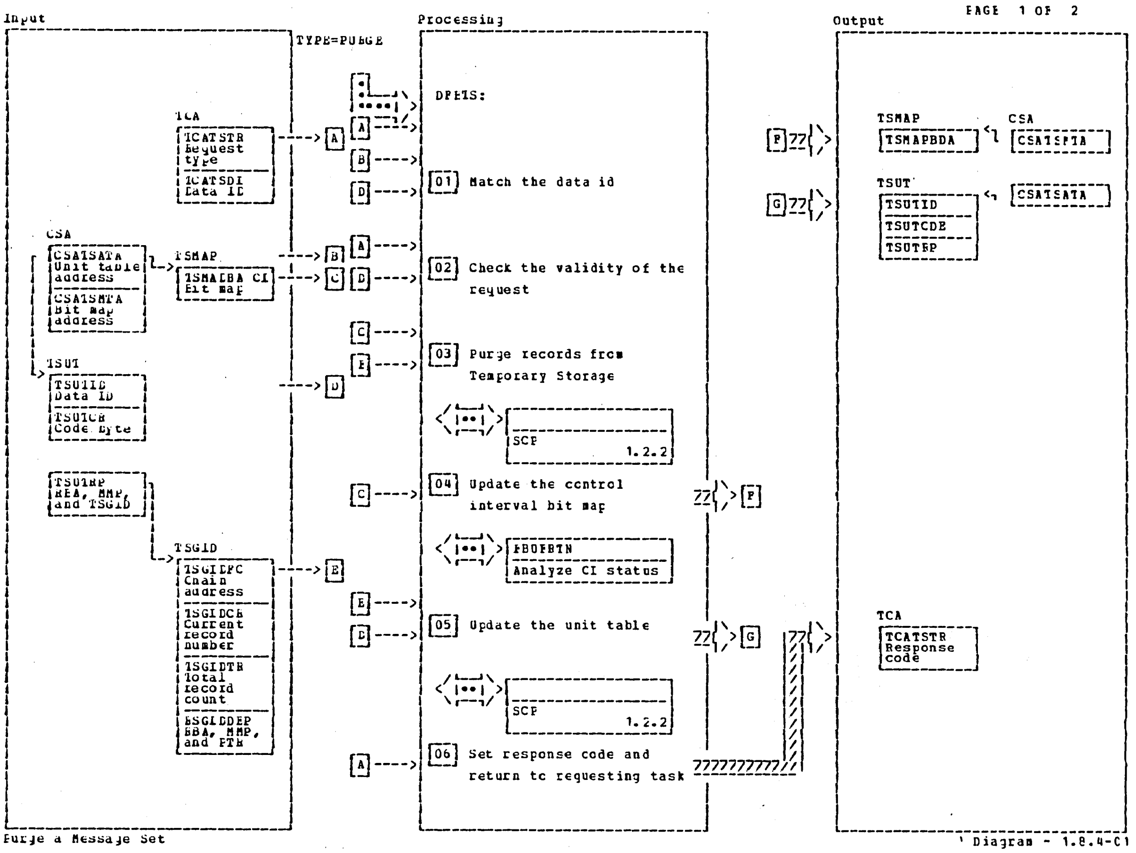
Diagram - 1.8.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The Unit Tables are scanned to determine if the DATAID was entered by a previous PUT or PUTQ. If a match is not found, the user receives an ID error response. | | TSP020 | |
| 02 A Release request is valid only for an entry created by a PUT. A Purge request is valid only for a message set created by a PUTQ. | | TSP100 | |
| 03 Find the record address and free the allocated space. | TSP600 | TSP700<br>TSP800 | |
| 04 If the data to be released are entries of a message set, free the associated TSGID(s). | | | |
| 05 The unit table entry is removed. If this was the last unit table entry, the unit table is freed. | | | |
| 06 Set response code in the TCA and return to caller. | | TSNXIT | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Release Data

Diagram - 1.8.3-01



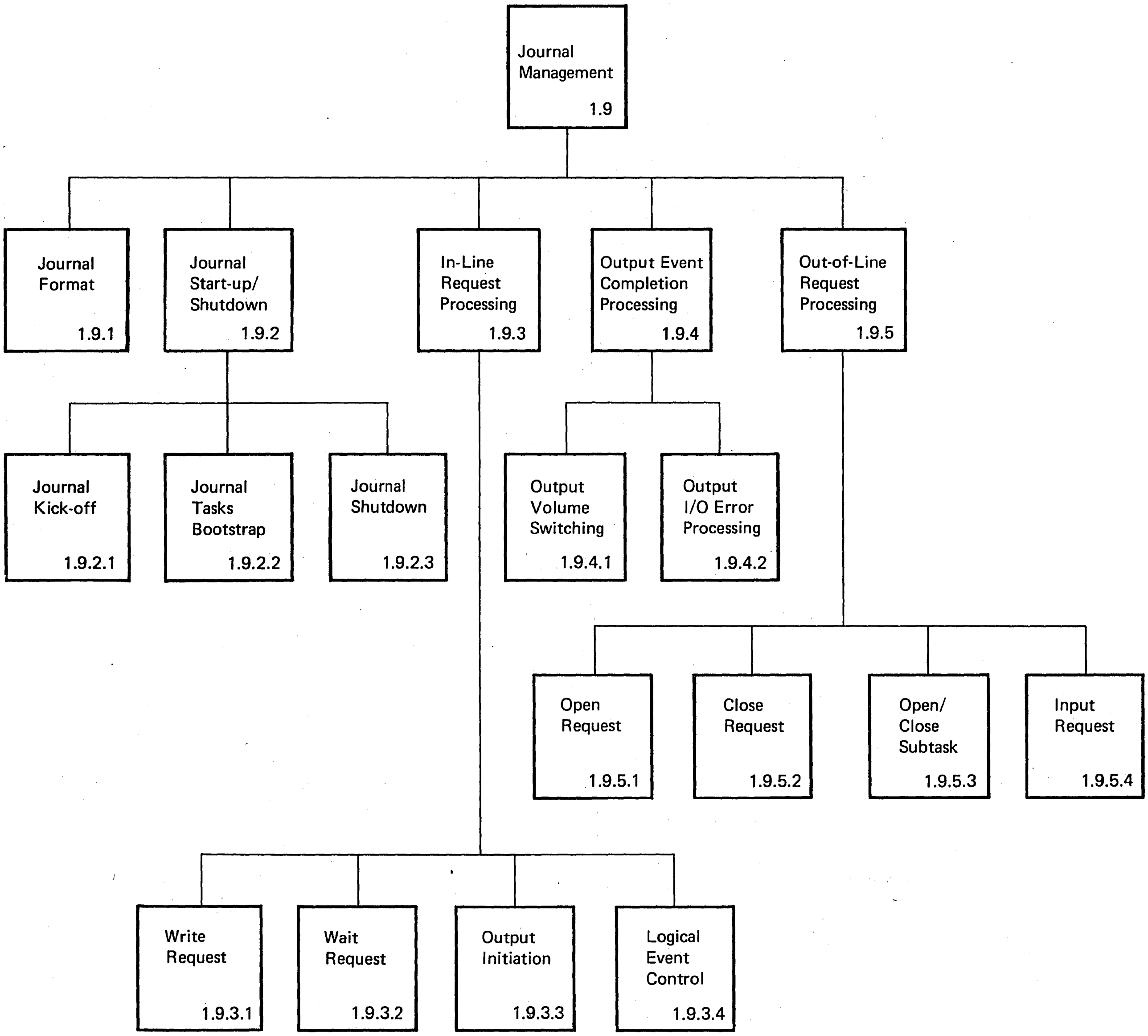
Purge a Message Set

Diagram - 1.8.4-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The unit tables are scanned to find a match to the data ID. If a match is not found, the user receives an ID error response. | | TSPUS1 | | | | | |
| 02 A purge request is valid only for a message set created by a PUTQ request. | | TSPEA1 | | | | | |
| 03 Main storage occupied by the records of a message set is released. Disk storage occupied by the records of a message set is made available for reuse. | | TSPMNF | | | | | |
| 04 The control interval bit map used in temporary disk storage is updated to reflect the status of the CI after the purge. | PBUFBIN | | | | | | |
| 05 The main storage occupied by the associated TSGIDs is released. When the records of the message set are released, the unit table entry that held the ID control information becomes available for reuse. | | TSPMFR | | | | | |

Purge a Message Set

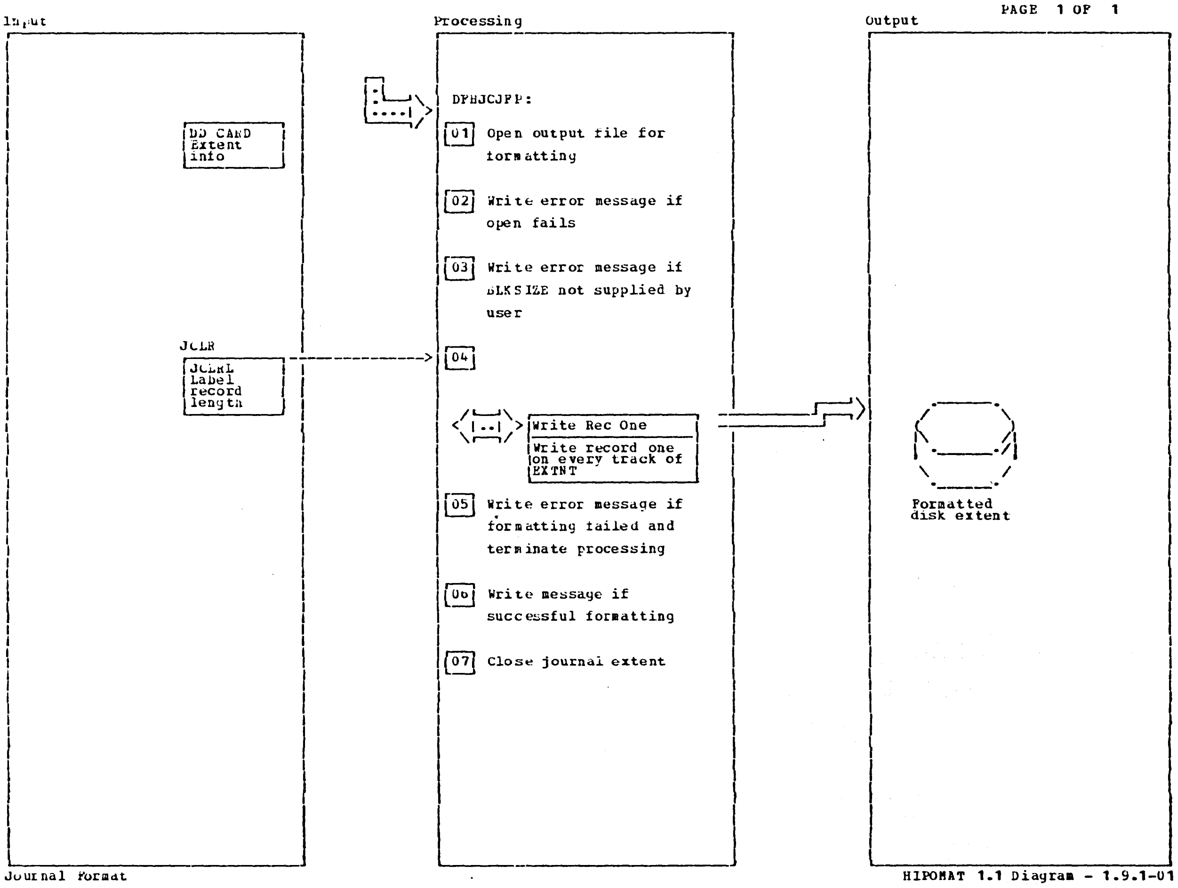
Diagram - 1.8.4-C1


Journal Management 1.9
Journal Format 1.9.1
Journal Start-up/ Shutdown 1.9.2
In-Line Request Processing 1.9.3
Output Event Completion Processing 1.9.4
Out-of-Line Request Processing 1.9.5
Journal Kick-off 1.9.2.1
Journal Tasks Bootstrap 1.9.2.2
Journal Shutdown 1.9.2.3
Output Volume Switching 1.9.4.1
Output I/O Error Processing 1.9.4.2
Open Request 1.9.5.1
Close Request 1.9.5.2
Open/ Close Subtask 1.9.5.3
Input Request 1.9.5.4
Write Request 1.9.3.1
Wait Request 1.9.3.2
Output Initiation 1.9.3.3
Logical Event Control 1.9.3.4

Journal Management

Diagram - 1.9.0-01

PAGE 1 OF 1



DFHJCJFP:

01 Open output file for formatting

02 Write error message if open fails

03 Write error message if BLKSIZE not supplied by user

04

05 Write error message if formatting failed and terminate processing

06 Write message if successful formatting

07 Close journal extent

Journal Format

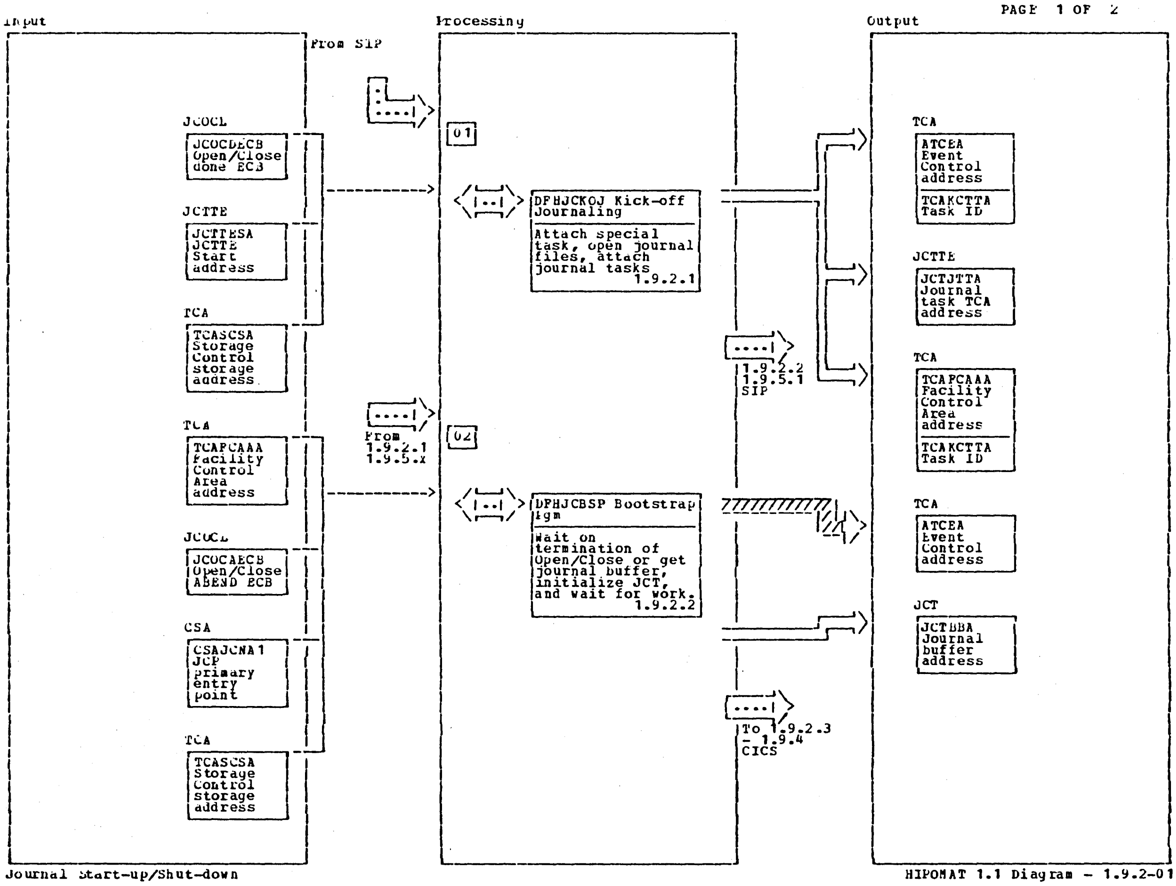
HIPOMAT 1.1 Diagram - 1.9.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 02 Message to operator: DFH4597 - Journal extent not initialized - unable to open DCB. Check DD card supplied. | | ERROR1 | |
| 03 Message to operator: DFH4598 - Journal extent not initialized - BLKSIZE not specified on DD card or in DSCB. | | ERROR2 | |
| 04 Only single, contiguous extents are formatted. A record one is written on every track with the length of the journal label record.<br>The record consists of binary zeros except the appropriate positions of the time and date in the label record are initialized to packed zeros. | | WRITER1 | |
| 05 Message to operator: DFH4596 - Journal extent not initialized - I/O error occurred. (Abend code: 456). | | ERROR | |
| 06 Message to operator: DFH4599 - | | EXIT | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| Journal extent initialized - nnnnn tracks available. | | | |

Journal Format

HIPOMAT 1.1 Diagram - 1.9.1-01



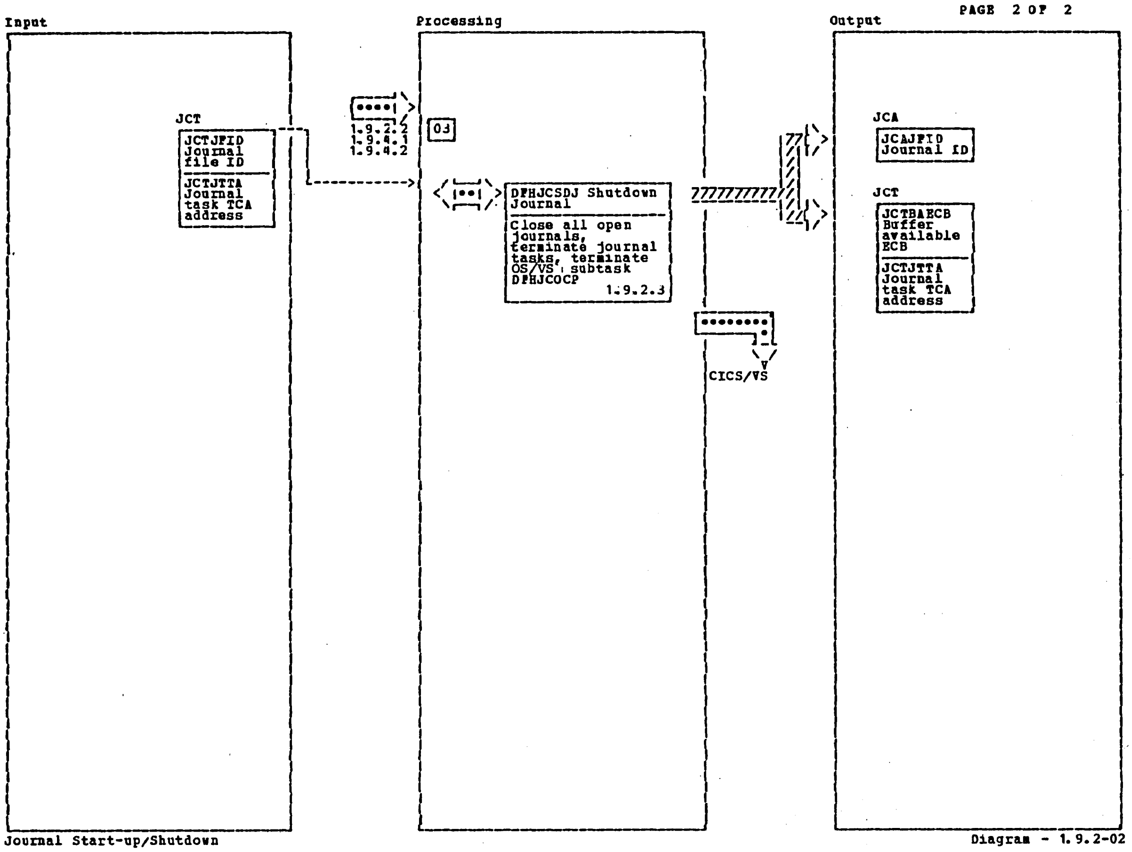
Journal Start-up/Shut-down

HIPOMAT 1.1 Diagram - 1.9.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 DFHJCKOJ gets control via CICS link from SIP. After it has opened the journal files and attached journal tasks, control is returned to SIP.<br>All attached tasks are transaction ID CSJC. One such task has the special purpose of waiting on termination of DFHJCOCP, the OS-subtask for Journal Open/Close's. The others, one per journal, are the actual journal tasks which expedite journal output processing. | | | | | | | |
| 02 DFHJCBSP is the initial program executed for all CSJC transactions. Depending on the field TCAFCAAA, it waits for termination of DFHJCOCP or passes control to DFHJCP. If TCAFCAAA is zero, the program waits for termination of DFHJCOCP. | | | | | | | |

Journal Start-up/Shut-down

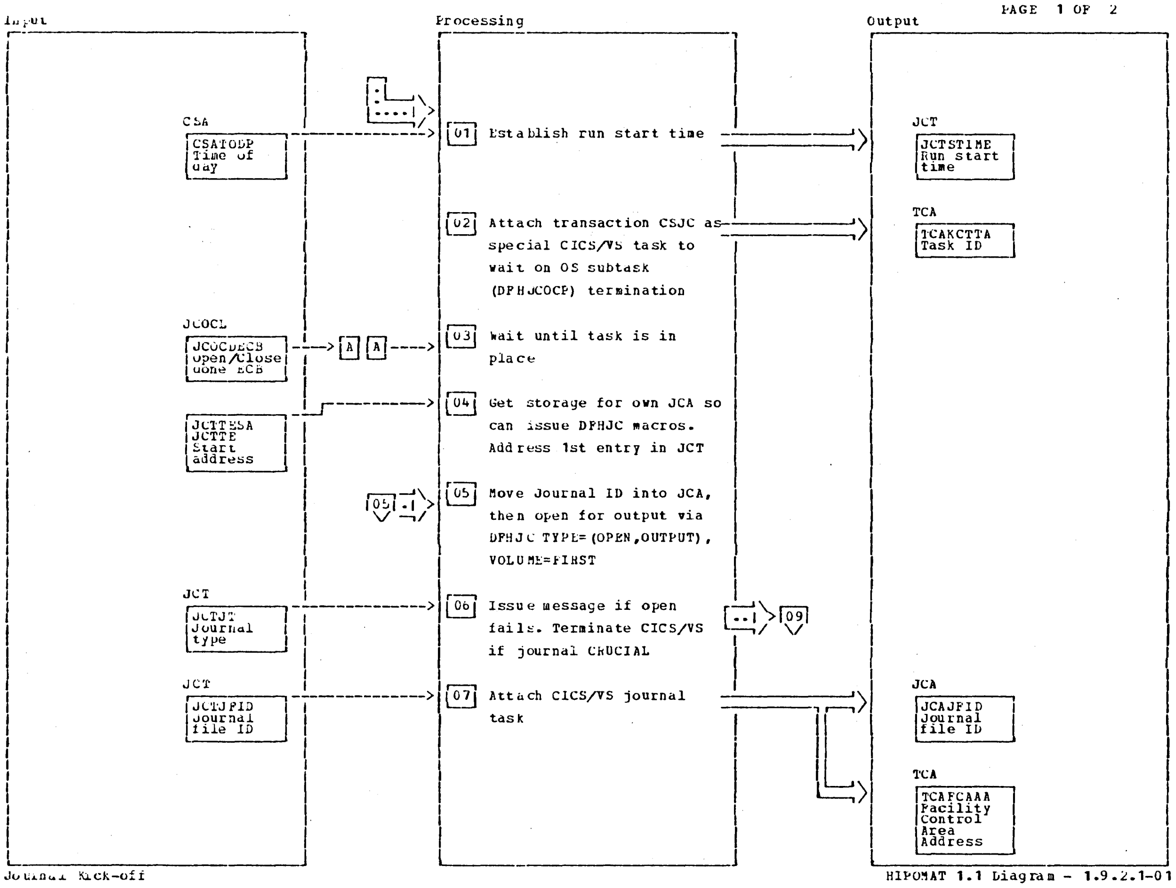
HIPOMAT 1.1 Diagram - 1.9.2-01


PAGE 2 OF 2
Input
Processing
Output
JCT
JCTJFID Journal file ID
JCTJTTA Journal task TCA address
1.9.2.2
1.9.4.1
1.9.4.2
03
DFHJCSDJ Shutdown Journal
Close all open journals, terminate journal tasks, terminate OS/VS subtask DFHJCOCP
1.9.2.3
CICS/VS
JCA
JCAJFID Journal ID
JCT
JCTBAECB Buffer available ECB
JCTJTTA Journal task TCA address

Journal Start-up/Shutdown Diagram - 1.9.2-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 DFHJCSDJ is called from DFHSTP when CICS/VS is closing down. Control is returned if Journaling was not active.<br>It is called from DFHJCIOE, DFHJCECV, or DFHJCBSP for some error conditions in Journal Management. | | | | | | | |

Journal Start-up/Shutdown Diagram - 1.9.2-02



01 Establish run start time

02 Attach transaction CSJC as special CICS/VS task to wait on OS subtask (DFHJCOCP) termination

03 Wait until task is in place

04 Get storage for own JCA so can issue DFHJC macros. Address 1st entry in JCT

05 Move Journal ID into JCA, then open for output via DFHJC TYPE=(OPEN,OUTPUT), VOLUME=FIRST

06 Issue message if open fails. Terminate CICS/VS if journal CRUCIAL

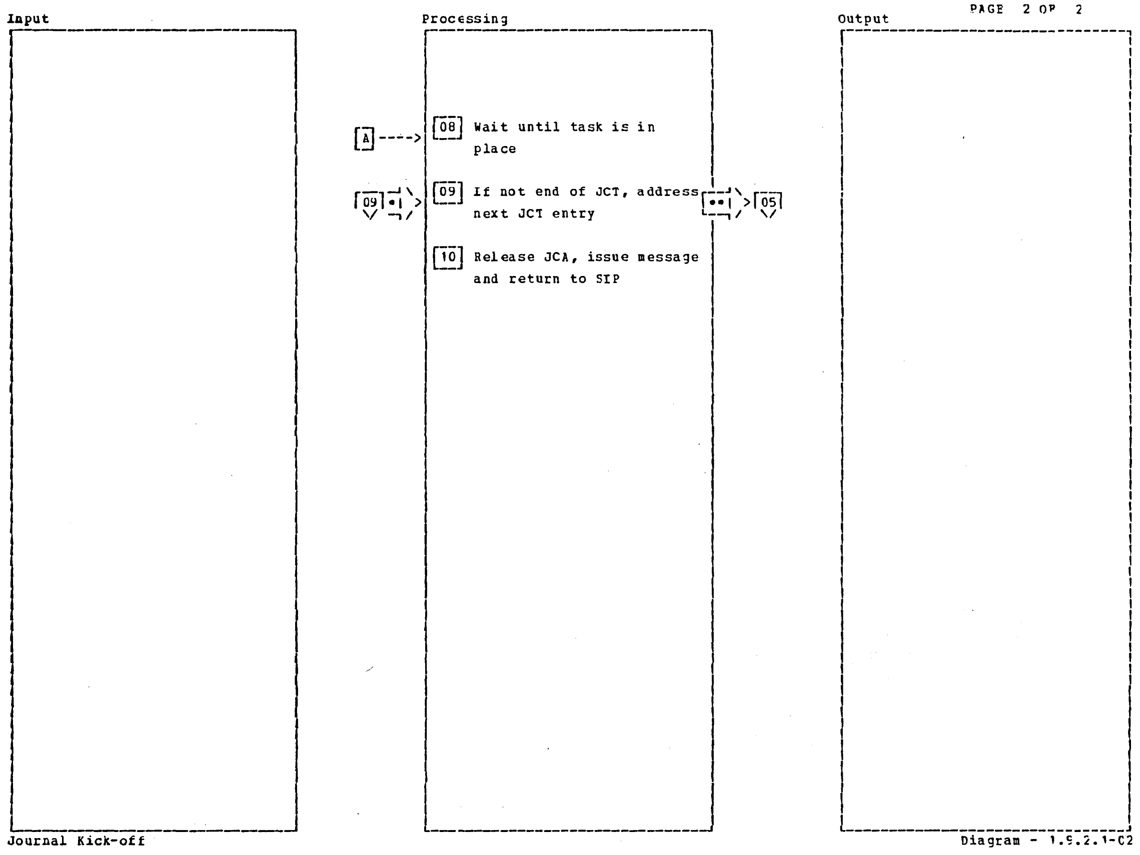
07 Attach CICS/VS journal task

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 02 Facility Control Area address in TCAFCAAA is zero. So CICS/VS task waits for termination of DFHJCOCP.<br>Transaction ID is CSJC for initial program DFHJCBSP. All such TCA's are identified as CICS/VS 'system tasks' by setting field TCAKCCTA to characters JJJ. | | | |
| 06 The message format is: 'DFH4501 - CICS/VS Journal nn not available - Initial open failure'. If Journal is CRUCIAL, CICS/VS is terminated with dump: OS ABEND 450 . | | B | |
| 07 The task is attached as under step 2 except the field TCAFCAAA is initialized to the JCT entry address.<br>Transaction ID is CSJC for initial program DFHJCBSP. All such TCAs are identified as CICS/VS 'System tasks' by setting field TCAKCTTA to | | D | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| characters JJJ. | | | |

Journal Kick-off HIPOMAT 1.1 Diagram - 1.9.2.1-01

PAGE 2 OF 2



08 Wait until task is in place

09 If not end of JCT, address next JCT entry

10 Release JCA, issue message and return to SIP

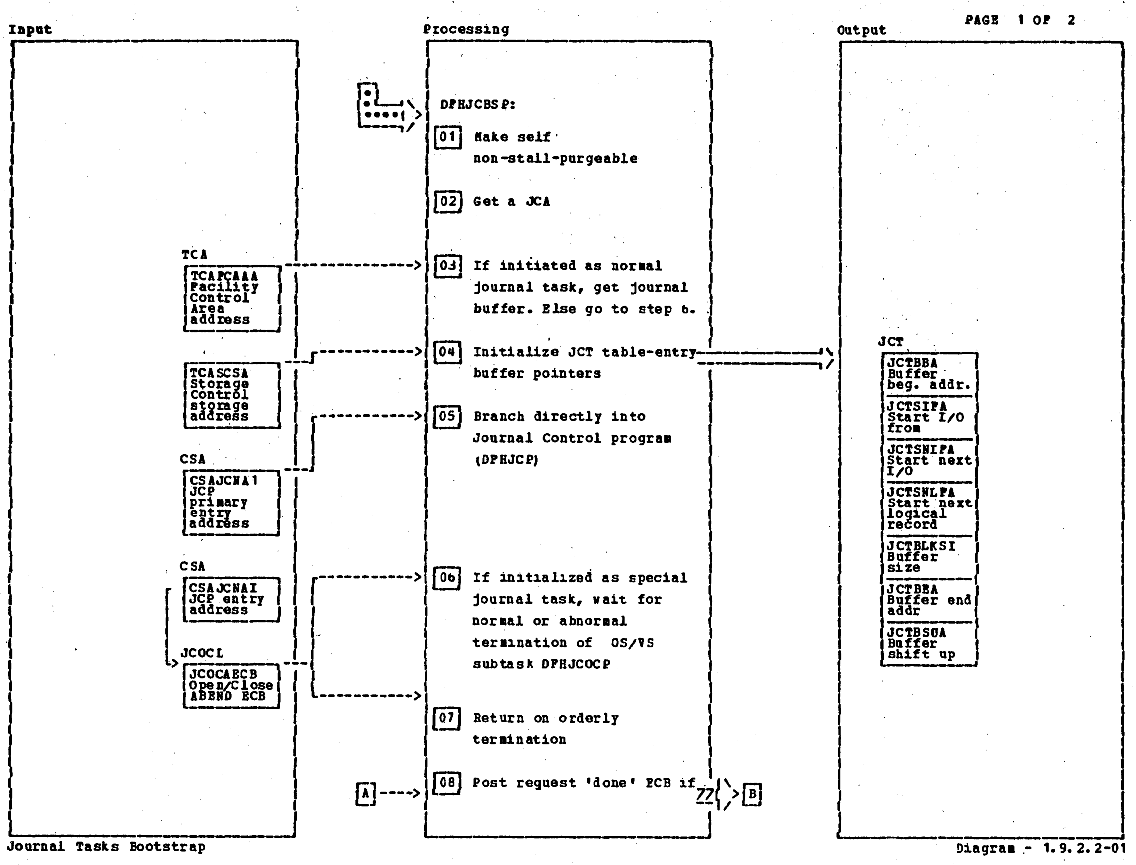
Journal Kick-off

Diagram - 1.9.2.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 10 The message format is: 'DFH4500 - xx of yy journals successfully opened'. | | F | | | | | |

Journal Kick-off

Diagram - 1.9.2.1-02


Input
Processing
Output
PAGE 1 OF 2
TCA
TCAFCAAA Facility Control Area address
TCASCSA Storage Control storage address
CSA
CSAJCNA1 JCP primary entry address
CSA
CSAJCNAI JCP entry address
JCOCL
JCOCAECB Open/Close ABEND ECB
A
B
JCT
JCTBBA Buffer beg. addr.
JCTSIFA Start I/O from
JCTSNIFA Start next I/O
JCTSNLFA Start next logical record
JCTBLKSI Buffer size
JCTBEA Buffer end addr
JCTBSUA Buffer shift up

DFHJCBSP:

01 Make self non-stall-purgeable

02 Get a JCA

03 If initiated as normal journal task, get journal buffer. Else go to step 6.

04 Initialize JCT table-entry buffer pointers

05 Branch directly into Journal Control program (DFHJCP)

06 If initialized as special journal task, wait for normal or abnormal termination of OS/VS subtask DFHJCOCP

07 Return on orderly termination

08 Post request 'done' ECB if

Journal Tasks Bootstrap

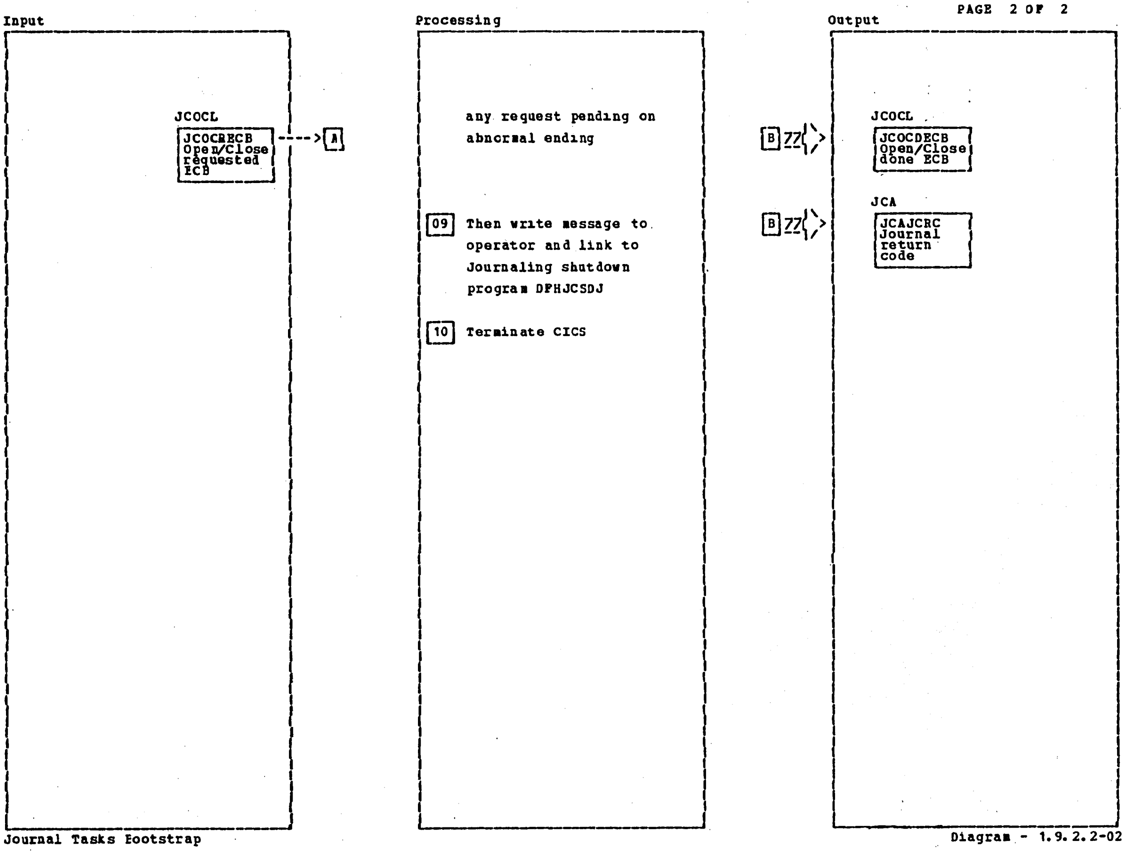
Diagram - 1.9.2.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 Task is initiated as 'normal' if TCAFCAAA is not zero. This field contains the JCT entry (JCTTE) address. | | | | | | | |
| 05 Entry point is on label JC208 in DFHJCP. Control never returns to this program: Journal tasks look in DFHJCP. | | | | | | | |
| Before control is lost, the SIP task (actually, program DFHJCKOJ at this time) is reactivated by posting JCTBALCB: the 'Buffer available' ECB. | | | | | | | |
| 06 Task is initiated as special journal task if TCAFCAAA is zero. | | | | | | | |
| 07 Bit one of byte two in JCOCAECB is zero for orderly termination. | | | | | | | |

Journal Tasks Bootstrap

Diagram - 1.9.2.2-01

PAGE 2 OF 2


Input
Processing
Output
JCOCL
JCOCRECB Open/Close requested ECB
A
B
ZZ
JCOCL
JCOCDECB Open/Close done ECB
B
ZZ
JCA
JCAJCRC Journal return code

any request pending on abnormal ending

09 Then write message to operator and link to Journaling shutdown program DFHJCSDJ

10 Terminate CICS

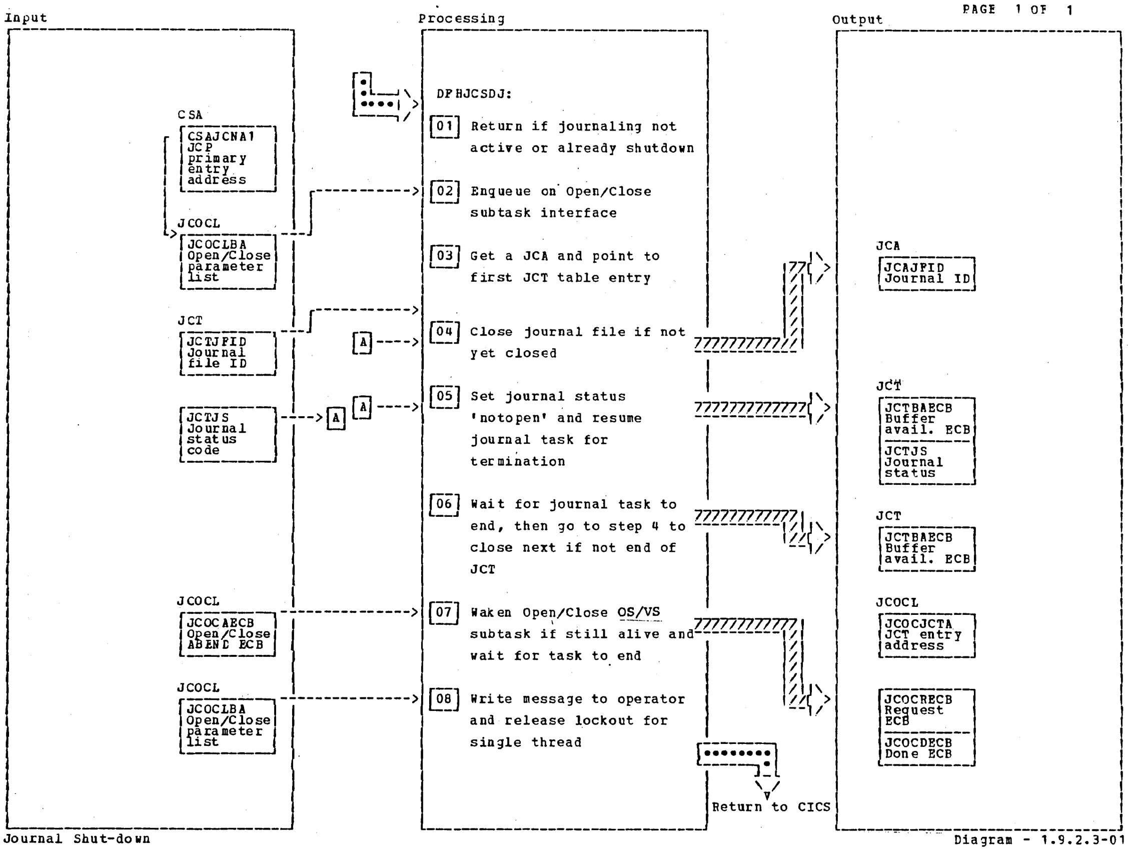
Journal Tasks Bootstrap

Diagram - 1.9.2.2-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 08 A pending journal request is canceled with I/O error return code. | | | | | | | |
| 09 The message format is: DFH454A - Journal Control subtask was abnormally terminated. | | SHUTDOWN | | | | | |
| 10 CICS/VS/OS terminated with a OS/VS DUMP macro instruction. | | | | | | | |

Journal Tasks Bootstrap

Diagram - 1.9.2.2-02


PAGE 1 OF 1
Input
Processing
Output
CSA
CSAJCNA1 JCP primary entry address
JCOCL
JCOCLBA Open/Close parameter list
JCT
JCTJFID Journal file ID
JCTJS Journal status code
A
A
A
JCOCL
JCOCAECB Open/Close ABEND ECB
JCOCL
JCOCLBA Open/Close parameter list
JCA
JCAJFID Journal ID
JCT
JCTBAECB Buffer avail. ECB
JCTJS Journal status
JCT
JCTBAECB Buffer avail. ECB
JCOCL
JCOCJCTA JCT entry address
JCOCRECB Request ECB
JCOCDECB Done ECB
Return to CICS

DFHJCSDJ:

01 Return if journaling not active or already shutdown

02 Enqueue on Open/Close subtask interface

03 Get a JCA and point to first JCT table entry

04 Close journal file if not yet closed

05 Set journal status 'notopen' and resume journal task for termination

06 Wait for journal task to end, then go to step 4 to close next if not end of JCT

07 Waken Open/Close OS/VS subtask if still alive and wait for task to end

08 Write message to operator and release lockout for single thread

Journal Shut-down

Diagram - 1.9.2.3-01

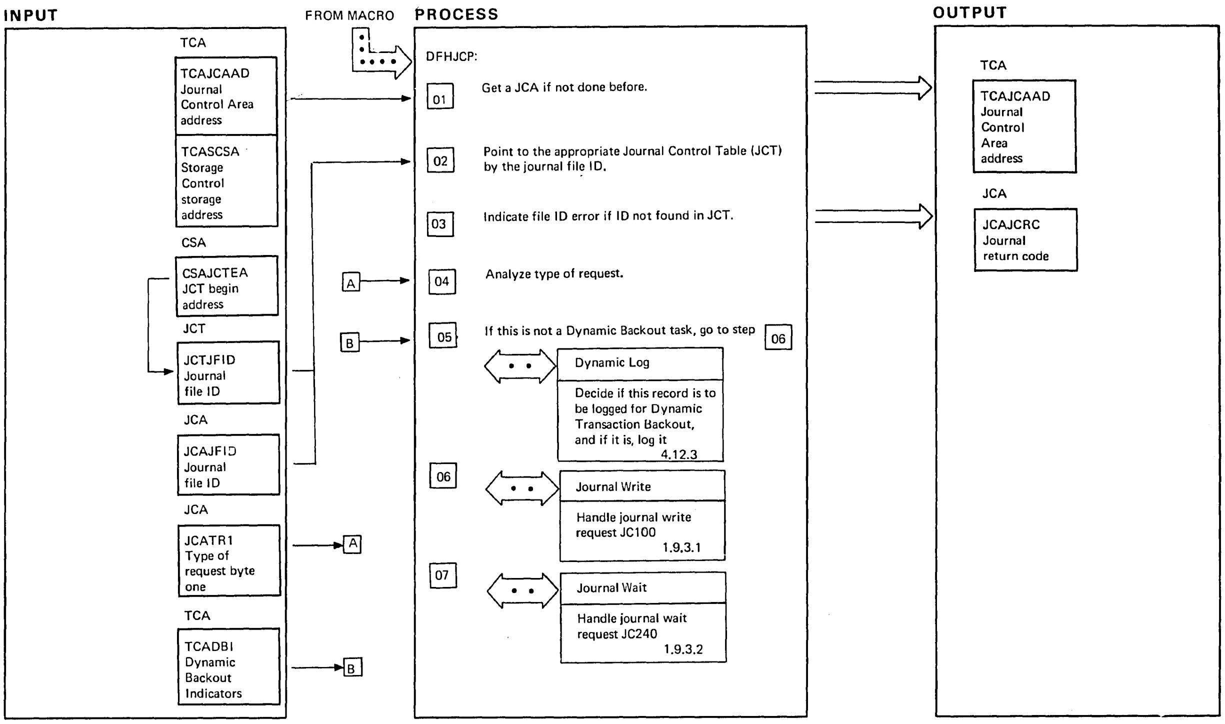
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 02 This ensures no other Open/Close requests or Jcurnaling shutdown can be active simultaneously. | | | |
| 04 If the OS/VS Open/Close subtask (DFHJCOCP) is already detached, an inline CLOSE is issued for the journal file. | | JCLOSE | |
| 06 Buffer available ECB (JCTBAECB) is posted by the terminating journal task, by code in DFHJCP. | | | |
| 07 The OS/VS subtask is forced to end by setting all bytes in JCOCJCTA to X'FF'. | | OSTEND | |
| 08 Message format is: DFH4510 - All open journals now closed. | | MESSAGES | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Journal Shut-down

Diagram - 1.9.2.3-01

Page 1 of 1


INPUT
TCA
TCAJCAAD Journal Control Area address
TCASCSA Storage Control storage address
CSA
CSAJCTEA JCT begin address
JCT
JCTJFID Journal file ID
JCA
JCAJFID Journal file ID
JCA
JCATR1 Type of request byte one
TCA
TCADBI Dynamic Backout Indicators
A
B
FROM MACRO
PROCESS
DFHJCP:
01 Get a JCA if not done before.
02 Point to the appropriate Journal Control Table (JCT) by the journal file ID.
03 Indicate file ID error if ID not found in JCT.
A
04 Analyze type of request.
B
05 If this is not a Dynamic Backout task, go to step 06
Dynamic Log
Decide if this record is to be logged for Dynamic Transaction Backout, and if it is, log it 4.12.3
06
Journal Write
Handle journal write request JC100 1.9.3.1
07
Journal Wait
Handle journal wait request JC240 1.9.3.2
OUTPUT
TCA
TCAJCAAD Journal Control Area address
JCA
JCAJCRC Journal return code

DFHJC In-Line Request Processing

Diagram - 1.9.3-01

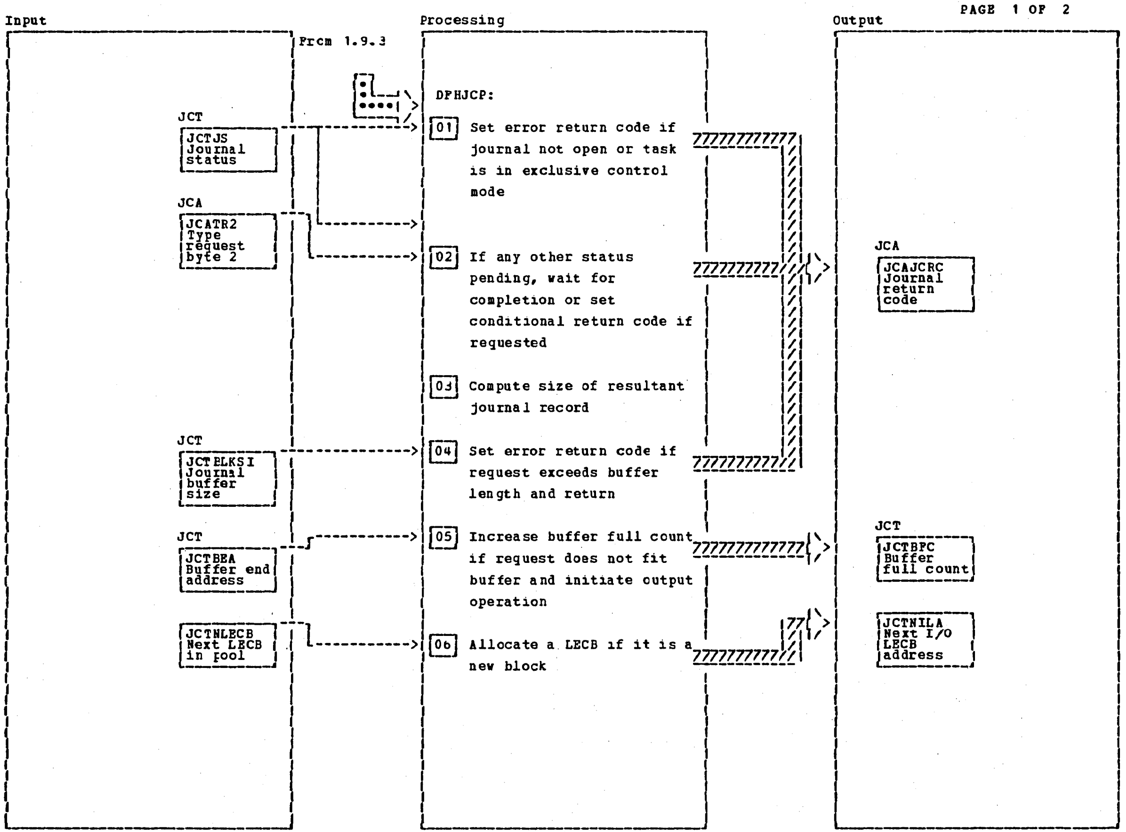
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Return on Register 14 if already got the JCA.<br>User task should be terminated by a Program Interrupt if no JCA was acquired. (No return code possible if no JCA.) | | | |
| 05 If TCADBTT in TCADBI is on and backout has not already started, i.e. TCADBIPM is not on, then Dynamic Transaction Backout Logging maybe indicated. | | JC150 | |
| 06 Put request is identical with a write request followed by a wait request. | | JC100 | |
| 07 For other macro requests, see chart 1.9.5. | | | 1.9.5 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

DFHJC In-Line Request Processing

Diagram - 1.9.3-01

PAGE 1 OF 2


Input
Processing
Output
From 1.9.3
DFHJCP:
JCT
JCTJS Journal status
JCA
JCATR2 Type request byte 2
JCT
JCTBLKSI Journal buffer size
JCT
JCTBEA Buffer end address
JCTNLECB Next LECB in pool
JCA
JCAJCRC Journal return code
JCT
JCTBFC Buffer full count
JCTNILA Next I/O LECB address

01 Set error return code if journal not open or task is in exclusive control mode

02 If any other status pending, wait for completion or set conditional return code if requested

03 Compute size of resultant journal record

04 Set error return code if request exceeds buffer length and return

05 Increase buffer full count if request does not fit buffer and initiate output operation

06 Allocate a LECB if it is a new block

Journal Write Request

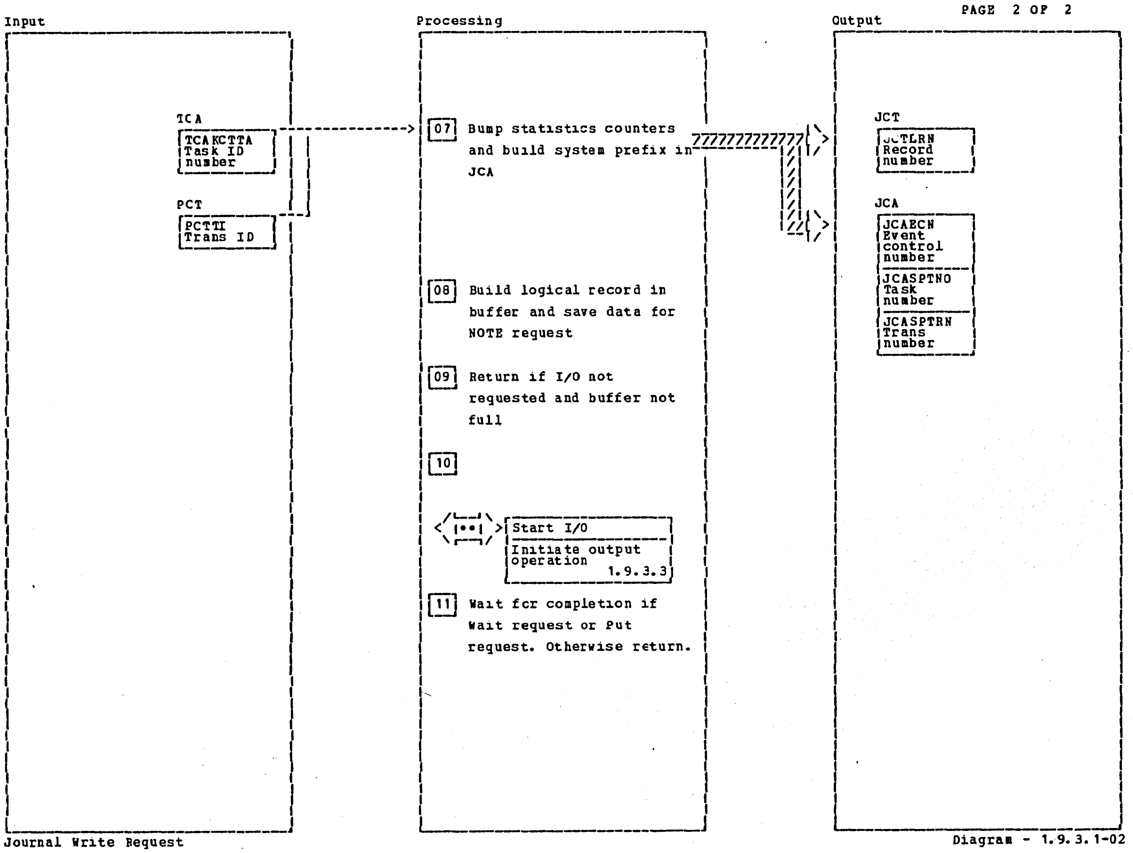
Diagram - 1.9.3.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 Record size depends on whether or not automatic journaling and user-prefix are supported. | | JC150 | | | | | |
| 05 Request is handled again if enough buffer available. | | JC160 | | | | | |

Journal Write Request

Diagram - 1.9.3.1-01

PAGE 2 OF 2



07 Bump statistics counters and build system prefix in JCA

08 Build logical record in buffer and save data for NOTE request

09 Return if I/O not requested and buffer not full

10

11 Wait for completion if Wait request or Put request. Otherwise return.

Journal Write Request

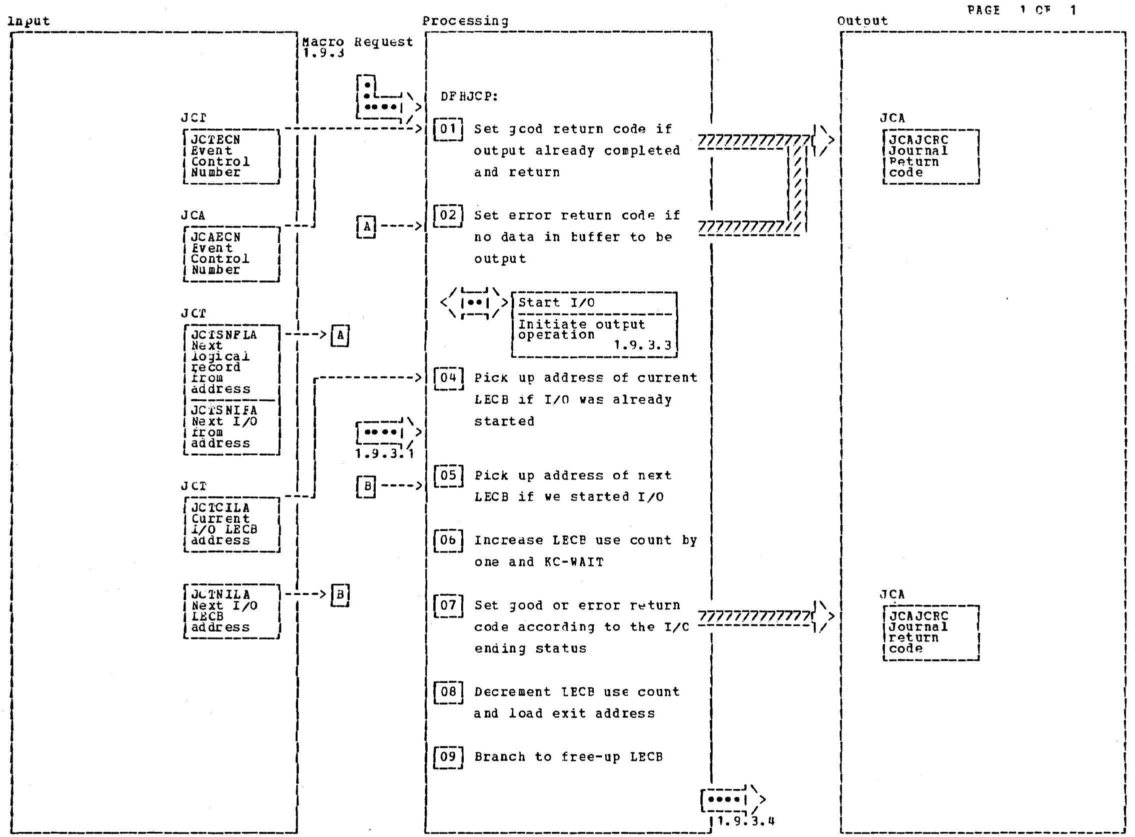
Diagram - 1.9.3.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 08 The record contains the following information<br>JCA SYS-PREFIX<br>RECORD ID<br>SYS-PREFIX length<br>User prefix<br>User data |  | JC214 |  |  |  |  |  |

Journal Write Request

Diagram - 1.9.3.1-02

PAGE 1 OF 1



Input

Macro Request 1.9.3

JCT
JCTECN Event Control Number

JCA
JCAECN Event Control Number

JCT
JCTSNFLA Next logical record from address
JCTSNIFA Next I/O from address

1.9.3.1

JCT
JCTCILA Current I/O LECB address

JCTNILA Next I/O LECB address

Processing

DFHJCP:

01 Set good return code if output already completed and return

02 Set error return code if no data in buffer to be output

Start I/O
Initiate output operation 1.9.3.3

04 Pick up address of current LECB if I/O was already started

05 Pick up address of next LECB if we started I/O

06 Increase LECB use count by one and KC-WAIT

07 Set good or error return code according to the I/O ending status

08 Decrement LECB use count and load exit address

09 Branch to free-up LECB

1.9.3.4

Outout

JCA
JCAJCRC Journal Return code

JCA
JCAJCRC Journal return code

Journal Wait Request

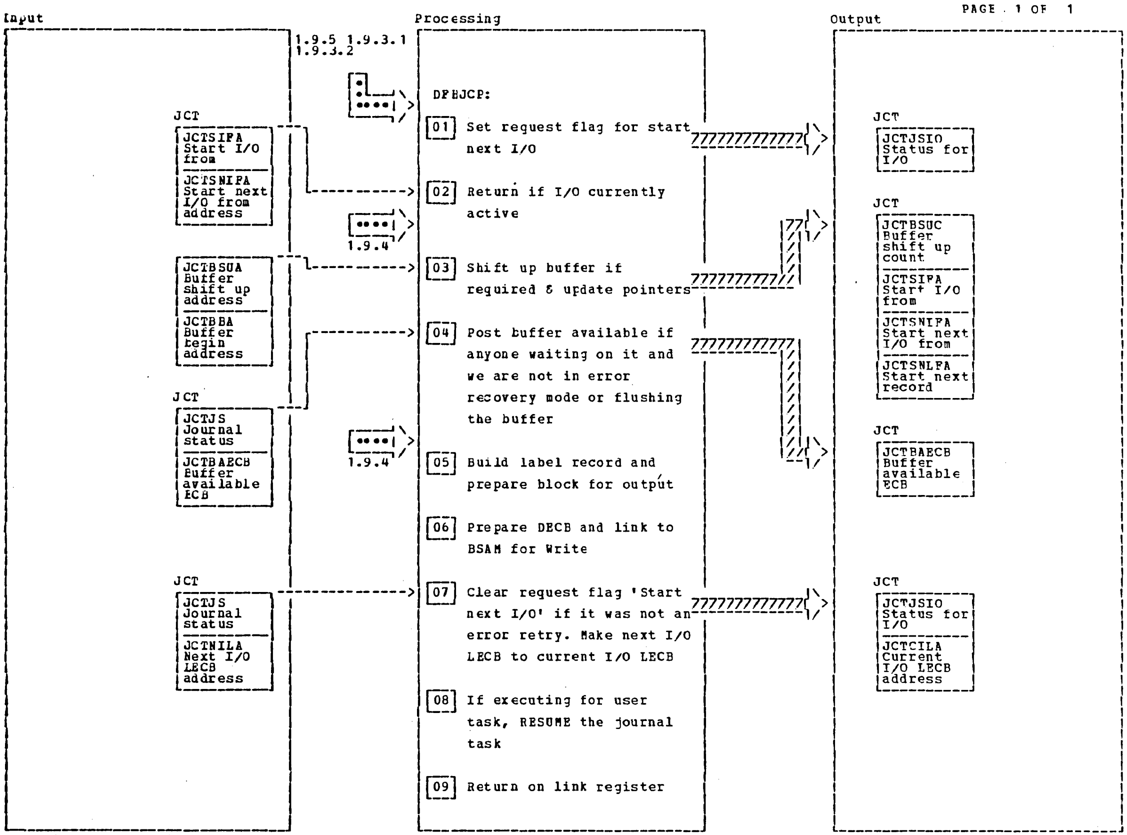
Diagram - 1.9.3.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Block has been written if event control number in JCT and JCA are equal. | | JC232 | | | | | |
| 05 Entry point in wait routine if PUT macro request. | | JC254 | | | | | |
| 06 | | | 1.9.3.4 | | | | |
| 07 The error is indicated in the LECB. | | | | | | | |
| 08 | | | 1.9.3.4 | | | | |

Journal Wait Request

Diagram - 1.9.3.2-01

PAGE 1 OF 1


Input
Processing
Output
1.9.5 1.9.3.1
1.9.3.2
1.9.4
1.9.4
JCT
JCTSIFA Start I/O from
JCTSNIFA Start next I/O from address
JCTBSUA Buffer shift up address
JCTBBA Buffer begin address
JCT
JCTJS Journal status
JCTBAECB Buffer available ECB
JCT
JCTJS Journal status
JCTNILA Next I/O LECB address
JCT
JCTJSIO Status for I/O
JCT
JCTBSUC Buffer shift up count
JCTSIFA Start I/O from
JCTSNIFA Start next I/O from
JCTSNLFA Start next record
JCT
JCTBAECB Buffer available ECB
JCT
JCTJSIO Status for I/O
JCTCILA Current I/O LECB address

DFHJCP:

01 Set request flag for start next I/O

02 Return if I/O currently active

03 Shift up buffer if required & update pointers

04 Post buffer available if anyone waiting on it and we are not in error recovery mode or flushing the buffer

05 Build label record and prepare block for output

06 Prepare DECB and link to BSAM for Write

07 Clear request flag 'Start next I/O' if it was not an error retry. Make next I/O LECB to current I/O LECB

08 If executing for user task, RESUME the journal task

09 Return on link register

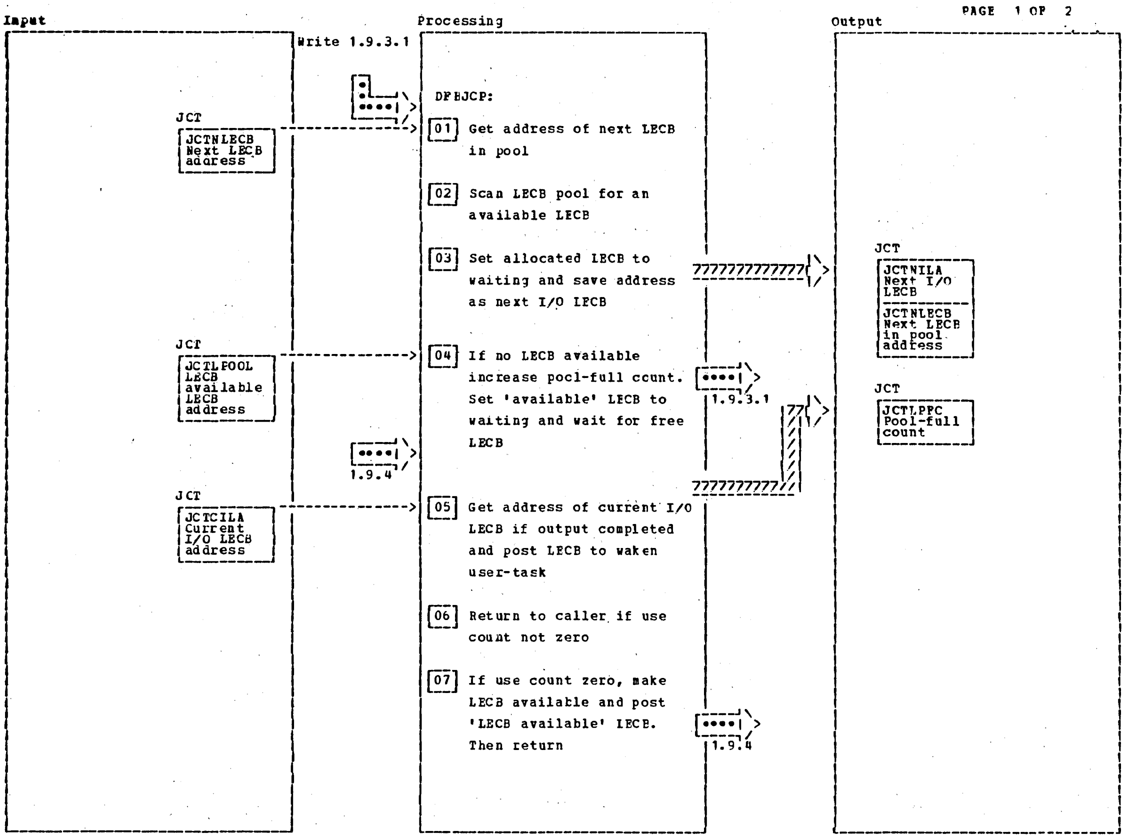
Journal Output Initiation

Diagram - 1.9.3.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 The record format is variable and preceded by 'LLbb'. | | JC263 | | | | | |

Journal Output Initiation

Diagram - 1.9.3.3-01


PAGE 1 OF 2
Input
Processing
Output
Write 1.9.3.1
JCT
JCTNLECB Next LECB address
JCT
JCTLPOOL LECB available LECB address
JCT
JCTCILA Current I/O LECB address
1.9.4
1.9.3.1
1.9.4
JCT
JCTNILA Next I/O LECB
JCTNLECB Next LECB in pool address
JCT
JCTLPFC Pool-full count

DFHJCP:

01 Get address of next LECB in pool

02 Scan LECB pool for an available LECB

03 Set allocated LECB to waiting and save address as next I/O LECB

04 If no LECB available increase pool-full count. Set 'available' LECB to waiting and wait for free LECB

05 Get address of current I/O LECB if output completed and post LECB to waken user-task

06 Return to caller if use count not zero

07 If use count zero, make LECB available and post 'LECB available' LECB. Then return

Logical Event Control (LECB)

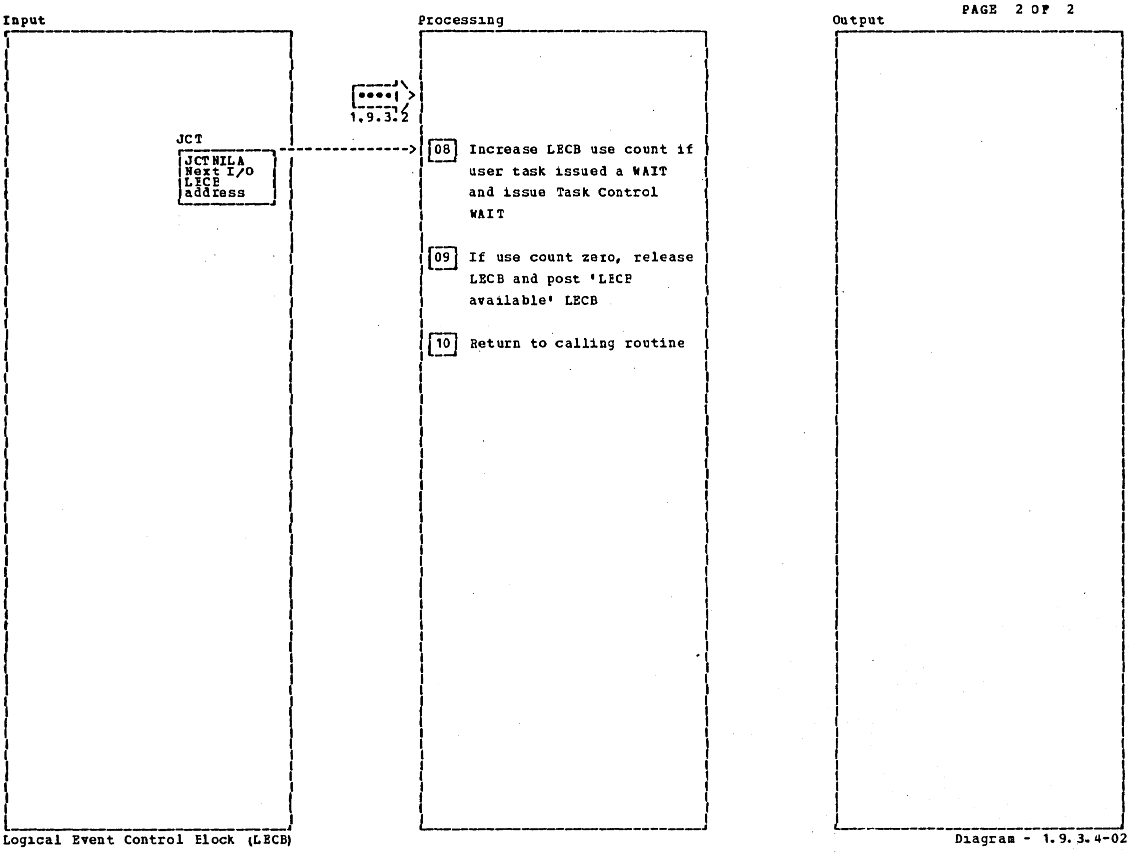
HIPOMAT 1.1 Diagram - 1.9.3.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The number of LECB's is three times the number of journals plus three.<br>Meaning of bits:<br>0 = Wait bit<br>1 = Completion bit<br>2 = I/O error<br>3-7 = Use count, i.e., count of user tasks waiting on the associated block. | | JC180 | | | | | |
| 04 The first LECB in pool is used as 'LECB available' LECB. It is posted available as soon as an LECB is available. No wait is issued on conditional return. | | JC190 | | | | | |
| 05 The journal task gets control for output completion processing. | | JC286 | | | | | |

Logical Event Control (LECB)

Diagram - 1.9.3.4-01

PAGE 2 OF 2



08 Increase LECB use count if user task issued a WAIT and issue Task Control WAIT

09 If use count zero, release LECB and post 'LECP available' LECB

10 Return to calling routine

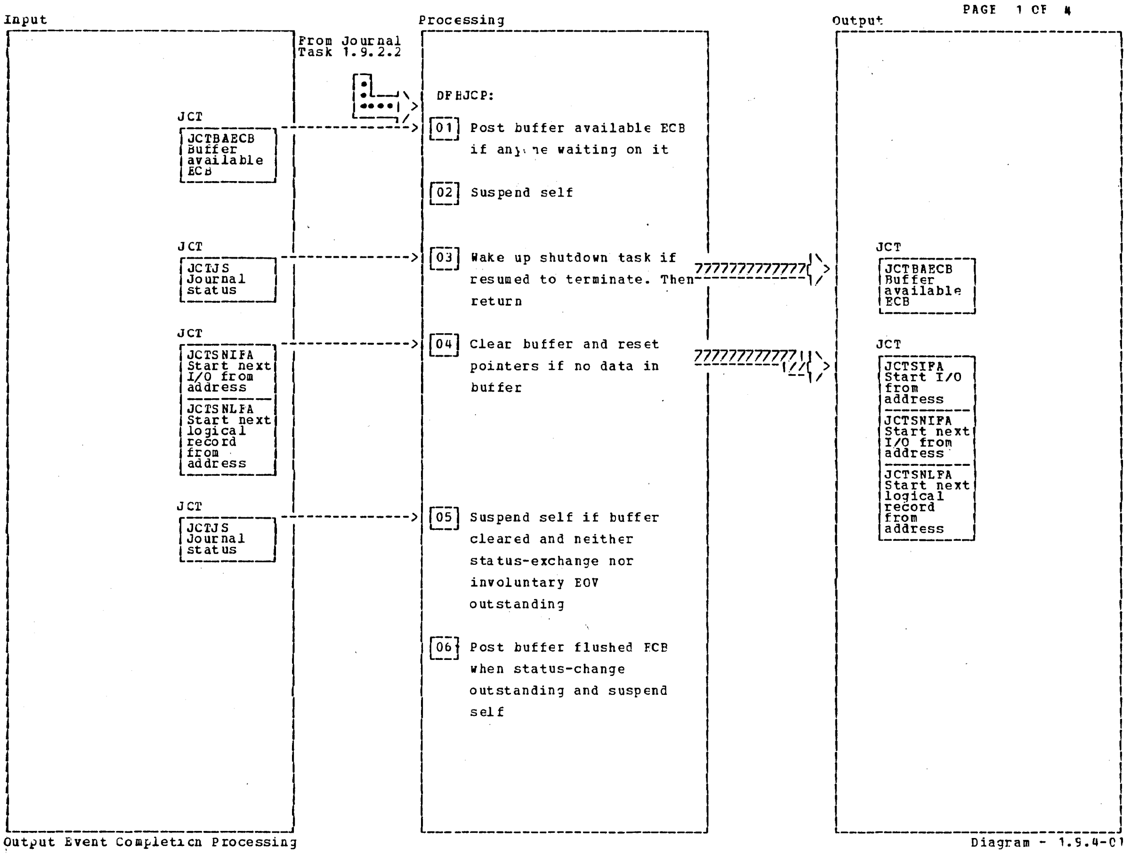
Logical Event Control Block (LECB)

Diagram - 1.9.3.4-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Logical Event Control Block (LECB)

Diagram - 1.9.3.4-02


PAGE 1 OF 4
Input
Processing
Output
From Journal Task 1.9.2.2
JCT
JCTBAECB Buffer available ECB
JCT
JCTJS Journal status
JCT
JCTSNIFA Start next I/O from address
JCTSNLFA Start next logical record from address
JCT
JCTJS Journal status
JCT
JCTBAECB Buffer available ECB
JCT
JCTSIFA Start I/O from address
JCTSNIFA Start next I/O from address
JCTSNLFA Start next logical record from address

DFHJCP:

01 Post buffer available ECB if any ne waiting on it

02 Suspend self

03 Wake up shutdown task if resumed to terminate. Then return

04 Clear buffer and reset pointers if no data in buffer

05 Suspend self if buffer cleared and neither status-exchange nor involuntary EOV outstanding

06 Post buffer flushed ECB when status-change outstanding and suspend self

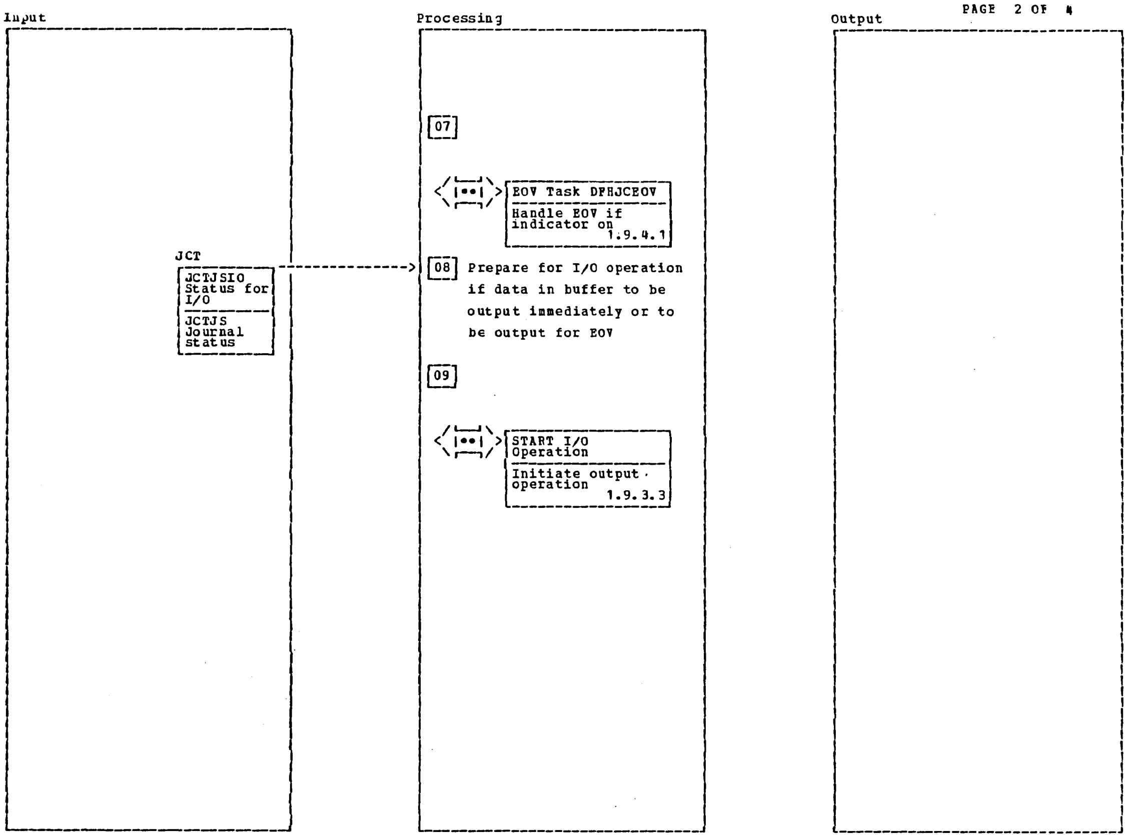
Output Event Completion Processing

Diagram - 1.9.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 Buffer is empty if 'Start next I/O from address' and 'Start next logical record from address' are equal. | | JC274 | | | | | |

Output Event Completion Processing

Diagram - 1.9.4-01



08 Prepare for I/O operation if data in buffer to be output immediately or to be output for EOV

Output Event Completicn Processing

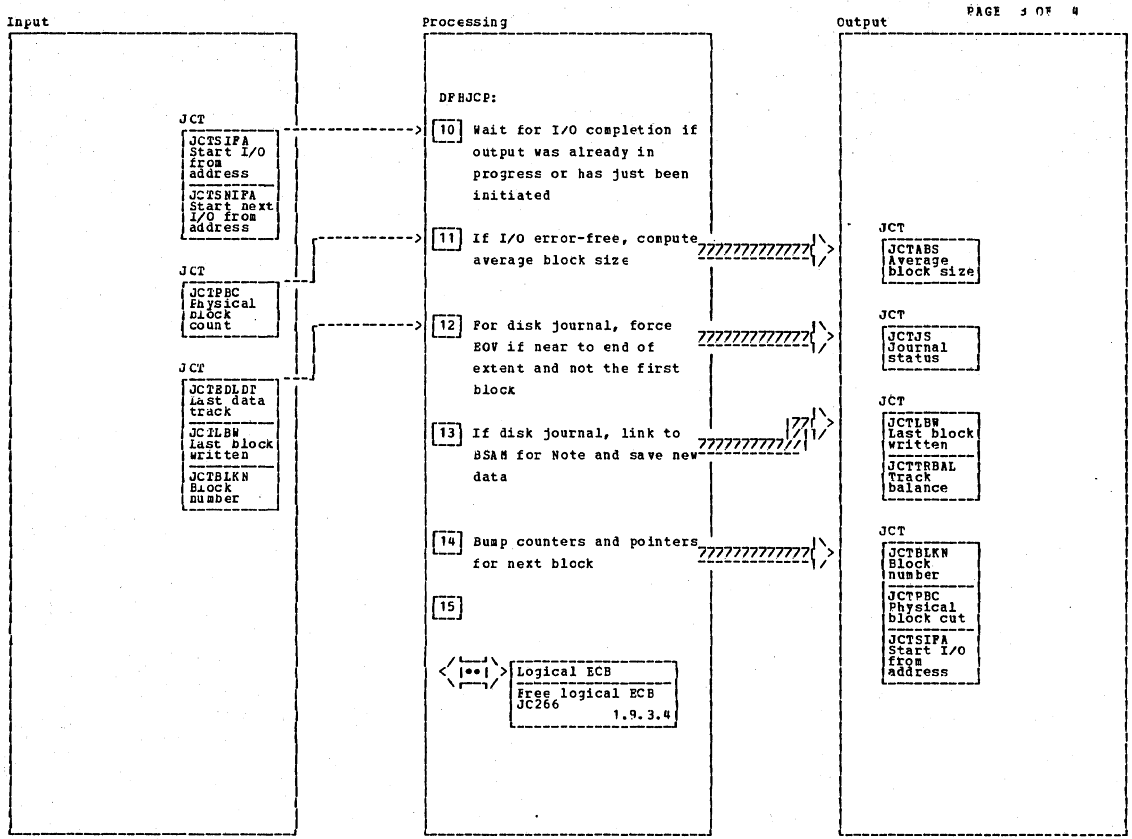
Diagram - 1.9.4-02

| Notes | Routine | Label | Ref | Notes | Routine | Latel | Ref |
|---|---|---|---|---|---|---|---|

Output Event Completicn Processing

Diagram - 1.9.4-02

PAGE 3 OF 4


Input

JCT

JCTSIFA Start I/O from address

JCTSNIFA Start next I/O from address

JCT

JCTPBC Physical block count

JCT

JCTEDLDT Last data track

JCTLBW Last block written

JCTBLKN Block number

Processing

DFHJCP:

10 Wait for I/O completion if output was already in progress or has just been initiated

11 If I/O error-free, compute average block size

12 For disk journal, force EOV if near to end of extent and not the first block

13 If disk journal, link to BSAM for Note and save new data

14 Bump counters and pointers for next block

15

Logical ECB

Free logical ECB JC266 1.9.3.4

Output

JCT

JCTABS Average block size

JCT

JCTJS Journal status

JCT

JCTLBW Last block written

JCTTRBAL Track balance

JCT

JCTBLKN Block number

JCTPBC Physical block cut

JCTSIFA Start I/O from address

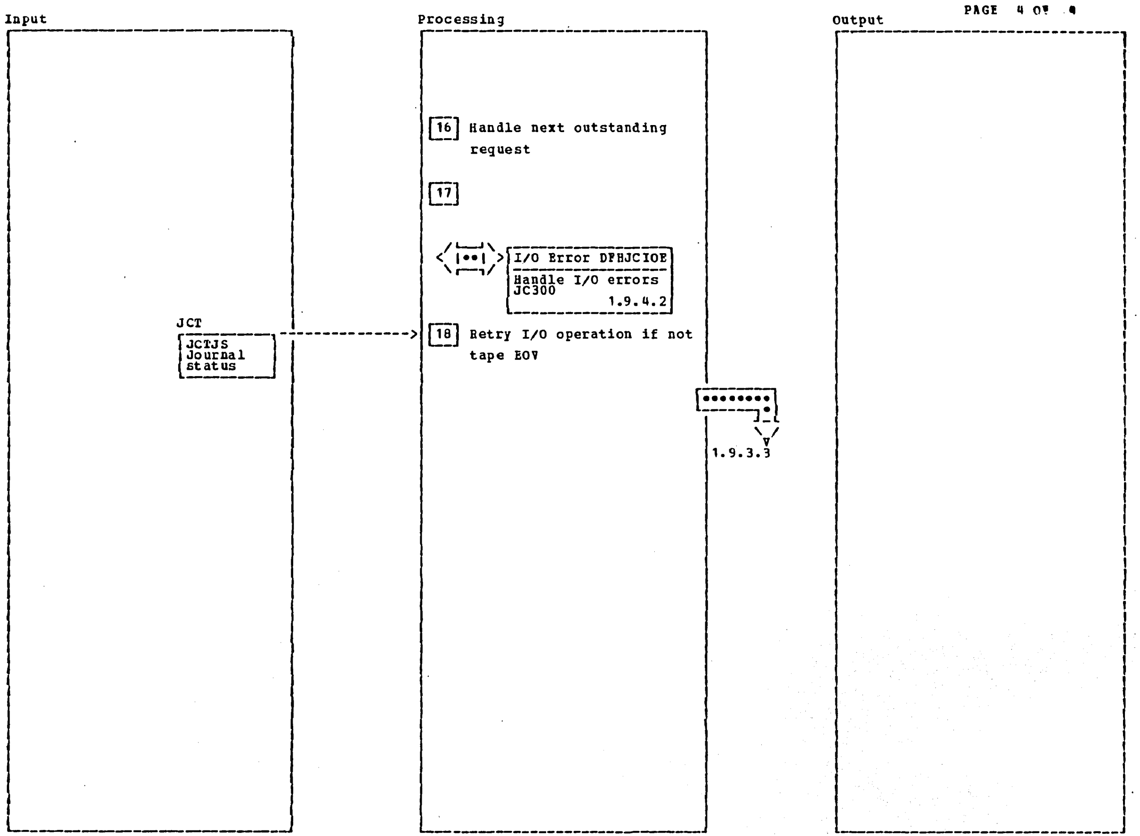
Output Event Completion Processing

Diagram - 1.9.4-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 10 An output operation is in progress if 'Start I/C from address' and 'Start next I/O from address' are not equal. | | JC272 | | | | | |
| 14 'Start next I/O from address' is moved to the field 'Start I/O from address'. | | JC286 | | | | | |

Output Event Completion Processing

Diagram - 1.9.4-03



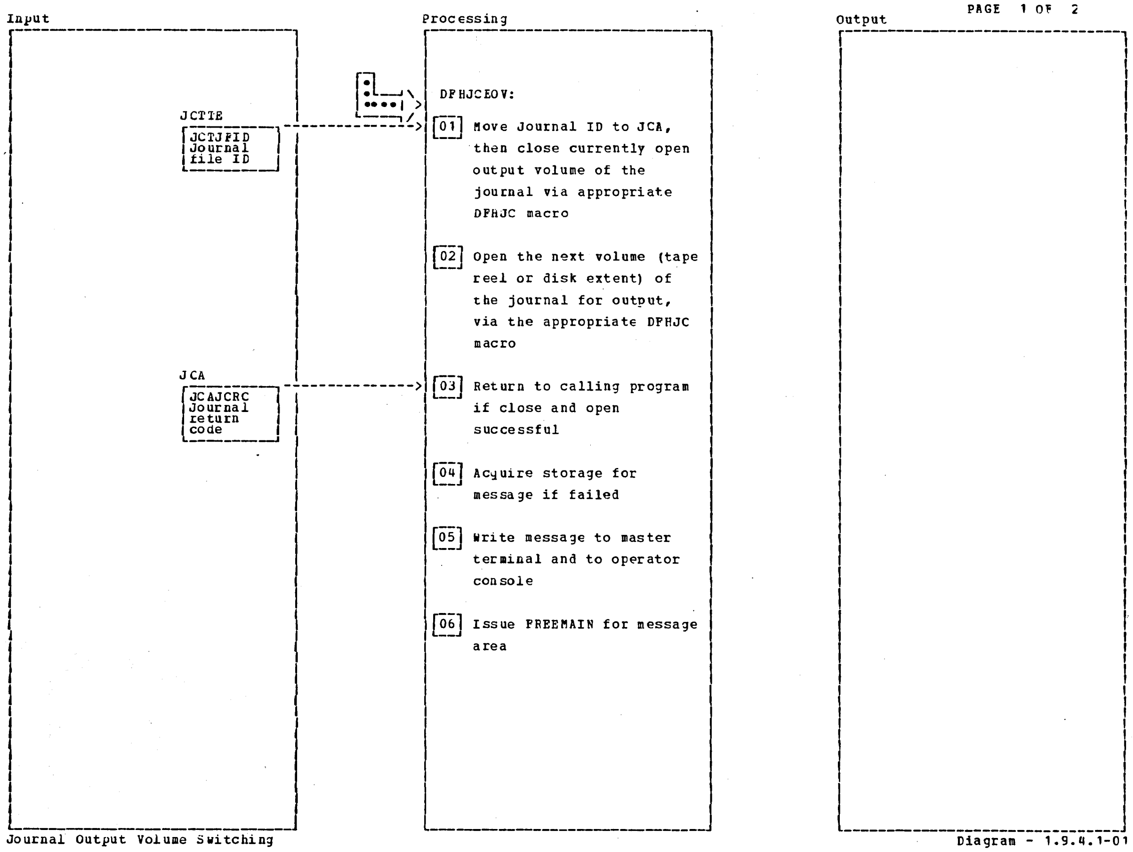
Output Event Completion Processing

Diagram - 1.9.4-04

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 18 Since end of reel on tape can't be predicted by the program, the I/O error can be caused by end of tape. In this case EOV is forced. | | | | | | | |

Output Event Completion Processing

Diagram - 1.9.4-04



DFHJCEOV:

01 Move Journal ID to JCA, then close currently open output volume of the journal via appropriate DFHJC macro

02 Open the next volume (tape reel or disk extent) of the journal for output, via the appropriate DFHJC macro

03 Return to calling program if close and open successful

04 Acquire storage for message if failed

05 Write message to master terminal and to operator console

06 Issue FREEMAIN for message area

Journal Output Volume Switching

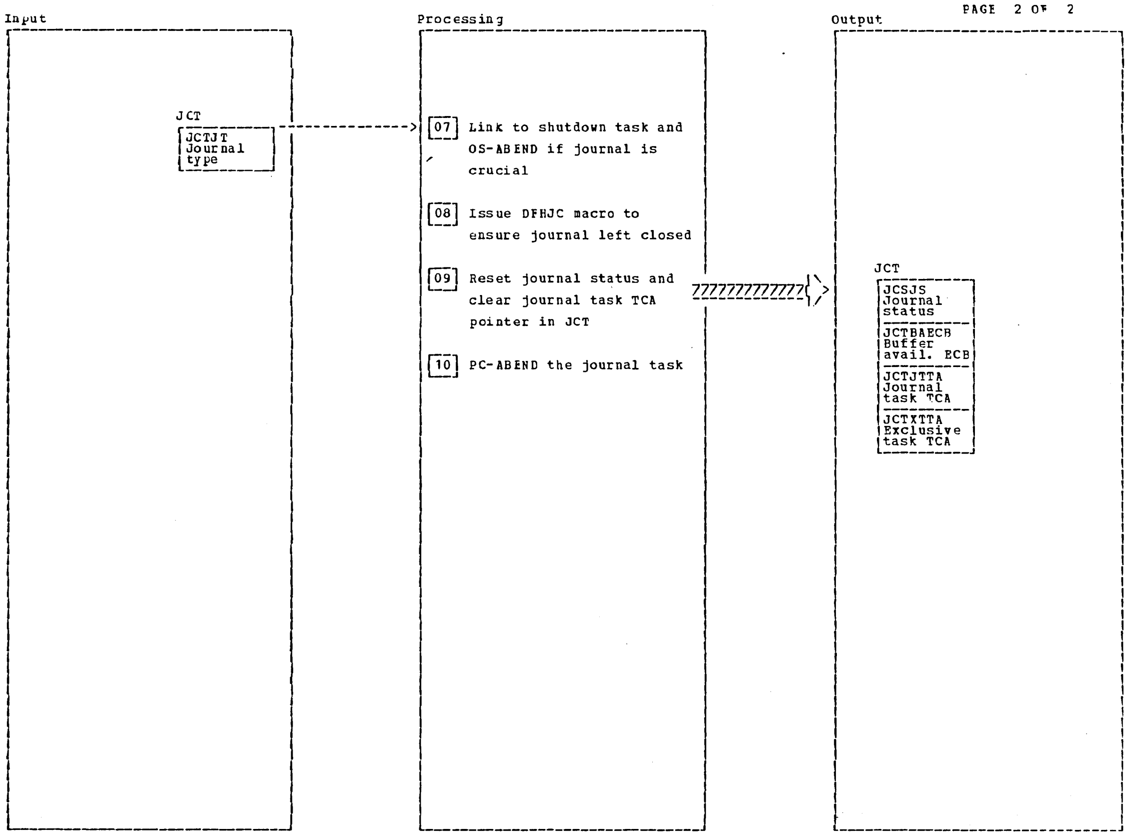
Diagram - 1.9.4.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 JCT-entry address is the Facility Control Address (field TCAFCAAA) of the invoking Journal task. | | | 1.9.2.1 | | | | |
| Note that the JCA - required for issuing DFHJC macros - was acquired for journal tasks when initialized. | | | 1.9.2.3 | | | | |
| 03 Calling program is actually mainline journal Control program (DFHJCP) executing for the Journal task which had detected output EOV. | | | | | | | |
| 05 Message format is: DFH452A - CICS/VS Journal no longer available - output volume-switch failure | | JC120 | | | | | |

Journal Output Volume Switching

Diagram - 1.9.4.1-01

PAGE 2 OF 2


Input
JCT
JCTJT Journal type
Processing
07 Link to shutdown task and OS-ABEND if journal is crucial
08 Issue DFHJC macro to ensure journal left closed
09 Reset journal status and clear journal task TCA pointer in JCT
10 PC-ABEND the journal task
Output
JCT
JCSJS Journal status
JCTBAECB Buffer avail. ECB
JCTJTTA Journal task TCA
JCTXTTA Exclusive task TCA

Journal Output Volume Switching

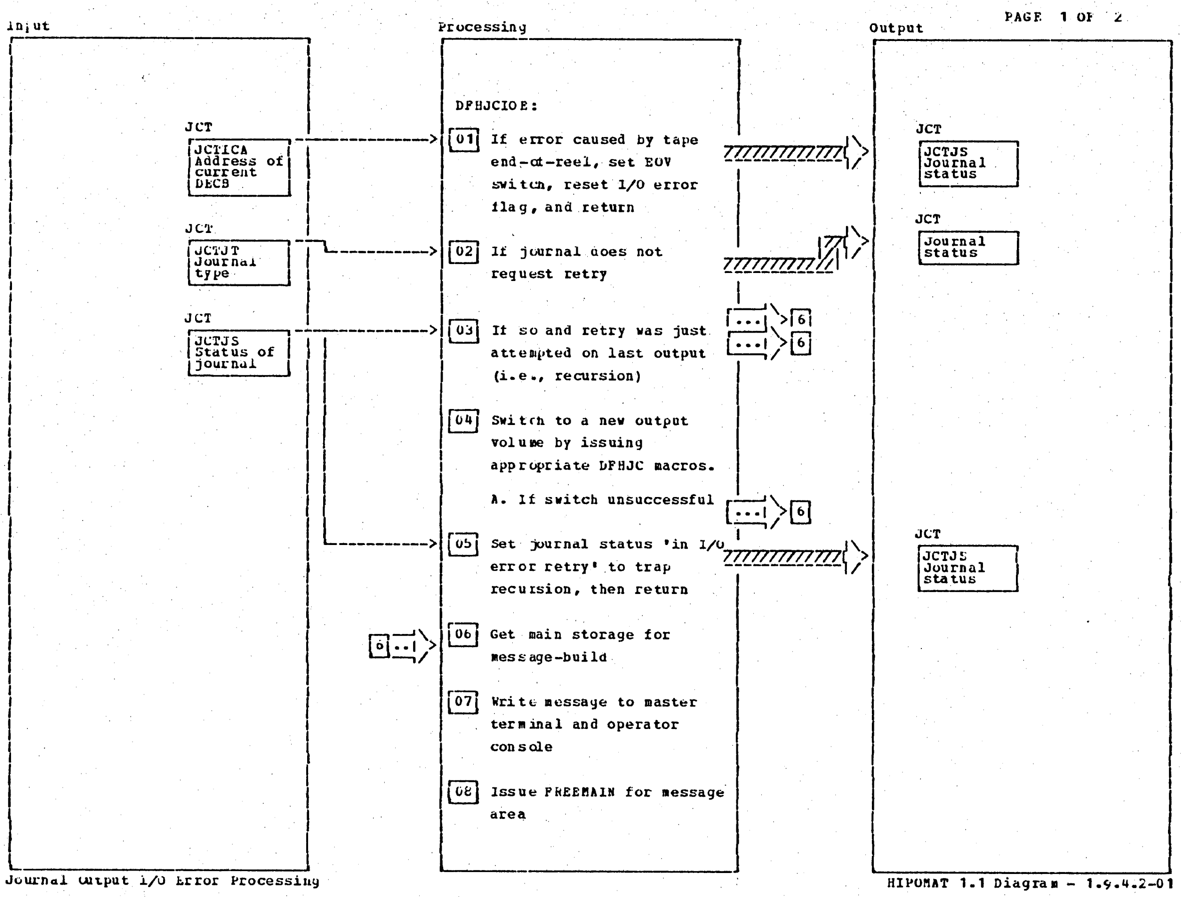
Diagram - 1.9.4.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 07 OS ABEND code is 452 | | | 1.9.2.3 | | | | |
| 08 | | | 1.9.5.2 | | | | |
| 10 CICS/VS ABEND code is AJCB | | JC140 | | | | | |

Journal Output Volume Switching

Diagram - 1.9.4.1-02

PAGE 1 OF 2



DFHJCIOE:

01 If error caused by tape end-of-reel, set EOV switch, reset I/O error flag, and return

02 If journal does not request retry

03 If so and retry was just attempted on last output (i.e., recursion)

04 Switch to a new output volume by issuing appropriate DFHJC macros.

A. If switch unsuccessful

05 Set journal status 'in I/O error retry' to trap recursion, then return

06 Get main storage for message-build

07 Write message to master terminal and operator console

08 Issue FREEMAIN for message area

Journal Output I/O Error Processing

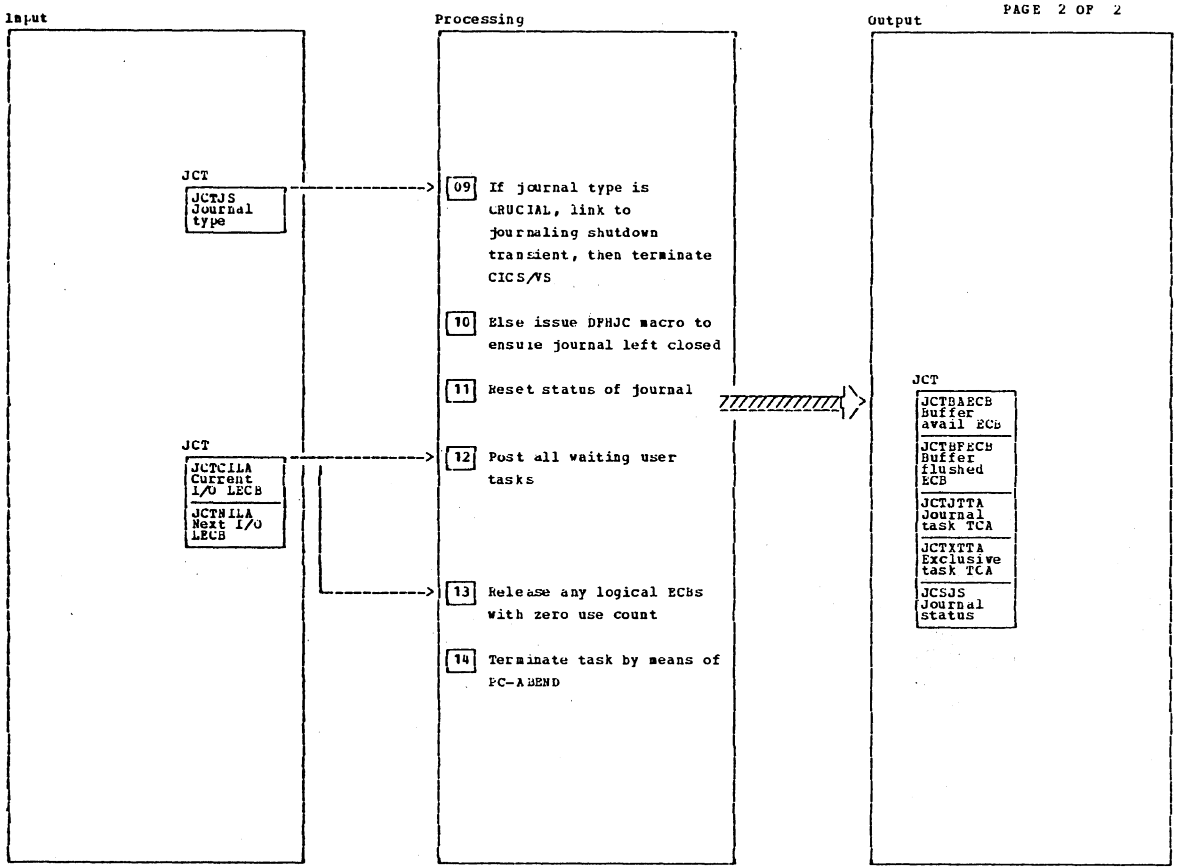
HIPOMAT 1.1 Diagram - 1.9.4.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Return is to mainline DFHJCP, to the section which performs output I/O completion (under journal task TCAs only) . | | | | | | | |
| 02 Byte JCTJT set to JCTJSR for journals generated with JOUROPT=RETRY in JCT. | | JC20 | | | | | |
| 03 Byte JCTJS bit JCTJCIOE is zeroed after every successful journal output, and is only set on later in this program (see step 5) to cause I/O retry by DFHJCP. | | | | | | | |
| 05 Return is to mainline DFHJCP, to the section which performs output I/O completion (under journal task TCAs only). | | | | | | | |
| 06 Enter this routine if output I/O error considered permanent. | | JC100 | | | | | |
| 07 Message format is: DFH453A - CICS/VS Journal no longer available - output I/O error. | | JC120 | | | | | |

Journal Output I/O Error Processing

HIPOMAT 1.1 Diagram - 1.9.4.2-01

PAGE 2 OF 2



09 If journal type is CRUCIAL, link to journaling shutdown transient, then terminate CICS/VS

10 Else issue DFHJC macro to ensure journal left closed

11 Reset status of journal

12 Post all waiting user tasks

13 Release any logical ECBs with zero use count

14 Terminate task by means of PC-ABEND

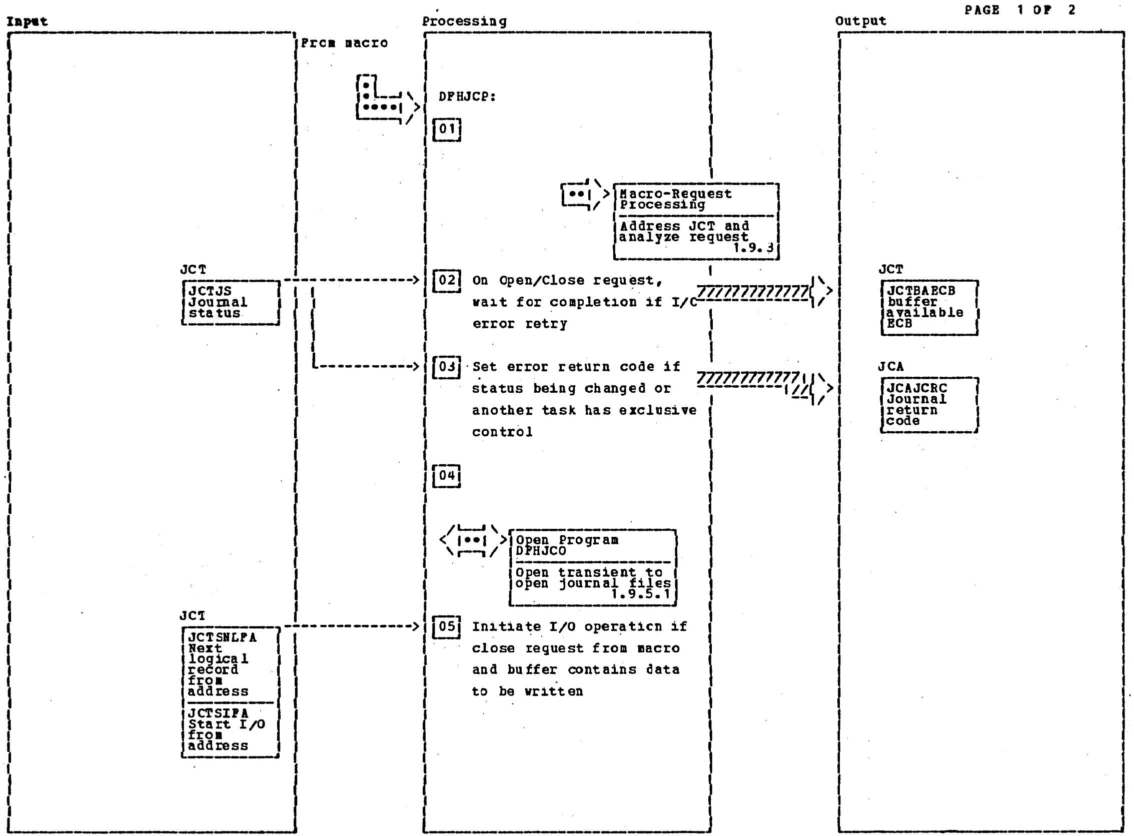
Journal Output I/O Error Processing

HIPOMAT 1.1 Diagram - 1.9.4.2-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 12 JCTCILA and JCTNILA point at LECBs associated with current block (which incurred the I/O error), and next block to be written to the journal. | | | | | | | |
| 13 If use count is not zero the LECB is posted with I/O error. | | JC290 | 1.9.3.4 | | | | |
| 14 Journal task's CICS/VS ABEND code is AJCA | | | | | | | |

Journal Output I/O Error Processing

HIPOMAT 1.1 Diagram - 1.9.4.2-02


PAGE 1 OF 2
Input
Processing
Output
Frcm macro
DFHJCP:
01
Macro-Request Processing
Address JCT and analyze request 1.9.3
JCT
JCTJS Journal status
JCT
JCTBAECB buffer available ECB
JCA
JCAJCRC Journal return code
04
Open Program DFHJCO
Open transient to open journal files 1.9.5.1
JCT
JCTSNLFA Next logical record from address
JCTSIFA Start I/O from address

02 On Open/Close request, wait for completion if I/O error retry

03 Set error return code if status being changed or another task has exclusive control

05 Initiate I/O operaticn if close request from macro and buffer contains data to be written

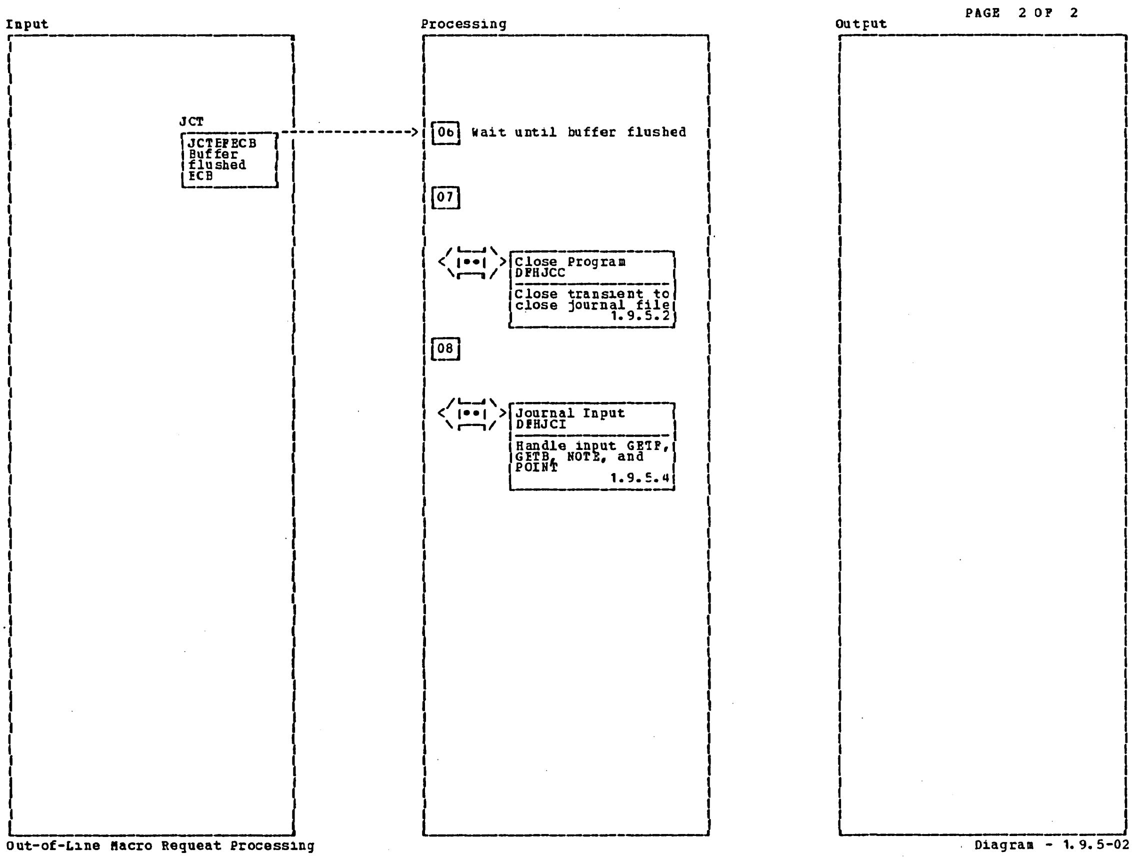
Out-of-Line Macro Request Processing

Diagram - 1.9.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 Journal is in change status if Program Control link is issued to the fcllowing tasks: DFHJCC, DFHJCO, DFHJCIOE and DFHJCEOV.<br>Journal is in transfer-control mode if the file is closed by a task or SIP has control during open for VOLUME=FIRST. | | | | | | | |
| 05 | | | 1.9.3.3 | | | | |

Out-of-Line Macro Request Processing

Diagram - 1.9.5-01

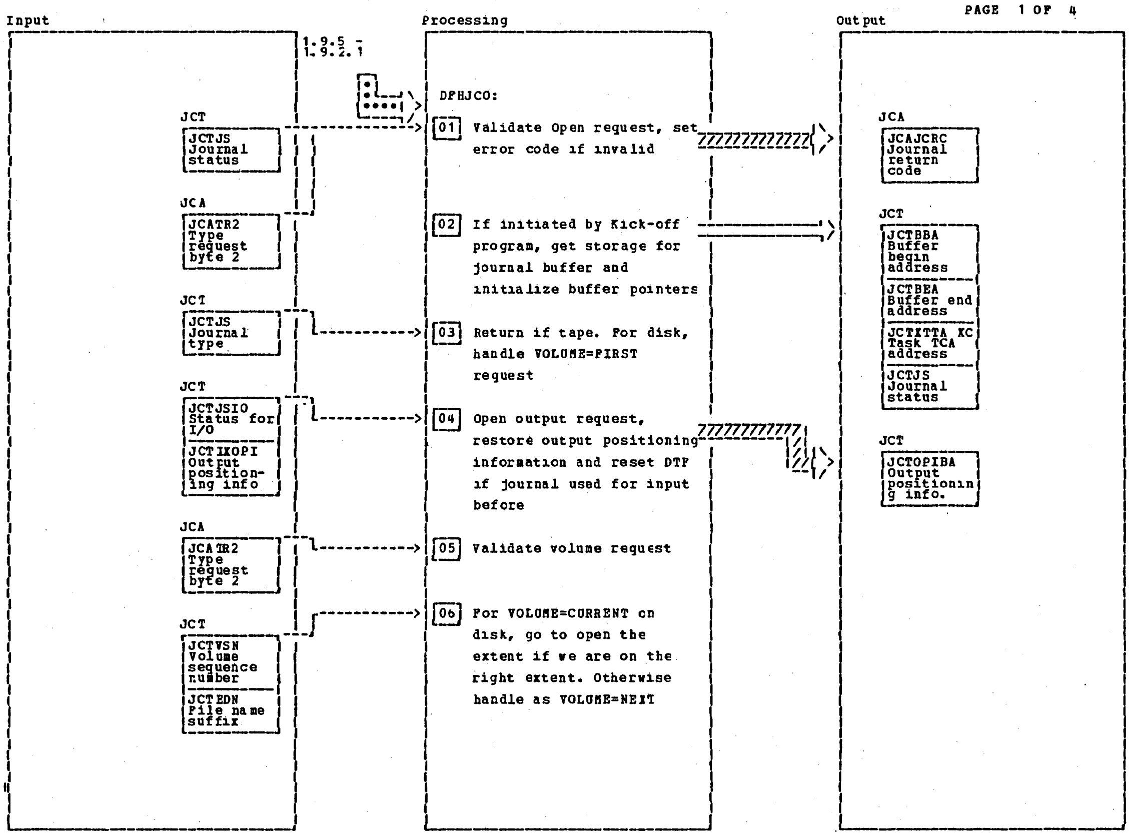


Out-of-Line Macro Requeat Processing Diagram - 1.9.5-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Out-of-Line Macro Requeat Processing Diagram - 1.9.5-02

PAGE 1 OF 4


Input
Processing
Output
1.9.5 -
1.9.2.1
JCT
JCTJS Journal status
JCA
JCATR2 Type request byte 2
JCT
JCTJS Journal type
JCT
JCTJSIO Status for I/O
JCTIXOPI Output position-ing info
JCA
JCATR2 Type request byte 2
JCT
JCTVSN Volume sequence number
JCTEDN File name suffix
JCA
JCAJCRC Journal return code
JCT
JCTBBA Buffer begin address
JCTBEA Buffer end address
JCTXTTA KC Task TCA address
JCTJS Journal status
JCT
JCTOPIBA Output positionin g info.

DFHJCO:

01 Validate Open request, set error code if invalid

02 If initiated by Kick-off program, get storage for journal buffer and initialize buffer pointers

03 Return if tape. For disk, handle VOLUME=FIRST request

04 Open output request, restore output positioning information and reset DTF if journal used for input before

05 Validate volume request

06 For VOLUME=CURRENT cn disk, go to open the extent if we are on the right extent. Otherwise handle as VOLUME=NEXT

Journal Open Request

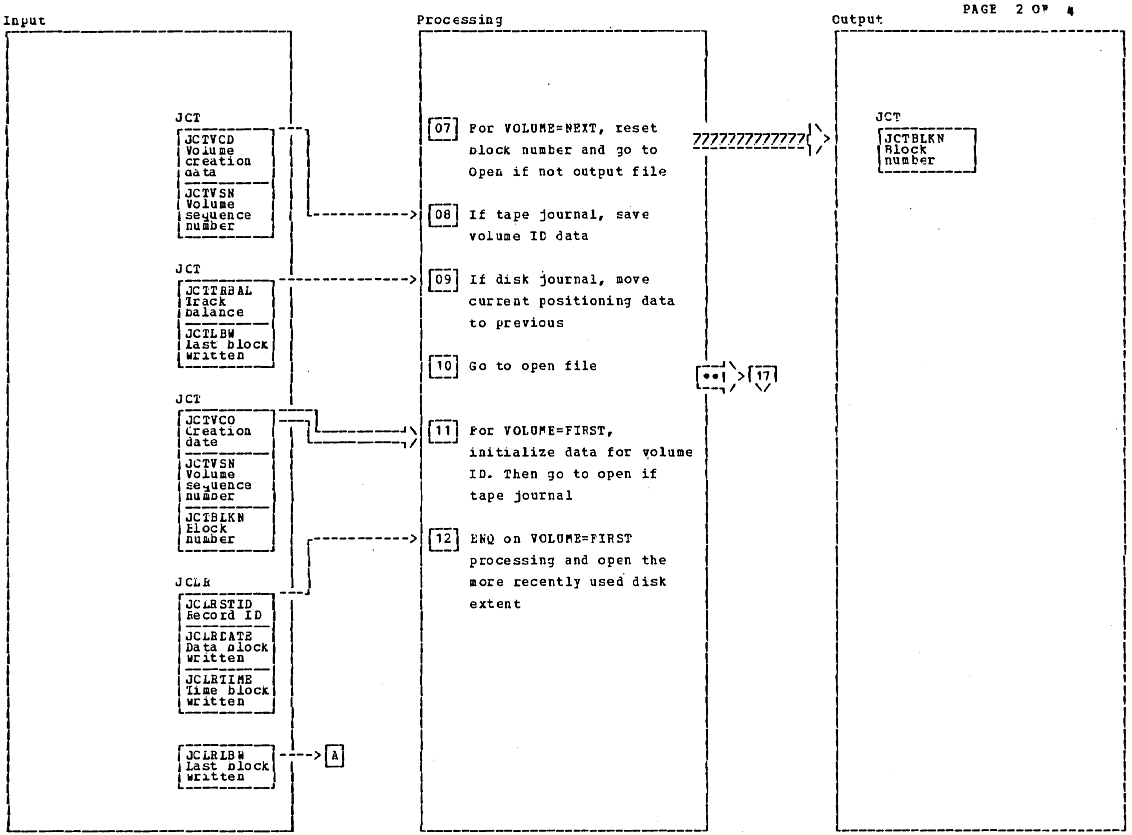
Diagram - 1.9.5.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Request is rejected if journal is not closed. If initiated by Kick-off program, type must be open and VOLUME=FIRST | | DFHJCO | | | | | |
| 02 For VOLUME=FIRST, select the more recently used extent At this time, Journal works in transfer control mode. | | JC100 | | | | | |
| 04 If journal was in input mode a VOL=NEXT request is reset and VOL=CURRENT is forced. | | JC200 | | | | | |
| 05 Valid vclume requests are: NEXT, FIRST, and CURRENT. | | J210 | | | | | |
| 06 ?f it is tape Journal or EOV flag is on, the request is changed to VOLUME=NEXT. | | JC215 | | | | | |

Journal Open Request

Diagram - 1.9.5.1-01

PAGE 2 OF 4


Input
Processing
Output
JCT
JCTVCD Volume creation data
JCTVSN Volume sequence number
JCT
JCTTRBAL Track balance
JCTLBW last block written
JCT
JCTVCO Creation date
JCTVSN Volume sequence number
JCTBLKN Block number
JCLR
JCLRSTID Record ID
JCLRDATE Data block written
JCLRTIME Time block written
JCLRLBW Last block written
A
17
JCT
JCTBLKN Block number

07 For VOLUME=NEXT, reset block number and go to Open if not output file

08 If tape journal, save volume ID data

09 If disk journal, move current positioning data to previous

10 Go to open file

11 For VOLUME=FIRST, initialize data for volume ID. Then go to open if tape journal

12 ENQ on VOLUME=FIRST processing and open the more recently used disk extent

Journal Open Request

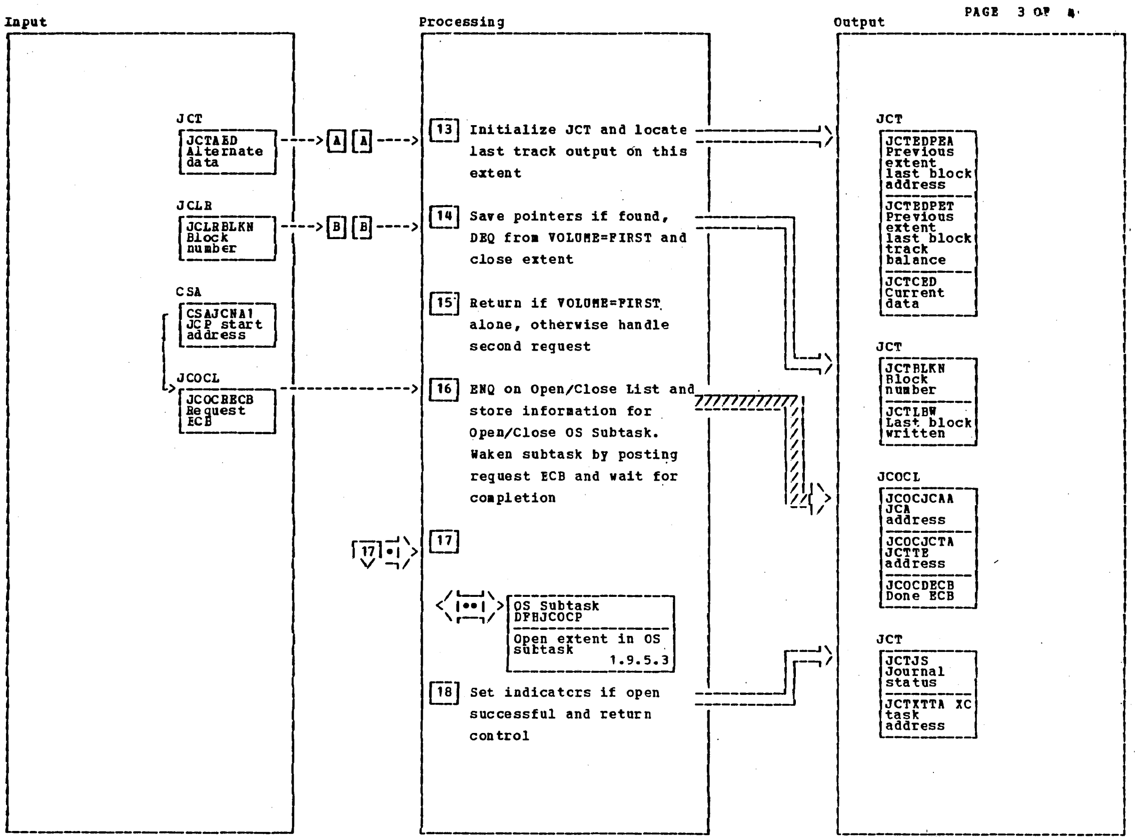
Diagram - 1.9.5.1-C2

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 07 For a flip-flop type of disk journal, swap volume sequence number and the device extent data. | | | | | | | |
| 10 If data has changed, the volume creation date is updated and, for tape, the volume sequence number is reset. | | JC312 | | | | | |
| 11 If SIP read this journal, free buffer storage and reset buffer pointer. Then handle it as VOLUME=CURRENT if disk and as VOLUME=NEXT if tape. | | JC4C0 | | | | | |
| 12 The more recently used extent is recognized by the date and time in the label record of relative track zero record one. Control is returned if extent had never been used. | | JC402 | | | | | |

Journal Open Request

Diagram - 1.9.5.1-C2

PAGE 3 OF 4



13 Initialize JCT and locate last track output on this extent

14 Save pointers if found, DEQ from VOLUME=FIRST and close extent

15 Return if VOLUME=FIRST alone, otherwise handle second request

16 ENQ on Open/Close List and store information for Open/Close OS Subtask. Waken subtask by posting request ECB and wait for completion

17

18 Set indicators if open successful and return control

Journal Open Request

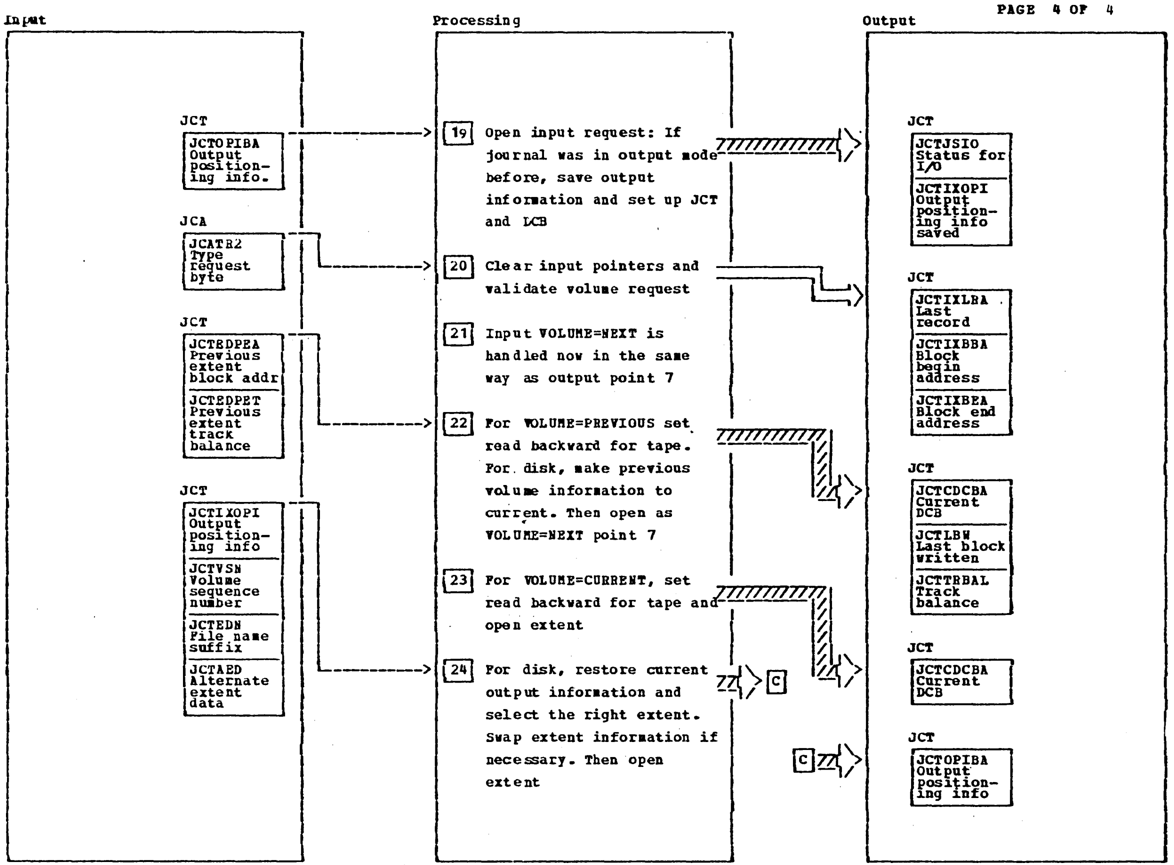
Diagram - 1.9.5.1-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 13 The format record one written by the Journal Formatting program guides to the track which contains the latest written record. | | JC500 | | | | | |
| 14 If an I/O error occurs during Open or reading or the extent is full, the appropriate flag is turned on and control is returned after closing the extent. | | JC580 | | | | | |
| 17 If OS subtask is detached, control is returned with I/O error flag on. OS subtask returns control by posting the 'done' ECB. | | | | | | | |
| 18 If request is issued by the kick-off program, the journal task is resumed before. | | | | | | | |

Journal Open Request

Diagram - 1.9.5.1-03

PAGE 4 OF 4


Input
Processing
Output
JCT
JCTOPIBA Output positioning info.
JCA
JCATR2 Type request byte
JCT
JCTEDPEA Previous extent block addr
JCTEDPET Previous extent track balance
JCT
JCTIXOPI Output positioning info
JCTVSN Volume sequence number
JCTEDN File name suffix
JCTAED Alternate extent data
JCT
JCTJSIO Status for I/O
JCTIXOPI Output positioning info saved
JCT
JCTIXLBA Last record
JCTIXBBA Block begin address
JCTIXBEA Block end address
JCT
JCTCDCBA Current DCB
JCTLBW Last block written
JCTTRBAL Track balance
JCT
JCTCDCBA Current DCB
JCT
JCTOPIBA Output positioning info
C
C

19 Open input request: If journal was in output mode before, save output information and set up JCT and DCB

20 Clear input pointers and validate volume request

21 Input VOLUME=NEXT is handled now in the same way as output point 7

22 For VOLUME=PREVIOUS set read backward for tape. For disk, make previous volume information to current. Then open as VOLUME=NEXT point 7

23 For VOLUME=CURRENT, set read backward for tape and open extent

24 For disk, restore current output information and select the right extent. Swap extent information if necessary. Then open extent

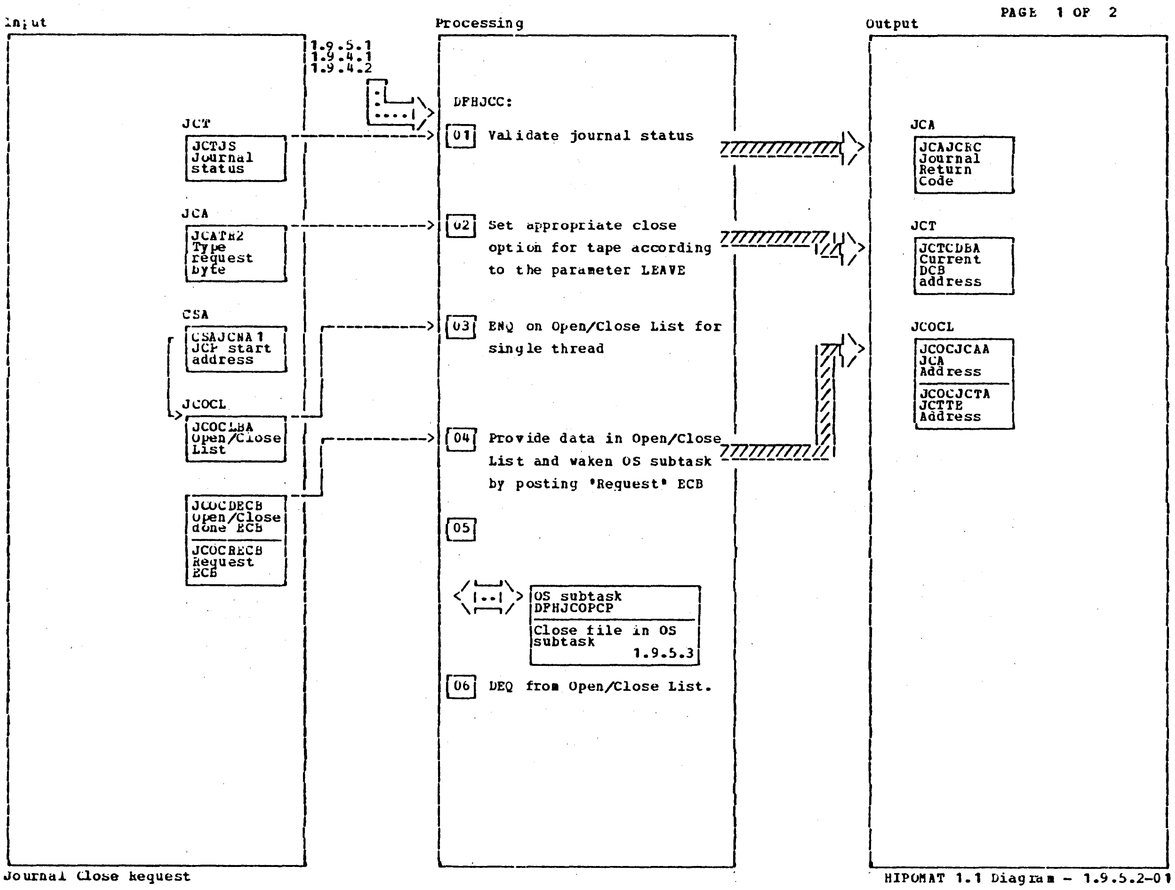
Journal Open Request

HIPOMAT 1.1 Diagram - 1.9.5.1-04

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 22 The first byte of the current DCB address contains the OS Open indicator. | | | | | | | |

Journal Open Request

HIPOMAT 1.1 Diagram - 1.9.5.1-04


PAGE 1 OF 2
Input
Processing
Output
1.9.5.1
1.9.4.1
1.9.4.2
DFHJCC:
JCT
JCTJS Journal status
01 Validate journal status
JCA
JCAJCRC Journal Return Code
JCA
JCATR2 Type request byte
02 Set appropriate close option for tape according to the parameter LEAVE
JCT
JCTCDBA Current DCB address
CSA
CSAJCNA 1 JCP start address
03 ENQ on Open/Close List for single thread
JCOCL
JCOCJCAA JCA Address
JCOCJCTA JCTTE Address
JCOCL
JCOCLBA Open/Close List
04 Provide data in Open/Close List and waken OS subtask by posting 'Request' ECB
JCOCDECB Open/Close done ECB
JCOCRECB Request ECB
05
OS subtask DFHJCOPCP
Close file in OS subtask
1.9.5.3
06 DEQ from Open/Close List.

Journal Close Request

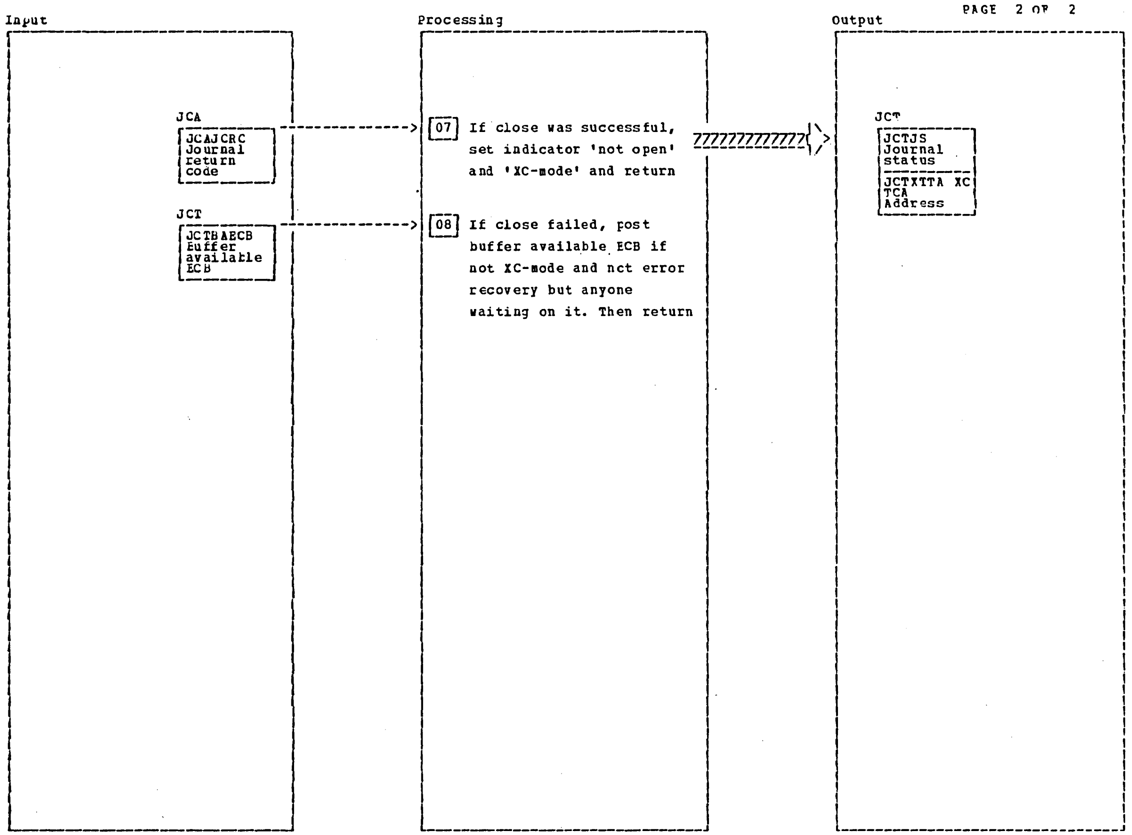
HIPOMAT 1.1 Diagram - 1.9.5.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 Close option for disk and for tape rew/unl is X'80': if tape to be left on drive X'30'. Tape in input mode is always rewound and unloaded. The OS close options are set in byte zero of DCB address. | | | | | | | |
| 05 If OS subtask is detached, return with I/O-error code. | | | | | | | |

Journal Close Request

HIPOMAT 1.1 Diagram - 1.9.5.2-01

PAGE 2 OF 2



Input

Processing

Output

JCA

JCAJCRC Journal return code

JCT

JCTBAECB Buffer available ECB

07 If close was successful, set indicator 'not open' and 'XC-mode' and return

08 If close failed, post buffer available ECB if not XC-mode and nct error recovery but anyone waiting on it. Then return

JCT

JCTJS Journal status

JCTXTTA XC TCA Address

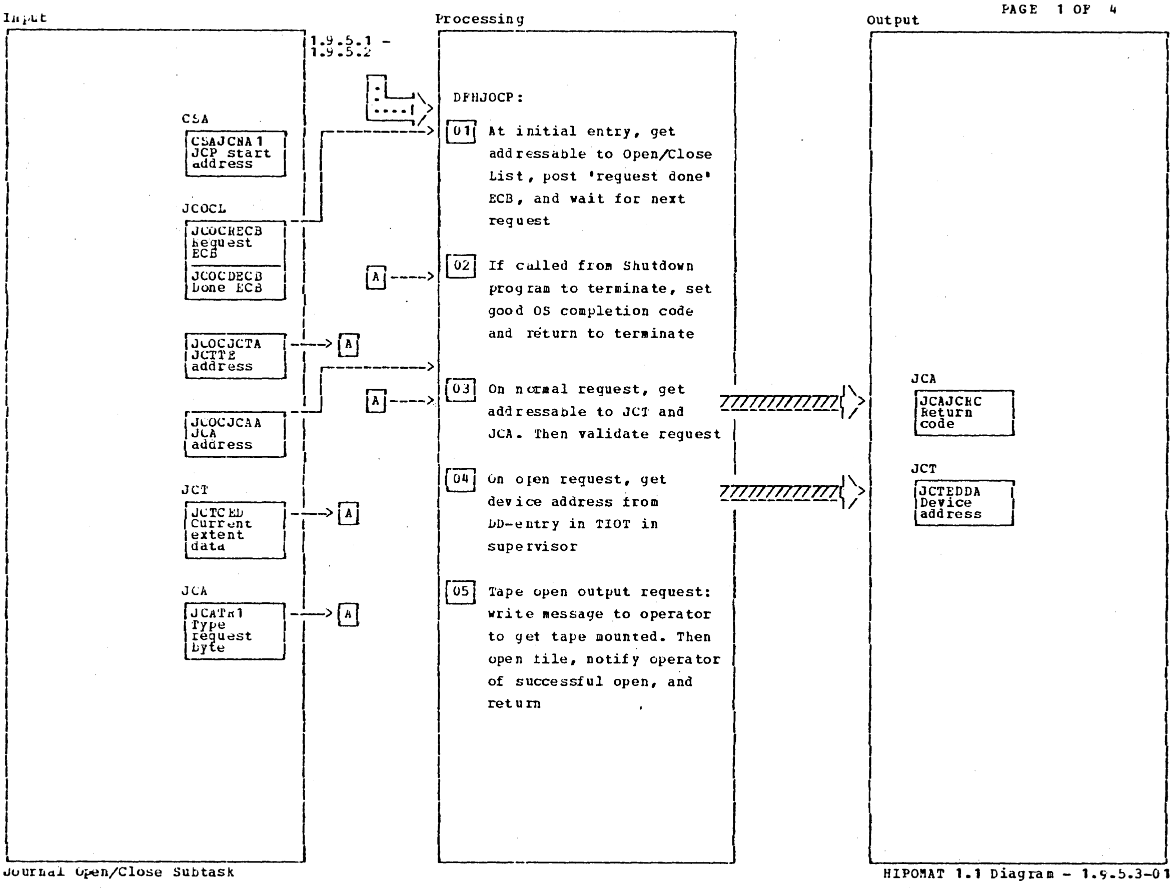
Journal Close Request

Diagram - 1.9.5.2-C2

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Journal Close Request

Diagram - 1.9.5.2-C2



DFHJOCP:

01 At initial entry, get addressable to Open/Close List, post 'request done' ECB, and wait for next request

02 If called from Shutdown program to terminate, set good OS completion code and return to terminate

03 On normal request, get addressable to JCT and JCA. Then validate request

04 On open request, get device address from DD-entry in TIOT in supervisor

05 Tape open output request: write message to operator to get tape mounted. Then open file, notify operator of successful open, and return

Journal Open/Close Subtask

HIPOMAT 1.1 Diagram - 1.9.5.3-01

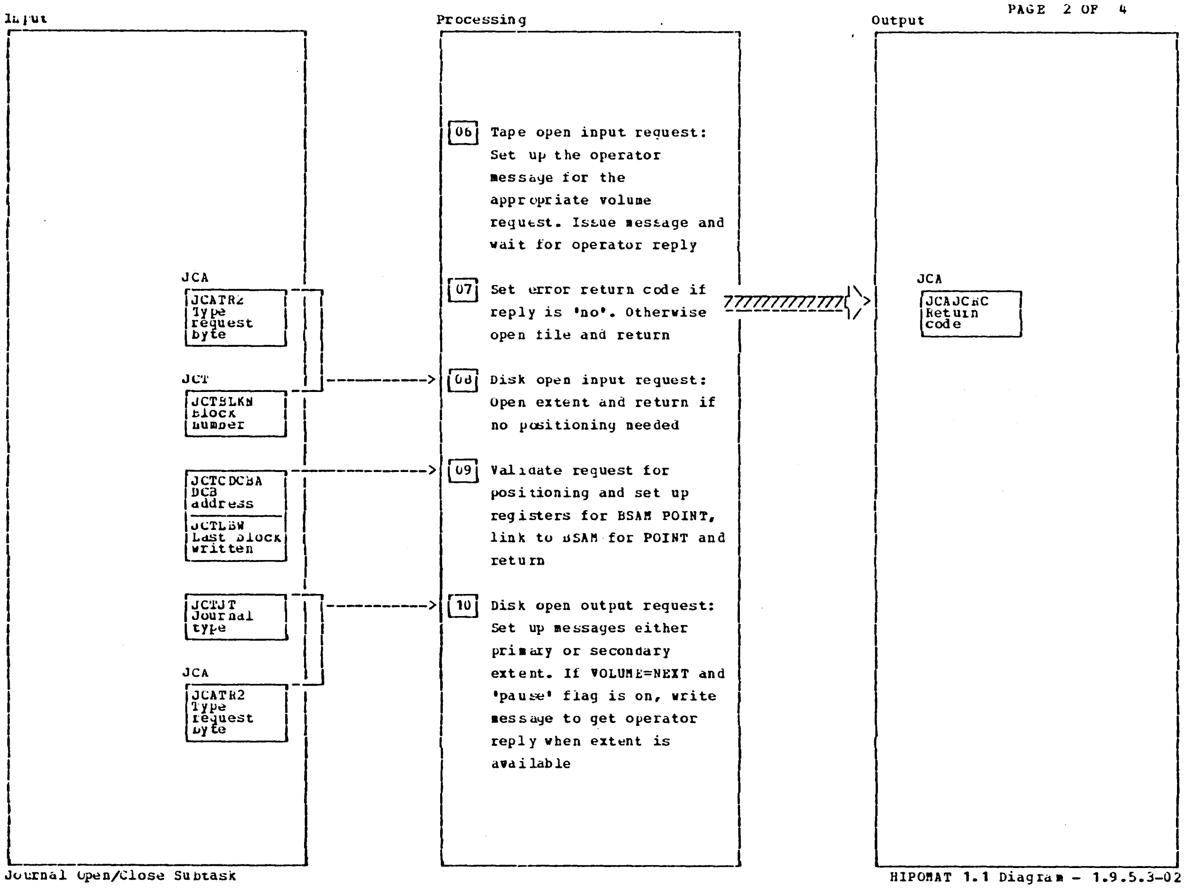
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 DFHJCOCP works as OS subtask. Control is passed to it via the 'request' ECB and returned via 'done' ECB. | | DFHJCOCP | |
| 02 If called to terminate journaling, all bytes of the JCTTE address in the Open/Close List are set to X'FF'. | | JC25 | |
| 04 If the device can't be found, the open is issued to get the error message. | | JC60 | |
| 05 If open fails, control is returned with I/O error indication. | | OPERTN | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Journal Open/Close Subtask

HIPOMAT 1.1 Diagram - 1.9.5.3-01

PAGE 2 OF 4



06 Tape open input request: Set up the operator message for the appropriate volume request. Issue message and wait for operator reply

07 Set error return code if reply is 'no'. Otherwise open file and return

08 Disk open input request: Open extent and return if no positioning needed

09 Validate request for positioning and set up registers for BSAM POINT, link to BSAM for POINT and return

10 Disk open output request: Set up messages either primary or secondary extent. If VOLUME=NEXT and 'pause' flag is on, write message to get operator reply when extent is available

Journal Open/Close Subtask

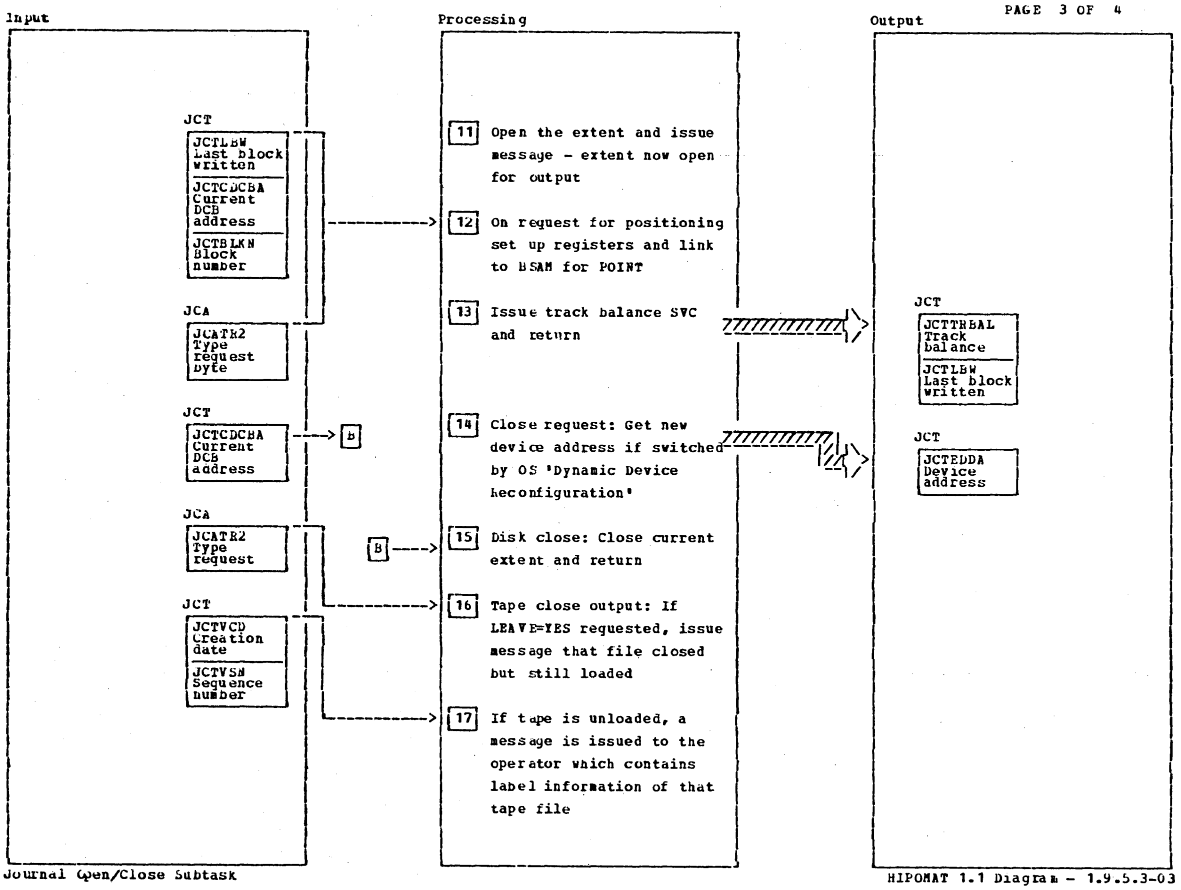
HIPOMAT 1.1 Diagram - 1.9.5.3-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 06 Volume specification can be: CURRENT, NEXT, PREVIOUS, or SPECIFIC input. |  | JC150 |  |  |  |  |  |
| 08 No positioning if volume current points to block number one. |  | JC330 |  |  |  |  |  |
| 10 Program loops on the message if reply is not YES. |  | JC320 |  |  |  |  |  |

Journal Open/Close Subtask

HIPOMAT 1.1 Diagram - 1.9.5.3-02

PAGE 3 OF 4



11 Open the extent and issue message - extent now open for output

12 On request for positioning set up registers and link to BSAM for POINT

13 Issue track balance SVC and return

14 Close request: Get new device address if switched by OS 'Dynamic Device Reconfiguration'

15 Disk close: Close current extent and return

16 Tape close output: If LEAVE=YES requested, issue message that file closed but still loaded

17 If tape is unloaded, a message is issued to the operator which contains label information of that tape file

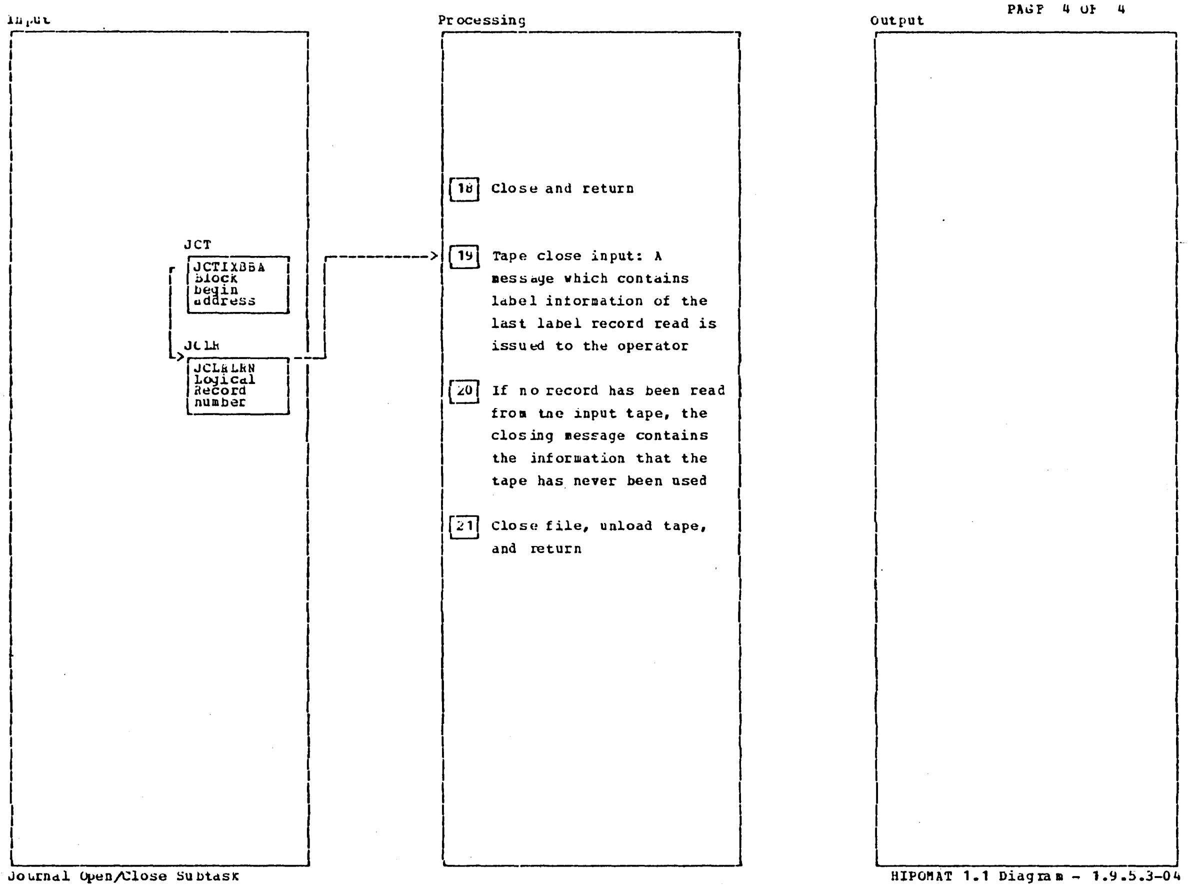
Journal Open/Close Subtask

HIPOMAT 1.1 Diagram - 1.9.5.3-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 12 POINT is not issued if VOLUME=CURRENT and block number is one or if the 'last block written' field is zero or if VOLUME=NEXT. | | POINTRT | | | | | |
| 13 Error exit is taken if SVC failed. | | POINTRTQ | | | | | |

Journal Open/Close Subtask

HIPOMAT 1.1 Diagram - 1.9.5.3-03



18 Close and return

19 Tape close input: A message which contains label information of the last label record read is issued to the operator

20 If no record has been read from the input tape, the closing message contains the information that the tape has never been used

21 Close file, unload tape, and return

Journal Open/Close Subtask

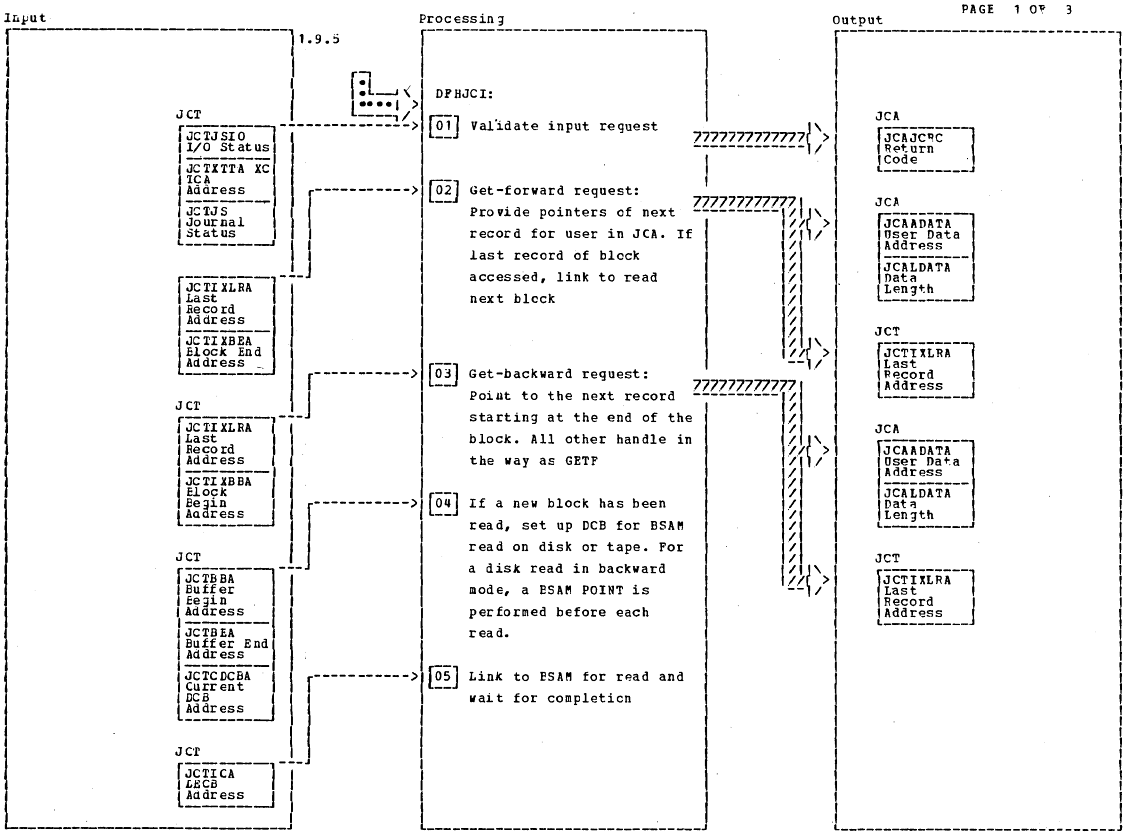
HIPOMAT 1.1 Diagram - 1.9.5.3-04

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Journal Open/Close Subtask

HIPOMAT 1.1 Diagram - 1.9.5.3-04

PAGE 1 OF 3



Input

1.9.5

JCT

JCTJSIO I/O Status

JCTXTTA XC TCA Address

JCTJS Journal Status

JCTIXLRA Last Record Address

JCTIXBEA Block End Address

JCT

JCTIXLRA Last Record Address

JCTIXBBA Block Begin Address

JCT

JCTBBA Buffer Begin Address

JCTBEA Buffer End Address

JCTCDCBA Current DCB Address

JCT

JCTICA LECB Address

Processing

DFHJCI:

01 Validate input request

02 Get-forward request: Provide pointers of next record for user in JCA. If last record of block accessed, link to read next block

03 Get-backward request: Point to the next record starting at the end of the block. All other handle in the way as GETF

04 If a new block has been read, set up DCB for BSAM read on disk or tape. For a disk read in backward mode, a BSAM POINT is performed before each read.

05 Link to BSAM for read and wait for completion

Output

JCA

JCAJCRC Return Code

JCA

JCAADATA User Data Address

JCALDATA Data Length

JCT

JCTIXLRA Last Record Address

JCA

JCAADATA User Data Address

JCALDATA Data Length

JCT

JCTIXLRA Last Record Address

Journal Input Request

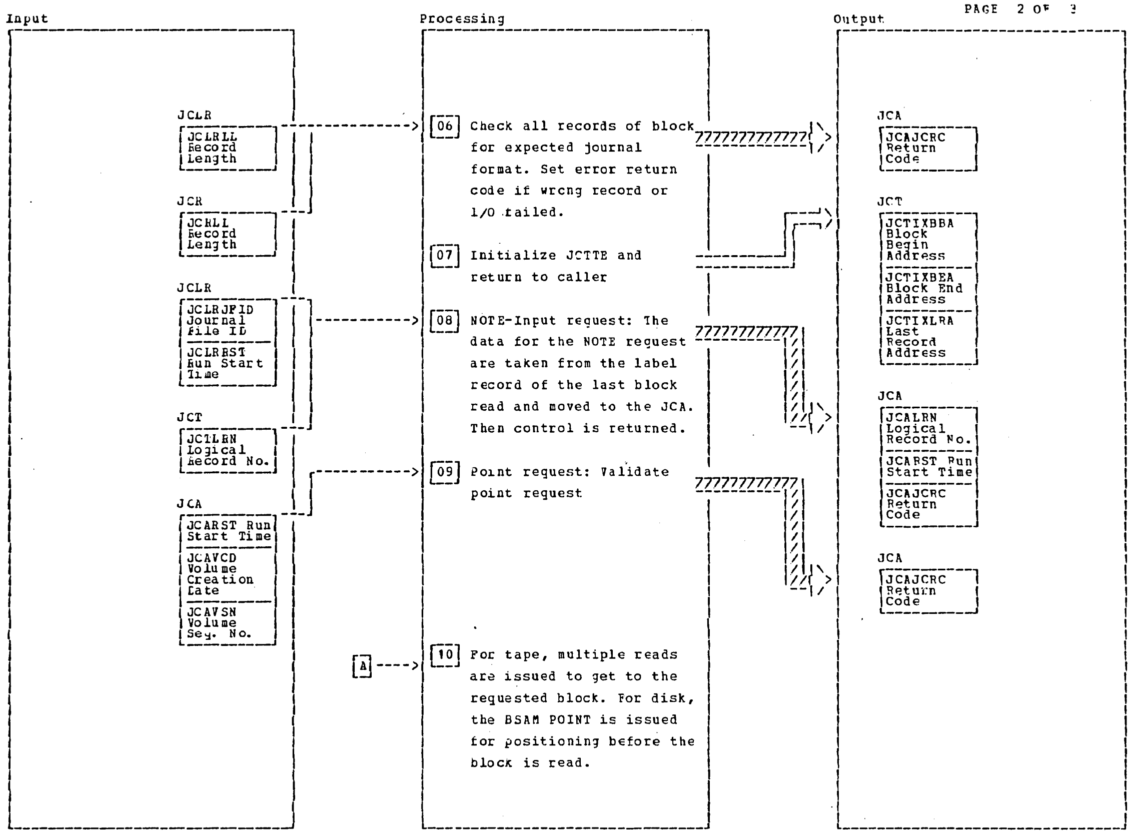
Diagram - 1.9.5.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 Control is returned with EOF indicator on if in backward mode block one has been accessed. POINT is not issued if first access. | | JC3C0 | | | | | |

Journal Input Request

Diagram - 1.9.5.4-01

PAGE 2 OF 3



06 Check all records of block for expected journal format. Set error return code if wrong record or I/O failed.

07 Initialize JCTTE and return to caller

08 NOTE-Input request: The data for the NOTE request are taken from the label record of the last block read and moved to the JCA. Then control is returned.

09 Point request: Validate point request

10 For tape, multiple reads are issued to get to the requested block. For disk, the BSAM POINT is issued for positioning before the block is read.

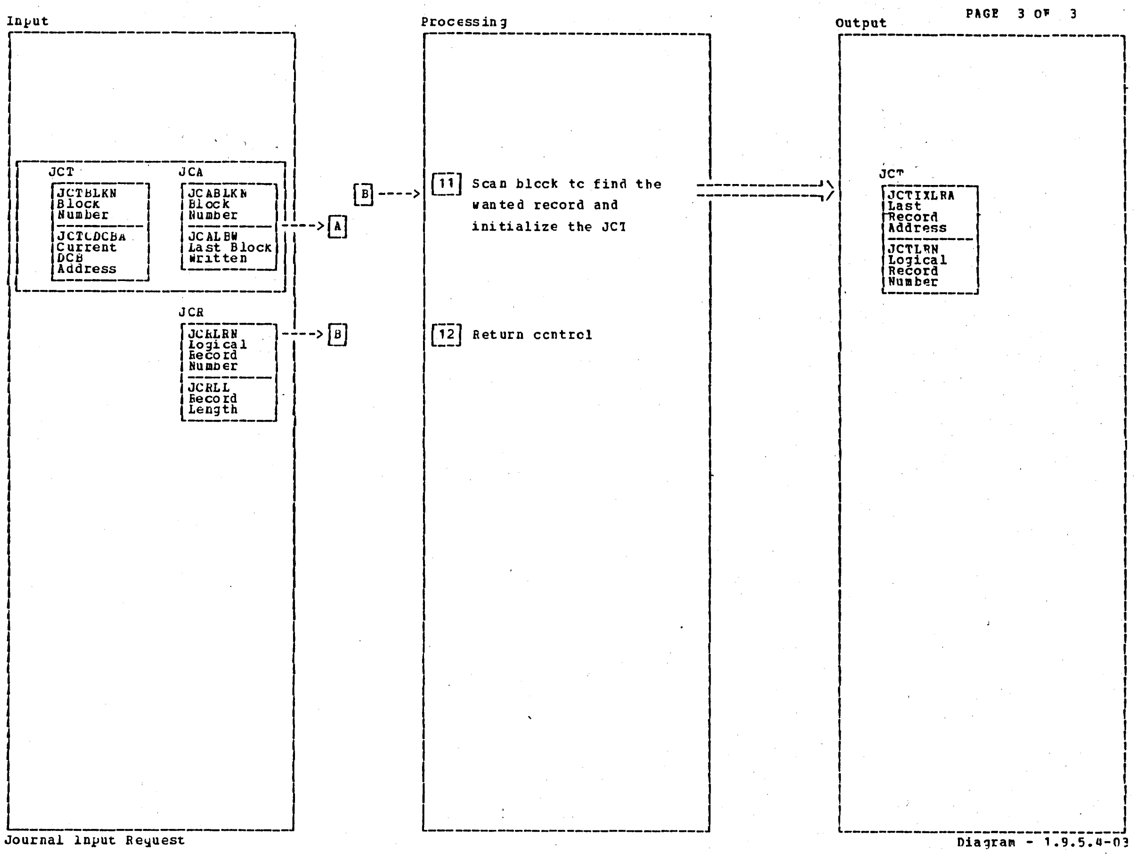
Journal Input Request

Diagram - 1.9.5.4-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 08 If no read has been issued before, an error return code is set and control is returned. | | | | | | | |
| 09 The request data are compared against the appropriate data of the label record. | | | | | | | |

Journal Input Request

Diagram - 1.9.5.4-02


PAGE 3 OF 3
Input
Processing
Output
JCT
JCA
JCTBLKN Block Number
JCTCDCBA Current DCB Address
JCABLKN Block Number
JCALBW Last Block Written
A
JCR
JCRLRN Logical Record Number
JCRLL Record Length
B
B
11 Scan block to find the wanted record and initialize the JCT
12 Return control
JCT
JCTIXLRA Last Record Address
JCTLRN Logical Record Number

Journal Input Request

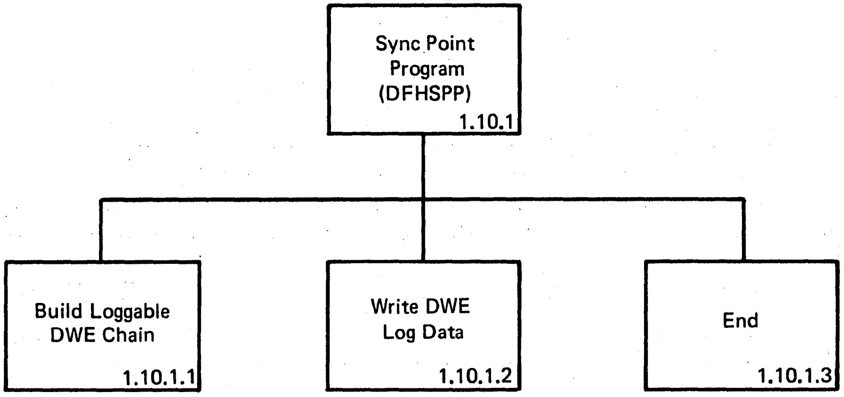
Diagram - 1.9.5.4-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 10 Control is returned with error indicator on if block number is not found. | | | | | | | |
| 11 Control is returned with error indicator on if block number is not found. | | | | | | | |

Journal Input Request

Diagram - 1.9.5.4-03

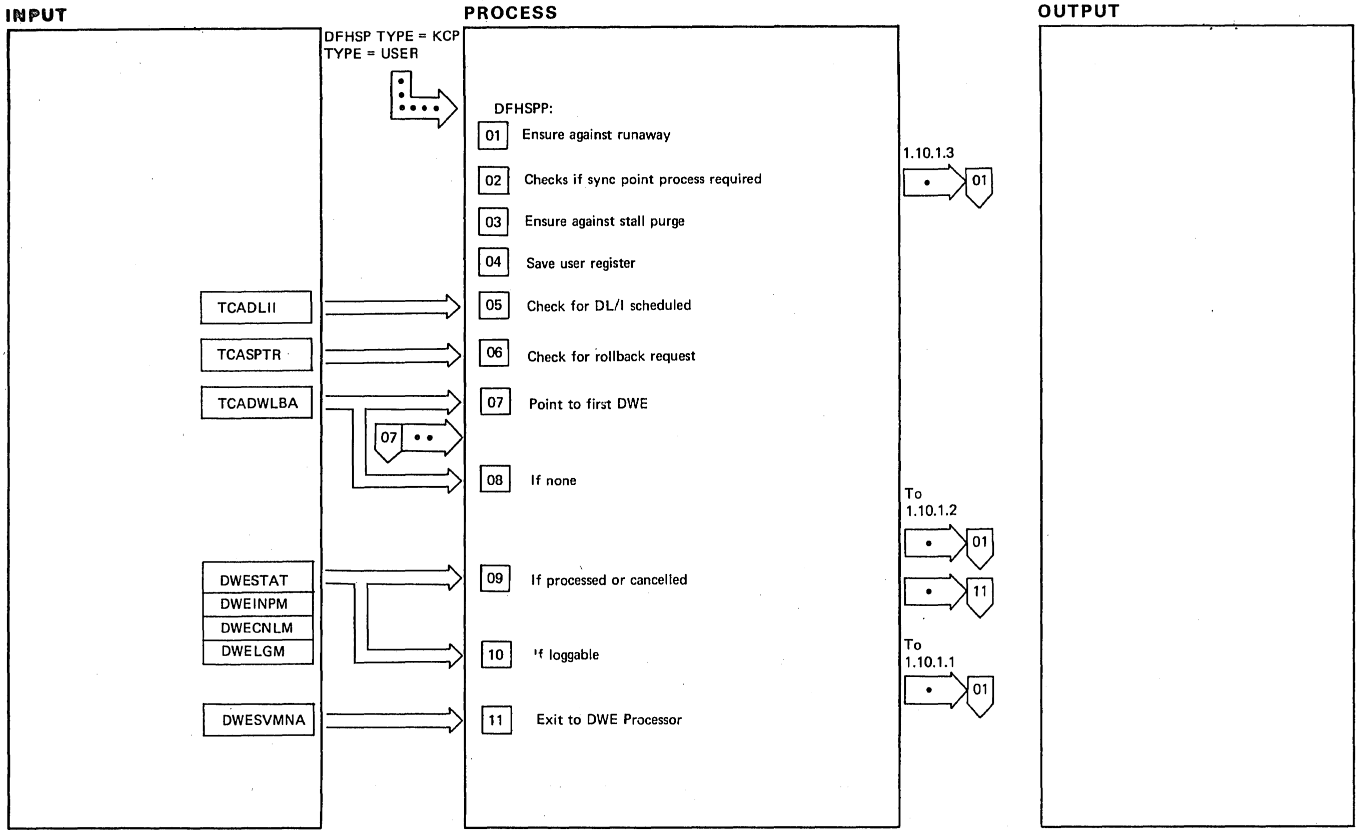
Page 1 of 1



Visual Index – Sync Point Program

Diagram – 1.10.0-01

Page 1 of 2



Sync Point Program (DFHSPP)

Diagram – 1.10.1-01

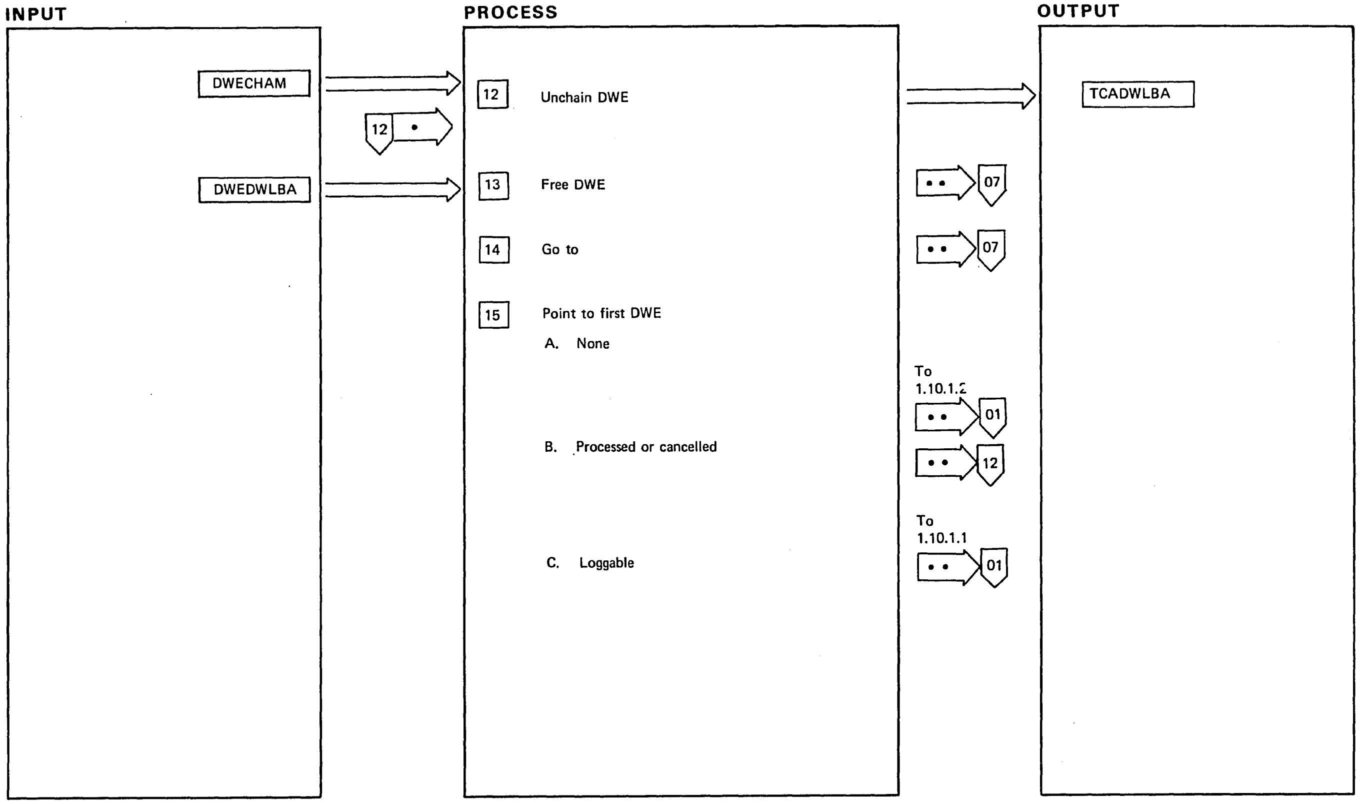
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 DFHIC TYPE = ENTER, CLASS = CICS/VS | ENTRY | DFHSPPEP | |
| 03 DFHKC TYPE = NO PURGE | | | |
| 05 Issue a call to DL/I task termination, if DL/I scheduled | | | |
| 06 Invokes DFHDBP to modify or reverse all changes to protected resources | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Sync Point Program (DFHSPP)

Diagram – 1.10.1-01

Page 2 of 2



12 Unchain DWE

13 Free DWE

14 Go to

15 Point to first DWE

A. None

B. Processed or cancelled

C. Loggable

Sync Point Program (DFHSPP)

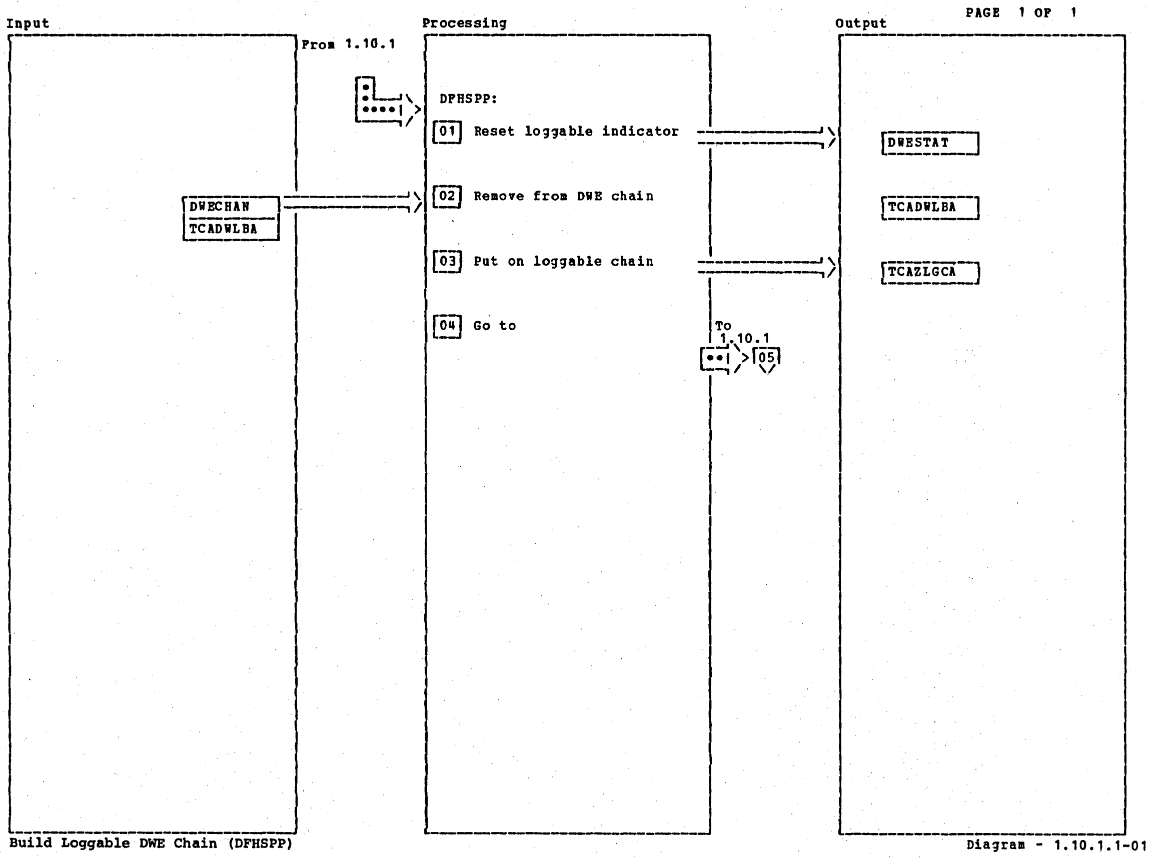
Diagram – 1.10.1-02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 12 DFHSC TYPE = FREEMAIN | | SPP02100 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Sync Point Program (DFHSPP)

Diagram – 1.10.1-02



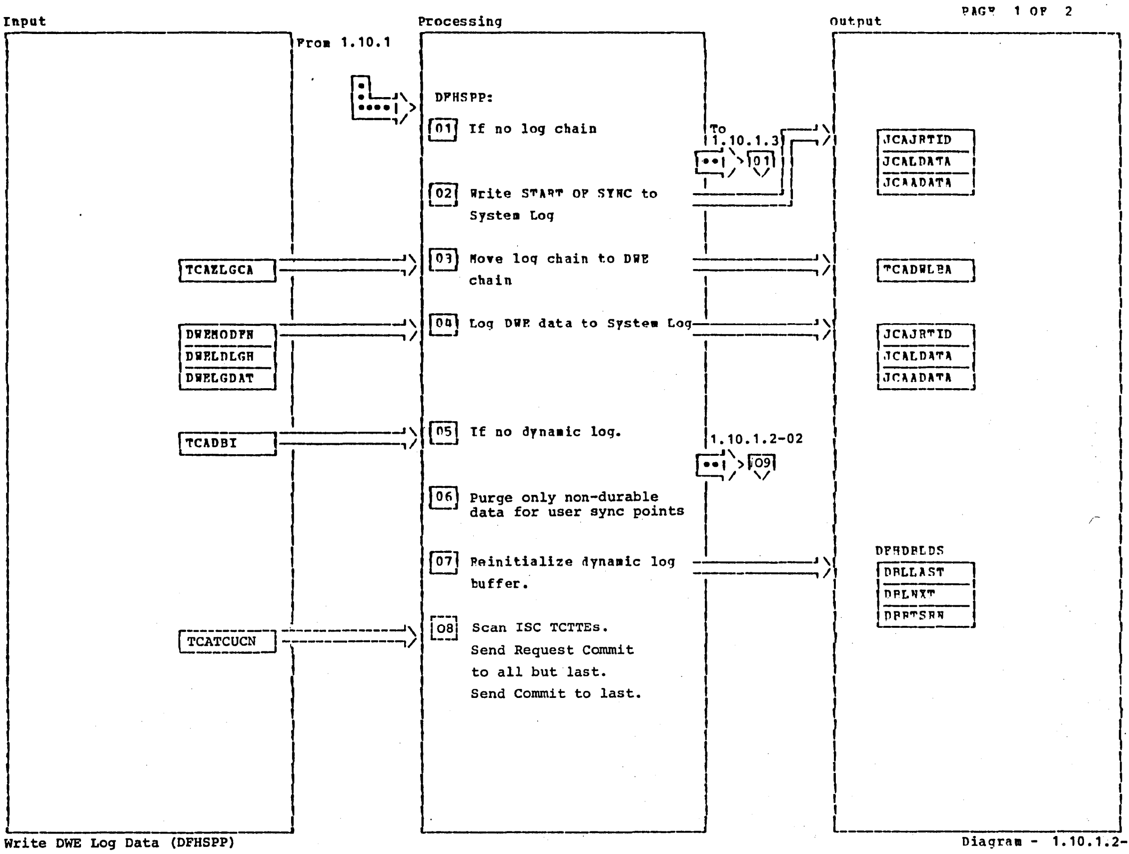
Build Loggable DWE Chain (DFHSPP)

Diagram - 1.10.1.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Reset the log indicator. | LOG<br>CHAIN<br>BUILD | SPP02020 | | | | | |
| 02 Remove from chain TCADWLBA. | | | | | | | |
| 03 Put on special log chain TCAZLGCA. | | | | | | | |
| 04 Return. | | | | | | | |

Build Loggablé DWE Chain (DFHSPP)

Diagram - 1.10.1.1-01


PAGE 1 OF 2
Input
Processing
Output
From 1.10.1
DFHSPP:
01 If no log chain
To 1.10.1.3
01
02 Write START OF SYNC to System Log
JCAJRTID
JCALDATA
JCAADATA
TCAZLGCA
03 Move log chain to DWE chain
TCADWLBA
DWEMODFN
DWELDLGH
DWELGDAT
04 Log DWE data to System Log
JCAJRTID
JCALDATA
JCAADATA
TCADBI
05 If no dynamic log.
1.10.1.2-02
09
06 Purge only non-durable data for user sync points
07 Reinitialize dynamic log buffer.
DFHDBLDS
DBLLAST
DBLNXT
DBBTSRN
TCATCUCN
08 Scan ISC TCTTEs. Send Request Commit to all but last. Send Commit to last.

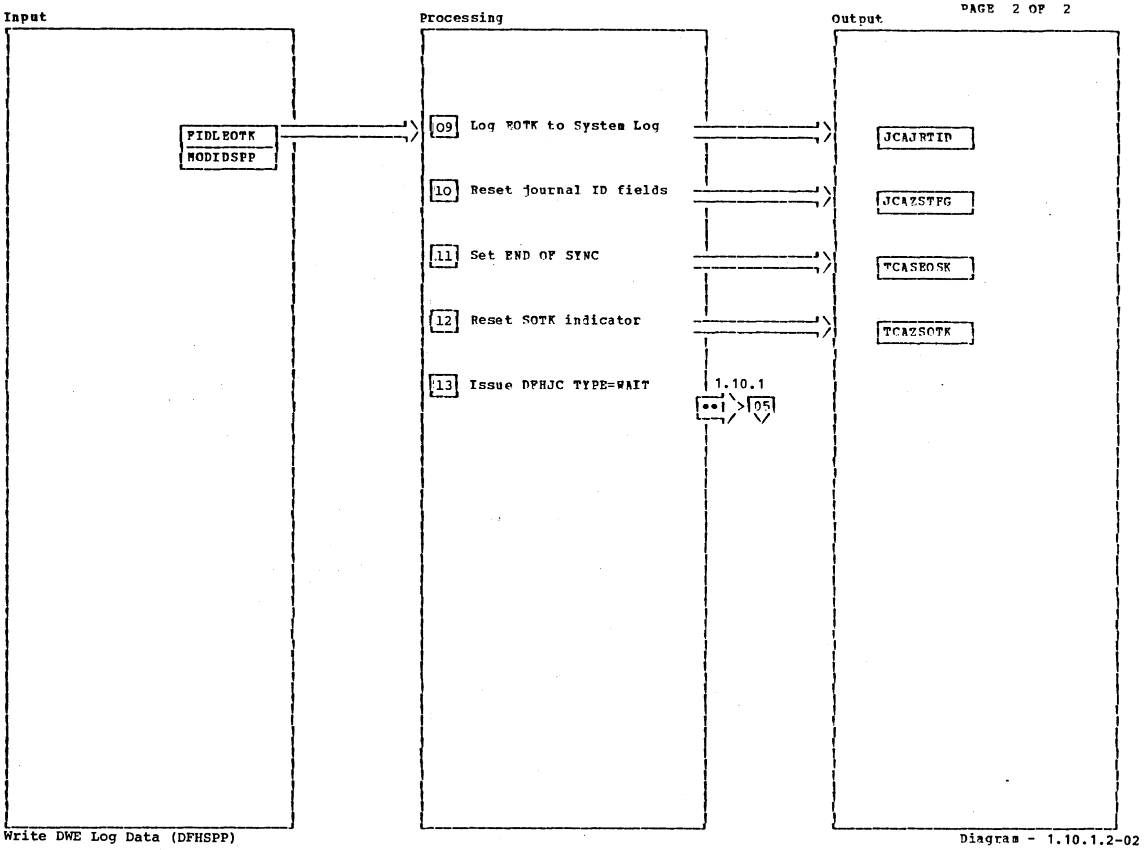
Write DWE Log Data (DFHSPP)

Diagram - 1.10.1.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 DFHJC TYPE=WRITE | WRITE SOSP | SPP00005 | | | | | |
| 04 DFHJC TYPE=WRITE | WRITE DATA | SPP00010 | | | | | |
| 08 DFHTC CTYPE=PREPARE<br>DFHTC CTYPE=COMMIT | | | | | | | |

Write DWE Log Data (DFHSPP)

Diagram - 1.10.1.2-01



Write DWE Log Data (DFHSPP)

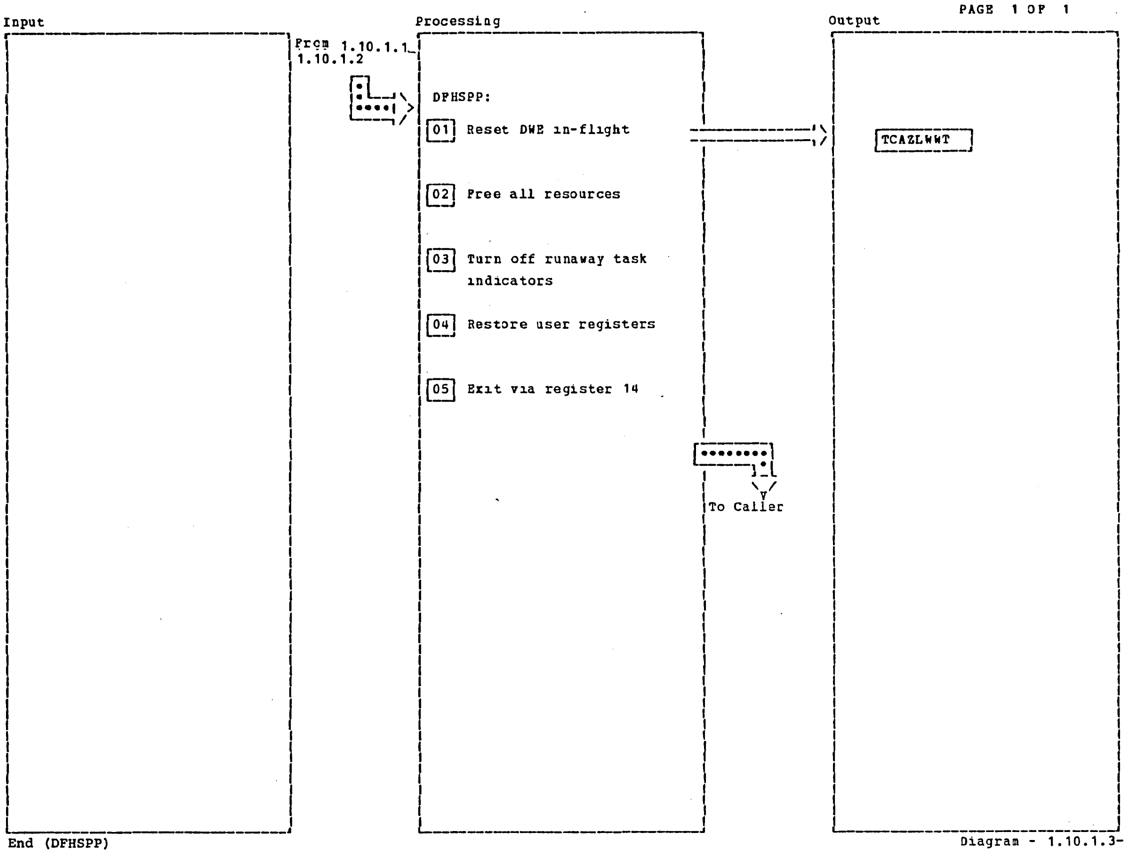
Diagram - 1.10.1.2-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 09 DFHJC TYPE=WRITE | LOGEOTK | SPP01000 | | | | | |
| 13 Journal Control | | SPP01030 | | | | | |

Write DWE Log Data (DFHSPP)

Diagram - 1.10.1.2-02

PAGE 1 OF 1



End (DFHSPP)

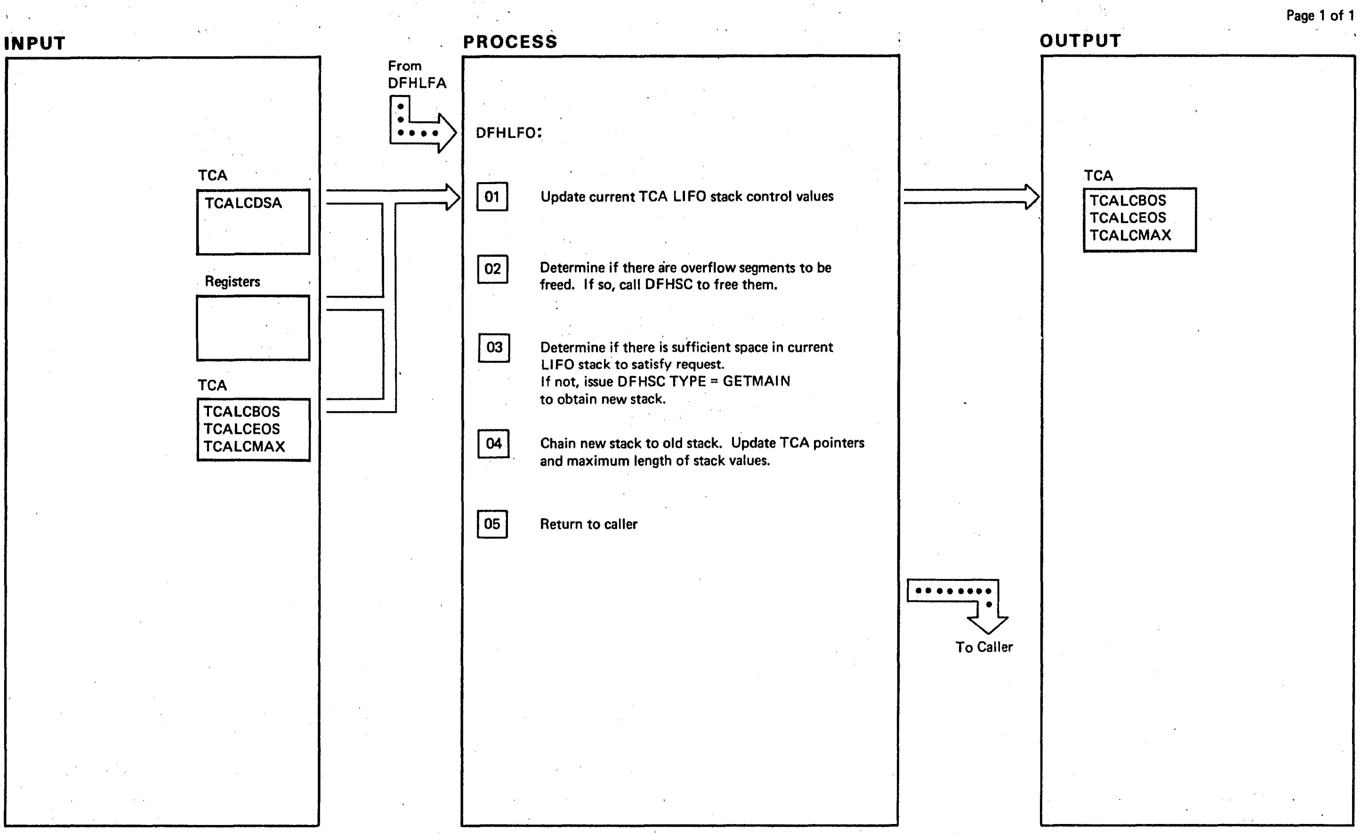
Diagram - 1.10.1.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Reset indicator | | SPP03000 | | | | | |
| 02 DFHKC TYPE=DEQALL | | | | | | | |
| 03 DFHIC TYPE=EXIT | | | | | | | |

End (DFHSPP)

Diagram - 1.10.1.3-01

Page 1 of 1



DFHLFO:

01 Update current TCA LIFO stack control values

02 Determine if there are overflow segments to be freed. If so, call DFHSC to free them.

03 Determine if there is sufficient space in current LIFO stack to satisfy request. If not, issue DFHSC TYPE = GETMAIN to obtain new stack.

04 Chain new stack to old stack. Update TCA pointers and maximum length of stack values.

05 Return to caller

LIFO Overflow (DFHLFO)

Diagram — 1.11.1-01

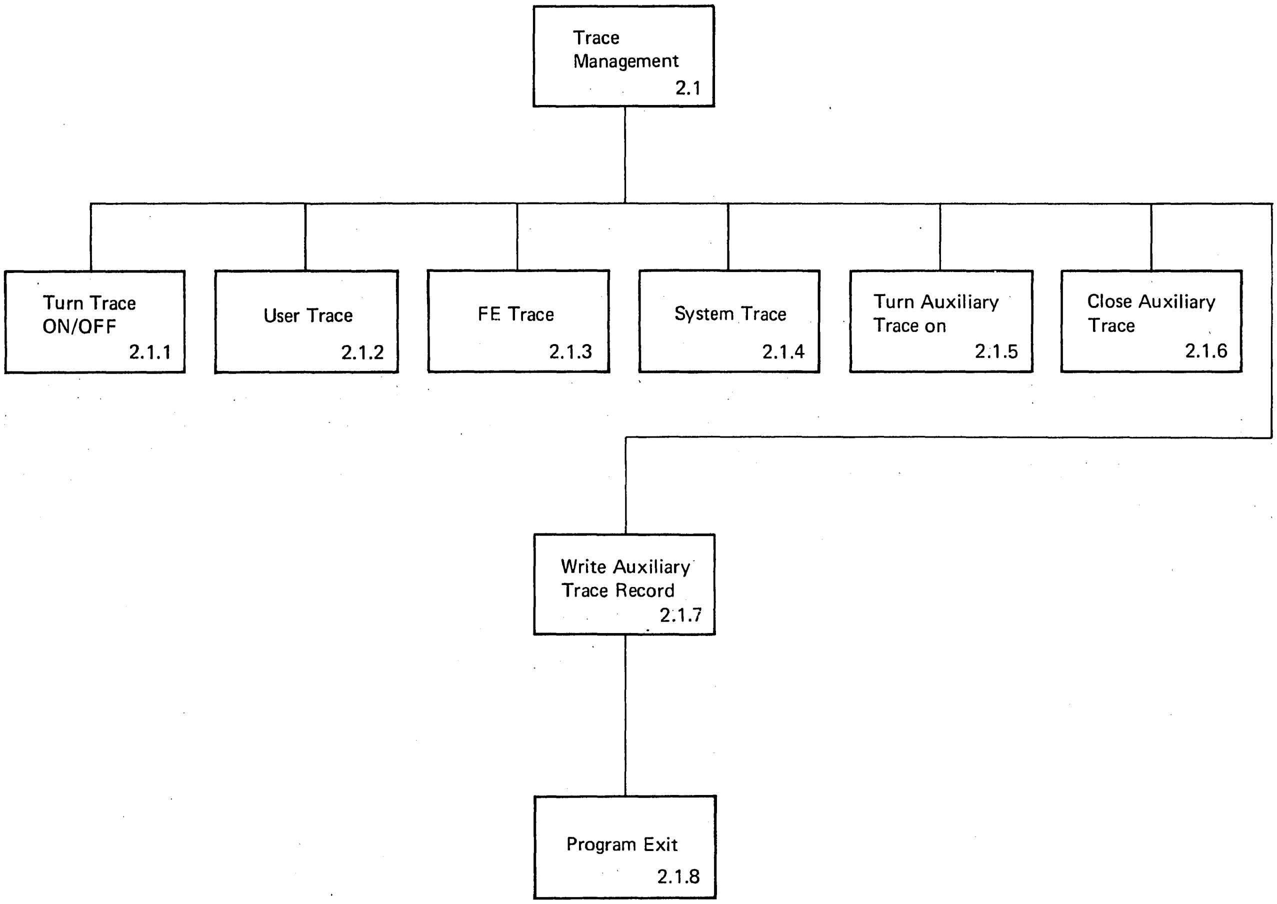
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 DFHLFO is entered whenever it appears there is no space in the current LIFO stack for the requesting module. Arguments are passed directly in registers. Registers are not saved as there is no space in which to save them. | | | |
| 02 There may already be LIFO overflow segments in existance which are not currently in use. | | | |
| 03 Required space is now available in the LIFO stack. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

LIFO Overflow (DFHLFO)

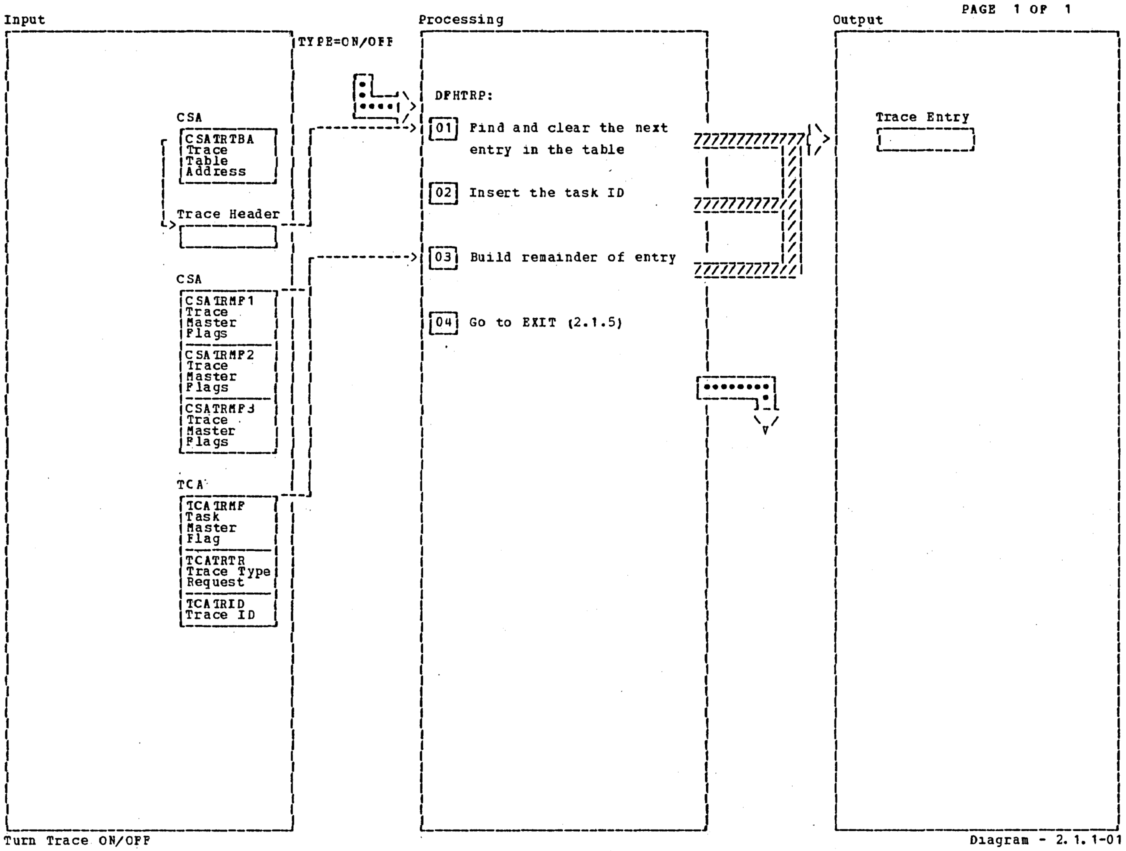
Diagram — 1.11.1-01

Page 1 of 1



Trace Management

Diagram - 2.1.0-01

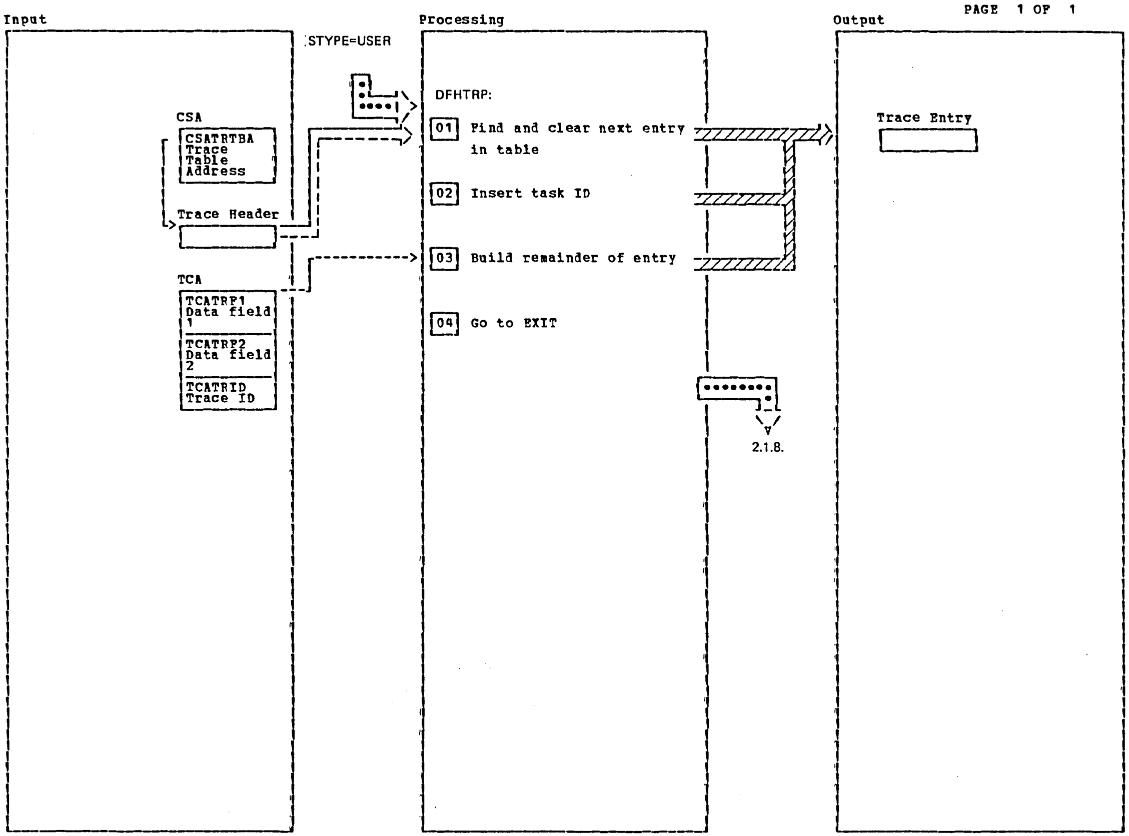


Turn Trace ON/OFF

Diagram - 2.1.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 This module is entered via a DFHTR TYPE=ON/OFF, STYPE=USER/SYSTEM FE macro, DFHTRACE macro, or DFHLFM macro. | | TRCKTR | | | | | |
| 02 | | TROK | | | | | |
| 03 TYPE=ON/OFF | | TRONOFF | | | | | |
| STYPE=USER | | TRNOSVS | | | | | |
| STYPE=SYSTEM | | TRGOON | | | | | |
| STYPE=FE | | TRFEE | | | | | |
| 04 The macro instruction expansion performs the turning ON or OFF of the trace flags in the CSA and TCA. | | | | | | | |

PAGE 1 OF 1



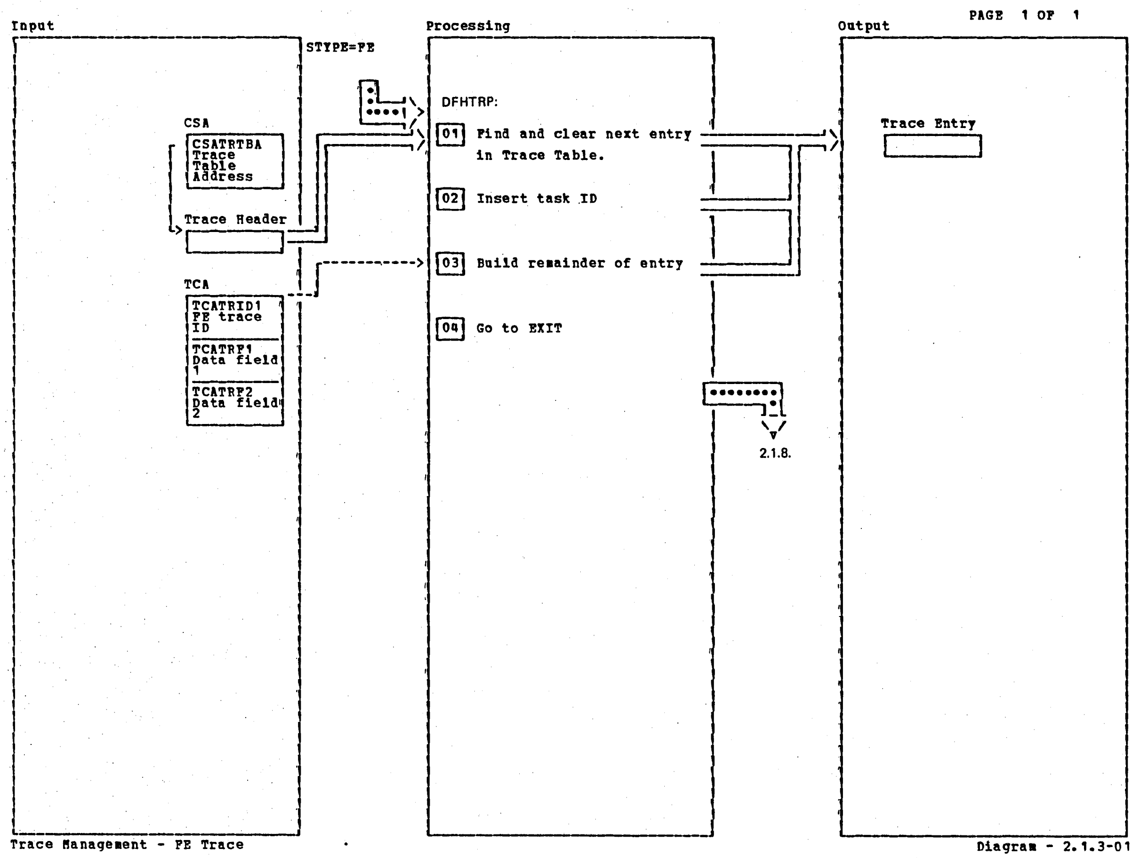
Trace Management - User Trace

Diagram - 2.1.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | TRCKTR | | | | | |
| 02 | | TROK | | | | | |
| 03 | | TRNOSYS | | | | | |

Trace Management - User Trace

Diagram - 2.1.2-01



Trace Management - PE Trace

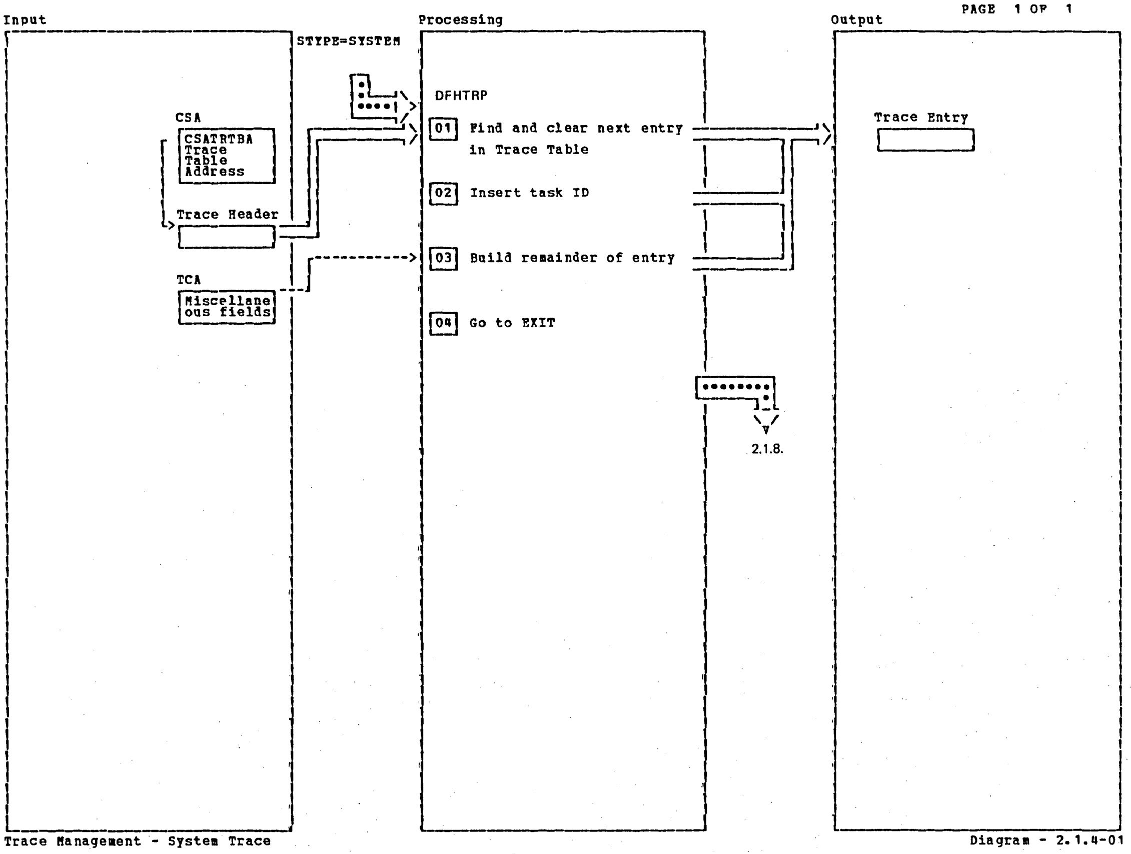
Diagram - 2.1.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | TRCKTR | |
| 02 | | TROK | |
| 03 | | TRPEE | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Trace Management - PE Trace

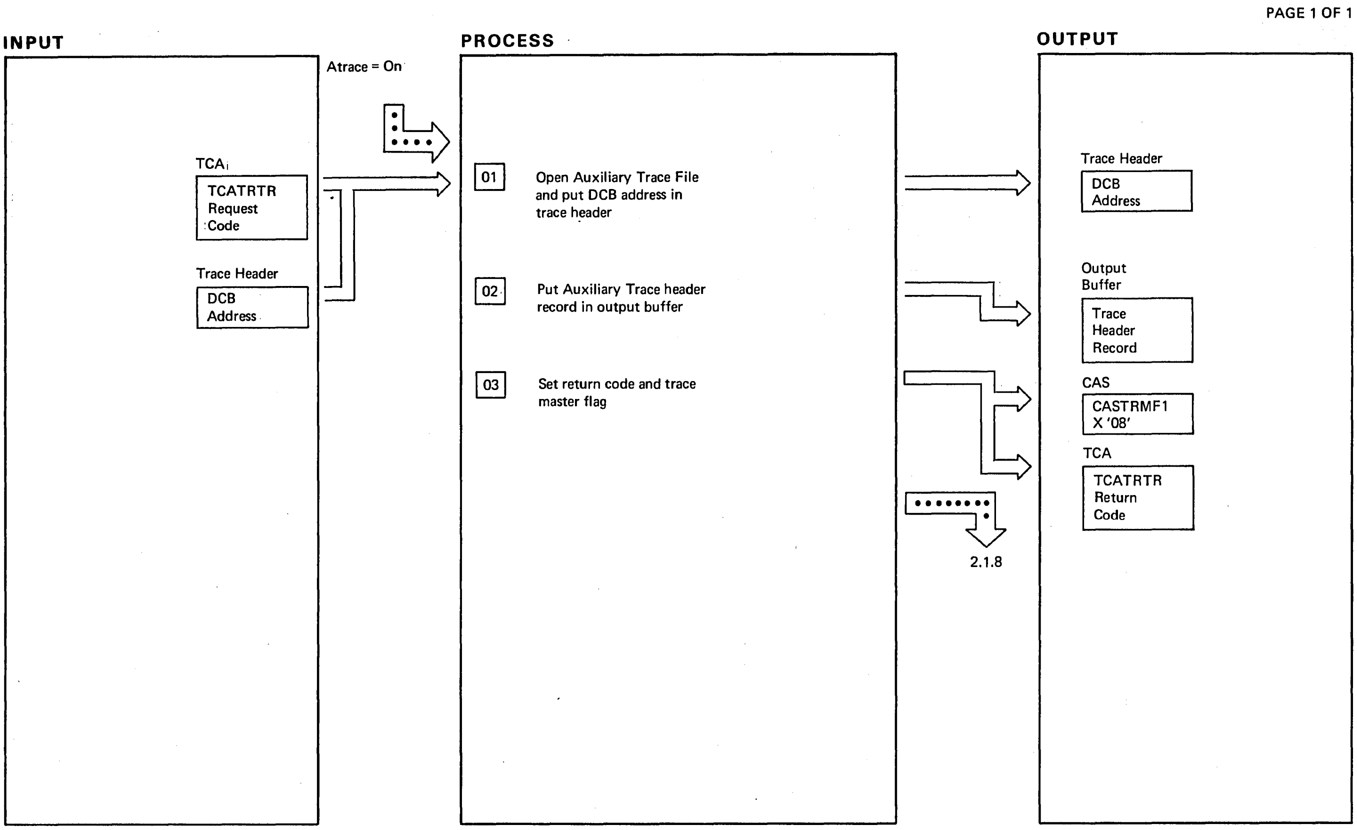
Diagram - 2.1.3-01



Trace Management - System Trace

Diagram - 2.1.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | TRCKTR | | | | | |
| 02 Data fields are taken from the TCA based on the ID and type request. | | TROK | | | | | |
| 03 | | TRGOON | | | | | |



01 Open Auxiliary Trace File and put DCB address in trace header

02 Put Auxiliary Trace header record in output buffer

03 Set return code and trace master flag

Turn Auxiliary Trace On

Diagram - 2.1.5-01

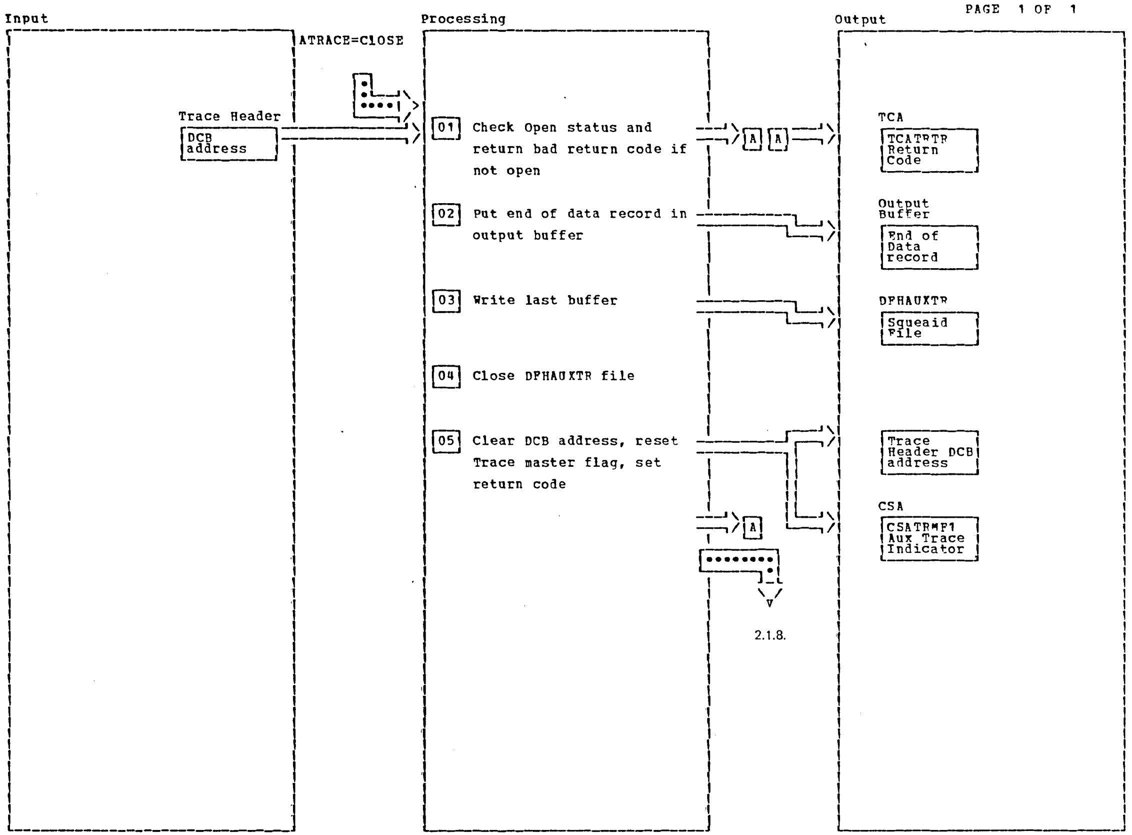
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 If OPEN fails a return code of X 'EQ' is returned. | | TRAXOPEN | |
| 02 | | TRAXOPOK | |
| 03 If OPEN successful, return code is X'00'. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Turn Auxiliary Trace On

Diagram - 2.1.5-01

PAGE 1 OF 1



01 Check Open status and return bad return code if not open

02 Put end of data record in output buffer

03 Write last buffer

04 Close DFHAUXTR file

05 Clear DCB address, reset Trace master flag, set return code

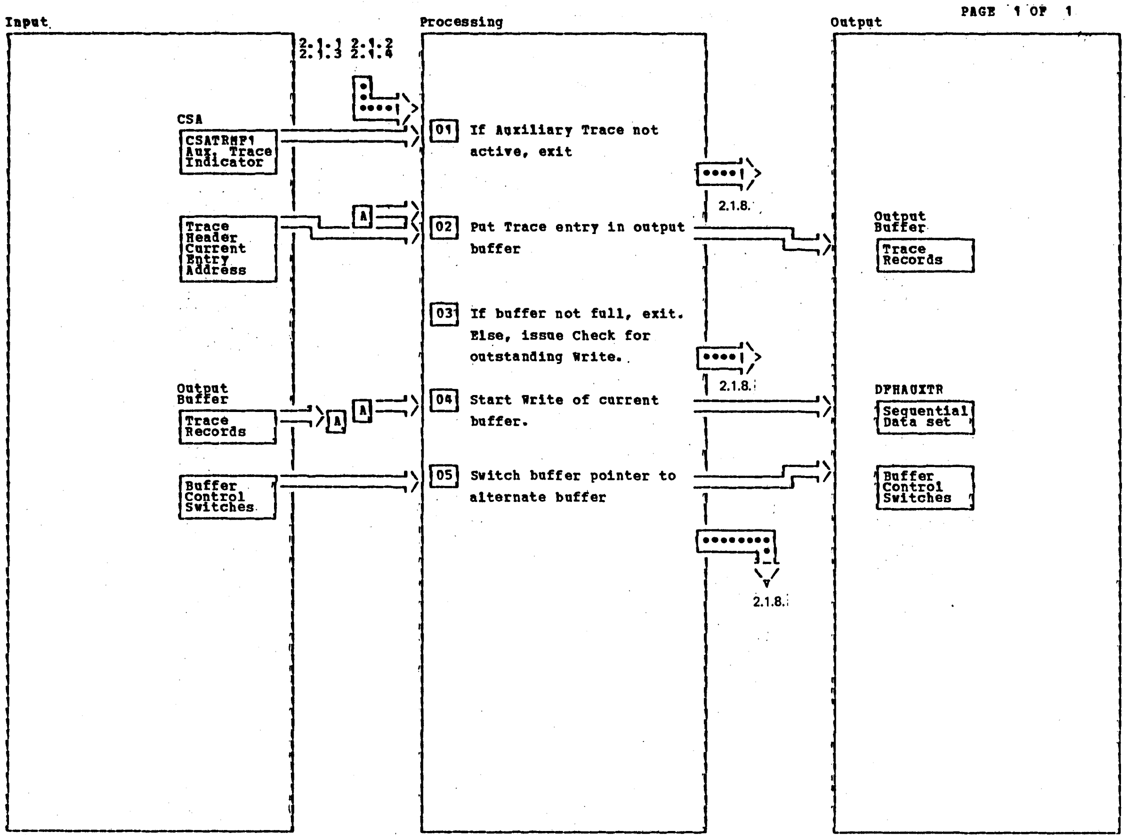
Close Auxiliary Trace

Diagram - 2.1.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | TRAUXCLS | | | | | |
| 02 | | TRAXCLOS | | | | | |
| 03 A check is issued if a previous WRITE is outstanding. The last WRITE is started and a check issued to wait for completion. | | TRAUXC2 | | | | | |
| 04 | | TRAUXSCL | | | | | |
| 05 | | | | | | | |

Close Auxiliary Trace

Diagram - 2.1.6-01


PAGE 1 OF 1
Input
Processing
Output
2.1.1 2.1.2
2.1.3 2.1.4
CSA
CSATRMF1 Aux. Trace Indicator
Trace Header Current Entry Address
Output Buffer
Trace Records
A
Buffer Control Switches
01 If Auxiliary Trace not active, exit
2.1.8.
02 Put Trace entry in output buffer
03 If buffer not full, exit. Else, issue Check for outstanding Write.
2.1.8.
04 Start Write of current buffer.
05 Switch buffer pointer to alternate buffer
2.1.8.
Output Buffer
Trace Records
DFHAUXTR
Sequential Data set
Buffer Control Switches

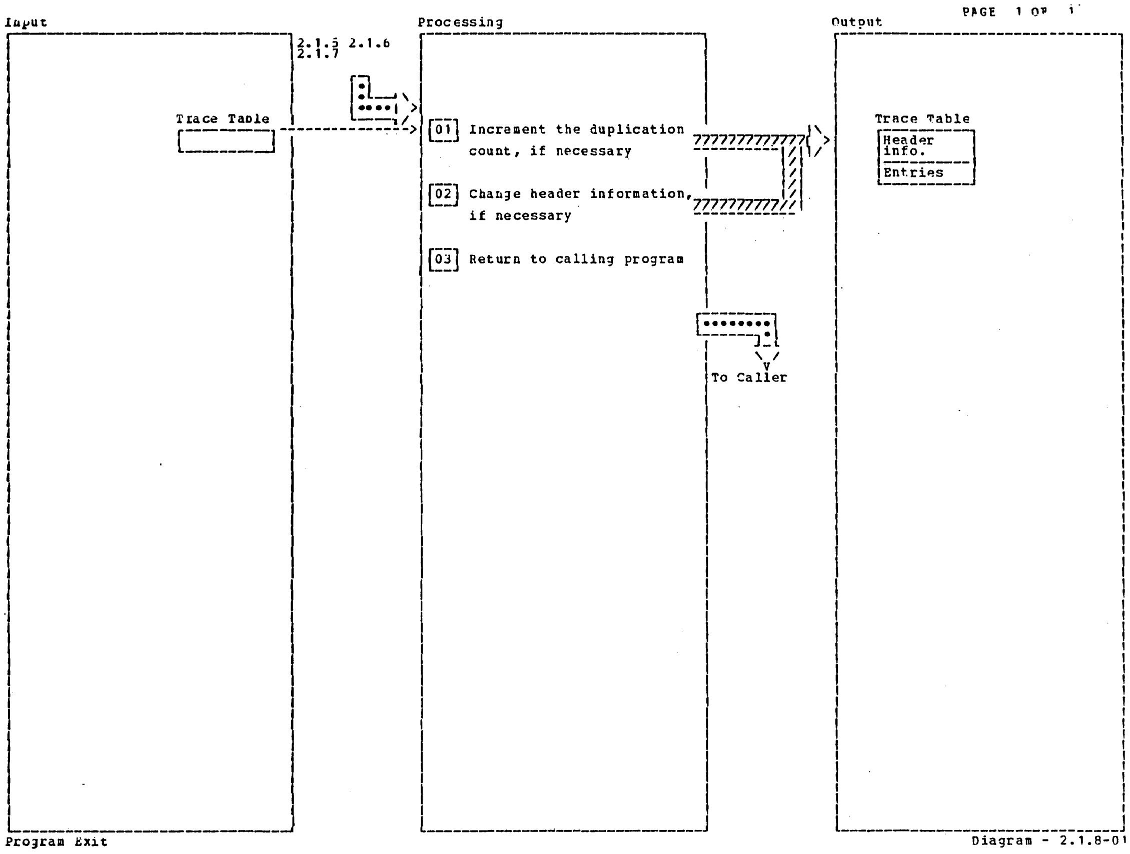
Write Auxiliary Trace Record

Diagram - 2.1.7-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | TRAXENT | | | | | |
| 02 If current entry is X'F0' no records are PUT. If previous entry is X'FD' both the previous and current records are PUT. Otherwise, the current entry is PUT. All entries are time-stamped with the contents of bits 13-44 of STCK. | | | | | | | |
| 03 | | | | | | | |
| 04 | | TRAUXPUT | | | | | |
| 05 Double buffering is used. | | | | | | | |

Write Auxiliary Trace Record

Diagram - 2.1.7-01

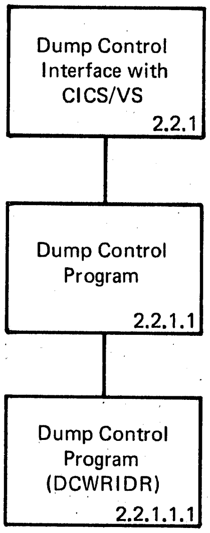


| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The count is incremented if the current entry is the same as a previous entry. | | TREXIT | | | | | |
| 02 If new entry has been built, update header information to show new current entry. | | TRST | | | | | |
| 03 | | TRREST | | | | | |

Program Exit

Diagram - 2.1.8-01

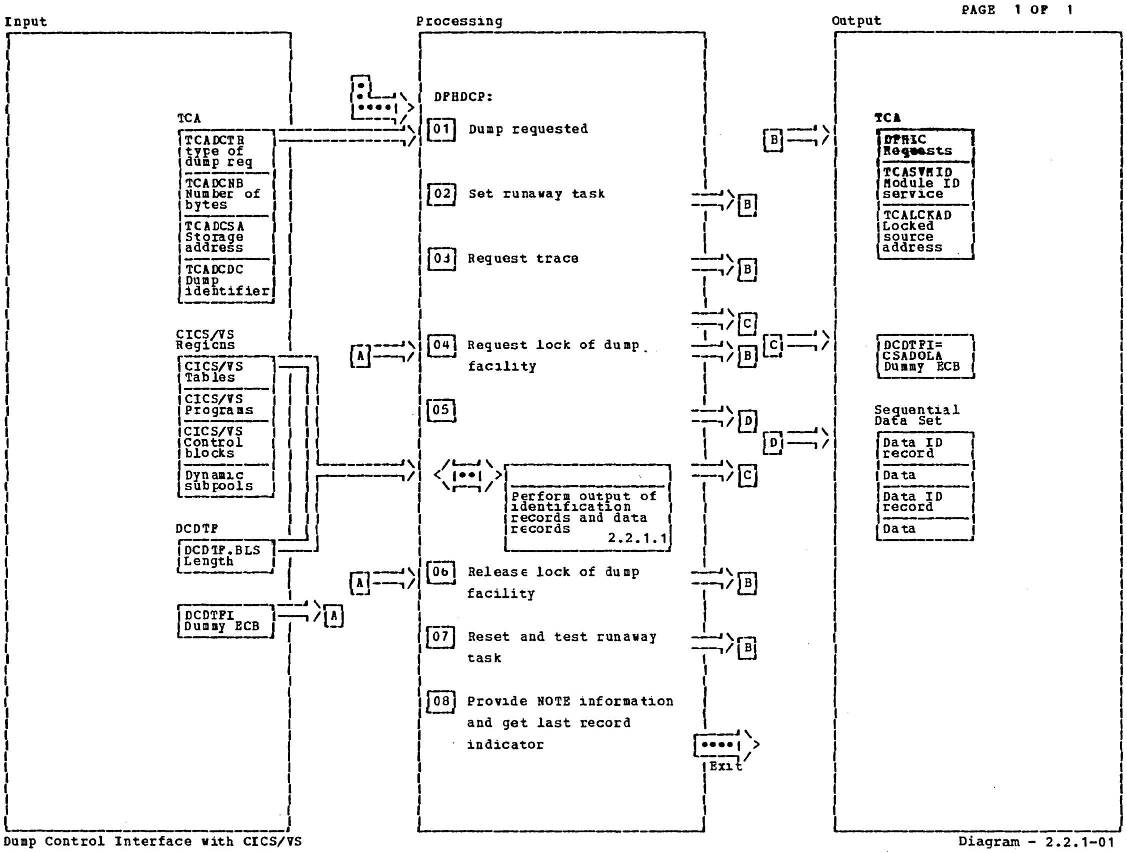
Page 1 of 1


Dump Control Interface with CICS/VS
2.2.1
Dump Control Program
2.2.1.1
Dump Control Program (DCWRIDR)
2.2.1.1.1

Visual Index – Dump Control

Diagram – 2.2.0-01

PAGE 1 OF 1



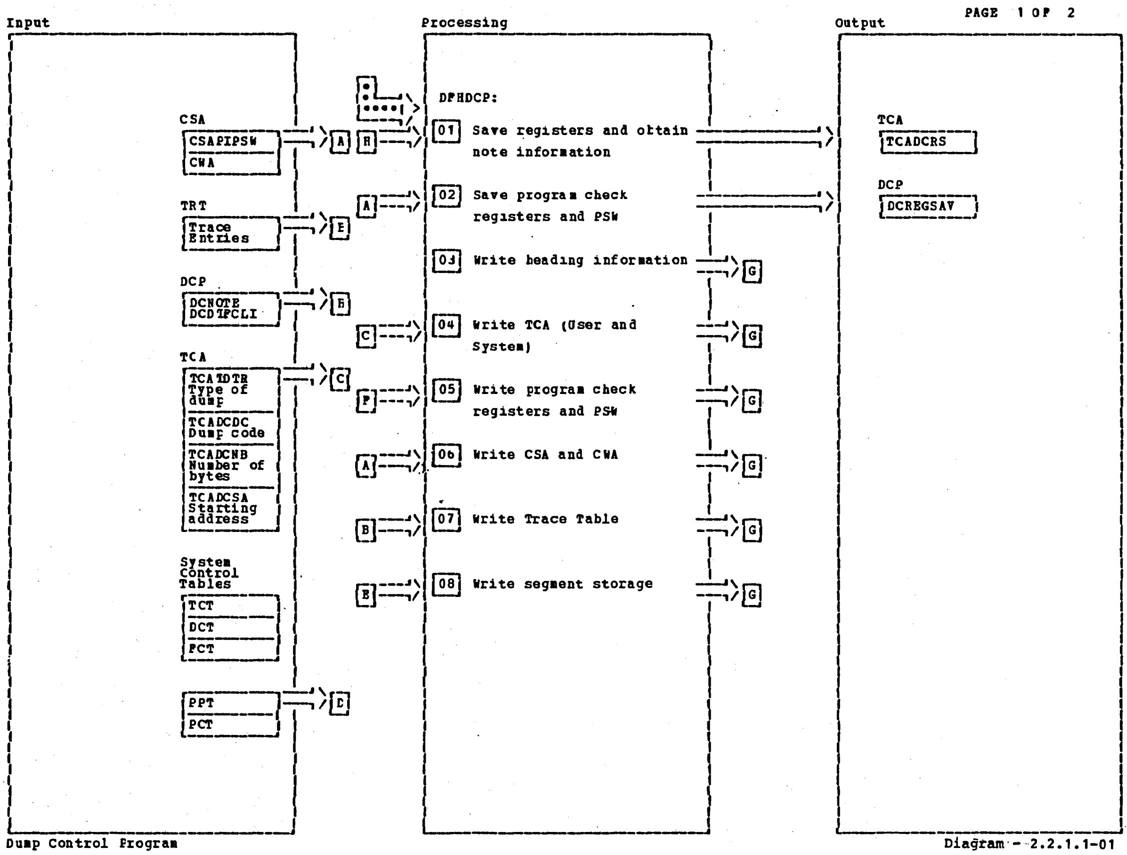
Dump Control Interface with CICS/VS

Diagram - 2.2.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 All dump requests supply TCADCTR and TCADCD. Optionally, TCADCNE and TCADCSA must be provided | | | | | | | |
| 05 Output of data identification records occurs from a fixed area in Dump Control prcgram. | | | 2.2.1.1 | | | | |
| Output of data records occurs from area which data normally occupies. Storage address and length of data records are contained in the registers (DCADRREG and DCLNGREG). | | | | | | | |

Dump Control Interface with CICS/VS

Diagram - 2.2.1-01



Dump Control Program

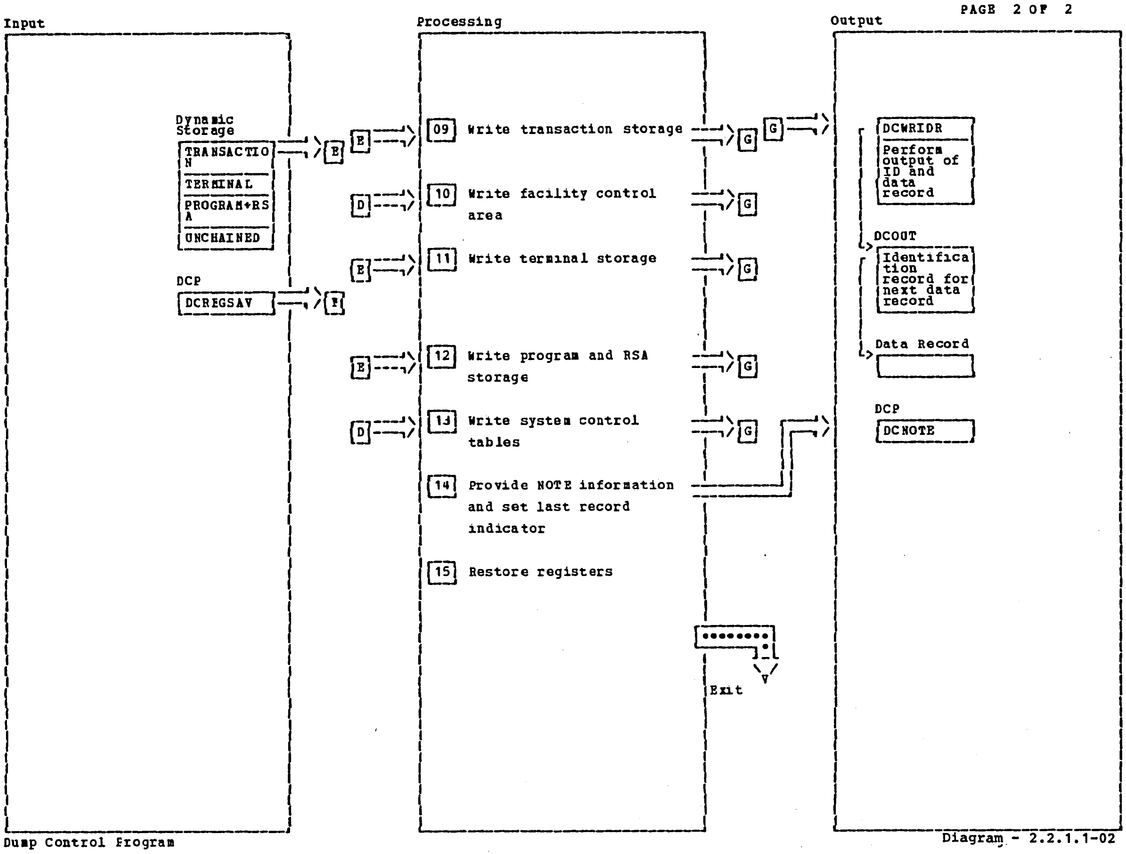
Diagram - 2.2.1.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 08 Conditionally performed based cn code in ICADCTR | | | | | | | |

Dump Control Program

Diagram - 2.2.1.1-01

PAGE 2 OF 2


Input
Processing
Output
Dynamic Storage
TRANSACTION
TERMINAL
PROGRAM+RSA
UNCHAINED
DCP
DCREGSAV
E
D
G
09 Write transaction storage
10 Write facility control area
11 Write terminal storage
12 Write program and RSA storage
13 Write system control tables
14 Provide NOTE information and set last record indicator
15 Restore registers
Exit
DCWRIDR
Perform output of ID and data record
DCOUT
Identification record for next data record
Data Record
DCP
DCNOTE

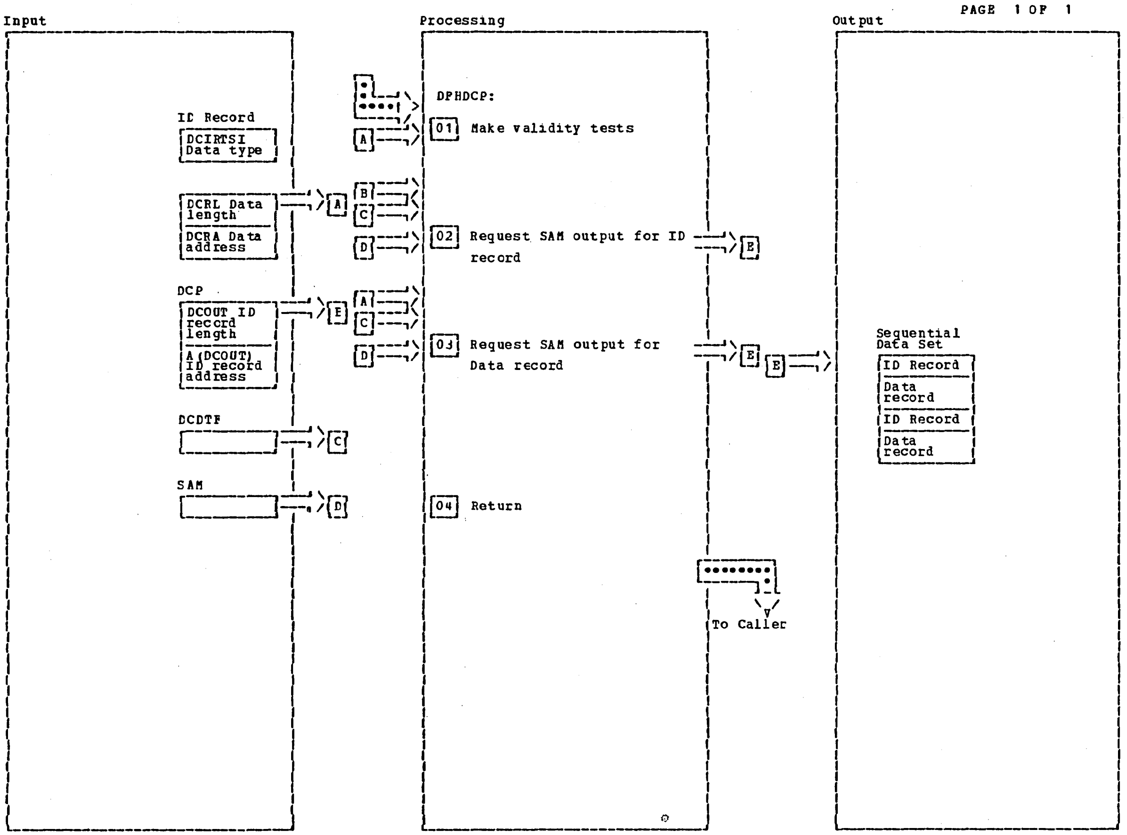
Dump Control Program

Diagram - 2.2.1.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 09 Conditionally performed based cn code in TCACTR | | | | | | | |
| 10 Conditionally performed based on code in TCACTR | | | | | | | |
| 11 Conditionally performed based on code in TCADCTR | | | | | | | |
| 12 Conditicnally performed based on code in TCADCTR | | | | | | | |
| 13 Conditicnally performed based cn code in TCADCTR | | | | | | | |
| 14 In case of EOF, the NOTE information will nct be provided. A Dump data set closed indicator (DCDTFCLI) will be set instead. | | | | | | | |

Dump Control Program

Diagram - 2.2.1.1-02



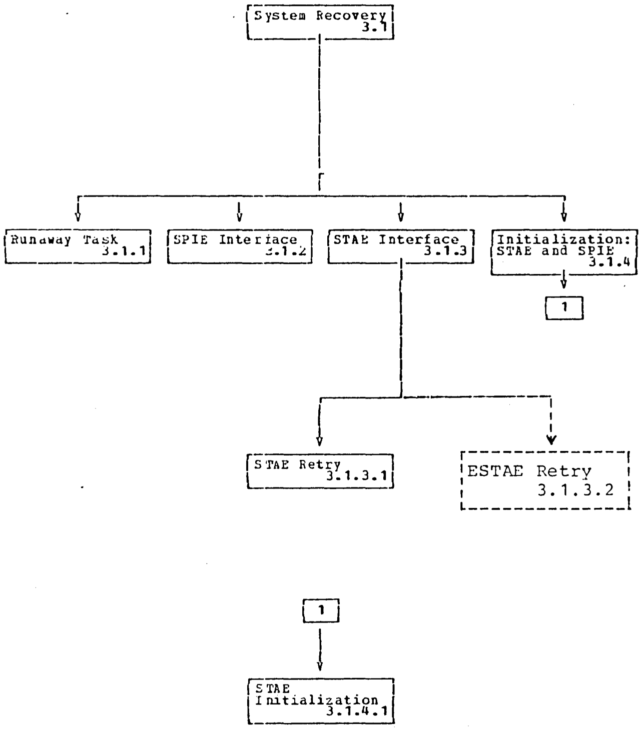
Dump Control Program (DCWRIDR)

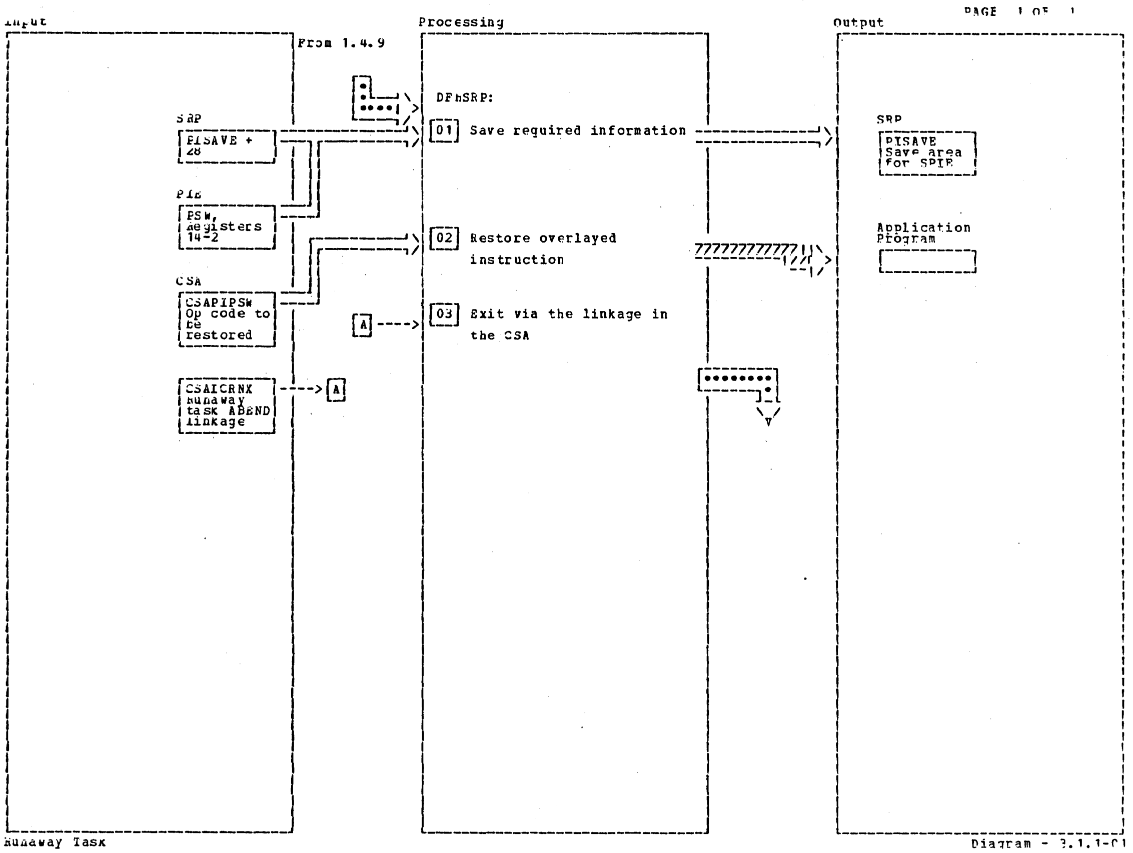
Diagram - 2.2.1.1.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Validate storage accounting. | | | | | | | |

Dump Control Program (DCWRIDR)

Diagram - 2.2.1.1.1-01


System Recovery 3.1
Runaway Task 3.1.1
SPIE Interface 3.1.2
STAE Interface 3.1.3
Initialization: STAE and SPIE 3.1.4
1
STAE Retry 3.1.3.1
ESTAE Retry 3.1.3.2
1
STAE Initialization 3.1.4.1


Input
Processing
Output
PAGE 1 OF 1
From 1.4.9
DFHSRP:
SRP
PISAVE + 28
PIE
PSW, Registers 14-2
CSA
CSAPIPSW Op code to be restored
CSAICRNX Runaway task ABEND linkage
A
01 Save required information
02 Restore overlayed instruction
A
03 Exit via the linkage in the CSA
SRP
PISAVE Save area for SPIE
Application Program

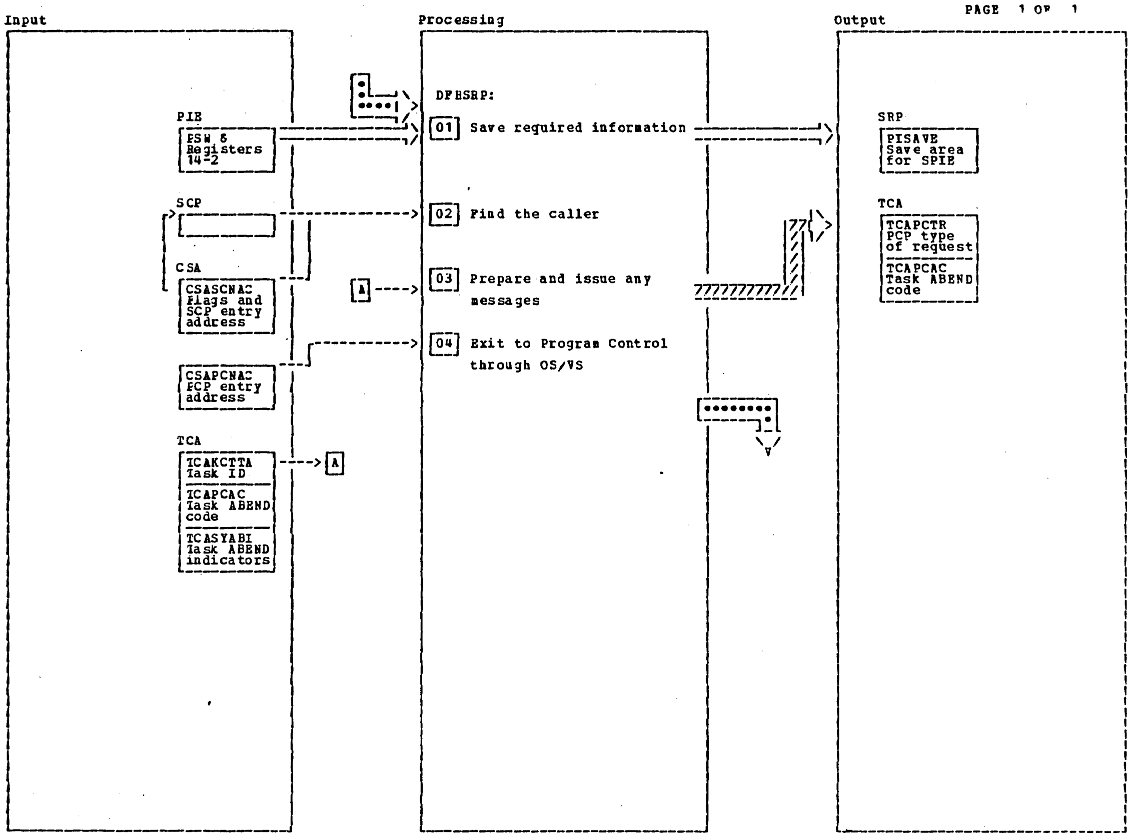
Runaway Task

Diagram - 3.1.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The System Recovery program is entered as a result of a program check caused by the Timer interrupt routine. (One instruction is changed to a NOP.) The op code and the address of the overlayed instruction are saved. The PSW and the registers are also saved. | | PIPSTART | | | | | |
| 02 | | | | | | | |
| 03 The task will later be abnormally terminated. | | PIPRNTK | | | | | |

Runaway Task

Diagram - 3.1.1-01


PAGE 1 OF 1
Input
Processing
Output
DFHSRP:
01 Save required information
02 Find the caller
03 Prepare and issue any messages
04 Exit to Program Control through OS/VS
PIE
PSW & Registers 14-2
SCP
CSA
CSASCNAC Flags and SCP entry address
CSAPCNAC PCP entry address
TCA
TCAKCTTA Task ID
TCAPCAC Task ABEND code
TCASYABI Task ABEND indicators
A
SRP
PISAVE Save area for SPIE
TCA
TCAPCTR PCP type of request
TCAPCAC Task ABEND code

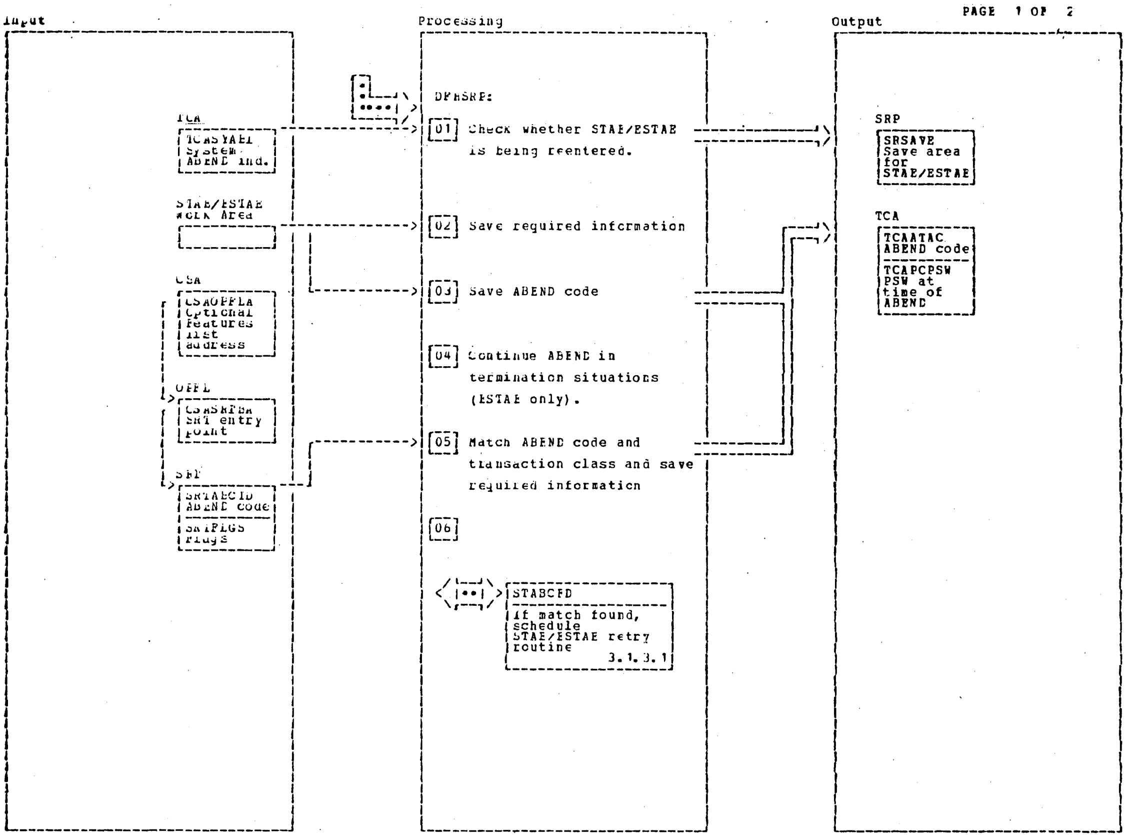
SPIE Interface

Diagram - 3.1.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The PSW and the registers are saved. | | PIPSTART | |
| 02 If the System Recovery program was entered during storage verification, the Storage Control program (SCP) sets a flag in the high-order byte in CSASCNAC. Control is now transferred back to SCP from SRP. The return address is given in the fullword which preceded the SCP entry point. | | | |
| If SRP was entered for any other reason, processing is continued. | | | |
| 03 If a system task is in control, message DFH0601 and OS/VS ABEND 601 are issued. | | PIPCICS | |
| If a recursive program check occurred, message DFH0602 and OS/VS ABEND 602 are issued. | | PIP0602 | |
| If a program check occurred in a STAE exit routine , message DFH0615 and OS/VS ABEND 615 are issued. | | | |
| Otherwise, a task ABEND is prepared with ABEND code ASRA. | | | |
| 04 | | DOCPIP08 | |

SPIE Interface

Diagram - 3.1.2-01


PAGE 1 OF 2
Input
Processing
Output
TCA
TCASYABI System ABEND ind.
STAE/ESTAE Work Area
CSA
CSAOPFLA Optional Features list address
OPFL
CSASRTBA SRT entry point
SRT
SRTABCID ABEND code
SRTFLGS Flags
DFHSRP:
01 Check whether STAE/ESTAE is being reentered.
02 Save required information
03 Save ABEND code
04 Continue ABEND in termination situations (ESTAE only).
05 Match ABEND code and transaction class and save required information
06
STABCFD
If match found, schedule STAE/ESTAE retry routine
3.1.3.1
SRP
SRSAVE Save area for STAE/ESTAE
TCA
TCAATAC ABEND code
TCAPCPSW PSW at time of ABEND

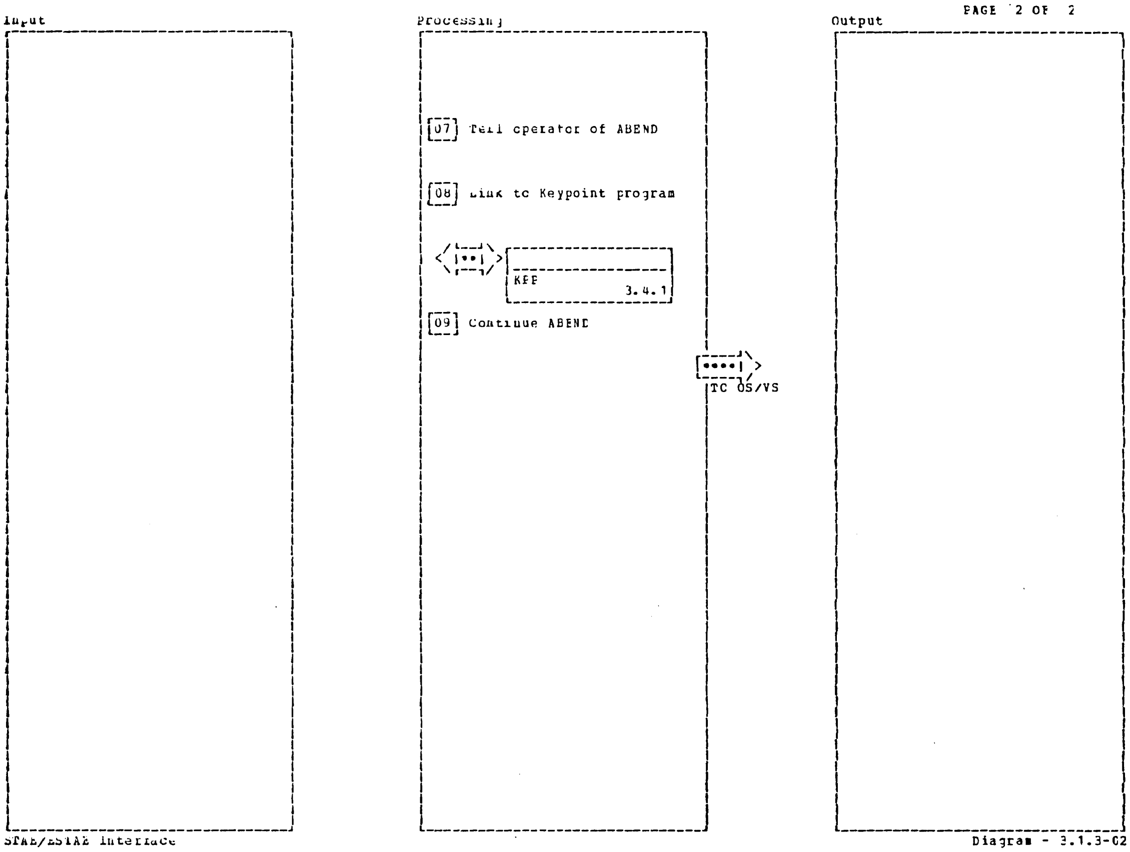
STAE/ESTAE Interface

Diagram - 3.1.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Message DFH0612 and OS/VS ABEND 612 are issued | | STEXRAB | |
| 02 Information is taken from STAE/ESTAE work area unless insufficient core available and then regs 0 and 1 provide some information. | | SRPSTEXR | |
| 03 | | STNCSWA | |
| 04 If CICS is being forced to close down by the system, message DFH0617 is issued and no attempt to recover is made. | | SRISCNLP | |
| 05 The SRT table is searched for a matching code. On STAE systems a 0F3 entry in the SRT will be ignored unless it has been deliberately generated by the user. | | | |
| 06 If ABEND occurred in system task, issue message DFH0613 and go to step 7. Retry routine is scheduled by returning to OS/VS with its address in reg 0, and a 4 in reg 15. | | STABCFD | |

STAE/ESTAE Interface

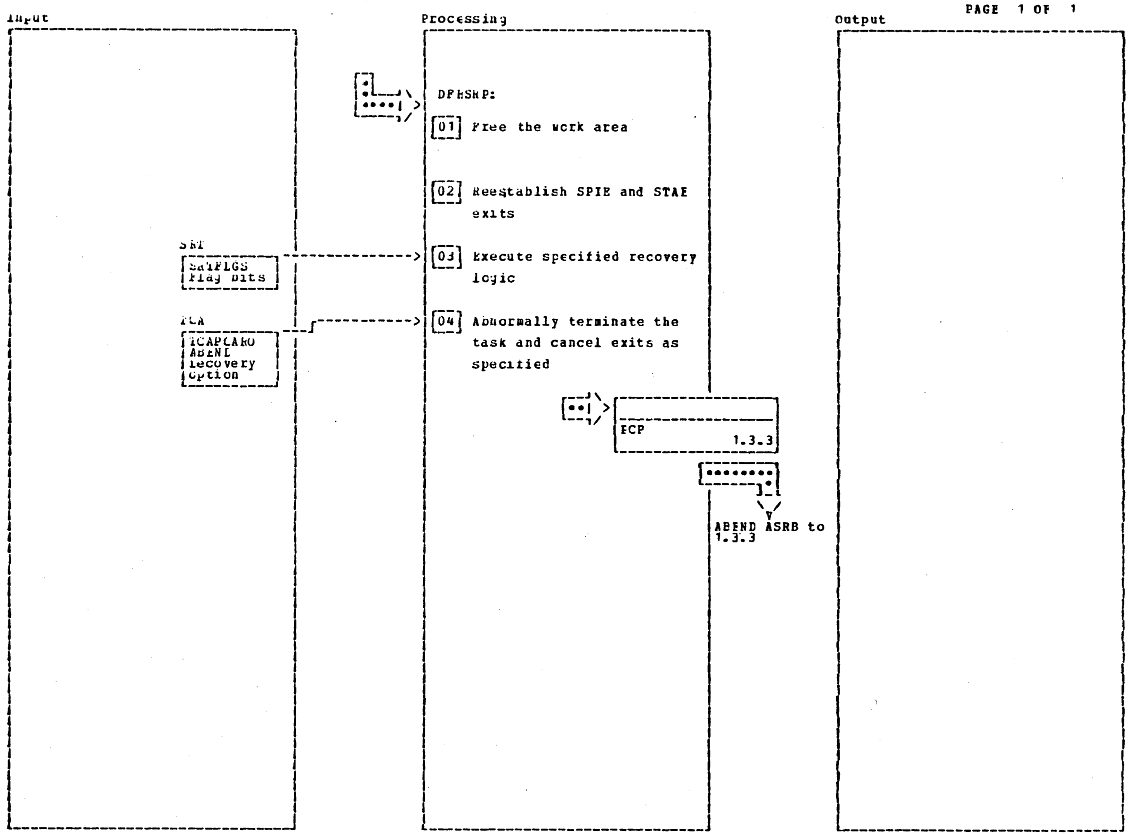
Diagram - 3.1.3-01



| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 07 Write message DFH0606. | | STEXKPP | | | | | |
| 08 Link to KPP if requested in SIT(ABKEOPT parameter) | | SRP*IO | | | | | |
| 09 Indicate ABEND to continue by 0 in reg 15. | | STNKPP | | | | | |

STAE/ESTAE Interface　　　Diagram - 3.1.3-02

PAGE 1 OF 1



DFHSRP:

01 Free the work area

02 Reestablish SPIE and STAE exits

03 Execute specified recovery logic

04 Abnormally terminate the task and cancel exits as specified

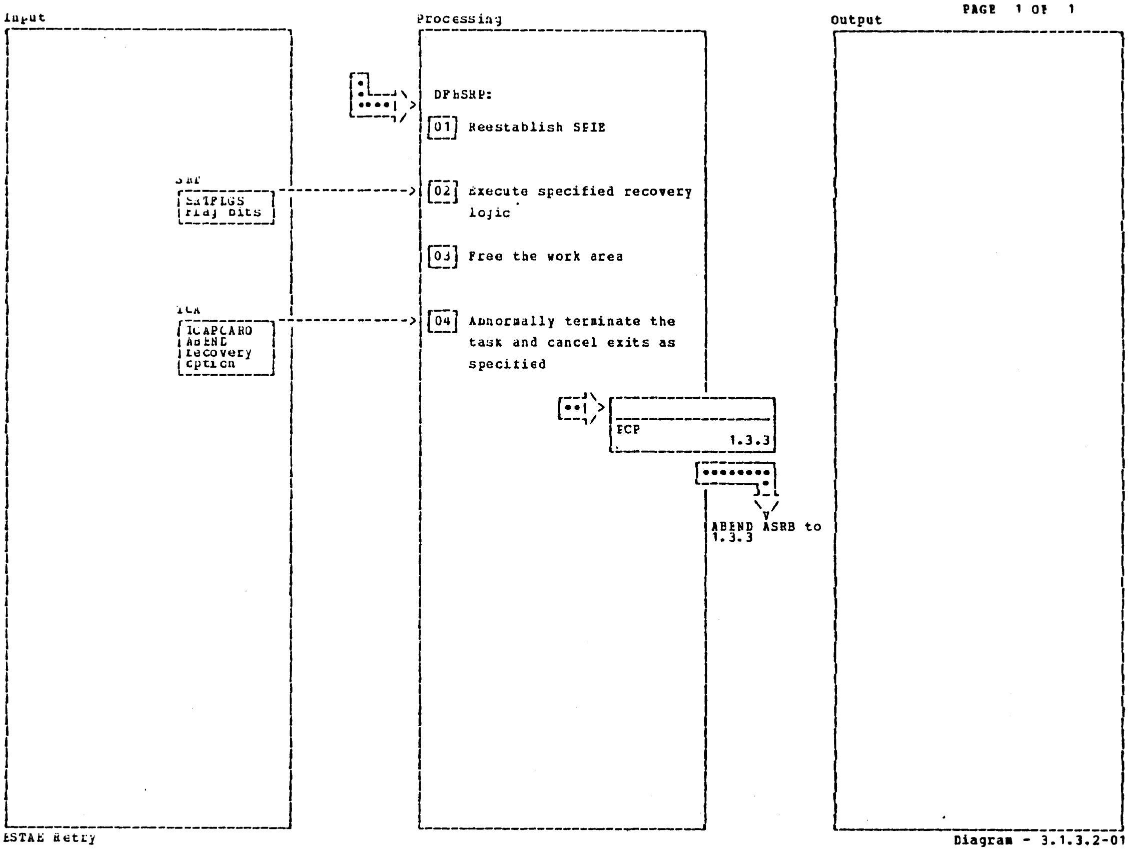
STAE Retry

Diagram - 3.1.3.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 An OS/VS FREEMAIN is issued if a STAE work area was acquired. |  | SRPSIRTR |  |  |  |  |  |
| 02 STAE exits are reestablished via the STAE initialization routine. |  |  | 3.1.4.1 |  |  |  |  |
| 03 Recovery logic can be a program to link to or a routine to branch to. In both cases a value of zero is passed in register 1. For routines an indicator is passed in register 3 showing that this is a STAE system. |  |  |  |  |  |  |  |
| 04 A Program Control ABEND is issued. (ABEND code is ASRB.) |  |  |  |  |  |  |  |

STAE Retry

Diagram - 3.1.3.1-01



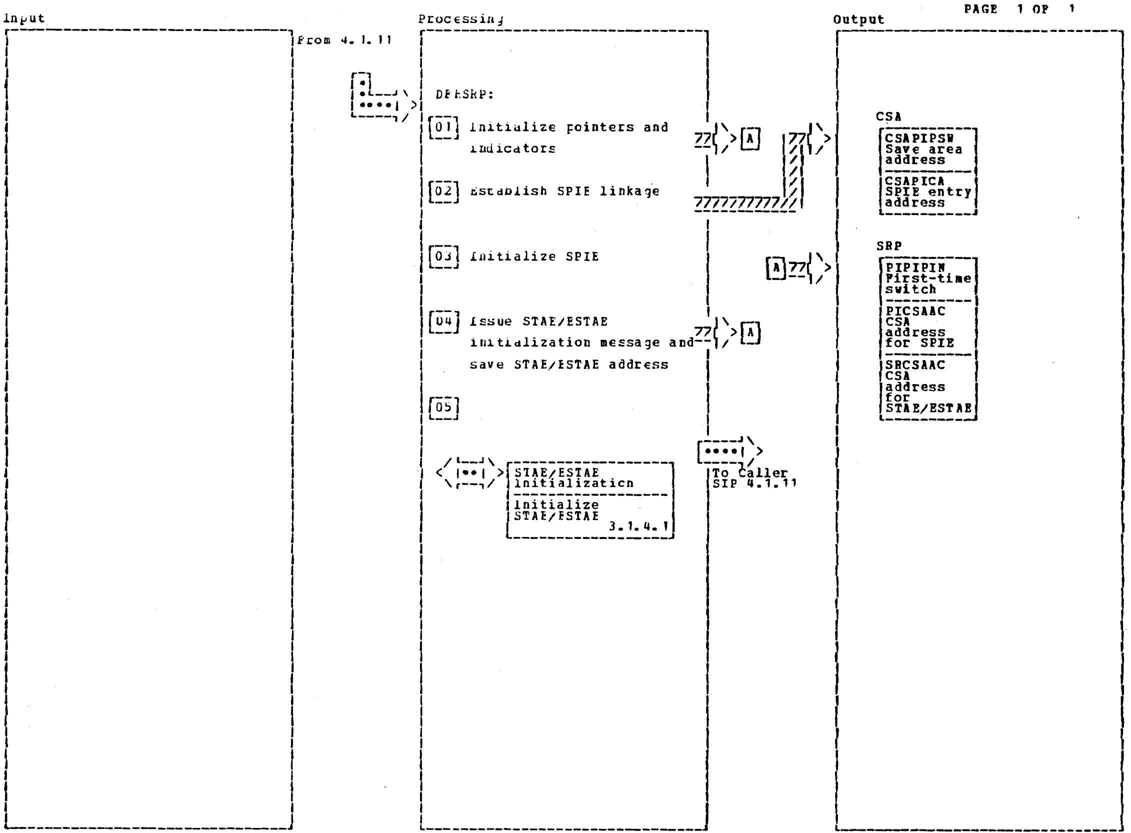
ESTAE Retry

Diagram - 3.1.3.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 STAE exits are restablished via the STAE initialization routine. | | | 3.1.4.1 | | | | |
| 02 Recovery logic can be a program to link to, or a routine to branch to. In both cases the ESTAE work area address is passed in register 1. For routines an indicator is passed in register 3 showing that this is an ESTAE system. | | | | | | | |
| 03 An OS/VS FREEMAIN is issued if an ESTAE work area was acquired. | | SRPSTRTE | | | | | |
| 04 A Program Control ABEND is issued. (ABEND code is ASRB.) | | | | | | | |

ESTAE Retry

Diagram - 3.1.3.2-01


PAGE 1 OF 1
Input
Processing
Output
From 4.1.11
DFHSRP:
01 Initialize pointers and indicators
02 Establish SPIE linkage
03 Initialize SPIE
04 Issue STAE/ESTAE initialization message and save STAE/ESTAE address
05
STAE/ESTAE Initialization
Initialize STAE/ESTAE 3.1.4.1
To Caller SIP 4.1.11
A
CSA
CSAPIPSW Save area address
CSAPICA SPIE entry address
SRP
PIPIPIN First-time switch
PICSAAC CSA address for SPIE
SRCSAAC CSA address for STAE/ESTAE

Initialization: STAE/ESTAE and SPIE

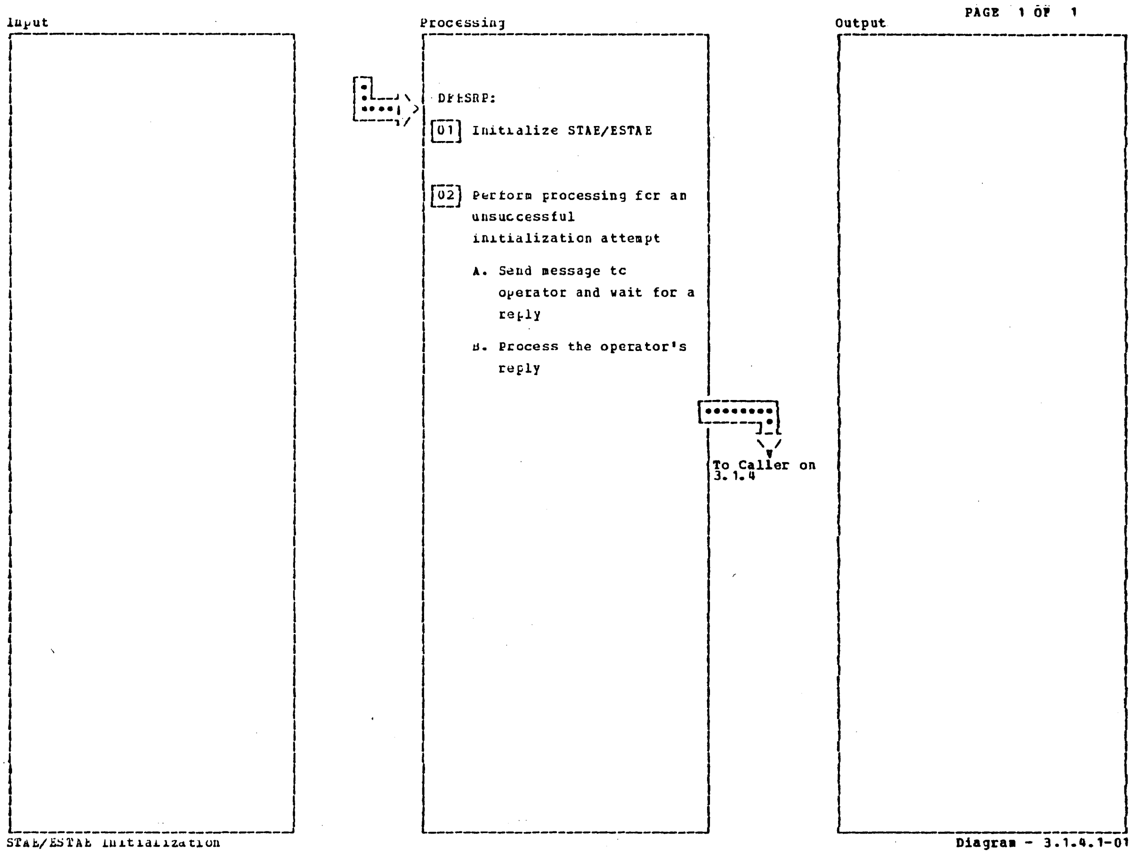
Diagram - 3.1.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Control is received from System initialization program to establish SPIE and STAE/ESTAE exits. The first-time switch is set and the SPIE address is saved. | | DFHSINA | | | | | |
| 02 | | PISVAREA | | | | | |
| 03 | | | | | | | |
| 04 Message DFH1500 is written. | | DFHSTNA | | | | | |
| 05 | | STW1OB | | | | | |

Initialization: STAE/ESTAE and SPIE

Diagram - 3.1.4-01

PAGE 1 OF 1



DFHSRP:

01 Initialize STAE/ESTAE

02 Perform processing for an unsuccessful initialization attempt

A. Send message to operator and wait for a reply

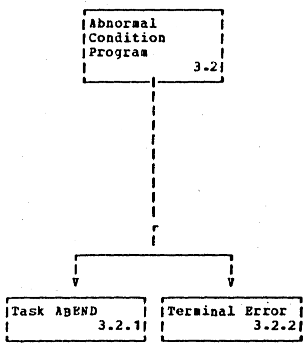
B. Process the operator's reply

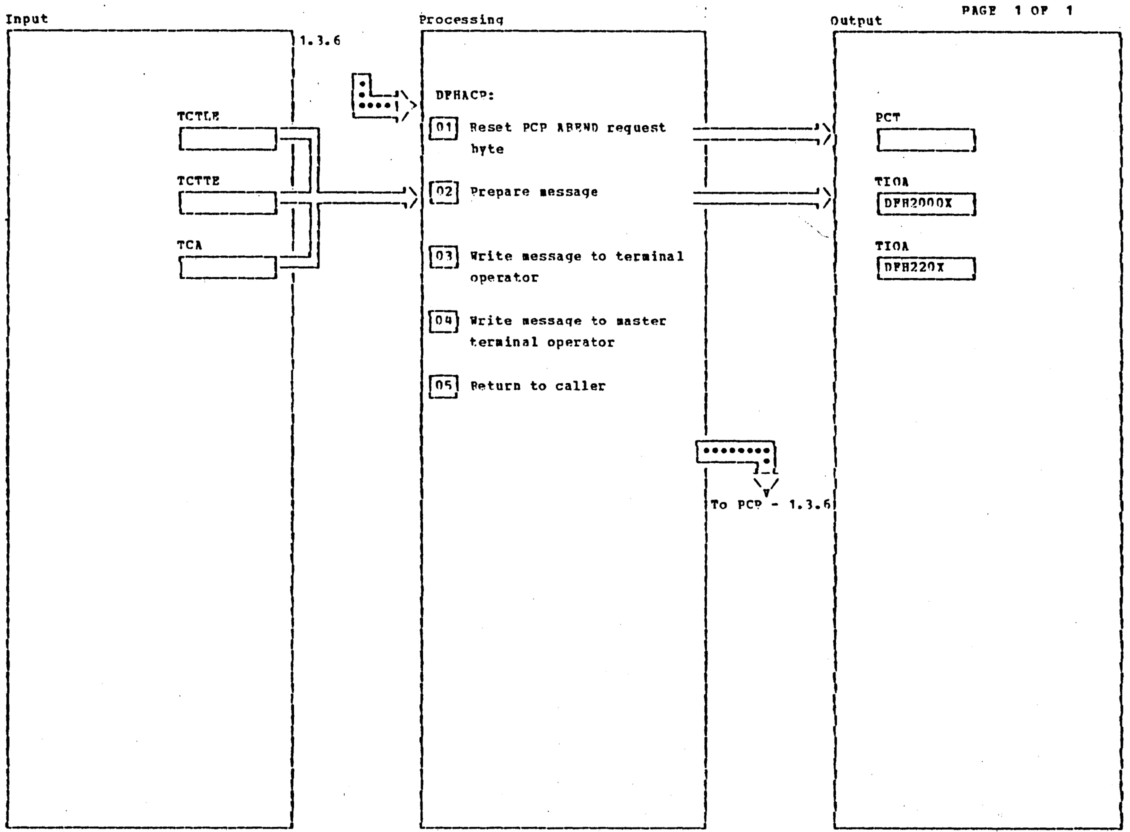
STAE/ESTAE Initialization — Diagram - 3.1.4.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 An OS/VS STAE/ESTAE macro instruction is issued.<br>If successful, control is returned to the caller. | | SRISSTAE | | | | | |
| 02 | | | | | | | |
| A. The operator is asked to specify either GO or CANCEL or CANCEL, DUMP. Messages DFH0604 and DFH0605 are issued. | | SRWTOLP | | | | | |
| B. If GO is specified, initialization continues. If CANCEL or CANCEL, DUMP is specified, CICS/VS is abnormally terminated with ABEND 605 with or without the DUMP parameter, depending on the operator's reply. | | SRPCKRSP | | | | | |

STAE/ESTAE Initialization — Diagram - 3.1.4.1-01

PAGE 1 OF 1


Abnormal Condition Program 3.2
Task ABEND 3.2.1
Terminal Error 3.2.2



Task ABEND

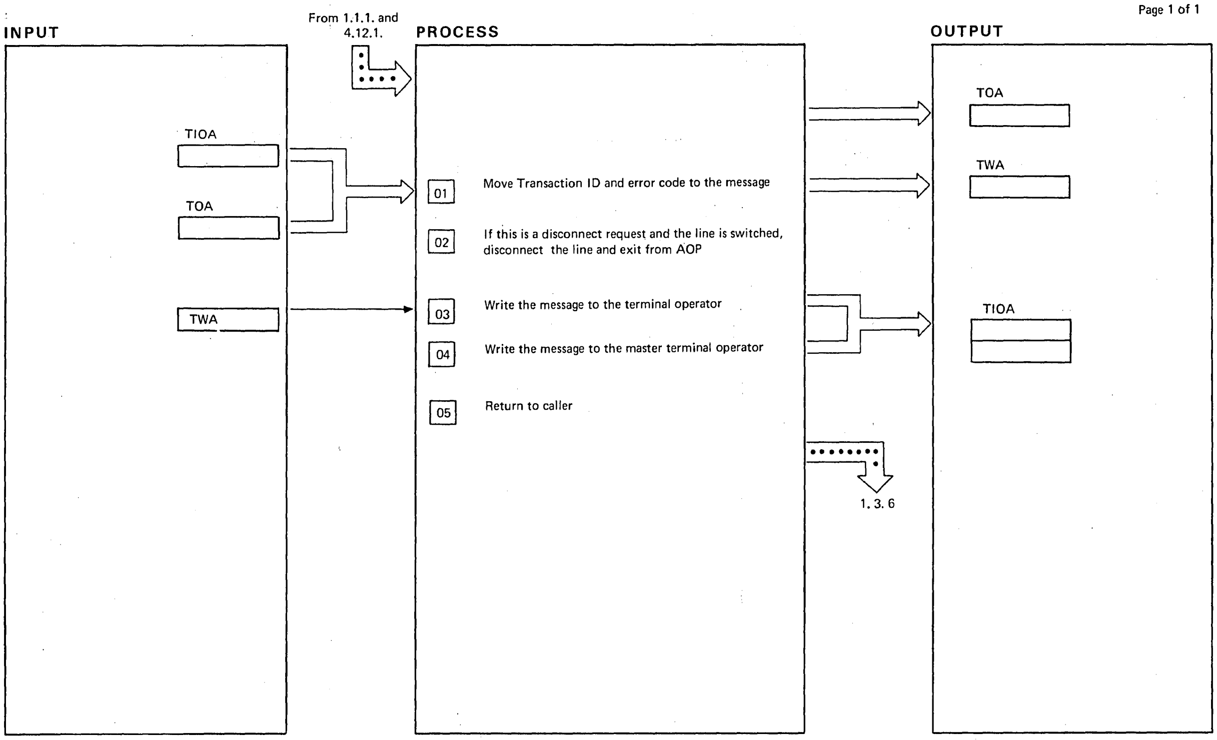
Diagram - 3.2.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 DFH220X messages are issued for Dynamic Transaction Backout tasks that have abended. | | | | | | | |
| 03 Bypass writinq to terminal operator if:<br>- Facility is not a terminal<br>- Facility Control Address is zero<br>- Terminal is out of service<br>- Switched line is not connected<br>- Terminal type is 3735, System/7, or 7770<br>- This is a 3790 running DIP | | | | | | | |
| 04 Bypass writing to master terminal operator if ABEND is due to stall ourge. | | | | | | | |

Task ABEND

Diagram - 3.2.1-01

Page 1 of 1



01 Move Transaction ID and error code to the message

02 If this is a disconnect request and the line is switched, disconnect the line and exit from AOP

03 Write the message to the terminal operator

04 Write the message to the master terminal operator

05 Return to caller

Terminal Error

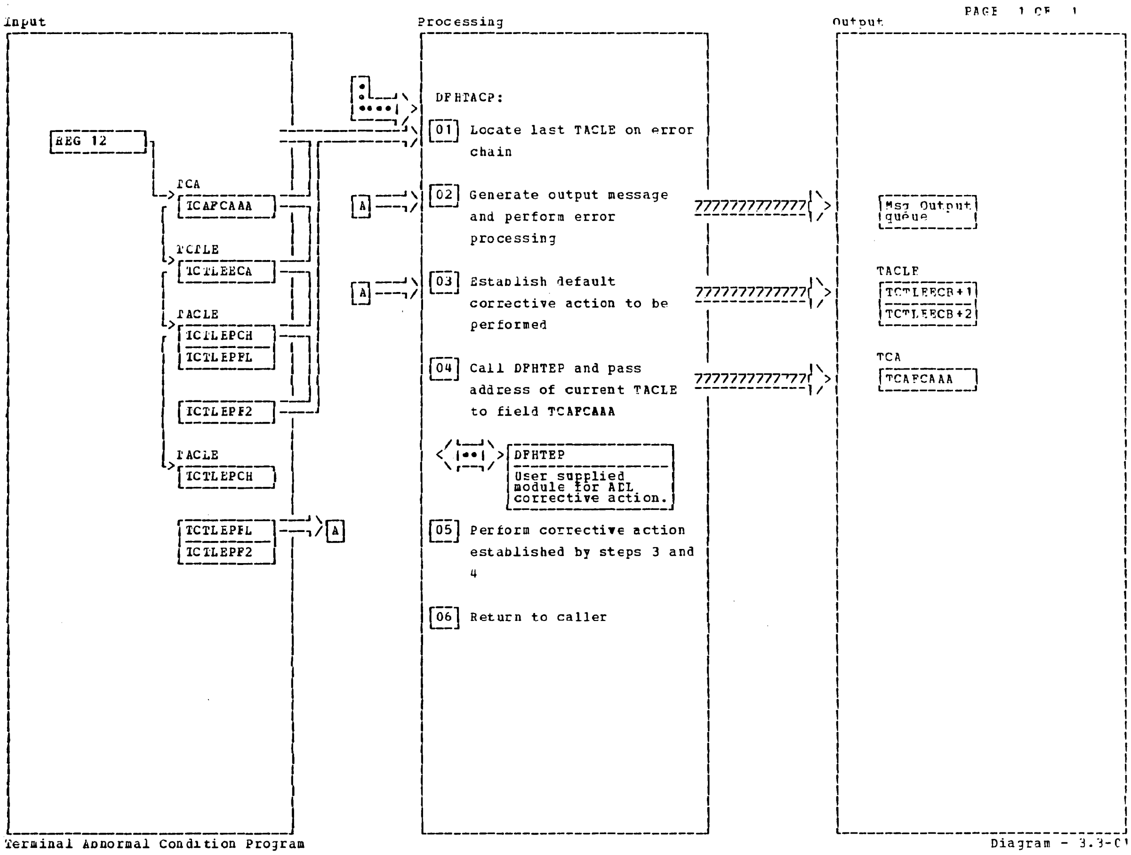
Diagram - 3.2.2-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 | | TRANSPB | |
| 02 If the line is out of service, return from AOP: otherwise, issue a DFHTC TYPE=(DISC, WAIT). | | | |
| 03 | | TRANSBR | |
| 04 | | TRANSBS | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Terminal Error

Diagram - 3.2.2-01



DFHTACP:

01 Locate last TACLE on error chain

02 Generate output message and perform error processing

03 Establish default corrective action to be performed

04 Call DFHTEP and pass address of current TACLE to field TCAFCAAA

DFHTEP
User supplied module for ADL corrective action.

05 Perform corrective action established by steps 3 and 4

06 Return to caller

Terminal Abnormal Condition Program

Diagram - 3.3-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The last TACLE entry is flagged by zero in the TCTLEPCH field. Since this field is used to store the address of a work area, the TCTECHLE bit is set on in the TCTLEPF2 field to indicate the last entry on the chain. The chain end location routine checks for either of these end-of-chain indicators. | | TACOEC | | | | | |
| 02 Processing at this time depends on the type of error that occurred. Refer to Table I for the processing routines and their order of execution. | | TACMSG | | | | | |
| 03 Default action depends on the type of error. Refer to table I for the default actions for each error code. | | TACMCOMP | | | | | |
| 04 DFHTEP is a user-supplied module for additional error analysis and correction. | | TACTEP | | | | | |

Terminal Abnormal Condition Program

Diagram - 3.3-C1

ERROR CODES

PROCESSING ROUTINES

| | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 8A | 8B | 8C | 8D | 8E | 8F | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 9A | 9B | 9C | 9D | 9E | 9F | A0 | A1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 3 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| B | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| C | | | 2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| D | | | | 2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| E | | | | | | 2 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| F | | | | | | | | 2 | | | | | | | | | | | | | | | | | | | | | | | | | |
| G | | | | | | | | | | | 2 | | | | | | | | | | | | | | | | | | | | | | |
| H | | | | | | | | | | | | 2 | | | | | | | | | | | | | | | | | | | | | |
| I | | | | | | | | | | | | | 2 | | | | | | | | | | | | | | | | | | | | |
| J | | | | | | | | | | | | | | 2 | | | | | | | | | | | | | | | | | | | |
| K | | | | | | | | | | | | | | | 2 | | | | | | | | | | | | | | | | | | |
| L | | | | | | | | | | | | | | | | | | | | | 2 | | | | | | | | | | | | |
| M | | | | | | | | | | | | | | | | | | | | | | | 2 | | | | | | | | | | |
| N | | | | 3 | 3 | 3 | 3 | 5 | 3 | 3 | 3 | 5 | 3 | 3 | 3 | 3 | 3 | | | 4 | 5 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 4 | 4 |
| O | | 3 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 2 | | | |
| Q | 5 | | 3 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| R | | 4 | | | 4 | | | | | | 4 | | | 4 | 4 | | 4 | | | | | | | | | | | | | | | 5 | 5 |
| S | | | | 5 | | 6 | 4 | 7 | 4 | 4 | | 7 | 4 | | | 4 | | | | 5 | 6 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | | |
| T | 2 | | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| U | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| V | | | | | 2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| W | | | | | | | | 3 | | | | 3 | | | | | | | | | | | | | | | | | | | | | |
| X | | | | | | | | 4 | | | | 4 | | | | | | | | 3 | 3 | | | | | | | | | | | | |
| Y | | | | | | | | | | | | | | | | 2 | | | | | | | | | | | | | | | | | |
| Z | | | | | | | | | | | | | | | | | 2 | | | | | | | | | | | | | | | | |
| AA | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| AB | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 2 | | |
| AC | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| AD | | | | | | 4 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| AE | | 2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| AF | 4 | | | | | 5 | | | | | | | | | | | | | | | | | | | | | | | | | | | 3 |
| BA | | | | 4 | | | | 6 | | | | 6 | | | | | | | | | | | | | | | | | | | | 3 | |
| BB | | | | | | | | | | | | | | | | | | | | 2 | | | | | | | | | | | | | |
| BC | | | | | | | | | | | | | | | | | | | | | | 2 | | | | | | | | | | | |
| BD | | | | | | | | | | | | | | | | | | | | | 4 | | | | | | | | | | | | |
| BE | | | | | | | | | | | | | | | | | | | | | | | | 2 | | | | | | | | | |
| BF | | | | | | | | | | | | | | | | | | | | | | | | | 2 | | | | | | | | |
| CA | | | | | | | | | | | | | | | | | | | | | | | | | | 2 | | | | | | | |
| CB | | | | | | | | | 2 | | | | | | | | | | | | | | | | | | | | | | | | |
| CC | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CD | | | | | | | | | | | | | | | | | | | | | | | | | | | 2 | | | | | | |
| CE | | | | | | | | | | | | | | | | | | | | | | | | | | | | 2 | | | | | |
| CF | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 2 | | | | |
| DA | | | | | | | | | | 2 | | | | | | | | | | | | | | | | | | | | | | | |
| DB | | | | | | | 2 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DC | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DD | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DE | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 2 | |
| DF | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 2 |

The matrix defines the selection of message routines based on error code. The sequence in which the routines are executed is indicated by the number in the column corresponding to the error code. For example, for error code X'88', the processing routines are executed in the following order: U, F, W, X, N, BA, S. Refer to Figure 2-4 for a list of the processing routines.

OPL6

Figure 2-3. TACP Message Construction Matrix

| Routine | Function |
|---|---|
| A | Generate message "DFH2501 MSG TO LONG, PLEASE RESUBMIT" |
| B | Not used |
| C | Generate message "DFH2503 AUTO OUTPUT HAS BEEN REQ,<br>PLEASE PREPARE TO RECEIVE" |
| D | Generate message prefix "DFH2502 TCT SEARCH ERROR" |
| E | Generate message prefix "DFH2505 POLLING LIST ERROR" |
| F | Generate message prefix "DFH2507 INPUT EVENT REJ" |
| G | Generate message prefix "DFH2512 OUTPUT BUFFER EXCEEDED" |
| H | Generate message prefix "DFH2506 OUTPUT EVENT REJ" |
| I | Generate message prefix "DFH2513 OUTPUT LENGTH ZERO" |
| J | Generate message prefix "DFH2514 NO OUTPUT AREA PROVIDED" |
| K | Generate message prefix "DFH2515 OUTPUT AREA EXCEEDED" |
| L | Generate message prefix "DFH2517 UNIT CHECK SNS=" |
| M | Generate message prefix "DFH2519 UNIT EXCEPTION S.N.O." |
| N | Generate message suffix "AT TERM xxxx, TRANS yyyy, hh.mm.ss"<br>or "ON LINE W/TERM, REL LINE zz, hh.mm.ss"<br>or "ON LINE W/TERM, hh.mm.ss" |
| O | Generate message suffix<br>"LINE, CNTRL, TERM xxxx, [REL LINE zz,] hh.mm.ss."<br>or "LINE, CNTRL W/TERM xxxx, [REL LINE zz,] hh.mm.ss"<br>or "LINE, TERM xxxx, [REL LINE zz,] hh.mm.ss"<br>or "LINE W/TERM xxxx, [REL LINE zz,] hh.mm.ss"<br>or "CNTRL, TERMxxxx, hh.mm.ss"<br>or "CNTRL W/TERM xxxx, hh.mm.ss"<br>or "TERM xxxx, hh.mm.ss" |
| P | Generate message prefix "DFH2508 UNAVAILABLE PRINTER" |
| Q | Write to terminal causing error. |
| R | Write to Destination "CSTL" |
| S | Write to Destination "CSMT" |
| T | Obtain terminal main storage area (message build area) |
| U | Obtain transient data main storage |
| V | Generate message prefix "DFH2531 INVALID WRITE REQUEST" |
| W | Generate message infix "RETURN CODE xx" |
| X | Converts hexadecimal byte into two printable characters |
| Y | Generate message prefix "DFH2513 PRINT QUEUED" |
| Z | Generate message prefix "DFH2531 1C FAILURE X"<br>where X may be "IOERROR" or "TRNIDER" or<br>"TRMIDER" or "INVREQ" |
| AA | Not used |
| AB | Generate message "DFH2534 INVALID DESTINATION" |
| AC | Set terminal out of service |
| AD | Set line out of service |
| AE | Generate message prefix "DFH2504" |
| AF | Obtain terminal statistics |
| BA | Obtain line statistics |
| BB | Generate message prefix "DFH2516 UNIT CHECK SNS=" |
| BC | Generate message prefix "DFH2518 UNIT EXCEPTION" |
| BD | Generate message infix ",S.N.O." |
| BE | Generate message prefix "DFH2520 NEGATIVE RESPONSE" |
| BF | Generate message prefix "DFH2521 UNDETERMINED UNIT ERROR" |
| CA | Generate message prefix "DFH2522 INTERCEPT REQUIRED" |
| CB | Generate message prefix "DFH2526 INTERV ON PRINTER"<br>or "DFH2527 INTERV REQ"<br>or "DFH2528 ERROR STATUS MSG XXXX RECEIVED" |
| CC | Not used |
| CD | Generate message prefix "DFH2523 INVALID COPY REQ" |
| CE | Generate message prefix "DFH2524 INVALID MSG BLOCK" |
| CF | Generate message prefix "DFH2525 INCMPLT MSG" |
| DA | Generate message prefix "DFH2510 TIME OUT" |
| DB | Generate message prefix "DFH2529 UNSOLICITED INPUT" |
| DC | Not used |
| DD | Not used |
| DE | Generate message prefix "DFH2530 INVALID READ REQUEST" |
| DF | Generate message prefix "DFH2509 INVALID DISC REQUEST" |

Note: Output messages may be composed of up to three parts in the form of "PREFIX INFIX SUFFIX". These three parts are put together by the various routines and then output to one of three possible destinations depending on the type of error.

OPL7

Figure 2-4. TACP Message Routines

| ERROR CODES | ROUTINE ORDER | | | |
|---|---|---|---|---|
| 81 | 8 | 1 | 2 | 7 |
| 83 | 7 | | | |
| 84 | 3 | 1/2 | 7 | |
| 85 | 1 | 2 | 7 | |
| 86 | 2 | 7 | | |
| 87 | 2 | 7 | | |
| 88 | 5 | 1 | 2 | 7 |
| 89 | 6 | 1 | 2 | 7 |
| 8A | 8 | 1 | 2 | 7 |
| 8B | 1 | 2 | 7 | |
| 8C | 5 | 1 | 2 | 7 |
| 8D | 1 | 2 | 7 | |
| 8E | 1 | 2 | 7 | |
| 8F | 1 | 2 | 7 | |
| 90 | 4 | 7 | | |
| 91 | 9 | 2 | 7 | |
| 94 | 6 | 1 | 2 | 7 |
| 95 | 6 | 1 | 2 | 7 |
| 96 | 6 | 1 | 2 | 7 |
| 97 | 6 | 1 | 2 | 7 |
| 98 | 6 | 1 | 2 | 7 |
| 99 | 5 | 1 | 2 | 7 |
| 9A | 7 | | | |
| 9B | 1 | 2 | 7 | |
| 9C | 6 | 1 | 2 | 7 |
| 9D | 6 | 1 | 2 | 7 |
| 9E | 2 | 7 | | |
| A0 | 1 | 2 | 7 | |
| A1 | 2 | 7 | | |

ROUTINE DESCRIPTION

1. Abend transaction
2. Link to DFHTEP
3. Put line in/out of service, as required
4. Unavailable printer processing
5. Put line or printer out of service
6. I/O error test
7. Test line for next operation
8. Set line disconnect required
9. Unavailable printer interval control error

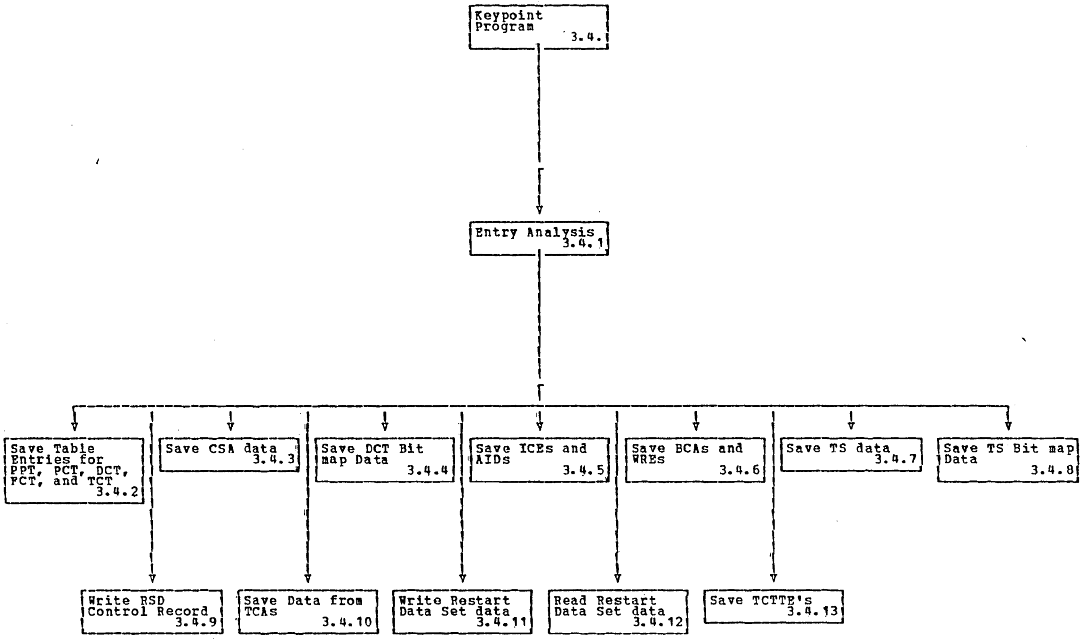
*Notes:*

- The information given above is a generalization of TACP's default error handling.
- The left-hand column contains the error code.
- The right-hand column shows the routines used and the order in which they are used.

*OPL82*

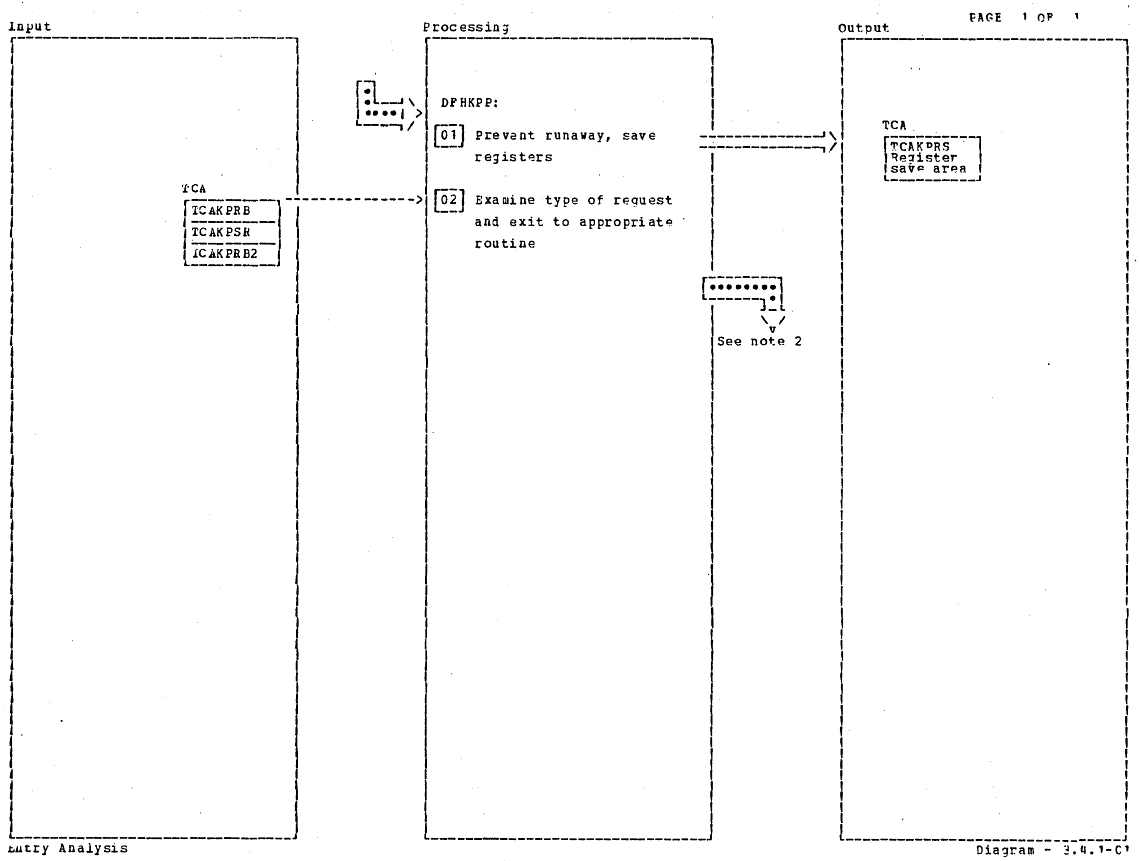
Figure 2-5. TACP Default Error Handling

PAGE 1 OF 1


Keypoint Program 3.4.
Entry Analysis 3.4.1
Save Table Entries for PPT, PCT, DCT, FCT, and TCT 3.4.2
Save CSA data 3.4.3
Save DCT Bit map Data 3.4.4
Save ICEs and AIDs 3.4.5
Save BCAs and WREs 3.4.6
Save TS data 3.4.7
Save TS Bit map Data 3.4.8
Write RSD Control Record 3.4.9
Save Data from TCAs 3.4.10
Write Restart Data Set data 3.4.11
Read Restart Data Set data 3.4.12
Save TCTTE's 3.4.13

Keypoint Program

HIPOMAT 1.1 Diagram - 3.4.-01

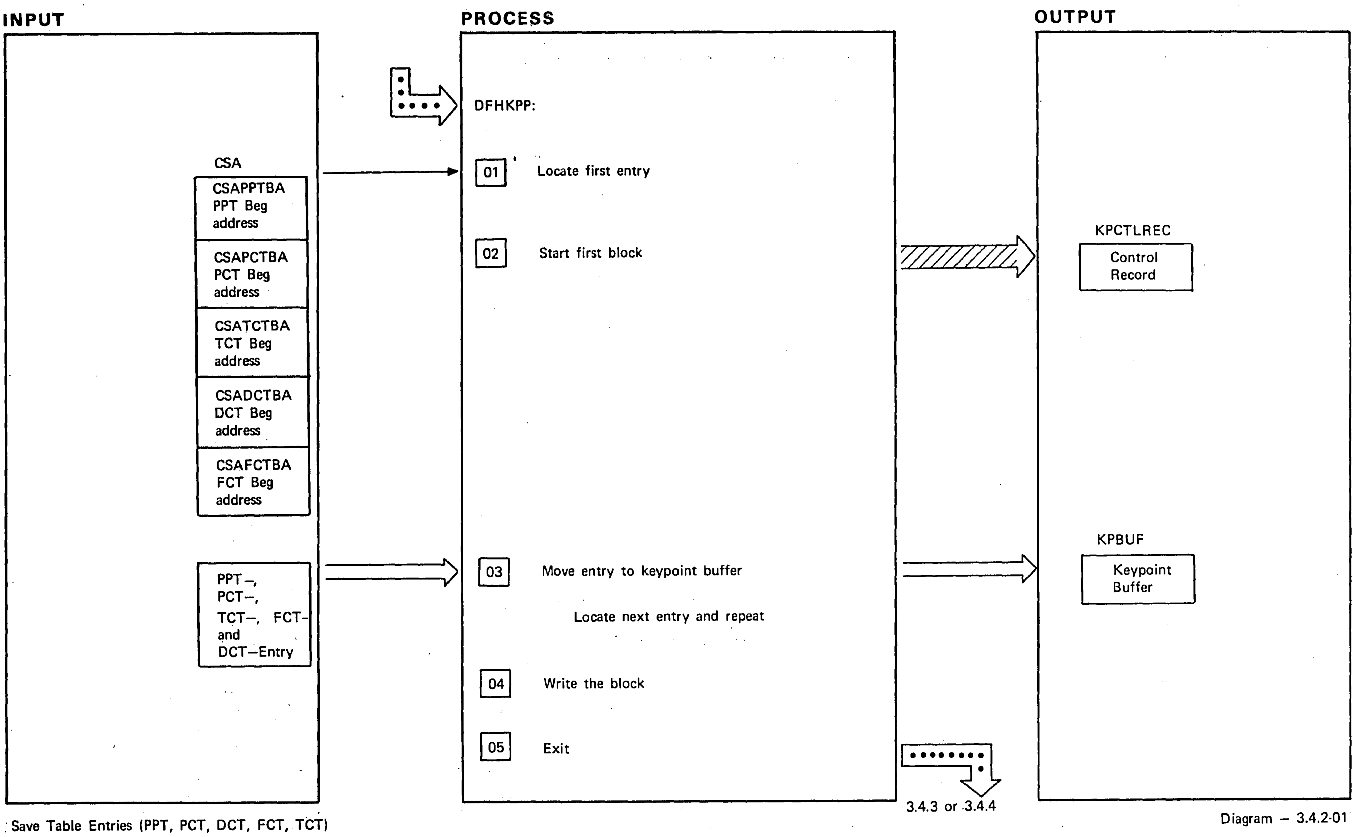

Input
Processing
Output
PAGE 1 OF 1
DFHKPP:
01 Prevent runaway, save registers
02 Examine type of request and exit to appropriate routine
TCA
TCAKPRB
TCAKPSR
TCAKPRB2
TCA
TCAKPRS Register save area
See note 2
Entry Analysis
Diagram - 3.4.1-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | DFHKPPNA | | | | | |
| 02 For warm keypoint | | | 3.4.2 | | | | |
| For activity keypoint: | | | | | | | |
| of DCT | | | 3.4.2 | | | | |
| of TCA | | | 3.4.10 | | | | |
| of TCTTE's | | | 3.4.13 | | | | |
| of all (DCT) | | | 3.4.2 | | | | |
| For Write Restart Data Set data | | | 3.4.11 | | | | |
| For Read Restart Data Set data | | | 3.4.12 | | | | |

Entry Analysis

Diagram - 3.4.1-C1

Page 1 of 1



Save Table Entries (PPT, PCT, DCT, FCT, TCT)

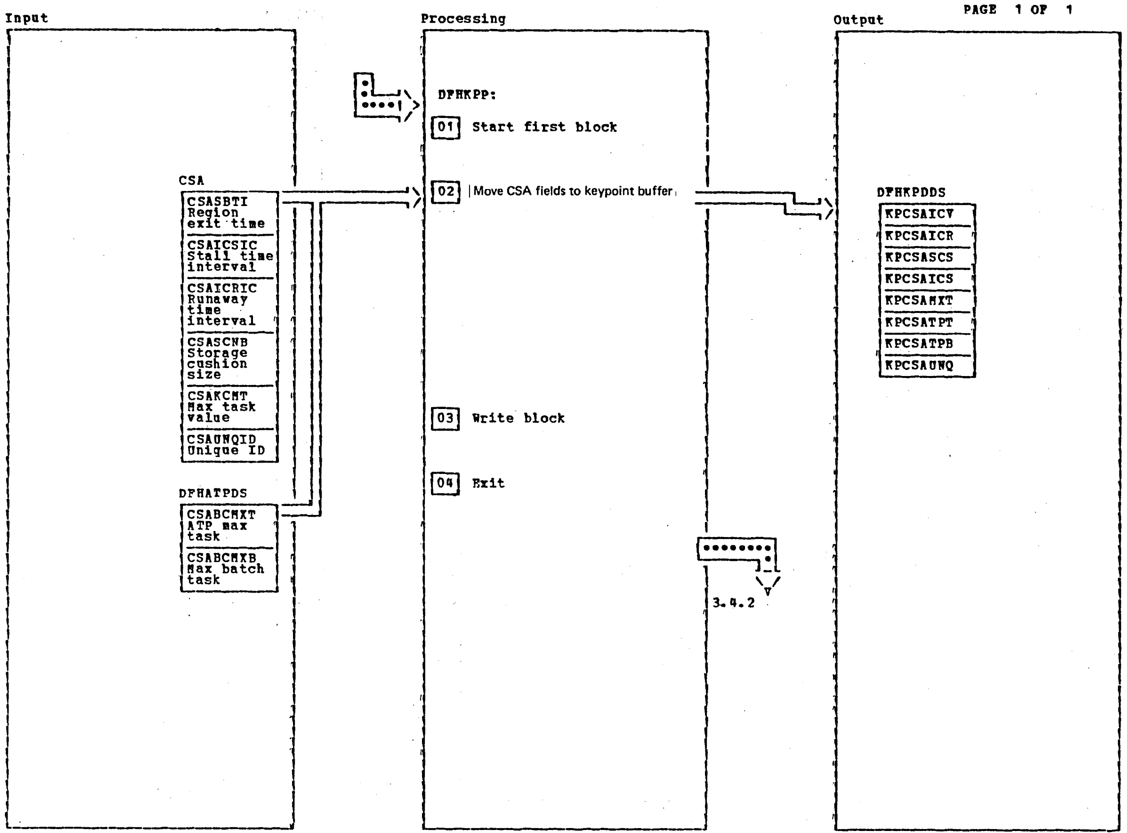
Diagram – 3.4.2-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 This logic is common for: | | | |
| – PPT | | KPPPT | |
| – PCT | | KPPCT | |
| – DCT | | KPDCT | |
| – FCT | | KPDCT | |
| – TCT | | KPDCT | |
| If no entry, the facility is marked absent in the control record and exit is taken. | | | |
| 02 | | KPBLK1 | |
| 04 | | KPWRITE | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Save Table Entries (PPT, PCT, DCT, FCT, TCT)

Diagram – 3.4.2-01



Save CSA Data

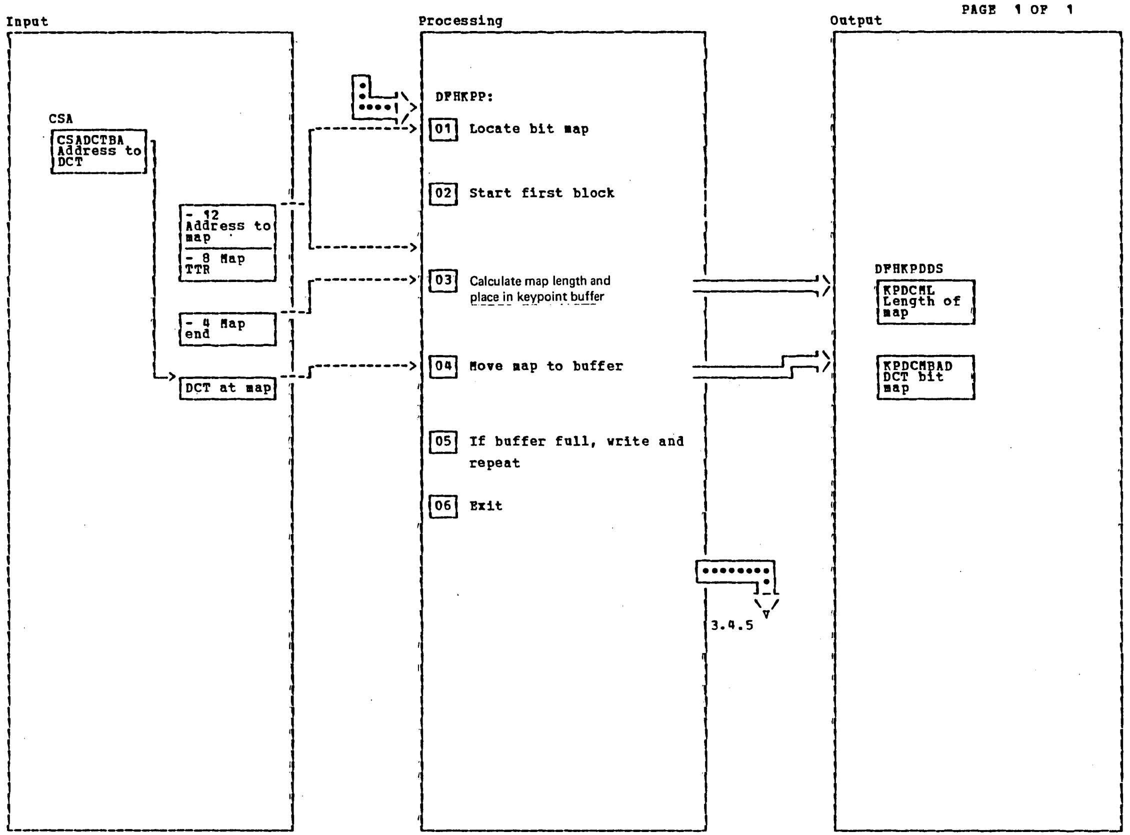
Diagram - 3.4.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | KPBLK1 | | | | | |
| 02 If ATP is not present, KPCSATPT and KPCSATPB are set to zeros | | | | | | | |
| 03 | | KPWRITE | | | | | |

Save CSA Data

Diagram - 3.4.3-01

PAGE 1 OF 1



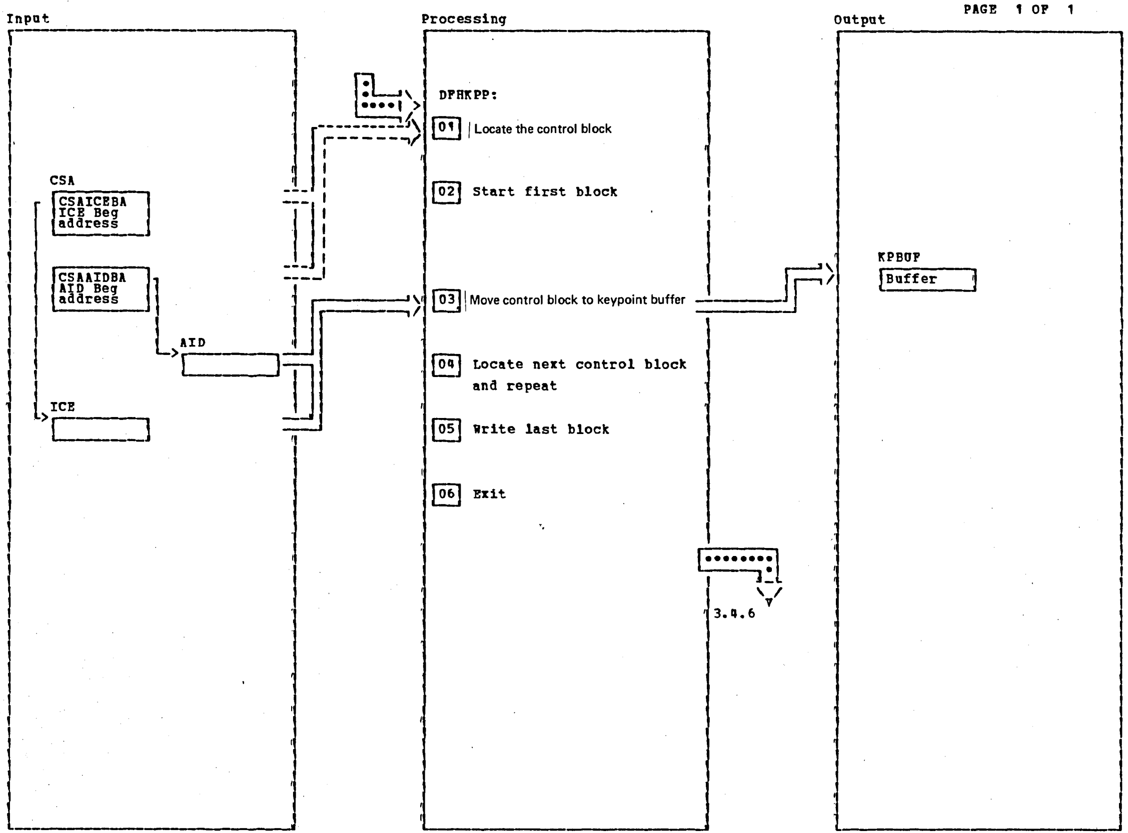
Save DCT Bit Map

Diagram - 3.4.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

Save DCT Bit Map

Diagram - 3.4.4-01



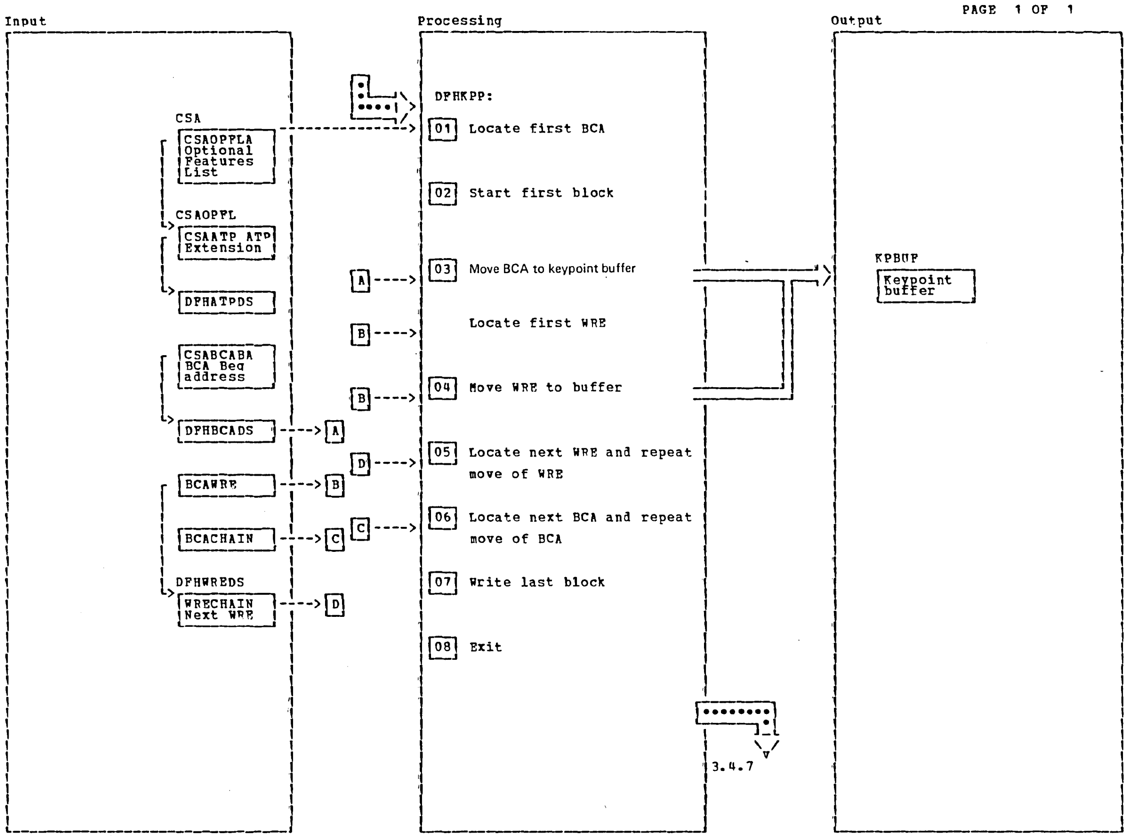
Save ICEs and AIDs

Diagram - 3.4.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 This logic is common for ICEs and AIDs | | | | | | | |
| - For ICEs | | KPICE | | | | | |
| - For AIDs | | KPAID | | | | | |
| If request is from DFHSTP, the addresses are in the TWA. | | | | | | | |
| If either types of control blocks are absent, this is indicated in the control record and exit is taken. | | | | | | | |
| 03 In case of ICE, the julian date is also moved to the buffer. | | | | | | | |

Save ICEs and AIDs

Diagram - 3.4.5-01



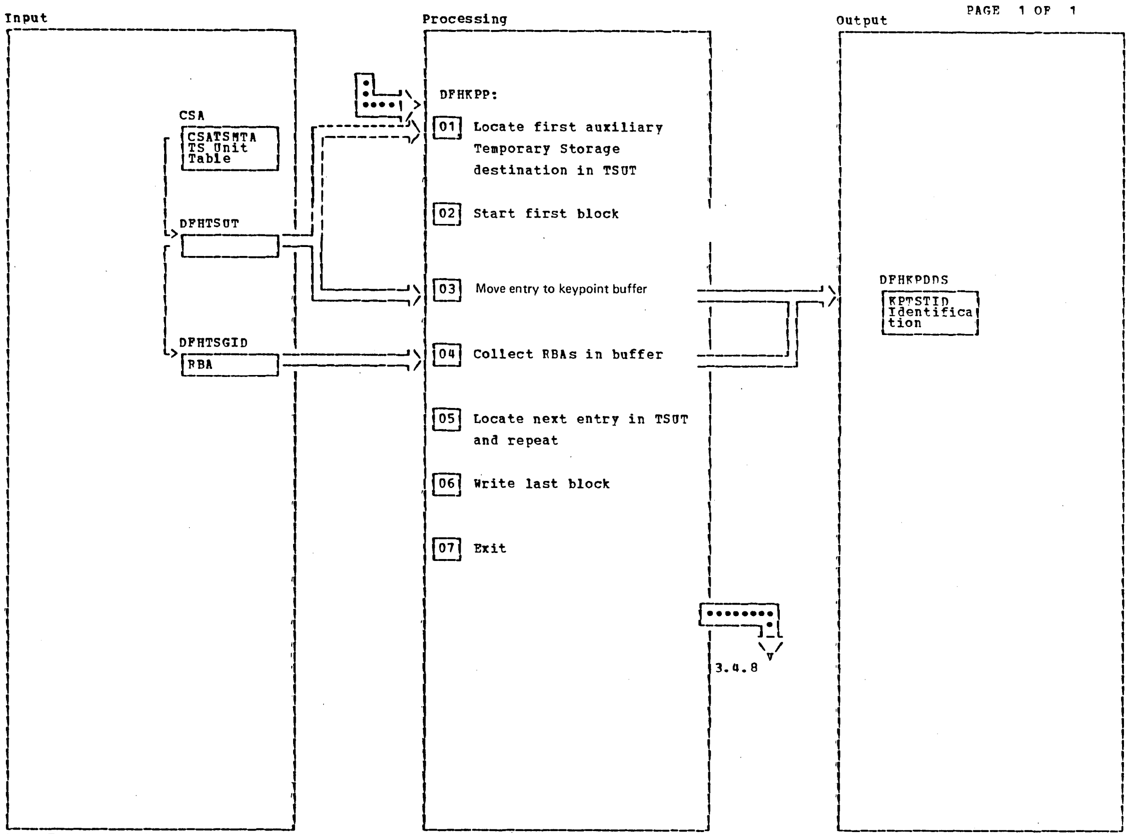
Save BCAs and WREs

Diagram - 3.4.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If request from DFHSTP, address is in TWA. If BCA not present, so is indicated in the control record, and exit taken. | | KPBCA | | | | | |

Save BCAs and WREs

Diagram - 3.4.6-01



PAGE 1 OF 1

DFHKPP:

01 Locate first auxiliary Temporary Storage destination in TSUT

02 Start first block

03 Move entry to keypoint buffer

04 Collect RBAs in buffer

05 Locate next entry in TSUT and repeat

06 Write last block

07 Exit

Save Temporary Storage Data

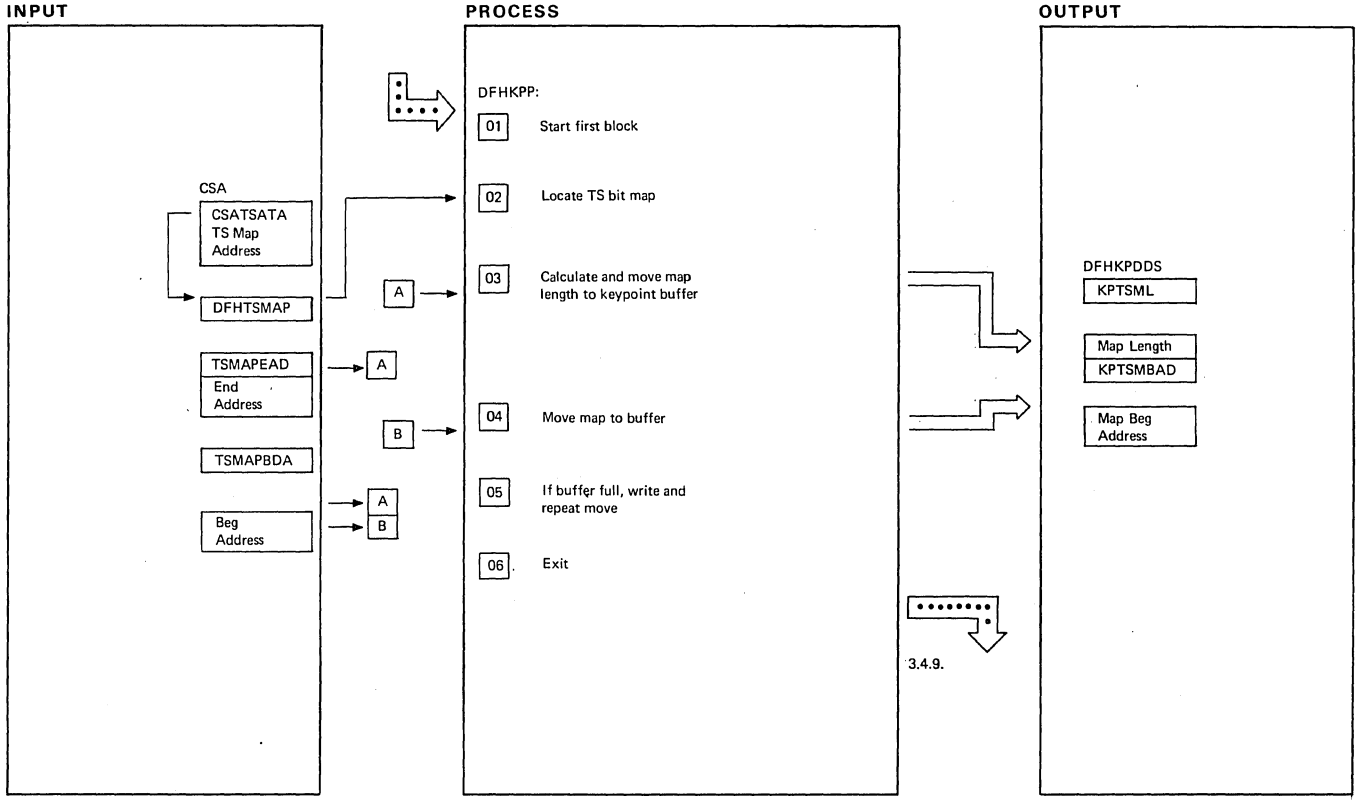
Diagram - 3.4.7-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If no TS in system, indicate this in the control record and exit. | | | | | | | |
| The locate is done by scanning the unit tables. | | KPTST00A | | | | | |
| If no aux entry found, this is indicated in the control record, and exit is taken (bit map data collection is also bypassed). | | | | | | | |
| 03 For single-RBA-dest. | | KPTST02A | | | | | |
| For mult-RBA-dest. | | KPTST05A | | | | | |
| 04 Only in case of mult-RBA-destination. | | KPTST07 | | | | | |

Save Temporary Storage Data

Diagram - 3.4.7-01

PAGE 1 OF 1



Save TS Bit Map Data

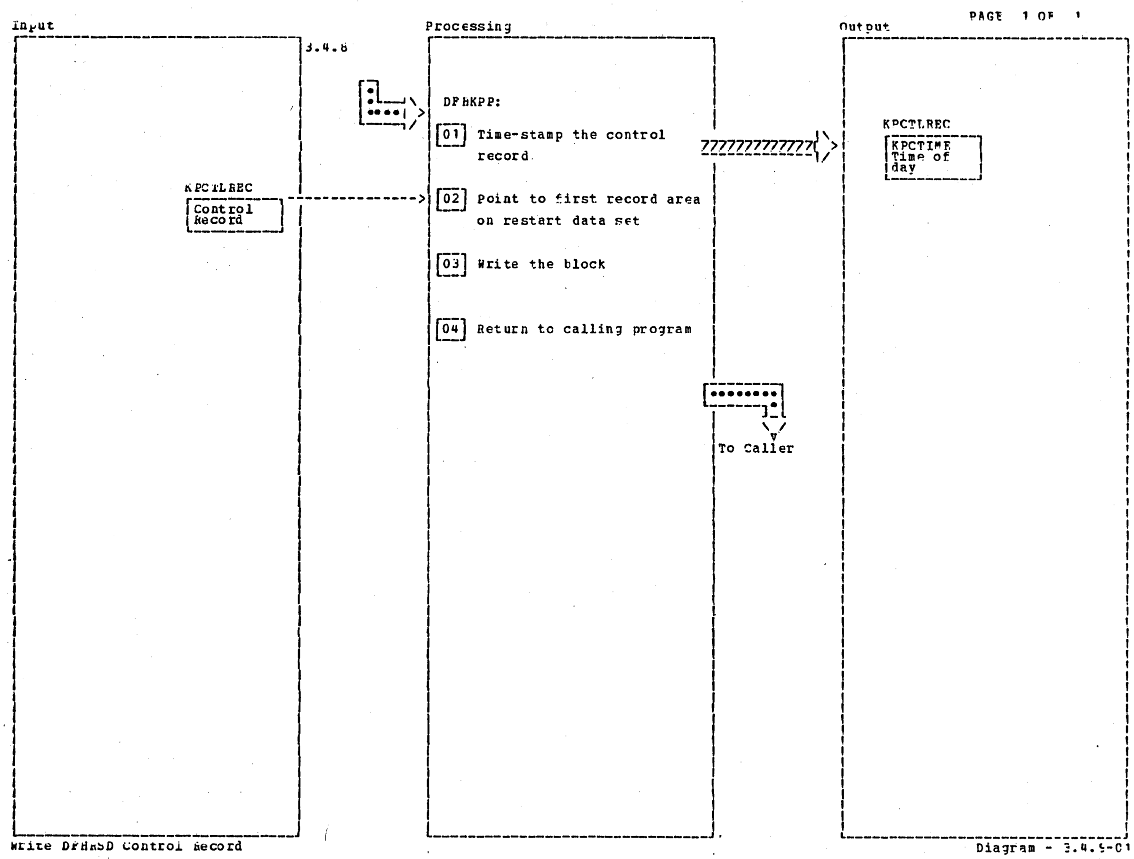
Diagram - 3.4.8-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Save TS Bit Map Data

Diagram - 3.4.8-01



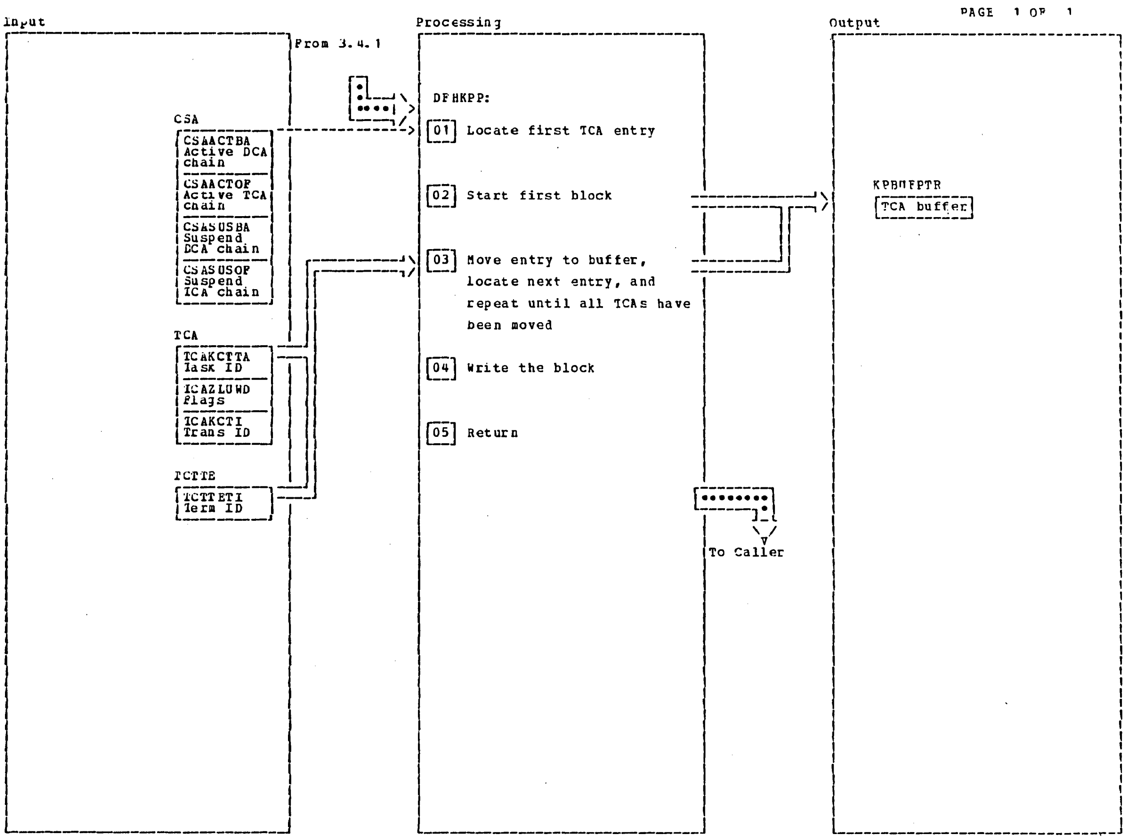
Write DFHASD Control Record

Diagram - 3.4.9-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Time taken by issuing operating system macro. | | | | | | | |
| 03 Subroutine | | KPWRITE | | | | | |
| 04 | | KPPRTN | | | | | |

Write DFHASD Control Record

Diagram - 3.4.9-01


PAGE 1 OF 1
Input
Processing
Output
From 3.4.1
CSA
CSAACTBA Active DCA chain
CSAACTOF Active TCA chain
CSASUSBA Suspend DCA chain
CSASUSOF Suspend TCA chain
TCA
TCAKCTTA Task ID
TCAZLUWD Flags
TCAKCTI Trans ID
TCTTE
TCTTETI Term ID
DFHKPP:
01 Locate first TCA entry
02 Start first block
03 Move entry to buffer, locate next entry, and repeat until all TCAs have been moved
04 Write the block
05 Return
To Caller
KPBUFPTR
TCA buffer

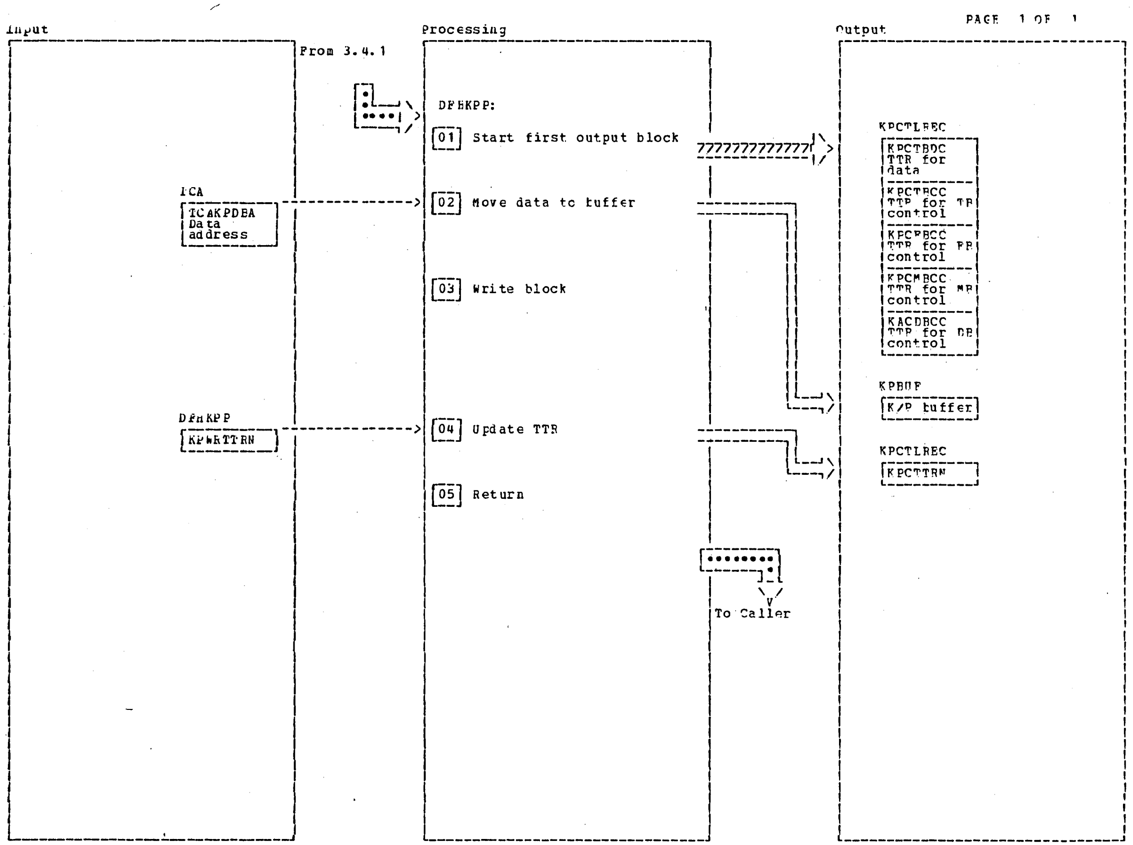
Save Data from TCAs

Diagram - 3.4.10-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 This logic is used to K/P all TCAs. | | KPTCA000 | | | | | |
| 02 Subroutine | | KPBLK1 | | | | | |
| 03 Only TCA's which have written anything to the system log, or are in DWE processing are taken. | | KPTCA015 | | | | | |
| 04 A conditional Journal Control request is issued. If not satisfied, a KC-chap is issued to cause a task switch, so that the Journal buffer will be emptied. The collection of TCA's is then repeated. | | KPJCWTCA | | | | | |
| 05 | | KPPRTN | | | | | |

Save Data from TCAs

Diagram - 3.4.10-01



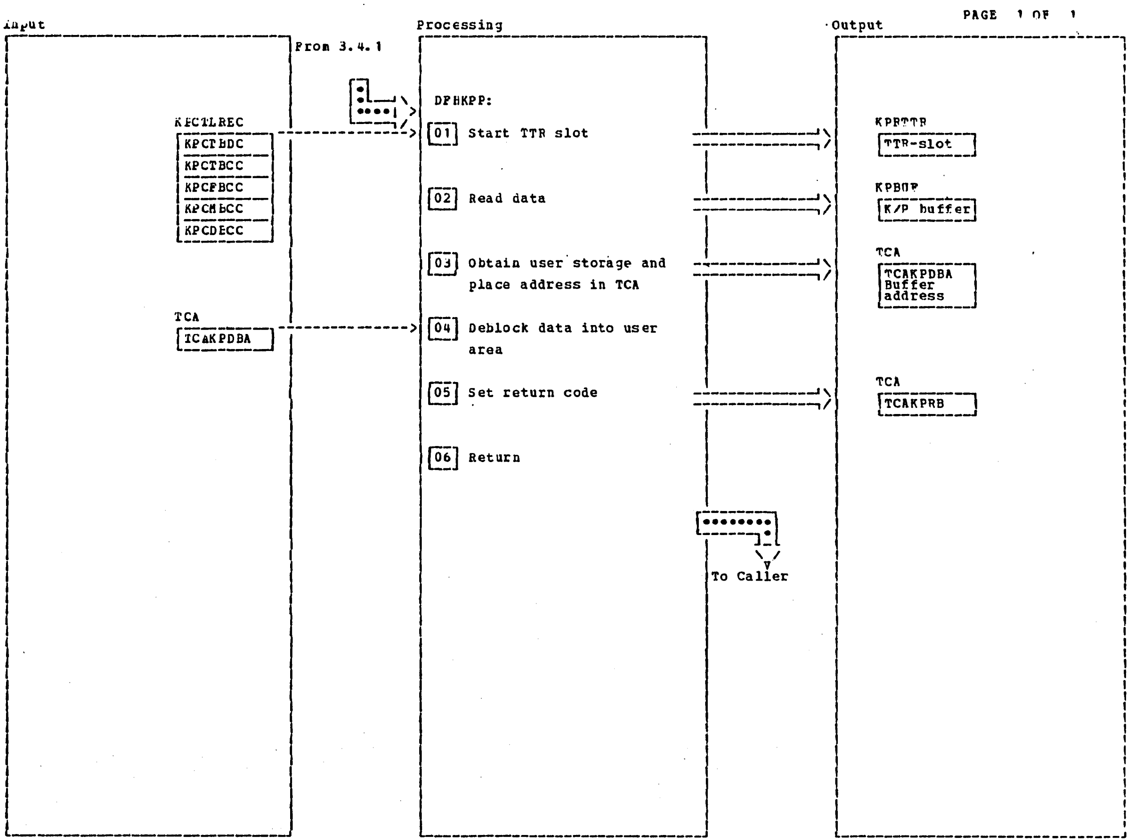
DFHKPP Write Restart Data SetData

Diagram - 3.4.11-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 This logic is used to write data records, the Transaction Backout Table, the File Backout Table, the DL/I Backout and the Message Backout Tables to the Restart data set. Depending on which table is written, the corresponding TTR-slot in the control record is primed. | | KPBLK1 | | | | | |
| 02 If user-record-size exceeds buffer space, the record will be written out spanned into more physical blocks. | | KPWIBD10 | | | | | |
| 03 Subroutine | | KPWRITE | | | | | |
| 04 Next available TTR is updated. | | KPNXTA | | | | | |
| 05 | | KPPRTN | | | | | |

DFHKPP Write Restart Data SetData

Diagram - 3.4.11-C1



DFHKPP Read Restart Data Set Data

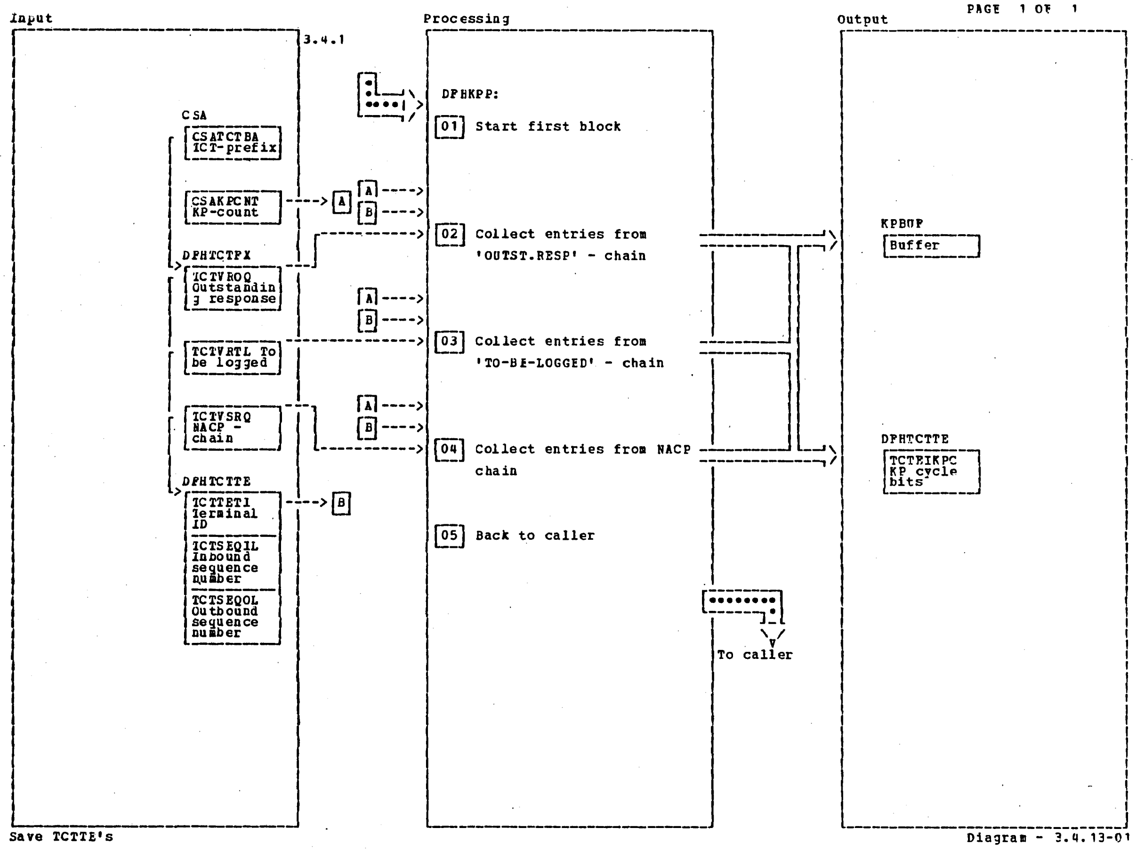
Diagram - 3.4.12-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 This logic is the same for all Read-requests. The starting TTR-address is picked up depending on the request. For read of | | | |
| TBODATA | | KPRTBD | |
| TBOCTL | | KPRTBS | |
| FBOCTL | | KPRFBC | |
| MBOCTL | | KPRMBC | |
| DBOCTL | | | |
| 02 Subroutine | | KPRTBDIO | |
| 03 Issue Storage Control macro - DFHSC TYPE=GETMAIN, CLASS=USER | | KPRTBD10 | |
| 04 If record is spanned over more than one physical blocks, several I/O operations may be required to give the whole record to the user area. | | KPRTBD12 | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 05 X'00' - good<br>X'01' - I/O error<br>X'02' - EOF | | | |
| 06 | | KPPRTN | |

DFHKPP Read Restart Data Set Data

Diagram - 3.4.12-01



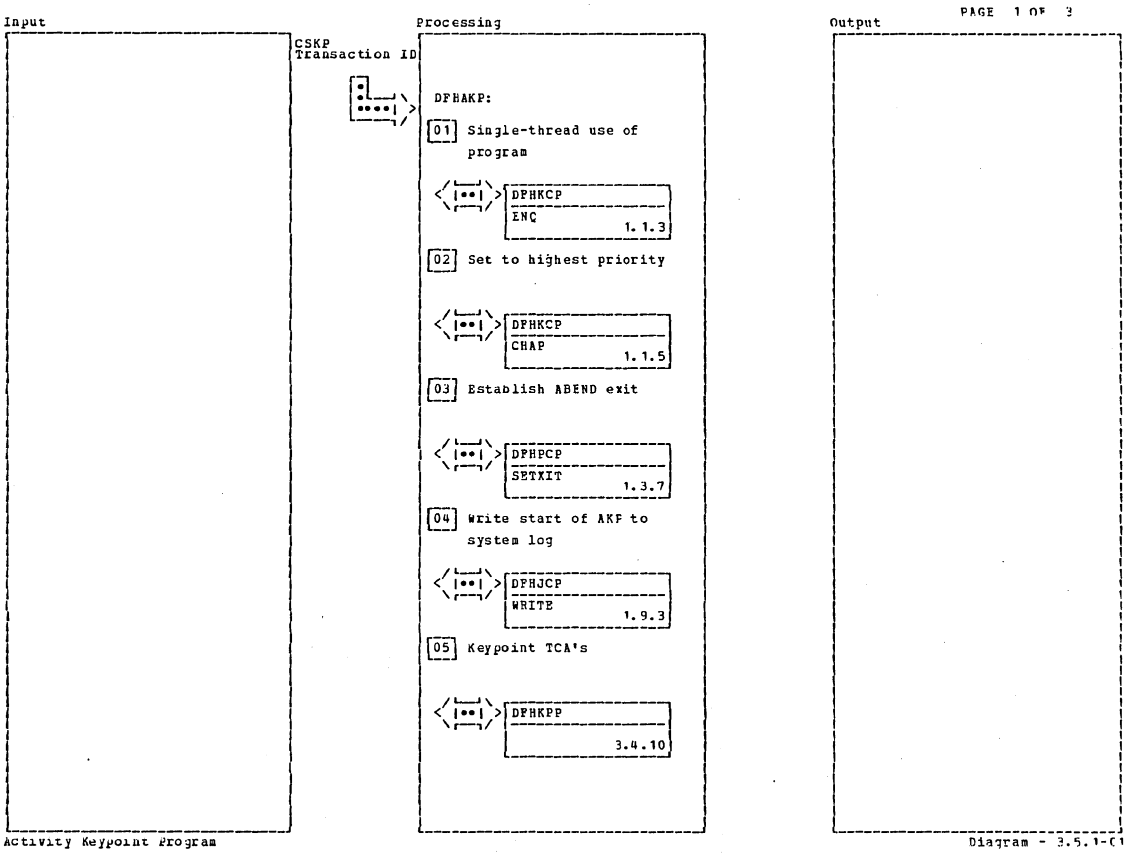
Save TCTTE's

Diagram - 3.4.13-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Subroutine | | KPBLK1 | | | | | |
| 02 When collecting an entry, the Keypoint cycle bit is set to reflect if the entry is taken during an odd or even keypoint. If I/O is required to collect all entries, the scan of the chain is restarted from the beginning, and an entry already taken is then bypassed.<br>Entries are collected at label KPTCTCOL for steps 2 and 3, and at label KPTCNCOL for step 4. | | | | | | | |
| 03 See note 2. | | | | | | | |
| 04 See note 2. | | | | | | | |

Save TCTTE's

Diagram - 3.4.13-01



Activity Keypoint Program — Diagram - 3.5.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 Abend exit labeled AKPABEND is active for entire duration of task. | | | 3.5.2 | | | | |

Activity Keypoint Program — Diagram - 3.5.1-01

PAGE 2 OF 3

Input

Processing

Output

06 Keypoint DCTs

DFHKPP
DCT 3.4.4

07 Keypoint TCTTEs

DFHKPP
TCTTE 3.4.13

08 Conditional link to user's Activity Keypoint program (DFHUAKP)

DFHPCP
LINK 1.3.1

09 Write end of AKP to system log

DFHJCP
WRITE 1.9.3

Activity Keypoint Program

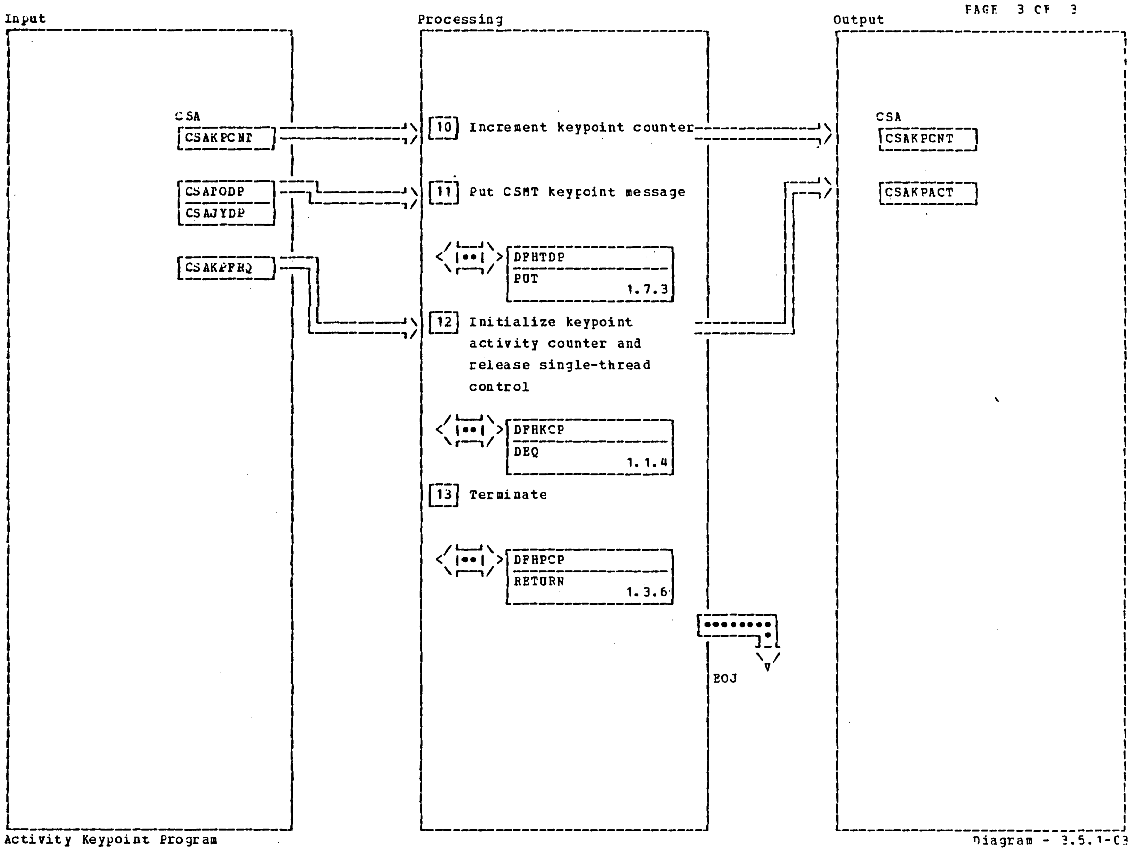
Diagram - 3.5.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

Activity Keypoint Program

Diagram - 3.5.1-02

PAGE 3 OF 3



Activity Keypoint Program

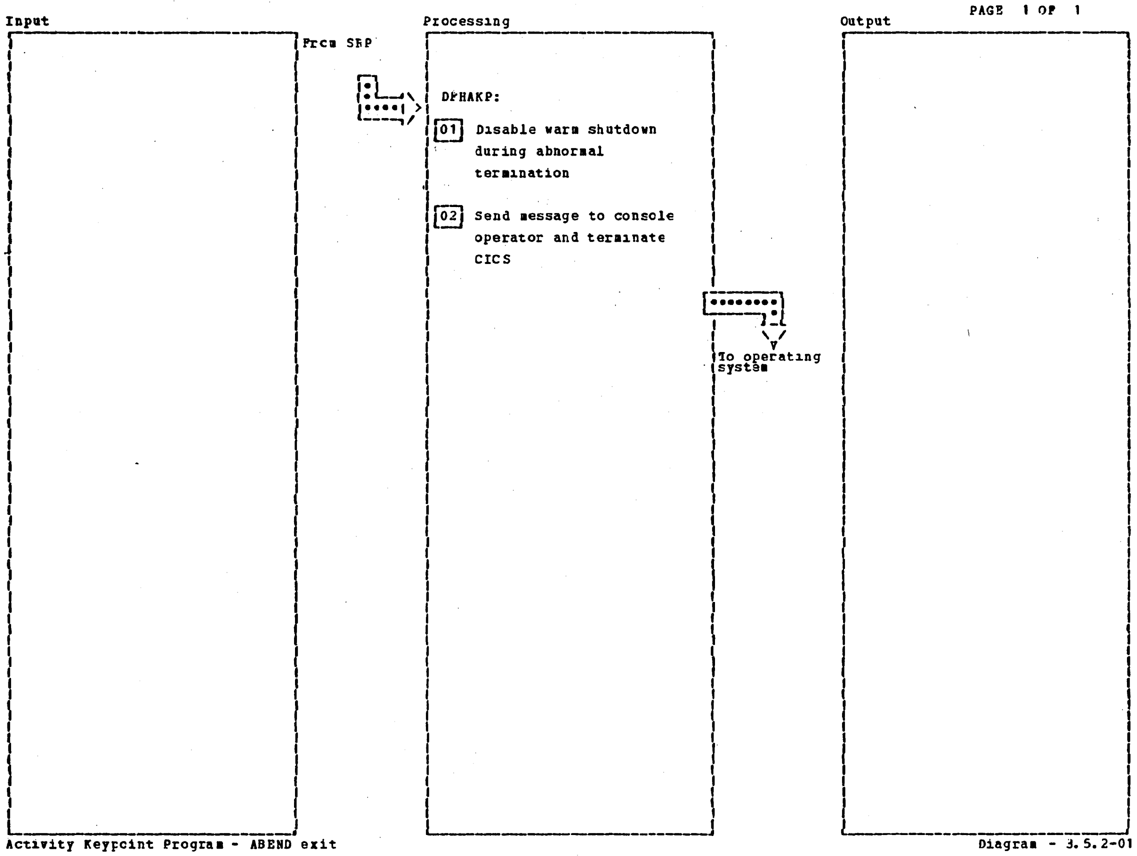
Diagram - 3.5.1-C3

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 11 Bypassed if it is the initial AKP taken during system initialization.<br>If TD-PUT fails the AKP task is abnormally terminated (AAKP) after releasing the ABEND exit. | | | AKPCOMP | | | | |
| 12 | | AKPCOMP | | | | | |

Activity Keypoint Program

Diagram - 3.5.1-C3

PAGE 1 OF 1



Input

Processing

Output

From SRP

DFHAKP:

01 Disable warm shutdown during abnormal termination

02 Send message to console operator and terminate CICS

To operating system

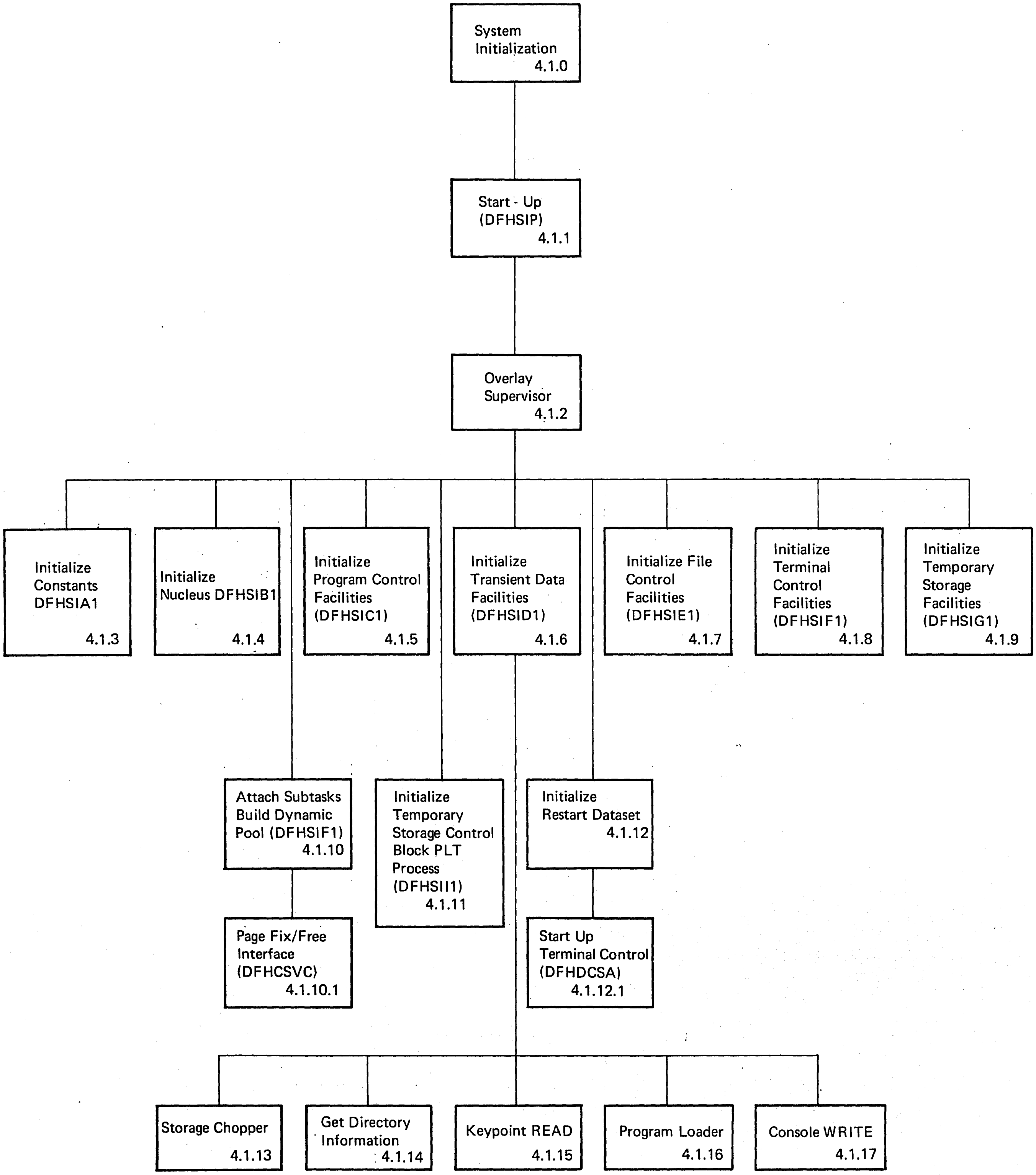
Activity Keypoint Program - ABEND exit

Diagram - 3.5.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | AKPABEND | | | | | |
| 02 Message DFH5802. | | | | | | | |

Activity Keypoint Program - ABEND exit

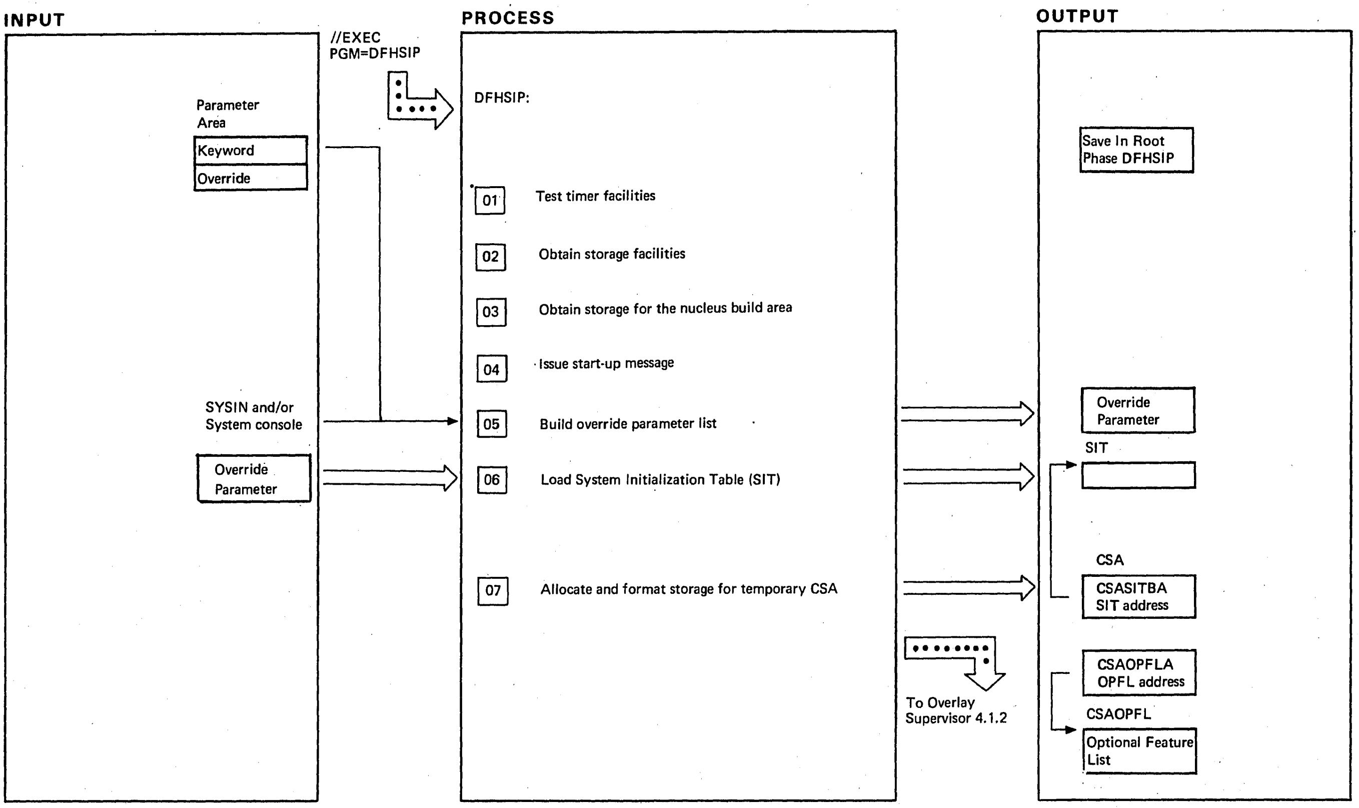
Diagram - 3.5.2-01


System Initialization 4.1.0
Start - Up (DFHSIP) 4.1.1
Overlay Supervisor 4.1.2
Initialize Constants DFHSIA1 4.1.3
Initialize Nucleus DFHSIB1 4.1.4
Initialize Program Control Facilities (DFHSIC1) 4.1.5
Initialize Transient Data Facilities (DFHSID1) 4.1.6
Initialize File Control Facilities (DFHSIE1) 4.1.7
Initialize Terminal Control Facilities (DFHSIF1) 4.1.8
Initialize Temporary Storage Facilities (DFHSIG1) 4.1.9
Attach Subtasks Build Dynamic Pool (DFHSIF1) 4.1.10
Page Fix/Free Interface (DFHCSVC) 4.1.10.1
Initialize Temporary Storage Control Block PLT Process (DFHSII1) 4.1.11
Initialize Restart Dataset 4.1.12
Start Up Terminal Control (DFHDCSA) 4.1.12.1
Storage Chopper 4.1.13
Get Directory Information 4.1.14
Keypoint READ 4.1.15
Program Loader 4.1.16
Console WRITE 4.1.17

System Initialization

Diagram - 4.1.0-01

Page 1 of 1



Start-Up (DFHSIP)

Diagram — 4.1.1-01

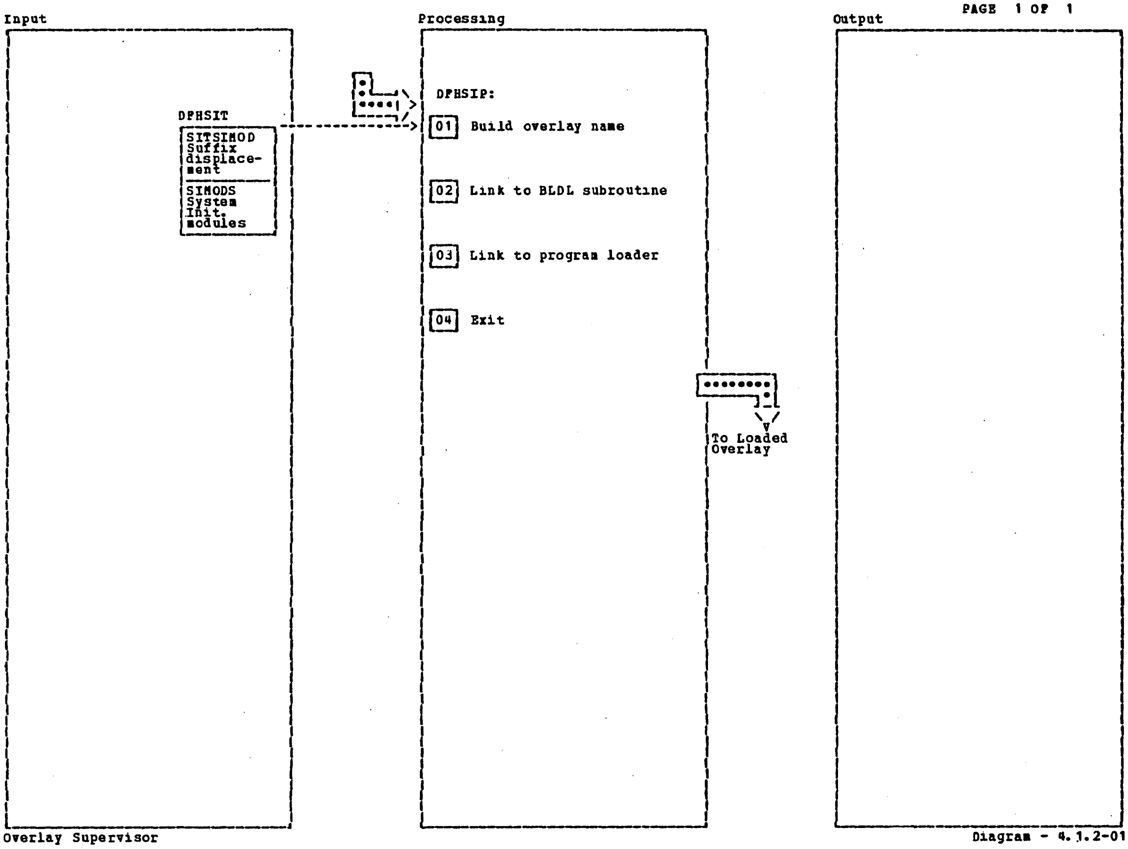
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 A STIMER macro is issued for ½ second. If interval expiration is not signalled within 5 seconds, message DFH1570A is issued. If TOD clock is not set (cc≠0 after STCK instruction) message DFH1569C is issued followed by ABEND 1569. | | | |
| 03 SIP obtains between 34K and 16 megabytes or storage via a variable condition GETMAIN. It then releases 10K via a FREEMAIN to allow for VS working space. | | SIGTMAIN | |
| 04 Message DFH1500 is issued. | | SIGOTCOR | 4.1.17 |
| 05 Read override parameter from the parameter area and/or from SYSIN and/or from the system console | OPENPARM<br>SIPCONSL | | |
| 06 The storage needed to load the table is obtained. | STORAGE CHOPPER | SITCONS | 4.1.13 |
| The information required is obtained | BLDL | | 4.1.14 |
| System Initialization Table is loaded MSG DFH1501 is issued. | PGM LOADER | | 4.1.16 |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 07 Storage is allocated for a temporary CSA at the high-end of the partition (just below the SIT). The storage is cleared and various fields are initialized. | STORAGE CHOPPER | | 4.1.13 |

Start-Up (DFHSIP)

Diagram — 4.1.1-01

PAGE 1 OF 1


Input
Processing
Output
DFHSIT
SITSIMOD Suffix displace-ment
SIMODS System Init. modules
DFHSIP:
01 Build overlay name
02 Link to BLDL subroutine
03 Link to program loader
04 Exit
To Loaded Overlay

Overlay Supervisor

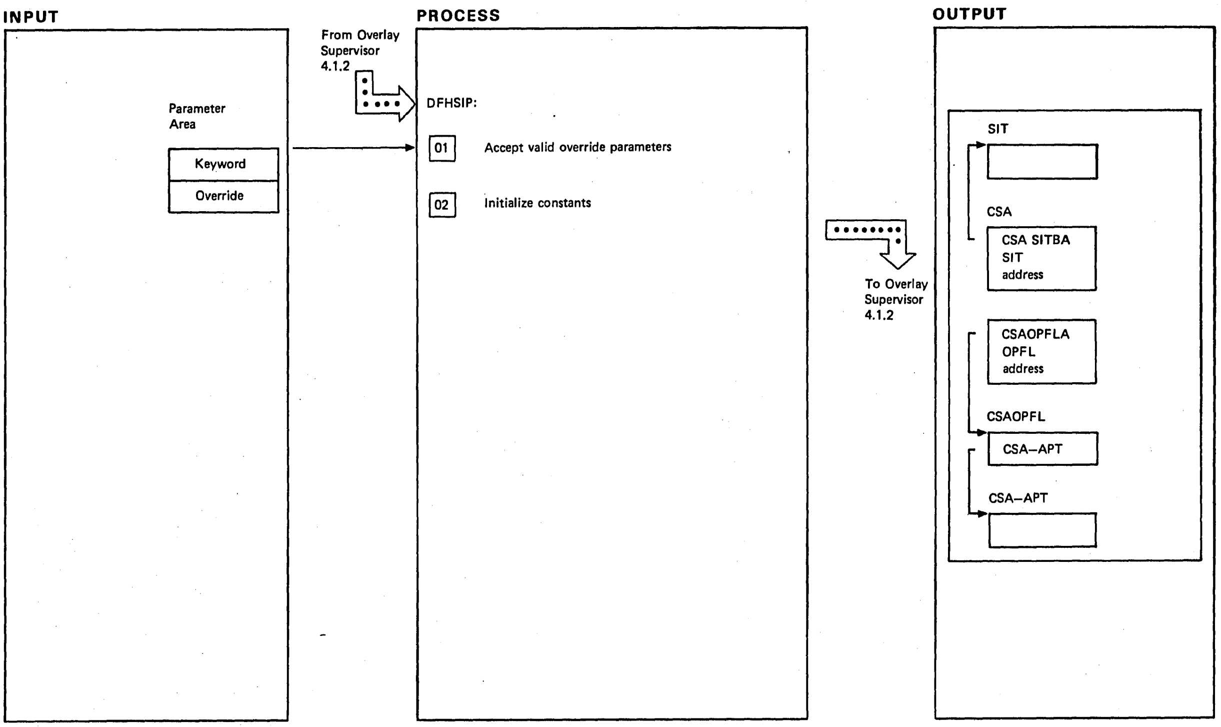
Diagram - 4.1.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The overlay name is formed by appending a two-character suffix to the basic name DFHSI. The suffix is located in a variable-length table and its displacement is given by a halfword index (SITSIMOD).<br>The index is increased by two upon exit to point to the next initialization routine.<br>If DL/I is included, the overlay supervis¢r calls overlay (DFHSIDL) - the OS/DLI overlay after DFHSIDI. | | OVERLSUP | |
| 02 Locate overlay in library. Issue LOAD TXT=NO macro instruction to OS/VS. | | | 4.1.14 |
| 03 Register 0 is used as the overlay area pointer. Issue LOAD macro instruction to OS/VS. | | | 4.1.16 |
| 04 The overlay base registers are loaded and the overlay supervisor transfers control to the next System Initialization routine by branching to its entry point. | | | |

Overlay Supervisor

Diagram - 4.1.2-01

Page 1 of 1



Initialize Constant (DFHSIA)

Diagram – 4.1.3–01

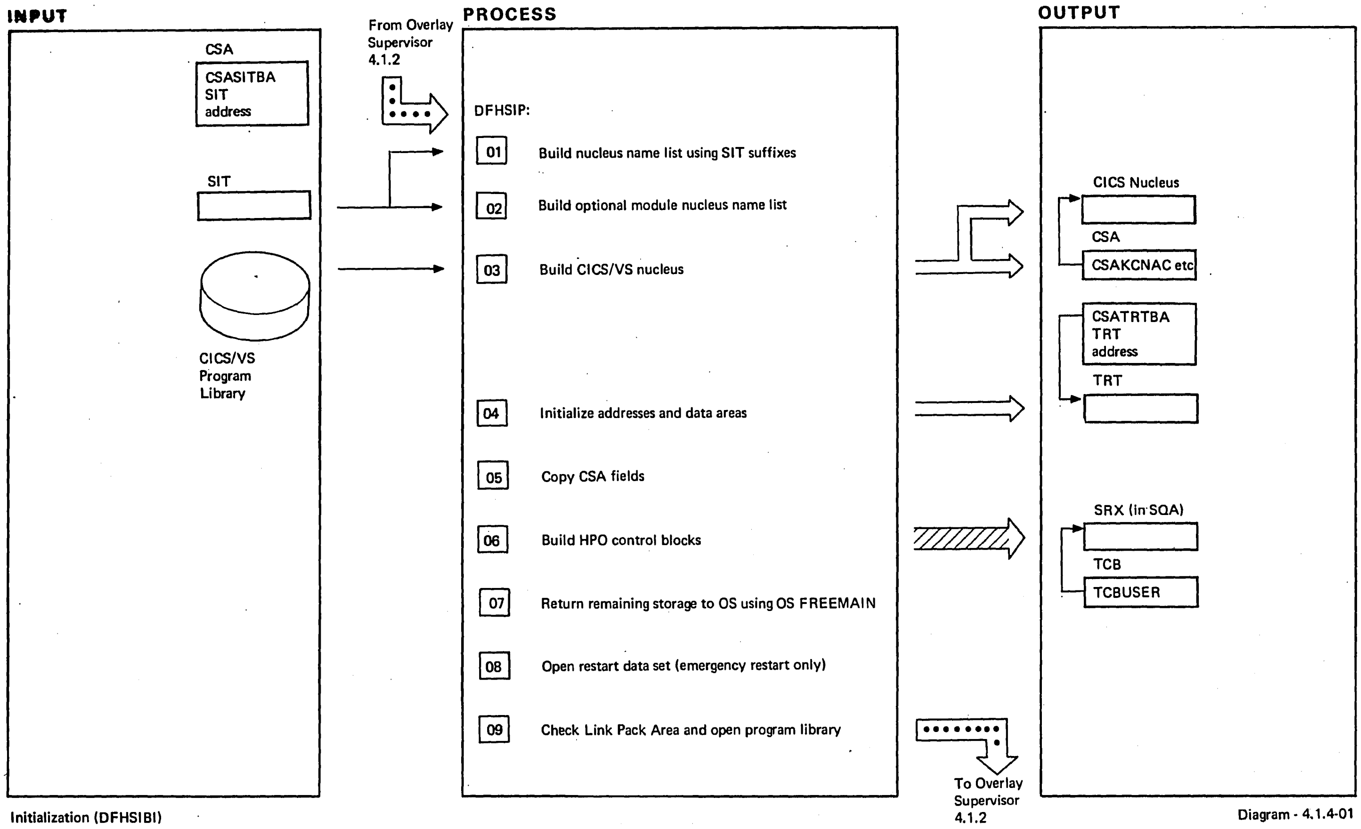
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Either message DFH1502 or DFH1503 is issued if the keyword or keyword data are invalid | Console WRITE | INVDTWRT<br>INVKEY | |
| 02 DL/I, CSA, and CSA – ATP constants are initialized | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Initialize Constant (DFHSIA)

Diagram – 4.1.3–01

Page 1 of 1


INPUT
CSA
CSASITBA
SIT
address
SIT
CICS/VS
Program
Library
From Overlay
Supervisor
4.1.2
PROCESS
DFHSIP:
01 Build nucleus name list using SIT suffixes
02 Build optional module nucleus name list
03 Build CICS/VS nucleus
04 Initialize addresses and data areas
05 Copy CSA fields
06 Build HPO control blocks
07 Return remaining storage to OS using OS FREEMAIN
08 Open restart data set (emergency restart only)
09 Check Link Pack Area and open program library
To Overlay
Supervisor
4.1.2
OUTPUT
CICS Nucleus
CSA
CSAKCNAC etc
CSATRTBA
TRT
address
TRT
SRX (in SQA)
TCB
TCBUSER

Initialization (DFHSIBI)

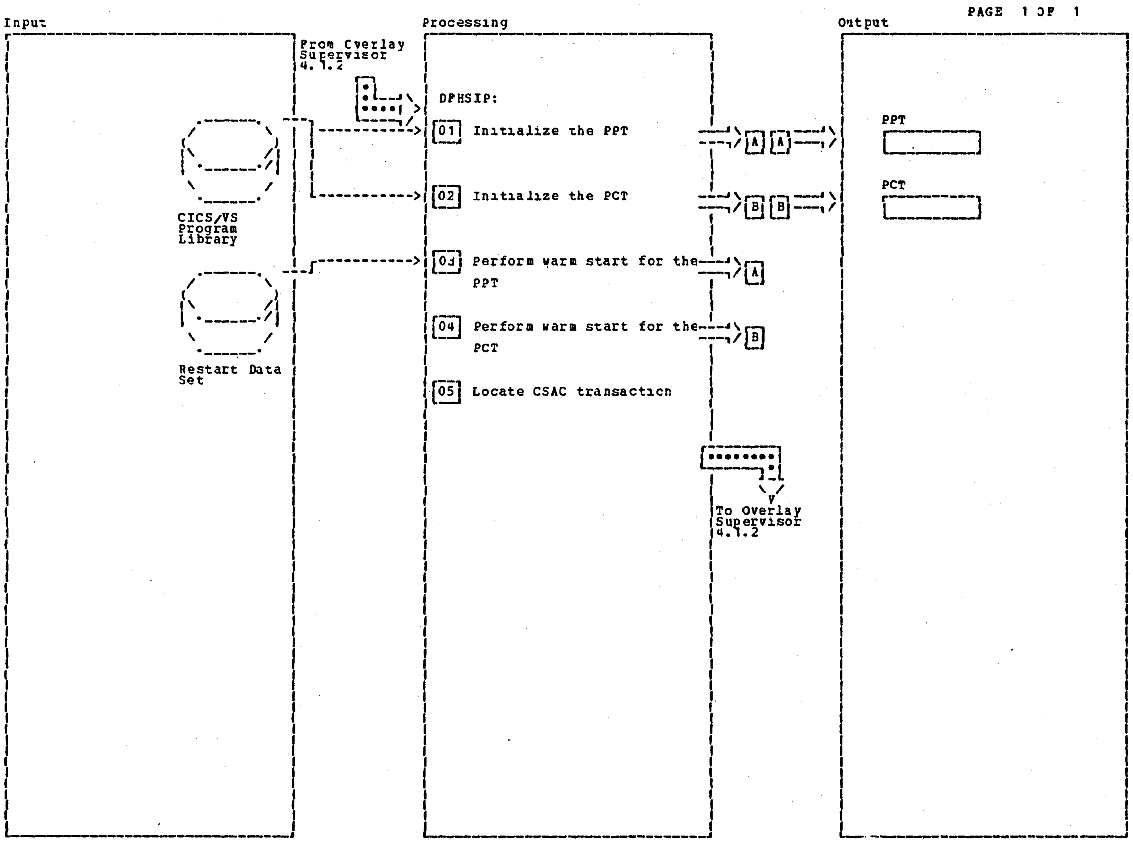
Diagram - 4.1.4-01

| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 01 | Message DFH15000 is issued | Console WRITE | DFHSIBNA | |
| 02 | The Nucleus Name List is an internal table that controls the names of modules loaded into the CICS/VS nucleus. The list is constructed by appending suffixes to modules where needed and by deleting modules not needed by the system. | | | |
| 03 | If required, a nucleus load table is loaded to control the load sequence of nucleus modules. The Internal default table is used to drive the loader to load the CICS nucleus, and to initialize addresses in CSA and CSA external. For HPO, certain modules are loaded into protected memory. | SIPBLDL<br>NUCBD<br>SIPCORE<br>SIPLDER | SINB00 | |
| 04 | Storage is obtained for other system data areas, and certain additional 'link edit' functions performed to interconnect nuclues modules and tables. The latter function applies to JCP, BMS modules, SCP, FCP, TCP, DLI and TRP.<br>• If 7770 suport is included DFHRWP70 is loaded.<br>• If keypoints are required a keypoint buffer is allocated and its address placed in KPP, or, if there is no system log, CICS abends with message DFH1563<br>• The trace table is initialized and its address placed in CSATRTBA | | | |

| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 05 | Copy addresses from the temporary CSA allocated by DFHSIP into the real CSA loaded by DFHSIB1 | | SIPRCSA | |
| 08 | If SITIND is not blank the restart data set is opened | SIPRSDRD | | |
| 09 | If PL/I shared library specified check that IBMBPSLA/IBMBPSMA are in link pack area. | | | |

Initialization (DFHSIBI)

Diagram - 4.1.4-01


PAGE 1 OF 1
Input
Processing
Output
From Overlay Supervisor 4.1.2
DFHSIP:
01 Initialize the PPT
02 Initialize the PCT
03 Perform warm start for the PPT
04 Perform warm start for the PCT
05 Locate CSAC transaction
CICS/VS Program Library
Restart Data Set
A
B
PPT
PCT
To Overlay Supervisor 4.1.2

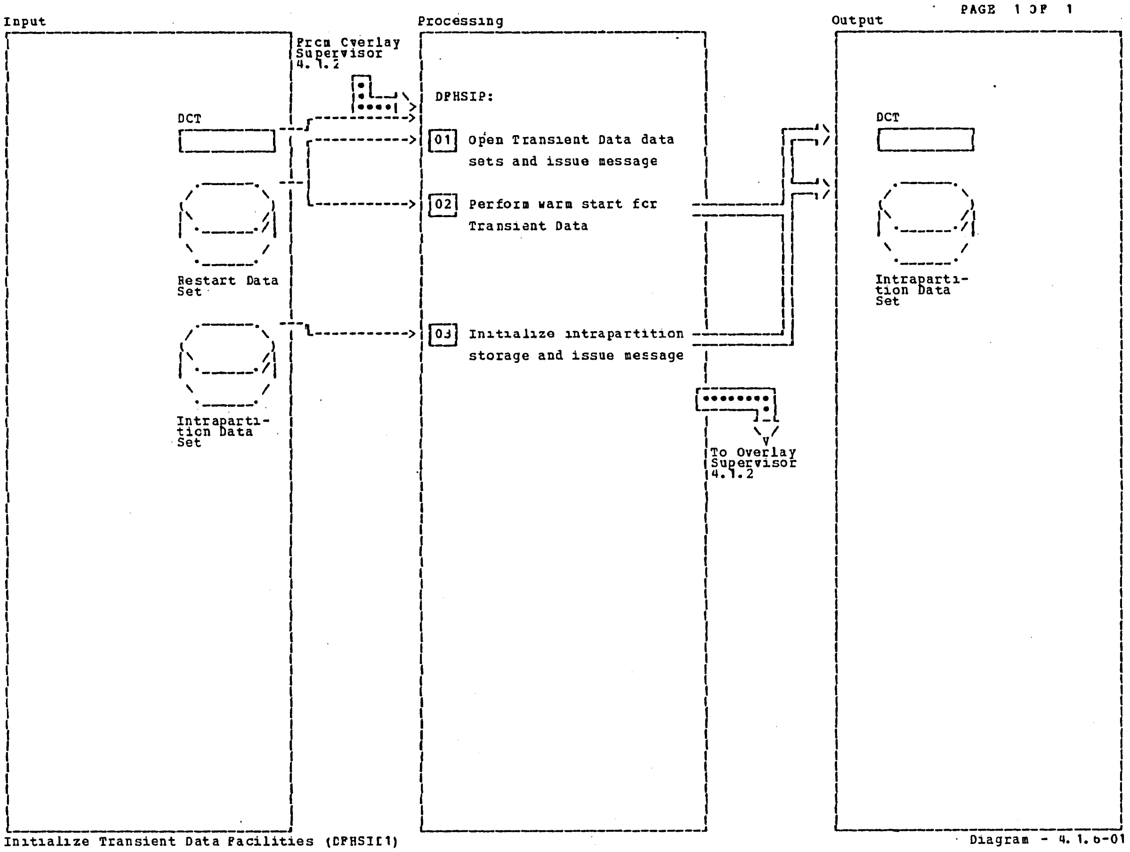
Initialize Program Control Facilities (DFHSIC1)

Diagram - 4.1.5-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The directory information is read from the BPL.<br>Error messages that may be issued are DFH1596a, DFH1594, and/or DFH1505. | BLDL | SIPPTLD | 4.1.14 |
| 02 The disable flags for invalid transactions are set. | | SIPLDPCT | |
| 03 The keypointed PPT entry flag is checked for disabled status as it is read from the restart data set. If the entry was keypointed in a disabled condition, the PPT is scanned and the matching entry is flagged as being disabled. If no warm start information was supplied, the PPT is cold started, and messages DFH1507 and DFH1511 are issued. | Keypoint READ | PPTWARM | 4.1.15 |
| 04 The keypointed PCT entry flag is checked for disabled status as it is read from the restart data set. If the entry was keypointed in a disabled condition, the PCT is scanned and the matching entry is flagged as being disabled. If no warm start information was supplied, the PCT is cold started and message DFH1508 is issued. | Keypoint READ | SIPPCTW | 4.1.15 |
| 05 If the CSAC transaction cannot be found, message DFH1597 is issued. Also, the disable flag ON and the PPT pointer set to zeros are checked in the PCT and message DFH1510 is issued. | Console WRITE | CSACSCN | |

Initialize Program Control Facilities (DFHSIC1)

Diagram - 4.1.5-01


PAGE 1 OF 1
Input
Processing
Output
From Overlay Supervisor 4.1.2
DFHSIP:
DCT
Restart Data Set
Intrapartition Data Set
01 Open Transient Data data sets and issue message
02 Perform warm start for Transient Data
03 Initialize intrapartition storage and issue message
To Overlay Supervisor 4.1.2
DCT
Intrapartition Data Set

Initialize Transient Data Facilities (DFHSIC1)

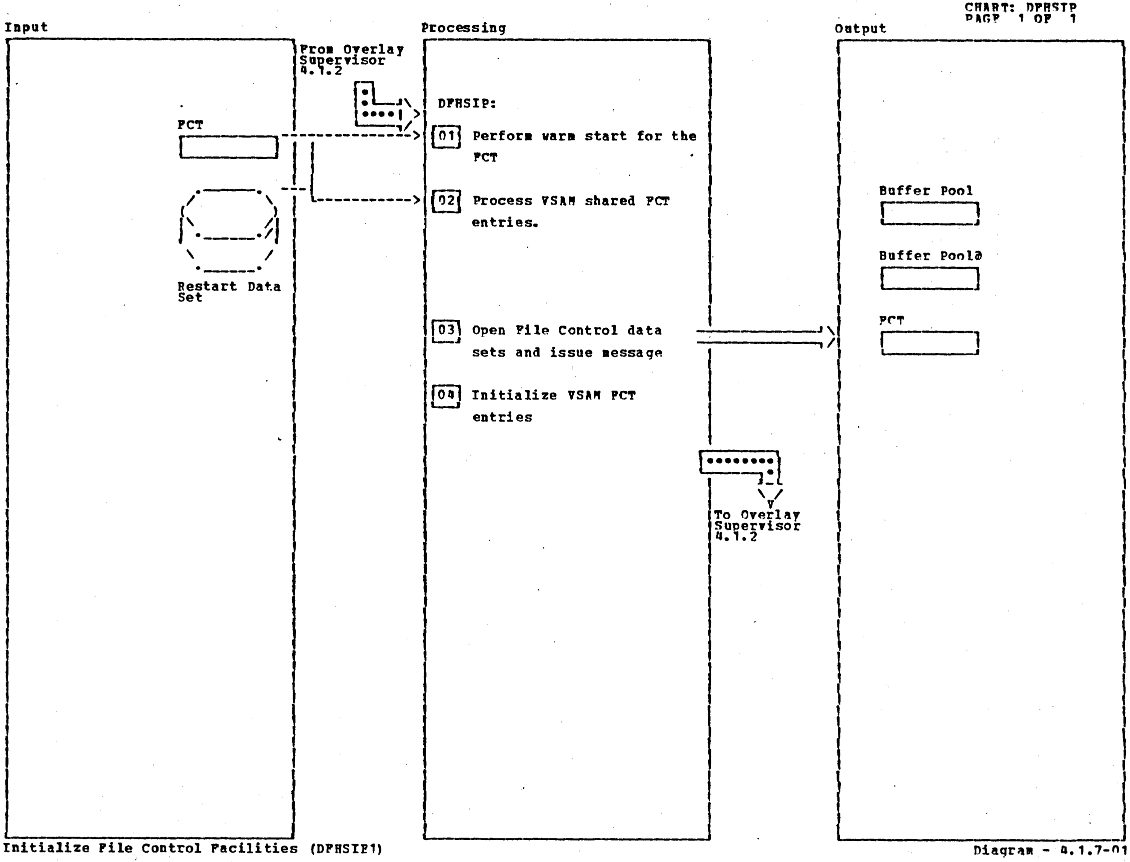
Diagram - 4.1.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Message DFH1500 is issued. | Console WRITE | SIPTD01B | | | | | |
| 02 The keypcinted DCT entry is read from the restart data set. The DCT table is scanned and the keypointed information is stored in the matching entry. If no warm start information was supplied, the DCT is cold started and messages DFH1560 and DFH1561 are issued. | Keypoint READ<br>Console WRITE | DCT1561 | | | | | |
| 03 Message DFH1500 is issued.<br>The control record (track 0 record 1) of intrapartition data set is read and no initialization is done if the control record can be read. (Track 0 is reserved for the control record.) | Console WRITE | SIPCOLD | | | | | |

Initialize Transient Data Facilities (DFHSIC1)

Diagram - 4.1.6-01

CHART: DFHSIP
PAGE 1 OF 1



01 Perform warm start for the PCT

02 Process VSAM shared PCT entries.

03 Open File Control data sets and issue message

04 Initialize VSAM PCT entries

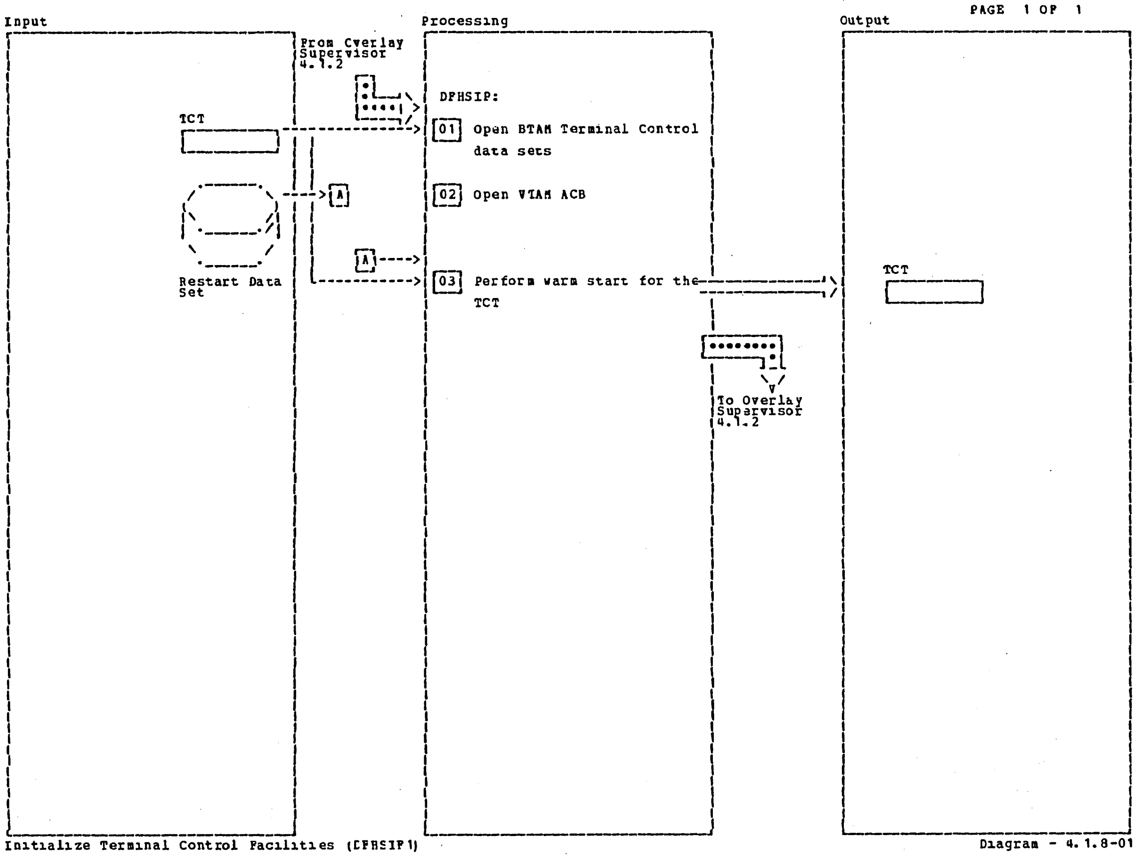
Initialize File Control Facilities (DFHSIE1) — Diagram - 4.1.7-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The keypointed PCT entry is read from the restart data set. The PCT table is then scanned and the keypointed control information is stored in the matching entry. If no warm start information is supplied, the PCT is cold started, and messages DFH1573 and DFH1574 are issued. | Keypoint READ and Console WRITE | FCTGETP AND FCT1574 | 4.1.14 |
| 02 Scan the PCT for VSAM shared entries. For each entry that is found, calculate the buffer number, maximum key length, and a running total of the string value specified. | | | |
| 03 Message DFH1500 is issued. | | FCTOPEN | 4.1.17 |
| 04 SHOWCB macro instruction is issued for key length, logical record length, and relative key position. This information is stored in the PCT. | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Initialize File Control Facilities (DFHSIE1) — Diagram - 4.1.7-01

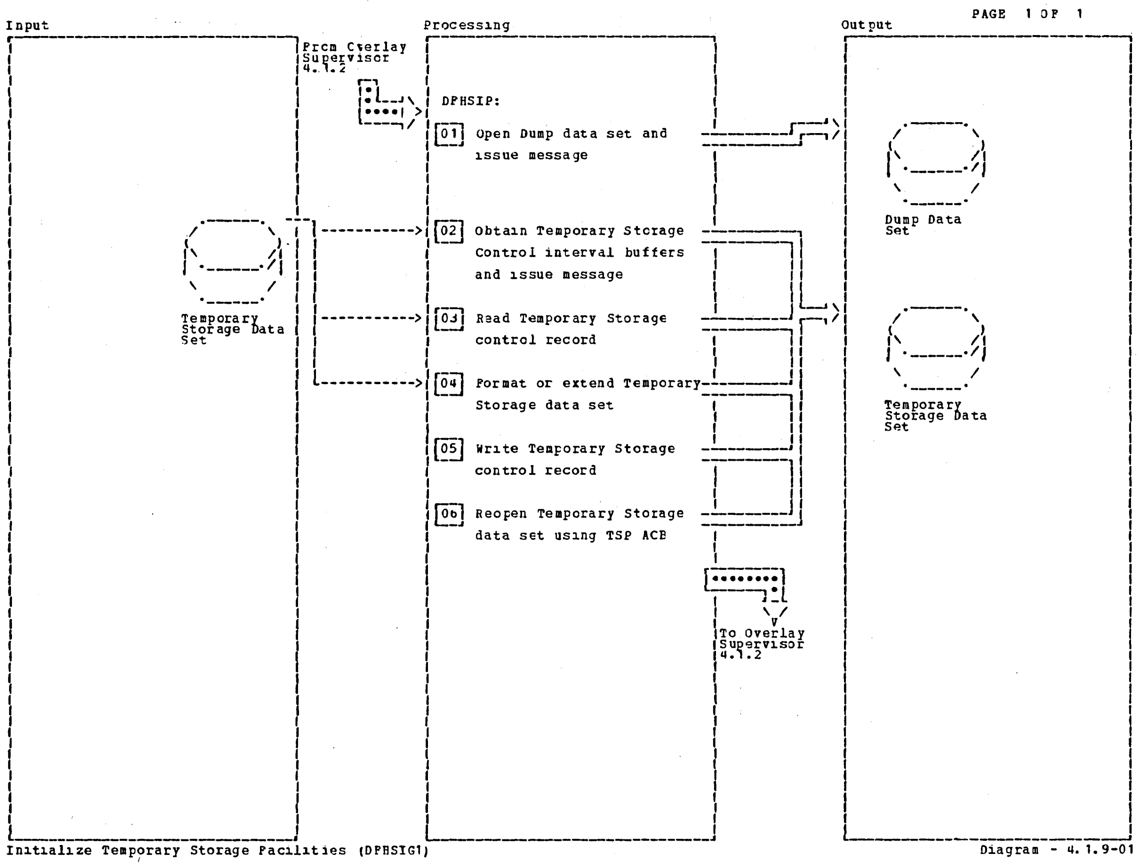
PAGE 1 OF 1


Input
Processing
Output
From Overlay Supervisor 4.1.2
DFHSIP:
TCT
01 Open BTAM Terminal Control data sets
02 Open VTAM ACB
A
A
03 Perform warm start for the TCT
Restart Data Set
To Overlay Supervisor 4.1.2
TCT

Initialize Terminal Control Facilities (DFHSIP1) Diagram - 4.1.8-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Message DFH1500 is issued.<br>If DFHTCP was not loaded, bypass loading of BTAM terminal control data sets. | Console WRITE | DFHSIPIB | | | | | |
| 02 Open VTAM ACB if generated. If open is not successful, issue message DFH1572; follow with message DFH1505. Issue message DFH1571 if both VTAM and BTAM are not available. | | | | | | | |
| 03 The keypointed TCT entry is read from the restart data set. The TCT is scanned and the keypointed status information is stored in the matching entry. If no warm start information was supplied, the TCT is cold started, and messages DFH1577 and DFH1575 are issued. | Keypoint READ and Console WRITE | TCTWARM | | | | | |

Initialize Terminal Control Facilities (DFHSIP1) Diagram - 4.1.8-01


PAGE 1 OF 1
Input
Processing
Output
From Overlay Supervisor 4.1.2
DFHSIP:
01 Open Dump data set and issue message
02 Obtain Temporary Storage Control interval buffers and issue message
03 Read Temporary Storage control record
04 Format or extend Temporary Storage data set
05 Write Temporary Storage control record
06 Reopen Temporary Storage data set using TSP ACB
To Overlay Supervisor 4.1.2
Temporary Storage Data Set
Dump Data Set
Temporary Storage Data Set

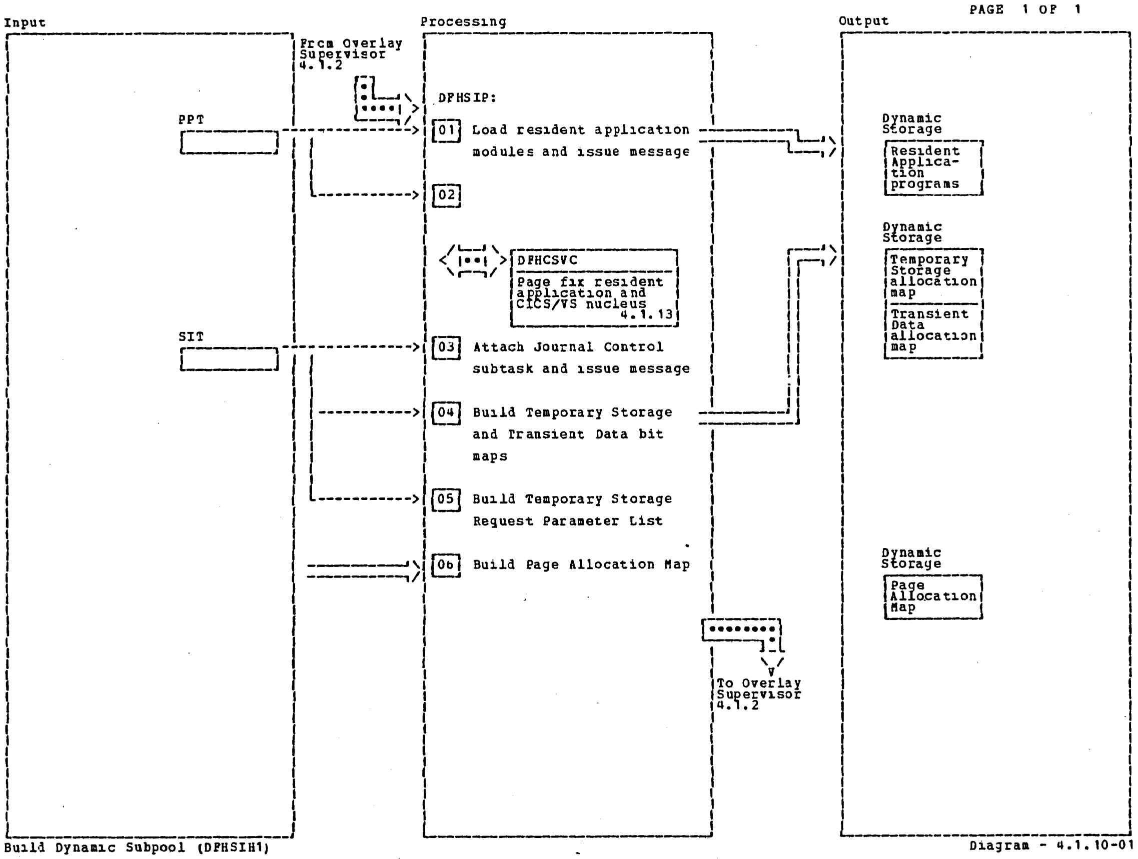
Initialize Temporary Storage Facilities (DFHSIG1)

Diagram - 4.1.9-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Message DFH1500 is issued. | Console WRITE | DFHSIG1B | | | | | |
| 02 Message DFH1500 is issued. | Console WRITE | TSPSTART | | | | | |
| 03 The first control interval of the Temporary Storage data set is reserved for control information. | | TEMPPGEX | | | | | |
| 04 If a format error occurred, message DFH1576 is issued. Format is done by writing dummy records to determine high relative byte address. | Console WRITE | SIPTSAB | | | | | |

Initialize Temporary Storage Facilities (DFHSIG1)

Diagram - 4.1.9-01


PAGE 1 OF 1
Input
Processing
Output
Frcm Overlay Supervisor 4.1.2
PPT
SIT
DFHSIP:
01 Load resident application modules and issue message
02
DFHCSVC
Page fix resident application and CICS/VS nucleus 4.1.13
03 Attach Journal Control subtask and issue message
04 Build Temporary Storage and Transient Data bit maps
05 Build Temporary Storage Request Parameter List
06 Build Page Allocation Map
To Overlay Supervisor 4.1.2
Dynamic Storage
Resident Applica-tion programs
Dynamic Storage
Temporary Storage allocation map
Transient Data allocation map
Dynamic Storage
Page Allocation Map

Build Dynamic Subpool (DFHSIH1)

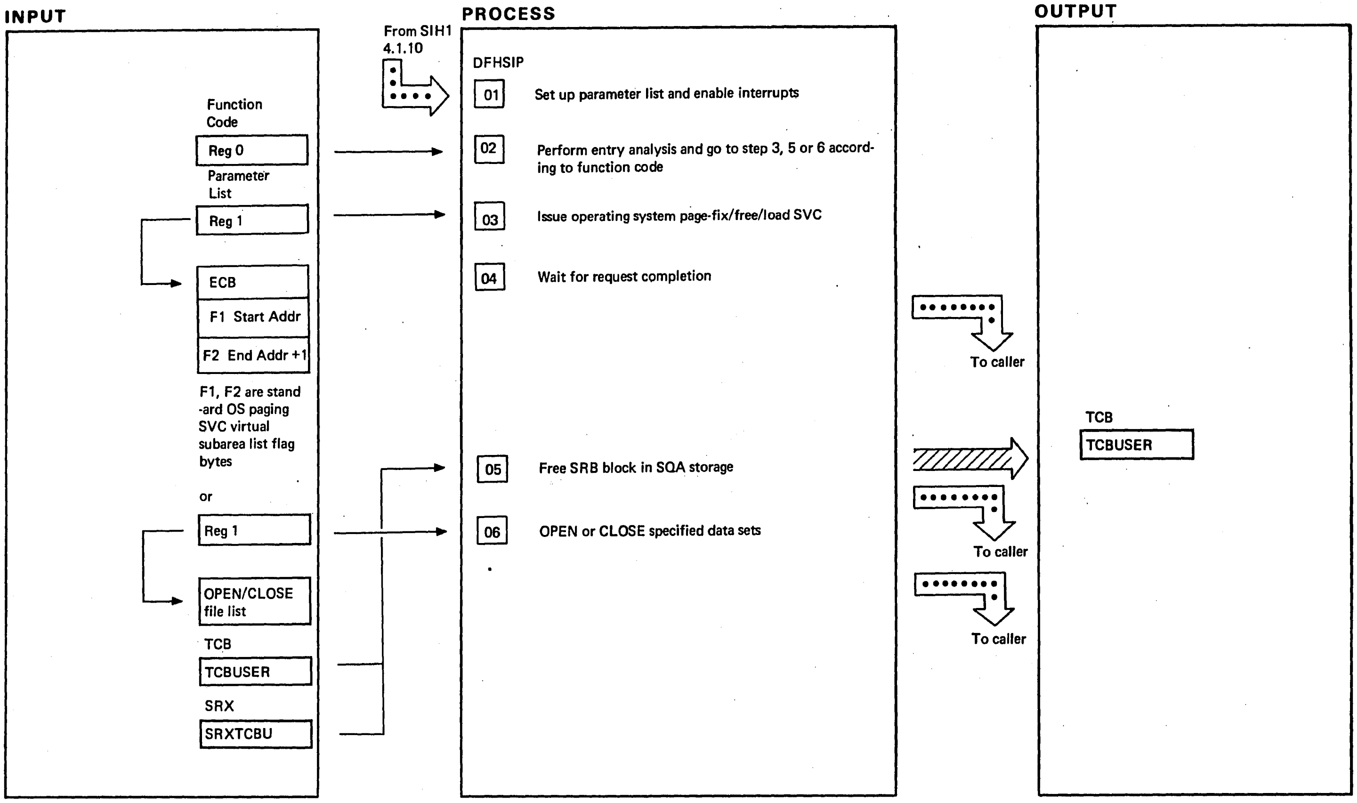
Diagram - 4.1.10-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 Resident applicaticns are fixed. The CICS/VS nucleus may also be fixed at this time.<br>If a page fix error occurs, message DFH1582 is issued. | DFHSVC | LDRESFX | | | | | |
| 03 Message DFH1500 is issued. | Conscle WRITE | | | | | | |

Build Dynamic Subpool (DFHSIH1)

Diagram - 4.1.10-01

Page 1 of 1


INPUT
Function Code
Reg 0
Parameter List
Reg 1
ECB
F1 Start Addr
F2 End Addr +1
F1, F2 are stand-ard OS paging SVC virtual subarea list flag bytes
or
Reg 1
OPEN/CLOSE file list
TCB
TCBUSER
SRX
SRXTCBU
From SIH1 4.1.10
PROCESS
DFHSIP
01 Set up parameter list and enable interrupts
02 Perform entry analysis and go to step 3, 5 or 6 according to function code
03 Issue operating system page-fix/free/load SVC
04 Wait for request completion
05 Free SRB block in SQA storage
06 OPEN or CLOSE specified data sets
To caller
To caller
To caller
OUTPUT
TCB
TCBUSER

Page Fix/Free Interface (DFHCSVC )

Diagram - 4.1.10.1 - 01

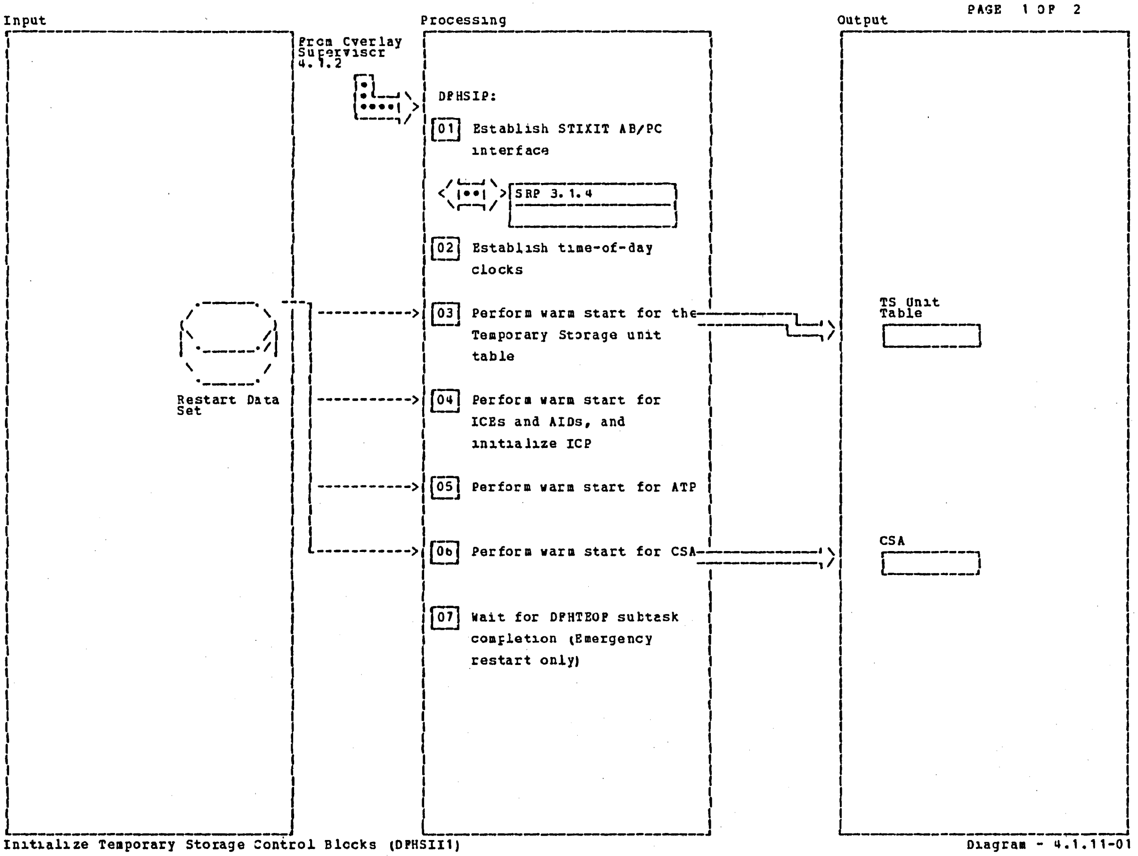
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The system mask is set for VS1. (It is assumed to be set for VS2.) | | CSVC100 | |
| 02 Request codes in reg 0 may be:<br>0 Paging Request<br>4 HPO release SRB block request<br>8 OPEN files<br>12 CLOSE files | | CSVC180 | |
| 03 For MVS register forms of input to the system paging SVC are used. The arguments to this SVC are loaded from the input parameter list (addressed by reg 1) where they were set up by the caller, usually by means of the DFHPAGE macro. For SVS and VS1, where the multiple-page register form of paging SVC is not available, page load assumes that the input parameter list is already page fixed, and issues a virtual subarea list form of paging SVC. For page fix/free the register form is used and the request performed one page at a time | | CSVC200 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 04 For page fix requests a wait is issued, if necessary, until the fix is complete.<br>For page load the caller is responsible for the wait.<br>In either case, on return, register 15 contains the standard operating system paging SVC return code (loaded from the ECB if a page fix load to wait). | | | |
| 05 For MVS/HPO the SRB in SQA storage addressed by the TCBUSER field in the TCB is freed, and the TCBUSER field restored to its orginal value | | CSVC300 | |
| 06 An OPEN/CLOSE interface is provided here for MVS/HPO to permit the requests to be issued in superviser state, which is necessary to allow VSAM to fix control blocks for ICIP | | CSVC400<br>CSVC500 | |

Page Fix/Free Interface (DFHCSVC)

Diagram - 4.1.10.1-01

PAGE 1 OF 2

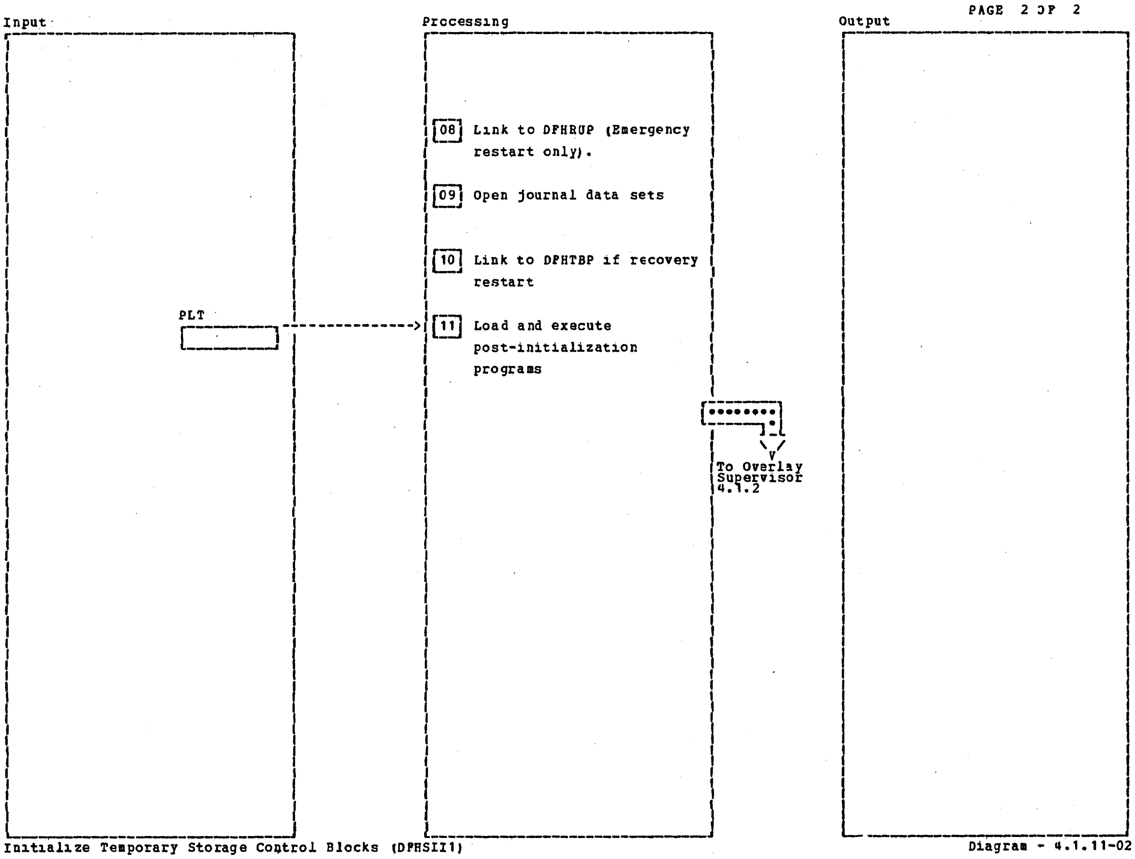

Input
Processing
Output
From Overlay Supervisor 4.1.2
DFHSIP:
01 Establish STIXIT AB/PC interface
SRP 3.1.4
02 Establish time-of-day clocks
Restart Data Set
03 Perform warm start for the Temporary Storage unit table
TS Unit Table
04 Perform warm start for ICEs and AIDs, and initialize ICP
05 Perform warm start for ATP
06 Perform warm start for CSA
CSA
07 Wait for DFHTEOF subtask completion (Emergency restart only)

Initialize Temporary Storage Control Blocks (DFHSII1) Diagram - 4.1.11-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 STIXIT AB/PC macro instruction are issued. Message DFH1500 is also issued. | Console WRITE | DFHSII1B | |
| 02 Issue DFHTC TYPE=RESET to establish the time-of-day clocks. If an error occurs, issue message DFH1543. | | | |
| 03 Read Keypcinted Temporary Storage entry from Restart data set. Setup all associated RBAs with the entry, then issue DFHTS TYPE=RESTORE to have TSP rebuild the Temporary Storage unit table. | | | |
| 04 Read Keypointed AID entries from Restart data set and build AID control blocks. Read Keypointed ICE entries from Restart data set and build ICE control blocks. | | | |
| 05 Read Keypointed ATP control blocks from Restart data set and build BCA and WRE control blocks. | | | |
| 06 Restore the following control fields in the CSA with the Keypcinted CSA data:<br>CSAKCMT - Max. tasks<br>CSASBTI - Region time<br>CSAICRIC - Runaway time int.<br>CSAICSIC - Stall time int.<br>CSASCNB - Storage cushion size<br>CSAONQID - CSA unique ID<br>CSABCMXT - ATP max. tasks<br>CSABCMXB - ATP max. batch | | | |
| 07 If emergency restart and system log is on tape, wait for completion of DFHTEOF. If DFHTEOF subtask fails, issue message DFH1585. | Console WRITE | CSABKIT | |

Initialize Temporary Storage Control Blocks (DFHSII1) Diagram - 4.1.11-01

PAGE 2 OF 2



08 Link to DFHRUP (Emergency restart only).

09 Open journal data sets

10 Link to DFHTBP if recovery restart

11 Load and execute post-initialization programs

Initialize Temporary Storage Control Blocks (DFHSII1)

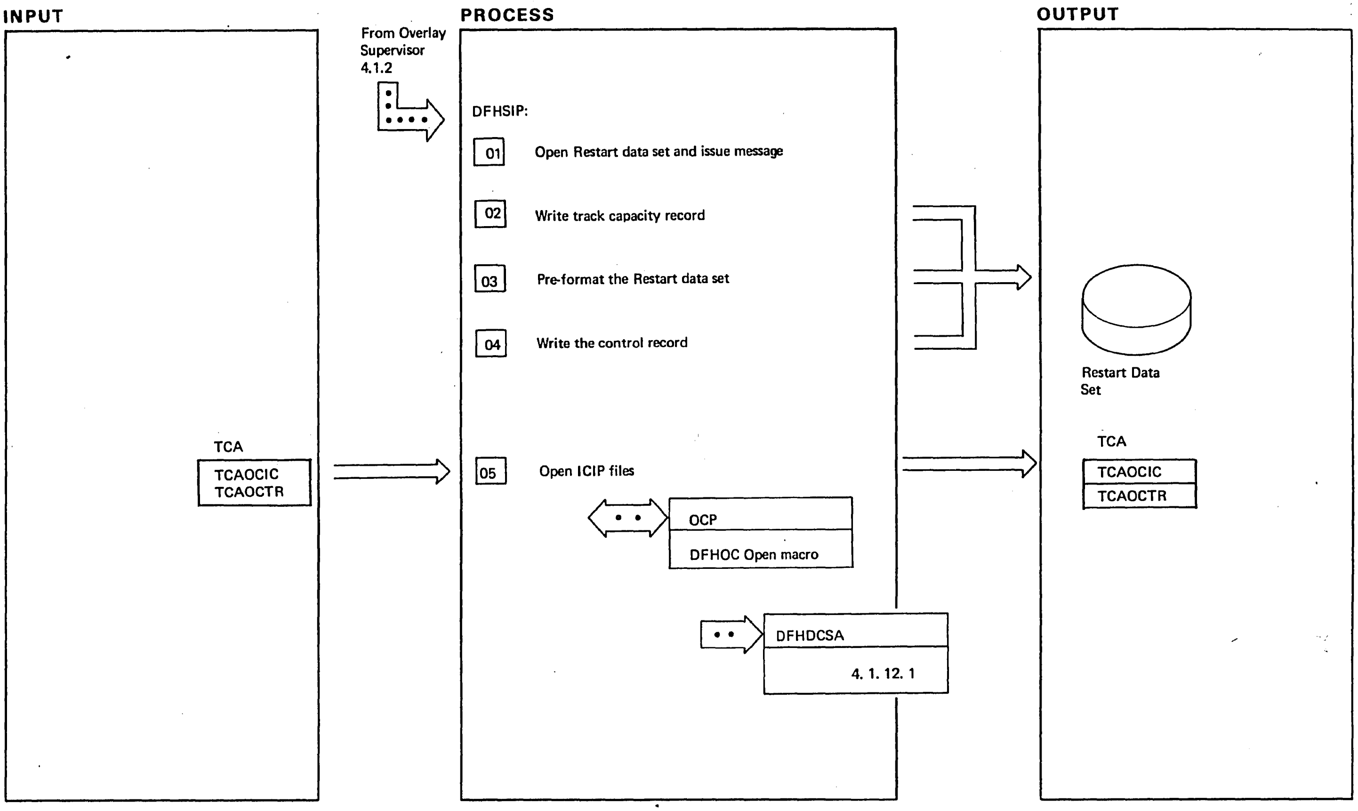
Diagram - 4.1.11-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 08 Issue DFHPC TYPE=LINK to DFHRUP if emergency restart. If Link fails issue message DFH1586. | Console WRITE | TDRSUBP | | | | | |
| 09 Message DFH1500 is issued. | Console WRITE | CSAEXIT | | | | | |
| 11 Messages DFH1578 and DFH1579 are issued. | Console WRITE | SIPPLTLP | | | | | |

Initialize Temporary Storage Control Blocks (DFHSII1)

Diagram - 4.1.11-02

Page 1 of 1


INPUT
PROCESS
OUTPUT
From Overlay Supervisor 4.1.2
DFHSIP:
01 Open Restart data set and issue message
02 Write track capacity record
03 Pre-format the Restart data set
04 Write the control record
05 Open ICIP files
OCP
DFHOC Open macro
DFHDCSA
4. 1. 12. 1
TCA
TCAOCIC
TCAOCTR
Restart Data Set
TCA
TCAOCIC
TCAOCTR

Initialize Restart Data Set (DFHSIJ1)

Diagram - 4.1.12 -01

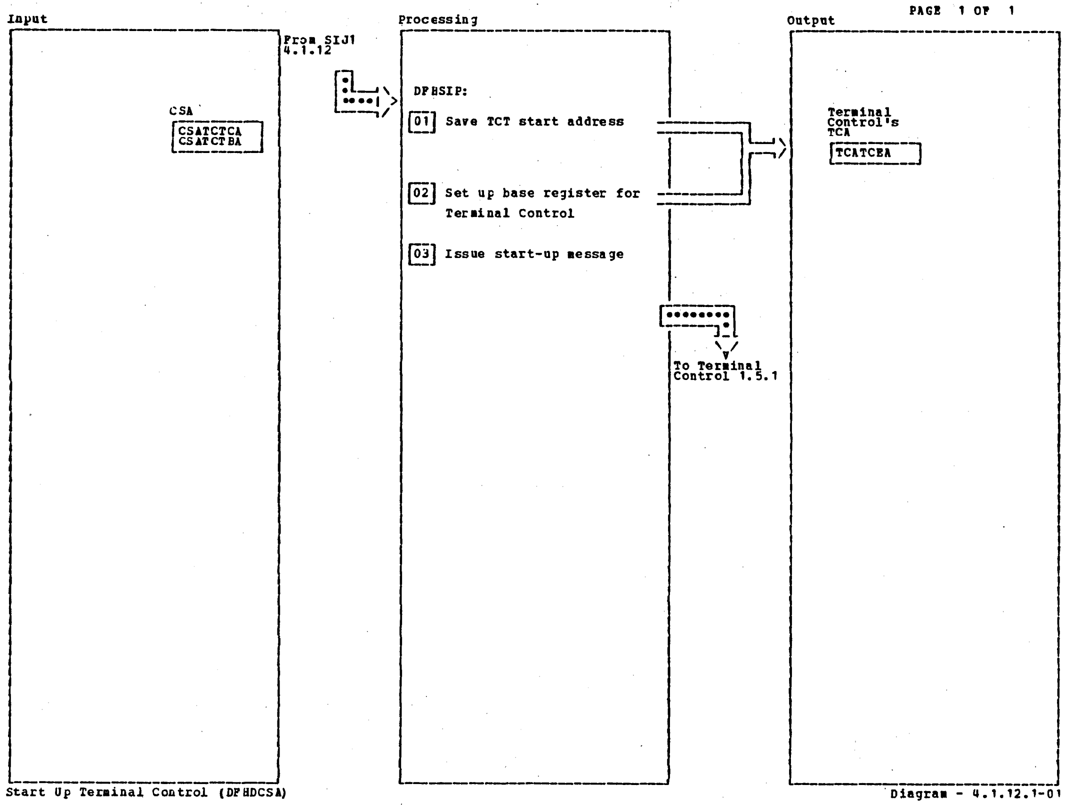
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Messages DFH1500, DFH1506 and DFH1504 are issued | Console WRITE | RSDCLDST<br>SIPEKAB<br>AND<br>EKPSYNAD | |
| 03 The block size specified in system generation of KPP is used to pre-format the Restart data set into fixed-length blocks of an undefined-length data set | | | |
| 05 Message 'DFH1500-ICIP Files being opened' is issued | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Initialize Restart Data Set (DFHSIJ1)

Diagram - 4.1.12 -01

PAGE 1 OF 1



01 Save TCT start address

02 Set up base register for Terminal Control

03 Issue start-up message

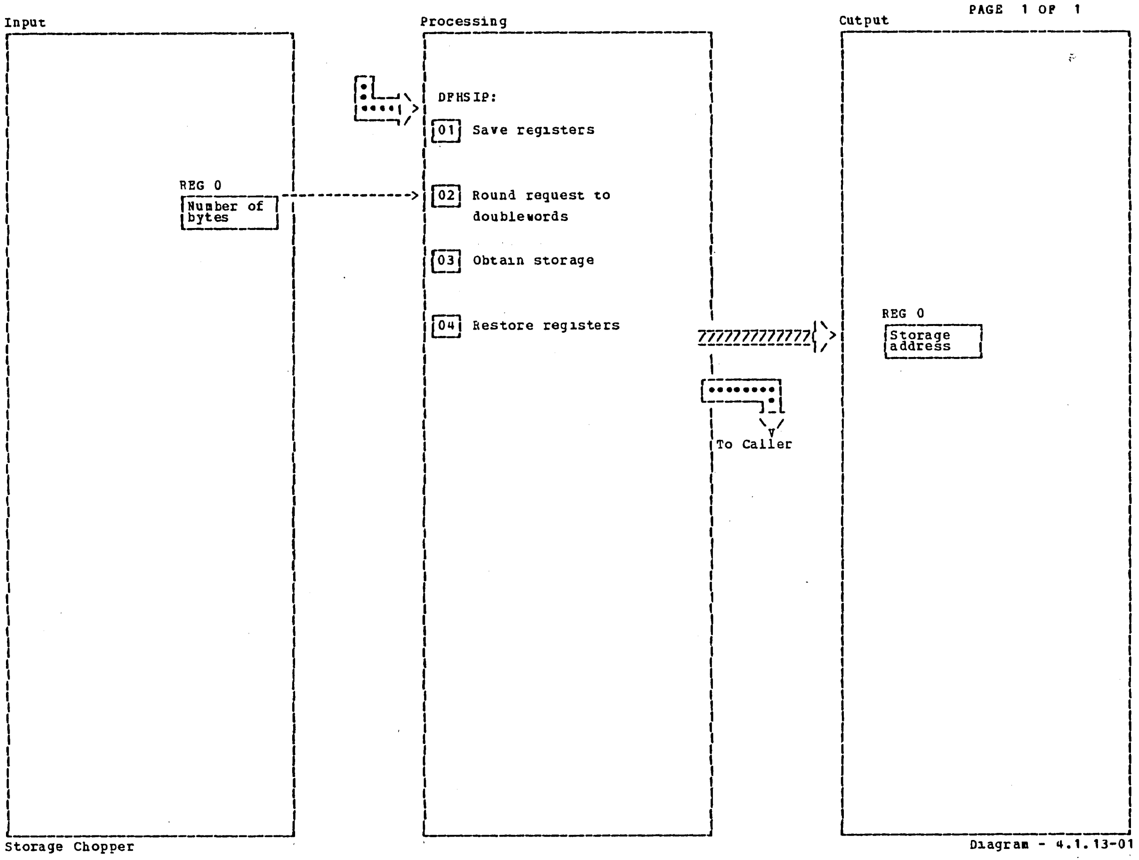
Start Up Terminal Control (DFHDCSA)

Diagram - 4.1.12.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 Message DFH1500 is issued. | | DFHDCSEA | | | | | |

Start Up Terminal Control (DFHDCSA)

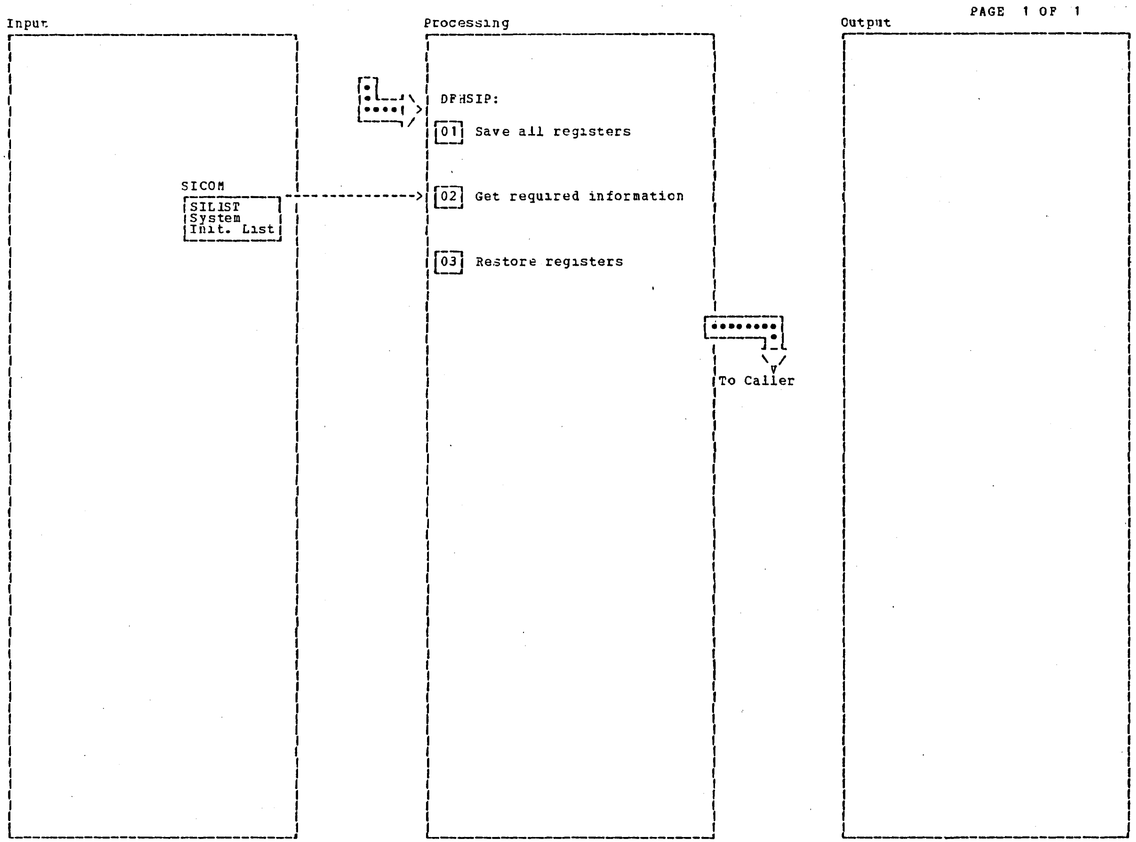
Diagram - 4.1.12.1-01



| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Save caller's registers | | SIGETCOR | | | | | |
| 03 The storage is allocated by SIP Storage Control is NOT called. If the request cannot be satisfied, ABEND code 1599 is issued. | | DIPLDAB1 | | | | | |
| 04 All registers except Register 0 remain unchanged. | | | | | | | |

Storage Chopper

Diagram - 4.1.13-01



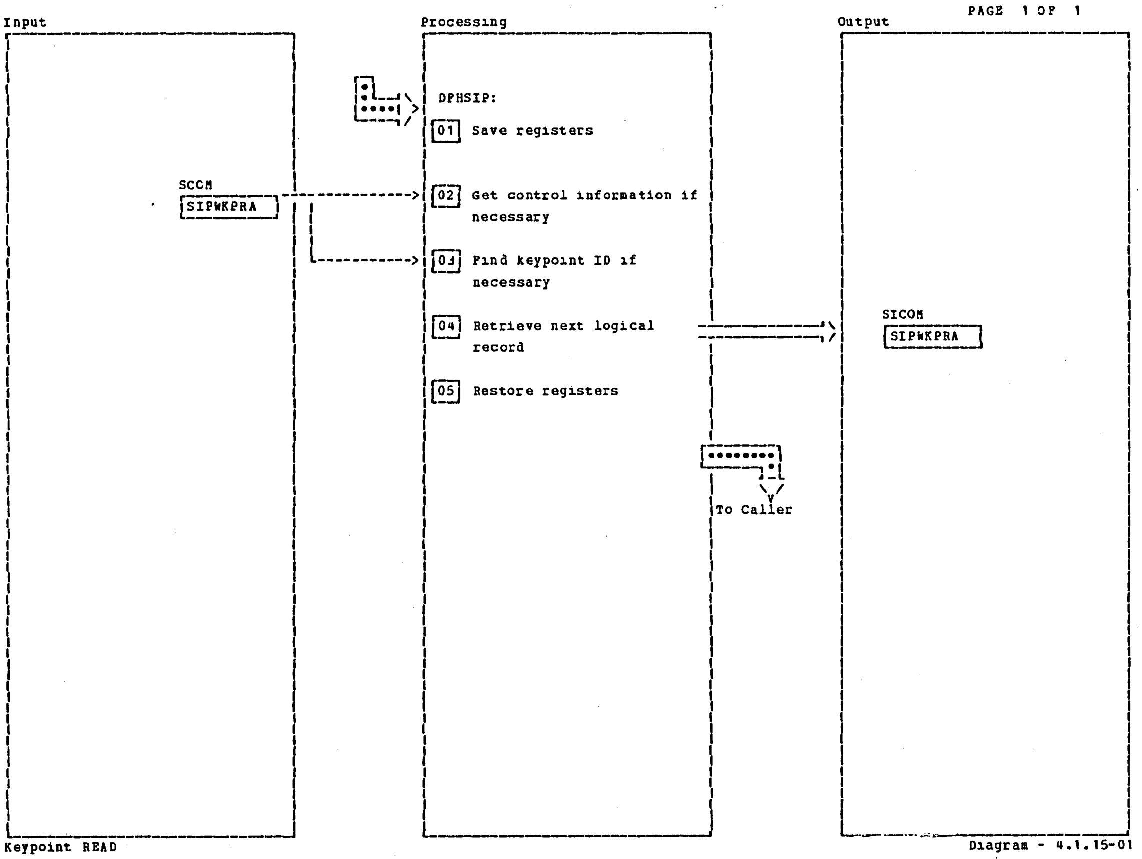
Get Directory Information about a Program

Diagram - 4.1.14-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | SIPBLDLE | | | | | |
| 02 An operating system LOAD with no text information is issued, pointing to the SI list which is an internal list in SIP. If the BLDL is unsuccessful and the program is not an application program, ABEND code 1596 is issued. | | SIP1596 | | | | | |

Get Directory Information about a Program

Diagram - 4.1.14-01



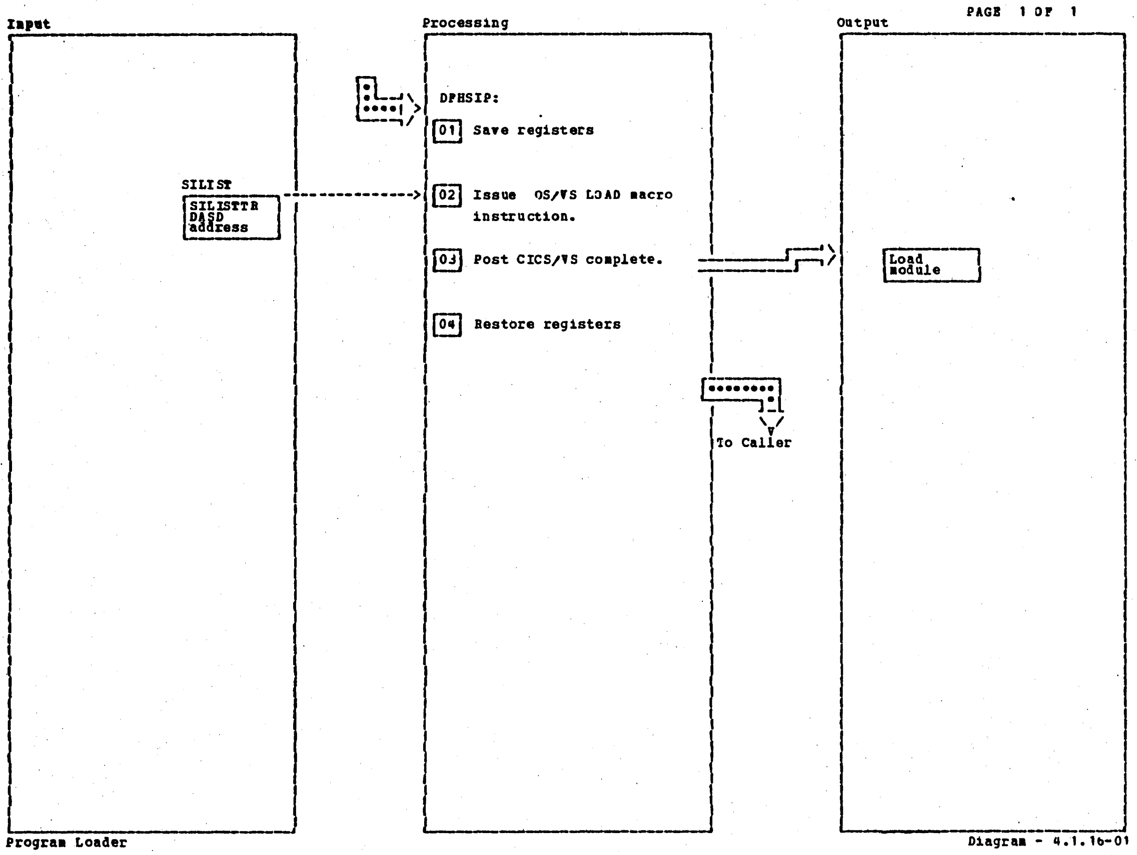
Keypoint READ

Diagram - 4.1.15-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | SIPDPPRD | | | | | |
| 02 The contrcl record is read only for the first request. The information is placed in an area in the Keypoint prcgram. | | RSDCTLRD | | | | | |
| 03 The Keypcint ID table is searched for GETFIRST requests only. (SICOM is an internal communication area in SIP.) | | RSDGETF | | | | | |
| 04 The keypcint records are deblocked and the next logical record is retrieved. | | RSDGETN | | | | | |
| 05 | | RSDEXIT | | | | | |

Keypoint REAC

Diagram - 4.1.15-01



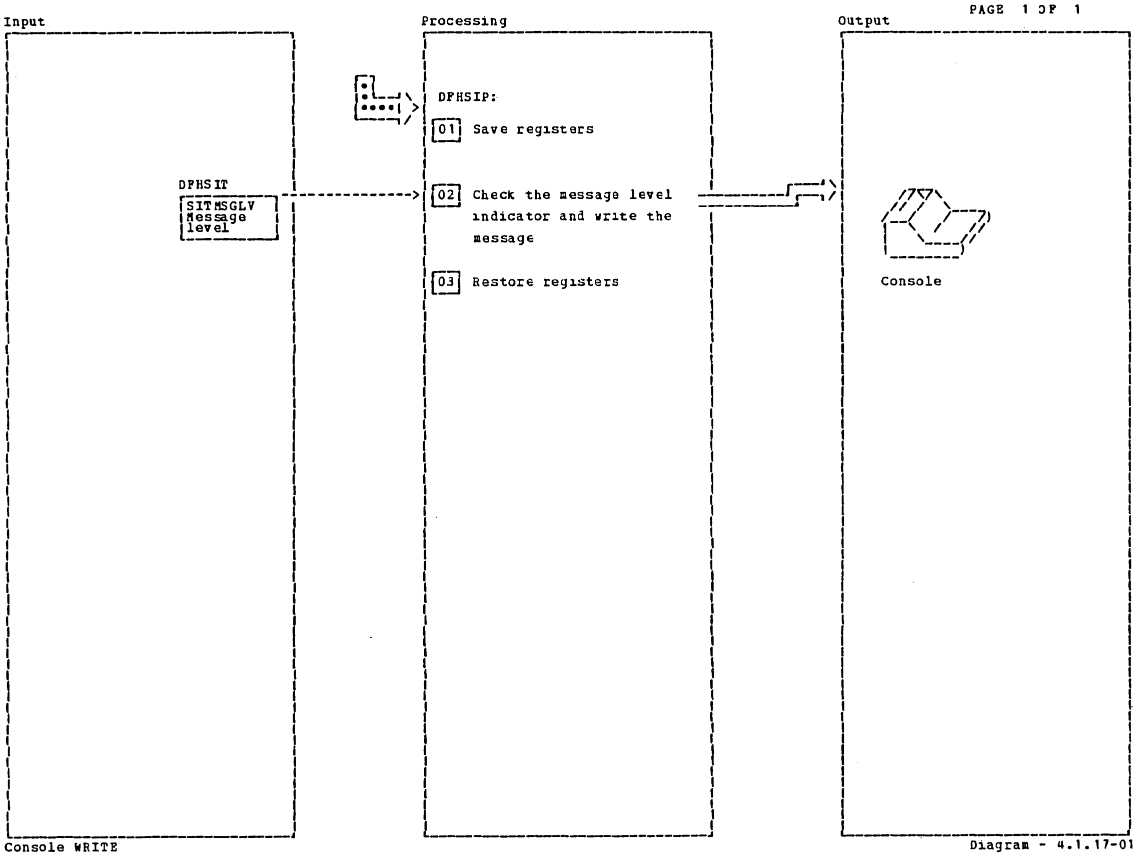
| Notes | Routine | Label | Ref |
|---|---|---|---|

| Notes | Routine | Label | Ref |
|---|---|---|---|

Program Loader

Diagram - 4.1.16-01

PAGE 1 OF 1



01 Save registers

02 Check the message level indicator and write the message

03 Restore registers

Console WRITE

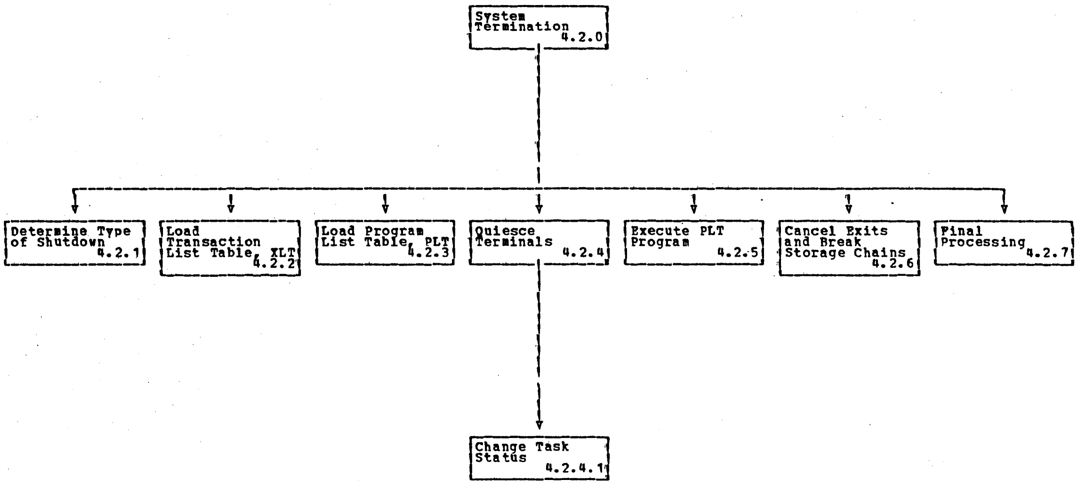
Diagram - 4.1.17-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 If the message level is zero, no message is written. The message is written only if the message level is one.<br>The message level is identified in the SIT. |  | SIPCONS |  |  |  |  |  |

Console WRITE

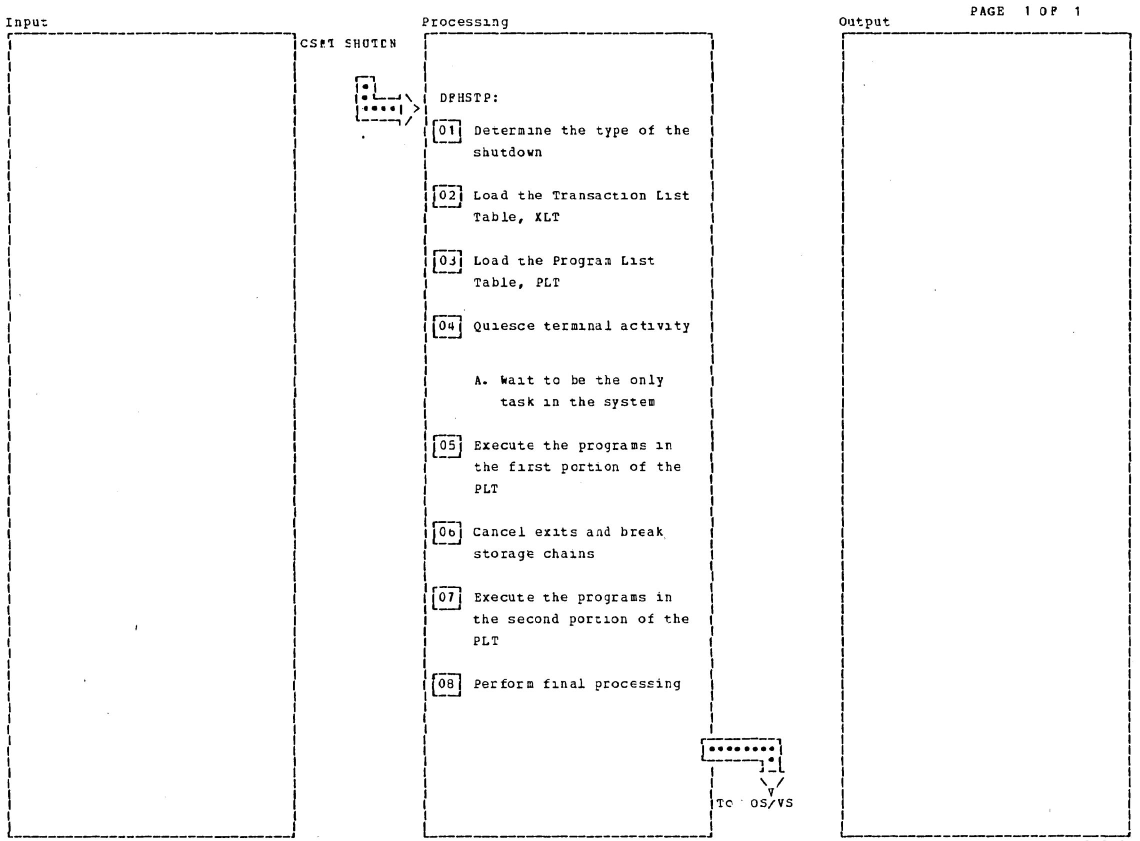
Diagram - 4.1.17-01

PAGE 1 OF 1


System Termination 4.2.0
Determine Type of Shutdown 4.2.1
Load Transaction List Table, XLT 4.2.2
Load Program List Table, PLT 4.2.3
Quiesce Terminals 4.2.4
Execute PLT Program 4.2.5
Cancel Exits and Break Storage Chains 4.2.6
Final Processing 4.2.7
Change Task Status 4.2.4.1

System Termination

Diagram - 4.2-01



DFHSTP:

[01] Determine the type of the shutdown

[02] Load the Transaction List Table, XLT

[03] Load the Program List Table, PLT

[04] Quiesce terminal activity

A. Wait to be the only task in the system

[05] Execute the programs in the first portion of the PLT

[06] Cancel exits and break storage chains

[07] Execute the programs in the second portion of the PLT

[08] Perform final processing

System Termination Overview

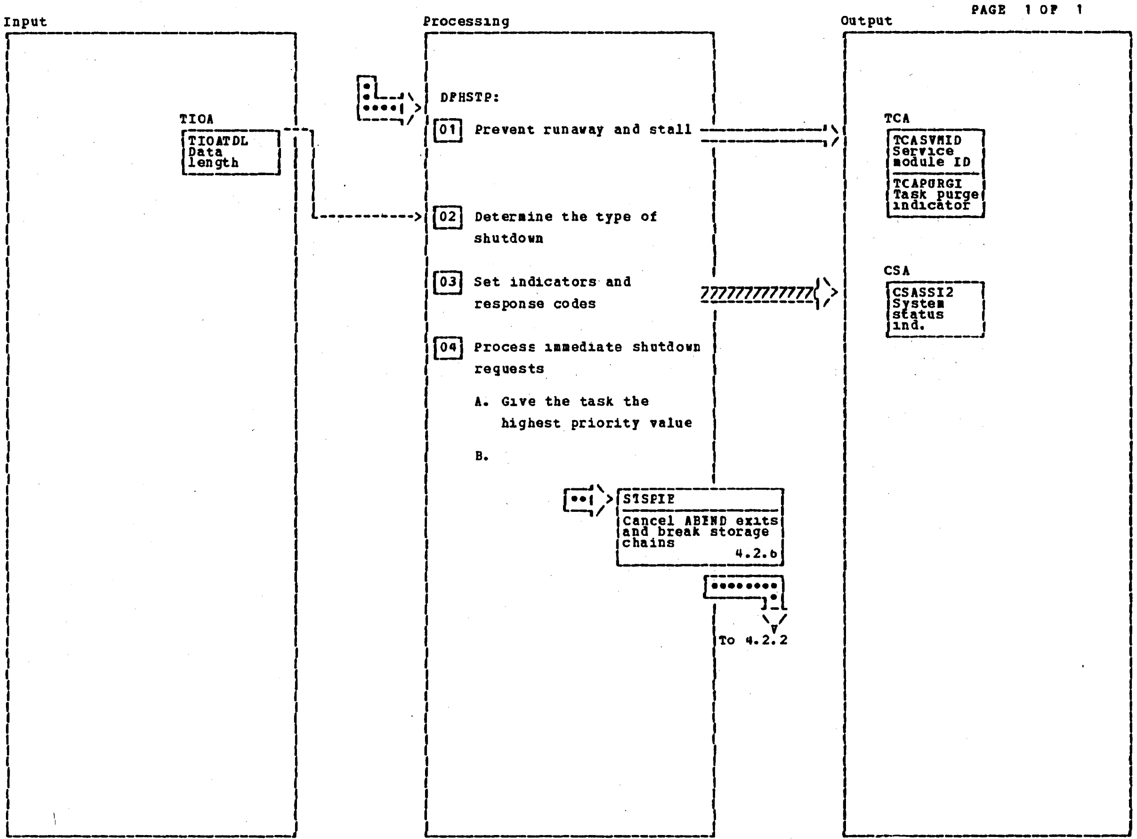
Diagram - 4.2.0-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| [01] | | STASKQ | 4.2.1 | | | | |
| [02] | | STXLTLD | 4.2.2 | | | | |
| [03] | | STPLTLD | 4.2.3 | | | | |
| [04] | | | 4.2.4 | | | | |
| A. | STCHAP | | 4.2.4.1 | | | | |
| [05] | | STPLTLNK | 4.2.5 | | | | |
| [06] | STSFIE | | 4.2.6 | | | | |
| [07] | | STPPLT2 | 4.2.5 | | | | |
| [08] | | STSHUDLI | 4.2.7 | | | | |

System Termination Overview

Diagram - 4.2.0-01

PAGE 1 OF 1


Input
Processing
Output
TIOA
TIOATDL Data length
DFHSTP:
01 Prevent runaway and stall
02 Determine the type of shutdown
03 Set indicators and response codes
04 Process immediate shutdown requests
A. Give the task the highest priority value
B.
STSPIE
Cancel ABEND exits and break storage chains
4.2.6
To 4.2.2
TCA
TCASVMID Service module ID
TCAPURGI Task purge indicator
CSA
CSASSI2 System status ind.

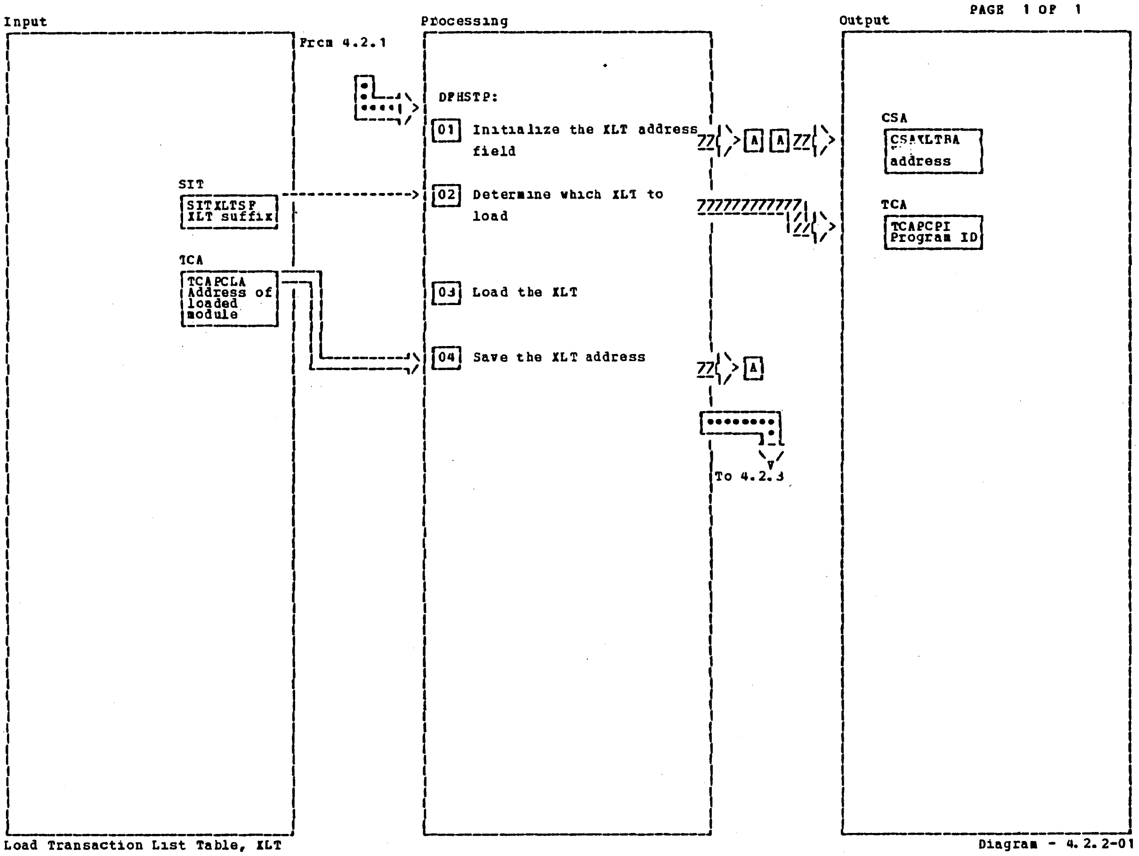
Determine Type of Shutdown

Diagram - 4.2.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 If the data length is zero, i.e., the shutdown type was not supplied, the operator is asked (by message DFH1700) to supply it. | | STASKQ | | | | | |
| 03 If the shutdcwn is not immediate, the busy-switch is set.<br>One of messages DFH1701, DFH1702, DFH1703, or DFH1704 is sent to the operator | | | | | | | |

Determine Type of Shutdown

Diagram - 4.2.1-01



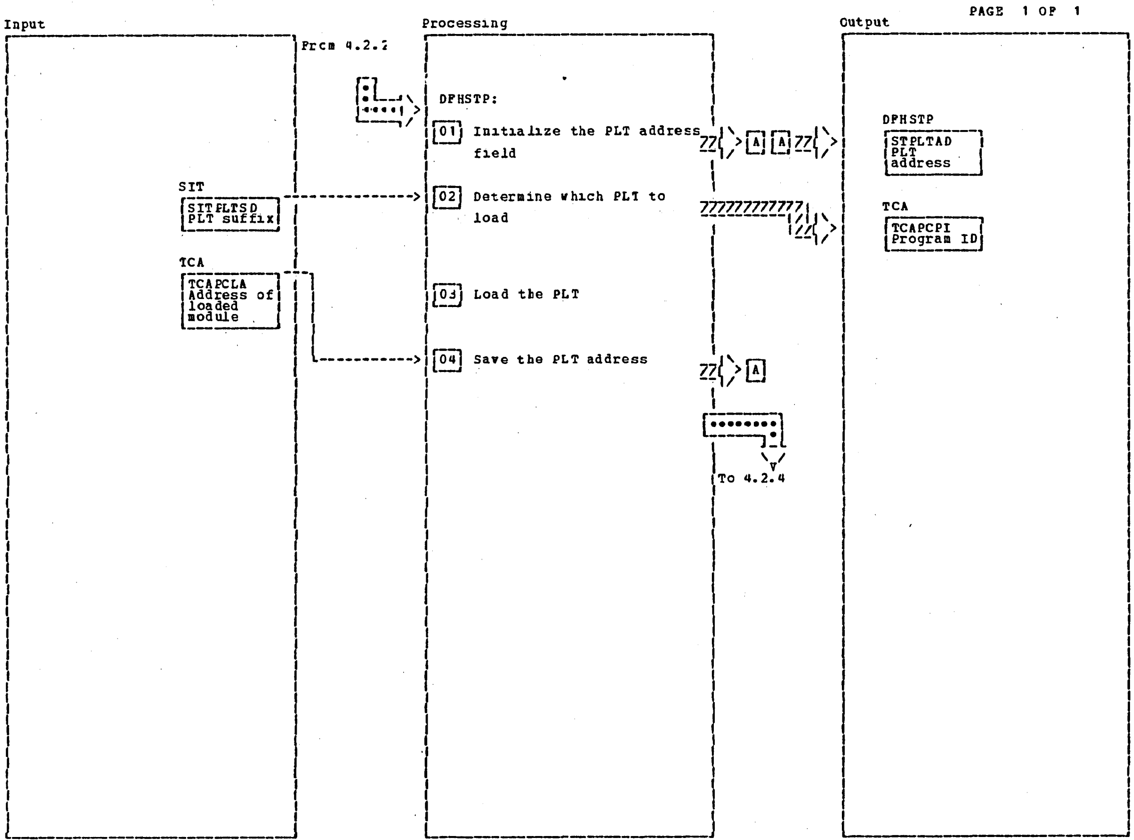
Load Transaction List Table, XLT

Diagram - 4.2.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The XLT address field is set tc zero. | | STXLTLD | | | | | |
| 02 The XLT suffix supplied in the shutdown request determines which XLT to lcad. If no suffix was given, the one specified by the XLT= operand of the DFHSIT is used.<br>If XLT=NO was specified, i.e., no table was supplied, the XLT load is bypassed. | | | | | | | |
| 03 If the table was not found, message DFH1750 is issued. | | STXLTLD1 | | | | | |

Load Transaction List Table, XLT

Diagram - 4.2.2-01



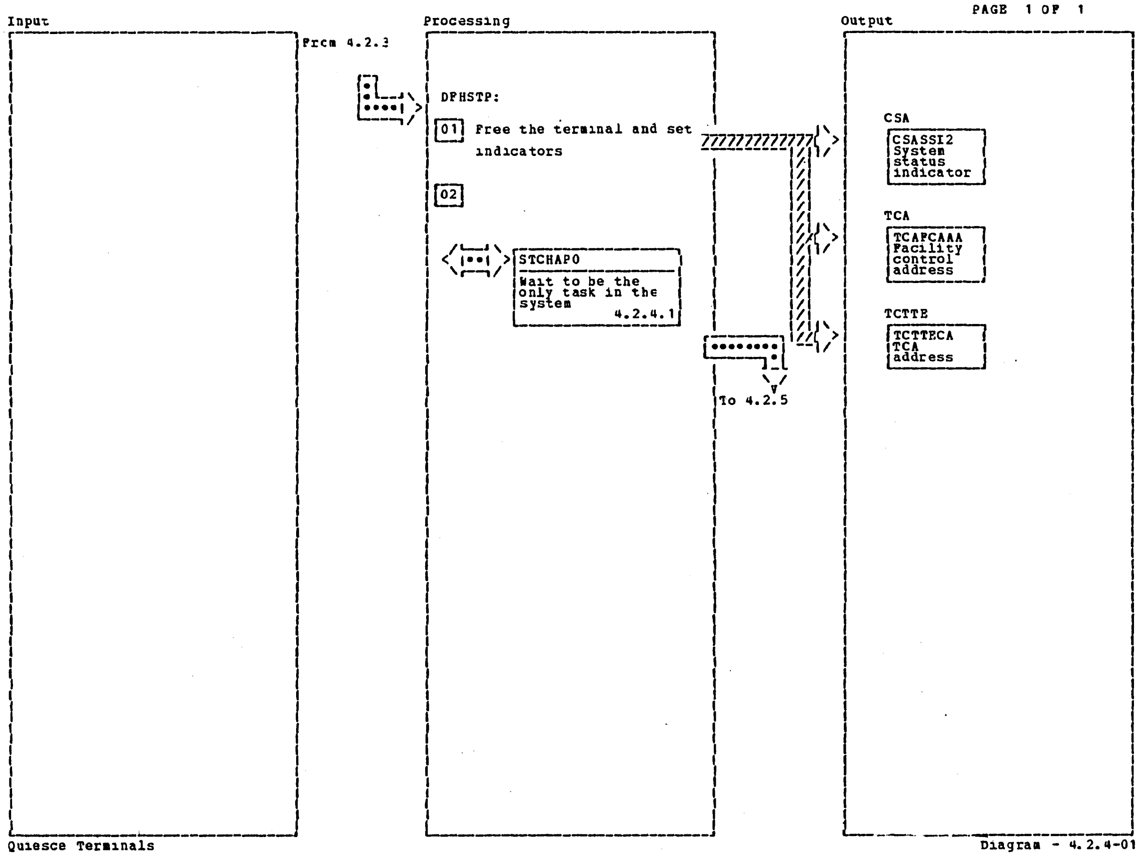
Load Program List Table, PLT

Diagram - 4.2.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The PLT address field is set to zero. | | STPLTLD | | | | | |
| 02 The PLT suffix supplied in the shutdown request determines which PLT to load. If no suffix was given, the one specified by PLTSD= operand of the DFHSIT initialization request is used.<br>If PLT=NO was specified, i.e., no table was supplied, the PLT load is bypassed. | | | | | | | |
| 03 If the table was not found, message DFH1750 is issued. | | STPLTLD1 | | | | | |

Load Program List Table, PLT

Diagram - 4.2.3-01


PAGE 1 OF 1
Input
Processing
Output
From 4.2.3
DFHSTP:
01 Free the terminal and set indicators
02
STCHAPO
Wait to be the only task in the system
4.2.4.1
To 4.2.5
CSA
CSASSI2 System status indicator
TCA
TCAFCAAA Facility control address
TCTTE
TCTTECA TCA address

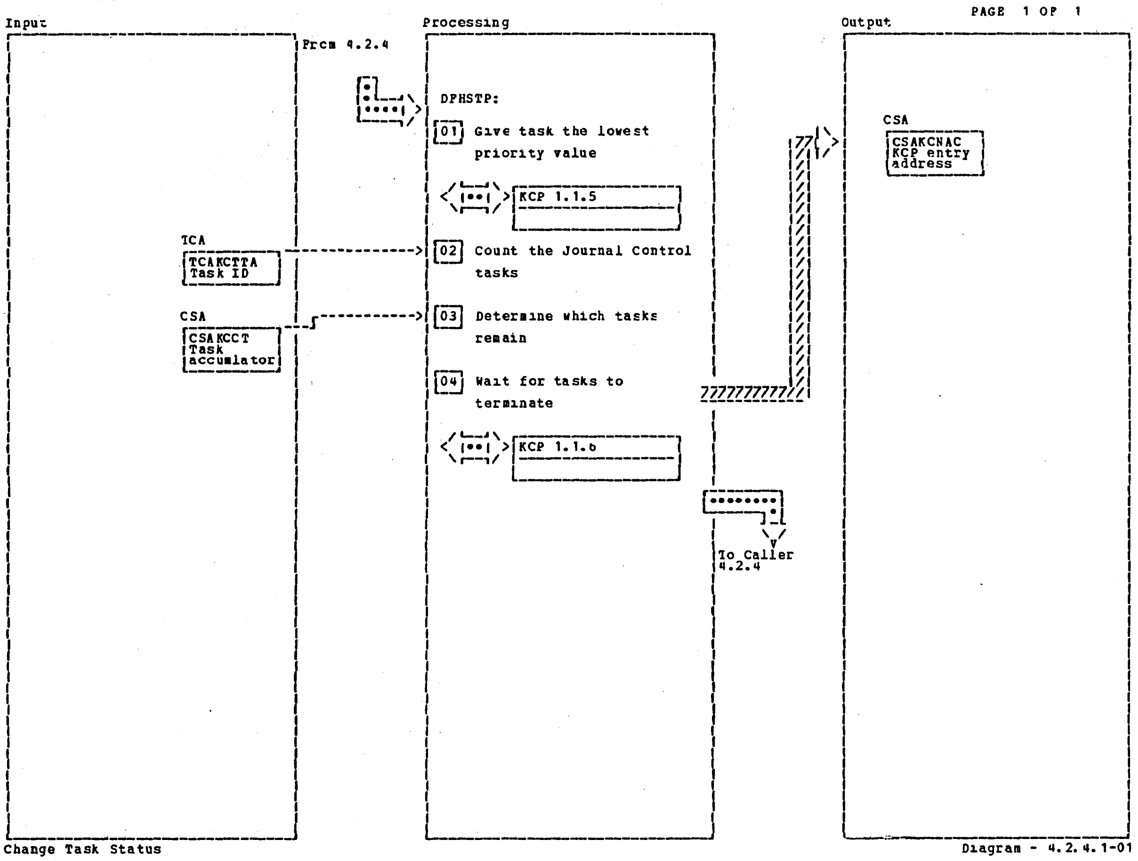
Quiesce Terminals

Diagram - 4.2.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The fields belonging to the freed terminal are set to zero. The terminal quiesce indicator is turned on. | | STPLTLDX | | | | | |

Quiesce Terminals

Diagram - 4.2.4-01


PAGE 1 OF 1
Input
Processing
Output
Frcm 4.2.4
DFHSTP:
01 Give task the lowest priority value
KCP 1.1.5
TCA
TCAKCTTA Task ID
02 Count the Journal Control tasks
CSA
CSAKCCT Task accumlator
03 Determine which tasks remain
04 Wait for tasks to terminate
KCP 1.1.6
To Caller 4.2.4
CSA
CSAKCNAC KCP entry address

Change Task Status

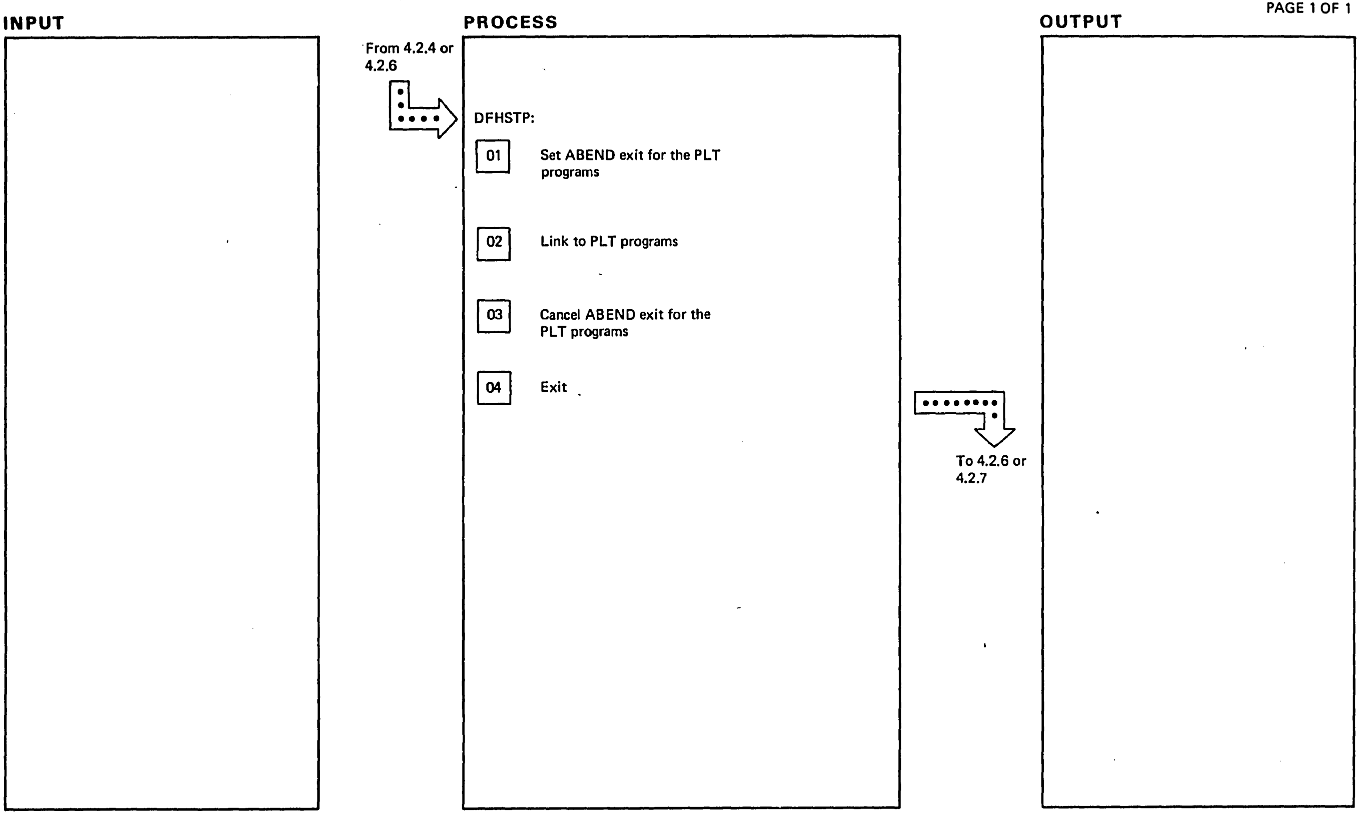
Diagram - 4.2.4.1-01

| Notes | Roctine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The priority value is set to zero tc make this task, i.e., System Terminaticn, the last task in the system. | STCHAFO | | | | | | |
| 02 The active TCA chain and the suspended TCA chain are scanned. Each task having JJJ as its ID is counted. | | STCHAP1 | | | | | |
| 03 If only the System Termination task and Journal Control tasks remain, i.e., the value calculated in Step 2 equals the value accumulated in CSAKCTT, no WAIT is issued. | | STCHAP6 | | | | | |
| 04 If any tasks other than Journal Control tasks remain, the ECBs posted ty Task Ccntrol for CICS DETACH requests are turned off and a WAIT is issued. When the WAIT is posted complete, this routine is re-entered. | | | | | | | |

Change Task Status

Diagram - 4.2.4.1-01

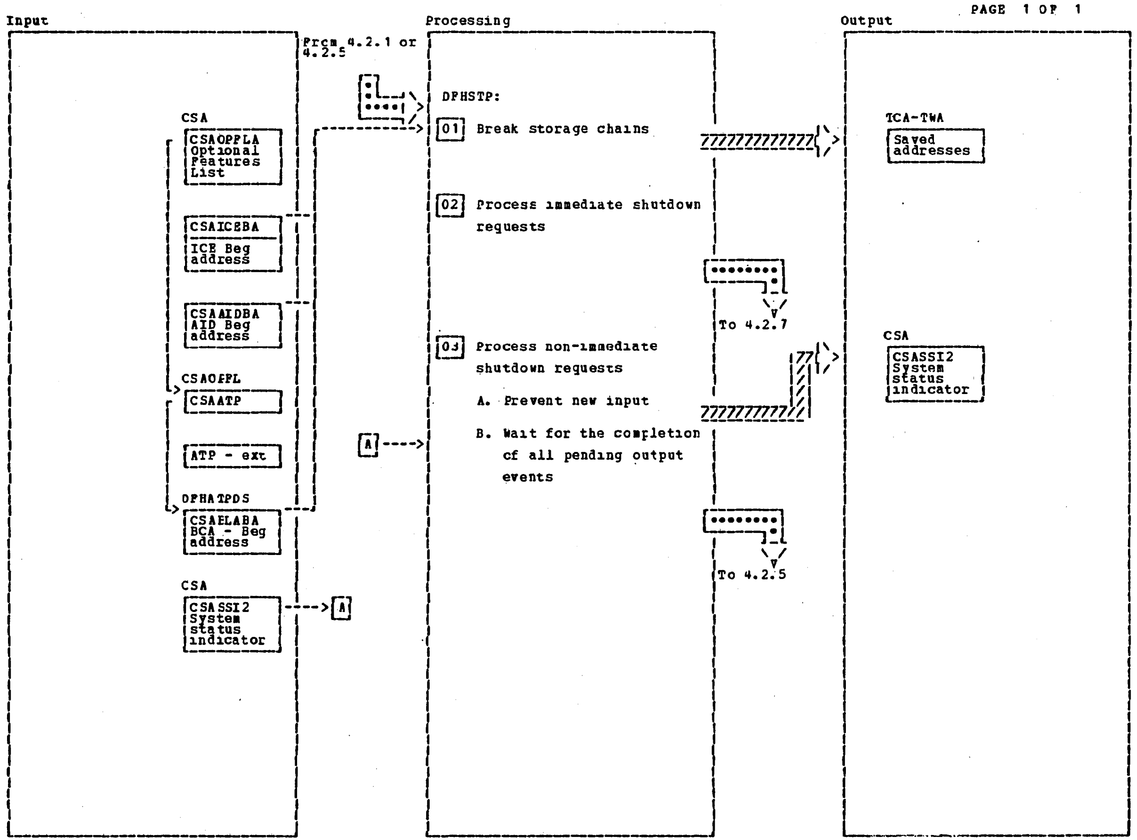
PAGE 1 OF 1



Execute PLT Program

Diagram — 4.2.5-01

| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 01 | The PLT, if one exists, may be divided into two portions. | | | |
| 02 | A Program Control LINK is issued. If a program error occurs, message DFH1780 is issued. If a program is not found, message DFH1752 is issued. The linking stops either when the DFHDELIM program is encountered or when the end of the PLT is reached. | | STPLTLNK for first portion<br>STPLT2 for second portion | |
| | The above processing is performed twice consecutively, once for each portion. If no table was loaded, this processing is bypassed. If only one portion exists, this processing is performed only once. | | | |
| 03 | | | | |
| 04 | After executing the first portion of the PLT programs, the exit is to Chart 4.2.6.<br>After executing the second portion of the PLT programs, the exit is to Chart 4.2.7. | | | |


PAGE 1 OF 1
Input
Processing
Output
From 4.2.1 or 4.2.5
CSA
CSAOPFLA Optional Features List
CSAICEBA ICE Beg address
CSAAIDBA AID Beg address
CSAOPFL
CSAATP
ATP - ext
DFHATPDS
CSAELABA BCA - Beg address
CSA
CSASSI2 System status indicator
A
DFHSTP:
01 Break storage chains
02 Process immediate shutdown requests
To 4.2.7
03 Process non-immediate shutdown requests
A. Prevent new input
B. Wait for the completion of all pending output events
To 4.2.5
TCA-TWA
Saved addresses
CSA
CSASSI2 System status indicator

Cancel Exits and Break Storage Chains

Diagram - 4.2.6-01

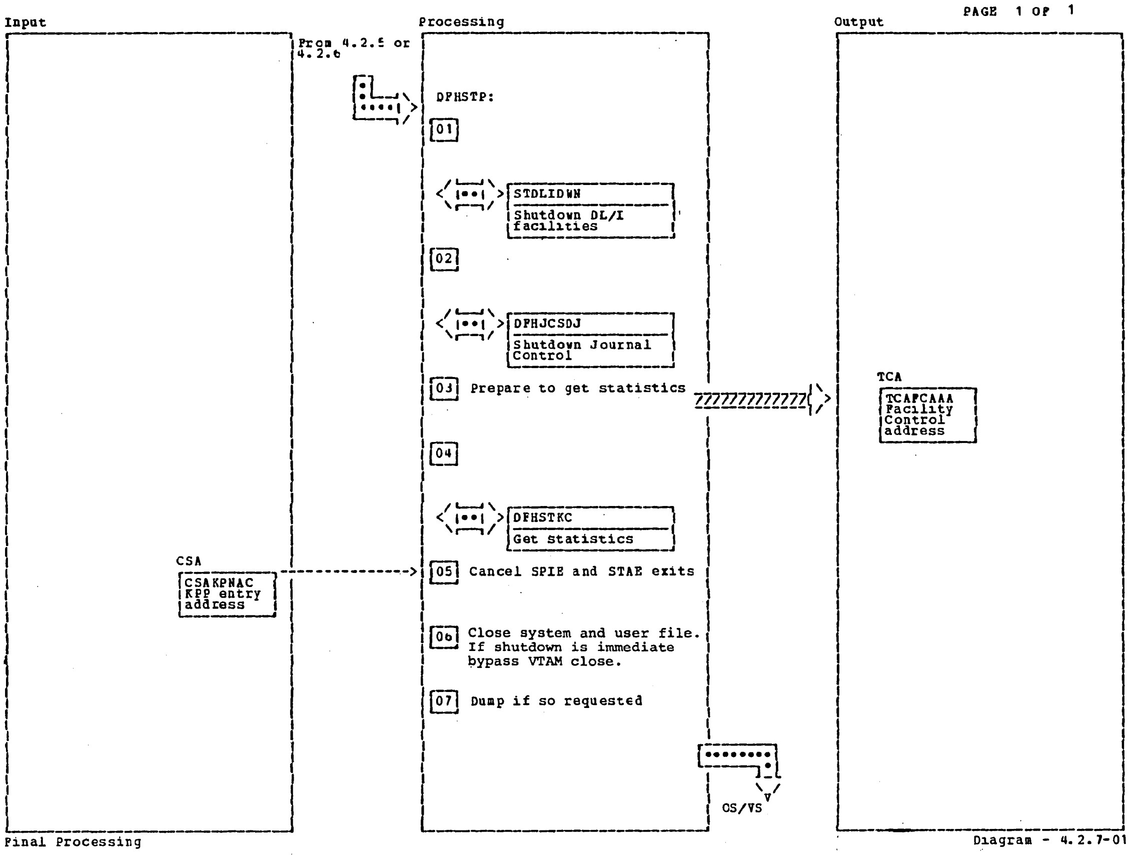
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The starting addresses of the BCA, AIC, and ICE chains are saved in the master terminals TWA. The original pointers are then set tc zeros. | | STBREAK | |
| 03 The igncre input switch is set to prevent new inputs.<br>When Terminal Ccntrol has served its last output, it posts this in the CSA. | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Cancel Exits and Break Storage Chains

Diagram - 4.2.6-01

PAGE 1 OF 1


Input
Processing
Output
From 4.2.5 or 4.2.6
DFHSTP:
01
STDLIDWN
Shutdown DL/I facilities
02
DFHJCSDJ
Shutdown Journal Control
03 Prepare to get statistics
TCA
TCAFCAAA Facility Control address
04
DFHSTKC
Get statistics
CSA
CSAKPNAC KPP entry address
05 Cancel SPIE and STAE exits
06 Close system and user file. If shutdown is immediate bypass VTAM close.
07 Dump if so requested
OS/VS

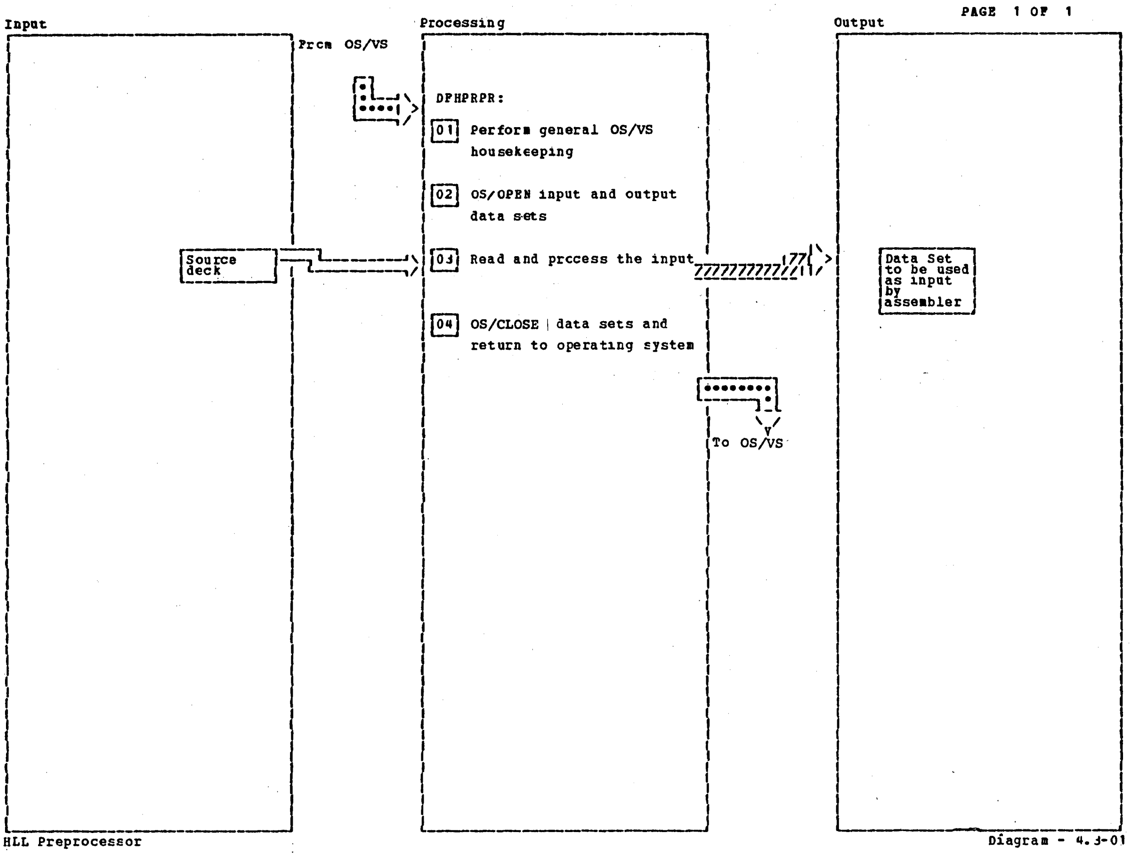
Final Processing

Diagram - 4.2.7-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 A File Control TYPE=DL/I,UNLD is issued. | | STSHUDLI | | | | | |
| 02 A Program Control LINK is issued for DFHJCSDJ. | | | | | | | |
| 03 The Facility Control address is set to zero. It is used to indicate all statistics. | | STPSTAT | | | | | |
| 04 A Program Control LINK is issued for DFHSTKC. | | STPSTAT | | | | | |
| 06 A warm Keypoint is requested. Upon return, one of the following messages is issued: DFH1790, DFH1792, DFH1794, or DFH1796. | | STKPLNK | | | | | |
| 07 If a dump was requested, message DFH1798 is issued. If a dump was not requested, message DFH1799 is issued. | | STKPEND | | | | | |

Final Processing

Diagram - 4.2.7-01


PAGE 1 OF 1
Input
Processing
Output
From OS/VS
DFHPRPR:
01 Perform general OS/VS housekeeping
02 OS/OPEN input and output data sets
Source deck
03 Read and process the input
Data Set to be used as input by assembler
04 OS/CLOSE data sets and return to operating system
To OS/VS

HLL Preprocessor

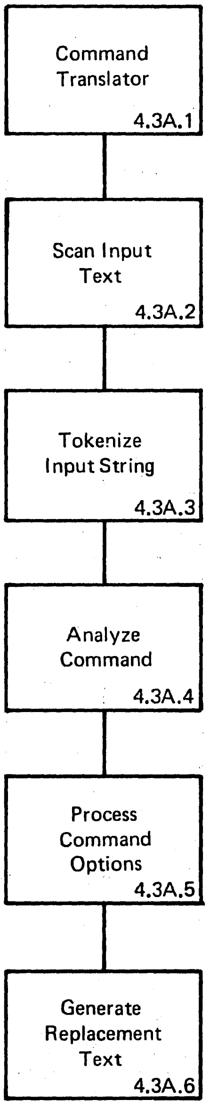
Diagram - 4.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | DFHPRNA | |
| 02 | | DOCPRP02 | |
| 03 The UPSI byte is checked to see if the 'strip' function is desired. If so 81 byte input is converted to 80 byte output. The first 16 columns are scanned for DFH or CALL. | | READ<br>DFHOPEN<br>PRSTRIPN | |
| If the input is not a CICS macro instruction or a DL/I call, the source record and REPRO are written. | | | |
| DL/I calls are changed to CICS calls, and REPRO and a new source record are punched. | | NOTCICS | |
| CICS macro instructions are left unchanged, unless INITIMG or CTLCHAR are found. In this case, the hex code is converted to decimal for ease of manipulation. | | CICS | |
| 04 When the end of the input is found, the END statement is written. | | EOD or PRSTRIPX | |

HLL Preprocessor

Diagram - 4.3-01

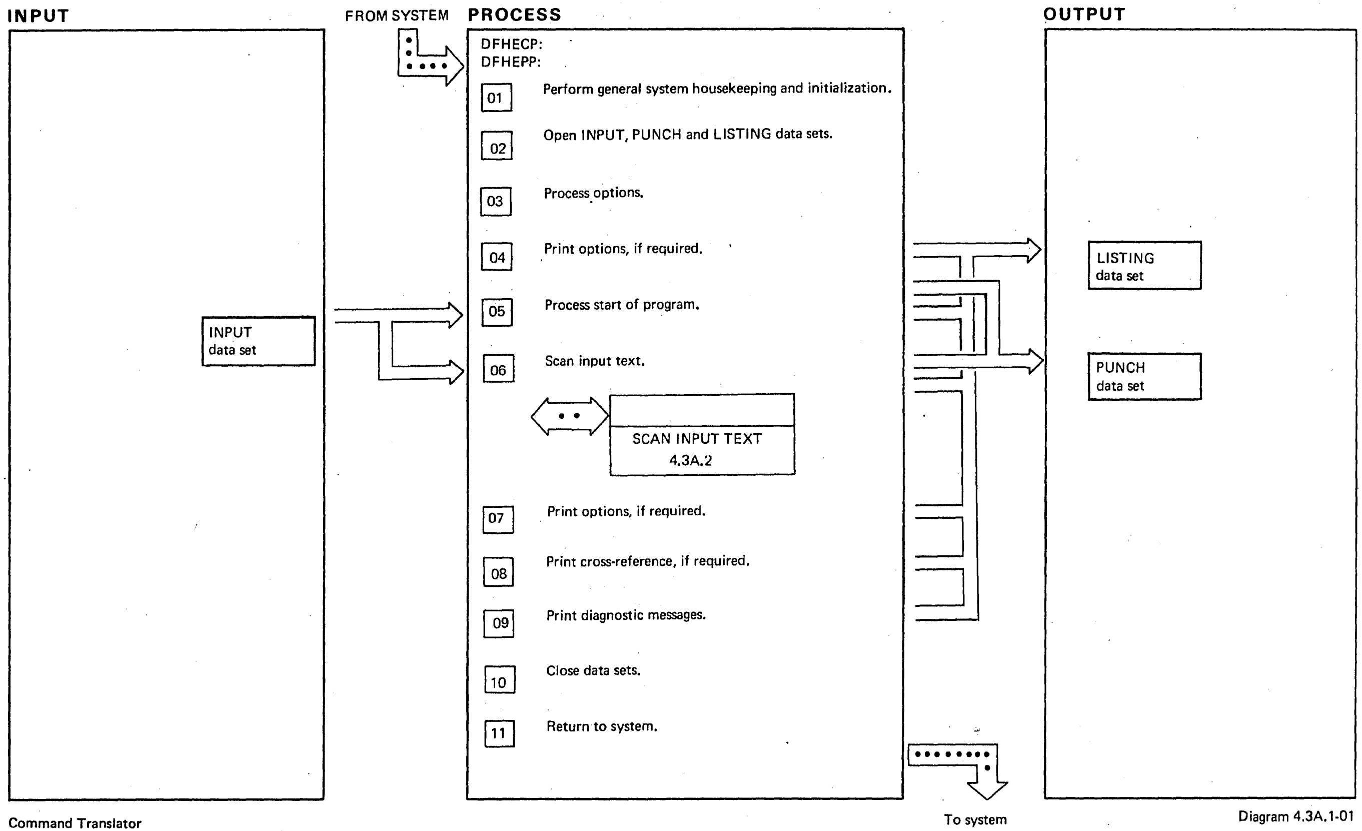
Page 1 of 1



Visual Index — Command Translator

Diagram — 4.3A.0-01

Page 1 of 1



DFHECP:
DFHEPP:

01 Perform general system housekeeping and initialization.

02 Open INPUT, PUNCH and LISTING data sets.

03 Process options.

04 Print options, if required.

05 Process start of program.

06 Scan input text.

07 Print options, if required.

08 Print cross-reference, if required.

09 Print diagnostic messages.

10 Close data sets.

11 Return to system.

Command Translator

Diagram 4.3A.1-01

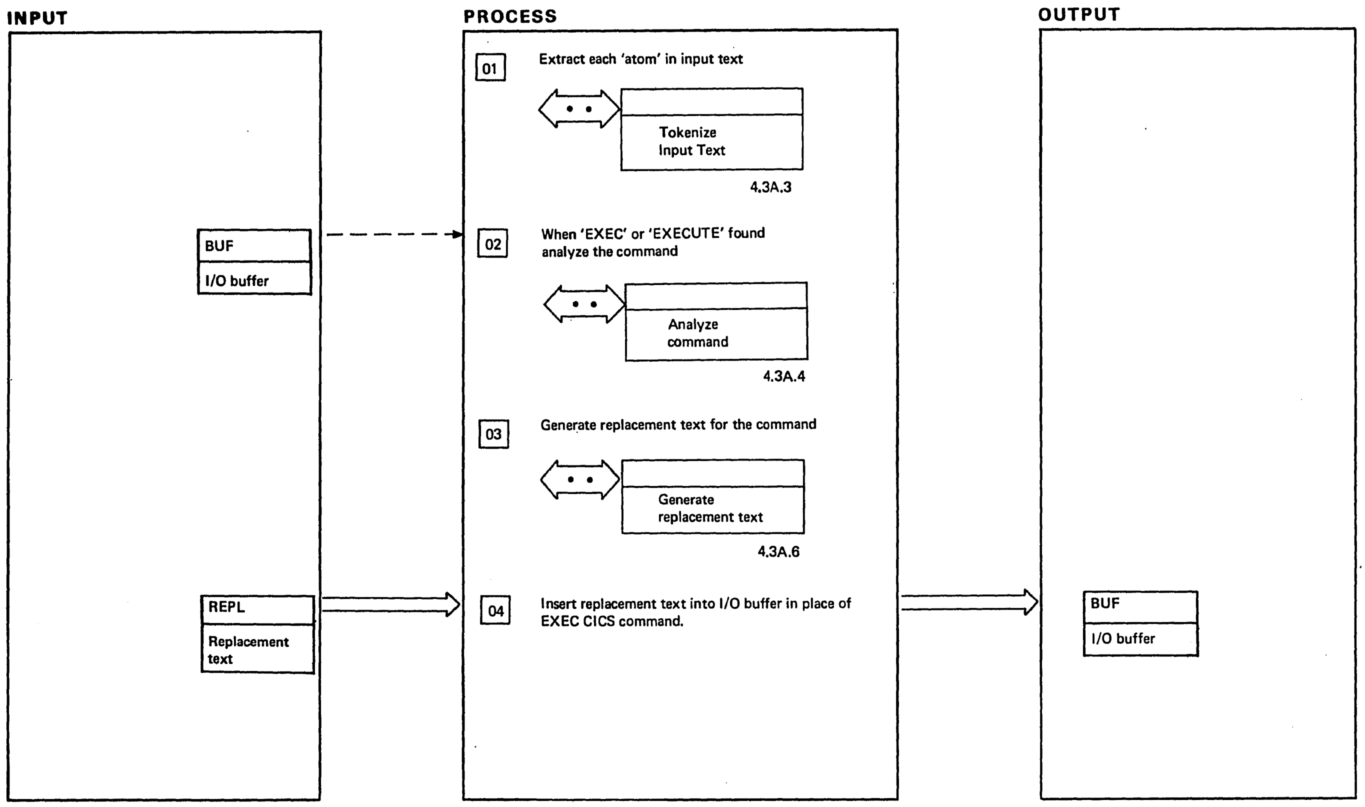
| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 01 | System housekeeping<br>1) Obtain all storage available.<br>2) Determine time and date.<br>3) Set up error exit routine. | DFHEN001 | | |
| 02 | If LISTING data set cannot be opened write to operator (OS only) and abend. If any data set cannot be opened terminate translation. | DFHEN315 | | |
| 03 | Options derived from<br>1) EXEC card (OS only).<br>2) *PROCESS or CBL cards.<br>When all options have been collected they are checked for consistency. | DFHEN005<br>DFHEN007<br>DFHEN008 | | |
| 04 | Options printed at head of listing for PL/I if OPTIONS option set. | DFHEN009 | | |
| 05 | For PL/I if input source start with a procedure statement specifying OPTIONS (MAIN) then an extra parameter is inserted at beginning of parameter list and a % INCLUDE statement generated to include the EIB definition.<br>For COBOL if input source starts with IDENTIFICATION DIVISION then extra parameters are inserted in the linkage section and a COPY statement included in the working storage section. | DFHEN010 | | |
| 07 | Options printed after input source for COBOL. | DFHEN009 | | |
| 08 | Cross-reference of all functions correctly used is printed if XREF option set. | DFHEN025 | | |

| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 09 | Diagnostic messages of severity greater or equal to FLAG value are printed. For PL/I they are sorted by severity order. For both PL/I and COBOL they are printed in lexical order.<br>For Assembler they are included as comments in the output, immediately following the commands to which they apply. They then appear as part of the assembly listing. | DFHEN026 | | |
| 10 | Close data sets. | DFHEN415 | | |
| 11 | For OS only a return code is generated according to highest level diagnostic issued. All storage is released. | | | |

Command Translator

Diagram 4.3A.1-01

Page 1 of 1



INPUT

BUF
I/O buffer

REPL
Replacement text

PROCESS

01 Extract each 'atom' in input text

Tokenize Input Text
4.3A.3

02 When 'EXEC' or 'EXECUTE' found analyze the command

Analyze command
4.3A.4

03 Generate replacement text for the command

Generate replacement text
4.3A.6

04 Insert replacement text into I/O buffer in place of EXEC CICS command.

OUTPUT

BUF
I/O buffer

Scan Input Text

Diagram 4.3A.2–01

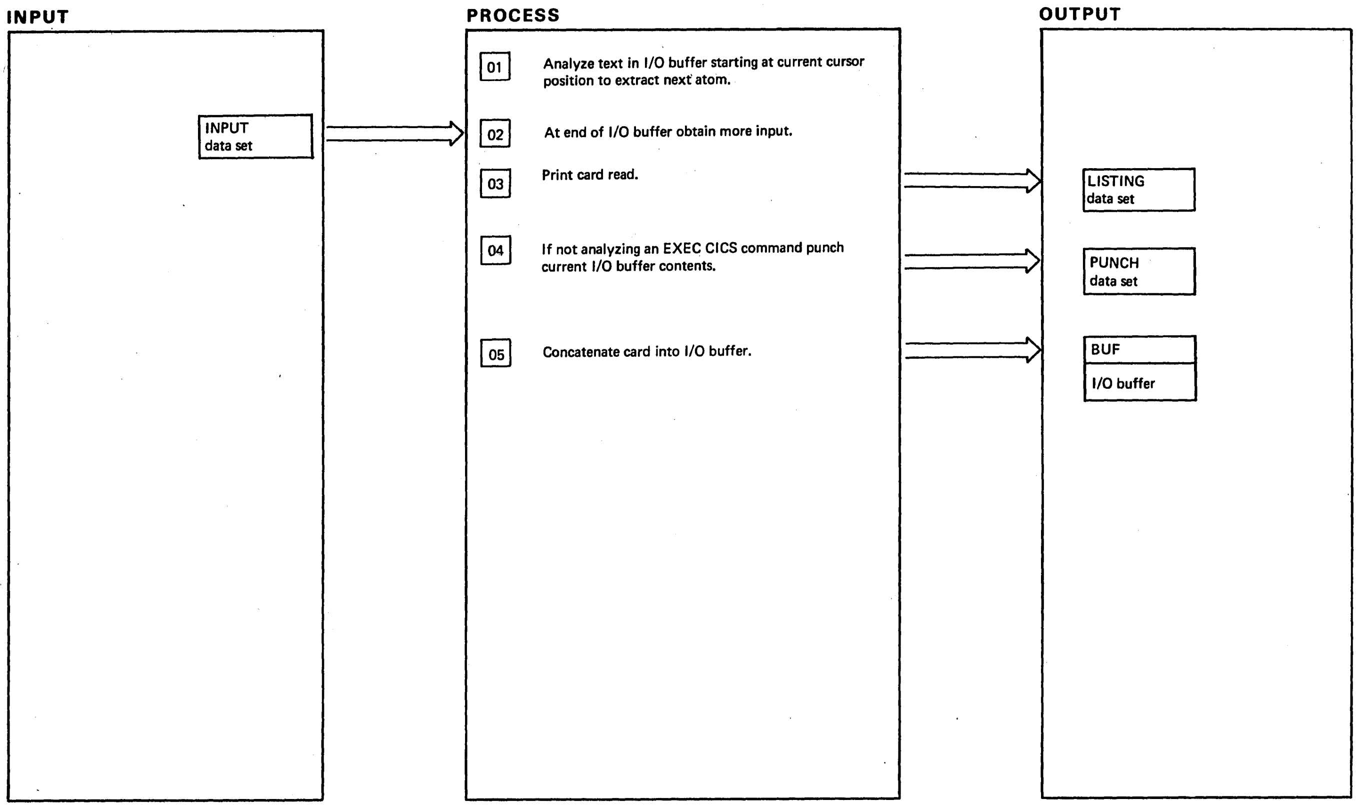
| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 01 | Input text regarded as a contiguous sequence of 'atoms' Read-in routine copes with Cobol continuation conventions. NOTE statements and NOTE paragraphs must not be searched for 'EXEC' or 'EXECUTE'<br>For Assembler only complete EXEC commands are placed in BUF and tokenized. Other text is separately analyzed. | SCANTXT | | |
| 02 | 'EXEC' may or may not be followed by 'CICS'. If 'CICS' appears then analysis successful even if command is to be deleted. If 'CICS' does not appear then analysis fails and no change is made to I/O buffer. | DFHEN019 | | |
| 03 | If analysis successful generate replacement text for command analysed. | DFHEN021 | | |
| 04 | Changed I/O buffer will be punched out when next card read in or when end of file encontered.<br>All text is maintained as card images for Assembler. Whenever analysis of a series of cards is complete they are punched out. The replacement string is also punched out directly without being placed in BUF. Diagnostics, if any, are then generated and punched out. | DFHEN018 | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Scan Input Text

Diagram 4.3A.2–01

Page 1 of 1



01 Analyze text in I/O buffer starting at current cursor position to extract next atom.

02 At end of I/O buffer obtain more input.

03 Print card read.

04 If not analyzing an EXEC CICS command punch current I/O buffer contents.

05 Concatenate card into I/O buffer.

Tokenize Input String

Diagram 4.3A.3–01

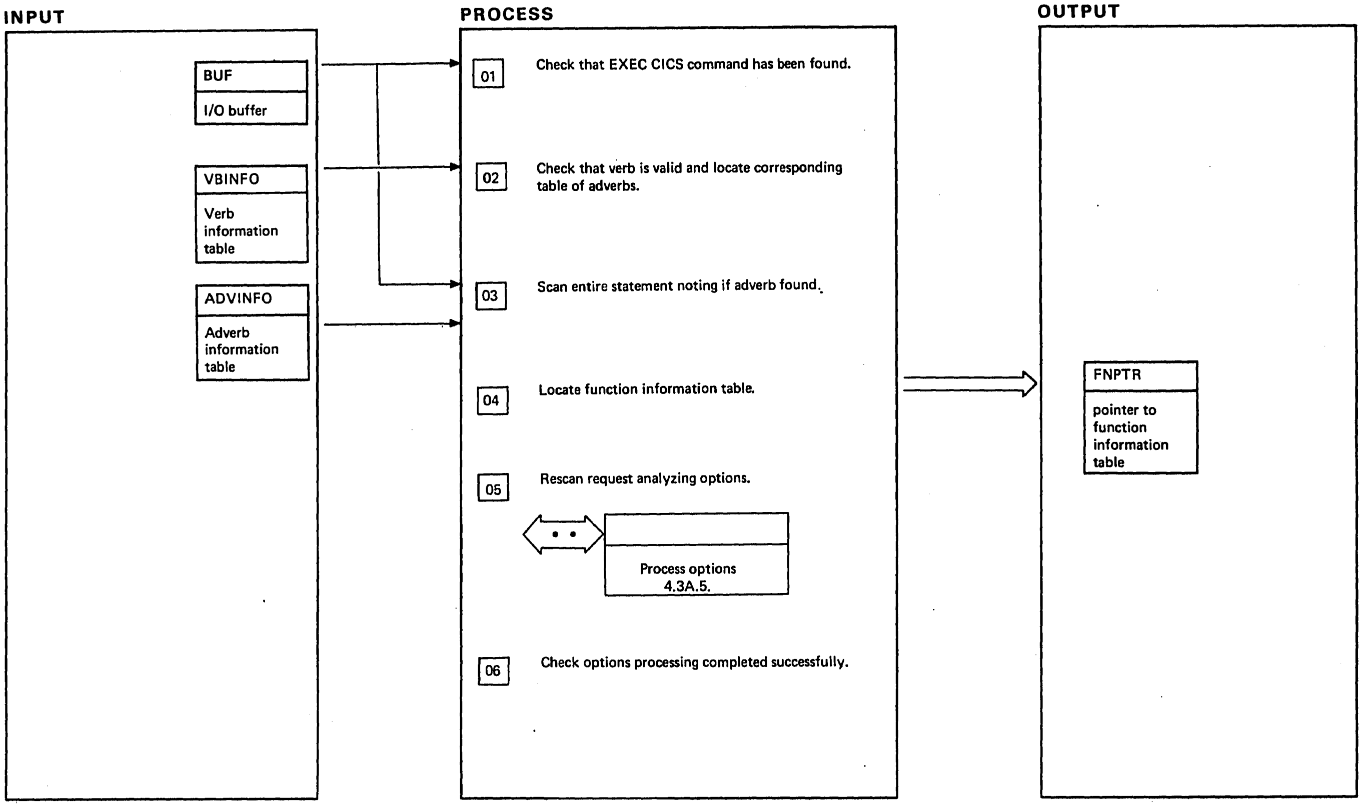
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Blanks and comments are skipped over. If atomization is successful the starting point and ending point of the atom are noted together with the atom type. The cursor is moved to the end of the atom. For Assembler this is all the atomization routine does. It does not control input and output of text. When the routine attempts to look beyond the end of BUF a condition code is set and return is made to the caller. | DFHEN011 | | |
| 02 When the atomization routine attempts to access the first character beyond the end of the I/O buffer the read-in routine is invoked to provide more input. | DFHEN014 | | |
| 03 If more input is required and is available the next card is read in and a statement identifier is generated for this card. If a source listing is required then the statement identifier and card image are printed. If not analyzing update current statement identifier. | DFHEN015<br>DFHEN115 | | |
| 04 If not EXEC command there is no further use for the contents of the I/O buffer which are therefore formatted into card images and punched out. | DFHEN017 | | |
| 05 Continuation conventions are dealt with so that contents of card data area can be concatenated into I/O buffer to provide a contiguous data stream so that atomization can continue from where it left off. | DFHEN215 | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Tokenize Input String

Diagram 4.3A.3–01

Page 1 of 1



01 Check that EXEC CICS command has been found.

02 Check that verb is valid and locate corresponding table of adverbs.

03 Scan entire statement noting if adverb found.

04 Locate function information table.

05 Rescan request analyzing options.

06 Check options processing completed successfully.

Analyze Command

Diagram 4.3A.4–01

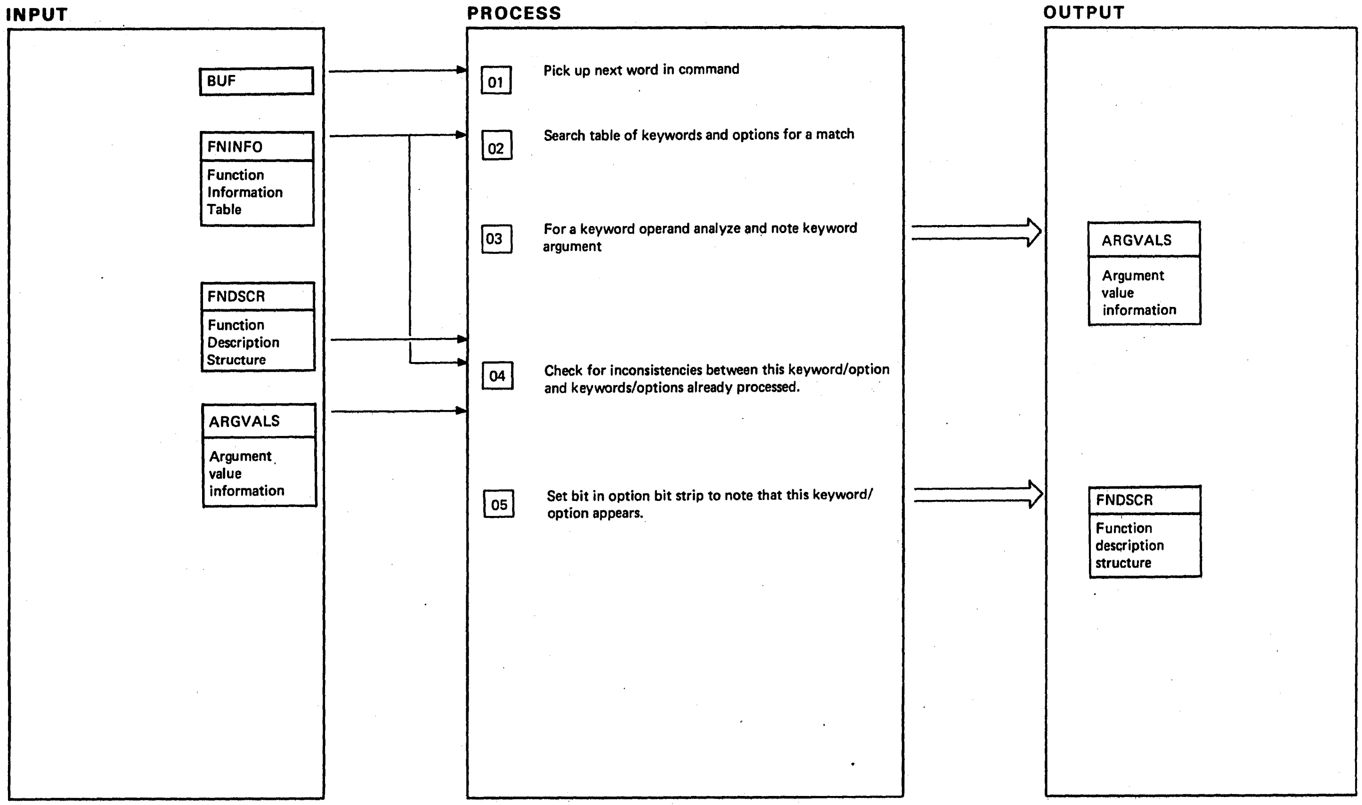
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Once EXEC CICS found routine committed to successful analysis | DFHEN019 | | |
| 02 Check verb in verb table and is supported by this version of the translator. If verb not found then attempt spelling correction. | SRCHVB | | |
| 03 The whole of the EXEC CICS command up to a correct command delimiter is scanned to ensure that whole statement is held as a contigious unmovable string in the input buffer. During the scan each option is checked against allowed adverbs list until a match is found. No spelling correction is done during this process.<br>For Assembler there is no delimiter but the whole command and only the command is already in the buffer. | PRESCAN<br>SRCHADV | | |
| 04 The verb-adverb combination uniquely defines a function information table which completely describes the allowed syntax for the command. | | | |
| 05 The EXEC CICS command is now rescanned noting options and argument values. | | | |
| 06 If options processing fails then command is deleted. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Analyze Command

Diagram 4.3A.4–01

Page 1 of 1


INPUT
BUF
FNINFO
Function Information Table
FNDSCR
Function Description Structure
ARGVALS
Argument value information
PROCESS
OUTPUT
ARGVALS
Argument value information
FNDSCR
Function description structure

01 Pick up next word in command

02 Search table of keywords and options for a match

03 For a keyword operand analyze and note keyword argument

04 Check for inconsistencies between this keyword/option and keywords/options already processed.

05 Set bit in option bit strip to note that this keyword/option appears.

Process Command Options

Diagram 4.3A.5–01

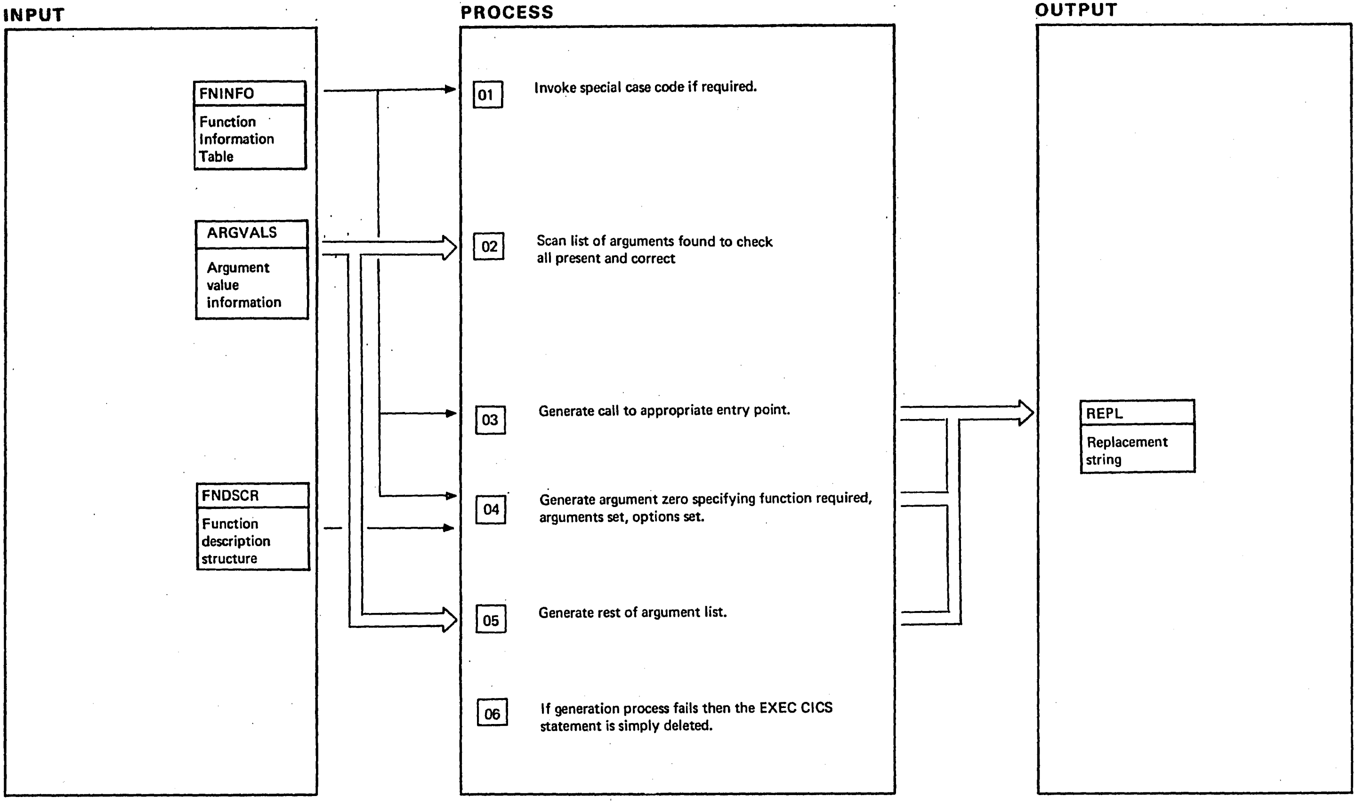
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 If next atom in text is not an identifier or statement delimiter an error has occurred. | DFHEN020 | | |
| 02 If no option matches then spelling correction is attempted.<br>If a match is found check that corresponding option or keyword is supported by this version of translator | SRCHOPT | | |
| 03 If option has associated argument Note 1) value of argument 2) syntactic type of argument 3) what keyword sets this argument | | | |
| 04 Function information table contains a list of pairs of options which are mutually exclusive. This list is scanned to see if current option keyword appears. If it does the option which is forbidden is examined and if it has already been noted as appearing then the current option is ignored. | CHKNIM | | |
| 05 For an option that sets an option bit check that the appropriate bit has not already been set to opposite value by another option that has already been processed. If it has then ignore this option. If not then set bit to appropriate value and note which option set the bit.<br>If next atom in text is not an identifier or statement delimiter a system error has occurred. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Process Command Options

Diagram 4.3A.5–01

Page 1 of 1


INPUT
FNINFO
Function Information Table
ARGVALS
Argument value information
FNDSCR
Function description structure
PROCESS
OUTPUT
REPL
Replacement string

01 Invoke special case code if required.

02 Scan list of arguments found to check all present and correct

03 Generate call to appropriate entry point.

04 Generate argument zero specifying function required, arguments set, options set.

05 Generate rest of argument list.

06 If generation process fails then the EXEC CICS statement is simply deleted.

Generate Replacement Text

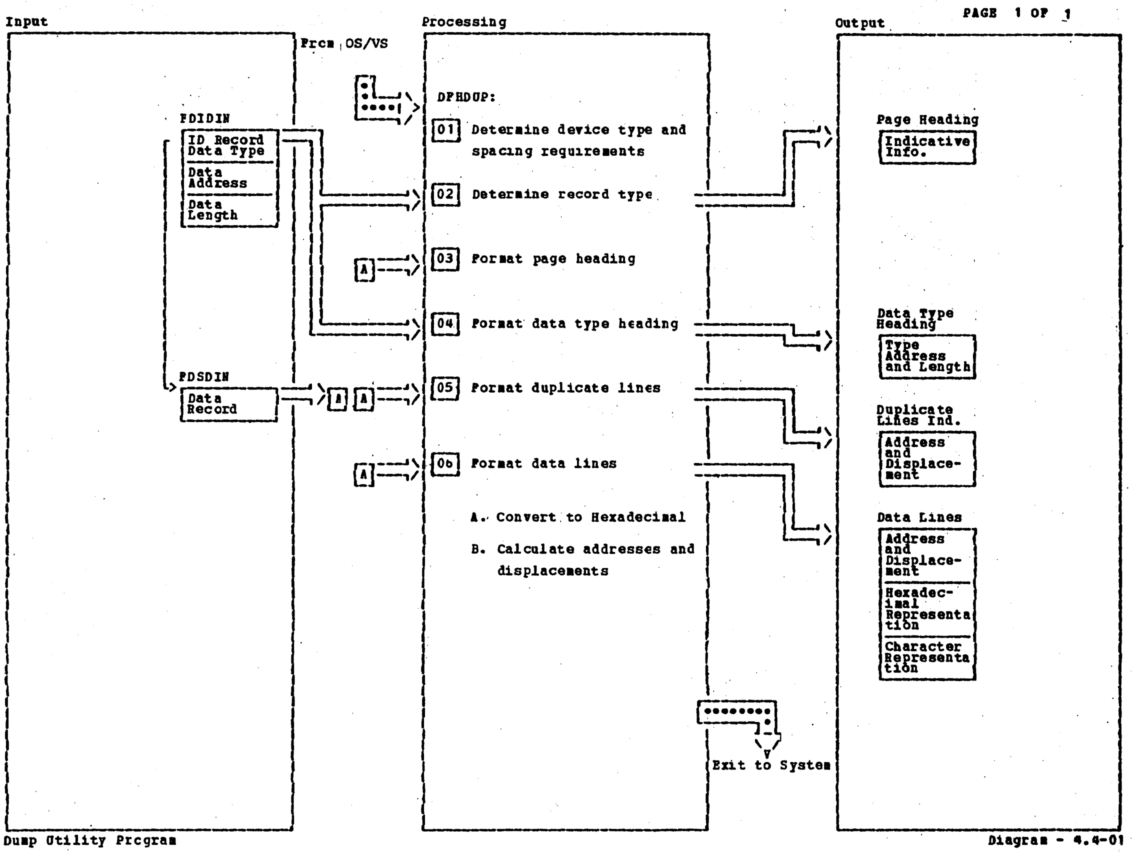
Diagram 4.3A.6–01

| | NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 01 | The function information table contains an entry point for special case code routine for this function. These entry points are all entry points to a single module which contains all the special case code | DFHEN021<br>DFHEN022 | | |
| 02 | The function information table contains an argument description list which specifies which arguments must appear in the call statement. This list is scanned in reverse order comparing with the arguments found to determine how many arguments must appear in the CALL statement.<br>The argument description list is then scaned forward;<br>A. To check that all arguments that must exist do exist<br>B. To default arguments that are optional and do not exist,<br>C. To check that mutually dependent arguments have been specified,<br>D. To give error messages for mandatory keyword arguments, that are missing. | | | |
| 03 | The generated call statement is built up in an adjustable length character string variable, beginning with 'call entry-point'. For Assembler a macro DFHEICAL is generated, with two sub-lists. The first is a list of codes which indicates what Assembler instructions should be generated by DFHEICAL for each argument. | DFHEN023 | | |

| | NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 04 | Argument zero consists of<br>1. a 2-byte encoding of the function required<br>2. a 24-bit bitstrip which specifies which arguments are genuine and which arguments are merely place holders<br>3. a bitstrip containing option settings.<br>The whole is represented by a character string argument, or for Assembler a hexadecimal string argument. | DFHEN024 | | |
| 05 | Each argument, which has already been evaluated and checked for consistency, is generated into the replacement string | | | |
| 06 | Generation may fail because;<br>1. analysis failed<br>2. inconsistency found between arguments<br>3. mandatory argument missing<br>For Assembler the replacement string is formatted and punched out. | | | |

Generate Replacement Text

Diagram 4.3A.6–01

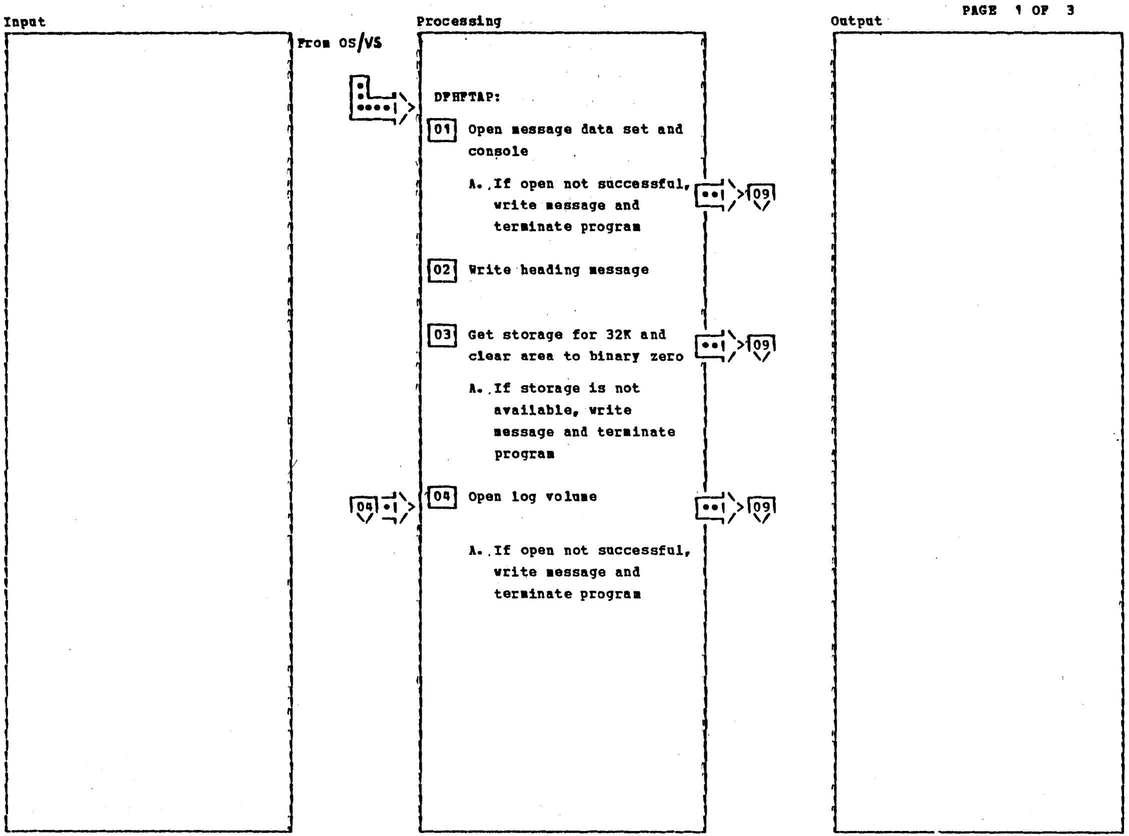


Dump Utility Program Diagram - 4.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 When DFHDUP is assembled to support more than one device type, DFHDUP sends a message to the operator console for defining the input device type. The UPSI byte is integrated to determine if single spacing is required. | | | | | | | |

Dump Utility Program Diagram - 4.4-01

PAGE 1 OF 3



Input

From OS/VS

Processing

DFHFTAP:

01 Open message data set and console

A. If open not successful, write message and terminate program

02 Write heading message

03 Get storage for 32K and clear area to binary zero

A. If storage is not available, write message and terminate program

04 Open log volume

A. If open not successful, write message and terminate program

Output

Format Tape

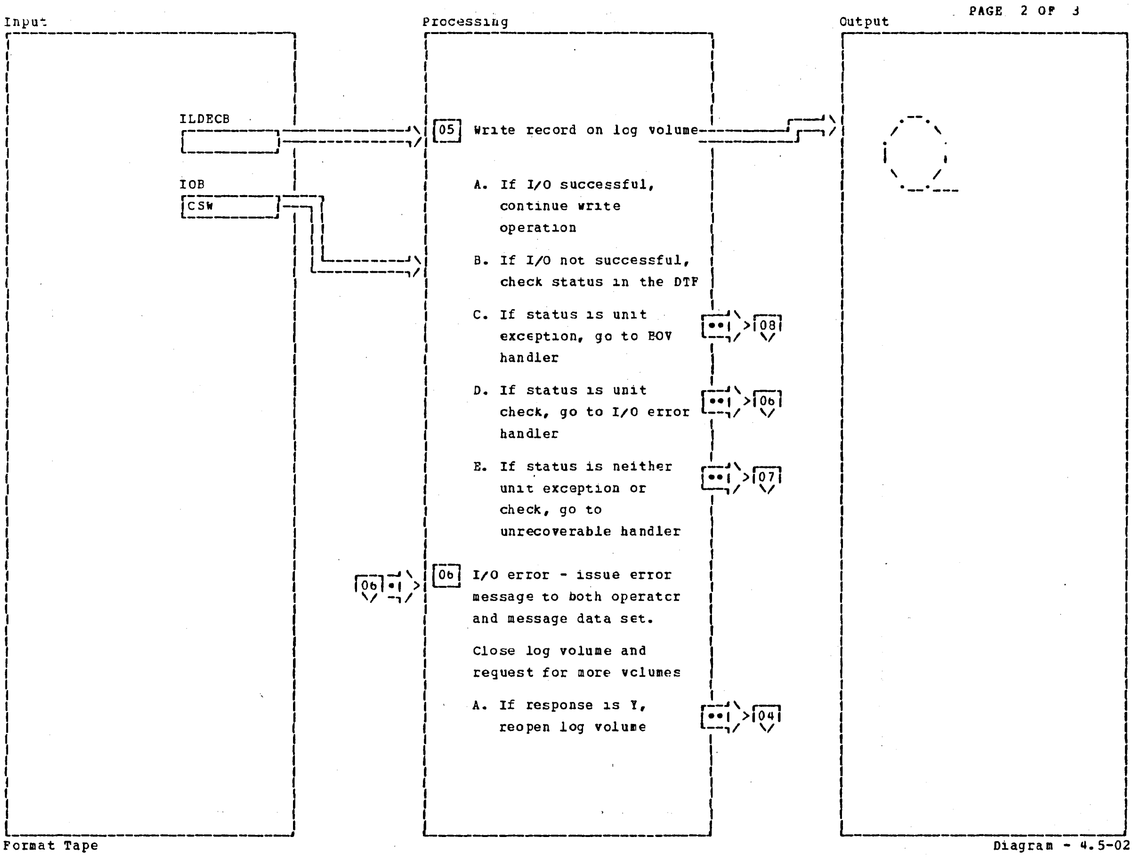
Diagram - 4.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Check DCBOFLGS for X'10' - successful open<br>A. DFH6101I unable to open message data set (programmer) | | | | | | | |
| 02 DFH6100 format tape (message data set) | | | | | | | |
| 03 Issue an OS/VS condition GETMAIN for 32K of storage. Check return code in register 15 for GETMAIN status.<br>A. DFH6102 unable to allocate working storage - program aborted (message data set) | | ILGETSTG | | | | | |
| 04 Check DCBOFLGS for X'10' - successful open.<br>A. DFH6103 unable to open log volume - program aborted (message data set) | | ILOPNLOG | | | | | |

Format Tape

Diagram - 4.5-01

PAGE 2 OF 3



05 Write record on log volume

A. If I/O successful, continue write operation

B. If I/O not successful, check status in the DTF

C. If status is unit exception, go to EOV handler

D. If status is unit check, go to I/O error handler

E. If status is neither unit exception or check, go to unrecoverable handler

06 I/O error - issue error message to both operator and message data set.

Close log volume and request for more volumes

A. If response is Y, reopen log volume

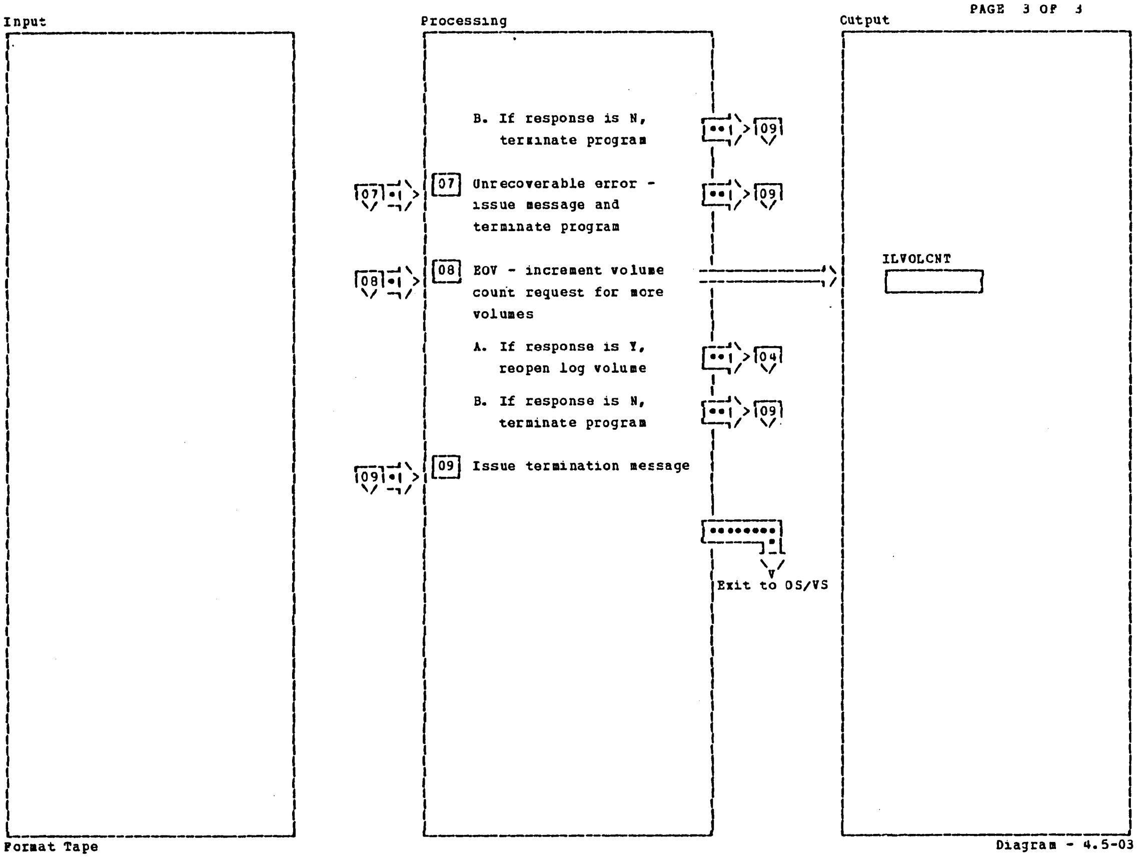
Format Tape

Diagram - 4.5-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 Write 32K binary zeros record to log data set. Check posted status in DECB (ILDECB) for a X'7F' (successful I/O). | | ILWRITE | | | | | |
| If post status is not X'7F', locate address of IOB from the DECB and check status of CSW. | | ILSYNRTN | | | | | |

Format Tape

Diagram - 4.5-02



B. If response is N, terminate program

07 Unrecoverable error - issue message and terminate program

08 EOV - increment volume count request for more volumes

A. If response is Y, reopen log volume

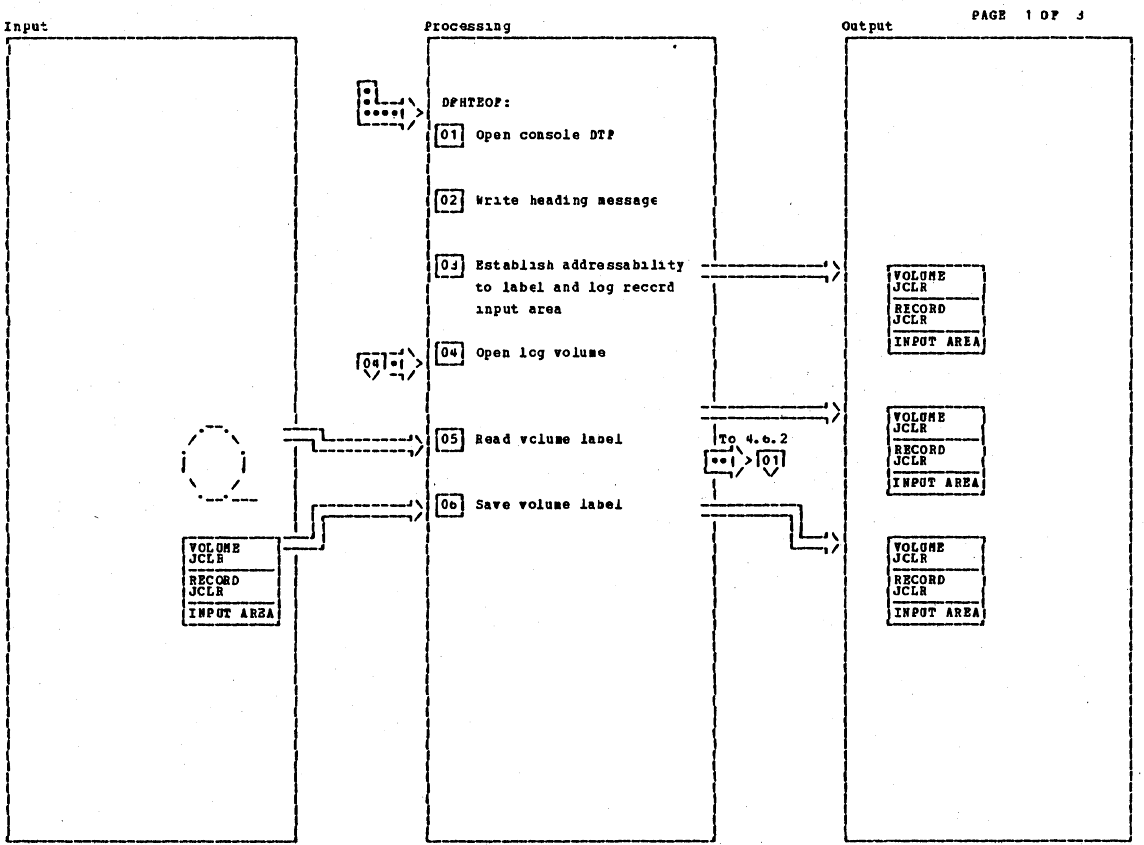
B. If response is N, terminate program

09 Issue termination message

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 06 Load sense data from the IOB and set up sense data as an index value into table ILERRTXT to insert error text into the error message.<br>DFH6104I discontinued processing on current log volume due to - XXXXXXXX - error, proceeding to next volume (operator and message data set)<br>DFH6110A more volumes to be initialized - Y or N (operator) | | | |
| 07 DFH6105 unrecoverable I/O error occurred - program aborted (message data set) | | | |
| 08 Increment field ILVOLCNT by 1. This field is used to keep a count of volumes that have been successfully initialized.<br>DFH6110A more volumes to be initialized - Y or N (operator) | | ILEOVRTN | |
| 09 Convert volume count in to termination message DFH6199. Close message and log data sets and return to OS/VS.<br>DFH6199 NNNN volume(s) initialized - log initialized completed (message data set) | | ILTERM | |

Format Tape

Diagram - 4.5-03



Tape End of File

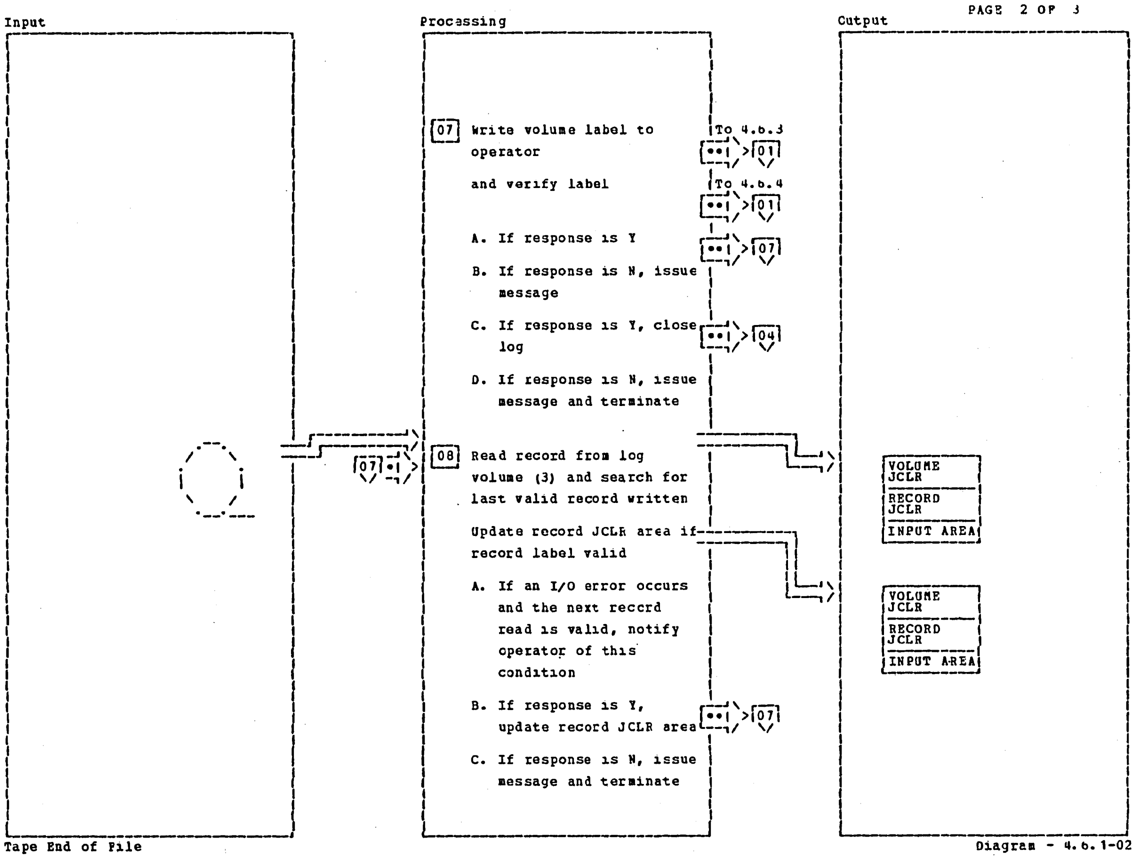
Diagram - 4.6.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 DFH2900I tape end of file program | | | | | | | |

Tape End of File

Diagram - 4.6.1-01

PAGE 2 OF 3



07 Write volume label to operator

and verify label

A. If response is Y

B. If response is N, issue message

C. If response is Y, close log

D. If response is N, issue message and terminate

08 Read record from log volume (3) and search for last valid record written

Update record JCLR area if record label valid

A. If an I/O error occurs and the next record read is valid, notify operator of this condition

B. If response is Y, update record JCLR area

C. If response is N, issue message and terminate

Tape End of File

Diagram - 4.6.1-02

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 04 DFH2906I volume label verification (operator)<br>DFH2907I label information - volume number YYDDD/NNN run HH.MM.SS block HH.MM.SS<br>DFH2908A is log volume valid - Y or N<br>B. DFH2904A swap volume Y or N (operator)<br>D. DFH2918 negative response to volume verification - program aborted | | | |
| 05 Read log record and verify the record JCL to the volume JCL in the following sequence:<br>- Volume creation date (JCLRVCD)<br>- Volume sequence (JCLRVSN)<br>- Run start time (JCLRRST) | | FTSCHEOF | |
| If the fields are valid, update the record JCLR area with the current label.<br>An I/O error is indicated by testing the FTBSPSW field for a X'02'. If the test proves to be positive and the comparison between the volume label and the next rec label after the I/O error had occurred is valid, a valid I/O error has occurred and the information is presented to the operator and the FTBSPSW reset to indicate no error occurred.<br>A. DFH2905I I/O error on previous rec - next record label valid<br>DFH2907I label information - ...... ......<br>DFH2909A continue processing - Y or N<br>C. DFH2910I I/O error occurred, program terminated | | FTMOVBLK | |

Tape End of File

Diagram - 4.6.1-02

PAGE 3 OF 3

Input

Processing

[09] Verify last valid record label (FTCONV - 4.6.3) with operator (FTWTOR - 4.6.4)

A. If response is Y → [10]

B. If response is N, issue message and exit

[10] → [10] Backspace to record following the last valid record and close the log volume

[11] Terminate program

Output

Tape End of File

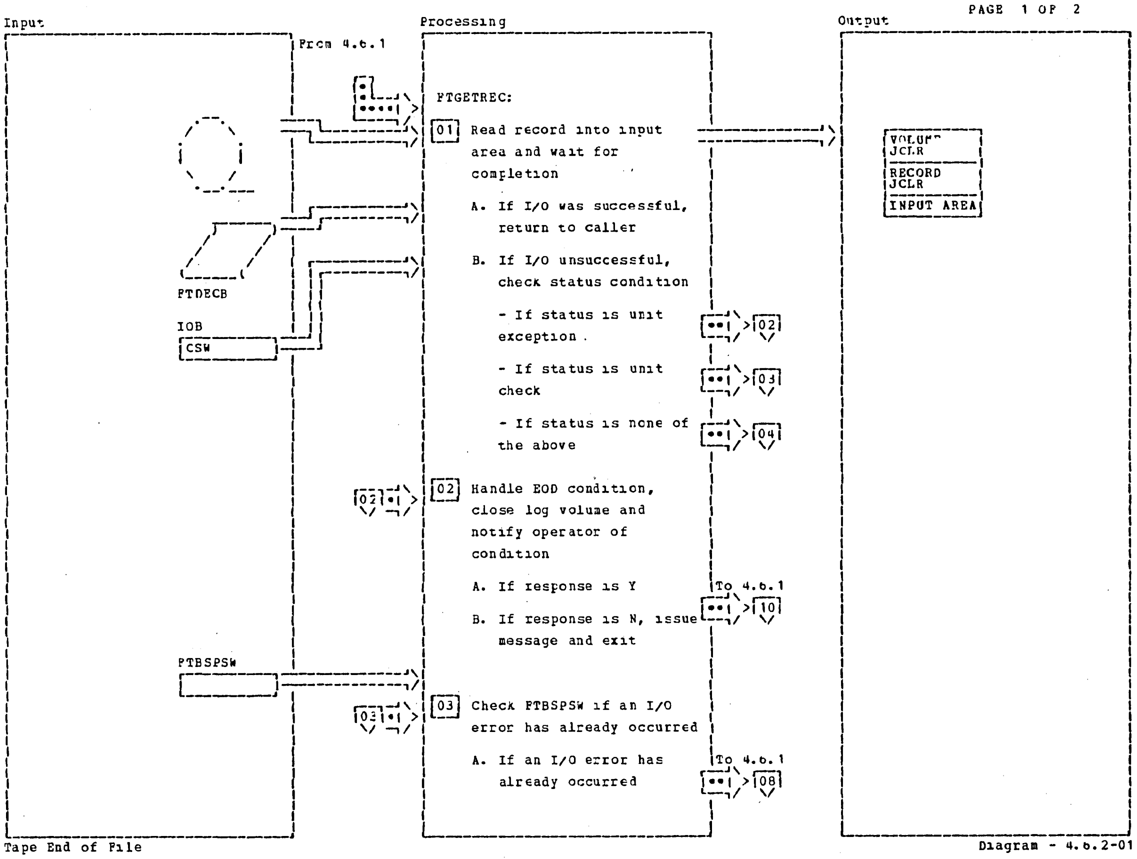
Diagram - 4.6.1-03

| Notes | Routine | Label | Ref |
|---|---|---|---|
| [06] A record is considered the last valid record when:<br>- two consecutive I/O errors occur<br>- unequal compare of the label fields between the record JCL and the volume JCL | | | |
| The record label is then written to the cperator for verification. | | FTBLKVER | |
| DFH291CI record label verificaticn<br>DFH2907I label information - ........ etc.<br>DFH2912A is record label valid - Y or N<br>B. DFH2913I neg rsp to record label verification - program aborted | | | |
| [07] The amount of records to be backspaced is kept in the field FTBSPSW. | | FTBSPREC | |
| [08] DFH2901I successful completion - DFHTEOF | | | |

Tape End of File

Diagram - 4.6.1-03

PAGE 1 OF 2



FTGETREC:

01 Read record into input area and wait for completion

A. If I/O was successful, return to caller

B. If I/O unsuccessful, check status condition

- If status is unit exception .
- If status is unit check
- If status is none of the above

02 Handle EOD condition, close log volume and notify operator of condition

A. If response is Y

B. If response is N, issue message and exit

03 Check FTBSPSW if an I/O error has already occurred

A. If an I/O error has already occurred

Tape End of File

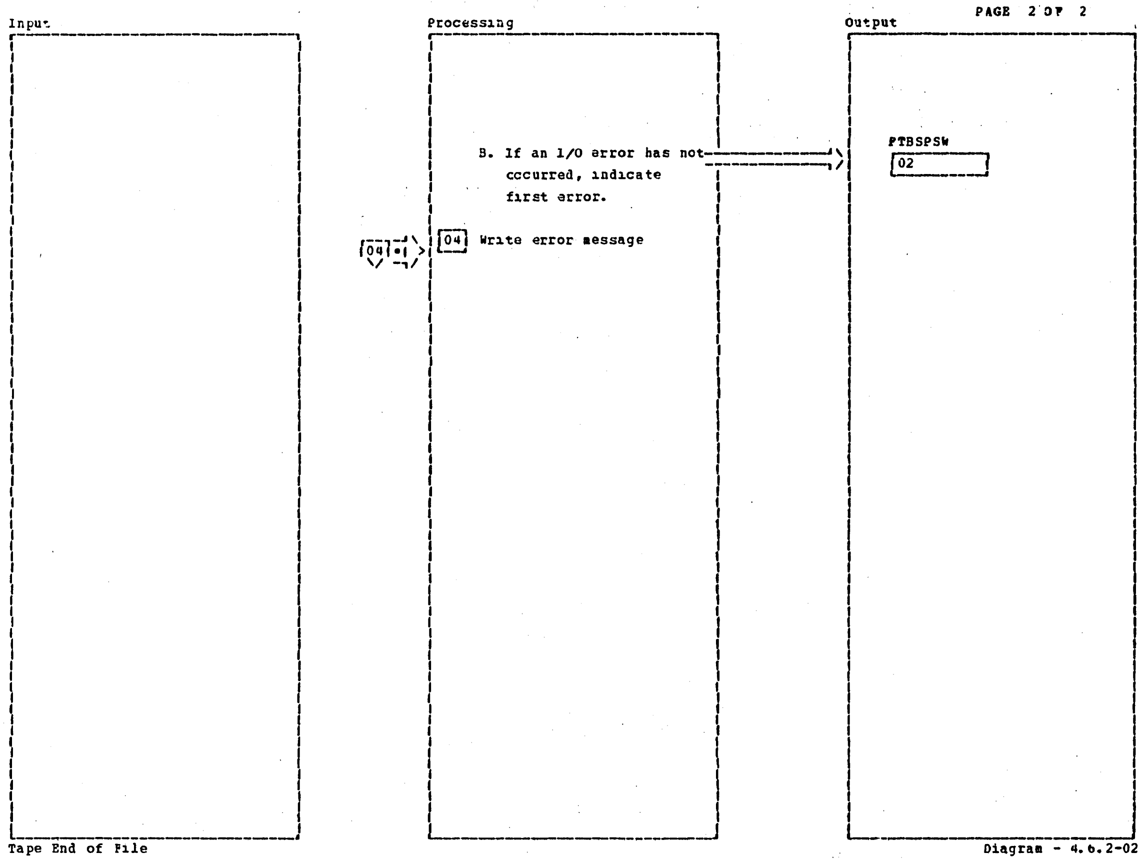
Diagram - 4.6.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Check error status in the DTF for type of error occurred. | | FTSYNAD | |
| 02 If error occurs during volume label processing issue:<br>DFH2919I EOD - no volume label / invalid volume mounted<br>DFH2914I EOD occurred - last record label verification<br>DFH2907I label information ...<br>DFH2915A is proper volume mounted - Y or N<br>B. DFH2920I EOD occurred - request program terminated | | FTEODAD | |
| 03 Check FTBSPSW for a previous I/O error. If an error has previously occurred, go to verify last valid record with operator. If an error has not previously occurred, set up error condition in message DFH2905I and indicate error in FTBSPSW. Then read next record on log volume. | | | |

Tape End of File

Diagram - 4.6.2-01

PAGE 2 OF 2



Input

Processing

Output

8. If an I/O error has not cccurred, indicate first error.

04 Write error message

FTBSPSW

02

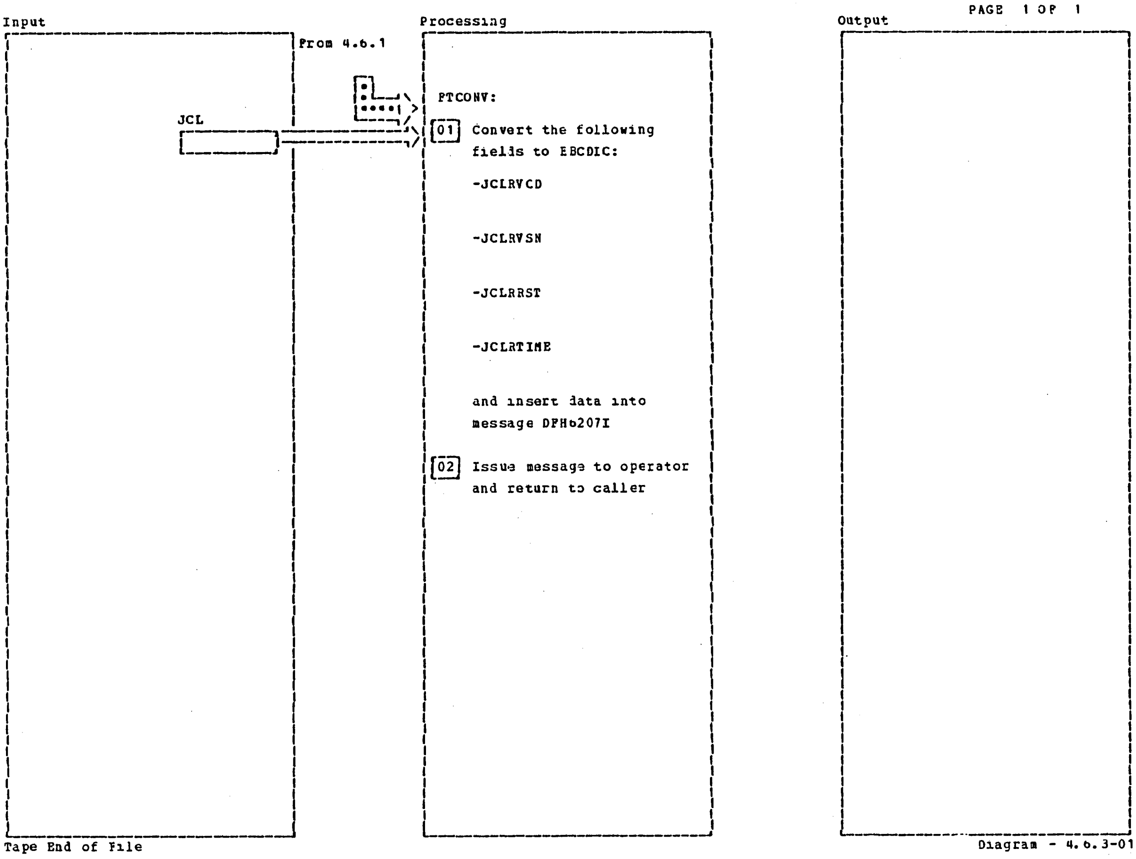
Tape End of File

Diagram - 4.6.2-02

| Notes | Rottine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 DFH2916I unrecoverable I/O error occurred - program terminated | | | | | | | |

Tape End of File

Diagram - 4.6.2-02



PAGE 1 OF 1

Input

From 4.6.1

JCL

Processing

FTCONV:

01 Convert the following fields to EBCDIC:

-JCLRVCD

-JCLRVSN

-JCLRRST

-JCLRTIME

and insert data into message DFH6207I

02 Issue message to operator and return to caller

Output

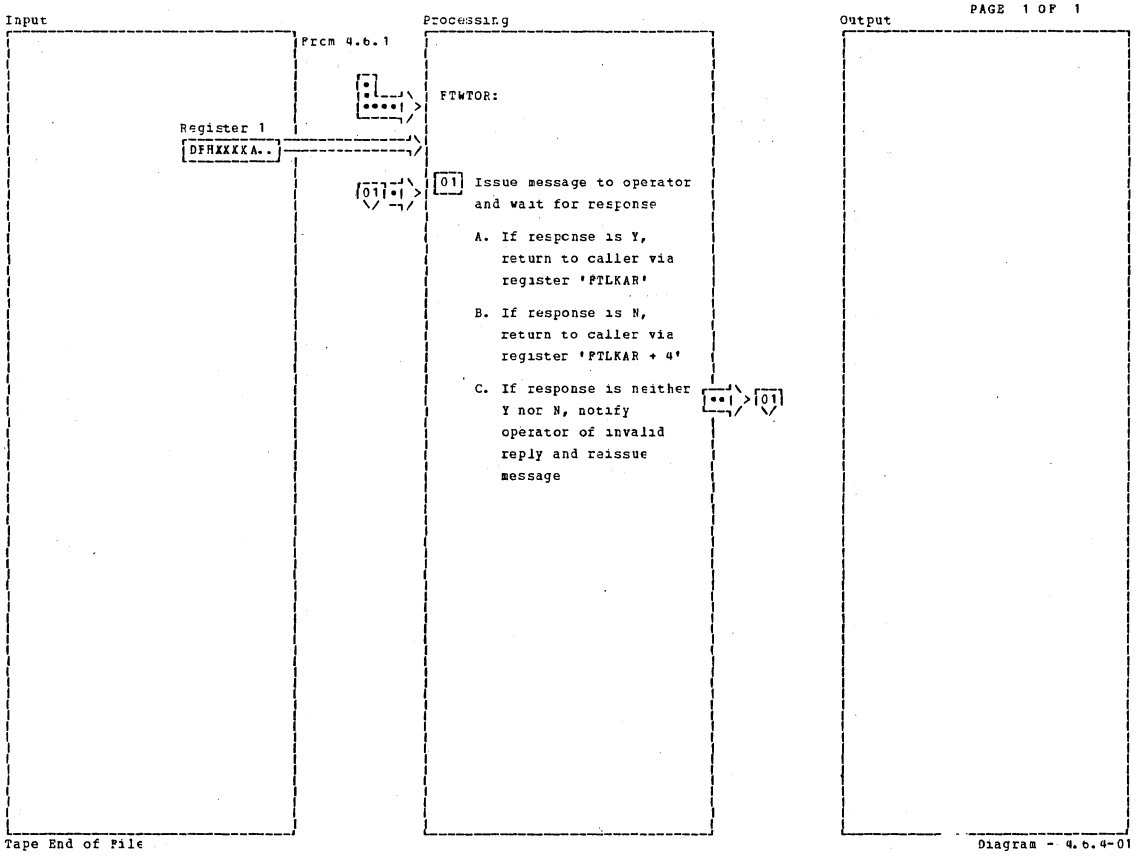
Tape End of File

Diagram - 4.6.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Convert packed decimal to printable characters and insert values into message.<br>E. DFH2907I label information - volume number YYDDD/NNN run HH.MM.SS block HH.MM.SS | DFHTEOF | FTCONV | | | | | |
| 02 See Note 1E. | | | | | | | |

Tape End of File

Diagram - 4.6.3-01



FTWTOR:

01 Issue message to operator and wait for response

A. If response is Y, return to caller via register 'FTLKAR'

B. If response is N, return to caller via register 'FTLKAR + 4'

C. If response is neither Y nor N, notify operator of invalid reply and reissue message

Tape End of File

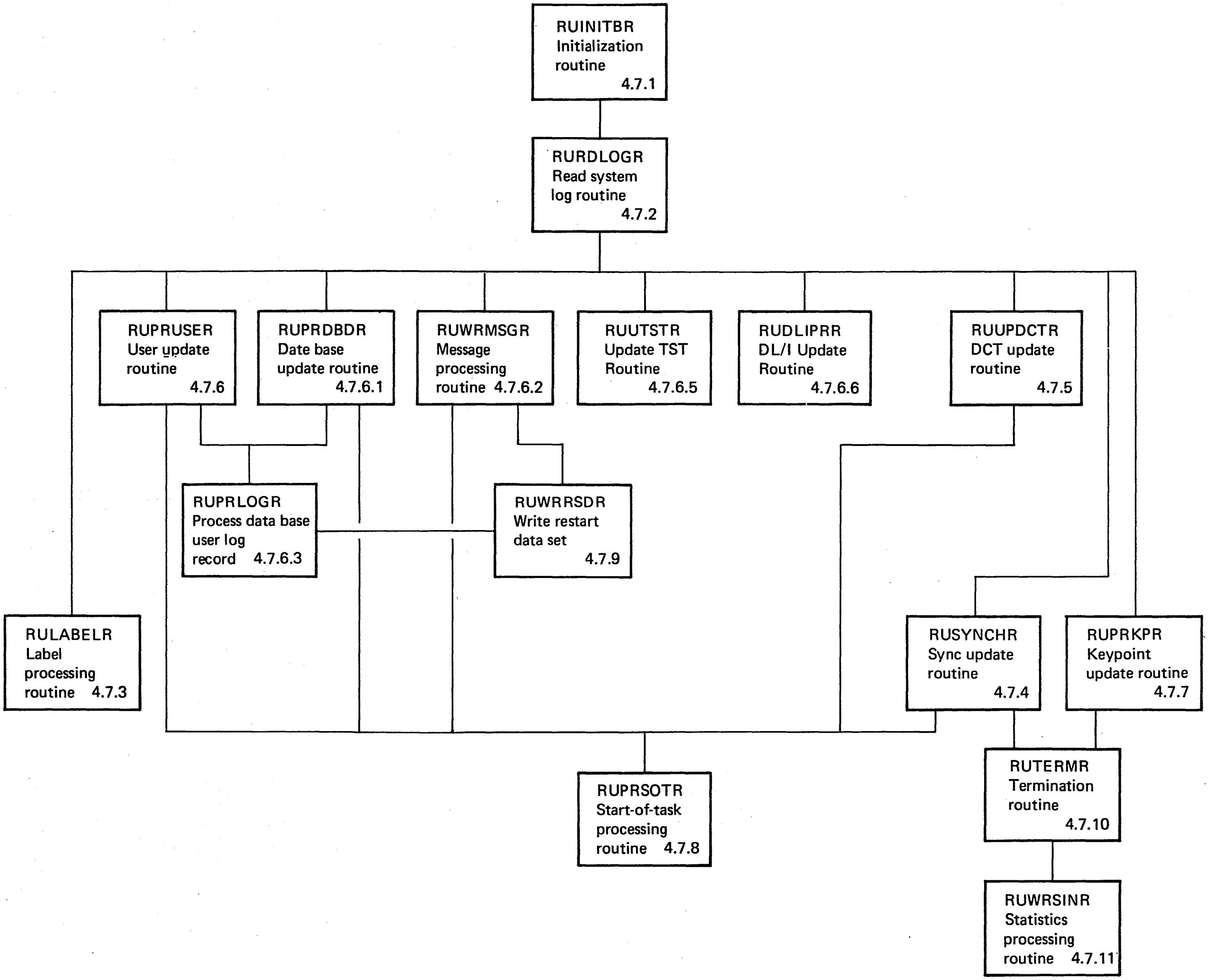
Diagram - 4.6.4-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Anticipate invalid response frcm operator: therefore save address of caller message area passed to this routine. | | FTWTOR | |
| Issue message to operator and wait for response. | | FTIWTOR | |
| If response is neither Y nor N, insert invalid response character in message DFH2917I, issue message to operator, restore caller's message address, and reissue message. | | | |
| C. DFH2917I incorrect reply - ? | | | |

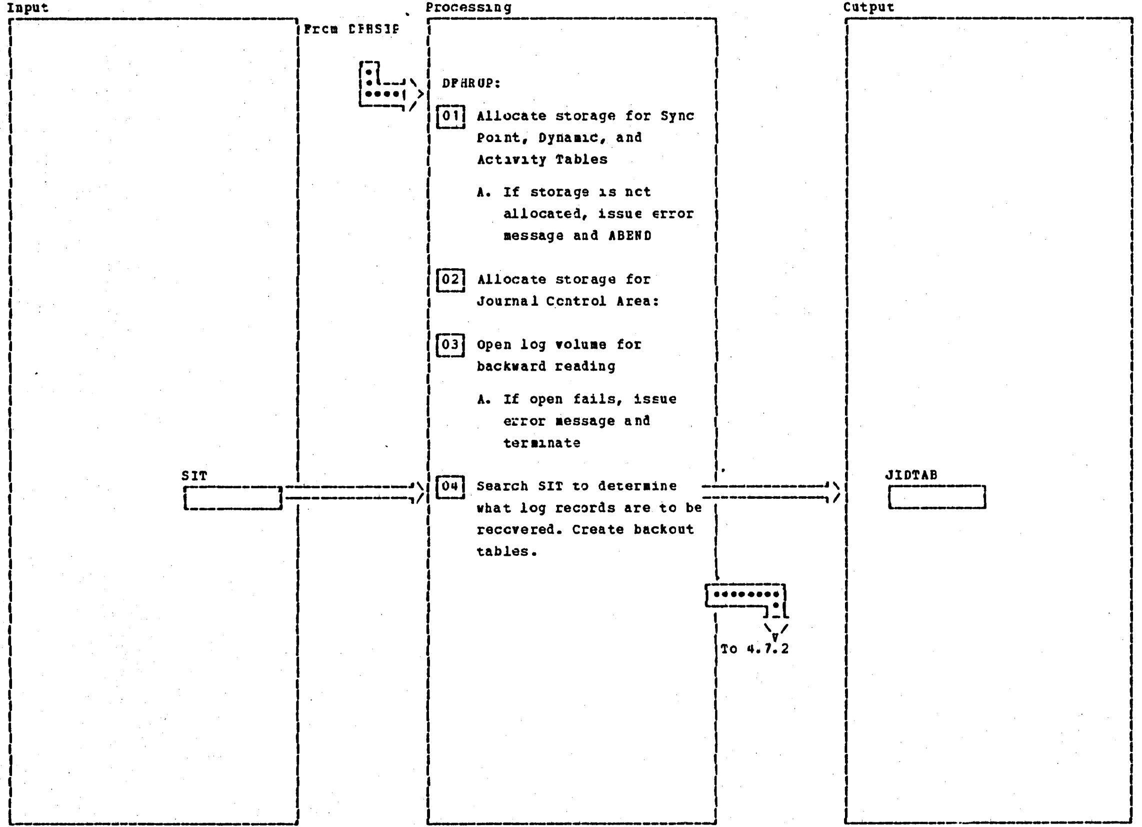
| Notes | Routine | Label | Ref |
|---|---|---|---|
| | | | |

Tape End of File

Diagram - 4.6.4-01


RUINITBR
Initialization routine
4.7.1
RURDLOGR
Read system log routine
4.7.2
RUPRUSER
User update routine
4.7.6
RUPRDBDR
Date base update routine
4.7.6.1
RUWRMSGR
Message processing routine 4.7.6.2
RUUTSTR
Update TST Routine
4.7.6.5
RUDLIPRR
DL/I Update Routine
4.7.6.6
RUUPDCTR
DCT update routine
4.7.5
RUPRLOGR
Process data base user log record 4.7.6.3
RUWRRSDR
Write restart data set
4.7.9
RULABELR
Label processing routine 4.7.3
RUSYNCHR
Sync update routine
4.7.4
RUPRKPR
Keypoint update routine
4.7.7
RUPRSOTR
Start-of-task processing routine 4.7.8
RUTERMR
Termination routine
4.7.10
RUWRSINR
Statistics processing routine 4.7.11

PAGE 1 OF 1


Input

Processing

Cutput

From DFHSIP

DFHRUP:

01 Allocate storage for Sync Point, Dynamic, and Activity Tables

A. If storage is nct allocated, issue error message and ABEND

02 Allocate storage for Journal Control Area:

03 Open log volume for backward reading

A. If open fails, issue error message and terminate

04 Search SIT to determine what log records are to be recovered. Create backout tables.

RUINITBR - Initialization Routine

Diagram - 4.7.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Allocate 2K of storage for each table. Initialize the first 4 words of the table with the following pointers:<br>Word 1 - pointer to next block of entries<br>Word 2 - pointer to beginning of block<br>Word 3 - pointer to available slot in the block<br>Word 4 - pointer to last entry in block<br>A. DFH2804I... ABEND code 2804 | | RUINITBR | |
| 03<br>A. DFH2803I... ABEND code 2803 | | | |
| 04 Scan option fields for X'04', in the SIT, indicating that recovery should be performed.<br>A. If recovery is not to be performed, zero the indexing factor in the JIDTAB table to prevent branching to the particular update processor. This causes branching to the user's update routine RUPROSER.<br>B. If recovery is to be performed, allocate 2K bytes of storage for backout tables. | | | |

RUINITBR - Initialization Routine

Diagram - 4.7.1-01

PAGE 1 OF 1

Input

Processing

Output

From 4.7.1

DFHRUP:

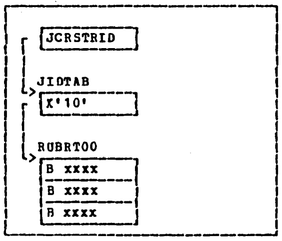
[01] Issue DFHJC macro. TYPE=GETB to read log records

A. If an I/O error occurs, issue message and terminate

B. If end-of-file occurs, close volume and open for previous volume

C. If any other errors occur, issue message and terminate

[02] The system ID in field JCRSTRID of the prefix portion of the log record is used as an index to branch to the appropriate update routine. See chart 4.7.



RURDLOGR - Read Log Routine

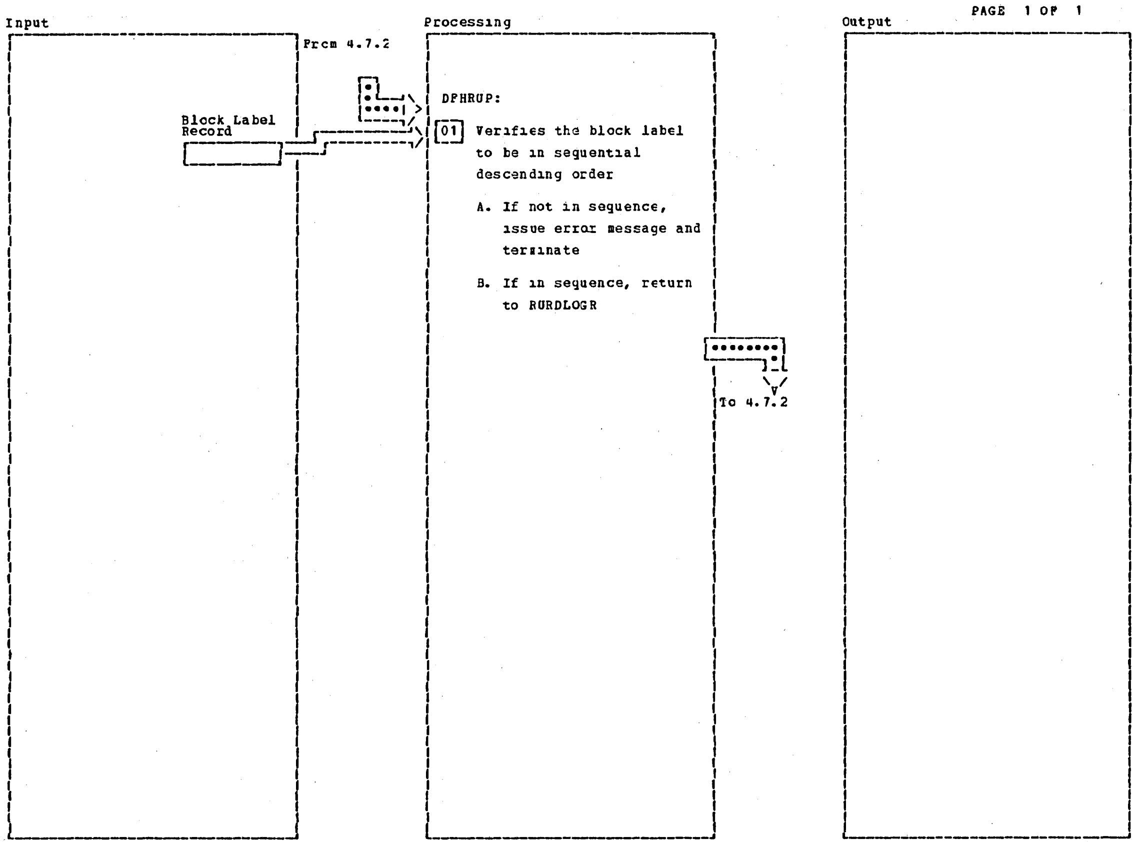
Diagram - 4.7.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| [01] | | RURDLOGR | |
| A. DFH2801I... ABEND code 2801 | | | |
| C. DFH2805I... ABEND code 2805 | | | |
| [02] DSECT DFHJCR is used to overlay the log record. At location JCRSTRID + 1 is the function ID that is utilized as an index value to retrieve another index value that is used to execute the appropriate branch instruction to the update processor. | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

RURDLOGR - Read Log Routine

Diagram - 4.7.2-01



01 Verifies the block label to be in sequential descending order

A. If not in sequence, issue error message and terminate

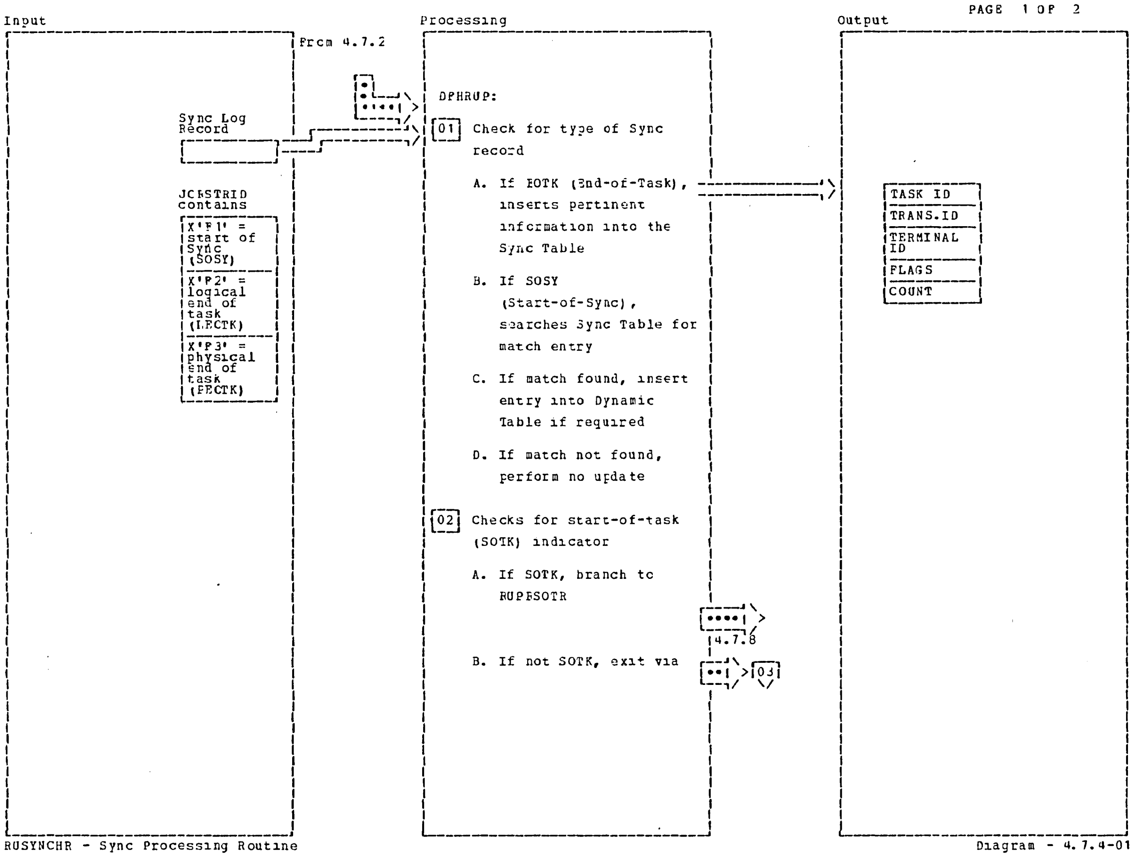
B. If in sequence, return to RURDLOGR

RULABELR - Label VerificationRoutine

Diagram - 4.7.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The first block label data is saved in fields: | | RULABELR | | | | | |
| - RURUNTME - run start time | | | | | | | |
| - RUDATE - date of run | | | | | | | |
| - RUBLKTME - time block record was written | | | | | | | |
| Thereof all other block label data is compared against this field. | | RULABE30 | | | | | |
| A. DFH2802I... ABEND code 2802 | | | | | | | |

PAGE 1 OF 2



DPHRUP:

01 Check for type of Sync record

A. If EOTK (End-of-Task), inserts pertinent information into the Sync Table

B. If SOSY (Start-of-Sync), searches Sync Table for match entry

C. If match found, insert entry into Dynamic Table if required

D. If match not found, perform no update

02 Checks for start-of-task (SOTK) indicator

A. If SOTK, branch to RUPESOTR

B. If not SOTK, exit via

RUSYNCHR - Sync Processing Routine

Diagram - 4.7.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 When a SOSY record is found the Dynamic Table is scanned for a matching entry. This search is by task ID. The entry is inserted into the Dynamic Table if an entry is nct found in the Dynamic Table with the same task ID. |  | RUSYNC30 |  |  |  |  |  |
| 02 SOTK is determine by checking the prefix portion of the record in the field JCSPF1 for a X'02' or X'04'. |  | RUSYNC80 |  |  |  |  |  |

PAGE 2 OF 2

Input

Processing

Output

03 • > 03 Checks for keypoint processed indication

A. If processed, branch to RUTERMR

•••• > 4.7.10

B. If not processed, return to RURDLOGR

•••• > 4.7.10

RUSYNCHR - Sync Processirg Routine

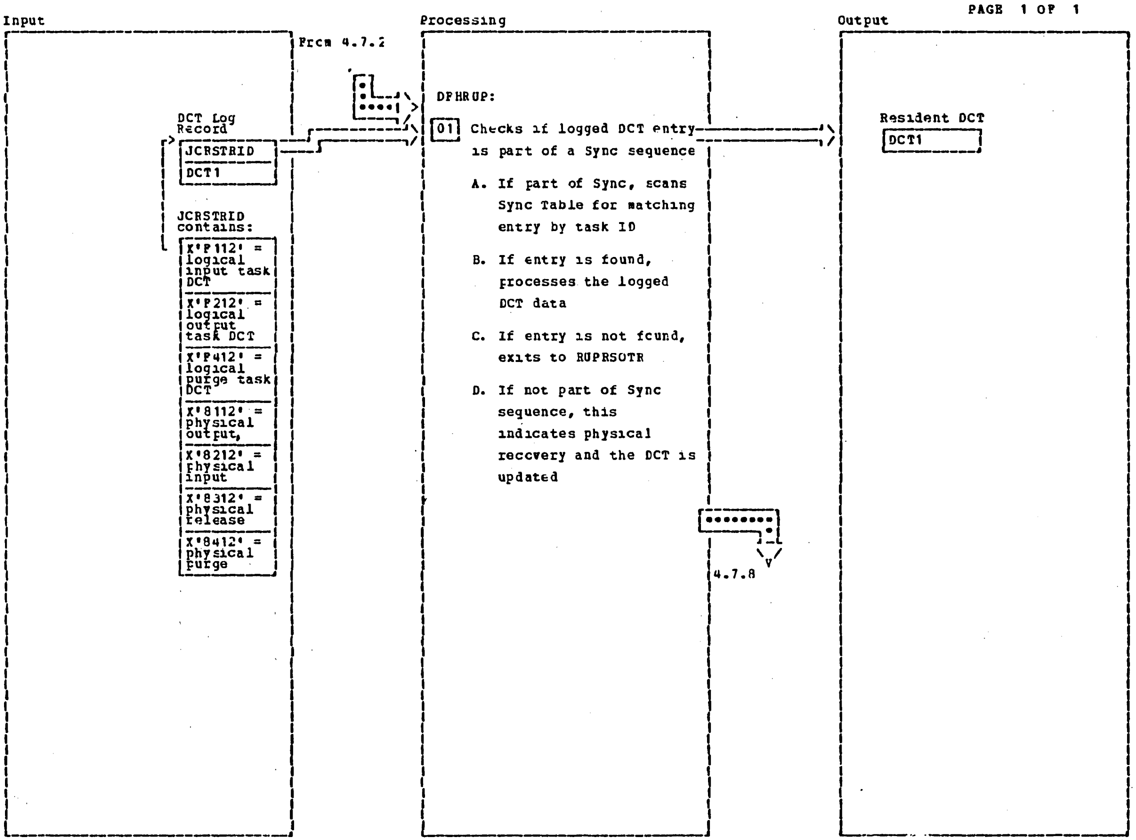
Diagram - 4.7.4-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 Keypoint processed is determined by check 'RUSTATUS' byte for a X'02'. | | | | | | | |

RUSYNCHR - Sync Frocessirg Routine

Diagram - 4.7.4-02

PAGE 1 OF 1


Input
Processing
Output
From 4.7.2
DCT Log Record
JCRSTRID
DCT1
JCRSTRID contains:
X'F112' = logical input task DCT
X'F212' = logical output task DCT
X'F412' = logical purge task DCT
X'8112' = physical output,
X'8212' = physical input
X'8312' = physical release
X'8412' = physical purge
Resident DCT
DCT1
4.7.8

DFHRUP:

01 Checks if logged DCT entry is part of a Sync sequence

A. If part of Sync, scans Sync Table for matching entry by task ID

B. If entry is found, processes the logged DCT data

C. If entry is not found, exits to RUPRSOTR

D. If not part of Sync sequence, this indicates physical recovery and the DCT is updated

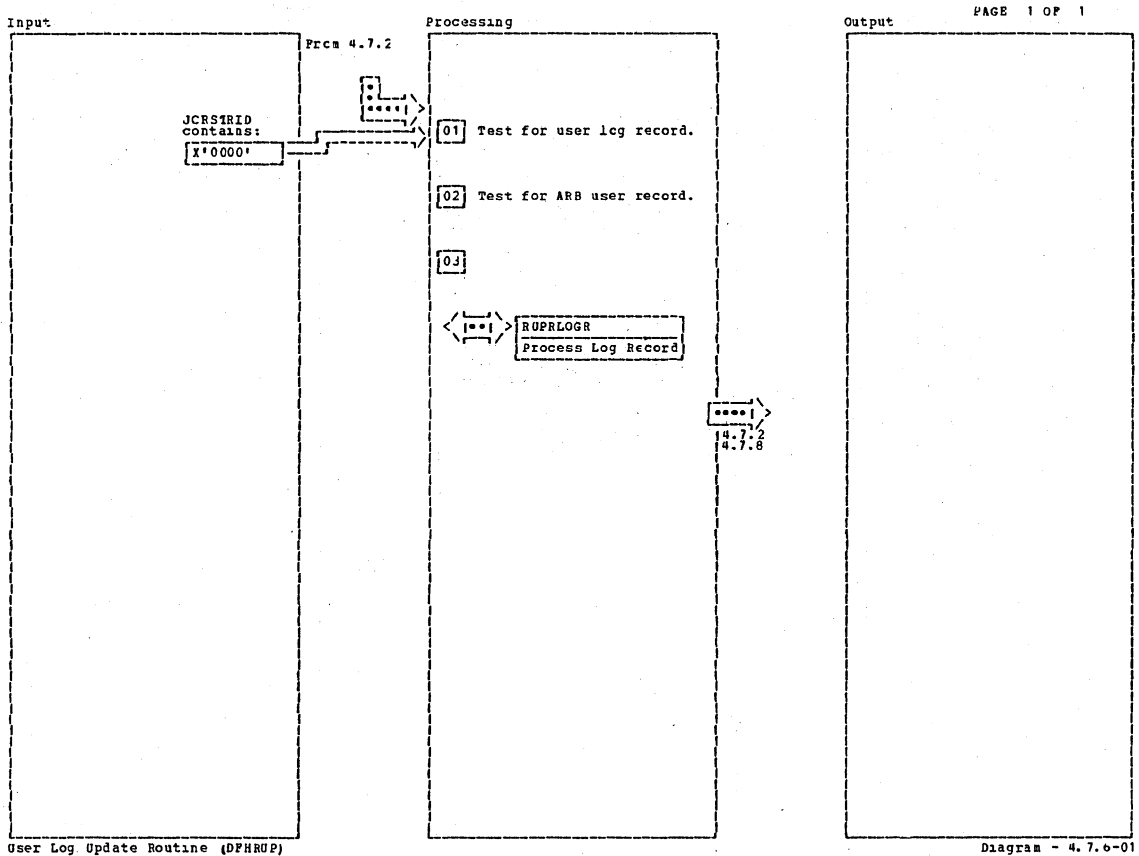
RUUPDCTR - Update DCT Entry Routine

Diagram - 4.7.5-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 If the logged DCT is part of the Sync sequence and a matching task ID is found in the Sync table, the resident DCT table is search for the matching destination ID. Depending on the type of record being processed, the appropriate TTRs are moved from the logged record to the resident DCT entry. | | RUUPDCTR | |
| For logical input task, the physical input pointer and the logical output pointer are retrieved and set in the resident DCT. | | RUUPD100 | |
| For logical output task, the physical output pointer and the logical input pointer are set into the resident DCT. | | RUPD200 | |
| For physical recovery, the input pointer is recovered and queue start is determined by queue type. | | RUUPD600 | |
| After the entry has been completely processed, a X'80' is set in the input pointer of the DCT entry to prevent any further updates to the entry. | | | |

RUUPDCTR - Update DCT Entry Routine

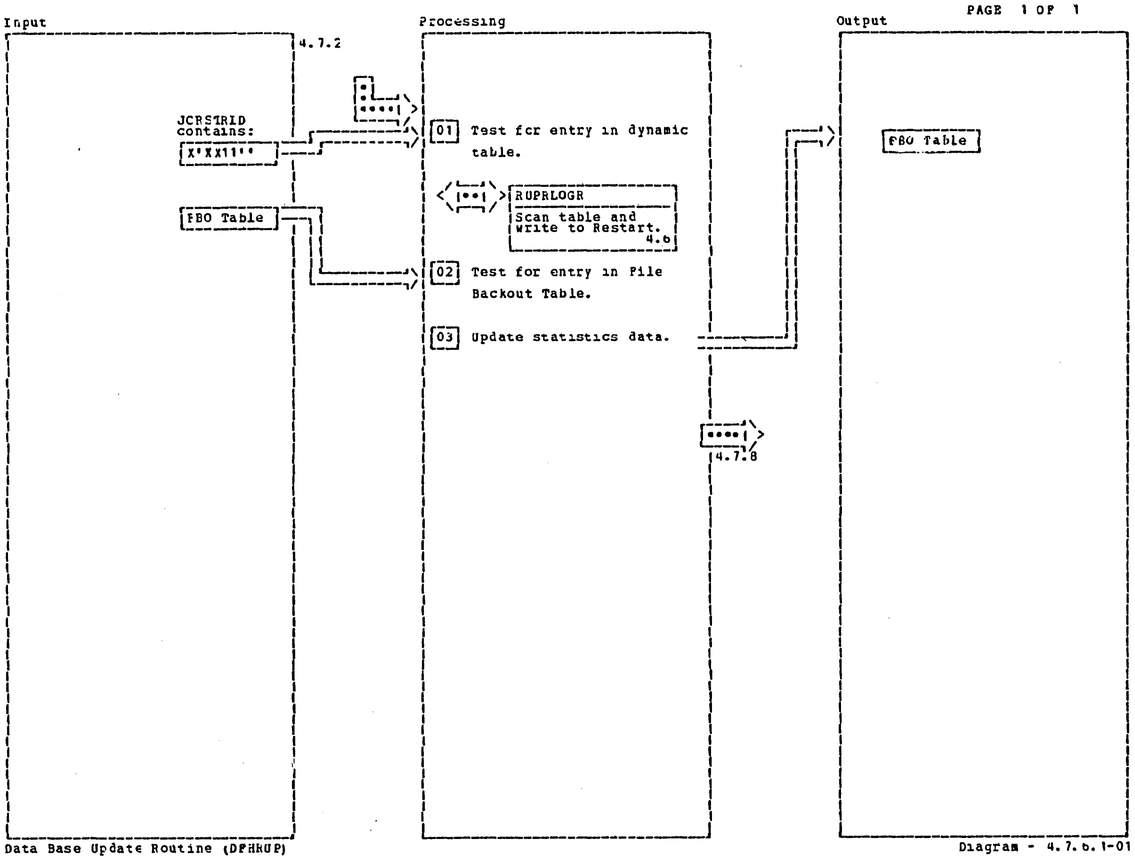
Diagram - 4.7.5-01



User Log Update Routine (DFHRUP)

Diagram - 4.7.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The test for user log record is by testing the JCRSTRID+1 location for X'00'. If the field is non-zero, this log record is not a user log record, but one marked as not to be recovered (determined earlier in RUINITER). | | | | | | | |
| 02 Test for user ARP record is CSKP in JCSPTRAN field or TCP in JCSPTASK of journal prefix. If any of these conditions are met and the end KP record has been processed but the KP previously flag is not set the record is written to the Restart Data Set. | | | | | | | |
| 03 BAL to RUPRLOGR routine to process user log record. | | | | | | | |


PAGE 1 OF 1
Input
Processing
Output
4.7.2
JCRS1RID contains:
X'XX11''
FBO Table
01 Test fcr entry in dynamic table.
RUPRLOGR
Scan table and write to Restart.
4.6
02 Test for entry in File Backout Table.
03 Update statistics data.
4.7.8
FBO Table

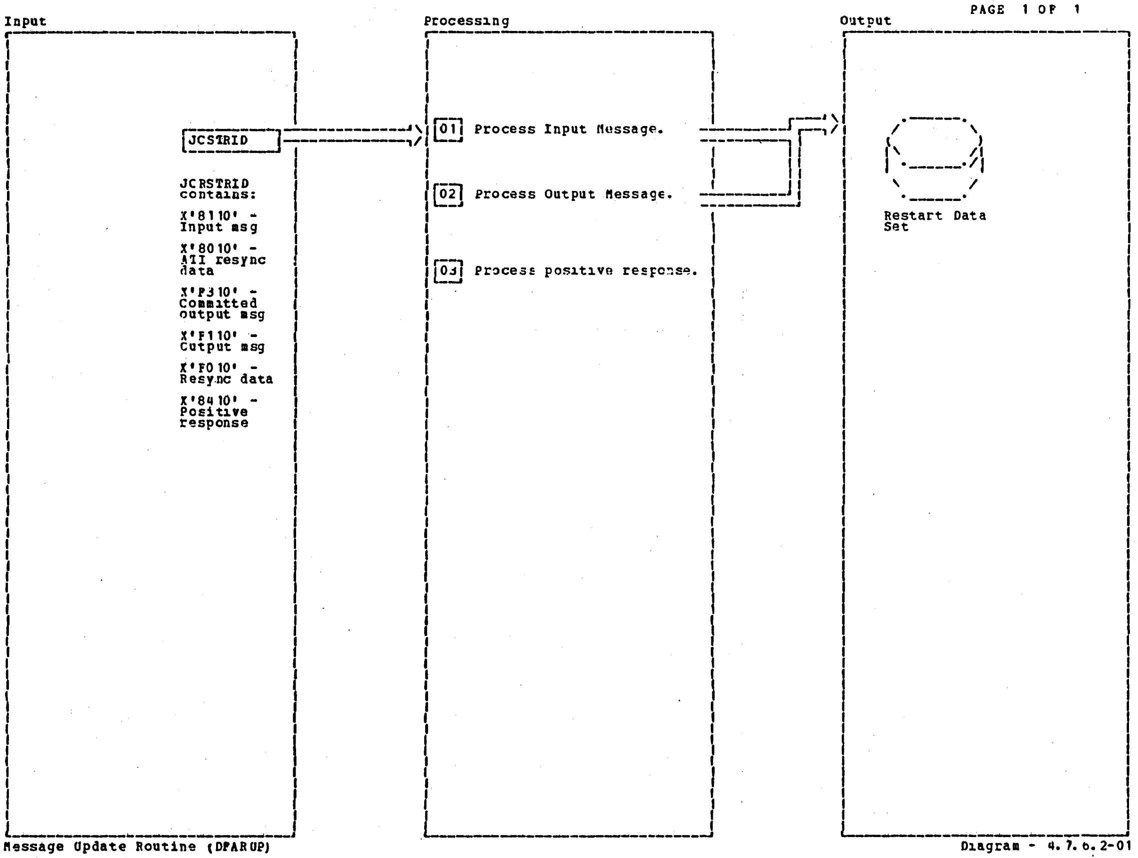
Data Base Update Routine (DFHRUP)

Diagram - 4.7.6.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 BAL to RUPRLOGF routine to scan for matching entry in Dynamic Table. Check for return code from RUPRLOGR. | | | | | | | |
| 02 If return code is non-zero, scan File Backout Table (FBO) for match. If no entry is found, create an entry in FBO table. | | | | | | | |
| 03 Update counters by record type (i.e., write, add, read, update, etc.). | | | | | | | |

Data Base Update Routine (DFHRUP)

Diagram - 4.7.6.1-01


PAGE 1 OF 1
Input
Processing
Output
JCSTRID
JCRSTRID contains:
X'8110' - Input msg
X'8010' - ATI resync data
X'F310' - Committed output msg
X'F110' - Output msg
X'F010' - Resync data
X'8410' - Positive response
01 Process Input Message.
02 Process Output Message.
03 Process positive response.
Restart Data Set

Message Update Routine (DFARUP)

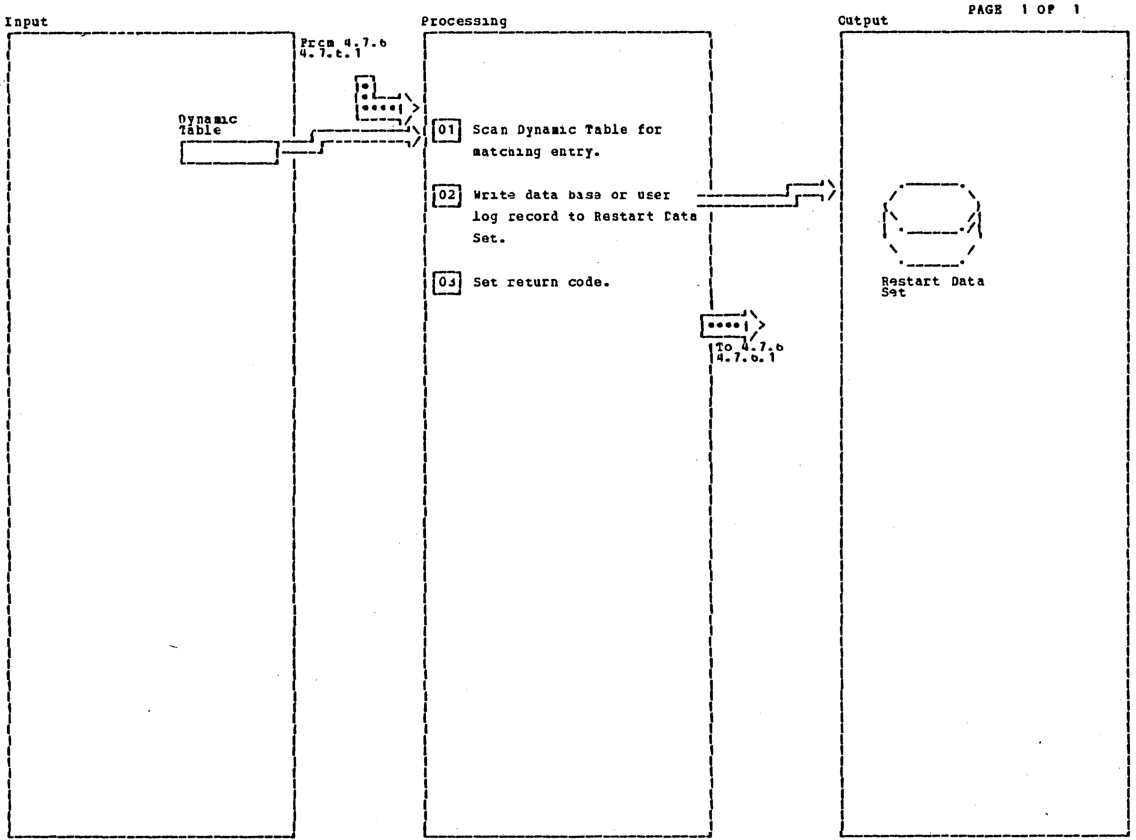
Diagram - 4.7.6.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Scan Dynamic Table for matching entry. | | RUWRMSGR | |
| If not found, write message to restart data set. | | | |
| Scan MBC Table for matching terminal entry and insert pertinent data. | | | |
| 02 Scan Sync Table for matching entry. | | RUOPMSGR | |
| If entry found, scan MBO table and insert pertinent information. | | | |
| If entry not found, exit. | | | |
| If not REOTK indicator, write to Restart data set. | | | |
| 03 Scan MBC table for matching terminal entry. | | RUORSPR | |
| If entry found, test for positive response bit. | | | |
| If set, exit. | | | |
| If not set, set positive response and reset awaiting indicator. | | | |
| If entry not found, execute entry in MBO Table and set awaiting indicator. | | | |

Message Update Routine (DFARUP)

Diagram - 4.7.6.2-01

PAGE 1 OF 1


Input
Processing
Output
From 4.7.6
4.7.6.1
Dynamic
Table
01 Scan Dynamic Table for matching entry.
02 Write data base or user log record to Restart Data Set.
03 Set return code.
To 4.7.6
4.7.6.1
Restart Data
Set

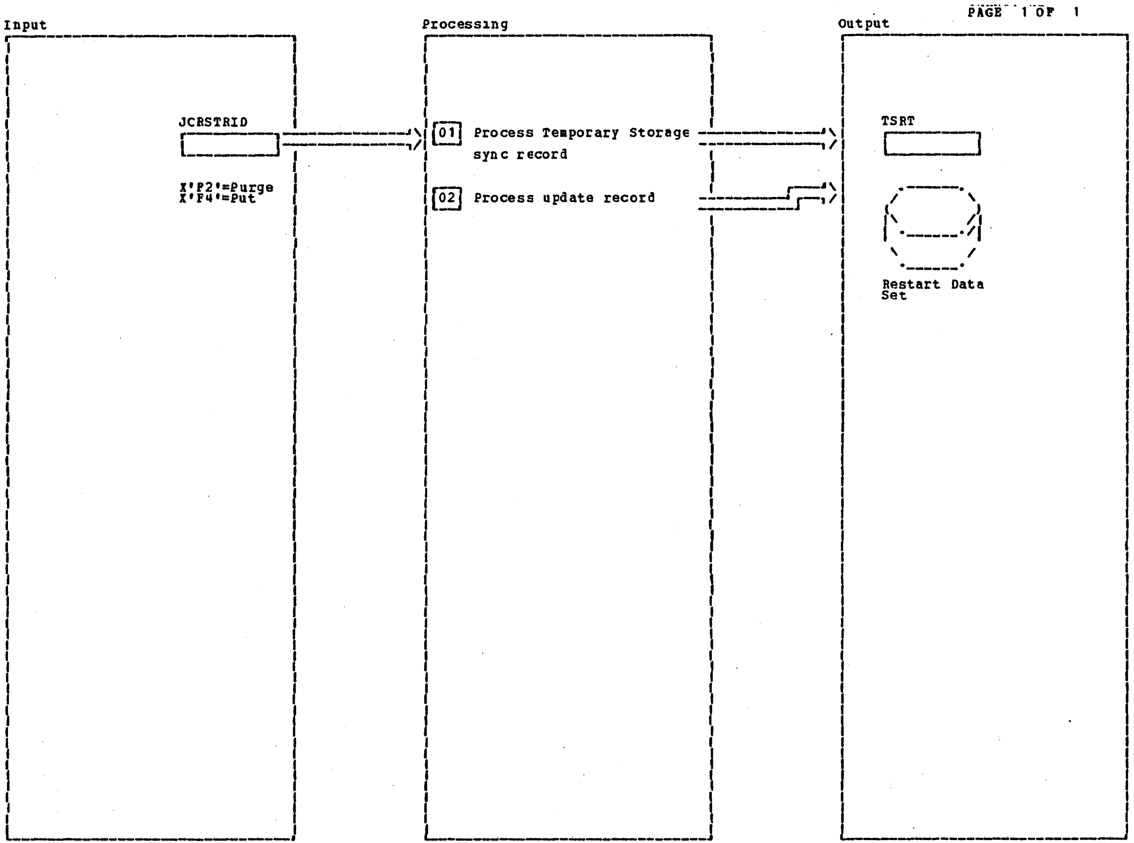
Process Data Ease/User Lcg Record (DFHRUP)

Diagram - 4.7.6.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Scan Dynamic Table for matching entry. If entry is not found, create entry. | | RUPRLOGR | | | | | |
| 02 If no end-of-task flag set in Dynamic Table entry, write Log Record to Restart data set. | | RUPR2004 | | | | | |
| 03 Set non-zero return code if log record written to restart data set and return to calling routine. | | | | | | | |

Process Data Base/User Lcg Record (DFHRUP)

Diagram - 4.7.6.3-01



RUUTSTR - Update TST Routine

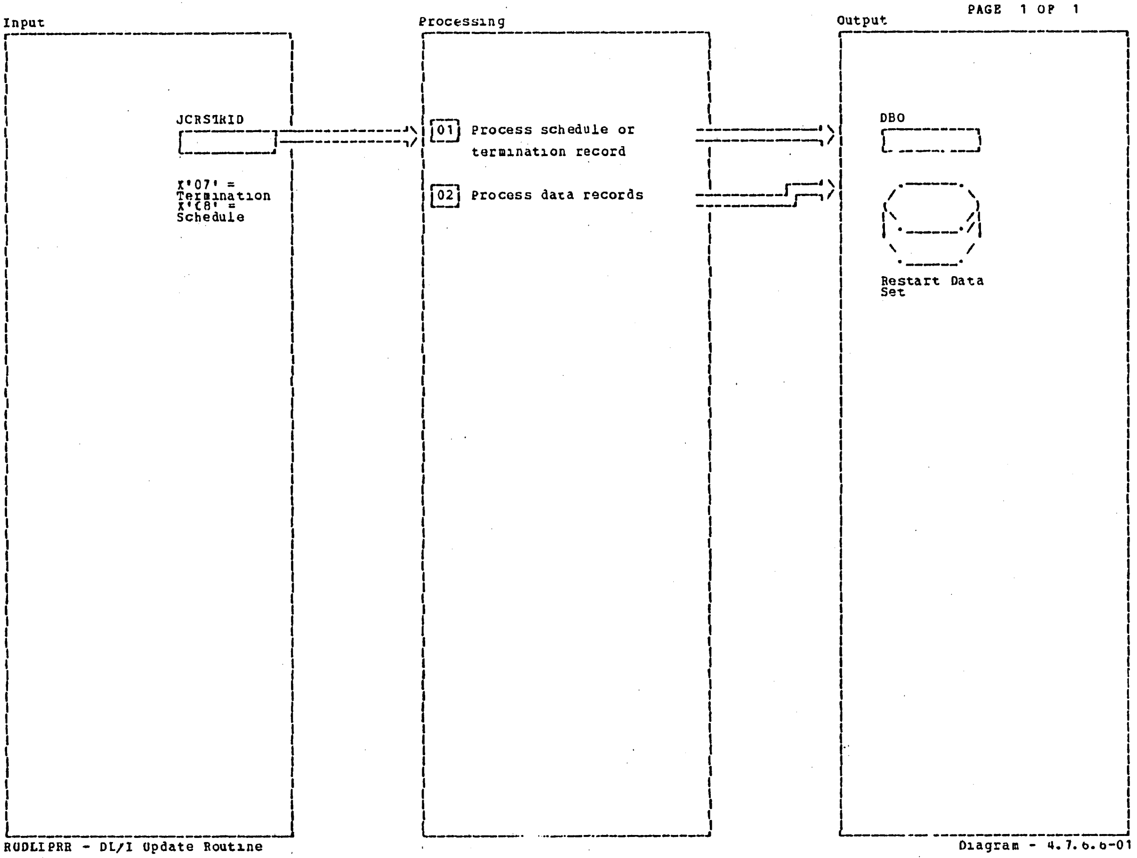
Diagram - 4.7.6.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Scan sync table for matching entry. If entry is found, scan the Temporary storage recovery table for matching entry. If entry is fcund and not marked processed, update data in entry and mark processed. If entry is not found, create entry and mark 'processed'. | | RUUPTSTR | | | | | |
| 02 Scan dynamic table for matching task ID. If matchirg entry is found and EOTK is set, exit. If EOTK, is not set, write record to restart data set. If entry is not found in dynamic table, create entry and write data record to restart data set. | | RUTS3000 | | | | | |

RUUTSTR - Update TST Routine

Diagram - 4.7.6.5-01

PAGE 1 OF 1



RUDLIPRR - DL/I Update Routine

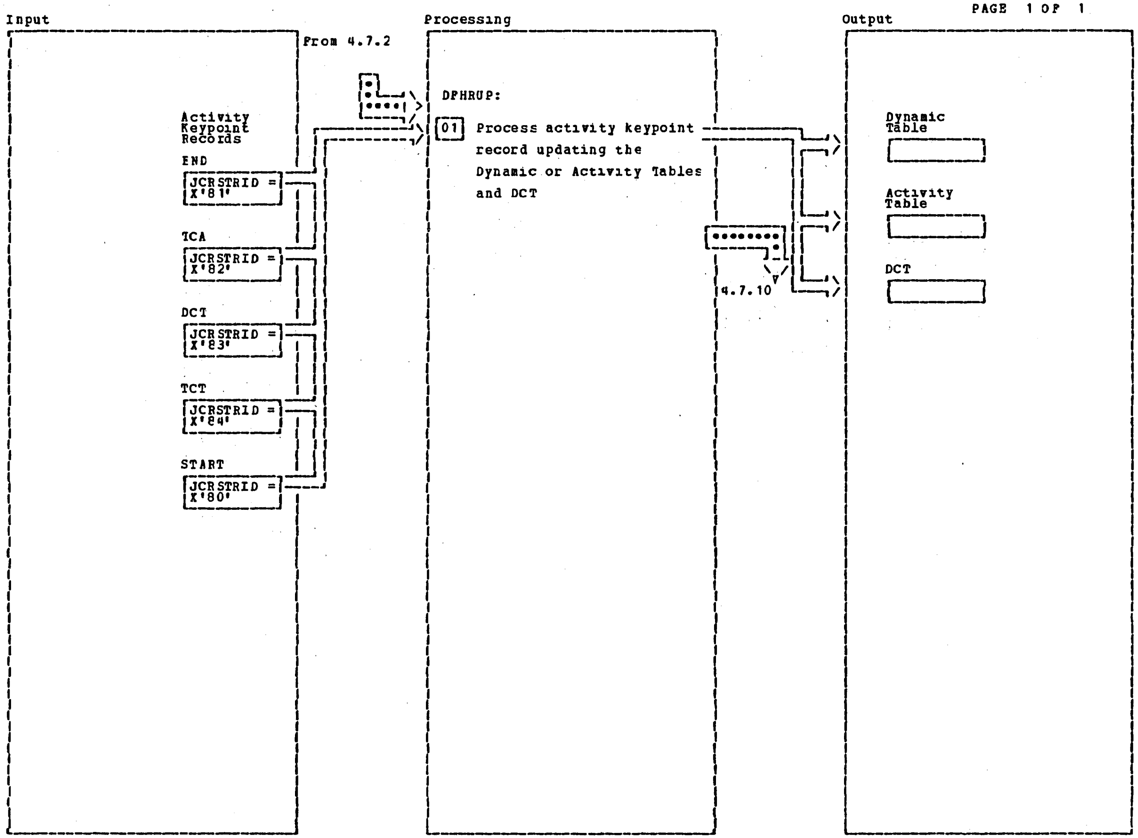
Diagram - 4.7.6.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Termination Record: Create entry in DBO table if matchirg entry cannot te found in the table. | | RUDLIPRR | | | | | |
| A. Schedule Record: Create entry in CBO table if matching entry cannot be found; mark entry as 'prccessed'. | | RUDL3000 | | | | | |
| 02 Data Records: Check for entry in DBO tatle marked 'processed'. If 'processed', set , exit. Otherwise, write data to restart data set. If entry is not found in DBO table, create entry and write data to the Restart Data Set. | | RUDL5000 | | | | | |

RUDLIPRR - DL/I Update Routine

Diagram - 4.7.6.6-01

PAGE 1 OF 1



RUPRKPR - Keypoint Processing Routine

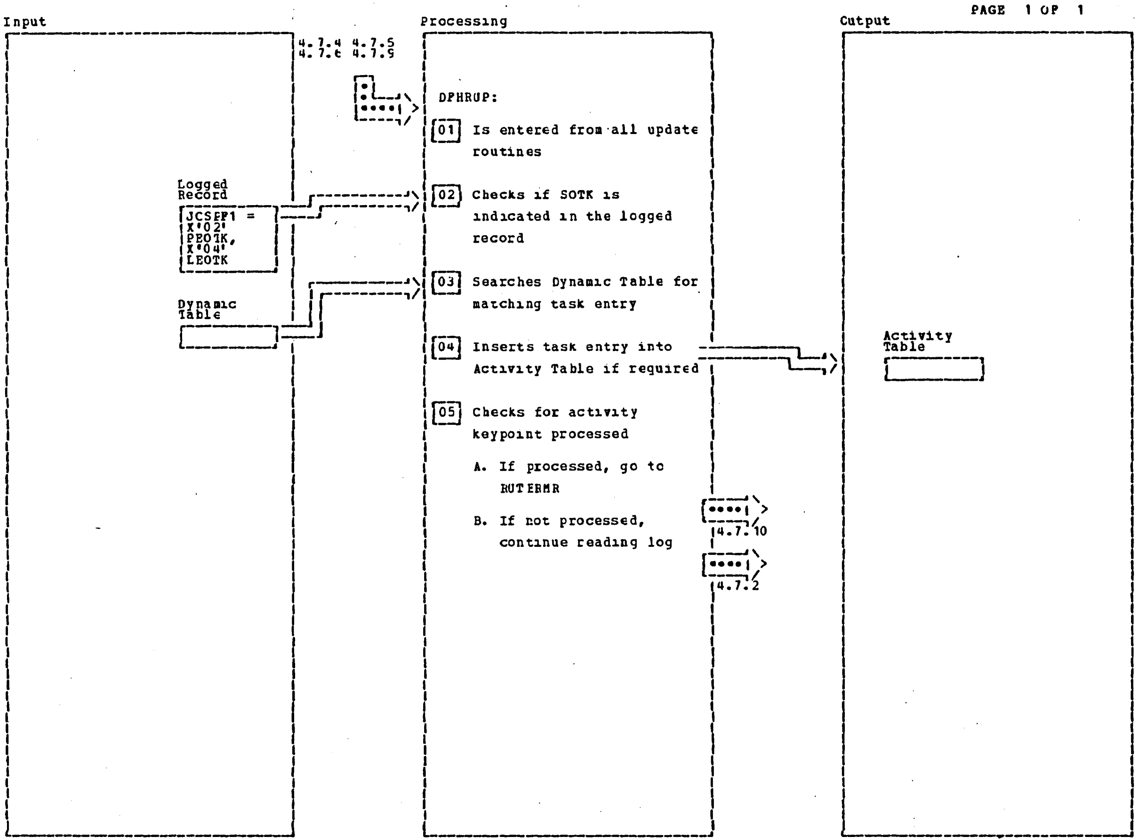
Diagram - 4.7.7-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If the first record read during an Activity Keypoint sequence is not the end keypoint record, the keypoint records are not processed.<br>Once the 'end' record has been established, the TCA, DCT, and TCT records are utilized to update the appropriate tables. When the start record is processed a flag (RUPKPPR) is set in field RUSTATUS to indicate that a keypoint has been processed and no further processing of the keypoint records is necessary. |  | RUPRKPR |  |  |  |  |  |

RUPRKPR - Keypoint Processing Routine

Diagram - 4.7.7-01

PAGE 1 OF 1



DPHRUP:

01 Is entered from all update routines

02 Checks if SOTK is indicated in the logged record

03 Searches Dynamic Table for matching task entry

04 Inserts task entry into Activity Table if required

05 Checks for activity keypoint processed

A. If processed, go to RUTERMR

B. If not processed, continue reading log

RUPRSOTR - Start-of-Task Processing Routine

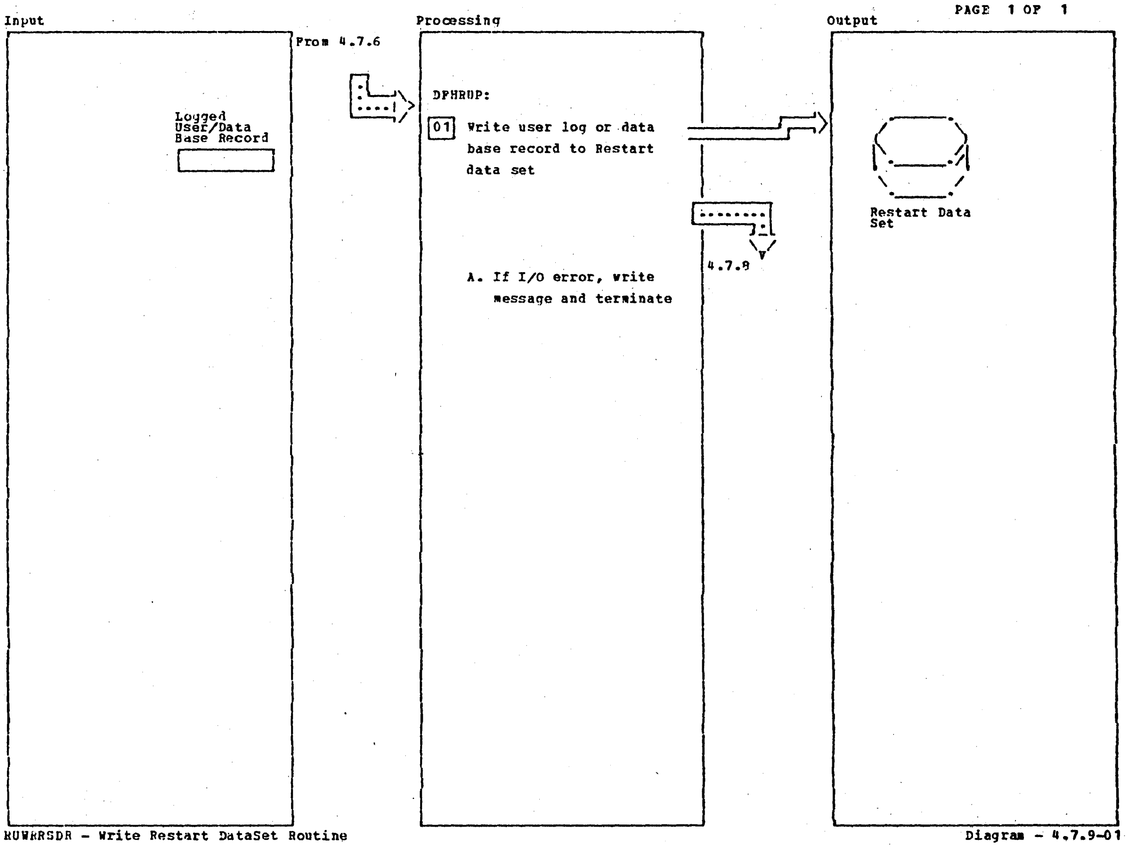
Diagram - 4.7.8-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 Checks the prefix field JCSPF1 for a X'C2' physical start-of-task (PSOTK) or X'04' logical start-of-task (LSOTK). If neither of these conditions exists, return tc read log RURDLOGR. |  | RUPRSOTR |  |  |  |  |  |
| 03 Searches the Dynamic Table for a matching task entry. If found, the following updates take place:<br>PSOTK = PEOTK, delete entry<br>PSOTK = LECTK, insert entry in Activity Table<br>PSOTK = LEOTK, no action<br>LSOTK = EECTK, no action |  | RUPRSO10 |  |  |  |  |  |
| 04 If entry is not found in the Dynamic lable, a task entry is created in the Activity Table. |  |  |  |  |  |  |  |

RUPRSOTR - Start-of-Task Processing Routine

Diagram - 4.7.8-01

PAGE 1 OF 1



Input

Processing

Output

From 4.7.6

Logged User/Data Base Record

DPHRUP:

01 Write user log or data base record to Restart data set

A. If I/O error, write message and terminate

4.7.8

Restart Data Set

RUWRRSDR - Write Restart DataSet Routine

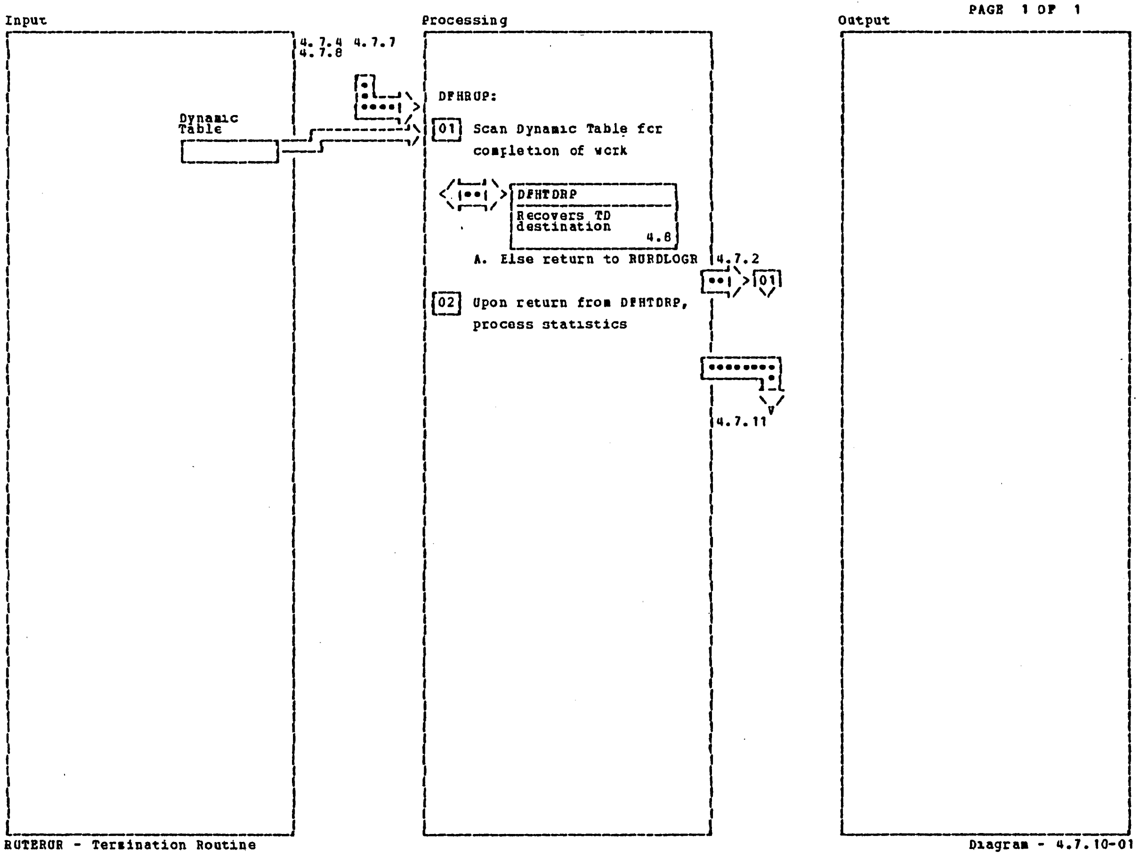
Diagram - 4.7.9-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The log record is written to the Restart data set by issuing macro DPHKP TYPE=WTBODATA.<br>A. DPH2808... ABEND code 2808 | | | | | | | |

RUWRRSDR - Write Restart DataSet Routine

Diagram - 4.7.9-01

PAGE 1 OF 1



DFHRUP:

[01] Scan Dynamic Table fcr completion of wcrk

DFHTDRP
Recovers TD destination 4.8

A. Else return to RURDLOGR

[02] Upon return from DFHTDRP, process statistics

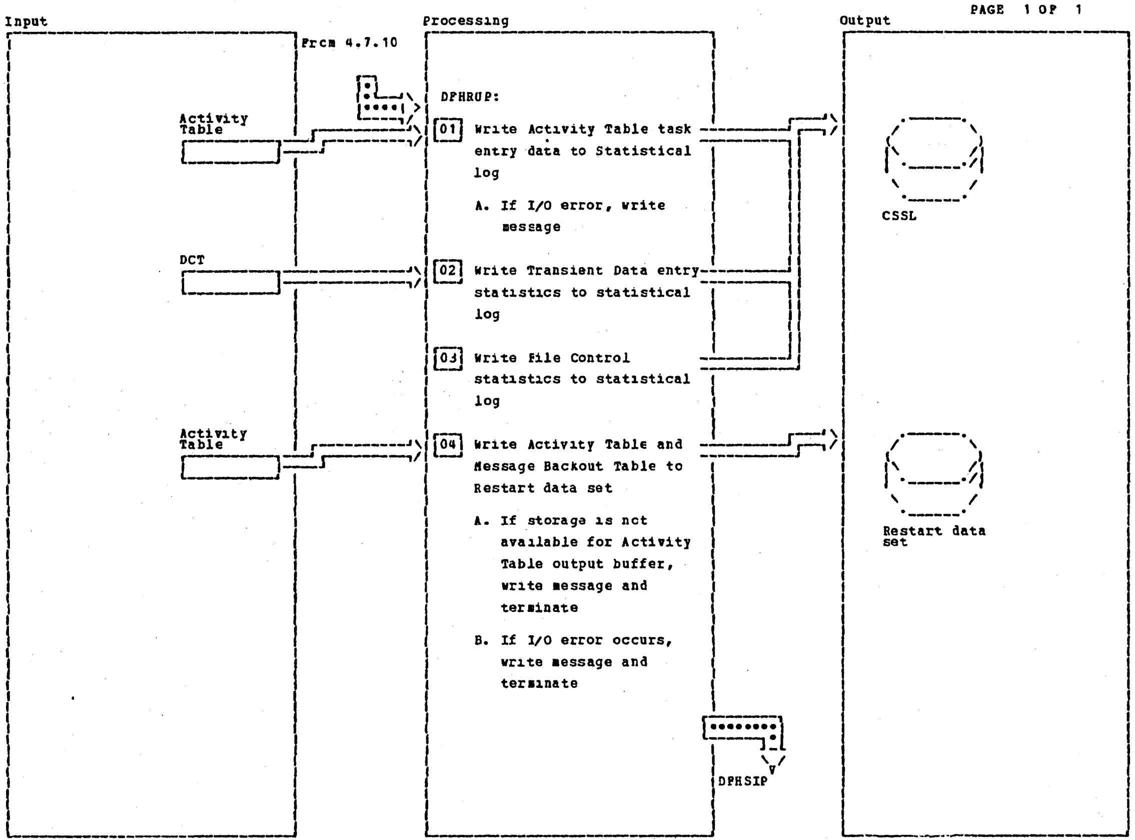
ROTEROR - Termination Routine

Diagram - 4.7.10-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| [01] The completion cf work is determined by scanning the Dynamic Table, to ensure that each task entry contains end-of-task conditions.<br>If this conditon exists, a link is issued to the Transient Data Recovery program (DFHTDRP).<br>If a task entry does not contain the end-cf-task condition, reading of the lcg records is continued until all entries in the Dynamic Table have completed a logical unit cf work. |  | ROTERM30 |  |  |  |  |  |

ROTEROR - Termination Routine

Diagram - 4.7.10-01


PAGE 1 OF 1
Input
Processing
Output
From 4.7.10
Activity Table
DCT
Activity Table
DFHRUP:
01 Write Activity Table task entry data to Statistical log
A. If I/O error, write message
02 Write Transient Data entry statistics to statistical log
03 Write File Control statistics to statistical log
04 Write Activity Table and Message Backout Table to Restart data set
A. If storage is not available for Activity Table output buffer, write message and terminate
B. If I/O error occurs, write message and terminate
DFHSIP
CSSL
Restart data set

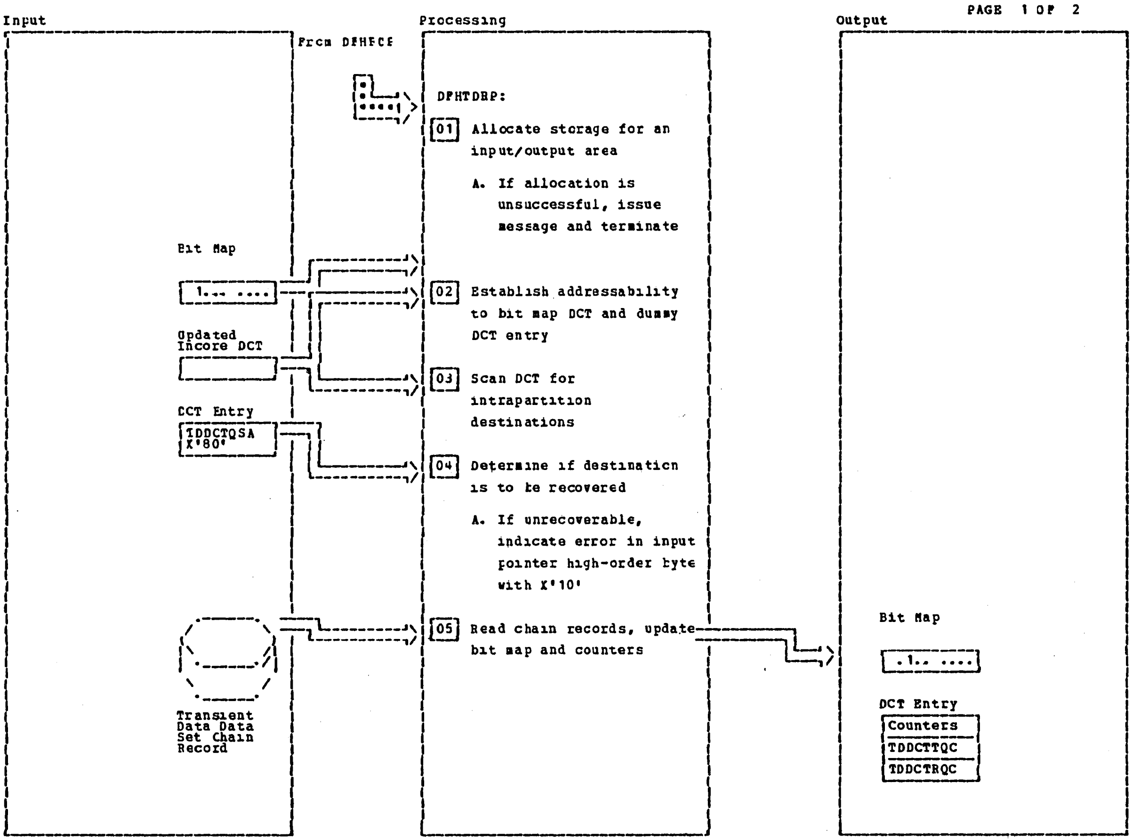
RUWRSINR - Write Statistics Routine

Diagram - 4.7.11-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Take statistical data from the task entries in the Activity Table and write the data to the statistical log. CSSL data consist of:<br>- Task ID<br>- Transaction ID<br>- Terminal ID<br>- Data Collected Count<br>A. DFH2807I... | | RUWRSINR | |
| 02 Scan DCT for Intrapartition entries and write statistical data. Data consist of:<br>- Destination ID<br>- Recovery status<br>- Total records in queue | | RUWRSI50 | |
| 04 The entries in the Activity Table and Message Backout Table are written to the Restart data set to be used later by a Transaction Backout program.<br>A. DFH2806I... ABEND code 2806<br>B. DFH2809I... ABEND code 2809 | | RUWRS200 | |

RUWRSINR - Write Statistics Routine

Diagram - 4.7.11-01



DFHTDRP:

01 Allocate storage for an input/output area

A. If allocation is unsuccessful, issue message and terminate

02 Establish addressability to bit map DCT and dummy DCT entry

03 Scan DCT for intrapartition destinations

04 Determine if destination is to be recovered

A. If unrecoverable, indicate error in input pointer high-order byte with X'10'

05 Read chain records, update bit map and counters

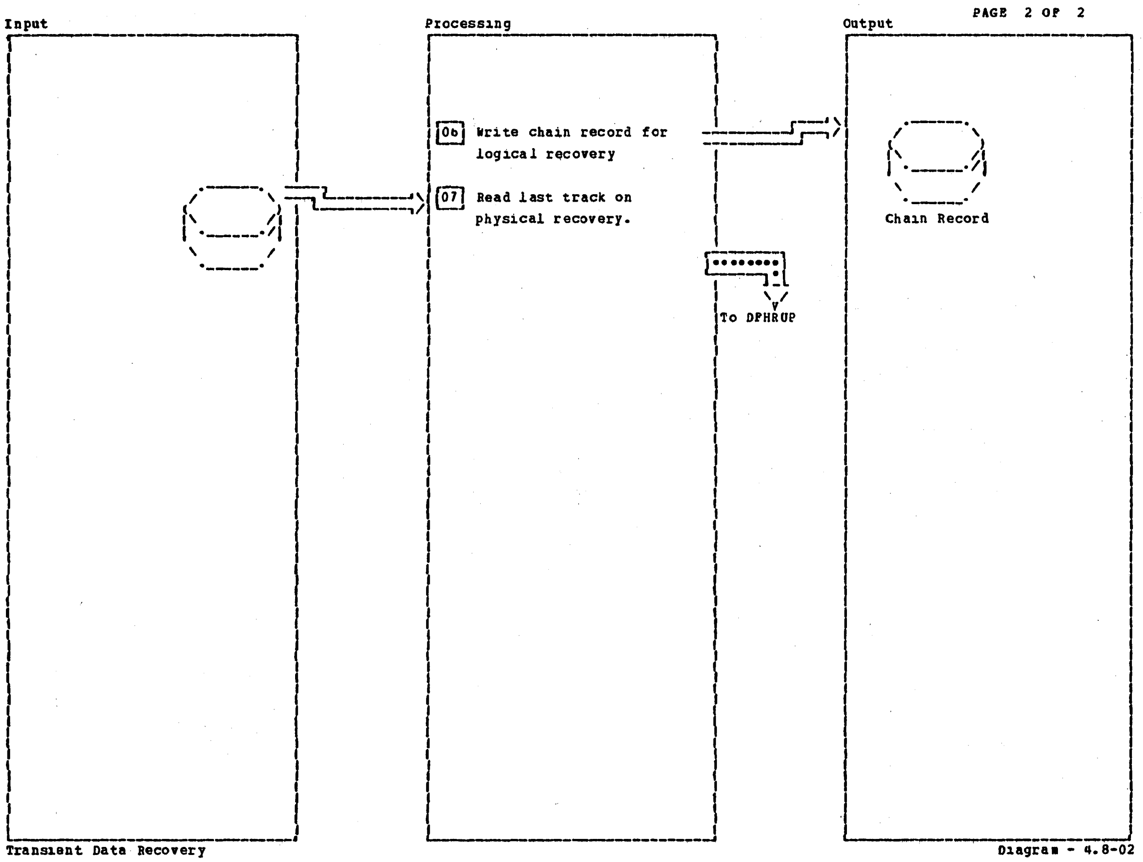
Transient Data Recovery

Diagram - 4.8-01

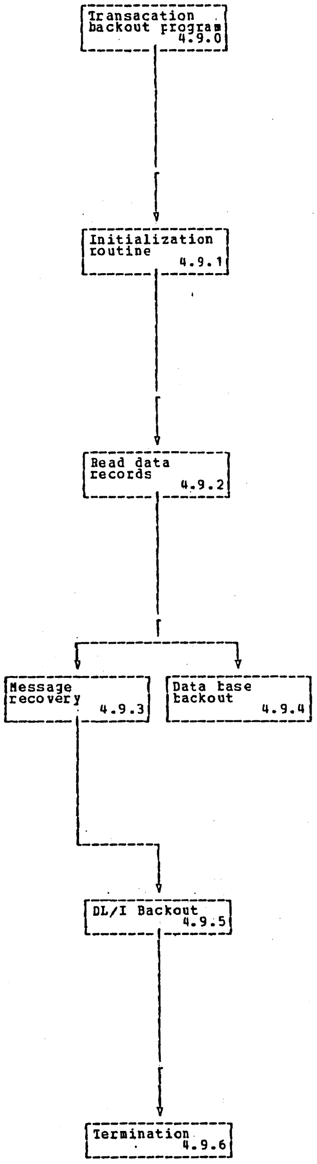
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01<br>A. DFH2810I... ABEND code 2810 | | | |
| 02 Pointer to bit map is at location - 12 from the DCT entry. Save bit map address in location TDRBMPTR. Store pointers to the input/output area and dummy DCT in the TCA. | | DFHTDR20 | |
| 03 Intrapartition destinations are determined by checking field TDDCTDT of the DCT for a X'80' (equates to TDINDTBU). | | TDR0000 | |
| 04 The destination is recovered if the queue start address (QSA) TDDCTQSA is nonzero and the high-order byte does not contain a X'80' (error in DFHRUP while updating the DCT). | | TDR0300 | |
| 05 Reading of chain records always begins with the QSA. As each chain record is read, the bit representing the track is flipped on in the bit map and the total queue count is incremented.<br>The chain is broken when no forward pointer exists for physical recovery or when the chain record track aquals the output pointer track (TDDCTODA).<br>For logical recovery, the counters are updated in the DCT. | | TDR1000 | |

Transient Data Recovery

Diagram - 4.8-01

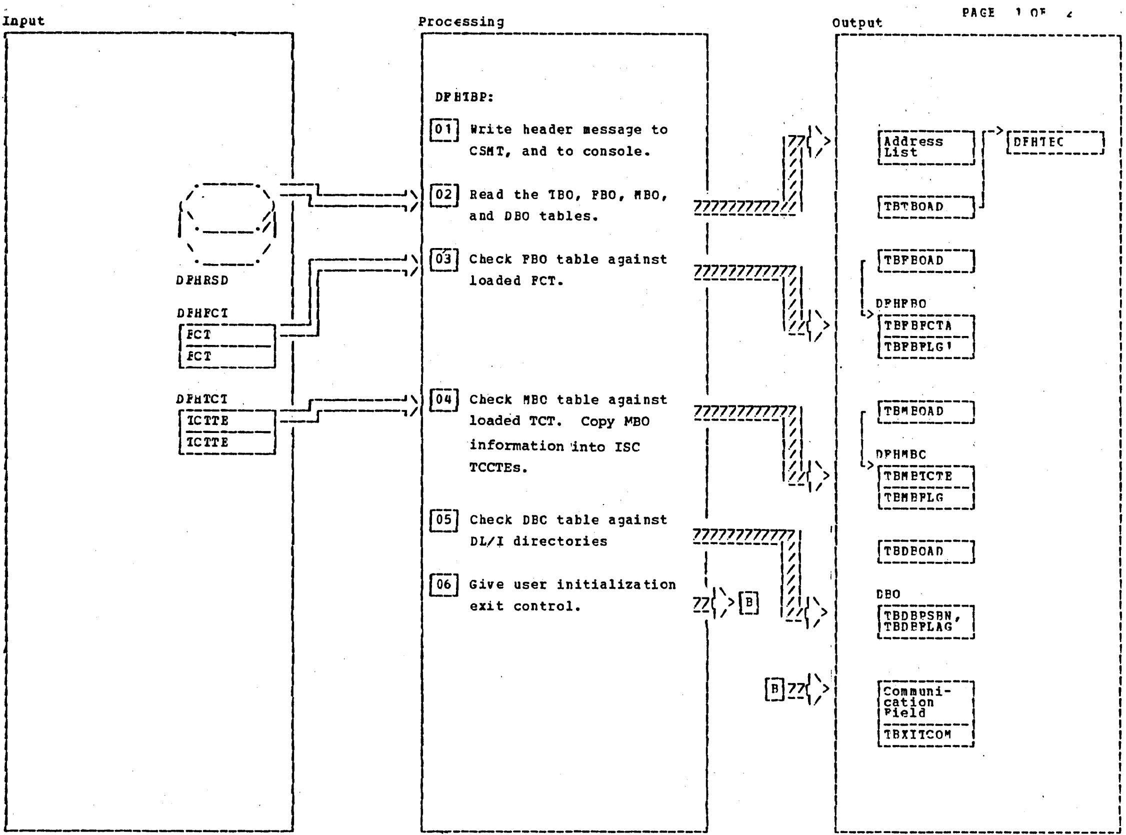


| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 06 For logical recovery, the last track of the destination is determined. If a forward chain pointer exists, the chai record is rewritten. If nc forward pointer exists, rewriting of chain record is bypassed. | | | | | | | |
| 07 The last track is read to determine the amcunt of records that exist cn the track and to determine whether the maximum data length has changed. | | TDR2000 | | | | | |



Transaction backout program

Diagram - 4.9-01



01 Write header message to CSMT, and to console.

02 Read the TBO, FBO, MBO, and DBO tables.

03 Check FBO table against loaded FCT.

04 Check MBO table against loaded TCT. Copy MBO information into ISC TCCTEs.

05 Check DBC table against DL/I directories

06 Give user initialization exit control.

Transaction Backout Program-Initialization Routine

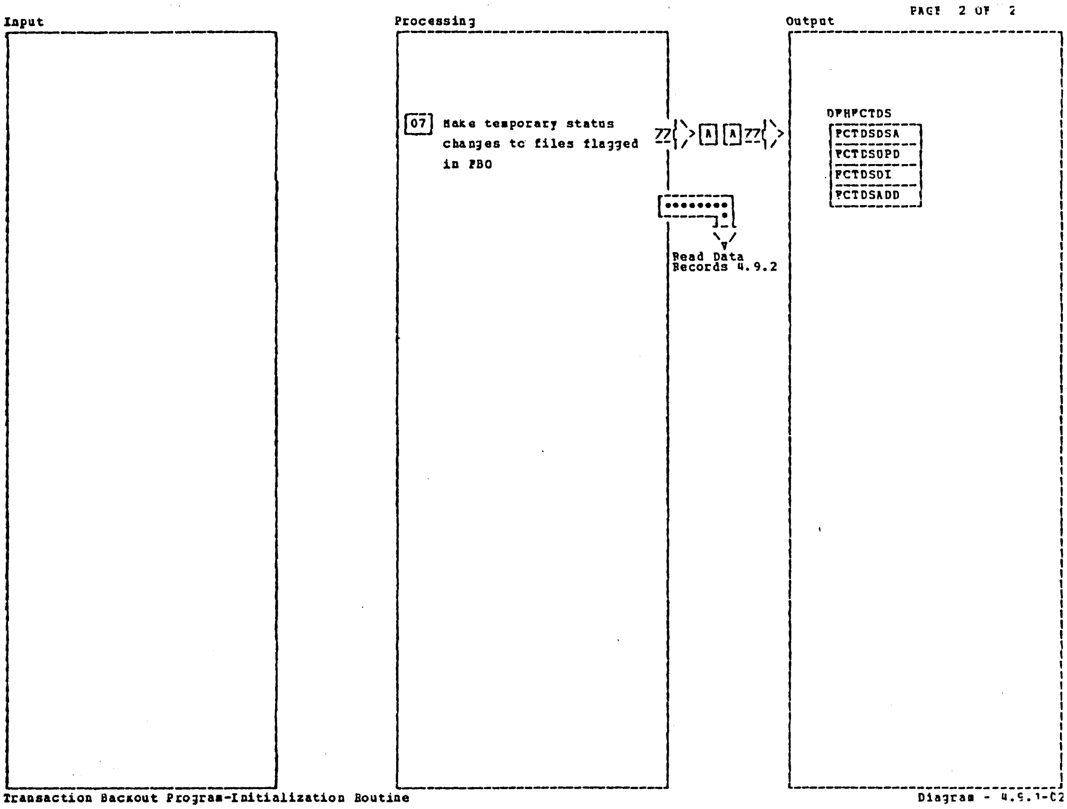
Diagram - 4.9.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Message DFH5701. | | | |
| 02 If TBO table is absent, message DFH5703 is issued. | | TBOC0100 | |
| If FBO table is missing,same message is issued. | | TB000130 | |
| If MBO table is missing, same message is issued. | | TBOC0160 | |
| If DBO table is missing, same message is issued. | | TB001000 | |
| If an I/O error occurs, message DFH5702 is issued and CICS/VS is terminated. | | | |
| The addresses to the tables are placed in fields TBTBOAD, TBFBOAD, TBMBOAD, and TBDBOAD respectively. | | | |
| 03 FBO entries without matching FCTs are flagged as absent. | | TB000210 | |
| Initial status of file is checked and FBO entries are flagged for appropriate | | TB000225 | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| temporary status changes. | | | |
| 04 MBO entries without matching TCTTEs are flagged as absent. | | TB000255 | |
| 05 DBO entries are checked against the DL/I PSB Directory and the DMB directory to ensure that the PSB's are schedulable. If the PSB is not schedulable, the entry is marked accordingly | | TB00105C | |
| If there is no DL/I support in the system, all entries are marked for no-action. | | | |
| 06 Via a BALR14 instruction. | | TB000275 | |

Transaction Backout Program-Initialization Routine

Diagram - 4.9.1-01



07 Make temporary status changes to files flagged in FBO

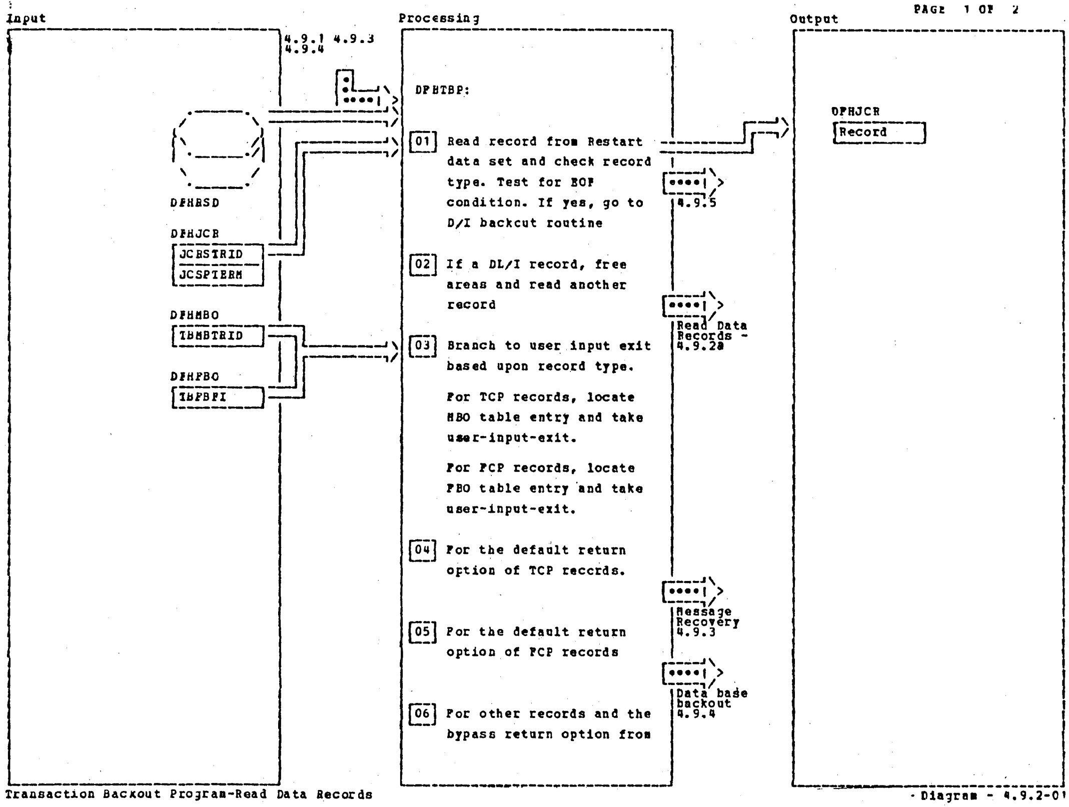
Transaction Backout Program-Initialization Routine

Diagram - 4.9.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 07 Open any defered open files that have inflight task backout information. |  | TBOC0310 |  |  |  |  |  |
| Issue message DFH5709 if no space. Issue message DFH5708 and give control to user exit. |  | TBO00315 |  |  |  |  |  |

Transaction Backout Program-Initialization Routine

Diagram - 4.9.1-02



DFHTBP:

01 Read record from Restart data set and check record type. Test for EOF condition. If yes, go to D/I backout routine

02 If a DL/I record, free areas and read another record

03 Branch to user input exit based upon record type.

For TCP records, locate HBO table entry and take user-input-exit.

For FCP records, locate FBO table entry and take user-input-exit.

04 For the default return option of TCP records.

05 For the default return option of FCP records

06 For other records and the bypass return option from

Transaction Backout Program-Read Data Records

Diagram - 4.9.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 If I/O error, issue message DFH5780, give user-input-exit control, then ABEND CICS/VS. | | TB000500<br>TB8C0000 | |
| If unmatched TCP record issue message DFH5710 with GO/CANCEL option. If GO return to read next record. | | TB000945 | |
| If unmatched FCP record issue message DFH5710 with GO/CANCEL option. If GO return to read next record | | TB000545 | |
| If EOF condition go to TB900000. | | | |
| 03 For TCP records, go to TB000900 for FCP records go to TB000530, for user records go to TB000560. | | TB0C0510 | |
| 04 User input exits come back to the following labels for default action. For TCP records, TB0C0910, for FCP records, TB000521, for user records at label, TB000565. | | | |
| 05 See note 3. | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 06 User input exits return to TB000524 to bypass the processing of the record. | | | |

Transaction Backout Program-Read Data Records

Diagram - 4.9.2-01

PAGE 2 OF 2

Input

Processing

the user input exit, free areas and read ancther record.

Read Data Records 4.9.2

Output

Transaction Backout Program-Read Data Reccrds

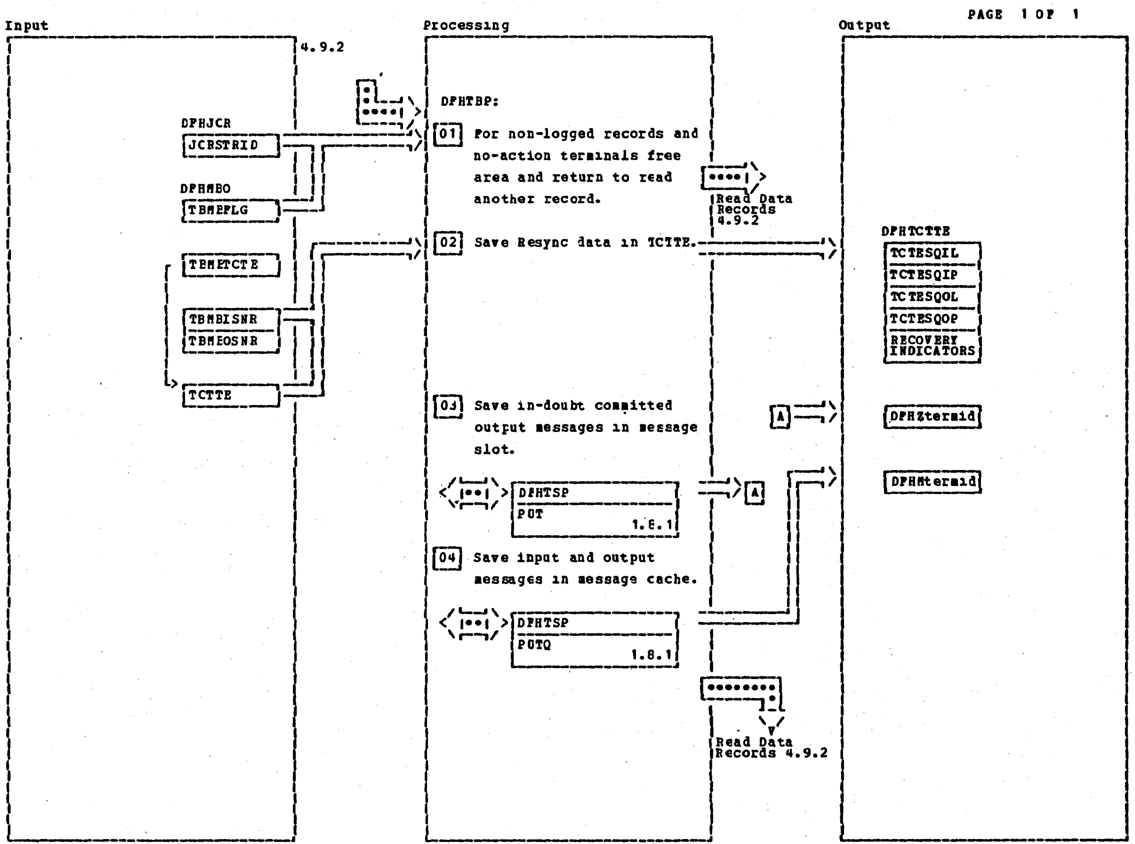
Diagram - 4.9.2-02

| Notes | Routine | Label | Ref |
|---|---|---|---|

| Notes | Routine | Label | Ref |
|---|---|---|---|

Transaction Backout Program-Read Data Records

Diagram - 4.9.2-02



01 For non-logged records and no-action terminals free area and return to read another record.

02 Save Resync data in TCTTE.

03 Save in-doubt committed output messages in message slot.

04 Save input and output messages in message cache.

Transaction Backout Program-Message Recovery

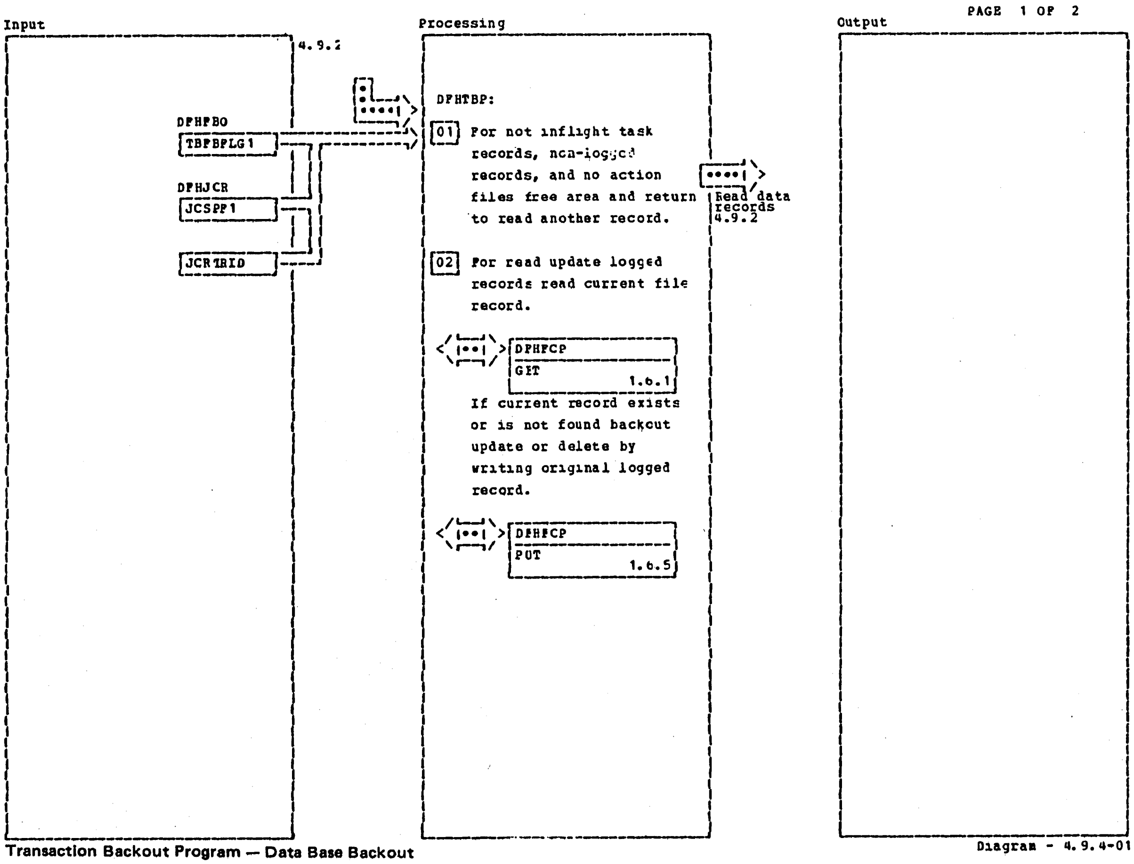
Diagram - 4.9.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | TB000910 | |
| 02 Sets indicators TCTEIBRS and TCTEIRPR. | | TB000925 | |
| 03 Issues DFHTS-PUT macro. Sets indicator TCTECOPF. | | | |
| 04 Issues DFHTS PUTQ macro. | | TB000915 | |

Transaction Backout Program-Message Recovery

Diagram - 4.9.3-01

PAGE 1 OF 2



DFHTBP:

01 For not inflight task records, non-logged records, and no action files free area and return to read another record.

02 For read update logged records read current file record.

If current record exists or is not found backout update or delete by writing original logged record.

Transaction Backout Program — Data Base Backout

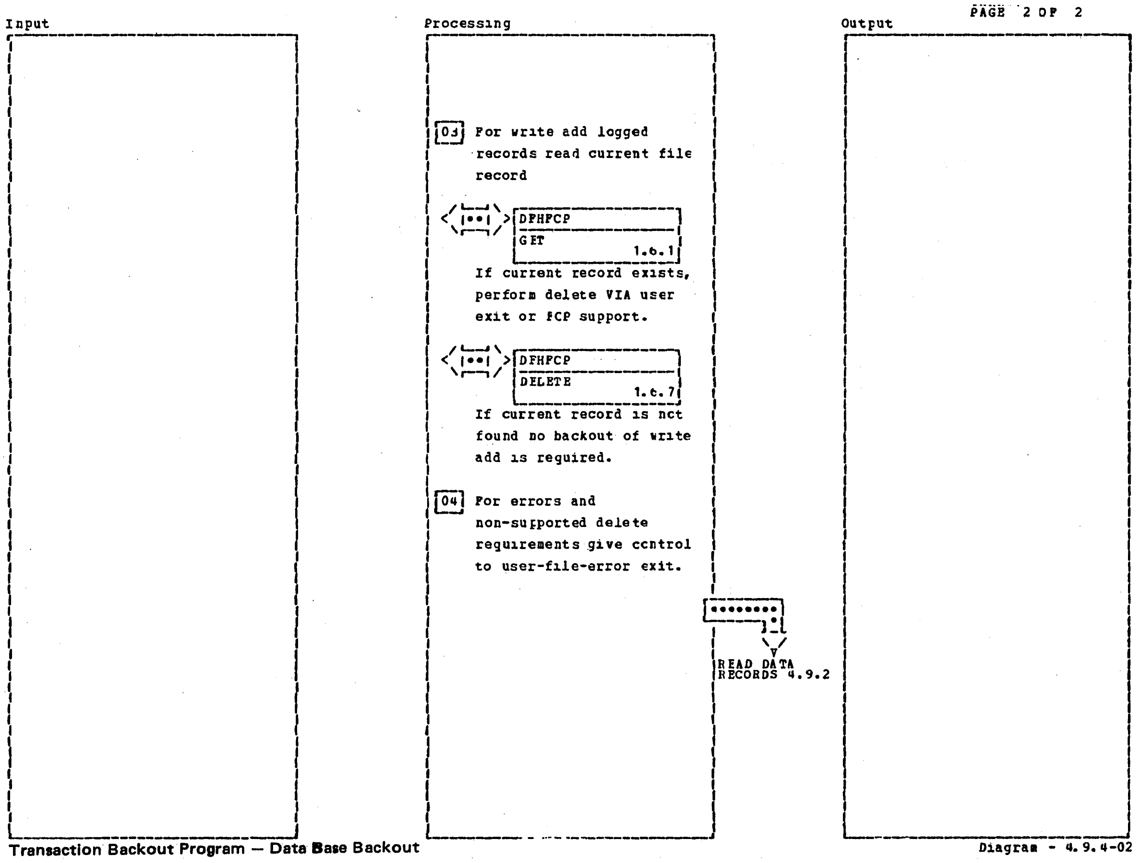
Diagram - 4.9.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | TB000521 | | | | | |
| 02 | | TB000700 | | | | | |

Transaction Backout Program — Data Base Backout

Diagram - 4.9.4-01

PAGE 2 OF 2



Input

Processing

Output

03 For write add logged records read current file record

DFHFCP
GET 1.6.1

If current record exists, perform delete VIA user exit or FCP support.

DFHFCP
DELETE 1.6.7

If current record is nct found no backout of write add is required.

04 For errors and non-supported delete requirements give control to user-file-error exit.

READ DATA RECORDS 4.9.2

Transaction Backout Program — Data Base Backout

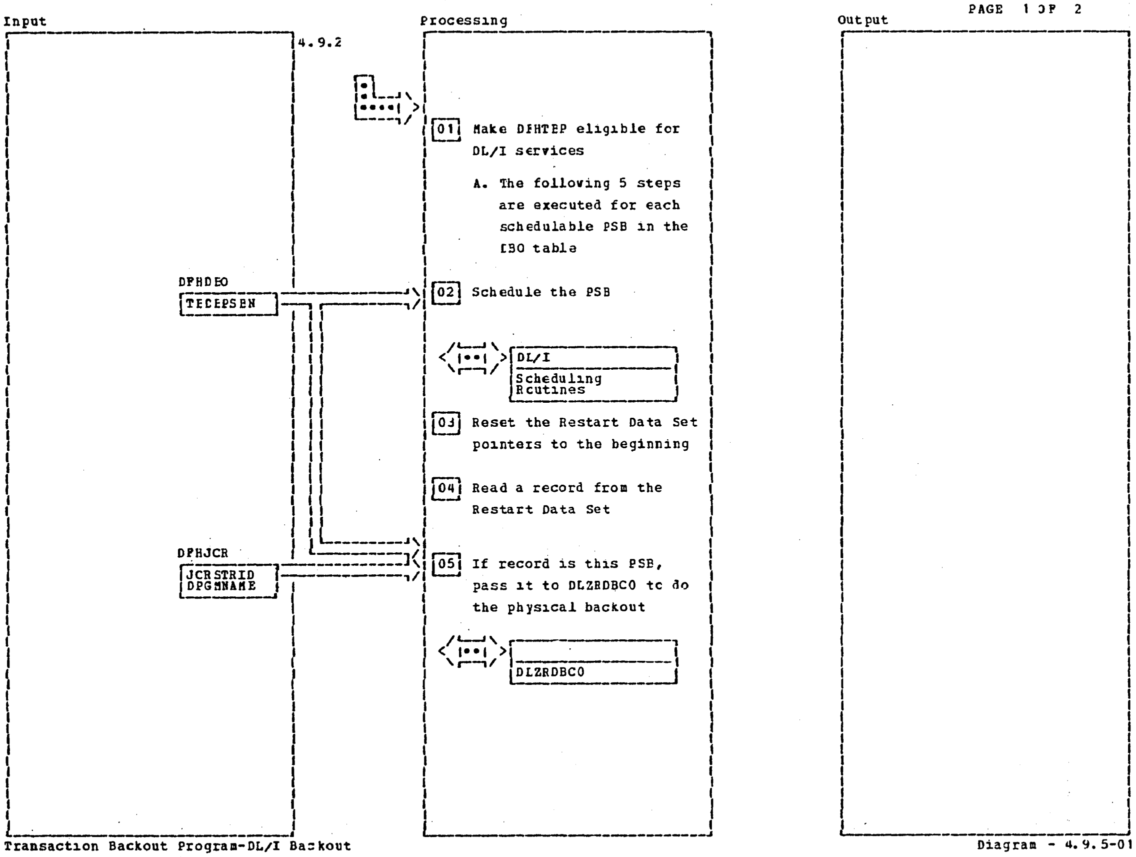
Diagram - 4.9.4-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 User file error exit is given control if a record added to a ISAM, BDAM, or VSAM ESDS data set is to be backed out (deleted). The user exit should logically delete the current record by marking it according to installation convention. TBP will then rewrite the logically deleted record. |  | TB000800 |  |  |  |  |  |
| 04 The communication byte TBXITCOM defines the type of error for user file error exit. |  |  |  |  |  |  |  |

Transaction Backout Program — Data Base Backout

Diagram - 4.9.4-02

PAGE 1 OF 2



01 Make DFHTBP eligible for DL/I services

A. The following 5 steps are executed for each schedulable PSB in the CBO table

02 Schedule the PSB

03 Reset the Restart Data Set pointers to the beginning

04 Read a record from the Restart Data Set

05 If record is this PSB, pass it to DLZRDBCO to do the physical backout

Transaction Backout Program-DL/I Backout

Diagram - 4.9.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The address of an ACT assembled inside DFHTBP is placed in the TCA and the DL/I SCD. The PPT entry for DFHTBP is flagged for DL/I. | | TB001500 | | | | | |
| 02 The number of the PSB directory entry for this PSB is placed in the ACT within DFHTBP, and a DL/I scheduling call issued. | | TB001510 | | | | | |
| 05 Each DL/I log record contains the PSB name. | | TB001550 | | | | | |

Transaction Backout Program-DL/I Backout

Diagram - 4.9.5-01

PAGE 2 OF 2

Input

Processing

06 At end of file on the Restart Data Set, unschedule the PSB

DL/I Unschedule Routines

07 When DBO table entries are processed, go to Transcation Backout Program Termination

Termination 4.9.6

Output

Transaction Backout Program-DL/I Backout

Diagram - 4.9.5-02

| Notes | Routine | Label | Ref |
|---|---|---|---|

| Notes | Routine | Label | Ref |
|---|---|---|---|

Transaction Backout Program-DL/I Backout

Diagram - 4.9.5-02

PAGE 1 OF 1

INPUT

4.9.2

PROCESS

DFHTBF

01 Go to user termination exit.

02 Send termination message

03 Reset Restart Data Set pointers to beginning.

04 Return.

1.3.6

OUTPUT

Transaction Backout Program-Termination

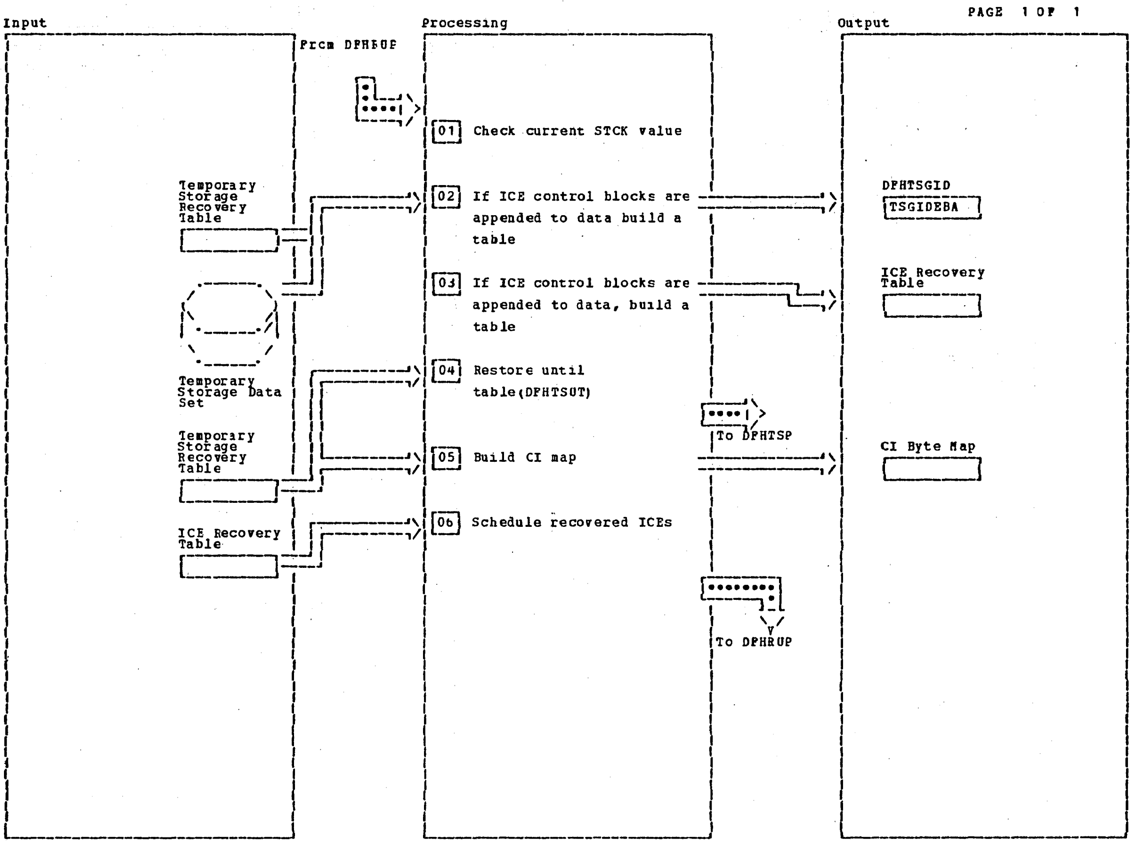
Diagram - 4.9.6-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 | | TB900000 | |
| 02 Message DFH5790 | | TB900030 | |
| 03 | | TBPCRET | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Transaction Backout Program-Termination

Diagram - 4.9.6-01


PAGE 1 OF 1
Input
Processing
Output
From DFHRUP
Temporary Storage Recovery Table
Temporary Storage Data Set
Temporary Storage Recovery Table
ICE Recovery Table
01 Check current STCK value
02 If ICE control blocks are appended to data build a table
03 If ICE control blocks are appended to data, build a table
04 Restore until table(DFHTSUT)
05 Build CI map
06 Schedule recovered ICEs
To DFHTSP
To DFHRUP
DFHTSGID
TSGIDEBA
ICE Recovery Table
CI Byte Map

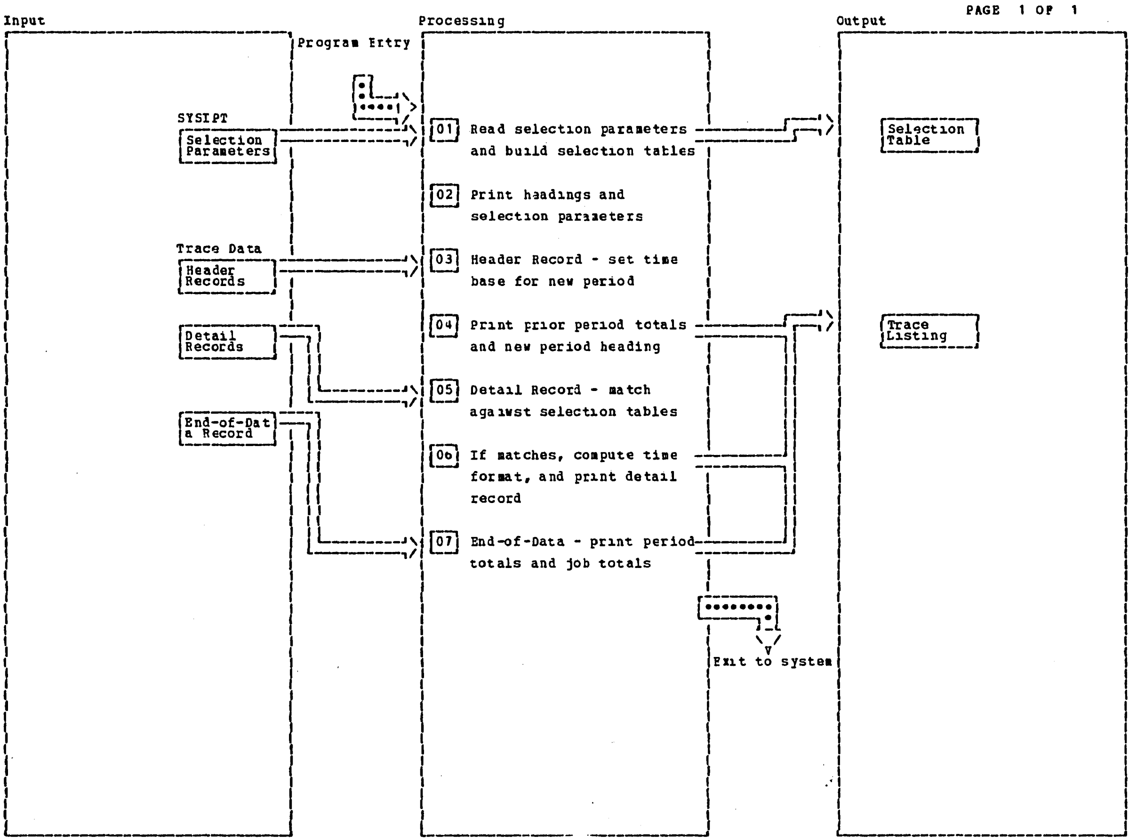
Temporary Storage Recovery Program

Diagram - 4.10-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Issues System/370 STCK request. If not set, notify operator. Compare STCK value with last keypointed STCK value. If not higher, notify operator. | | TSRP0010 | |
| 02 Each record on the data set has the data identification included in the record header, along with record number for queue-type data. The recovery table is scanned to see if the data is to be recovered. If more than one record with the same DATAID is found, the data with the highest STCK value is taken. The auxiliary storage address is stored in the TS Recovery Table. If the current STCK value is less than the STCK value of a record, CICS restart is terminated. | | TSRP0050 | |
| 03 Durning the scanning of the data set, if a recoverable record has an ICE appended, create a table and move the ICE into the table in ascending sequence by time created, based on STCK value in the record prefix. | | | |
| 04 After the data set has processed, DFHTSRP points to each entry in the recovery table and calls DFHTSP to create an entry in the unit table (DFHTSUT). | | | |
| 05 The record length given in the auxiliary address is usad to update the CI use map to reflect the number of available segments in the CI. | | | |
| 06 ICEs are chained off the CSA via link to either DFHICP or DFHKCP for each ICE. | | | |

Temporary Storage Recovery Program

Diagram - 4.10-01


PAGE 1 OF 1
Input
Processing
Output
Program Entry
SYSIPT
Selection Parameters
01 Read selection parameters and build selection tables
Selection Table
02 Print headings and selection parameters
Trace Data
Header Records
03 Header Record - set time base for new period
Detail Records
04 Print prior period totals and new period heading
Trace Listing
05 Detail Record - match against selection tables
End-of-Dat a Record
06 If matches, compute time format, and print detail record
07 End-of-Data - print period totals and job totals
Exit to system

Trace Utility Program

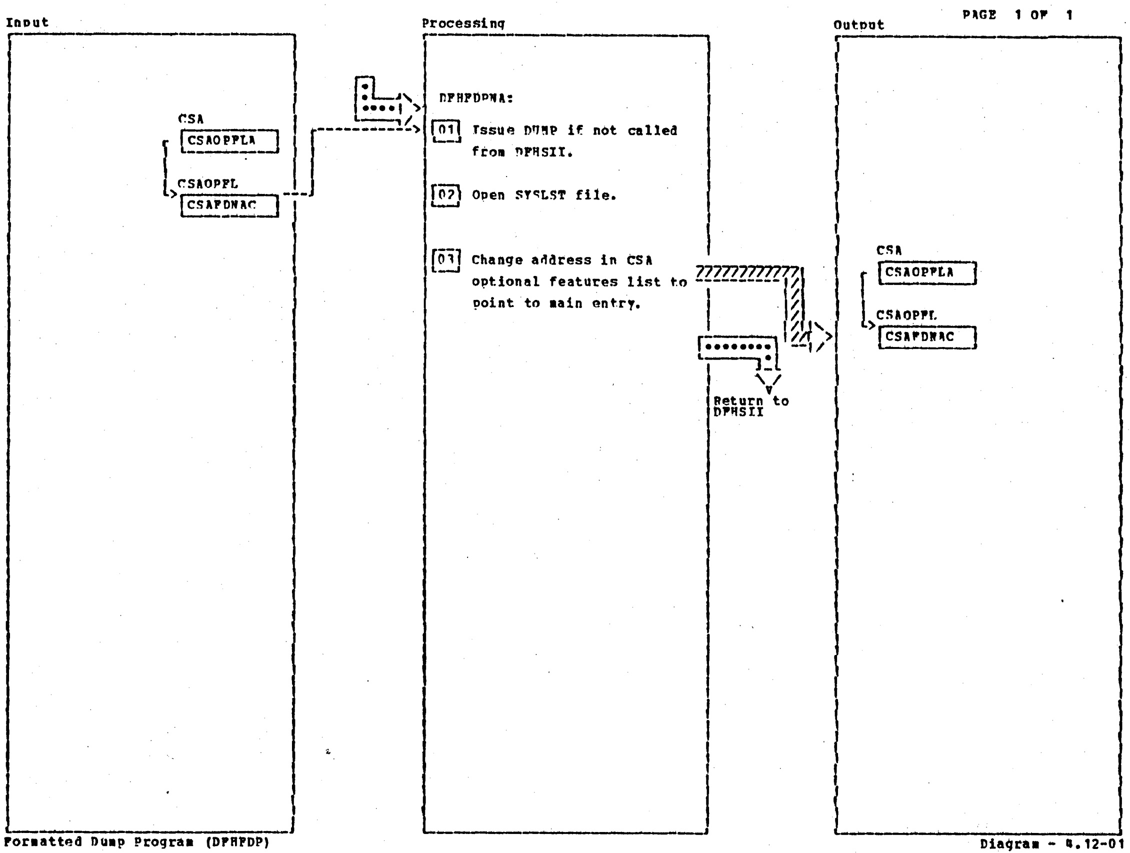
Diagram - 4.11-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If no input, ALL is assumed. | | TUINP | | | | | |
| 02 | | | | | | | |
| 03 Base is CICS time and saved value of STCK instruction. | | TUHED | | | | | |
| 04 | | TUPRTSHD | | | | | |
| 05 Record is not printed if not selected. | | TUDET | | | | | |
| 06 Time is determined by converting displacement of STCK value to 1/100 sec and adding to CICS time. | | TUDTF | | | | | |
| 07 | | ATREODAD | | | | | |

Trace Utility Program

Diagram - 4.11-01

PAGE 1 OF 1



01 Issue DUMP if not called from DFHSII.

02 Open SYSLST file.

03 Change address in CSA optional features list to point to main entry.

Formatted Dump Program (DFHFDP)

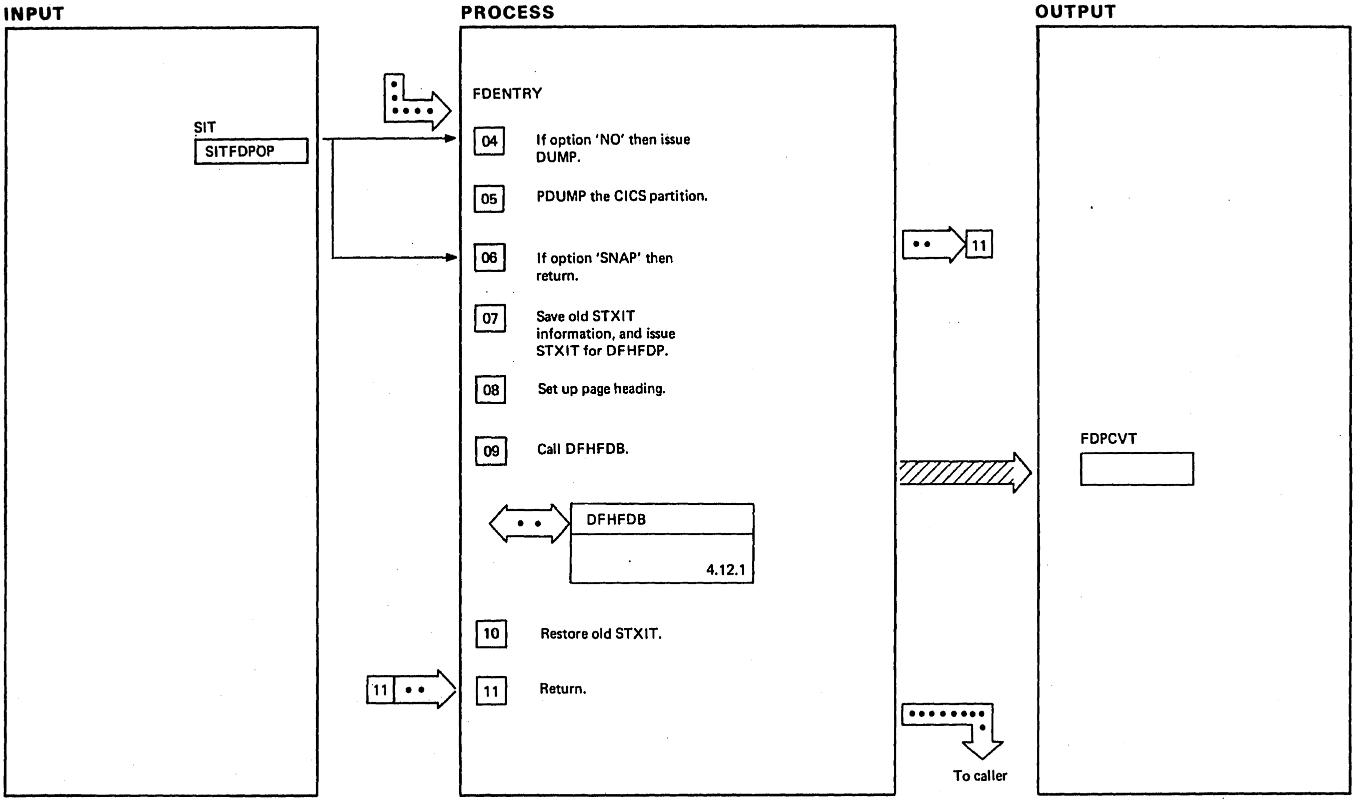
Diagram - 4.12-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 For the call to initialize DFHFDP, DFHSII sets a bit in the top byte of CSAFDNAC. | | | | | | | |

Formatted Dump Program (DFHFDP)

Diagram - 4.12-01

PAGE 2 OF 2


INPUT
SIT
SITFDPOP
PROCESS
FDENTRY
04 If option 'NO' then issue DUMP.
05 PDUMP the CICS partition.
06 If option 'SNAP' then return.
07 Save old STXIT information, and issue STXIT for DFHFDP.
08 Set up page heading.
09 Call DFHFDB.
DFHFDB
4.12.1
10 Restore old STXIT.
11 Return.
11
11
To caller
OUTPUT
FDPCVT

Formatted Dump Program (DFHFDP)

Diagram - 4.12-02

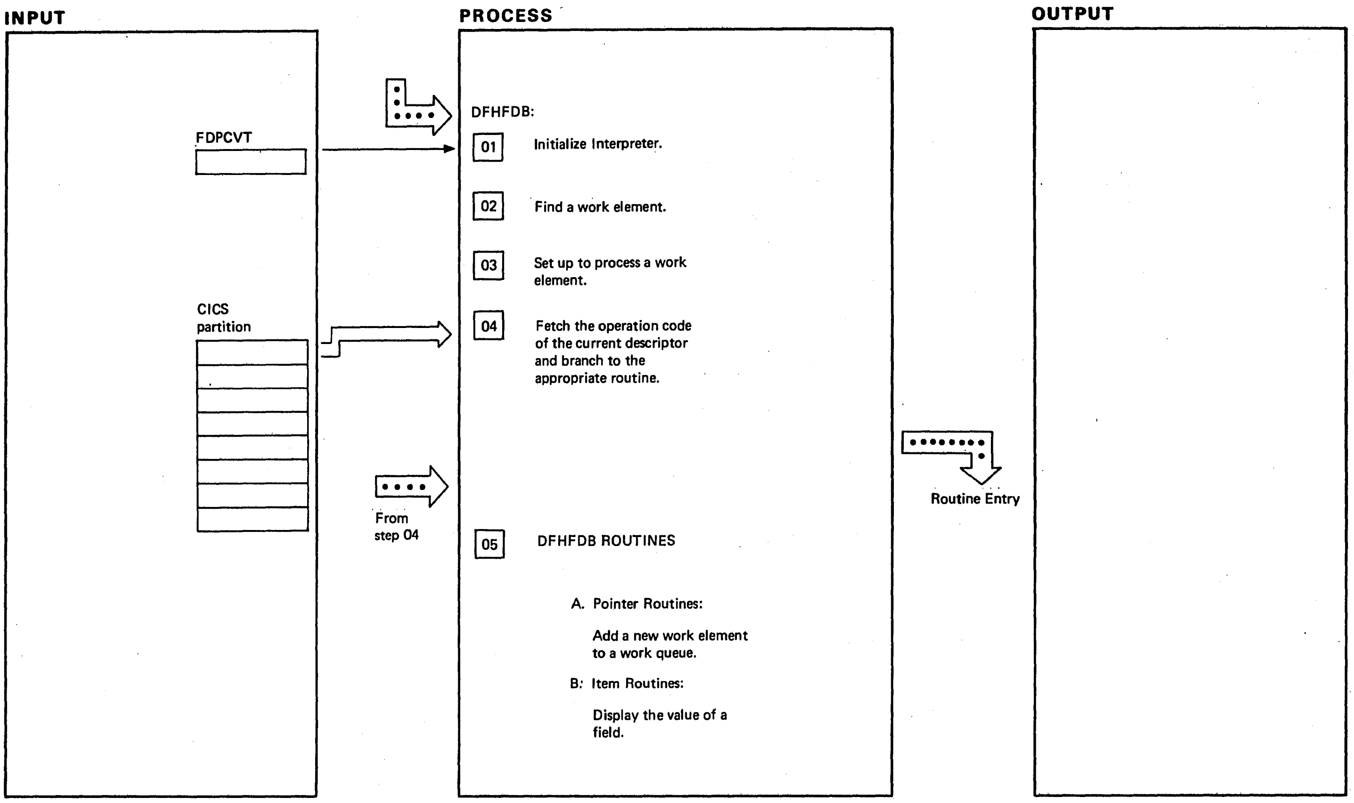
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Formatted Dump Program (DFHFDP)

Diagram - 4.12-02

PAGE 1 OF 2



DFHFDB:

01 Initialize Interpreter.

02 Find a work element.

03 Set up to process a work element.

04 Fetch the operation code of the current descriptor and branch to the appropriate routine.

05 DFHFDB ROUTINES

A. Pointer Routines:

Add a new work element to a work queue.

B: Item Routines:

Display the value of a field.

Formatted Dump Program (DFHFDP)

Diagram - 4.12.1-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 DFHFDB is an interpreter that processes descriptor tables contained in DFHFDC. | | | |
| 02 There is a work element for every instance of a control block. A work element contains the address of its control block and a pointer to the appropriate descriptor table in DFHFDC. The work elements are chained in work queues. | | | |
| 04 Each descriptor begins with a code that indicates which routine in DFHFDB will process it. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Formatted Dump Program (DFHFDP)

Diagram - 4.12.1-01

PAGE 2 OF 2

INPUT

PROCESS

C. Checking Routines:

Check that a field is valid.

D. Print Routine:

Print a hexadecimal dump of a control block.

E. Exit Routine:

De-allocate all work elements and return to caller.

OUTPUT

Formatted Dump Program (DFHFDP)

Diagram - 4.12.1-02

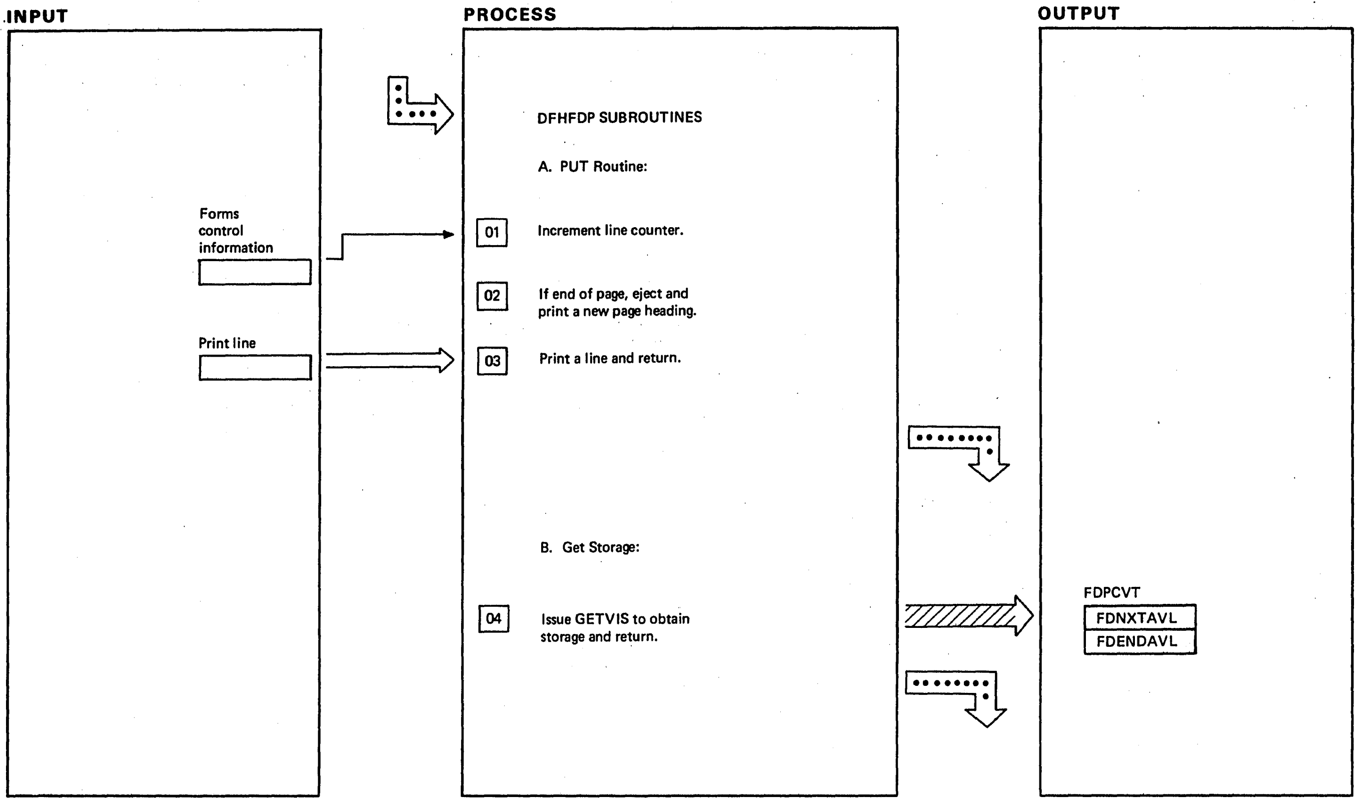
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Formatted Dump Program (DFHFDP)

Diagram - 4.12.1-02

PAGE 1 OF 2



INPUT

Forms control information

Print line

PROCESS

DFHFDP SUBROUTINES

A. PUT Routine:

01 Increment line counter.

02 If end of page, eject and print a new page heading.

03 Print a line and return.

B. Get Storage:

04 Issue GETVIS to obtain storage and return.

OUTPUT

Formatted Dump Program (DFHFDP)

Diagram - 4.12.2-01

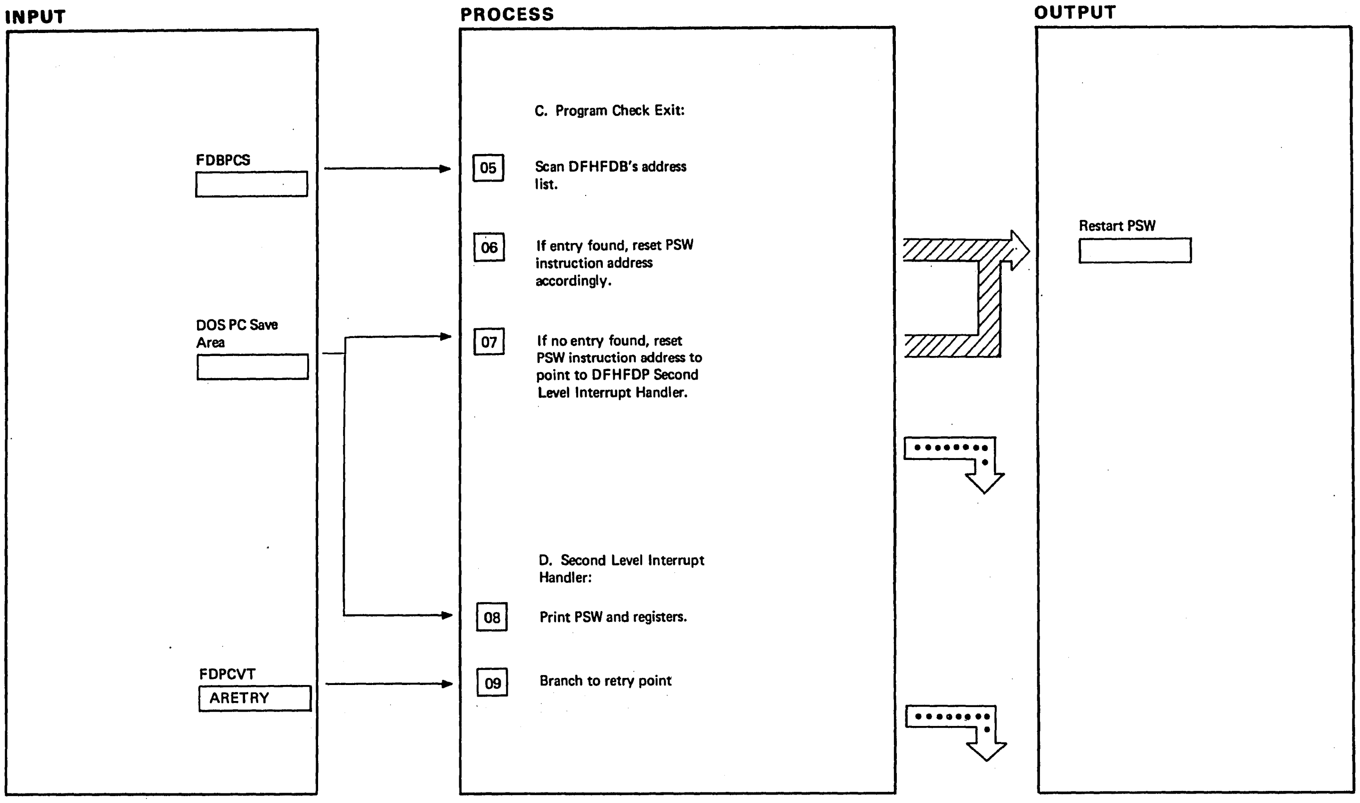
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The control information supplied by the caller contains the carriage control character to be used, the space used by this line of print, and the space required between this line and the bottom of the page. | | | |
| 02 Place beginning and end addresses of area obtained in the DFHFDP communications area. | | | |
| 03 DFHFDB has a list of pairs of addresses, each pair being a possible program check location and its corresponding restart point. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Formatted Dump Program (DFHFDP)

Diagram - 4.12.2-01

PAGE 2 OF 2



C. Program Check Exit:

05 Scan DFHFDB's address list.

06 If entry found, reset PSW instruction address accordingly.

07 If no entry found, reset PSW instruction address to point to DFHFDP Second Level Interrupt Handler.

D. Second Level Interrupt Handler:

08 Print PSW and registers.

09 Branch to retry point

Formatted Dump Program (DFHFDP)

Diagram - 4.12.2-02

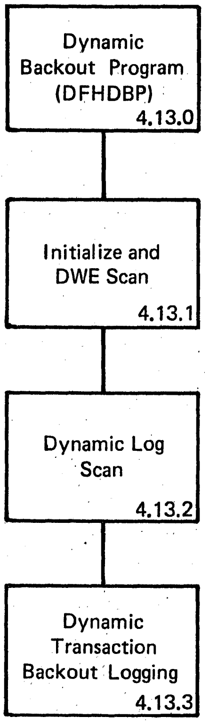
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Formatted Dump Program (DFHFDP)

Diagram - 4.12.2-02

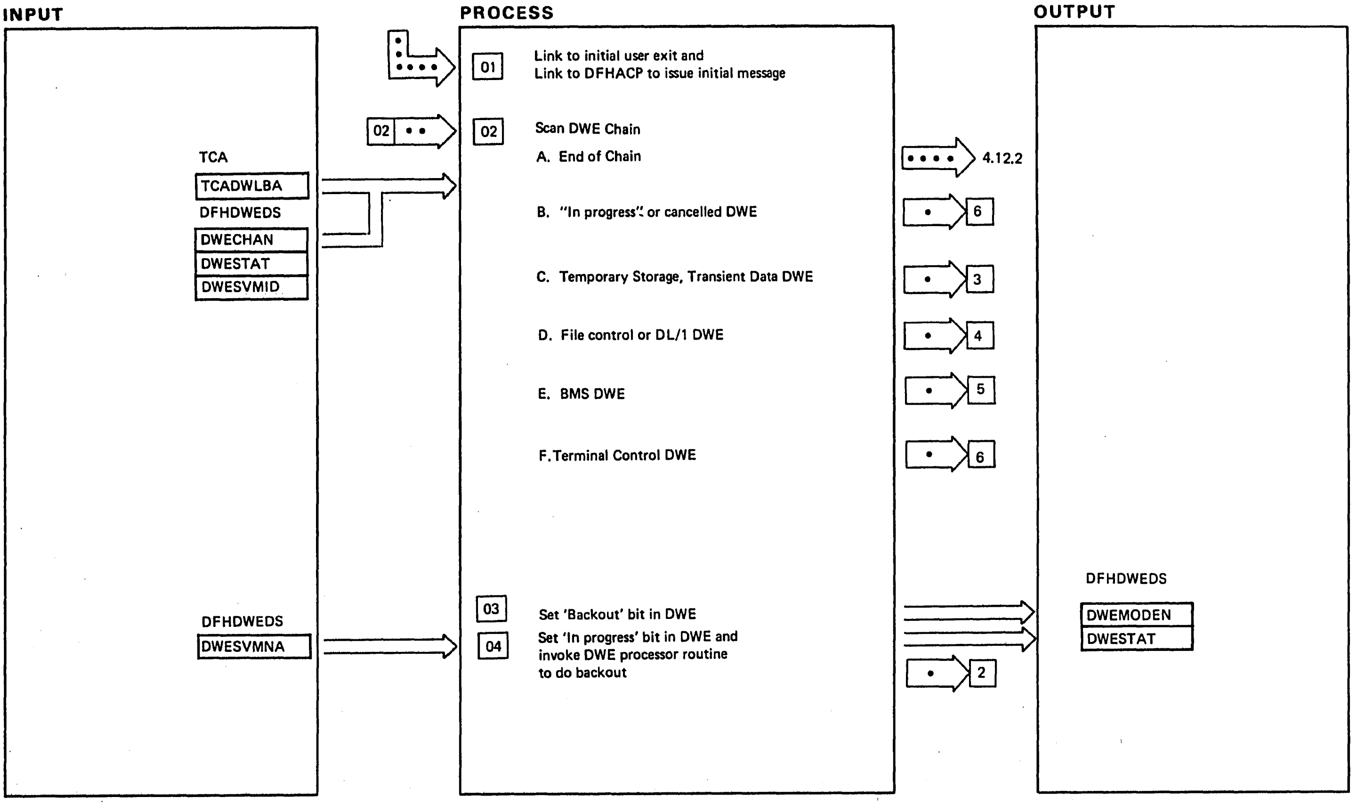
Page 1 of 1


Dynamic Backout Program (DFHDBP)
4.13.0
Initialize and DWE Scan
4.13.1
Dynamic Log Scan
4.13.2
Dynamic Transaction Backout Logging
4.13.3

Visual Index — Dynamic Backout Program

Diagram — 4.13.0-01

Page 1 of 2


INPUT
PROCESS
OUTPUT
01
Link to initial user exit and
Link to DFHACP to issue initial message
02
02
Scan DWE Chain
A. End of Chain
4.12.2
B. "In progress" or cancelled DWE
6
C. Temporary Storage, Transient Data DWE
3
D. File control or DL/1 DWE
4
E. BMS DWE
5
F.Terminal Control DWE
6
TCA
TCADWLBA
DFHDWEDS
DWECHAN
DWESTAT
DWESVMID
03
Set 'Backout' bit in DWE
04
Set 'In progress' bit in DWE and
invoke DWE processor routine
to do backout
2
DFHDWEDS
DWESVMNA
DFHDWEDS
DWEMODEN
DWESTAT

Initialise and DWE scan (DFHDBP)

Diagram — 4.13.1-01

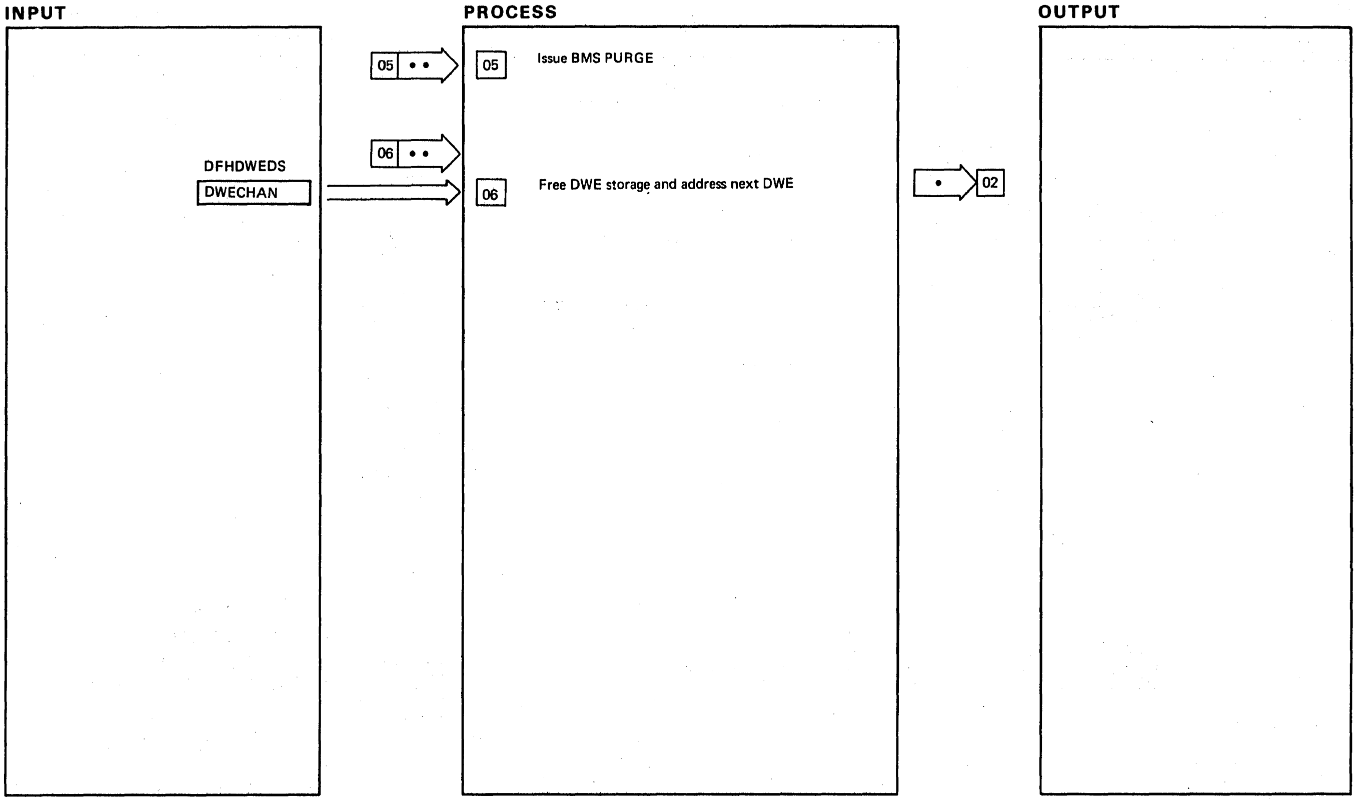
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Message DFH2205. DFHACP is used to write the message because it has the logic required to write to the user's Terminal. | | | |
| 04 The DWE processor routine is addressed from the DWE. This is used as a con—venient mechanism for accessing the backout code in DFHTSP and DFHTDP. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Initialise and DWE scan (DFHDBP)

Diagram — 4.13.1-01

Page 2 of 2



05 Issue BMS PURGE

06 Free DWE storage and address next DWE

Initialise and DWE scan (DFHDBP)

Diagram — 4.13.1-02

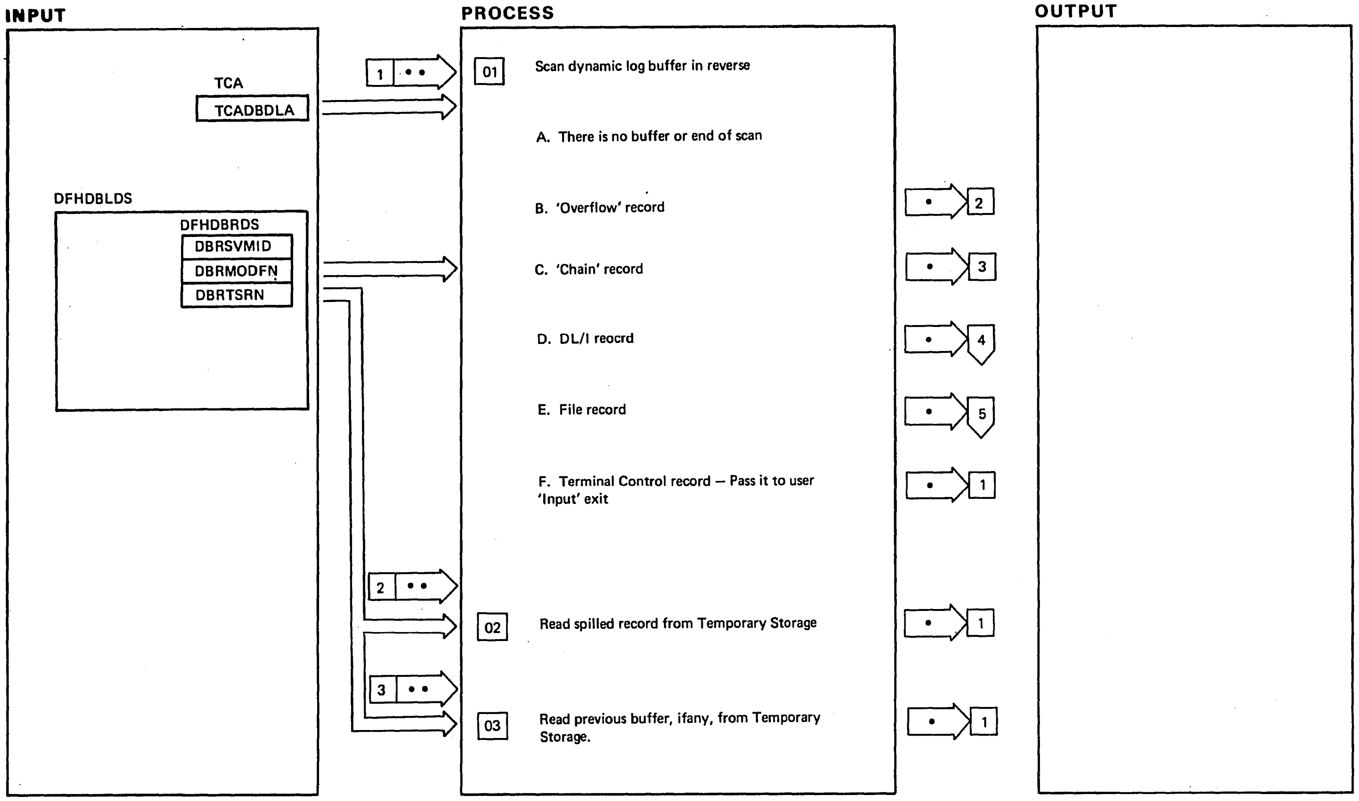
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 05 Clears out any partially built BMS message — Prevents it being written at th end of the transaction. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Initialise and DWE scan (DFHDBP)

Diagram — 4.13.1-02

Page 1 of 3



01 Scan dynamic log buffer in reverse

A. There is no buffer or end of scan

B. 'Overflow' record

C. 'Chain' record

D. DL/I reocrd

E. File record

F. Terminal Control record — Pass it to user 'Input' exit

02 Read spilled record from Temporary Storage

03 Read previous buffer, ifany, from Temporary Storage.

Dynamic Log scan (DFHDBP)

Diagram — 4.13.2-01

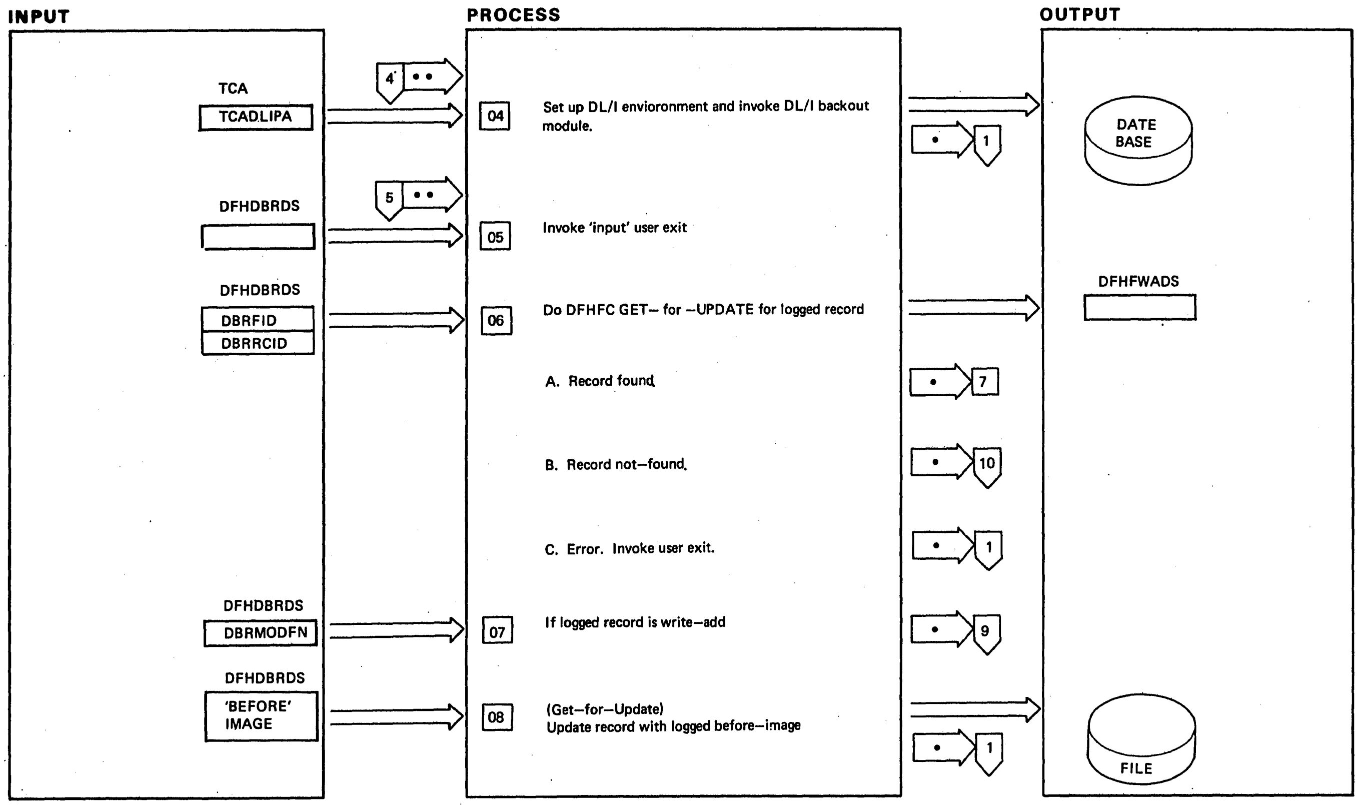
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 & 03 TS queue name is 'DTB' prefixed by X 'FF' and suffixed by the task number | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Dynamic Log scan (DFHDBP)

Diagram — 4.13.2-01

Page 2 of 3



Dynamic Log scan (DFHDBP)

Diagram – 4.13.2-02

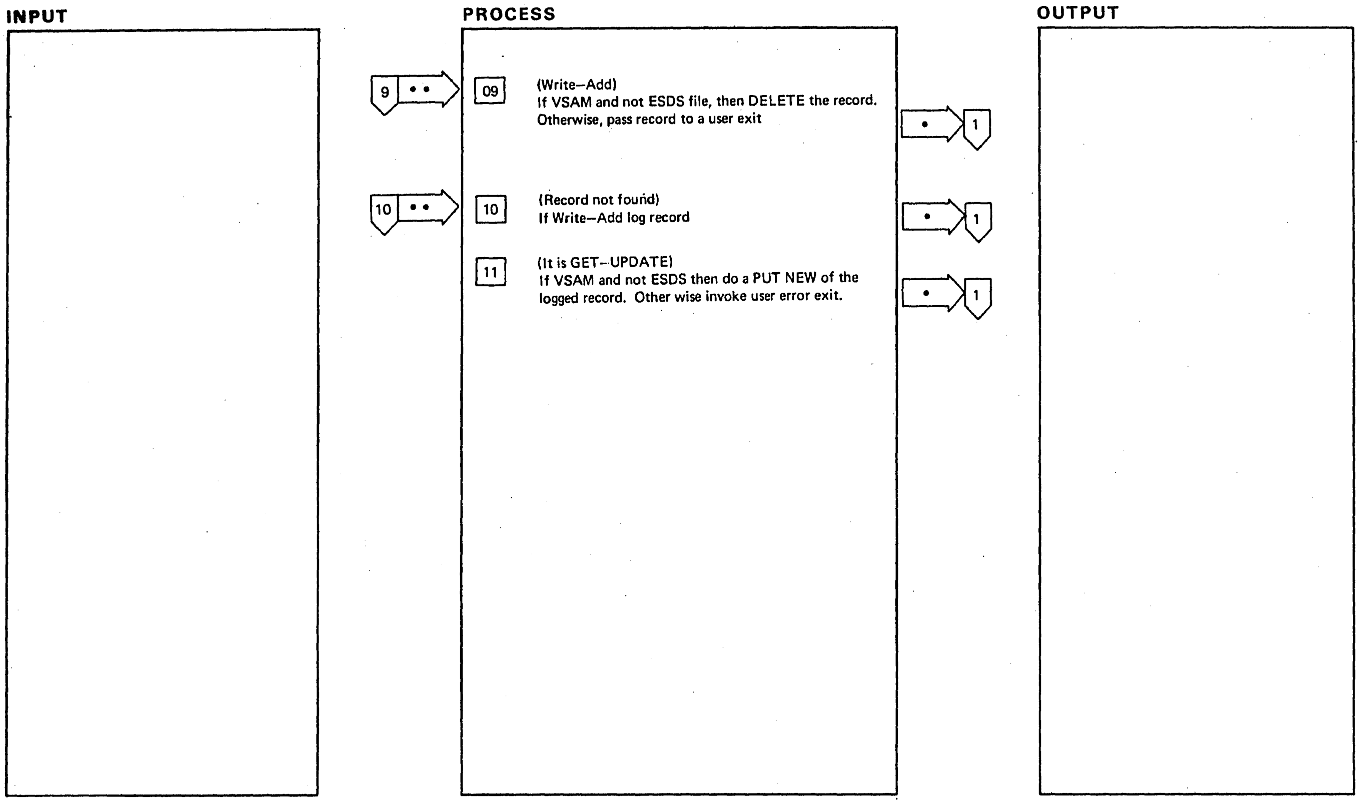
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 04 DLZRDBCO | | | |
| 07 Only WRITE–ADD'S or get for UPDATE'S are logged | | | |
| 08 Errors are passed to a user error exit. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Dynamic Log scan (DFHDBP)

Diagram – 4.13.2-02

Page 3 of 3



09 (Write–Add)
If VSAM and not ESDS file, then DELETE the record.
Otherwise, pass record to a user exit

10 (Record not found)
If Write–Add log record

11 (It is GET–UPDATE)
If VSAM and not ESDS then do a PUT NEW of the logged record. Other wise invoke user error exit.

Dynamic Log scan (DFHDBP)

Diagram – 4.13.2-03

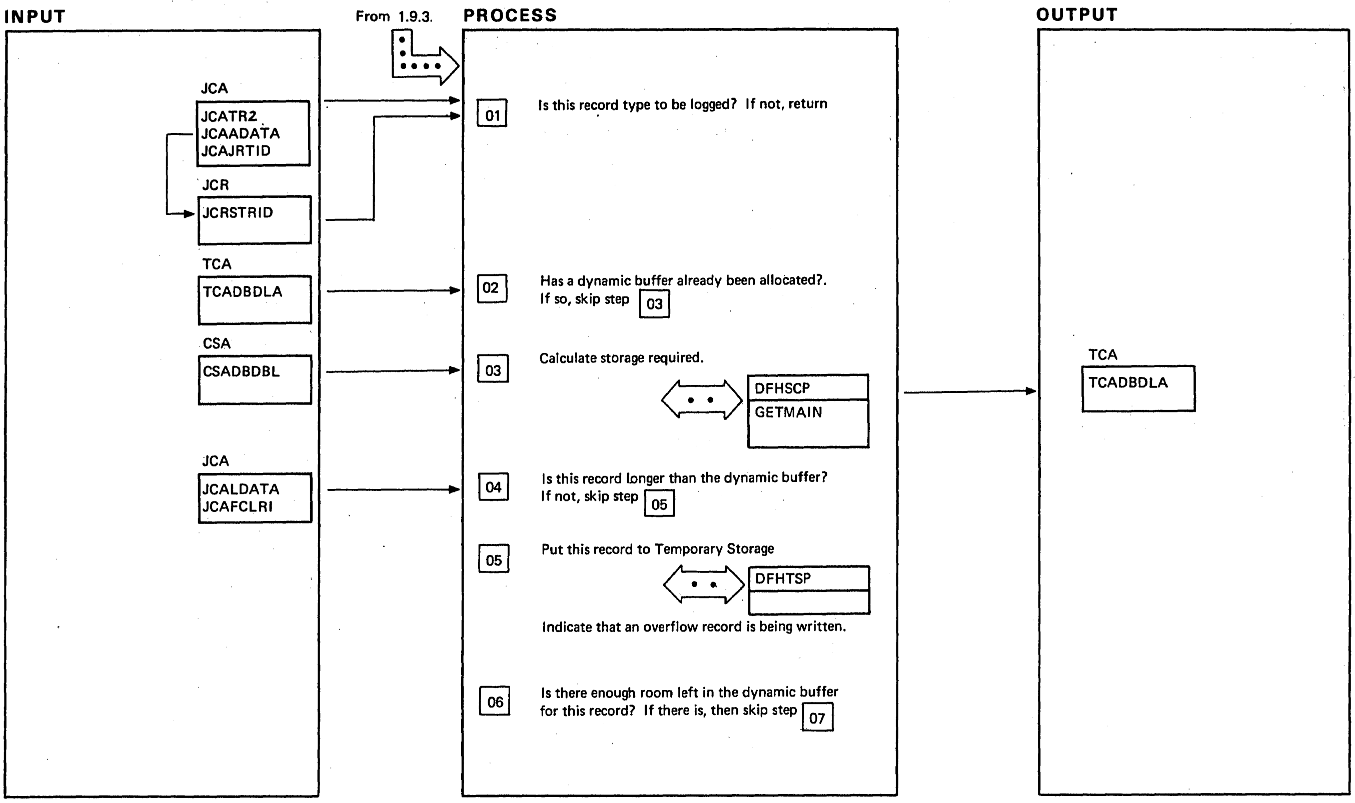
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Dynamic Log scan (DFHDBP)

Diagram – 4.13.2-03

Page 1 of 2


INPUT
From 1.9.3.
PROCESS
OUTPUT
JCA
JCATR2
JCAADATA
JCAJRTID
JCR
JCRSTRID
TCA
TCADBDLA
CSA
CSADBDBL
JCA
JCALDATA
JCAFCLRI
01
Is this record type to be logged? If not, return
02
Has a dynamic buffer already been allocated?. If so, skip step 03
03
Calculate storage required.
DFHSCP
GETMAIN
04
Is this record longer than the dynamic buffer? If not, skip step 05
05
Put this record to Temporary Storage
DFHTSP
Indicate that an overflow record is being written.
06
Is there enough room left in the dynamic buffer for this record? If there is, then skip step 07
TCA
TCADBDLA

Dynamic Transaction Backout Logging

Diagram — 4.13.3-01

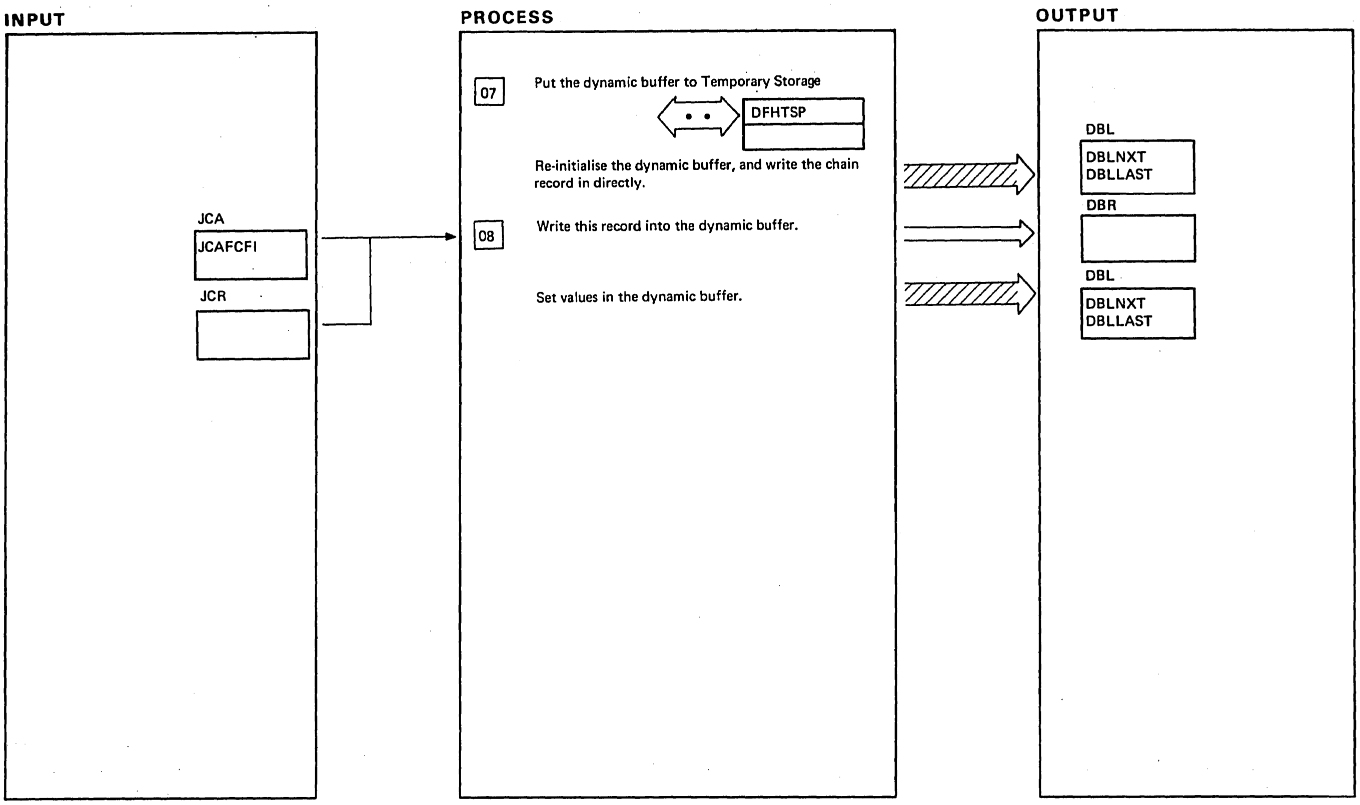
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Types of record to be logged:- DL/I type 50-52 (OS)<br>50-51 (DOS)<br>Terminal Control first input message.<br>File control Get—Update,<br>Write—Add. | | | |
| 05 The Temporary Storage queue name is constructed as: X'FF'//DTB//Task ID. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Dynamic Transaction Backout Logging

Diagram — 4.13.3-01

Page 2 of 2



07 Put the dynamic buffer to Temporary Storage

Re-initialise the dynamic buffer, and write the chain record in directly.

08 Write this record into the dynamic buffer.

Set values in the dynamic buffer.

Dynamic Transaction Backout Logging

Diagram – 4.13.3-02

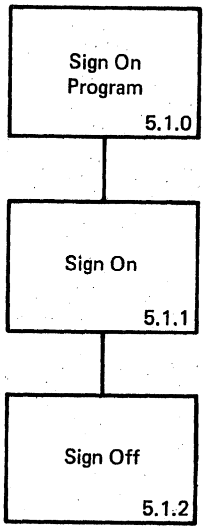
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 07 The Temporary Storage queue-names is the same as for step. | | | |
| 08 For file control records, the file-id as well as the record data must be logged. For other records, just the data is logged. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Dynamic Transaction Backout Logging

Diagram – 4.13.3-02

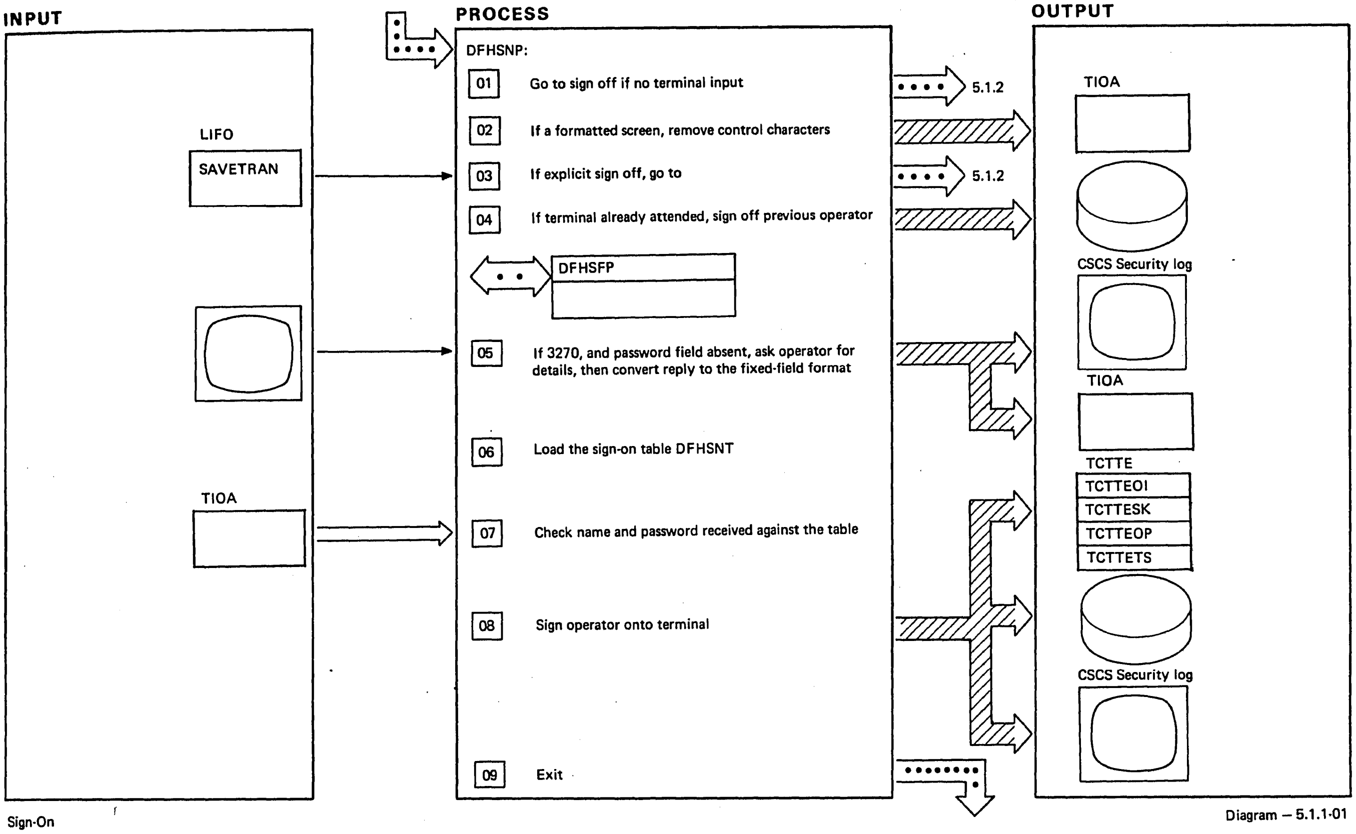
Page 1 of 1



Visual Index — Sign On Program

Diagram — 5.1.0-01

Page 1 of 1



INPUT

LIFO

SAVETRAN

TIOA

PROCESS

DFHSNP:

01 Go to sign off if no terminal input

02 If a formatted screen, remove control characters

03 If explicit sign off, go to

04 If terminal already attended, sign off previous operator

DFHSFP

05 If 3270, and password field absent, ask operator for details, then convert reply to the fixed-field format

06 Load the sign-on table DFHSNT

07 Check name and password received against the table

08 Sign operator onto terminal

09 Exit

OUTPUT

TIOA

CSCS Security log

TIOA

TCTTE

TCTTEOI

TCTTESK

TCTTEOP

TCTTETS

CSCS Security log

Sign-On

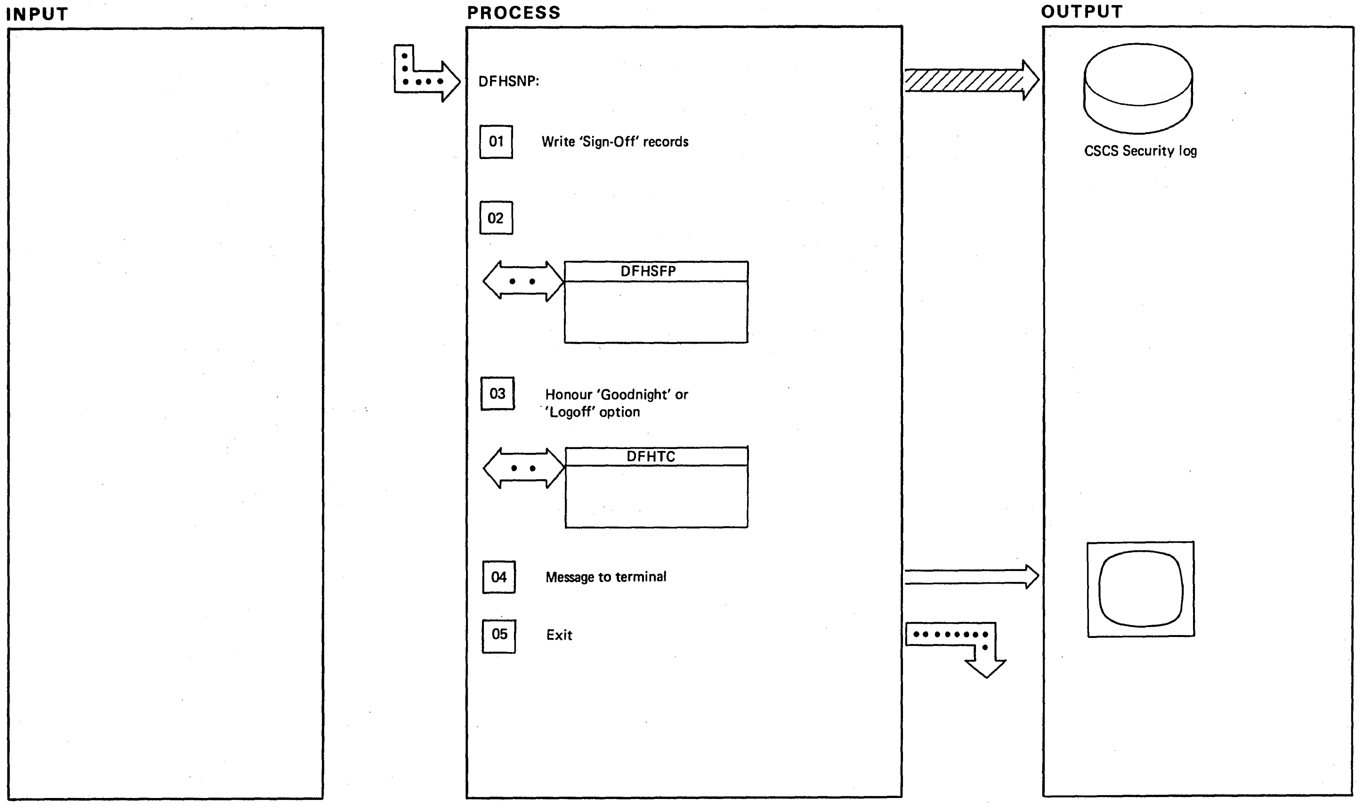
Diagram — 5.1.1-01

| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 03 | Transactor code of CSSF or 8888, obtained from TCAKOID | | CSSNNFT | |
| 04 | Link to sign-off program | DFHSFP | | |
| 05 | The screen format requests operator name in one field, and password in another, non-display, field | | | |
| 06 | Abandon processing with appropriate messages, if NAME field absent or table not found | | CSSNINEB | |
| 07 | In case of error, compose messages to security log (CSCS), terminal operator, and CSML, then exit | | CSSNLSB | |
| 08 | If the security log (CSCS) is not defined in the DCT, all references to it are non-operations | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Sign-On

Diagram — 5.1.1-01



Sign-Off

Diagram – 5.1.2-01

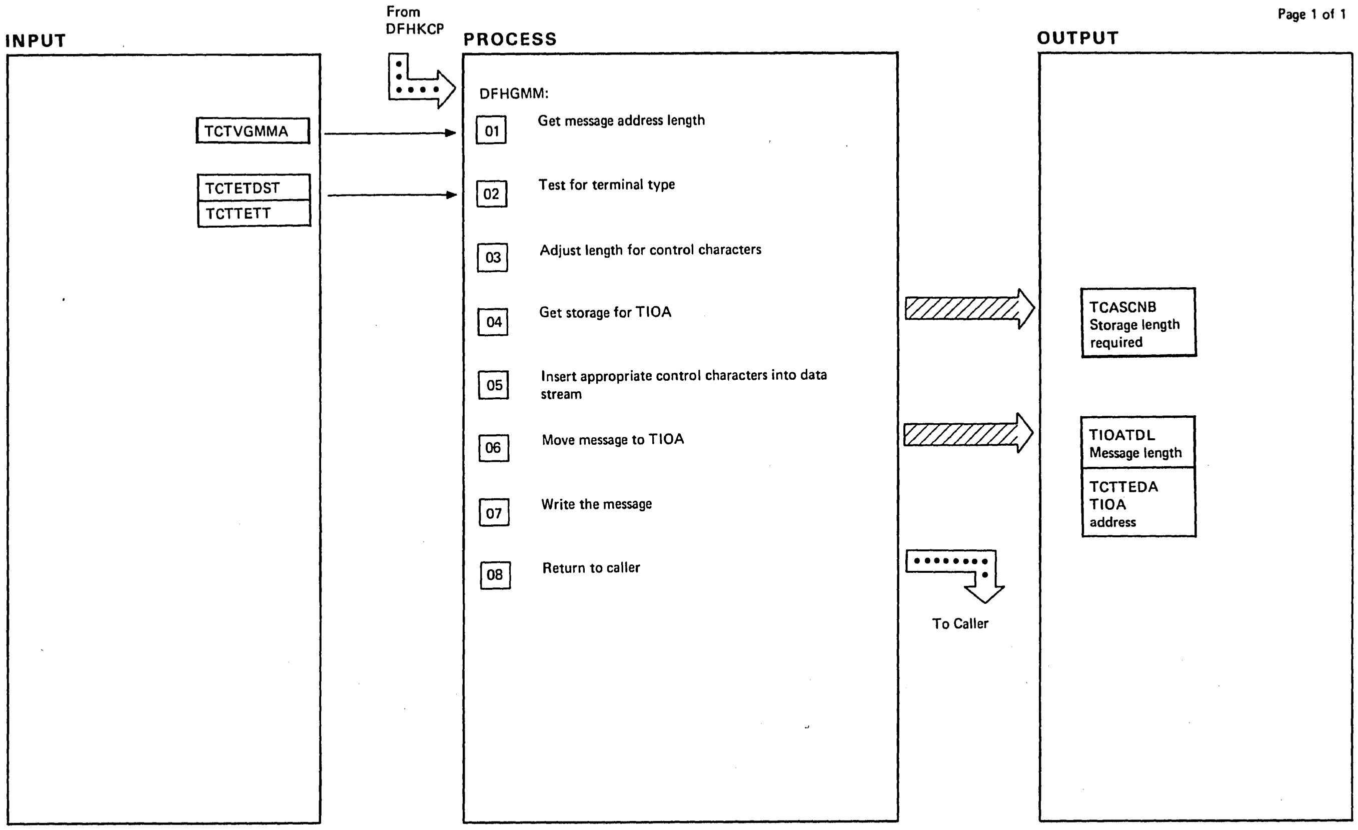
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 | DFHSNP | CSSNPSF | |
| 02 Module DFHSFP writes a message to destination CSML, and resets the status of the TCTTE to 'unattended'. | DFHSFP | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Sign-Off

Diagram – 5.1.2-01

Page 1 of 1



"Good Morning" Message (DFHGMM)

Diagram – 5.1.3-01

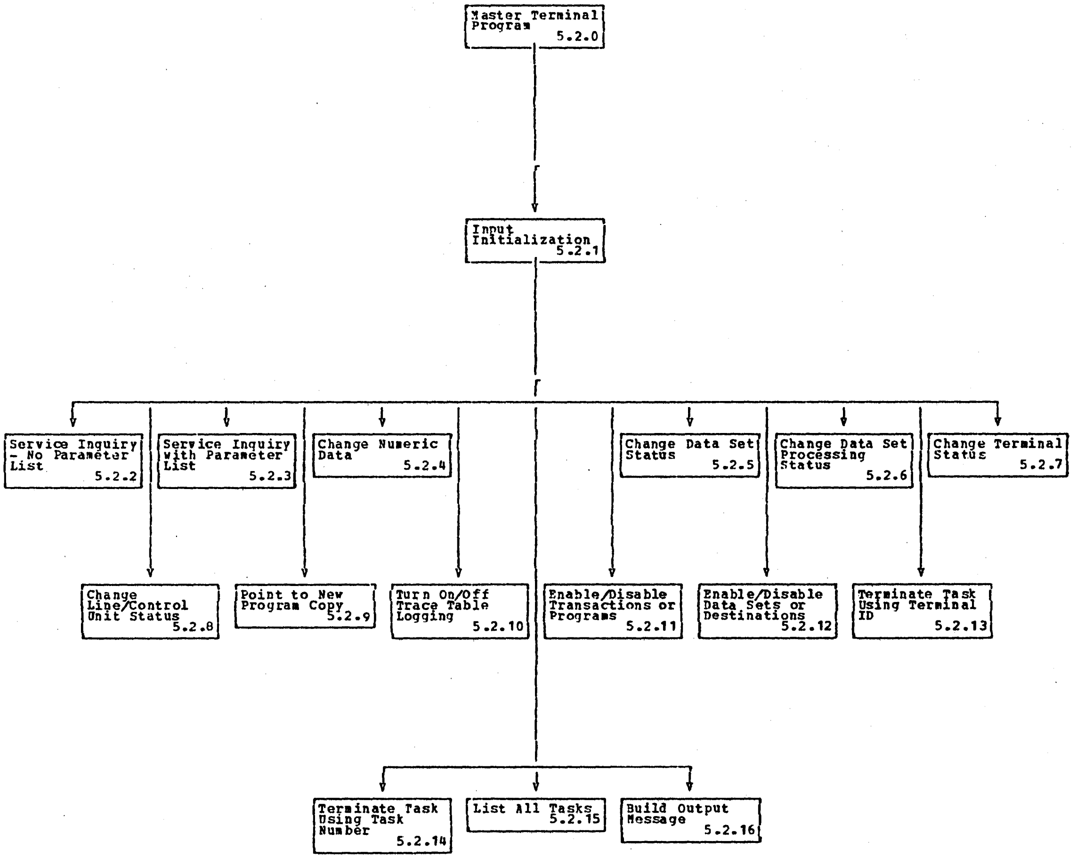
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 Terminal types can be 3270, SCS basic, or 3600 | | | |
| 03 For 3270 and for CICS/VS logo panel | | | |
| 04 GETMAIN using DFHSC | | SCGMSETL | |
| 07 DFHTC WRITE, WAIT in case intervention required | | CSGMWRIT | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

"Good Morning" Message (DFHGMM)

Diagram – 5.1.3-01

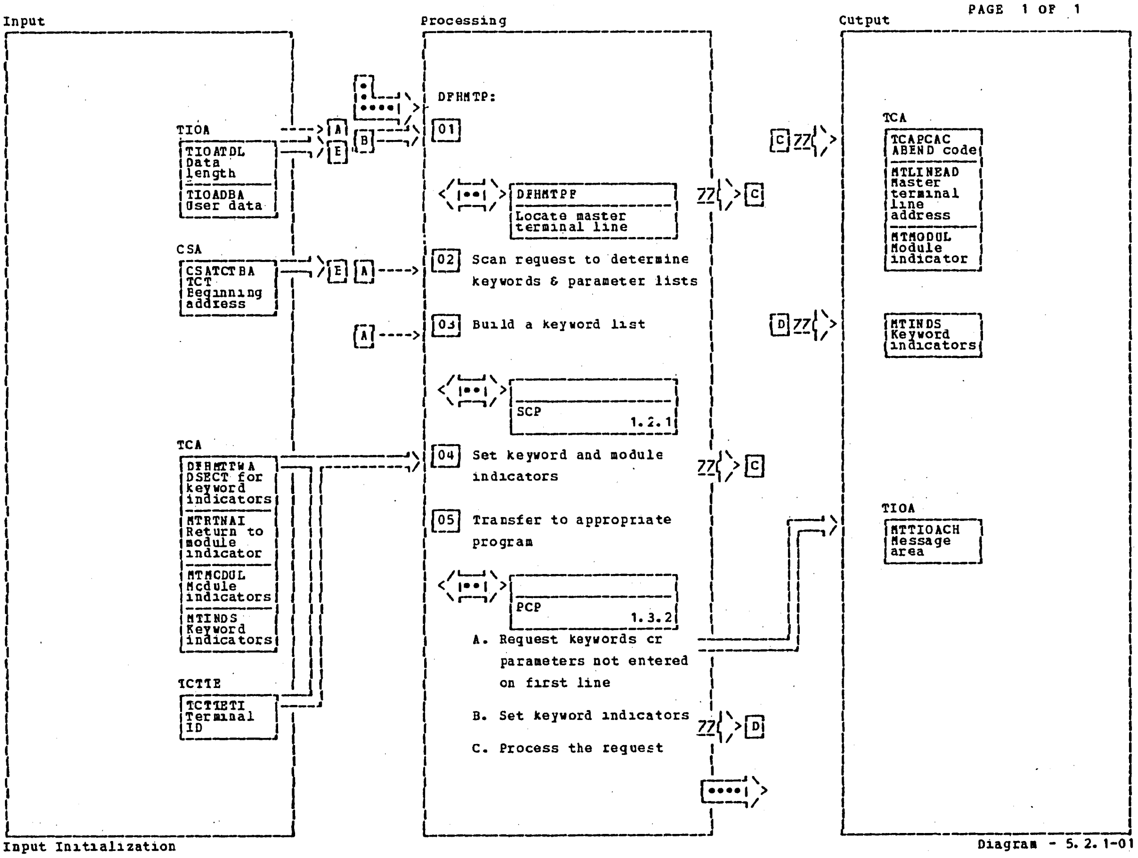
PAGE 1 OF 1


Master Terminal Program 5.2.0
Input Initialization 5.2.1
Service Inquiry - No Parameter List 5.2.2
Service Inquiry with Parameter List 5.2.3
Change Numeric Data 5.2.4
Change Data Set Status 5.2.5
Change Data Set Processing Status 5.2.6
Change Terminal Status 5.2.7
Change Line/Control Unit Status 5.2.8
Point to New Program Copy 5.2.9
Turn On/Off Trace Table Logging 5.2.10
Enable/Disable Transactions or Programs 5.2.11
Enable/Disable Data Sets or Destinations 5.2.12
Terminate Task Using Terminal ID 5.2.13
Terminate Task Using Task Number 5.2.14
List All Tasks 5.2.15
Build Output Message 5.2.16

Master Terminal Program

Diagram - 5.2-01

PAGE 1 OF 1


Input
Processing
Cutput
TIOA
TIOATDL Data length
TIOADBA User data
CSA
CSATCTBA TCT Beginning address
TCA
DFHMTWA DSECT for keyword indicators
MTRTNAI Return to module indicator
MTMODUL Module indicators
MTINDS Keyword indicators
TCTTE
TCTTETI Terminal ID
A
B
E
DFHMTP:
01
DFHMTPF
Locate master terminal line
02 Scan request to determine keywords & parameter lists
03 Build a keyword list
SCP
1.2.1
04 Set keyword and module indicators
05 Transfer to appropriate program
PCP
1.3.2
A. Request keywords or parameters not entered on first line
B. Set keyword indicators
C. Process the request
C
D
TCA
TCAPCAC ABEND code
MTLINEAD Master terminal line address
MTMODUL Module indicator
MTINDS Keyword indicators
TIOA
MTTIOACH Message area

Input Initialization

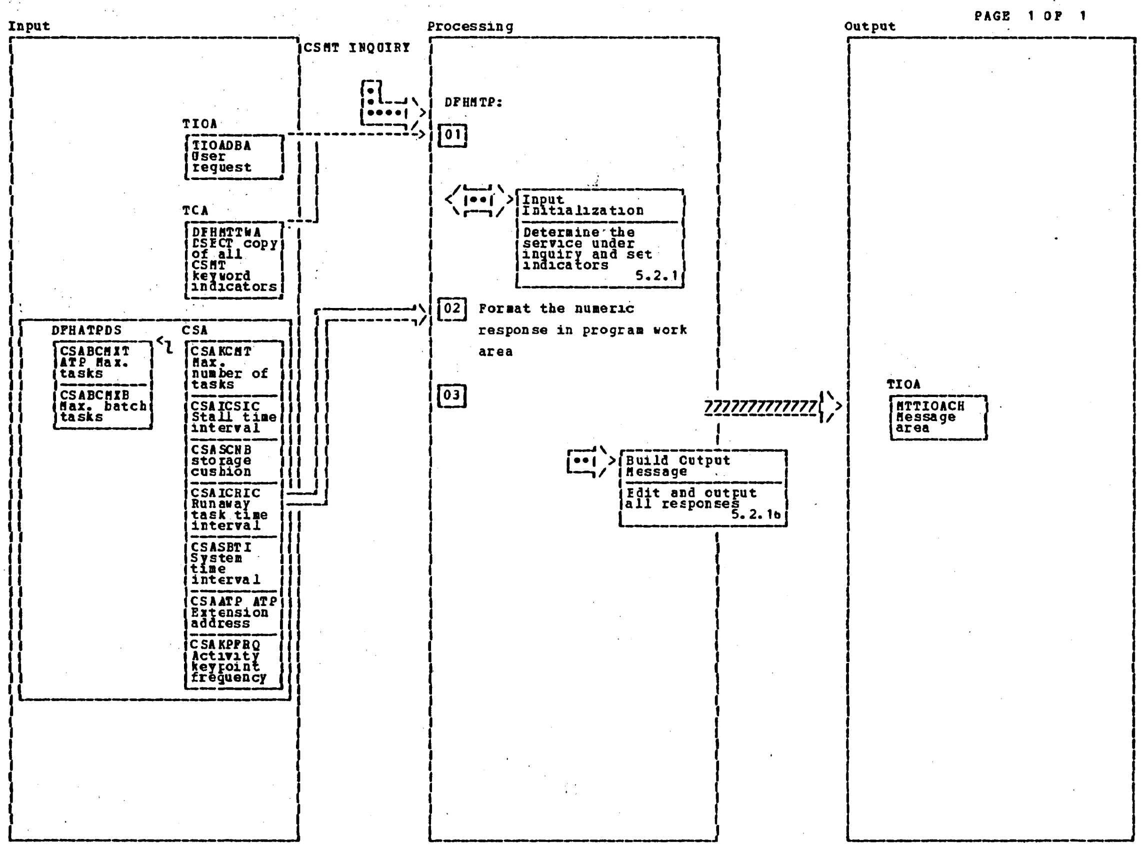
Diagram - 5.2.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 DFHMTPF finds the line address, adjusting for pooled lines. | DFHMTPA<br>DFHMTPF | MTLOADF<br>MTGETMLN | |
| 02 Trailing blanks and 3270 control characters are removed. | DFHMTPA | MTGOTMLN | |
| If CANCEL was requested, the task is terminated. | | | |
| If no keywords were specified, input is requested from the user. | MTWMSGA1 | | |
| To build the keyword list, the input is scanned for new line symbols, numeric parameters, '=', and ','. | DFHMTPA | MTSCNIPT | |
| If a keyword equals parameter was specified, the keyword ID table is searched for a matching entry. | DFHMTPA | MTEQUAL | |
| 03 | DFHMTPA | MTWRDCNA | |
| 04 | DFHMTPA | MTSRCHSK | |
| 05 If no keyword indicator is on, | DFHMTA | DOCMTP01 | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| ABEND code 'AMTA ' is moved to the TCAPCAC field. Otherwise, a DFHPC TYPE=XCTL is issued. | | | |
| A. The module affected requests the missing keywords or parameters. | All DFHMTP modules | | |
| B. The requesting modules set the indicators. | All DFHMTP modules | | |
| C. See chart 5.2 | | | |

Input Initialization

Diagram - 5.2.1-01


PAGE 1 OF 1
Input
CSMT INQUIRY
TIOA
TIOADBA User request
TCA
DFHMTTWA CSECT copy of all CSMT keyword indicators
DFHATPDS
CSABCMXT ATP Max. tasks
CSABCMXB Max. batch tasks
CSA
CSAKCMT Max. number of tasks
CSAICSIC Stall time interval
CSASCNB storage cushion
CSAICRIC Runaway task time interval
CSASBTI System time interval
CSAATP ATP Extension address
CSAKPFRQ Activity keypoint frequency
Processing
DFHMTP:
01
Input Initialization
Determine the service under inquiry and set indicators
5.2.1
02
Format the numeric response in program work area
03
Build Output Message
Edit and output all responses
5.2.16
Output
TIOA
MTTIOACH Message area

Service Inquiry - No Parameter List

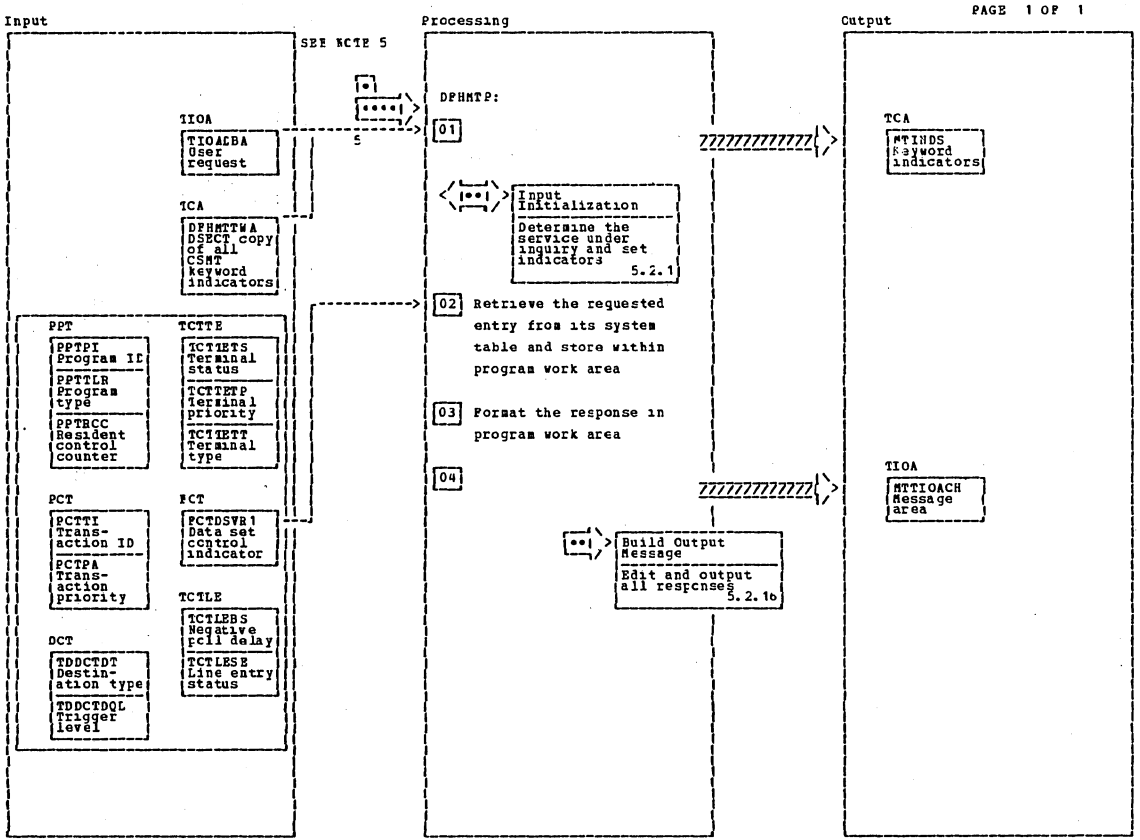
Diagram - 5.2.2-01

| Notes | Rottine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | | | | | | |
| 02 RNAWAY | DFHMTPA | DOCMTP03 | | | | | |
| STALL | DFHMTPE | DOCMTP45 | | | | | |
| TIME | DFHMTPA | MTTIMERT | | | | | |
| ATPMXT | DFHMTPB | MTATBMTR | | | | | |
| BMAXT | DFHMTPB | MTATBMTR | | | | | |
| CUSH | DFHMTPB | DOCMTP14 | | | | | |
| MAXT | DFHMTPB | DOCMTP15 | | | | | |
| AKPFRQ | DFHMTA | MTAKP00 | | | | | |
| 03 The leading zeros are suppressed and at least one digit is outputed. | All DFHMTP modules | MTMMDC | 5.2.16 | | | | |

Service Inquiry - No Parameter List

Diagram - 5.2.2-01

PAGE 1 OF 1


Input
Processing
Cutput
SEE NCTE 5
TIOA
TIOALBA User request
TCA
DFHMTTWA DSECT copy of all CSMT keyword indicators
PPT
PPTPI Program ID
PPTTLR Program type
PPTRCC Resident control counter
TCTTE
TCTTETS Terminal status
TCTTETP Terminal priority
TCTTETT Terminal type
PCT
PCTTI Trans-action ID
PCTPA Trans-action priority
FCT
FCTDSVR1 Data set control indicator
TCTLE
TCTLEBS Negative pcll delay
TCTLESE Line entry status
DCT
TDDCTDT Destin-ation type
TDDCTDQL Trigger level
5
DFHMTP:
01
Input Initialization
Determine the service under inquiry and set indicators 5.2.1
02 Retrieve the requested entry from its system table and store within program work area
03 Format the response in program work area
04
Build Output Message
Edit and output all respcnses 5.2.16
TCA
MTINDS Keyword indicators
TIOA
MTTIOACH Message area

Service Inquiry - With Parameter List

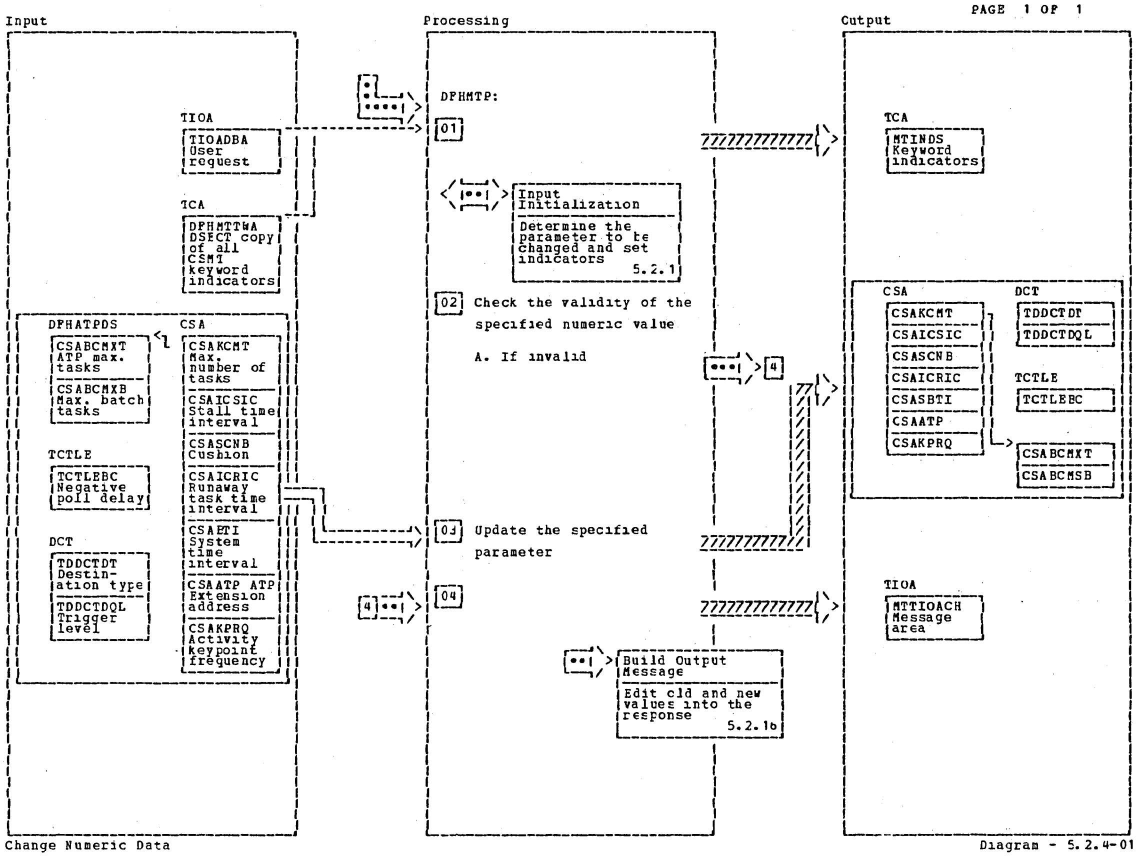
Diagram - 5.2.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 02 Possible inquiries and their tables are: | | | |
| TRIGER-DCT | CFEMTPE | DOCMTP43 | |
| TERMNL-TCT | DPHMTPC | DOCMTP27 | |
| DATBAS-FCT | DPHMTPB | DOCMTP18 | |
| TRNACT-PCT | CFHMTPG | MTTRANRT | |
| CNTRL-TCT | CFHMTPF | DOCMTP50 | |
| PRIOR-PCT | DPHMTPE | MTTRANRT | |
| PRIOR-TCT | CFHMTPC | DOCMTP27 | |
| NEGP-TCT | DPHMTPE | DOCMTP16 | |
| LINE-TCT | DPHMTPF | DOCMTP49 | |
| PGRM-PPT | CFEMTPE | DOCMTP44 | |
| 03 NEGP, TRIGER, and PRIORITY generate a decimal response. | | | |
| 04 In numeric responses, the leading zeros are suppressed. | All CFHMTP modules | MTMMDC | 5.2.16 |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 05 An example of a service inquiry with a parameter list is:<br>CSMT DATABAS,INQ,FILEID= | | | |

Service Inquiry - With Parameter List

Diagram - 5.2.3-01


PAGE 1 OF 1
Input
Processing
Cutput
TIOA
TIOADBA User request
TCA
DPHMTTWA DSECT copy of all CSMT keyword indicators
DFHATPDS
CSABCMXT ATP max. tasks
CSABCMXB Max. batch tasks
TCTLE
TCTLEBC Negative poll delay
DCT
TDDCTDT Destin-ation type
TDDCTDQL Trigger level
CSA
CSAKCMT Max. number of tasks
CSAICSIC Stall time interval
CSASCNB Cushion
CSAICRIC Runaway task time interval
CSAETI System time interval
CSAATP ATP Extension address
CSAKPRQ Activity keypoint frequency
DFHMTP:
01
Input Initialization
Determine the parameter to be changed and set indicators 5.2.1
02 Check the validity of the specified numeric value
A. If invalid
4
03 Update the specified parameter
4
04
Build Output Message
Edit cld and new values into the response 5.2.16
TCA
MTINDS Keyword indicators
CSA
CSAKCMT
CSAICSIC
CSASCNB
CSAICRIC
CSASBTI
CSAATP
CSAKPRQ
DCT
TDDCTDT
TDDCTDQL
TCTLE
TCTLEBC
CSABCMXT
CSABCMSB
TIOA
MTTIOACH Message area

Change Numeric Data

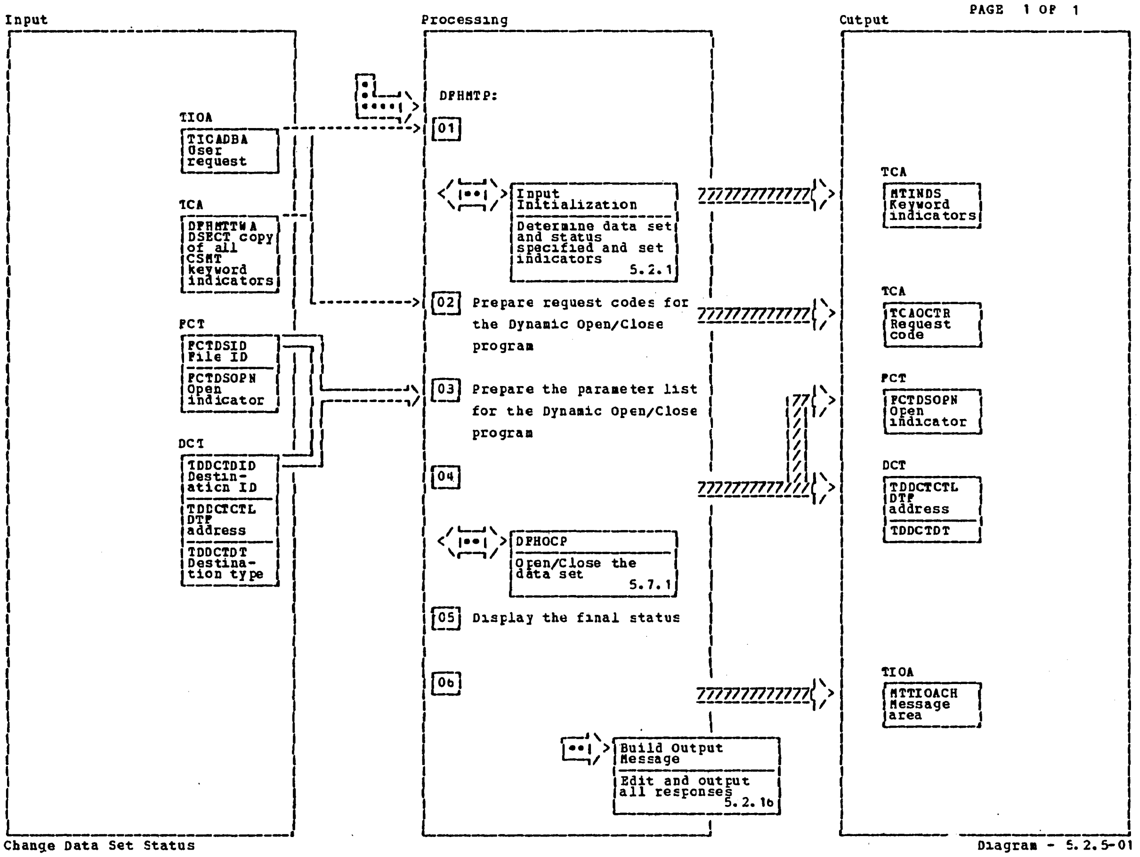
Diagram - 5.2.4-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | | |
| 02 The range cf allowable values for the specified parameter is: | | | |
| RNAWAY: Time value to 27962020 | CFHMTPA | DOCMTP03 | |
| STALL: Time value to 32767 | CFHMTPE | DOCMTP45 | |
| TIME: 100 to 27962020 | DFHMTPA | MTTIMERT | |
| ATPMXT: less than cr equal BMAXT | CFHMTPE | MTATBMTR | |
| BMAXT: less than or equal MAXT or ATPMXT | CFHMTPB | MTATBMTR | |
| CUSH: 20 to 65535 | DFHMTPE | DOCMTP14 | |
| MAXT: 2 to 999 | CFHMTPB | DOCMTP15 | |
| CLASS MAXT: 1 to 999 | | | |
| ACTIVE MAXT: 1 to 999 | | | |
| TRIGER: 0 to 255 | CFEMTPE | DOCMTP43 | |
| NEGP: less than or equal 2000 | CFEMTPB | DOCMTP16 | |
| AKPFRQ: 200 to 65535 | DFHMTA | MTAKPF00 | |
| IOCP: 0 to 50 | | | |
| 03 | All DFHMTP modules | MTMMDC | 5.2.16 |

Change Numeric Data

Diagram - 5.2.4-01

PAGE 1 OF 1


Input
Processing
Cutput
DFHMTP:
TIOA
TICADBA User request
TCA
DFHMTTWA DSECT copy of all CSMT keyword indicators
PCT
FCTDSID File ID
FCTDSOPN Open indicator
DCT
TDDCTDID Destination ID
TDDCTCTL DTF address
TDDCTDT Destination type
01
Input Initialization
Determine data set and status specified and set indicators 5.2.1
02 Prepare request codes for the Dynamic Open/Close program
03 Prepare the parameter list for the Dynamic Open/Close program
04
DFHOCP
Open/Close the data set 5.7.1
05 Display the final status
06
Build Output Message
Edit and output all responses 5.2.16
TCA
MTINDS Keyword indicators
TCA
TCAOCTR Request code
FCT
FCTDSOPN Open indicator
DCT
TDDCTCTL DTF address
TDDCTDT
TIOA
MTTIOACH Message area

Change Data Set Status

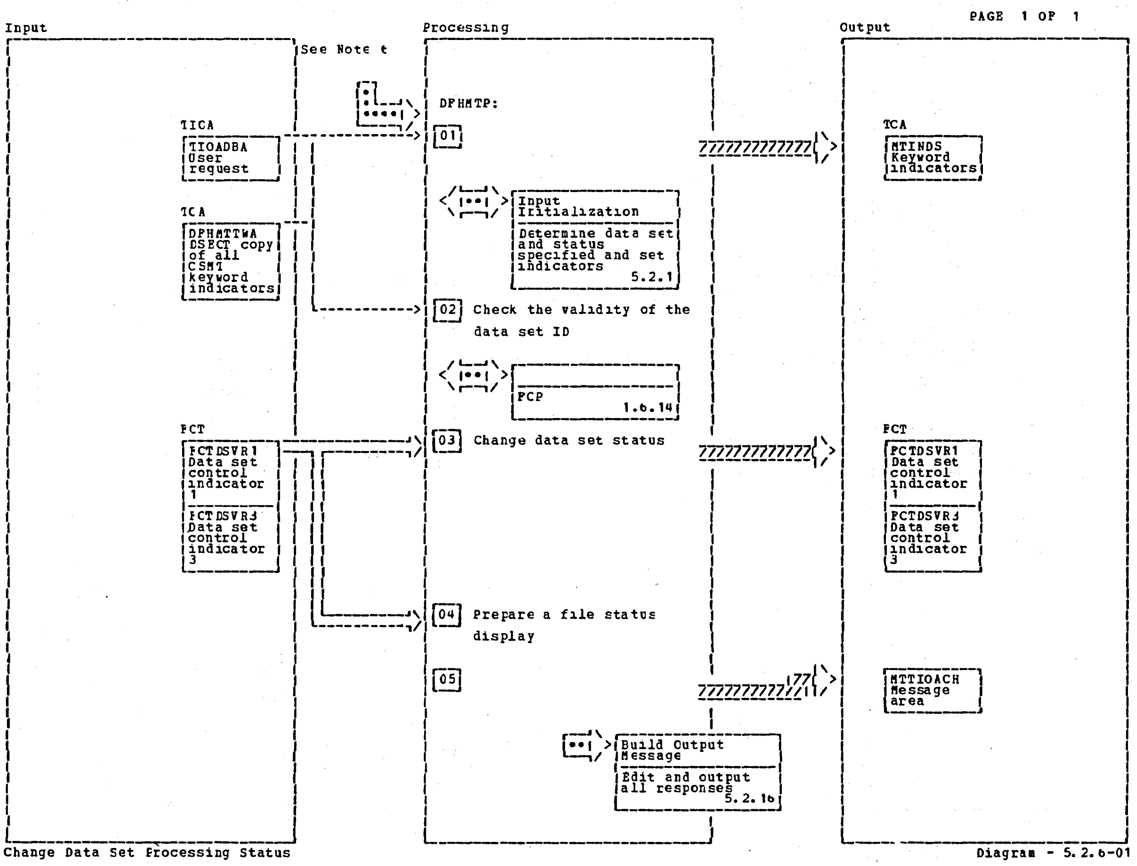
Diagram - 5.2.5-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 02 Possible statuses for data sets are OPEN and CLOSE | CFHMTPD | DCOMTP33 | |
| Possible statuses for Dump data sets are OPEN, CLOSE, and SWITCH. | DFHMTPD | DCOMTP37 | |
| 03 | CFHMTPD | DCOMTP34 | |
| 04 For extrapartition destinations, the first byte of TDDCTCTL is set to Open/Close. | CFHMTPD | DOCMTP39 | 5.7.1 |
| 05 Extrapartition destination status change. | DFHMTPD | MTTDLUP1 | |
| Data base status change. | DFHMTPB | MTFILDSX | |
| 06 Extrapartition destination status change | CFHMTPD | MTMMDC | 5.2.14 |
| Data base status change. | DFHMTPB | MTMMDC | 5.2.14 |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| | | | |

Change Data Set Status

Diagram - 5.2.5-01


PAGE 1 OF 1
Input
Processing
Output
See Note 6
TICA
TIOADBA User request
TCA
DFHMTTWA DSECT copy of all CSMT keyword indicators
DFHMTP:
01
Input Initialization
Determine data set and status specified and set indicators
5.2.1
02 Check the validity of the data set ID
FCP
1.6.14
FCT
FCTDSVR1 Data set control indicator 1
FCTDSVR3 Data set control indicator 3
03 Change data set status
04 Prepare a file status display
05
Build Output Message
Edit and output all responses
5.2.16
TCA
MTINDS Keyword indicators
FCT
FCTDSVR1 Data set control indicator 1
FCTDSVR3 Data set control indicator 3
MTTIOACH Message area

Change Data Set Processing Status

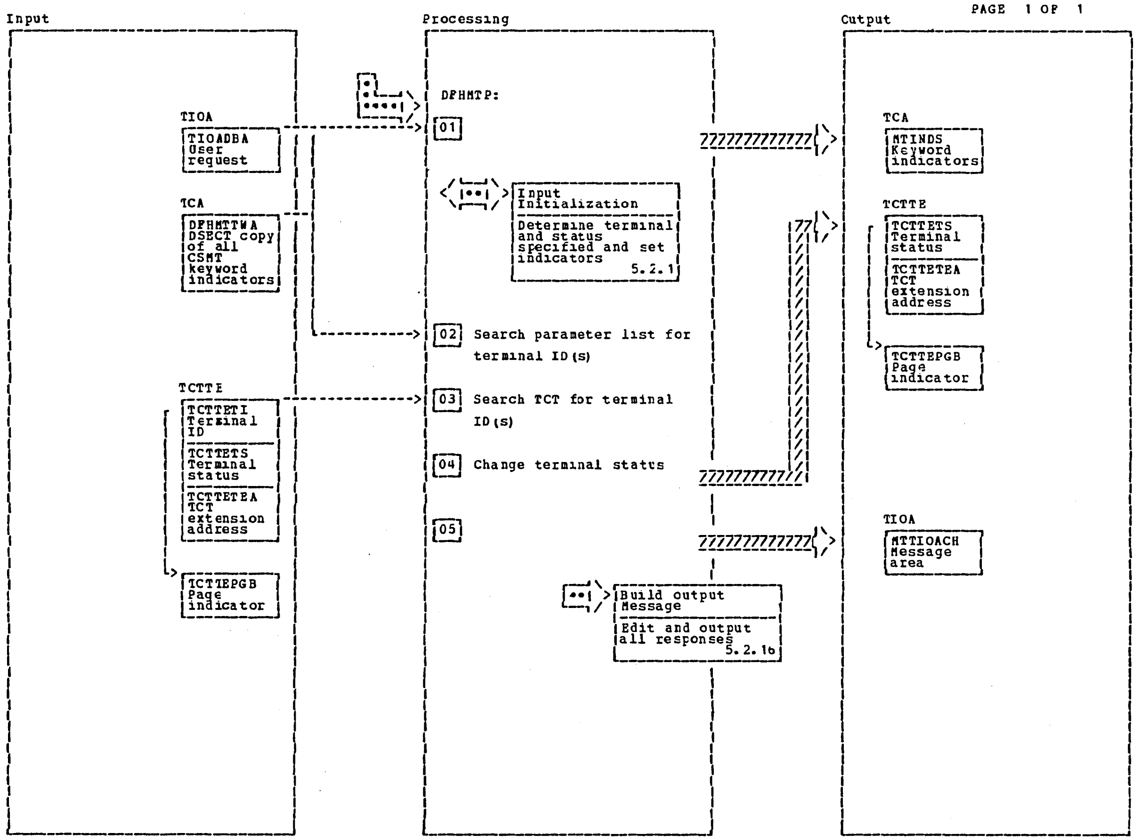
Diagram - 5.2.6-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | | |
| 02 A File Control LOCATE macro instruction is issued to find the FCT entry. | DFHMTPB | DOCMTP18 | |
| 03 | DFHMTPB | MTFILCHG | |
| 04 | DFHMTPB | MTFILDSX | |
| 05 | DFHMTPB | MTMMDC | 5.2.16 |
| 06 CSMT DATBAS,status,FILEID= | | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Change Data Set Processing Status

Diagram - 5.2.6-01


PAGE 1 OF 1
Input
Processing
Cutput
DFHMTP:
TIOA
TIOADBA User request
TCA
DFHMTTWA DSECT copy of all CSMT keyword indicators
TCTTE
TCTTETI Terminal ID
TCTTETS Terminal status
TCTTETEA TCT extension address
TCTTEPGB Page indicator
01
Input Initialization
Determine terminal and status specified and set indicators 5.2.1
02 Search parameter list for terminal ID(s)
03 Search TCT for terminal ID(s)
04 Change terminal status
05
Build output Message
Edit and output all responses 5.2.16
TCA
MTINDS Keyword indicators
TCTTE
TCTTETS Terminal status
TCTTETEA TCT extension address
TCTTEPGB Page indicator
TIOA
MTTIOACH Message area

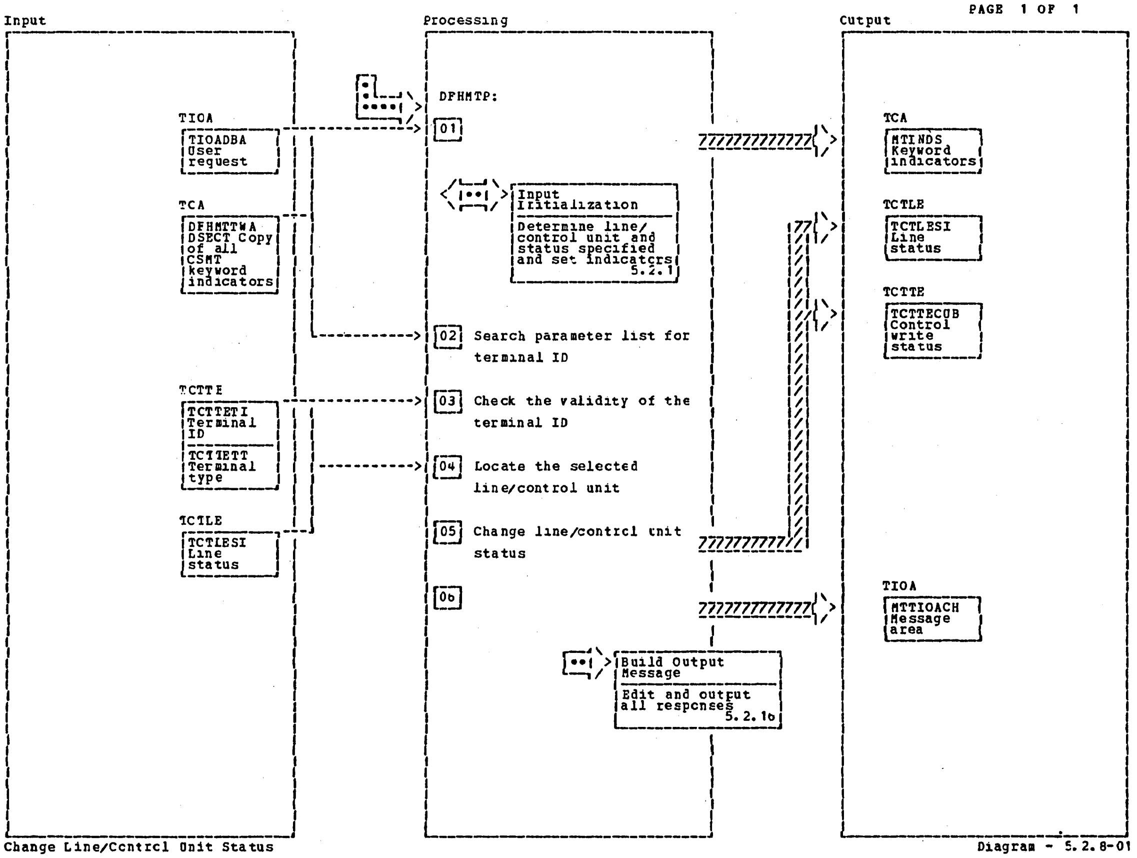
Change Terminal Status

Diagram - 5.2.7-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | | |
| 02 To define the terminal(s) to be changed, the following keywords may be used: | CFHMTPC | | |
| SINGLE (used with TERMID=) specifies cnly cne terminal ID. | | DOCMTP22 | |
| LIST (used with TERMID=) specifies a list of terminal IDs separated by commas. | | DOCMTP23 | |
| CLASS (used with CLASID=) specifies a class of terminals using their TLT suffix. | | DOCMTP24 | |
| 03 | DPHETPC | DOCMTP27 | |
| 04 The allowable processing status changes are: RECEIVE, INPUT, TRANSACTION, TRANSIEVE, AUTOPAGE, and PAGE.<br>The allowable service status changes are: INSERV, OUTSRV, ACQUIRE, and RELEASE. | DFHETPC | DOCMTP28 | |
| 05 | All DFHMTP modules | MTMMDC | 5:2.16 |

Change Terminal Status

Diagram - 5.2.7-01


PAGE 1 OF 1
Input
Processing
Output
TIOA
TIOADBA User request
TCA
DFHMTTWA DSECT Copy of all CSMT keyword indicators
TCTTE
TCTTETI Terminal ID
TCTTETT Terminal type
TCTLE
TCTLESI Line status
DFHMTP:
01
Input Initialization
Determine line/control unit and status specified and set indicators 5.2.1
02 Search parameter list for terminal ID
03 Check the validity of the terminal ID
04 Locate the selected line/control unit
05 Change line/control unit status
06
Build Output Message
Edit and output all responses 5.2.16
TCA
MTINDS Keyword indicators
TCTLE
TCTLESI Line status
TCTTE
TCTTECUB Control write status
TIOA
MTTIOACH Message area

Change Line/Control Unit Status

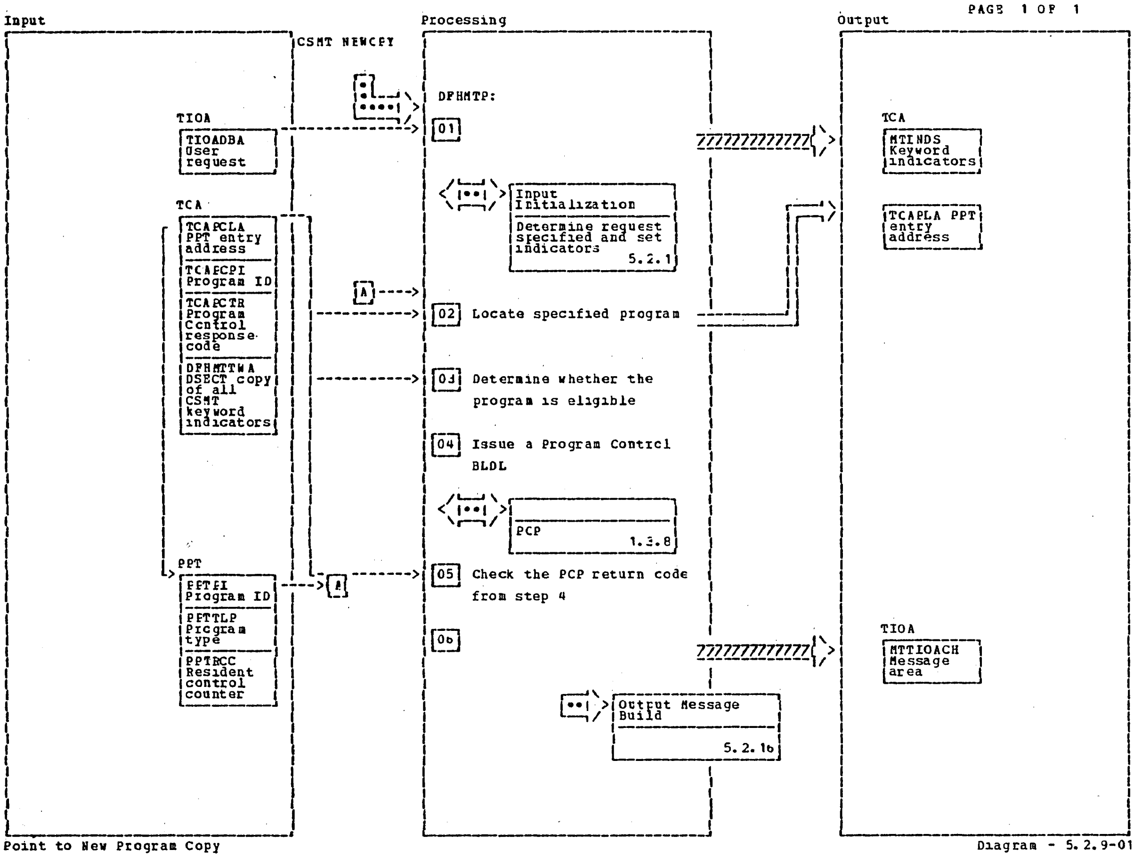
Diagram - 5.2.8-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | | | | | | |
| 02 If no TERMID given, a TERMID is requested. | DFHMTPF | MTLINCTL | | | | | |
| 03 If TERMID is not in TCT, a new TERMID is requested. | DFHMTPF | MTLINCTL | | | | | |
| 04 Line: | DFHMTPF | DOCMTP49 | | | | | |
| Control Unit: | DFHMTPF | DOCMTP50 | | | | | |
| 05 Line: | DFHMTPF | MTCLINE | | | | | |
| Control Unit: | DFHMTPF | MTCCNTRL | | | | | |
| An LOPEN macro instruction is issued to put the line/control unit in service. Since line status for a VTAM terminal is not possible, a message is issued. | | | | | | | |
| 06 | DFHMTPF | MTMMDC | 5.2.16 | | | | |

Change Line/Control Unit Status

Diagram - 5.2.8-01

PAGE 1 OF 1



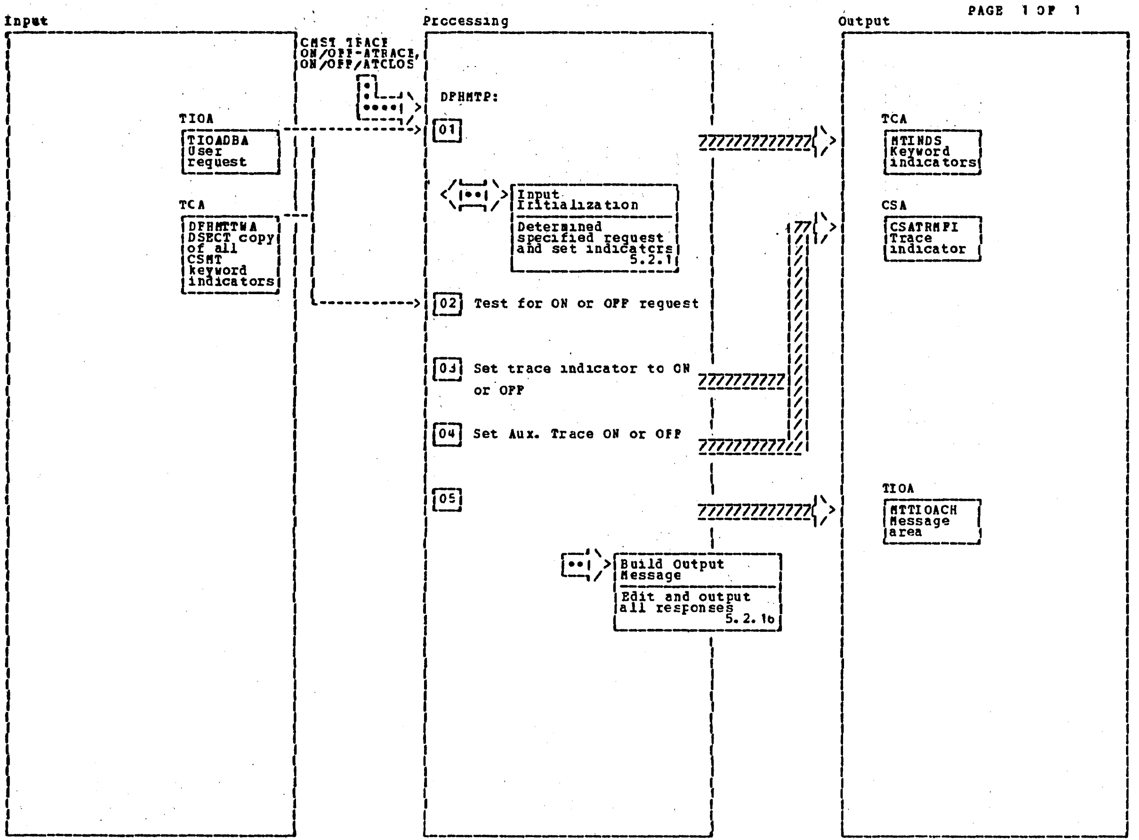
Point to New Program Copy

Diagram - 5.2.9-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | | | | | | |
| 02 | CFHMTPE | MTNEWCPR | | | | | |
| 03 Programs used by one or more tasks are ineligible. | CFEMTPE | MTNEWCPR | | | | | |
| 04 BLDL function allows PCP to obtain new TTR, program size, and entry point displacement for PPT entry. | CFEMTPE | MTNEWCPR | 1.3.8 | | | | |
| 05 | DFHMTPE | MTNEWCPR | | | | | |
| 06 | DFHMTPP | MTMMDC | 5.2.16 | | | | |

Point to New Program Copy

Diagram - 5.2.9-01


PAGE 1 OF 1
Input
Processing
Output
CMST TRACE ON/OFF-ATRACE, ON/OFF/ATCLOS
TIOA
TIOADBA User request
TCA
DFHMTTWA DSECT copy of all CSMT keyword indicators
DFHMTP:
01
Input Initialization
Determined specified request and set indicators 5.2.1
02 Test for ON or OFF request
03 Set trace indicator to ON or OFF
04 Set Aux. Trace ON or OFF
05
Build Output Message
Edit and output all responses 5.2.16
TCA
MTINDS Keyword indicators
CSA
CSATRMFI Trace indicator
TIOA
MTTIOACH Message area

Turn On/Off Trace Table Logging

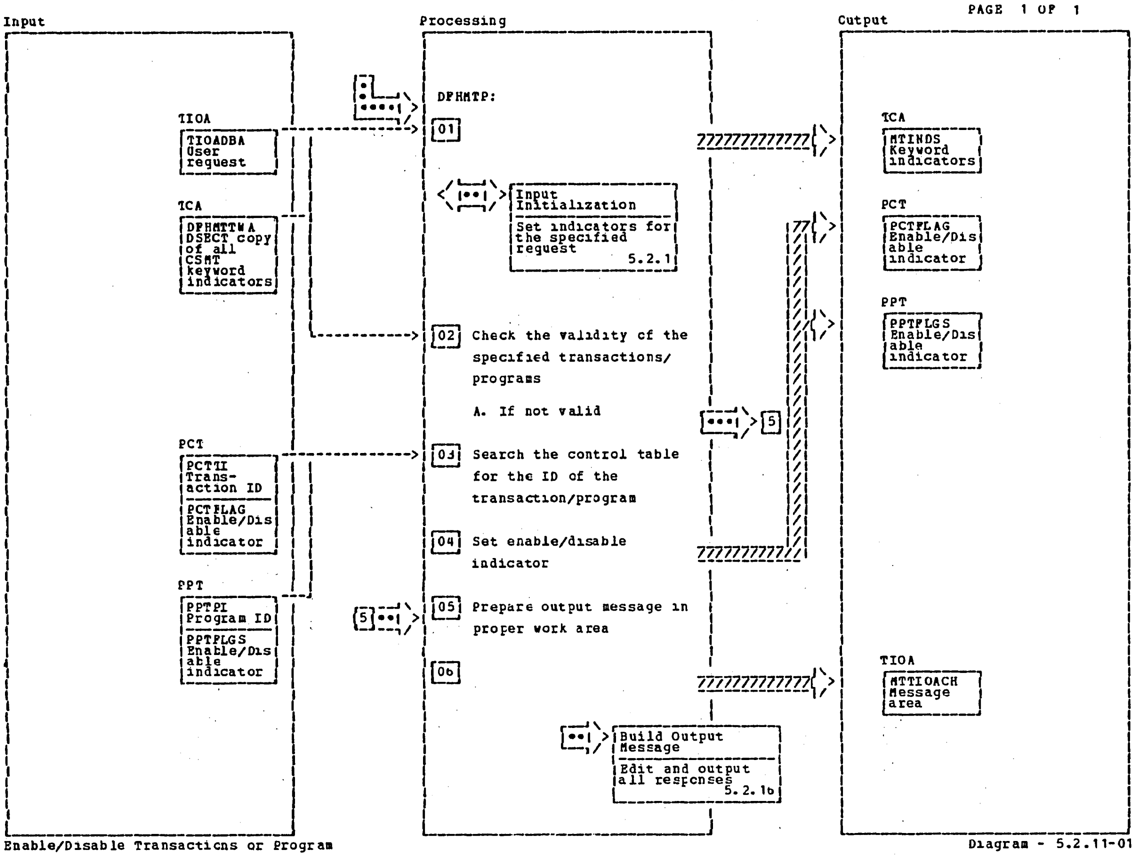
Diagram - 5.2.10-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | | | | | | |
| 02 | DFHMTPB | DCCMTP17 | | | | | |
| 03 | DFHMTPB | DCCMTP17 | | | | | |
| 04 | DFHMTPB | | | | | | |
| 05 | DFHMTPB | MTMMDC | 5.2.16 | | | | |

Turn On/Off Trace Table Logging

Diagram - 5.2.10-01

PAGE 1 OF 1


Input
Processing
Cutput
TIOA
TIOADBA User request
TCA
DFHMTTWA DSECT copy of all CSMT keyword indicators
PCT
PCTTI Trans-action ID
PCTFLAG Enable/Disable indicator
PPT
PPTPI Program ID
PPTFLGS Enable/Disable indicator
DFHMTP:
01
Input Initialization
Set indicators for the specified request 5.2.1
02 Check the validity of the specified transactions/ programs
A. If not valid
5
03 Search the control table for the ID of the transaction/program
04 Set enable/disable indicator
5
05 Prepare output message in proper work area
06
Build Output Message
Edit and output all responses 5.2.16
TCA
MTINDS Keyword indicators
PCT
PCTFLAG Enable/Disable indicator
PPT
PPTFLGS Enable/Disable indicator
TIOA
MTTIOACH Message area

Enable/Disable Transactions or Program

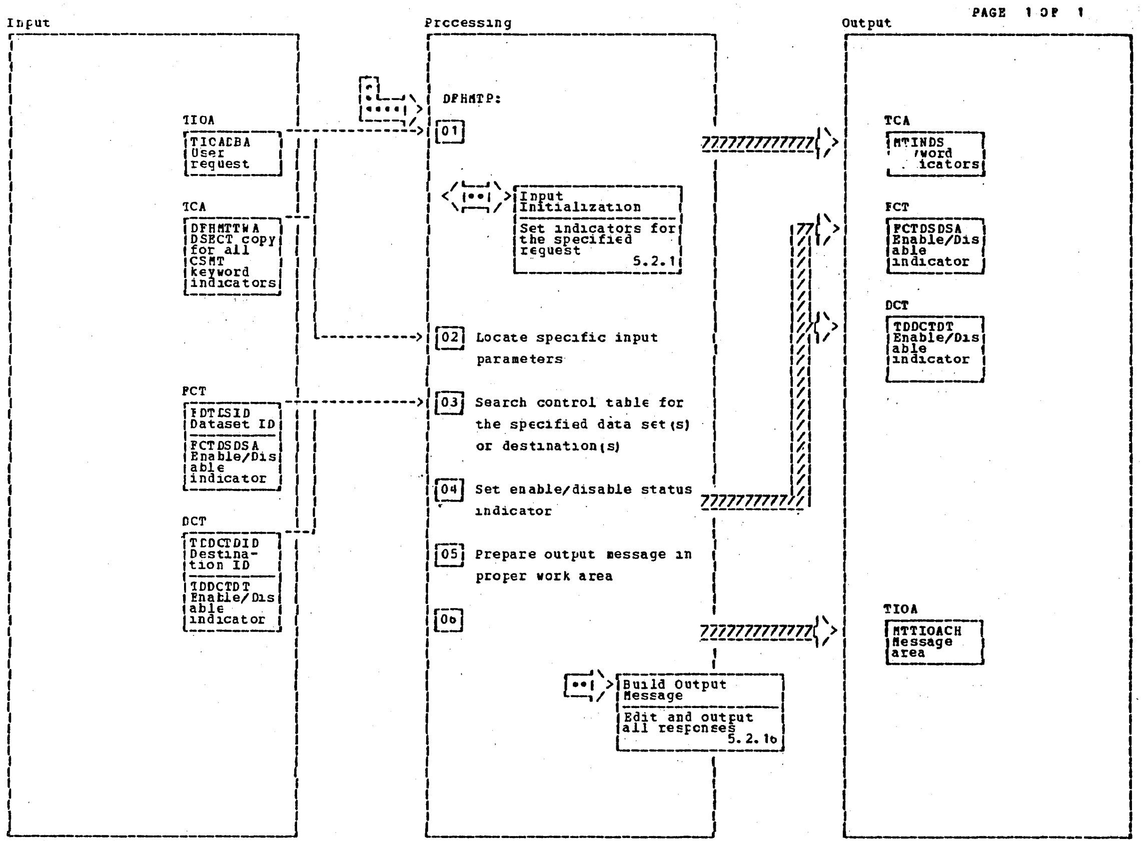
Diagram - 5.2.11-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | | |
| 02 The allowable parameters for transaction/ program specification are: | | | |
| SINGLE | DFHMTPG | MTSINGRT | |
| LIST | DFHMTPG | MTLISTRT | |
| CLASS | DFHMTPC | MTCLASRT | |
| ALL | DFHMTPG | MTALLRTN | |
| 03 The PPT is searched for the program ID. | DFHMTPG | MTPRED | |
| The PCT is searched for the transaction ID. | DFHMTPG | MTTPED | |
| 04 If PPT disabled, set disable indicator for all PCT entries that specify the PPT entry. | | | |
| Do not enable PPT or PCT entries if programs are not available. | | | |
| 05 | DFHMTPG | MTWMTTST | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 06 | DFHMTPG | MTMMDC | 5.2.16 |

Enable/Disable Transactions or Program

Diagram - 5,2.11-01



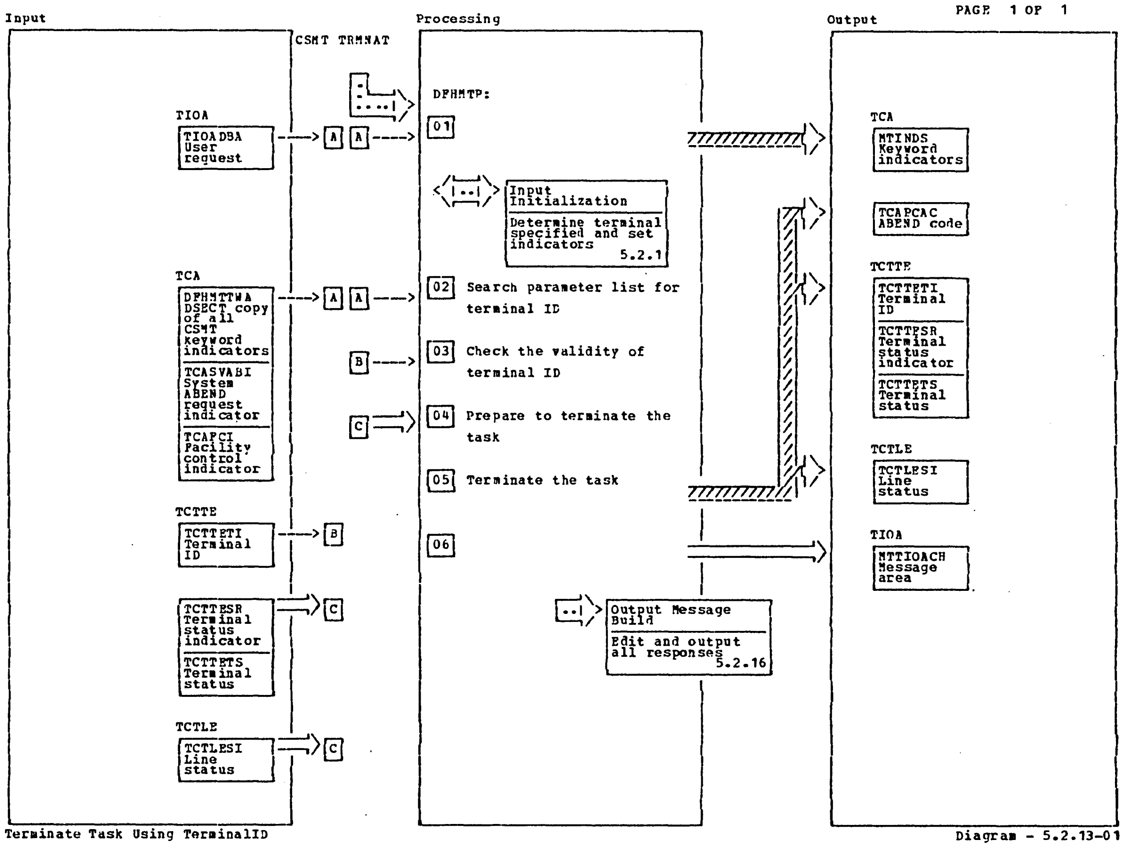
Enable/Disable Data Sets or Destinations

Diagram - 5.2.12-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | | | | | | |
| 02 Data set IDs are specified by the FILEID= OR ALL parameters. | DFHMTPB | DOCMTP18 | | | | | |
| Extrapartition destination IDs are specified by the DESTID= parameter. | DFHMTPD | DOCMTP36 | | | | | |
| 03 The FCT is searched for the data set ID. | DFHMTPB | DOCMTP18 | | | | | |
| The DCT is searched for the extrapartition destination ID. | DFHMTPD | DOCMTP36 | | | | | |
| 04 | | | | | | | |
| 05 The enabled/disabled constant is added to the message. | | | | | | | |
| 06 Data set message output. | DFHMTPB | MTMMDC | 5.2.14 | | | | |
| Extrapartition destination message output. | DFHMTPD | MTMMDC | 5.2.14 | | | | |

Enable/Disable Data Sets or Destinations

Diagram - 5.2.12-01


PAGE 1 OF 1
Input
Processing
Output
CSMT TRMNAT
DFHMTP:
TIOA
TIOADBA User request
A
A
01
TCA
MTINDS Keyword indicators
TCAPCAC ABEND code
Input Initialization
Determine terminal specified and set indicators
5.2.1
TCA
DFHMTTWA DSECT copy of all CSMT keyword indicators
TCASVABI System ABEND request indicator
TCAFCI Facility control indicator
02 Search parameter list for terminal ID
03 Check the validity of terminal ID
04 Prepare to terminate the task
05 Terminate the task
06
B
C
TCTTE
TCTTETI Terminal ID
TCTTESR Terminal status indicator
TCTTETS Terminal status
TCTLE
TCTLESI Line status
TCTTE
TCTTETI Terminal ID
TCTTESR Terminal status indicator
TCTTETS Terminal status
TCTLE
TCTLESI Line status
TIOA
MTTIOACH Message area
Output Message Build
Edit and output all responses
5.2.16

Terminate Task Using TerminalID

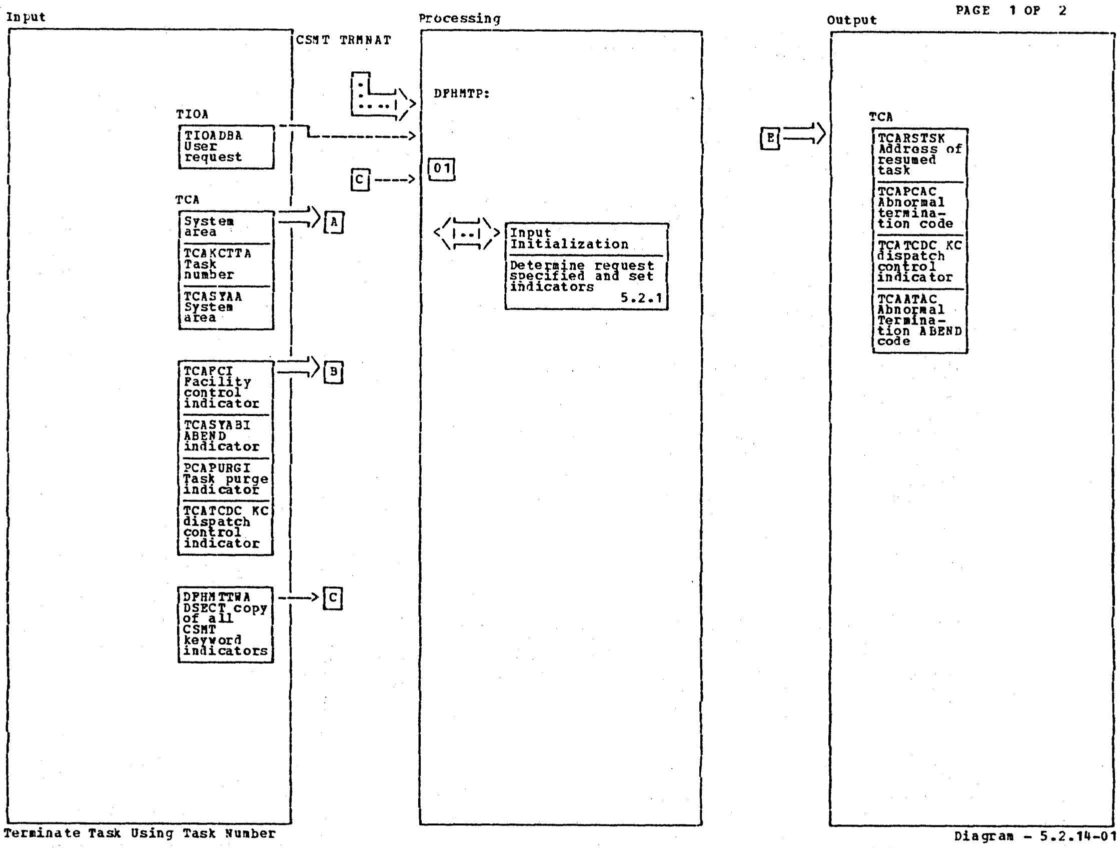
Diagram - 5.2.13-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 02 | DFHMTPF | DOCMTP51 | |
| 03 | DFHMTPF | DOCMTP51 | |
| 04 The specified terminal must be out of service to terminate the task. | DFHMTPF | DOCMTP54 | |
| If the terminal is out of service, issue ABEND code AMT1. | | MTTEXSTS | |
| 05 If no TCP WAIT was issued, a deierred ABEND is issued, (1) with ABEND code AMTH. -TCAABMTP indicator set in TCASYABI- | DFHMTPF | MTTEXSTS | |
| If a TCP WAIT was issued, all puigeable tasks are purged, (2) by setting the TCADCABD indicator in TCATCDC. | DFHMTPF | MTSTESTS | Notes 3 and 4 of 5.2.14 |
| 06 | DFHMTPF | MTMMDC | 5.2.16 |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Terminate Task Using TerminalID

Diagram - 5.2.13-01


PAGE 1 OF 2
Input
Processing
Output
CSMT TRMNAT
TIOA
TIOADBA User request
TCA
System area
TCAKCTTA Task number
TCASYAA System area
A
TCAFCI Facility control indicator
TCASYABI ABEND indicator
TCAPURGI Task purge indicator
TCATCDC KC dispatch control indicator
B
DFHMTTWA DSECT copy of all CSMT keyword indicators
C
DFHMTP:
C
01
Input Initialization
Determine request specified and set indicators
5.2.1
E
TCA
TCARSTSK Address of resumed task
TCAPCAC Abnormal termina-tion code
TCATCDC KC dispatch control indicator
TCAATAC Abnormal Termina-tion ABEND code

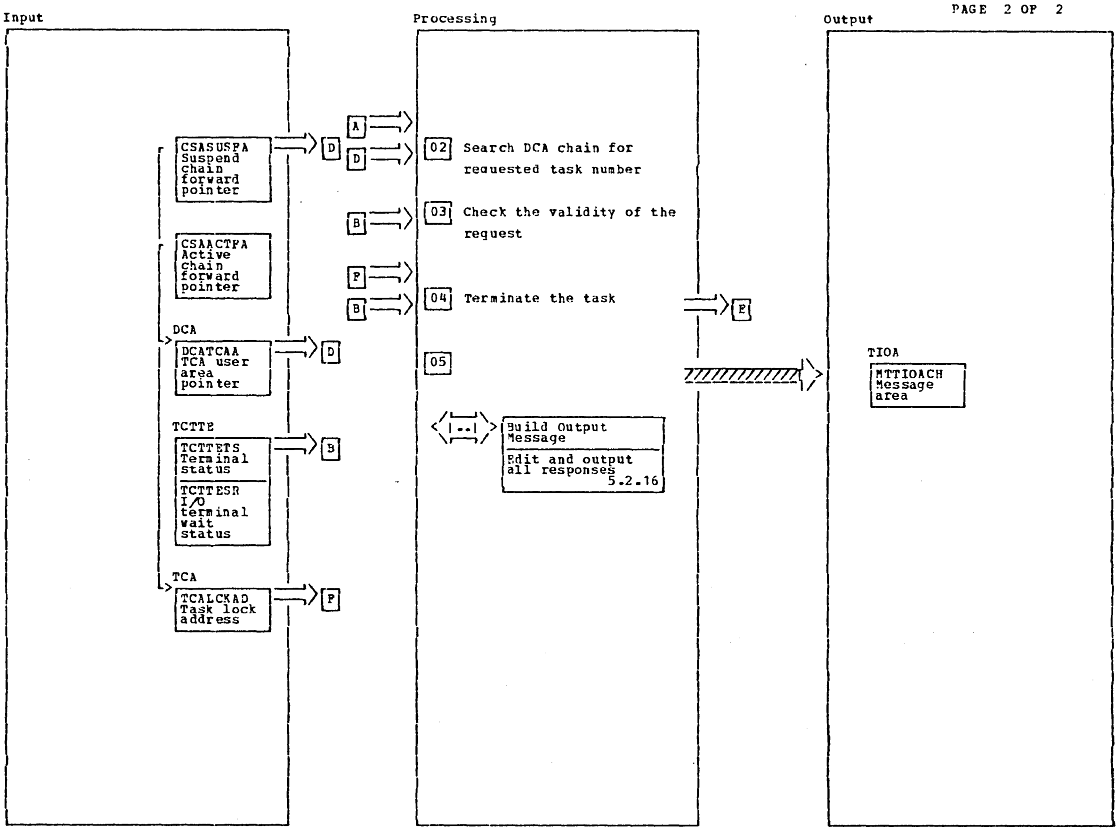
Terminate Task Using Task Number

Diagram - 5.2.14-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The user is also asked if termination is to be immediate. | DFHMTPF | MTGTMNID | | | | | |

Terminate Task Using Task Number

Diagram - 5.2.14-01


PAGE 2 OF 2
Input
Processing
Output
CSASUSFA Suspend chain forward pointer
CSAACTFA Active chain forward pointer
DCA
DCATCAA TCA user area pointer
TCTTE
TCTTETS Terminal status
TCTTESR I/O terminal wait status
TCA
TCALCKAD Task lock address
A
B
D
E
F
TIOA
MTTIOACH Message area
Build Output Message
Edit and output all responses 5.2.16

02 Search DCA chain for requested task number

03 Check the validity of the request

04 Terminate the task

05

Terminate Task Using Task Number

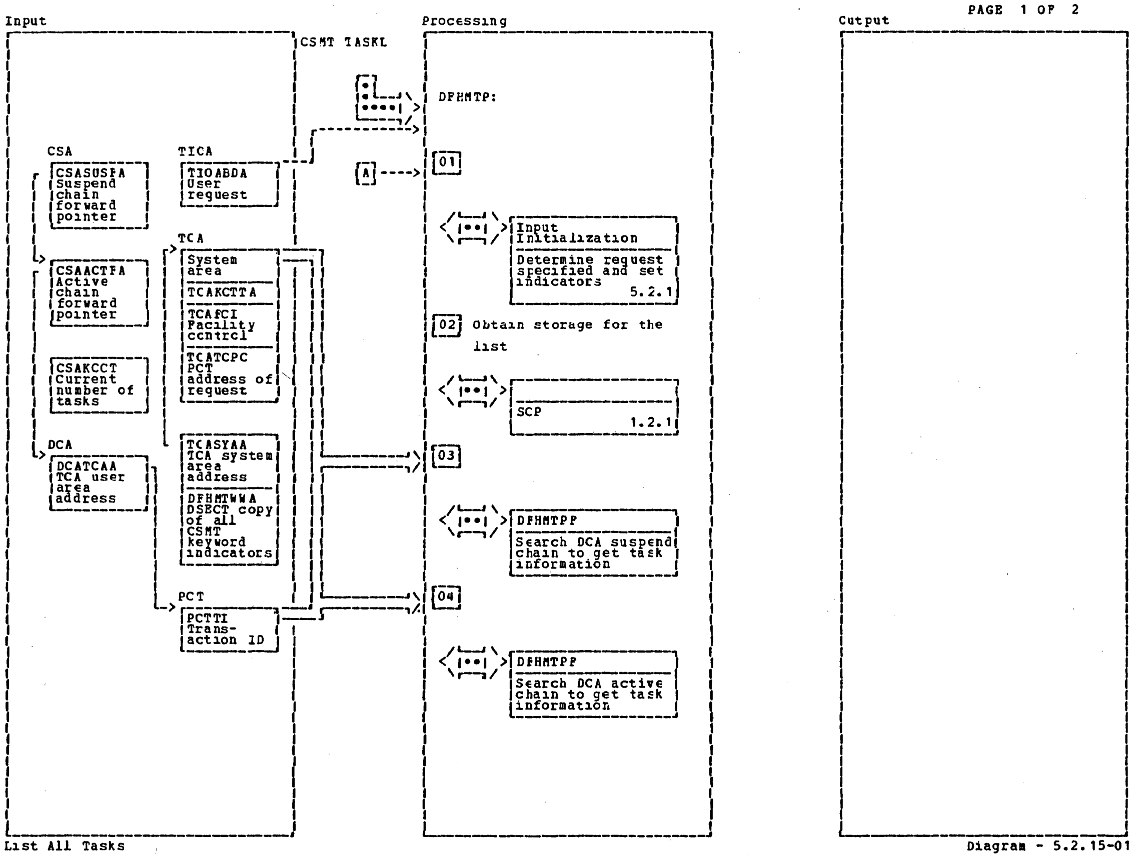
Diagram - 5.2.14-02

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 02 The chain is searched from low to high priority.<br>The DCA entries point to the TCA which contains the task number.<br>The suspend chain is searched first, the active chain second. If the task number is not found, an error message is issued and control is returned to CICS. | DFHMTPF | MTSKTERM | |
| 03 A task is ineligible for termination if:<br>A. An ABEND is in progress.<br>b. An ABEND has been deferred and a not-immediate ABEND is requested.<br>C. The terminal is not out of service.<br>D. The task is non-dispatchable and non-stall purgeable.<br>A task is eligible for teimination if: | DFHMTPF | MTFOUNDT | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| E. A terminal WAIT was issued.<br>F. An ABEND was previously deferred but an immediate ABEND is now requested.<br>G. The task is non-dispatchable and stall purgeable.<br>H. The task is waiting under a CICS LOCK. | | | |
| 04 For eligible tasks, the direct ABEND indicator is set -TCADCABD set on in TCATCDC- code, AMT, is moved to TCAPCAC. A DFHKC TYPE=RESUME macro instruction is issued.<br>For ineligible tasks, an error message is issued. | | | |
| 05 | DFHMTPF | MTMMDC | 5.2.16 |

Terminate Task Using Task Number

Diagram - 5.2.14-02


PAGE 1 OF 2
Input
Processing
Cutput
CSMT TASKL
DFHMTP:
CSA
CSASUSFA Suspend chain forward pointer
CSAACTFA Active chain forward pointer
CSAKCCT Current number of tasks
DCA
DCATCAA TCA user area address
TICA
TIOABDA User request
TCA
System area
TCAKCTTA
TCAFCI Facility control
TCATCPC PCT address of request
TCASYAA TCA system area address
DFHMTWWA DSECT copy of all CSMT keyword indicators
PCT
PCTTI Trans-action ID
A
01
Input Initialization
Determine request specified and set indicators 5.2.1
02 Obtain storage for the list
SCP 1.2.1
03
DFHMTPP
Search DCA suspend chain to get task information
04
DFHMTPP
Search DCA active chain to get task information

List All Tasks

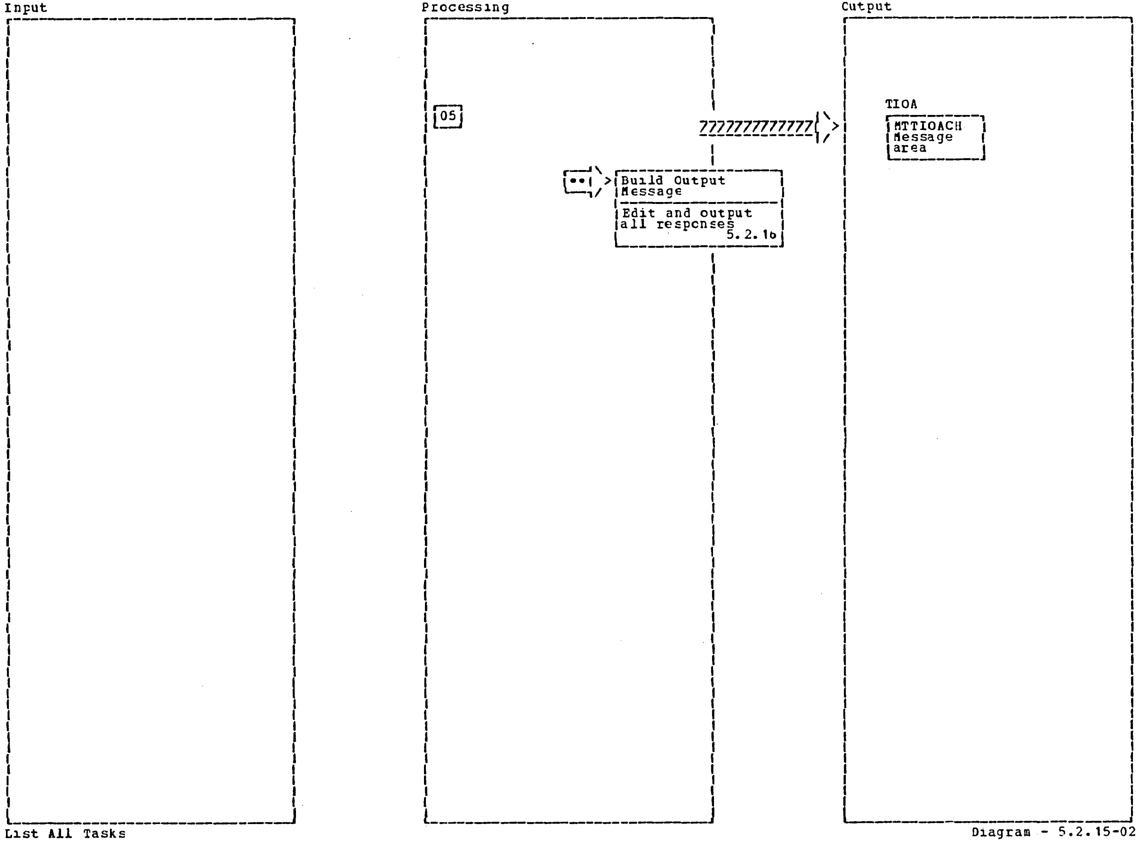
Diagram - 5.2.15-01

| Notes | Rottine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | | | | | | |
| 02 Total Storage=L(T+3)+H where: | DFHMTPP | MTASKLST | | | | | |
| T is the number of tasks, | | | | | | | |
| L is the length of the output line, | | | | | | | |
| H is the length of the heading, | | | | | | | |
| 3 is a safety margin. | | | | | | | |
| 03 The chain is searched from low to high priority. | DFHMTPP | MTLOOPSH | | | | | |
| This subroutine gets one line of output information. | DFHMTPP | MTASKLIN | | | | | |
| 04 The chain is search from low to high priority. (The last DCA is for the TCP which is not included in the output.) | DFHMTPP | MTLOOP2 | | | | | |

List All Tasks

Diagram - 5.2.15-01

PAGE 2 OF 2



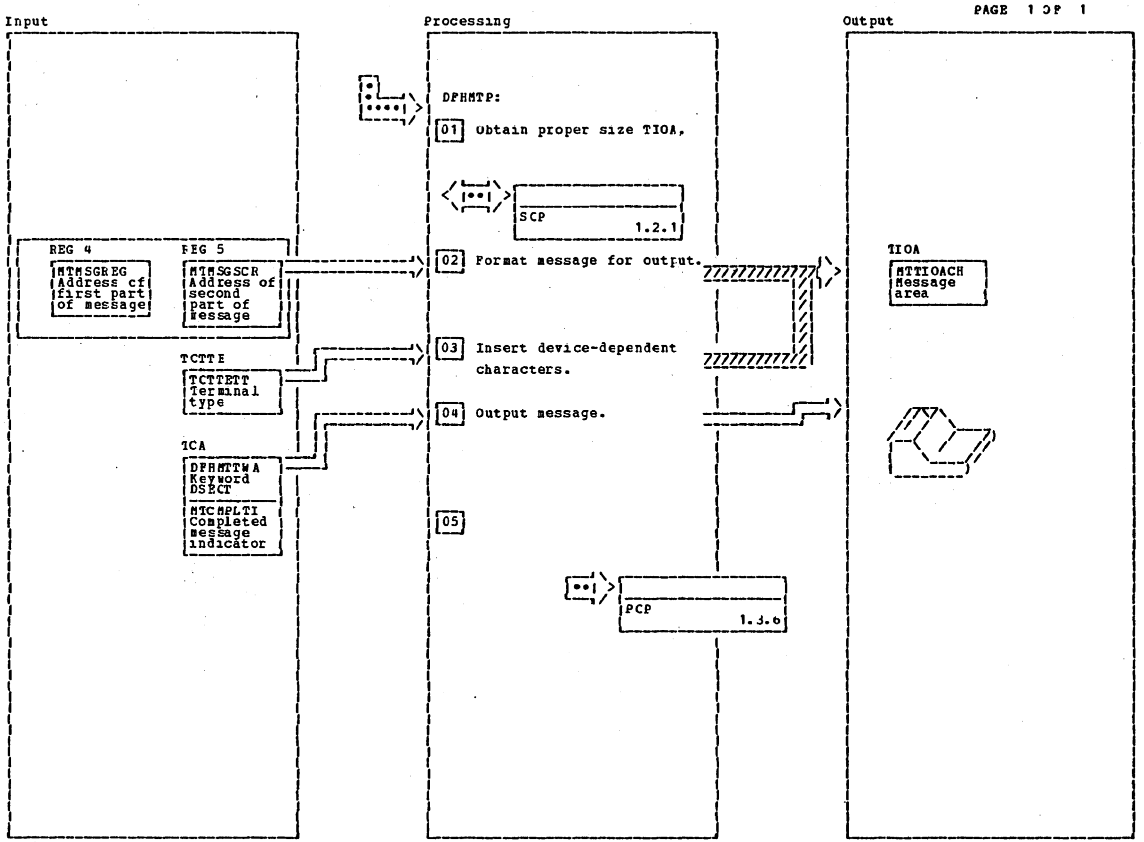
List All Tasks

Diagram - 5.2.15-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 | DFHMTPF | MTMMDC | 5.2.16 | | | | |

List All Tasks

Diagram - 5.2.15-02


PAGE 1 OF 1
Input
Processing
Output
DFHMTP:
01 Obtain proper size TIOA.
SCP 1.2.1
REG 4
MTMSGREG Address of first part of message
REG 5
MTMSGSCR Address of second part of message
02 Format message for output.
TIOA
MTTIOACH Message area
TCTTE
TCTTETT Terminal type
03 Insert device-dependent characters.
TCA
DFHMTTWA Keyword DSECT
MTCMPLTI Completed message indicator
04 Output message.
05
PCP 1.3.6

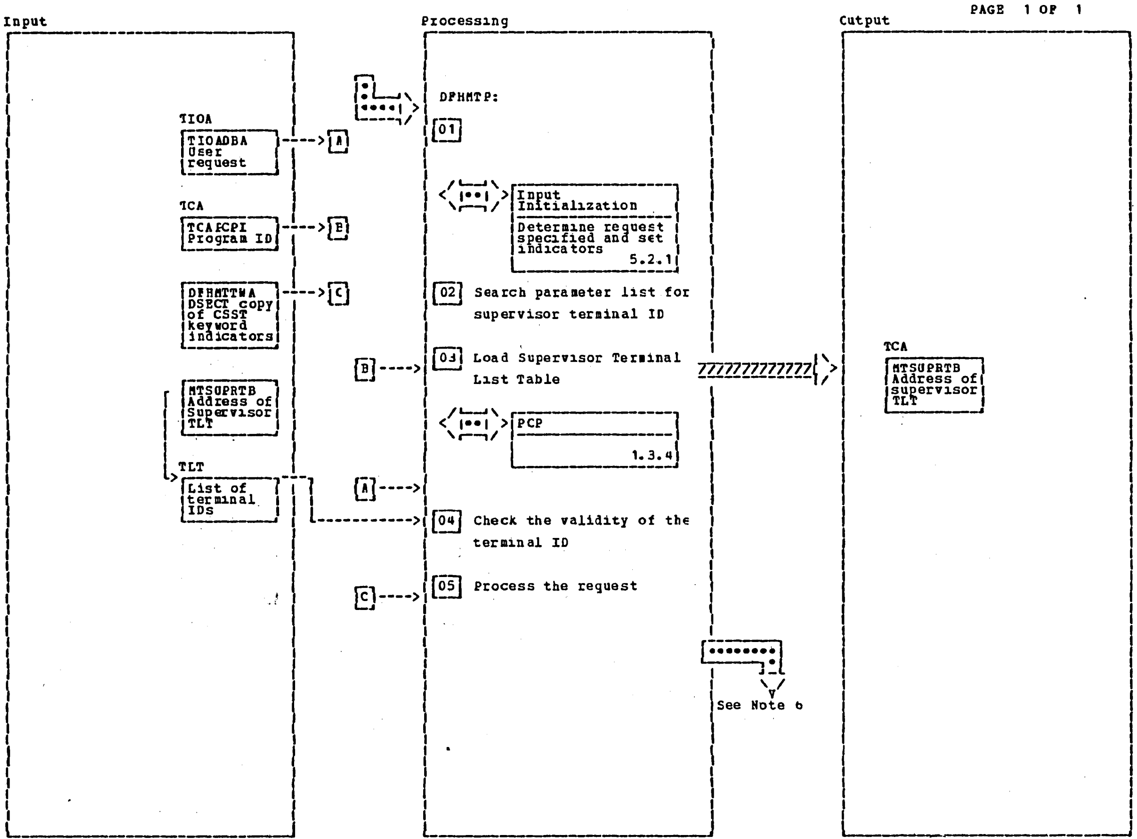
Build Output Message

Diagram - 5.2.16-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The original TIOA is used if it is large enough. | All DFHMTP modules | MTTTIOL | | | | | |
| 02 The message is formatted into not more than 40-character lines. Carriage returns are inserted where needed. | All DFHMTP modules | MTMMDC | | | | | |
| 03 Special formatting is required for 2780, 3270, TWX, VTAM, and bisync programmable terminals. | All DFHMTP modules | MTENDSET | | | | | |
| 04 If this is not the final write of the task, a Terminal Control WRITE, READ, and WAIT is issued; then control is returned to the caller. | All DFHMTP modules | MTTCPWWR | | | | | |
| For the final write, a Terminal Control WRITE is issued. | All DFHMTP modules | MTTCPWW | | | | | |
| At the end of the final write, the data and time is written. | All DFHMTP modules | MTOUTDT2 | | | | | |
| 05 A DFHPC TYPE=RETURN is issued. | DFHMTPA | MTRETURN | | | | | |

Build Output Message

Diagram - 5.2.16-01



Supervisor Terminal

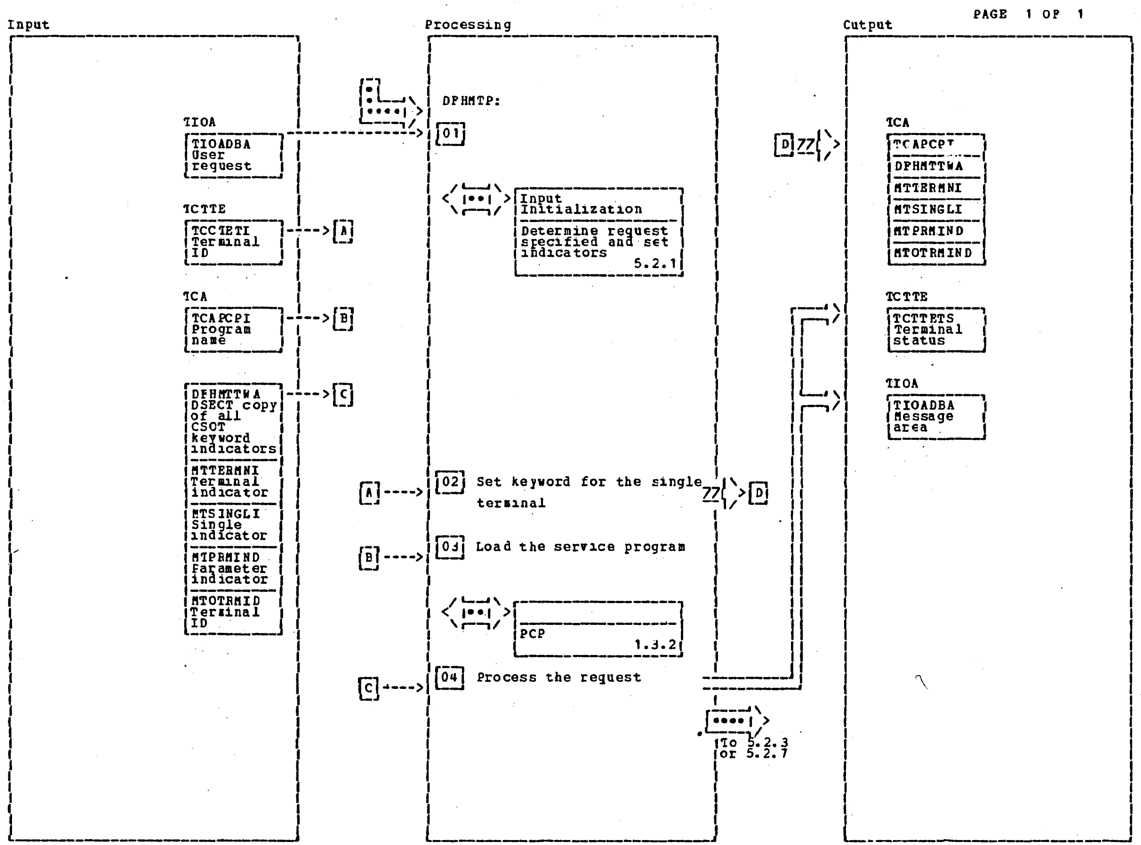
Diagram - 5.3.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 02 A search is made for the following requests: | | | |
| Inquire/change the status of a line/control unit | DFHMTPF | MTCSSTRT | |
| Inquire/change the status of terminal | DFHMTPF | MTCSSTRT | |
| Terminate a task | DFHMTPC | MTCSSTRT | |
| 03 A Program Control LOAD is issued for the specified terminal list table. | | | |
| 04 The specified terminal ID must be in the TLT: if not, format a response and go to 5.2.16. | DFHMTPC<br>OR<br>DFHMTPF | MTSCHSID | 5.2.16 |
| 05 The allowable requests are: | | | |
| Change terminal status | | | 5.2.7 |
| Change line/control unit status | | | 5.2.8 |
| Inquire about terminal or line/control unit status | | | 5.2.3 |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| Terminate a task | | | 5.2.13 |
| 06 To 5.2.3, 5.2.7, 5.2.8, or 5.2.13 | | | |

Supervisor Terminal

Diagram - 5.3.1-01



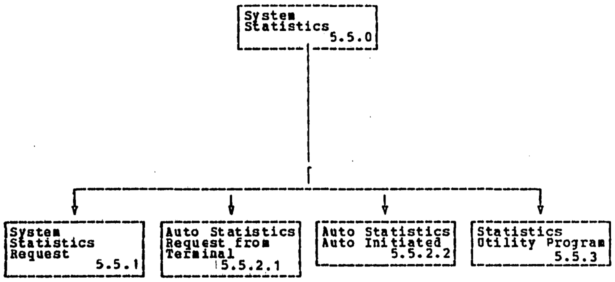
Operator Terminal

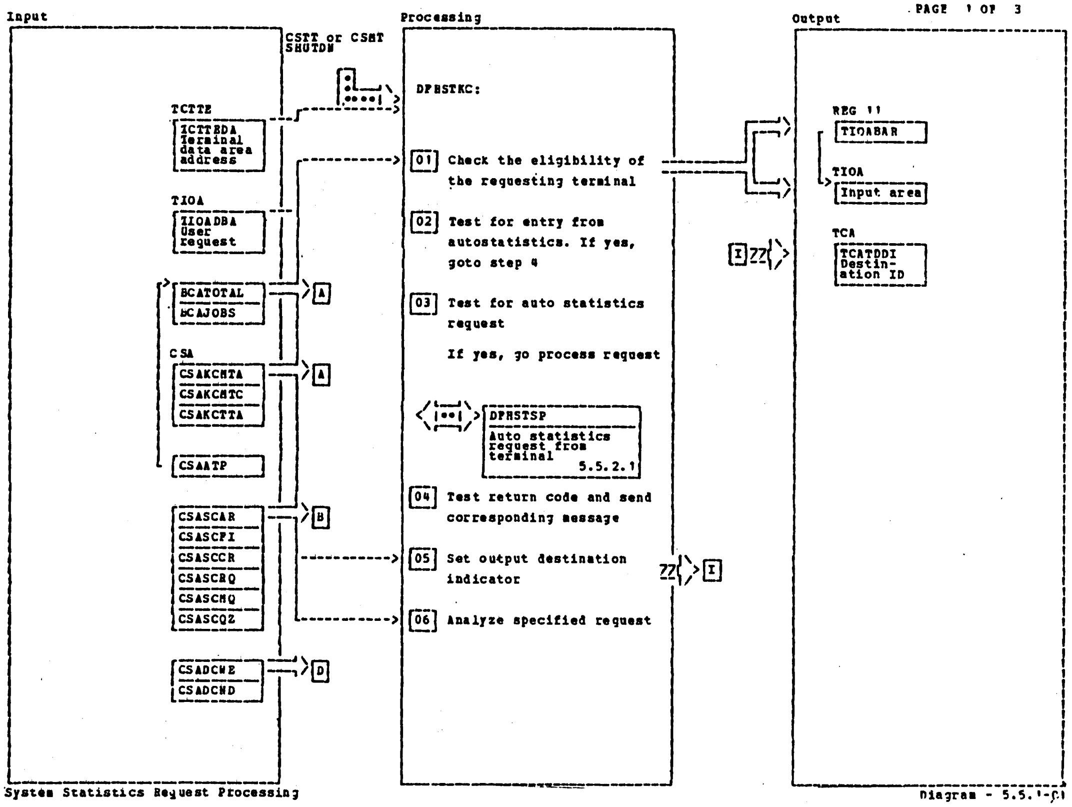
Diagram - 5.4.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 02 In the TCA, the terminal, parameter, and single indicators are turned on. | DFHMTPA | MTWMSGA | |
| 03 Module DFHMTPC is loaded via a Program Control XCTL. | DFHMTPA | MTLOADCA | |
| 04 Allowable operator terminal requests are: | | | |
| Inquiry | DFHMTPC | MTSNGBAL | 5.2.3 |
| Processing status change | DFHMTPC | MTCTRMST | 5.2.7 |
| Service status cannot be changed. | | | |

Operator Terminal

Diagram - 5.4.1-01


System Statistics 5.5.0
System Statistics Request 5.5.1
Auto Statistics Request from Terminal 5.5.2.1
Auto Statistics Auto Initiated 5.5.2.2
Statistics Utility Program 5.5.3


PAGE 1 OF 3
Input
Processing
Output
CSTT or CSMT SHUTDN
DFHSTKC:
TCTTE
TCTTEDA Terminal data area address
TIOA
TIOADBA User request
BCATOTAL
BCAJOBS
CSA
CSAKCMTA
CSAKCMTC
CSAKCTTA
CSAATP
CSASCAR
CSASCPI
CSASCCR
CSASCRQ
CSASCMQ
CSASCQZ
CSADCWE
CSADCWD
A
A
B
D
01 Check the eligibility of the requesting terminal
02 Test for entry from autostatistics. If yes, goto step 4
03 Test for auto statistics request
If yes, go process request
DFHSTSP
Auto statistics request from terminal 5.5.2.1
04 Test return code and send corresponding message
05 Set output destination indicator
06 Analyze specified request
ZZ
I
REG 11
TIOABAR
TIOA
Input area
TCA
TCATDDI Destination ID
I
ZZ

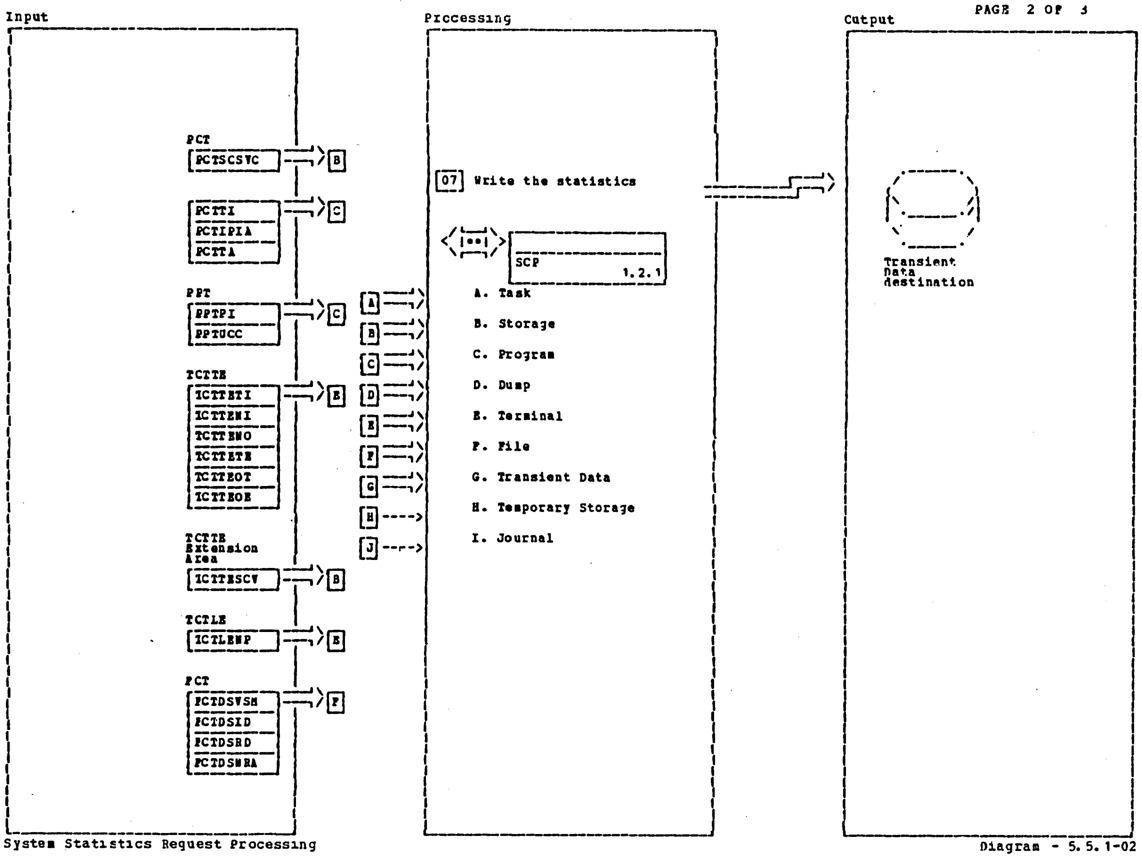
System Statistics Request Processing

Diagram - 5.5.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If the system statistics are requested via a CSMT SHUTDN request, this check is not made. | DFHSTKC | DFHSTKC | | | | | |
| 02 Test indicator in CSA (CSA XCTKC) which indicates a request from auto Initiated Auto Statistics to send a message. | | | | | | | |
| 03 CSSL is the default destination. | DFHSTKC | DFHSTKCQ | | | | | |
| 04 Determine whether AOR or SOR was requested and check the labels field for four-character destinations.<br>The CSMT SHUTDN request always outputs all relevant statistics. | DFHSTKC | TSTCODE | | | | | |

System Statistics Request Processing

Diagram - 5.5.1-01



System Statistics Request Processing

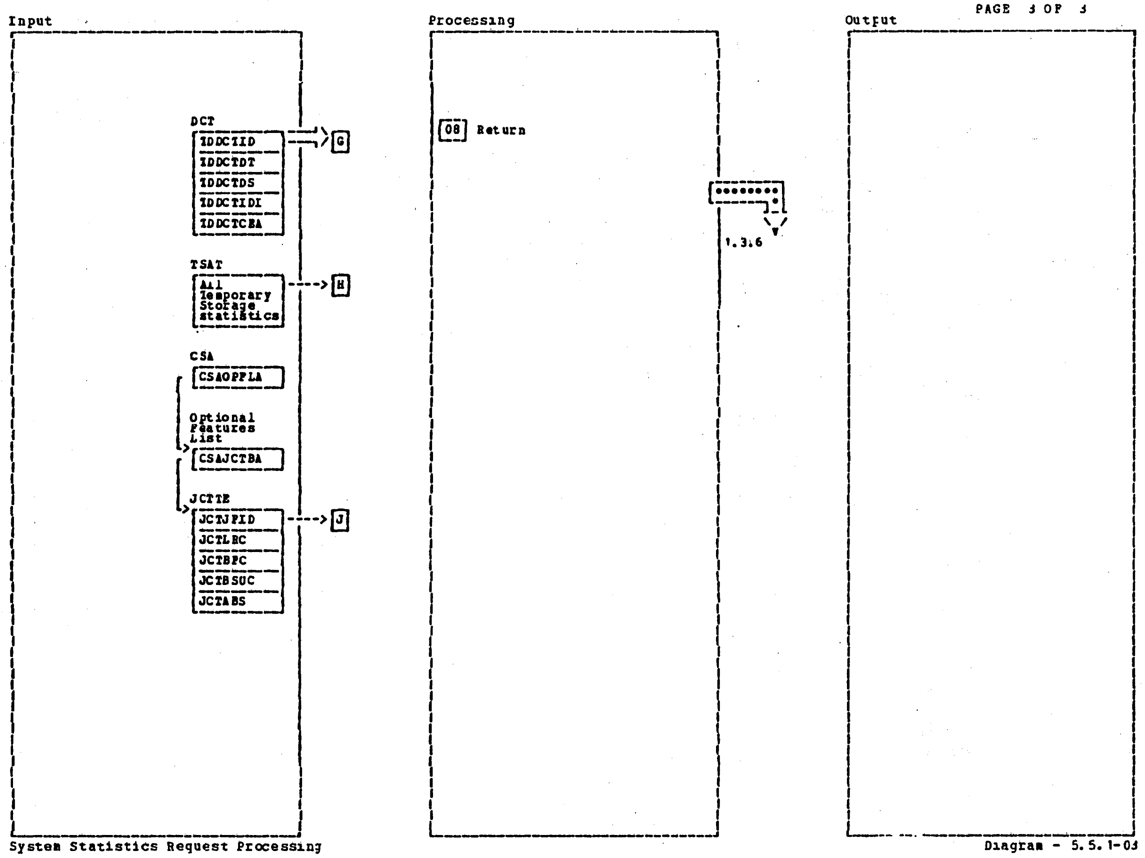
Diagram - 5.5.1-02

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 05 A Storage Control GETMAIN is issued for each statistic requested. A Transient Data PUT is issued to send each statistic to the specified destination. | | | |
| A. Task | DFHSTKC | GO | |
| B. Storage | DFHSTKC | STORAGE | |
| C. Program | DFHSTKC | PROGRAM | |
| D. Dump | DFHSTKC | DUMP | |
| E. Terminal | DFHSTTR | DFHSTTRN | |
| F. File | DFHSTTR | FILE | |
| G. Transient Data | DFHSTTD | DFHSTTDN | |
| H. Temporary Storage | DFHSTTD | TEMPSTOR | |
| I. Journal | DFHSTTD | JOURNST | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

System Statistics Request Processing

Diagram - 5.5.1-02



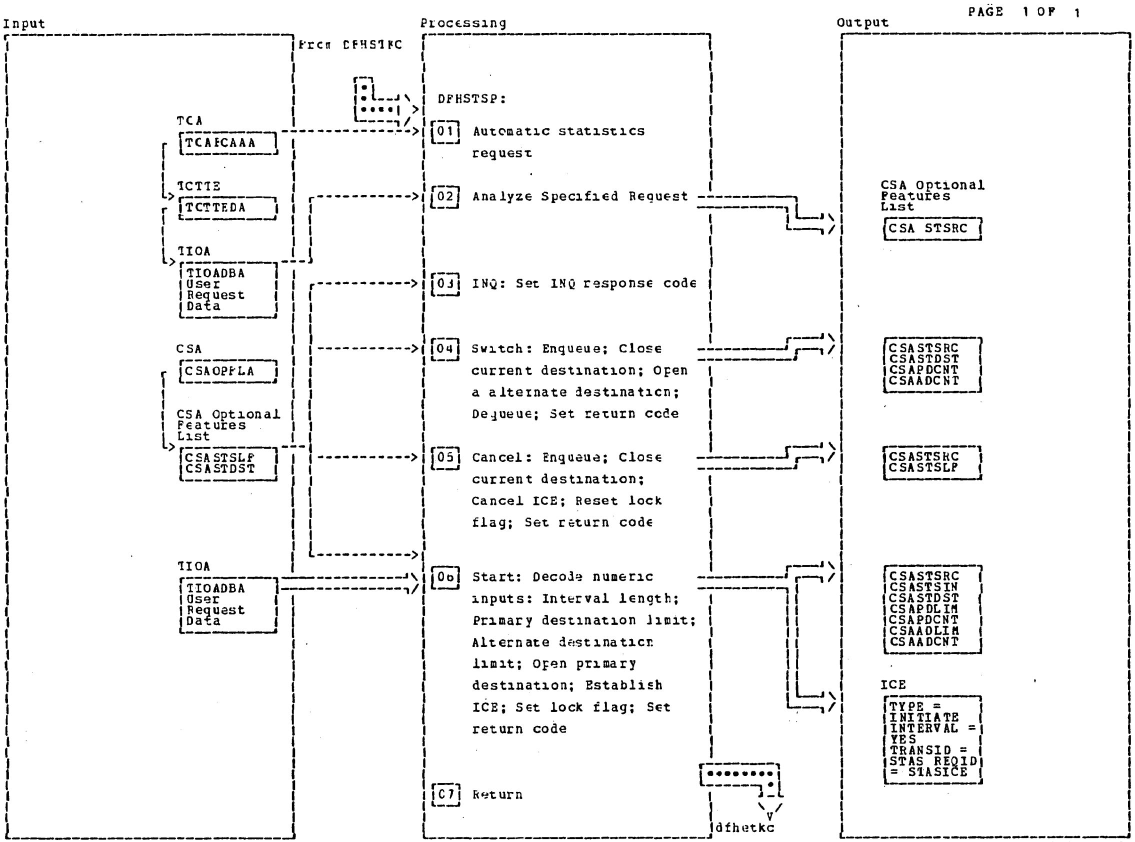
System Statistics Request Processing

Diagram - 5.5.1-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|

System Statistics Request Processing

Diagram - 5.5.1-03


PAGE 1 OF 1
Input
Processing
Output
From DFHSTKC
DFHSTSP:
TCA
TCAFCAAA
TCTTE
TCTTEDA
TIOA
TIOADBA
User
Request
Data
CSA
CSAOPFLA
CSA Optional
Features
List
CSASTSLP
CSASTDST
TIOA
TIOADBA
User
Request
Data
01 Automatic statistics request
02 Analyze Specified Request
03 INQ: Set INQ response code
04 Switch: Enqueue; Close current destination; Open a alternate destination; Dequeue; Set return code
05 Cancel: Enqueue; Close current destination; Cancel ICE; Reset lock flag; Set return code
06 Start: Decode numeric inputs: Interval length; Primary destination limit; Alternate destination limit; Open primary destination; Establish ICE; Set lock flag; Set return code
C7 Return
dfhetkc
CSA Optional
Features
List
CSA STSRC
CSASTSRC
CSASTDST
CSAPDCNT
CSAADCNT
CSASTSRC
CSASTSLP
CSASTSRC
CSASTSIN
CSASTDST
CSAPDLIM
CSAPDCNT
CSAADLIM
CSAADCNT
ICE
TYPE =
INITIATE
INTERVAL =
YES
TRANSID =
STAS REQID
= STASICE

Automatic Statistics Request from Terminal

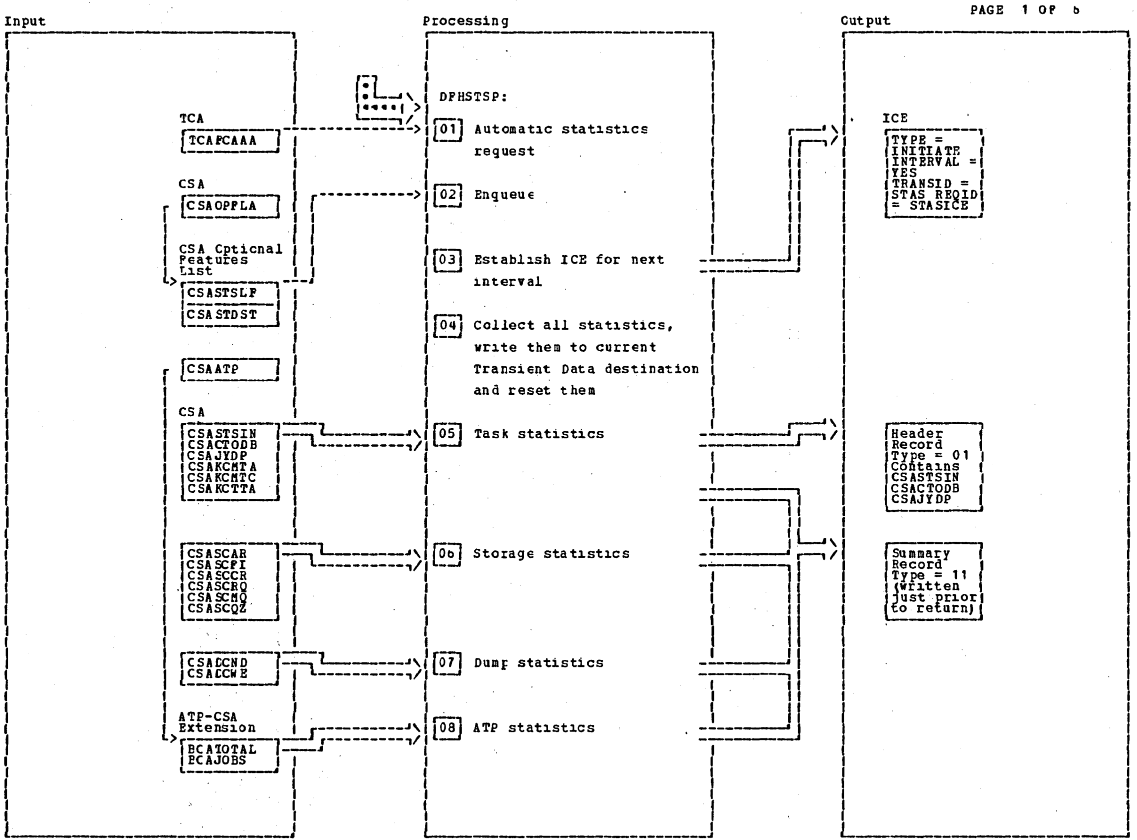
Diagram - 5.5.2.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The absence of a TCTTE implies automatic initiation. | DFHSTSP | DFHSTSPN | 5.5.2.2 |
| 02 The TIOA contains a request to START, CANCEL, SWITCH, or INQ. | DFHSTSP | | |
| 03 In the case of a INQ, DFHSTSP merely sets a return code. The data for the message exists in the CSA Optional Features List. | DFHSTSP | STQUERY | |
| 04 The Switch routine enqueues on the address of the lock flag to prevent use by the automatic task during switching. Switch puts the new destination ID in CSASTDST and resets the appropriate count. | DFHSTSP | STSWITCH | |
| 05 The Cancel routine enqueues on the address of the lock flag to prevent use by the automatic task during close. | DFHSTSP | TRMINATE | |
| 06 The interval length is a required input. If the destination limits are not specified, zero is stored in the CSAand implies no limit. | | | |
| 07 | DFHSTSP | RETURN | |

Automatic Statistics Request from Terminal

Diagram - 5.5.2.1-01

PAGE 1 OF 6



Automatic Statistics Program Automatic Initialization

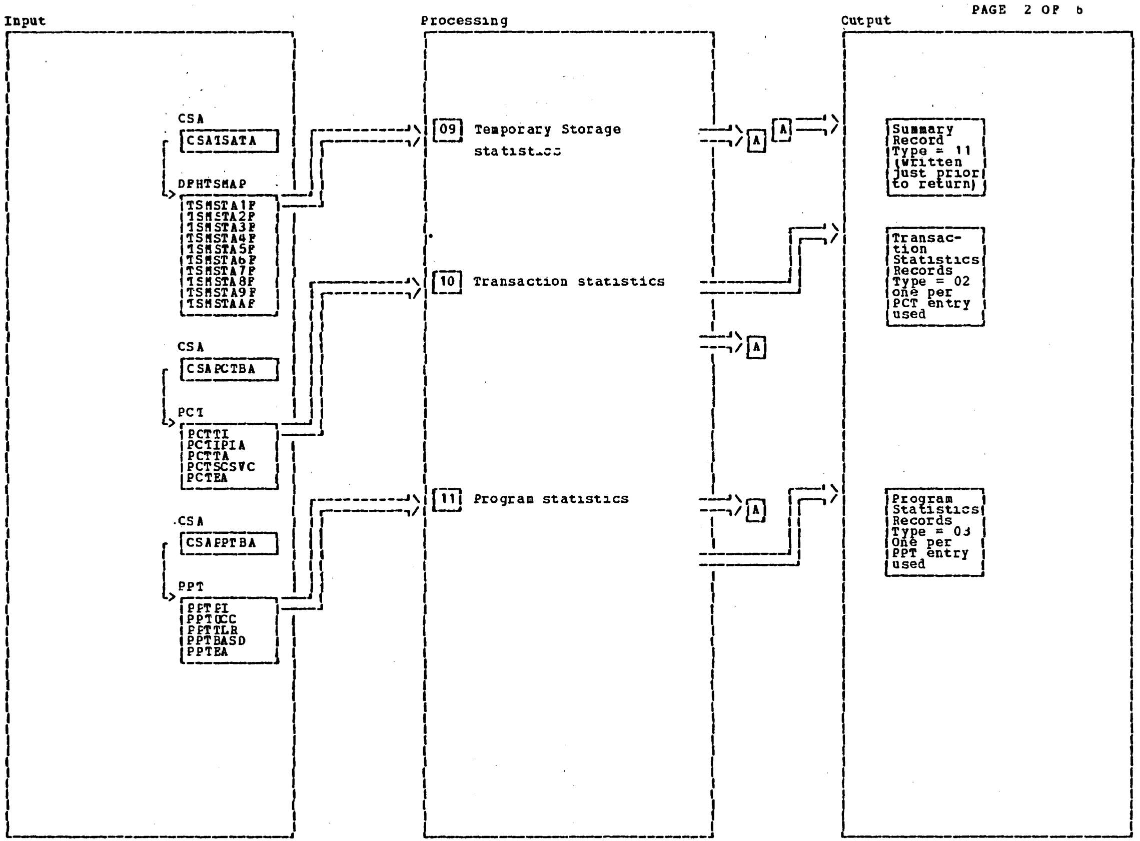
Diagram - 5.5.2.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The presence of a facility implies a user request. | DFHSTSP | DFHSTSPN | 5.5.2.1 | | | | |
| 02 The Data Collection routine enqueues on the address of the lock flag to prevent Switch and Cancel frcm closing output destination during use. | DFHSTSP | AUTOINIT | | | | | |
| 03 The ICE must be established first to closely approximate the requested interval. | DFHSTSP | AUTOINIT | | | | | |
| 04 The value in CSAKCTTA is never reset. | | | | | | | |
| 05 Data is entered into summary record tc be written later. | DFHSTSP | TASK | | | | | |
| 06 Same as note 5. | DFHSTSP | STORAGE | | | | | |
| 07 Same as nete 5. | DFHSTSP | DUMP | | | | | |
| 08 Same as note 5. | DFHSTSP | TASK | | | | | |

Automatic Statistics Program Automatic Initialization

Diagram - 5.5.2.2-01

PAGE 2 OF 6


Input
Processing
Output
CSA
CSATSATA
DFHTSMAP
TSMSTA1F
TSMSTA2F
TSMSTA3F
TSMSTA4F
TSMSTA5F
TSMSTA6F
TSMSTA7F
TSMSTA8F
TSMSTA9F
TSMSTAAF
CSA
CSAPCTBA
PCT
PCTTI
PCTIPIA
PCTTA
PCTSCSVC
PCTEA
CSA
CSAPPTBA
PPT
PPTFI
PPTUCC
PPTTLR
PPTBASD
PPTEA
09 Temporary Storage statistics
10 Transaction statistics
11 Program statistics
A
A
A
A
Summary Record Type = 11 (written just prior to return)
Transaction Statistics Records Type = 02 one per PCT entry used
Program Statistics Records Type = 03 One per PPT entry used

Automatic Statistics Program Automatic Initialization

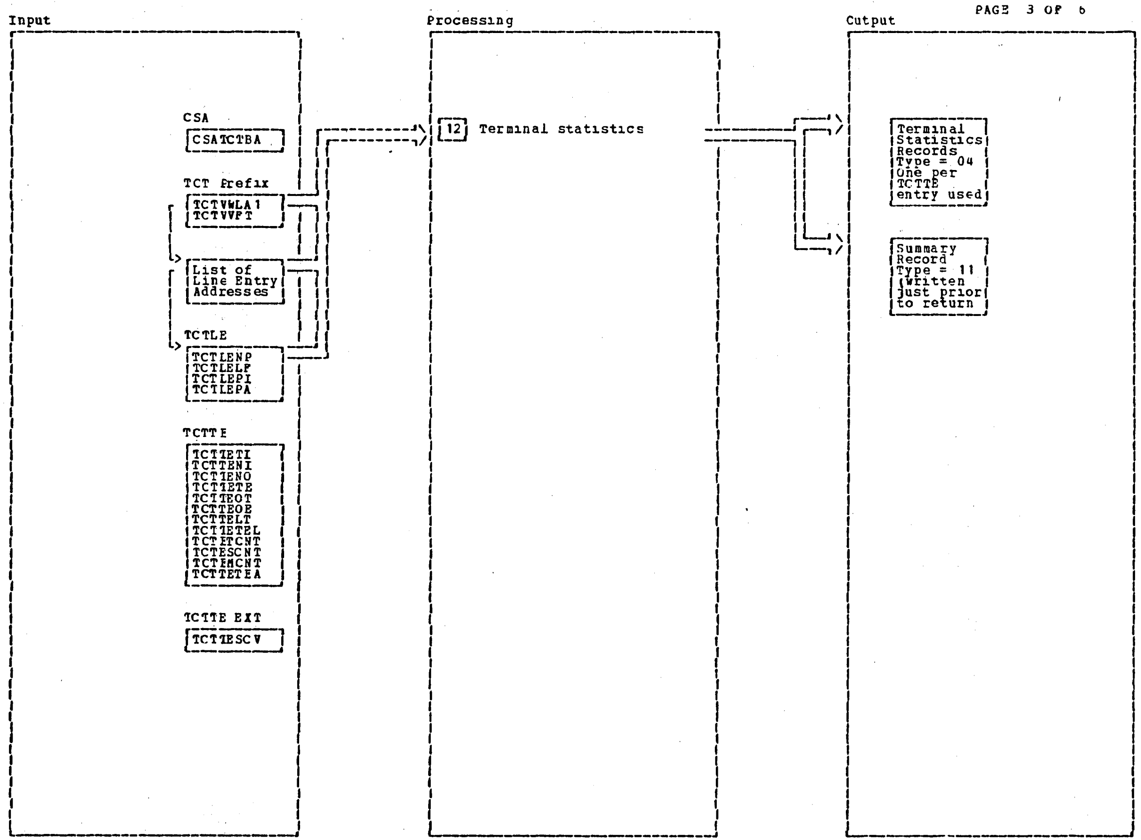
Diagram - 5.5.2.2-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 09 Same as note 5. | DFHSTSP | TEMPSTOR | | | | | |
| 10 | DFHSTSP | TRANSACT | | | | | |
| 11 | DFHSTSP | PROGRAM | | | | | |

Automatic Statistics Program Automatic Initialization

Diagram - 5.5.2.2-02

PAGE 3 OF 6



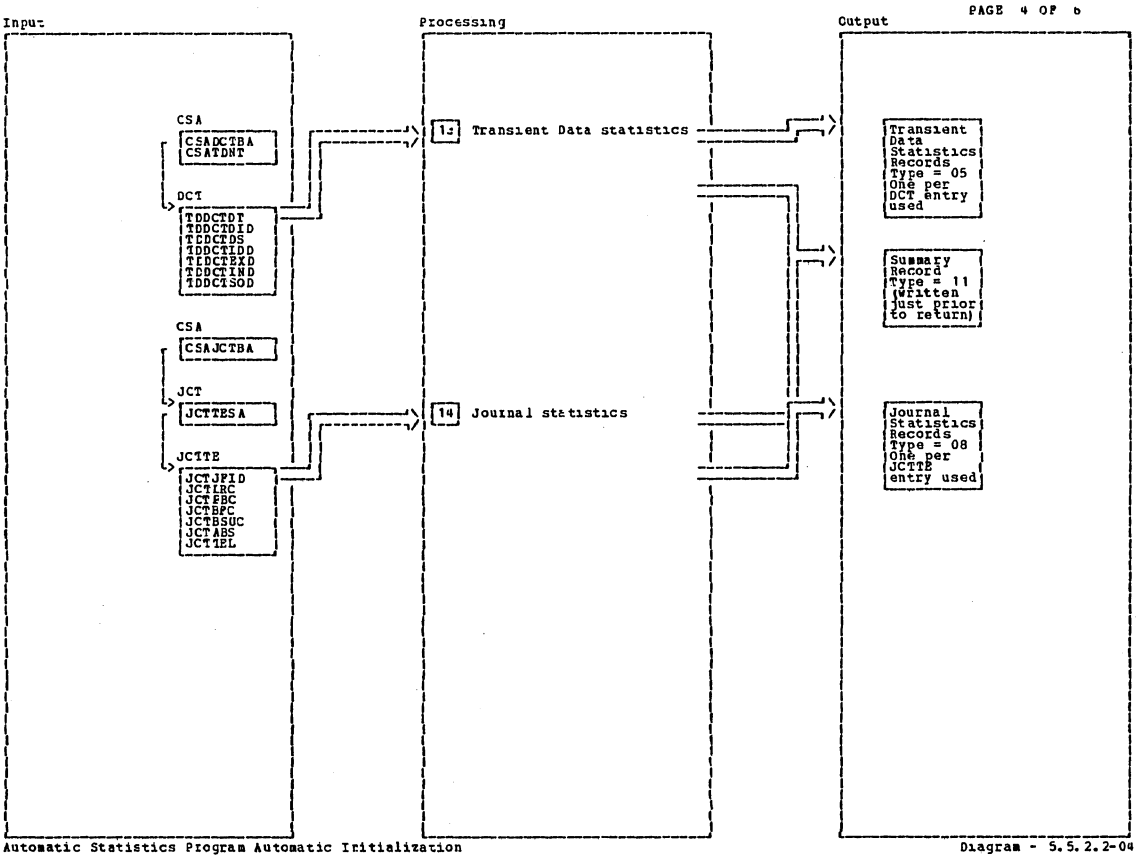
Automatic Statistics Program Automatic Initialization

Diagram - 5.5.2.2-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 12 | DFHSTSP | TERMINAL | | | | | |

Automatic Statistics Program Automatic Initialization

Diagram - 5.5.2.2-03


PAGE 4 OF 6
Input
Processing
Output
CSA
CSADCTBA
CSATDNT
DCT
TDDCTDT
TDDCTDID
TDDCTDS
TDDCTIDD
TDDCTEXD
TDDCTIND
TDDCTSOD
CSA
CSAJCTBA
JCT
JCTTESA
JCTTE
JCTJFID
JCTIRC
JCTFBC
JCTBFC
JCTBSUC
JCTABS
JCTTEL
13 Transient Data statistics
14 Journal statistics
Transient Data Statistics Records Type = 05 One per DCT entry used
Summary Record Type = 11 (written just prior to return)
Journal Statistics Records Type = 08 One per JCTTE entry used

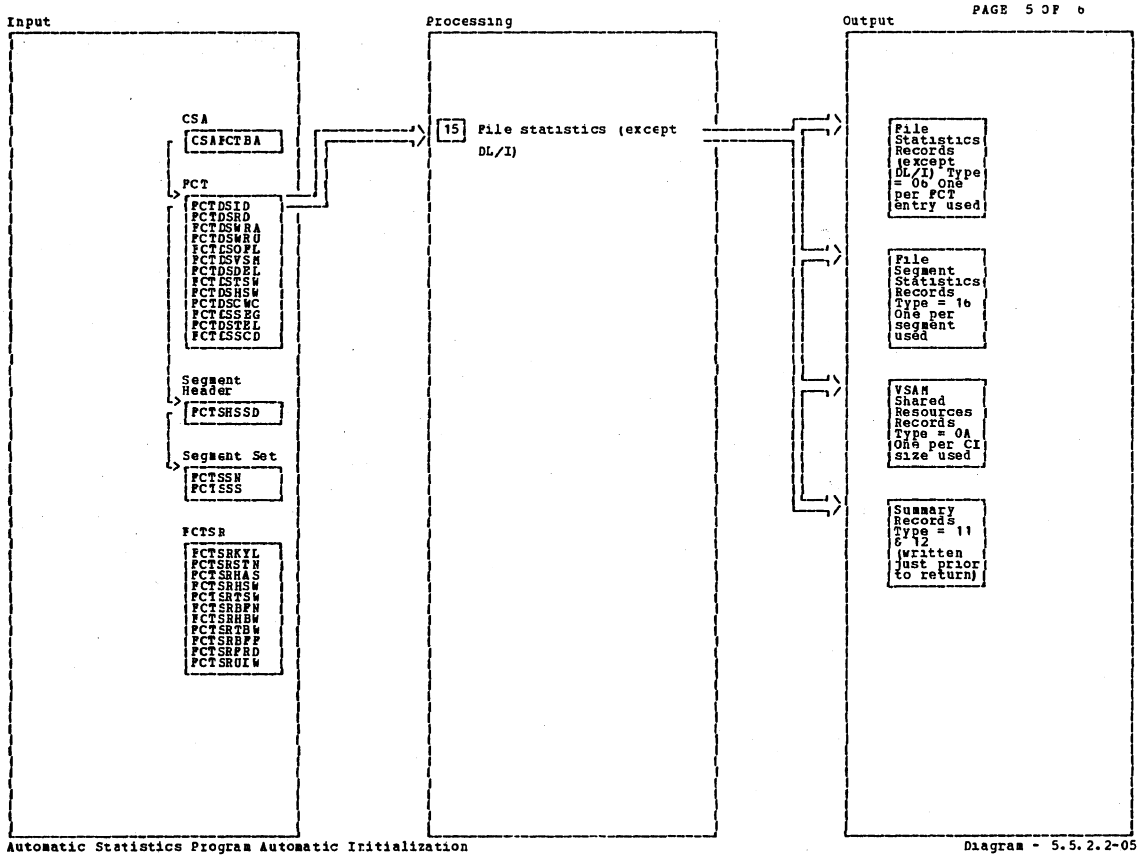
Automatic Statistics Program Automatic Initialization

Diagram - 5.5.2.2-04

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 13 | DFHSTSP | TRANDATA | | | | | |
| 14 | DFHSTSP | JOURNAL | | | | | |

Automatic Statistics Program Automatic Initialization

Diagram - 5.5.2.2-04

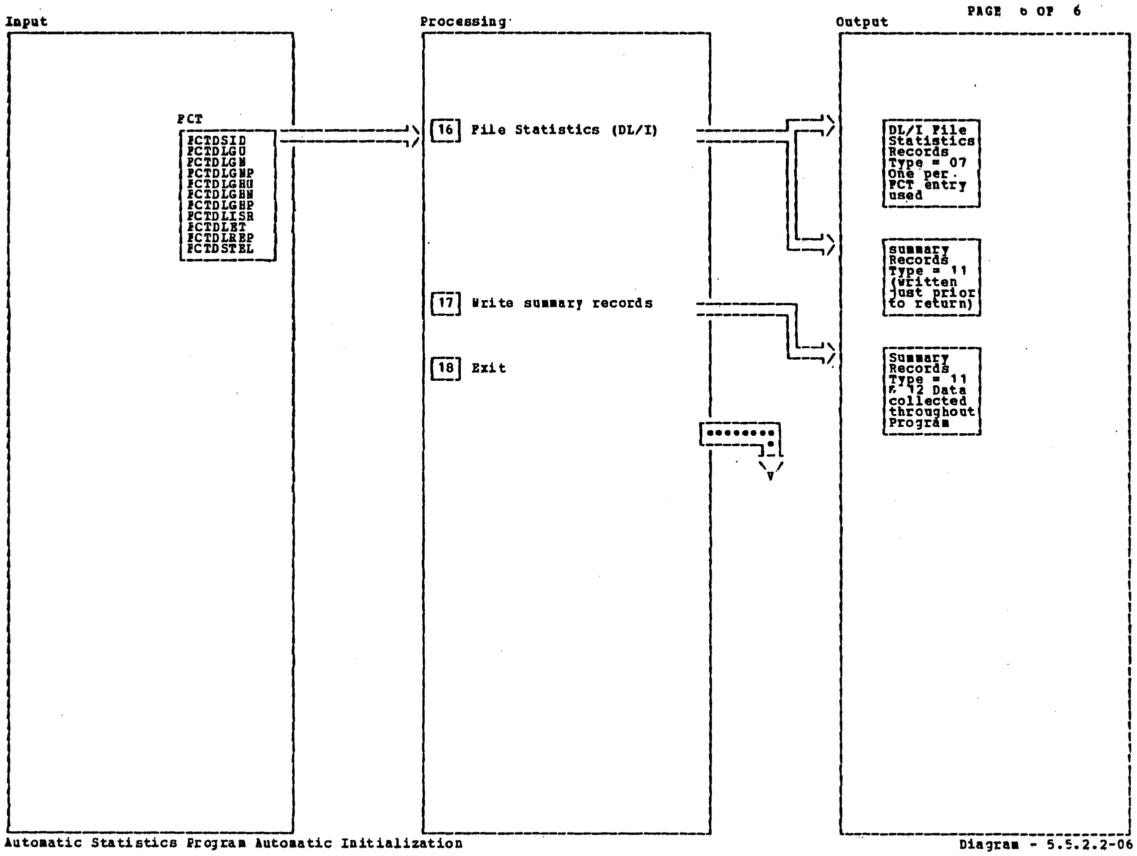

Input
CSA
CSAFCTBA
FCT
FCTDSID
FCTDSRD
FCTDSWRA
FCTDSWRU
FCTDSOPL
FCTDSVSM
FCTDSDEL
FCTDSTSW
FCTDSHSW
FCTDSCWC
FCTDSSEG
FCTDSTEL
FCTDSSCD
Segment Header
FCTSHSSD
Segment Set
FCTSSN
FCTSSS
FCTSR
FCTSRKYL
FCTSRSTN
FCTSRHAS
FCTSRHSW
FCTSRTSW
FCTSRBFN
FCTSRHBW
FCTSRTBW
FCTSRBFP
FCTSRFRD
FCTSRUIW
Processing
15 File statistics (except DL/I)
Output
File Statistics Records (except DL/I) Type = 06 One per FCT entry used
File Segment Statistics Records Type = 16 One per segment used
VSAM Shared Resources Records Type = 0A One per CI size used
Summary Records Type = 11 & 12 (written just prior to return)

Automatic Statistics Program Automatic Initialization Diagram - 5.5.2.2-05

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 15 Two passes of tae FCT are made. The first gathers all statistics execpt DL/I and notes if DL/I files are present. | DFHSTSP | FILE | | | | | |

Automatic Statistics Program Automatic Initialization Diagram - 5.5.2.2-05

PAGE 6 OF 6


Input
Processing
Output
FCT
FCTDSID
FCTDLGU
FCTDLGN
FCTDLGNP
FCTDLGHU
FCTDLGHN
FCTDLGHP
FCTDLISR
FCTDLET
FCTDLREP
FCTDSTEL
16 File Statistics (DL/I)
17 Write summary records
18 Exit
DL/I File Statistics Records Type = 07 One per. FCT entry used
summary Records Type = 11 (written just prior to return)
Summary Records Type = 11 & 12 Data collected throughout Program

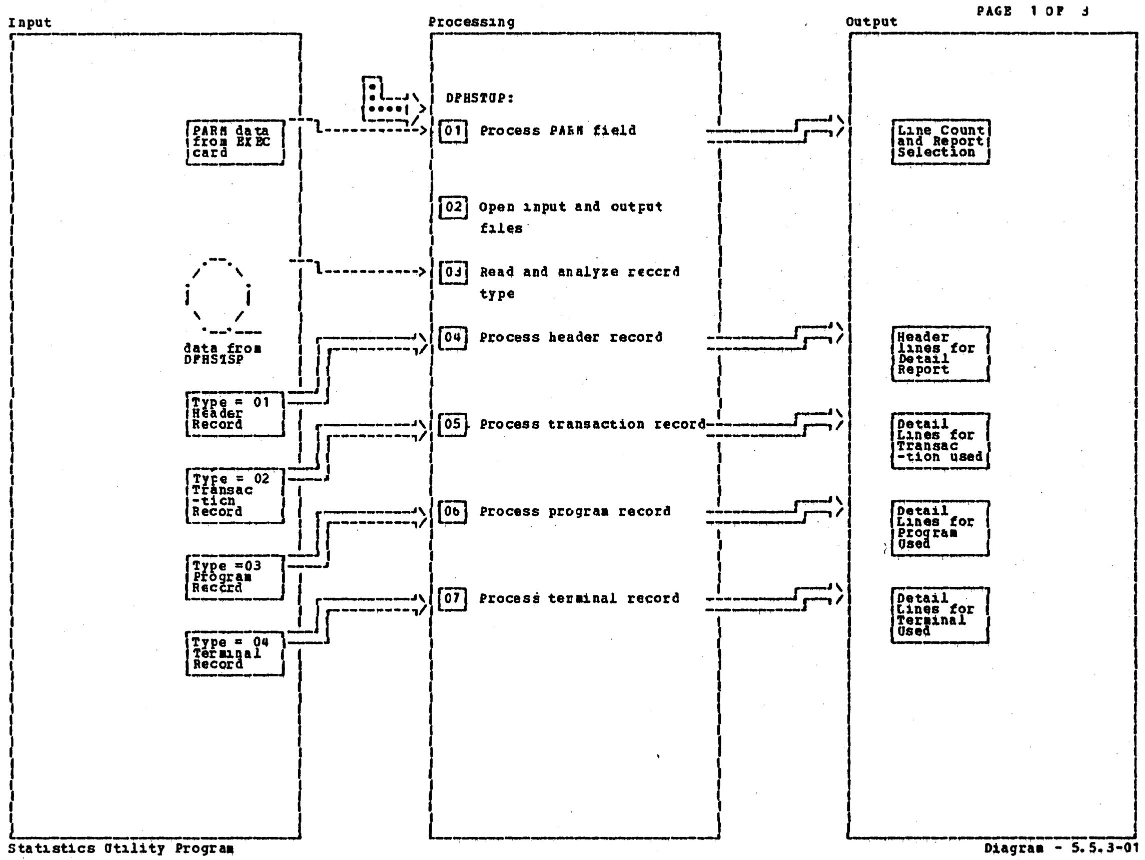
Automatic Statistics Program Automatic Initialization

Diagram - 5.5.2.2-06

| Notes | Boutine | Label | Bef | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 16 The second pass over the FCT is made only for DL/I files, if any are present. | DFHSTSP | DLIFILES | | | | | |
| 17 | DFHSTSP | TOTALS | | | | | |

Automatic Statistics Program Automatic Initialization

Diagram - 5.5.2.2-06


PAGE 1 OF 3
Input
Processing
Output
PARM data from EXEC card
data from DFHSTSP
Type = 01 Header Record
Type = 02 Transac -tion Record
Type =03 Program Record
Type = 04 Terminal Record
DFHSTUP:
01 Process PARM field
02 Open input and output files
03 Read and analyze record type
04 Process header record
05 Process transaction record
06 Process program record
07 Process terminal record
Line Count and Report Selection
Header lines for Detail Report
Detail Lines for Transac -tion used
Detail Lines for Program Used
Detail Lines for Terminal Used

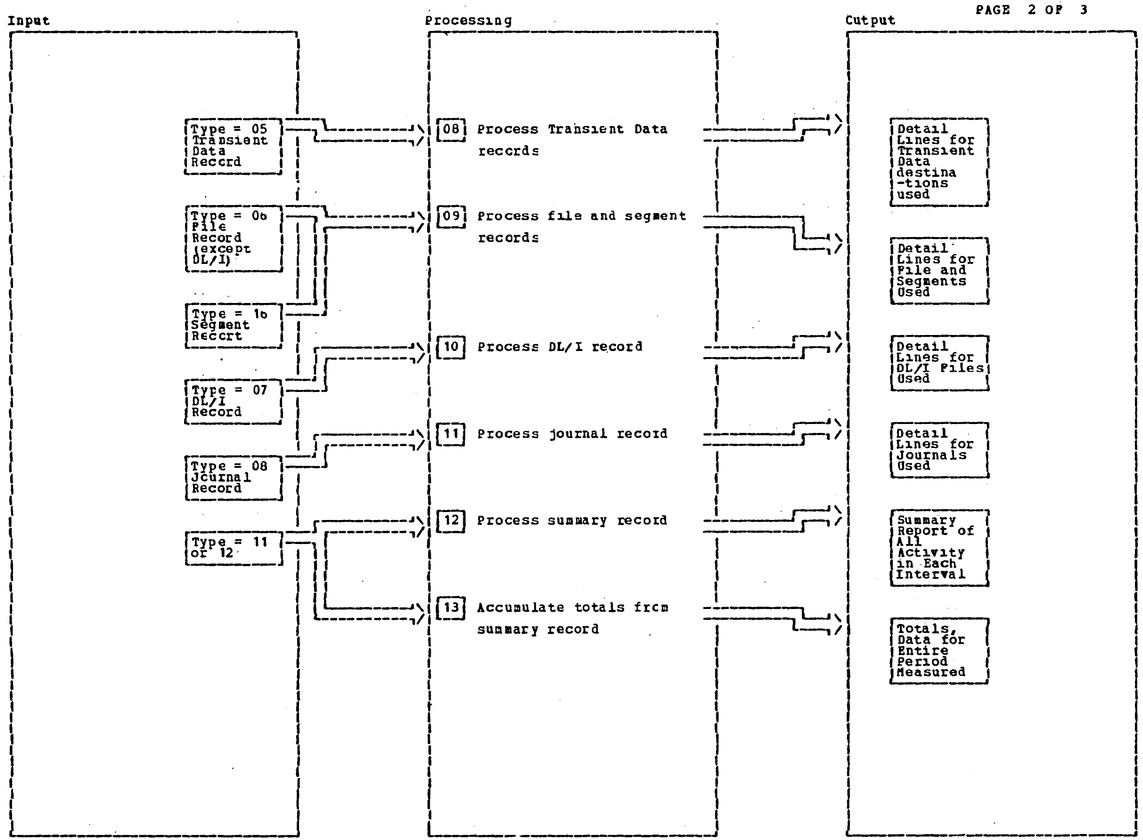
Statistics Utility Program Diagram - 5.5.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The PARM data can specify the number of lines to be printed per page of report and if the detail report is to be produced. | DFHSTUP | PARMCH | |
| 02 | CFHSTUP | OPENFILE | |
| 03 If the detail report was not specified, only type II(summary) records are processed. | CFESTUP | GETREC | |
| 04 | DFHSTUF | INTHEADR | |
| 05 | DFHSTUF | TRANSACT | |
| 06 | DFHSTUP | PROGRAM | |
| 07 | CFHSTUF | TERMINAL | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Statistics Utility Program Diagram - 5.5.3-01

PAGE 2 OF 3


Input
Processing
Cutput
Type = 05 Transient Data Record
Type = 06 File Record (except DL/I)
Type = 16 Segment Record
Type = 07 DL/I Record
Type = 08 Journal Record
Type = 11 or 12
08 Process Transient Data records
09 Process file and segment records
10 Process DL/I record
11 Process journal record
12 Process summary record
13 Accumulate totals from summary record
Detail Lines for Transient Data destina-tions used
Detail Lines for File and Segments Used
Detail Lines for DL/I Files Used
Detail Lines for Journals Used
Summary Report of All Activity in Each Interval
Totals, Data for Entire Period Measured

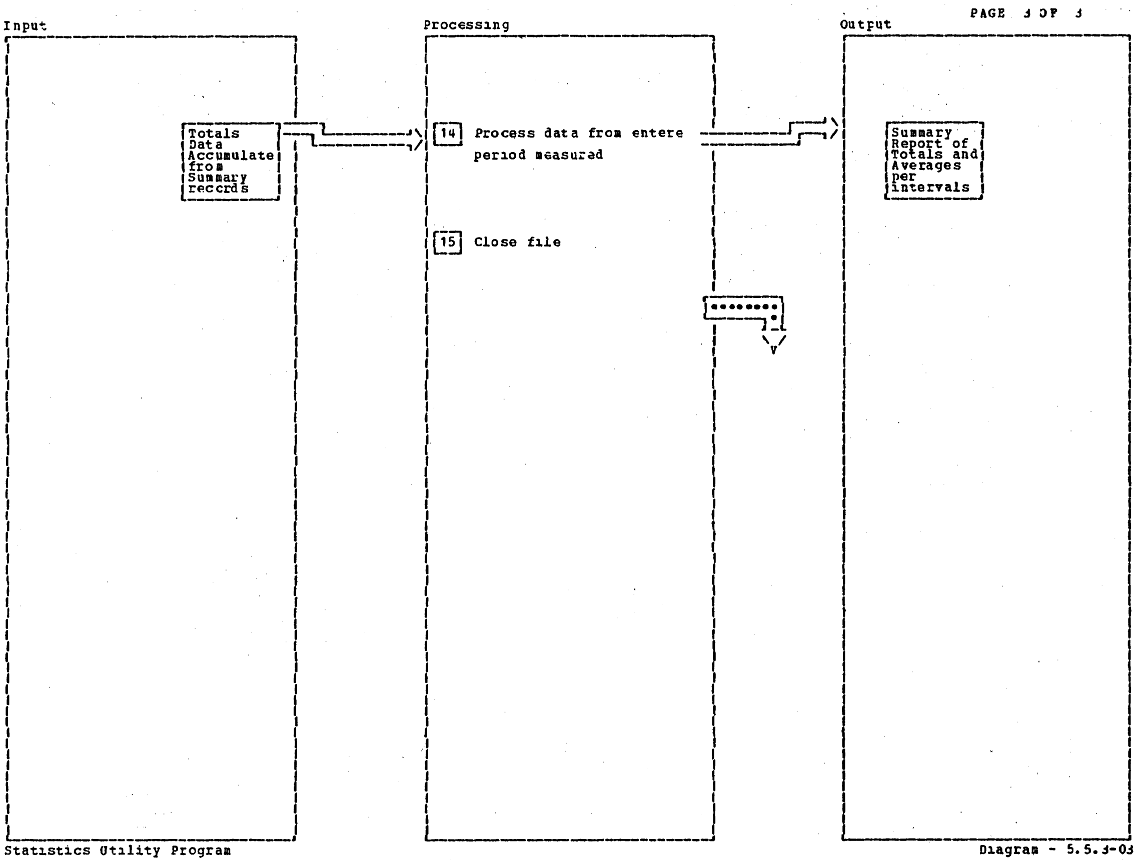
Statistics Utility Program

Diagram - 5.5.3-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 08 | DFHSTUP | TRANDATA | | | | | |
| 09 | DFHSTUP | FILE | | | | | |
| 10 | DFHSTUP | DLIFILE | | | | | |
| 11 | DFHSTUP | JOURNAL | | | | | |
| 12 Summary data is collected for up to 10 intervals and printed after 10 intervals or at end of file. | DFHSTUP | TOTALS | | | | | |
| 13 In addition, totals for the entire period measured are accumulated from all the summary records to be printed at end of file. | DFHSTUP | TOTALS1 | | | | | |

Statistics Utility Program

Diagram - 5.5.3-02


PAGE 3 OF 3
Input
Processing
Output
Totals Data Accumulate from Summary records
14 Process data from entere period measured
15 Close file
Summary Report of Totals and Averages per intervals

Statistics Utility Program

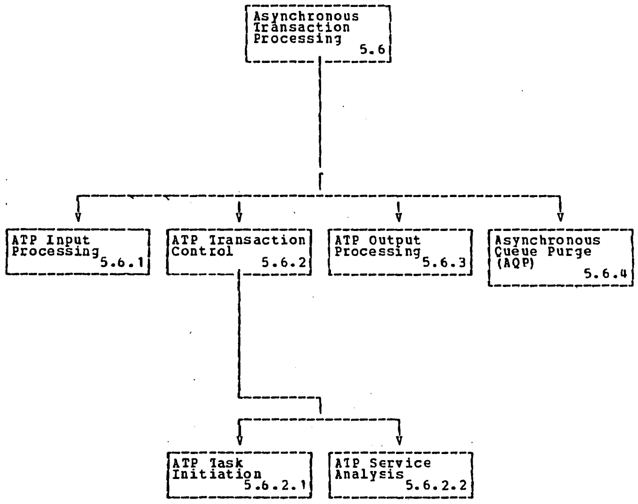
Diagram - 5.5.3-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 14 All totals are divided by the number of intervals; totals and averages are printed. | DFHSTUP | PERIOD | | | | | |

Statistics Utility Program

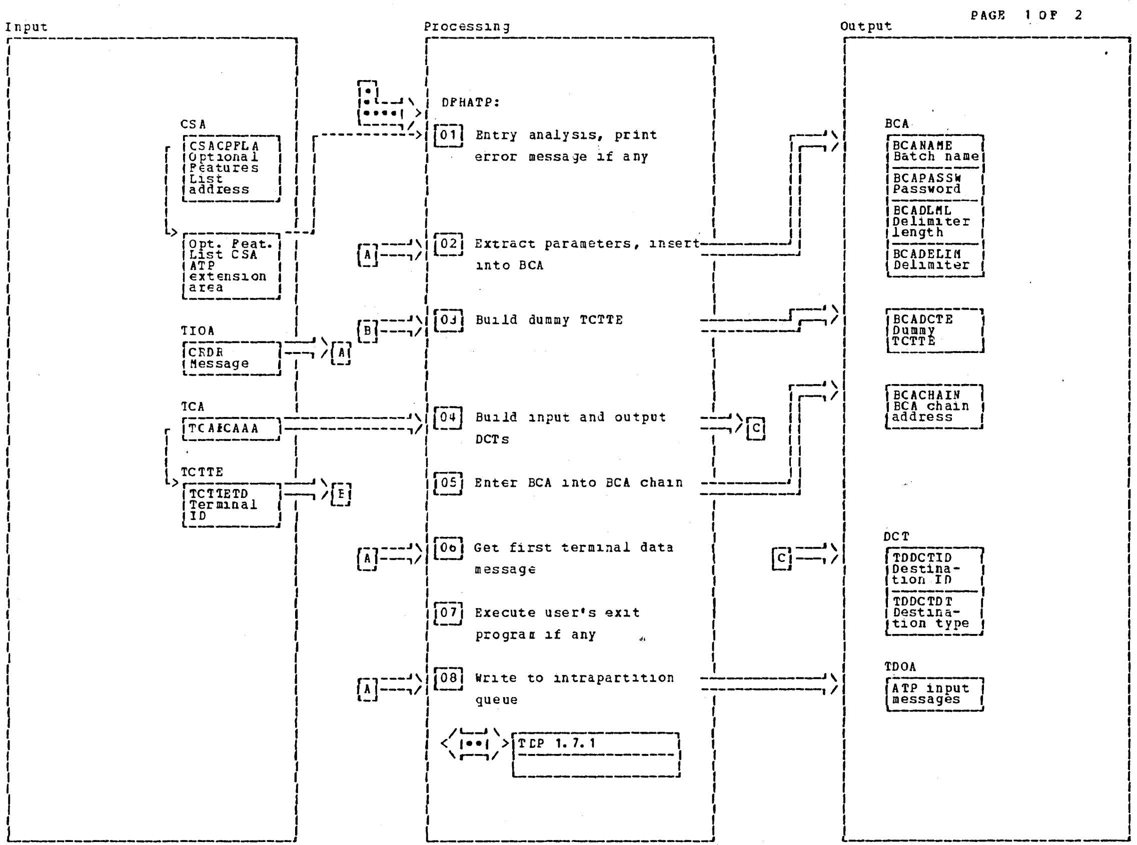
Diagram - 5.5.3-03

PAGE 1 OF 1


Asynchronous Transaction Processing 5.6
ATP Input Processing 5.6.1
ATP Transaction Control 5.6.2
ATP Output Processing 5.6.3
Asynchronous Queue Purge (AQP) 5.6.4
ATP Task Initiation 5.6.2.1
ATP Service Analysis 5.6.2.2

Asyncaronous Iransaction Processing

Diagram - 5.6-C1


PAGE 1 OF 2
Input
Processing
Output
CSA
CSACPPLA Optional Features List address
Opt. Feat. List CSA ATP extension area
TIOA
CRDR Message
TCA
TCAFCAAA
TCTTE
TCTTETD Terminal ID
DFHATP:
01 Entry analysis, print error message if any
02 Extract parameters, insert into BCA
03 Build dummy TCTTE
04 Build input and output DCTs
05 Enter BCA into BCA chain
06 Get first terminal data message
07 Execute user's exit program if any
08 Write to intrapartition queue
TCP 1.7.1
A
B
C
E
BCA
BCANAME Batch name
BCAPASSW Password
BCADLML Delimiter length
BCADELIM Delimiter
BCADCTE Dummy TCTTE
BCACHAIN BCA chain address
DCT
TDDCTID Destination ID
TDDCTDT Destination type
TDOA
ATP input messages

ATP Input Processing

Diagram - 5.6.1-01

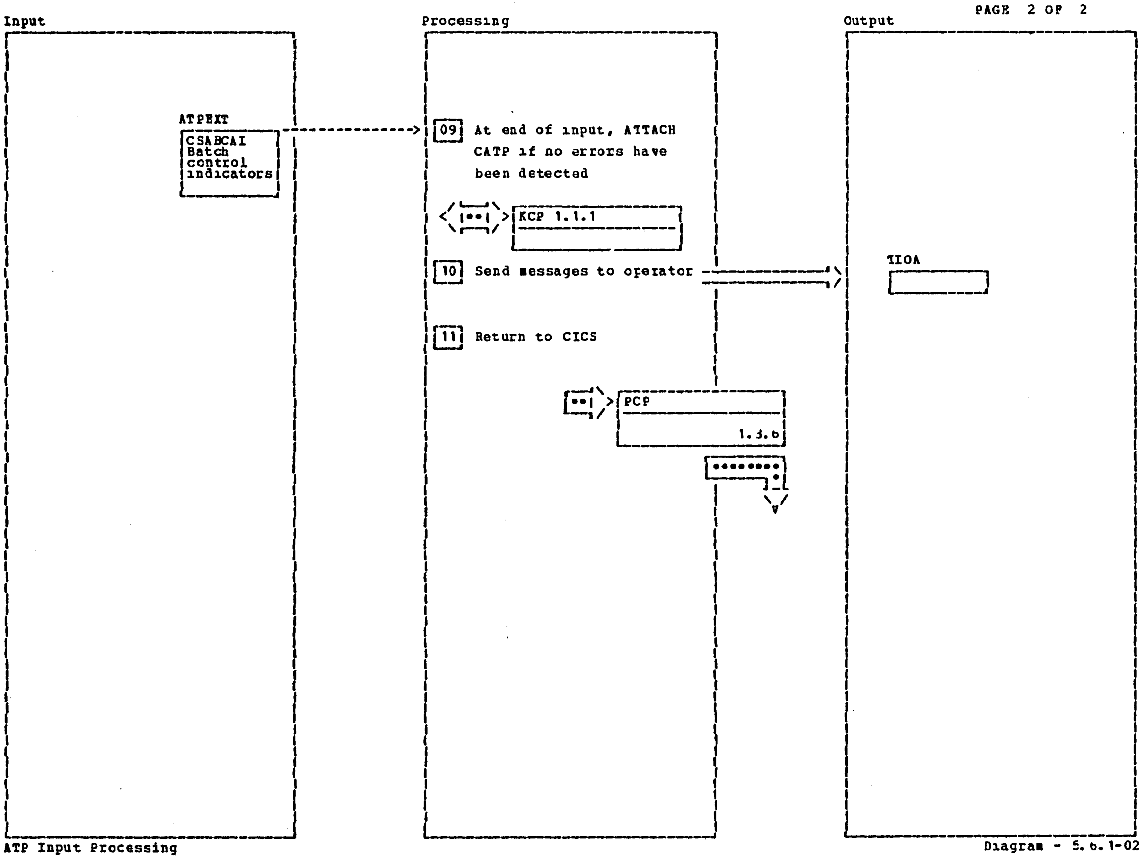
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 DFHRD1 is entered from DFHRD2 from the Terminal Control program in response to a CRDR command. | DFHRD1 | DFHRD1NA | |
| 03 The CRDR TCTTE is copied to the BCA and freed from the TCA and terminal storage. | | | |
| 04 The Destination IDs are constructed from task ID concatenated with a destination character. | | | |
| 05 | | RDRCHAIN | |
| 06 | DFHRD2 | | |
| 07 | | RDRXNTRY | |
| 08 Input messages are blocked in the TDOA. When the block is full, a DFHTD TYPE=PUT macro instruction is issued. | | RDRTDPUT | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

ATP Input Processing

Diagram - 5.6.1-01

PAGE 2 OF 2



09 At end of input, ATTACH CATP if no errors have been detected

10 Send messages to operator

11 Return to CICS

ATP Input Processing

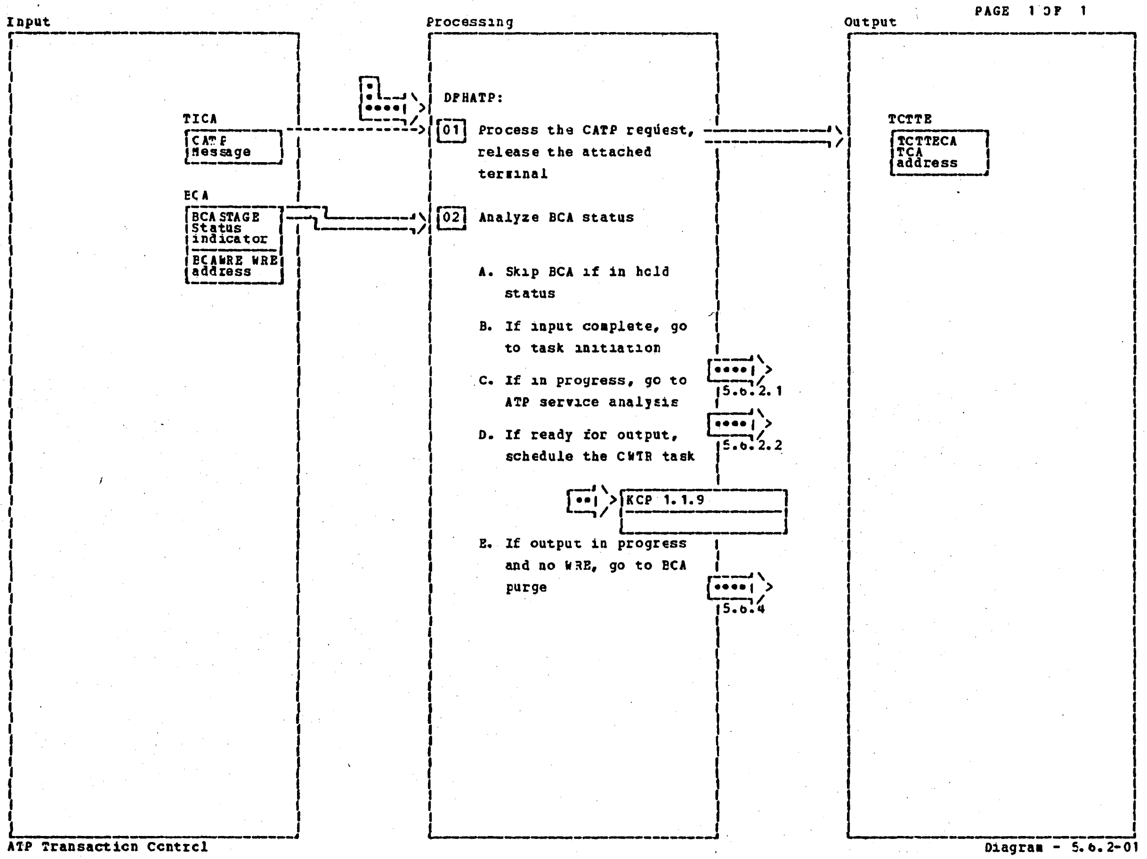
Diagram - 5.6.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 09 A DFHKC TYPE=ATTACH macro instruction is issued. | | | | | | | |
| 10 | DFHFC1 | RDRMSGP | | | | | |
| 11 A DFHPC TYPE=RETURN macro instruction is issued. | | RDRPAN | | | | | |

ATP Input Processing

Diagram - 5.6.1-02

PAGE 1 OF 1



DFHATP:

01 Process the CATP request, release the attached terminal

02 Analyze BCA status

A. Skip BCA if in hold status

B. If input complete, go to task initiation

C. If in progress, go to ATP service analysis

D. If ready for output, schedule the CWTR task

E. If output in progress and no WRE, go to BCA purge

ATP Transaction Control

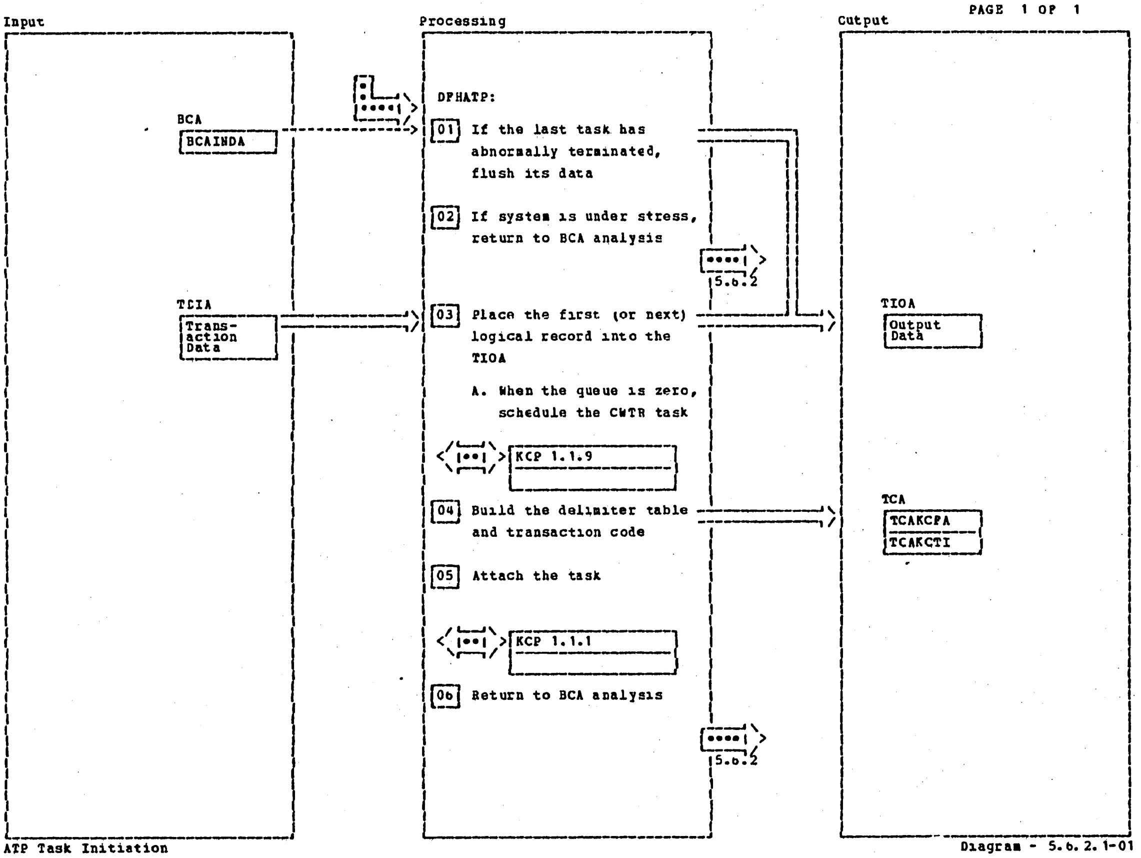
Diagram - 5.6.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 ATP is attached by a CRDR transaction or by the Terminal Control program in response to a terminal command (CATP). If attached by CRDR, step 1 is bypassed. | CREATP | | | | | | |
| 02 | | ATPBCANL | | | | | |
| D. A DFHKC TYPE=SCEEDULE macro is issued. | | ATPOPSCH | | | | | |

ATP Transaction Control

Diagram - 5.6.2-01

PAGE 1 OF 1



DFHATP:

01 If the last task has abnormally terminated, flush its data

02 If system is under stress, return to BCA analysis

03 Place the first (or next) logical record into the TIOA

A. When the queue is zero, schedule the CWTR task

04 Build the delimiter table and transaction code

05 Attach the task

06 Return to BCA analysis

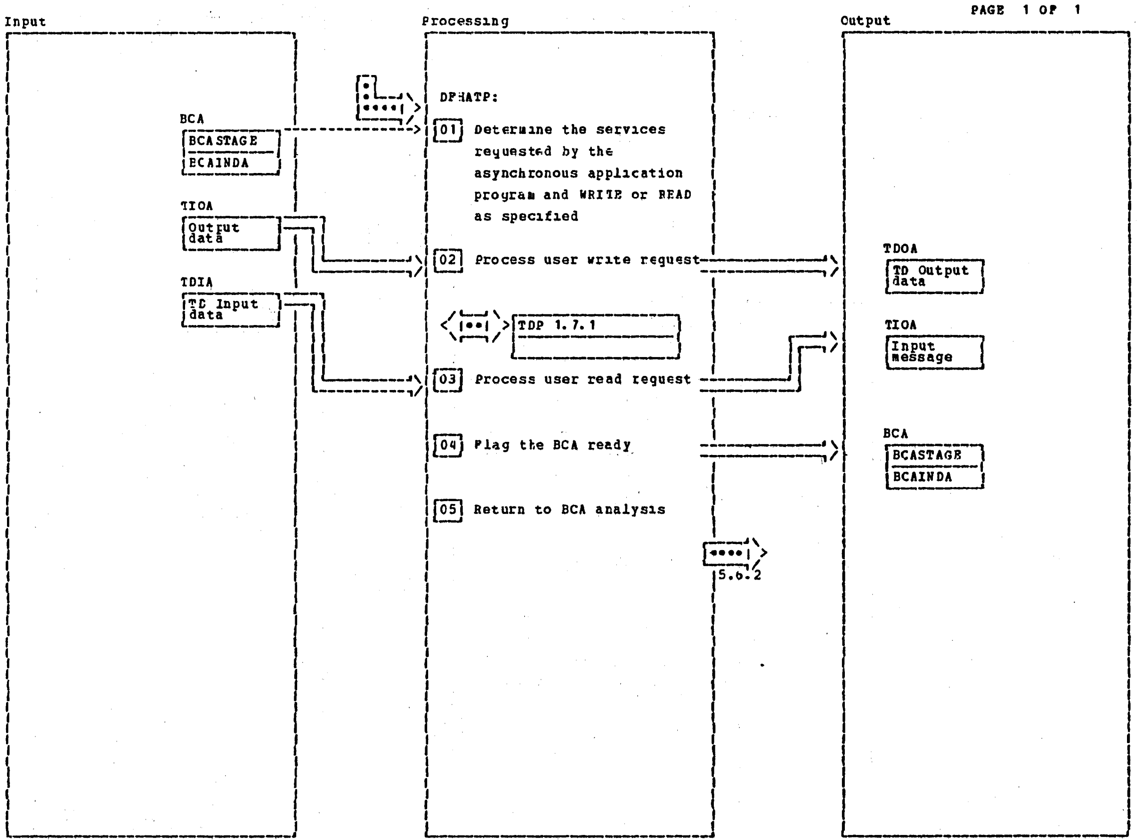
ATP Task Initiation

Diagram - 5.6.2.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 This entry is used when one asynchronous task terminates and another may be ready to start. | | ATPINITP | | | | | |
| 02 This entry is used when a batch is marked ready to process. | | ATPINIT | | | | | |
| 03 | | ATPFETCH | | | | | |
| A. A DFHKC TYPE=SCHEDULE macro instruction is issued. | | ATPOPSCH | | | | | |
| 04 Delimiter table is built from the field start and field separator characters of the SIT. | | ATPATTCH | | | | | |
| 05 A DFHKC TYPE=ATTACH macro instruction is issued. | | ATPATCHB | | | | | |

ATP Task Initiation

Diagram - 5.6.2.1-01


PAGE 1 OF 1
Input
Processing
Output
BCA
BCASTAGE
BCAINDA
TIOA
Output data
TDIA
TD Input data
DFHATP:
01 Determine the services requested by the asynchronous application program and WRITE or READ as specified
02 Process user write request
TDP 1.7.1
03 Process user read request
04 Flag the BCA ready
05 Return to BCA analysis
5.6.2
TDOA
TD Output data
TIOA
Input message
BCA
BCASTAGE
BCAINDA

ATP Service Analysis

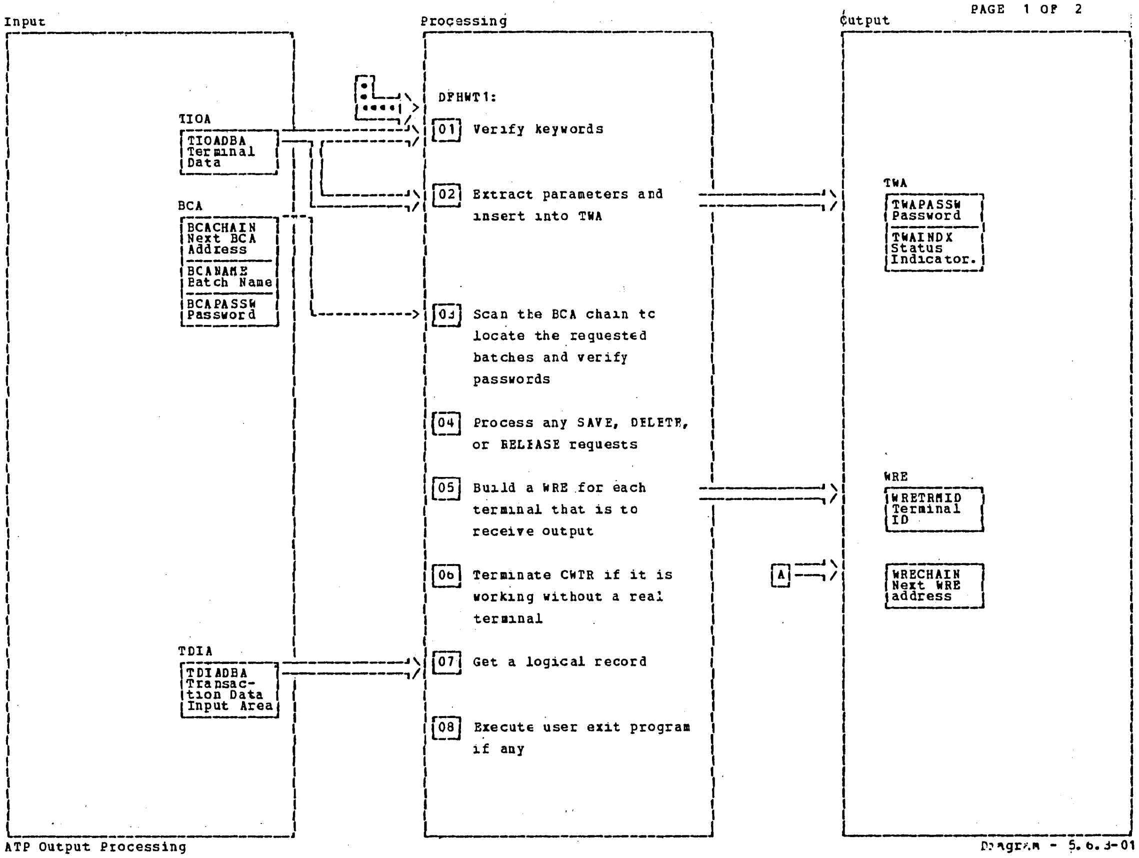
Diagram - 5.6.2.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 This routine is entered each time a BCA indicates an in progress status. | | ATPSVANL | | | | | |
| 02 Data is moved to the output buffer. When buffer is full, a DFHTD TYPE=PUT macro instruction is issued. | | ATPSVAPT | | | | | |
| 03 The next logical input record is moved from the TDIA buffer to the TIOA. | | ATPSVARD | | | | | |

ATP Service Analysis

Diagram - 5.6.2.2-01

PAGE 1 OF 2



DFHWT1:

01 Verify keywords

02 Extract parameters and insert into TWA

03 Scan the BCA chain to locate the requested batches and verify passwords

04 Process any SAVE, DELETE, or RELEASE requests

05 Build a WRE for each terminal that is to receive output

06 Terminate CWTR if it is working without a real terminal

07 Get a logical record

08 Execute user exit program if any

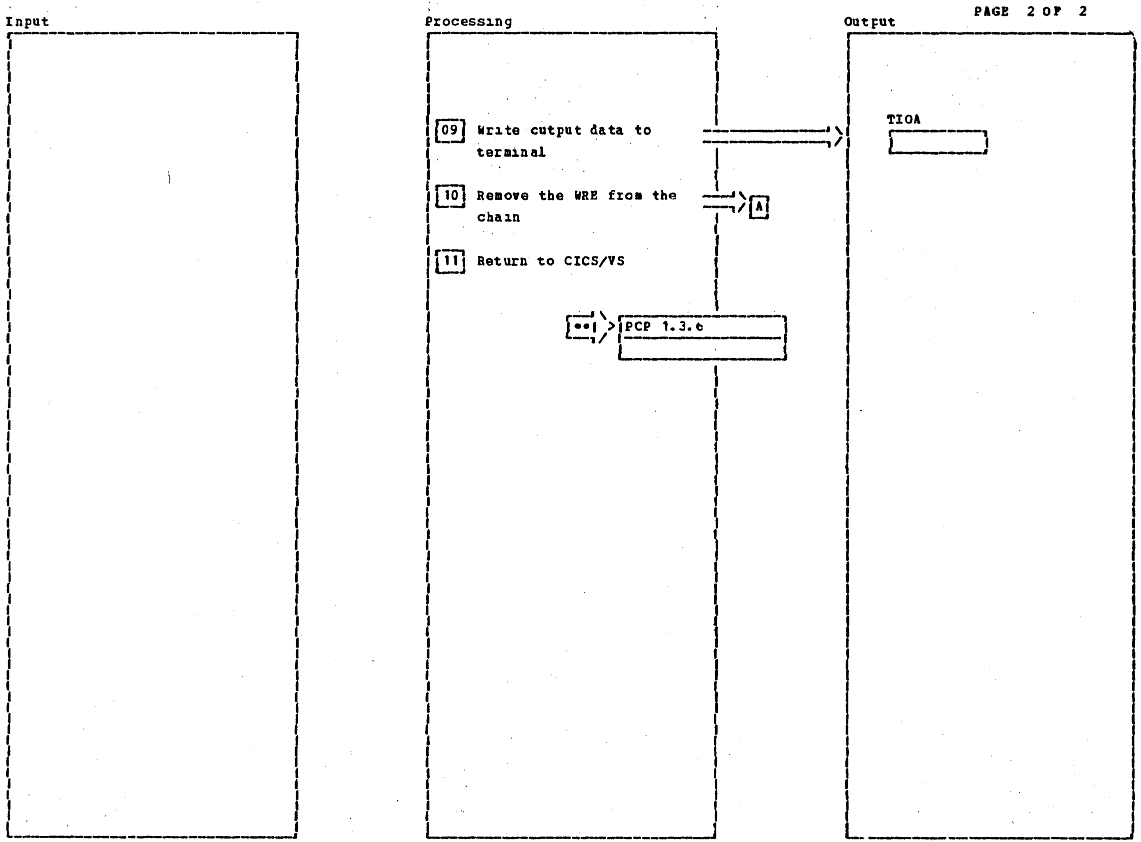
ATP Output Processing

Diagram - 5.6.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 DFHWT1 is entered in response to a CWTR or CSMT request. | DFHWT1 | | | | | | |
| 02 The request is scanned for the following keywords: DELETE, SAVE, RELEASE, and STATUS. | | | | | | | |
| 03 | | WTRGETAL | | | | | |
| 05 The WRE chain is scanned for duplicates and the WRE entered in the chain. | | | | | | | |
| 07 | DFHWT2 | | | | | | |
| 08 User may insert a record. | | | | | | | |

ATP Output Processing

Diagram - 5.6.3-01



ATP Output Processing

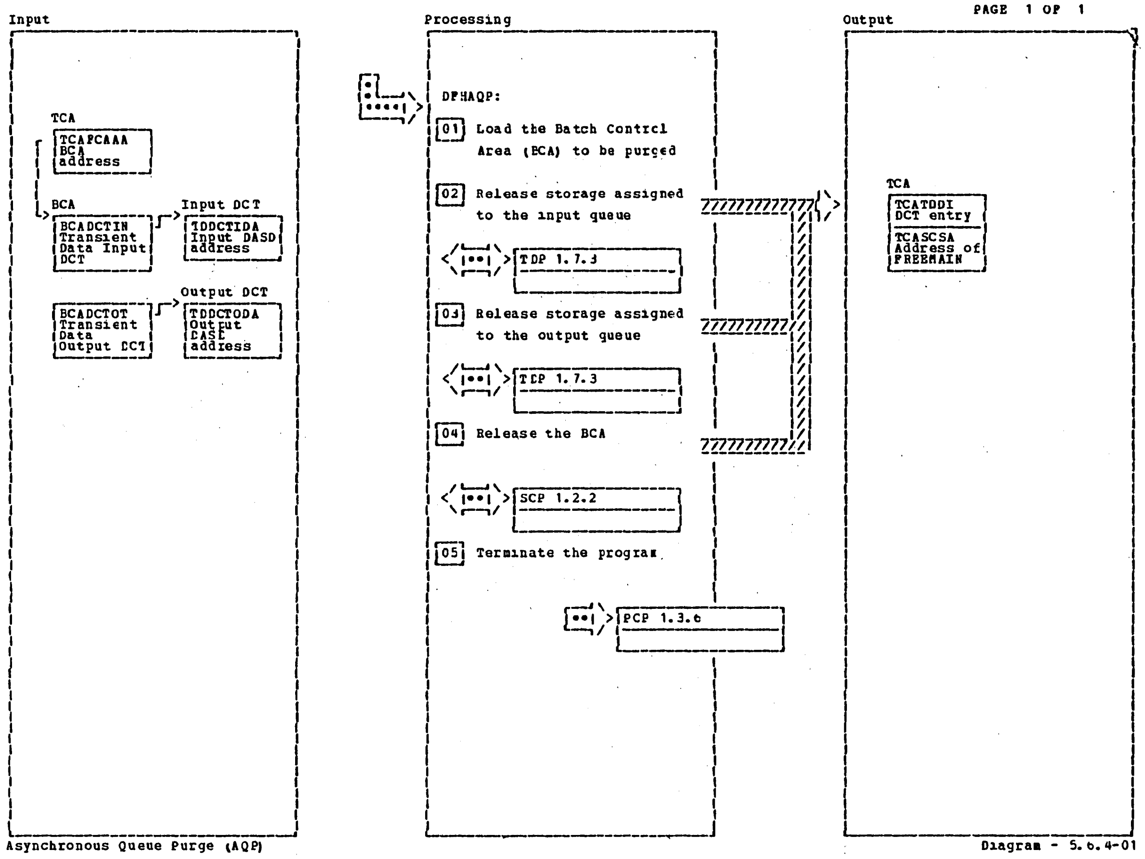
Diagram - 5.6.3-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 10 | | WRTUNCHN | | | | | |
| 11 A DFHPC TYPE=RETURN macro instruction is issued. | | | | | | | |

ATP Output Processing

Diagram - 5.6.3-02

PAGE 1 OF 1

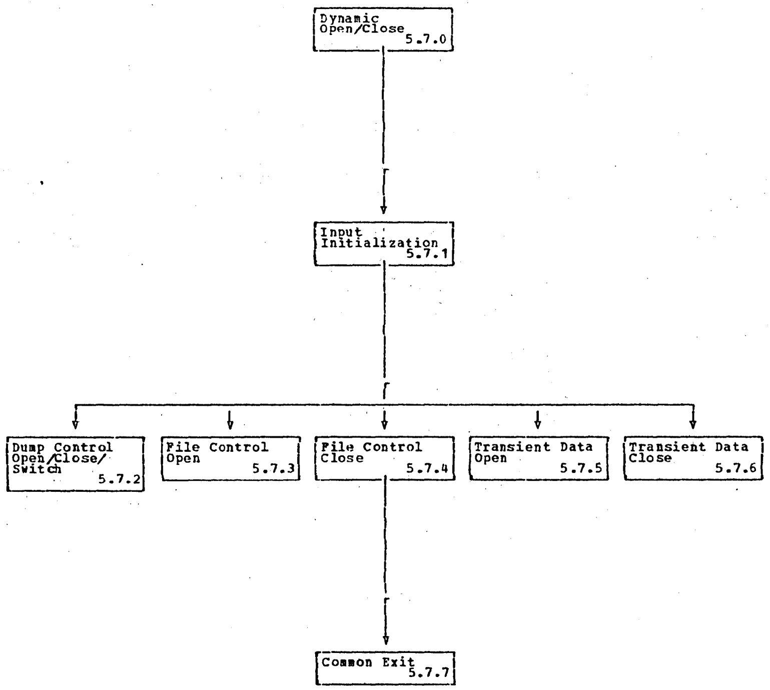

Input
Processing
Output
TCA
TCAFCAAA BCA address
BCA
BCADCTIN Transient Data Input DCT
Input DCT
TDDCTIDA Input DASD address
BCADCTOT Transient Data Output DCT
Output DCT
TDDCTODA Output DASD address
DFHAQP:
01 Load the Batch Control Area (BCA) to be purged
02 Release storage assigned to the input queue
TDP 1.7.3
03 Release storage assigned to the output queue
TDP 1.7.3
04 Release the BCA
SCP 1.2.2
05 Terminate the program
PCP 1.3.6
TCA
TCATDDI DCT entry
TCASCSA Address of FREEMAIN

Asynchronous Queue Purge (AQP) Diagram - 5.6.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 This module can be entered from either DFHATP or DFHWTR. | | | | | | | |
| 02 A Transient Data PURGE is issued to release the storage. | | AQPURGE | | | | | |
| 03 A Transient Data PURGE is issued to release the storage. | | | | | | | |
| 04 A Storage Control FREEMAIN is issued. | | AQPRLSE | | | | | |
| 05 A Program Control RETURN is issued. | | | | | | | |

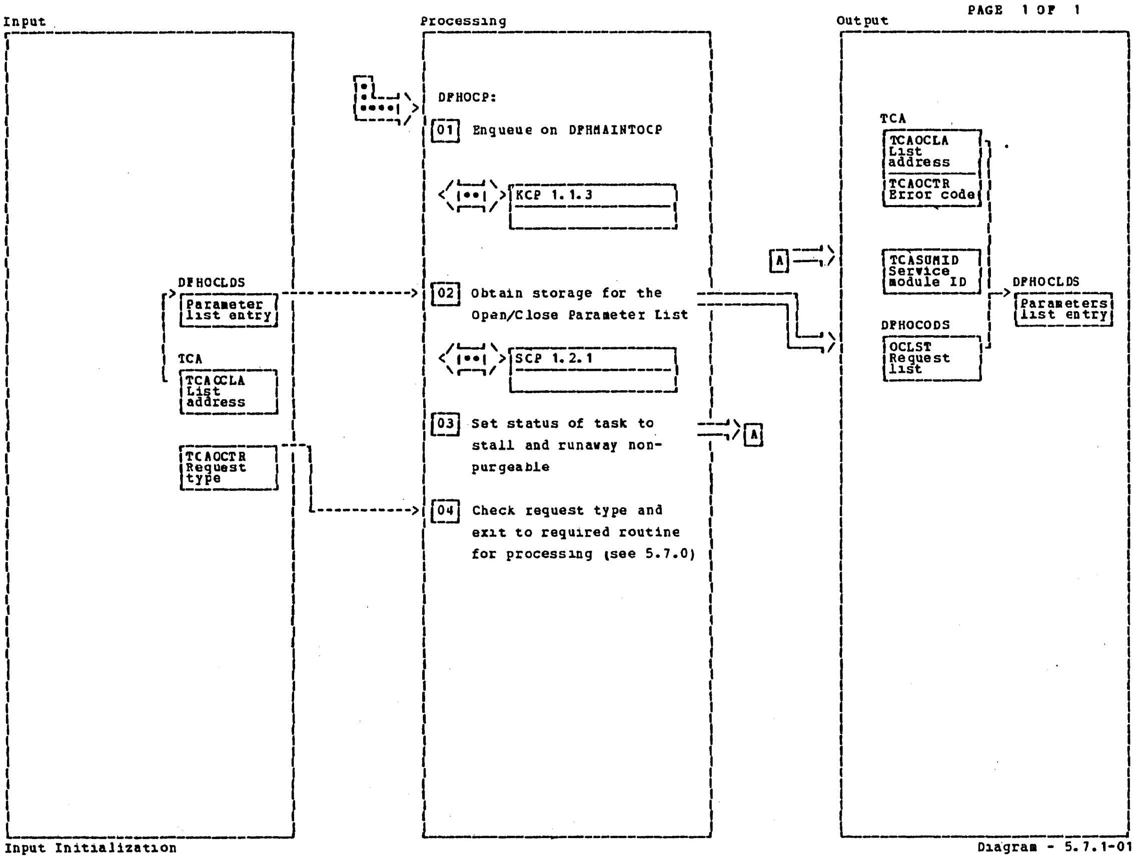
Asynchronous Queue Purge (AQP) Diagram - 5.6.4-01

PAGE 1 OF 1


Dynamic Open/Close 5.7.0
Input Initialization 5.7.1
Dump Control Open/Close/ Switch 5.7.2
File Control Open 5.7.3
File Control Close 5.7.4
Transient Data Open 5.7.5
Transient Data Close 5.7.6
Common Exit 5.7.7

Dynamic Open/Close

Diagram - 5.7.0-01


PAGE 1 OF 1
Input
Processing
Output
DFHOCP:
01 Enqueue on DFHMAINTOCP
KCP 1.1.3
02 Obtain storage for the Open/Close Parameter List
SCP 1.2.1
03 Set status of task to stall and runaway non-purgeable
04 Check request type and exit to required routine for processing (see 5.7.0)
DFHOCLDS
Parameter list entry
TCA
TCAOCLA List address
TCAOCTR Request type
TCA
TCAOCLA List address
TCAOCTR Error code
TCASUMID Service module ID
DFHOCODS
OCLST Request list
DFHOCLDS
Parameters list entry
A
A

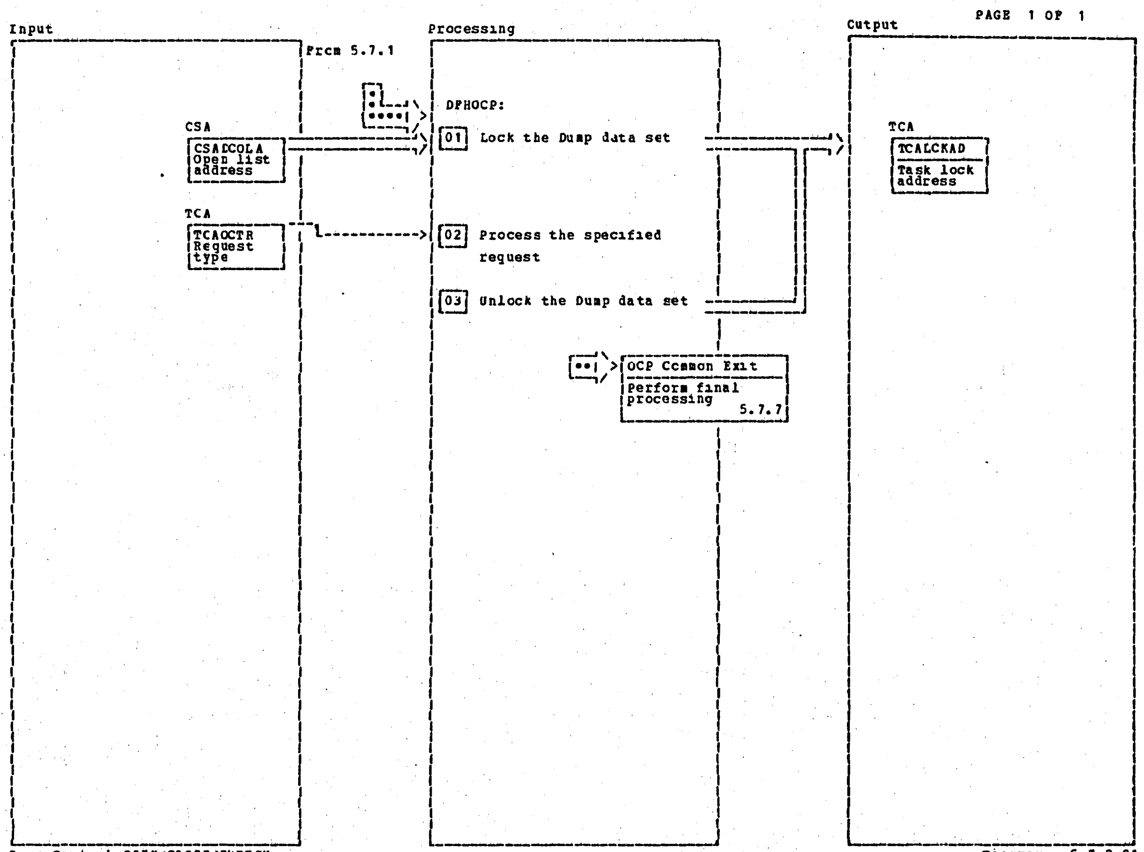
Input Initialization

Diagram - 5.7.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 This module is entered via a DFHOC TYPE=OPEN/CLOSE/SWICH or DFHPC TYPE=LINK macro instruction. | | OCPGMLPE | | | | | |
| 02 The number of requests cannot exceed the number of entries in the Open/Close Parameter List. A Storage Control GETMAIN is issued to obtain enough storage for all requests. | | OCPGMLP | | | | | |
| 03 | | OCPRGBP | | | | | |
| 04 | | OCPLSTBP | | | | | |

Input Initialization

Diagram - 5.7.1-01



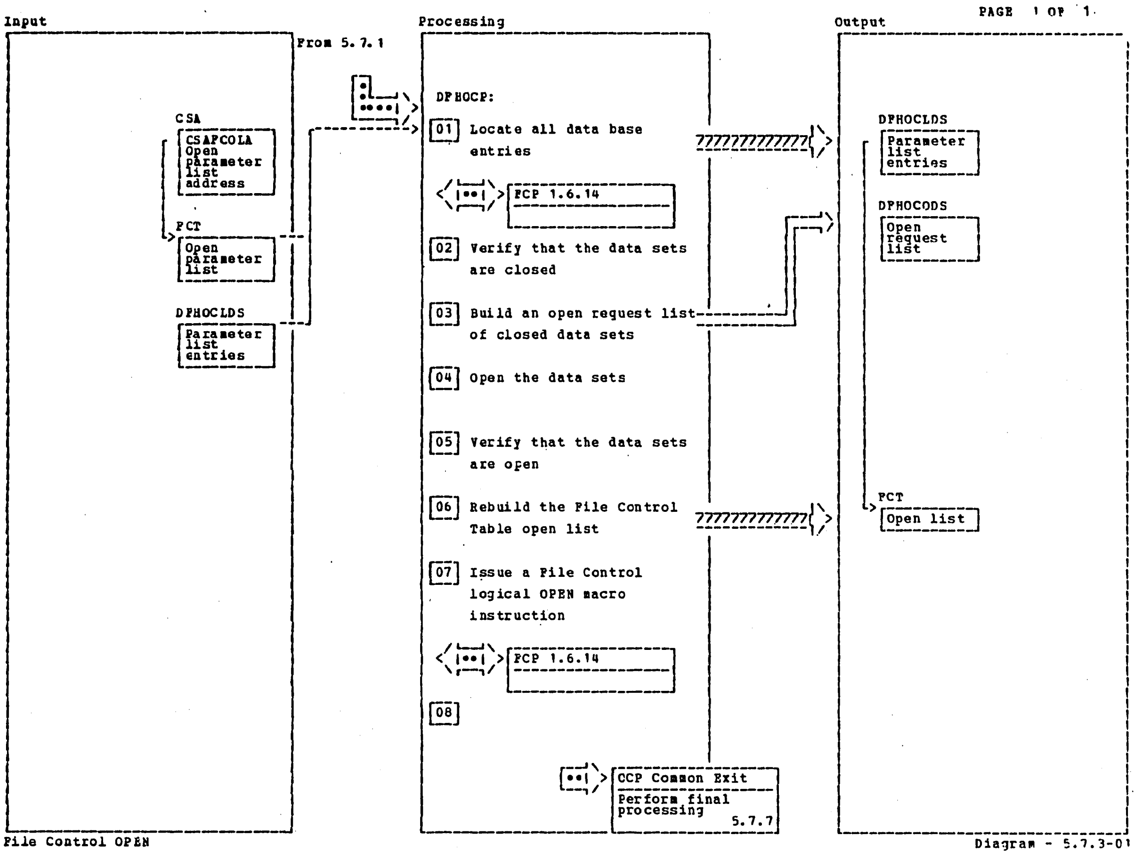
Dump Control OPEN/CLOSE/SWITCH

Diagram - 5.7.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Lock on the single thread lock in the Dump Control program. | | OCPDPCL | | | | | |
| 02 If TYPE=SWITCH or TYPE=CLOSE was specified, the Dump data set is closed. | | OCPDPCL | | | | | |
| If TYPE=SWITCH was specified the filename in the DTF is switched if the DUMP data set is disk. | | OCPDPCLX | | | | | |
| If TYPE=OPEN or TYPE=SWITCH was specified, the Dump data set is opened. | | OCPDPOP | | | | | |

Dump Control OPEN/CLOSE/SWITCH

Diagram - 5.7.2-01


PAGE 1 OF 1
Input
Processing
Output
From 5.7.1
CSA
CSAFCOLA
Open parameter list address
FCT
Open parameter list
DFHOCLDS
Parameter list entries
DFHOCP:
01 Locate all data base entries
FCP 1.6.14
02 Verify that the data sets are closed
03 Build an open request list of closed data sets
04 Open the data sets
05 Verify that the data sets are open
06 Rebuild the File Control Table open list
07 Issue a File Control logical OPEN macro instruction
FCP 1.6.14
08
OCP Common Exit
Perform final processing
5.7.7
DFHOCLDS
Parameter list entries
DFHOCODS
Open request list
FCT
Open list

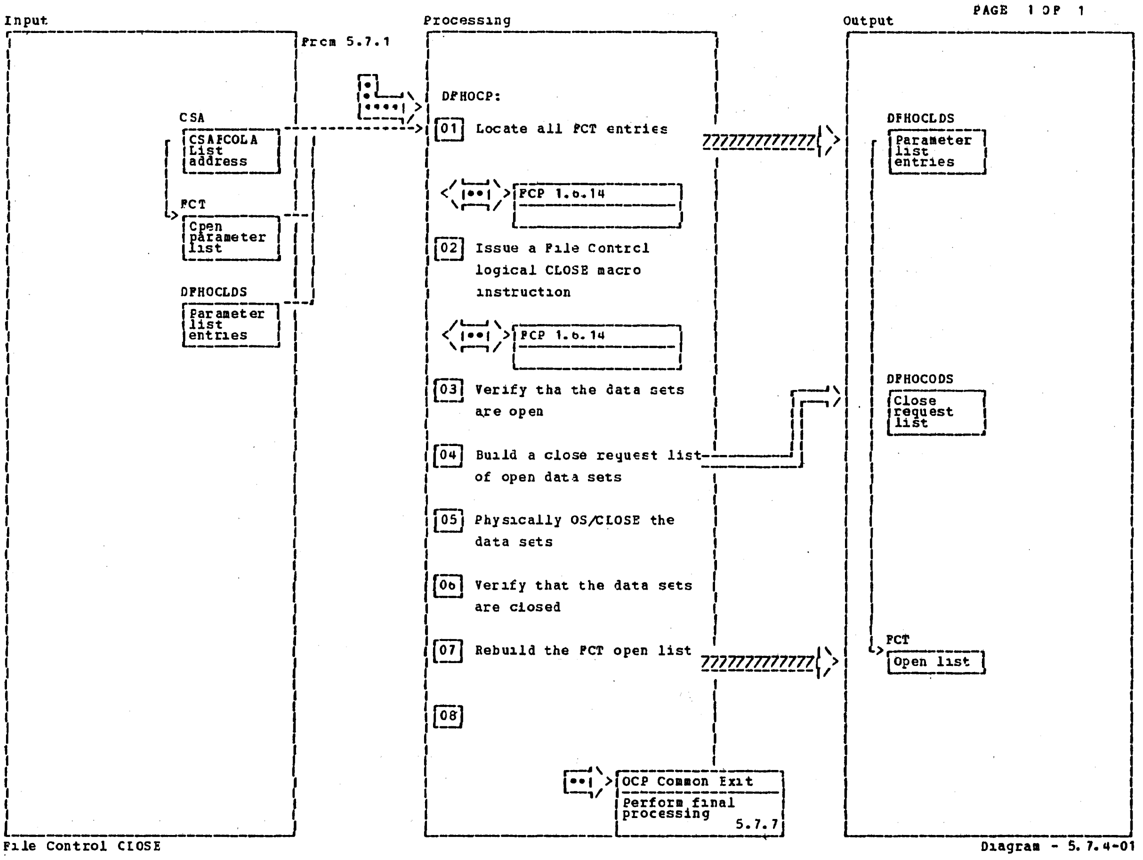
File Control OPEN

Diagram - 5.7.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 A File Control LOCATE macro is issued. | | OCPBBTN | | | | | |
| 02 | | OCPDBSTR | | | | | |
| 03 | | OCPHLST | | | | | |
| 04 This is an OS/VS physical open. | | OCPHLST | | | | | |
| 05 If the data set uses VSAM, the required FCT information is added. | | OCPFCLP2 | | | | | |
| 06 | | OCPDBOLP | | | | | |
| 07 | | OCPDBAO1 | | | | | |

File Control OPEN

Diagram - 5.7.3-01



01 Locate all FCT entries

02 Issue a File Control logical CLOSE macro instruction

03 Verify tha the data sets are open

04 Build a close request list of open data sets

05 Physically OS/CLOSE the data sets

06 Verify that the data sets are closed

07 Rebuild the FCT open list

08

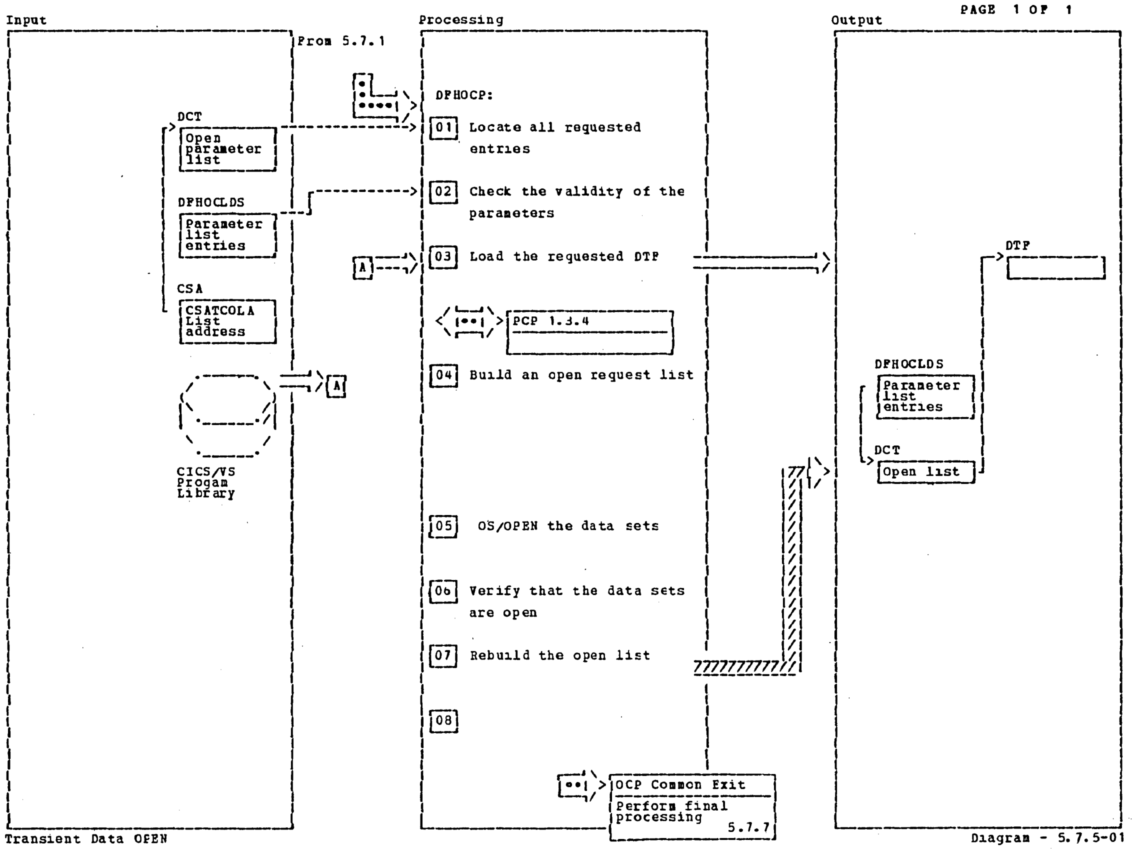
File Control CLOSE

Diagram - 5.7.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 A File Control LOCATE is issued to obtain the actual FCT address of the symbolically named entries. | | OCPDBRTN | | | | | |
| 02 | | OCPBCLRT | | | | | |
| 03 | | OCPBCLRL | | | | | |
| 04 | | OCPDBCLI | | | | | |
| 05 | | OCPDBCLI | | | | | |
| 06 | | OCPRESTL | | | | | |
| 07 | | OCPSCHE2 | | | | | |

File Control CLOSE

Diagram - 5.7.4-01



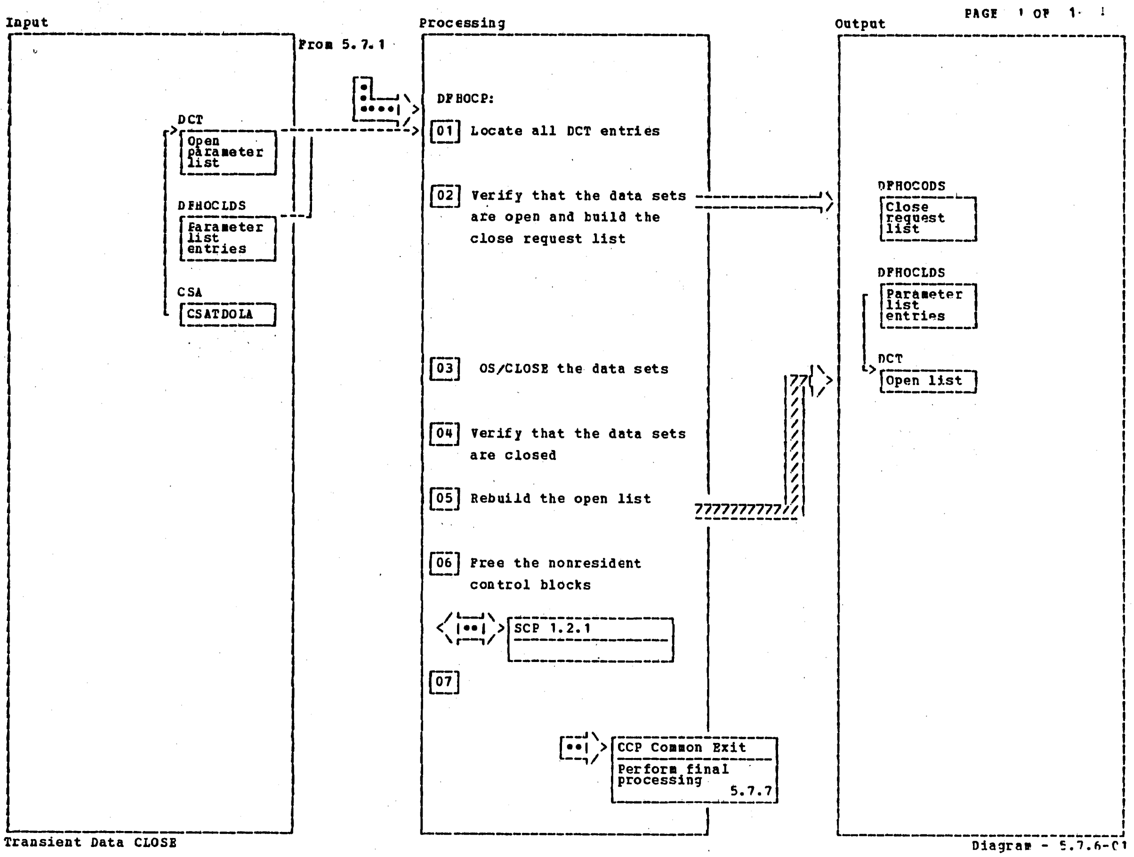
Transient Data OPEN

Diagram - 5.7.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | TDENFIND | | | | | |
| 02 This check is made only for nonresident data sets | | OCPRPOPN | | | | | |
| 03 The DTF is loaded only for nonresident data sets. It is loaded via a conditional program load with the RELOAD=YES parameter specified. | | LOADRTN | | | | | |
| 04 | | TDOPENEX | | | | | |
| 05 | | TDOPENEX | | | | | |
| 06 | | TDOPENLP | | | | | |
| 07 | | TDERRBYP | | | | | |

Transient Data OPEN

Diagram - 5.7.5-01



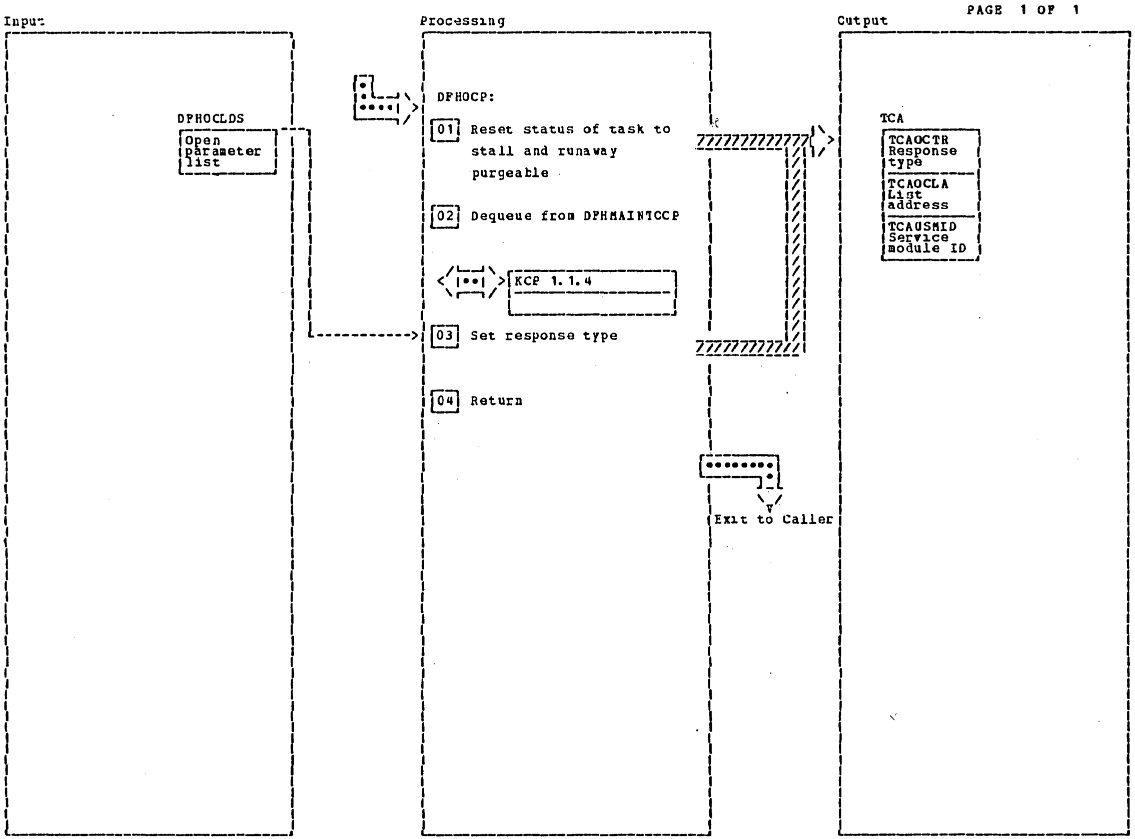
Transient Data CLOSE

Diagram - 5.7.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | OCTDRTN | | | | | |
| 02 | | OCPTOCLS | | | | | |
| 03 | | TOOCLAAD | | | | | |
| 04 | | DOCOCP20 | | | | | |
| 05 | | OCPOLSTF | | | | | |
| 06 | | OCPOLSTF | | | | | |

Transient Data CLOSE

Diagram - 5.7.6-01



Common Exit

Diagram - 5.7.7-01

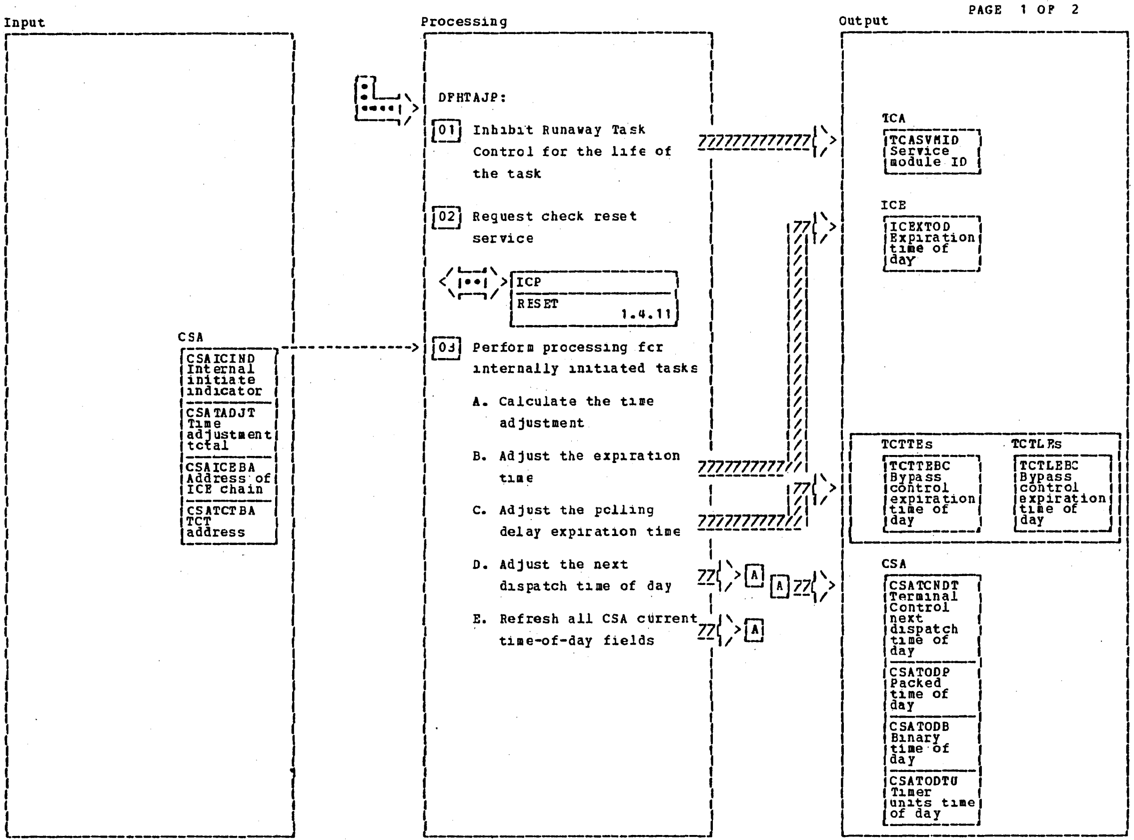
| Notes | Rcutine | Label | Ref |
|---|---|---|---|
| 01 This is the common exit for all OCP processing. | | OCPXT | |
| 02 | | OCTEXIT1 | |
| 03 The TCACCTR field contains a composite response of all errors. | | OCPEXIT1 | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

Common Exit

Diagram - 5.7.7-01

PAGE 1 OF 2


Input
Processing
Output
CSA
CSAICIND Internal initiate indicator
CSATADJT Time adjustment total
CSAICEBA Address of ICE chain
CSATCTBA TCT address
DFHTAJP:
01 Inhibit Runaway Task Control for the life of the task
02 Request check reset service
ICP
RESET 1.4.11
03 Perform processing for internally initiated tasks
A. Calculate the time adjustment
B. Adjust the expiration time
C. Adjust the polling delay expiration time
D. Adjust the next dispatch time of day
E. Refresh all CSA current time-of-day fields
A
A
TCA
TCASVMID Service module ID
ICE
ICEXTOD Expiration time of day
TCTTEs
TCTTEBC Bypass control expiration time of day
TCTLEs
TCTLEBC Bypass control expiration time of day
CSA
CSATCNDT Terminal Control next dispatch time of day
CSATODP Packed time of day
CSATODB Binary time of day
CSATODTU Timer units time of day

Time Adustment Program

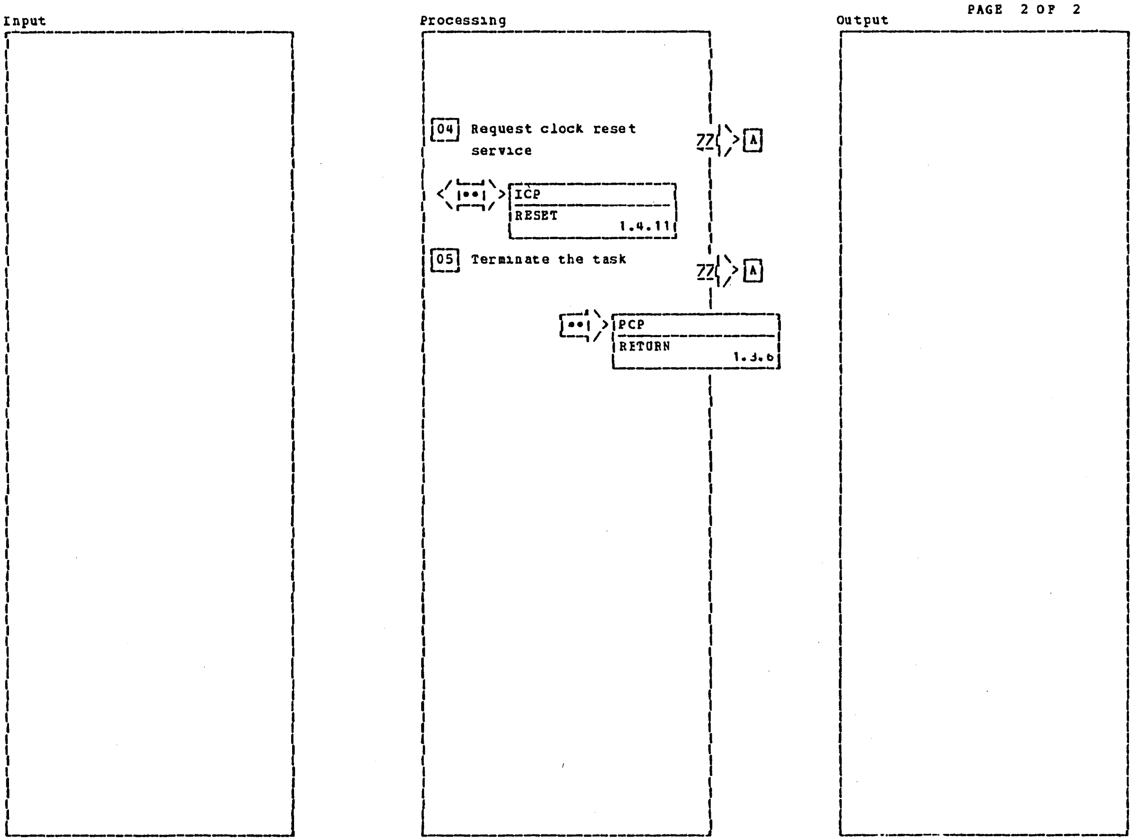
Diagram - 5.8.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | DFHTODNA | |
| 02 The clock reset services of ICP are requested to obtain the current 24-hour clock timer units value and adjustment amount. | | | |
| 03 The task that executes this program can be initiated internally if midnight occurs or if invoked through master terminal. | | | |
| If internal CICS tests detect a clock roll-back condition, an indicator in the CSA is set to initiate the task. | | | |
| A. If the difference between the time of day at the time the reset condition was detected and the prior time of day is greater than 23-1/2 hours, the program assumes that midnight had occurred and its sets the adjustment time to 24 hours. | | | |
| If the difference is less than 23-1/2 hours, that difference will be used as the adjustment amount. | | | |

Time Adustment Program

Diagram - 5.8.1-01

PAGE 2 OF 2



04 Request clock reset service

ICP
RESET
1.4.11

05 Terminate the task

PCP
RETURN
1.3.6

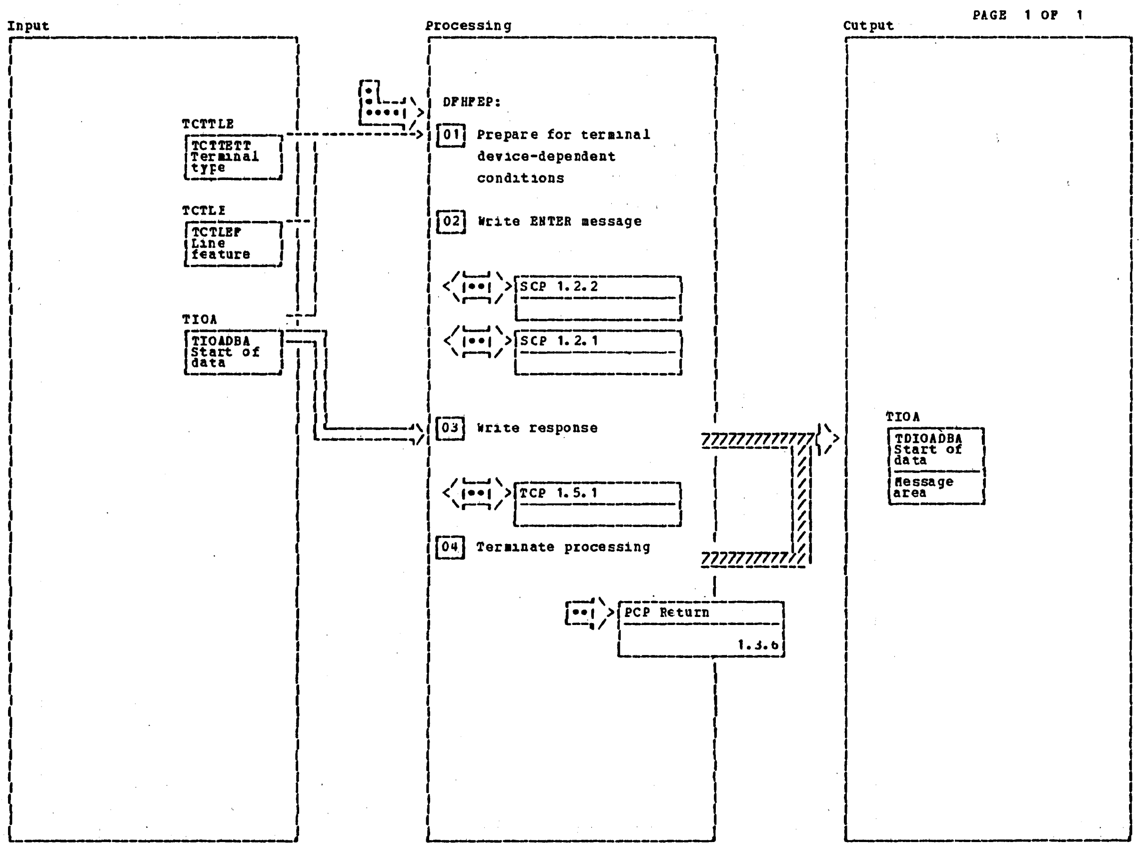
Time Adustment Program

Diagram - 5.8.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 The clock reset services of ICP are requested to establish the new base clock and time unit values and to refresh the current date and time-of-day values in the CSA. | | | | | | | |
| 05 A Program Control RETURN is issued. | | | | | | | |

Time Adustment Program

Diagram - 5.8.1-02



Terminal Test

Diagram - 5.9.1-01

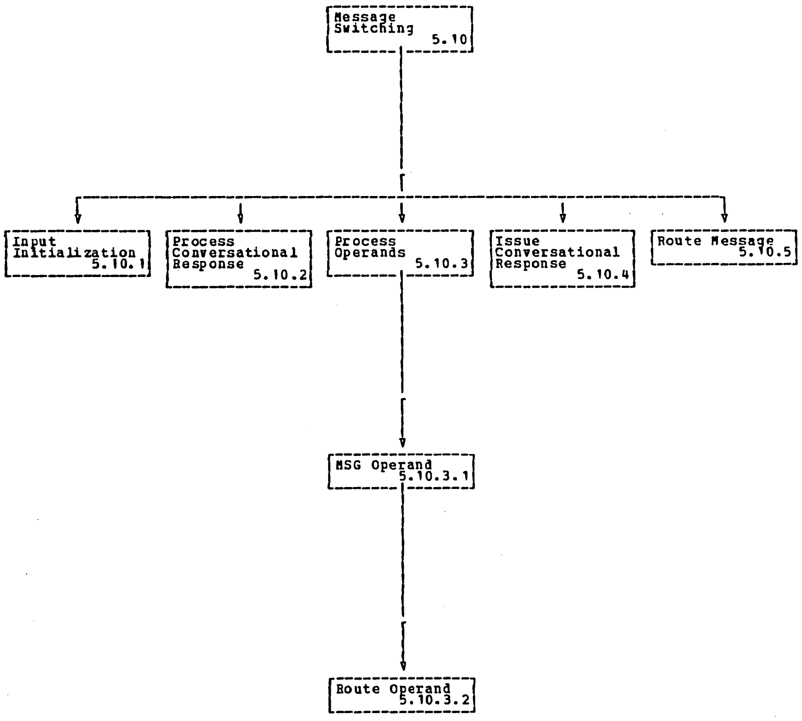
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Handle special conditions for: | DFHFEP | | |
| TWX | | TWXMESS | |
| 2740 Model 2 | | FELS | |
| Bisync device | | FEBTE | |
| Hard-copy device | | INCIDLES | |
| VTAM device | | FE360MSG | |
| 02 A DFHSC TYPE=FREEMAIN, RELEASE=ALL macro instruction is issued.<br>Storage for the message is obtained via a Storage Control GETMAIN. To write the message, a Terminal Control WRITE/READ/WAIT is issued. | | | |
| 03 If print was requested, the character set is printed via a Terminal Control WRITE/READ/WAIT. If end was entered, the DFH3301 completion message is issued. Otherwise, the input is echoed. | DFHFEP | SENDCHAR | |

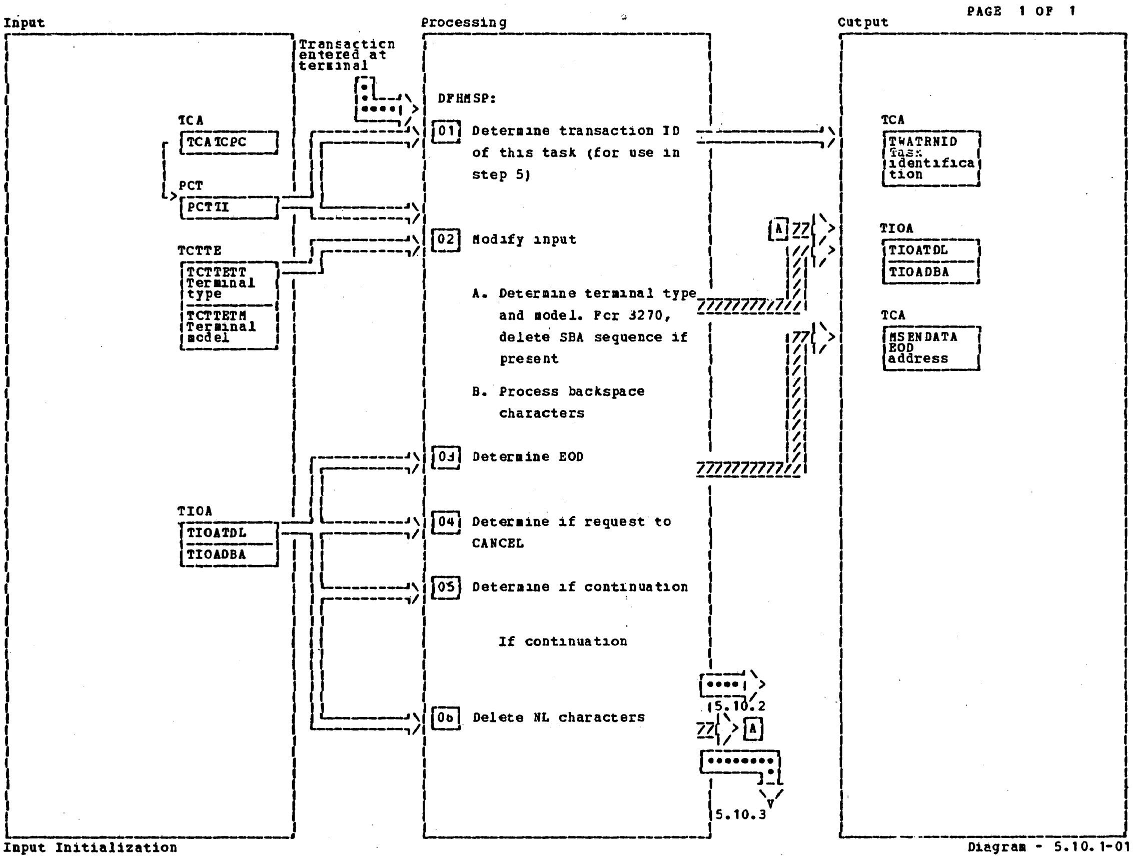
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 04 Completion message DFH3301 is issued. | DFHFEP | RETURN | |

Terminal Test

Diagram - 5.9.1-01

PAGE 1 OF 1


Message Switching 5.10
Input Initialization 5.10.1
Process Conversational Response 5.10.2
Process Operands 5.10.3
Issue Conversational Response 5.10.4
Route Message 5.10.5
MSG Operand 5.10.3.1
Route Operand 5.10.3.2


PAGE 1 OF 1
Input
Processing
Cutput
Transacticn entered at terminal
TCA
TCAICPC
PCT
PCTII
TCTTE
TCTTETT Terminal type
TCTTETM Terminal model
TIOA
TIOATDL
TIOADBA
DFHMSP:
01 Determine transaction ID of this task (for use in step 5)
02 Modify input
A. Determine terminal type and model. For 3270, delete SBA sequence if present
B. Process backspace characters
03 Determine EOD
04 Determine if request to CANCEL
05 Determine if continuation
If continuation
06 Delete NL characters
5.10.2
A
5.10.3
TCA
TWATRNID Task identification
TIOA
TIOATDL
TIOADBA
TCA
MSENDATA EOD address
A

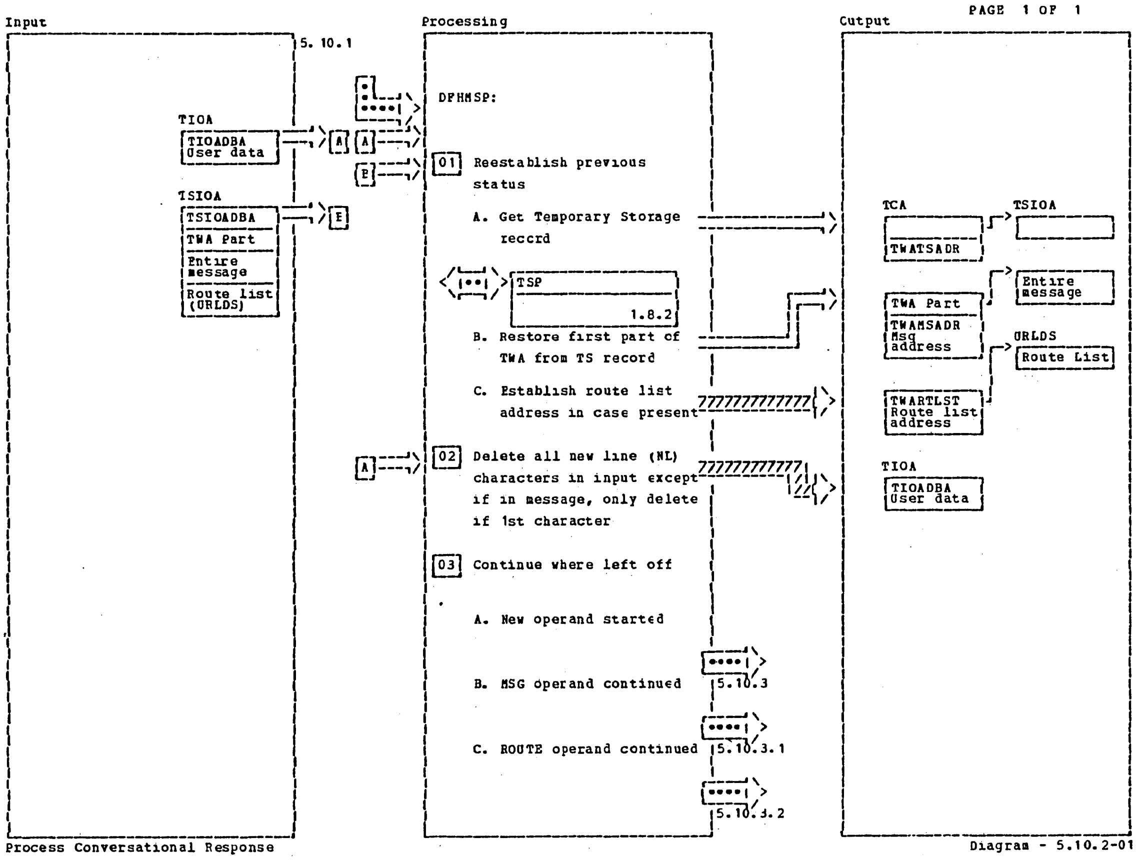
Input Initialization

Diagram - 5.10.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 02 | | | |
| A. Delete 3270 SBA sequence if first 3 characters at TIOADBA. | | MSMDL1BY | |
| B. Delete backspace characters and same number of preceding data characters. | | MSSCANBS | |
| 04 If CANCEL entered at EOD, or on 3275/3277, 2 successive CLEAR operations (1 CLEAR operation allows continuation). If CANCEL, issue termination response and return to CICS. | | MSNONL<br>MSMDLIBY<br>MSRSTRT | |
| 05 The first 4 characters of the TIOA are compared with the task identification stored in TWATRNIC. If not equal, this is a continuation (go to MSCONTIN). If equal, treat as a new start. If transaction identification is followed by 'C', delete first 5 characters of input and go to MSCONTIN. | | MSNOCNCL | 5.10.4<br>STEP 3 |
| 06 Delete all new line (NL) characters in input except if, in message, only delete if 1st character. | | MSFINDNL | |

Input Initialization

Diagram - 5.10.1-01



Process Conversational Response

Diagram - 5.10.2-01

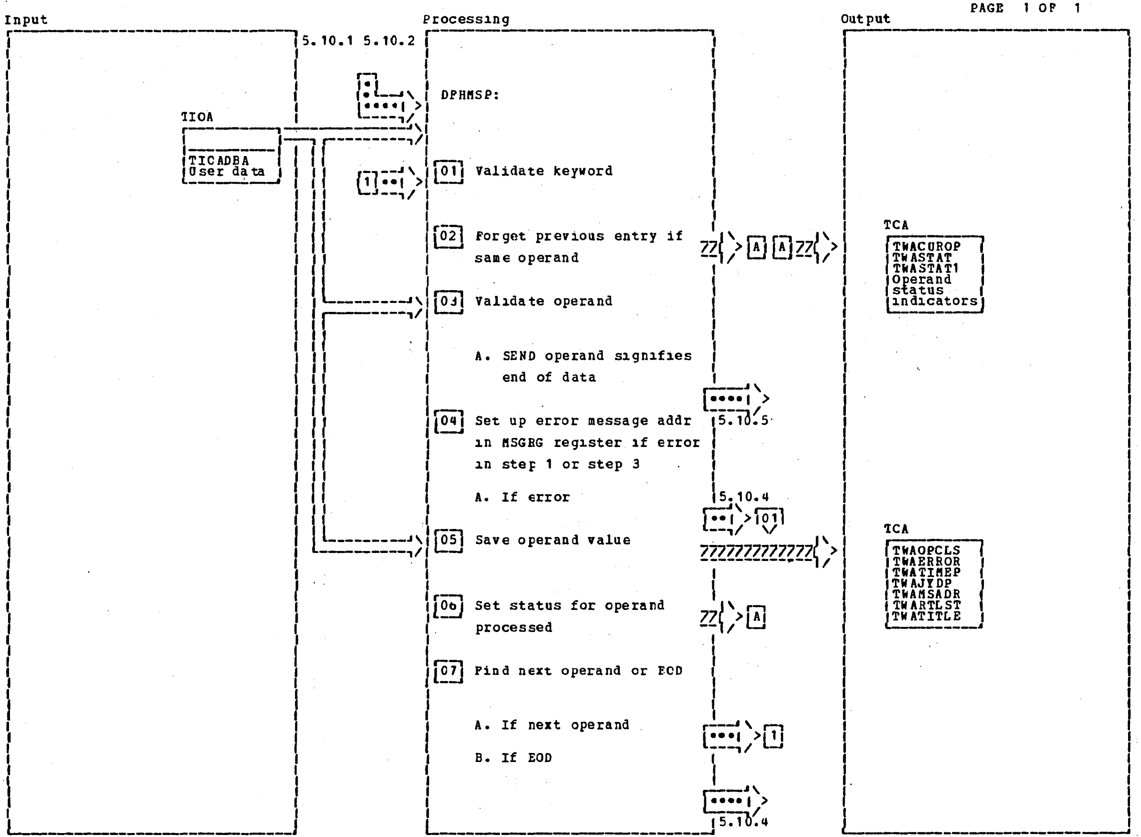
| Notes | Rcutine | Label | Ref |
|---|---|---|---|
| 01 | | | |
| A. TS record name is /1 byte - X'FC' / 4 bytes - Termid / 3 bytes - C'MSG'/ address of TS record is stored in TWATSADR. | | MSCONTIN | |
| B. Restore 1st part of TWA from TSICA. Set up address of message (if present in TSIOA - MSG indicator on in TWASTAT) in TWAMSADR. | | MSCLRBYP | |
| C. Set up TWARTLST to where route list would start if present (fullword-aligned after end of message - message length in TWAMSLNG) | | MSCLRBYP | |
| D. Check input and continue with one of the following conditions<br>- New operand started<br>- MSG or ROUTE operand continued<br>- Errcr condition corrected | | MSCLRBYP | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| | | | |

Process Conversaticnal Response

Diagram - 5.10.2-01

PAGE 1 OF 1



DFHMSP:

01 Validate keyword

02 Forget previous entry if same operand

03 Validate operand

A. SEND operand signifies end of data

04 Set up error message addr in MSGRG register if error in step 1 or step 3

A. If error

05 Save operand value

06 Set status for operand processed

07 Find next operand or EOD

A. If next operand

B. If EOD

Process Operands

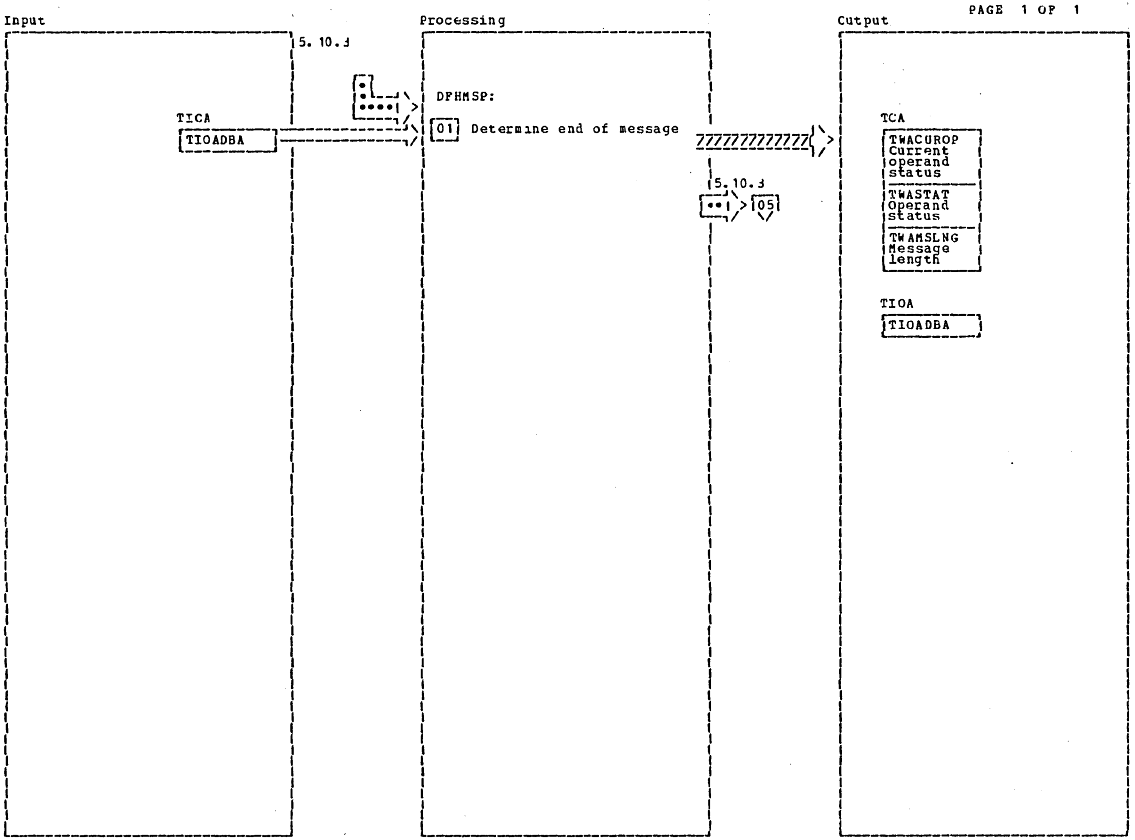
Diagram - 5.10.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Scan for first character of keyword. | | MSNTRY | |
| 02 Turn off corresponding indicator in TWASTAT or TWASTAT1.<br>Turn on corresponding indicator in TWACUROP to show operand currently processing. (This is checked on continuation to determine if MSG or ROUTE operands are being continued.) | | Start of each operand routine | |
| 03 Such as OPCLASS values in range 1 to 24, ERRTERM 1 to 4 characters, etc. Some validating can only be done after the SEND operand is received - i.e., combination of TIME and DATE too far in future.<br>NOTE: For M, R, S operand processing see the following charts M 5.10.3.1, R 5.10.3.2, S 5.10.3.2. The S operand gives control to Route Message 5.10.5. | | Each operand routine | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 04 | | MSOPERR and in each operand routine | |
| 05 Operand values are stored in first part of TWA except for MSG and ROUTE where address of message or route list is stored in TWAMSADR and TWARTLST. | | Each operand routine | |
| 06 Turn on corresponding indicator in TWASTAT or TWASTAT1. | | Each operand routine | |

Process Operands

Diagram - 5.10.3-01

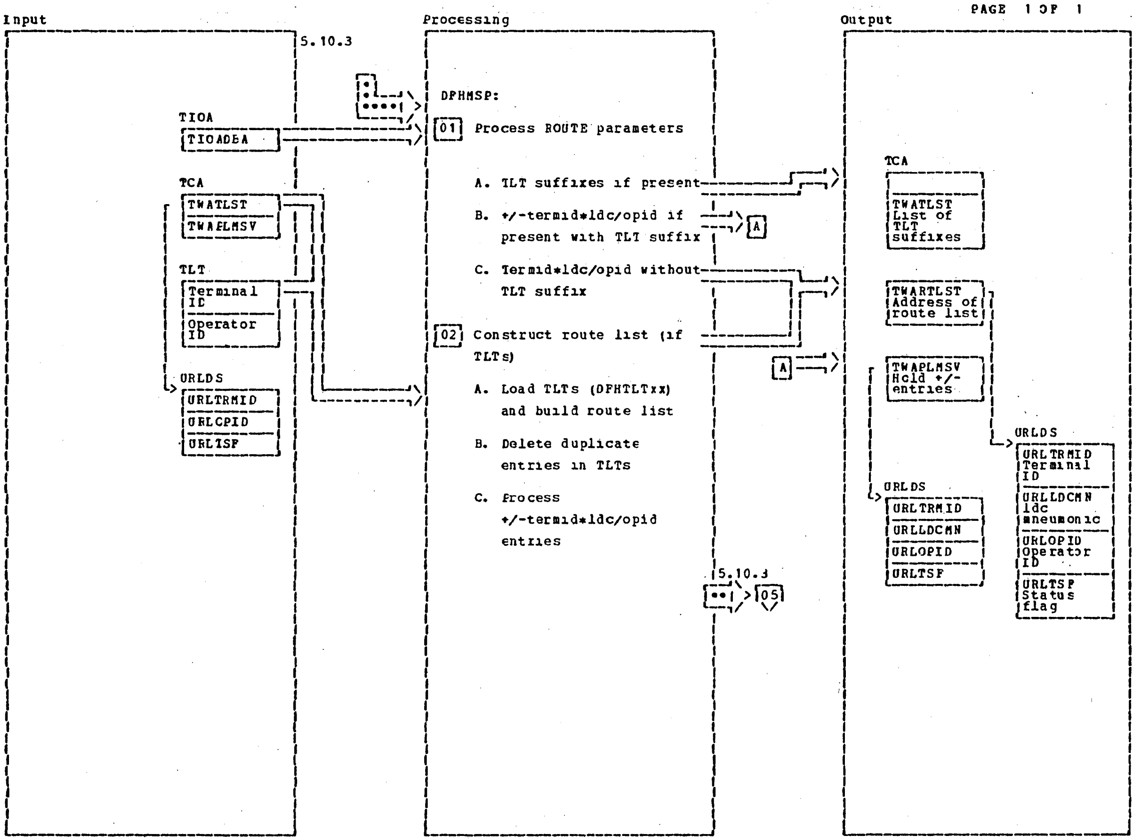


MSG Operand Diagram - 5.10.3.1-01

| Notes | Rcutine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Scan for ending gucte and find<br>A. EOD - message is continued on next input.<br>B. Quote followed by space - message is ended and MSG indicator turned on in TWASTAT.<br>C. Quote followed by comma - same as B.<br>D. Quote followed by quote - the first quote is part of message and the second quote is deleted (TIOA data shifted to start of TIOA by 1 byte).<br>E. Quote fcllowed by EOD - the QUOTE indicatcr is turned on in TWACUROP. If the next input starts with quote, it is processed like D; otherwise, like B.<br>F. Quote followed by any other character - processed like B but an error message is issued for an invalid operand |  | MSMSG4 |  | separator. |  |  |  |

MSG Operand Diagram - 5.10.3.1-01

PAGE 1 OF 1


Input
5.10.3
TIOA
TIOADBA
TCA
TWATLST
TWAPLMSV
TLT
Terminal ID
Operator ID
URLDS
URLTRMID
URLOPID
URLTSP
Processing
DFHMSP:
01 Process ROUTE parameters
A. TLT suffixes if present
B. +/-termid*ldc/opid if present with TLT suffix
A
C. Termid*ldc/opid without TLT suffix
02 Construct route list (if TLTs)
A. Load TLTs (DFHTLTxx) and build route list
B. Delete duplicate entries in TLTs
C. Process +/-termid*ldc/opid entries
A
5.10.3
05
Output
TCA
TWATLST List of TLT suffixes
TWARTLST Address of route list
TWAPLMSV Hold +/- entries
URLDS
URLTRMID
URLLDCMN
URLOPID
URLTSP
URLDS
URLTRMID Terminal ID
URLLDCMN ldc mneumonic
URLOPID Operator ID
URLTSP Status flag

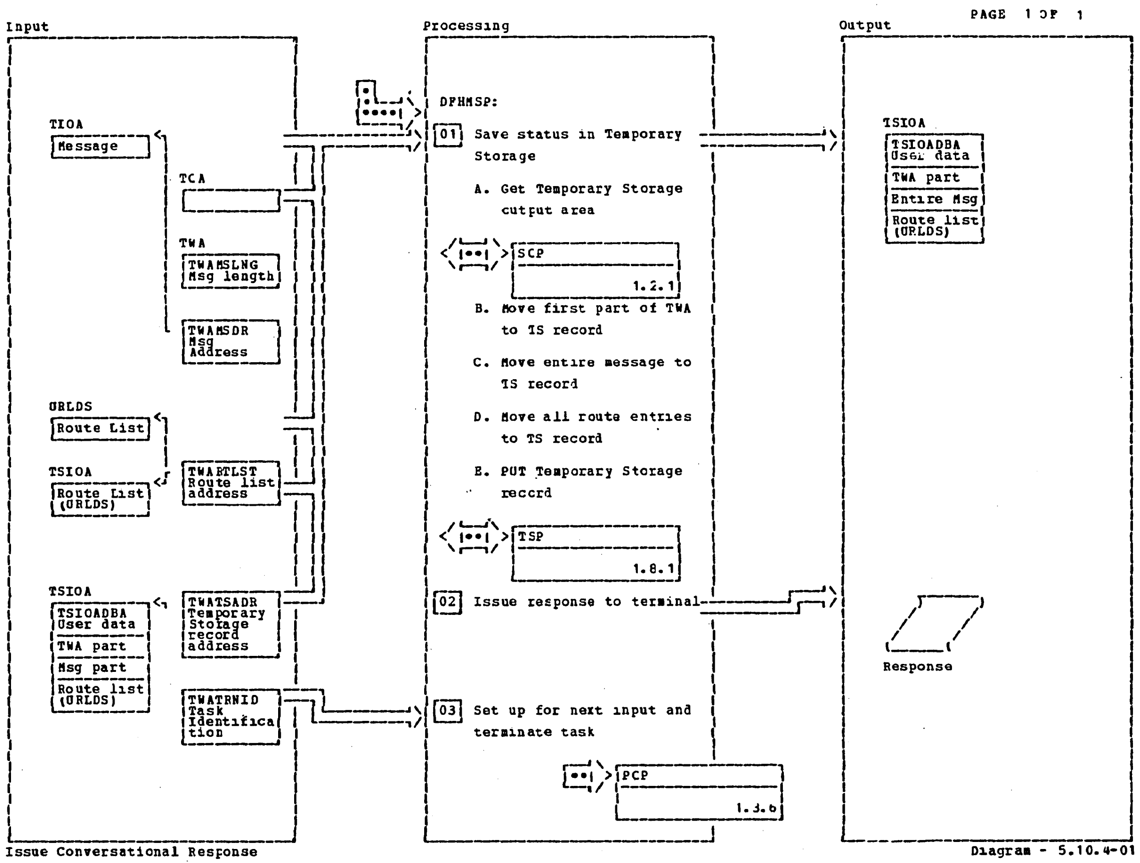
ROUTE Operand

Diagram - 5.10.3.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | | |
| A. TLT suffixes are saved in TWATLST (max. number of entries specified by &MAXSUFX or MAXSUFX). | | MSPERIOD | |
| B. +/-termid*ldc/opid entries are saved in the form of a user- supplied route list where the address is stored in TWAPLMSV. The + or - is saved in URLTSP. | | MSLSTDON | |
| C. If no TLT suffixes are present, the entries are used to build a user-supplied route list whose address is stored in TWARTLST. | | MSNDTID | |
| 02 | | | |
| A. Load TLTs (DFHTLTxx) using suffixes stored in TWATLST and build a user-supplied route list. | | MSCHKTLT | |
| B. If more than one TLT, do not include Termid*ldc/opid entries if defined previously in TLT already processed. | | MSCHKDUP | |
| C. Add any + entries if not previously defined and delete all - entries if previously defined. Each +/- entry (saved at step 1B) is processed against the entire route list built from all the TLT's (at steps 2A an 2B). | | MSTLLDON | |

ROUTE Operand

Diagram - 5.10.3.2-01



DFHMSP:

01 Save status in Temporary Storage

A. Get Temporary Storage output area

B. Move first part of TWA to TS record

C. Move entire message to TS record

D. Move all route entries to TS record

E. PUT Temporary Storage reccrd

02 Issue response to terminal

03 Set up for next input and terminate task

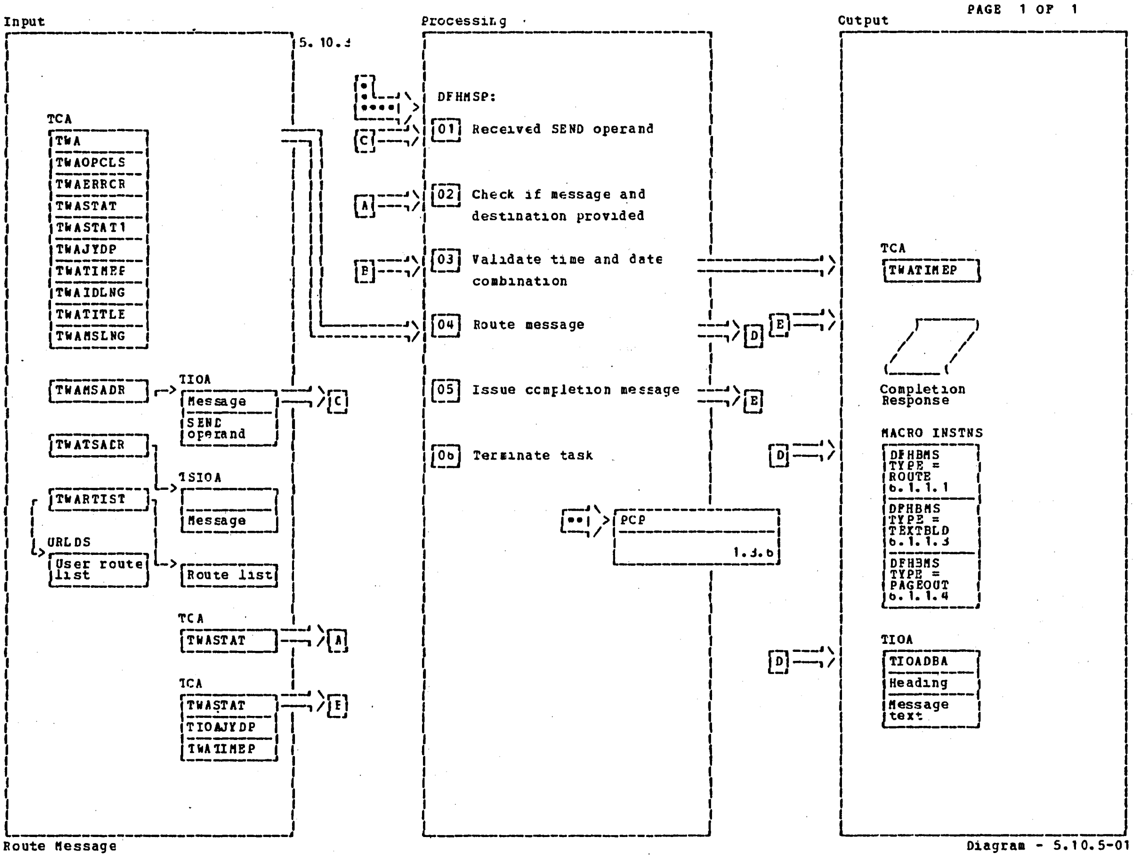
Issue Conversational Response

Diagram - 5.10.4-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | | |
| A. Size of output area is sum of the following pieces if still active:<br>- TWA first part<br>- Previcus msg part (old TS record)<br>- Current msg part (from TIOA)<br>- Previcus rcute list (old TS record)<br>- Current route list (built from current input or chained off cld TS record route list) | | MSINPTBY | |
| B. All cf the TWA fields thru TWAMSLNG | | MSNPREV4 | |
| C. Message can ccnsist of:<br>- Piece in old TS record (if msg indicator on in TWASTAT field in TS reccrd)<br>- Part in current TIOA | | MSMSGMV | |
| D. Route list is pointed to by TWARTLST and may include entries in TS record if processed in previous input. Route list is started on full-word boundary. | | MSRTEMOV | |
| 02 Response msg address is in MSGRG register upon entry to MSCNVRS (ccnversation routine). MSINARG register pcints to variable data to te appended to response. | | MSCNVRS<br>MSSBABYP | |
| 03 If continuation is possible, effectively issue a DFHPC TYPE=RETURN,TRANSID= to cause same task to start automatically on next input, otherwise issue a standard DFHPC TYPE=RETURN. | | MSGSENT | 5.10.1<br>STEP 1<br>STEP 5 |
| Response to terminal is via 3275/3277 DFHTC TYPE=WRITE to position on bottom line of screen, all other terminals DFHBMS TYPE=(EDIT,OUT). | | | |

Issue Conversational Response

Diagram - 5.10.4-01



Route Message

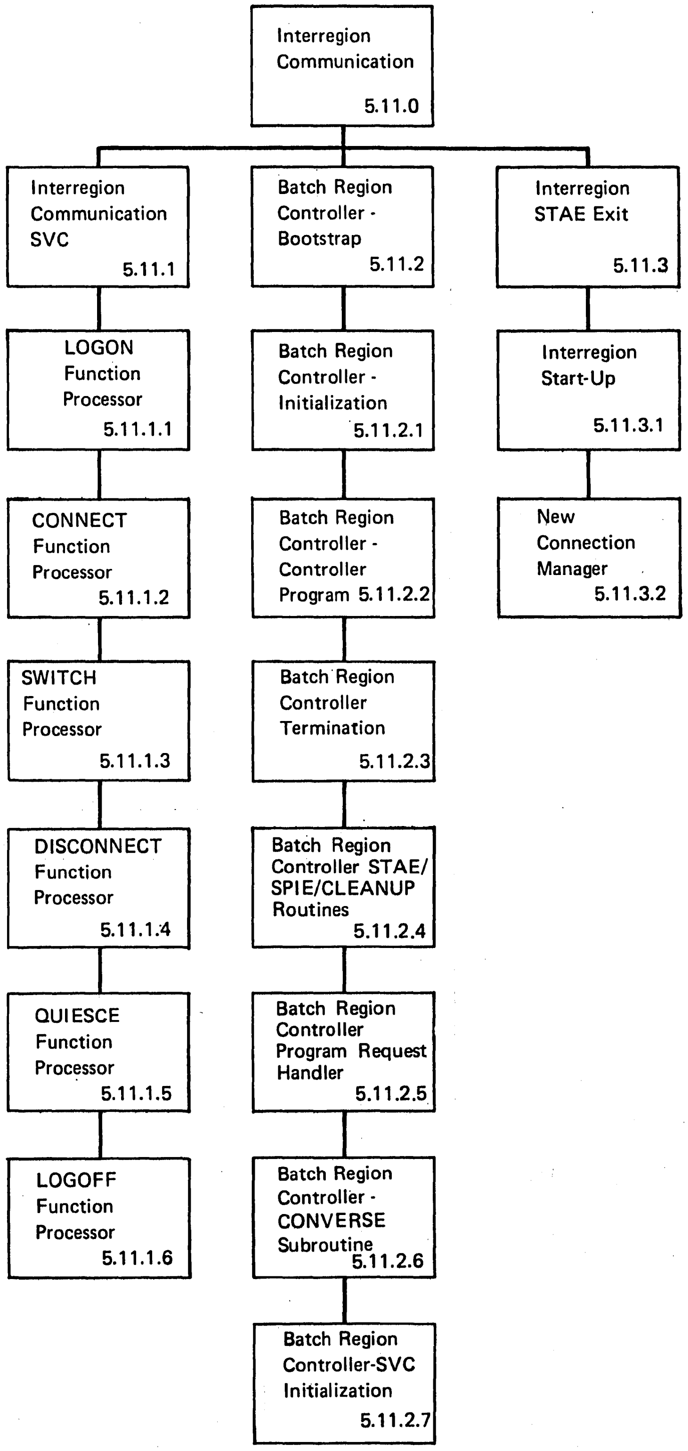
Diagram - 5.10.5-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | MSBMSRT | |
| 02 MSG and ROUTE or OPCLASS operands are required as a minimum, else error message. | | MSNOMSG<br>MSNORTE | |
| 03 Time and date combination must be less than 100 hours from start of current day, else error message. | | MSTIMDON | |
| 04 TWASTAT and TWASTAT1 indicate the operands in effect. The active TWA fields set up during operand processing are used to set up the TCA and issue one of several forms of the DFHBMS TYPE=ROUTE macro. | | MSBMSRT | 6.1.1.1 |
| A TIOA-like area consisting of a time/date and originator's identification heading (if HEADING is specified) and the message is created and a DFHBMS TYPE=TEXTBLD macro is issued. | | | 6.1.1.3 |
| If the message has a piece in the current input and was continued from the previous input, two TEXTBLDs are issued one for each part. | | | |
| A DFHBMS TYPE=PAGEOUT macro is issued to indicate that the message is completed and should be routed. | | | 6.1.1.4 |

Route Message

Diagram - 5.10.5-01

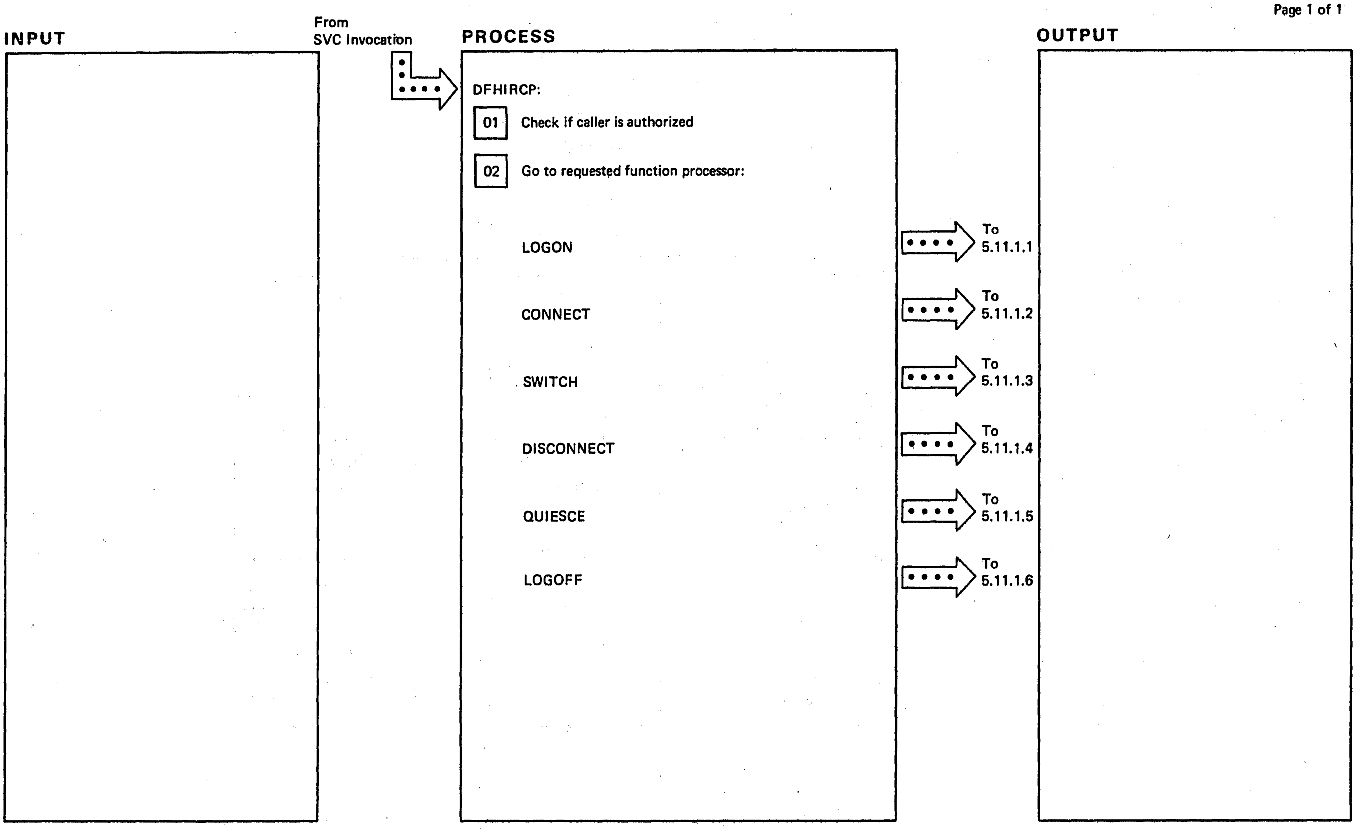
Page 1 of 1


Interregion Communication 5.11.0
Interregion Communication SVC 5.11.1
Batch Region Controller - Bootstrap 5.11.2
Interregion STAE Exit 5.11.3
LOGON Function Processor 5.11.1.1
Batch Region Controller - Initialization 5.11.2.1
Interregion Start-Up 5.11.3.1
CONNECT Function Processor 5.11.1.2
Batch Region Controller - Controller Program 5.11.2.2
New Connection Manager 5.11.3.2
SWITCH Function Processor 5.11.1.3
Batch Region Controller Termination 5.11.2.3
DISCONNECT Function Processor 5.11.1.4
Batch Region Controller STAE/ SPIE/CLEANUP Routines 5.11.2.4
QUIESCE Function Processor 5.11.1.5
Batch Region Controller Program Request Handler 5.11.2.5
LOGOFF Function Processor 5.11.1.6
Batch Region Controller - CONVERSE Subroutine 5.11.2.6
Batch Region Controller-SVC Initialization 5.11.2.7

Interregion Communication — Visual Index

Diagram — 5.11-01

Page 1 of 1



Interregion Communication SVC

Diagram — 5.11.1-01

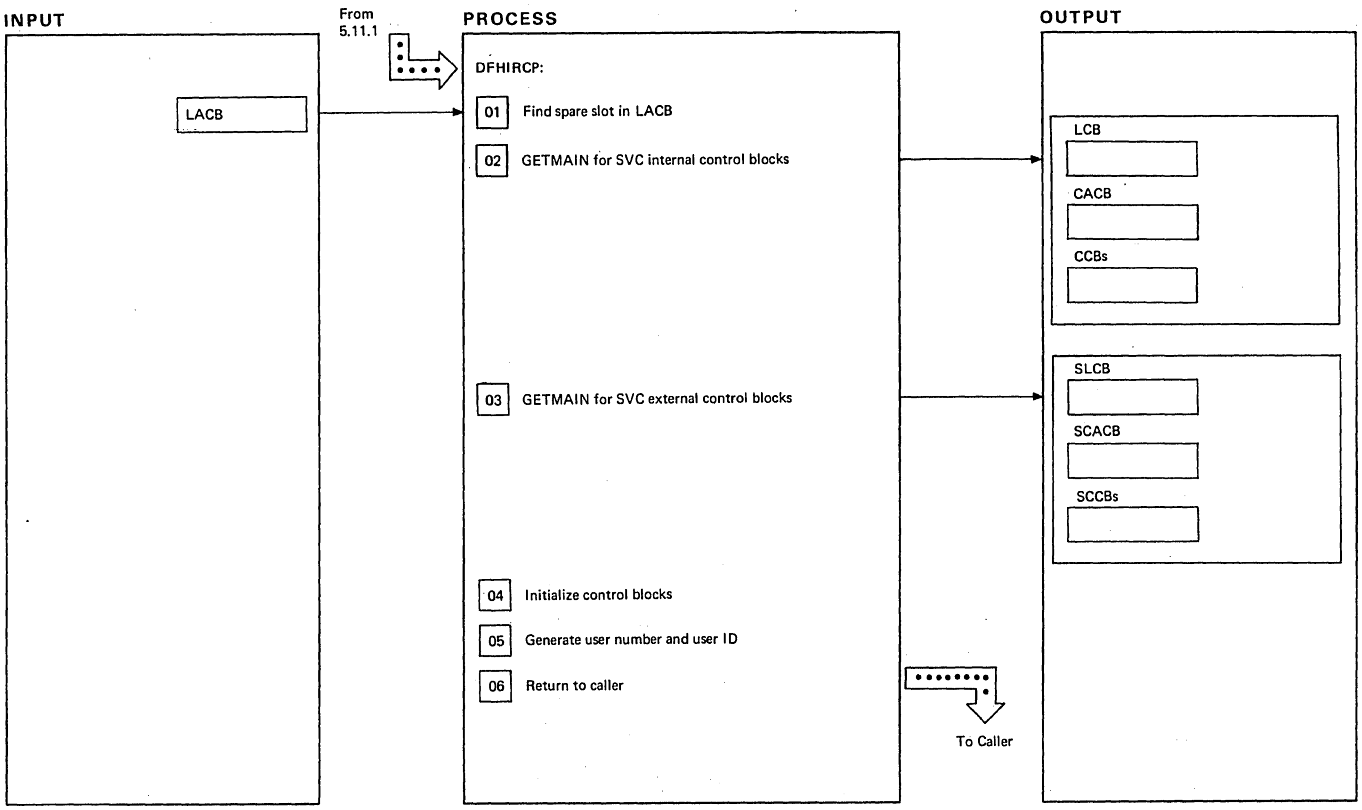
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Uses DFHAUTH macro to check that caller had authority to use interregion SVC | DFHIRCP | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Interregion Communication SVC

Diagram — 5.11.1-01

Page 1 of 1



DFHIRCP:

01 Find spare slot in LACB

02 GETMAIN for SVC internal control blocks

03 GETMAIN for SVC external control blocks

04 Initialize control blocks

05 Generate user number and user ID

06 Return to caller

LOGON Function Processor

Diagram — 5.11.1.1-01

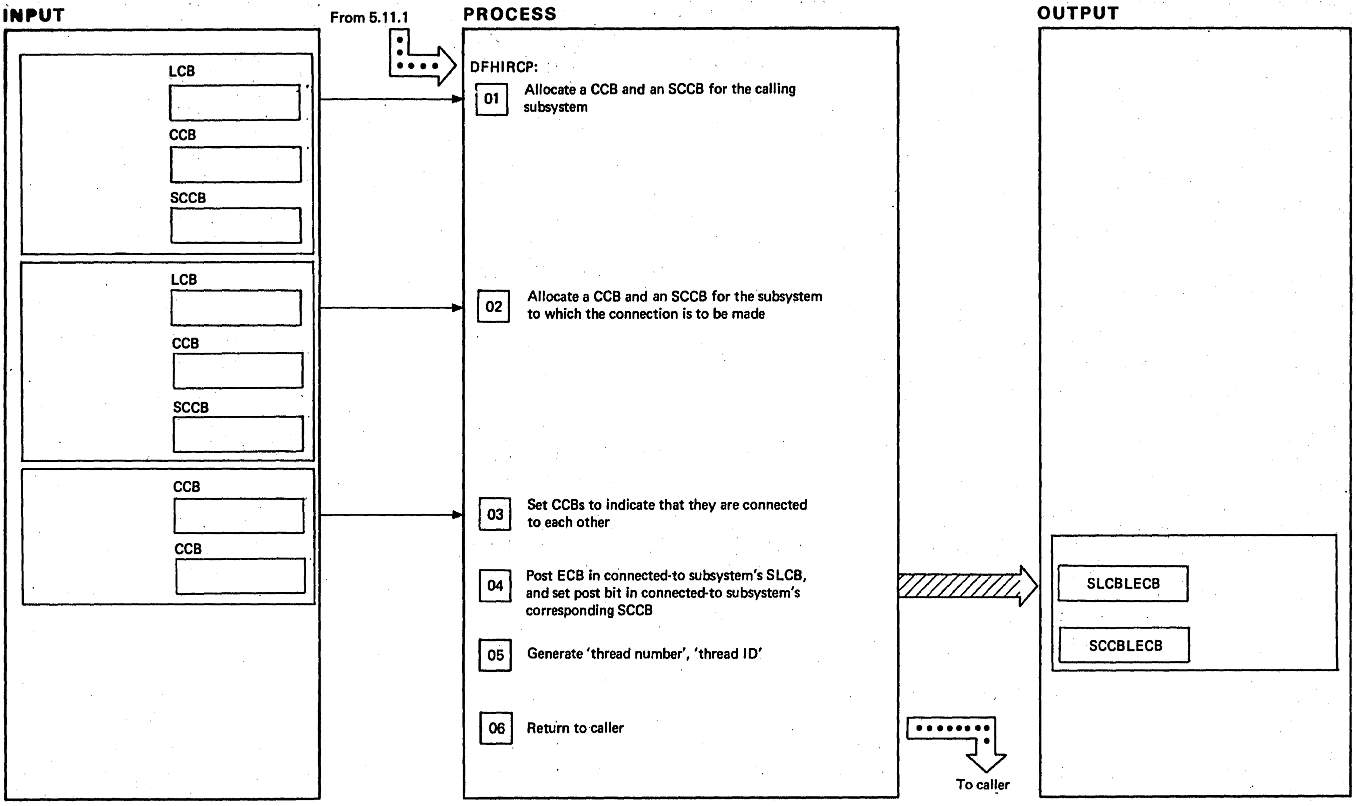
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 LACB has one active field per logged-on user | | IRCP2000 | |
| 02 Blocks are in key 0 fetch protected storage | | | |
| 03 Blocks are in user key storage, and are accessed by the user subsystem | | | |
| 05 User number and user ID are passed by the user on subsequent calls for identification and security purposes | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

LOGON Function Processor

Diagram — 5.11.1.1-01

Page 1 of 1



01 Allocate a CCB and an SCCB for the calling subsystem

02 Allocate a CCB and an SCCB for the subsystem to which the connection is to be made

03 Set CCBs to indicate that they are connected to each other

04 Post ECB in connected-to subsystem's SLCB, and set post bit in connected-to subsystem's corresponding SCCB

05 Generate 'thread number', 'thread ID'

06 Return to caller

CONNECT Function Processor

Diagram — 5.11.1.2-01

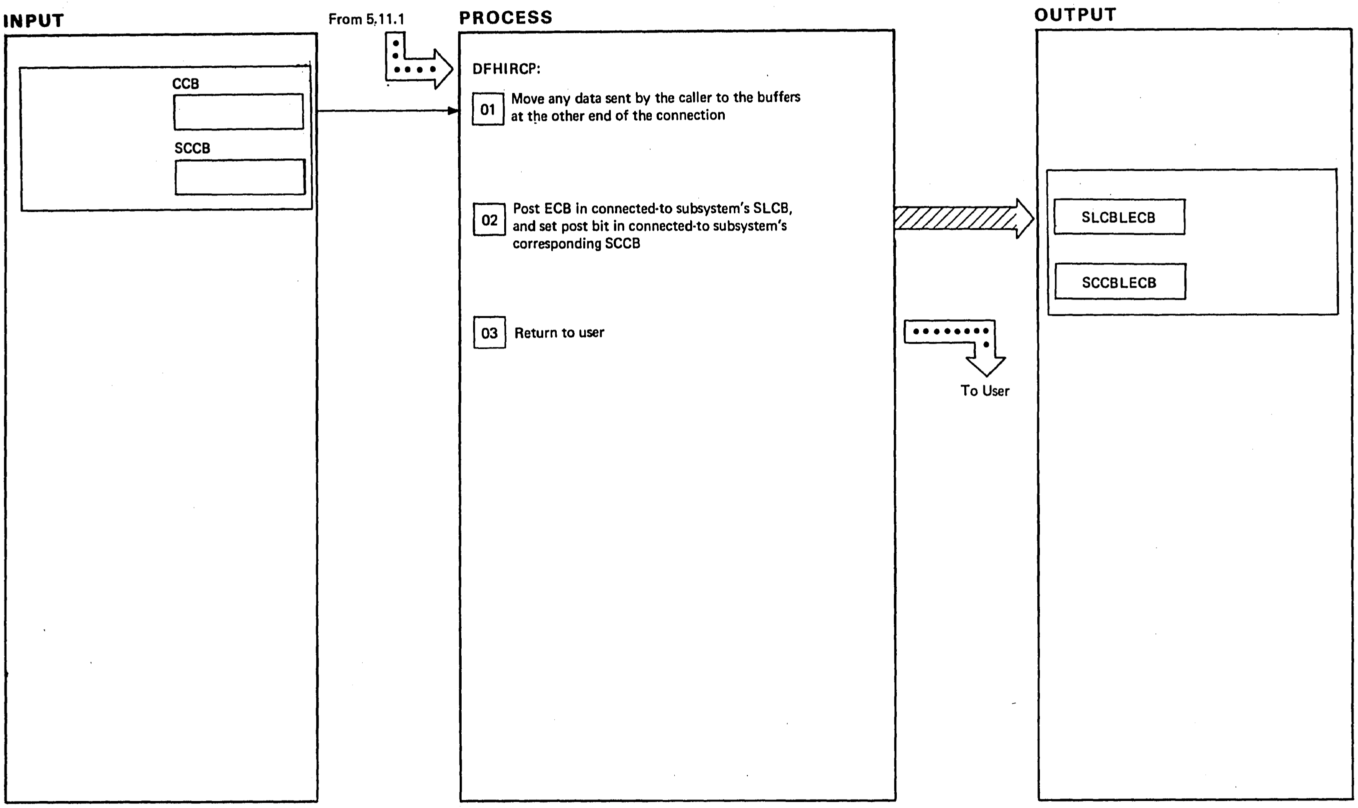
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 There is one CCB/SCCB pair for each connection in which a subsystem can participate at any one time. | | IRCP4000 | |
| 04 The ECB in the SLCB is posted using an operating system POST. The SCCB ECB is posted by setting the X'40' bit. | | | |
| 05 The 'thread number' and 'thread ID' (identifying the connection to be used) are passed for thread identification and security purposes by the user on subsequent calls which are specific to this connection | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

CONNECT Function Processor

Diagram — 5.11.1.2-01

Page 1 of 1



DFHIRCP:

01 Move any data sent by the caller to the buffers at the other end of the connection

02 Post ECB in connected-to subsystem's SLCB, and set post bit in connected-to subsystem's corresponding SCCB

03 Return to user

SWITCH Function Processor

Diagram — 5.11.1.3-01

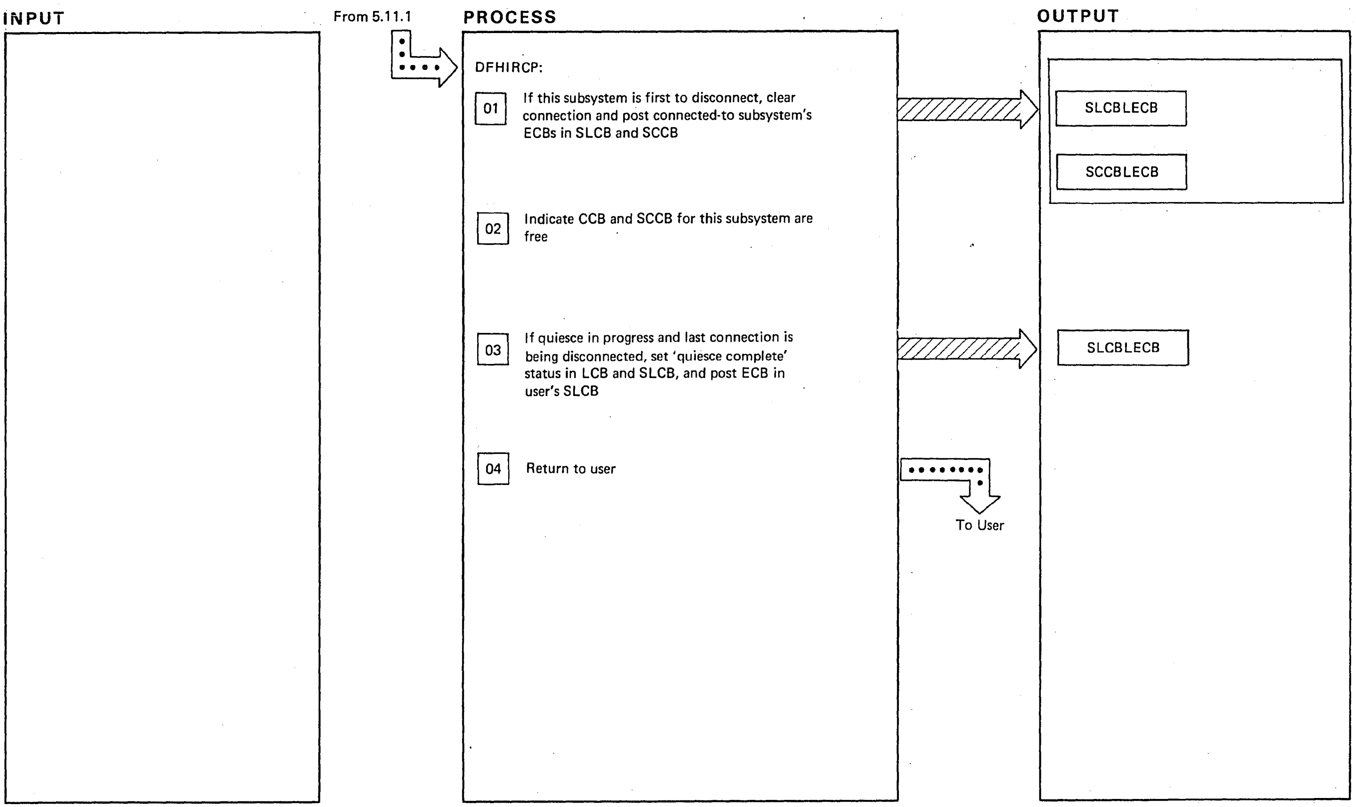
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

SWITCH Function Processor

Diagram — 5.11.1.3-01

Page 1 of 1



INPUT

From 5.11.1

PROCESS

DFHIRCP:

01 If this subsystem is first to disconnect, clear connection and post connected-to subsystem's ECBs in SLCB and SCCB

02 Indicate CCB and SCCB for this subsystem are free

03 If quiesce in progress and last connection is being disconnected, set 'quiesce complete' status in LCB and SLCB, and post ECB in user's SLCB

04 Return to user

To User

OUTPUT

SLCBLECB

SCCBLECB

SLCBLECB

DISCONNECT Function Processor

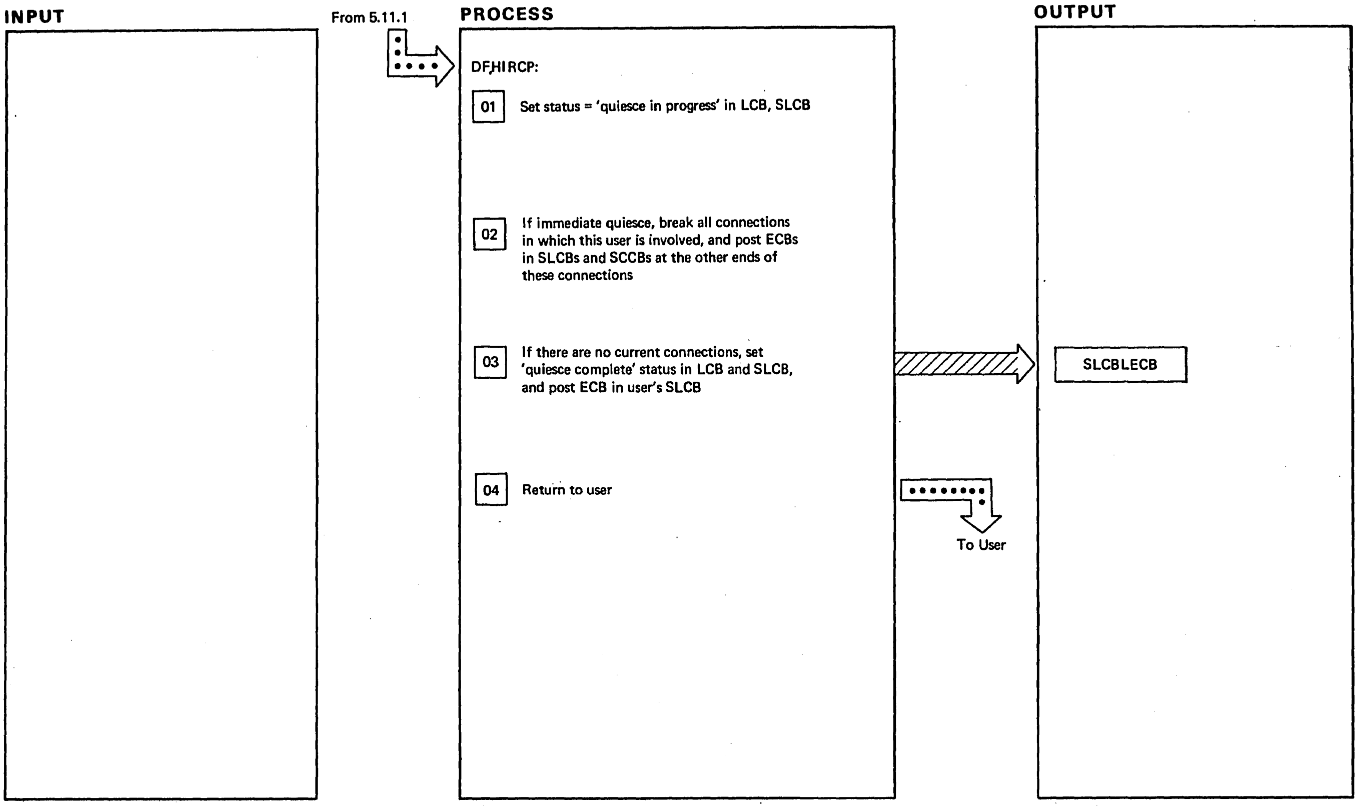
Diagram — 5.11.1.4-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Both sides of connection must issue a DISCONNECT request in order that CCB/SCCB pair on each side is freed | | IRCP5000 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

DISCONNECT Function Processor

Diagram — 5.11.1.4-01



DFHIRCP:

01 Set status = 'quiesce in progress' in LCB, SLCB

02 If immediate quiesce, break all connections in which this user is involved, and post ECBs in SLCBs and SCCBs at the other ends of these connections

03 If there are no current connections, set 'quiesce complete' status in LCB and SLCB, and post ECB in user's SLCB

04 Return to user

QUIESCE Function Processor

Diagram — 5.11.1.5-01

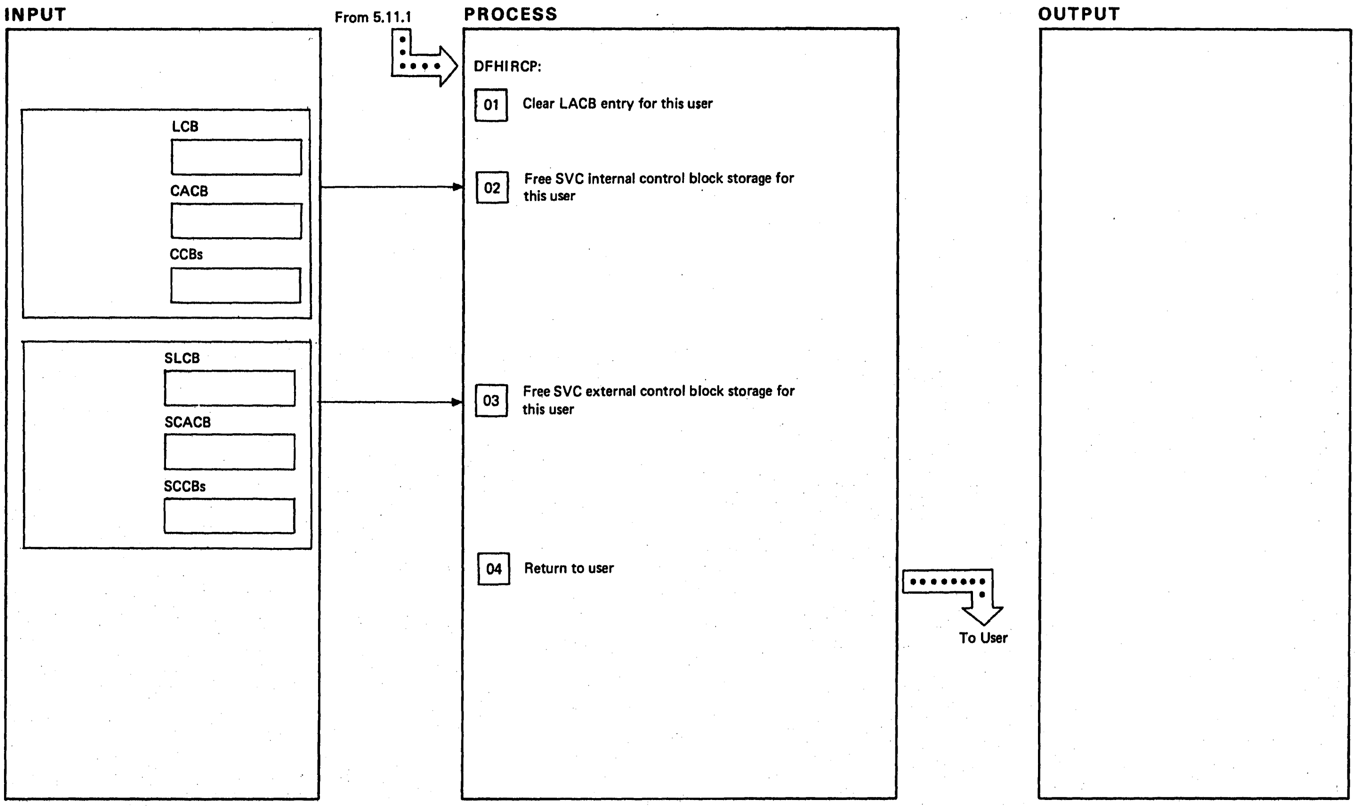
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 Posting indicates to connected-to subsystems that connections are broken | | IRCP7000 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

QUIESCE Function Processor

Diagram — 5.11.1.5-01

Page 1 of 1


INPUT
From 5.11.1
PROCESS
OUTPUT
LCB
CACB
CCBs
SLCB
SCACB
SCCBs
DFHIRCP:
01 Clear LACB entry for this user
02 Free SVC internal control block storage for this user
03 Free SVC external control block storage for this user
04 Return to user
To User

LOGOFF Function Processor

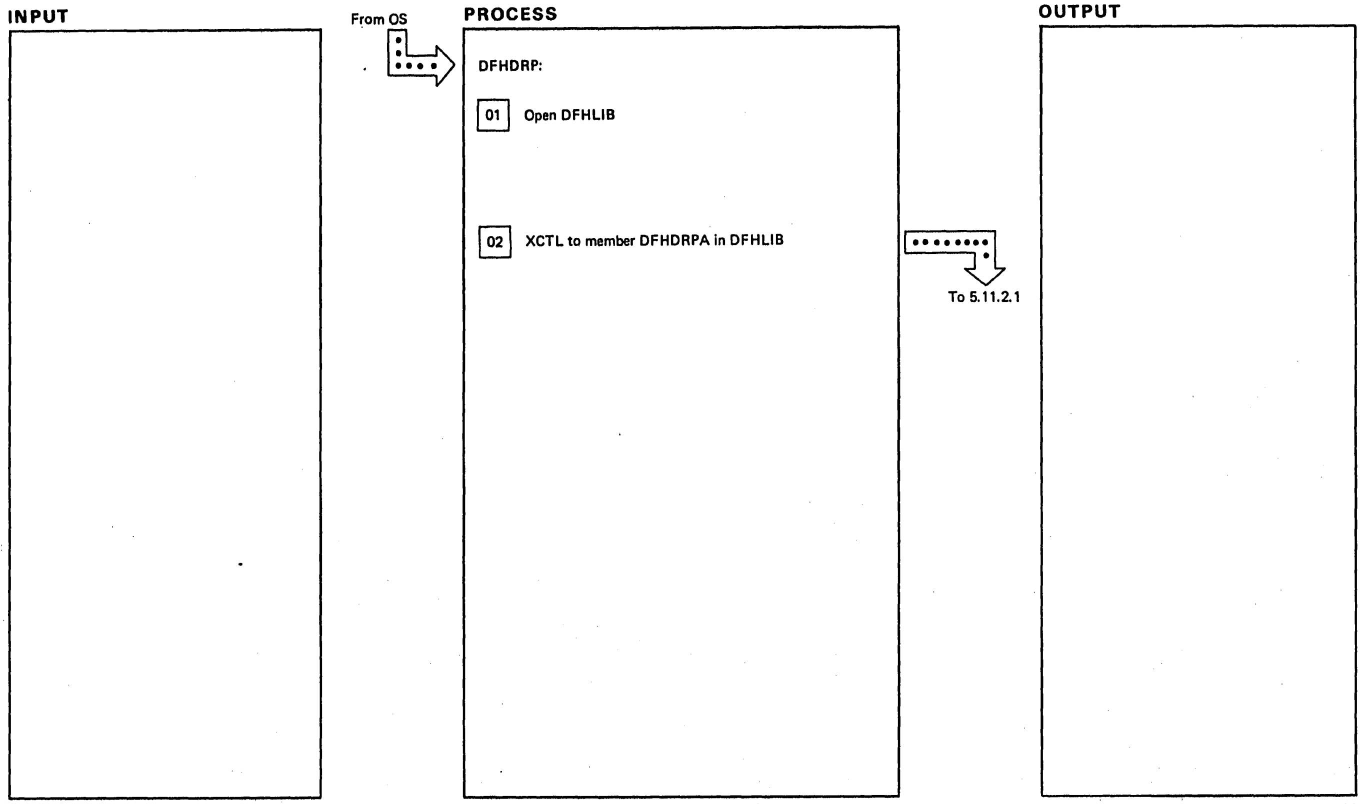
Diagram — 5.11.1.6-01

| NOTES | ROUTINE | LABEL | REFERENCE | NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|---|---|---|

LOGOFF Function Processor

Diagram — 5.11.1.6-01

Page 1 of 1



DFHDRP:

01 Open DFHLIB

02 XCTL to member DFHDRPA in DFHLIB

Batch Region Controller - Bootstrap

Diagram — 5.11.2-01

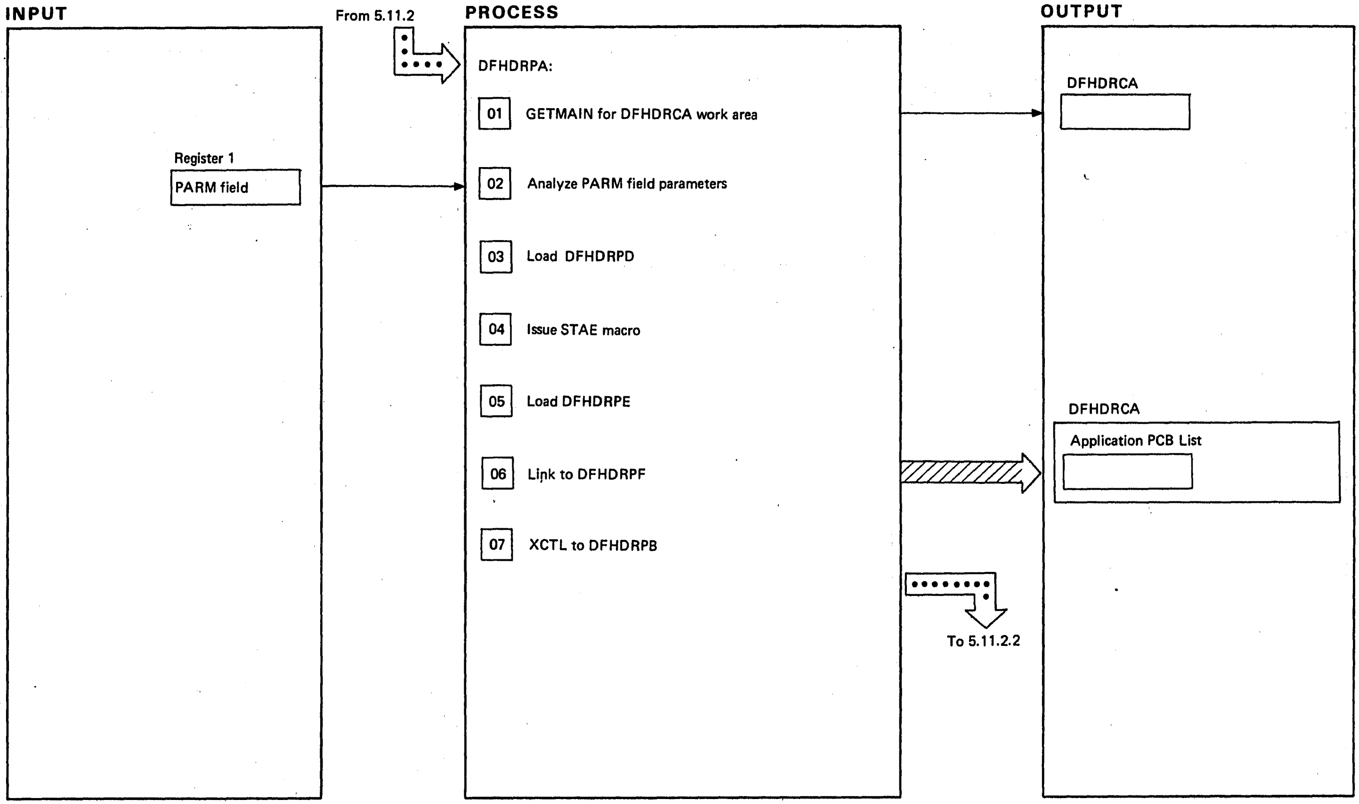
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 DFHLIB is the DD name of the (APF) authorized library, CICS.LOADLIB1 | | | |
| 02 DFHDRPA is the batch region initialization module, and needs to be (APF) authorized | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Batch Region Controller - Bootstrap

Diagram — 5.11.2-01

Page 1 of 1



Batch Region Controller - Initialization

Diagram — 5.11.2.1-01

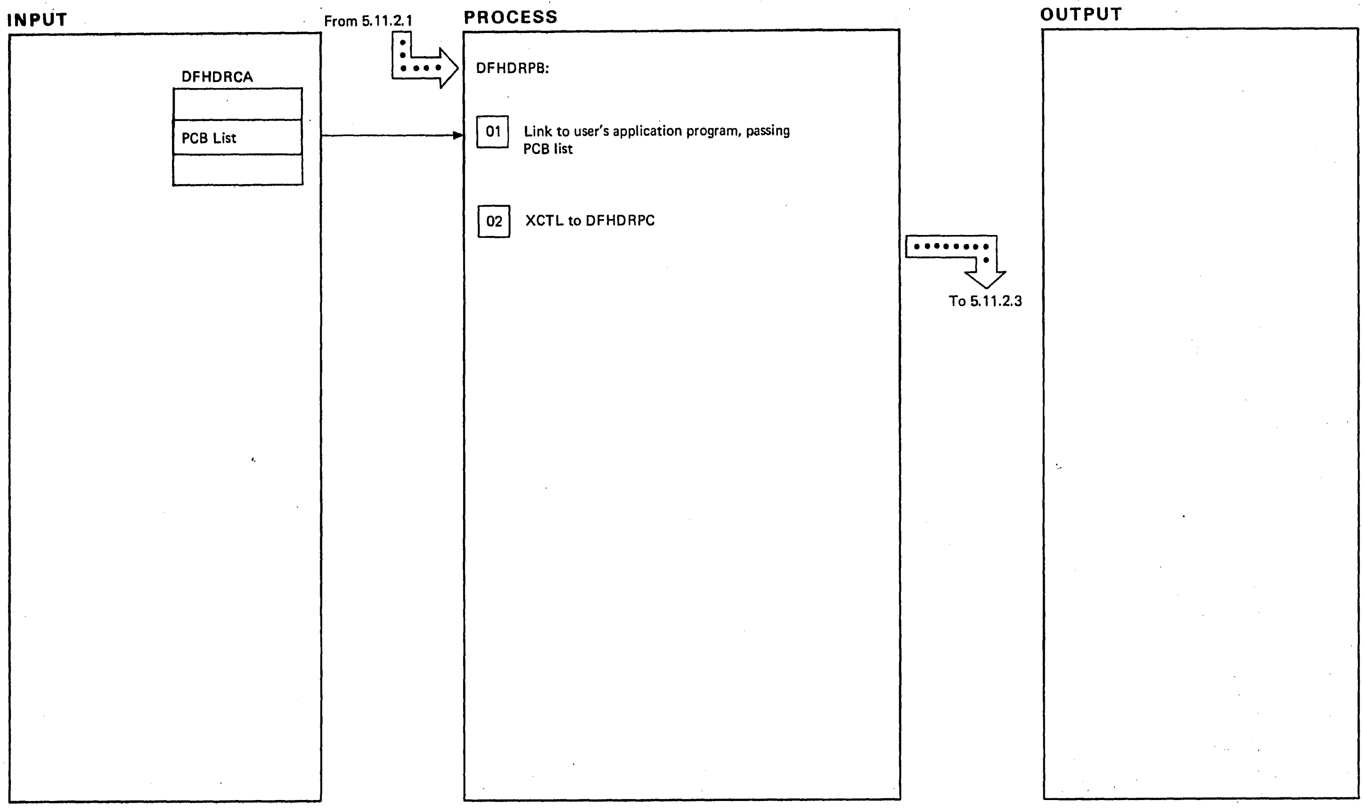
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 DFHDRCA is the batch region control block | | | |
| 03 DFHDRPD contains STAE exit, SPIE exit, and cleanup routines | | | |
| 05 DFHDRPE contains the Program Request Handler, and Converse Routine (which communicates with the CICS/VS region) | | | |
| 06 DFHDRPF performs initialization required to set up the link to the CICS/VS region, and invokes CICS/VS to schedule the application's PSB | | | |
| 07 Batch initialization complete | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Batch Region Controller - Initialization

Diagram — 5.11.2.1-01

Page 1 of 1



Batch Region Controller - Program Controller

Diagram — 5.11.2.2-01

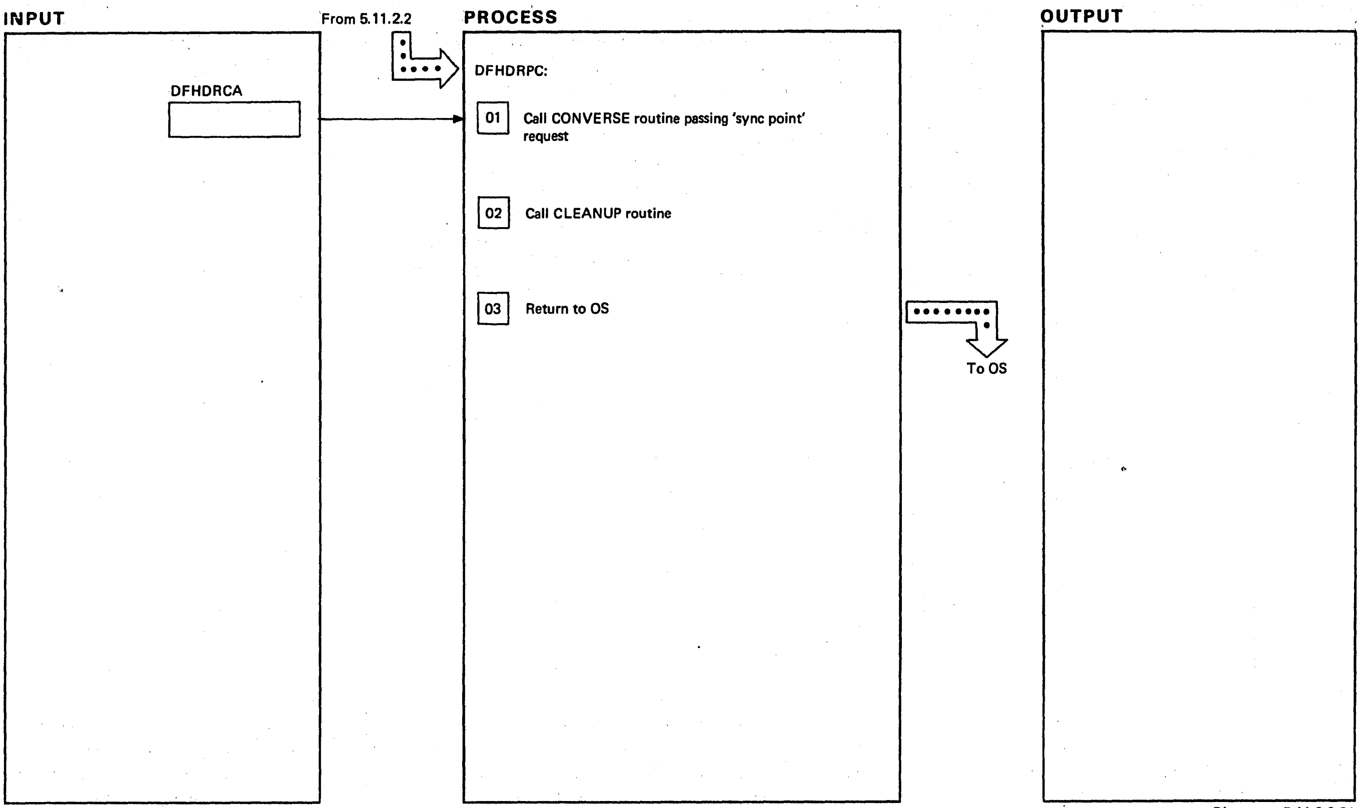
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 DFHDRPC is the batch region termination module | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Batch Region Controller - Program Controller

Diagram — 5.11.2.2-01

Page 1 of 1



DFHDRPC:

01 Call CONVERSE routine passing 'sync point' request

02 Call CLEANUP routine

03 Return to OS

Batch Region Controller - Termination

Diagram – 5.11.2.3-01

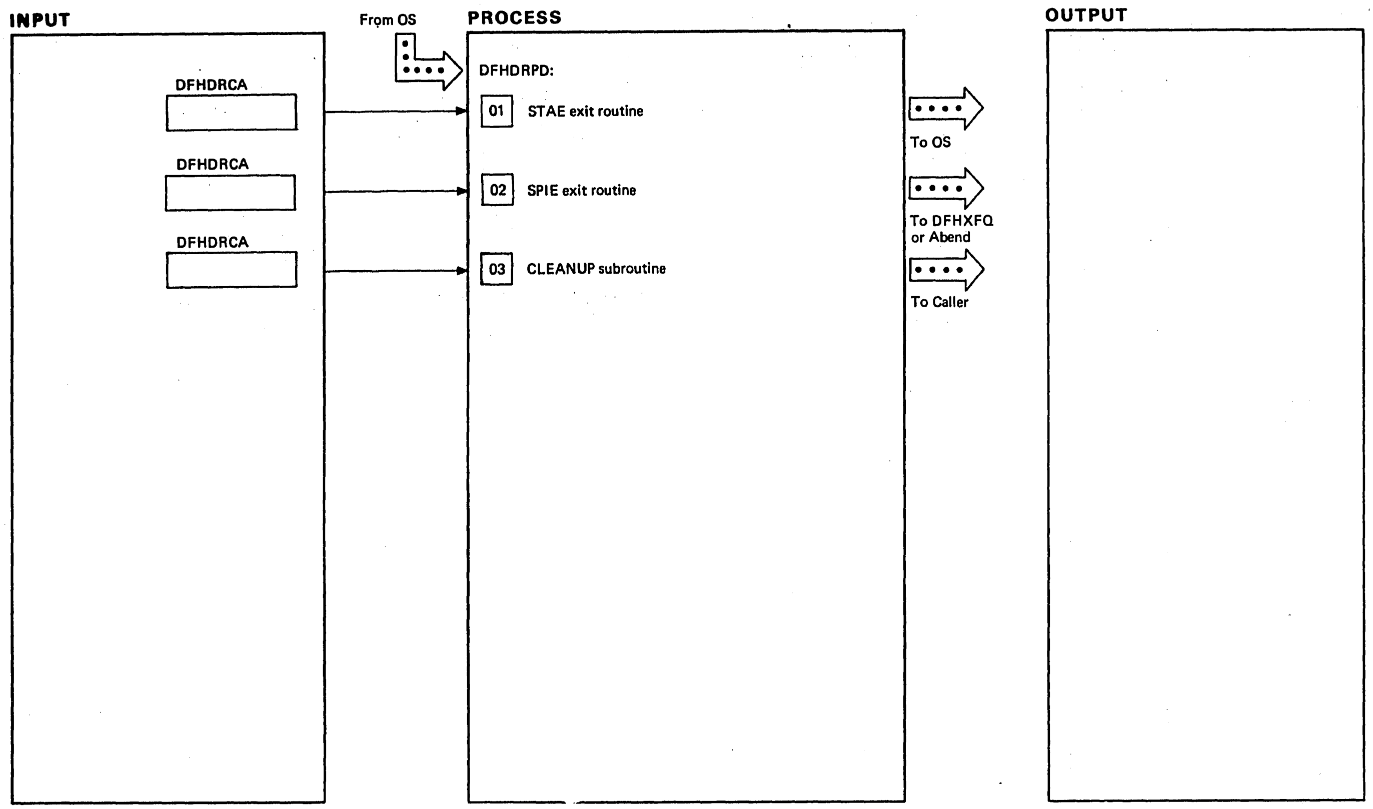
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 CONVERSE routine (in DFHDRPE) sends request to CICS/VS, and awaits reply. 'Sync Point' request informs CICS/VS that batch processing is complete | | | |
| 02 CLEANUP routine (in DFHDRPD) frees links with CICS/VS and the interregion SVC | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Batch Region Controller - Termination

Diagram – 5.11.2.3-01

Page 1 of 1



Batch Region Controller - STAE/SPIE/CLEANUP Routines

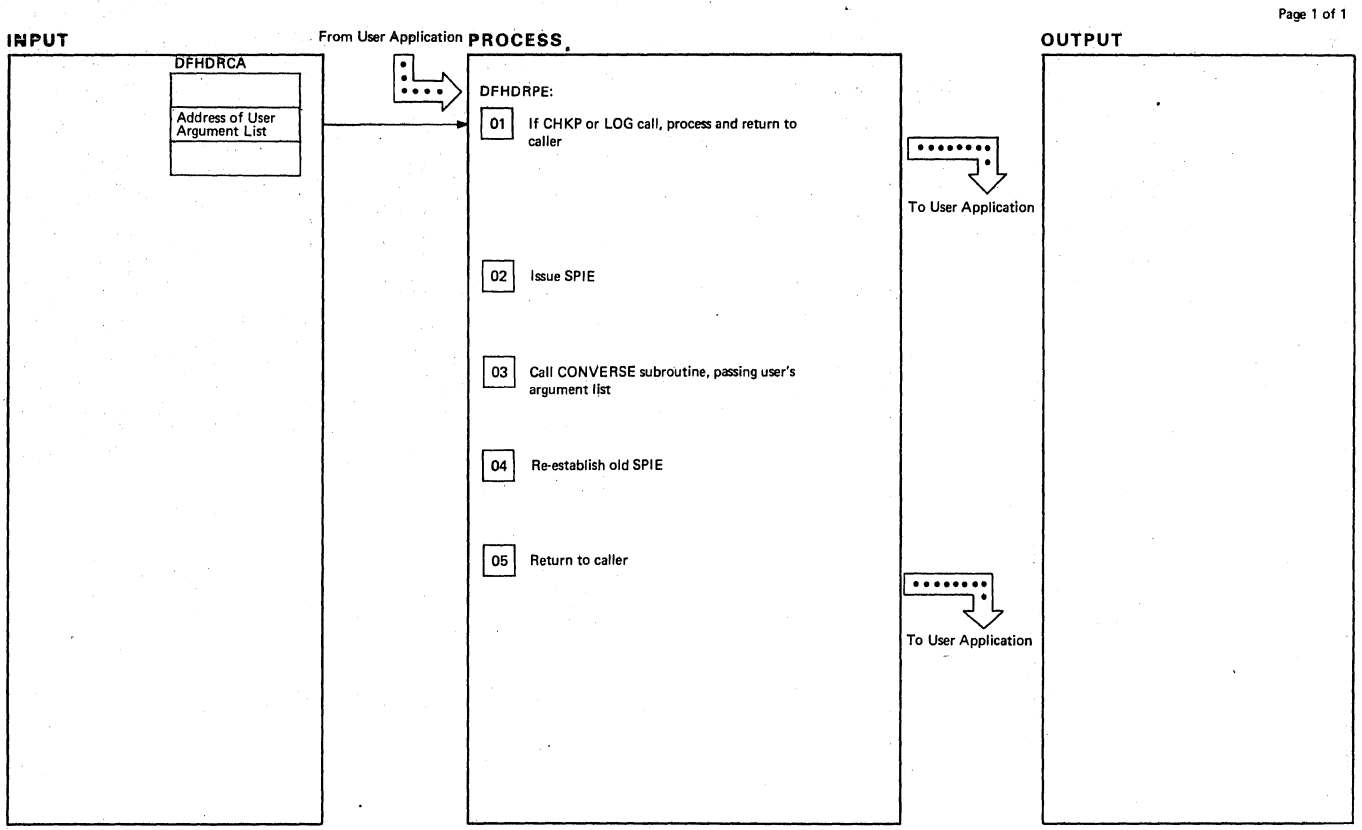
Diagram — 5.11.2.4-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Calls CLEANUP subroutine and continues abend | | STA000 | |
| 02 If program check caused by excessive data move by the batch transformer (DFHXFQ), then return to DFHXFQ; otherwise call CLEANUP subroutine and abend | | SPI000 | |
| 03 Issues DISCONNECT and LOGOFF requests to the interregion SVC | | CLN000 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Batch Region Controller - STAE/SPIE/CLEANUP Routines

Diagram — 5.11.2.4-01



DFHDRPE:

01 If CHKP or LOG call, process and return to caller

02 Issue SPIE

03 Call CONVERSE subroutine, passing user's argument list

04 Re-establish old SPIE

05 Return to caller

Batch Region Controller - Program Request Handler

Diagram – 5.11.2.5-01

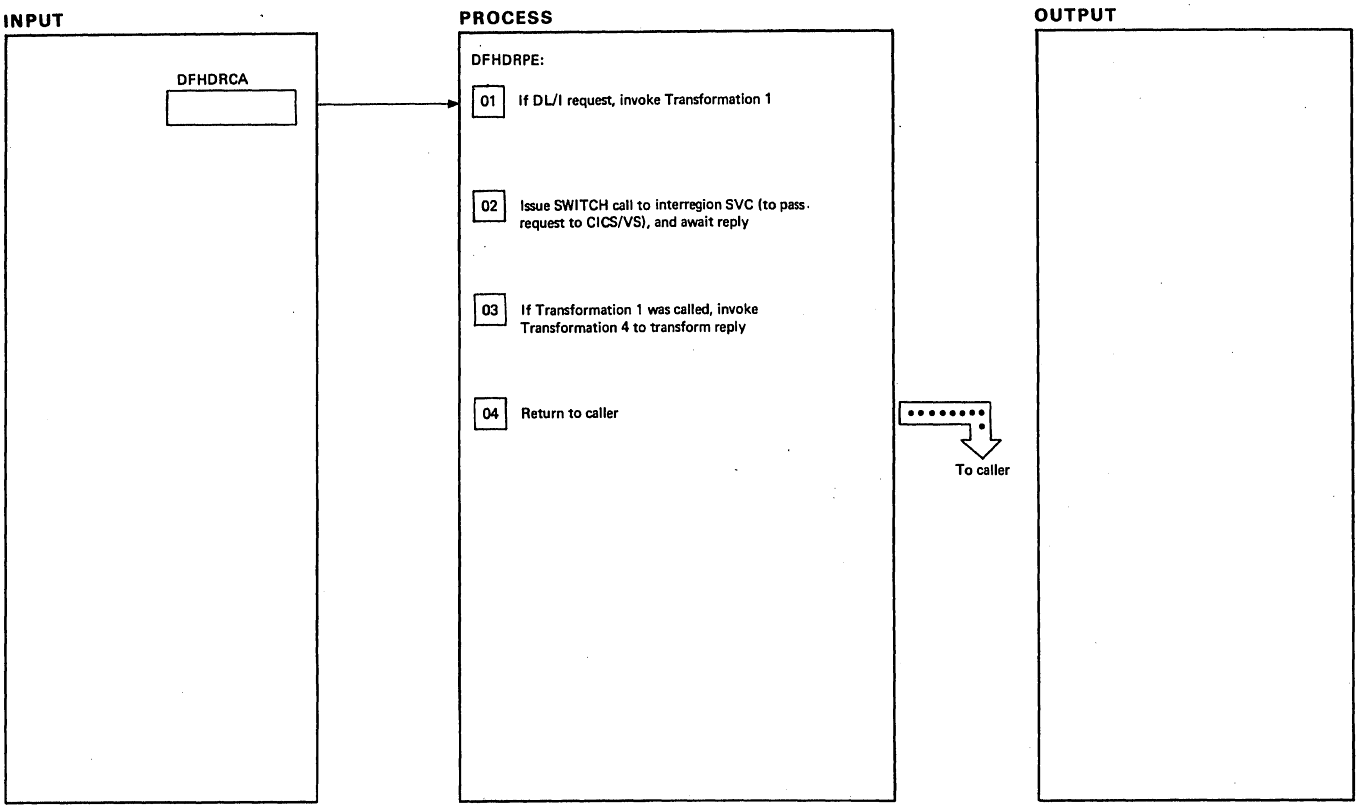
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Program Request Handler receives control when the user's application issues a DL/I request. The language interface routine (DFSLI000) leaves the address of the uer's argument list in the work area (DFHDRCA) | | PRH000 | |
| 03 CONVERSE subroutine transmits request to CICS/VS and processes reply | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Batch Region Controller - Program Request Handler

Diagram – 5.11.2.5-01

Page 1 of 1



DFHDRPE:

01 If DL/I request, invoke Transformation 1

02 Issue SWITCH call to interregion SVC (to pass request to CICS/VS), and await reply

03 If Transformation 1 was called, invoke Transformation 4 to transform reply

04 Return to caller

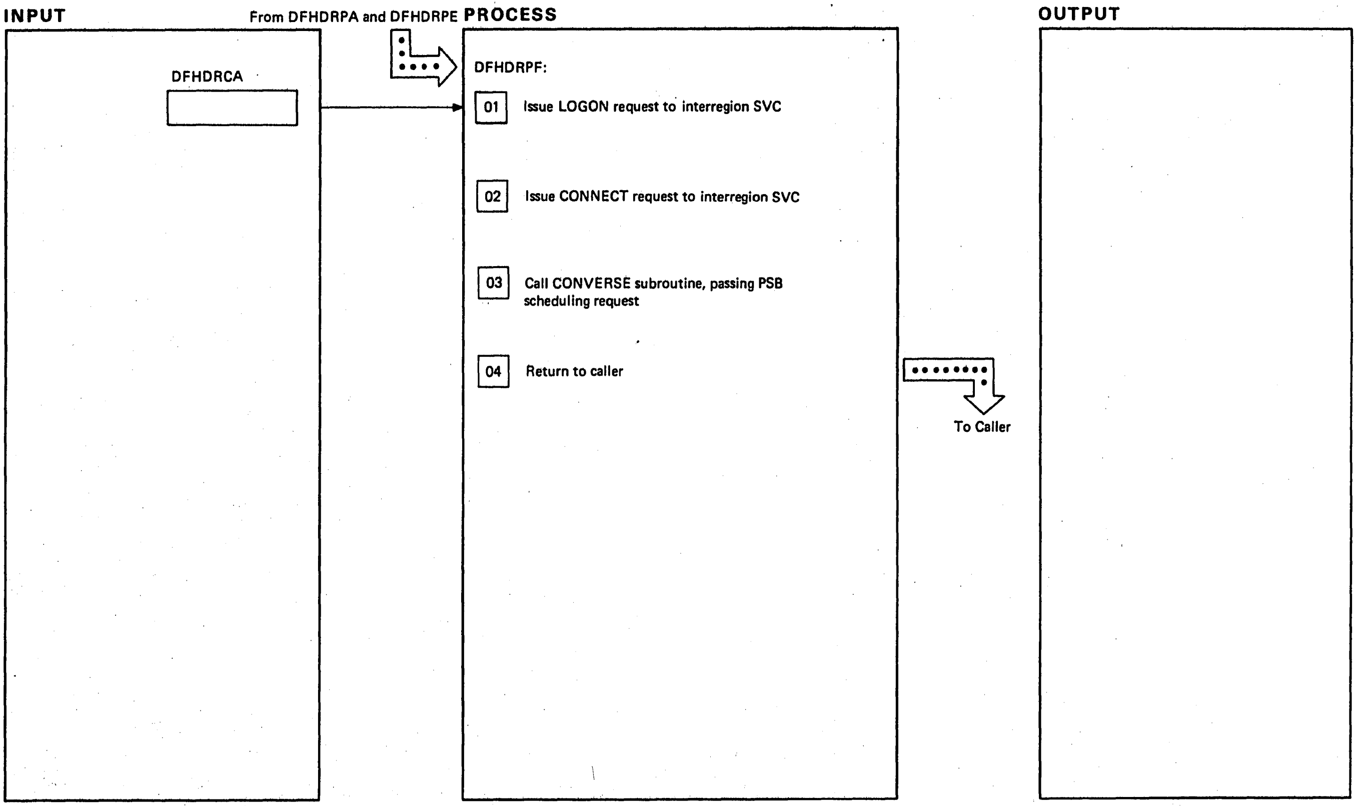
Batch Region Controller - CONVERSE Subroutine

Diagram — 5.11.2.6-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 If data to be transmitted is a DL/I request (as opposed to, for example, a sync point request), then the argument list must be transformed into a format which does not contain addresses, so that it may be interpreted by the CICS/VS region. Transformation 1 (in DFHXFQ) performs this transformation | | CVR000 | |
| 03 The reply from CICS/VS must be used to set the variables passed (for example, PCB, input I/O area) in the original DL/I argument list. Transformation 4 (in DFHXFQ) does this | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Page 1 of 1



01 Issue LOGON request to interregion SVC

02 Issue CONNECT request to interregion SVC

03 Call CONVERSE subroutine, passing PSB scheduling request

04 Return to caller

Batch Region Controller - SVC Initialization

Diagram – 5.11.2.7-01

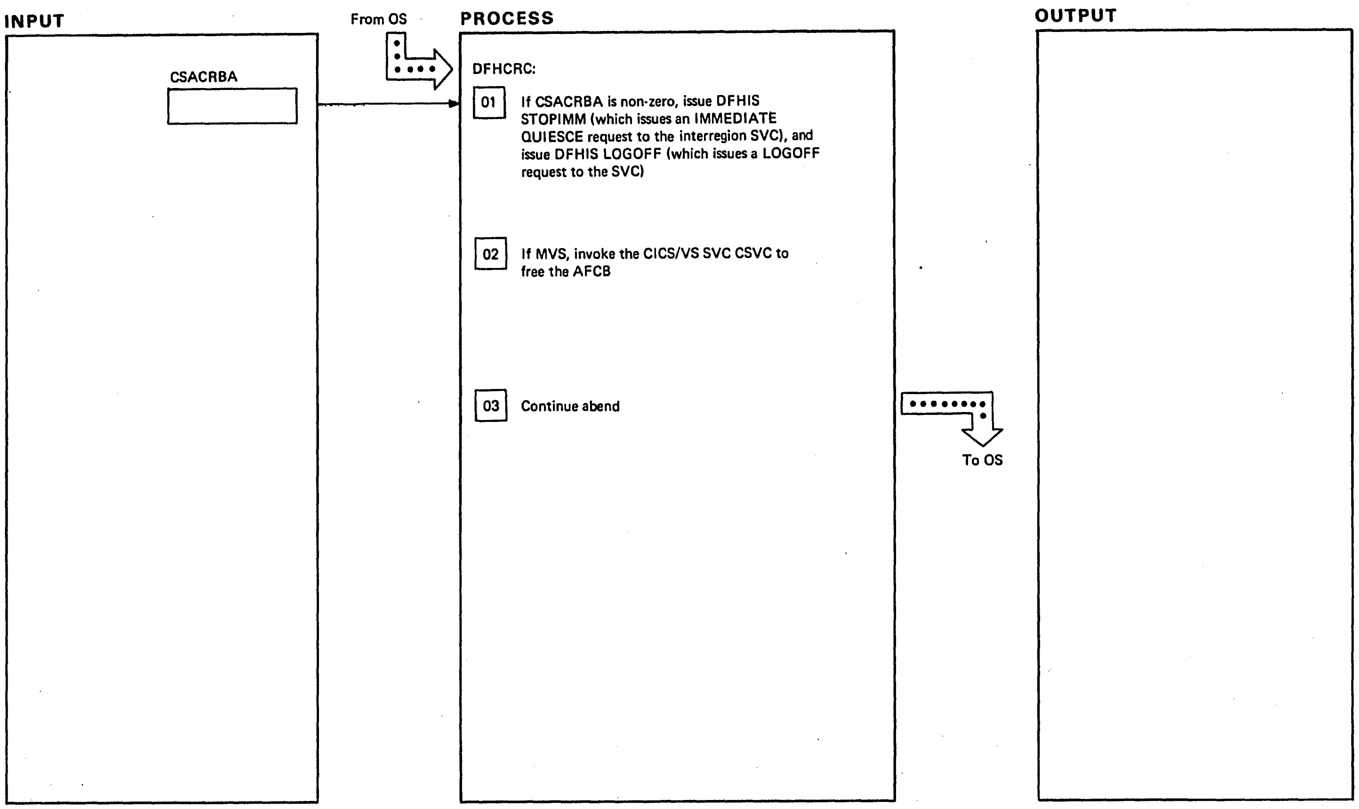
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Make batch region known to the SVC | | | |
| 02 Request connection to CICS/VS region that owns the PSB to be shared | | | |
| 03 Pass the name of the PSB specified in the PARM field of the EXEC card | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Batch Region Controller - SVC Initialization

Diagram – 5.11.2.7-01

Page 1 of 1



DFHCRC:

01 If CSACRBA is non-zero, issue DFHIS STOPIMM (which issues an IMMEDIATE QUIESCE request to the interregion SVC), and issue DFHIS LOGOFF (which issues a LOGOFF request to the SVC)

02 If MVS, invoke the CICS/VS SVC CSVC to free the AFCB

03 Continue abend

Interregion STAE Exit

Diagram — 5.11.3-01

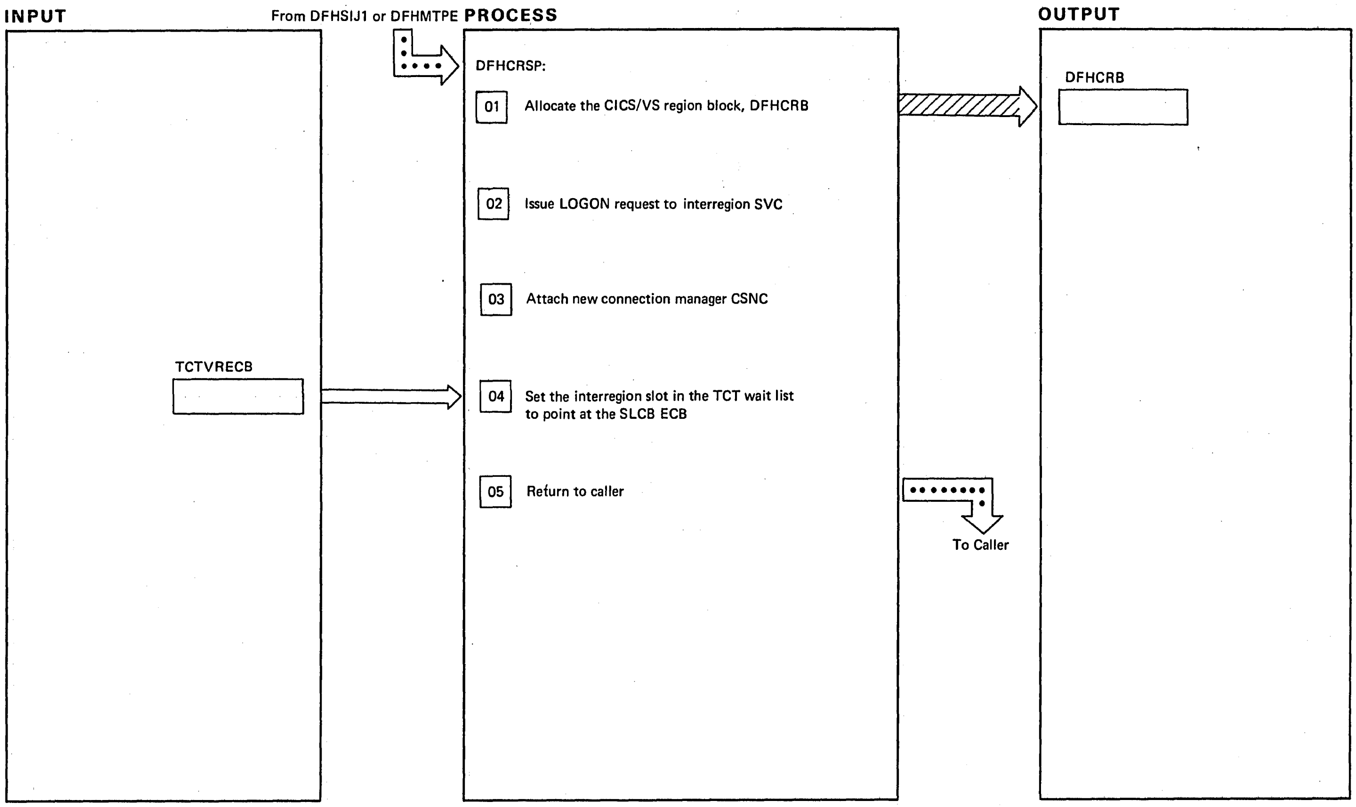
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 DFHCRC is the STAE exit which cleans up interregion communication. If CSACRBA is zero, interregion communication is inactive | | | |
| 02 The AFCB is the block built by the DFHAUTH macro | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Interregion STAE Exit

Diagram — 5.11.3-01

Page 1 of 1



INPUT

From DFHSIJ1 or DFHMTPE

PROCESS

OUTPUT

DFHCRSP:

01 Allocate the CICS/VS region block, DFHCRB

02 Issue LOGON request to interregion SVC

03 Attach new connection manager CSNC

04 Set the interregion slot in the TCT wait list to point at the SLCB ECB

05 Return to caller

Interregion Start-Up

Diagram – 5.11.3.1-01

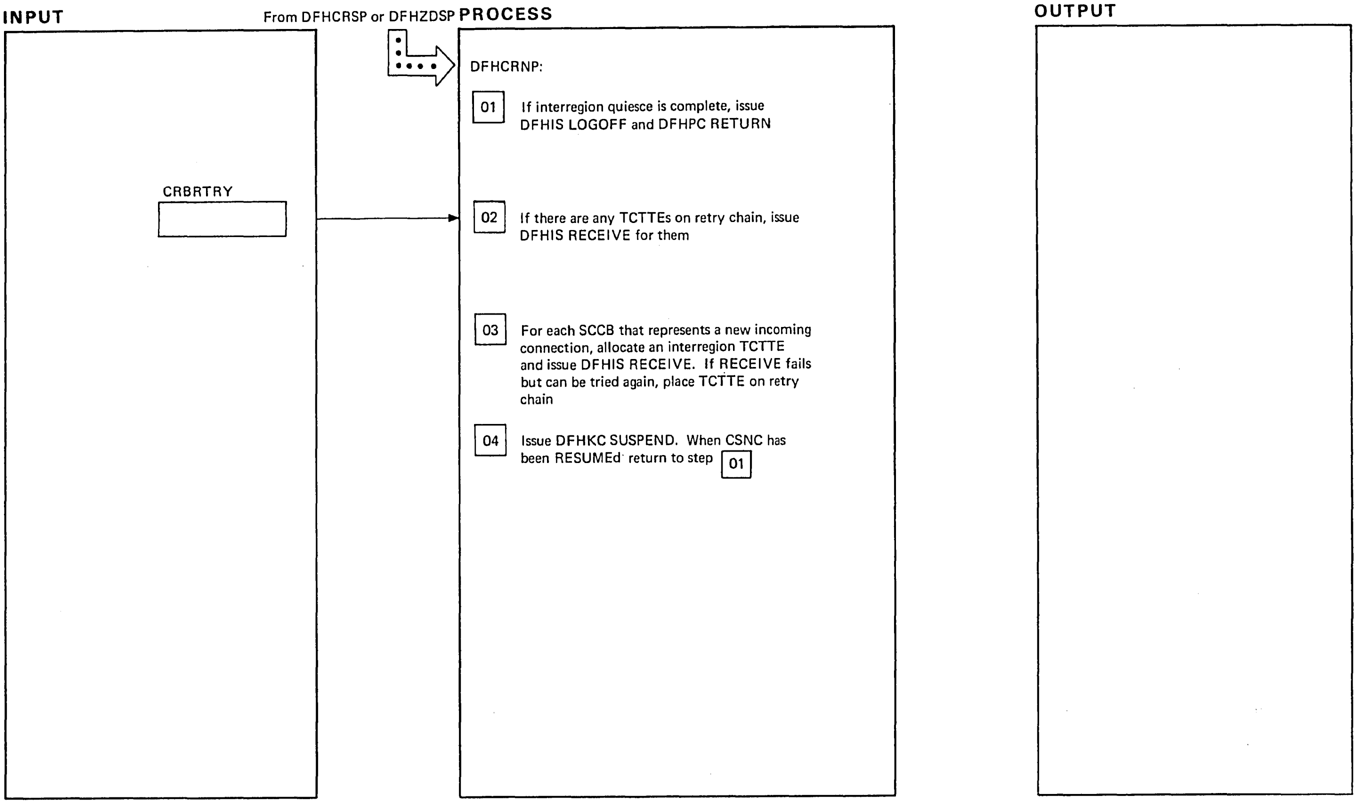
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 DFHCRSP is the interregion start-up program | | | |
| 03 The program corresponding to CSNC is DFHCRNP. CSNC remains attached until the interregion session is closed | | | |
| 04 The interregion SVC posts this ECB when there is work for CICS/VS. The ZCP dispatcher, ZDSP, notices the posting and RESUMEs CSNC if there are any new connections to handle, or if Interregion Quiesce is complete | | CRS040 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Interregion Start-Up

Diagram 5.11.3.1-01

Page 1 of 1



01 If interregion quiesce is complete, issue DFHIS LOGOFF and DFHPC RETURN

02 If there are any TCTTEs on retry chain, issue DFHIS RECEIVE for them

03 For each SCCB that represents a new incoming connection, allocate an interregion TCTTE and issue DFHIS RECEIVE. If RECEIVE fails but can be tried again, place TCTTE on retry chain

04 Issue DFHKC SUSPEND. When CSNC has been RESUMEd return to step 01

New Connection Manager

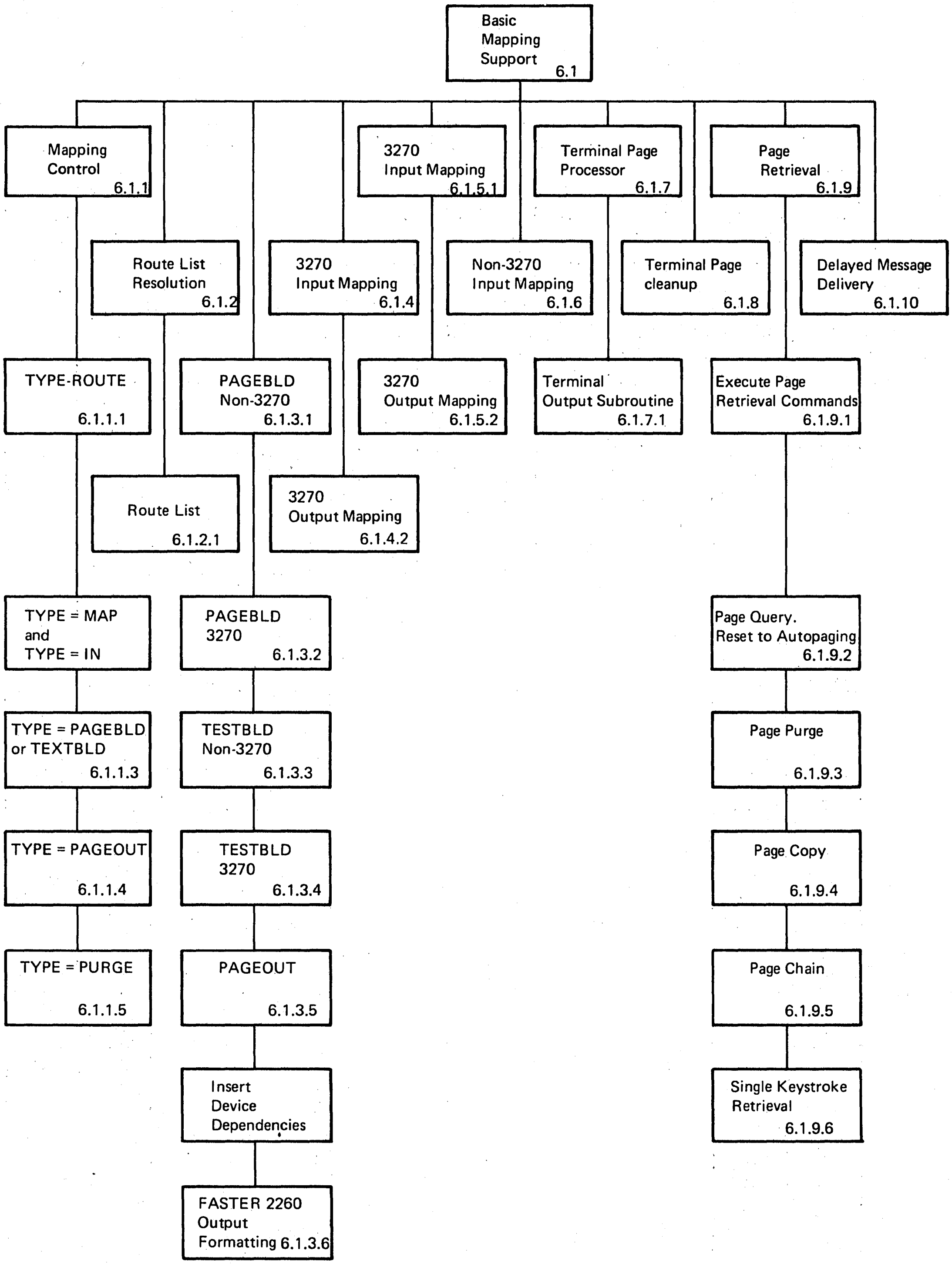
Diagram – 5.11.3.2-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 If quiesce is complete, LOGOFF from SVC and complete transaction | | CRN030 | |
| 02 DFHIS RECEIVE attaches the mirror transaction, CSMI. If the attach fails, but can be tried again (for example, MAXTASKS), then the interregion TCTTE is placed on the retry chain | | CRN050 | |
| 03 The SCCBs are scanned using the SCACB. These blocks are all generated by the interregion SVC. (See also note 02 above) | | CRN150 | |
| 04 CSNC is only active when new incoming connections are to be handled, and when quiesce is complete. Incoming data on existing connections is awaited, using DFHKC TYPE=WAIT, by the mirror transaction, CSMI. CSNC is RESUMEd by DFHZDSP, the DFHZCP dispatcher | | CRN260 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

New Connection Manager

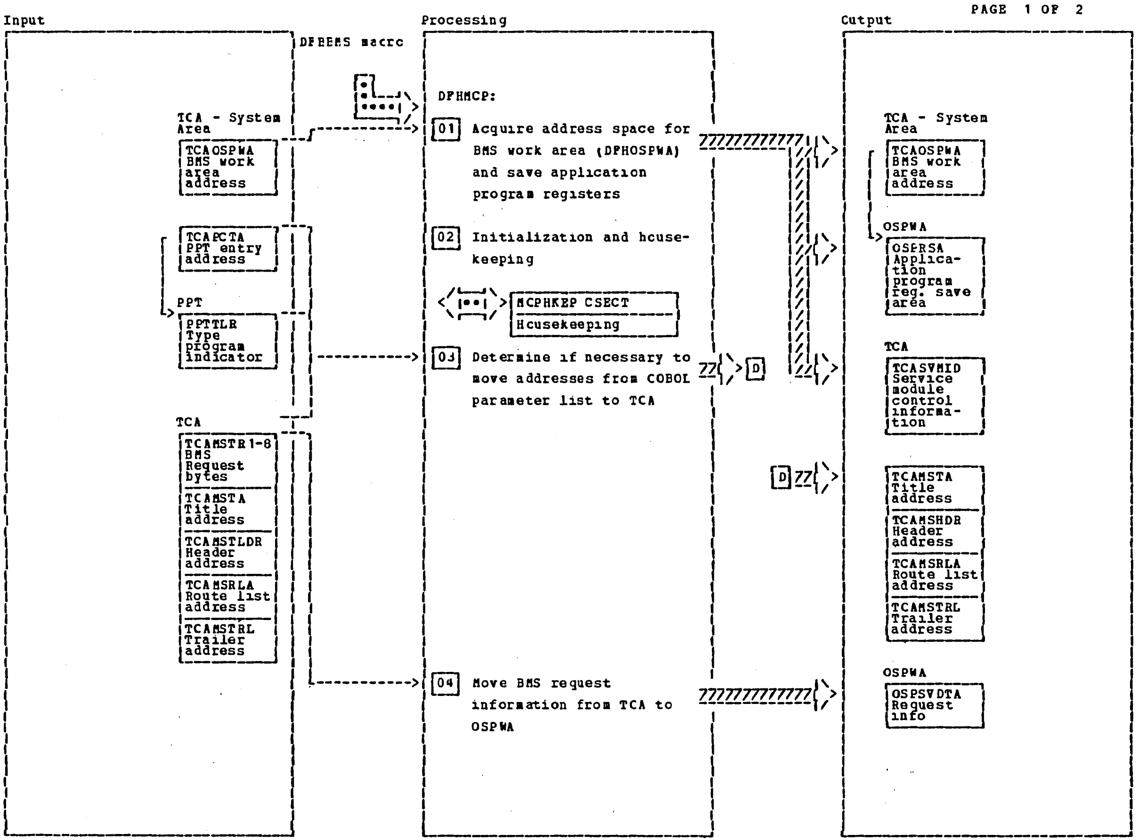
Diagram – 5.11.3.2-01


Basic Mapping Support 6.1
Mapping Control 6.1.1
Route List Resolution 6.1.2
3270 Input Mapping 6.1.5.1
3270 Input Mapping 6.1.4
Non-3270 Input Mapping 6.1.6
Terminal Page Processor 6.1.7
Terminal Page cleanup 6.1.8
Page Retrieval 6.1.9
Delayed Message Delivery 6.1.10
TYPE-ROUTE 6.1.1.1
PAGEBLD Non-3270 6.1.3.1
3270 Output Mapping 6.1.5.2
Terminal Output Subroutine 6.1.7.1
Execute Page Retrieval Commands 6.1.9.1
Route List 6.1.2.1
3270 Output Mapping 6.1.4.2
TYPE = MAP and TYPE = IN
PAGEBLD 3270 6.1.3.2
Page Query. Reset to Autopaging 6.1.9.2
TYPE = PAGEBLD or TEXTBLD 6.1.1.3
TESTBLD Non-3270 6.1.3.3
Page Purge 6.1.9.3
TYPE = PAGEOUT 6.1.1.4
TESTBLD 3270 6.1.3.4
Page Copy 6.1.9.4
TYPE = PURGE 6.1.1.5
PAGEOUT 6.1.3.5
Page Chain 6.1.9.5
Insert Device Dependencies
Single Keystroke Retrieval 6.1.9.6
FASTER 2260 Output Formatting 6.1.3.6

Basic Mapping Support

Diagram – 6.1.0-01

PAGE 1 OF 2


Input
DFHBMS macro
TCA - System Area
TCAOSPWA BMS work area address
TCAPCTA PPT entry address
PPT
PPTTLR Type program indicator
TCA
TCAMSTR1-8 BMS Request bytes
TCAMSTA Title address
TCAMSTLDR Header address
TCAMSRLA Route list address
TCAMSTRL Trailer address
Processing
DFHMCP:
01 Acquire address space for BMS work area (DFHOSPWA) and save application program registers
02 Initialization and housekeeping
MCPHKEP CSECT
Housekeeping
03 Determine if necessary to move addresses from COBOL parameter list to TCA
D
04 Move BMS request information from TCA to OSPWA
Cutput
TCA - System Area
TCAOSPWA BMS work area address
OSPWA
OSPRSA Application program reg. save area
TCA
TCASVMID Service module control information
TCAMSTA Title address
TCAMSHDR Header address
TCAMSRLA Route list address
TCAMSTRL Trailer address
OSPWA
OSPSVDTA Request info

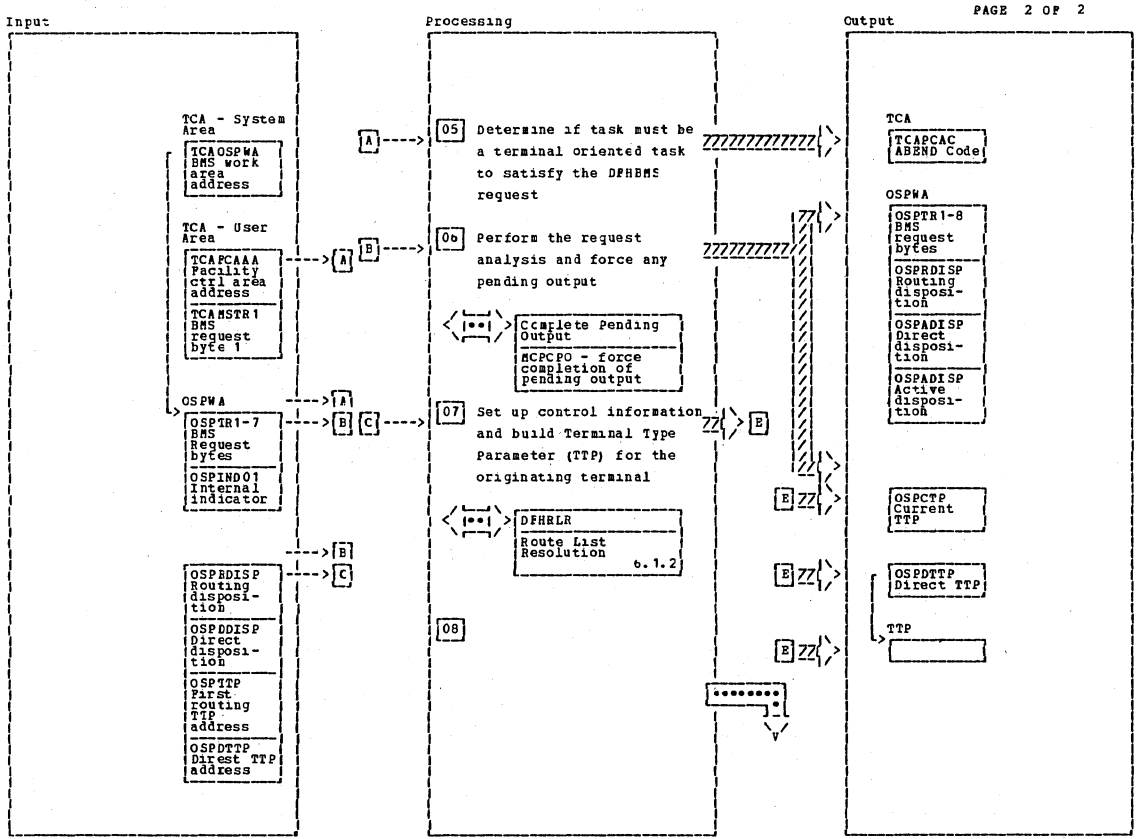
BMS Control: Request Analysis

Diagram - 6.1.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Some application programs registers are temporarily saved in the TCA until the OSPWA is acquired.<br>Indicate to Interval Control Program that, if the Runaway Task interval expires during the execution of BMS ervice programs, abnormal termination should be deferred until control has been returned to the application program. |  | DFHMCPNA |  |  |  |  |  |
| 02 Set up DWE exit address.<br>Initiate CSPG to monitor Purge Delay (PRGDLAY), if sysgened by the user. |  |  |  |  |  |  |  |
| 03 |  | MCPMC1ST |  |  |  |  |  |
| 04 This consists of all information in the TCA common communication area except the common register save area. |  |  |  |  |  |  |  |

BMS Control: Request Analysis

Diagram - 6.1.1-01


PAGE 2 OF 2
Input
Processing
Output
TCA - System Area
TCAOSPWA BMS work area address
TCA - User Area
TCAFCAAA Facility ctrl area address
TCAMSTR1 BMS request byte 1
OSPWA
OSPTR1-7 BMS Request bytes
OSPIND01 Internal indicator
OSPRDISP Routing disposi-tion
OSPDDISP Direct disposi-tion
OSPTTP First routing TTP address
OSPDTTP Direst TTP address
A
B
C
05 Determine if task must be a terminal oriented task to satisfy the DFHBMS request
06 Perform the request analysis and force any pending output
Complete Pending Output
MCPCPO - force completion of pending output
07 Set up control information and build Terminal Type Parameter (TTP) for the originating terminal
DFHRLR
Route List Resolution 6.1.2
08
E
TCA
TCAPCAC ABEND Code
OSPWA
OSPTR1-8 BMS request bytes
OSPRDISP Routing disposi-tion
OSPADISP Direct disposi-tion
OSPADISP Active disposi-tion
OSPCTP Current TTP
OSPDTTP Direct TTP
TTP

BMS Control: Request Analysis

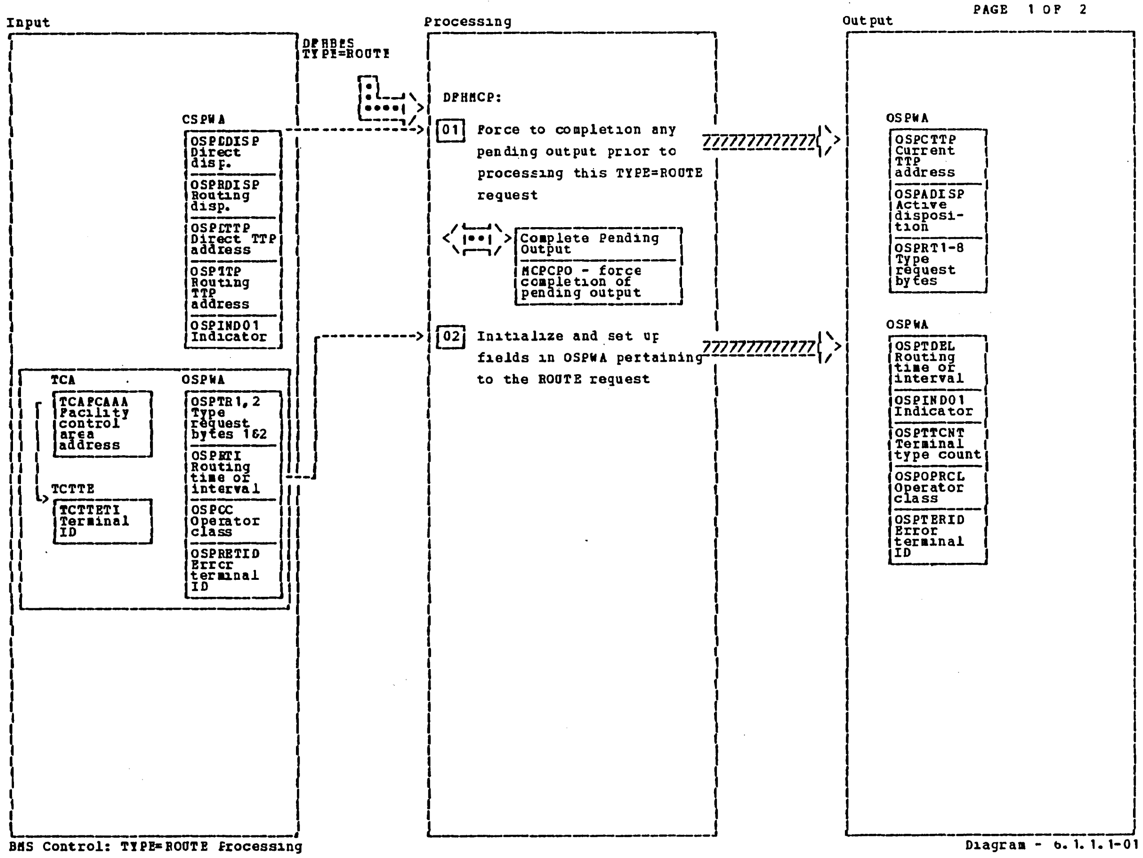
Diagram - 6.1.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 Transaction ABEND ABM3 is issued if the task is not attached to a terminal and should have been. | | | | | | | |
| 06 If it is necessary, previously built output will be forced to completion so that the current request can be processed. | | MCPBYTCK | | | | | |
| 08 Pass control to the routine processing specific request: see overview. | | | 6.1 | | | | |

BMS Control: Request Analysis

Diagram - 6.1.1-02

PAGE 1 OF 2


Input
DFHBMS
TYPE=ROUTE
CSPWA
OSPDDISP Direct disp.
OSPRDISP Routing disp.
OSPDTTP Direct TTP address
OSPITP Routing TTP address
OSPIND01 Indicator
TCA
TCAFCAAA Facility control area address
TCTTE
TCTTETI Terminal ID
OSPWA
OSPTR1,2 Type request bytes 1&2
OSPRTI Routing time or interval
OSPCC Operator class
OSPRETID Error terminal ID
Processing
DFHMCP:
01 Force to completion any pending output prior to processing this TYPE=ROUTE request
Complete Pending Output
MCPCPO - force completion of pending output
02 Initialize and set up fields in OSPWA pertaining to the ROUTE request
Output
OSPWA
OSPCTTP Current TTP address
OSPADISP Active disposition
OSPRT1-8 Type request bytes
OSPWA
OSPTDEL Routing time or interval
OSPIND01 Indicator
OSPTTCNT Terminal type count
OSPOPRCL Operator class
OSPTERID Error terminal ID

BMS Control: TYPE=ROUTE Processing

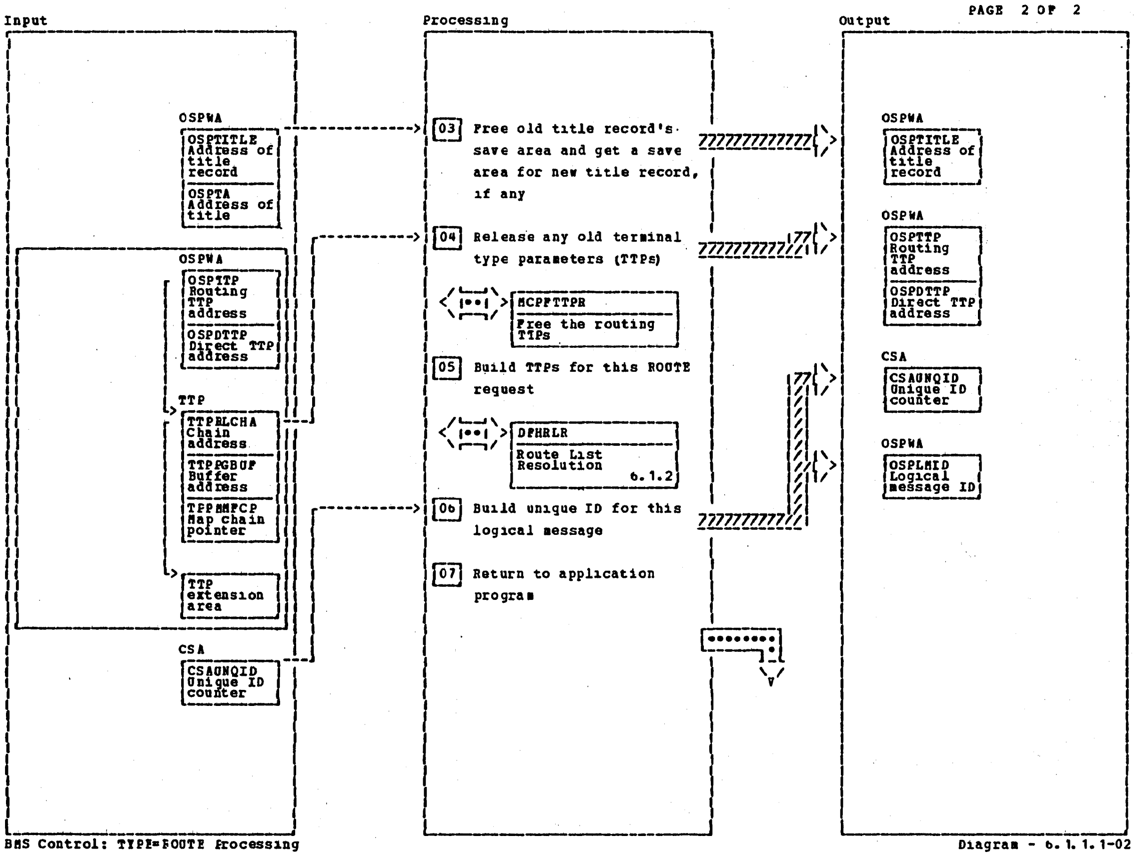
Diagram - 6.1.1.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The pending output that is forced to completion may be for the direct terminal or the routed terminals. | | MCPROUTE | | | | | |
| 02 Fields from the ROUTE request must be saved to be used at the completion of the logical message built under this routing environment. | | MCPORTE | | | | | |

BMS Control: TYPE=ROUTE Processing

Diagram - 6.1.1.1-01

PAGE 2 OF 2


Input
Processing
Output
OSPWA
OSPTITLE Address of title record
OSPTA Address of title
OSPWA
OSPTTP Routing TTP address
OSPDTTP Direct TTP address
TTP
TTPELCHA Chain address
TTPPGBUF Buffer address
TTPBMFCP Map chain pointer
TTP extension area
CSA
CSAUNQID Unique ID counter
DFHCPFTTPR
Free the routing TTPs
DFHRLR
Route List Resolution
6.1.2
OSPWA
OSPTITLE Address of title record
OSPWA
OSPTTP Routing TTP address
OSPDTTP Direct TTP address
CSA
CSAUNQID Unique ID counter
OSPWA
OSPLMID Logical message ID

03 Free old title record's save area and get a save area for new title record, if any

04 Release any old terminal type parameters (TTPs)

05 Build TTPs for this ROUTE request

06 Build unique ID for this logical message

07 Return to application program

BMS Control: TYPE=ROUTE Processing

Diagram - 6.1.1.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 | | DFHCPCKTTL | | | | | |
| 04 Also releases any storage chained off the TTPs. | | DFHCPCKTTP | | | | | |
| 05 | | DFHCPBTTP | | | | | |
| 07 | | DFHCPEXIT | | | | | |

BMS Control: TYPE=ROUTE Processing

Diagram - 6.1.1.1-02

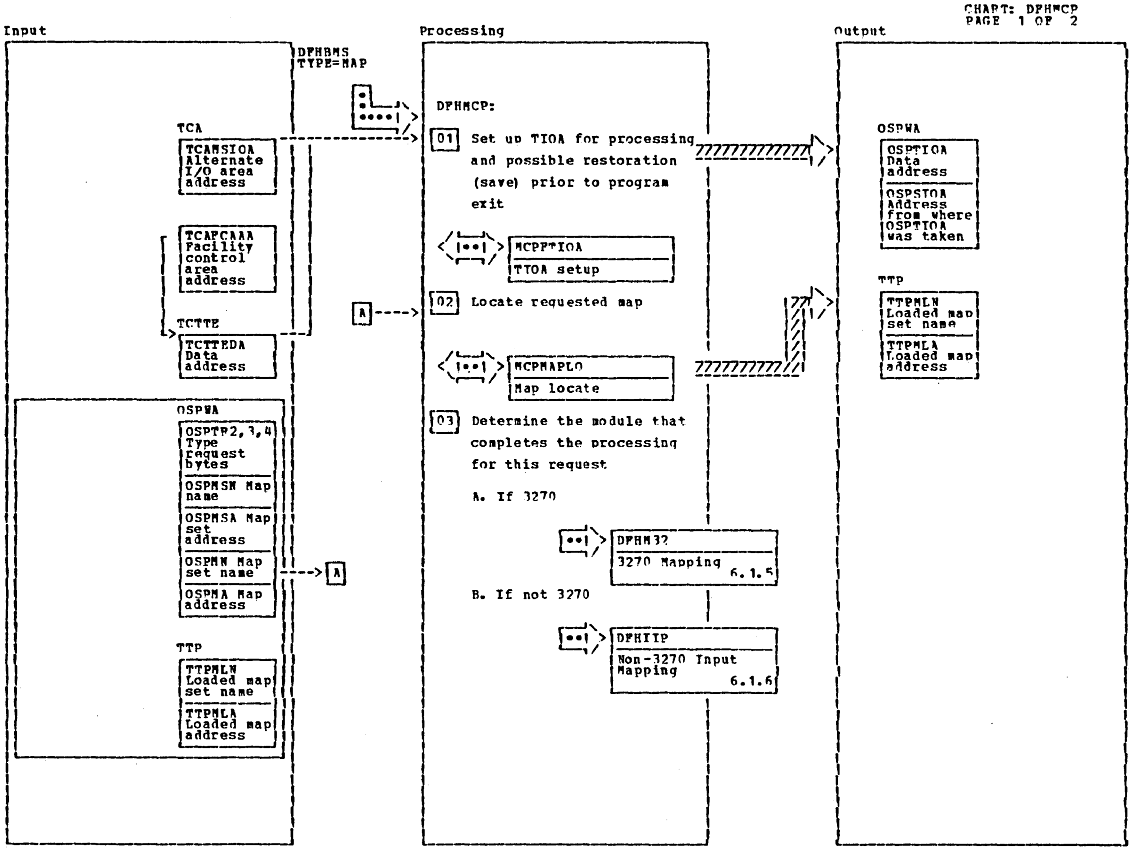


CHART: DFHMCP
PAGE 1 OF 2
Input
Processing
Output
DFHBMS
TYPE=MAP
TCA
TCAMSIOA Alternate I/O area address
TCAFCAAA Facility control area address
TCTTE
TCTTEDA Data address
OSPWA
OSPTP2,3,4 Type request bytes
OSPMSN Map name
OSPMSA Map set address
OSPMN Map set name
OSPMA Map address
TTP
TTPMLN Loaded map set name
TTPMLA Loaded map address
A
DFHMCP:
01 Set up TIOA for processing and possible restoration (save) prior to program exit
MCPPTIOA
TIOA setup
A
02 Locate requested map
MCPMAPLO
Map locate
03 Determine the module that completes the processing for this request
A. If 3270
DFHM32
3270 Mapping 6.1.5
B. If not 3270
DFHIIP
Non-3270 Input Mapping 6.1.6
OSPWA
OSPTIOA Data address
OSPSTOA Address from where OSPTIOA was taken
TTP
TTPMLN Loaded map set name
TTPMLA Loaded map address

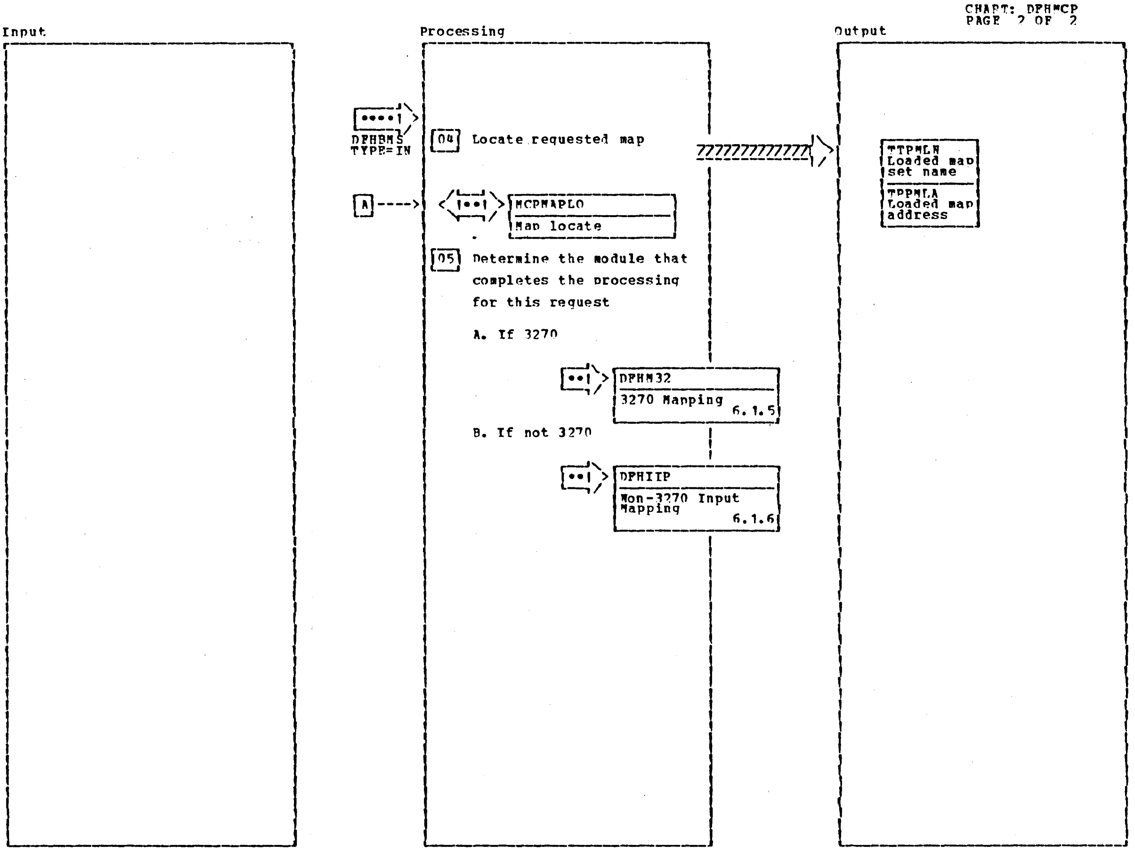
BMS Control: TYPE=MAP and TYPE=IN Processing

Diagram - 6.1.1.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If no TIOA supplied, terminate with 'ABM2'. | | MCPMAP | | | | | |
| 02 Deletes old map set if no longer needed. Checks if map too large for page. Terminates with 'ABMO' if map not found or 'ABMI' if map is not an input mode map. | | MCPGTMAP | | | | | |
| If there is a Tab map, its address is stored in TTPTFMA, and the format indicator is moved to TTPTFMI to indicate actual horizontal/vertical tabs. | MCPMAPLO | | | | | | |
| If there is no Tab map, TTPTFMA is set to a negative value, and TTPTFMI remains x'00'. | | | | | | | |

BMS Control: TYPE=MAP and TYPE=IN Processing

Diagram - 6.1.1.2-01



BMS Control: TYPE=MAP and TYPE=IN Processing

Diagram - 6.1.1.2-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

BMS Control: TYPE=MAP and TYPE=IN Processing

Diagram - 6.1.1.2-02

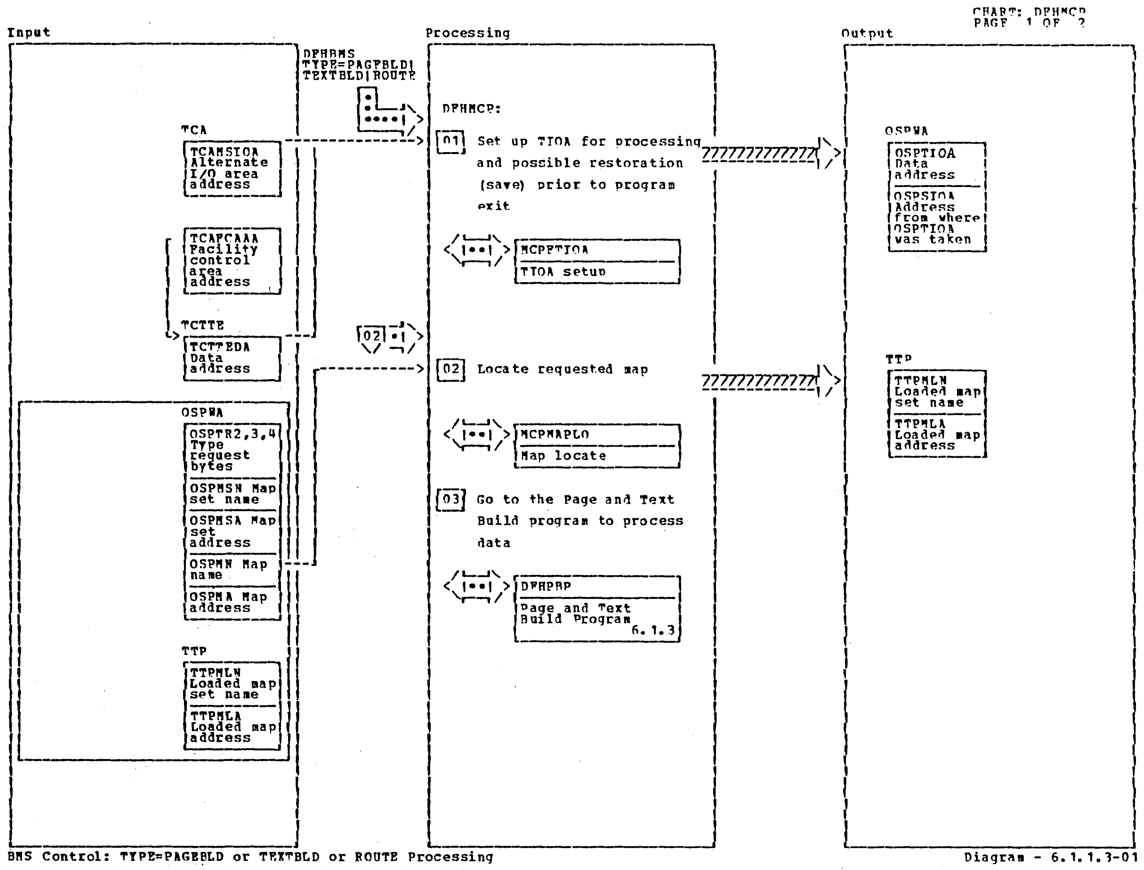


CHART: DFHMCP
PAGE 1 OF 2
Input
Processing
Output
DFHBMS
TYPE=PAGEBLD|
TEXTBLD|ROUTE
TCA
TCAMSIOA Alternate I/O area address
TCAFCAAA Facility control area address
TCTTE
TCTTEDA Data address
OSPWA
OSPTR2,3,4 Type request bytes
OSPMSN Map set name
OSPMSA Map set address
OSPMN Map name
OSPMA Map address
TTP
TTPMLN Loaded map set name
TTPMLA Loaded map address
DFHMCP:
01 Set up TIOA for processing and possible restoration (save) prior to program exit
MCPFTIOA
TIOA setup
02
02 Locate requested map
MCPMAPLO
Map locate
03 Go to the Page and Text Build program to process data
DFHPBP
Page and Text Build Program
6.1.3
OSPWA
OSPTIOA Data address
OSPSIOA Address from where OSPTIOA was taken
TTP
TTPMLN Loaded map set name
TTPMLA Loaded map address

BMS Control: TYPE=PAGEBLD or TEXTBLD or ROUTE Processing — Diagram - 6.1.1.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | MCPGTIOA | | | | | |
| 02 Deletes old map set if no longer needed. Checks if map too large for page. Terminates with 'ABMO' if map not found or 'ABMO' if map not output mode. The map locate just pertains to PAGEBLD requests. | | MCPGETMP | | | | | |
| 03 The Page Build program does all the page formatting and device dependence. | | MCPGOPG | | | | | |

BMS Control: TYPE=PAGEBLD or TEXTBLD or RCUTE Processing — Diagram - 6.1.1.3-01

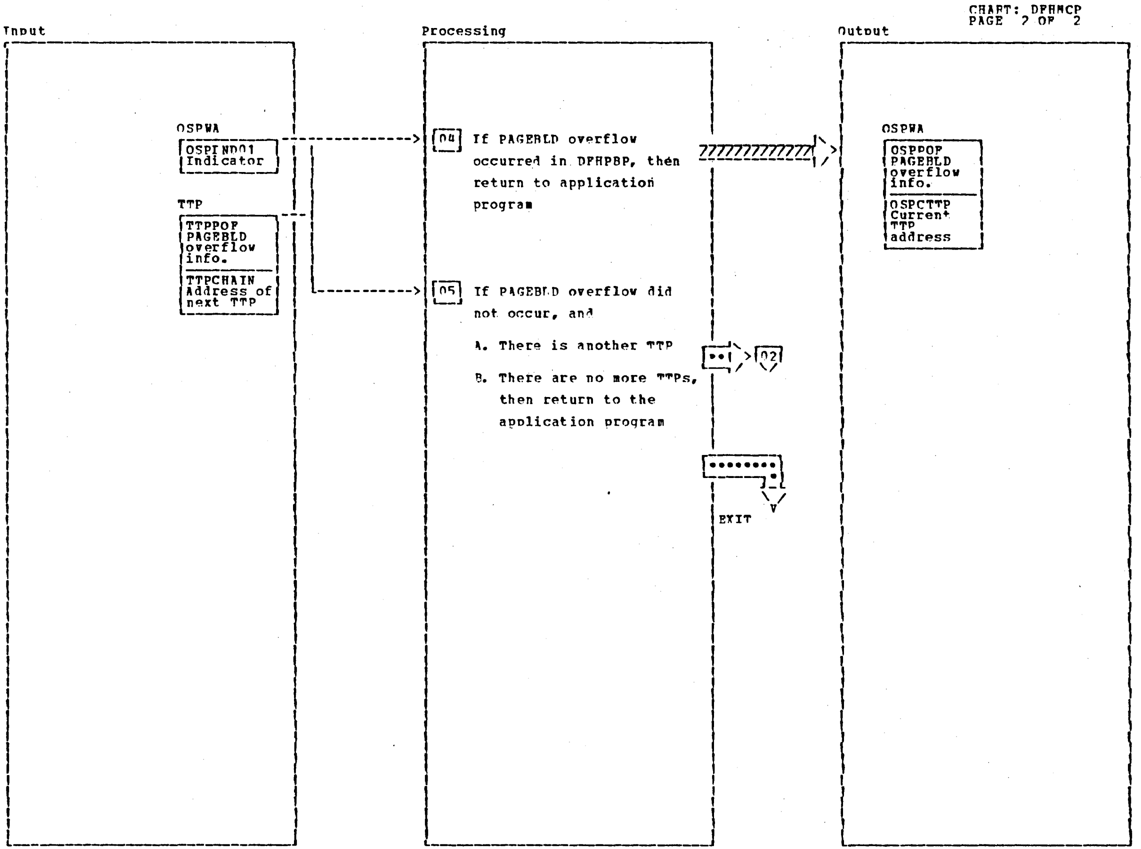


CHART: DFHMCP
PAGE 2 OF 2
Input
Processing
Output
OSPWA
OSPINDO1 Indicator
TTP
TTPPOF PAGEBLD overflow info.
TTPCHAIN Address of next TTP
OSPWA
OSPPOF PAGEBLD overflow info.
OSPCTTP Current TTP address
02
EXIT

04 If PAGEBLD overflow occurred in DFHPBP, then return to application program

05 If PAGEBLD overflow did not occur, and

A. There is another TTP

B. There are no more TTPs, then return to the application program

BMS Control: TYPE=PAGEBLD or TEXTBLD or ROUTE Processing

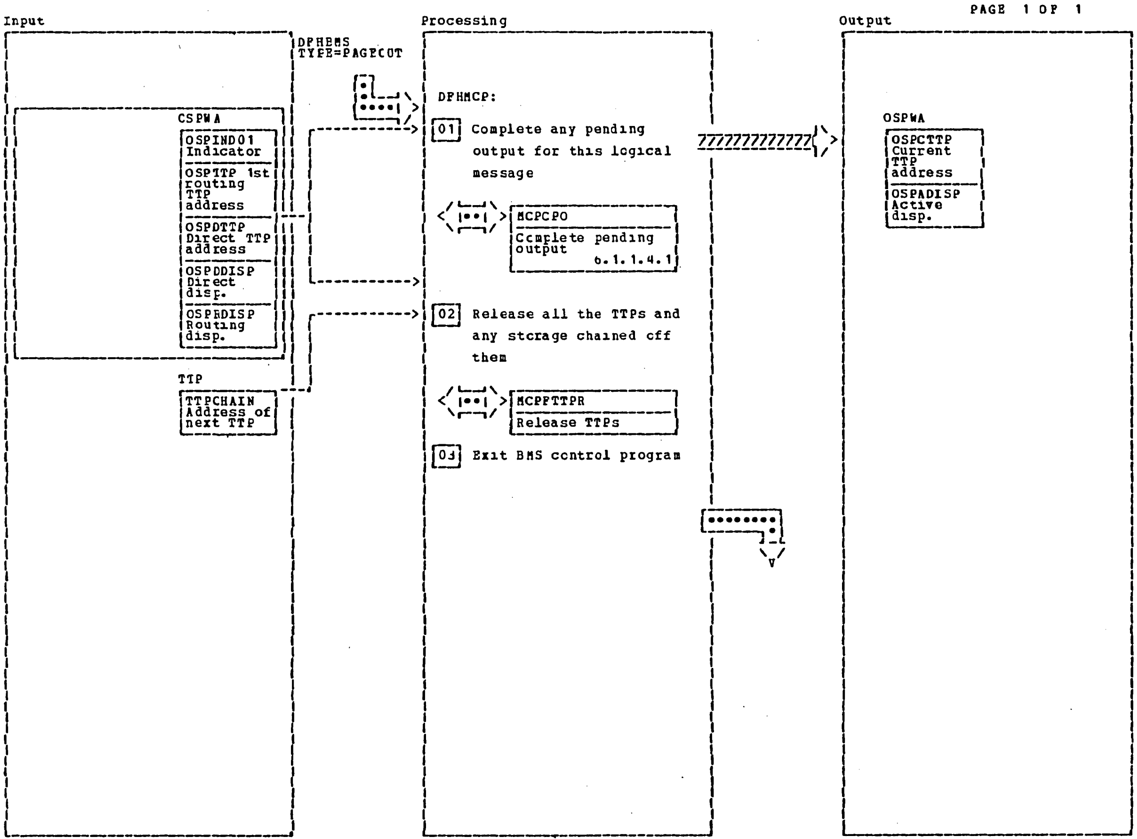
Diagram - 6.1.1.3-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 Return control to the application program immediately following the macro expansion, except when OFLOW was specified. In that case control is given to the application program at the location specified via the OFLOW operand. | | | | | | | |

BMS Control: TYPE=PAGEBLD or TEXTBLD or ROUTE Processing

Diagram - 6.1.1.3-02

PAGE 1 OF 1



BMS Control: TYPE=PAGEOUT Processing

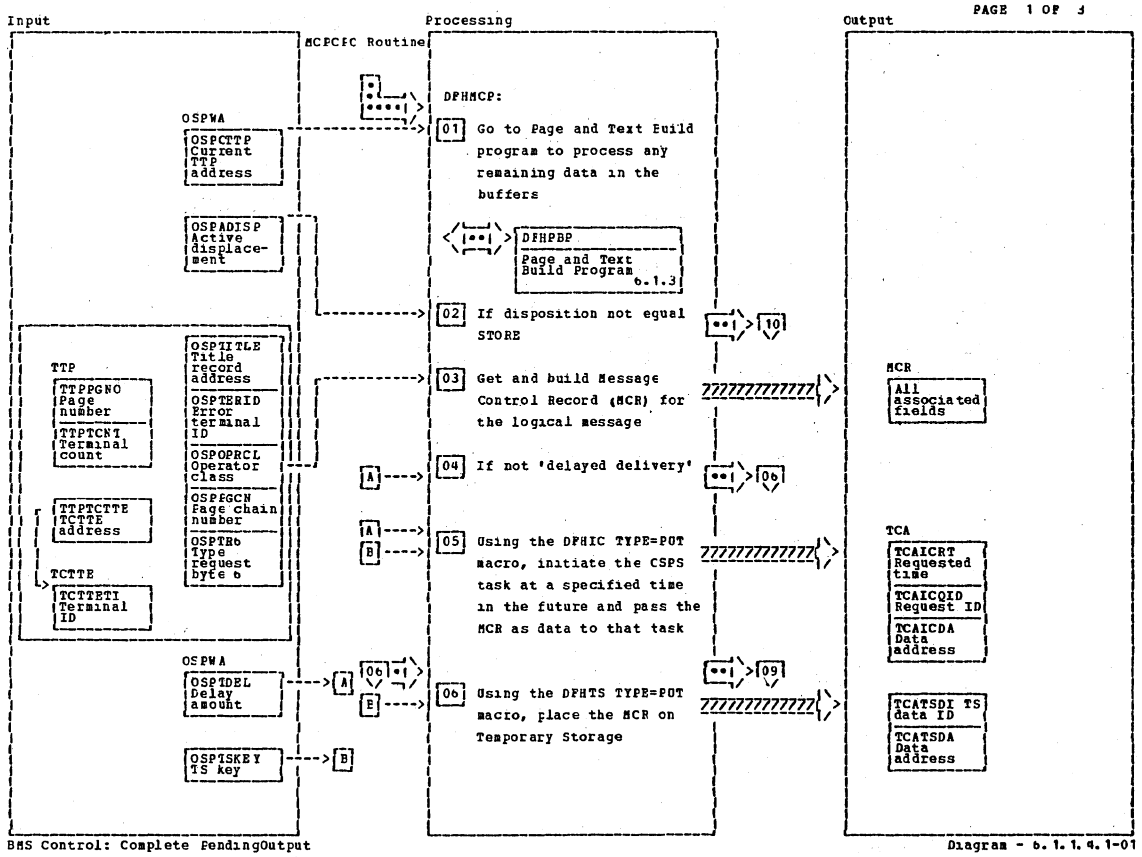
Diagram - 6.1.1.4-01

| Notes | Rcutine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 Practically all the PAGEOUT processing takes place in the MCPCPO subroutine. The mainline sets up pcinters for MCPCPO. | | MCPPGOUT | | | | | |
| 02 The storage chained off the TTP is the page buffer, map copy, and map TIOA. | | MCPTTPFR | | | | | |
| 03 Free areas chained off the OSPWA like the title record save area and the returned page list. Re-initialize OSPWA. | | MCPEXIT | | | | | |

BMS Control: TYPE=PAGEOUT Processing

Diagram - 6.1.1.4-01

PAGE 1 OF 3



DFHMCP:

01 Go to Page and Text Build program to process any remaining data in the buffers

02 If disposition not equal STORE

03 Get and build Message Control Record (MCR) for the logical message

04 If not 'delayed delivery'

05 Using the DFHIC TYPE=PUT macro, initiate the CSPS task at a specified time in the future and pass the MCR as data to that task

06 Using the DFHTS TYPE=PUT macro, place the MCR on Temporary Storage

BMS Control: Complete PendingOutput

Diagram - 6.1.1.4.1-01

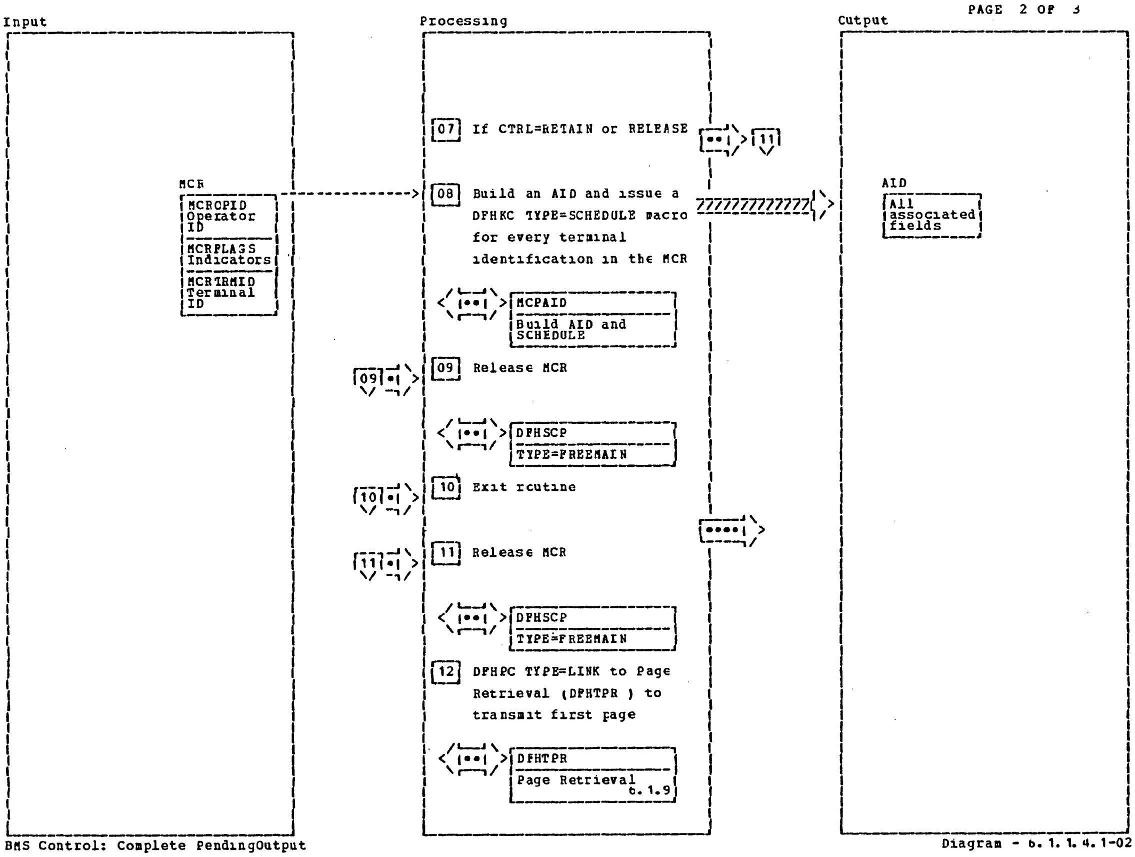
| Notes | Rottine | Label | Ref |
|---|---|---|---|
| 01 Any partially built pages must be completed according to OUT, STORE, or RETURN. | | MCPPAGNG | |
| 02 Disposition equal to STORE indicates that the pages have been put on Temporary Storage. This requires the building of a Message Control Record (MCR) and the initiation of a task (CSPS) to later retrieve these pages from Temporary Storage or LINK to a program (DFHTPR) to retrieve these pages now. | | | |
| 03 Included in this MCR is a list of terminal identifications that are to receive the logical message. | | MCPNODDS | |
| 04 | | MCPTSTRT | |
| 05 The CSPS task uses the program DFHTPS which builds an AID and issues a DFHKC TYPE=SCHEDULE macro for every terminal identification in the MCR. | | | 6.1.10 |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 06 | | MCPTSPUT | |

BMS Control: Complete PendingOutput

Diagram - 6.1.1.4.1-01

PAGE 2 OF 3



07 If CTRL=RETAIN or RELEASE

08 Build an AID and issue a DFHKC TYPE=SCHEDULE macro for every terminal identification in the MCR

09 Release MCR

10 Exit routine

11 Release MCR

12 DFHPC TYPE=LINK to Page Retrieval (DFHTPR ) to transmit first page

BMS Control: Complete PendingOutput

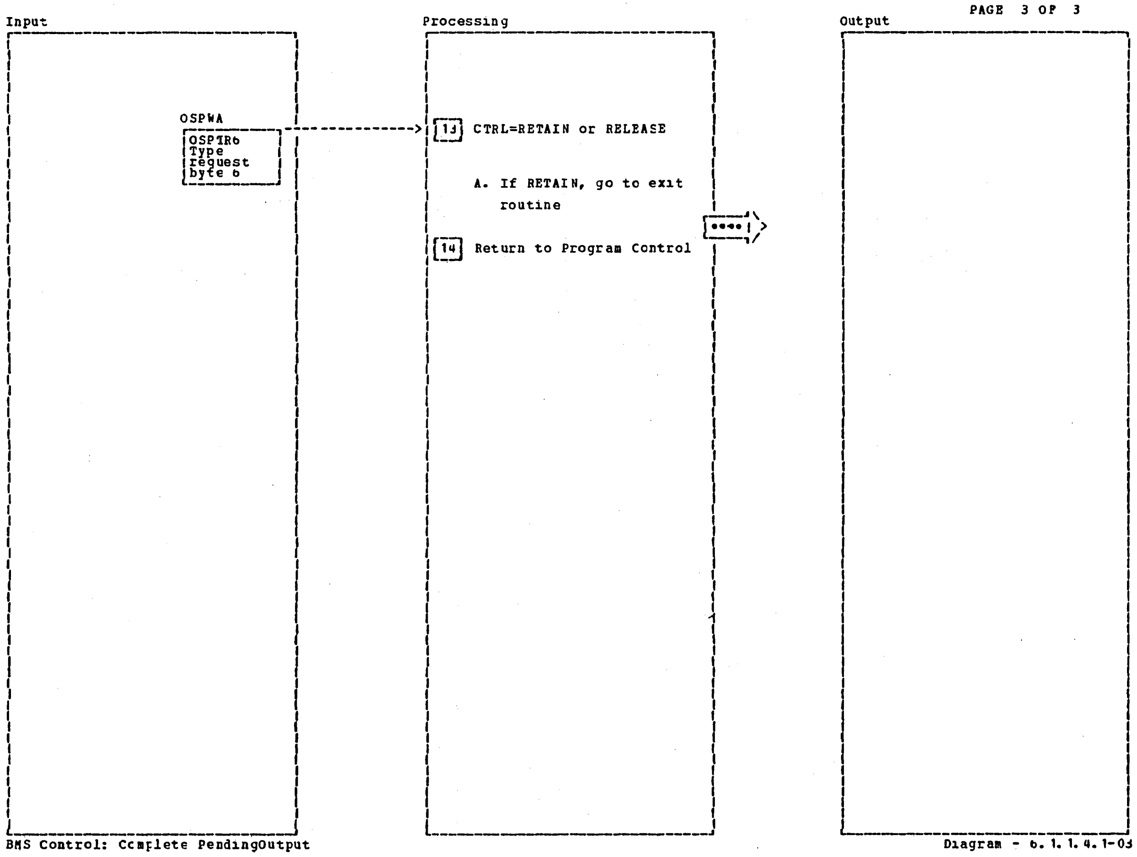
Diagram - 6.1.1.4.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 07 CTRL=RETAIN or RELEASE indicates that the user desires the first page of output to be transmitted to the terminal (DFHTPR) and control returned according to RETAIN or RELEASE. If neither RETAIN nor RELEASE is specified, other tasks are scheduled to transmit the pages and control is returned to the application program. | | | | | | | |
| 08 The address of the AID is passed to Task Control in TCAKCPA. | | | | | | | |
| 09 | | MCPFRMCR | | | | | |
| 10 Prior to exit, a check is made for another TTP. If there is one, pointers are reset and control repeats at step 01. If not another TTP, exit routine. | | MCPCKMOR | | | | | |
| 11 | | MCPMCRFR | | | | | |

BMS Control: Complete PendingOutput

Diagram - 6.1.1.4.1-02

PAGE 3 OF 3



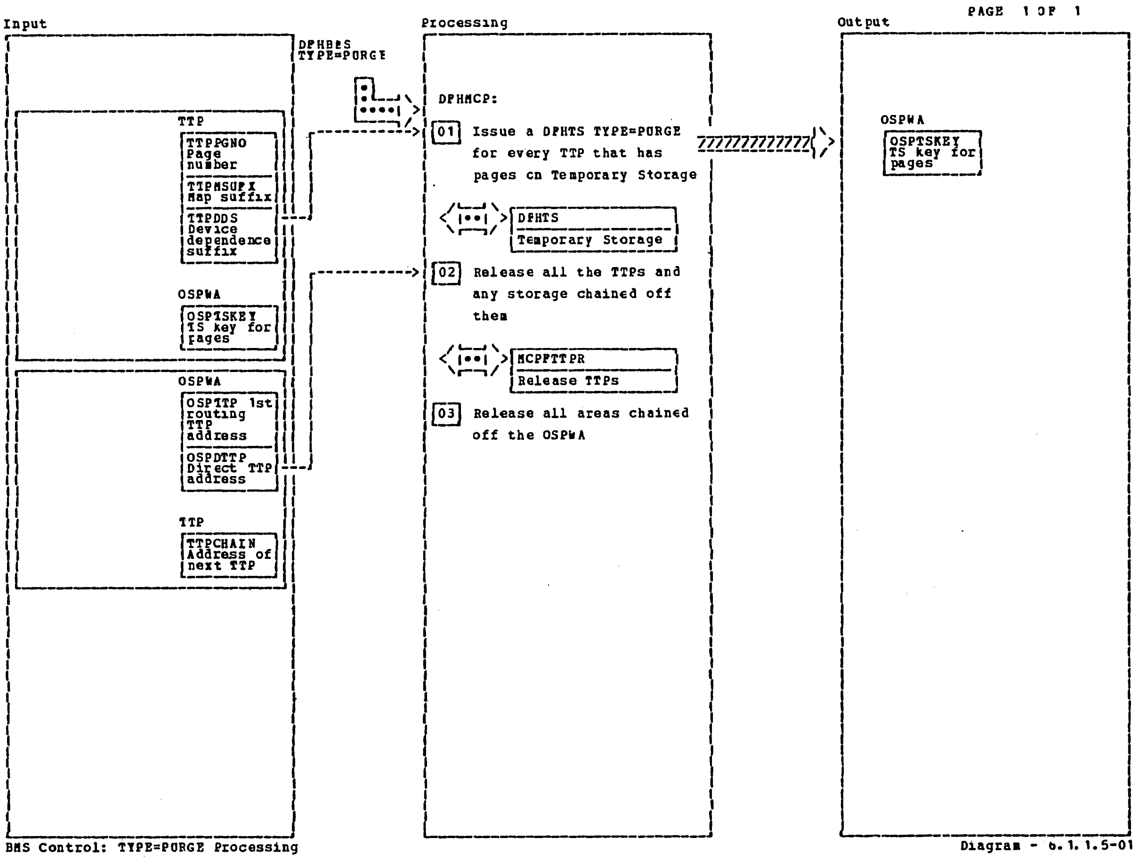
BMS Control: Ccmplete PendingOutput

Diagram - 6.1.1.4.1-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 14 Return via a DFHPC TYPE=RETURN | | | | | | | |

BMS Control: Ccmplete PendingOutput

Diagram - 6.1.1.4.1-03



DFHMCP:

01 Issue a DFHTS TYPE=PURGE for every TTP that has pages cn Temporary Storage

02 Release all the TTPs and any storage chained off them

03 Release all areas chained off the OSPWA

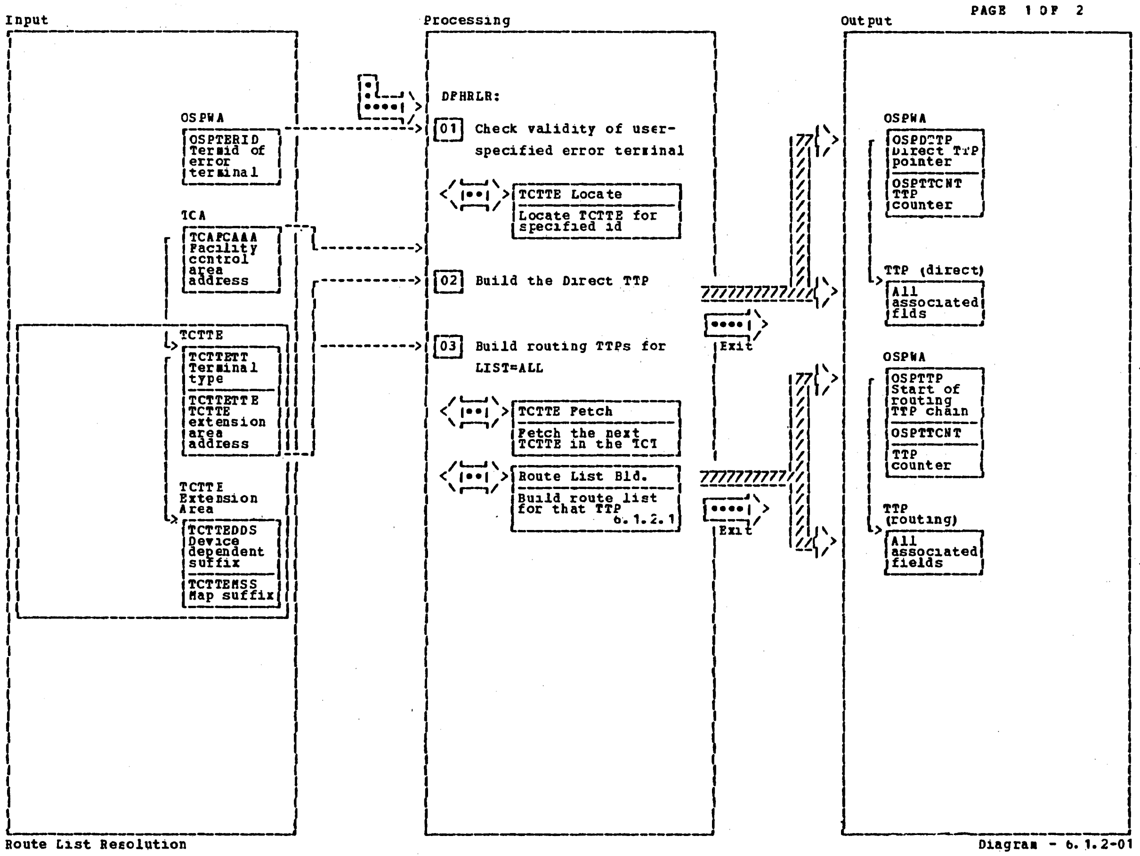
BMS Control: TYPE=PURGE Processing

Diagram - 6.1.1.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | MCPPURGE | | | | | |
| 02 | | MCPGOTTP | | | | | |
| 03 These areas include the title record save area and the returned page list. | | MCPMISC | | | | | |

BMS Control: TYPE=PURGE Processing

Diagram - 6.1.1.5-01

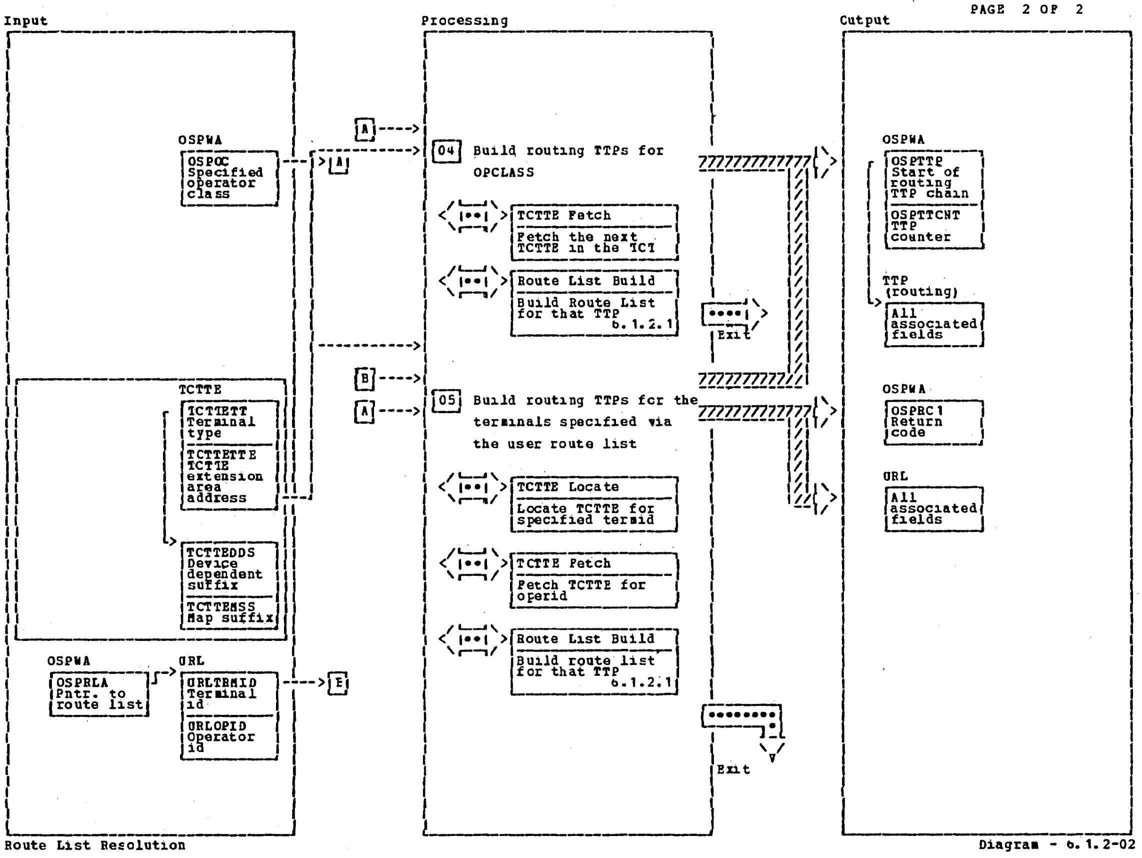

PAGE 1 OF 2
Input
Processing
Output
OSPWA
OSPTERID Termid of error terminal
TCA
TCAFCAAA Facility control area address
TCTTE
TCTTETT Terminal type
TCTTETTE TCTTE extension area address
TCTTE Extension Area
TCTTEDDS Device dependent suffix
TCTTEMSS Map suffix
DFHRLR:
01 Check validity of user-specified error terminal
TCTTE Locate
Locate TCTTE for specified id
02 Build the Direct TTP
03 Build routing TTPs for LIST=ALL
TCTTE Fetch
Fetch the next TCTTE in the TCT
Route List Bld.
Build route list for that TTP 6.1.2.1
Exit
Exit
OSPWA
OSPDTTP Direct TTP pointer
OSPTTCNT TTP counter
TTP (direct)
All associated flds
OSPWA
OSPTTP Start of routing TTP chain
OSPTTCNT TTP counter
TTP (routing)
All associated fields

Route List Resolution Diagram - 6.1.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 After processing here, control is passed to one of the following: Step 2, Step 3, Step 4, or Step 5. | | | | | | | |
| 02 | | RLRONLY | | | | | |
| 03 The TCTTE FETCH routine will return the address of the next TCTTE for a terminal that is supported by BMS. | | RLRALL | | | | | |

Route List Resolution Diagram - 6.1.2-01

PAGE 2 OF 2


Input
Processing
Output
OSPWA
OSPOC Specified operator class
A
04 Build routing TTPs for OPCLASS
TCTTE Fetch
Fetch the next TCTTE in the TCT
Route List Build
Build Route List for that TTP 6.1.2.1
Exit
B
05 Build routing TTPs for the terminals specified via the user route list
TCTTE Locate
Locate TCTTE for specified termid
TCTTE Fetch
Fetch TCTTE for operid
Route List Build
Build route list for that TTP 6.1.2.1
Exit
TCTTE
TCTTETT Terminal type
TCTTETTE TCTTE extension area address
TCTTEDDS Device dependent suffix
TCTTEMSS Map suffix
OSPWA
OSPRLA Pntr. to route list
URL
URLTRMID Terminal id
URLOPID Operator id
E
OSPWA
OSPTTP Start of routing TTP chain
OSPTTCNT TTP counter
TTP (routing)
All associated fields
OSPWA
OSPRC1 Return code
URL
All associated fields

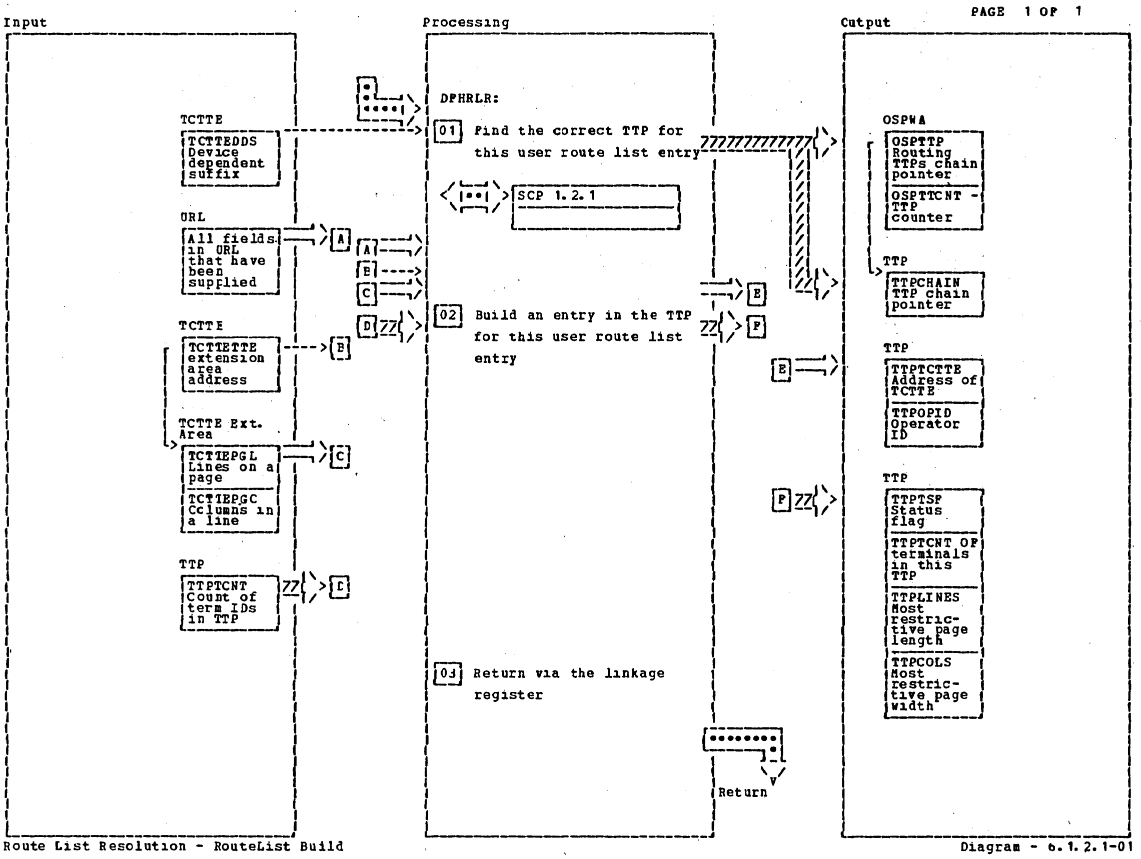
Route List Resolution

Diagram - 6.1.2-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 The TCTTE FETCH routine will return the address of the next TCTTE for a terminal that is supported by BMS. | | RLBOPCL | | | | | |
| 05 If URLTRMID is specified in the URL entry, the TCTTE LOCATE is executed. If URLTRMID is not specified, the TCTTE FETCH is executed. | | RLBLIST | | | | | |

Route List Resolution

Diagram - 6.1.2-02


PAGE 1 OF 1
Input
Processing
Cutput
TCTTE
TCTTEDDS Device dependent suffix
URL
All fields in URL that have been supplied
TCTTE
TCTTETTE extension area address
TCTTE Ext. Area
TCTTEPGL Lines on a page
TCTTEPGC Columns in a line
TTP
TTPTCNT Count of term IDs in TTP
A
B
C
D
E
F
DFHRLR:
01 Find the correct TTP for this user route list entry
SCP 1.2.1
02 Build an entry in the TTP for this user route list entry
03 Return via the linkage register
Return
OSPWA
OSPTTP Routing TTPs chain pointer
OSPTTCNT - TTP counter
TTP
TTPCHAIN TTP chain pointer
TTP
TTPTCTTE Address of TCTTE
TTPOPID Operator ID
TTP
TTPTSF Status flag
TTPTCNT OF terminals in this TTP
TTPLINES Most restrictive page length
TTPCOLS Most restrictive page width

Route List Resolution - RouteList Build

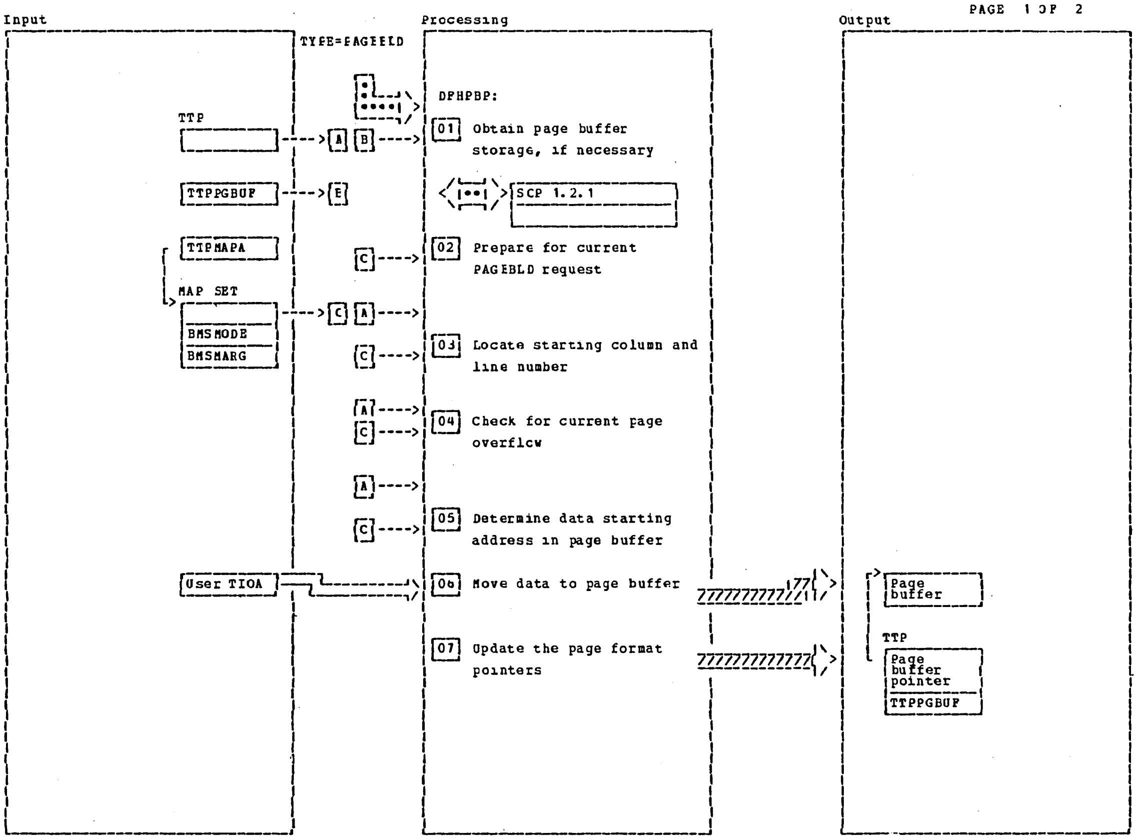
Diagram - 6.1.2.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If there are no TTPs or cannot find the one needed, issue a DFHSC TYPE=GETMAIN for another TTP. | | RLRRLBLD | | | | | |
| 02 For every URL entry, a TTP route list entry is built in the proper TTP. | | RLRBLDL | | | | | |

Route List Resolution - RouteList Build

Diagram - 6.1.2.1-01

PAGE 1 OF 2



01 Obtain page buffer storage, if necessary

02 Prepare for current PAGEBLD request

03 Locate starting column and line number

04 Check for current page overflow

05 Determine data starting address in page buffer

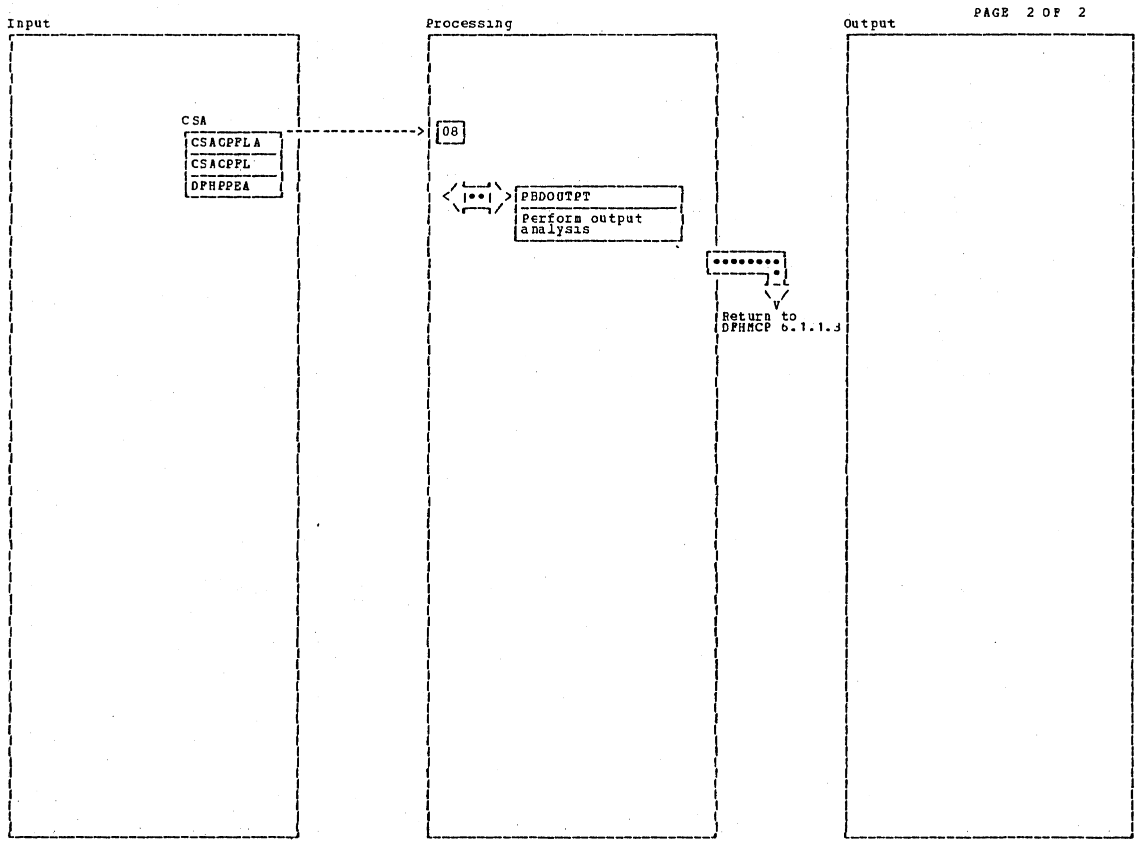
06 Move data to page buffer

07 Update the page format pointers

DFHPBP: PAGEBLD, Non-3270 Output Mapping

Diagram - 6.1.3.1-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 On the first request for a page, a buffer is obtained and formatted with blanks and device dependent characters. | | PBD00050 and PBD00054 | |
| If a buffer exists from a previous request, a new buffer is not obtained. | | PBD00017 | |
| 02 If JUSTIFY=FIRST was specified, TYPE=PAGEOUT processing is simulated before continuing processing. | | | |
| 04 If the map will not fit on the current page, the overflow indicator is set and control is returned to DFHMCP. | | | |
| 05 If DATA=BLOCK was specified and if no field specifications were provided in the map set, then the data is positioned according to column and line. | | PBD01130 | |
| For FIELD data, the data is positioned according to the field specifications provided by the user in the map set. | | PBD01140 | |



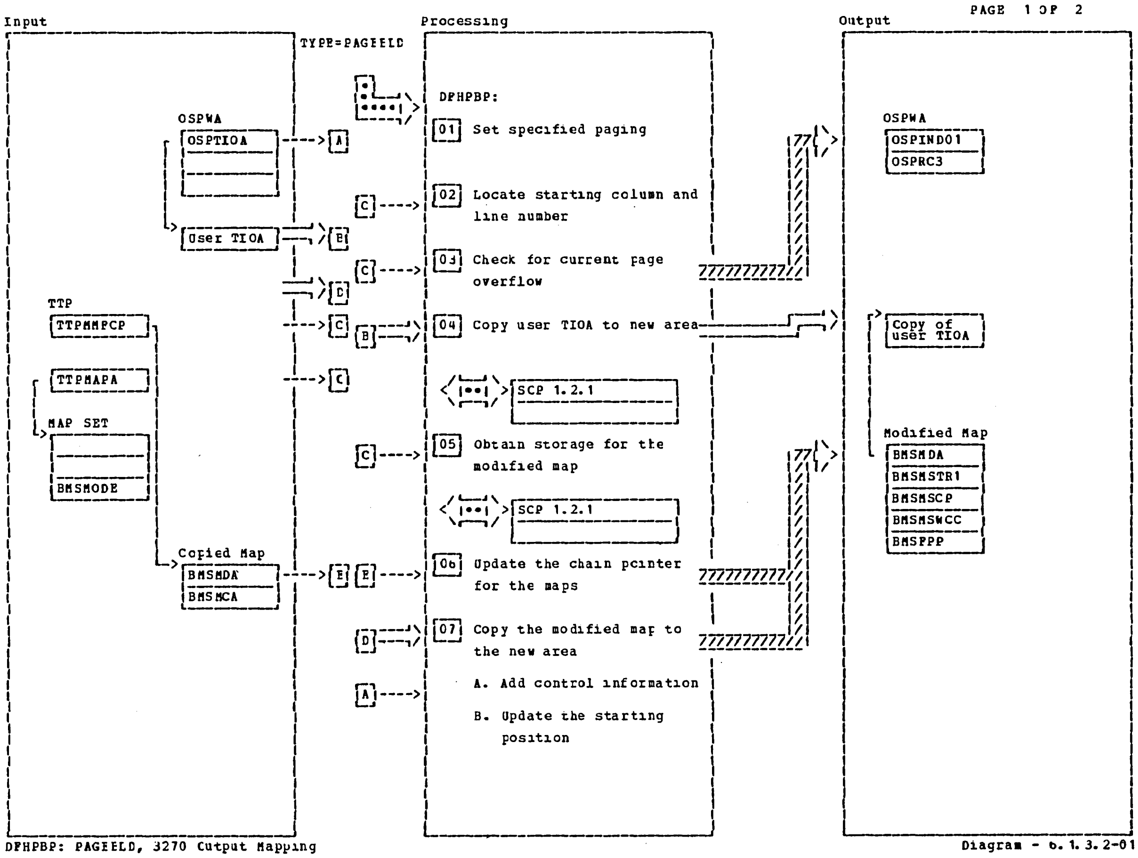
DFHPBP: PAGEBLD, Non-3270 Output Mapping

Diagram - 6.1.3.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 08 If JUSTIFY=LAST was specified, TYPE=PAGEOUT processing is simulated. | | PBD01300 | | | | | |

DFHPBP: PAGEBLD, Non-3270 Output Mapping

Diagram - 6.1.3.1-02


PAGE 1 OF 2
Input
Processing
Output
TYPE=PAGEBLD
OSPWA
OSPTIOA
User TIOA
TTP
TTPMMPCP
TTPMAPA
MAP SET
BMSMODE
Copied Map
BMSMDA
BMSMCA
A
B
C
D
E
DFHPBP:
01 Set specified paging
02 Locate starting column and line number
03 Check for current page overflow
04 Copy user TIOA to new area
SCP 1.2.1
05 Obtain storage for the modified map
SCP 1.2.1
06 Update the chain pointer for the maps
07 Copy the modified map to the new area
A. Add control information
B. Update the starting position
OSPWA
OSPINDO1
OSPRC3
Copy of user TIOA
Modified Map
BMSMDA
BMSMSTR1
BMSMSCP
BMSMSWCC
BMSPPP

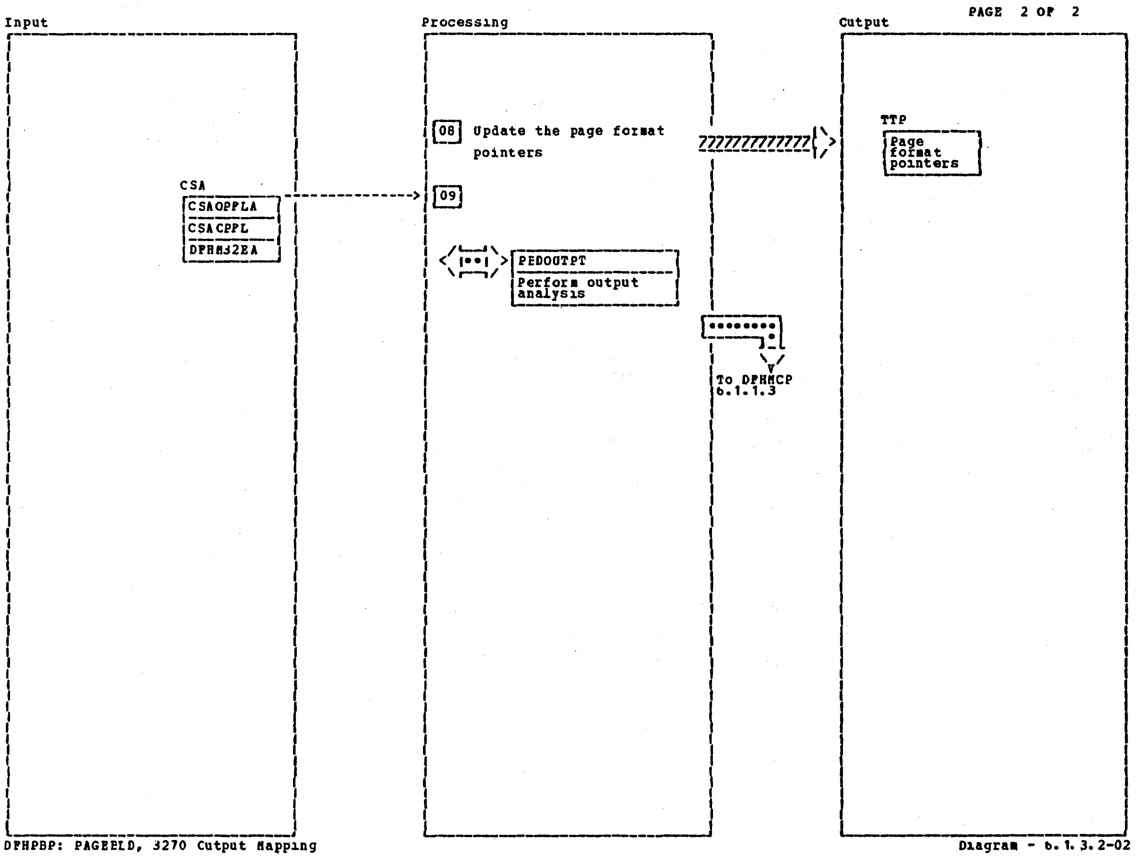
DFHPBP: PAGEELD, 3270 Cutput Mapping

Diagram - 6.1.3.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The user specifies the required paging. If JUSTIFY=FIRST was specified, TYPE=PAGEOUT processing is simulated before continuing processing. | | PBD00100 | | | | | |
| 02 | | PBD01000 | | | | | |
| 03 If the map will not fit on the current page, the overflow indicator is set and control is returned to DFHMCP. | | PBD01120 | 6.1.3 | | | | |
| 06 The chain of previous maps is traced to update the pointer. | | | | | | | |
| 07 | | | | | | | |
| B. The field specifications are used to calculate the starting position. | | | | | | | |

DFHPBP: PAGEBLD, 3270 Output Mapping

Diagram - 6.1.3.2-01



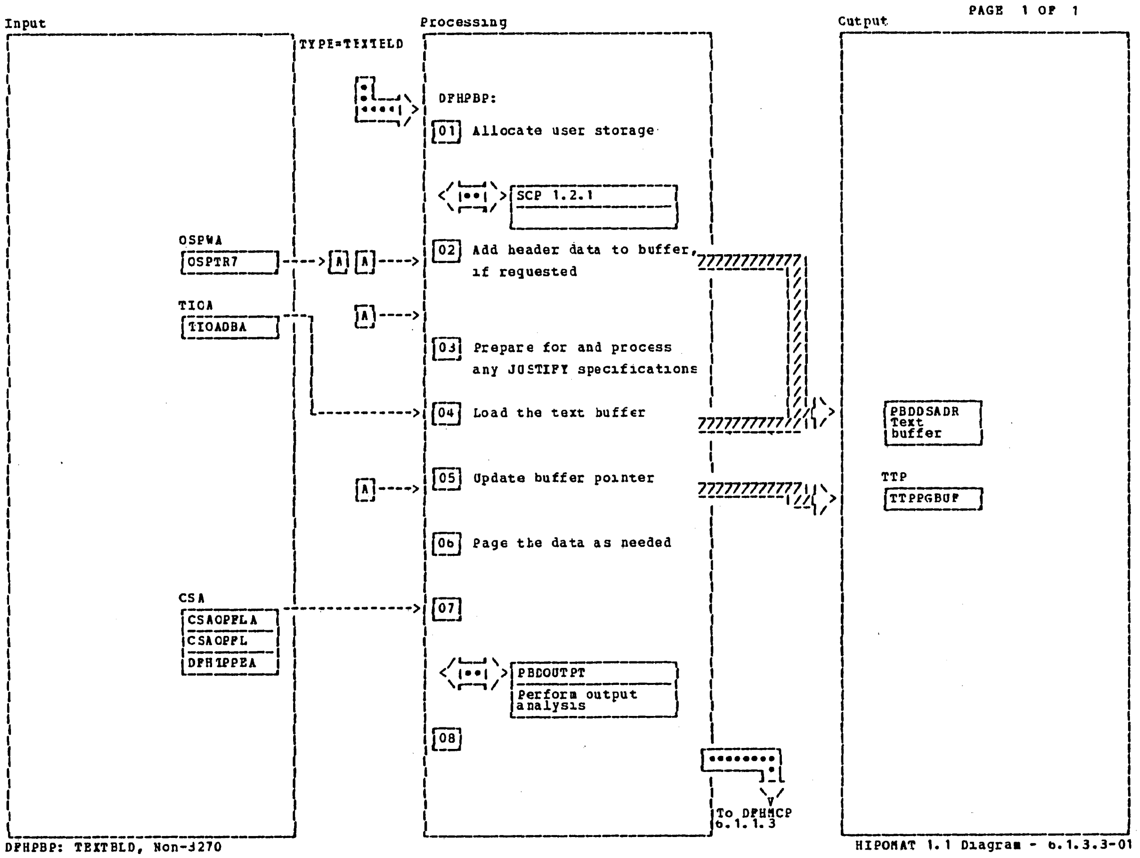
DFHPBP: PAGEBLD, 3270 Output Mapping

Diagram - 6.1.3.2-02

| Notes | Rottine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 09 If the page is full, PBDOUTPT links to DFHM32 to output the page before returning control to DFHMCP. | | | 6.1.5.2 | | | | |

DFHPBP: PAGEBLD, 3270 Output Mapping

Diagram - 6.1.3.2-02



DFHPBP: TEXTBLD, Non-3270

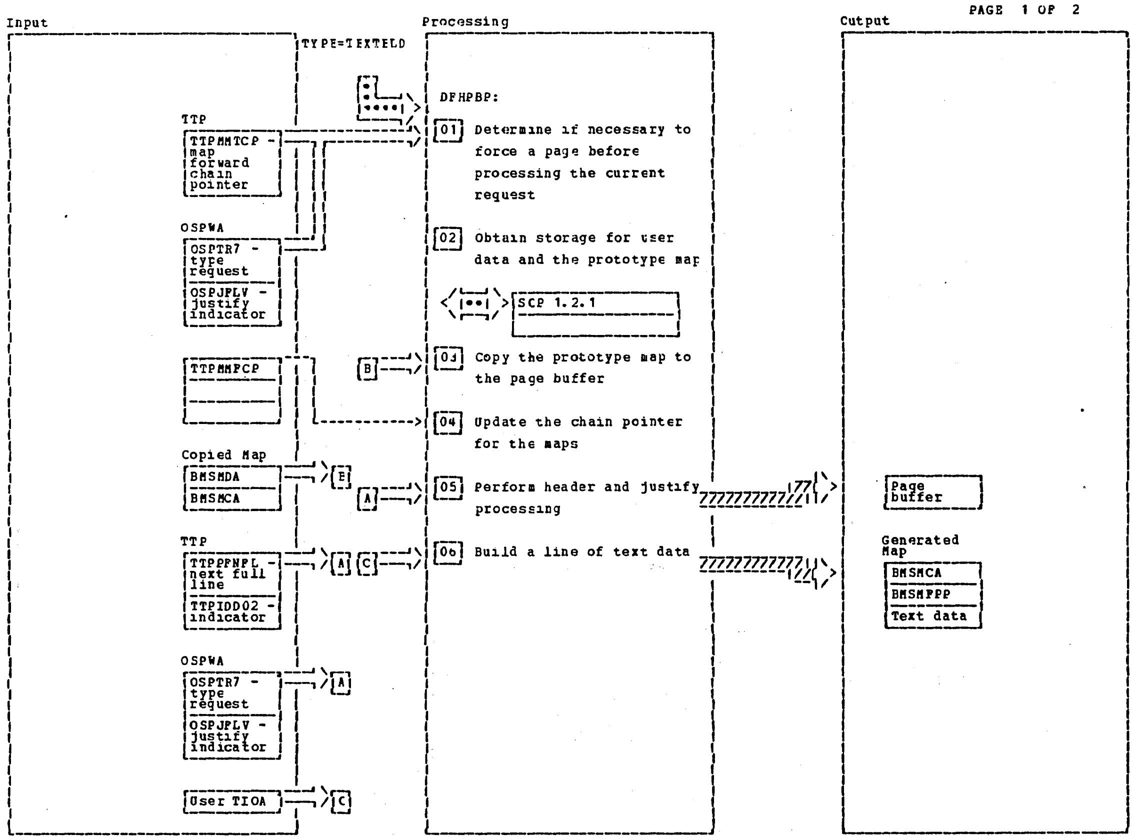
HIPOMAT 1.1 Diagram - 6.1.3.3-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 For first requests, a page buffer is allocated. | | PBD00017 | |
| For subsequent requests, the same page buffer is used. | | | |
| For TYPE=OUT requests, only enough space for user data is allocated. | | | |
| For TYPE=OUT,EDIT requests, a complete page buffer is not obtained and TTPPGBUF is not updated. | | | |
| 03 | | PBD20002 | |
| 06 If the data will not fit on the page, trailer data is added to the page, the page is forced out, and a new page is used. | | | |
| 07 For incomplete pages, PBDOUTPT returns control to the Mapping Control program (DFHMCP). | | PBD015RT | |
| For complete pages, PBDOUTPT links DFHDSB for terminal dependent processing. | | PBD01510 | 6.1.7 |
| 08 Return control to DFHMCP when the user data is exhausted, else repeat from step 4. | | | |

DFHPBP: TEXTBLD, Non-3270

Diagram - 6.1.3.3-01

PAGE 1 OF 2


Input
Processing
Cutput
TYPE=TEXTBLD
DFHPBP:
TTP
TTPMMTCP - map forward chain pointer
OSPWA
OSPTR7 - type request
OSPJFLV - justify indicator
TTPMMFCP
Copied Map
BMSMDA
BMSMCA
TTP
TTPPFNFL - next full line
TTPIDD02 - indicator
OSPWA
OSPTR7 - type request
OSPJFLV - justify indicator
User TIOA
B
E
A
A
C
A
C
01 Determine if necessary to force a page before processing the current request
02 Obtain storage for user data and the prototype map
SCP 1.2.1
03 Copy the prototype map to the page buffer
04 Update the chain pointer for the maps
05 Perform header and justify processing
06 Build a line of text data
Page buffer
Generated Map
BMSMCA
BMSMFPP
Text data

DFHPBP: TEXTBLC, 3270

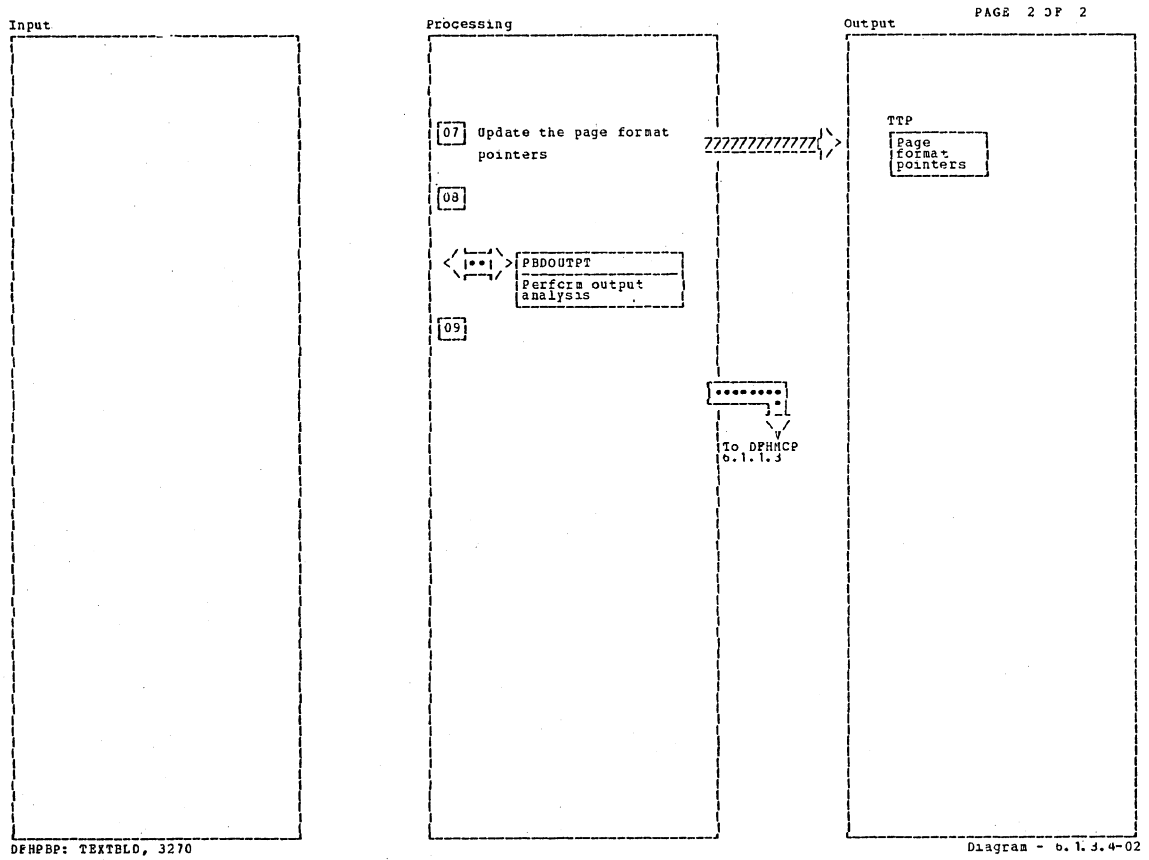
Diagram - 6.1.3.4-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 | | PBD00015 | |
| 02 The prototype map is generated in this prcgram.<br>Storage required = D+H+T+M Where:<br>D is the length of the user data,<br>H is the length of the header data,<br>T is the length of the trailer data,<br>M is the maximum map size. | | PBDMOPPO | |
| 03 The map definition and one field specification are moved to the acquired storage area. | 3270 Text build initial setup | PBD23000 | |
| 05 | | PBD20002 | |
| 06 As each line of text is built, the map field specifications are updated. | 327C Text Fuild, Map Build | PBD24000 | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| | | | |

DFHPBP: TEXTBLD, 3270

Diagram - 6.1.3.4-01



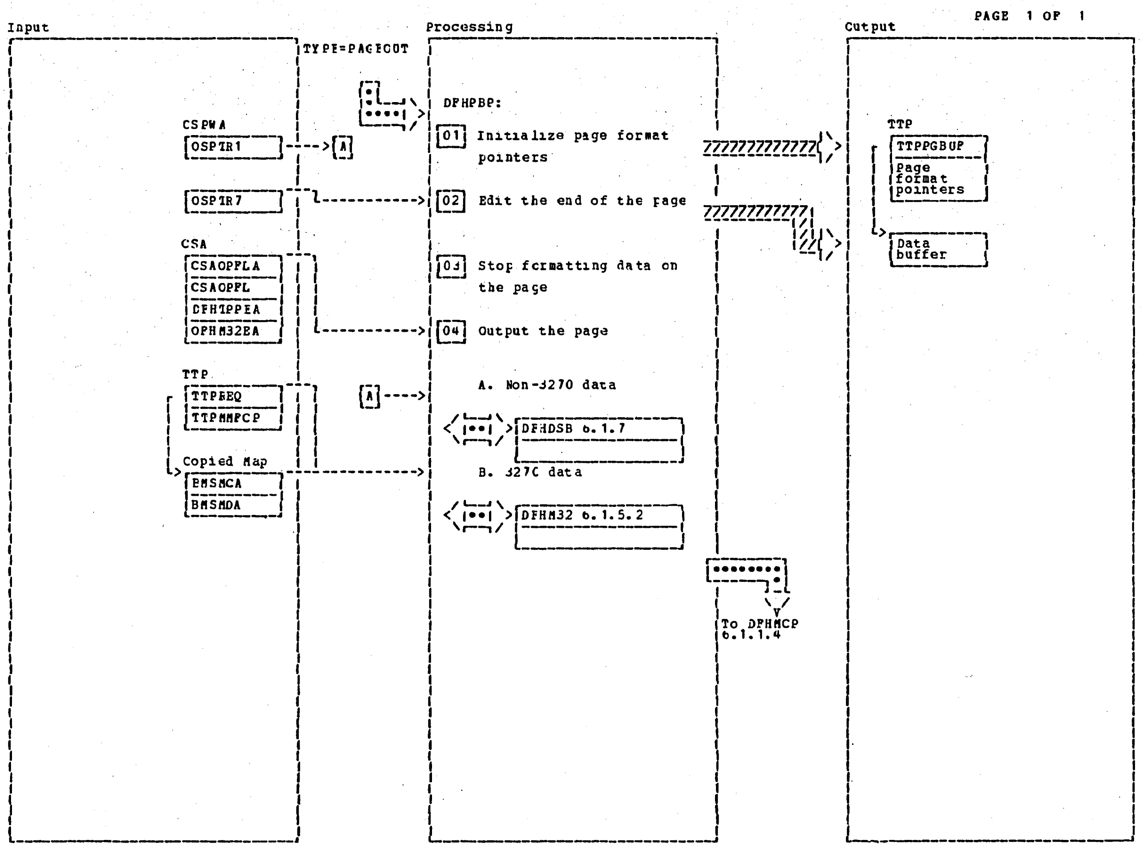
DFHPBP: TEXTBLD, 3270

Diagram - 6.1.3.4-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 08 For page overflow and full page conditions, control passes to DFHM32 for output. | | PBD01520 | 6.1.5.2 | | | | |
| 09 Return ccntrcl to DFHMCP when data is exhausted, else repeat from step 6. | | | | | | | |

DFHPBP: TEXTBLD, 3270

Diagram - 6.1.3.4-02



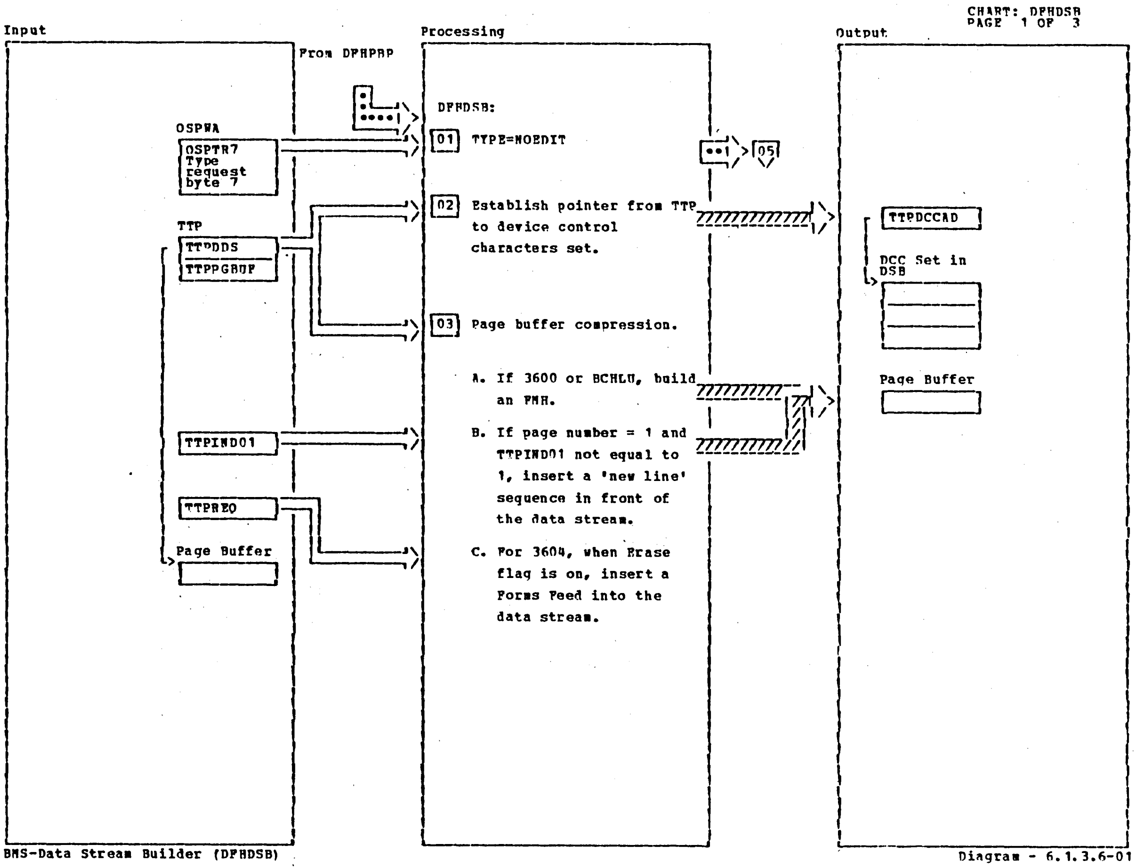
PAGEOUT: Terminate Output Request

Diagram - 6.1.3.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The page format pointers are reset to their initial values. | | PBD16610 | | | | | |
| 02 Device-dependent characters are added to the end of the page if the EDIT parameter was specified. | | | | | | | |
| Trailer data is added to the end of the page if TRAILER=YES was specified. | | PBDTRPJT | | | | | |
| 03 Device-dependent characters are added to the buffer to cause any lines remaining on the page to be skipped. | | PBD1NLP6 | | | | | |
| 04 | | PBDOUTPT | | | | | |

PAGEOUT: Terminate Output Request

Diagram - 6.1.3.5-01



DFHDSB:

01 TYPE=NOEDIT

02 Establish pointer from TTP to device control characters set.

03 Page buffer compression.

A. If 3600 or BCHLU, build an FMH.

B. If page number = 1 and TTPINDO1 not equal to 1, insert a 'new line' sequence in front of the data stream.

C. For 3604, when Erase flag is on, insert a Forms Feed into the data stream.

BMS-Data Stream Builder (DFHDSB)

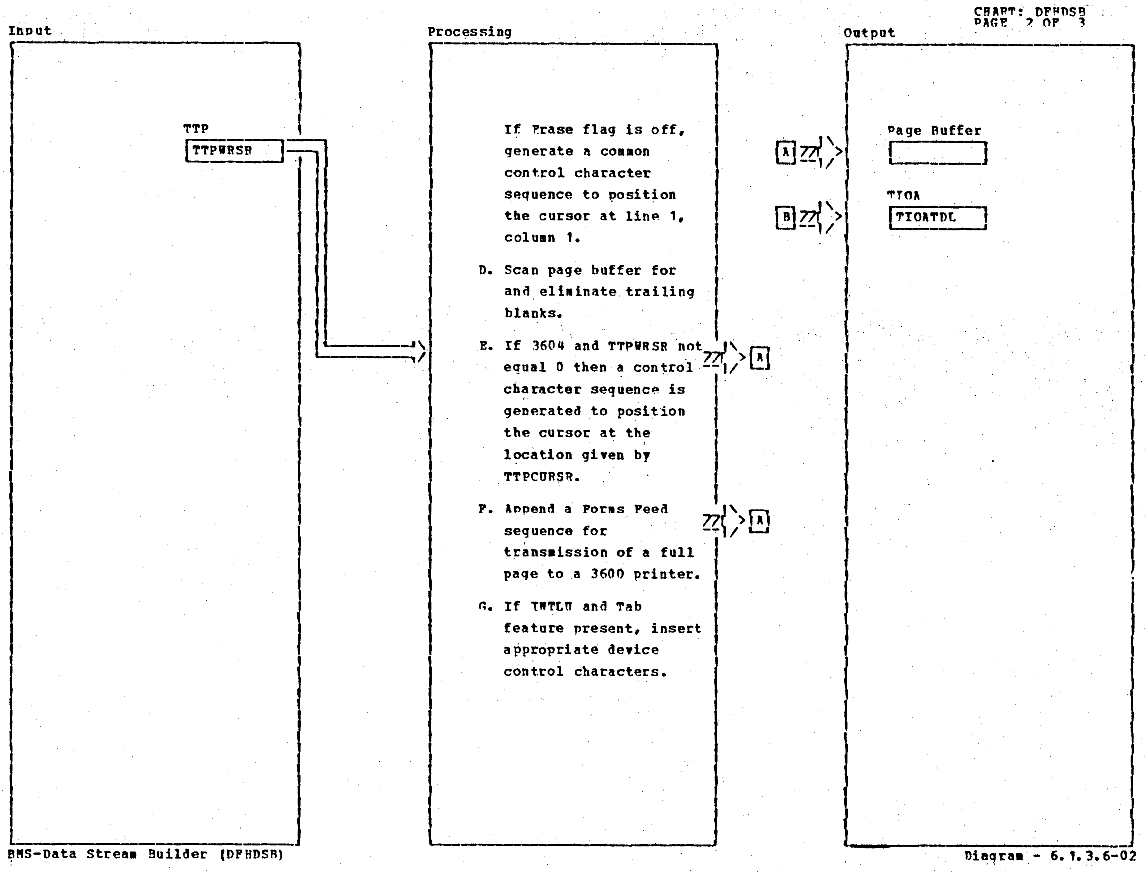
Diagram - 6.1.3.6-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 If the Data Stream Build program is entered with TYPE=NOEDIT, no data stream building is performed. | | | |
| 02 Using the given device dependent suffix in the TTP, the address of the accompanying device control characters set is stored in the TTPDCCAD field of the TTP. | DSBCC00 | | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

BMS-Data Stream Builder (DFHDSB)

Diagram - 6.1.3.6-01



If Erase flag is off, generate a common control character sequence to position the cursor at line 1, column 1.

D. Scan page buffer for and eliminate trailing blanks.

E. If 3604 and TTPWRSR not equal 0 then a control character sequence is generated to position the cursor at the location given by TTPCURSR.

F. Append a Forms Feed sequence for transmission of a full page to a 3600 printer.

G. If TWTLU and Tab feature present, insert appropriate device control characters.

BMS-Data Stream Builder (DFHDSB)

Diagram - 6.1.3.6-02

| Notes | Routine | Label | Ref |
|---|---|---|---|

| Notes | Routine | Label | Ref |
|---|---|---|---|

BMS-Data Stream Builder (DFHDSB)

Diagram - 6.1.3.6-02

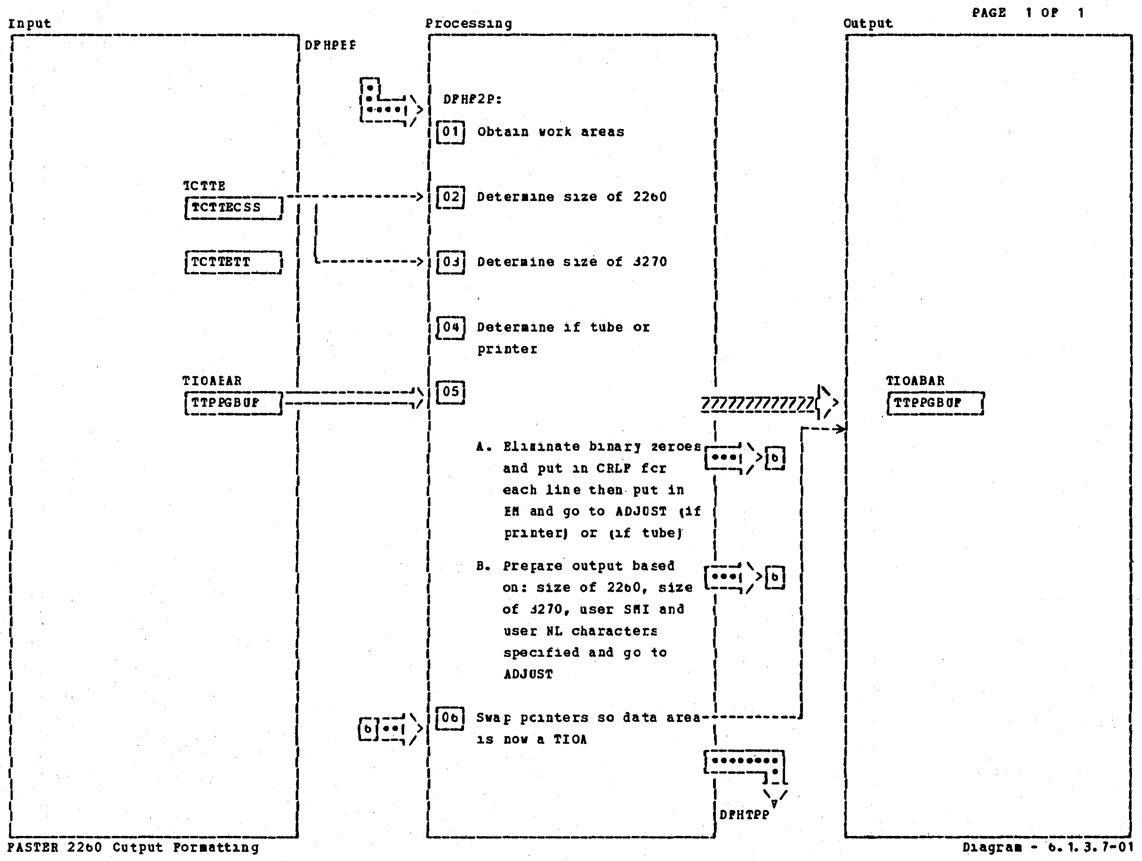
CHART: DFHDSB
PAGE 3 OF 3

| Input | Processing | Output |
|---|---|---|
| | H. If BCHLU, an IRS (Inter-Record Separator) is used instead of the NL character to separate records.<br>04 Provide total data length after compression of the page buffer. > B<br>05<br>< ••• > DFHTPP<br>06 Return to caller | |

BMS-Data Stream Builder (DFHDSB) Diagram - 6.1.3.6-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03<br>D. The page buffer is scanned line by line for trailing blanks. Trailing blanks are eliminated, thus reducing the effective length of the data stream to be transmitted. The logical new line character which terminates each line is replaced by its physical equivalents from the device control character set. The physical new line characters are inserted after the first nonblank character encountered when backscanning a particular line. The data from any of the following lines are moved after the preceding lines physical new line sequence. This buffer compression is continued until the stopper character is encountered. | | | | | | | |

BMS-Data Stream Builder (DFHDSB) Diagram - 6.1.3.6-03


PAGE 1 OF 1
Input
Processing
Output
DFHPEP
DFHF2P:
01 Obtain work areas
TCTTE
TCTTECSS
02 Determine size of 2260
TCTTETT
03 Determine size of 3270
04 Determine if tube or printer
TIOABAR
TTPPGBUF
05
A. Eliminate binary zeroes and put in CRLF for each line then put in EM and go to ADJUST (if printer) or (if tube)
6
B. Prepare output based on: size of 2260, size of 3270, user SMI and user NL characters specified and go to ADJUST
6
TIOABAR
TTPPGBUF
6
06 Swap pointers so data area is now a TIOA
DFHTPP

FASTER 2260 Output Formatting

Diagram - 6.1.3.7-01

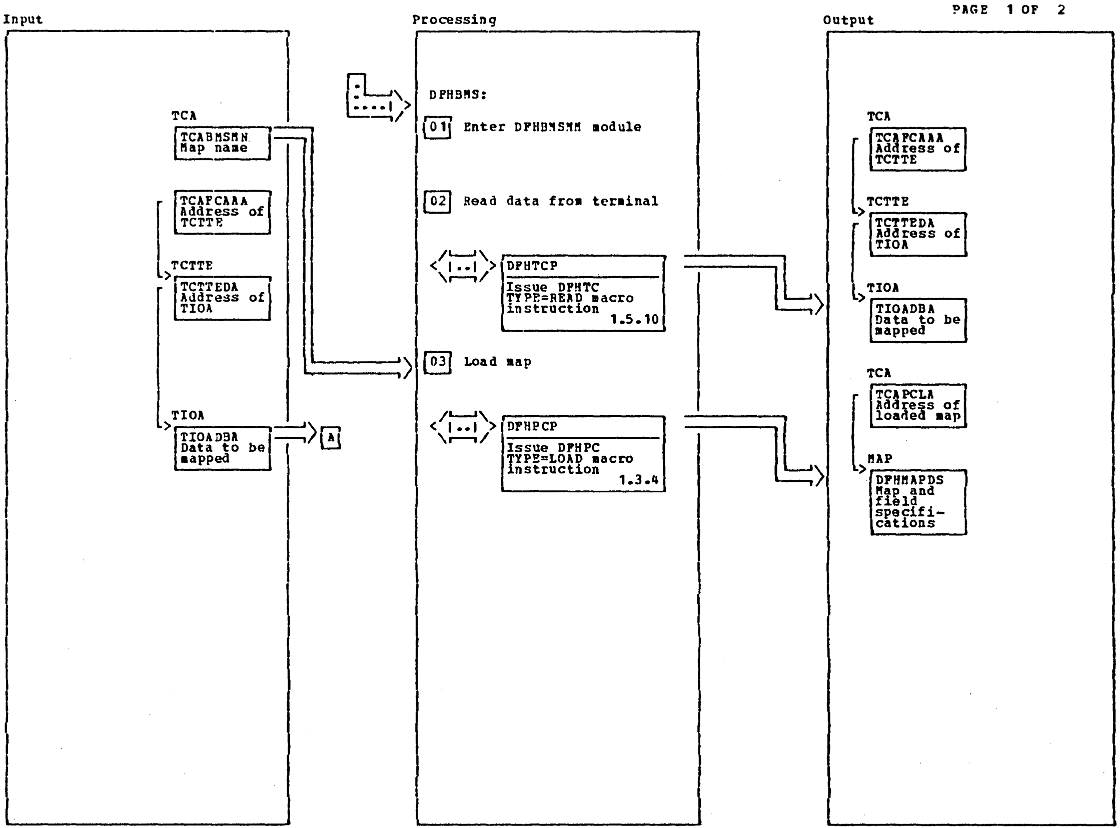
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 Issue GETMAIN (one for Terminal, one for user). To establish work buffers. | | DFH2PNA | |
| 02 Test for 2260 screen size (240,480,960). | | | |
| 03 Test for 3270 screen size (480, 1920). | | CK32705Z | |
| 04 Check if display or printer. | | TRMLOOP | |
| 05 | | | |
| A. If printer, eliminate binary zeroes and insert CRLF (carriage return / line feed) for each line, EM (end of message) on last line. | | PRINTER | |
| B. If display prepare output (either 40 characters per line or 80 characters per line and number of lines based on 2260 and/or 3270 sizes, inserting user coded SMI or new line sysmbols as required). | | TUBE | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 06 Put prepared output in TIOA, adjust pointers and return. | | ADJUST | |

FASTER 2260 Output Formatting

Diagram - 6.1.3.7-01

PAGE 1 OF 2


Input
Processing
Output
TCA
TCABMSMN Map name
TCAFCAAA Address of TCTTE
TCTTE
TCTTEDA Address of TIOA
TIOA
TIOADBA Data to be mapped
A
DFHBMS:
01 Enter DFHBMSMM module
02 Read data from terminal
DFHTCP
Issue DFHTC TYPE=READ macro instruction 1.5.10
03 Load map
DFHPCP
Issue DFHPC TYPE=LOAD macro instruction 1.3.4
TCA
TCAFCAAA Address of TCTTE
TCTTE
TCTTEDA Address of TIOA
TIOA
TIOADBA Data to be mapped
TCA
TCAPCLA Address of loaded map
MAP
DFHMAPDS Map and field specifi- cations

Pre-VS 3270 Input Mapping Operation - DFHBMSMM

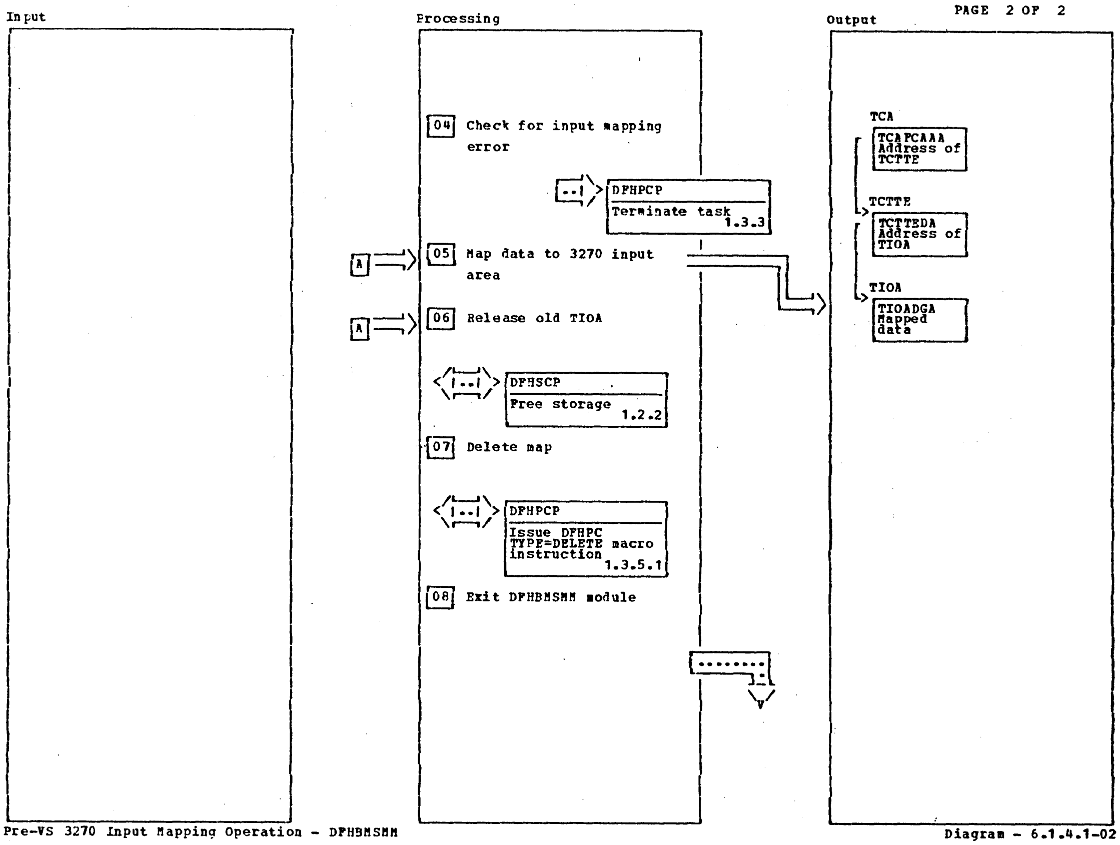
Diagram - 6.1.4.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 DFHBMSSM is loaded and entered via a DFHBMS TYPE=IN or MAP macro instruction from the application program when CICS V2.3 object code is executed on CICS/VS.<br>DFHBMSMM is entered from the DFHMCP module when a map assembled under CICS V2.3 is loaded by a CICS/VS DFHBMS macro request. | DFHBMSMM | DOCBMS01 | | | | | |
| 02 When DFHBMS TYPE=IN is requested, the data to be mapped is read from the terminal via a Terminal Control READ. | | DOCBMS10 | | | | | |
| 03 If the name of the map, and not the address of the map, has been passed by a DFHBMS request, the map is loaded via a Program Control LOAD. | | | | | | | |

Pre-VS 3270 Input Mapping Operation - DFHBMSMM

Diagram - 6.1.4.1-01

PAGE 2 OF 2



Pre-VS 3270 Input Mapping Operation - DFHBMSMM

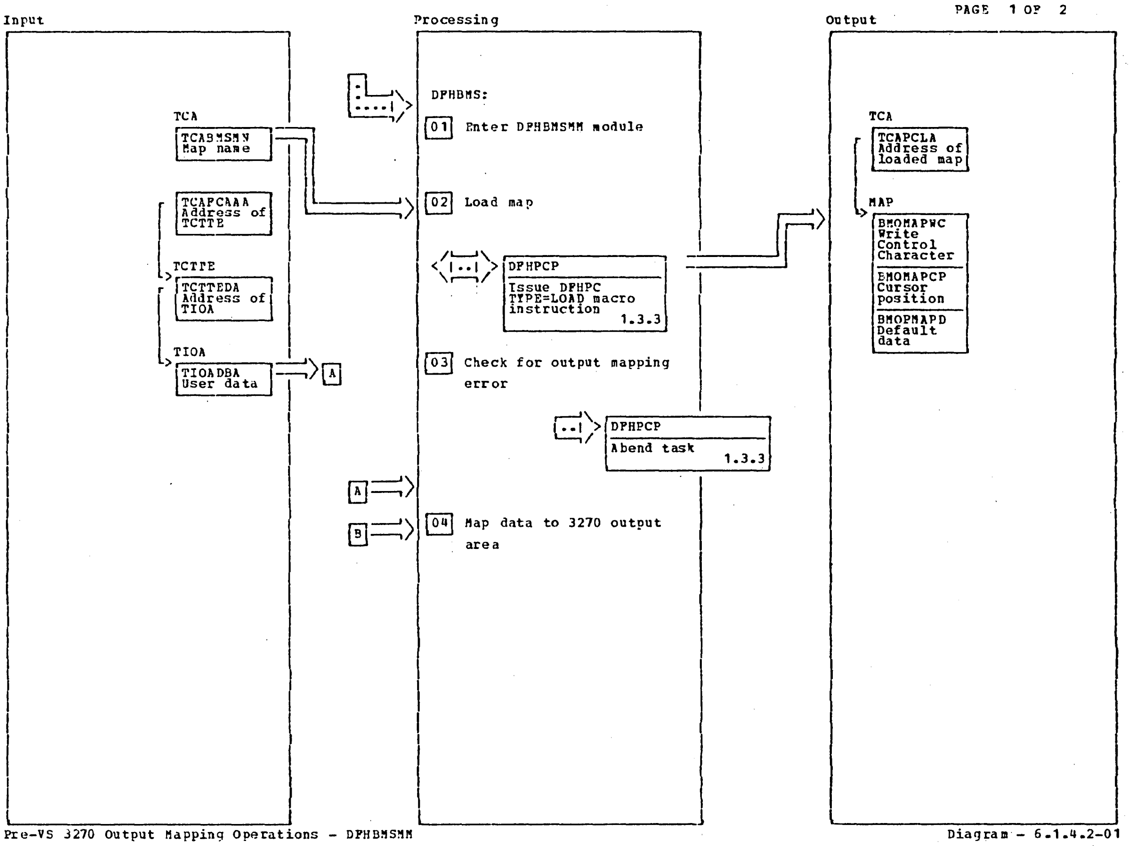
Diagram - 6.1.4.1-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 04 If the terminal is not a 3270 (ABMT) or if an output map has been loaded (ABMI), the data is not mapped and the transaction is abnormally terminated. | | | | | | | |
| 05 | | DOCBMS06 | | | | | |
| 06 If a DFHBMS TYPE=SAVE has not been requested, the old TIOA is released via a Storage Control FREEMAIN. | | BMINDUN | | | | | |
| 07 If the name of the map, and not the address of the map, has been passed by a DFHBMS request, the map is deleted via a Program Control DELETE. | | BMNOWAIT | | | | | |
| 08 Return to caller. | | DOCBMS09 | | | | | |

Pre-VS 3270 Input Mapping Operation - DFHBMSMM

Diagram - 6.1.4.1-02

PAGE 1 OF 2



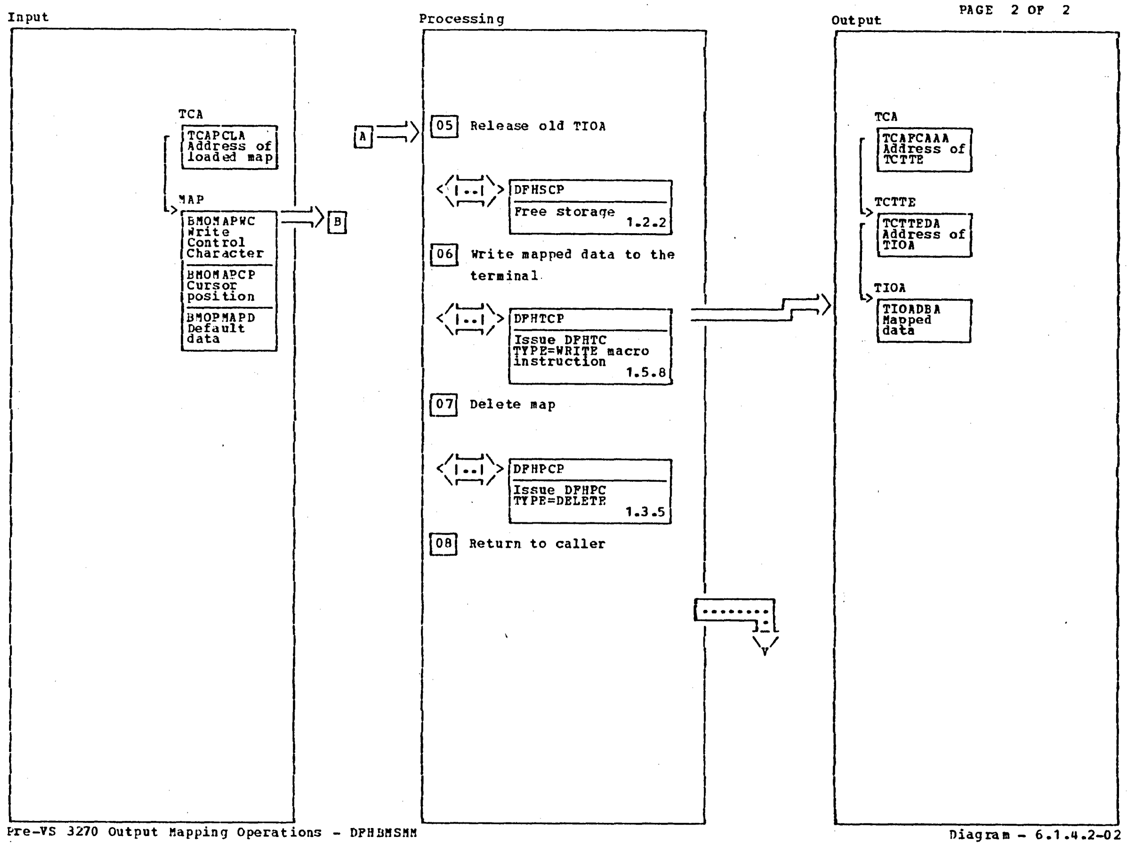
Pre-VS 3270 Output Mapping Operations - DFHBMSMM

Diagram - 6.1.4.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 DFHBMSMM is loaded and entered via a DFHBMS TYPE=OUT macro instruction from the application program when CICS V2.3 object code is executed on CICS/VS.<br>DFHBMSMM is entered from the DFHMCP module when a map assembled under CICS V2.3 is loaded by a CICS/VS DFHBMS macro request. | DFHBMSMM | DOCBMS01 | |
| 02 If the name of the map, and not the address of the map, has been passed by a DFHBMS request, the map is loaded via a Program Control LOAD. | | DOCBMS08 | |
| 03 If the terminal is not a 3270 (ABMT) or if an input map has been loaded (ABMO), the data is not mapped and the transaction is abnormally terminated. | | BMNOTOMP | |
| 04 The cursor position, the Write Control Character, and the data are formatted for 3270 output.<br>If DATA=ONLY is specified, fields containing user data in the TIOA are mapped.<br>If DATA=YES is specified, fields containing user data are mapped along with default data contained in the map. User data always overrides default data.<br>If DATA=NO is specified, default data mapped. | | DOCBMS04 | |

Pre-VS 3270 Output Mapping Operations - DFHBMSMM

Diagram - 6.1.4.2-01



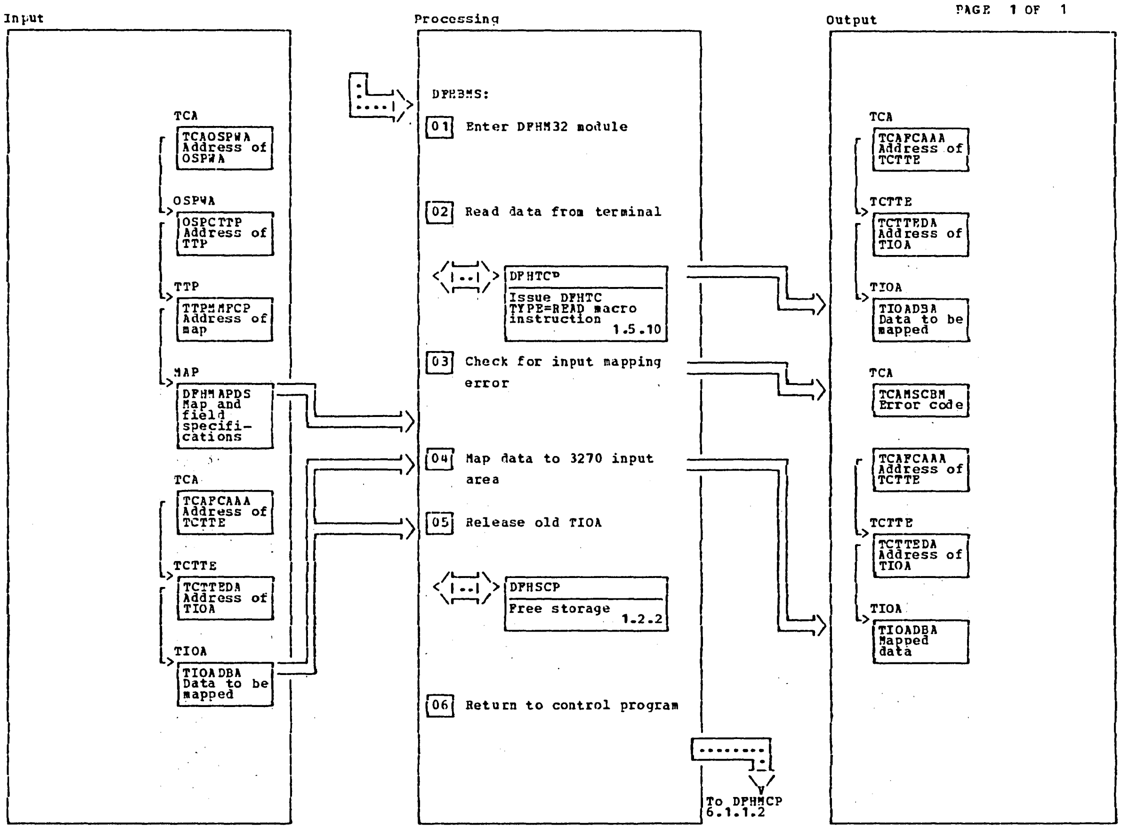
Pre-VS 3270 Output Mapping Operations - DFHBMSMM

Diagram - 6.1.4.2-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 If a DFHBMS TYPE=SAVE has not been requested, the old TIOA is released via a Storage Control FREEMAIN. | | BMOENOMP | | | | | |
| 06 | | BMNOERS | | | | | |
| 07 If the name of the maps and not the address of the map, has been passed by a DFHBMS request, the map is deleted via a Program Control DELETE. | | BMNOWAIT | | | | | |
| 08 | | DOCBMS09 | | | | | |

Pre-VS 3270 Output Mapping Operations - DFHBMSMM

Diagram - 6.1.4.2-02


PAGE 1 OF 1
Input
Processing
Output
TCA
TCAOSPWA Address of OSPWA
OSPWA
OSPCTTP Address of TTP
TTP
TTPMMPCP Address of map
MAP
DFHMAPDS Map and field specifi-cations
TCA
TCAFCAAA Address of TCTTE
TCTTE
TCTTEDA Address of TIOA
TIOA
TIOADBA Data to be mapped
DFHBMS:
01 Enter DFHM32 module
02 Read data from terminal
DFHTCP
Issue DFHTC TYPE=READ macro instruction
1.5.10
03 Check for input mapping error
04 Map data to 3270 input area
05 Release old TIOA
DFHSCP
Free storage
1.2.2
06 Return to control program
To DFHMCP 6.1.1.2
TCA
TCAFCAAA Address of TCTTE
TCTTE
TCTTEDA Address of TIOA
TIOA
TIOADBA Data to be mapped
TCA
TCAMSCBM Error code
TCAFCAAA Address of TCTTE
TCTTE
TCTTEDA Address of TIOA
TIOA
TIOADBA Mapped data

3270 Input Mapping Operations- DFHM32

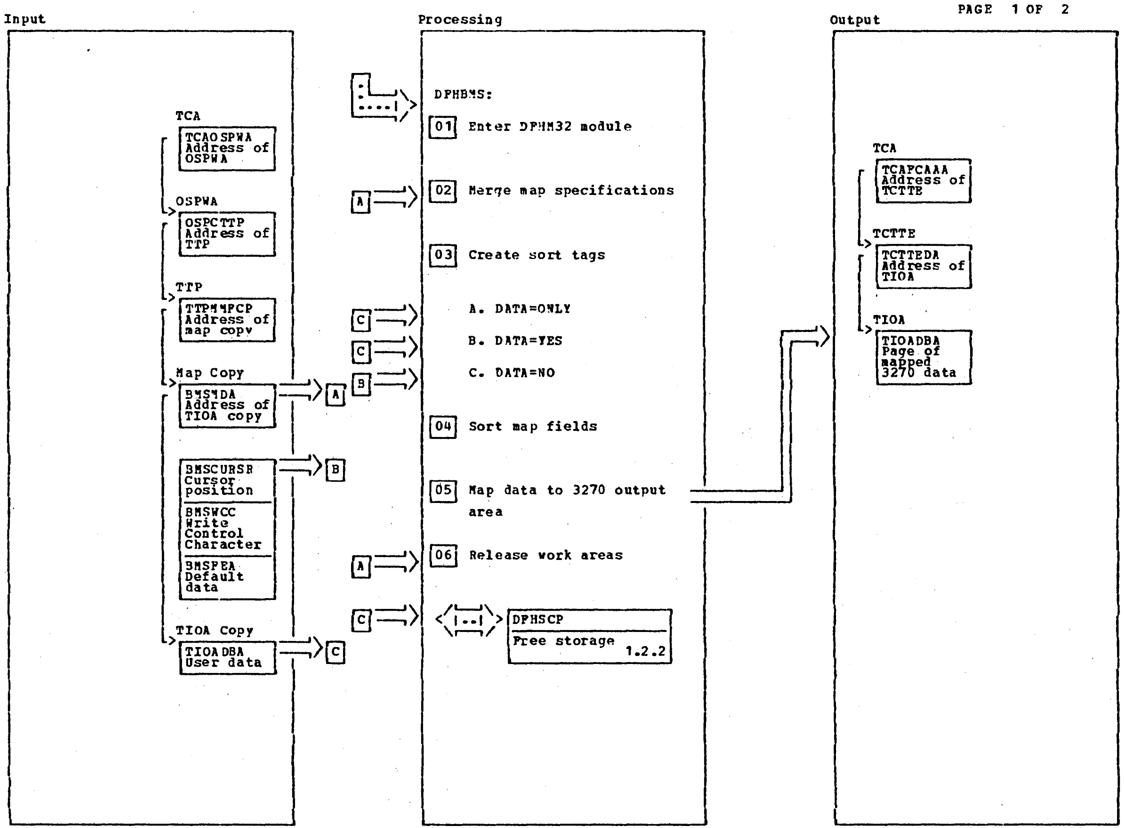
Diagram - 6.1.5.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The DFHM32 module is entered via a DFHBMS TYPE=IN or MAP macro instruction. | DFHM32 | DOCM3201 | | | | | |
| 02 When DFHBMS TYPE=IN is requested, the data to be mapped is read from the terminal via a Terminal Control READ. | | DOCM3211 | | | | | |
| 03 If a TIOA does not contain a data length and a SBA sequence, the data cannot be mapped and an error code is returned in the TCA. | | DOCM3213 | | | | | |
| 04 | | DOCM3212 | | | | | |
| 05 If a DFHBMS TYPE=SAVE has not been requested, the old TIOA is released via a Storage Control FREEMAIN. | | DOCM3214 | | | | | |
| 06 | | DOCM3210 | | | | | |

3270 Input Mapping Operations- DFHM32

Diagram - 6.1.5.1-01

PAGE 1 OF 2



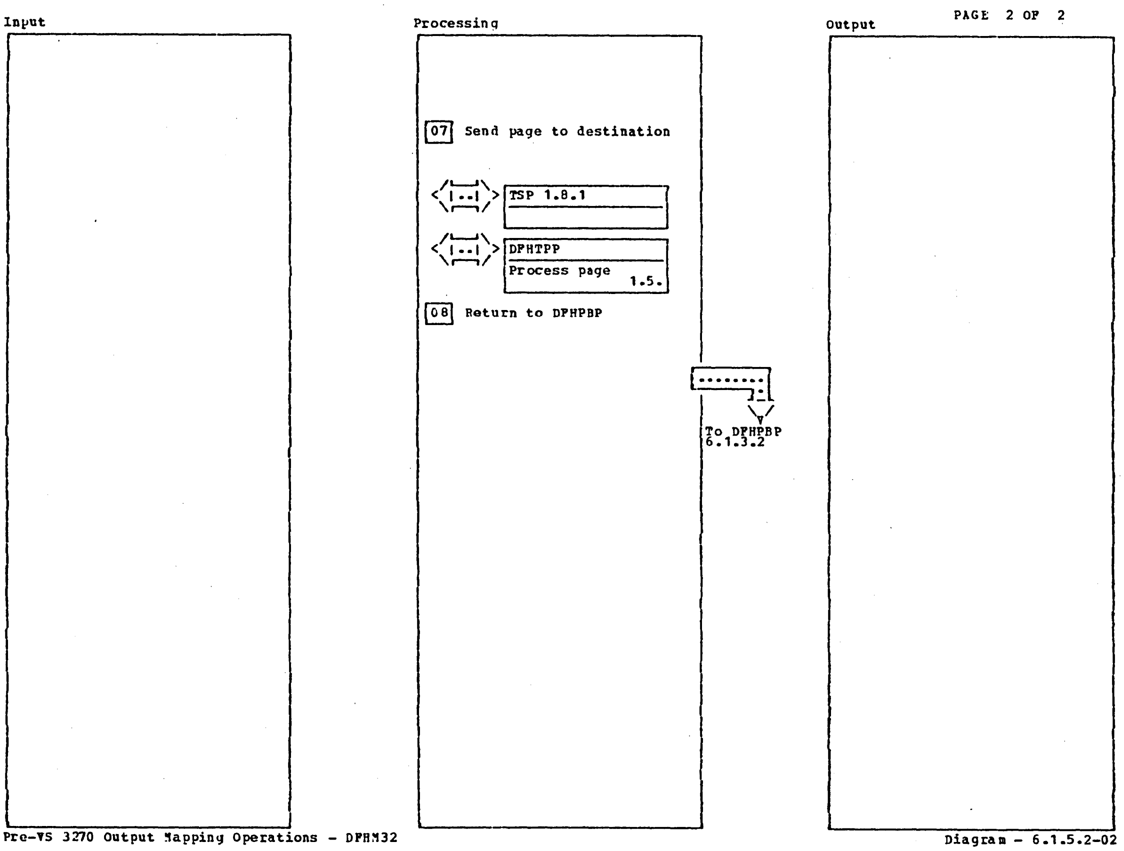
Pre-VS 3270 Output Mapping Operations - DFHM32

Diagram - 6.1.5.2-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The DFHM32 module is entered via a DFHBMS TYPE=PAGEBLD, TEXTBLD, OUT, STORE, or RETURN macro instruction. | DFHM32 | DOCM3201 | |
| 02 When multiple PAGEBLD or TEXTBLD requests are grouped into one page, the specifications are merged into one map.<br>The WCC from the first requested map in the page determines the WCC for the page unless it is overridden by a WCC from a DFHBMS CTRL operand request.<br>The cursor position from the last requested map in the page determines the cursor position on the page unless it is overridden by a DFHBMS CURSOR operand request. | | DOCM3202 | |
| 03 | | | |
| A. When DATA=ONLY is specified, fields containing user data in the TIOA are tagged for mapping. | | DOCM3203 | |
| B. When DATA=YES is specified, fields containing user data in the TIOA are tagged for mapping; otherwise, fields containing default data in the map are tagged for mapping. | | DOCM3205 | |
| C. When DATA=NO is specified, fields containing default data in the map are tagged for mapping. | | DOCM3205 | |
| 04 Sorting occurs only when map fields are specified out of sequence or when multiple PAGEBLD or TEXTBLD requests are grouped into one page. | | DOCM3206 | |
| 05 | | DOCM3207 | |
| 06 Release all map copies, TIOA copies, and related mapping work areas via a Storage Control FREEMAIN. | | | |

Pre-VS 3270 Output Mapping Operations - DFHM32

Diagram - 6.1.5.2-01



Pre-VS 3270 Output Mapping Operations - DFHM32

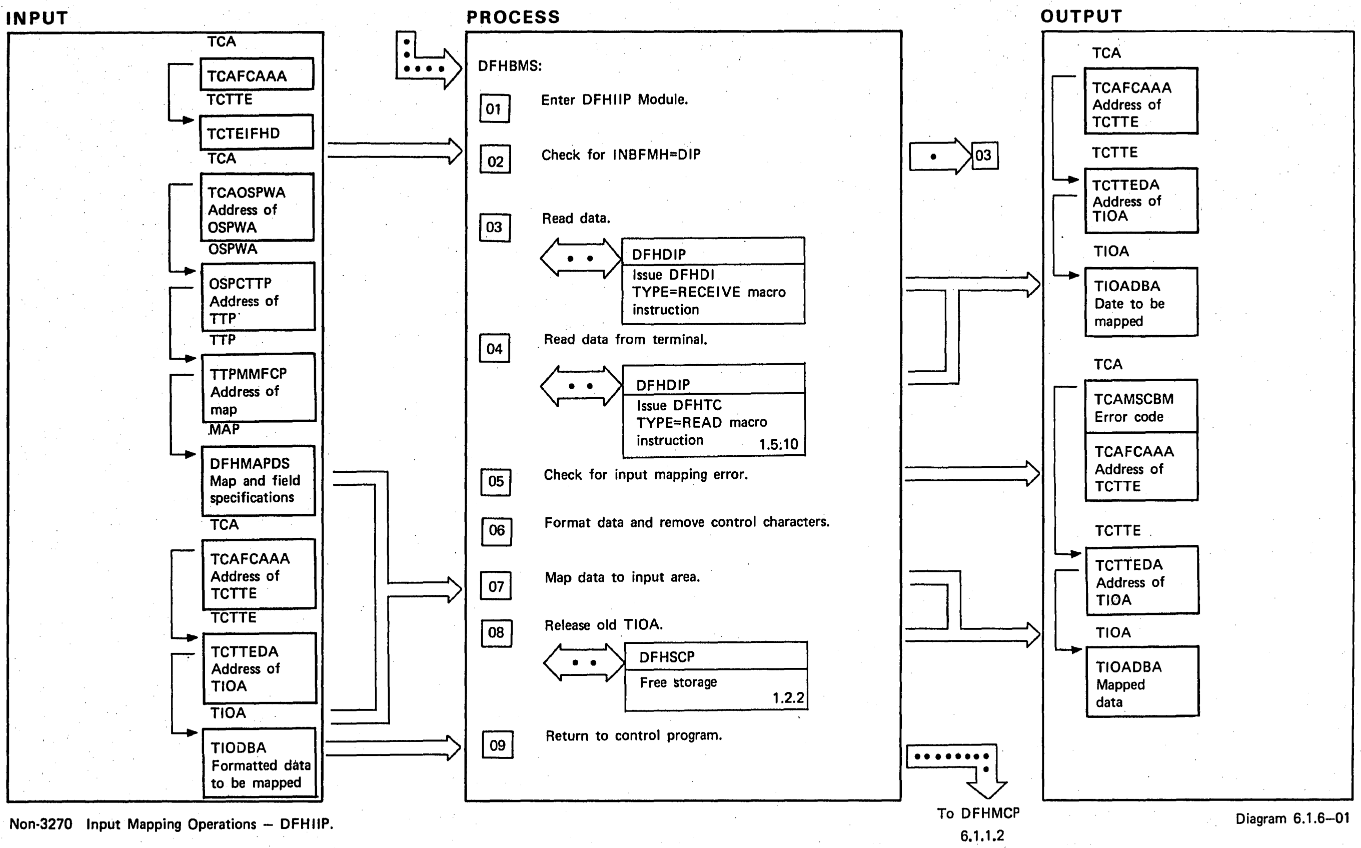
Diagram - 6.1.5.2-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 07 The Terminal Page Processor writes the page to the originating terminal when TYPE=OUT is specified, or places the page on Temporary Storage when DFHBMS TYPE=STORE is specified. | | DOCM3209 | | | | | |
| 08 | | DOCM3210 | | | | | |

Pre-VS 3270 Output Mapping Operations - DFHM32

Diagram - 6.1.5.2-02

Page 1 of 1



Non-3270 Input Mapping Operations – DFHIIP.

Diagram 6.1.6–01

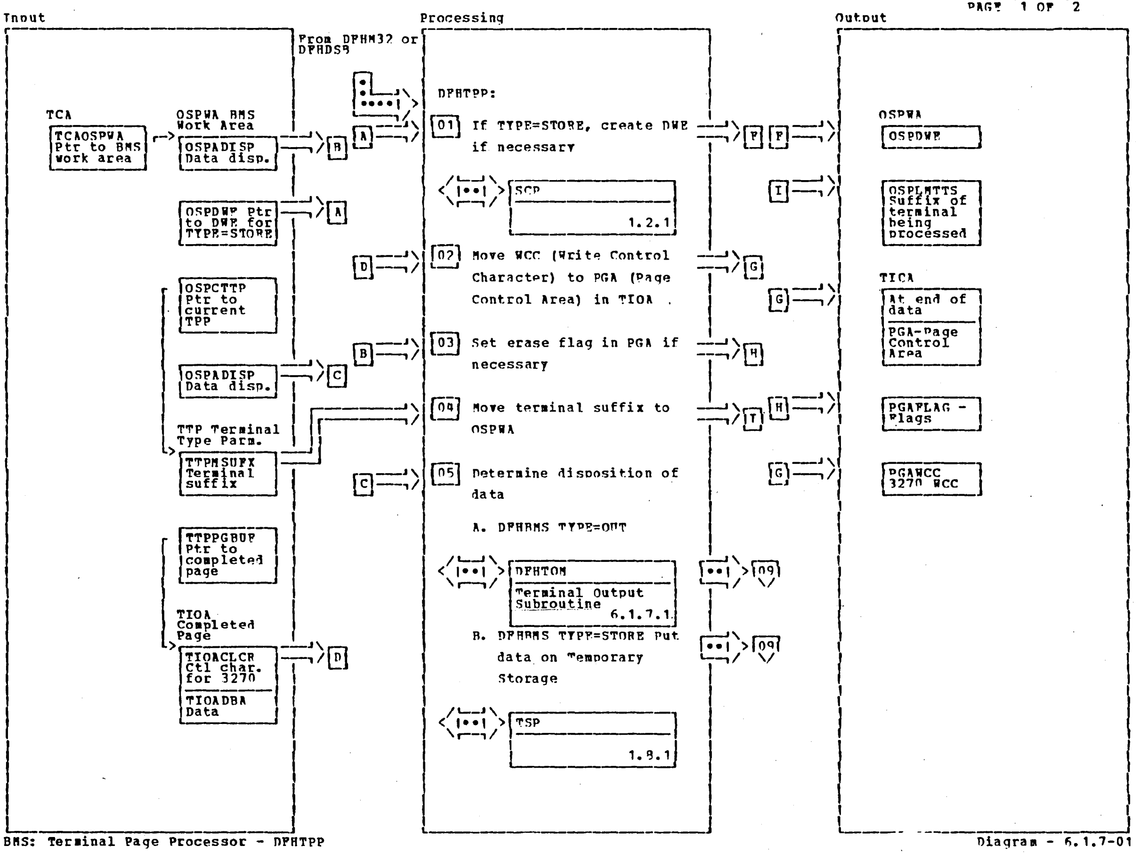
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The DFHIIP module is entered via a DFHBMS TYPE=IN or MAP macro instruction. | | | |
| 02 For the BLU, INBFMH=DIP may be specified on the PCT to give Batch Data Management on input. | | | |
| 03 When INSBMH=DIP is requested, data data is read using a Data Interchange RECEIVE. | | | |
| 04 When DFHBMS TYPE=IN is requested the data to be mapped is read from terminal via a Terminal Control READ. | | | |
| 05 If a TIOA does not Contain a data length or the map does not a length and width as specified by DFHMDI SIZE=(line,col),,the data cannot be mapped and an error code is returned in the TCA. | | | |
| 06 | | | |
| 07 | | | |
| 08 If a DFHBMS TYPE=SAVE has not been requested, the old TIOA is released via a Storage Control FREE-MAIN. | | | |
| 09 | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Non-3270 Input Mapping Operations – DFHIIP.

Diagram 6.1.6–01

CHART: DFHTPP
PAGE 1 OF 2


Input
Processing
Output
From DFHM32 or DFHDS3
TCA
TCAOSPWA Ptr to BMS work area
OSPWA BMS Work Area
OSPADISP Data disp.
OSPDWE Ptr to DWE for TYPE=STORE
OSPCTTP Ptr to current TTP
OSPADISP Data disp.
TTP Terminal Type Parm.
TTPMSUFX Terminal suffix
TTPPGBUF Ptr to completed page
TIOA Completed Page
TIOACLCR Ctl char. for 3270
TIOADBA Data
DFHTPP:
01 If TYPE=STORE, create DWE if necessary
SCP
1.2.1
02 Move WCC (Write Control Character) to PGA (Page Control Area) in TIOA .
03 Set erase flag in PGA if necessary
04 Move terminal suffix to OSPWA
05 Determine disposition of data
A. DFHBMS TYPE=OUT
DFHTOM
Terminal Output Subroutine
6.1.7.1
B. DFHBMS TYPE=STORE Put data on Temporary Storage
TSP
1.8.1
09
09
OSPWA
OSPDWE
OSPLMTTS Suffix of terminal being processed
TIOA
At end of data
PGA-Page Control Area
PGAFLAG - Flags
PGAWCC 3270 WCC
A
B
C
D
F
G
H
I

BMS: Terminal Page Processor - DFHTPP

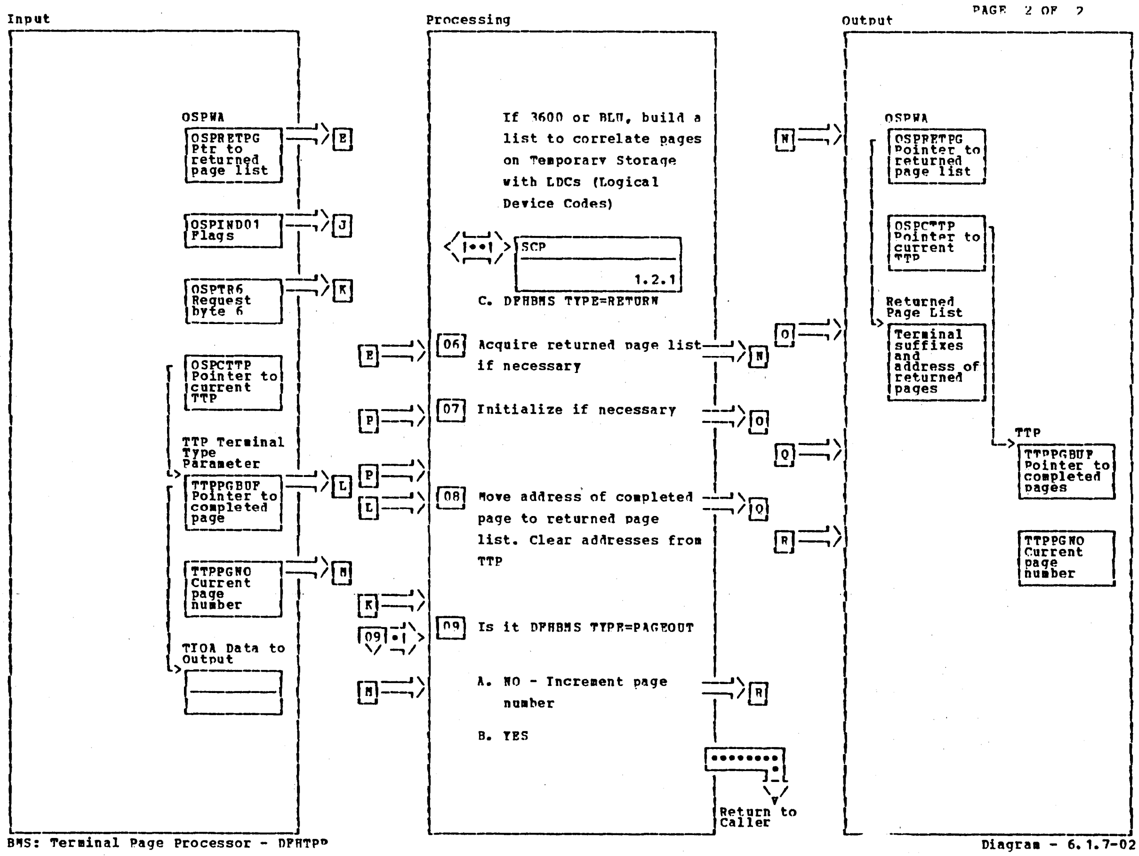
Diagram - 6.1.7-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 03 TIOA is shown here. TIOA and TSIOA both have 12-byte prefixes. The only difference is that the TSIOA data length must include the field with the length. Therefore 4 is added to the TIOA data length here. | | | | | | | |
| 05 If device is a 2740 Model 2 with Buffer Receive, suffix is changed from F to R. | | | | | | | |

BMS: Terminal Page Processor - DFHTPP

Diagram - 6.1.7-01

PAGE 2 OF 2


Input
Processing
Output
OSPWA
OSPRETPG Ptr to returned page list
OSPIND01 Flags
OSPTR6 Request byte 6
OSPCTTP Pointer to current TTP
TTP Terminal Type Parameter
TTPPGBUF Pointer to completed page
TTPPGNO Current page number
TIOA Data to Output
E
J
K
L
M
N
O
P
Q
R
SCP
1.2.1
09
OSPWA
OSPRETPG Pointer to returned page list
OSPCTTP Pointer to current TTP
Returned Page List
Terminal suffixes and address of returned pages
TTP
TTPPGBUF Pointer to completed pages
TTPPGNO Current page number
Return to Caller

If 3600 or BLU, build a list to correlate pages on Temporary Storage with LDCs (Logical Device Codes)

C. DFHBMS TYPE=RETURN

06 Acquire returned page list if necessary

07 Initialize if necessary

08 Move address of completed page to returned page list. Clear addresses from TTP

09 Is it DFHBMS TYPE=PAGEOUT

A. NO - Increment page number

B. YES

BMS: Terminal Page Processor - DFHTPP

Diagram - 6.1.7-02

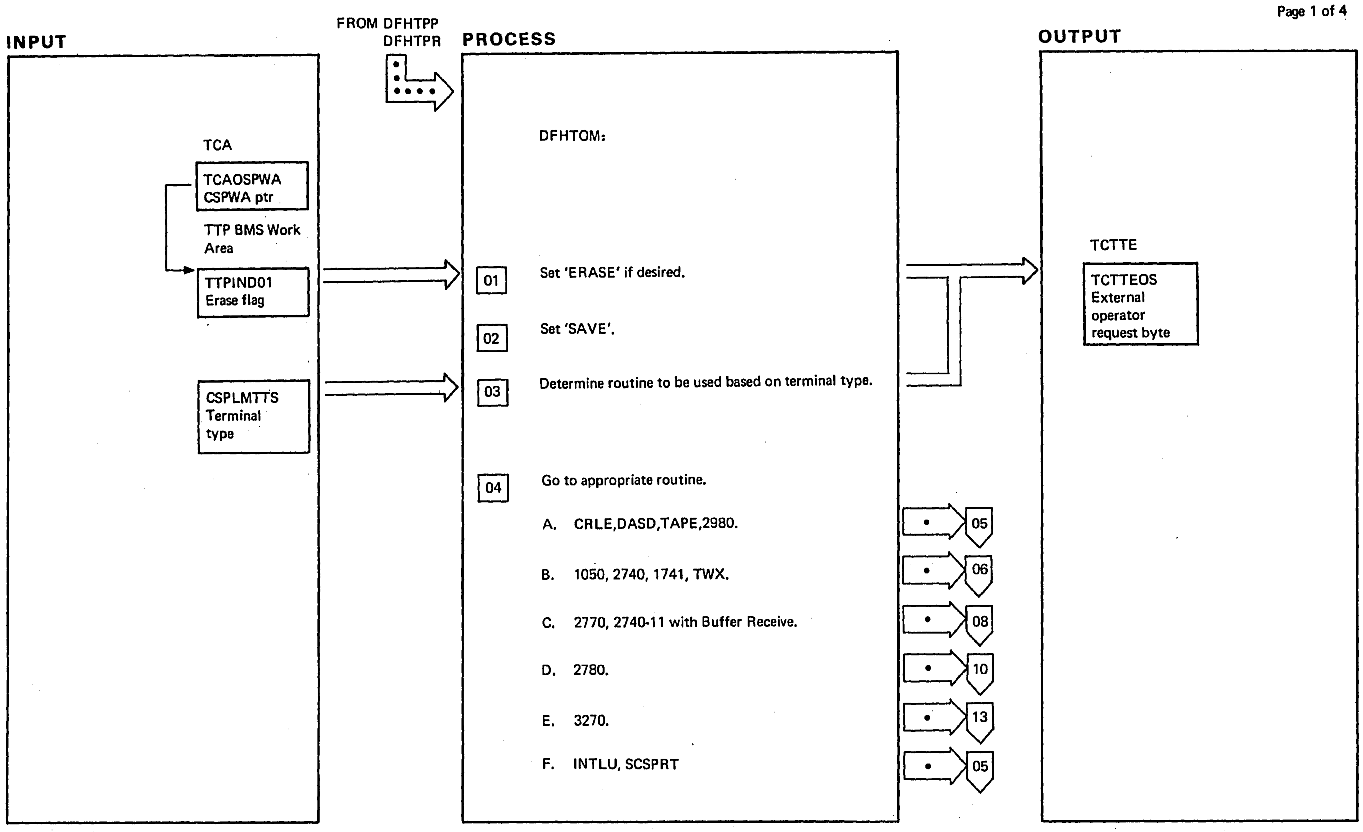
| Notes | Routine | Label | Ref |
|---|---|---|---|

| Notes | Routine | Label | Ref |
|---|---|---|---|

BMS: Terminal Page Processor - DFHTPP

Diagram - 6.1.7-02

Page 1 of 4



DFHTOM:

01 Set 'ERASE' if desired.

02 Set 'SAVE'.

03 Determine routine to be used based on terminal type.

04 Go to appropriate routine.

A. CRLE,DASD,TAPE,2980.

B. 1050, 2740, 1741, TWX.

C. 2770, 2740-11 with Buffer Receive.

D. 2780.

E. 3270.

F. INTLU, SCSPRT

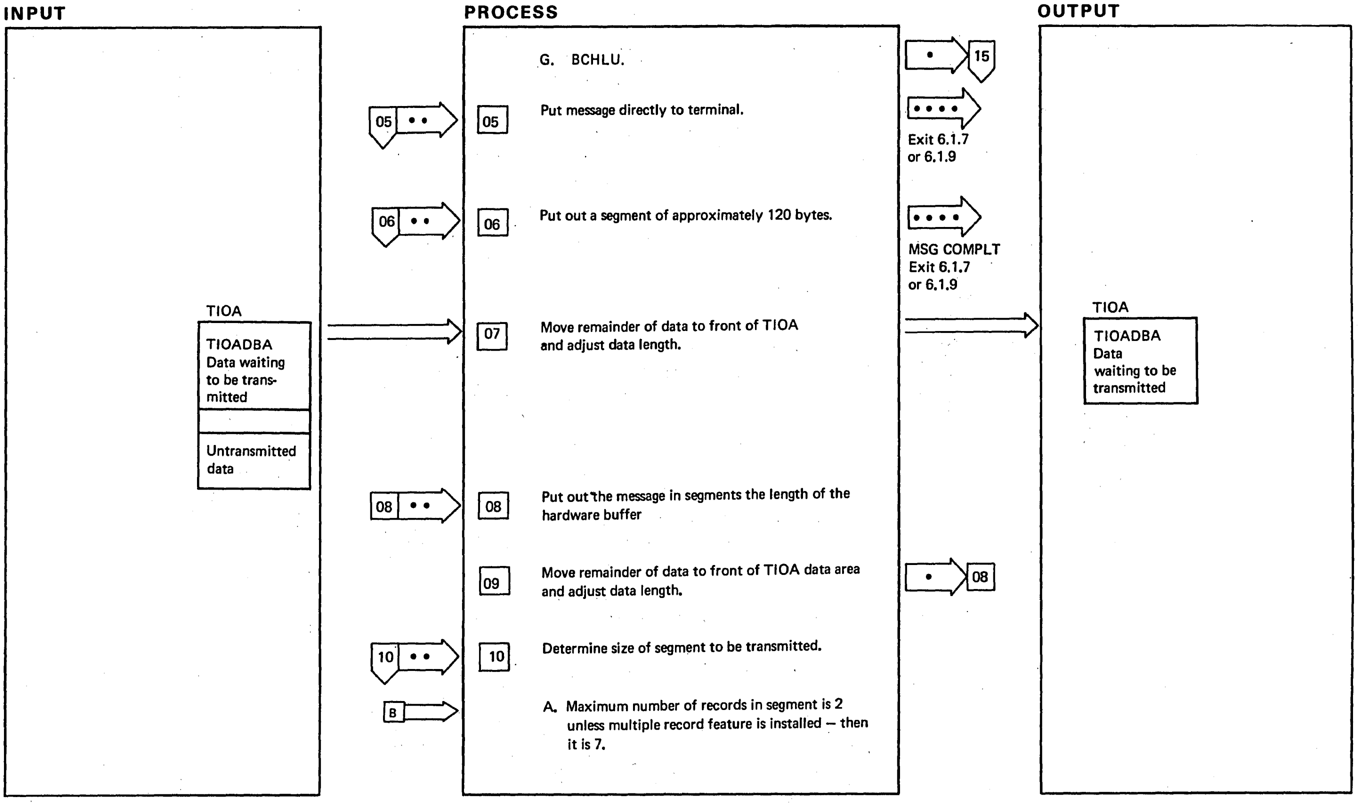
Terminal Output Subroutine – DFHTOM

Diagram – 6.1.7.1.– 01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 This routine is copied as inline code into programs DFHTPP and DFHTPR. | | TPOUTPUT | |
| 02 The TIOA must be saved since it is probably not on the TCTTE storage chair and therefore Terminal Control may not free it. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Page 2 of 4



G. BCHLU.

05 Put message directly to terminal.

06 Put out a segment of approximately 120 bytes.

07 Move remainder of data to front of TIOA and adjust data length.

08 Put out the message in segments the length of the hardware buffer

09 Move remainder of data to front of TIOA data area and adjust data length.

10 Determine size of segment to be transmitted.

A. Maximum number of records in segment is 2 unless multiple record feature is installed — then it is 7.

Terminal Output Subroutine — DFHTOM

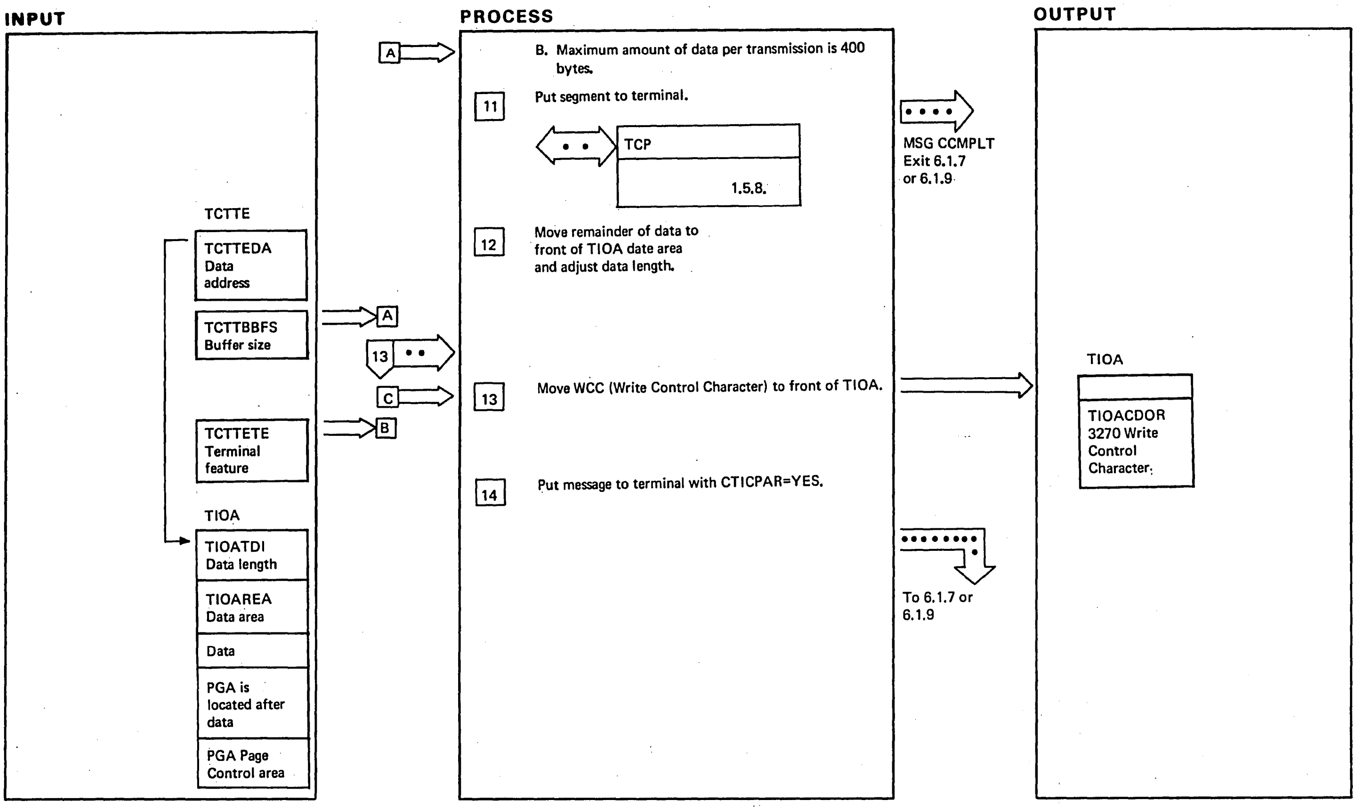
Diagram — 6.1.7.1.— 02

| | NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 05 | This processing is for CRLP, TAPE, DASD, 2980. For these terminals, Terminal Control program handles segmenting. | CBLP<br>TAPE<br>DASD<br>2890–182<br>2980–4<br>ALL | TPTRMA<br>TPTRMC<br>TPTRMC<br>TPTR<br>TPTRMR<br>TPTRMPUT | |
| 06 | This processing is for unbuffered start-stop terminals — 1050, 2740-1, 2741, TWX. The segment size is small to minimize retransmission due to errors. The size of the segment to be transmitted is determined by searching for a new line —X'15'— after the 120th byte of the message. | TWX<br>1050<br>2740<br>2741<br>ALL | TPTRMD<br>TMPRME<br>TPTBMF<br>TPTBMG<br>TPSTSTOP | |
| 07 | This processing is for 2770 and 2740-11 with Buffer Receive. For 2740 the actual buffer size is divided by 2 as a rough but fast allowance for the fact that the buffer must also accommodate upshift and downshift characters. | 2740<br>2770<br>BOTH | TPTFMH<br>TPTFMI<br>TPBUFPUT | |
| 08 | See steps 6-7 for details of data flow. | | | |
| 10 | This processing is for 2780.<br>A. A segment is made up of whole records separated by unit separator characters (IUS = X'1F'). | 2780 | TPTFMJ | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Terminal Output Subroutine — DFHTOM

Diagram — 6.1.7.1.— 02



B. Maximum amount of data per transmission is 400 bytes.

11 Put segment to terminal.

12 Move remainder of data to front of TIOA date area and adjust data length.

13 Move WCC (Write Control Character) to front of TIOA.

14 Put message to terminal with CTICPAR=YES.

Terminal Output Subroutine – DFHTOM

Diagram – 6.1.7.1 – 03

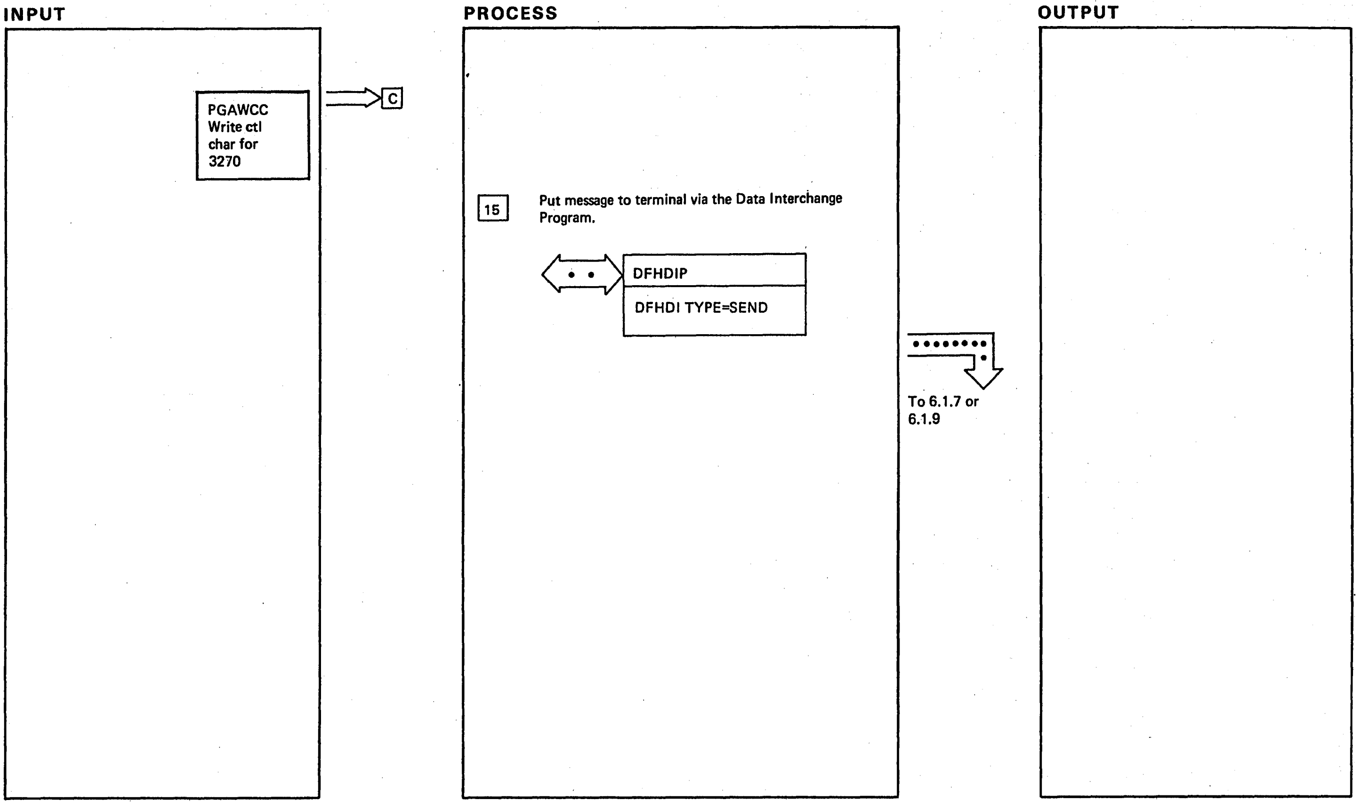
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 12 See steps 6-7 for details of data flow. | | | |
| 13 This processing is performed for all models of the 3270 Information Display System. | 3270 – 1<br>3270 – 2<br>BOTH | TPTBML<br>TPTBMM<br>TPTBMM | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Terminal Output Subroutine – DFHTOM

Diagram – 6.1.7.1.– 03

Page 4 of 4



Put message to terminal via the Data Interchange Program.

Terminal Output Subroutine — DFHTOM

Diagram — 6.1.7.1. — 04

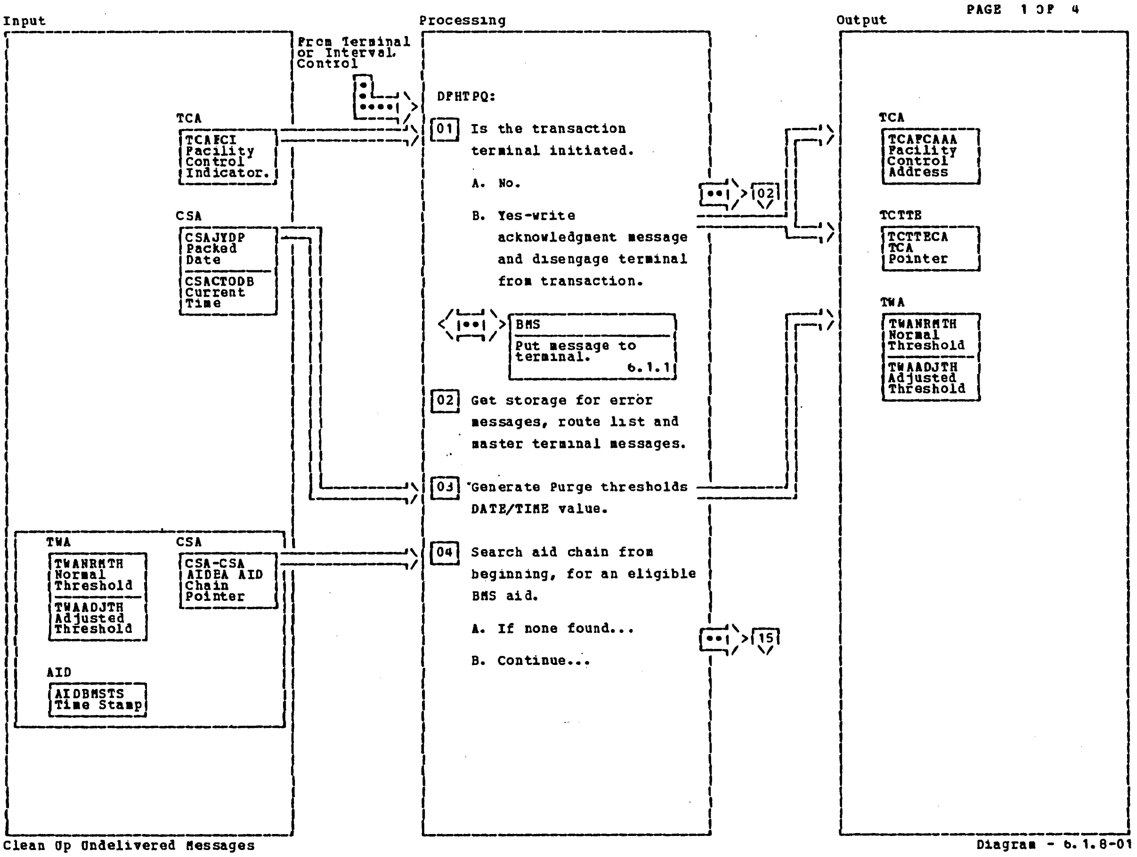
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 15 The Data Interchange Program handles destination selection for the BLU. | BLU | TPTRMY | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Terminal Output Subroutine — DFHTOM

Diagram — 6.1.7.1 — 04

PAGE 1 OF 4



Input
Processing
Output
From Terminal or Interval Control
TCA
TCAFCI Facility Control Indicator.
CSA
CSAJYDP Packed Date
CSACTODB Current Time
TWA
TWANRMTH Normal Threshold
TWAADJTH Adjusted Threshold
CSA
CSA-CSA AIDEA AID Chain Pointer
AID
AIDBMSTS Time Stamp
02
15
BMS
Put message to terminal.
6.1.1
TCA
TCAFCAAA Facility Control Address
TCTTE
TCTTECA TCA Pointer
TWA
TWANRMTH Normal Threshold
TWAADJTH Adjusted Threshold

DFHTPQ:

01 Is the transaction terminal initiated.

A. No.

B. Yes-write acknowledgment message and disengage terminal from transaction.

02 Get storage for error messages, route list and master terminal messages.

03 Generate Purge thresholds DATE/TIME value.

04 Search aid chain from beginning, for an eligible BMS aid.

A. If none found...

B. Continue...

Clean Up Undelivered Messages

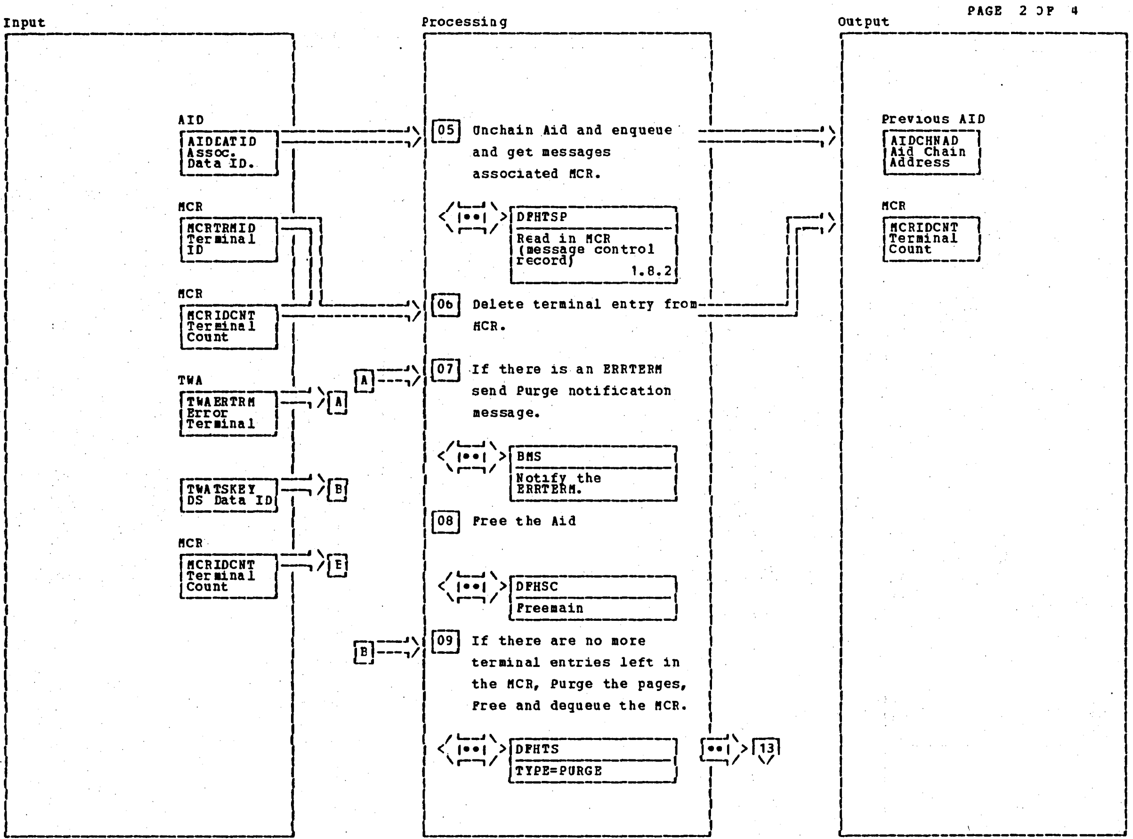
Diagram - 6.1.8-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 B also releases all previous TIOAs and gets user storage for subsequent messages. | | | | | | | |
| 03 Time stamp is composed of two bytes of packed day of the year and two bytes at the current time in 256 second units (2nd and 3rd bytes of CSACTODB).<br>The adjusted threshold, TWAADJTH, is used for checking when current time is this year. | | | | | | | |
| 04 An eligible BMS aid is one that has yet to be initiated and is older than the purge delay intervals. | | TPQAID | | | | | |

Clean Up Undelivered Messages

Diagram - 6.1.8-01

PAGE 2 OF 4



05 Unchain Aid and enqueue and get messages associated MCR.

06 Delete terminal entry from MCR.

07 If there is an ERRTERM send Purge notification message.

08 Free the Aid

09 If there are no more terminal entries left in the MCR, Purge the pages, Free and dequeue the MCR.

Clean Up Undelivered Messages

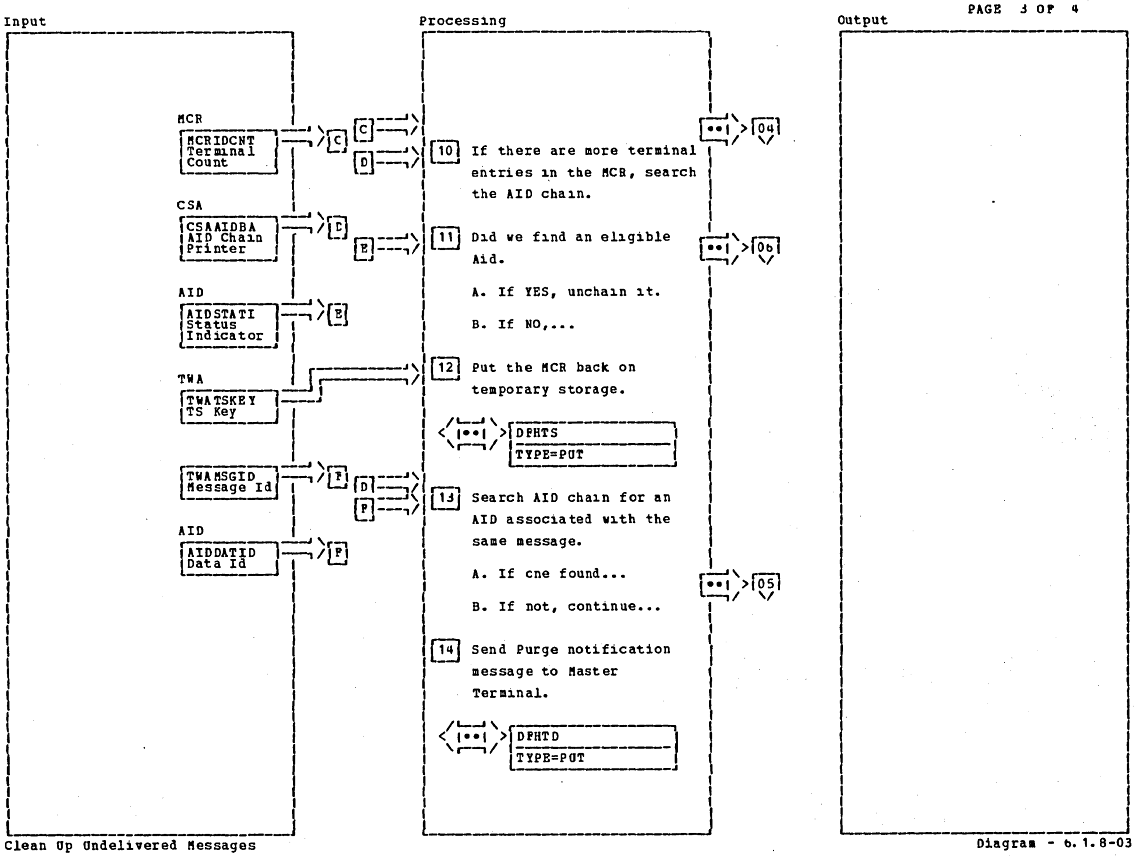
Diagram - 6.1.8-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 | | TPQUCAID | | | | | |
| 06 | | TPGDELT | | | | | |
| 07 Prior to sending the message to the ERRTERM, a DFHBMS TYPE=ROUTE macro instruction has to be issued to the error terminal. | | | | | | | |
| 09 If the MCR that was just processed was for a BMS system message, then control resumes at step 04. | | | | | | | |

Clean Up Undelivered Messages

Diagram - 6.1.8-02

PAGE 3 OF 4



10 If there are more terminal entries in the MCR, search the AID chain.

11 Did we find an eligible Aid.

A. If YES, unchain it.

B. If NO,...

12 Put the MCR back on temporary storage.

13 Search AID chain for an AID associated with the same message.

A. If one found...

B. If not, continue...

14 Send Purge notification message to Master Terminal.

Clean Up Undelivered Messages

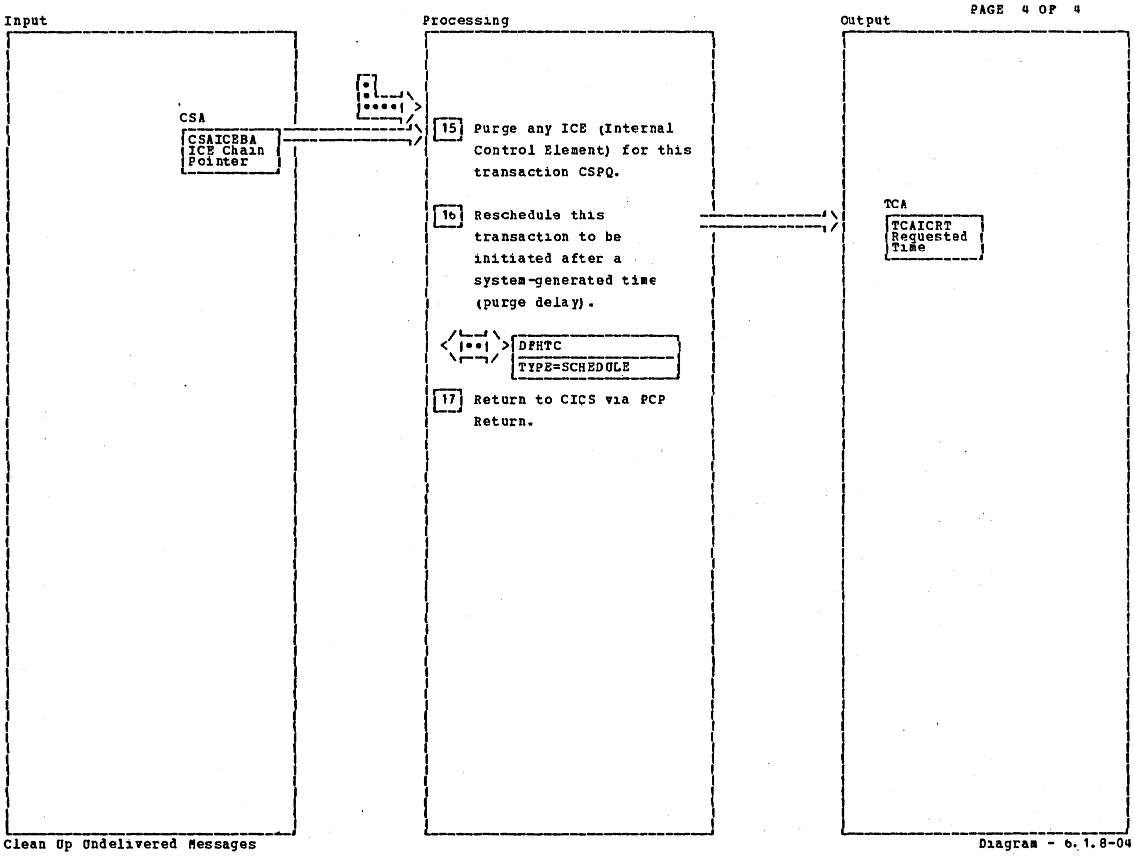
Diagram - 6.1.8-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 10 This AID chain search starts from the beginning and searches for an eligible AID with the same MCR as the last AID processed. | | TPQACS2 | | | | | |
| 12 Also free and dequeue the MCR. | | TPQMCRBK | | | | | |
| 13 This AID chain search starts from the beginning and searches for an eligible AID for the same message as the last AID processed. | | TPQACS3 | | | | | |

Clean Up Undelivered Messages

Diagram - 6.1.8-03

PAGE 4 OF 4



15 Purge any ICE (Internal Control Element) for this transaction CSPQ.

16 Reschedule this transaction to be initiated after a system-generated time (purge delay).

17 Return to CICS via PCP Return.

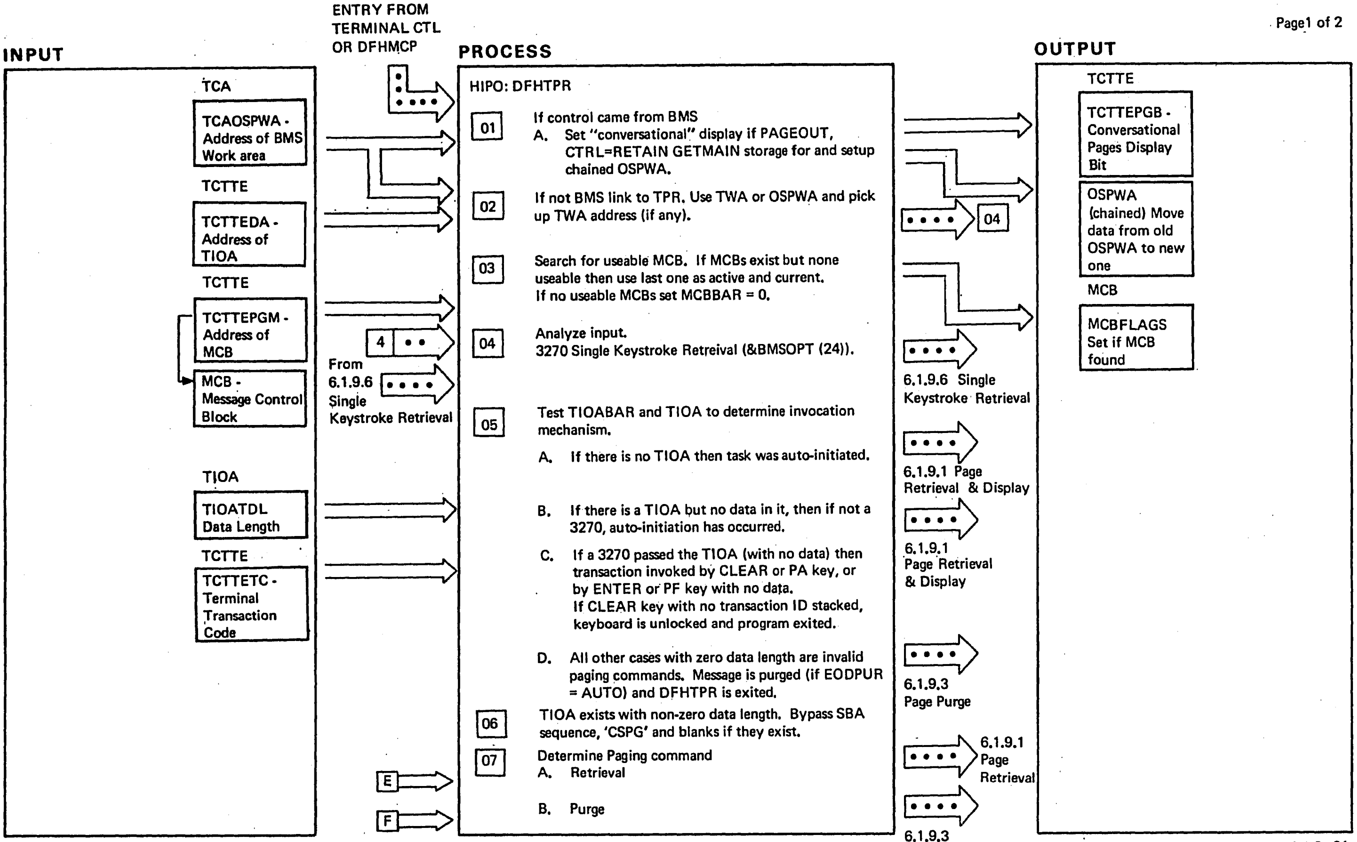
Clean Up Undelivered Messages

Diagram - 6.1.8-04

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 14 | | TPQEXIT | | | | | |
| 15 | | TPQICEX | | | | | |

Clean Up Undelivered Messages

Diagram - 6.1.8-04


Page 1 of 2
INPUT
TCA
TCAOSPWA - Address of BMS Work area
TCTTE
TCTTEDA - Address of TIOA
TCTTE
TCTTEPGM - Address of MCB
MCB - Message Control Block
TIOA
TIOATDL Data Length
TCTTE
TCTTETC - Terminal Transaction Code
ENTRY FROM TERMINAL CTL OR DFHMCP
4
From 6.1.9.6 Single Keystroke Retrieval
E
F
PROCESS
HIPO: DFHTPR
04
6.1.9.6 Single Keystroke Retrieval
6.1.9.1 Page Retrieval & Display
6.1.9.1 Page Retrieval & Display
6.1.9.3 Page Purge
6.1.9.1 Page Retrieval
6.1.9.3 Page Purge
OUTPUT
TCTTE
TCTTEPGB - Conversational Pages Display Bit
OSPWA (chained) Move data from old OSPWA to new one
MCB
MCBFLAGS Set if MCB found

01 If control came from BMS
A. Set "conversational" display if PAGEOUT, CTRL=RETAIN GETMAIN storage for and setup chained OSPWA.

02 If not BMS link to TPR. Use TWA or OSPWA and pick up TWA address (if any).

03 Search for useable MCB. If MCBs exist but none useable then use last one as active and current. If no useable MCBs set MCBBAR = 0.

04 Analyze input. 3270 Single Keystroke Retreival (&BMSOPT (24)).

05 Test TIOABAR and TIOA to determine invocation mechanism.
A. If there is no TIOA then task was auto-initiated.
B. If there is a TIOA but no data in it, then if not a 3270, auto-initiation has occurred.
C. If a 3270 passed the TIOA (with no data) then transaction invoked by CLEAR or PA key, or by ENTER or PF key with no data. If CLEAR key with no transaction ID stacked, keyboard is unlocked and program exited.
D. All other cases with zero data length are invalid paging commands. Message is purged (if EODPUR = AUTO) and DFHTPR is exited.

06 TIOA exists with non-zero data length. Bypass SBA sequence, 'CSPG' and blanks if they exist.

07 Determine Paging command
A. Retrieval
B. Purge

BMS: Page Retrieval Program — DFHTPR: Entry Analysis

Diagram 6.1.9–01

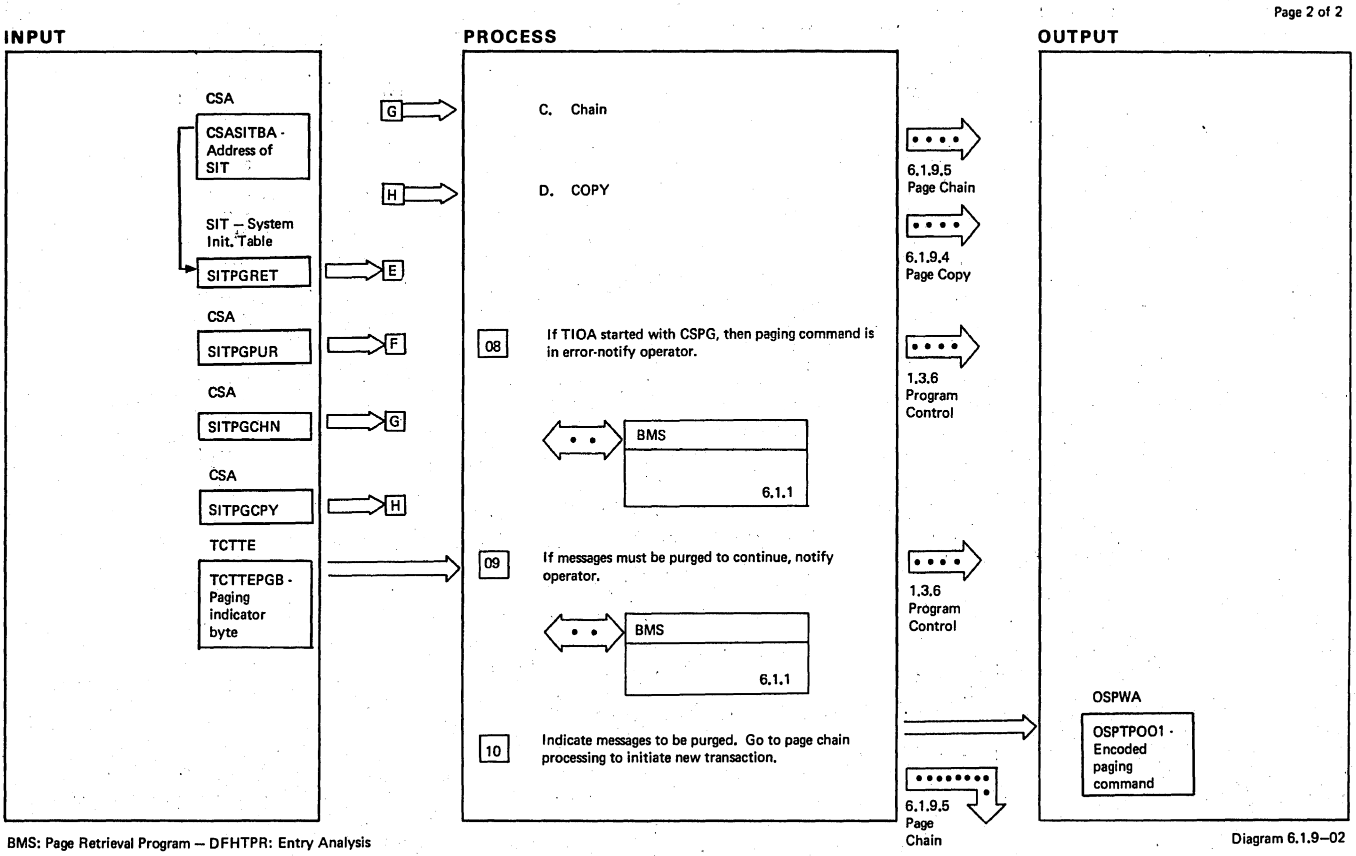
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 | TPMCBCEK | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

BMS: Page Retrieval Program — DFHTPR: Entry Analysis

Diagram 6.1.9–01

Page 2 of 2



C. Chain

D. COPY

08 If TIOA started with CSPG, then paging command is in error-notify operator.

09 If messages must be purged to continue, notify operator.

10 Indicate messages to be purged. Go to page chain processing to initiate new transaction.

BMS: Page Retrieval Program — DFHTPR: Entry Analysis

Diagram 6.1.9—02

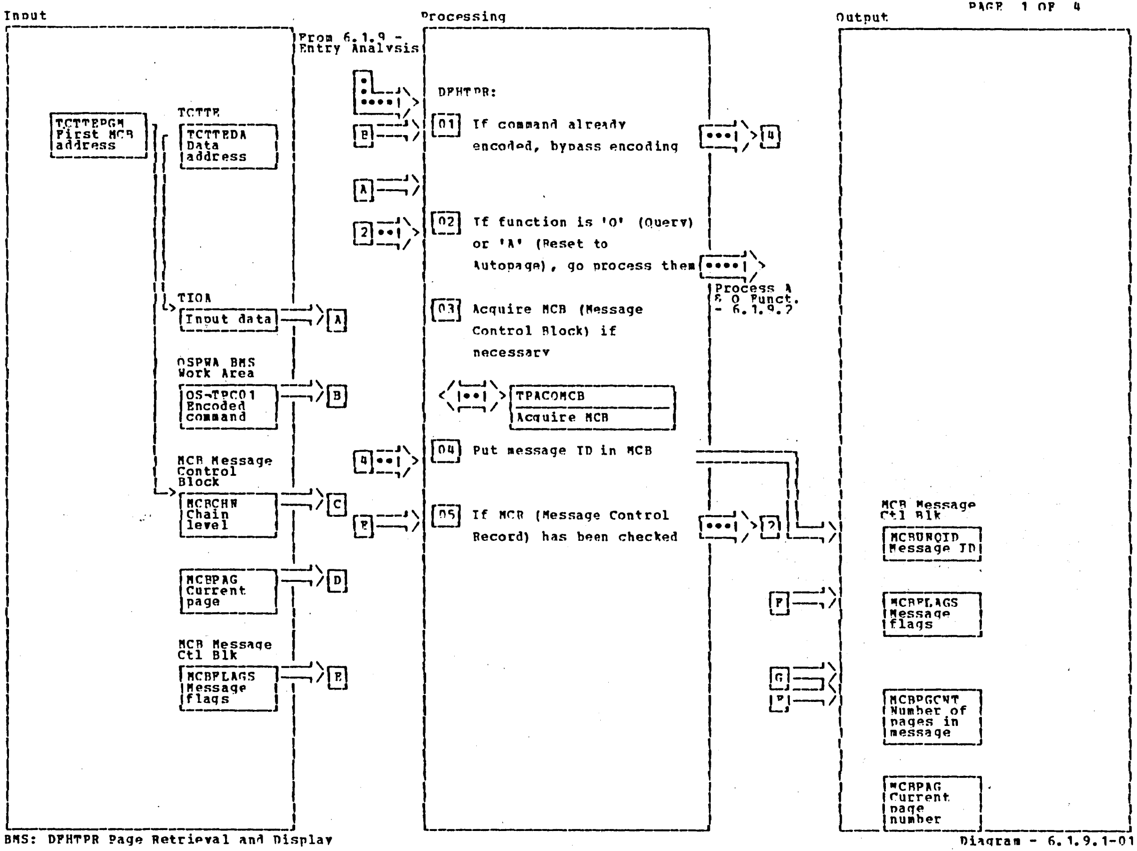
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 05 | TPBNCCPR | | |
| 06 | TPBCHKCX | | |
| 08 | TPBCFODP | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

BMS: Page Retrieval Program — DFHTPR: Entry Analysis

Diagram 6.1.9—02

CHART: DFHTPR
PAGE 1 OF 4


Input
Processing
Output
TCTTEPGM First MCB address
TCTTE
TCTTEDA Data address
TIOA
Input data
OSPWA BMS Work Area
OS-TPC01 Encoded command
MCB Message Control Block
MCBCHN Chain level
MCBPAG Current page
MCB Message Ctl Blk
MCBFLAGS Message flags
A
B
C
D
E
From 6.1.9 - Entry Analysis
2
4
DFHTPR:
01 If command already encoded, bypass encoding
02 If function is 'Q' (Query) or 'A' (Reset to Autopage), go process them
03 Acquire MCB (Message Control Block) if necessary
TPACOMCB
Acquire MCB
04 Put message ID in MCB
05 If MCR (Message Control Record) has been checked
Process A & Q Funct. - 6.1.9.2
MCB Message Ctl Blk
MCBUNQID Message ID
F
MCBFLAGS Message flags
G
MCBPGCNT Number of pages in message
MCBPAG Current page number

BMS: DFHTPR Page Retrieval and Display

Diagram - 6.1.9.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | TPENCRET | | | | | |
| 03 This routine is not charted. It obtains an MCB from the shared subpool and chains it to the previous MCB on the TCTTE. | | TPACOMCB | | | | | |
| 04 | | TPRMRGID | | | | | |
| 05 | | TPRDMCR | | | | | |

BMS: DFHTPR Page Retrieval and Display

Diagram - 6.1.9.1-01

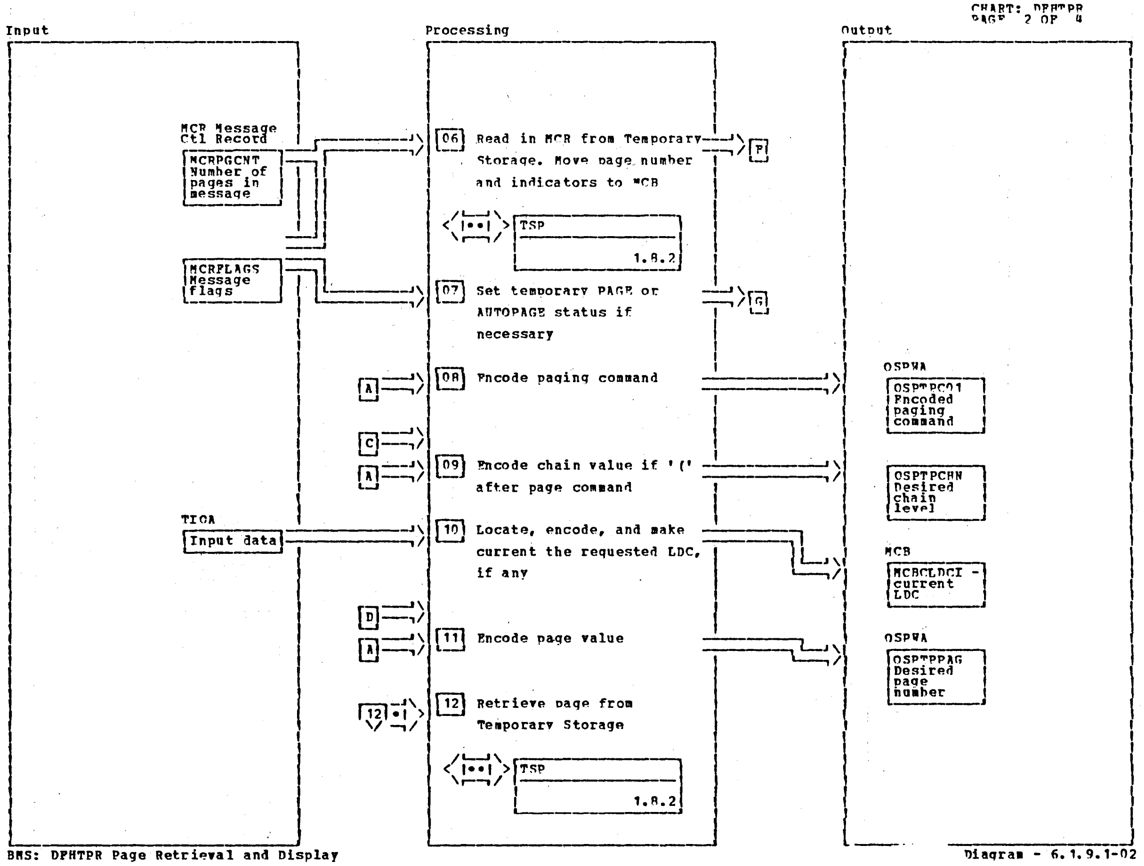


CHART: DFHTPR
PAGE 2 OF 4
Input
Processing
Output
MCR Message Ctl Record
MCRPGCNT Number of pages in message
MCRFLAGS Message flags
TIOA
Input data
06 Read in MCR from Temporary Storage. Move page number and indicators to MCB
TSP
1.8.2
F
07 Set temporary PAGE or AUTOPAGE status if necessary
G
A
08 Encode paging command
C
A
09 Encode chain value if '(' after page command
10 Locate, encode, and make current the requested LDC, if any
D
A
11 Encode page value
12
12 Retrieve page from Temporary Storage
TSP
1.8.2
OSPWA
OSPTPC01 Encoded paging command
OSPTPCHN Desired chain level
MCB
MCBCLDCI - current LDC
OSPWA
OSPTPPAG Desired page number

BMS: DFHTPR Page Retrieval and Display

Diagram - 6.1.9.1-02

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 07 | | TPRMCBPA | |
| 08 | | TPERCHN | |
| 09 | | TPERCH1 | |
| 11 | | TPERETPG | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

BMS: DFHTPR Page Retrieval and Display

Diagram - 6.1.9.1-02

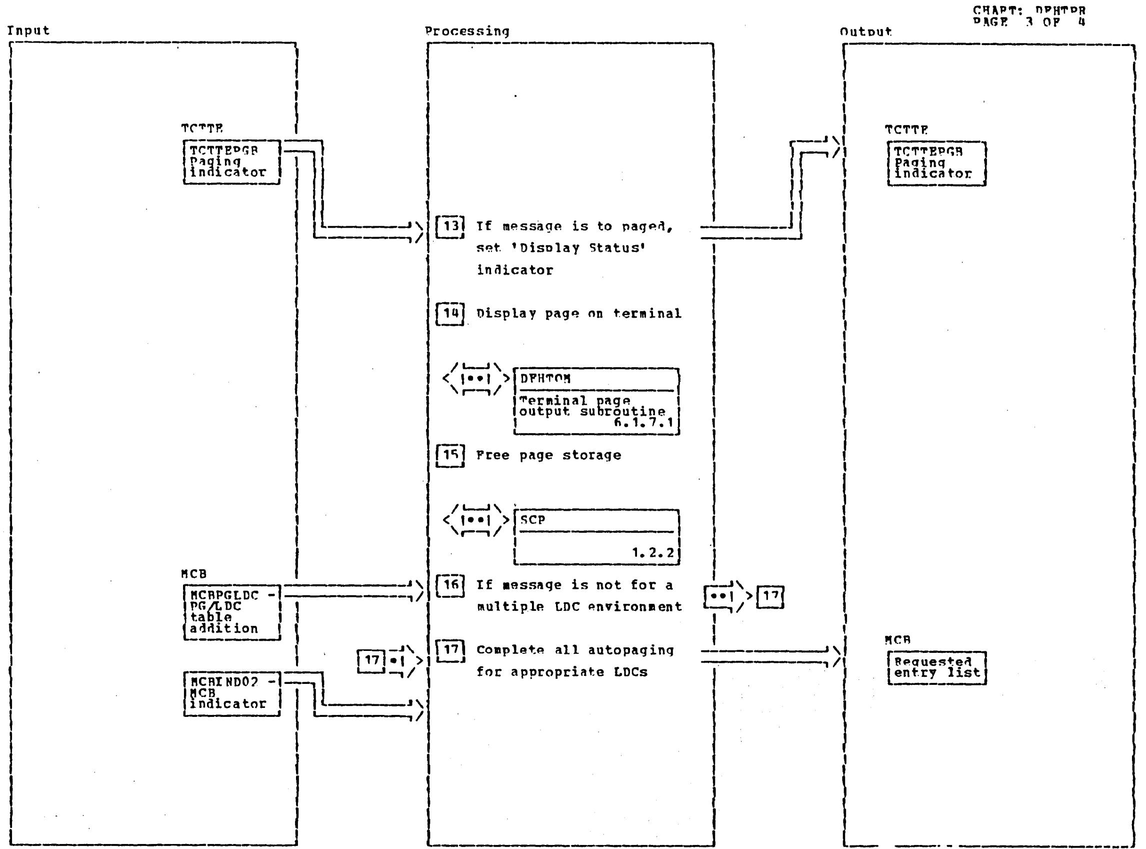


CHART: DFHTPR
PAGE 3 OF 4
Input
Processing
Output
TCTTE
TCTTEPGB
Paging indicator
13 If message is to paged, set 'Display Status' indicator
14 Display page on terminal
DFHTOM
Terminal page output subroutine 6.1.7.1
15 Free page storage
SCP
1.2.2
MCB
MCBPGLDC - PG/LDC table addition
16 If message is not for a multiple LDC environment
17
MCBIND02 - MCB indicator
17 Complete all autopaging for appropriate LDCs
TCTTE
TCTTEPGB
Paging indicator
MCB
Requested entry list

BMS: DFHTPR Page Retrieval and Display

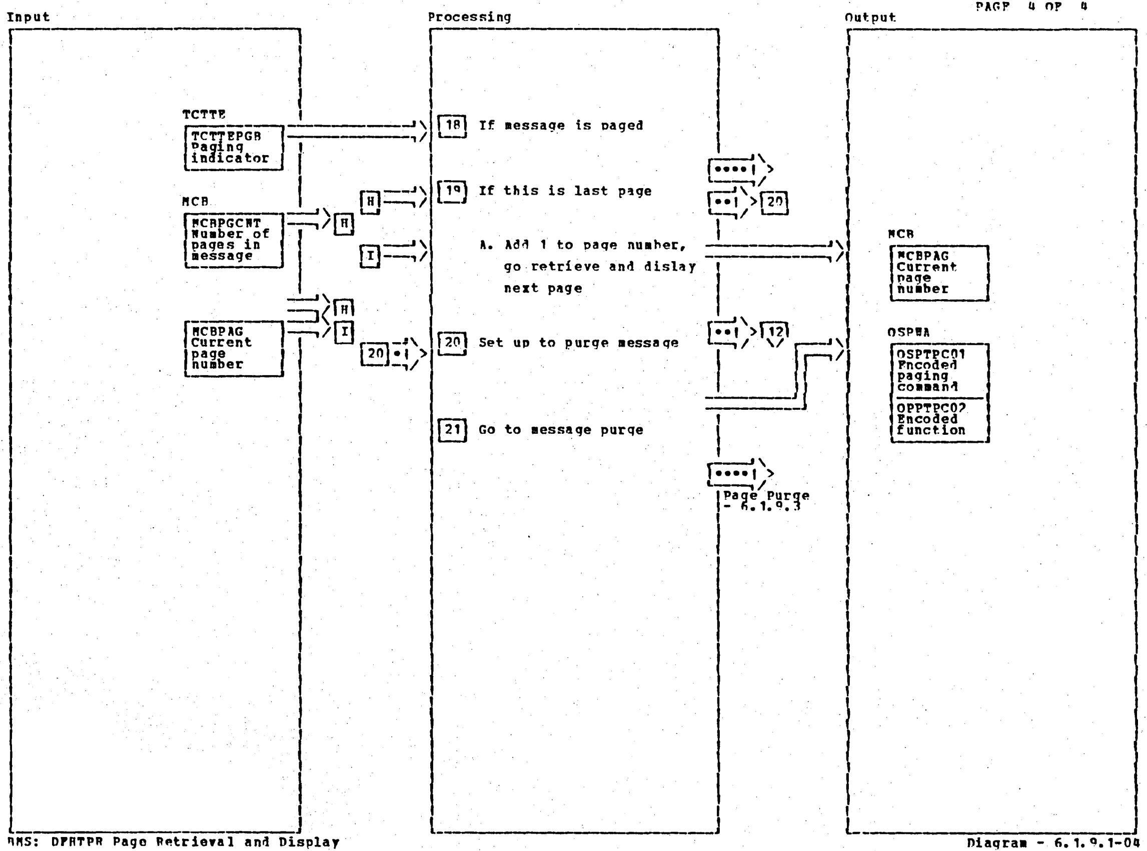
Diagram - 6.1.9.1-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 12 | | TPEXRET | | | | | |
| 15 This is an open subroutine. | | TPOUTPUT | | | | | |
| 16 | | TPRNPCHK | | | | | |
| 17 This routine saves information on the currently active LDC, completes messages for any and all autopaged LDCs, and notes if there are any paged messages left. | | | | | | | |

BMS: DFHTPR Page Retrieval and Display

Diagram - 6.1.9.1-03

CHART: DFHTPR
PAGE 4 OF 4



RMS: DFHTPR Page Retrieval and Display

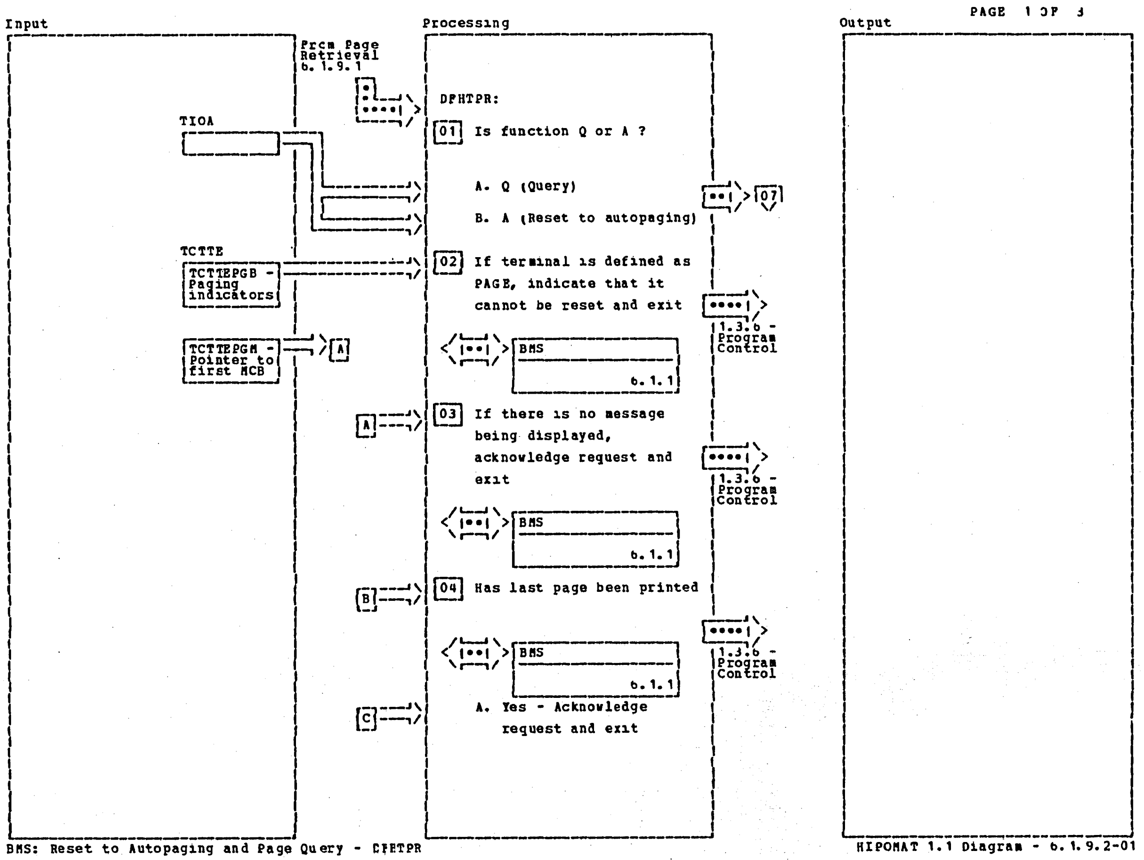
Diagram - 6.1.9.1-04

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 19 This step applies, if message is to be displayed one page at a time. | TPRNPCHK | | |
| 20 | | TPERRAUT | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

RMS: DFHTPR Page Retrieval and Display

Diagram - 6.1.9.1-04


PAGE 1 OF 3
Input
Processing
Output
Prcm Page Retrieval 6.1.9.1
DFHTPR:
TIOA
01 Is function Q or A ?
A. Q (Query)
B. A (Reset to autopaging)
07
TCTTE
TCTTEPGB - Paging indicators
02 If terminal is defined as PAGE, indicate that it cannot be reset and exit
1.3.6 - Program Control
TCTTEPGM - Pointer to first MCB
A
BMS
6.1.1
A
03 If there is no message being displayed, acknowledge request and exit
1.3.6 - Program Control
BMS
6.1.1
B
04 Has last page been printed
1.3.6 - Program Control
BMS
6.1.1
C
A. Yes - Acknowledge request and exit

BMS: Reset to Autopaging and Page Query - DFHTPR

HIPOMAT 1.1 Diagram - 6.1.9.2-01

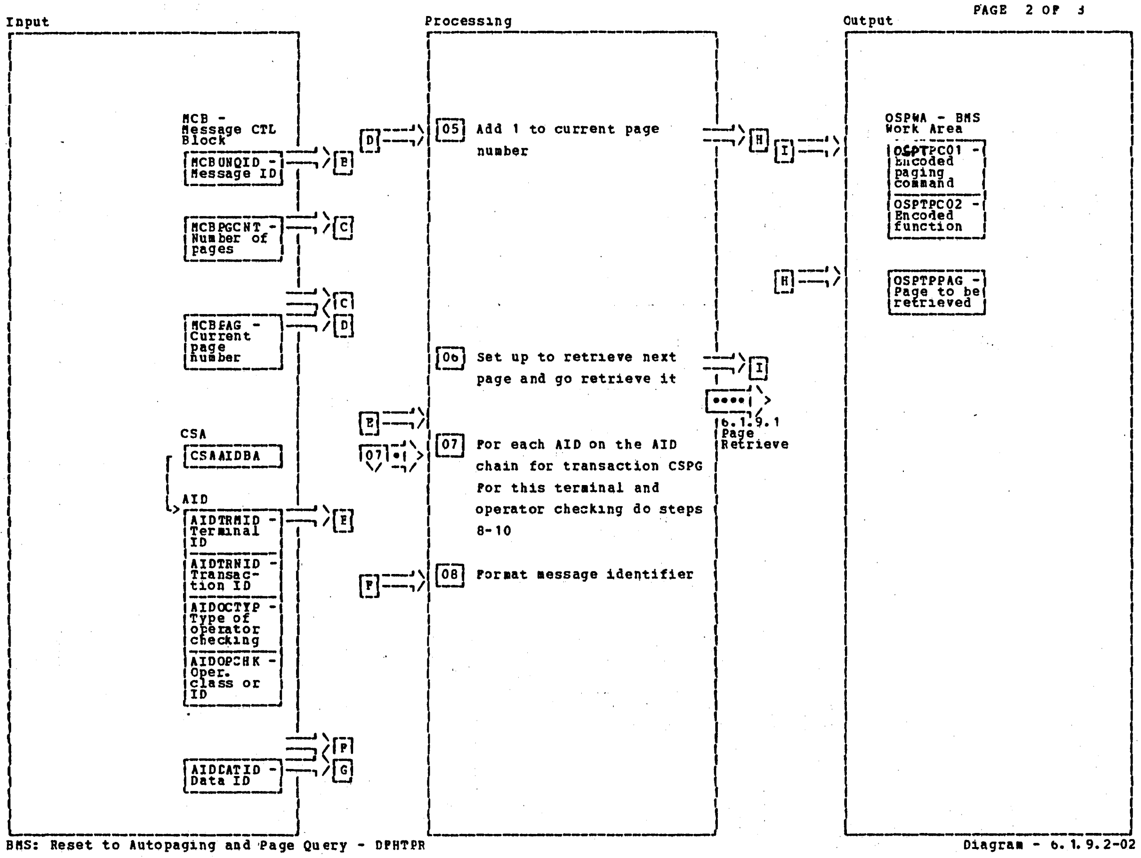
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 A message may have been sent to a hard-copy terminal with CTRL=PAGE. The message is retrieved by the operator 1 page at a time until the operator invokes this function. Then all the remaining pages are displayed. | TPRETA | | |
| 02 If the message is for a multiple LDC environment, all LDCs are checked for PAGE status before an error message is sent to the operator. | | TPRCKTRM | |
| 03 | | TPERETA | |
| 04 | | TPRCKIFW | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

BMS: Reset to Autopaging and Page Query - DFHTPR

Diagram - 6.1.9.2-01

PAGE 2 OF 3


Input
Processing
Output
MCB - Message CTL Block
MCBUNQID - Message ID
MCBPGCNT - Number of pages
MCBPAG - Current page number
CSA
CSAAIDBA
AID
AIDTRMID - Terminal ID
AIDTRNID - Transac-tion ID
AIDOCTYP - Type of operator checking
AIDOPCHK - Oper. class or ID
AIDDATID - Data ID
B
C
D
E
F
G
H
I
07
6.1.9.1 Page Retrieve
OSPWA - BMS Work Area
OSPTPC01 - Encoded paging command
OSPTPC02 - Encoded function
OSPTPPAG - Page to be retrieved

05 Add 1 to current page number

06 Set up to retrieve next page and go retrieve it

07 For each AID on the AID chain for transaction CSPG for this terminal and operator checking do steps 8-10

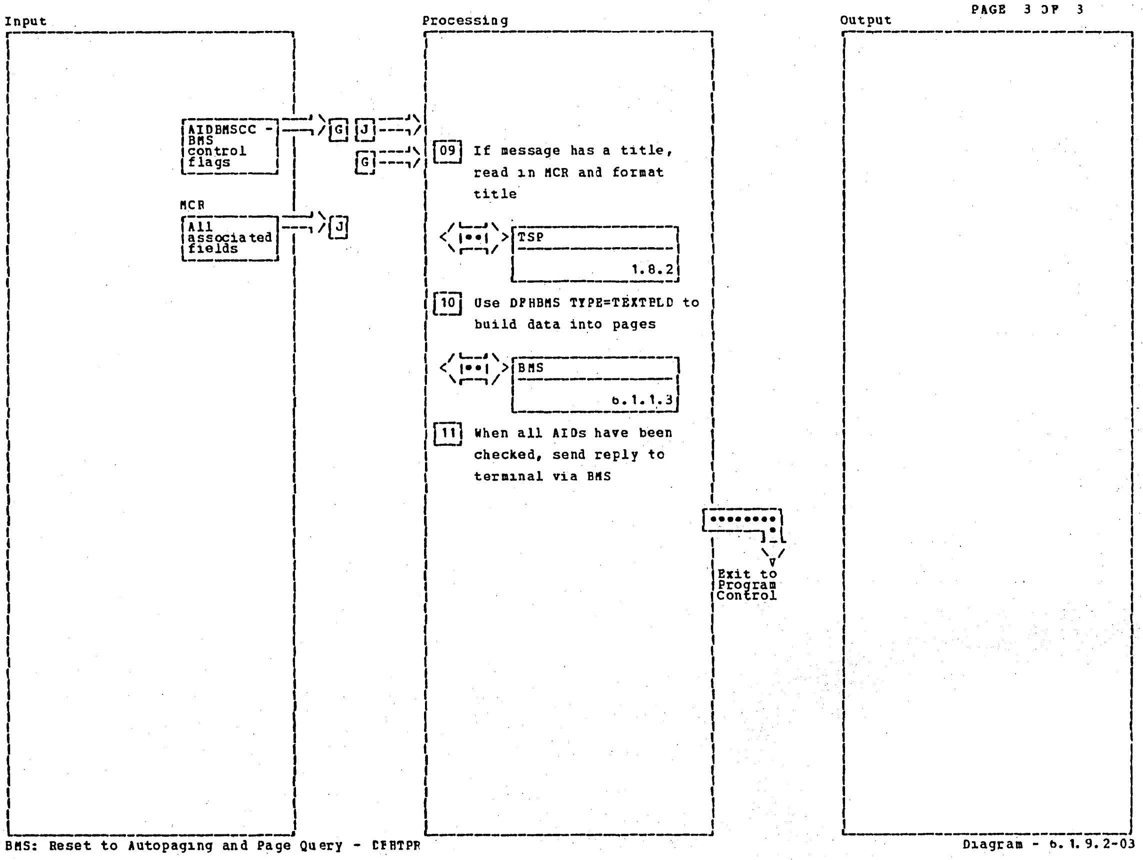
08 Format message identifier

BMS: Reset to Autopaging and Page Query - DFHTPR Diagram - 6.1.9.2-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 If the message is for multiple LDC environment, every LDC that can be reset to autopage status is processed, one at a time, until completion. | | TPRAPCK | | | | | |
| 07 | | TPRETQ | | | | | |
| 08 | | TPQDAID | | | | | |

BMS: Reset to Autopaging and Page Query - DFHTPR Diagram - 6.1.9.2-02

PAGE 3 OF 3



09 If message has a title, read in MCR and format title

10 Use DFHBMS TYPE=TEXTBLD to build data into pages

11 When all AIDs have been checked, send reply to terminal via BMS

BMS: Reset to Autopaging and Page Query - DFHTPR

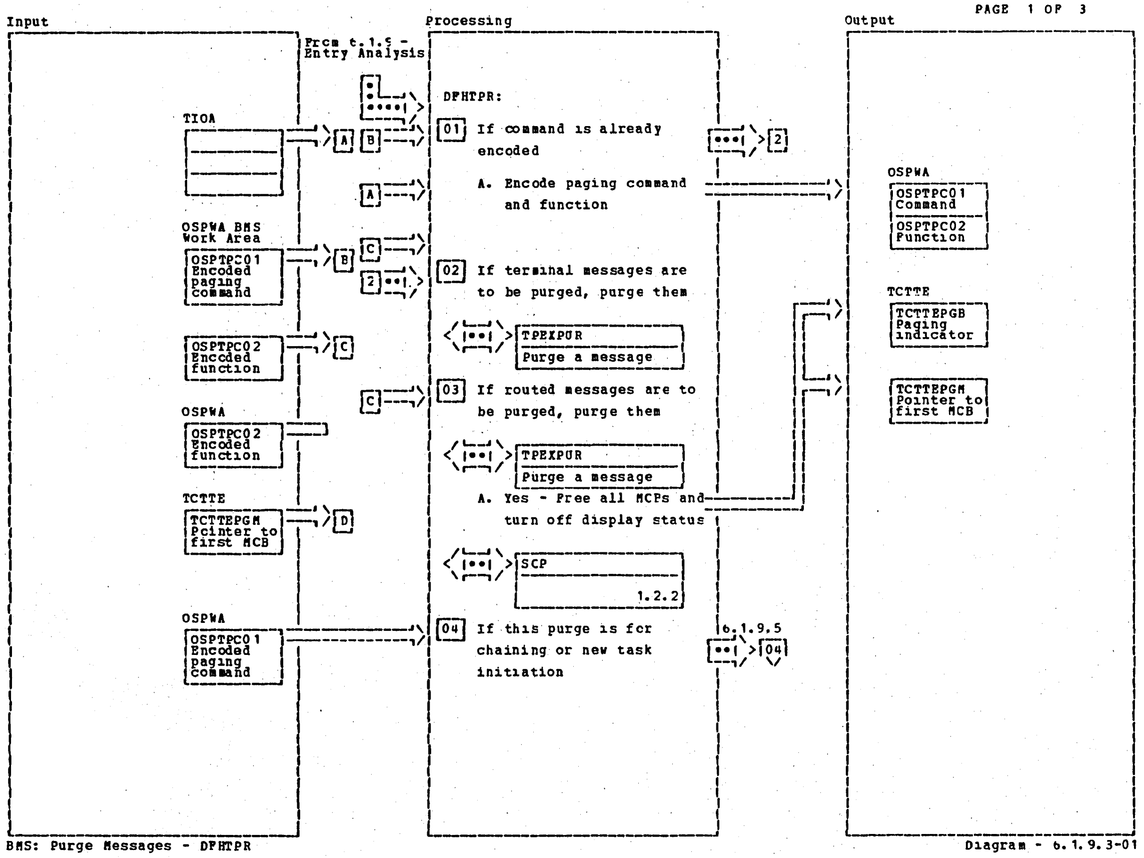
Diagram - 6.1.9.2-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 09 | | TPQMCRG | | | | | |
| 10 For 3270 only. For others, DFHBMS TYPE=OUT is used. | 3270<br>OTHER | TPQPUT32<br>TPQPUTL | | | | | |
| 11 | | TPQDISP | | | | | |

BMS: Reset to Autopaging and Page Query - DFHTPR

Diagram - 6.1.9.2-03

PAGE 1 OF 3


Input
Processing
Output
From 6.1.9 - Entry Analysis
TIOA
A
B
OSPWA BMS Work Area
OSPTPC01 Encoded paging command
OSPTPC02 Encoded function
C
2
OSPWA
OSPTPC02 Encoded function
TCTTE
TCTTEPGM Pointer to first MCB
D
OSPWA
OSPTPC01 Encoded paging command
OSPWA
OSPTPC01 Command
OSPTPC02 Function
TCTTE
TCTTEPGB Paging indicator
TCTTEPGM Pointer to first MCB
6.1.9.5
04

DFHTPR:

01 If command is already encoded

A. Encode paging command and function

02 If terminal messages are to be purged, purge them

TPEXPUR — Purge a message

03 If routed messages are to be purged, purge them

TPEXPUR — Purge a message

A. Yes - Free all MCPs and turn off display status

SCP — 1.2.2

04 If this purge is for chaining or new task initiation

BMS: Purge Messages - DFHTPR

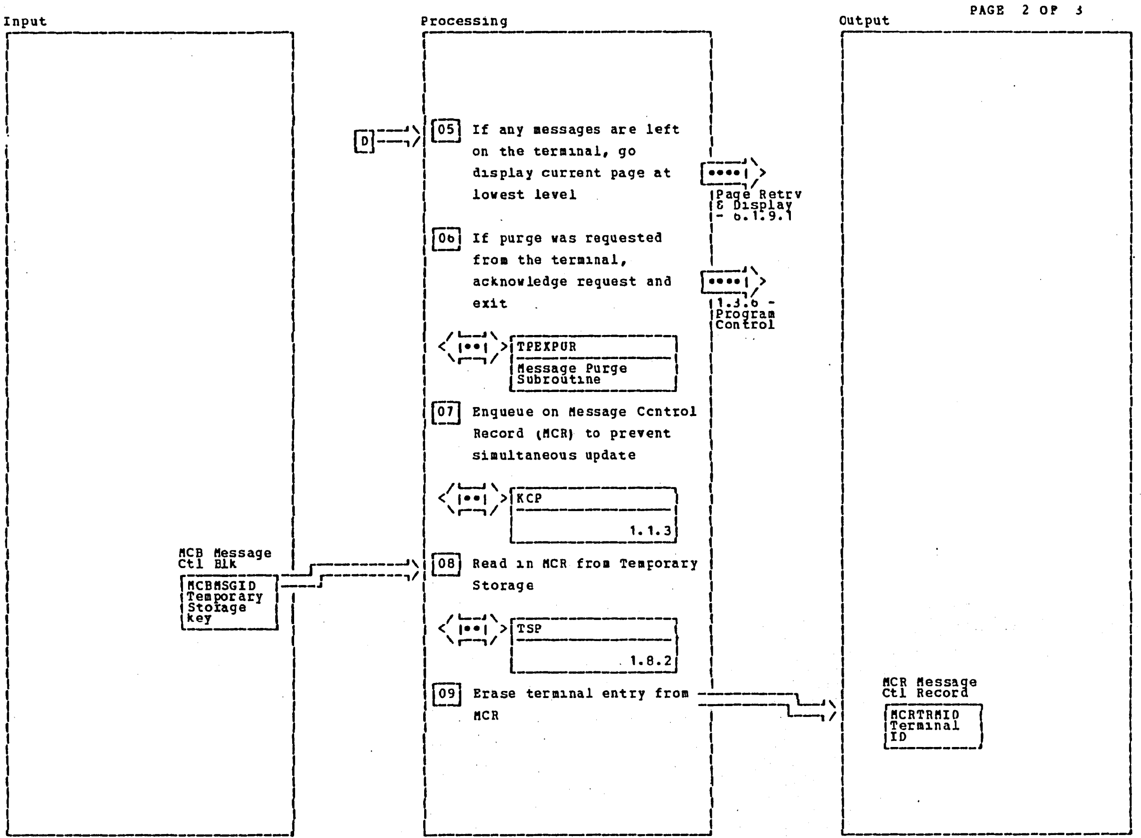
Diagram - 6.1.9.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 This routine is charted in steps 8 thru 12 of this diagram. | | TPPTERM | | | | | |
| 03 This routine is charted in steps 8 thru 12 of this diagram. | | TPPROUTN | | | | | |
| 04 If all messages are purged from terminal, all MCBs are freed. Otherwise they are left for use by succeeding chained transactions, if any. | TPRCUTE | | | | | | |

BMS: Purge Messages - DFHTPR

Diagram - 6.1.9.3-01

PAGE 2 OF 3



05 If any messages are left on the terminal, go display current page at lowest level

06 If purge was requested from the terminal, acknowledge request and exit

07 Enqueue on Message Control Record (MCR) to prevent simultaneous update

08 Read in MCR from Temporary Storage

09 Erase terminal entry from MCR

BMS: Purge Messages - DPHTPR

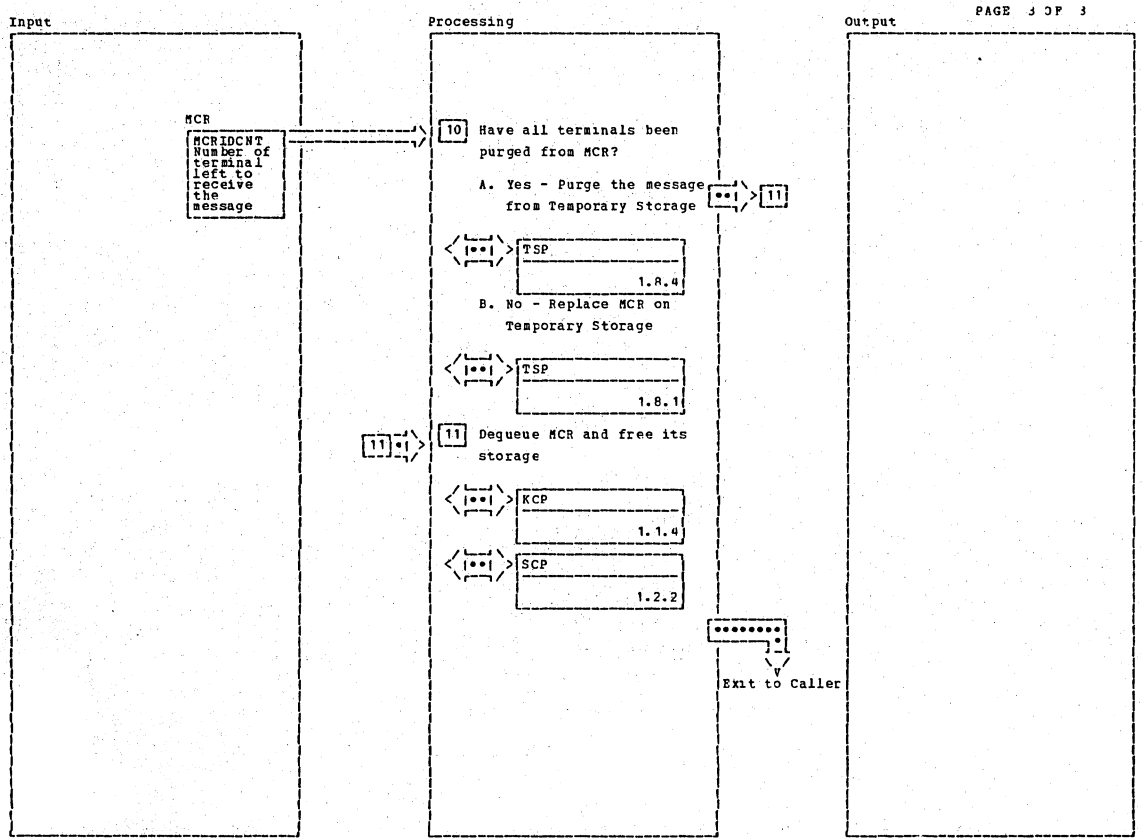
Diagram - 6.1.9.3-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 08 | TPEXEOR | TPEXPUR | | | | | |
| 10 | | TPPUR1<br>TPPTRMPR | | | | | |

BMS: Purge Messages - DPHTPR

Diagram - 6.1.9.3-02

PAGE 3 OF 3



10 Have all terminals been purged from MCR?

A. Yes - Purge the message from Temporary Storage

B. No - Replace MCR on Temporary Storage

11 Dequeue MCR and free its storage

BMS: Purge Messages - DFHTPR

Diagram - 6.1.9.3-03

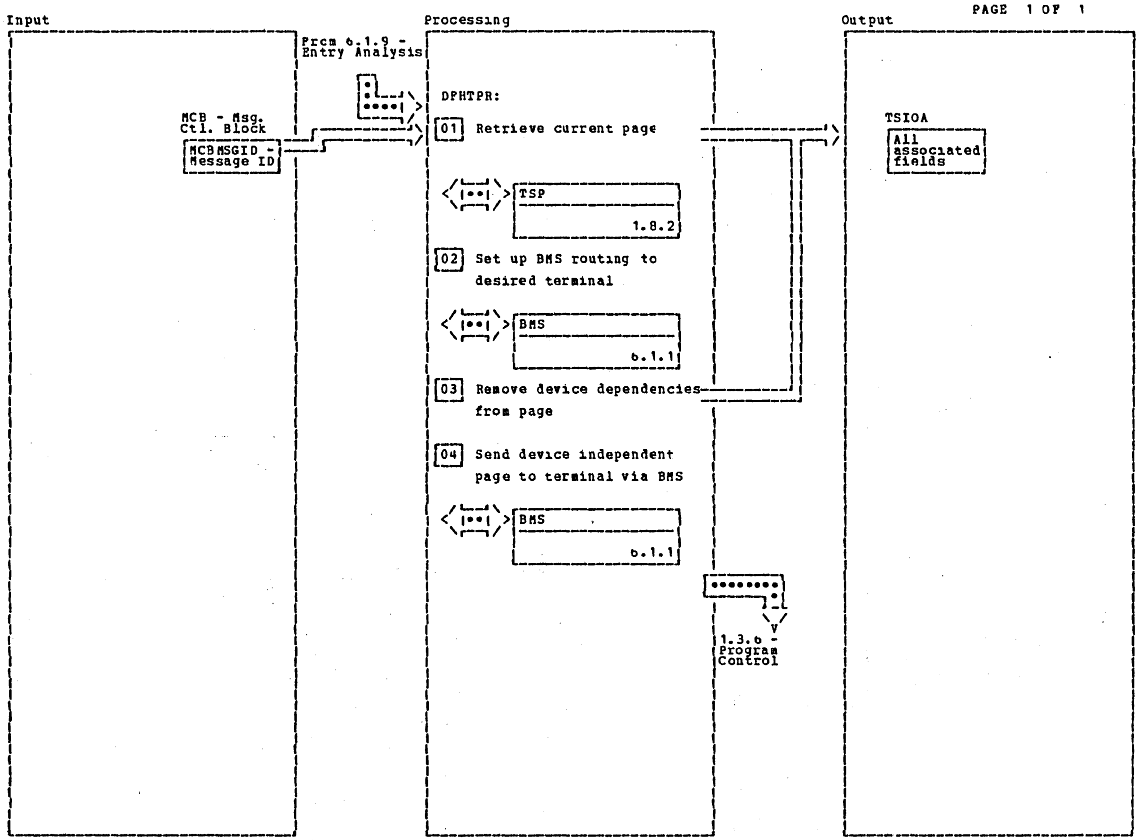
| Notes | Routine | Label | Ref |
|---|---|---|---|
| 11 | A | TPPTSPUR | |
| | B | TPPUPMCR | |

| Notes | Routine | Label | Ref |
|---|---|---|---|

BMS: Purge Messages - DFHTPR

Diagram - 6.1.9.3-03

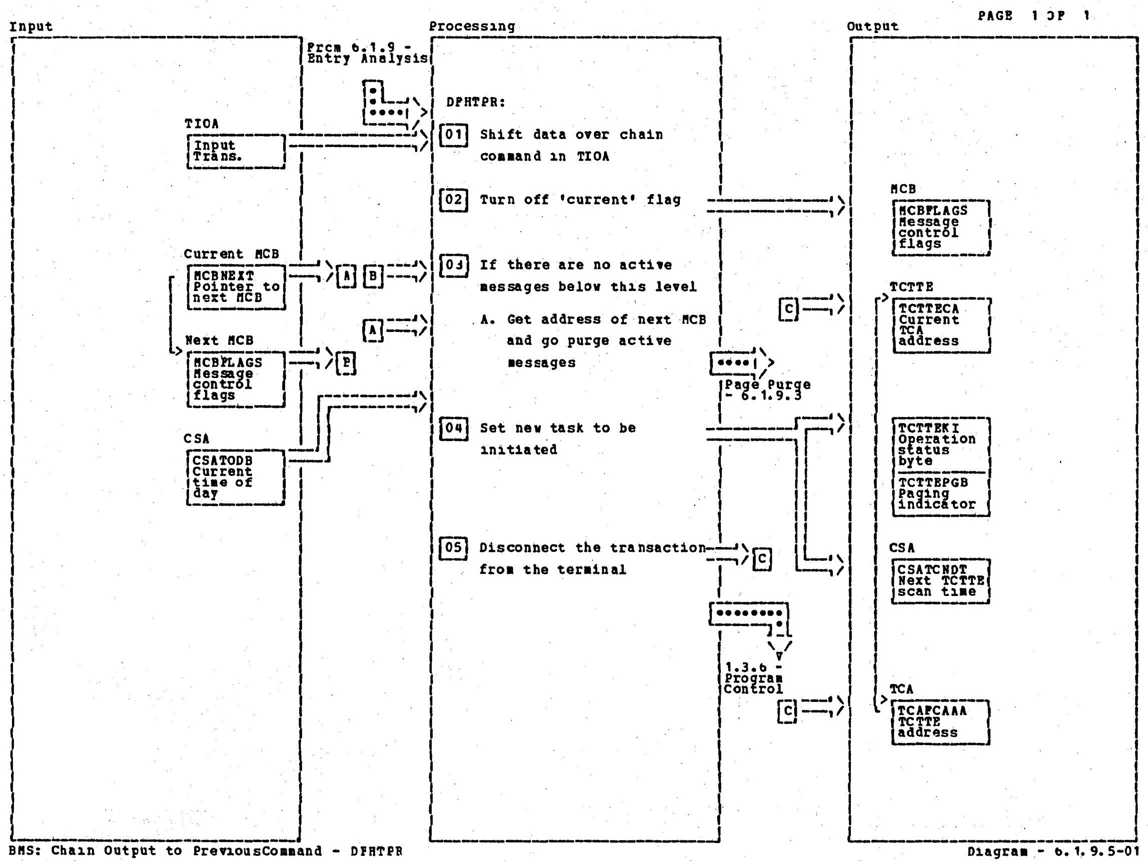
PAGE 1 OF 1



BMS: Copy a Page to Another Terminal

Diagram - 6.1.9.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | TPENCCOP | TPENCCOP | | | | | |
| 02 | | TPECMCBY | | | | | |
| 03 If copy is to go to identical model 3270, page is sent to terminal with device dependencies via DFHBMS TYPE=(STORE,NOEDIT). | | IIFORMAT | | | | | |
| 04 | | IIINDUN1 | | | | | |


PAGE 1 OF 1
Input
Processing
Output
From 6.1.9 - Entry Analysis
DFHTPR:
TIOA
Input Trans.
01 Shift data over chain command in TIOA
02 Turn off 'current' flag
MCB
MCBFLAGS Message control flags
Current MCB
MCBNEXT Pointer to next MCB
A
B
03 If there are no active messages below this level
A. Get address of next MCB and go purge active messages
Next MCB
MCBFLAGS Message control flags
Page Purge - 6.1.9.3
TCTTE
TCTTECA Current TCA address
CSA
CSATODB Current time of day
04 Set new task to be initiated
TCTTEKI Operation status byte
TCTTEPGB Paging indicator
05 Disconnect the transaction from the terminal
C
CSA
CSATCNDT Next TCTTE scan time
1.3.6 - Program Control
TCA
TCAFCAAA TCTTE address

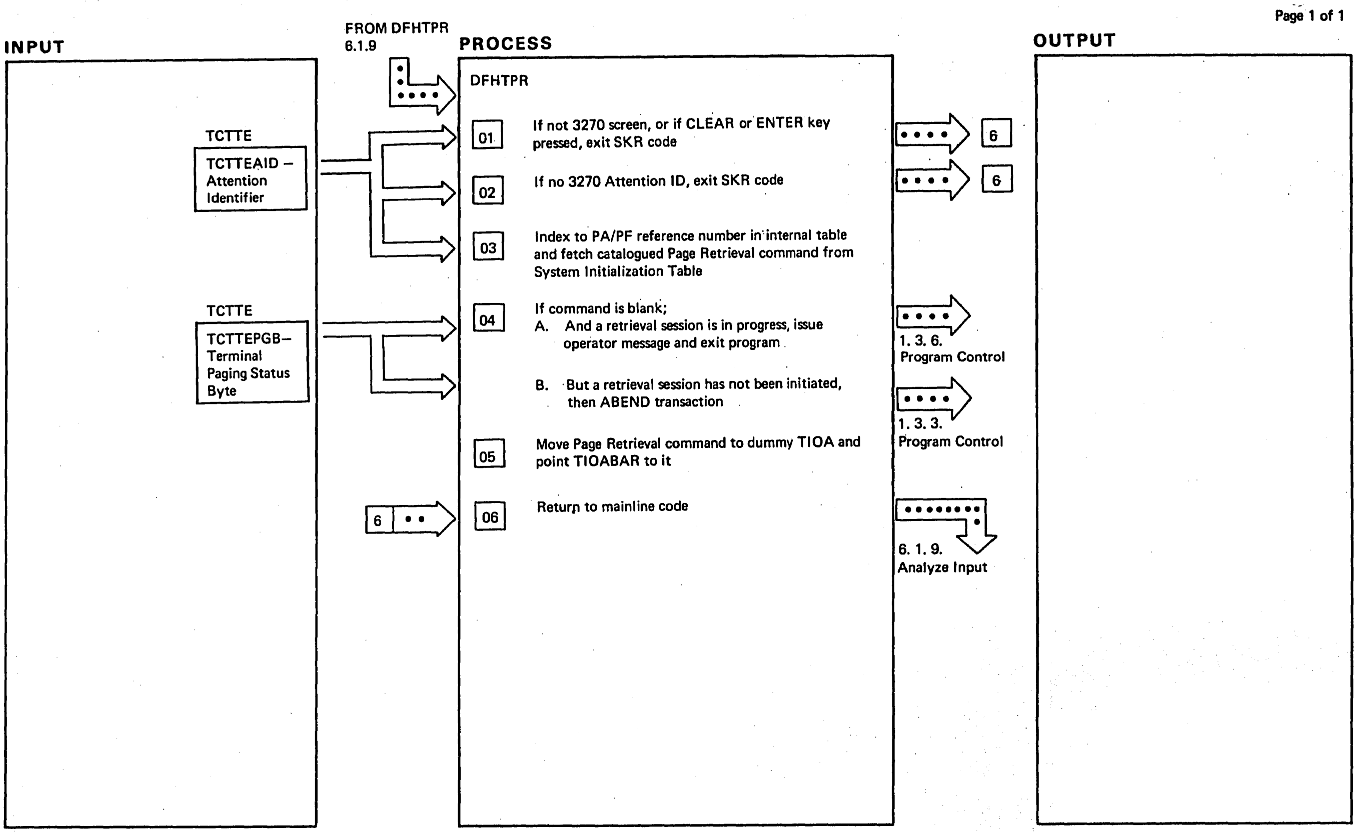
BMS: Chain Output to PreviousCommand - DFHTPR

Diagram - 6.1.9.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | TPENCCHN | TPENCCHN | | | | | |
| 02 | | TPENEXEC | | | | | |
| 03 | | TPNEODLP | | | | | |
| 04 | | TPNNINIT | | | | | |
| 05 A new scan of the TCTTE is performed and the changed transaction is initiated. | | | | | | | |

BMS: Chain Output to PreviousCommand - DFHTPR

Diagram - 6.1.9.5-01



DFHTPR

01 If not 3270 screen, or if CLEAR or ENTER key pressed, exit SKR code

02 If no 3270 Attention ID, exit SKR code

03 Index to PA/PF reference number in internal table and fetch catalogued Page Retrieval command from System Initialization Table

04 If command is blank;
A. And a retrieval session is in progress, issue operator message and exit program.
B. But a retrieval session has not been initiated, then ABEND transaction

05 Move Page Retrieval command to dummy TIOA and point TIOABAR to it

06 Return to mainline code

BMS: Page Retrieval Program – DFHTPR: Single Keystroke Retrieval

Diagram 6.1.9.6–01

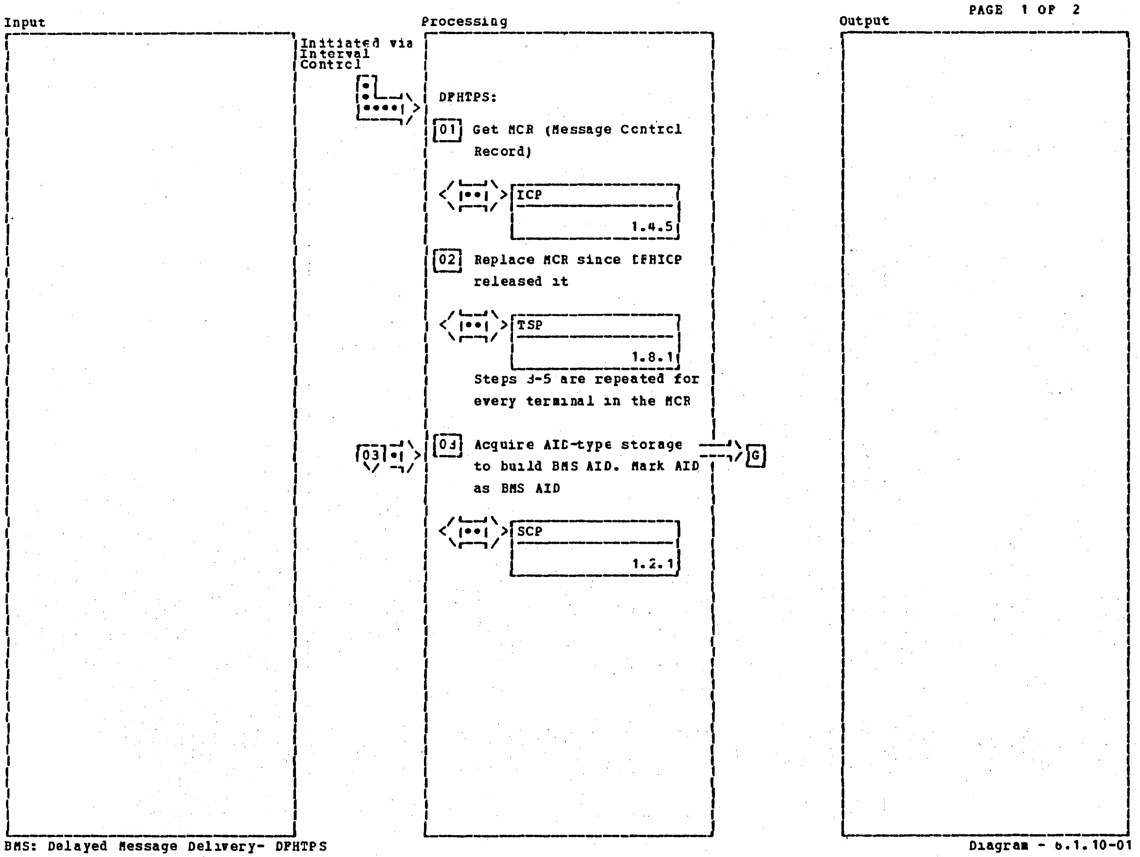
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

BMS: Page Retrieval Program – DFHTPR: Single Keystroke Retrieval

Diagram 6.1.9.6–01

PAGE 1 OF 2



Input

Processing

Output

Initiated via Interval Control

DFHTPS:

01 Get MCR (Message Control Record)

ICP 1.4.5

02 Replace MCR since DFHICP released it

TSP 1.8.1

Steps 3-5 are repeated for every terminal in the MCR

03 Acquire AID-type storage to build BMS AID. Mark AID as BMS AID

G

SCP 1.2.1

BMS: Delayed Message Delivery- DFHTPS

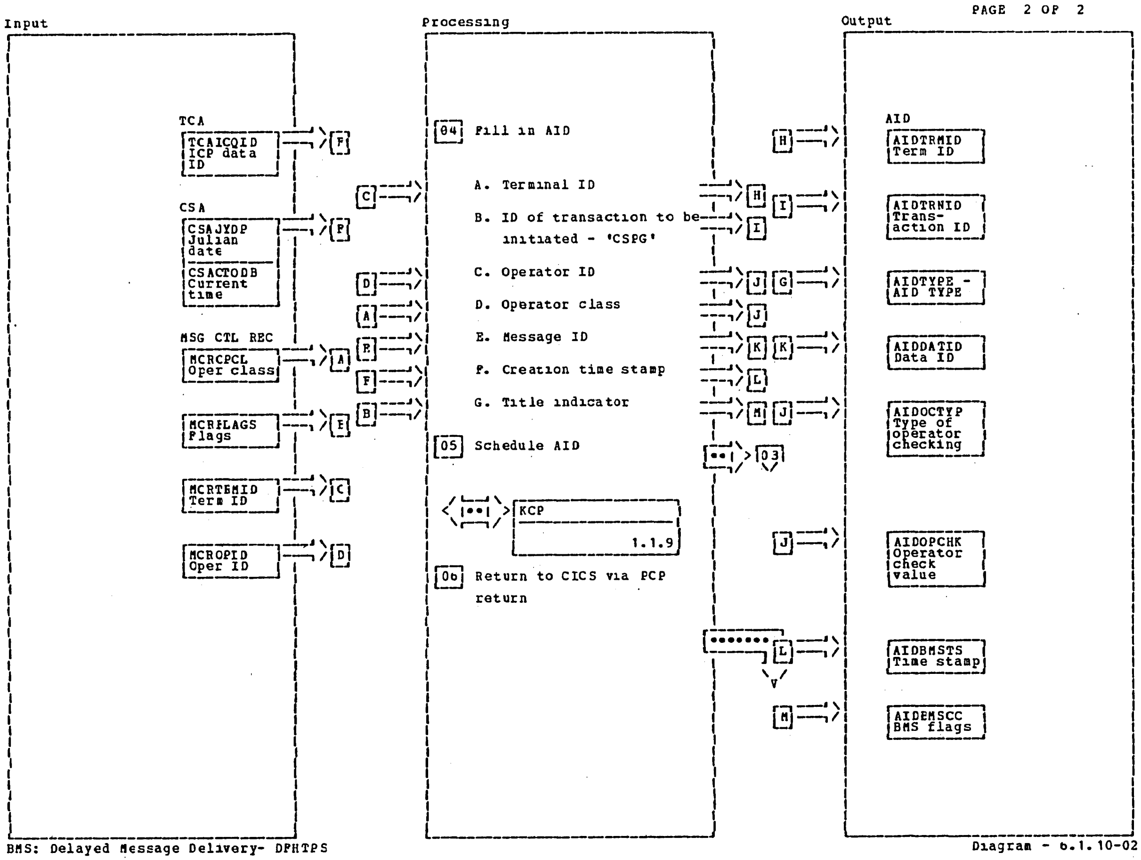
Diagram - 6.1.10-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 MCR is stored on Temporary Storage and this transaction is initiated by DFHMCP via DFHIC TYPE=PUT. Retrieval is via DFHIC TYPE=GET which releases MCR from Temporary Storage. | | | | | | | |
| 02 MCR is replaced on Temporary Storage via DFHTS TYPE=PUT. | | | | | | | |

BMS: Delayed Message Delivery- DFHTPS

Diagram - 6.1.10-01

PAGE 2 OF 2



BMS: Delayed Message Delivery- DFHTPS

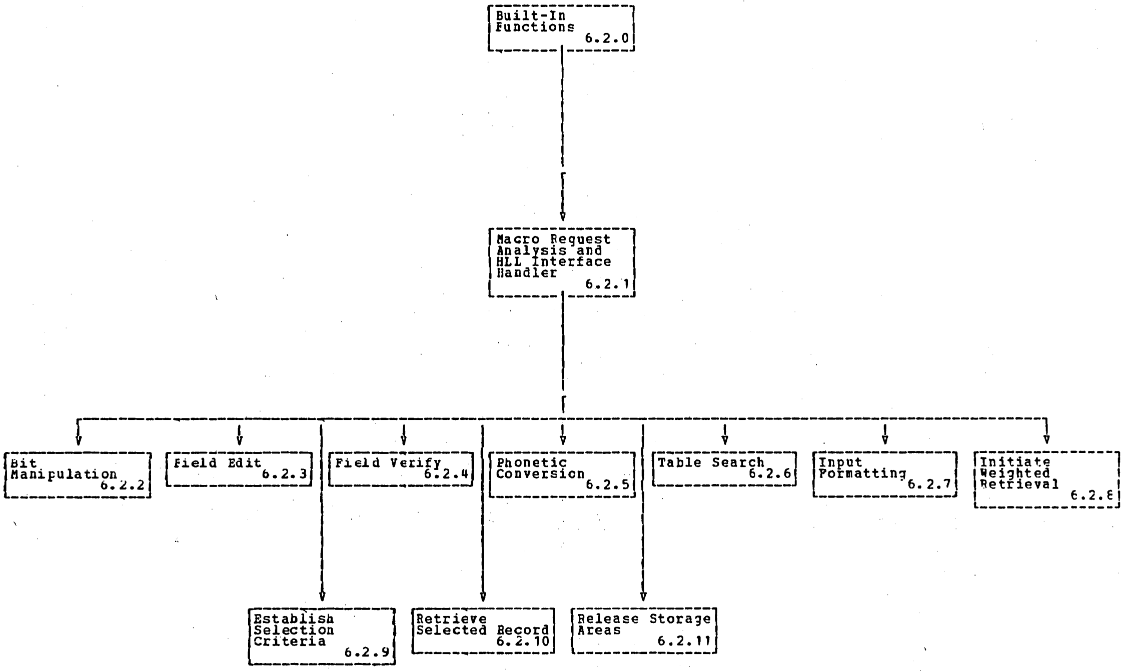
Diagram - 6.1.10-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 05 AID is scheduled via DFHKC TYPE=SCHEDULE. | | | | | | | |

BMS: Delayed Message Delivery- DFHTPS

Diagram - 6.1.10-02

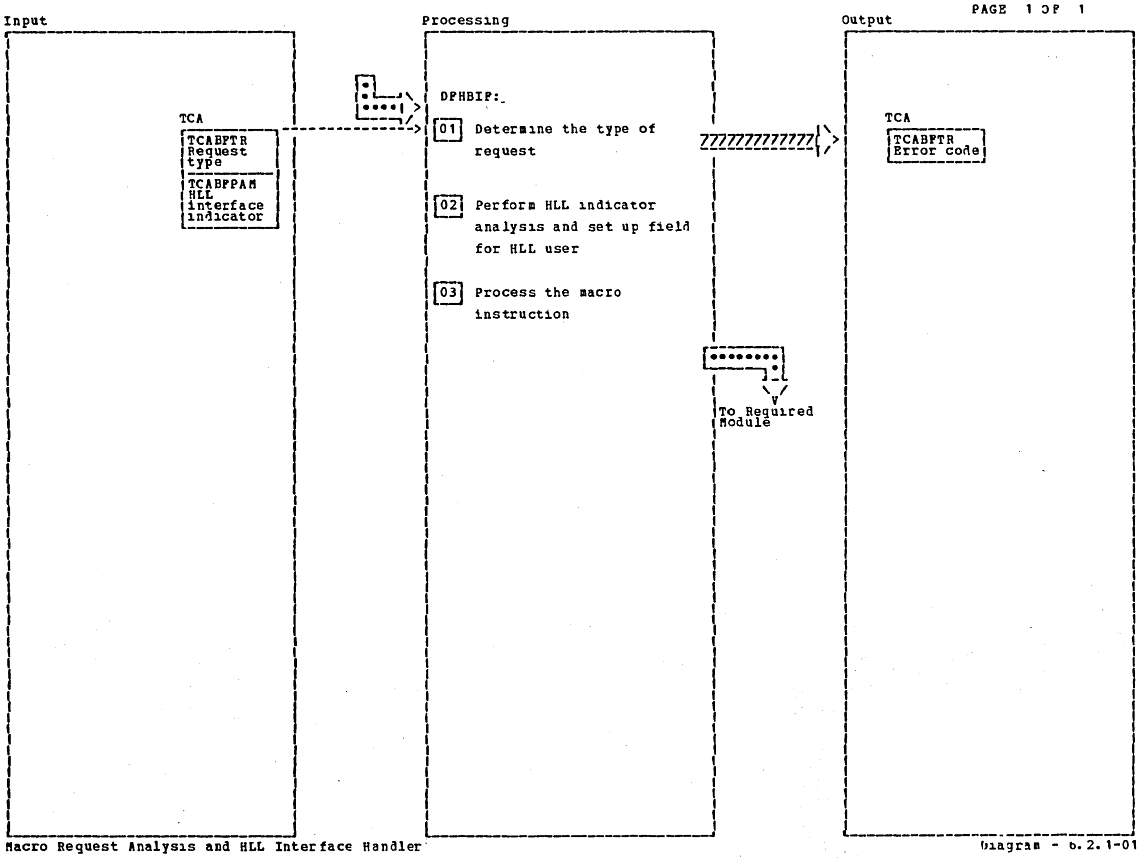
PAGE 1 OF 1


Built-In Functions 6.2.0
Macro Request Analysis and HLL Interface Handler 6.2.1
Bit Manipulation 6.2.2
Field Edit 6.2.3
Field Verify 6.2.4
Phonetic Conversion 6.2.5
Table Search 6.2.6
Input Formatting 6.2.7
Initiate Weighted Retrieval 6.2.8
Establish Selection Criteria 6.2.9
Retrieve Selected Record 6.2.10
Release Storage Areas 6.2.11

Built-In-Functions

Diagram - 6.2-01

PAGE 1 OF 1



01 Determine the type of request

02 Perform HLL indicator analysis and set up field for HLL user

03 Process the macro instruction

Macro Request Analysis and HLL Interface Handler

Diagram - 6.2.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 Initialize address pointers for HLL user. | | | | | | | |
| Initialize TCABITF if the Bit Manipulation macro instruction was given. | | BFHLBIT | | | | | |
| Initialize TCAFLD if DEEDIT was specified. | | BFHLEDT | | | | | |
| Initialize TCAINA1 and TCAINA2 if INFORMAT was specified. | | BFHLINF | | | | | |
| Initialize TCACKFD if FVERIFY was specified. | | BFHLCK | | | | | |
| Initialize TCATSA1, TCATSA2, TCATSA3, TCATSA4, and TCATSA5 if TSEARCH was specified. | | BFHLTSH | | | | | |
| Initialize TCAWTRI if WTRETST was specified. | | BFHLWRT | | | | | |
| Initialize TCAWPA1, TCAWPA3, and TCAWPA4 if WTRTPARM was specified. | | BFHLWPM | | | | | |

Macro Request Analysis and HLL Interface Handler

Diagram - 6.2.1-01

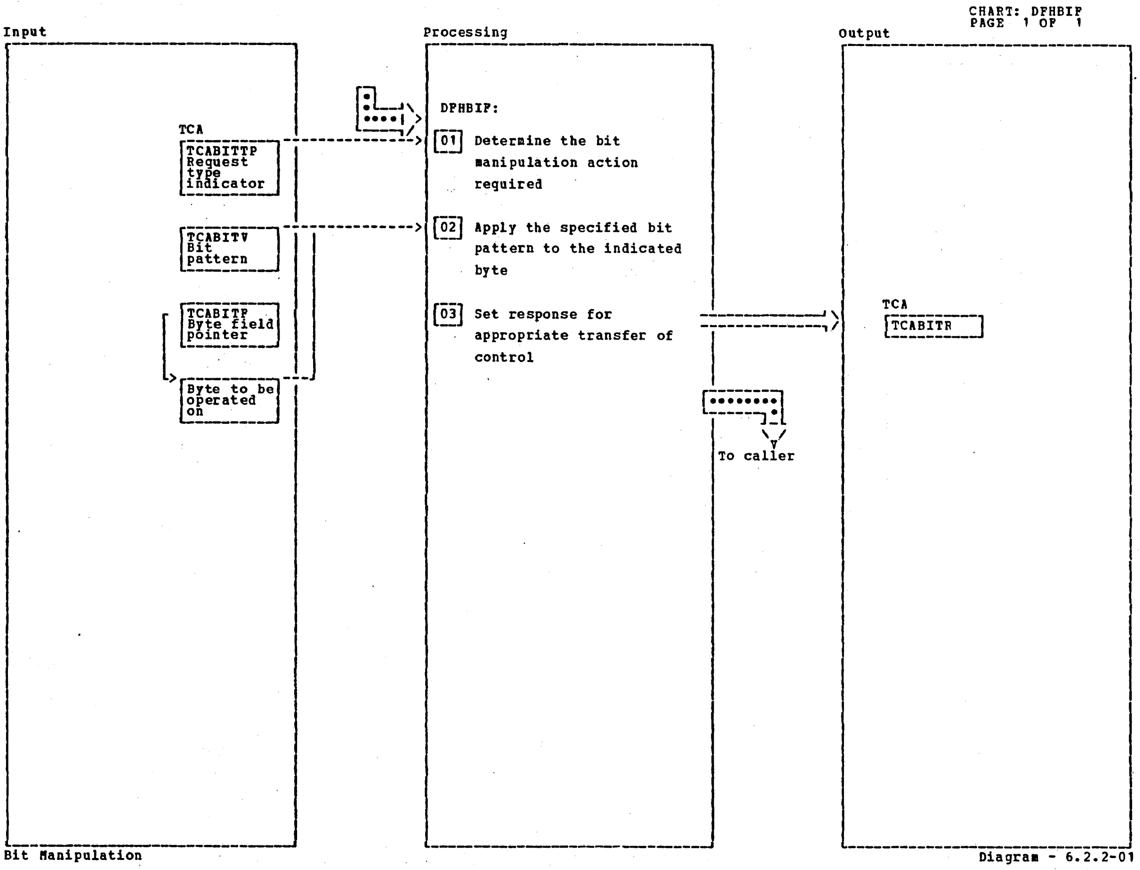


CHART: DFHBIF
PAGE 1 OF 1

Input

TCA

TCABITTP Request type indicator

TCABITV Bit pattern

TCABITF Byte field pointer

Byte to be operated on

Processing

DFHBIF:

01 Determine the bit manipulation action required

02 Apply the specified bit pattern to the indicated byte

03 Set response for appropriate transfer of control

To caller

Output

TCA

TCABITR

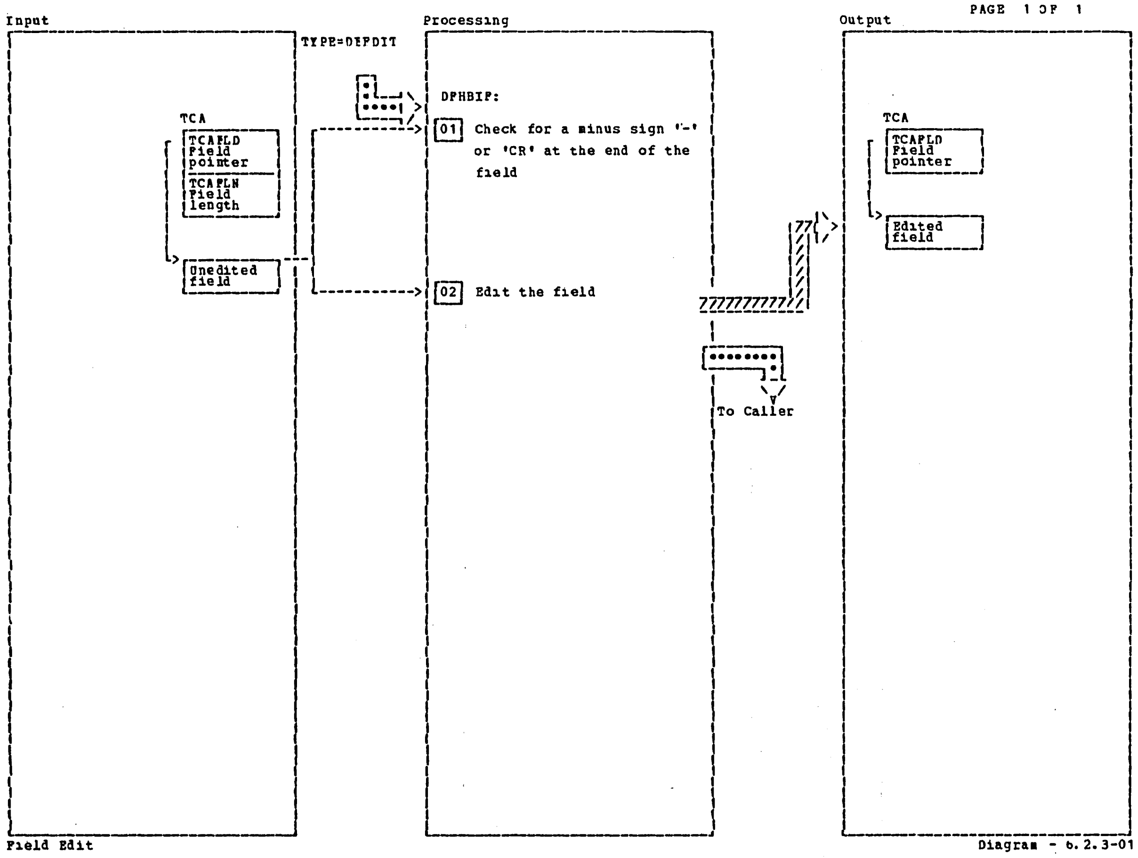
Bit Manipulation

Diagram - 6.2.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The allowable operands are BITSETON, BITSETOFF, BITFLIP, and BITEST. | | | | | | | |
| 02 BITSETON | | BIFTST4 | | | | | |
| BITSETOFF | | BIFTST8 | | | | | |
| BITFLIP | | BIFTSTC | | | | | |
| BITEST | | BIFTST0 | | | | | |
| 03 For BITSETON, BITSETOFF, of BITFLIP, all bits in the specified byte are on (or off) after the operation, and control is transferred to the address specified by BITON= (or BITOFF=) operand. | | | | | | | |
| 04 For BITEST, all bits that are tested under bit pattern mask, are on (of off), and control is transferred to the address specified by BITON= (or BITOFF=) operand. | | | | | | | |

Bit Manipulation

Diagram - 6.2.2-01



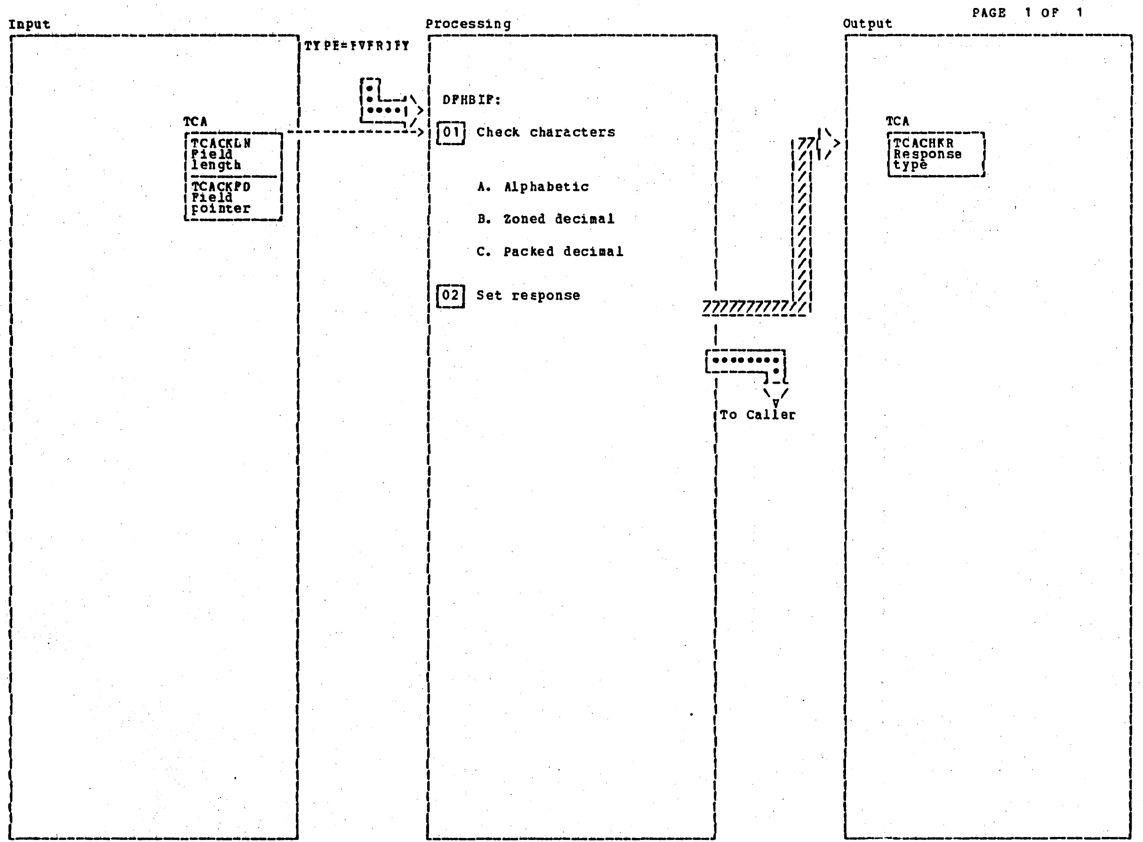
Field Edit

Diagram - 6.2.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 If the field ends with a minus sign or 'CR', a negative zone is placed over the low-order byte. | | BIFDED0 | | | | | |
| 02 All non-numeric characters are removed. | | BIFDED4 | | | | | |

Field Edit

Diagram - 6.2.3-01



Field Verify

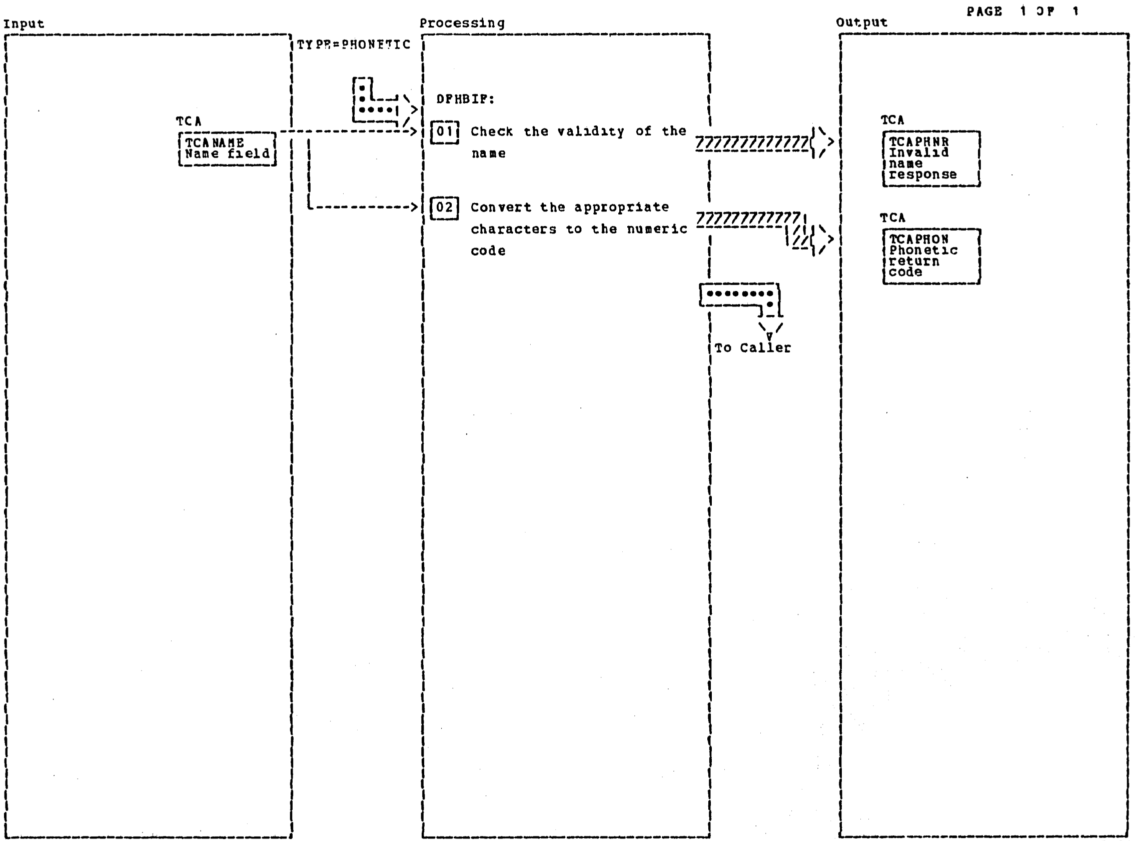
Diagram - 6.2.4-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| The field function determines what type of data is to be processed. Control is transferred to the appropriate routine in the application program based on the type of data. | | | | | | | |
| Character check | | | | | | | |
| A. Alphabetic | | BIFCHK4 | | | | | |
| B. Zoned decimal | | BIFCHK8 | | | | | |
| C. Packed decimal | | BIFCHKC | | | | | |

Field Verify

Diagram - 6.2.4-01

PAGE 1 OF 1



Phonetic Conversion

Diagram - 6.2.5-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The first character must be alphabetic. | | BIFPHN0 | | | | | |
| 02 The first character remains unchanged. The Translate Table is used to convert the character to numeric code. | | BIFPHN2 | | | | | |

Phonetic Conversion

Diagram - 6.2.5-01

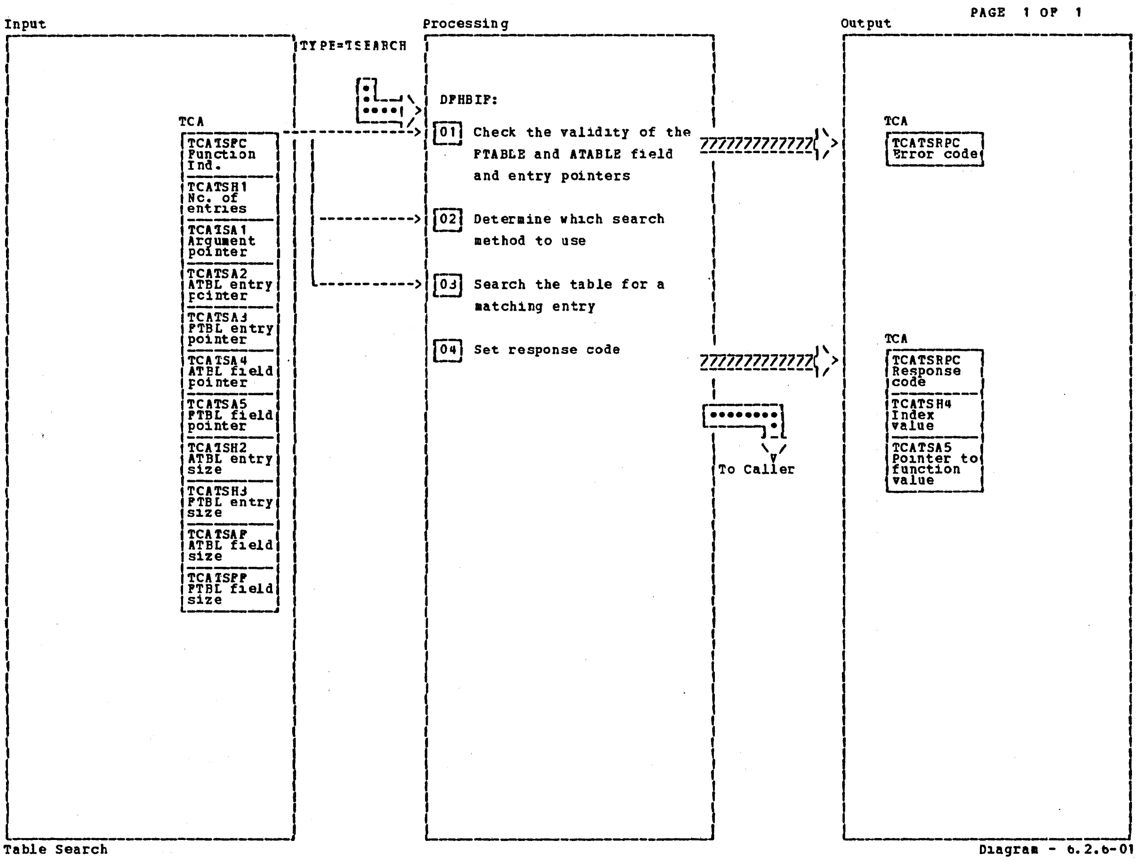


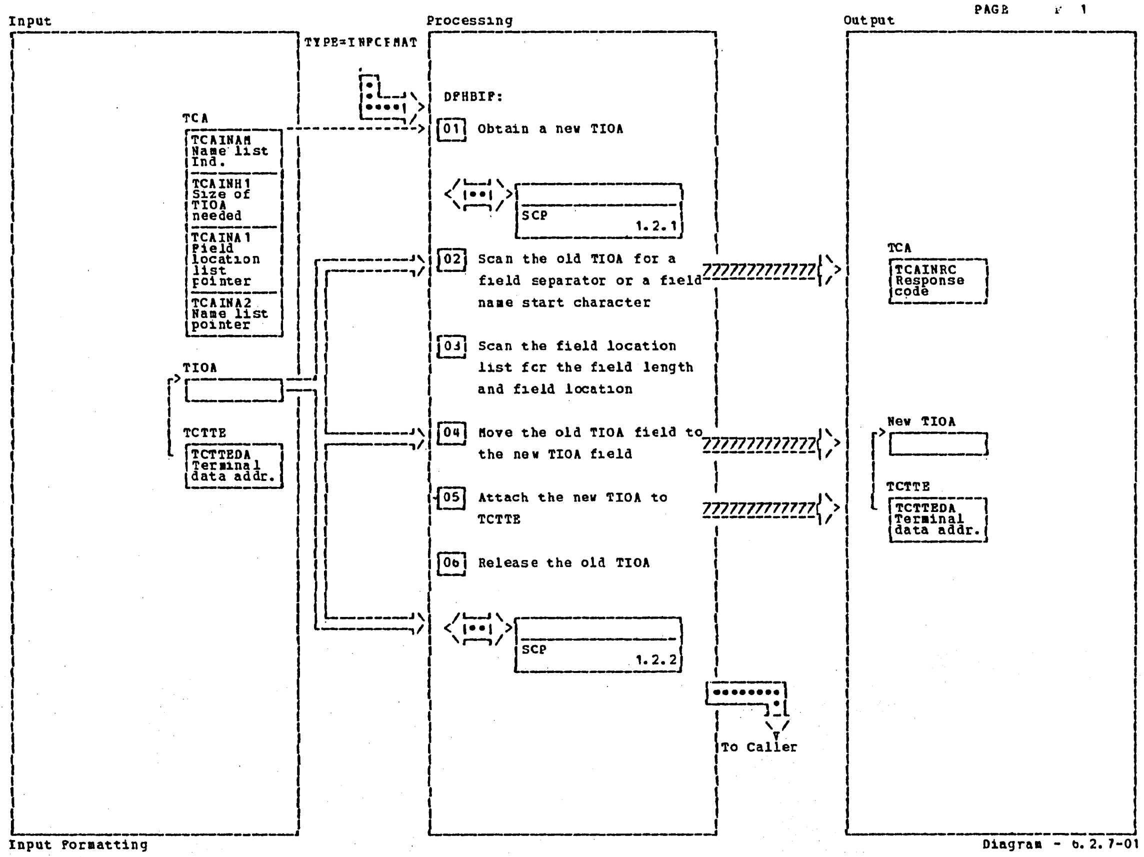
Table Search

Diagram - 6.2.6-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | BIFTSH | | | | | |
| 02 A linear search is made if the ORDER= operand is not specified. | | BIFTSH0 | | | | | |
| A binary search is made if the ORDER= operand is specified. | | | | | | | |
| 03 Linear search. | | BIFTSH2 | | | | | |
| Binary search. | | BIFTSH4 | | | | | |
| 04 A match or no match value is returned. | | BIFTSH6 | | | | | |

Table Search

Diagram - 6.2.6-01


Input
Processing
Output
TYPE=INPCFMAT
DFHBIF:
TCA
TCAINAM Name list Ind.
TCAINH1 Size of TIOA needed
TCAINA1 Field location list pointer
TCAINA2 Name list pointer
TIOA
TCTTE
TCTTEDA Terminal data addr.
01 Obtain a new TIOA
SCP 1.2.1
02 Scan the old TIOA for a field separator or a field name start character
03 Scan the field location list fcr the field length and field location
04 Move the old TIOA field to the new TIOA field
05 Attach the new TIOA to TCTTE
06 Release the old TIOA
SCP 1.2.2
To Caller
TCA
TCAINRC Response code
New TIOA
TCTTE
TCTTEDA Terminal data addr.

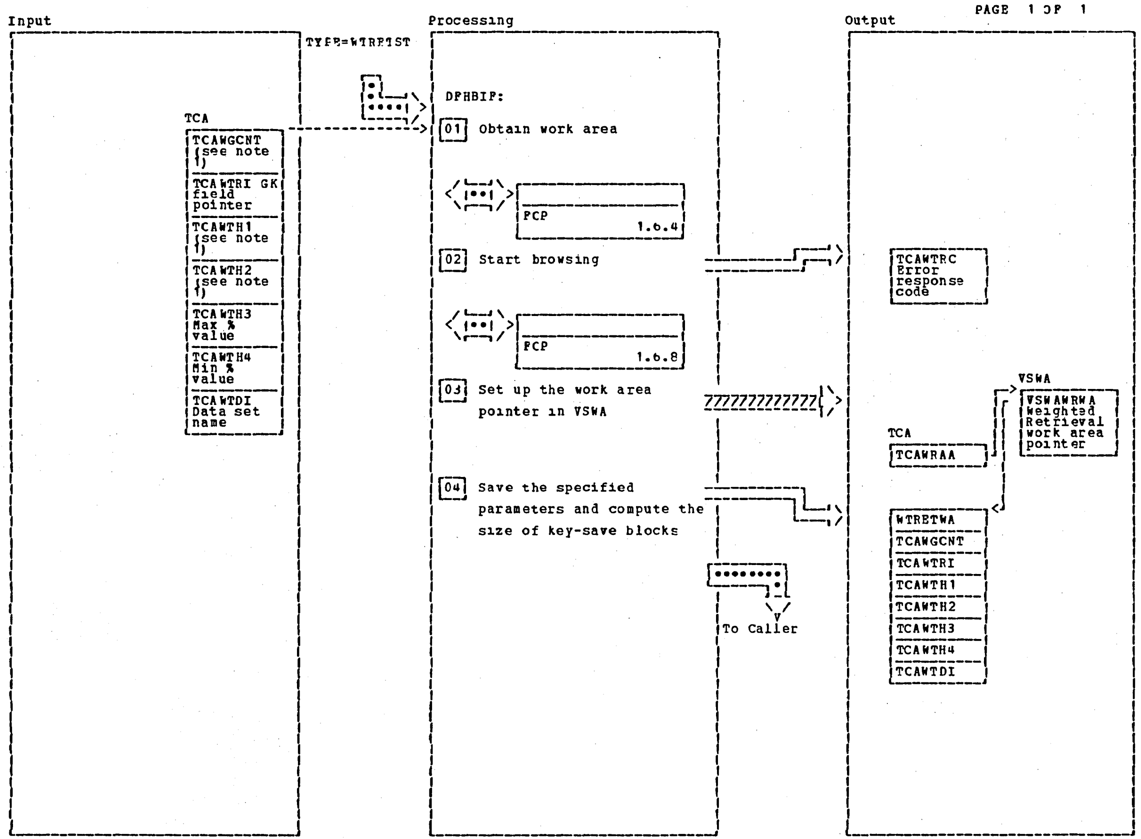
Input Formatting

Diagram - 6.2.7-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | BIFINF | | | | | |
| 02 | BIFINF30 | BIFINF2 | | | | | |
| 03 | BIFINF40 | | | | | | |
| 04 | BIFINF20 | | | | | | |
| 05 | | BIFINF26 | | | | | |
| 06 | | BIFINF28 | | | | | |

Input Formatting

Diagram - 6.2.7-01



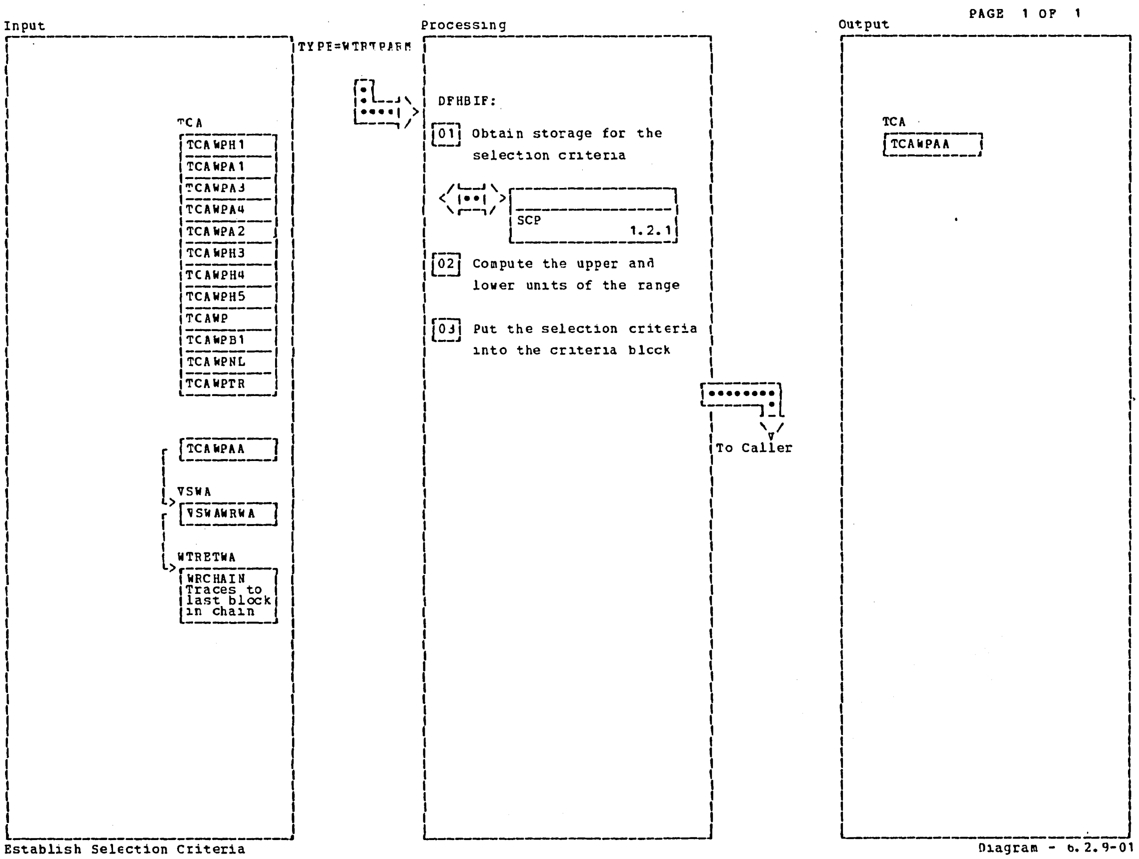
Initiate Weighted Retrieval

Diagram - 6.2.8-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 TCAWGCNT indicates the maximum number of records to be made available to the application program (NRECDS).<br>TCAWTH1 indicates the maximum number of records that can be read.<br>TCAWTH2 indicates where Weighted Retrieval is to begin examination of records. | | BIFWRT | | | | | |
| 02 A File Control SETL is issued to initiate the browse operation. | | | | | | | |
| 03 | | BFWRTO | | | | | |
| 04 | | BFWRTO | | | | | |

Initiate Weighted Retrieval

Diagram - 6.2.8-01



Establish Selection Criteria

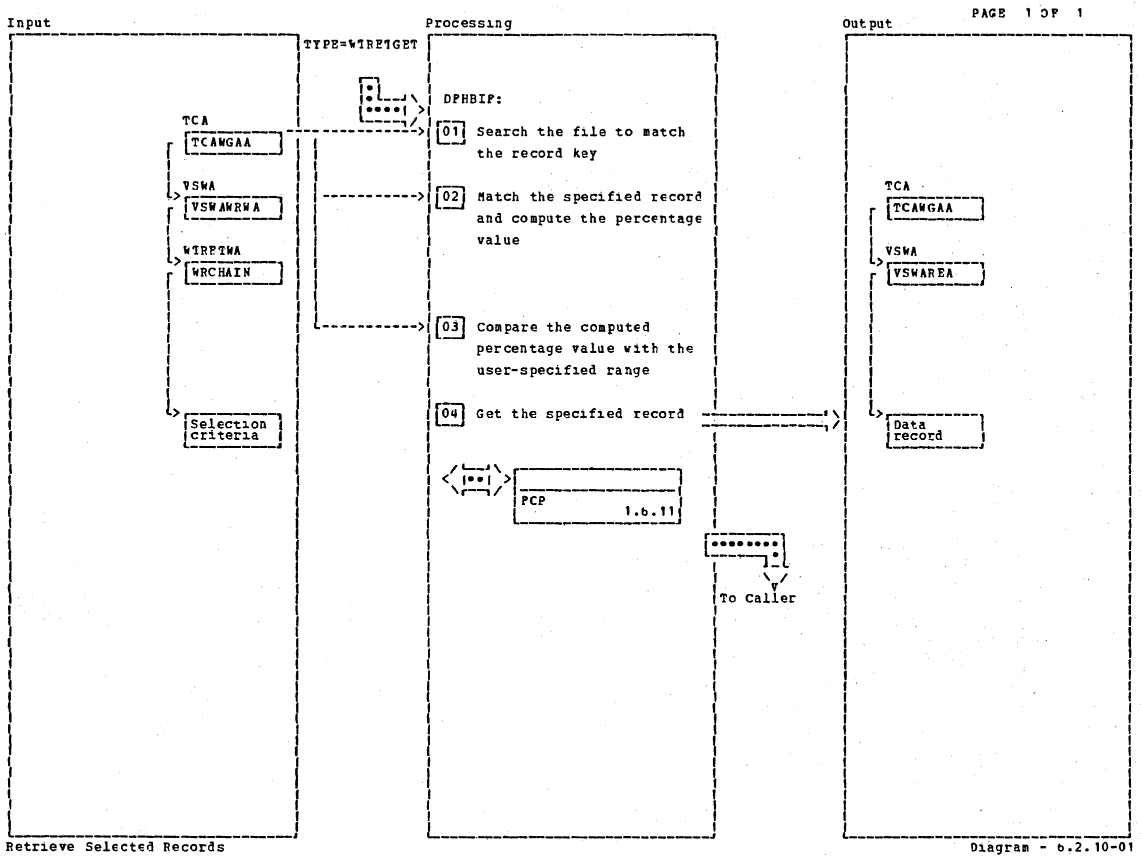
Diagram - 6.2.9-01

| Notes | Routine | Label | Ref |
|---|---|---|---|
| 01 The selecticn criteria's block chain is traced to the last block. | | BIFWPM2 | |
| 02 | | BIPWPM30 | |
| 03 | | BIPWPM20 | |

| Notes | Routine | Label | Ref |
|---|---|---|---|
| | | | |

Establish Selection Criteria

Diagram - 6.2.9-01


PAGE 1 OF 1
Input
Processing
Output
TYPE=WTRETGET
DFHBIF:
TCA
TCAWGAA
VSWA
VSWAWRWA
WTRETWA
WRCHAIN
Selection criteria
01 Search the file to match the record key
02 Match the specified record and compute the percentage value
03 Compare the computed percentage value with the user-specified range
04 Get the specified record
PCP
1.6.11
To Caller
TCA
TCAWGAA
VSWA
VSWAREA
Data record

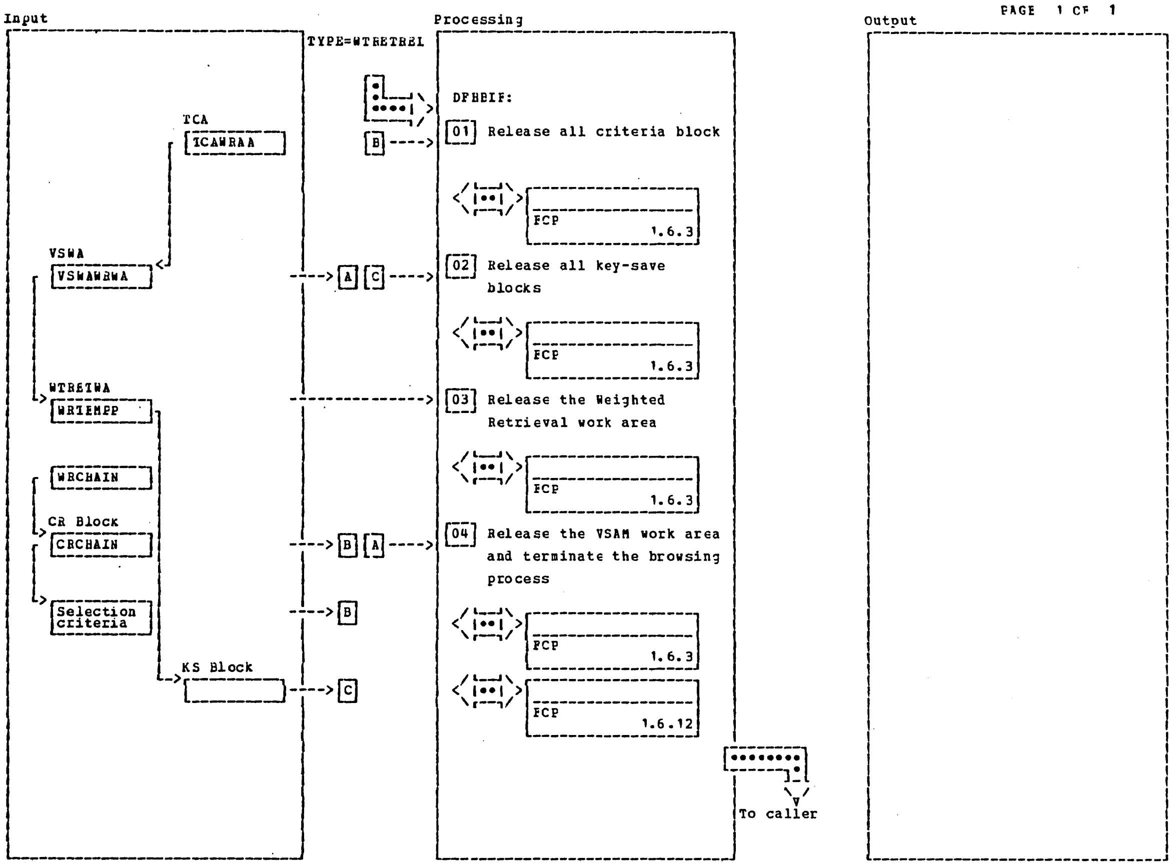
Retrieve Selected Records

Diagram - 6.2.10-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 The first three process steps apply only for the first WTRETGET request. The first WTRETGET request causes all acceptable record keys to be saved in a key-save block for use in subsequent WTRETGET requests. | | BIFWGT | | | | | |
| 04 A File Control GETNEXT is issued to retrieve the record. | | BIFWGT50 | | | | | |

Retrieve Selected Records

Diagram - 6.2.10-01



Release Storage Areas

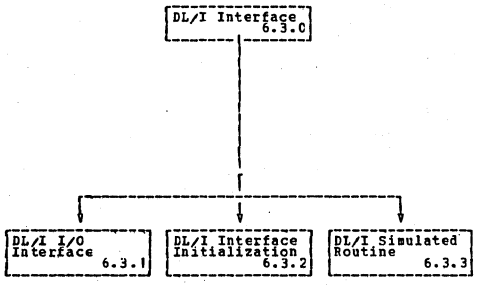
Diagram - 6.2.11-01

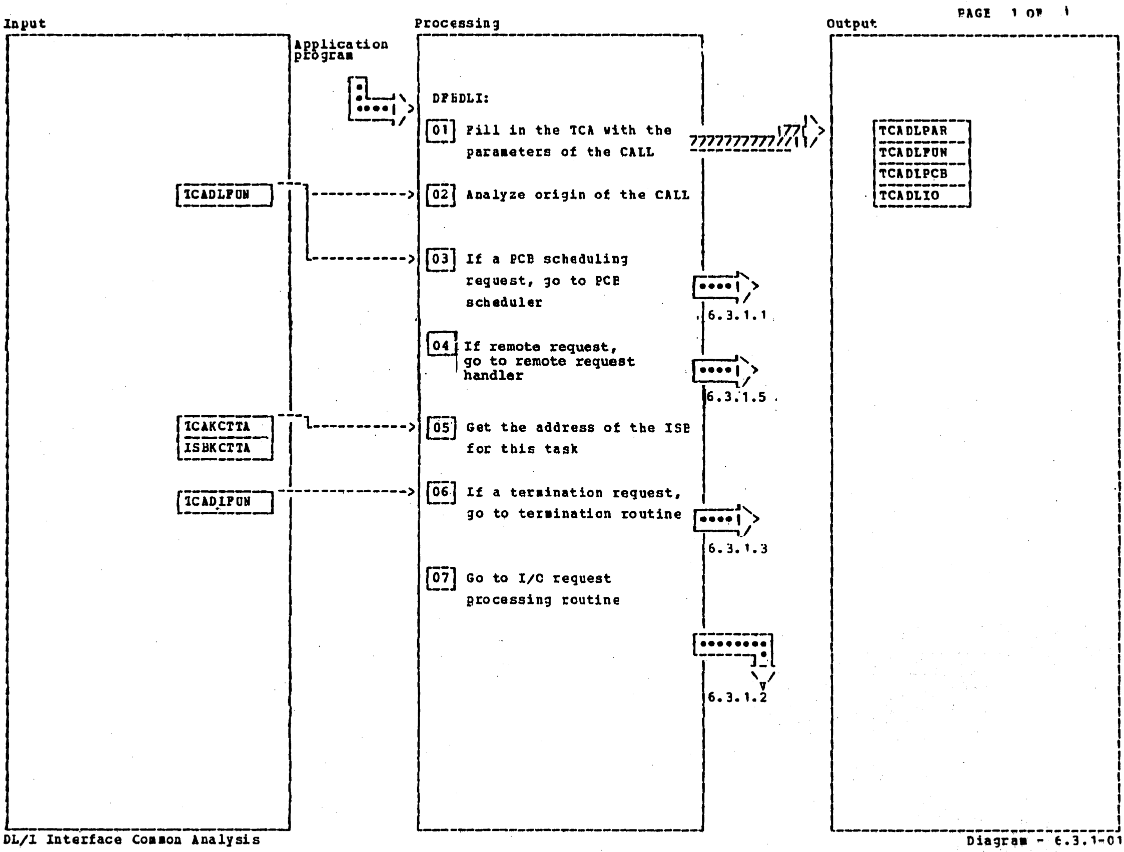
| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | BIFWRL | | | | | |
| 02 | | BIFWRLE | | | | | |
| 03 | | BIFWRLE | | | | | |
| 04 A File Control ESETL is issued. | | BIFWRLA | | | | | |

Release Storage Areas

Diagram - 6.2.11-01

PAGE 1 OF 1


DL/I Interface 6.3.0
DL/I I/O Interface 6.3.1
DL/I Interface Initialization 6.3.2
DL/I Simulated Routine 6.3.3



DFHDLI:

01 Fill in the TCA with the parameters of the CALL

02 Analyze origin of the CALL

03 If a PCB scheduling request, go to PCB scheduler

04 If remote request, go to remote request handler

05 Get the address of the ISB for this task

06 If a termination request, go to termination routine

07 Go to I/O request processing routine

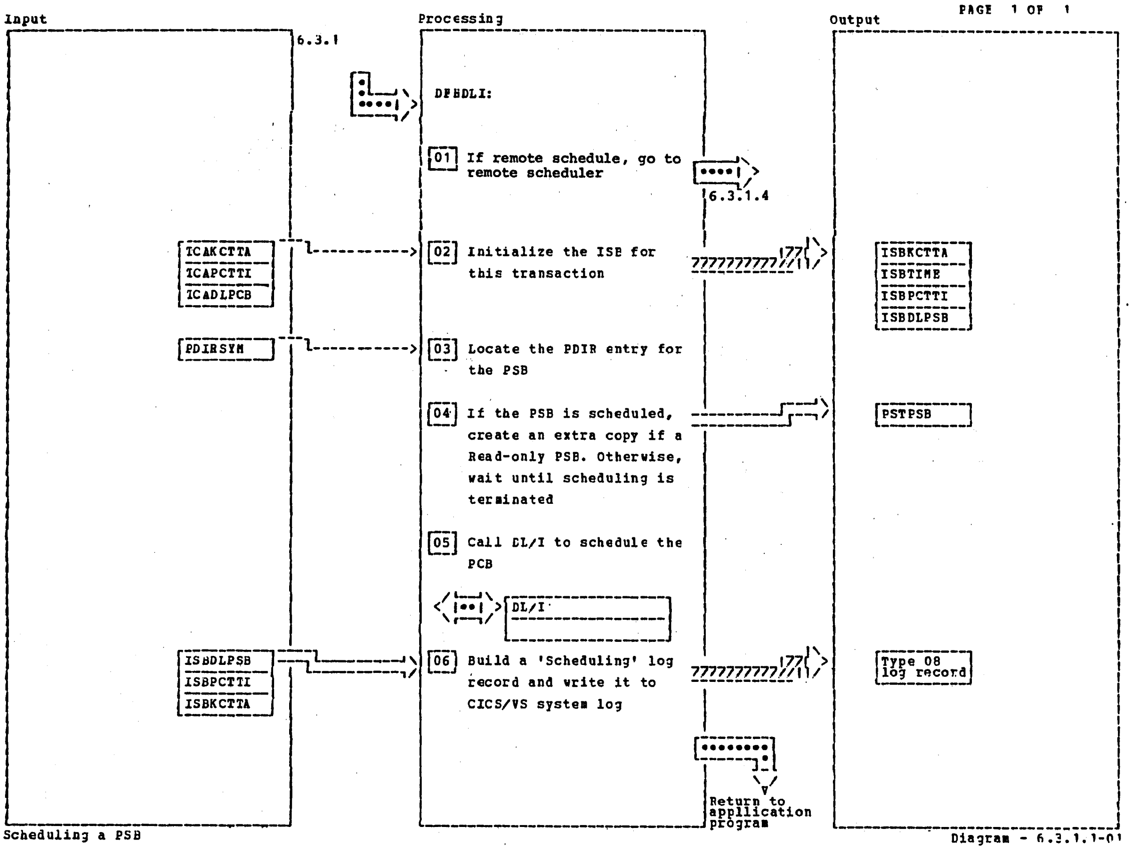
DL/I Interface Common Analysis

Diagram - 6.3.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | DLINBP | | | | | |
| 02 Analyze vhether CALL or DFHFC request, and also source language of calling program. | | DLOBIGIN | | | | | |

DL/I Interface Common Analysis

Diagram - 6.3.1-01



01 If remote schedule, go to remote scheduler

02 Initialize the ISB for this transaction

03 Locate the PDIR entry for the PSB

04 If the PSB is scheduled, create an extra copy if a Read-only PSB. Otherwise, wait until scheduling is terminated

05 Call DL/I to schedule the PCB

06 Build a 'Scheduling' log record and write it to CICS/VS system log

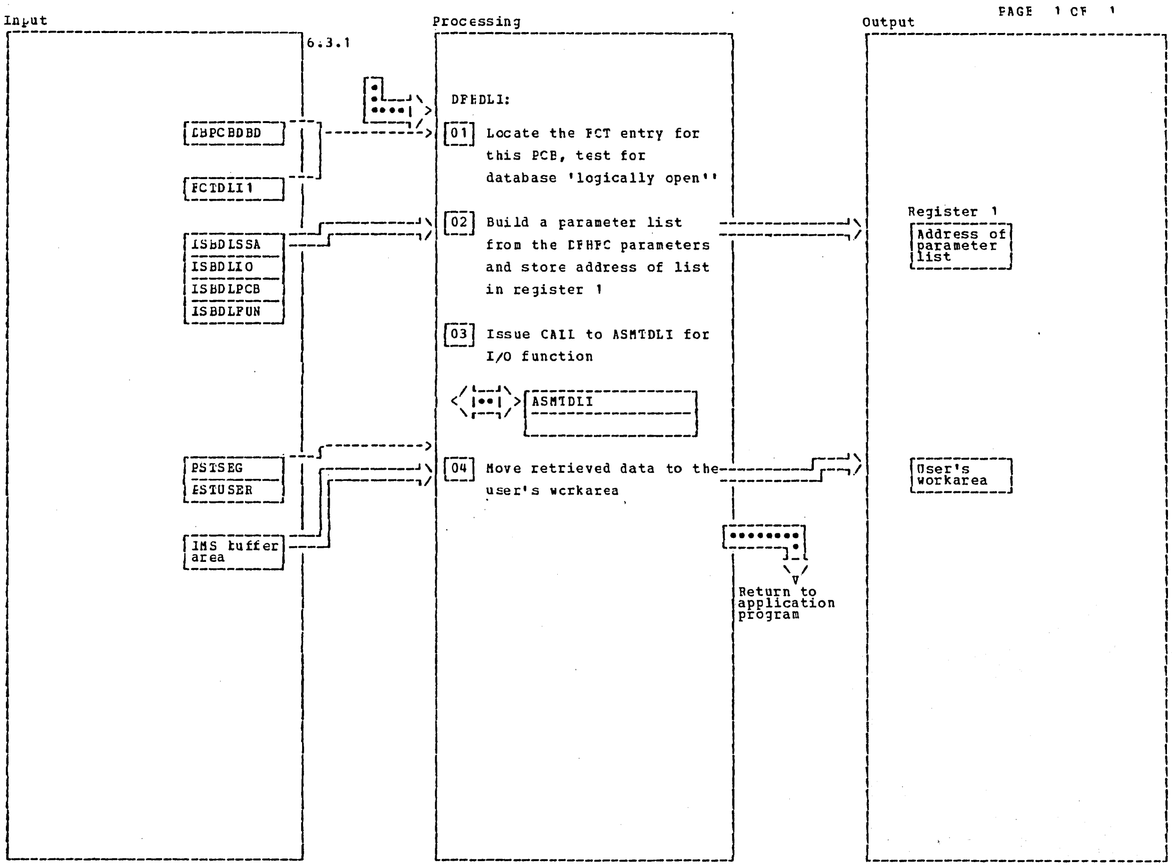
Scheduling a PSB

Diagram - 6.3.1.1-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 02 | | DFHDLPCP | | | | | |
| 03 | | DLPSBCB | | | | | |
| 04 | | DLPDLOOK | | | | | |
| 05 | | DLIFCALL | | | | | |
| 06 DFHJC TYPE= (WRITE, DL/I, WAIT) | | DLPCBX | | | | | |

Scheduling a PSB

Diagram - 6.3.1.1-01


PAGE 1 OF 1

Input

6.3.1

CBPCBDBD

FCTDLI1

ISBDLSSA

ISBDLIO

ISBDLPCB

ISBDLFUN

PSTSEG

PSTUSER

IMS buffer area

Processing

DFHDLI:

01 Locate the FCT entry for this PCB, test for database 'logically open''

02 Build a parameter list from the DFHFC parameters and store address of list in register 1

03 Issue CALL to ASMTDLI for I/O function

ASMTDLI

04 Move retrieved data to the user's workarea

Return to application program

Output

Register 1

Address of parameter list

User's workarea

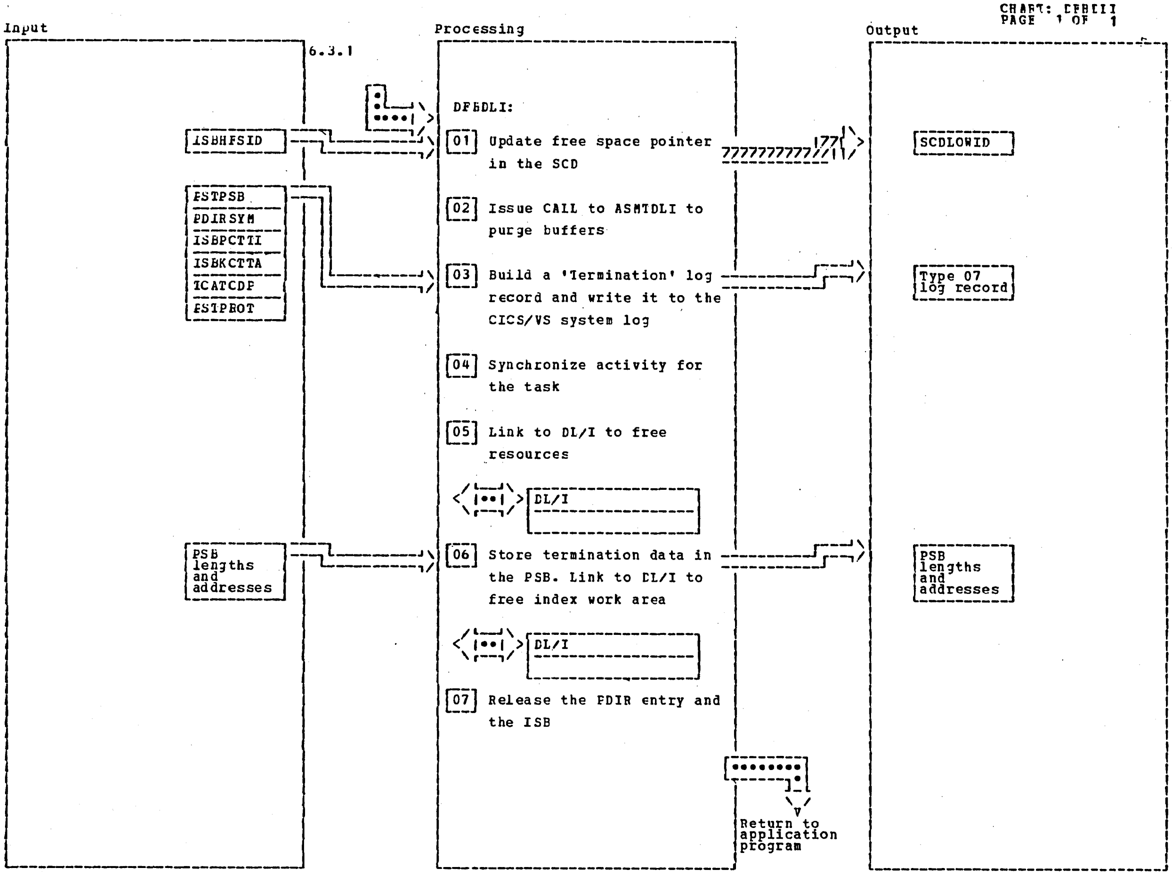
DL/I I/O Request

Diagram - 6.3.1.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 |  | DLLOCFCT |  |  |  |  |  |
| 02 |  | DLLBUILD |  |  |  |  |  |
| 03 |  | DLICALL |  |  |  |  |  |
| 04 |  | DLIMOVE |  |  |  |  |  |

DL/I I/O Request

Diagram - 6.3.1.2-01



DFHDLI:

01 Update free space pointer in the SCD

02 Issue CALL to ASMTDLI to purge buffers

03 Build a 'Termination' log record and write it to the CICS/VS system log

04 Synchronize activity for the task

05 Link to DL/I to free resources

06 Store termination data in the PSB. Link to DL/I to free index work area

07 Release the FDIR entry and the ISB

Terminate a PCB

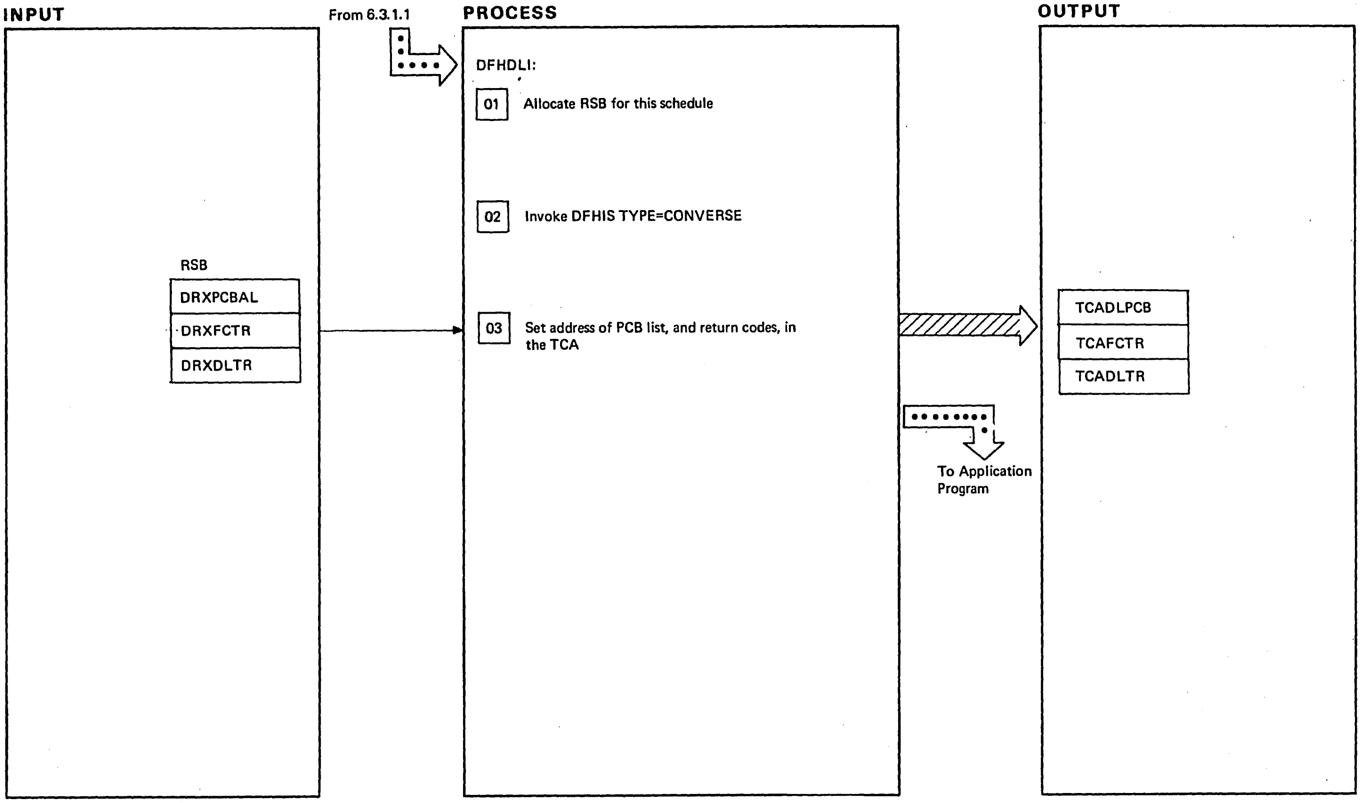
Diagram - 6.3.1.3-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 | | DFHDLI | | | | | |
| 02 | | DLCIUNS | | | | | |
| 03 | | DLTLOG | | | | | |
| 04 A DFHJP macro instruction is issued unless the DL/I TERM CALL was issued by the sync point ptogram. | DLOGTBE | | | | | | |
| 05 | | DLTFREE | | | | | |
| 07 | | DLTFRISB | | | | | |

Terminate a PCB

Diagram - 6.3.1.3-01

Page 1 of 1


INPUT
From 6.3.1.1
PROCESS
OUTPUT
DFHDLI:
01 Allocate RSB for this schedule
02 Invoke DFHIS TYPE=CONVERSE
03 Set address of PCB list, and return codes, in the TCA
RSB
DRXPCBAL
DRXFCTR
DRXDLTR
TCADLPCB
TCAFCTR
TCADLTR
To Application Program

Remote Scheduler

Diagram — 6.3.1.4-01

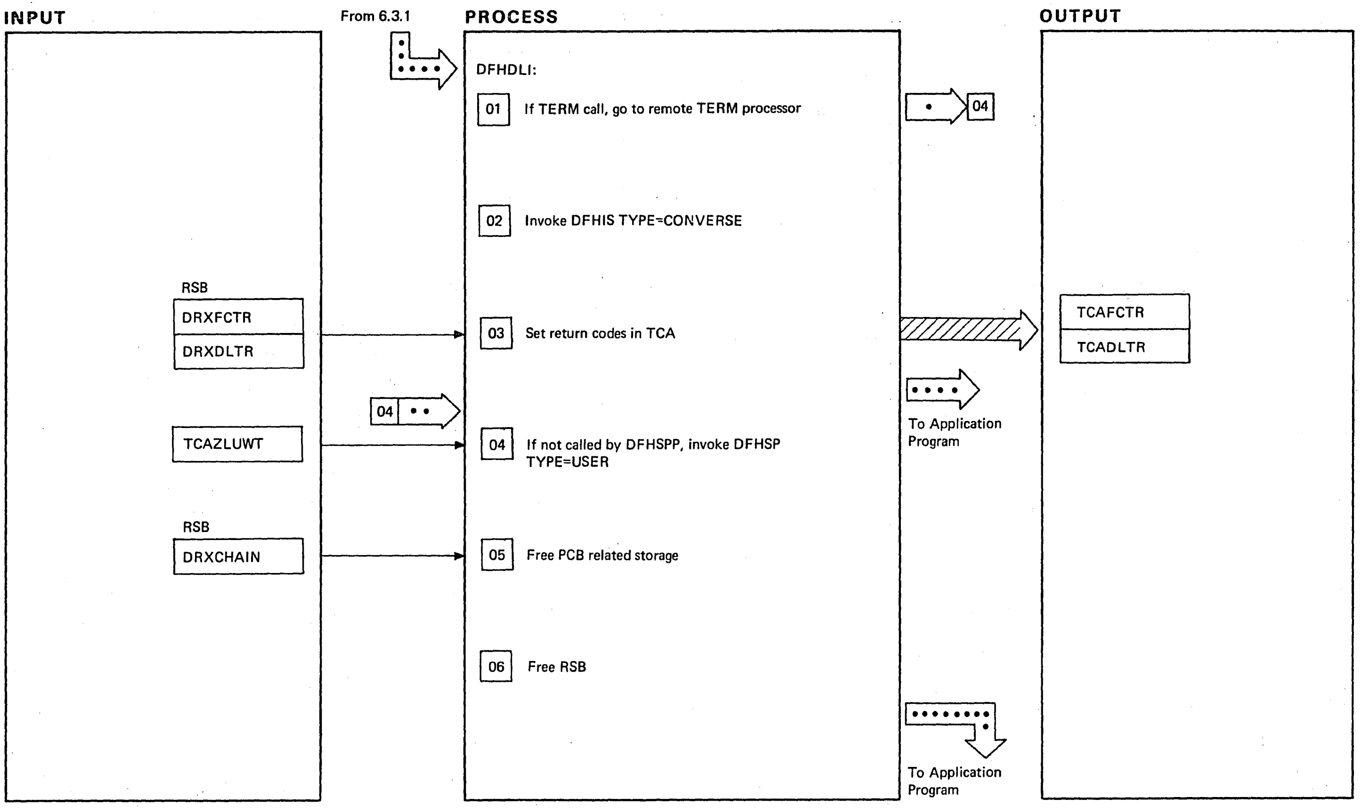
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The required PSB is on a remote CICS/VS system. Allocate Remote Scheduling Block (RSB) for duration of this schedule (i.e., until TERM call) | | REMPSB | |
| 02 Transmit scheduling request to remote system | | REMP030 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Remote Scheduler

Diagram — 6.3.1.4-01

Page 1 of 1



DFHDLI:

01 If TERM call, go to remote TERM processor

02 Invoke DFHIS TYPE=CONVERSE

03 Set return codes in TCA

04 If not called by DFHSPP, invoke DFHSP TYPE=USER

05 Free PCB related storage

06 Free RSB

Remote Request Handler

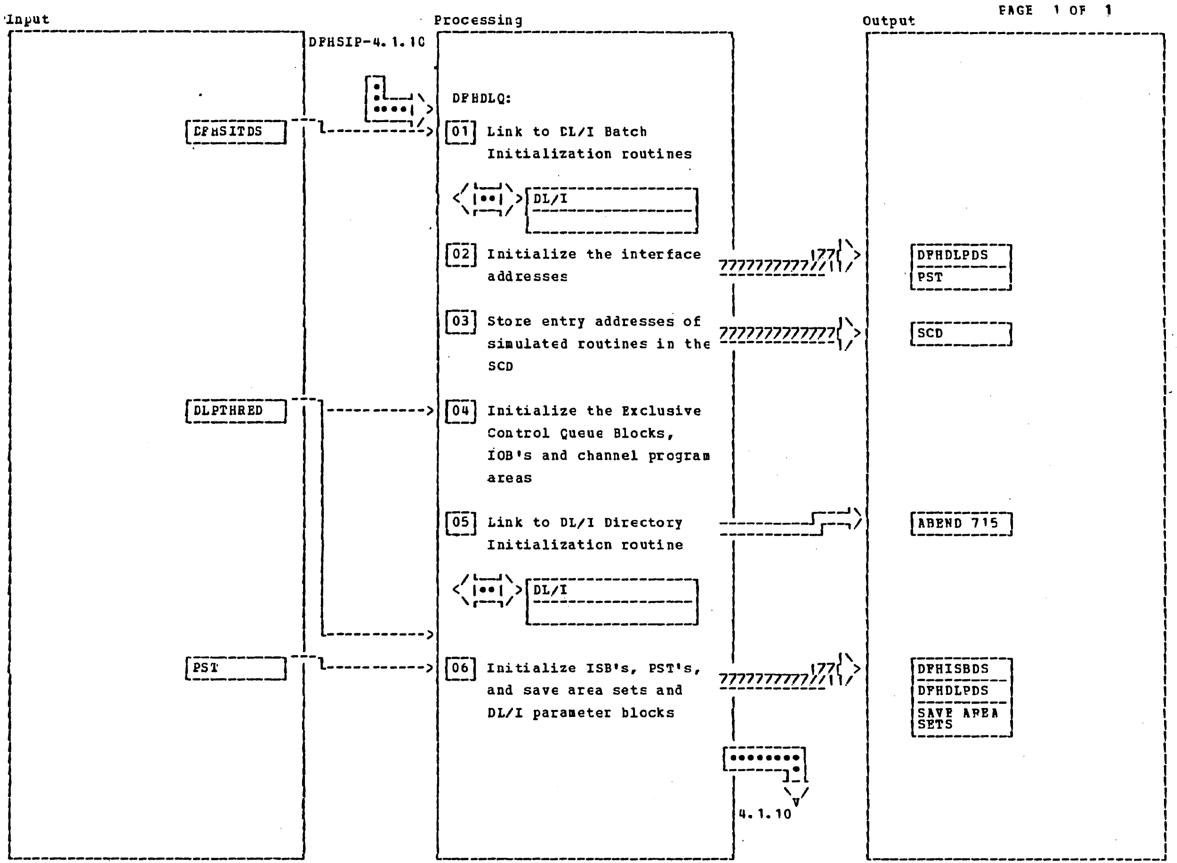
Diagram – 6.3.1.5-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Request is for data base residing on remote CICS/VS system | | REMDB | |
| 02 Transmit remote request to remote CICS/VS system owning the data base | | REMD060 | |
| 04 If DFHDLI was not invoked by DFHSPP for this TERM call, a sync point must be issued | | REMD500 | |
| 05 Local copy of PCBs, created by DFHXFP during schedule response processing | | REMD510 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Remote Request Handler

Diagram – 6.3.1.5-01



DFHDLQ:

01 Link to DL/I Batch Initialization routines

02 Initialize the interface addresses

03 Store entry addresses of simulated routines in the SCD

04 Initialize the Exclusive Control Queue Blocks, IOB's and channel program areas

05 Link to DL/I Directory Initialization routine

06 Initialize ISB's, PST's, and save area sets and DL/I parameter blocks

DL/I interface Initialization

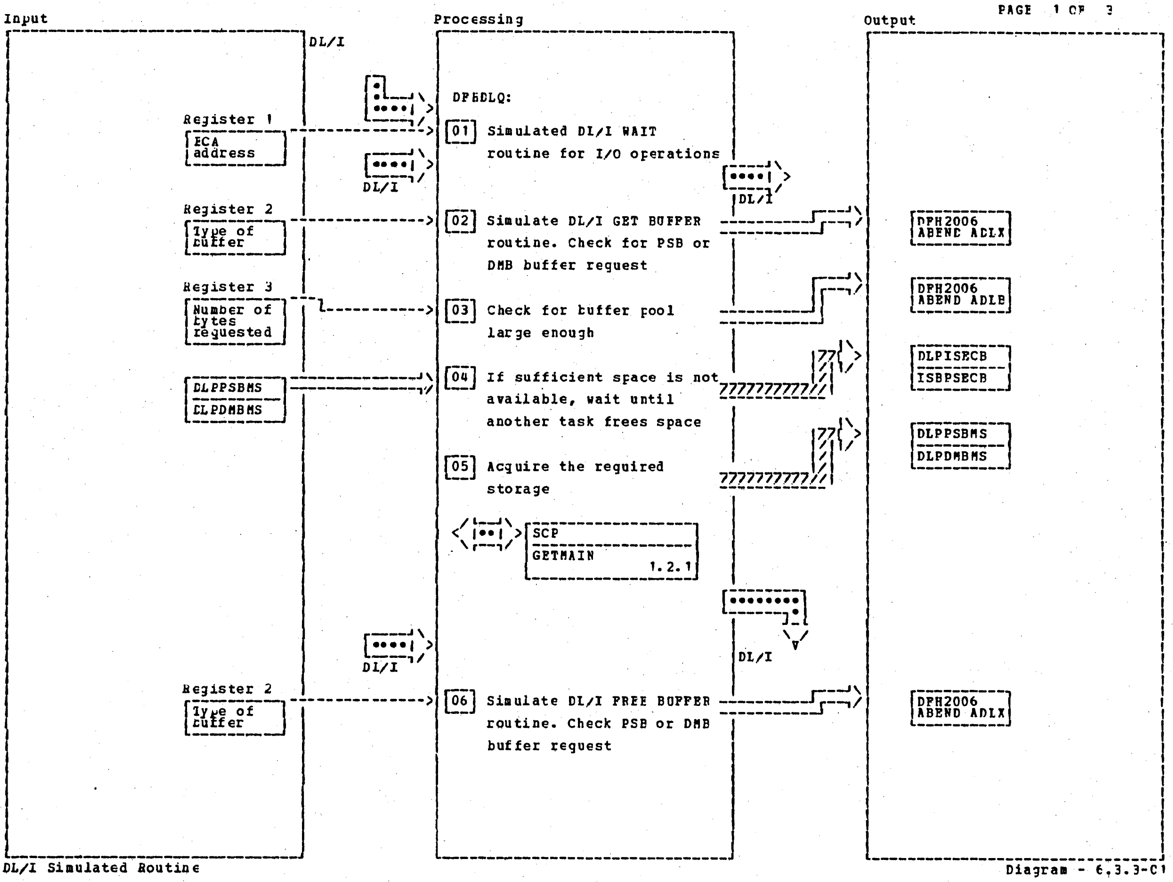
Diagram - 6.3.2-01

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 DFSBBA00 checks the parameters passed by DFHSIP and loads and initializes PXPARMS, ECPARMS, LIPARMS, and PCPARMS. DFSBBLD0 loads the PSB's and DMB's from the ACBLIB. DFSDLBN0 initializes the PST. | | DFHBBA00 | | | | | |
| 02 | | DLGETIA | | | | | |
| 03 | | DLINSCD | | | | | |
| 05 DFSIIND0 initializes the PSB and DMB directories. If any errors are detected, an OS ABEND 715 is issued. | | | | | | | |
| 06 A copy of DFSBNUC's PST is made for each 'thread' and its' address is placed in the corresponding ISB at ISBPST. | | DLINIT | | | | | |

DL/I interface Initialization

Diagram - 6.3.2-01

PAGE 1 OF 3



Input

Register 1
ECA address

Register 2
Type of buffer

Register 3
Number of bytes requested

DLPPSBMS
DLPDMBMS

Register 2
Type of buffer

Processing

DFHDLQ:

01 Simulated DL/I WAIT routine for I/O operations

02 Simulate DL/I GET BUFFER routine. Check for PSB or DMB buffer request

03 Check for buffer pool large enough

04 If sufficient space is not available, wait until another task frees space

05 Acquire the required storage

SCP
GETMAIN 1.2.1

06 Simulate DL/I FREE BUFFER routine. Check PSB or DMB buffer request

Output

DFH2006
ABEND ADLX

DFH2006
ABEND ADLB

DLPISECB
ISBPSECB

DLPPSBMS
DLPDMBMS

DFH2006
ABEND ADLX

DL/I Simulated Routine

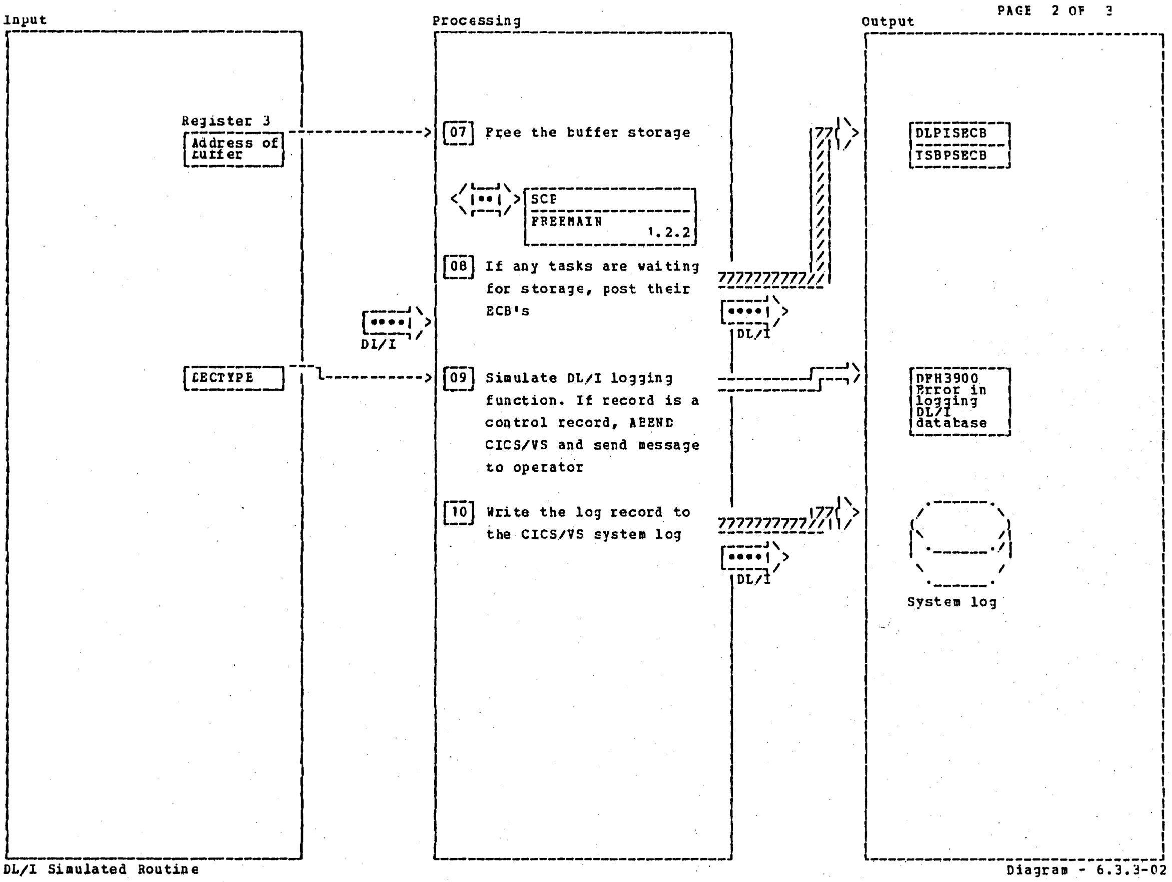
Diagram - 6.3.3-C1

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 01 DFHKC TYPE=WAIT | | IWAIT | | | | | |
| 02 | | IGETBUF | | | | | |
| 03 | | DLGBUF | | | | | |
| 04 Register 0 contains the address of the SCD. If short on storage and register 0 is negative, return to DL/I immediately. If register 0 is positive, issue a DFHKC TYPE=WAIT macro instruction. | | | | | | | |
| 05 DFHSC TYPE=GETMAIN | | | | | | | |
| 06 | | IGETBUF | | | | | |

DL/I Simulated Routine

Diagram - 6.3.3-C1

PAGE 2 OF 3



[07] Free the buffer storage

[08] If any tasks are waiting for storage, post their ECB's

[09] Simulate DL/I logging function. If record is a control record, ABEND CICS/VS and send message to operator

[10] Write the log record to the CICS/VS system log

DL/I Simulated Routine

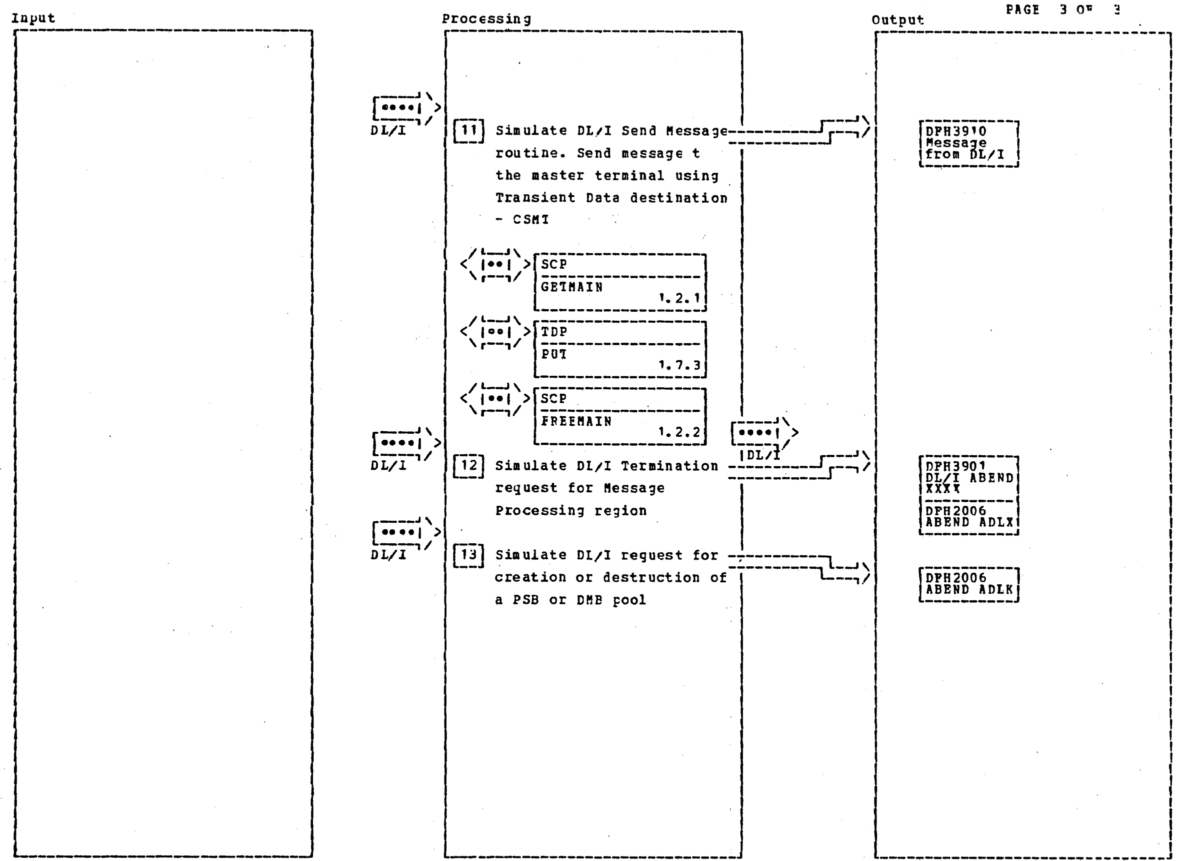
Diagram - 6.3.3-02

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| [07] | | DLFEF | | | | | |
| [09] Types 50, 51, and 52 are valid. The record is printed via a DFHDC TYPE=PARTIAL. | | DFSFLOIO | | | | | |
| [10] DFHJC TYPE=GETJCA and DFHJC TYPE=(WRITE, DL/I, PARTIAL). | | FLPUT | | | | | |

DL/I Simulated Routine

Diagram - 6.3.3-02

PAGE 3 OF 3



11 Simulate DL/I Send Message routine. Send message t the master terminal using Transient Data destination - CSMT

12 Simulate DL/I Termination request for Message Processing region

13 Simulate DL/I request for creation or destruction of a PSB or DMB pool

DL/I Simulated Routine

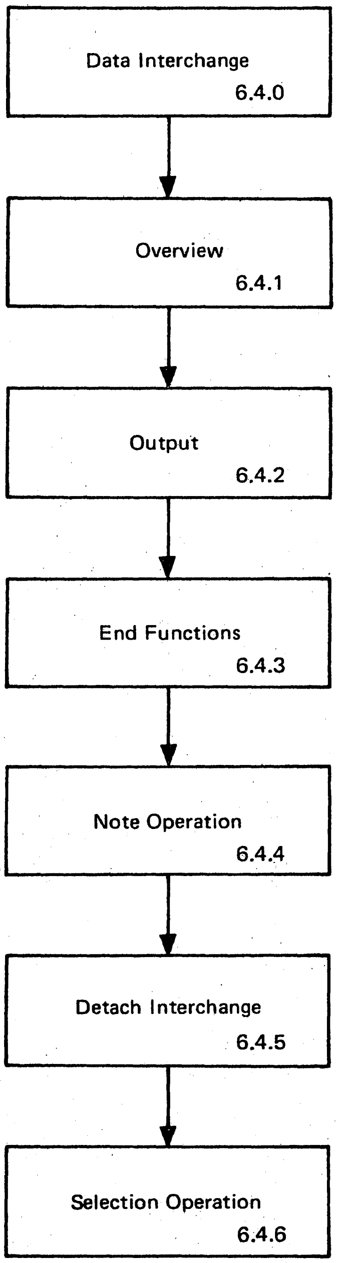
Diagram - 6.3.3-03

| Notes | Routine | Label | Ref | Notes | Routine | Label | Ref |
|---|---|---|---|---|---|---|---|
| 11 DFHSC TYPE=GETMAIN<br>DFHTD TYPE=PUT<br>DFHSC TYPE=FREEMAIN | | DFSCLMR0 | | | | | |
| 12 DL/I cannot continue processing the request. | | IASE | | | | | |
| 13 This implies a logic error in the interface. | | ICREAT | | | | | |

DL/I Simulated Routine

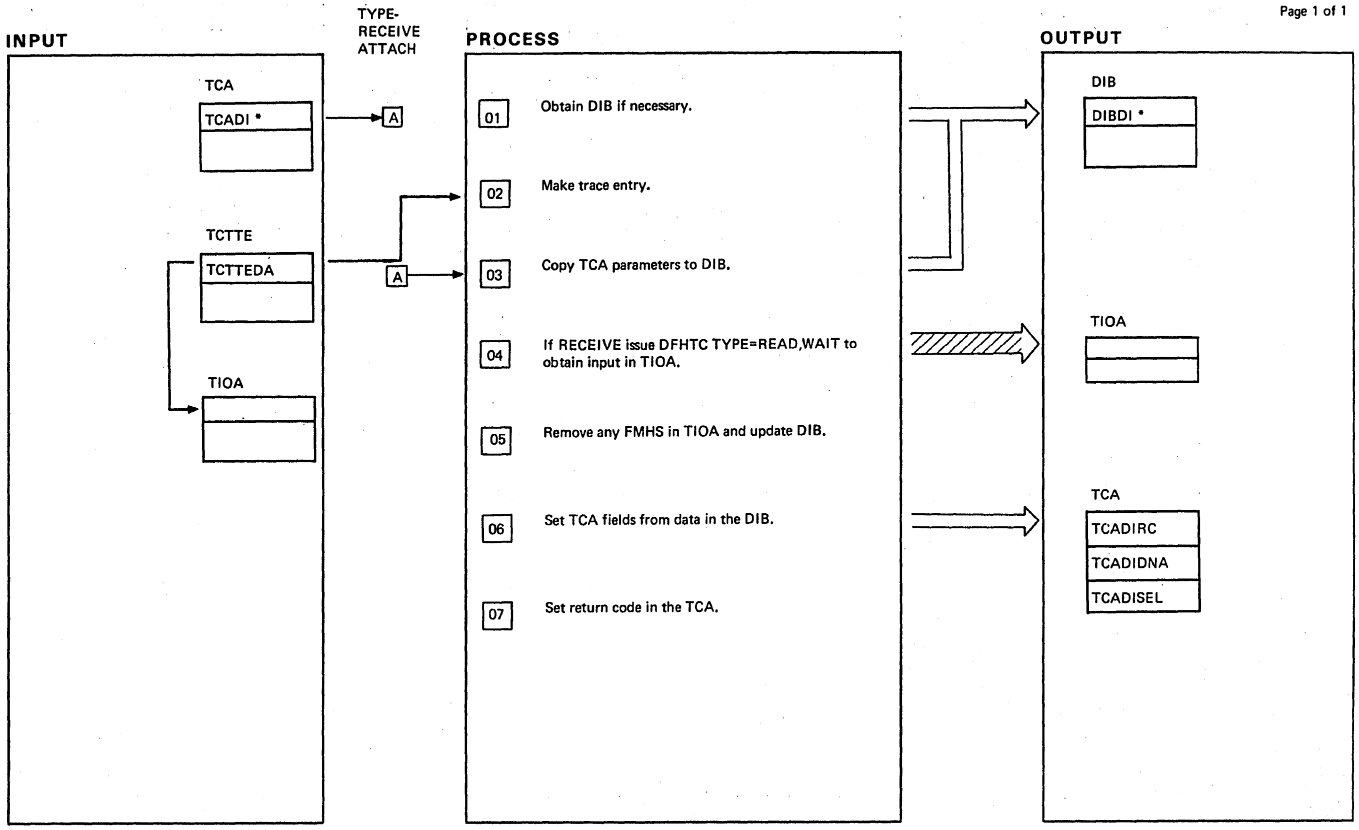
Diagram - 6.3.3-03

Page 1 of 1



Data Interchange — Visual Index

Diagram — 6.4.0-01



01 Obtain DIB if necessary.

02 Make trace entry.

03 Copy TCA parameters to DIB.

04 If RECEIVE issue DFHTC TYPE=READ,WAIT to obtain input in TIOA.

05 Remove any FMHS in TIOA and update DIB.

06 Set TCA fields from data in the DIB.

07 Set return code in the TCA.

Data Interchange — Input

Diagram 6.4.1–01

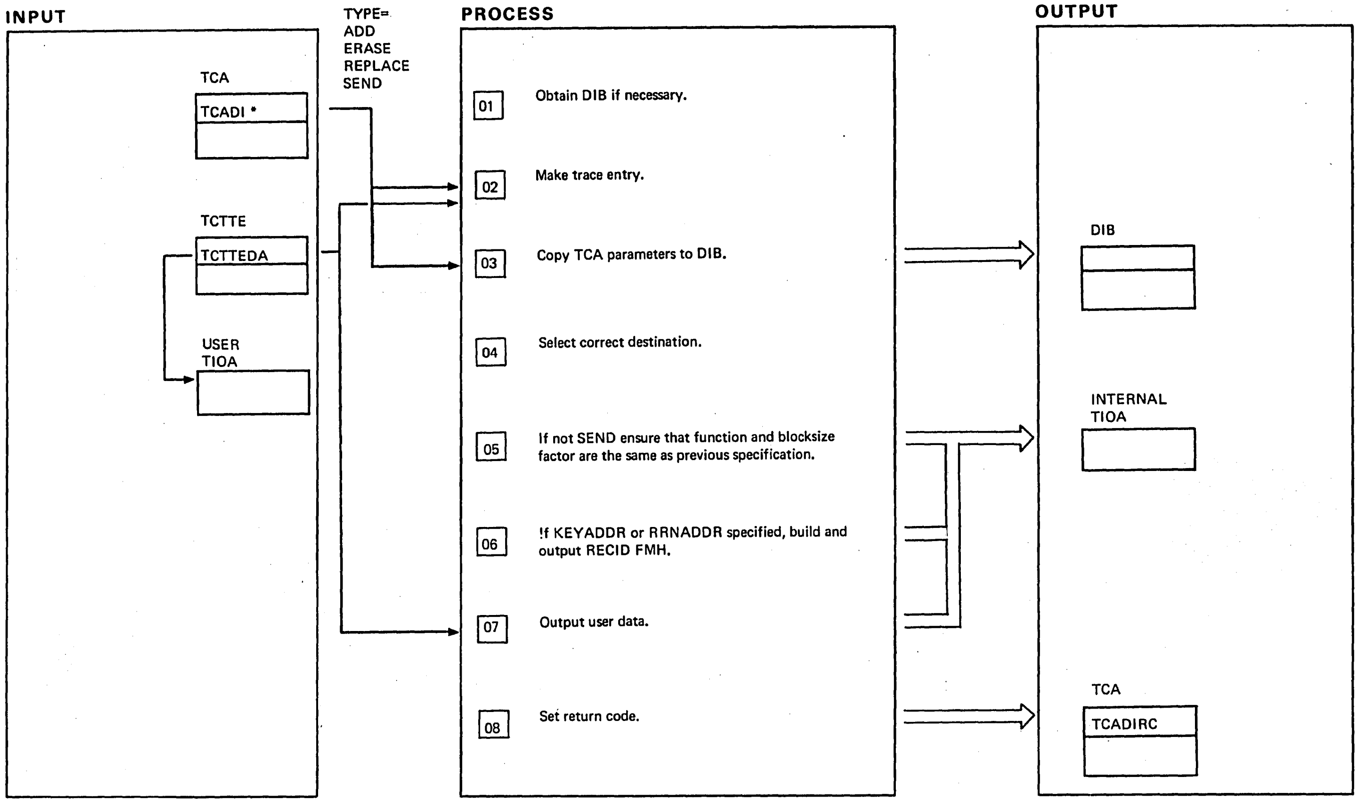
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 04 For CTYPE=ATTACH TIOA already contains the data to process. | | INPUTRTE | |
| 05 | | INPUT-FMH | |
| 06 | | INPRETNO | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Data Interchange — Input

Diagram 6.4.1–01

Page 1 of 1



01 Obtain DIB if necessary.

02 Make trace entry.

03 Copy TCA parameters to DIB.

04 Select correct destination.

05 If not SEND ensure that function and blocksize factor are the same as previous specification.

06 !f KEYADDR or RRNADDR specified, build and output RECID FMH.

07 Output user data.

08 Set return code.

Data Interchange — Output

Diagram 6.4.2—01

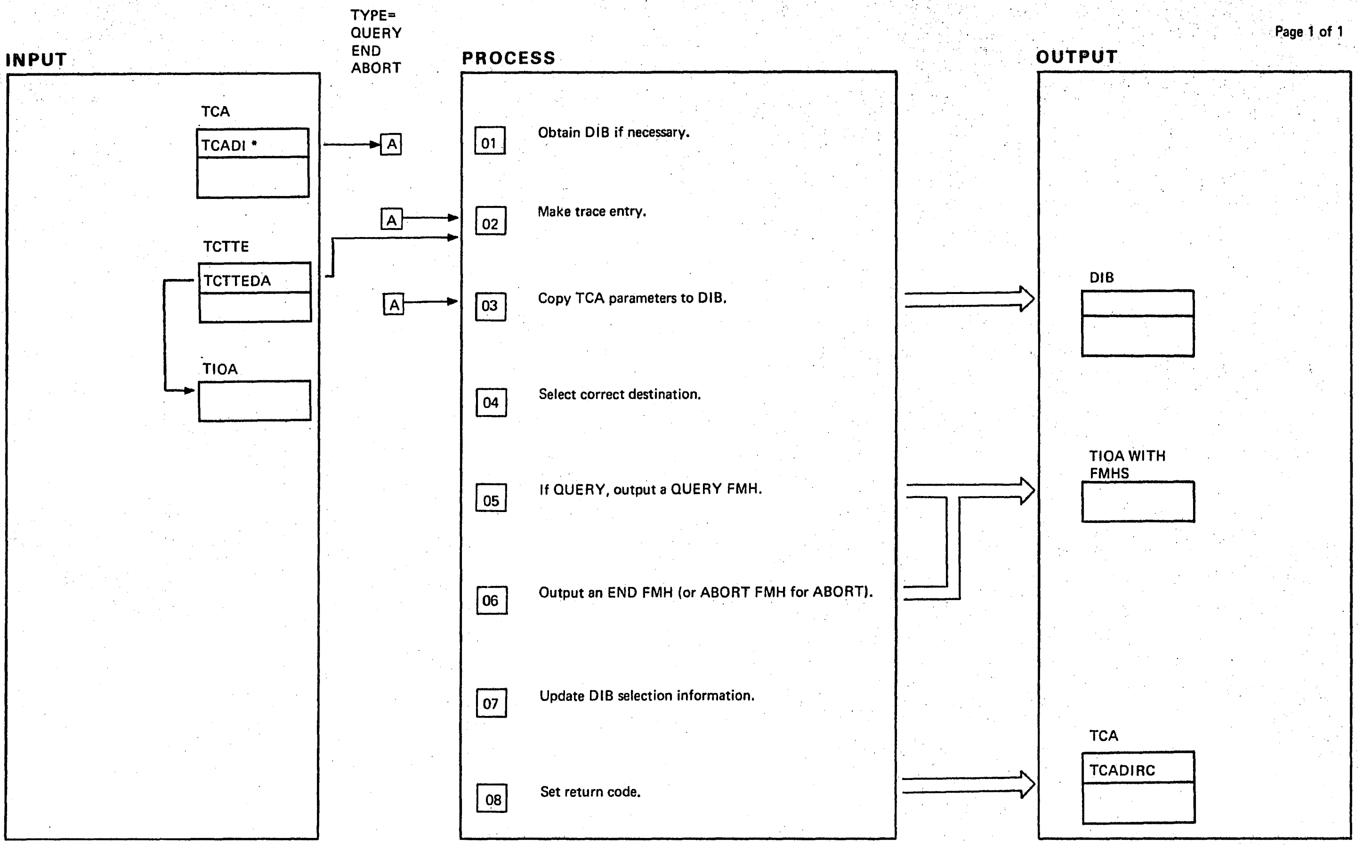
| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 04 | Selecting the correct destination involves possibly suspending/resuming/ending the current destination and beginning new destination. | | DESTCHEK | |
| 05 | | | SARE0010 | |
| 06 | | | SARERIDT | |
| 07 | | | DISENRTE | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Data Interchange — Output

Diagram 6.4.2—01

Page 1 of 1



01 Obtain DIB if necessary.

02 Make trace entry.

03 Copy TCA parameters to DIB.

04 Select correct destination.

05 If QUERY, output a QUERY FMH.

06 Output an END FMH (or ABORT FMH for ABORT).

07 Update DIB selection information.

08 Set return code.

Data Interchange — End Functions

Diagram 6.4.3–01

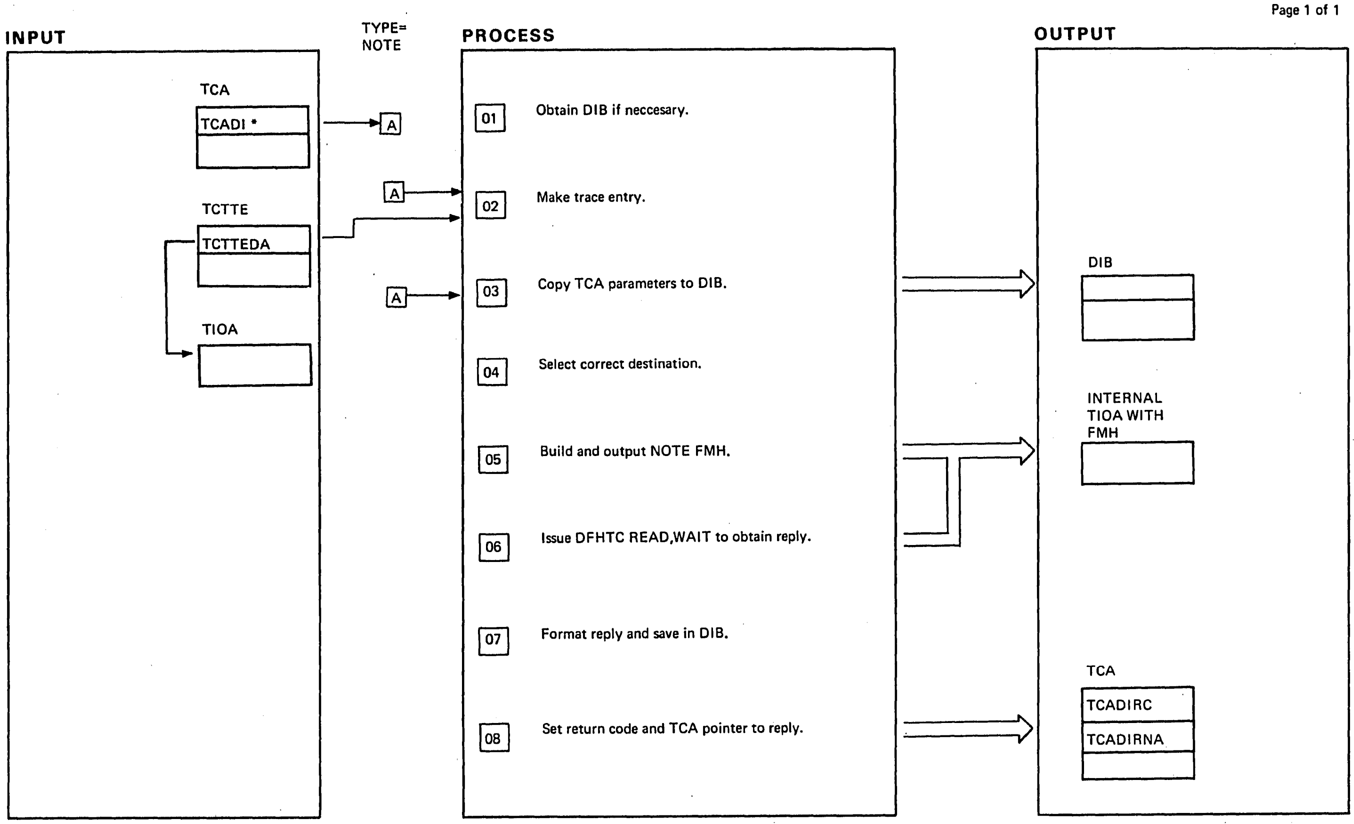
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 04 See 6.4.6 | | DESTCHEK | |
| 05 | | FMHBQUER | |
| 06 | | DESELECT | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Data Interchange — End Functions

Diagram 6.4.3–01

Page 1 of 1



01 Obtain DIB if neccesary.

02 Make trace entry.

03 Copy TCA parameters to DIB.

04 Select correct destination.

05 Build and output NOTE FMH.

06 Issue DFHTC READ,WAIT to obtain reply.

07 Format reply and save in DIB.

08 Set return code and TCA pointer to reply.

Data Interchange — Note Operation

Diagram 6.4.4–01

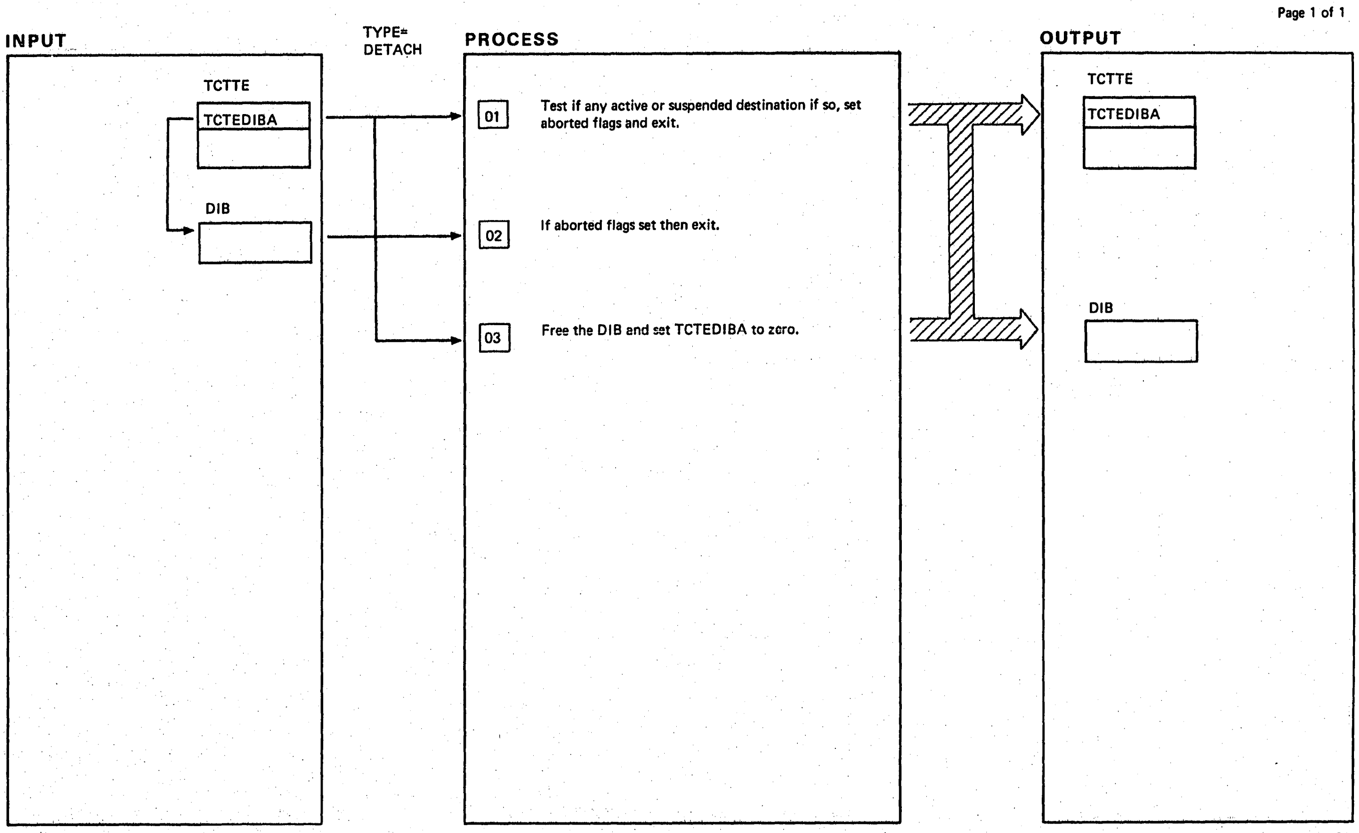
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 04. See 6.4.6 | | DESTCHEK | |
| 05 | | FMHBNOTE | |
| 06 | | FMHBN010 | |
| 07 | | FMHINP40 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Data Interchange — Note Operation

Diagram 6.4.4–01

Page 1 of 1



01 Test if any active or suspended destination if so, set aborted flags and exit.

02 If aborted flags set then exit.

03 Free the DIB and set TCTEDIBA to zero.

Data Interchange — Detach Operation

Diagram 6.4.5–01

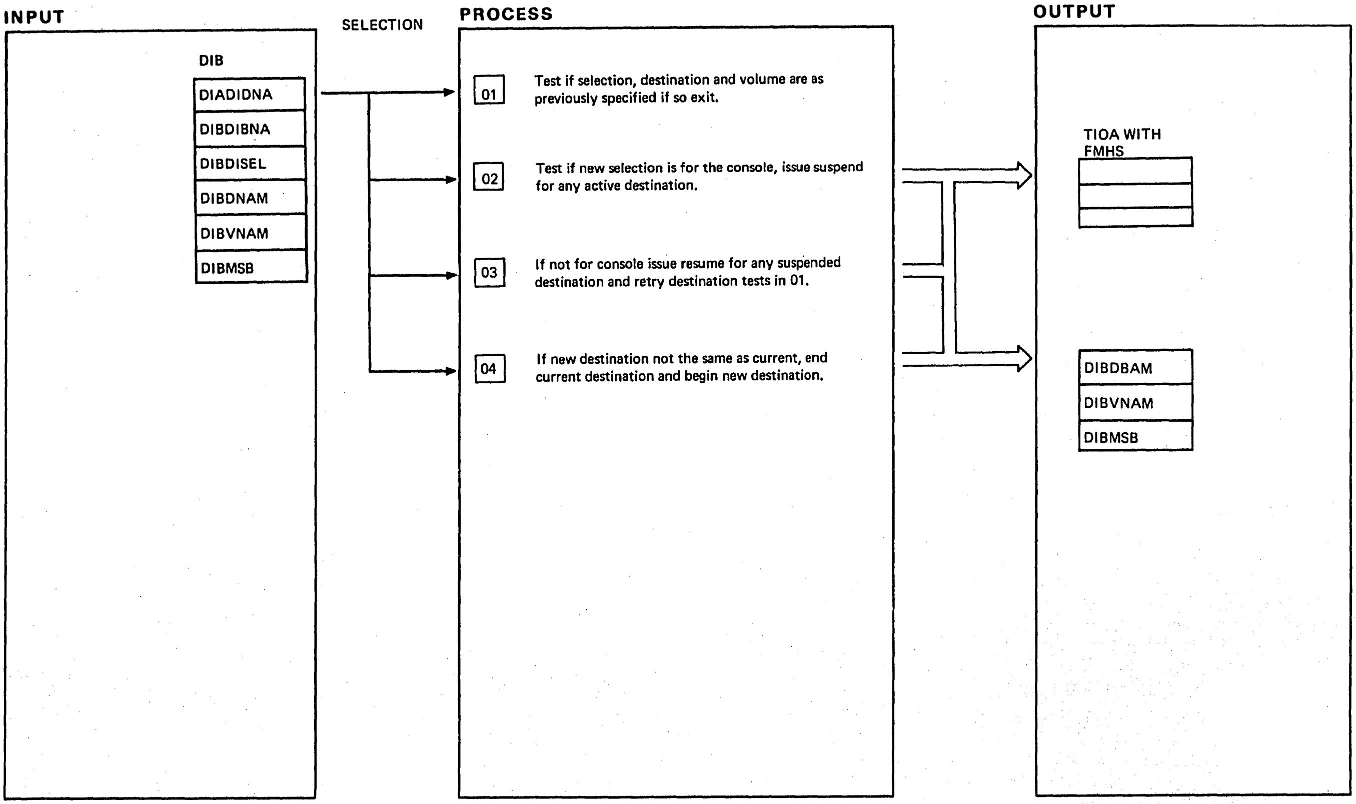
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 | | DETACH | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Data Interchange — Detach Operation

Diagram 6.4.5–01

Page 1 of 1



01 Test if selection, destination and volume are as previously specified if so exit.

02 Test if new selection is for the console, issue suspend for any active destination.

03 If not for console issue resume for any suspended destination and retry destination tests in 01.

04 If new destination not the same as current, end current destination and begin new destination.

Data Interchange – Selection Operation

Diagram 6.4.6–01

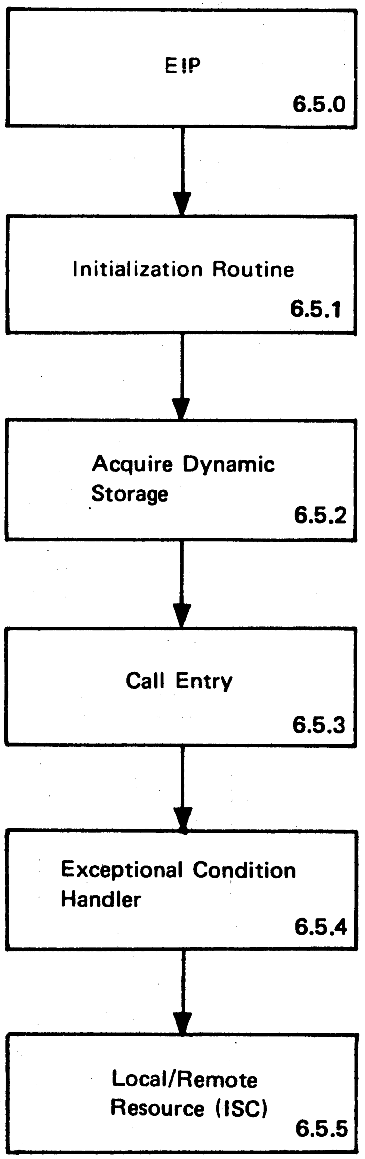
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 | | DESTCHEK | |
| 02 | | DESTEND | |
| 03 | | DESTC100 | |
| 04 | | DESTC300 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Data Interchange – Selection Operation

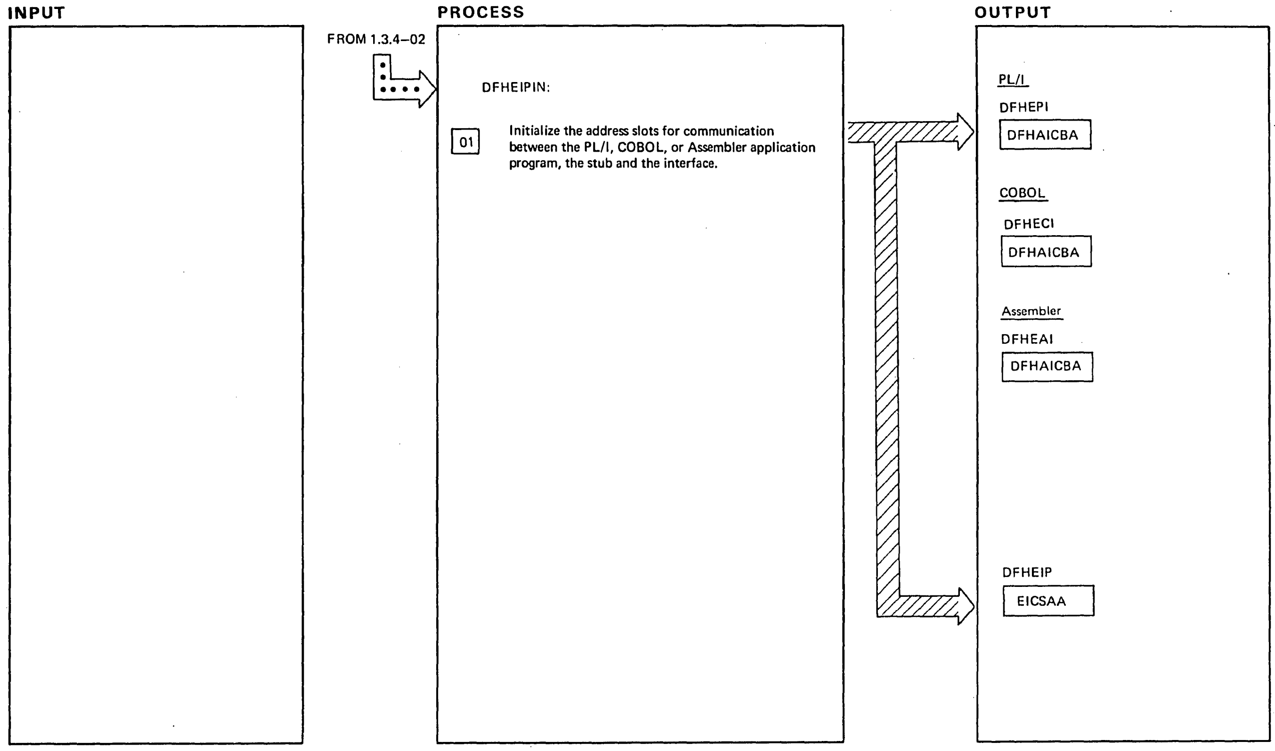
Diagram 6.4.6–01

Page 1 of 1



EIP – Visual Index

Diagram – 6.5.0-01



01 Initialize the address slots for communication between the PL/I, COBOL, or Assembler application program, the stub and the interface.

EIP Initialization routine

Diagram – 6.5.1-01

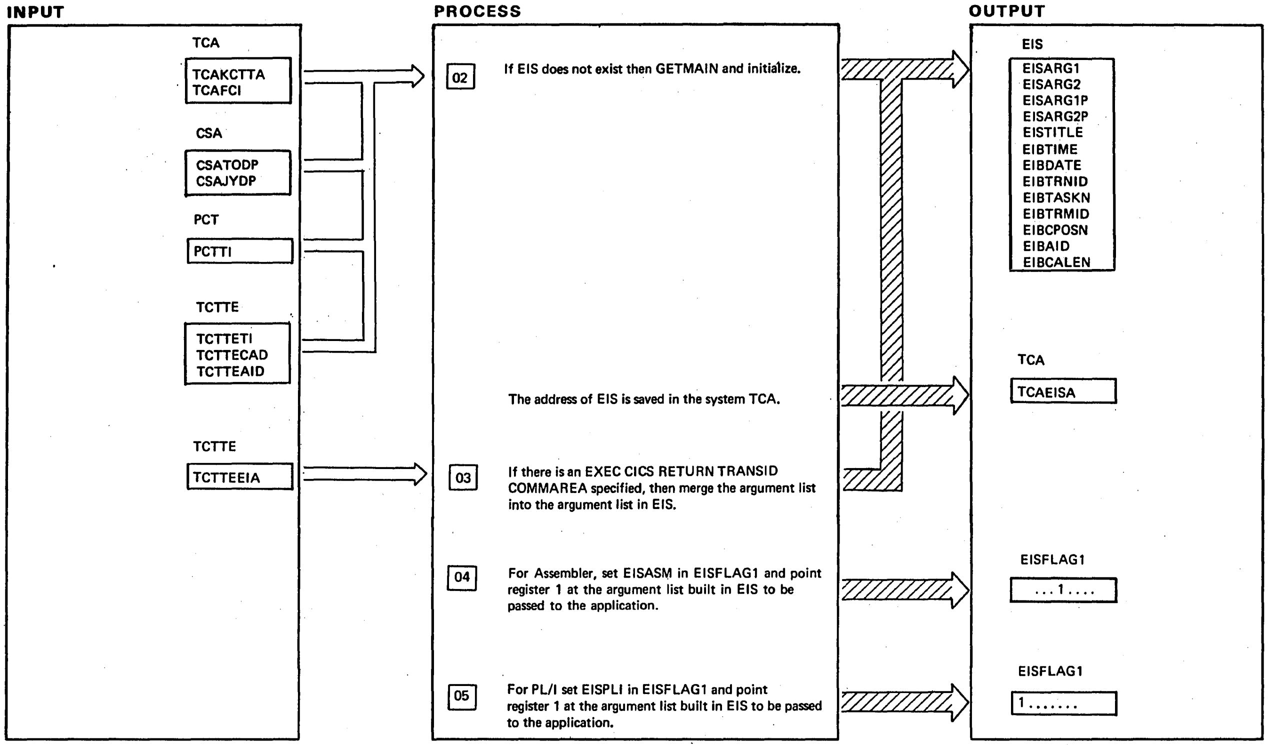
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 PCP passes register 1 to DFHEIPIN which points at an argument list that contains the address of the slot in the stub which is to contain the address of the call entry point to EIP–DFHEIPCN. | | DFHEIPIN | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

EIP Initialization routine

Diagram – 6.5.1-01

Page 2 of 3



02 If EIS does not exist then GETMAIN and initialize.

The address of EIS is saved in the system TCA.

03 If there is an EXEC CICS RETURN TRANSID COMMAREA specified, then merge the argument list into the argument list in EIS.

04 For Assembler, set EISASM in EISFLAG1 and point register 1 at the argument list built in EIS to be passed to the application.

05 For PL/I set EISPLI in EISFLAG1 and point register 1 at the argument list built in EIS to be passed to the application.

EIP Initialization routine

Diagram — 6.5.1-02

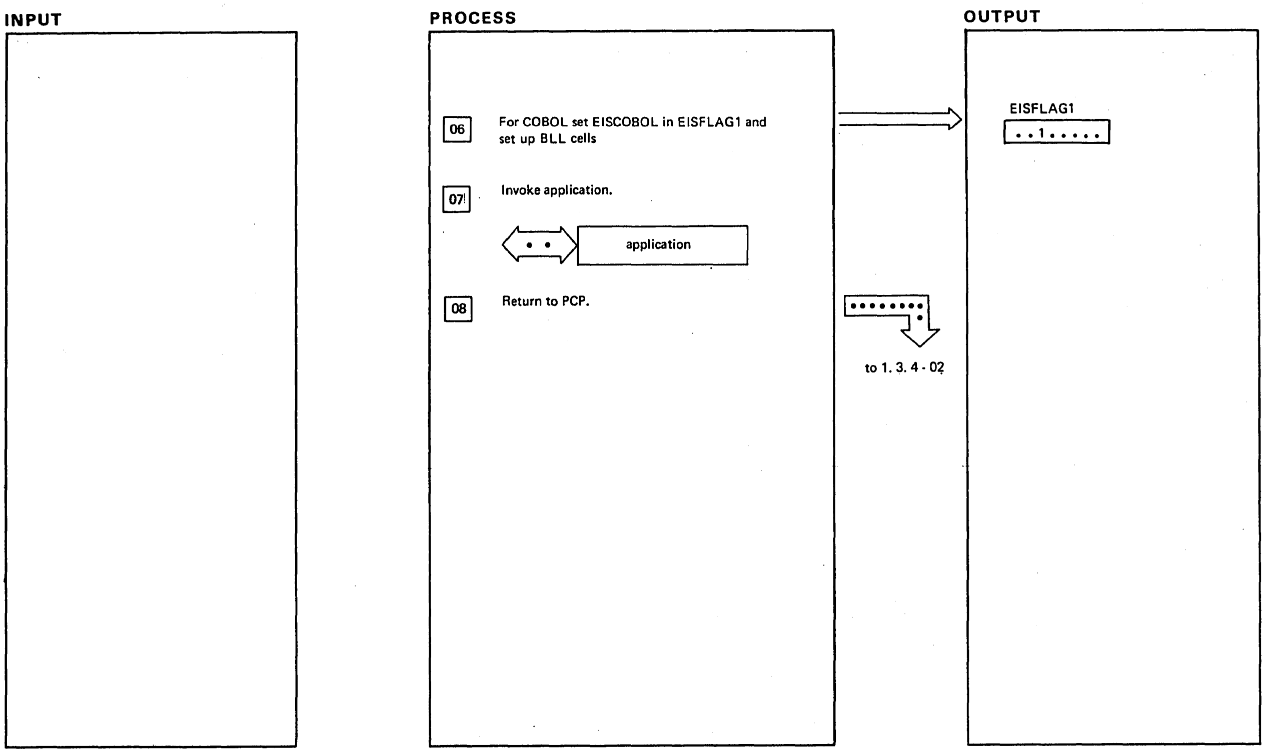
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 EIS will already exist if the application program has been invoked by an EXEC CICS LINK or XCTL. (This EIS contains an argument list to be passed to the application).<br>If EIS did not already exist then the argument list to be passed to the application is built in the system part of EIS.<br>EIS will also exist (but will not be initialized) if the task has previously been executed and has used the EXEC interface. In this case KCP acquires EIS, along with the TCA, and EIP just initializes it. | | | |
| 03 The argument list was built in CLASS = SHARED storage by the EXEC CICS RETURN function.<br>It is addressed by TCTTEEIA. | | | |
| 04 EISASM will be used in EIP where there are dependencies on Assembler. | | | |
| 05 EISPLI will be used in EIP when there are dependencies on PL/I (eg. length of labels). | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

EIP Initialization routine

Diagram — 6.5.1-02

Page 3 of 3



INPUT

PROCESS

06 For COBOL set EISCOBOL in EISFLAG1 and set up BLL cells

07 Invoke application.

08 Return to PCP.

OUTPUT

EISFLAG1

EIP Initialization routine

Diagram — 6.5.1-03

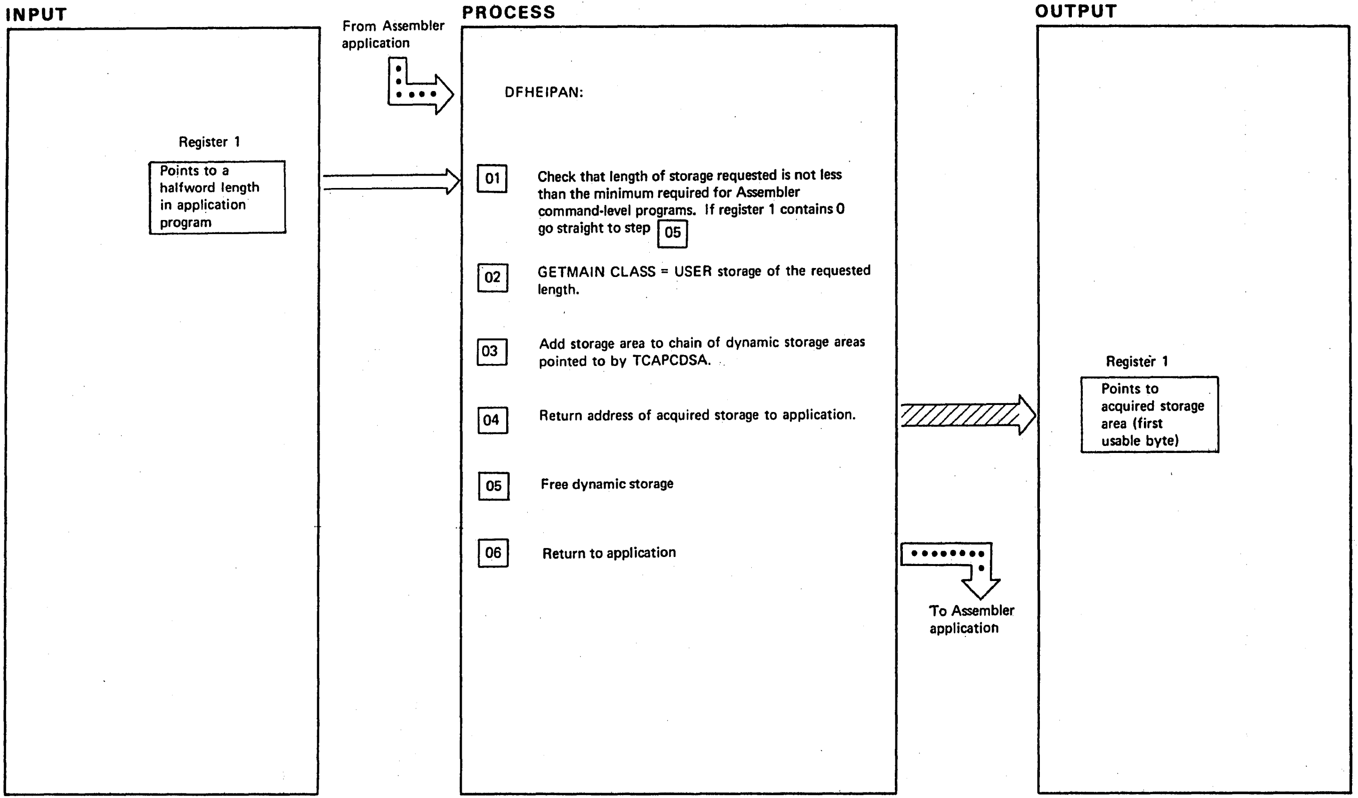
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 06 The BLL cells are<br>1. EIB<br>2. DFHCOMMAREA<br>3. points to itself<br>4. CSA | | | |
| 07 The application is invoked from EIP and it will return to this point in EIP | | | |
| 08 EIP normally returns to PCP after the point of invocation of EIP but control also reaches this point in EIP for an EXEC CICS XCTL in a PL/I program. In this case control passes to the XCTL code in PCP. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

EIP Initialization routine

Diagram — 6.5.1-03

Page 1 of 1



INPUT

Register 1

Points to a halfword length in application program

From Assembler application

PROCESS

DFHEIPAN:

01 Check that length of storage requested is not less than the minimum required for Assembler command-level programs. If register 1 contains 0 go straight to step 05

02 GETMAIN CLASS = USER storage of the requested length.

03 Add storage area to chain of dynamic storage areas pointed to by TCAPCDSA.

04 Return address of acquired storage to application.

05 Free dynamic storage

06 Return to application

To Assembler application

OUTPUT

Register 1

Points to acquired storage area (first usable byte)

Acquire Dynamic Storage for Command-Level Assembler Programs

Diagram – 6.5.2-01

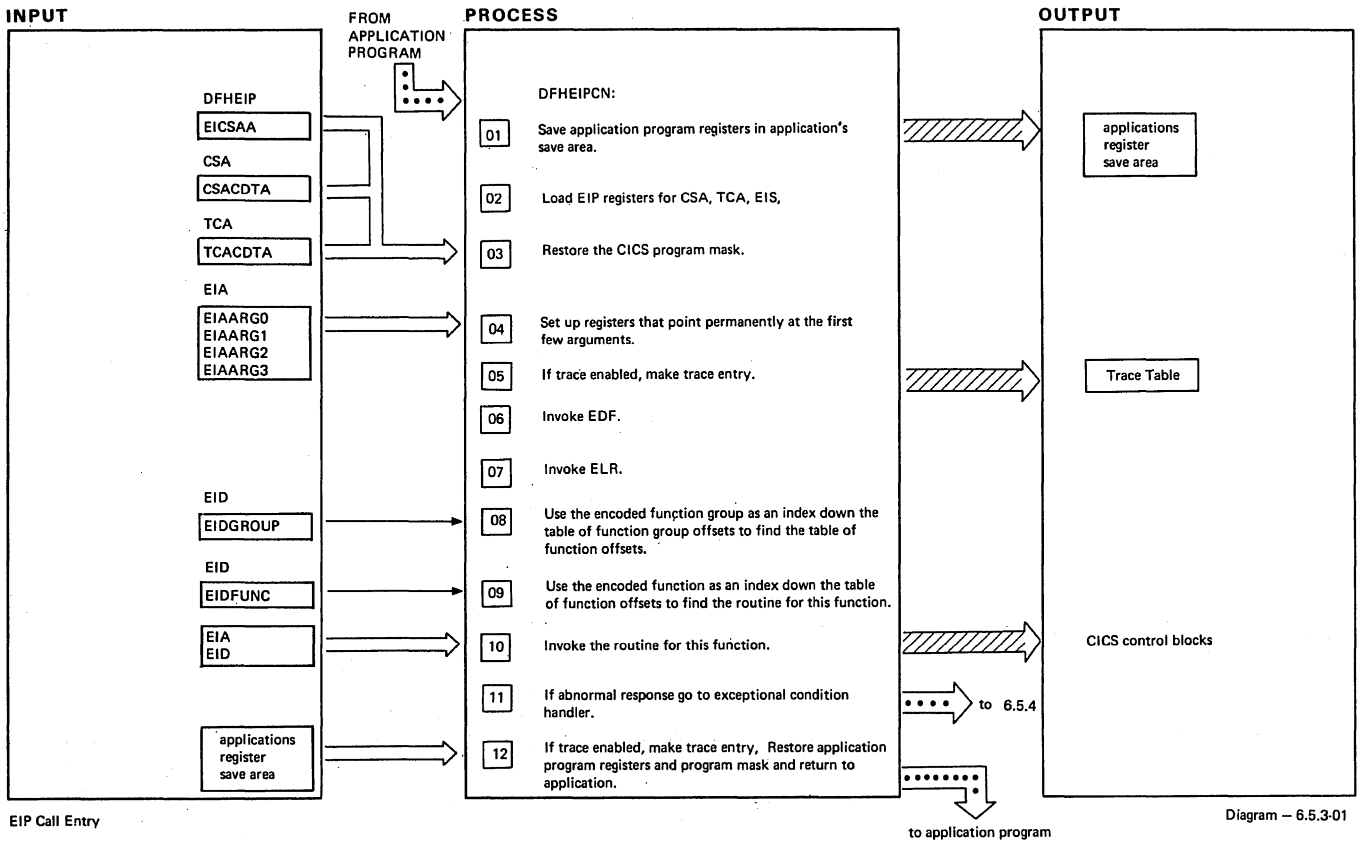
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The prolog code in an Assembler program which uses the command-level-interface is generated by the DFHEIENT macro. This prolog invokes DFHEIPAN to obtain the dynamic storage that the application will require, with register 1 pointing at the length of storage required.<br>The epilog code in an Assembler program which uses the command-level-interface is generated by the DFHEIRET macro. This epilog code invokes DFHEIPAN to free the dynamic storage. In this case register 1 contains 0. |  | DFHEIPAN |  |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Acquire Dynamic Storage for Command-Level Assembler Programs

Diagram – 6.5.2-01

Page 1 of 1



INPUT

DFHEIP
EICSAA
CSA
CSACDTA
TCA
TCACDTA
EIA
EIAARG0
EIAARG1
EIAARG2
EIAARG3
EID
EIDGROUP
EID
EIDFUNC
EIA
EID
applications register save area

FROM APPLICATION PROGRAM

PROCESS

DFHEIPCN:

01 Save application program registers in application's save area.

02 Load EIP registers for CSA, TCA, EIS,

03 Restore the CICS program mask.

04 Set up registers that point permanently at the first few arguments.

05 If trace enabled, make trace entry.

06 Invoke EDF.

07 Invoke ELR.

08 Use the encoded function group as an index down the table of function group offsets to find the table of function offsets.

09 Use the encoded function as an index down the table of function offsets to find the routine for this function.

10 Invoke the routine for this function.

11 If abnormal response go to exceptional condition handler.

12 If trace enabled, make trace entry, Restore application program registers and program mask and return to application.

OUTPUT

applications register save area

Trace Table

CICS control blocks

to 6.5.4

to application program

EIP Call Entry

Diagram – 6.5.3-01

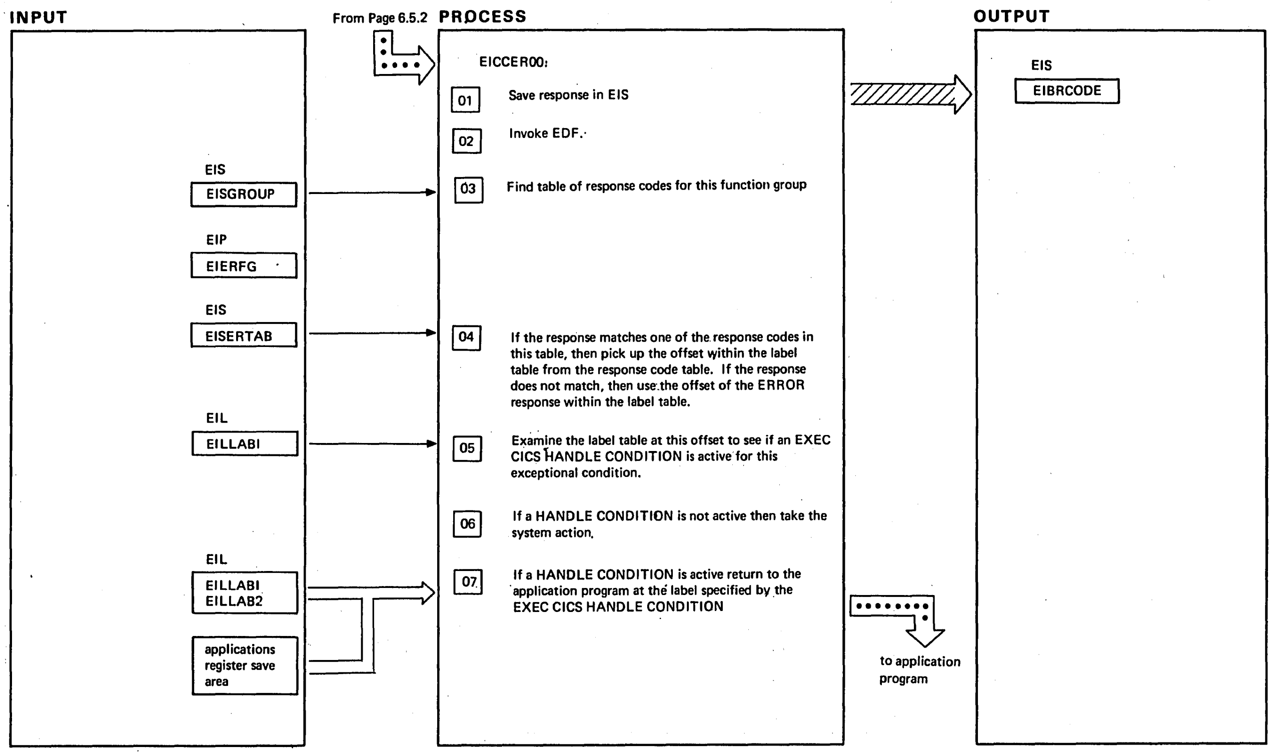
| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 01 | For COBOL they are saved in the TGT, for PL/I they are saved in the DSA, and for Assembler they are saved in DFHEISTG. In each case this is the register save area that register 13 points to when EIP is invoked. | | | |
| 02 | The CSA address was stored on DFHEIP by DFHEIPIN | | | 6.5.1 |
| 03 | The application program may run with a different program mask from CICS. (eg. PL/I) | | | |
| 04 | To minimise code size and maximise performance, four registers are reserved to point at the first four arguments (the other arguments are addressed indirectly via the argument list which is pointed at by register 1) | | | |
| 05 | The EIP trace entry records the application program's register 14, the function group and function encoding, the application program's save area address. | | EICC40 | |
| 06 | If debug mode is on in EIS, then invoke EDF. | | | |
| 07 | If ISC = YES was specified at CICS initialization, invoke DFHELR. | | | 6.5.5 |
| 08 | Function groups are by CICS component, e.g. TC, FC, TD etc. | | EICC60 | |
| 09 | Examples of functions are READ, WRITE, READNEXT within the FC function group. | | | |

| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 10 | There are separate load modules for each of the separate function groups of EIP, e.g., DFHETC, DFHEFC, DFHETD, etc. The individual function code may invoke routines in the nucleus of EIP, DFHEIP.<br>The routine for each function takes the arguments passed by a call statement and stores information in EIS and/or CICS control blocks. This may be done in individual function code or in routines which are common across a function group. If required, the appropriate CICS/VS module is then involved. | | | |
| 11 | Abnormal responses are handled separately. | | | 6.5.4 |
| 12 | The EIP trace entry for normal return, records the application program's register 14, the function group and function encoding and the response code. | | DFHEIPCX | |

EIP Call Entry

Diagram – 6.5.3-01

Page 1 of 1



EICCER00:

01 Save response in EIS

02 Invoke EDF.

03 Find table of response codes for this function group

04 If the response matches one of the response codes in this table, then pick up the offset within the label table from the response code table. If the response does not match, then use the offset of the ERROR response within the label table.

05 Examine the label table at this offset to see if an EXEC CICS HANDLE CONDITION is active for this exceptional condition.

06 If a HANDLE CONDITION is not active then take the system action.

07 If a HANDLE CONDITION is active return to the application program at the label specified by the EXEC CICS HANDLE CONDITION

EIP Exceptional Condition Handler

Diagram — 6.5.4-01

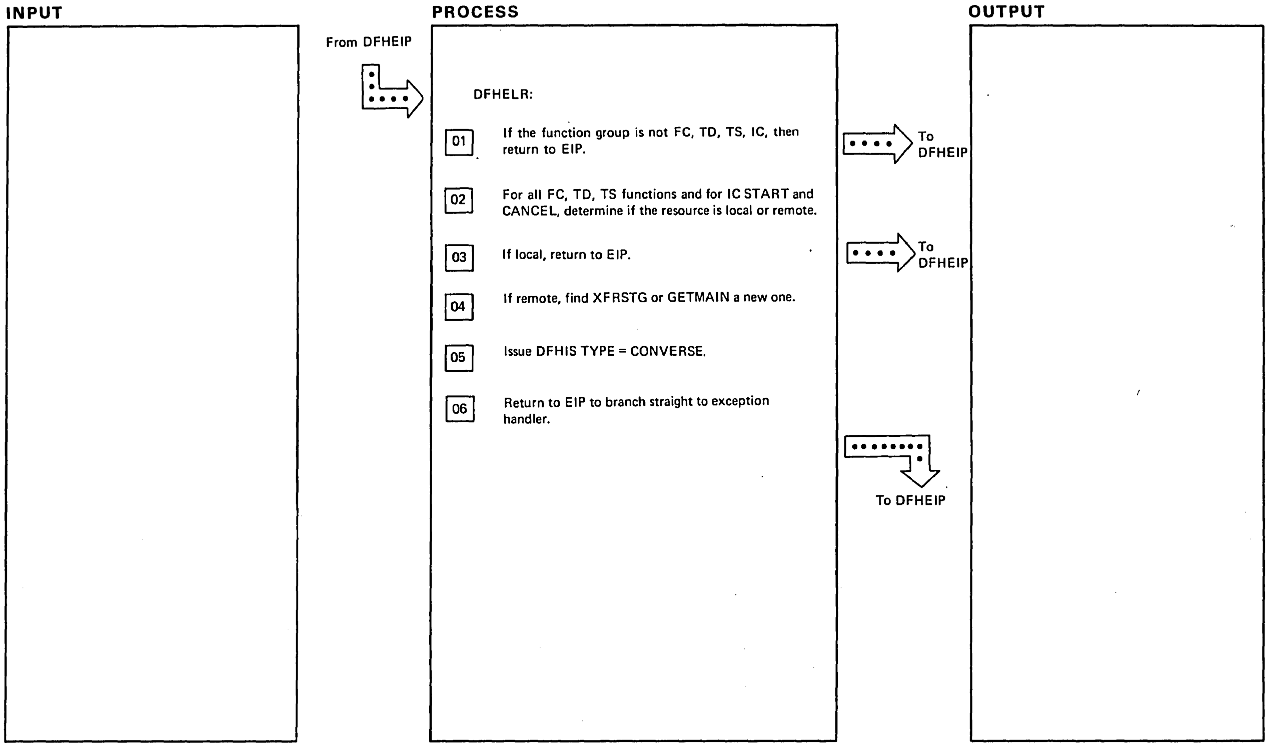
| | NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 01 | The response is saved in the user area of EIS (EIB) for the application to access. | | EICCER00 | |
| 02 | If debug mode is on in EIS, then invoke EDF. | | | |
| 03 | The response codes from CICS/VS are only unique within a CICS/VS component, so the function group is used to find the convert table of response codes for this component. | | EICCER20 | |
| 04 | | | EICCER30 | |
| 05 | | | EICCER60 | |
| 06 | For some exceptional conditions the system action is to return normally to the applications (via DFHEIPCX on 6.5.2-01). For most exceptional conditions the system action is to abend with an ABEND code that begins AEI or AEY. The last character of the abend code indicates the exceptional condition. | | EICCER70 | |
| 07 | The EIP trace entry for abnormal return, records the address of the exceptional condition label, the function group and function encoding, the response code. For COBOL, return is to a GO TO DEPENDING ON EIBGDI statement with EIBGDI set up by EIP.<br>For PL/1, return uses the PL/1 abnormal GOTO code. | | EICCND50 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

EIP Exceptional Condition Handler

Diagram — 6.5.4-01

Page 1 of 1



DFHELR:

01 If the function group is not FC, TD, TS, IC, then return to EIP.

02 For all FC, TD, TS functions and for IC START and CANCEL, determine if the resource is local or remote.

03 If local, return to EIP.

04 If remote, find XFRSTG or GETMAIN a new one.

05 Issue DFHIS TYPE = CONVERSE.

06 Return to EIP to branch straight to exception handler.

Local/Remote Resource Determination (Intersystem Communication)

Diagram – 6.5.5-01

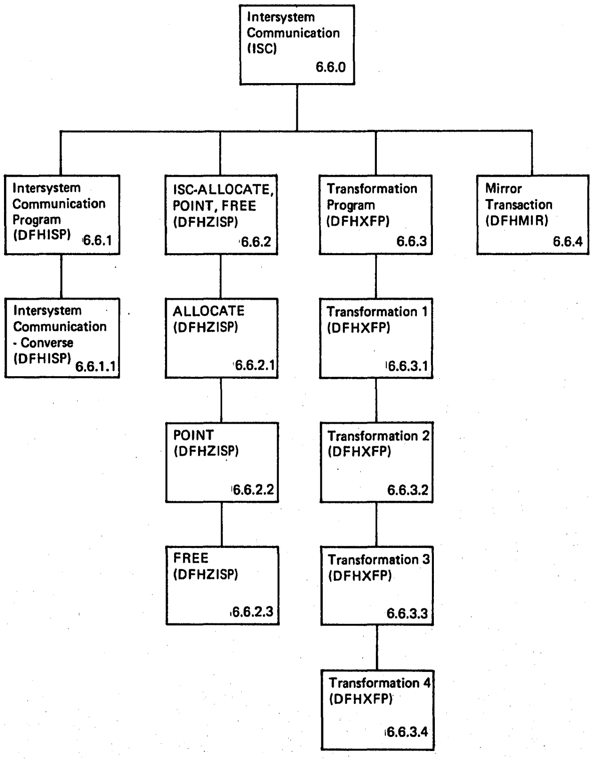
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 A resource is remote if the SYSID option is specified or if the entry in the associated CICS table indicates the resource to be remote. The CICS tables involved are:<br>FCT for FC<br>DCT for TD<br>TST for TS<br>PCT for IC | | | |
| 03 | LRCC00 | | |
| 04 XFRSTG is used to pass information through ISP to XFP. | | | |
| 05 | LRCC80 | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Local/Remote Resource Determination (Intersystem Communication)

Diagram – 6.5.5-01

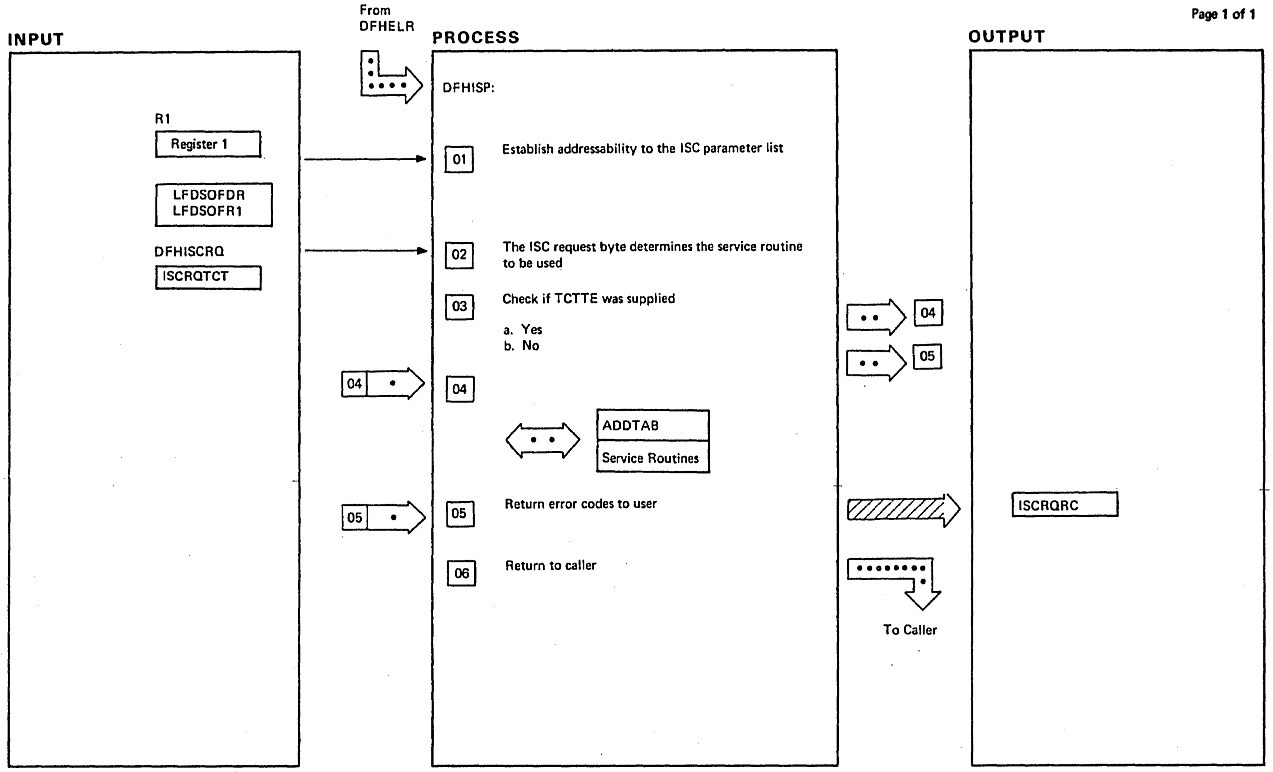
Page 1 of 1


Intersystem Communication (ISC) 6.6.0
Intersystem Communication Program (DFHISP) 6.6.1
ISC-ALLOCATE, POINT, FREE (DFHZISP) 6.6.2
Transformation Program (DFHXFP) 6.6.3
Mirror Transaction (DFHMIR) 6.6.4
Intersystem Communication - Converse (DFHISP) 6.6.1.1
ALLOCATE (DFHZISP) 6.6.2.1
POINT (DFHZISP) 6.6.2.2
FREE (DFHZISP) 6.6.2.3
Transformation 1 (DFHXFP) 6.6.3.1
Transformation 2 (DFHXFP) 6.6.3.2
Transformation 3 (DFHXFP) 6.6.3.3
Transformation 4 (DFHXFP) 6.6.3.4

Intersystem Communication – Visual Index

Diagram – 6.6.0-01

Page 1 of 1


INPUT
From DFHELR
PROCESS
OUTPUT
R1
Register 1
LFDSOFDR
LFDSOFR1
DFHISCRQ
ISCRQTCT
DFHISP:
01
Establish addressability to the ISC parameter list
02
The ISC request byte determines the service routine to be used
03
Check if TCTTE was supplied
a. Yes
b. No
04
05
04
ADDTAB
Service Routines
05
Return error codes to user
06
Return to caller
04
05
ISCRQRC
To Caller

Intersystem Communication Program (DFHISP)

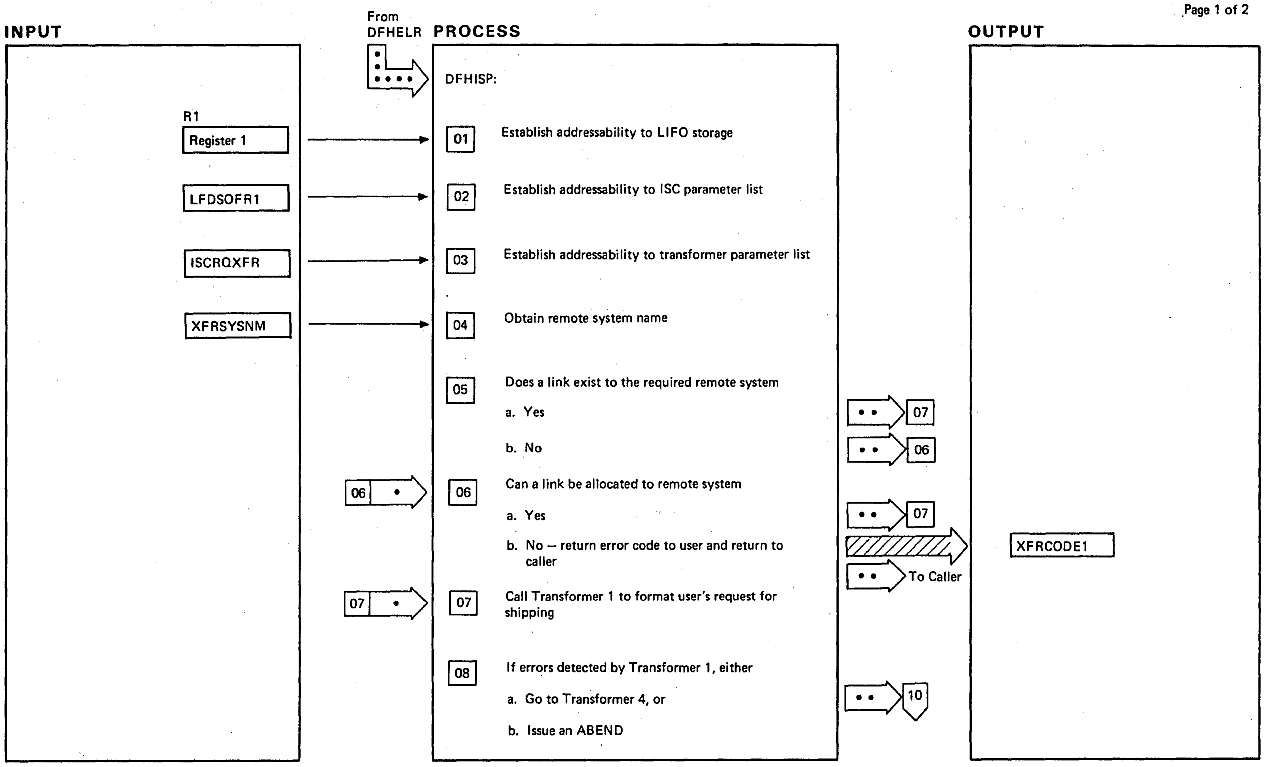
Diagram – 6.6.1-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The address of the ISC parameter list is in the register save area of the LIFO stack | | | |
| 02 The ISC request byte is used as an index to an address list of service routines | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Intersystem Communication Program (DFHISP)

Diagram – 6.6.1-01



DFHISP:

01 Establish addressability to LIFO storage

02 Establish addressability to ISC parameter list

03 Establish addressability to transformer parameter list

04 Obtain remote system name

05 Does a link exist to the required remote system
- a. Yes → 07
- b. No → 06

06 Can a link be allocated to remote system
- a. Yes → 07
- b. No — return error code to user and return to caller

07 Call Transformer 1 to format user's request for shipping

08 If errors detected by Transformer 1, either
- a. Go to Transformer 4, or → 10
- b. Issue an ABEND

Intersystem Communication — Converse (DFHISP)

Diagram — 6.6.1.1-01

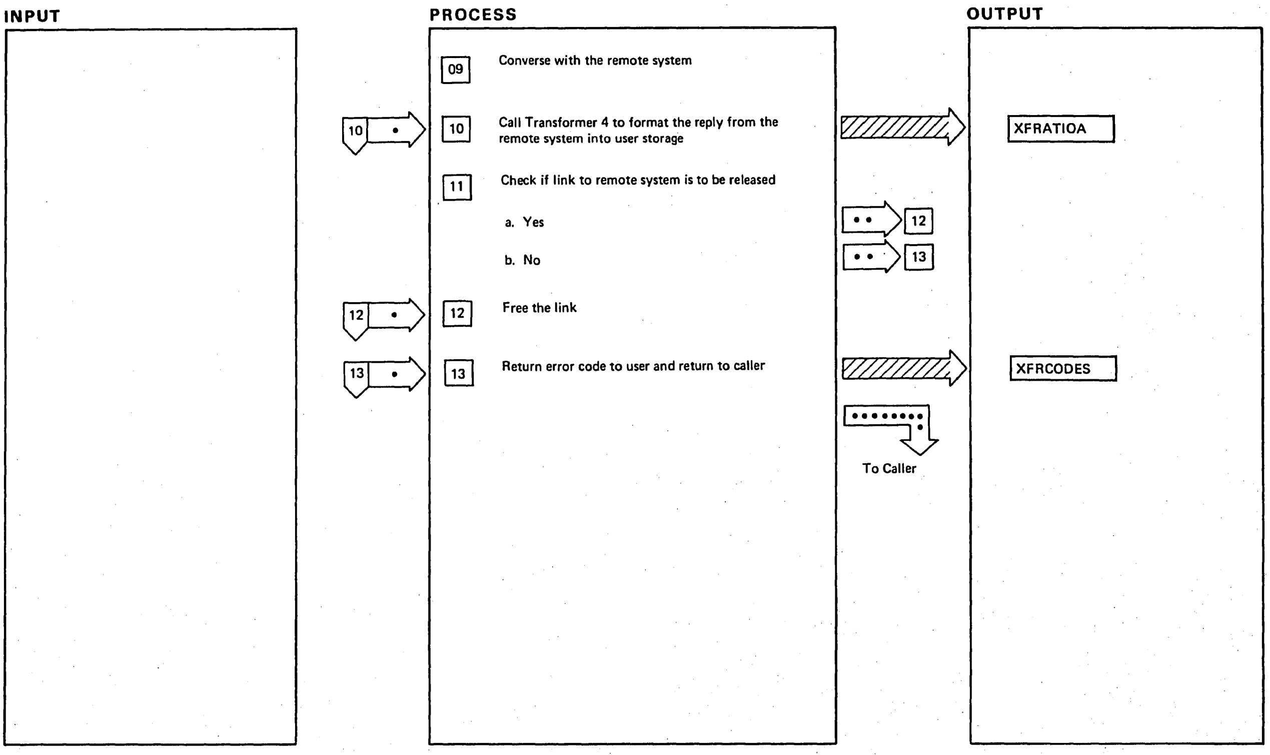
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 05 DFHTC TYPE = POINT,<br>REQUID = CSMI,<br>TRMIDNT = YES | | | |
| 06 DFHTC TYPE = ALLOCATE,<br>REQUID = CSMI,<br>TRMIDNT = YES | | | |
| 07 DFHIS TYPE = XFORM,<br>XFNUM = 1,<br>XFSTG = DFHXFRDS<br>DFHXFRDS is the start of the transformer parameter list | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Intersystem Communication — Converse (DFHISP)

Diagram — 6.6.1.1-01

Page 2 of 2



INPUT

PROCESS

OUTPUT

09 Converse with the remote system

10 Call Transformer 4 to format the reply from the remote system into user storage

11 Check if link to remote system is to be released

a. Yes

b. No

12 Free the link

13 Return error code to user and return to caller

Intersystem Communication – Converse (DFHISP)

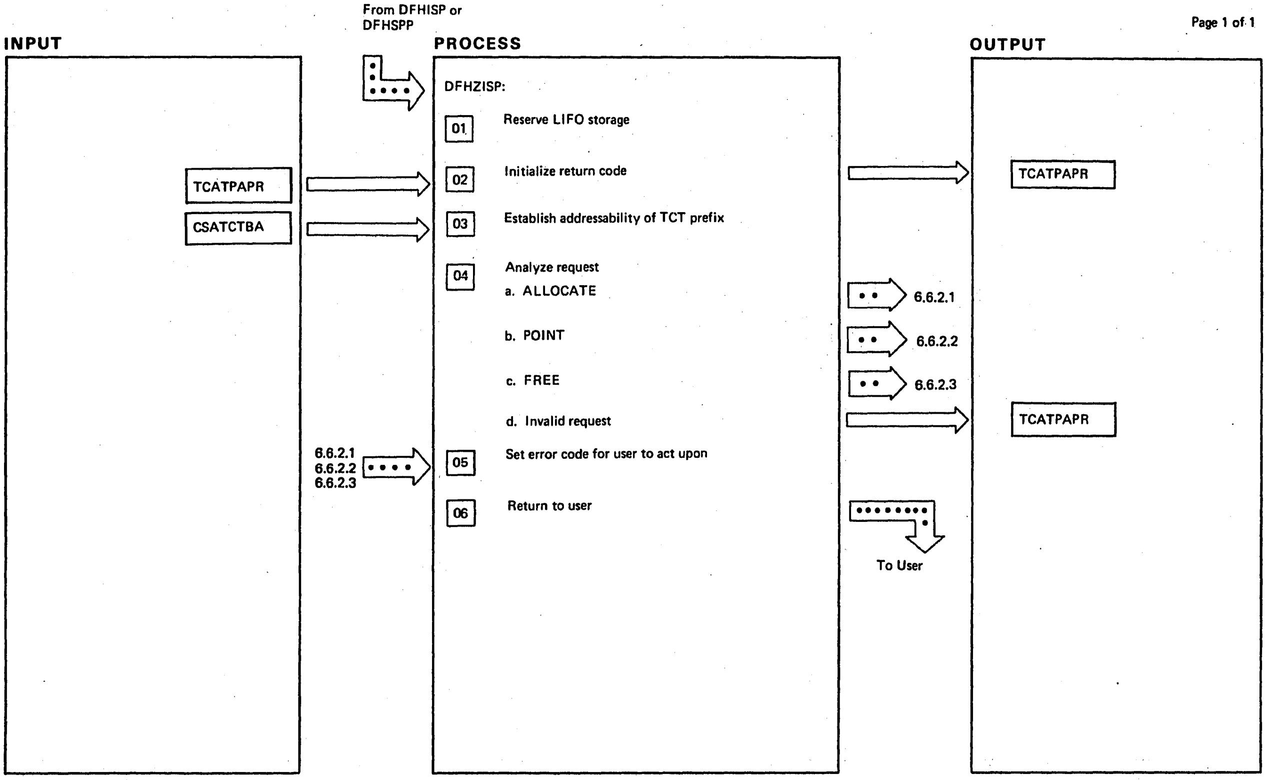
Diagram – 6.6.1.1-02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 09 DFHTC TYPE = (WRITE, WAIT, READ),<br>TERM = YES,<br>FMH = YES | | | |
| 10 DFHIS TYPE = XFORM,<br>XFNUM = 4,<br>XFSTG = DFHXFRDS<br>DFHXFRDS is the start of the transformer parameter list | | | |
| 12 DFHTC TYPE = FREE,<br>TERM = YES | | | |
| 13 DFHLFM TYPE = RETURN,<br>PLIST = LFMPLZ<br>LFMPLZ is LIFO parameter list | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Intersystem Communication – Converse (DFHISP)

Diagram – 6.6.1.1-02



ISC – ALLOCATE, POINT, FREE (DFHZISP)

Diagram – 6.6.2-01

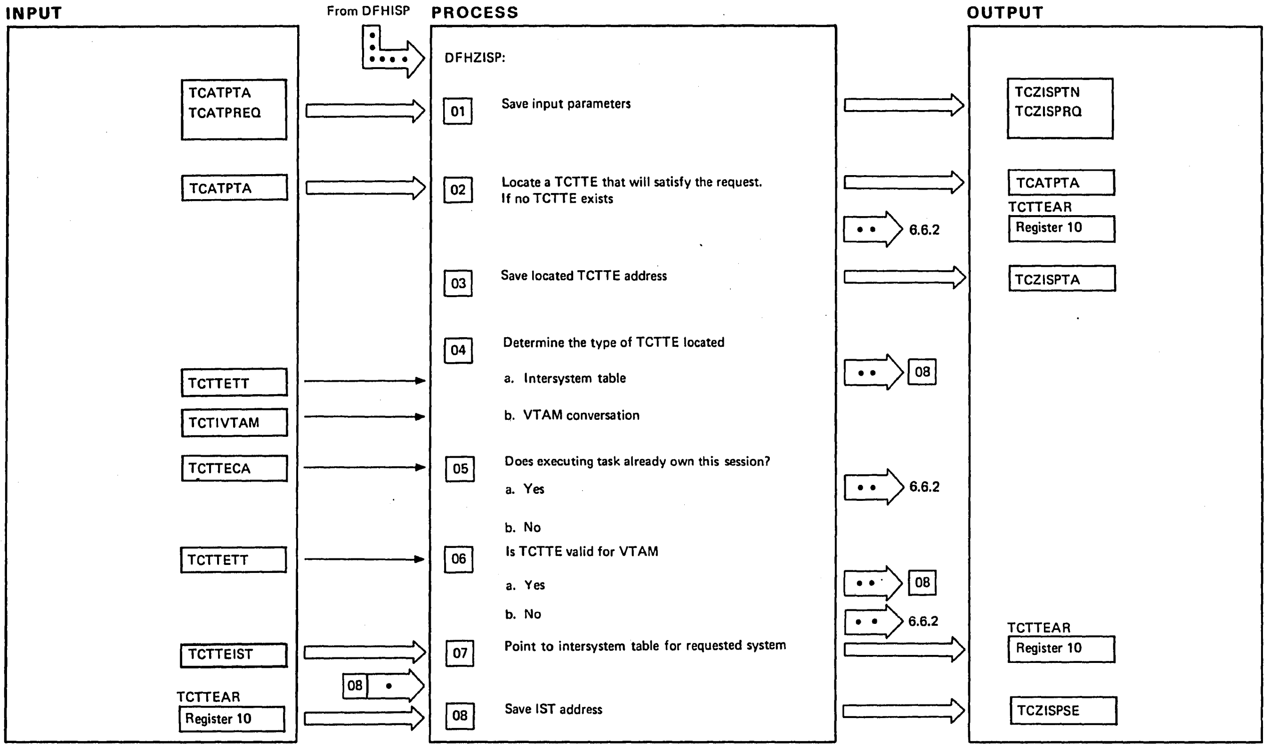
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 06 DFHLFM TYPE = RETURN,<br>PLIST = TCZISLFM | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

ISC – ALLOCATE, POINT, FREE (DFHZISP)

Diagram – 6.6.2-01

Page 1 of 4



INPUT

PROCESS

OUTPUT

From DFHISP

DFHZISP:

01 Save input parameters

Input: TCATPTA, TCATPREQ — Output: TCZISPTN, TCZISPRQ

02 Locate a TCTTE that will satisfy the request. If no TCTTE exists → 6.6.2

Input: TCATPTA — Output: TCATPTA, TCTTEAR Register 10

03 Save located TCTTE address

Output: TCZISPTA

04 Determine the type of TCTTE located

a. Intersystem table → 08

b. VTAM conversation

Input: TCTTETT, TCTIVTAM

05 Does executing task already own this session?

a. Yes → 6.6.2

b. No

Input: TCTTECA

06 Is TCTTE valid for VTAM

a. Yes → 08

b. No → 6.6.2

Input: TCTTETT

07 Point to intersystem table for requested system

Input: TCTTEIST — Output: TCTTEAR Register 10

08 Save IST address

Input: TCTTEAR Register 10 — Output: TCZISPSE

ALLOCATE (DFHZISP)

Diagram – 6.6.2.1-01

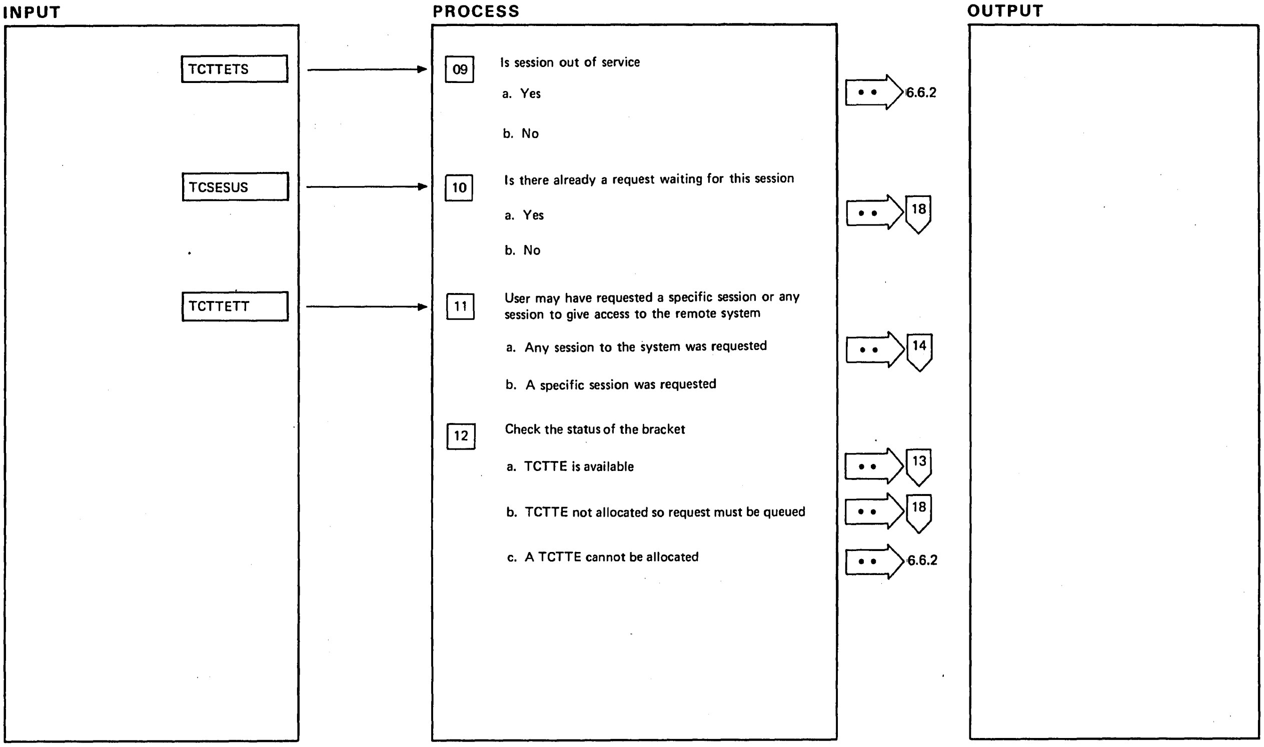
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 02 DFHTC CTYPE = LOCATE,<br>TERM = ID<br>The remote system identification is in TCATPTA | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

ALLOCATE (DFHZISP)

Diagram – 6.6.2.1-01

Page 2 of 4



INPUT

TCTTETS

TCSESUS

TCTTETT

PROCESS

09 Is session out of service

a. Yes → 6.6.2

b. No

10 Is there already a request waiting for this session

a. Yes → 18

b. No

11 User may have requested a specific session or any session to give access to the remote system

a. Any session to the system was requested → 14

b. A specific session was requested

12 Check the status of the bracket

a. TCTTE is available → 13

b. TCTTE not allocated so request must be queued → 18

c. A TCTTE cannot be allocated → 6.6.2

OUTPUT

ALLOCATE (DFHZISP)

Diagram – 6.6.2.1-02

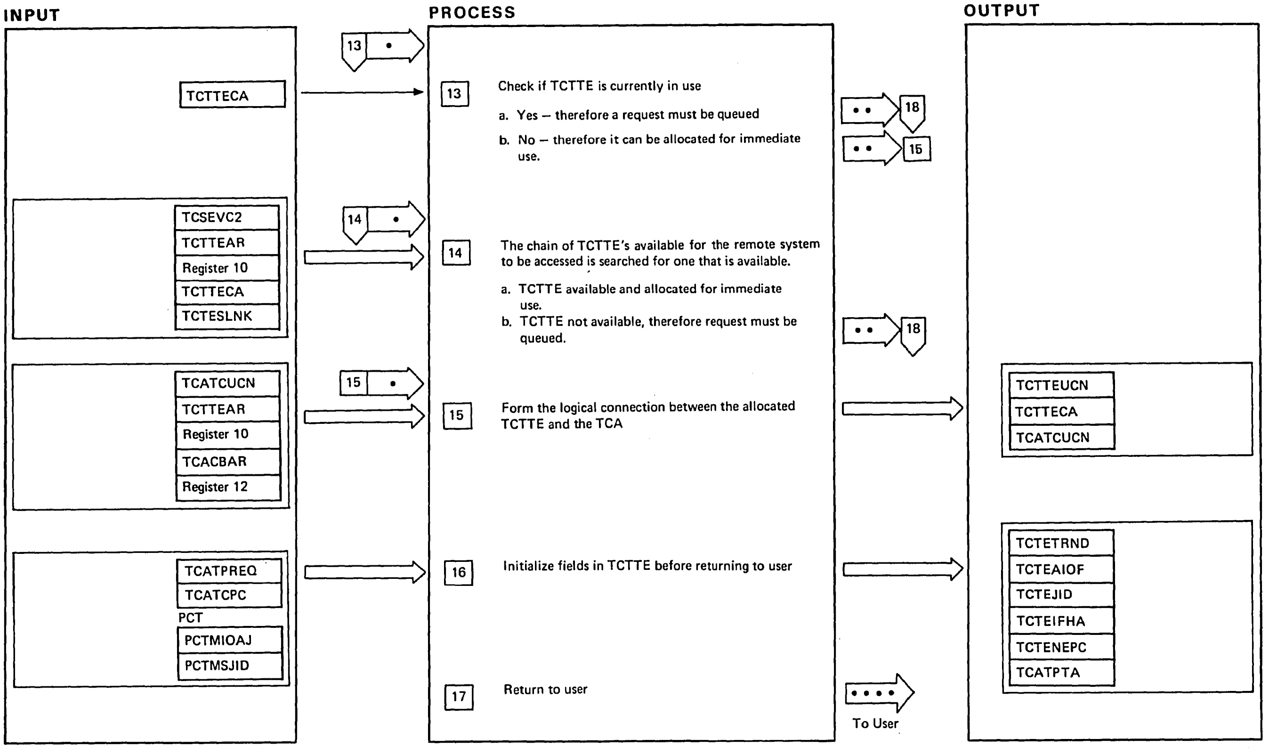
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 12 DFHTCPCM TYPE = BRACHECK<br>The TCTTE address is in TCATPTA | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

ALLOCATE (DFHZISP)

Diagram – 6.6.2.1-02

Page 3 of 4



13 Check if TCTTE is currently in use

a. Yes — therefore a request must be queued

b. No — therefore it can be allocated for immediate use.

14 The chain of TCTTE's available for the remote system to be accessed is searched for one that is available.

a. TCTTE available and allocated for immediate use.

b. TCTTE not available, therefore request must be queued.

15 Form the logical connection between the allocated TCTTE and the TCA

16 Initialize fields in TCTTE before returning to user

17 Return to user

ALLOCATE (DFHZISP)

Diagram — 6.6.2.1-03

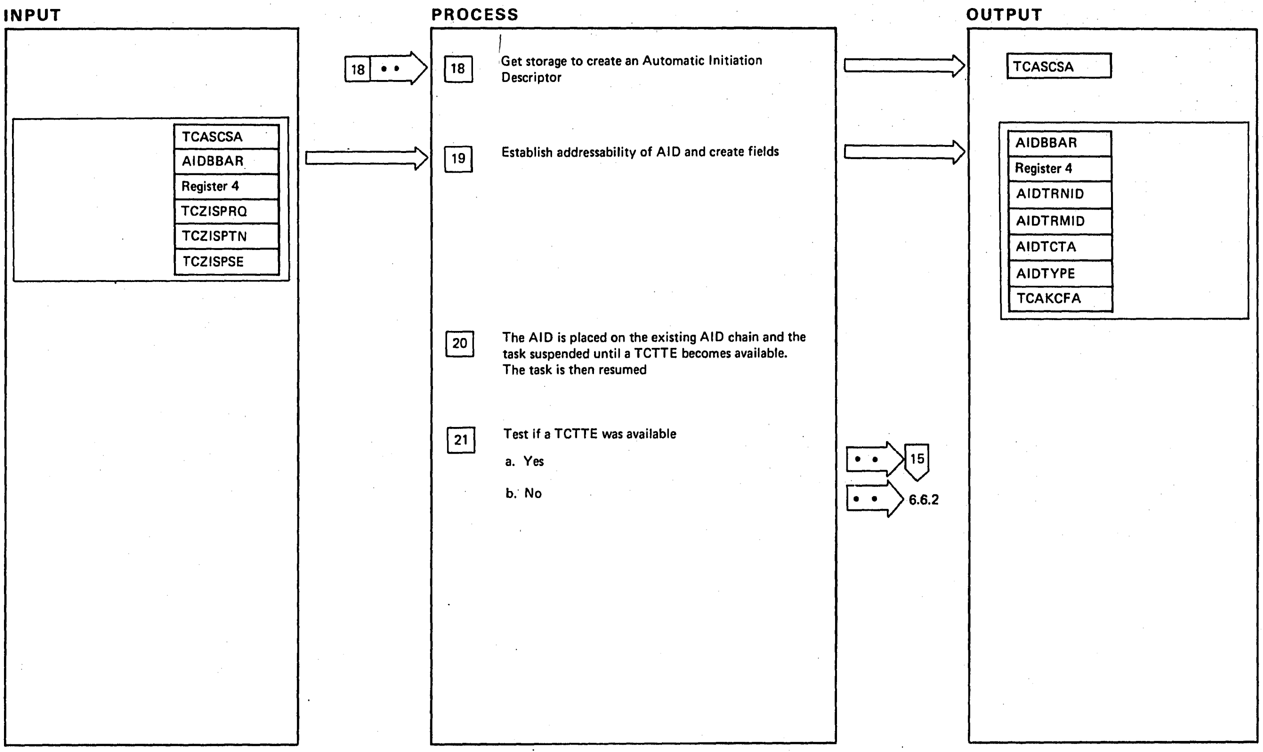
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 14 DFHTCPCM TYPE = BRACHECK<br>The TCTTE address is in TCATPTA | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

ALLOCATE (DFHZISP)

Diagram — 6.6.2.1-03

Page 4 of 4



18 Get storage to create an Automatic Initiation Descriptor

19 Establish addressability of AID and create fields

20 The AID is placed on the existing AID chain and the task suspended until a TCTTE becomes available. The task is then resumed

21 Test if a TCTTE was available

a. Yes

b. No

ALLOCATE (DFHZISP)

Diagram – 6.6.2.1-04

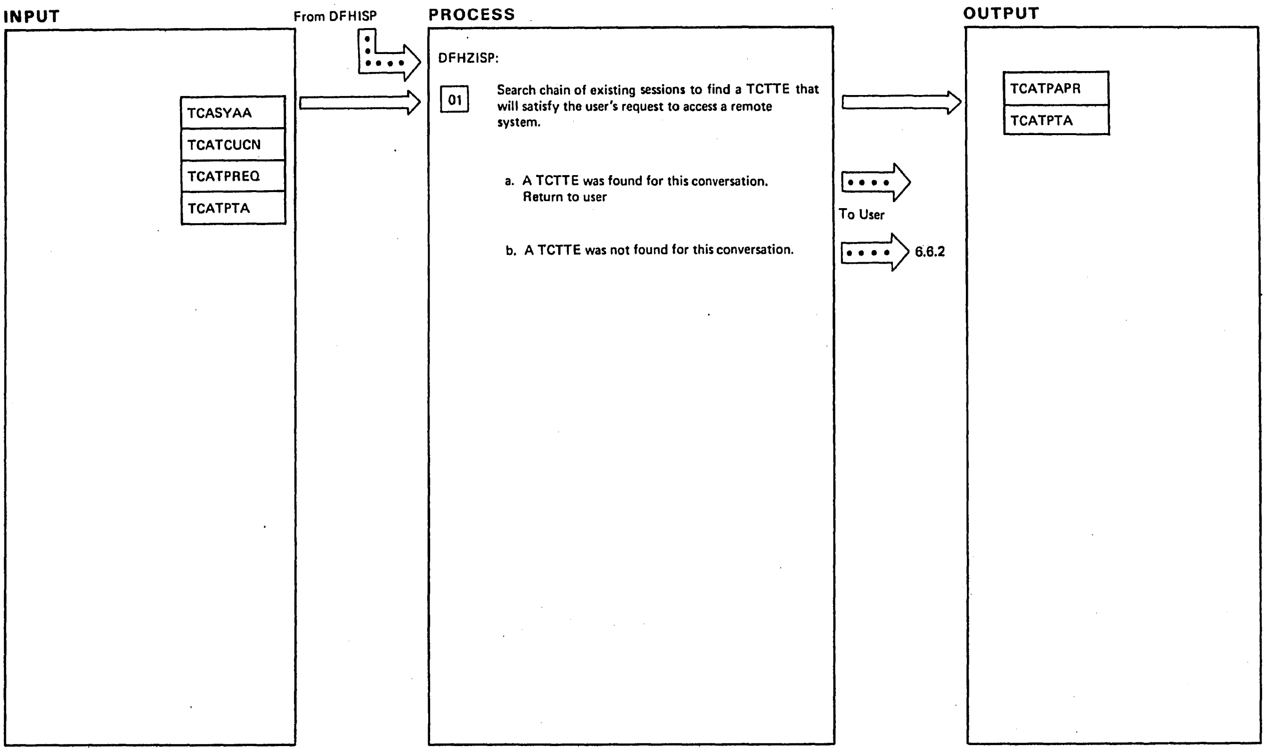
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 18 DFHSC TYPE = GETMAIN,<br>CLASS = AID,<br>INITIMG = 00,<br>NUMBYTE = AIDXTAD | | | |
| 20 DFHKC TYPE = SCHEDULE,<br>CLASS = IS,<br>TRMIDNT = YES | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

ALLOCATE (DFHZISP)

Diagram – 6.6.2.1-04

Page 1 of 1



INPUT

TCASYAA

TCATCUCN

TCATPREQ

TCATPTA

From DFHISP

PROCESS

DFHZISP:

01 Search chain of existing sessions to find a TCTTE that will satisfy the user's request to access a remote system.

a. A TCTTE was found for this conversation. Return to user

b. A TCTTE was not found for this conversation.

To User

6.6.2

OUTPUT

TCATPAPR

TCATPTA

POINT (DFHZISP)

Diagram – 6.6.2.2-01

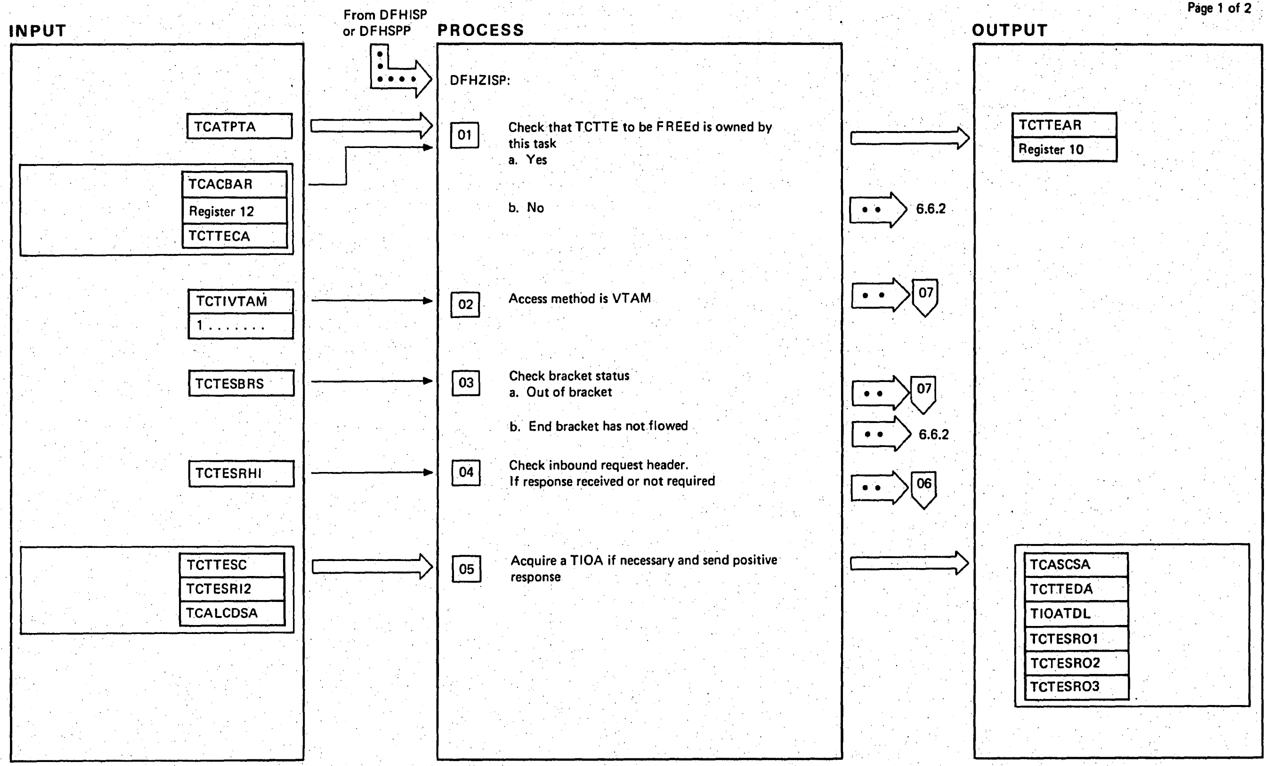
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

POINT (DFHZISP)

Diagram – 6.6.2.2-01

Page 1 of 2



INPUT

TCATPTA

TCACBAR
Register 12
TCTTECA

TCTIVTAM
1 . . . . . . .

TCTESBRS

TCTESRHI

TCTTESC
TCTESRI2
TCALCDSA

From DFHISP or DFHSPP

PROCESS

DFHZISP:

01 Check that TCTTE to be FREEd is owned by this task
a. Yes
b. No → 6.6.2

02 Access method is VTAM → 07

03 Check bracket status
a. Out of bracket → 07
b. End bracket has not flowed → 6.6.2

04 Check inbound request header. If response received or not required → 06

05 Acquire a TIOA if necessary and send positive response

OUTPUT

TCTTEAR
Register 10

TCASCSA
TCTTEDA
TIOATDL
TCTESRO1
TCTESRO2
TCTESRO3

FREE (DFHZISP)

Diagram – 6.6.2.3-01

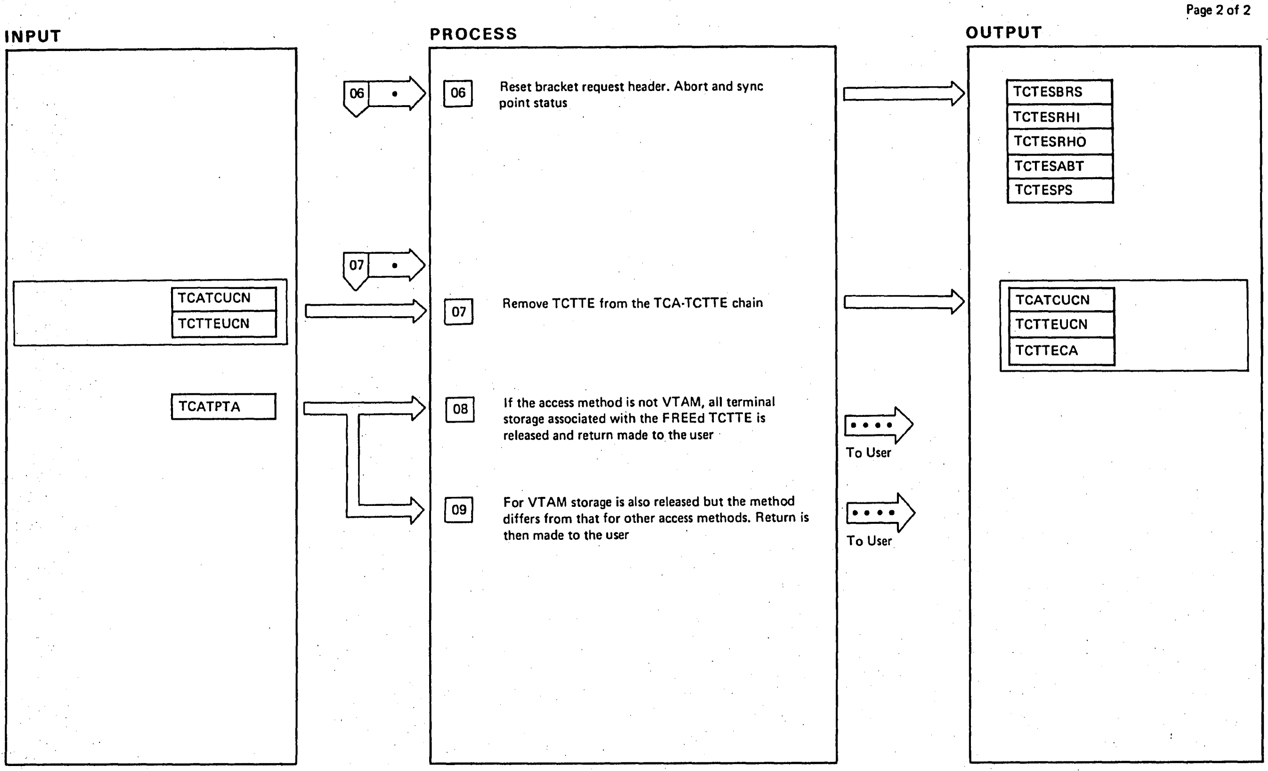
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 05 If the task does not have a TIOA then one must be acquired:<br>DFHSC TYPE = GETMAIN,<br>CLASS = TERMINAL,<br>NUMBYTE = 12<br>with zero data length so that a positive response can be sent:<br>DFHIS TYPE = IOR,<br>PLIST = ZISCRQ,<br>REQUEST = WRITEL,<br>TERM = TCTTEAR | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

FREE (DFHZISP)

Diagram – 6.6.2.3-01

Page 2 of 2



06 Reset bracket request header. Abort and sync point status

07 Remove TCTTE from the TCA-TCTTE chain

08 If the access method is not VTAM, all terminal storage associated with the FREEd TCTTE is released and return made to the user

09 For VTAM storage is also released but the method differs from that for other access methods. Return is then made to the user

FREE (DFHZISP)

Diagram – 6.6.2.3-02

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 08 DFHSC TYPE = FREEMAIN,<br>CLASS = TERMINAL,<br>RELEASE = ALL,<br>TERM = YES | | | |
| 09 DFHTC CTYPE = DETACH,<br>TERM = YES | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

FREE (DFHZISP)

Diagram – 6.6.2.3-02

From DFHISP
or DFHMIR

Page 1 of 1

**INPUT**

**PROCESS**

DFHXFP:

01 Transformation 1 – transform requested command and parameter list into a form suitable for transmission to remote system

02 Transformation2 – transform received request into a form suitable for execution

03 Transformation 3 – transform the command response into a suitable form for transmission of the reply.

04 Transformation 4 – transform reply into a form expected by application program.

Transformation 1
6.6.3.1

Transformation 2
6.6.3.2

Transformation 3
6.6.3.3

Transformation 4
6.6.3.4

**OUTPUT**

Transformation Program (DFHXFP)

Diagram – 6.6.3-01

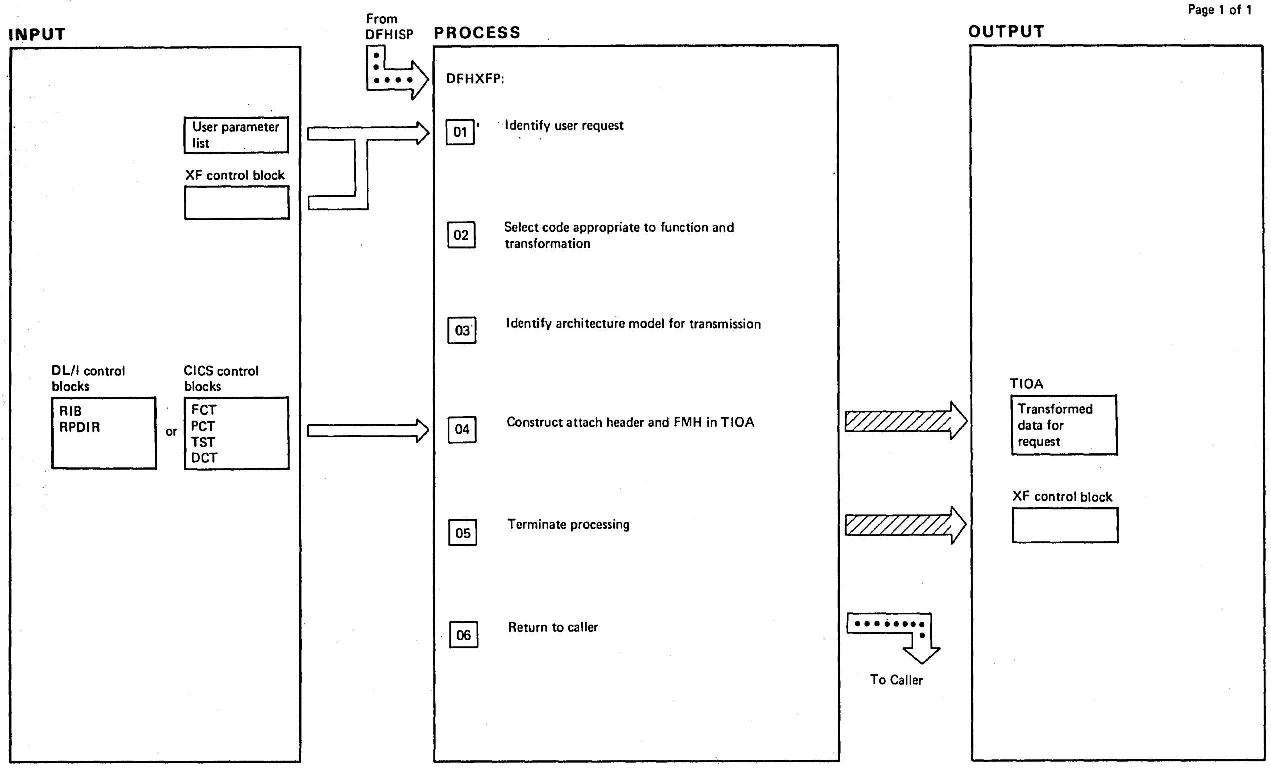
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Transformation Program (DFHXFP)

Diagram – 6.6.3-01

Page 1 of 1



Transformation 1 (DFHXFP)

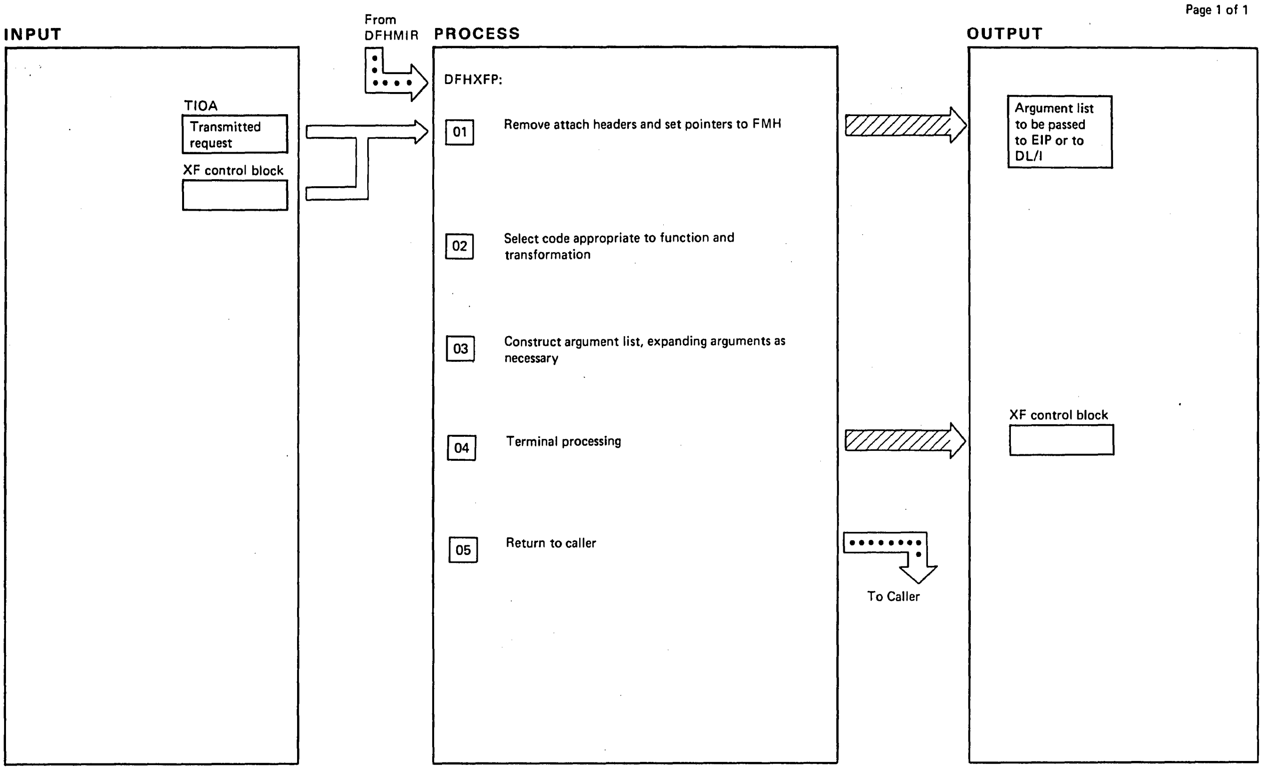
Diagram – 6.6.3.1-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 This is the paramater list passed from the application program to the EXEC interface program or to the DL/I interface program |  | INIT0100 |  |
| 02 |  | CMN0000 |  |
| 03 04 Done by routines such as TSWQ0100 |  |  |  |
| 05 |  | CMN0100 |  |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Transformation 1 (DFHXFP)

Diagram – 6.6.3.1-01



01 Remove attach headers and set pointers to FMH

02 Select code appropriate to function and transformation

03 Construct argument list, expanding arguments as necessary

04 Terminal processing

05 Return to caller

Transformation 2 (DFHXFP)

Diagram – 6.6.3.2-01

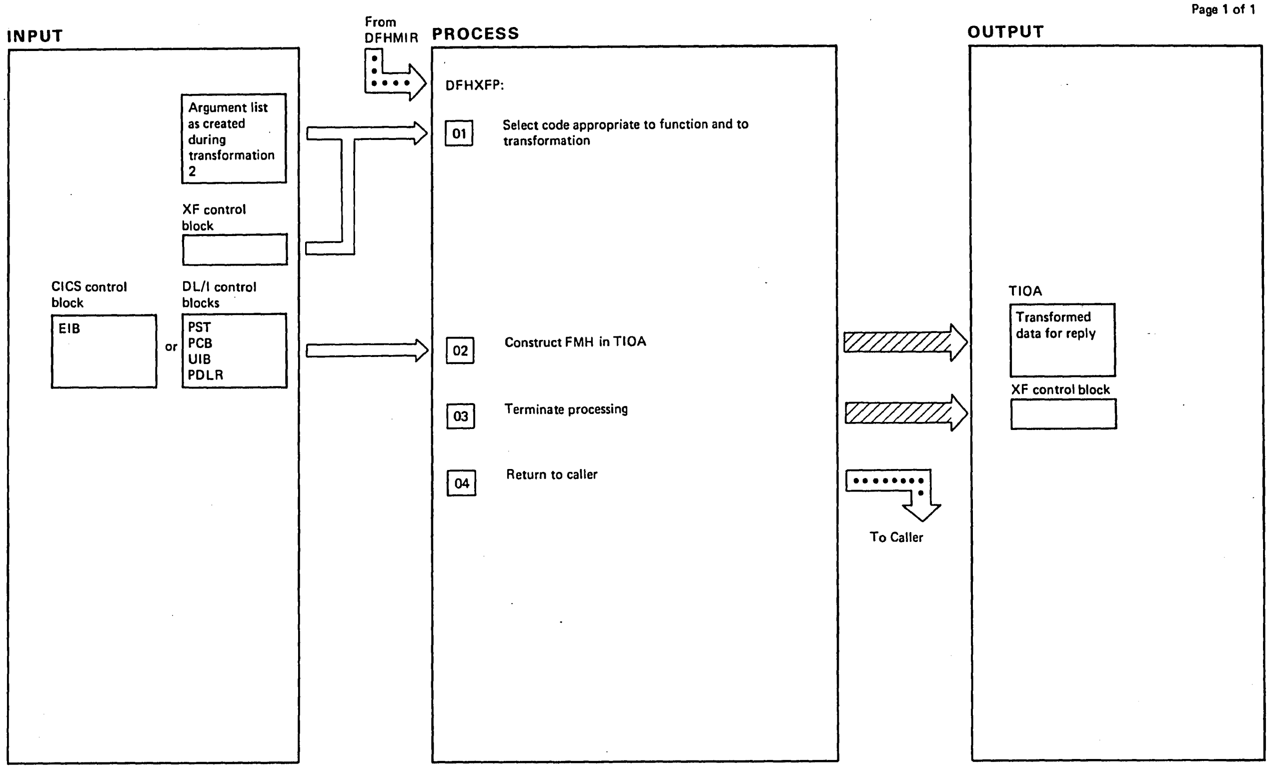
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 This is the data as constructed during transformation 1 | | INIT0200 | |
| 02 | | CMN0000 | |
| 03 Done by routines such as TSWQ0200 | | | |
| 04 | | CMN0200 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Transformation 2 (DFHXFP)

Diagram – 6.6.3.2-01

Page 1 of 1



Transformation 3 (DFHXFP)

Diagram — 6.6.3.3-01

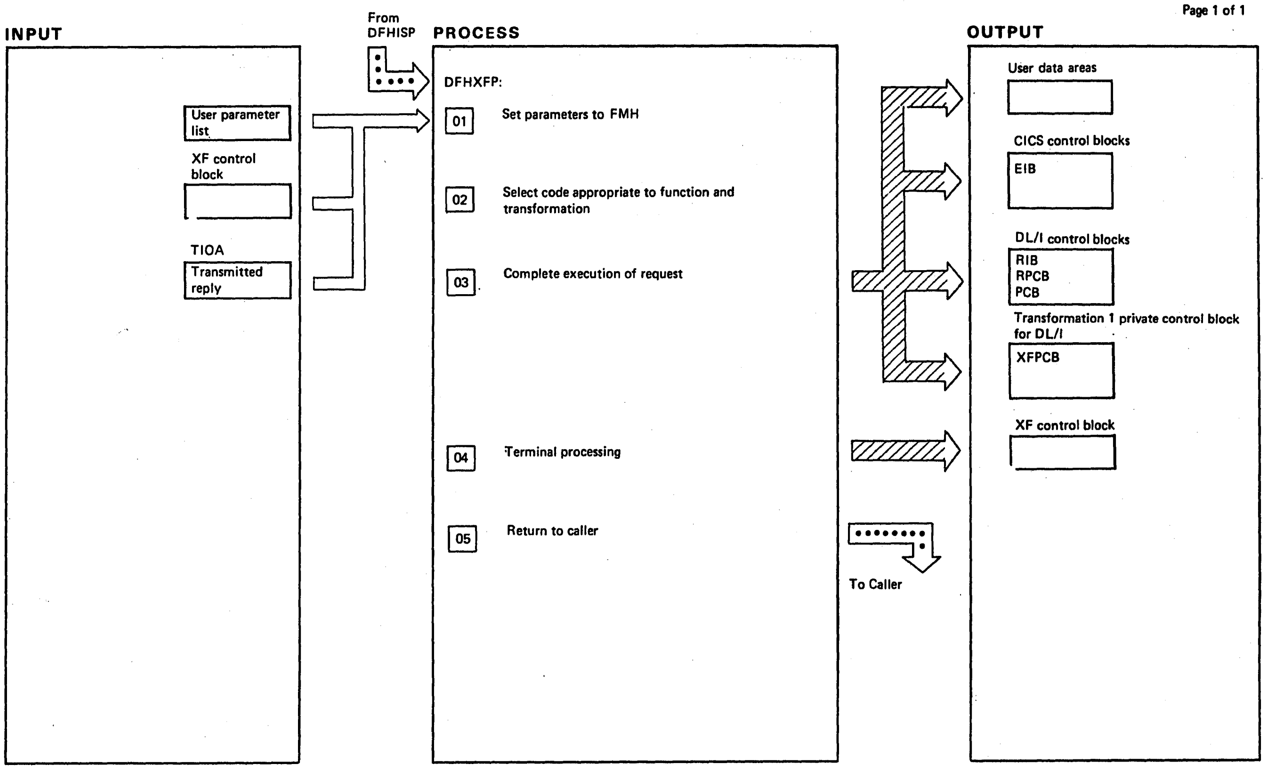
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 This is the argument list as constructed during transformation 2 | | INIT0300<br>CMN0000 | |
| 02 Done by routines such as TSWQ0300 | | | |
| 03 | | CMN0300 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Transformation 3 (DFHXFP)

Diagram — 6.6.3.3-01

Page 1 of 1



Transformation 4 (DFHXFP)

Diagram – 6.6.3.4-01

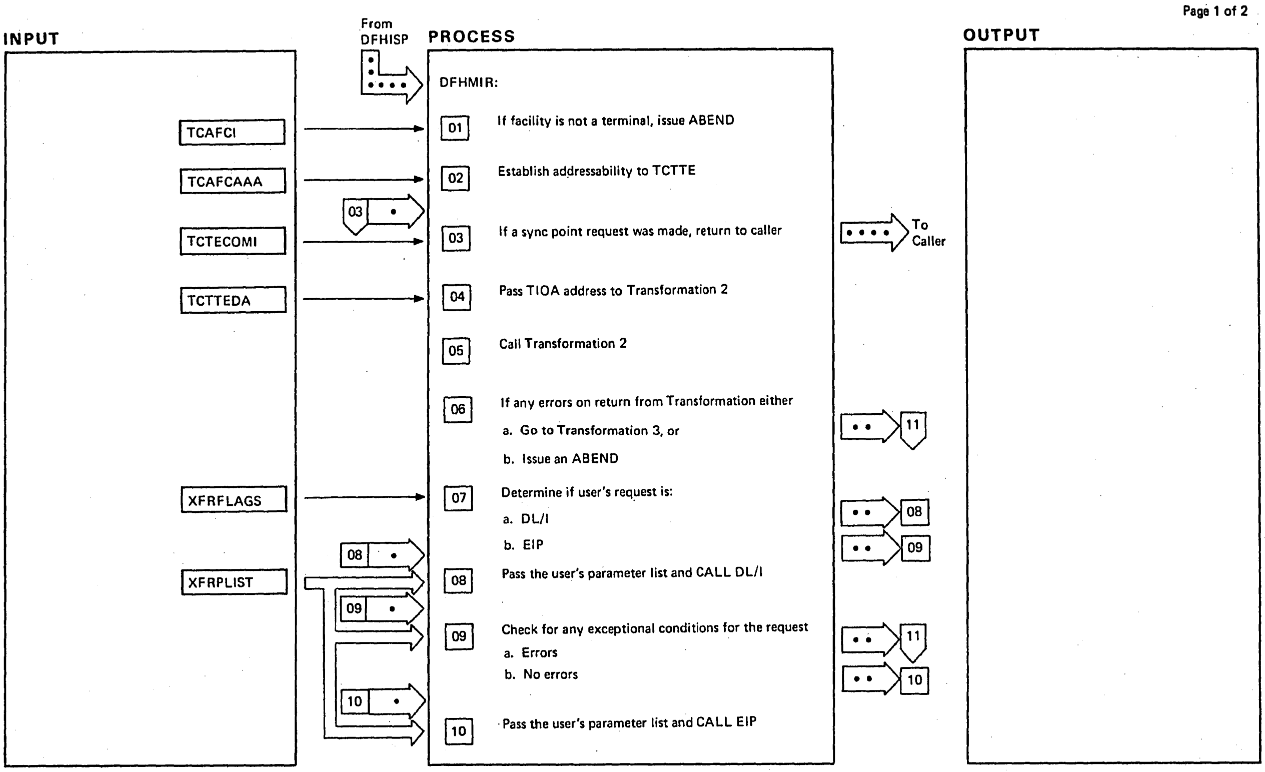
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 This is the data as constructed during transformation 3 | | INIT0400 | |
| 02 | | CMN0000 | |
| 03 Done by routines such as TSWQ0400 | | | |
| 04 | | CMN0400 | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Transformation 4 (DFHXFP)

Diagram – 6.6.3.4-01

Page 1 of 2


INPUT
From DFHISP
PROCESS
OUTPUT
DFHMIR:
TCAFCI
TCAFCAAA
TCTECOMI
TCTTEDA
XFRFLAGS
XFRPLIST
01 If facility is not a terminal, issue ABEND
02 Establish addressability to TCTTE
03 If a sync point request was made, return to caller
To Caller
04 Pass TIOA address to Transformation 2
05 Call Transformation 2
06 If any errors on return from Transformation either
a. Go to Transformation 3, or
b. Issue an ABEND
11
07 Determine if user's request is:
a. DL/I
b. EIP
08
09
08 Pass the user's parameter list and CALL DL/I
09 Check for any exceptional conditions for the request
a. Errors
b. No errors
11
10
10 Pass the user's parameter list and CALL EIP

Mirror Transaction (DFHMIR)

Diagram — 6.6.4-01

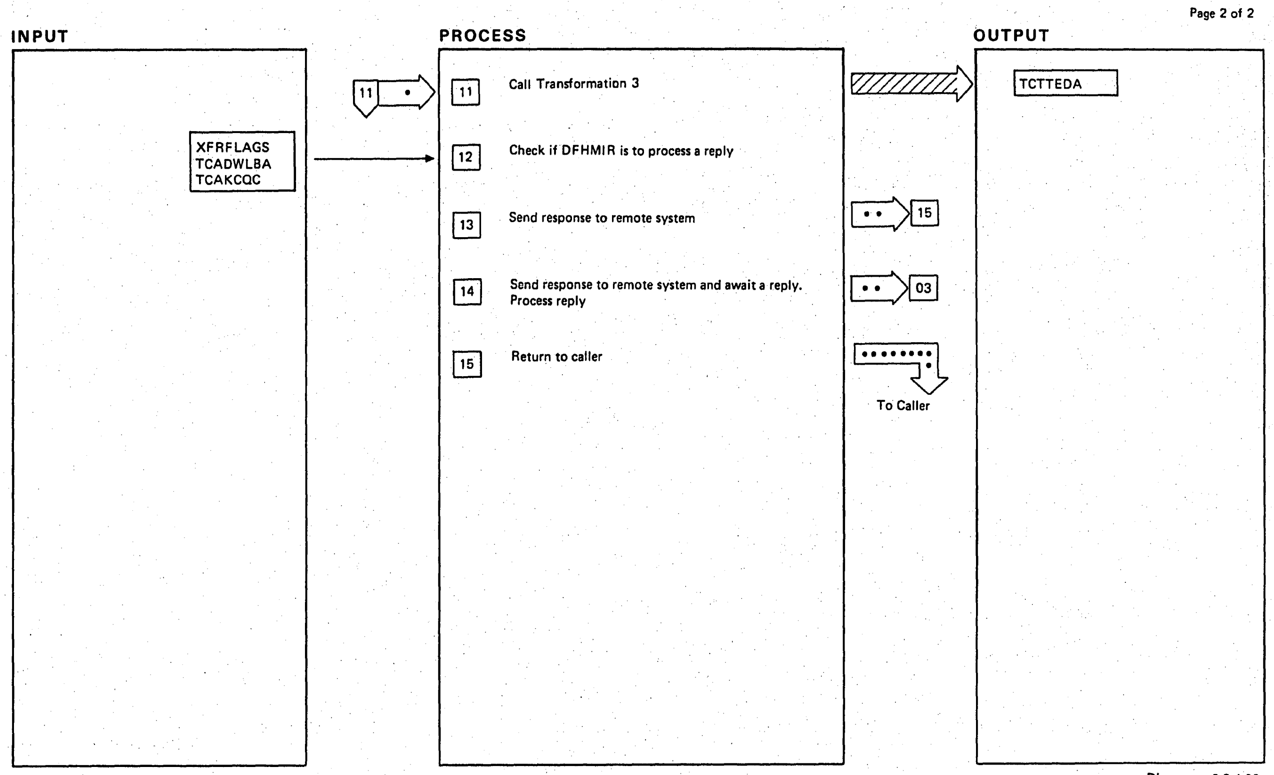
| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 01 | DFHPC TYPE = ABEND, ABCODE = AISA | | | |
| 03 | The sync point will be issued by DFHKCP | | | |
| 04 | The TIOA contains data about the user's request and was formatted by Transformation 1 on the user side of the intersystem link | | | |
| 05 | DFHIS TYPE=XFORM,<br>XFNUM=2,<br>XFSTG=XFRSAA<br>XFRSAA is the start of the transformation parameter list | | | |
| 06 | This action depends on the severity of the errors<br>b. DFHPC TYPE = ABEND,<br>ABCODE = AISB | | | |
| 08 | CALLDLI ASMTDLI MF=(E,1) | | | |
| 09 | EXEC CICS HANDLE CONDITION ERROR | | | |
| 10 | DFHEICAL | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| | | | |

Mirror Transaction (DFHMIR)

Diagram — 6.6.4-01

Page 2 of 2


INPUT
PROCESS
OUTPUT
XFRFLAGS
TCADWLBA
TCAKCQC
11
11 Call Transformation 3
12 Check if DFHMIR is to process a reply
13 Send response to remote system
15
14 Send response to remote system and await a reply. Process reply
03
15 Return to caller
To Caller
TCTTEDA

Mirror Transaction (DFHMIR)

Diagram – 6.6.4-02

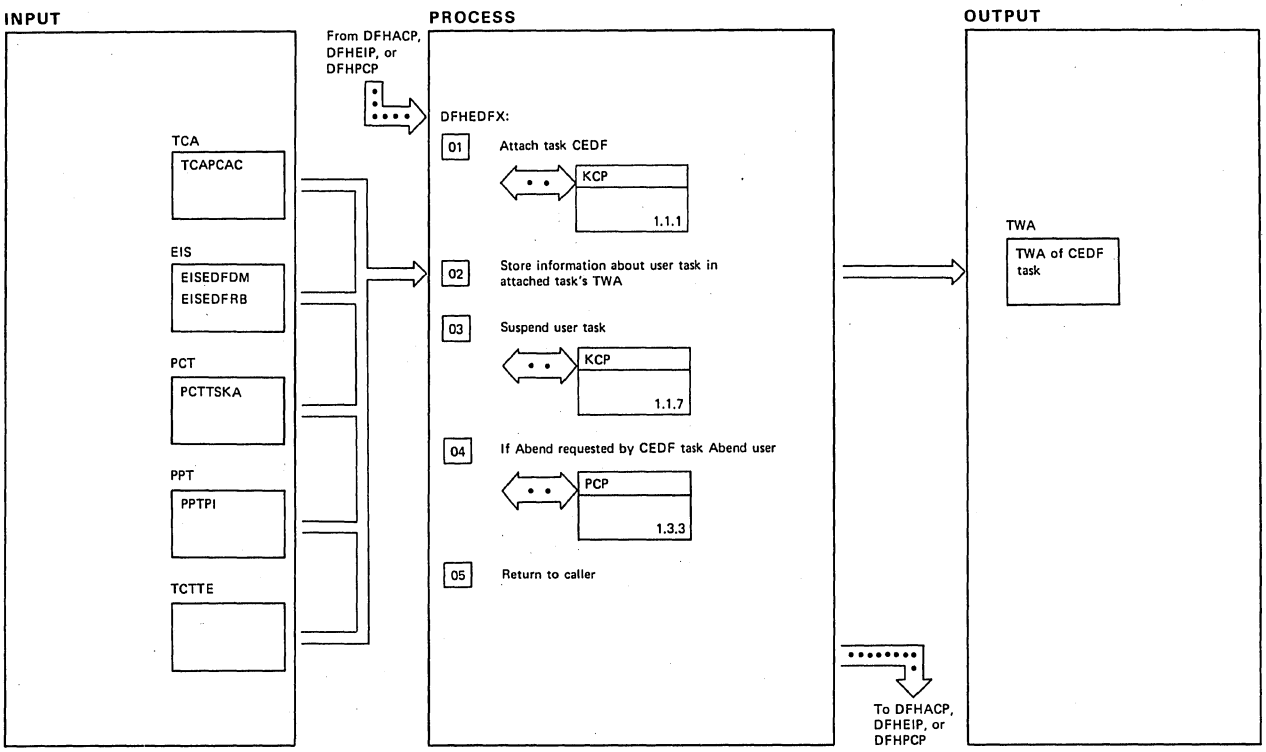
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 11 DFHIS TYPE = XFORM,<br>XFNUM = 3,<br>XFSTG = XFRSAA<br>XFRSAA is the start of the transformation parameter list | | | |
| 13 DFHTC TYPE = (WRITE, WAIT, LAST),<br>FMH = YES | | | |
| 14 DFHTC TYPE = (WRITE, WAIT, READ),<br>FMH = YES | | | |
| 15 DFHLFM TYPE = PCRETURN | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

Mirror Transaction (DFHMIR)

Diagram – 6.6.4-02

Page 1 of 1



EDF Task Switch Program

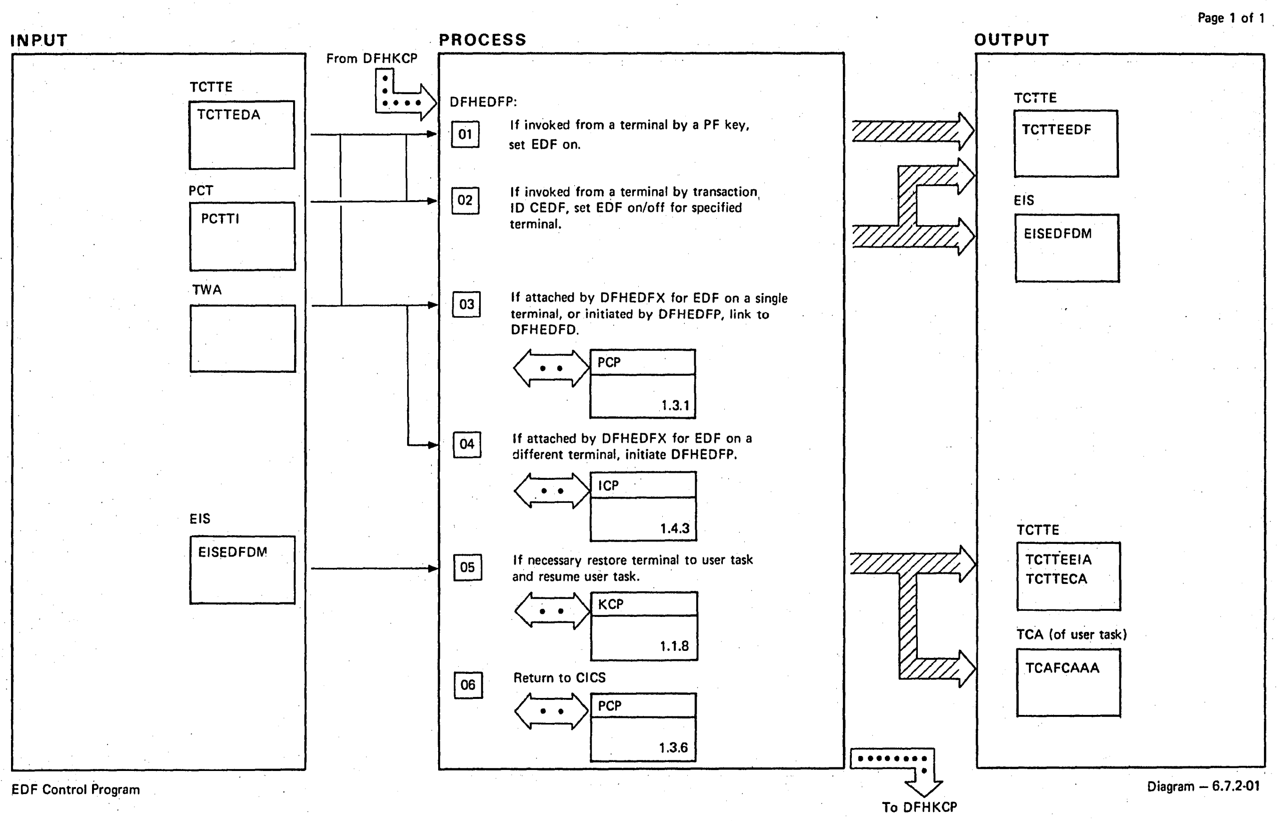
Diagram – 6.7.1-01

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 Do not attach another CEDF task if this task is already under the control of a CEDF task | | | |
| 02 Most information is copied directly. In some instances the address of the data is recorded. The CEDF task must take care not to write back into the user task's control blocks if the user task is purged. | | | |
| 03 The user task should be resumed by CEDF but may be resumed by KCP in a stall situation. | | | |
| 04 User may instruct CEDF to Abend the user task | | | |
| 05 Caller may be EIP, PCP, or ACP | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

EDF Task Switch Program

Diagram – 6.7.1-01


Page 1 of 1
INPUT
PROCESS
OUTPUT
From DFHKCP
TCTTE
TCTTEDA
PCT
PCTTI
TWA
EIS
EISEDFDM
DFHEDFP:
01 If invoked from a terminal by a PF key, set EDF on.
02 If invoked from a terminal by transaction ID CEDF, set EDF on/off for specified terminal.
03 If attached by DFHEDFX for EDF on a single terminal, or initiated by DFHEDFP, link to DFHEDFD.
PCP
1.3.1
04 If attached by DFHEDFX for EDF on a different terminal, initiate DFHEDFP.
ICP
1.4.3
05 If necessary restore terminal to user task and resume user task.
KCP
1.1.8
06 Return to CICS
PCP
1.3.6
To DFHKCP
TCTTE
TCTTEEDF
EIS
EISEDFDM
TCTTE
TCTTEEIA
TCTTECA
TCA (of user task)
TCAFCAAA

EDF Control Program

Diagram – 6.7.2-01

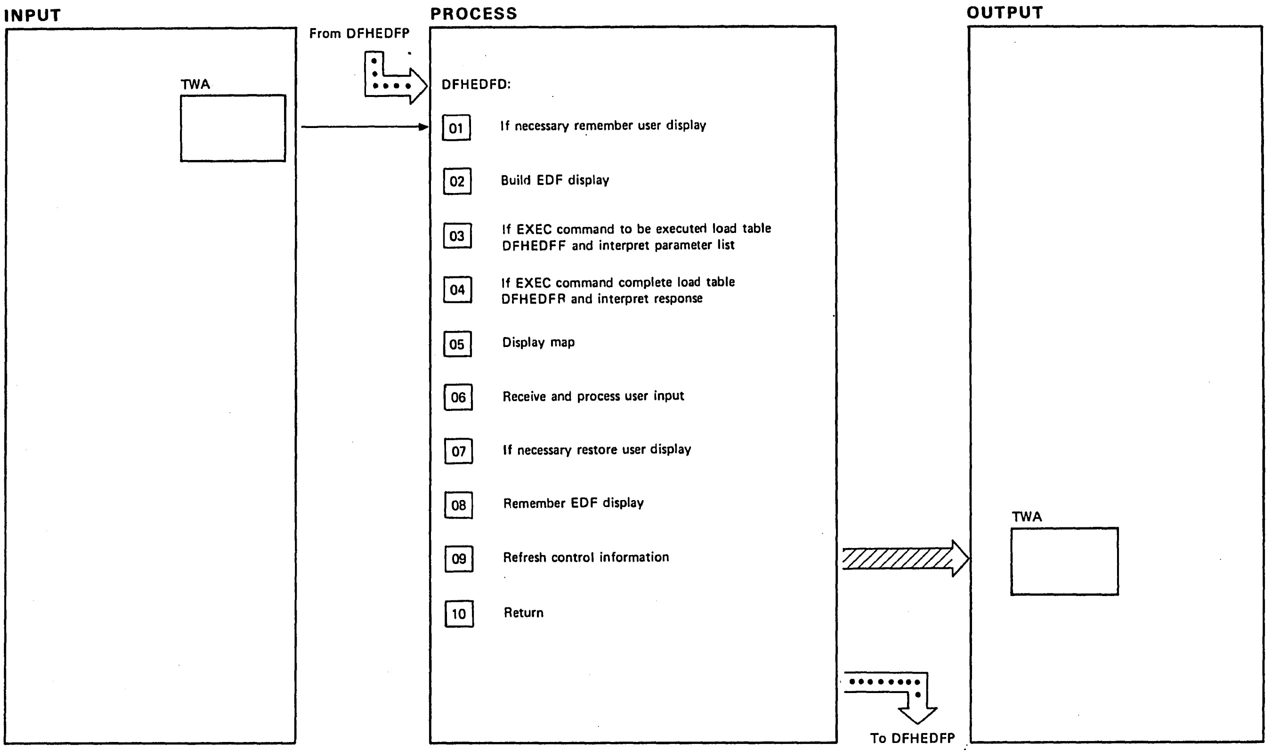
| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|
| 01 The PF key always means that the Execution Diagnostic Facility (EDF) is to be switched on for the direct terminal. | | | |
| 02 Input message is analyzed and named terminal located. If HLPI task is active but not yet in EDF mode, switch EDF bit in EIS; otherwise switch bit in TCTTE. | | | |
| 03 Using EDF on a separate terminal an IC PUT macro starts the CEDF task at the display terminal. The associated data is retrieved and stored in the TWA. | | | |
| 04 Using EDF on a separate terminal DFHEDFP is first invoked to initiate a task on the display terminal. The data stored in the TWA is passed to the initiated task by using an IC PUT macro. | | | |
| 05 Using EDF on a single terminal the terminal is removed from the CEDF task and restored to the user. | | | |
| 06 The CEDF task terminates, the user task having been allowed to continue. | | | |

| NOTES | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|

EDF Control Program

Diagram – 6.7.2-01

Page 1 of 1


INPUT
TWA
From DFHEDFP
PROCESS
DFHEDFD:
01 If necessary remember user display
02 Build EDF display
03 If EXEC command to be executed load table DFHEDFF and interpret parameter list
04 If EXEC command complete load table DFHEDFR and interpret response
05 Display map
06 Receive and process user input
07 If necessary restore user display
08 Remember EDF display
09 Refresh control information
10 Return
To DFHEDFP
OUTPUT
TWA

EDF Display Program

Diagram – 6.7.3-01

| NOTES | | ROUTINE | LABEL | REFERENCE |
|---|---|---|---|---|
| 01 | If the EDF display terminal is the same as the user terminal and the user screen may be different from the last saved screen, issue a read buffer and store a copy of the buffer on a temporary storage queue named X'FD01' \|\| TERMINAL – ID | | | |
| 02 | There are five types of display:<br>1. Function display, for example initialization, about to execute command<br>2. EIB display<br>3. Storage display<br>4. Stop conditions<br>5. Remembered display | | | |
| 03 | DFHEDFF is a table that relates particular arguments to their keyword and specifies the data type, length, etc. | | | |
| 04 | DFHEDFR is a table that relates particular responses to their exceptional condition. | | | |
| 05 | The display can have one of three formats defined by map.<br>DFHEDFM<br>DFHEDFN<br>DFHEDFP<br>all of which reside in mapset<br>DFHEDFM | | | |
| 06 | The user may modify data on the screen and then press ENTER or a PF key. The modified data is checked and then used to overwrite user data. The action defined for the PF key is then performed. | | | |
| 07 | If the user task is about to use the EDF display terminal in some way, restore the remembered user display. | | | |
| 08 | Remember current EDF display on temporary storage queue. | | | |
| 09 | Refresh control information (for example, number of current display) and store on temporary storage for next invocation of CEDF task. | | | |
| 10 | Return to DFHEDFP. | | | |

EDF Display Program

Diagram – 6.7.3-01

# SECTION 3. PROGRAM ORGANIZATION

CICS/OS/VS source-program listings are the key to the organization of CICS/OS/VS. You get into the listings from the method of operation (HIPO) diagrams. That is, you are aware of the general function being performed; that leads you to a set of HIPO charts; the HIPO charts identify specific modules involved in that function. Once you have located the module or routine name that interests you in the diagrams, you are ready to turn to a source-program listing to find the additional information you require.

CICS/OS/VS code is well commented. The processing steps within a module are described throughout the code. Many of the modules interact extensively with other CICS/OS/VS components during execution. Certain control blocks (the format and content of which are detailed in Section 2 of *CICS/OS/VS Program Logic: Data Areas*) provide vital system information.

Interface diagrams are given in this section for modules which interact frequently with other modules or make frequent use of external control blocks. The diagrams are supported by notes, keyed by numerical indicators (1, 2, and so on) to specific portions of the diagrams.

For the most part, modules are documented in this section in alphabetic order according to their symbolic names (DFHACP, and so on). However, certain modules are discussed in logical groupings, to aid you in using this section. The logical groupings are: Asynchronous Transaction Processing, Basic Mapping Support, and Intersystem Communication.

Not all modules need be described by interface diagrams. Hence, you will not find interface diagrams for all modules in this section. The modules that are not described (as well as those that are) are listed in the Directory (Section 1 of *CICS/OS/VS Program Logic: Data Areas*). As noted above, your next point of reference should be the CICS/OS/VS source code.

## ABNORMAL CONDITION PROGRAM (DFHACP)

The Abnormal Condition program (DFHACP) is a system service program used to analyze abnormal conditions that occur within the system and inform the terminal operator of problems. Errors can be classified as belonging in either of two broad categories:

- Task abnormal conditions, which are detected by CICS/VS management programs and are often due to an application program destroying system control information. When this happens, the task is terminated, the terminal operator is, if possible, informed of the error, and the error is logged at destination CSMT (computer system master terminal.)

- Operator errors, such as invalid transaction identifications, security key violations, or failure of an operator to sign on the system before attempting to communicate with CICS/VS. When this happens, the terminal operator is notified, and the error is logged at destination CSMT.

The notes below refer to Figure 3-1.

Notes:

1. DFHACP is invoked by Task Control whenever an invalid transaction code is detected.

2. DFHACP is invoked by Program Control whenever a task is abnormally terminated. The Sign-On Table (SNT) is loaded to obtain the operator identification (if present) for error messages. DFHACP returns to Program Control after the error message has been issued. When a task is abnormally terminated because of a stall purge condition, the stall purge count is increased by one and the transaction identification (from the Program Control Table) is included in the error message.

3. DFHACP communicates with Storage Control to obtain and release Terminal Input/Output Areas (TIOAs) and Transient Data Output Areas (TDOAs) for writing error messages.

4. If a user-written Program Error program (PEP) is provided, DFHACP issues a DFHPC TYPE=LINK macro instruction to transfer control to the program so that it can execute its function. All user-written PEPs communicate their requests through the TCA. The CSA is used to obtain the addresses of programs, and the Terminal Control Table terminal entry and line entry (TCTTE and TCTLE) are used to determine the status of the terminal and the line.

5. If a transaction that is using Dynamic Transaction Backout abends, DFHDBP will link to DFHACP to write messages to the user terminal and to CSMT.

6. Error messages are written to Transient Data destination, CSMT.

7. DFHACP communicates with Dump Control to initiate a complete dump, identified by the code AACA.

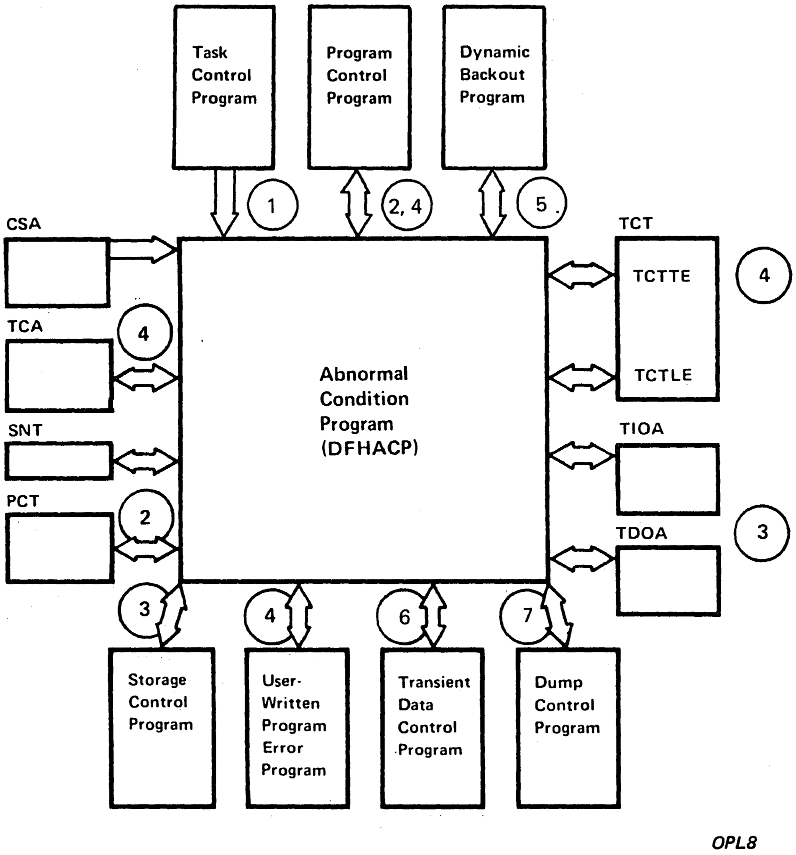


Figure 3-1. Abnormal Condition Program Interfaces

## ACTIVITY KEYPOINT PROGRAM (DFHAKP)

The Activity Keypoint program (DFHAKP) is invoked by transaction CSKP when activity keypointing is required. It invokes the Keypoint program (DFHKPP) to perform the keypointing. When control is returned to DFHAKP, an exit can be taken to a user-written routine (DFHUAKP) for further processing. DFHAKP writes a time stamp to the master terminal (CSMT) at completion of activity keypointing. For further details, see the HYPO diagram for DFHAKP in Section 2 and "Keypoint Program (DFHKPP)" in this section.

## ASYNCHRONOUS TRANSACTION PROCESSING GROUP (ATP)

The Asynchronous Transaction Control program (part of the Asynchronous Transaction Processing group, ATP) controls the initiation and data handling of all asynchronous tasks submitted as part of a batch. The batch consists of one or more CICS/VS transactions, along with any associated data, entered into CICS/VS through the Asynchronous Transaction Input Processing programs (DFHRD1 and DFHRD2).

When an entire batch has been submitted, the transactions are executed asynchronously with other possible terminal activity by the originating terminal. When all transactions have been processed, the output of the batch can be transmitted back to a terminal, depending on how the batch was entered, by the Asynchronous Transaction Output Processing programs (DFHWT1 and DFHWT2). If not transmitted automatically, the output remains queued until it is requested by the originating terminal or an alternate terminal. A batch may also be deleted by the Asynchronous Queue Purge program (DFHAQP).

ATP is executed as a unique CICS/VS task, with its own TCA, and may be resident and active only when one or more batches exist within CICS/VS.

### ASYNCHRONOUS TRANSACTION INPUT PROCESSOR (DFHRD1, DFHRD2)

The Asynchronous Transaction Input Processor (RDR) communicates directly with five functions of the CICS/VS System Management component: Storage Control, Program Control, Terminal Control, Task Control, and Transient Data Control. RDR communicates indirectly with the user-exit program as well as with the ATP modules AQP and ATP through the Terminal Input/Output Area (TIOA) and the Transient Data intrapartition queue (constructed in a Transient Data Output Area, or TDOA), respectively.

The notes below refer to Figure 3-2.

Notes:

1. The first phase of RDR (DFHRD1) is attached by Terminal Control (by means of a Task Control DFHKC TYPE=ATTACH macro instruction) in response to a CRDR command entered from a terminal.

2. DFHRD1 interprets the CRDR request and returns any messages to the requesting terminal.

3. DFHRD1 places a Batch Control Area (BCA) containing batch status information on a BCA chain. Storage Control services are requested by means of DFHSC TYPE=GETMAIN macro instruction to obtain a storage area for the BCA.

4. The second phase of RDR (DFHRD2) reads the batch data, which consists of transaction-initiating records, input data records, and one or more delimiters.

5. DFHRD2 interacts with any user exit routines (for more about these routines, see the CICS/VS System Programmer's Reference Manual.

6. DFHRD2 queues the input data onto a Transient Data intrapartition queue. Transient Data Control services are requested by means of DFHTD TYPE=PUT requests. Communication with Storage Control is required to obtain and release the TDOA.

7. When an entire batch has been read, the Asynchronous Transaction Control program (DFHATP) is either attached (by means of a Task Control DFHKC TYPE=ATTACH macro instruction) or marked "ready to run."

8. The TCAFCAAA field of the RDR TCA points to the terminal associated with this ATP task.

9. If a batch is to be deleted, the Asynchronous Queue Purge program (DFHAQP) is attached to perform the purging of data from the Transient Data intrapartition queue.

10. RDR communicates with Program Control by means of DFHPC TYPE=LOAD, TYPE=XCTL, and TYPE=RETURN macro instructions requesting Program Control services.

11. RDR communicates with Terminal Control by means of DFHTC macro instructions for WRITE, SAVE, WAIT, and GET operations on terminal data.

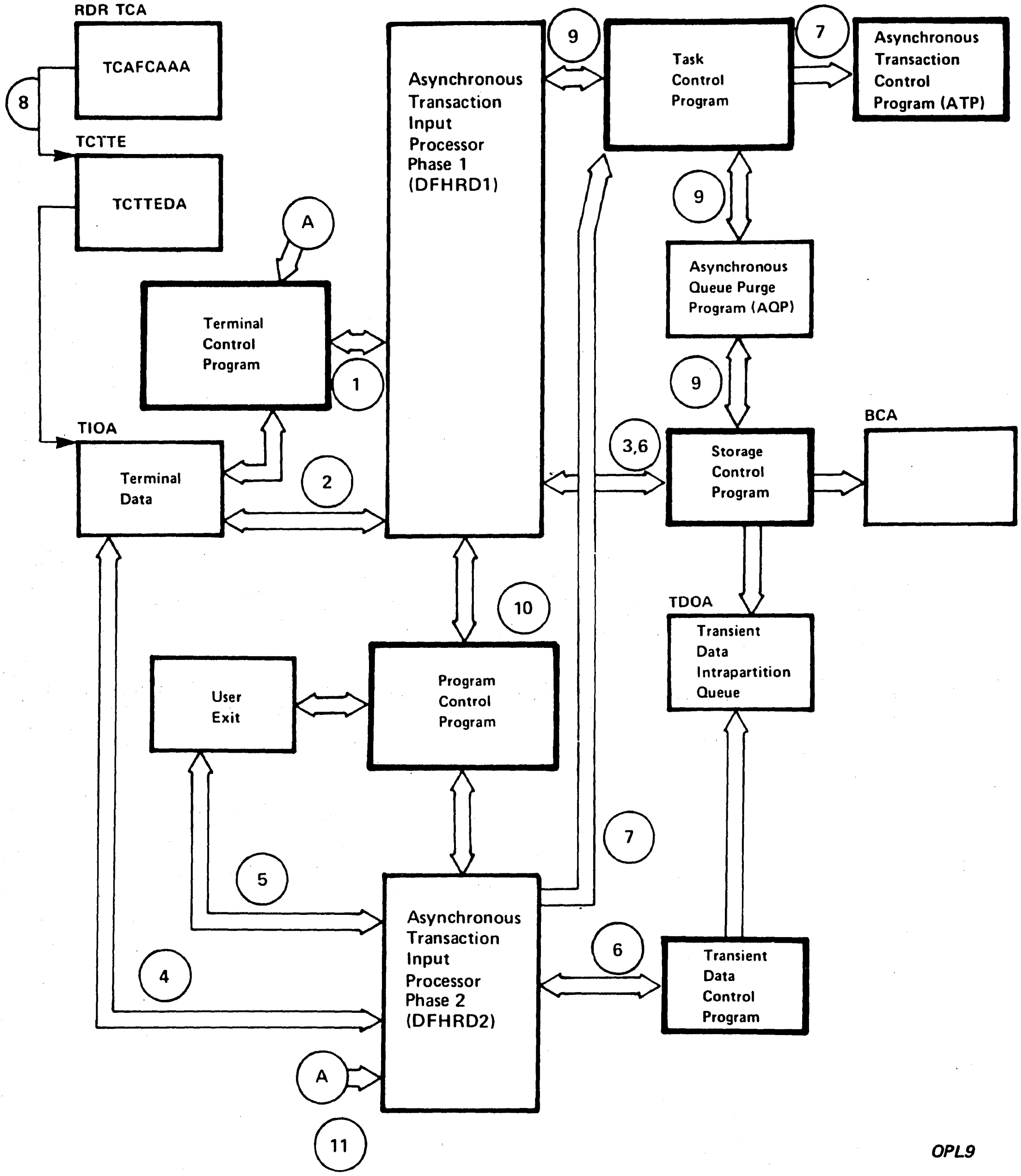

RDR TCA
TCAFCAAA
8
TCTTE
TCTTEDA
A
Terminal Control Program
1
TIOA
Terminal Data
2
Asynchronous Transaction Input Processor Phase 1 (DFHRD1)
9
Task Control Program
7
Asynchronous Transaction Control Program (ATP)
9
Asynchronous Queue Purge Program (AQP)
9
3,6
Storage Control Program
BCA
TDOA
Transient Data Intrapartition Queue
10
User Exit
Program Control Program
7
5
4
Asynchronous Transaction Input Processor Phase 2 (DFHRD2)
6
Transient Data Control Program
A
11
OPL9

Figure 3–2. Asynchronous Transaction Input Processor Interfaces

## ASYNCHRONOUS TRANSACTION OUTPUT PROCESSOR (DFHWT1, DFHWT2)

The Asynchronous Transaction Output Processor (WTR) communicates directly with five functions of the CICS/VS System Management component: Storage Control, Program Control, Terminal Control, Task Control, and Transient Data Control. WTR communicates indirectly with user-exit programs as well as with the ATP modules AQP and ATP through the Terminal Input/Output Area (TIOA) and the Transient Data intrapartition queue (constructed in a Transient Data Input Area, or TDIA), respectively.

The notes below refer to Figure 3-3.

Notes:

1. The first phase of WTR (DFHWT1) is attached by Terminal Control (by means of a Task Control DFHKC TYPE=ATTACH macro instruction) in response to a CWTR command entered from a terminal. Output is scheduled and actually occurs if CWTR is attached to a real terminal.

2. DFHWT1 interprets the CWTR request and returns any messages to the requesting terminal.

3. The BCA chain is searched to locate any batches requested by the CWTR command. A Write Request Element (WRE) is built for each terminal that is to receive output, with Storage Control services requested by means of DFHSC TYPE=GETMAIN to obtain a storage area for the WRE.

4. Preparatory to output, a Terminal Input/Output Area (TIOA) is acquired by means of a Storage Control request. DFHWT2 writes all records to be sent to a terminal.

5. DFHWT2 interacts with any user-exit routines (for more about these routines, see the CICS/VS System Programmer's Reference Manual.

6. DFHWT2 retrieves the data from a Transient Data intrapartition queue, utilizing Transient Data Control services.

7. At the end of a CWTR operation initiated by a WRE, the WRE is removed from the WRE chain and its storage released by means of a Storage Control DFHSC TYPE=FREEMAIN. After all WREs have been removed from a BCA, if SAVE was not specified, the BCA is removed from the BCA chain and its storage released in a similar manner. DFHWT2 interfaces with the Asynchronous Queue Purge program (AQP), which is attached by means of a Task Control DFHKC TYPE=ATTACH macro instruction to perform this processing.

8. If RELEASE was specified in the CWTR command, the Asynchronous Transaction Control program (ATP) is attached if it is not attached. The storage is released, and the task is terminated.

9. The TCAFCAAA field of the WTR TCA points to the terminal associated with this ATP task.

10. WTR communicates with Program Control by means of DFHPC TYPE=LOAD, LINK, XCTL, and RETURN macro instructions requesting Program Control services.

11. WTR communicates with Terminal Control by means of DFHTC TYPE=PUT requests to write out terminal messages.

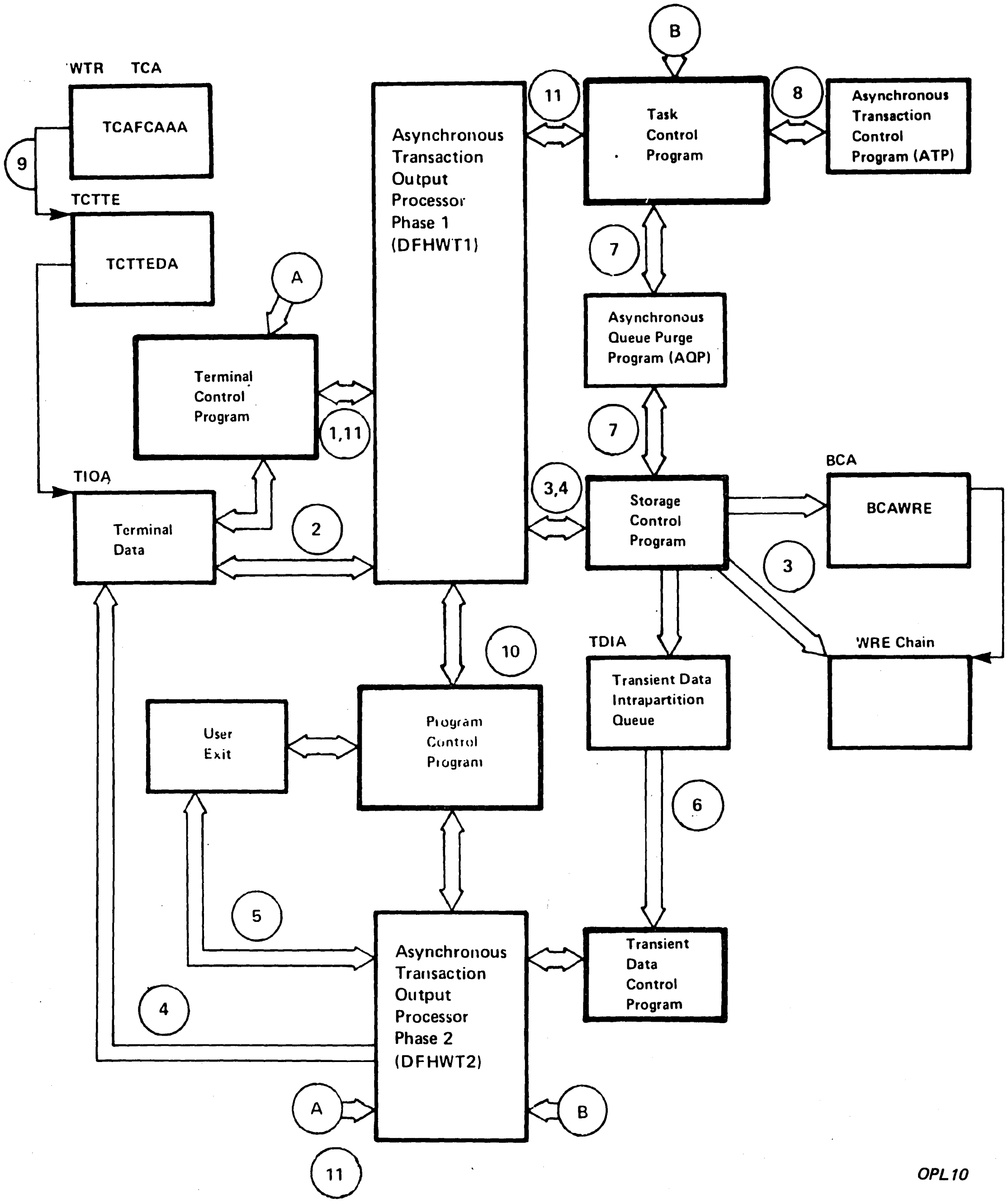

B
WTR TCA
TCAFCAAA
9
TCTTE
TCTTEDA
11
Asynchronous Transaction Output Processor Phase 1 (DFHWT1)
Task Control Program
8
Asynchronous Transaction Control Program (ATP)
7
Asynchronous Queue Purge Program (AQP)
A
Terminal Control Program
1,11
7
TIOA
Terminal Data
2
3,4
Storage Control Program
BCA
BCAWRE
3
10
TDIA
WRE Chain
Transient Data Intrapartition Queue
User Exit
Program Control Program
6
5
Asynchronous Transaction Output Processor Phase 2 (DFHWT2)
Transient Data Control Program
4
A
B
11
OPL10

Figure 3-3. Asynchronous Transaction Output Processor Interfaces

## ASYNCHRONOUS TRANSACTION CONTROL PROGRAM (DFHATP)

The Asynchronous Transaction Control program (DFHATP) communicates directly with five functions of the CICS/VS System Management component: Storage Control, Program Control, Terminal Control, Task Control, and Transient Data Control. ATP communication with the application and with the ATP modules RDR, AQP, and WTR is performed indirectly through the Terminal Input/Output Area (TIOA), Transient Data intrapartition queues, and the Batch Control Area (BCA) chain.

The notes below refer to Figure 3-4.

Notes:

1. ATP is attached by either a CRDR transaction (RDR) or the Terminal Control program (DFHTCP) in response to a terminal command (CATP or an application program read or write request).

2. Any DFHTC TYPE=READ or TYPE=WRITE request issued by an application program causes ATP to extract data from the Transient Data intrapartition input queue, or to place data onto the Transient Data intrapartition output queue, respectively. All rules that apply to the handling of Terminal Input/Output Areas (TIOAs) when a task is directly connected to a terminal also apply to tasks being run asynchronously. For this purpose, each task initiated by ATP is given the address of the dummy TCTTE, which is available in the BCA.

3. Whenever a BCA is to be deleted, the Asynchronous Queue Purge program (DFHAQP) performs the purging of data from the Transient Data queues used to process batches.

4. When all transactions in a batch have been processed, the output of the batch can be transmitted automatically to a terminal.

5. ATP communicates with Task Control by means of DFHKC requests for WAIT, ATTACH, RESUME, and SCHEDULE functions.

6. ATP communicates with Transient Data Control for PUT and GET requests of the data on Transient Data intrapartition queues.

7. ATP communicates with Program Control by means of DFHPC TYPE=RETURN macro instructions.

8. ATP communicates with Storage Control by means of DFHSC TYPE=GETMAIN and TYPE=FREEMAIN requests for input/output areas and BCA storage.

9. ATP communicates with Terminal Control by means of DFHTC TYPE=WRITE requests to write out terminal messages.

10. The initial application program for each valid transaction code is attached by DFHATP via Task Control.

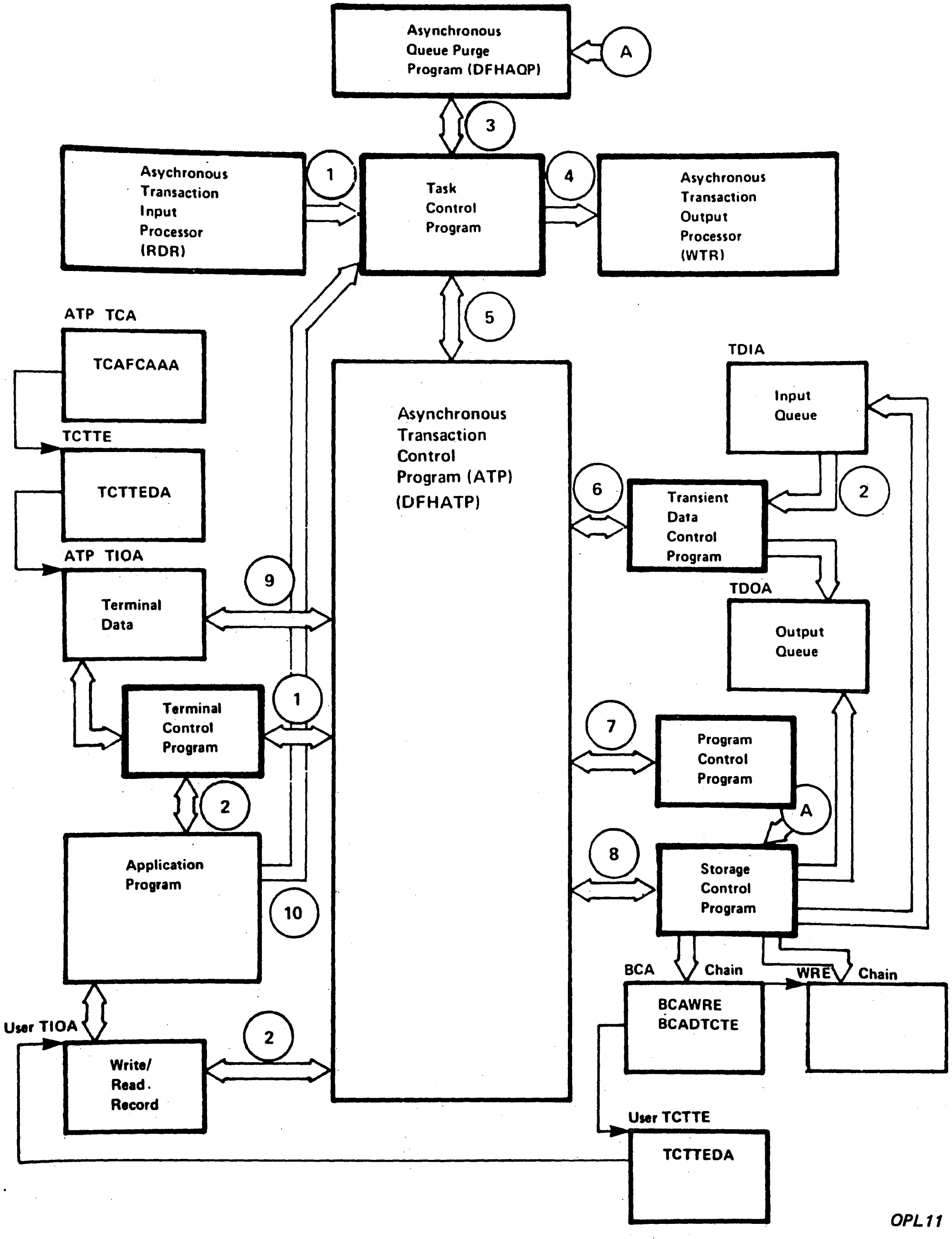


Figure 3-4. Asynchronous Transaction Control Interfaces

## BUILT-IN FUNCTIONS PROGRAM (DFHBFP)

| The Built-In Functions program (DFHBFP) may be generated with either or | both of two options: (1) the basic set, which includes Table Search, Phonetic Conversion, Field Edit, Field Verify, Bit Manipulation, and Input Formatting; and (2) Weighted Retrieval. If generated, any of these functions can be called by any application program. The Weighted Retrieval built-in function communicates with Storage Control (DFHSCP) and File Control (DFHFCP) during weighted retrieval operations as detailed in Figure 3-5 and the notes below.

Notes:

1. When DFHBFP services are utilized in an application program, the symbolic storage definition for the communication area of the Built-In Functions program (TCACCCA) must by copied into the common control communication area of the application program communication section of the program's TCA. This copying is achieved by means of a DFHBFTCA macro instruction, which must immediately follow the statement that copies the TCA and the user's definition of a TWA, if any, in the application program.

2. DFHBFP services are requested by means of DFHBIF macro instructions, which establish fields in the requesting program's TCA for communication with DFHBFP.

3. Storage Control services are used by the Input Formatting built-in function to obtain a new TIOA and by the Weighted Retrieval built-in function to obtain a work area for internal use.

4. File Control services are used by the Weighted Retrieval built-in function to perform browse operations.

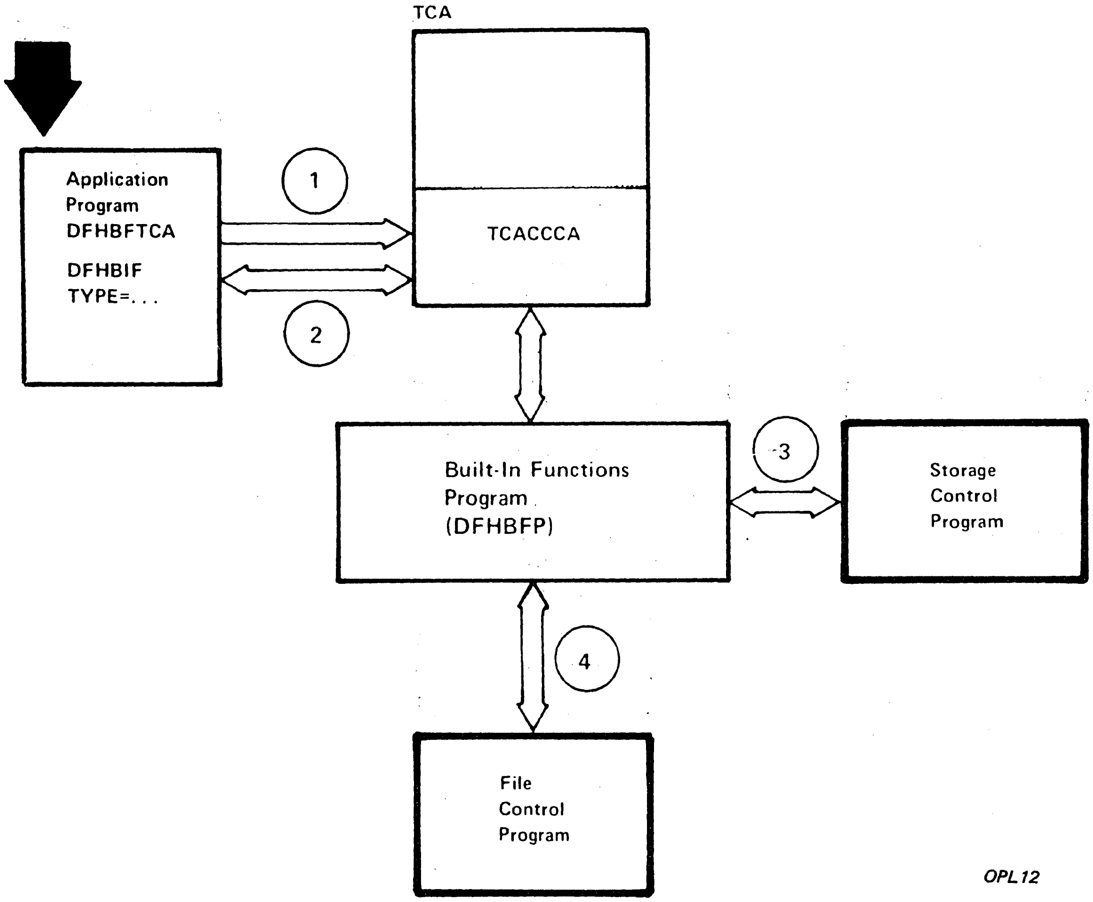


Figure 3–5. Built–In Functions Interface

## BASIC MAPPING SUPPORT (BMS)

Basic Mapping Support (BMS) is provided by means of a number of modules, each of which interfaces with other BMS modules, CICS/VS management components, and application programs. The maps that are handled by CICS/VS BMS may be new maps, created to utilize CICS/VS BMS mapping capabilities, or old maps, created for pre-VS BMS. The interrelationships of CICS/VS components providing mapping services to pre-VS programs and CICS/VS programs requesting mapping services are summarized in Figure 3-6. Additional details for specific programs within CICS/VS Basic Mapping Support are given on the pages that follow.

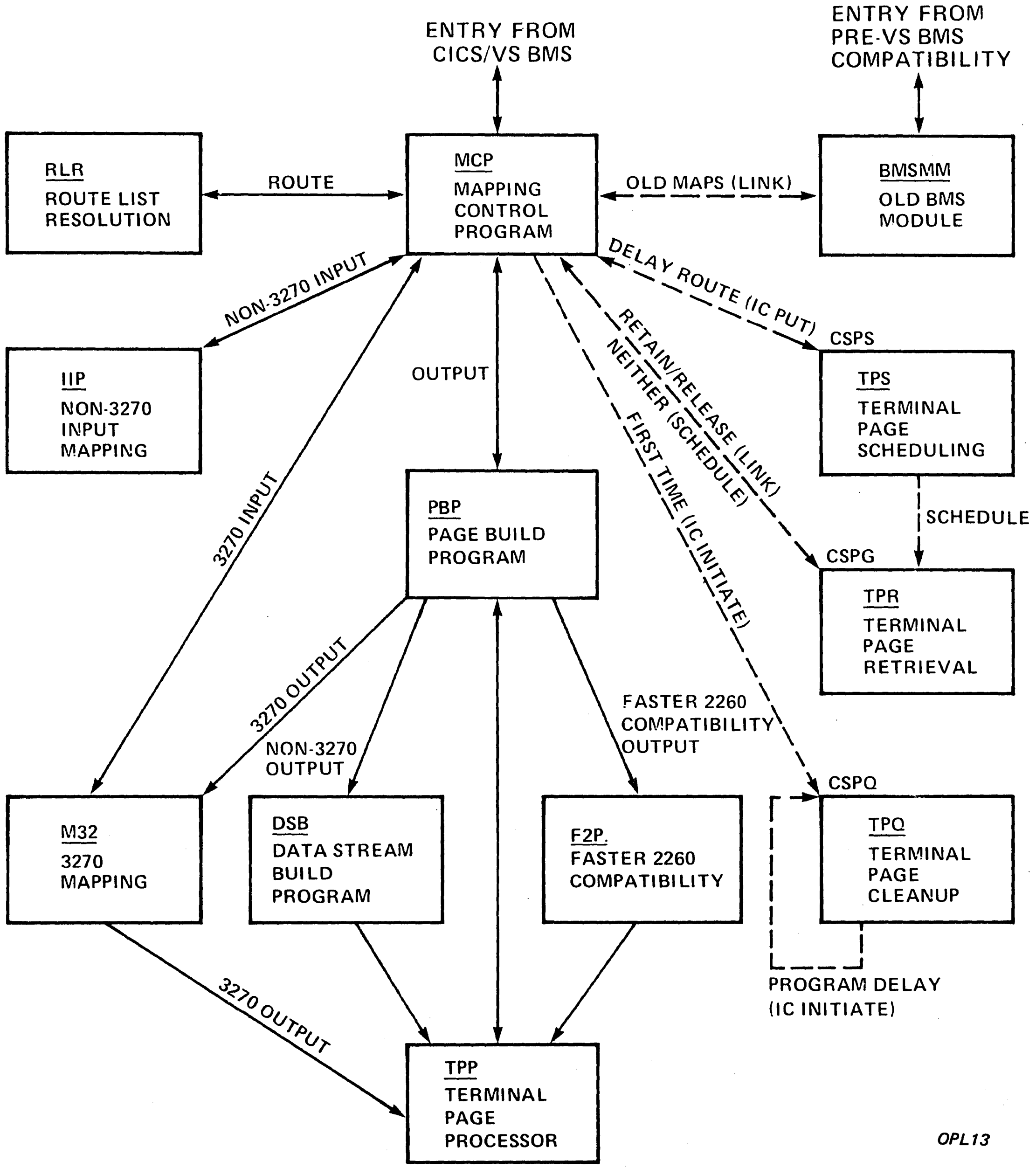


Figure 3-6. CICS/VS Basic Mapping Support

## PRE-VS BMS MAPPING MODULE (DFHBMSMM, BMS)

The pre-VS BMS Mapping Module (DFHBMSMM) is called in response to requests for BMS services coded in the form established for CICS (that is, prior to CICS/VS).

The notes below refer to Figure 3-7.

Notes:

1. A pre-CICS/VS DFHBMS TYPE=IN, MAP, or OUT macro request by an application program communicating with a 3270 terminal passes information via the TCA through the Program Control program (DFHPCP) to DFHBMSMM.

2. A CICS/VS DFHBMS TYPE=IN, MAP, or OUT macro instruction using pre-CICS/VS maps and DSECTs to communicate with a 3270 terminal passes information via the TCA through the Mapping Control program (DFHMCP) to DFHBMSMM.

3. Maps are either passed by the application program or loaded by DFHBMSMM.

4. The address of a Terminal Input/Output Area (TIOA) is supplied by the application program for TYPE=MAP or TYPE=OUT requests and by Terminal Control (DFHTCP) for TYPE=IN.

5. Terminal Control is used to read the data for a TYPE=IN request.

6. DFHBMSMM communicates with Storage Control (DFHSCP) to obtain and release work areas and buffers for mapping operations.

7. DFHBMSMM communicates with Program Control (DFHPCP) to load and delete maps required for mapping operations.

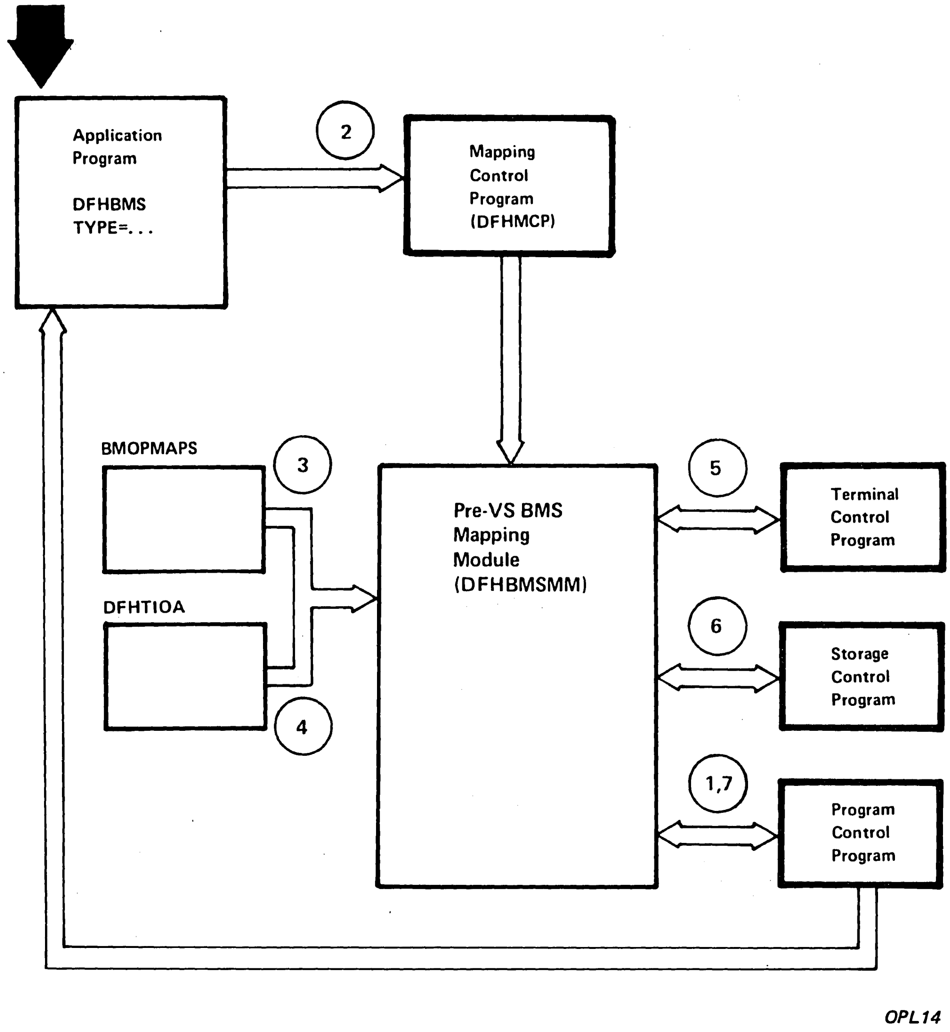


Figure 3-7. Pre-VS BMS Mapping Module Interfaces

## DATA STREAM BUILD (DFHDSB, BMS)

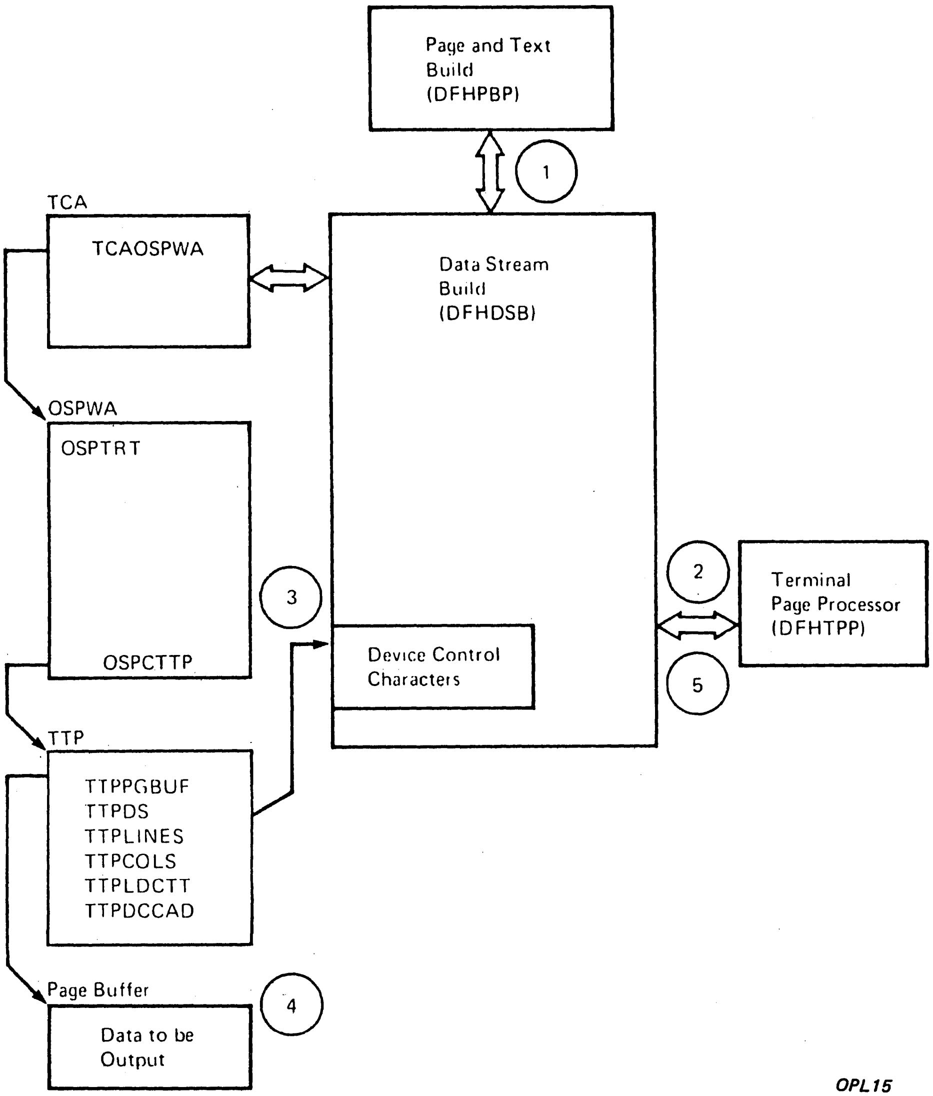
The Data Stream Build program addresses the page buffer, which was composed by the Page and Text Build program (DFHPBP). The page buffer contains lines of output data that are to be written to a terminal other than a 3270. The number of lines is contained in the TTPLINES field. The following functions are performed by the Data Stream Build program on the data in the page buffer:

- Truncates trailing blanks within data lines.
- Substitutes strings of physical device control characters for logical new-line characters that terminate each line of data.
- Provides a format management header (FMH) for some VTAM-supported devices.
- Allows horizontal and/or vertical tab processing.

The notes below refer to Figure 3-8.

Notes:

1. DFHDSB is entered from the Page Build program to process the page buffer.
2. For TYPE=NOEDIT, page buffer compression is skipped and control is given to the Terminal Page Processor (DFHTPP).
3. If not TYPE=NOEDIT, the appropriate device control characters for the target device are selected for substitution.
4. The page buffer containing the data to be compressed is located through the address stored at TTPPGBUF.
5. After compression of the page buffer data, DFHTPP is called to provide disposition of the page.



ıgure 3-8. Data Stream Build Interfaces

## NON-3270 INPUT MAPPING (DFHIIP, BMS)

The Non-3270 Input Mapping program (DFHIIP) is called in response to requests for BMS services involving terminals other than 3270 devices.

The notes below refer to Figure 3-9.

Notes:

1. A DFHBMS TYPE=IN or TYPE=MAP request by an application program communicating with other than a 3270 terminal passes information via the TCA through the Mapping Control program (DFHMCP) to DFHIIP.
2. The map required for an operation is either passed by the application program or loaded by DFHMCP.
3. The address of a Terminal Input/Output Area (TIOA) is supplied by the application program for TYPE=MAP and by Terminal Control for TYPE=IN.
4. Terminal Control is used to read the data for a TYPE=IN request except when the input is from a batch LU with INBFMH=DIP specified in the PCT, in which case DFHIIP interfaces with DFHDIP to read in the data.
5. DFHIIP communicates with Storage Control (DFHSCP) to obtain and release buffers for mapping operations.

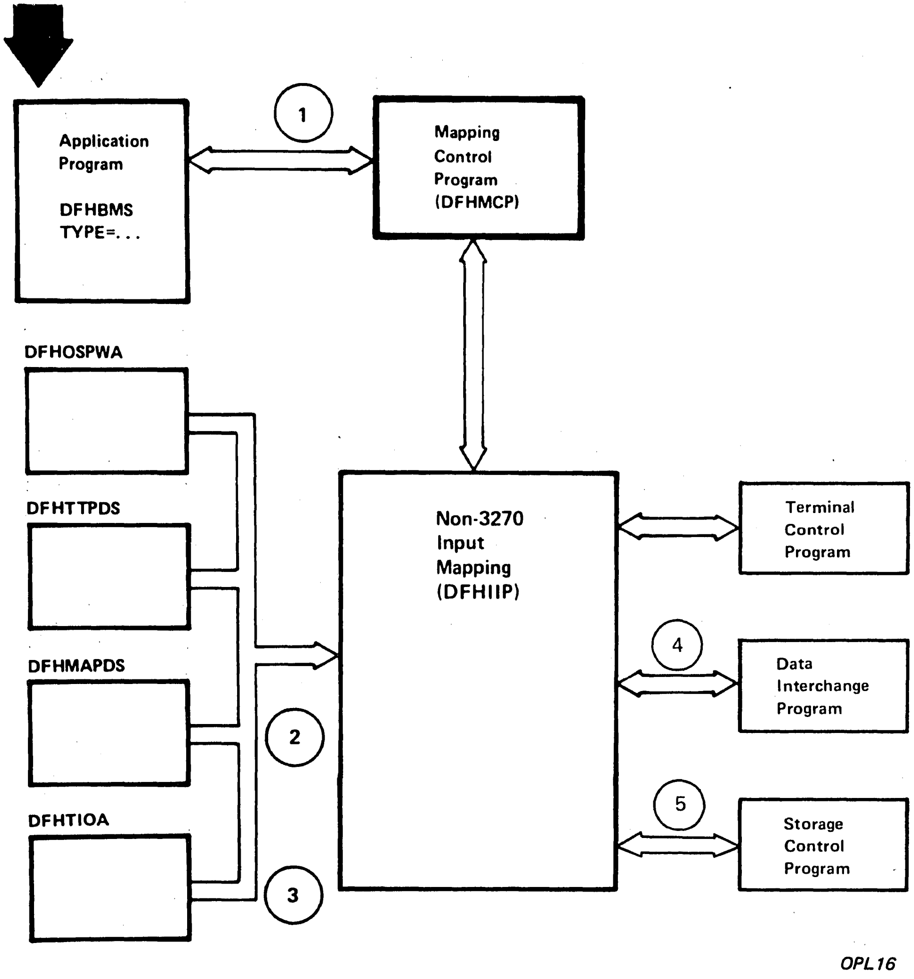


Figure 3-9. Non-3270 Input Mapping Interfaces

## MAPPING CONTROL PROGRAM (DFHMCP, BMS)

The Mapping Control program (DFHMCP) is the interface between application programs and the modules which perform mapping, message switching, page and text building, device-dependent output preparation, and message disposition to terminals, temporary storage areas, or the application program.

The notes below refer to Figure 3-10.

Notes:

1. This program is entered when an application program issues a DFHBMS request for Basic Mapping Support services.

2. It may also be called by Task Control to process a Deferred Work Element (DWE) if an application program terminates and there are partial pages in storage or the Message Control Record (MCR) created during execution of the task has not been placed in temporary storage.

3. The expansion of the DFHBMS macro instruction and the application program insert data in TCA fields labeled TCAMSxxx.

4. The following information is returned to the requestor in fields of the TCA: error codes, page overflow information, and (if TYPE=RETURN was specified in the request), a list of completed pages.

5. A Terminal Control DFHTC TYPE=SAVE macro instruction is issued if TYPE=SAVE was specified in the DFHBMS macro instruction.

6. DFHMCP communicates with Temporary Storage Control to put the MCR for routed or stored messages (TYPE=ROUTE and/or TYPE=STORE was specified). A DFHTS TYPE=PURGE macro instruction is issued to request that a message be purged from temporary storage if a DFHBMS TYPE=PURGE request is issued.

7. DFHMCP communicates with Storage Control to:

   a. Acquire and free storage in which the MCR is built (TYPE=PAGEOUT after TYPE=STORE and/or TYPE=ROUTE)

   b. Acquire and free storage in which to copy the message title (TYPE=ROUTE,TITLE=symbolic address or YES)

   c. Acquire storage to build Automatic Initiate Descriptors (AIDs) for non-routed messages or routed messages to be delivered immediately (TYPE=PAGEOUT)

   d. Acquire a BMS work area (OSPWA) at the time of the initial BMS request

   e. Acquire and free an area used for user request data if a TYPE=PAGEOUT must be simulated before processing the user's request

   f. Free the returned page list (TYPE=PURGE)

   g. Free map copies if TYPE=PAGEOUT and pages were being built in response to TYPE=PAGEBLD requests

   h. Free Terminal Type Parameters (TTPs) (TYPE=PAGEOUT)

8. DFHMCP communicates with Program Control to

   a. Load and delete map sets

   b. Link to the Page Retrieval program (DFHTPR) to process one or more pages of a message if TYPE=PAGEOUT and CTRL=RETAIN or CTRL=RELEASE

   c. Abnormally terminate a task if uncorrectable errors occur

   d. Link to the BMS Mapping Module (DFHBMSMM) if a map created prior to CICS/VS is loaded

9. DFHMCP communicates with Interval Control to

   a. Initiate transaction CSPQ (see HIPO 6.1.8)

   b. Obtain the current time of day, which is then used to time-stamp AIDs for routed messages

   c. Initiate transaction CSPS (see HIPO 6.1.10) for messages to be delivered at some future time

10. DFHMCP communicates with Task Control to schedule transaction CSPG (see HIPO 6.1.9) for every terminal that is to receive a routed message to be delivered immediately.

11. Transient Data Control is used to send error and informational messages to the master terminal.

12. Route List Resolution (DFHRLR) is used to collect terminals from a user-supplied route list or from the entire TCT by terminal type, and build a Terminal Type Parameter (TTP), which controls message building, for each terminal type. It is also used to build a one-element TTP for the originating terminal.

13. Non-3270 Input Mapping (DFHIIP) is used to process TYPE=IN or TYPE=MAP macro instructions for a terminal other than a 3270.

14. 3270 Mapping (DFHM32) is used to process TYPE=IN or TYPE=MAP macro instructions for a 3270 terminal.

15. Page and Text Build (DFHPBP) processes all output requests (TYPE=OUT, TYPE=STORE, or TYPE=RETURN).

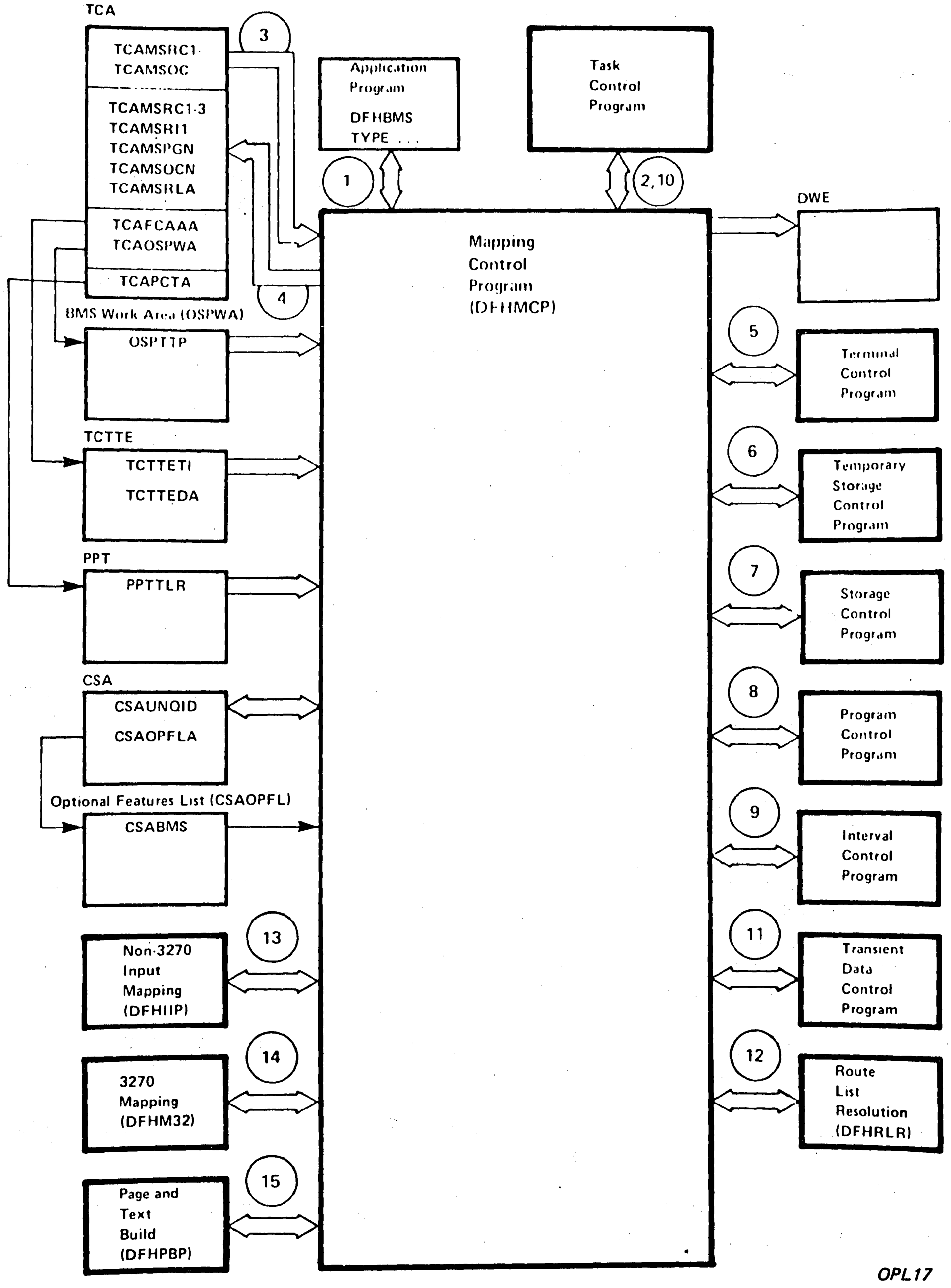


Figure 3-10. Mapping Control Program Interfaces

This page is left blank intentionally

## 3270 MAPPING (DFHM32, BMS)

The 3270 Mapping program (DFHM32) is called in response to requests for BMS services involving terminals of the 3270 Information Display System.

The notes below refer to Figure 3-11.

Notes:

1. A DFHBMS TYPE=PAGEBLD, TEXTBLD, OUT, STORE, or RETURN macro request by an application program communicating with a 3270 terminal passes information via the TCA through the Mapping Control program (DFHMCP) and the Page and Text Build program (DFHPBP) to DFHM32.
2. A DFHBMS TYPE=IN or TYPE=MAP macro request by an application program communicating with a 3270 terminal passes information via the TCA through the Message Control program (DFHMCP) to DFHM32.
3. Maps are either passed by the application program or loaded by DFHMCP.
4. The address of a Terminal Input/Output Area (TIOA) is supplied by Terminal Control (DFHTCP) for TYPE=IN requests and by the application program for all other requests.
5. Terminal Control is used to read the data for a TYPE=IN request.
6. DFHM32 communicates with Storage Control (DFHSCP) to obtain and release buffers for mapping operations.
7. All output requests (see note 1) are sent to a designated destination by the Terminal Page Processor (DFHTPP).

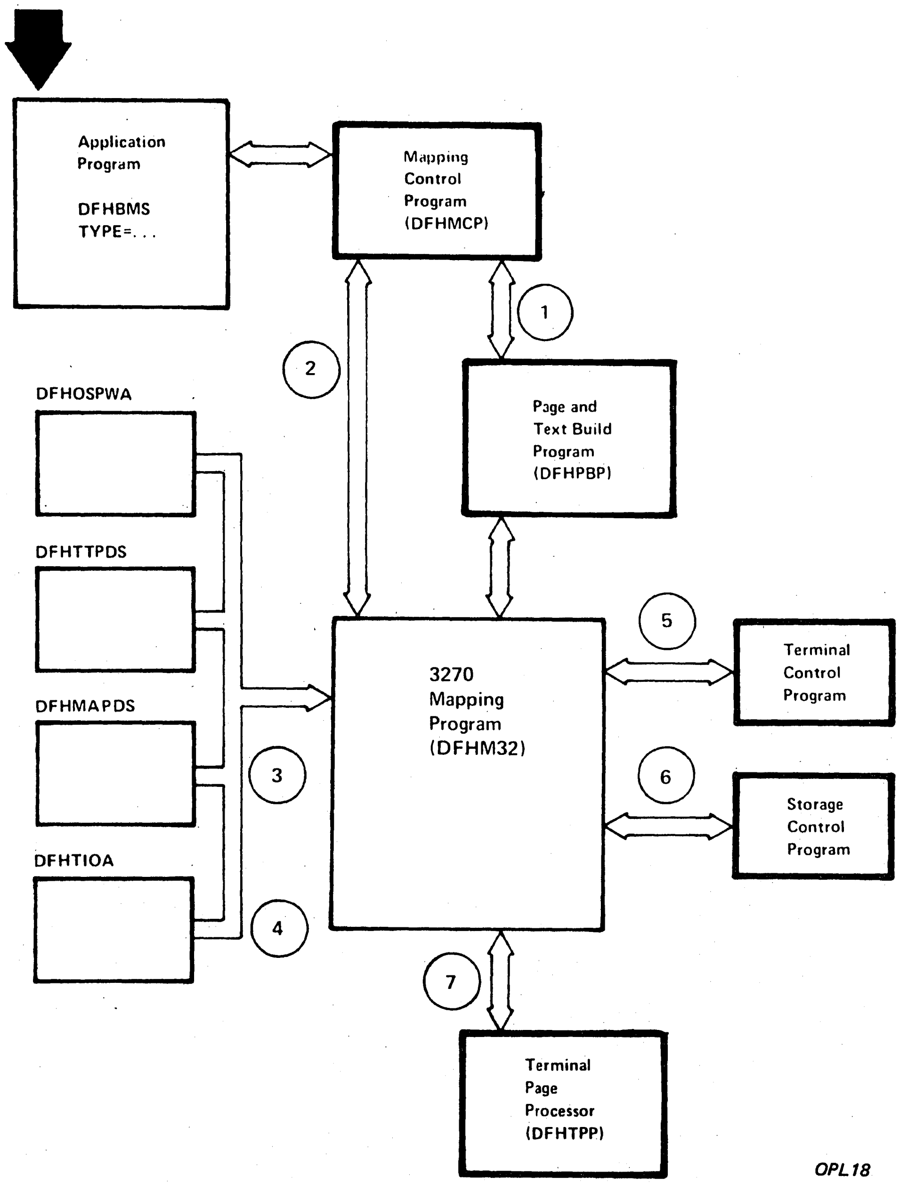


Figure 3-11. 3270 Mapping Program Interfaces

## PAGE AND TEXT BUILD (DFHPBP, BMS)

The Page and Text Build program (DFHPBP) processes all BMS output requests (DFHBMS TYPE=OUT, STORE, RETURN, or PAGEOUT). It performs the following functions:

- positions the data in the page, either by actually placing it in a buffer or by copying it and adjusting the map for a 3270 (TYPE=PAGEBLD)
- places the data into the page buffer (TYPE=TEXTBLD)
- inserts device dependent control characters for other than 3270 Information Display System devices

The notes below refer to Figure 3-12.

Notes

1. DFHPBP is entered from the Mapping Control program (DFHMCP) to process all BMS output requests. It is called once for each Terminal Type Parameter (TTP) on the TTP chain pointed to by OSPTTP. The current TTP in the chain is pointed to by OSPCTTP.

2. DFHPBP returns control to DFHMCP when request processing is complete, or when the page must be written out before a TYPE=PAGEBLD request can be processed and an OFLOW=symbolic address operand was specified.

3. OSPTR2, OSPTR3,...,OSPTR7 contain request data from the DFHBMS macro expansion. OSPRC1 and OSPRC3 contain return codes to be examined by DFHMCP.

4. For a TYPE=PAGEBLD request for a 3270, the map is copied and chained to the TTP. For a TYPE=TEXTBLD request for a 3270, a dummy map is created and chained to the TTP. When a page is complete, control is given to 3270 Mapping (DFHM32), which combines the map copies chained to the TTP and maps the data.

5. DFHPBP communicates with Storage Control to

   a. Acquire and free buffers in which pages are built

   b. Acquire storage for copies of maps for TYPE=TEXTBLD or TYPE=PAGEBLD

   c. Acquire storage for a copy of the user's data for TYPE=TEXTBLDor TYPE=PAGEBLD

6. DFHPBP requests Program Control to abnormally terminate a transaction (DFHPC TYPE=ABEND) if certain uncorrectable errors occur.

7. TYPE=TEXTBLD request for a 3270 causes a map set consisting of one dummy map to be passed to 3270 Mapping (DFHM32). The map has one field with attributes FREEKB and FRESET.

8. If the page is being constructed for a 3270, control is given to DFHM32 to map the data and then to DFHTPP to output the page. Otherwise, device dependencies are inserted in the page and control is given to the Terminal Page Processor (DFHTPP) to output the page.

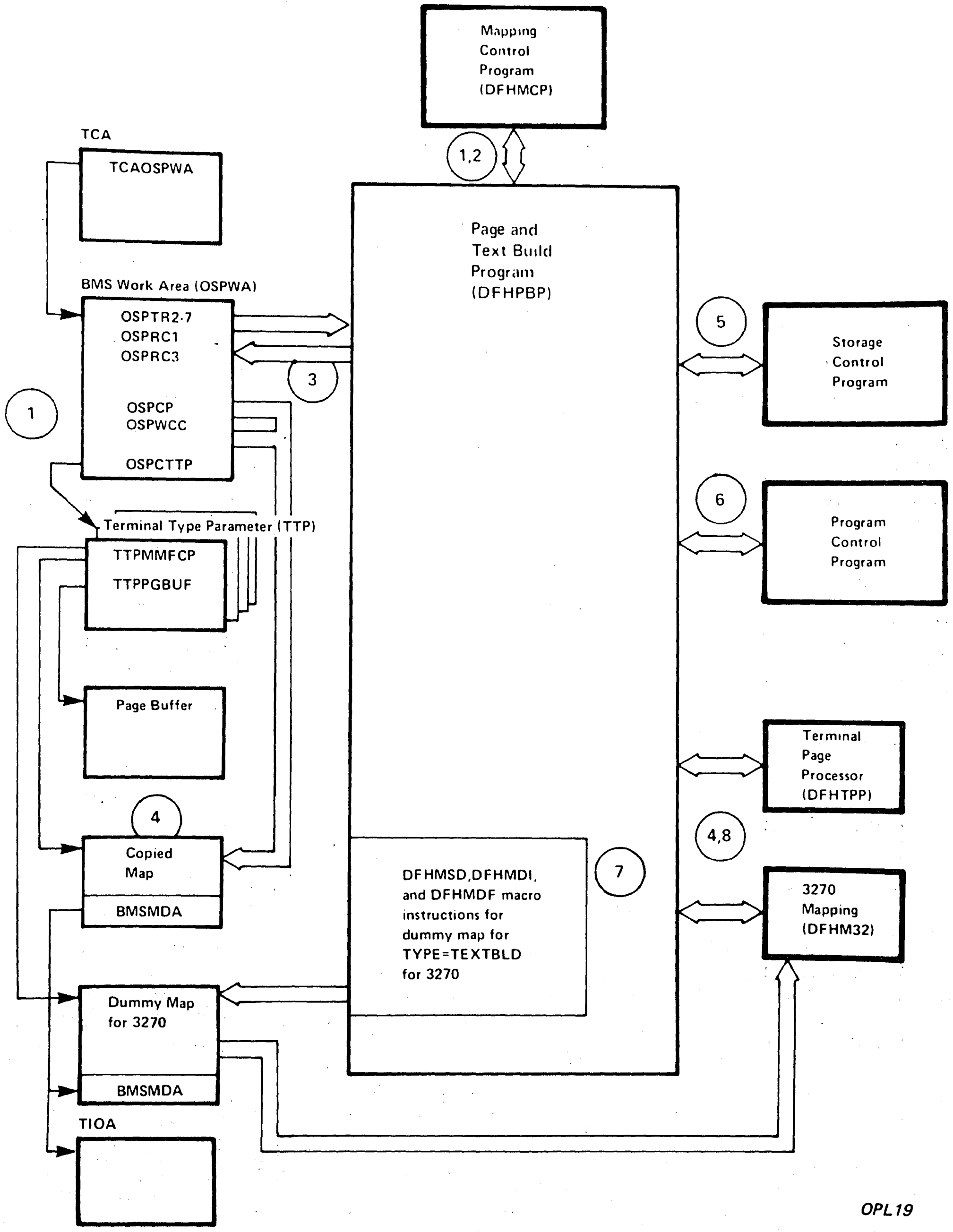


Figure 3–12. Page and Text Build Program Interfaces

## ROUTE LIST RESOLUTION PROGRAM (DFHRLR, BMS)

The Route List Resolution program (DFHRLR) builds Terminal Type Parameters (TTPs), which are the main blocks for building and outputting data in BMS.

The notes below refer to Figure 3-13.

Notes:

1. DFHRLR is called by the Mapping Control program (DFHMCP) to determine the grouping of terminal destinations.

2. If data is to be routed, DFHRLR groups the terminals in the user's route list by terminal type and builds a routing TTP for each type. The address of the first routing TTP in the chain of TTPs is placed in OSPTTP.

3. If data is not to be routed, a direct TTP is built for the originating terminal and its address is placed in OSPDTTP.

4. DFHRLR communicates with Storage Control to acquire storage for the TTP.

5. Program Control services are requested by means of a DFHPC TYPE=ABEND macro instruction if certain uncorrectable errors occur.

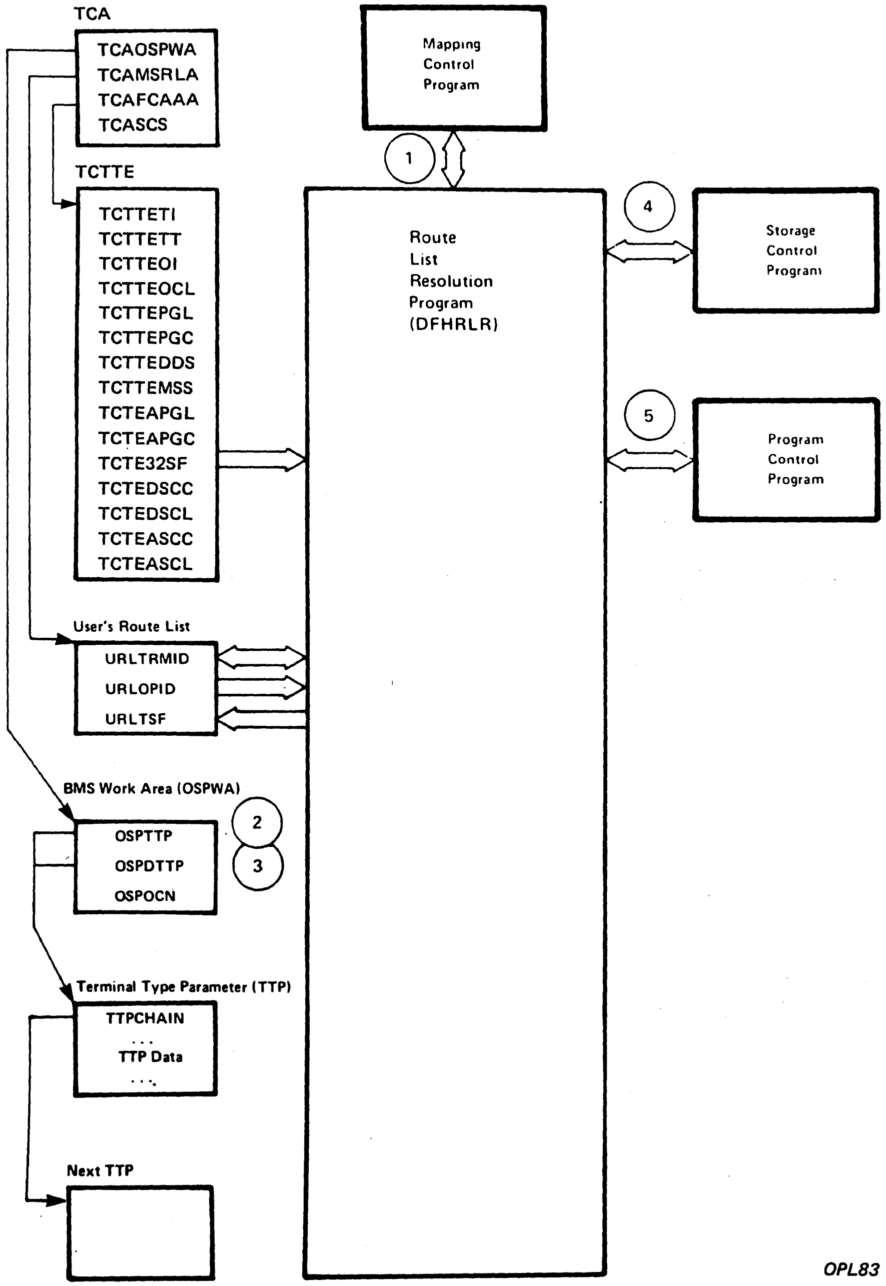


Figure 3-13. Route List Resolution Program Interfaces

## TERMINAL PAGE PROCESSOR (DFHTPP, BMS)

The Terminal Page Processor (DFHTPP) puts completed pages to a destination specified in the BMS output request (TYPE=OUT sends to the originating terminal; TYPE=STORE directs to temporary storage; and TYPE=RETURN directs to a list of completed pages that are returned to the application program).

The notes below refer to Figure 3-14.

Notes:

1. DFHTPP is entered from 3270 Mapping (DFHM32) for 3270s, from FASTER 2260 Compatibility (DFHF2P) for FASTER 2260 compatibility output, and from Data Stream Build (DFHDSB) for other devices.

2. DFHTPP communicates with Storage Control to obtain.

   a. The return list (to store the address of completed pages to be returned to the programmer).

   b. Deferred Work Elements (DWEs), which ensure that message control information is written to disk even if the programmer neglects to issue a DFHBMS TYPE=PAGEOUT request.

   c. Storage for a list that correlates pages on temporary storage with the logical device codes for which they are destined.

3. Temporary Storage Control is used to store pages and the Message Control Record (MCR) for messages stored on Temporary Storage.

4. The Terminal Type Parameter (TTP) controls the formatting of a message for a particular terminal type, for example, 2741. TTPPGBUF contains the address of a completed page.

5. The Terminal Output macro instruction (DFHTOM) is issued to provide an open subroutine assembled within DFHTPP that puts a completed page out to the terminal.

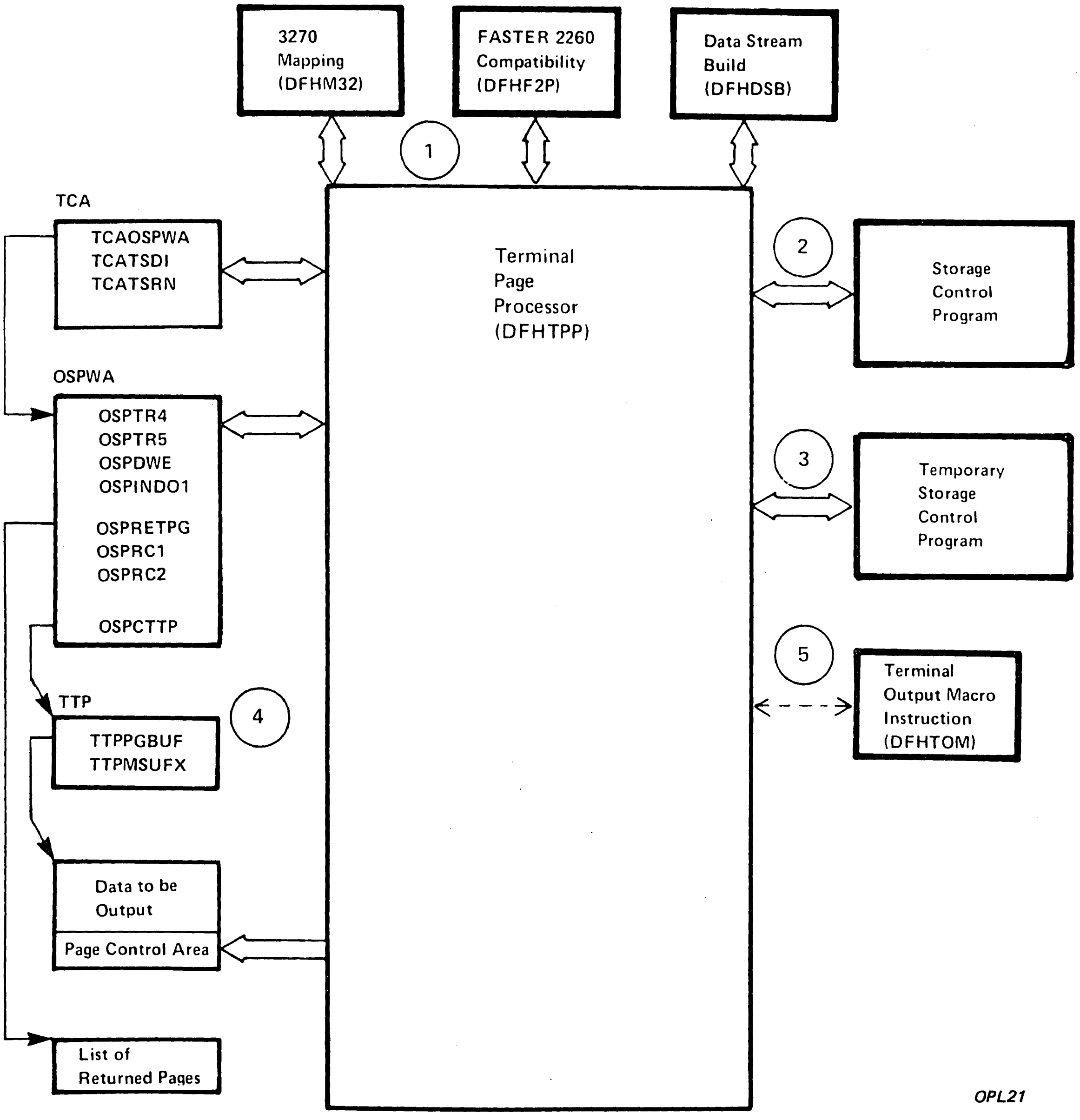


Figure 3—14. Terminal Page Processor Interfaces

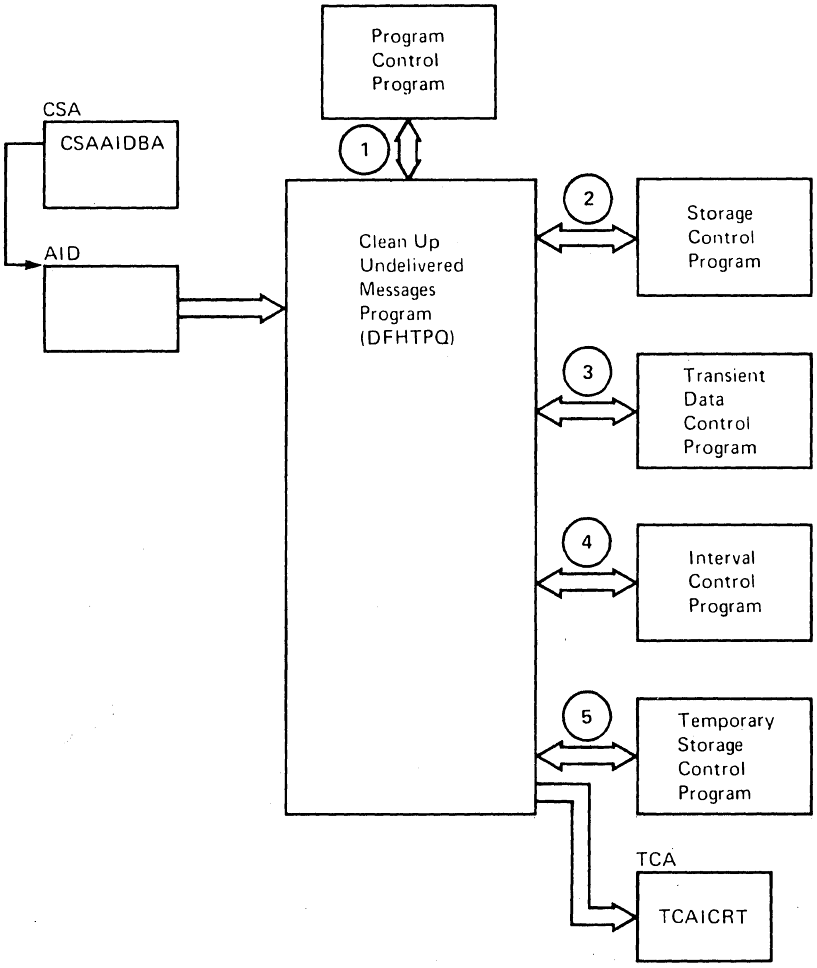
## CLEANUP UNDELIVERED MESSAGES PROGRAM (DFHTPQ, BMS)

The Cleanup Undelivered Messages program (DFHTPQ) checks the chain of Automatic Initiate Descriptors (AIDs) to detect and delete AIDs that have been on the chain for longer than the purge-delay time specified at system generation (DFHSG PROGRAM=BMS,PRGDLAY=hhmm).

The notes below refer to Figure 3-15.

Notes:

1. DFHTPQ is initiated the first time by the Mapping Control program (DFHMCP) via Interval Control. Thereafter, it reinitiates itself (see note 4).
2. DFHTPQ communicates with Storage Control to free AIDs which have been purged and to acquire storage for notification messages.
3. Transient Data Control is used to send notification messages.
4. Interval Control is used to obtain the current time and to reinitiate this task (DFHTPQ).
5. DFHTPQ communicates with Temporary Storage Control to retrieve and replace Message Control Records (MCRs) and to purge messages.


Program Control Program
CSA
CSAAIDBA
1
AID
Clean Up Undelivered Messages Program (DFHTPQ)
2
Storage Control Program
3
Transient Data Control Program
4
Interval Control Program
5
Temporary Storage Control Program
TCA
TCAICRT

OPL22

Figure 3-15. Cleanup Undelivered Messages Program Interfaces

## PAGE RETRIEVAL PROGRAM (DFHTPR, BMS)

The Page Retrieval program (DFHTPR) processes messages built by BMS and placed in temporary storage.

The notes below refer to Figure 3-16.

Notes:

1. DFHTPR can be initiated as a standalone transaction (CSPG or user defined paging command, for example, P/, or 3270 PA/PF keys), or linked to from BMS conversational operation. (DFHBMS TYPE=PAGEOUT, CTRL=RELEASE|RETAIN). The functions of DFHTPR are:

   a. Display the first page of a routed message.

   b. Display subsequent pages of a message at a terminal for which TYPE=PAGEOUT,CTRL=AUTOPAGE was specified.

   c. Process paging commands from a terminal.

   d. Process transaction CSPG (see HIPO 6.1.9) when it is entered at the terminal.

   e. Purge a message displayed at the terminal if the terminal is in display status and other than a paging command is entered at the terminal.

2. DFHTPR is entered from the BMS Mapping Control program (DFHMCP) to display the first page of a message originated at the terminal if CTRL=RETAIN was specified in the BMS request. DFHTPR reads from the terminal and processes paging commands until other than a paging command is entered.

3. DFHTPR communicates with Storage Control to

   a. Acquire and free Message Control Blocks (MCBs).

   b. Free Message Control Record (MCR) storage

   c. Acquire storage for informational and error messages to be sent to the destination terminal and the master terminal

   d. Free an Automatic Initiate Descriptor (AID) taken off the AID chain

   e. Acquire and free storage for a route list constructed in response to a COPY command entered at a terminal

   f. Acquire a TIOA into which to place a device-independent page when performing the COPY function

4. Temporary Storage Control is used to retrieve and replace MCRs and to retrieve and purge pages.

5. Basic Mapping Support is used to display error and informational messages at a requesting terminal and to send a page to the destination terminal in the COPY function.

6. Task Control is used to retain exclusive control of a MCR while it is being updated.

7. DFHTPR communicates with Interval Control during error processing when a Temporary Storage identification error is returned while attempting to retrieve a MCR. Up to four retries (each consisting of a one-second wait followed by another attempt to read the MCR) are performed. (The error may be due to the fact that an MCR has been temporarily released because another task is updating it. If so, the situation may correct itself, in which case a retry is successful.)

8. Terminal Control is used to read in the next portion of terminal input after a page or informational message is sent to the terminal when TYPE=PAGEOUT,CTRL=RETAIN was specified.

9. Transient Data Control is used to send error or informational messages to the master terminal.

10. The Terminal Output macro instruction (DFHTOM) is issued to provide an open subroutine that puts a completed page out to the terminal.

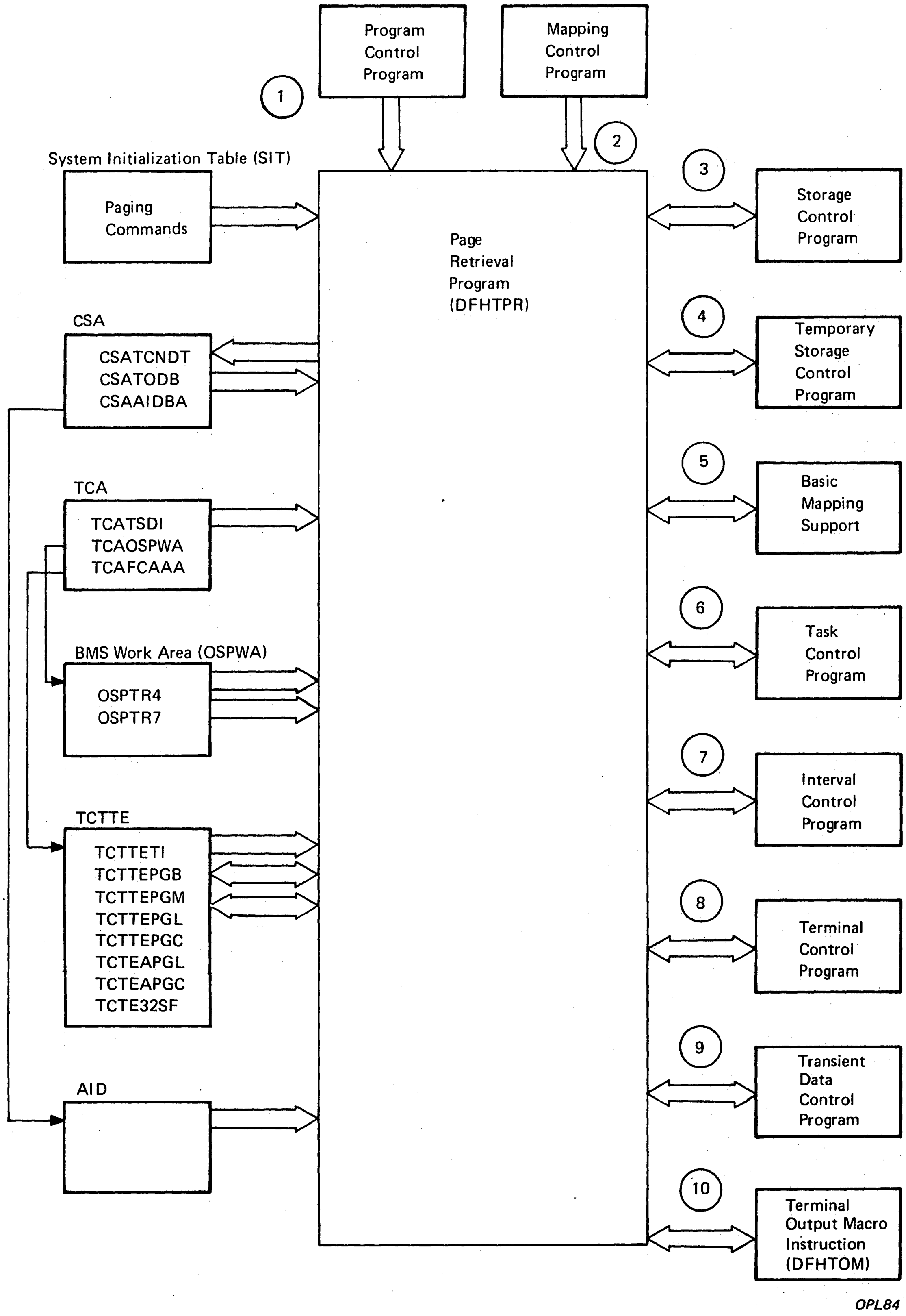


Figure 3-16. Page Retrieval Program Interfaces

This page is left blank intentionally

## TERMINAL PAGE SCHEDULING PROGRAM (DFHTPS, BMS)

The Terminal Page Scheduling program (DFHTPS) processes messages that have been scheduled for delayed delivery. An Automatic Initiate Descriptor (AID) is built and scheduled for every terminal specified in the Message Control Record (MCR) that Interval Control provides.

The notes below refer to Figure 3-17.

Notes:

1. DFHTPS is called by Program Control when a delayed message is to be sent.

2. DFHTPS communicates with Interval Control to obtain the Message Control Record (MCR) that has the IDs of all terminals that are to receive the message.

3. DFHTPS communicates with Temporary Storage Control to replace the MCR, since Interval Control released the MCR while retrieving it.

4. Interval Control has created an Interval Control Element (ICE) for the time-dependent request that is now to be serviced. DFHTPS refers to many fields of the MCR (supplied via Interval Control) when building the AIDs required for the message.

5. DFHTPS communicates with Storage Control to acquire storage for the AIDs that it constructs.

6. After the AIDs have been constructed, DFHTPS communicates with Task Control to schedule the AIDs for processing.

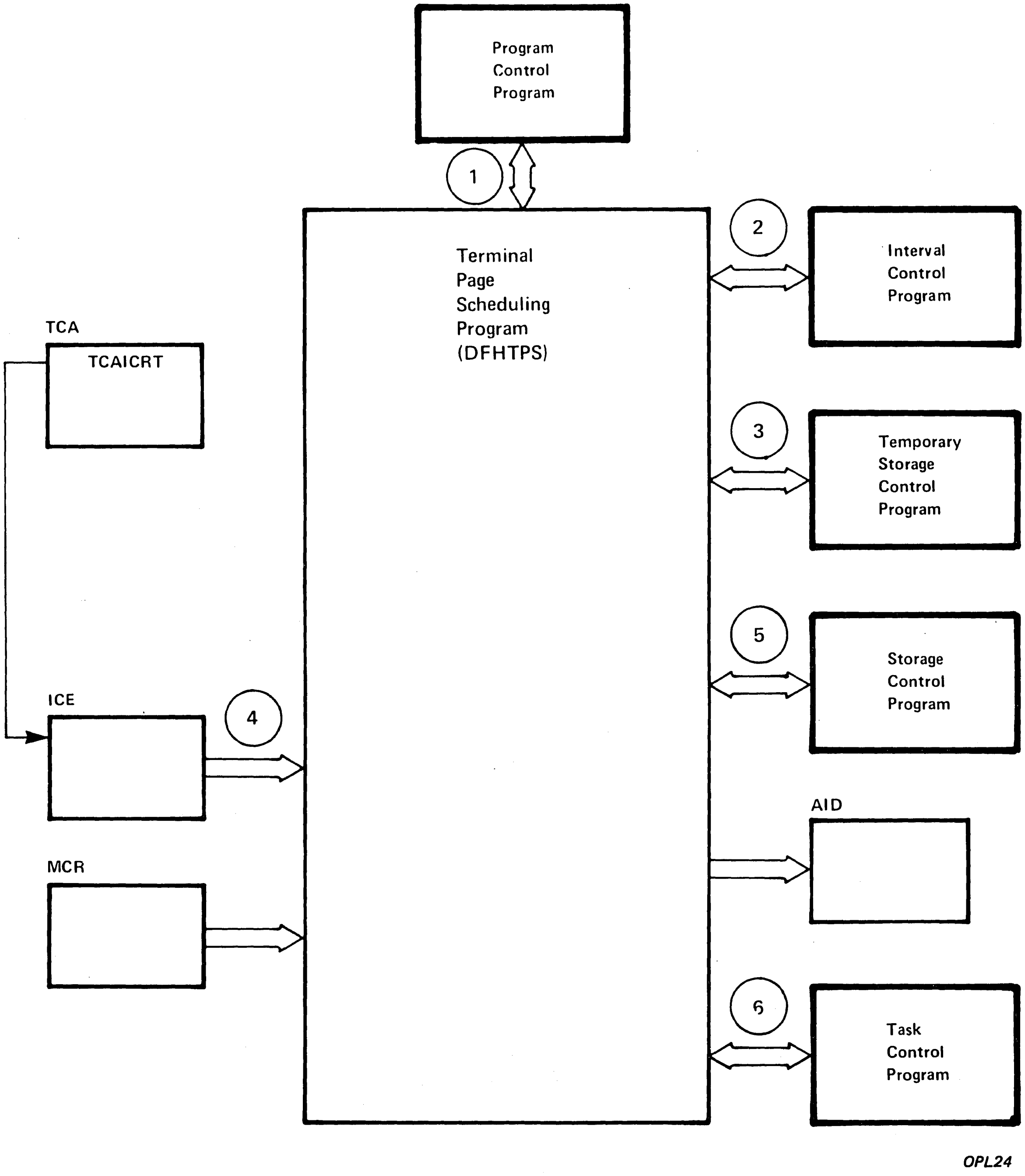


Figure 3-17. Terminal Page Scheduling Program Interfaces

## DYNAMIC ALLOCATION PROGRAM (MVS ONLY)

The program runs as a CICS/VS transaction, using CICS/VS function at the command level wherever possible. It does not modify any CICS/VS control blocks. Only the DYNALLOC function is available through the program; any manipulation of the environment before or after the DYNALLOC request must be done by other means.

The flow in a normal invocation is as follows. The main program, DFH99M receives control from CICS/VS, and carries out initialization. This includes determining the screen size and allocating input and output buffer sections, and issuing initial messages. It then invokes DFH99GI to get the input command from the terminal. Upon return, if the command was null, the main program terminates, issuing a final message.

The command obtained has its start and end addresses stored in the global communications area, COMM. The main program allocates storage for tokenised text, and calls DFH99TK to tokenise the command. If errors were detected during this stage, further analysis of the command is bypassed.

Following successful tokenising, the main program calls DFH99FP to analyse the verb keyword. DFH99FP calls DFH99LK to look up the verb keyword in the table, DFH99T. DFH99LK calls DFH99MT if an abbreviation is possible. Upon finding the matching verb, DFH99FP puts the address of the operand section of the table into COMM, and puts the function code into the dynalloc request block.

The main program now calls DFH99KO to process the operand keywords. Each keyword in turn is looked up in the table by calling DFH99LK, and the value coded for the keyword is checked against the attributes in the table. DFH99KO then starts off a text unit with the appropriate code, and, depending on the attributes the value should have, calls a convert routine.

For character and numeric string, DFH99CC is called. It validates the string, and puts its length and value into the text unit.

For binary, DFH99BC is called. It validates the value, converts it to binary of the required length, and puts its length and value into the text unit.

For keyword values, DFH99KC is called. It looks up the value in the description part of the keyword table using DFH99LK, and puts the coded equivalent value and its length into the text unit.

When a keyword specifying a returned value is encountered, DHF99KO makes an entry on the returned value chain, which is anchored in COMM. This addresses the keyword entry in DFH99T, the text unit where the value will be returned, and the next entry. In this case the convert routine is still called, but it only reserves storage in the text unit, setting the length to the maximum and the value to zeros.

When all the operand keywords have been processed, DFH99KO returns to the main program, which calls DFH99DY to issue the dynalloc request.

DFH99DY sets up the remaining parts of the parameter list, and if no errors too severe have been previously detected, a subtask is ATTACHed to issue the DYNALLOC SVC. A WAIT EVENT is then issued against the subtask termination ECB. When the subtask ends, and CICS dispatches the program again, the DYNALLOC return code is captured from the subtask

ECB, and the error and reason codes from the DYNALLOC request block, and a message is issued to give these values to the terminal.

DFH99DY then returns to the main program, which calls DFH99RP to process returned values. DFH99RP scans the returned value chain, and for each element issues a message containing the keyword and the value found in the text unit. If a returned value corresponds to a keyword value, DFH99KR is called to look up the value in the description, and issue the message.

Processing of the command is now complete, and the main program reinitializes for the next one, and loops back to the point where it calls DFH99GI.

Messages are issued at many places, using macros. The macro expansion ends with a call to DFH99MP, which ensures that a new line is started for each new message, and calls DFH99ML, the message editor. Input to the message editor is a list of tokens, and each one is picked up in turn and converted to displayable text. For each piece of text, DFH99TX is called, which inserts the text into the output buffer, starting a new line if necessary. This arranges that a word is never split over two lines.

At the end of processing the command, the main program calls DFH99MP with no parameters, which causes it to send the output buffer to the terminal, and initialize it to empty.

## DYNAMIC BACKOUT PROGRAM (DFHDBP)

The Dynamic Backout Program (DFHDBP) is invoked by the Program Control Program during ABEND processing for transactions for which dynamic transaction backout has been specified.

DFHDBP backs out the changes made by the transaction to recoverable resources. It does this by first scanning the DWE chain to identify changes to Transient Data, Temporary storage, and AIDs. DFHDBP then scans the dynamic log to pick up file control and DL/I 'before' images, and 'durable' data, so that the files, data bases, and initial images of TCTUA and COMMAREA may be restored.

For a restartable transaction, either the user-written program DFHRTY is invoked or the default rules are applied, to determine if restart is required. If restart is required, DFHPCP is invoked.

The notes below refer to Figure 3-18.

Notes:

1. DFHPCP invokes DFHDBP using XCTL.
2. DFHACP is linked to issue message DFH2205 to the user's terminal.
3. The DWE list is scanned. DFHTDP is invoked for Transient Data DWEs, and DFHTSP is invoked for Temporary Storage DWEs, to backout changes to the destinations. DFHFCP is invoked for File Control DWEs to perform any outstanding RELEASE.
4. The Dynamic Log is scanned and DFHFC macro instructions issued to reverse file control changes. The DL/I backout module is invoked to backout DL/I changes.
5. 'Spilled' Dynamic Log entries are recovered from temporary storage.
6. Transfer control to DFHACP.
7. DFHPC TYPE=RETRY is invoked to restart the transaction.
8. The retry program (DFHRTY) is invoked to decide whether restart is appropriate.

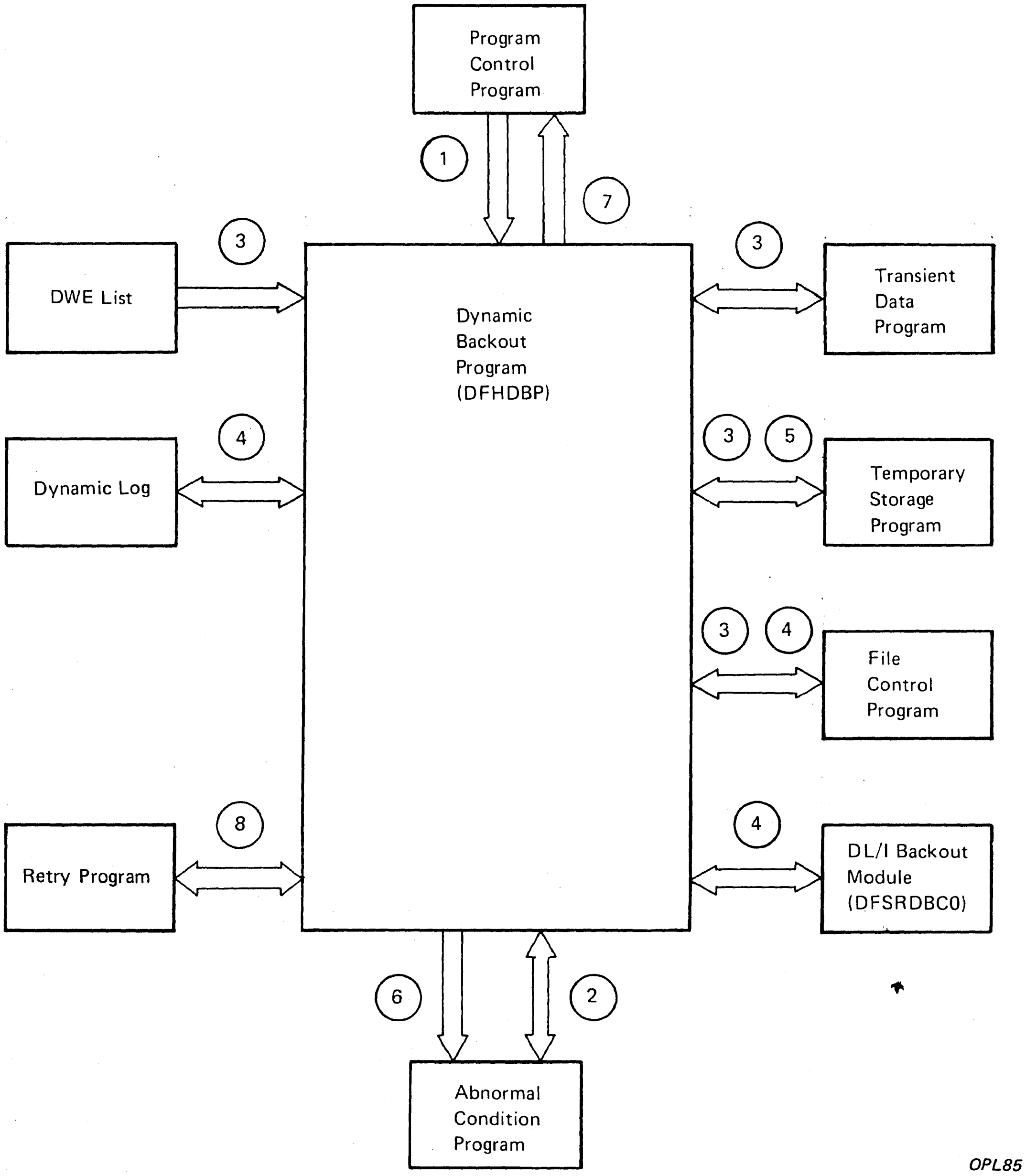


Figure 3-18. Dynamic Backout Program

## DUMP MANAGEMENT (DFHDCP)

The Dump Control program (DFHDCP) dumps specified areas of storage to tape or disk (to be printed later by the Dump Utility program). It functions in response to a requesting task, performing the service requested and returning control to the requesting task upon completion of the dump operation.

The notes below refer to Figure 3-19.

Notes:

1. DFHDCP may be called by either an application program or a CICS/VS management program to write out the contents of main storage at any time. The DFHDC macro instruction sets the type of request switch (TCADCTR), number of bytes to be dumped (TCADCNB), storage address (TCADCSA), and dump identifier (TCADCDC) in the requesting program's TCA.

2. The Common System Area (CSA) and Task Control Area (TCA) are always the first areas to be dumped. Additional areas that may be dumped include: transaction storage, Trace Table, terminal storage, program and register save areas, and system control tables.

3. DFHDCP communicates with Interval Control, Trace Control, and Program Control to set runaway task control, request and release a lock of the dump facility, request a trace, and reset and test runaway task control.

4. Dump Control refers to a Data Event Control Block (DECB) to determine address (DCECBIOA) and length (DCECBIOL). DCDTFI identifies a dummy ECB, referenced in the lock and release of the dump facility.

5. Output is written to a sequential data set on tape or disk. Records have a variable blocked format. A single record may be a CICS/VS system control table, one TCA, or an entire program. Each logical record contains a one byte identifier to indicate the type of control block, table, or program that it contains.

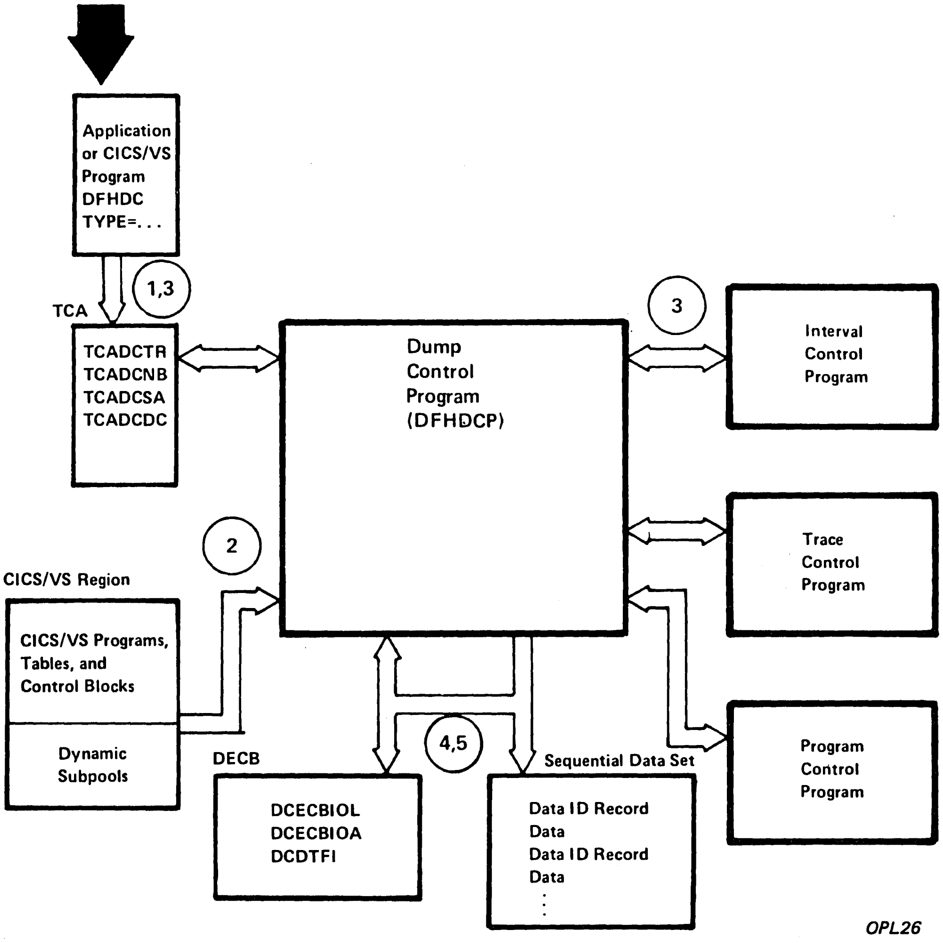


Figure 3-19. Dump Control Interfaces

## DATA INTERCHANGE PROGRAM (DFHDIP)

The data interchange program is designed as a function manager for SNA devices. It is invoked by DFHDI requests directly or by the BMS routines from DFHBMS requests (that is, the BMS routines issue DFHDI requests) DFHDIP provides the following functions:

1. Determines whether a new output destination has been specified and if so, builds appropriate FMHs to select the new destination, and outputs these FMHs to the SNA device via terminal control.

2. Invokes the appropriate subroutine to perform the desired function:

   ADD — build ADD FMH, transmit it and the user data.
   REPLACE — build REPLACE FMH, transmit it and the user data.
   ERASE — build ERASE FMH, RECID FMH and transmit it.
   NOTE — build NOTE FMH, transmit it, return the reply to the user.
   QUERY — build QUERY FMH, transmit it, and output END FMH.
   SEND — output user data.
   WAIT — wait for completion of the I/O.
   END — build END FMH and transmit it.
   ABORT — build ABORT FMH and transmit it.
   ATTACH — remove FMH from initial input.
   DETACH — free the storage used by DFHDIP.
   RECEIVE — read a complete record from the logical device.

3. Sets the appropriate return code.

The notes below refer to Figure 3-20.

Notes

1. The application program issues DFHDI requests.

2. DFHDIP receives control via the macro invocation.

3. If no storage has been obtained for the data interchange block (DIB) then storage control is invoked. The storage is chained to the TCTTE.

4. A trace entry is made.

5. If logging is present (protected task and message integrity) and if a destination change or function change occurs on output then temporary storage is invoked to write the DIB to recoverable temporary storage.

6. Terminal control is invoked to output any built FMH and also to output the user data. (DFHTC TYPE=WRITE is issued). For input requests DFHTC TYPE=READ requests are issued to obtain a non-null input record.

7. Any errors obtained from the device are decoded and placed in the TCA return code slot. If no errors were detected then a return code of zero is returned.

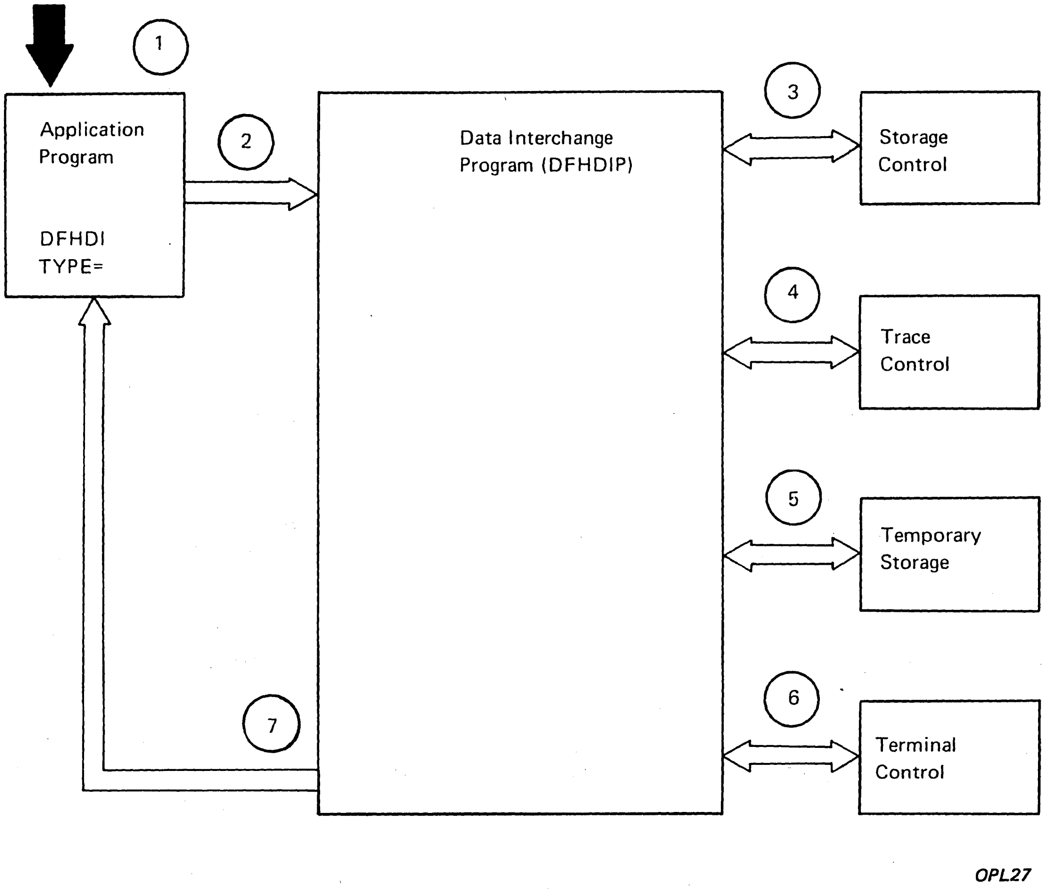


Figure 3–20. Data Interchange Program

## DL/I INTERFACE (DFHDLI)

The CICS/OS/VS-DL/I Interface program (DFHDLI) communicates directly with user-written application programs, DL/I, and other CICS/VS functions. The DL/I Interface accepts requests for DL/I processing from application programs as well as CICS/OS/VS Service modules.

The notes below refer to Figure 3-21.

Notes:

1. When DL/I functions are requested by an application program through execution of a File Control macro instruction (DFHFC TYPE=DLI, xxx), fields within the common communications area of the TCA are filled with appropriate entries to communicate with the DL/I Interface. The functions can also be requested by a CALL or a CALLDLI macro instruction, in which case DFHDLI will fill in the required fields in the TCA. If the request is for a local data base, DFHDLI communicates the requested function to IMS/VS service modules. If the request is for a remote data base, DFHDLI invokes the ISC CONVERSE service to transmit the request to the appropriate system. In addition to processing DL/I input/output requests, DFHDLI will, upon request, schedule and terminate Program Specification Blocks (PSB).

2. DFHDLI receives and communicates requests from the Master Terminal program (DFHMTP) to logically close a DL/I data base (DBRCVR) or logically close for updating a DL/I data base (DBDUMP).

3. DFHDLI receives requests from the Open/Close program (DFHOCP) to physically open or close a DL/I data base. These requests are normally issued by DFHOCP on behalf of DFHMTP.

4. DFHDLI receives requests to terminate DL/I processing (UNLD). This request is accepted only from the System Termination program (DFHSTP).

5. In servicing the requests mentioned above, DFHDLI communicates these requests to DL/I modules and also uses various CICS/OS/VS modules. DFHDLI issues requests to File Control to perform a LOCATE function, to Journal Control to perform logging functions, to Storage Control to obtain required storage, to Task Control to perform the WAIT function, and to Program Control in the event of abnormal termination. DFHDLI also uses the DL/I Interface Parameters and the DL/I Interface Scheduling Block (ISB) for storing and retrieving control information. If the request is for a remote data base, then a Remote Scheduling Block (RSB) is used instead of the DL/I Interface Scheduling Block.

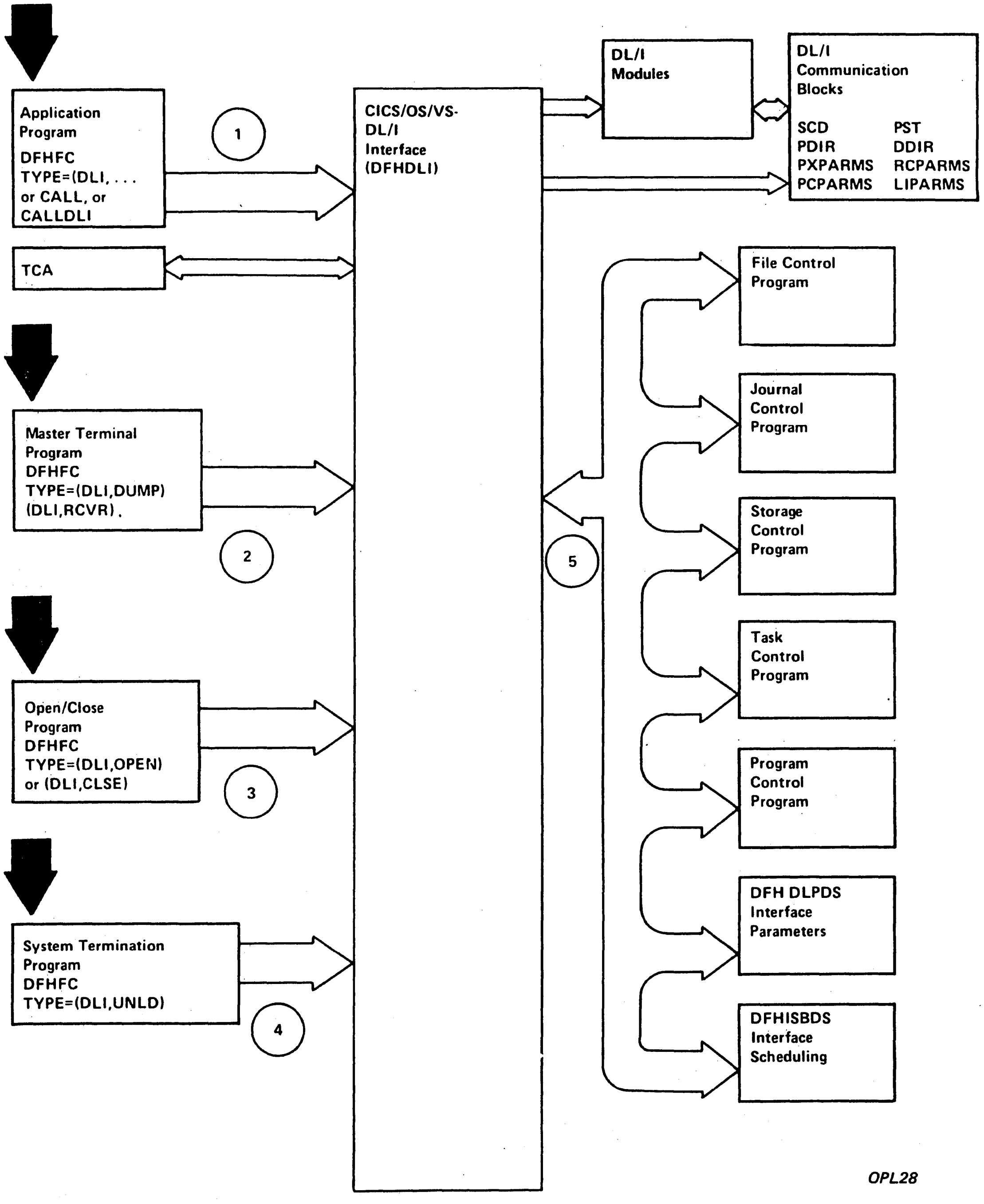

Application Program DFHFC TYPE=(DLI, . . . or CALL, or CALLDLI
TCA
Master Terminal Program DFHFC TYPE=(DLI,DUMP) (DLI,RCVR).
Open/Close Program DFHFC TYPE=(DLI,OPEN) or (DLI,CLSE)
System Termination Program DFHFC TYPE=(DLI,UNLD)
1
2
3
4
5
CICS/OS/VS-DL/I Interface (DFHDLI)
DL/I Modules
DL/I Communication Blocks
SCD
PST
PDIR
DDIR
PXPARMS
RCPARMS
PCPARMS
LIPARMS
File Control Program
Journal Control Program
Storage Control Program
Task Control Program
Program Control Program
DFH DLPDS Interface Parameters
DFHISBDS Interface Scheduling
OPL28

Figure 3-21. CICS/OS/VS-DL/I Interface

## DL/I INTERFACE INITIALIZATION (DFHDLQ)

The CICS/OS/VS—DL/I Interface Initialization Program consists of two phases of execution — an initialization phase and a DL/I substitute function phase. In the initialization phase, DFHDLQ communicates with the System Initialization program (DFHSIP) and the DL/I initialization functions. In the DL/I substitute function phase, DFHDLQ provides the interface to mask DL/I function requests and substitutes appropriate CICS/VS functions.

The notes below refer to Figure 3-22.

Notes:

Phase 1

1. Initialization values are passed to DFHDLQ from the System Initialization Program (DFHSIP) in the DL/I extension (DFHLISTA) of the System Initialization Table (DFHSIT) and also in the DL/I Interface Parameters (DFHDLPDS).
2. DFHDLQ invokes the IMS initialization modules which initialize DL/I areas and control blocks.
3. DFHDLQ then initializes parameter blocks in the CICS/VS Interface Parameters (DFHDLPDS) and the Interface Scheduling Blocks (DFHISBDS).
4. DFHDLQ then reinitializes some of the DL/I control blocks in the SCD and PST. Pointers are modified to point to the DFHDLQ DL/I Substitute Function Phase instead of IMS service modules.

Phase 2

1. DL/I passes control to DFHDLQ when requesting certain DL/I services whose pointers have been masked to point to DFHDLQ. Some of the services processed through DFHDLQ are indicated below.
2. A DL/I request for its Wait routine passes control to DFHDLQ, which then issues the WAIT through CICS/VS Task Control.
3. A DL/I request for main storage for PSB and DMB buffers and workspace is intercepted by DFHDLQ which then issues the request through CICS/VS Storage Control.
4. A DL/I request for logging of data base records is intercepted by DFHDLQ which then issues the request to CICS/VS Journal Control to perform the logging function.
5. DL/I request to send a message is intercepted by DFHDLQ, which then sends a message to the Master Terminal destination via CICS/VS Transient Data.
6. A DL/I request for abnormal termination is intercepted by DFHDLQ which requests task abnormal termination via the CICS/VS Program Control.
7. After servicing the DL/I service requested, DFHDLQ returns control to IMS to resume processing.

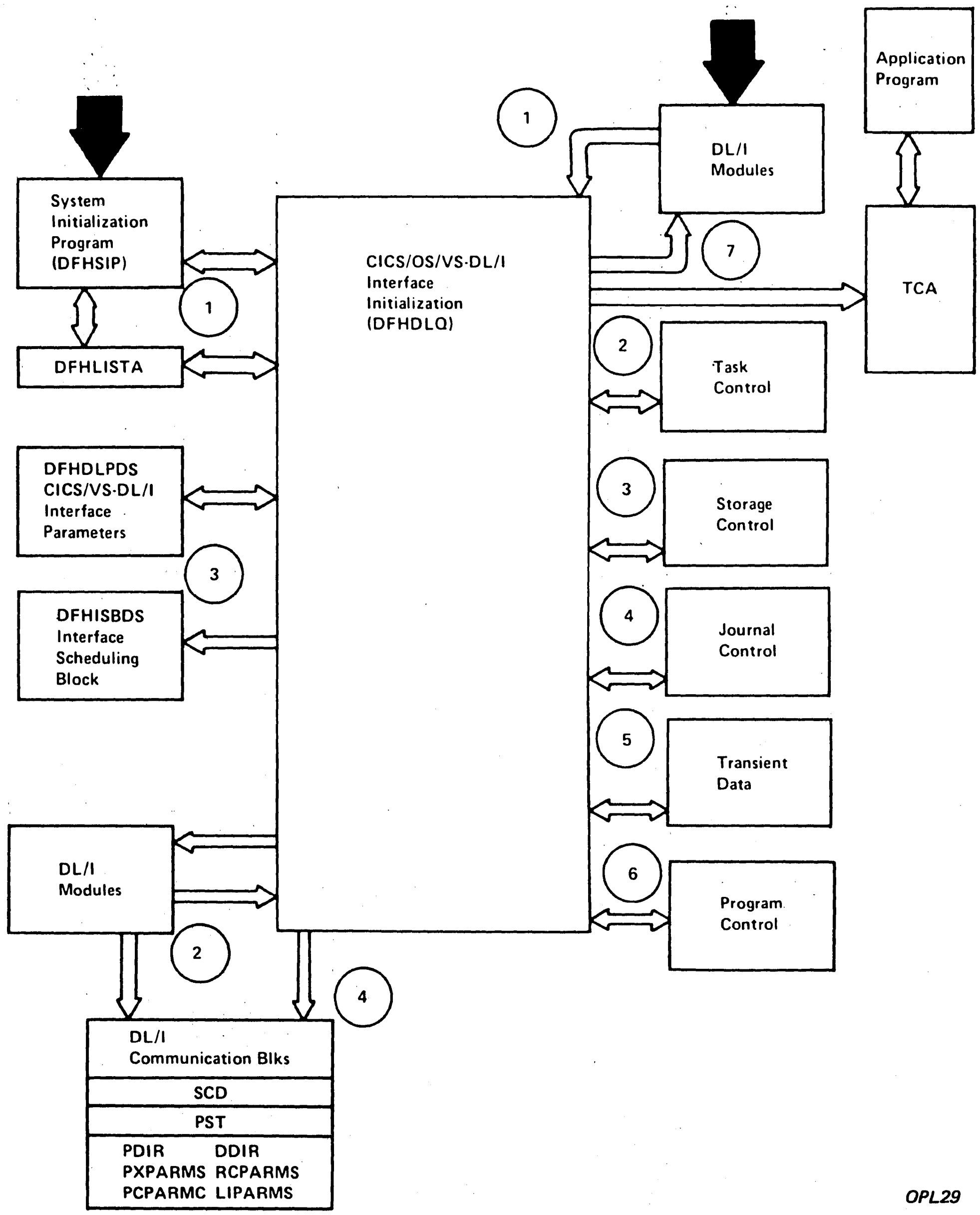


Figure 3-22. CICS/OS/VS-DL/I Interface Initialization Interface

## EXEC INTERFACE (DFHEIP)

The EXEC Interface program is used by application programs using the Command-Level Interface. It invokes the required CICS/VS services on behalf of the application program.

The EXEC Interface program is divided into a number of separate modules. It consists of the EXEC Interface program nucleus (DFHEIP) and one module for each of the CICS/VS management modules. These separate modules are called the EXEC Interface stubs and are named with the following convention:

the stub for DFHFCP is DFHEFC,
the stub for DFHTDP is DFHETD,
and so on.

The notes below refer to Figure 3-23.

Notes:

1. When DFHPCP loads an application program that uses the Command-Level Interface, it invokes DFHEIP initialization routine to set up the Command-Level Interface environment. DFHEIP then invokes the application.

2. The initialization routine obtains and initializes the EXEC Interface Structure (EIS). This contains an EXEC Interface Block (EIB) which is used to pass information to the application program.

3. The statements inserted by the command-language translator for each EXEC CICS command include a CALL statement that causes the application to invoke DFHEIP. The nucleus module DFHEIP then invokes the appropriate EXEC Interface stub.

4. The arguments passed by the application are moved into CICS/VS control blocks by the appropriate EXEC interface stub.

5. The appropriate CICS/VS management module is invoked by the appropriate EXEC interface stub.

6. Information is returned to the application program by DFHEIP via EIB.

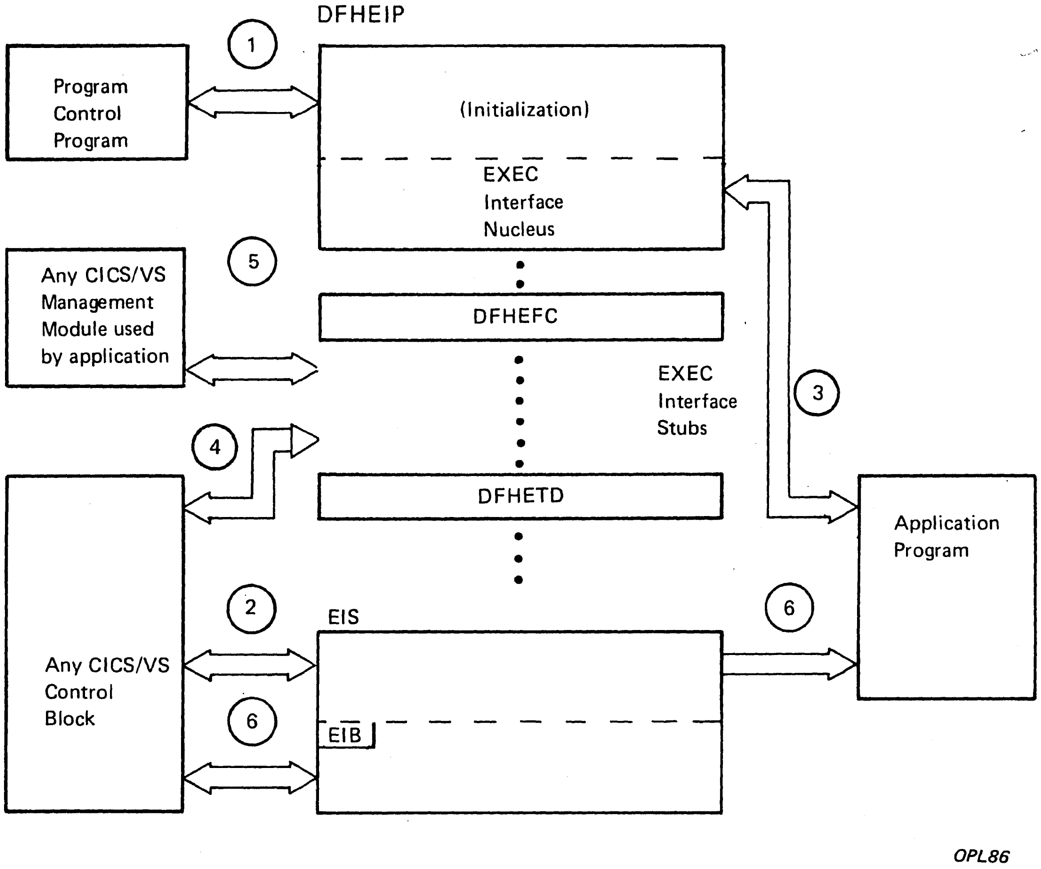


Figure 3-23. EXEC Interface Program Interfaces

## FILE MANAGEMENT (DFHFCP AND DFHFCD)

There are two file control programs, DFHFCP and DFHFCD. DFHFCP processes all VSAM requests, and in addition processes certain requests which may be ISAM or BDAM requests. These common requests which DFHFCP processes are:

OPEN
CLOSE
LOCATE
RELEASE
GETAREA
ESETL

The second file control module processes the remaining ISAM and BDAM requests.

The File Control programs communicate directly with other CICS/VS functions, the standard access methods, and user-written application programs (see Figure 3-24). The on-page connectors show the relationships of the logical elements of the programs.

When file services are requested by another CICS/VS module, File Control locates the necessary File Control Table (FCT) entries according to the OPEN/CLOSE/LOCATE Parameter List. It may perform a logical open or close on those entries.

When file services are requested by an application program (through execution of a File Control (DFHFC) macro instruction), fields within the common communication area of the TCA are filled with appropriate entries to communicate with File Control. The File Control interface is determined accordingly; that is, File Control may:

- Request Storage Control to acquire any required storage areas
- Communicate with the standard access methods to request that any required I/O operations be performed
- Request Task Control to place the application program (requesting task) in a wait state until I/O operations are completed as required
- Request Journal Control to perform any automatic journaling required
- Return the address of the appropriate control block to the requesting task in the field TCAFCAA of the task's TCA
- Request Task Control to enter and leave SRB mode

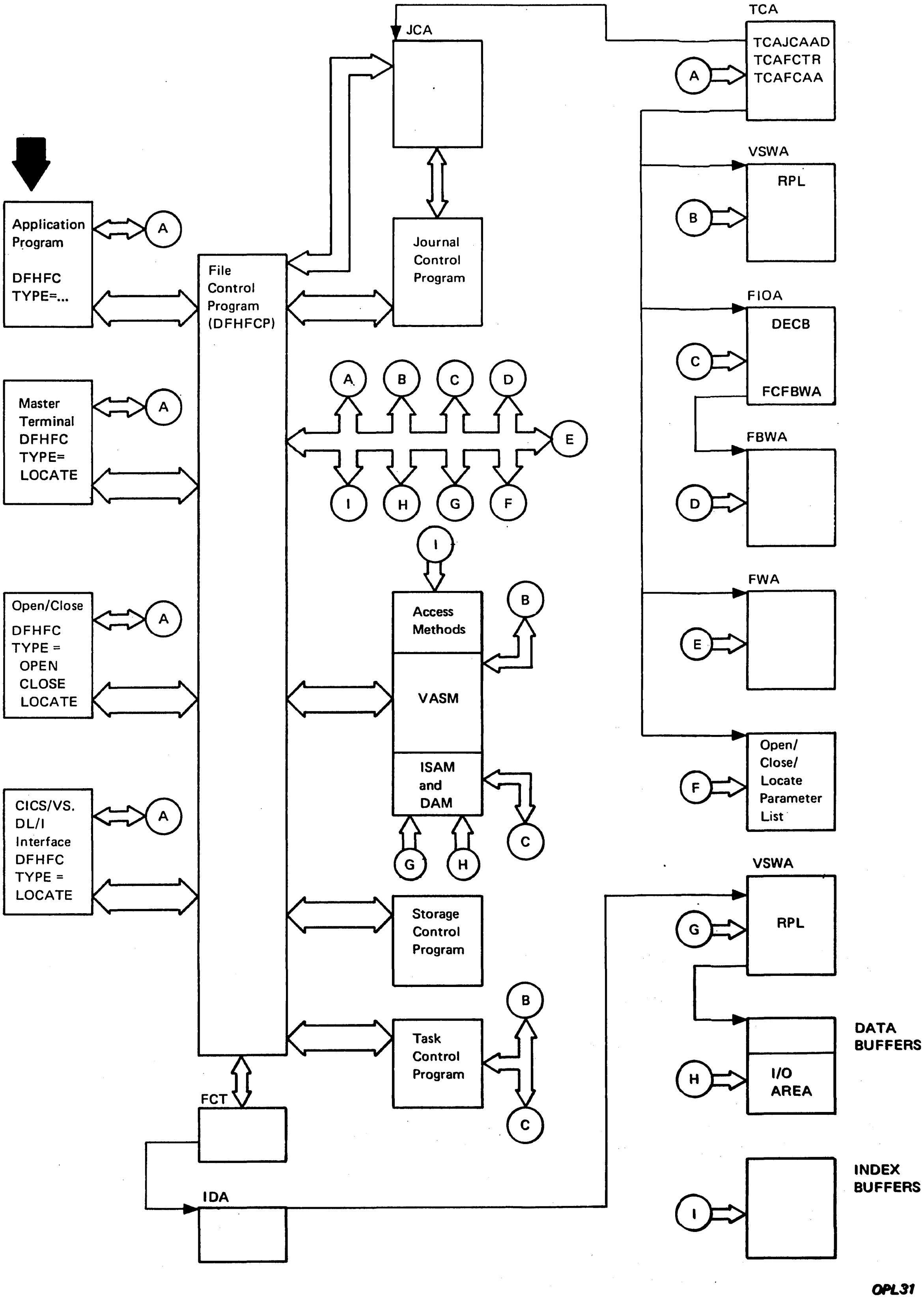


Figure 3-24. File Control Interface

## FORMATTED DUMP PROGRAM (DFHFDP)

The Formatted Dump Program (FDP) will be invoked by abnormal termination of CICS/VS, by specification of DUMP at CICS shut-down time, by the issue of a CSMT SNAP command by the Master Terminal Operator, or if a CICS/VS task is abnormally terminated due to a program interrupt (ABEND code ASRA). The output of the Formatted Dump Program is written to the dump data set and may be printed by means of the Dump Utility Program.

The output of the FDP can be in one of three forms, depending on the specification made either at CICS start-up via one of the start parameters, or at system generation by the FDP parameter of the DFHSIT macro. The three forms of output thus specified are:

1. a dump of the supervisor and CICS partition [DUMP (ABEND)],

2. a dump of the CICS partition only [PDUMP (SNAP)], or

3. a dump of the CICS partition [PDUMP (SNAP)] followed by a series of control blocks, each dumped in as logical an order as possible. Fields to be highlighted in each control block will precede the hexadecimal dump of the appropriate control block.

The Formatted Dump Program (DFHFDP) consists of three modules: DFHFDA, DFHFDB, and DFHFDC.

DFHFDA is the control module of the FDP, and acts as an interface module to CICS/VS and the operating system. It has only one entry point which is used for taking a dump with or without formatting. It contains all system-dependent code and output routines. It also contains the code which is to be executed at CICS initialization time. According to the option selected at CICS start-up time, DFHFDA will issue DUMP (ABEND), or issue PDUMP (SNAP) and return, or issue PDUMP (SNAP), call DFHFDB and return.

The main routines and subroutines of DFHFDA are:

- Initialization (at CICS/VS start-up)

- Main dump routine

- Subroutines used by DFHFDB

- First level program check handler

- Second level program check handler

- Initial working storage area

- Communication area for DFHFDP (DFHFDPDS)

DFHFDB is the module which performs the bulk of the work in producing the formatted dump. It consists mainly of an interpreter which "executes" the text contained in DFHFDC. It calls DFHFDA to perform all operating system-dependent functions. The functions of its main routines are:

- Initialization – reinitializes the necessary areas of the interpreter so that the FDP is serially reusable.

- Primes the interpreter by starting it with a pointer to the Common System Area (CSA) and the text string for the CSA, which must be the first string in DFHFDC.
- Queue scan routine – this is the "scheduler" of the interpreter; it decides which control block should be processed next.
- Work element preparation – having decided which control block is to be processed next, generates the necessary pointers and control information.
- IFETCH – fetches the operation code of the next descriptor and branches to the appropriate descriptor processing routine.
- Descriptor processing routines
- Termination processing routines
- Error processing routines

Module DFHFDC contains the text which is interpreted by DFHFDB, and consists of two CSECTs. The first of these contains fixed length entries, one for each type of control block to be dumped, and acts as an index to the second CSECT which contains the text strings, one for each control block type. The text strings contain "instructions" which describe to the interpreter where it will find pointers to other control blocks, which fields should be formatted, etc. All lengths and offsets used in DFHFDC are determined from the appropriate DSECT, so that any change of position or length in a DSECT will be reflected in the FDP by reassembly of DFHFDC only.

## FASTER 2260 COMPATIBILITY (DFHFIP, DFHF2P)

The FASTER 2260 compatibility routines provide 2260 compatibility on a 3270 for users of the FASTER Language Facility (FLF). However, 2260 compatibility is not available for 3270 support through VTAM.

DFHFIP is invoked by the Terminal Control program after each successful Read operation on a 3270 terminal that was defined as being FASTER 2260 compatible. All input that is not in 3270 native mode (no SBA as a first character) is rearranged to provide to the application program only the data between the start-of-message indicator (SMI), if present, and the cursor. If a new-line character (NL) is encountered, the remainder of the 2260 logical line is dropped.

DFHF2P is invoked by the BMS program for each page of output that is not in 3270 native mode and is destined for a 3270 that was defined as being FASTER 2260 compatible. The program arranges the output page to appear as it would on a 2260 by inserting user-defined SMI and NL characters in the data stream.

## FORMAT TAPE (DFHFTAP)

To prevent invalid recovery due to erroneous data on the System Log, all tape volumes used for this purpose should be formatted prior to use. Formatting magnetic tapes facilitates finding the end of file for mounted volumes if the system abnormally terminates without writing an end-of-file mark. This function can be performed by the standalone program DFHFTAP, which provides the following services:

- Opens message data set
- Acquires 32K of storage for a record area
- Opens tape volume and writes binary zero records

If any of these services cannot be performed successfully, a message is returned and the program is terminated. Otherwise, binary zeros are written until the end-of-file condition is encountered or an I/O error occurs. If an I/O error occurs, no recovery of the write error is performed and no more formatting occurs. The volume is closed and another volume is requested. If no other volume is to be formatted, the program is terminated. If end of file occurs (normal end), the volume is closed and another volume is requested. If no other volume is to be formatted, the program is terminated; otherwise, processing continues with opening and writing of records until all volumes have been formatted.

## TIME MANAGEMENT (DFHICP)

Time Management consists of the Interval Control program (DFHICP), which can be divided into two functional areas. The first of these is that portion of logic that services Interval Control (DFHIC) macro instructions and is executed under control of the requesting task's TCA. The second functional area is that of detecting the expiration of time-dependent events. This function is performed for the Task Dispatcher.

The notes below refer to Figure 3-25.

Notes:

1. Issuing a DFHIC macro instruction causes a type-of-request code to be set in TCAICTR, a field in the requesting task's TCA. Processing enters the Interval Control macro instruction service logic. The register contents are saved in TCAICRS and the requested service is performed or initiated. While performing the service, Interval Control interfaces with other CICS/VS management functions (Storage Control, Task Control, and Temporary Storage Control). Interval Control builds Interval Control Elements (ICEs) for time-dependent requests made via DFHIC macro instructions. The ICEs are chained off the CSA (CSAICEBA) in expiration time sequence (ICEXTOD). Control is returned to the requesting CICS/VS module or user-written program when the requested service has been queued or performed. The DFHIC TYPE=RESET macro service (used to resynchronize the CICS/VS time-of-day with the operating system) interfaces with the timer facilities of the operating system.

2. The Task Dispatcher activates Task Control's TCA for intermodule communication and branches to the expiration analysis logic of the Interval Control program. This logic checks the ICE chain entries, in expiration time sequence, determining whether or not each has expired. Provided certain conditions do not exist, for example, any unfinished 'Post-Initialization' programs, or a 'short-on-storage' condition, each expired ICE is removed from the chain, and processing related to the service requested through the DFHIC macro instruction is performed or initiated. During expiration analysis, Interval Control interfaces with Storage Control and uses Task Control services as well. Since the normal processing flow in the Task Control program is from the requested service logic, the expiration analysis logic is reentered by again activating Task Control's TCA as described above. The time remaining until the next ICE expires is returned in TCAICRT.

   One of the functions of the Task Dispatcher is to continually refresh the current time-of-day values retained in the CSA (CSATODP and CSACTODB). During this process, resetting the clock at midnight can be detected.

3. During expiration analysis, a CICS/VS system task to adjust the expiration times of day to reflect the occurrence of midnight and to reset the CICS/VS current time-of-day and date values to coincide with the operating system can be initiated. The CICS/VS-provided Time Adjustment program performs these functions by adjusting the expiration times in the ICEs (as well as other CICS/VS-maintained expiration times), and then changing the time-of-day values in the CSA to equal the operating system clock.

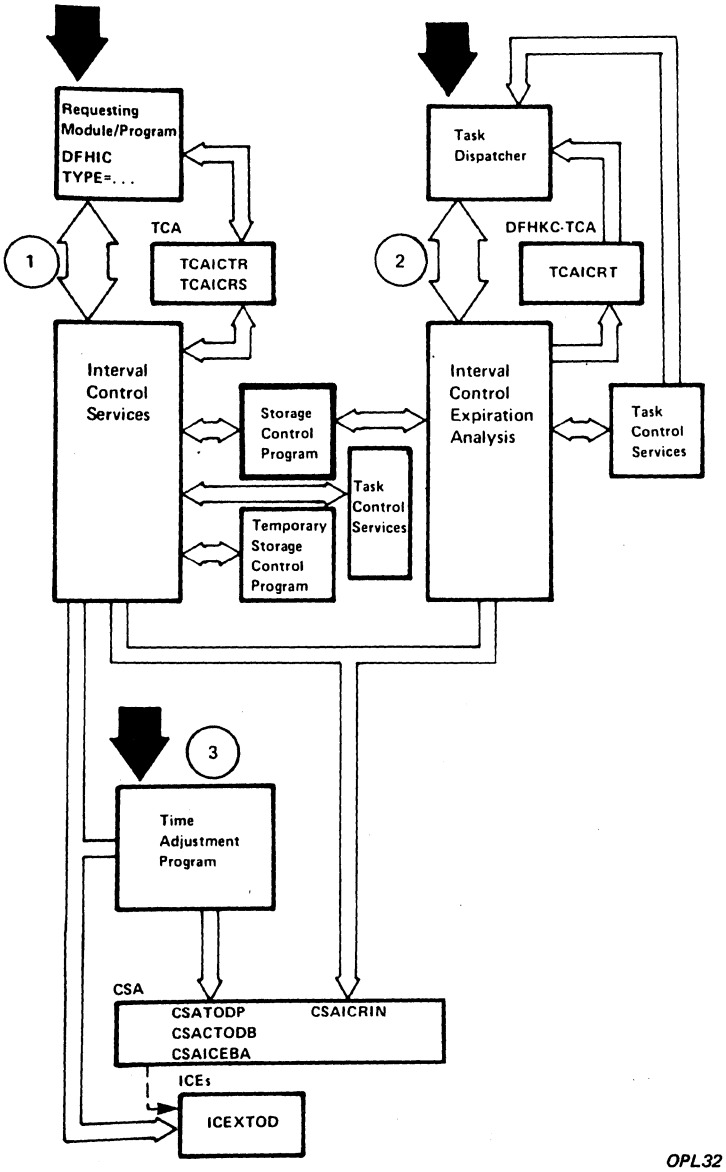


Figure 3-25. Interval Control Interface

## INTERREGION COMMUNICATION (FOR SHARED DATA BASES)

The interregion communication facility enables an IMS/VS DL/I data base to be accessed by CICS/OS/VS and by concurrently active DL/I batch programs in other regions. All requests for access to the data base are coordinated in one region – the CICS/OS/VS region.

DL/I requests from application programs in batch regions are handled, in the first instance, by CICS-DL/I Batch Region Controllers (one in each batch region) instead of by the IMS/VS DB batch region controller. Using an interregion SVC routine, the batch region controller ships the DL/I request to the CICS/OS/VS region where a mirror transaction handles the request as if the request had come from an ISC-connected CICS/VS system.

The DL/I response is subsequently returned to the calling batch region, again by use of the interregion SVC routine.

The interregion communication facility for shared data bases is provided by three sets of modules:

1. Interregion Communication SVC module (DFHIRCP)

2. Batch Region modules (DFHDRP, DFHDRPA/B/C/D/E/F)

3. CICS/OS/VS Region modules (DFHCRC, DFHCRSP, DFHCRNP, DFHZCX).

### INTERREGION COMMUNICATION SVC MODULE (DFHIRCP)

DFHIRCP provides the SVC services required for two or more regions to communicate with each other. The six SVC services are described below.

Note: In the following description of the IRC SVC services, the term subsystem is used in a general sense to mean an application in one region that needs to communicate with an application in another region. (In the data-base-sharing context, the term subsystem refers to CICS/OS/VS in its region or to an IMS/VS job step in a batch region.)

LOGON The LOGON service prepares the subsystem for communication with other subsystems. The LOGON service allows the subsystem to identify itself to the SVC routine and to specify how many simultaneous connections with other subsystems are to be allowed. During LOGON, control blocks are allocated, some in key 0 storage (for use only by the SVC) and others in user-key storage (for use by the SVC and the subsystem).

The following control blocks are allocated in key 0 storage:

- LCB – logon control block. There is one LCB per logged-on subsystem and it is the basic control block used for control of all subsystems.

- CACB – connection address control block. There is one CACB per logged-on subsystem and it contains the addresses of all connection control blocks (CCBs) for the subsystem.

- CCB – connection control block. The LOGON request causes 'n' CCBs to be created per logged-on subsystem, where 'n' is the maximum number of simultaneous connections with other subsystems. ('n' is passed as a parameter to the LOGON service by the user.) The CCB controls a connection to another subsystem.

The following control blocks are allocated in user-key storage:

- SLCB – subsystem logon control block. Within the SLCB is an ECB (event control block) that the SVC routine posts (with an OS/VS POST macro instruction) when the subsystem has any interregion activity to perform; only one SLCB is allocated per subsystem.

- SCACB – subsystem connection address control block. There is one SCACB per logged-on subsystem and it contains the addresses of all subsystem connection control blocks (SCCBs) for the subsystem.

- SCCB – subsystem connection control block. The LOGON request causes 'n' SCCBs to be created, where 'n' is the allowable number of simultaneous connections with other subsystems. The SVC routine allocates one of the SCCBs to represent one end of a connection only when another subsystem issues a CONNECT request. In each allocated SCCB is an ECB that is posted (the SVC routine sets the post bit on) when the subsystem has any new input from the other (connected) subsystem. The SCCB remains allocated until the connected subsystem issues a DISCONNECT request.

- Buffers. Buffers are allocated to hold input to the subsystem from other subsystems.

<u>CONNECT</u> After it has logged on, a subsystem issues a CONNECT request to the SVC routine in order to establish a connection with another subsystem. The SVC routine effects this connection by allocating, in both subsystems, an unused CCB/SCCB pair to represent the connection. The CONNECT request then posts the connected-to subsystem to inform it that it has a new incoming connection. (In data-base sharing, it is the batch region that issues the CONNECT request — as a result of which the SVC routine allocates a CCB/SCCB pair in both the CICS/OS/VS region and the batch region.)

<u>SWITCH</u> After a subsystem has established a connection, the SWITCH service is used to pass control (and optionally data) to the other subsystem. All DL/I requests and responses are shipped between regions by means of the SWITCH service. In performing the SWITCH service, the SVC routine causes the ECBs in the SLCB and the SCCB of the other subsystem to be posted.

<u>DISCONNECT</u> The DISCONNECT service breaks connection with another subsystem; it frees those control blocks that were associated with that particular connection, and posts the other subsystem to say that the disconnection has occurred. The other subsystem then issues its own DISCONNECT to free its own control blocks that were associated with the connection.

QUIESCE The QUIESCE service prevents further connections being established to or from the requesting subsystem, and posts the SLCB ECB when all existing connections are broken. If the request is for an immediate quiesce, the SVC immediately disconnects all active connections to and from the subsystem, at both ends, and posts all the SCCBs involved.

LOGOFF The LOGOFF service removes the subsystem from interregion activity, and the SVC frees all control blocks allocated for the subsystem.

## Invocation of the SVC

On invocation of the SVC, register 1 is set to contain the address of a parameter list. Each element of the parameter list contains the address of a sublist. Each sublist represents a single SVC request. Thus the SVC can process several requests during a single invocation.

The parameter list and sublists are built, and the SVC is invoked, by using the DFHIR macro. For each SVC request, the parameters passed in the sublist to the SVC are coded as operands on the DFHIR macro. The operands are detailed below. The underscored operands represent items set (rather than used) by the SVC during the request.

TYPE=LOGON,MYNAME=,USERNUM=,USERID=,SLCBAD=,MXTHRDS=,BUFSZ=

MYNAME
the name of this subsystem.

USERNUM
a number allocated to this subsystem, and to be used for subsequent SVC requests.

USERID
an ID allocated to this subsystem (for security purposes) and to be used for subsequent SVC requests.

SLCBAD
the address of this subsystem's SLCB.

MXTHRDS
the maximum number of simultaneous connections which can be active for this subsystem. The SVC allocates this number of CCBs and SCCBs for this subsystem.

BUFSZ
the buffer size for this subsystem. There is one buffer for each connection to this subsystem, and the buffer is used to hold data sent to this subsystem from the other end of the connection. The buffer must be large enough to contain such data.

TYPE=CONNECT,TONAME=,USERNUM=,USERID=,THRDNUM=,THRDID=,SCCBAD=

TONAME
the name of the subsystem to which the connection is to be made.

USERNUM
as provided by LOGON.

USERID
as provided by LOGON.

THRDNUM
a number allocated to this connection by the SVC, and to be used for subsequent connection-specific requests to the SVC.

THRDID
an ID allocated to this connection by the SVC (for security purposes) and to be used for subsequent connection-specific requests to the SVC.

SCCBAD
the address of the SCCB for the connection.

TYPE=SWITCH,USERNUM=,THRDNUM=,THRDID=,DATA=

USERNUM
as provided by LOGON.

THRDNUM
as provided by CONNECT.

THRDID
as provided by CONNECT.

DATA
data to be passed to the other end of the connection.

TYPE=DISC,USERNUM=,THRDUM=,THRDID=,DSCTYP=

USERNUM
as provided by LOGON.

THRDNUM
as provided by CONNECT.

THRDID
as provided by CONNECT.

DSCTYP
'N' for normal disconnect.

'A' for abnormal disconnect.

TYPE=QUIESCE,USERNUM=,USERID=,QUITYP=

USERNUM
as provided by LOGON.

USERID
as provided by LOGON.

QUITYP
'I' for immediate quiesce.

'N' for normal quiesce.

TYPE=LOGOFF,USERNUM=,USERID=

USERNUM
as provided by LOGON.

USERID
as provided by LOGON.

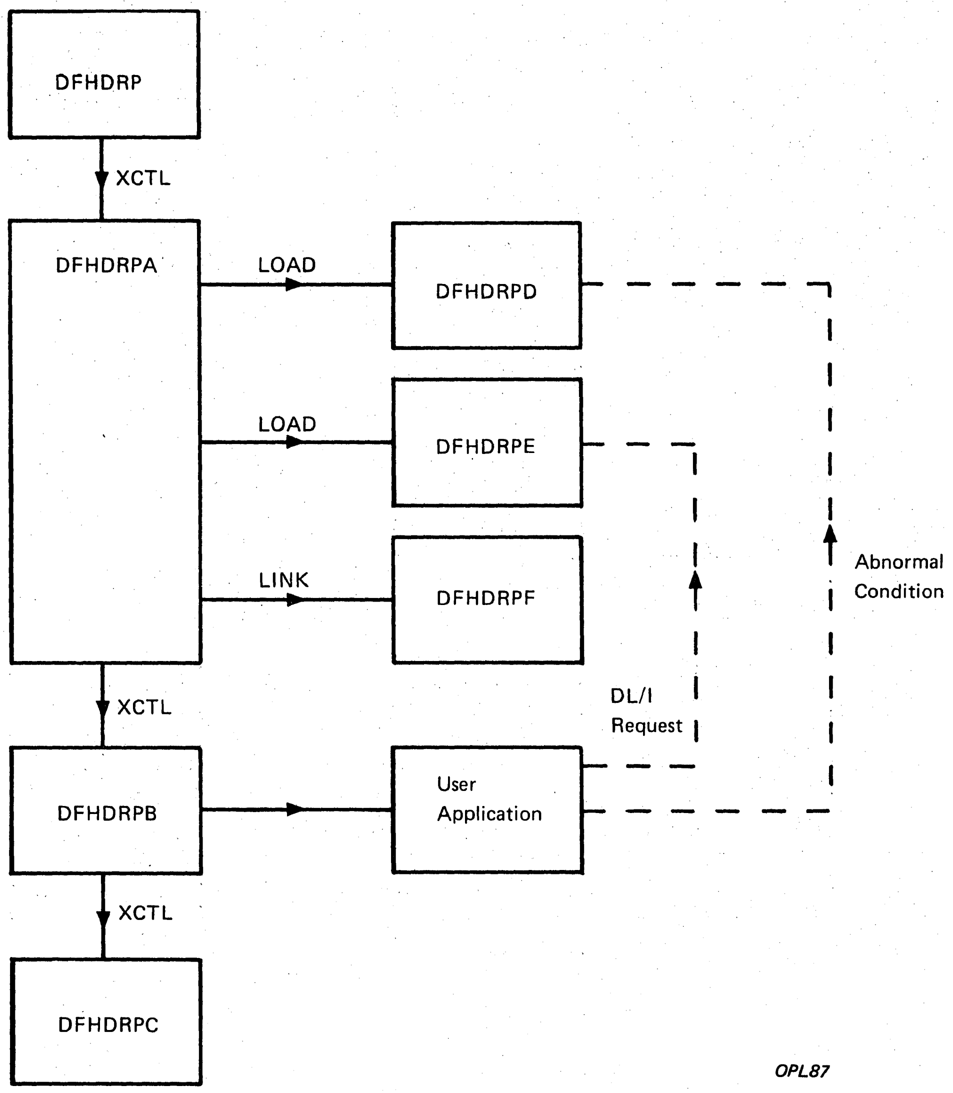


Figure 3-26. Interregion Communication – Batch Region Modules

## BATCH REGION MODULES (DFHDRP,DFHDRPA/B/C/D/E/F)

The batch region modules control the execution of the data-base-sharing job step in the batch region (see Figure 3-26). The functions of each module are described below.

### Batch Region Control (DFHDRP)

DFHDRP is the first module to be executed during a batch data-base-sharing job step. The module resides in the link pack area, and is APF (Authorized Program Facility)-authorized. DFHDRP opens the APF-authorized library, DFHLIB, in which all the batch region controller modules (DFHDRPA-F) reside, and transfers control to DFHDRPA. Using this mechanism the batch region controller modules are authorized, but the user's application program (in STEPLIB or JOBLIB) is not.

### Batch Region Initialization (DFHDRPA)

DFHDRPA processes the user parameters in the JCL EXEC statement, and then loads DFHDRPD and DFHDRPE. Finally, DFHDRPA links to DFHDRPF to establish a connection to the CICS/OS/VS partition/region, before passing control to DFHDRPB.

### Application Program Control (DFHDRPB)

DFHDRPB initiates execution of the batch application program. When the application finishes, DFHDRPB invokes DFHDRPC.

### Batch Region Termination (DFHDRPC)

DFHDRPC invokes the converse routine in DFHDRPE in order to send CICS/OS/VS a sync point request to indicate that DL/I activity is complete. DFHDRPC then invokes the cleanup routine in DFHDRPD to disconnect from the CICS/OS/VS region, and to logoff.

### Batch Region Cleanup (DFHDRPD)

This module contains the cleanup routine, and the STAE and SPIE exits. The cleanup routine is invoked when the batch job step is to terminate, either normally or abnormally. The routine issues a DISCONNECT request to the SVC, in order to break the connection to CICS/OS/VS, and then issues a LOGOFF request.

The STAE and SPIE exits invoke the cleanup routine before continuing the abnormal termination.

## DL/I Request Handling (DFHDRPE)

DFHDRPE contains two routines, the Program Request Handler routine and the Converse routine.

The Program Request Handler receives control, via the link edit stub (DFSLI000), when the application program issues a DL/I request. The routine then invokes the Converse routine in order to send the request to CICS/OS/VS and await a reply.

If the request is a DL/I request, the Converse routine invokes DFHXFQ (which is link edited with DFHDRPE) to transform the DL/I argument list into a format which can be processed in the CICS/OS/VS region/partition. The formatted data is sent by issuing a SWITCH request to the SVC. When the SVC posts the batch SLCB ECB to indicate that the reply has arrived, the Converse routine again invokes DFHXFQ to transform the reply into the format required by the user.

## SVC Initialization (DFHDRPF)

DFHDRPF issues a LOGON request and a CONNECT request to the SVC routine to establish a connection with CICS/OS/VS. It then passes a DL/I argument list (representing a scheduling request for the user's PSB) to the Converse routine in DFHDRPE, so that CICS/OS/VS may schedule the PSB.

## CICS/OS/VS REGION MODULES (DFHCRSP,DFHCRNP,DFHZCX,DFHCRC)

The CICS/OS/VS region modules control the handling of DL/I requests from other regions. The functions of each module are described below.

## CICS/OS/VS Interregion Communication Startup (DFHCRSP)

DFHCRSP is optionally invoked at either system initialization (by DFHSIJ1), or via a CSMT IRC,BEGIN command from the master terminal. DFHCRSP issues a LOGON request to the SVC and attaches CSNC (the New Connection Manager, represented by DFHCRNP). The SLCB ECB is also added to the wait list which is embedded in the DFHTCT table. Thus, the Terminal Control Dispatcher, DFHZDSP (within DFHZCP), will receive control when the SLCB ECB is posted. Finally, DFHCRSP allocates the CICS/OS/VS Region Block (CRB), which controls CICS/OS/VS interregion activity.

## New Connection Manager—Transaction CSNC (DFHCRNP)

Transaction CSNC is attached by DFHCRSP and immediately suspends itself. When the Terminal Control Dispatcher (DFHZDSP) finds either there are new incoming connections from other regions/partitions, or that interregion quiesce is complete, CSNC is resumed by DFHZDSP.

If quiesce is complete, DFHCRNP issues DFHIS TYPE=LOGOFF (see description of DFHCRC), and then DFHPC TYPE=RETURN.

If a new incoming connection exists, DFHCRNP allocates an interregion TCTTE (generated as part of DFHTCT TYPE=IRCBCH), places the incoming data in a TIOA, and invokes DFHIS TYPE=RECEIVE in order to attach the mirror transaction, CSMI. Thereafter, CSMI handles DL/I requests on behalf of the batch program in the same way as it does for a remote ISC-connected system.

If DFHIS TYPE=RECEIVE fails for a temporary reason (for example, MAXTASKS), the TCTTE is chained from the CRB in preparation for a later retry.

## CICS/OS/VS Region Communication (DFHZCX)

DFHZCX contains several routines which are used for interregion communication.

I/O Request Routine (IOR) This routine is invoked when data is to be sent to the connected (batch) subsystem. For instance, when the mirror transaction, CSMI, has performed the DL/I request and wishes to send the response to the requesting batch region/partition, it issues a DFHTC TYPE=(WRITE,WAIT,READ) which is recognized as an interregion request (the TCTTE is an interregion TCTTE), and passed to IOR. The IOR routine issues a SWITCH request to the SVC and awaits a reply (it waits on the ECB contained in the connection's SCCB).

STOP Routine This routine is invoked when the interregion session is to be quiesced. This happens during:

- Abend processing (see description of DFHCRC)
- System termination (via DFHSTP)
- Master terminal command CSMT IRC,END

The routine issues a QUIESCE request to the SVC.

FREE Routine This routine is invoked during sync point processing and issues a DISCONNECT request to the SVC.

LOGOFF Routine This routine is invoked when interregion quiesce is complete. This happens when:

- Terminal Control Dispatcher notices that quiesce is complete, and resumes CSNC (see description of DFHCRNP).
- System termination (via DFHSTP).
- Abend processing (see description of DFHCRC).

The routine issues a LOGOFF request to the SVC.

## Interregion Abnormal Exit (DFHCRC)

DFHCRC is the interregion STAE exit module. The STAE macro is issued in DFHSIB1, and when DFHSRP continues an ABEND, DFHCRC receives control. DFHCRC issues DFHIS TYPE=STOPIMM which (from within DFHZCX) issues an IMMEDIATE QUIESCE request to the SVC. Then DFHCRC issues DFHIS TYPE=LOGOFF which (from within DFHZCX) issues a LOGOFF request to the SVC.

## INTERSYSTEM COMMUNICATION (ISC)

The intersystem communication (ISC) facility (see Figures 3-27, 3-28, and 3-29) enables separate CICS/VS systems to be connected so that a transaction in one system is able to retrieve data from, to send data to, or to initiate a transaction in, another CICS/VS system. The facility is available to application programs that use the command-level interface.

### ISC HANDLING OF CICS/VS EXEC REQUESTS

The user issues a command-level request, which is handled by the EXEC interface program (DFHEIP). DFHEIP invokes DFHELR, which determines whether the request is for a local or a remote resource. If the resource is local, DFHELR returns control to DFHEIP, which invokes the appropriate CICS/VS modules locally.

If the resource is remote, DFHELR allocates a storage area into which it places the name of the remote system, address of the EXEC parameter list, and the address of the table (FCT, DCT, etc.) for the requested resource. Finally, DFHELR issues a DFHIS TYPE=CONVERSE macro, which passes control to the intersystem program DFHISP.

DFHISP determines whether a usable session is already established for the task. If no session is established DFHISP issues an allocate request to establish one. DFHISP then invokes DFHXFP to transform the requested command and parameter list into a form suitable for transmission. (This is known as "transformation 1".) A DFHTC TYPE=(WRITE, WAIT, READ) macro is then issued by DFHISP, and terminal control transmits the request to the remote system. If the NOCHECK option is specified on the EXEC request then a DFHTC TYPE=(WRITE,LAST) macro is issued by DFHISP, to optimize the flows since no reply is now necessary.

On receiving the request, terminal control in the remote system attaches the mirror transaction program (DFHMIR). DFHMIR allocates storage, as in the requesting system, in which the addresses of the session TCTTE and the TIOA are placed. The mirror program then invokes DFHXFP to transform the received request into a form suitable for execution by DFHEIP. (This is known as "transformation 2".)

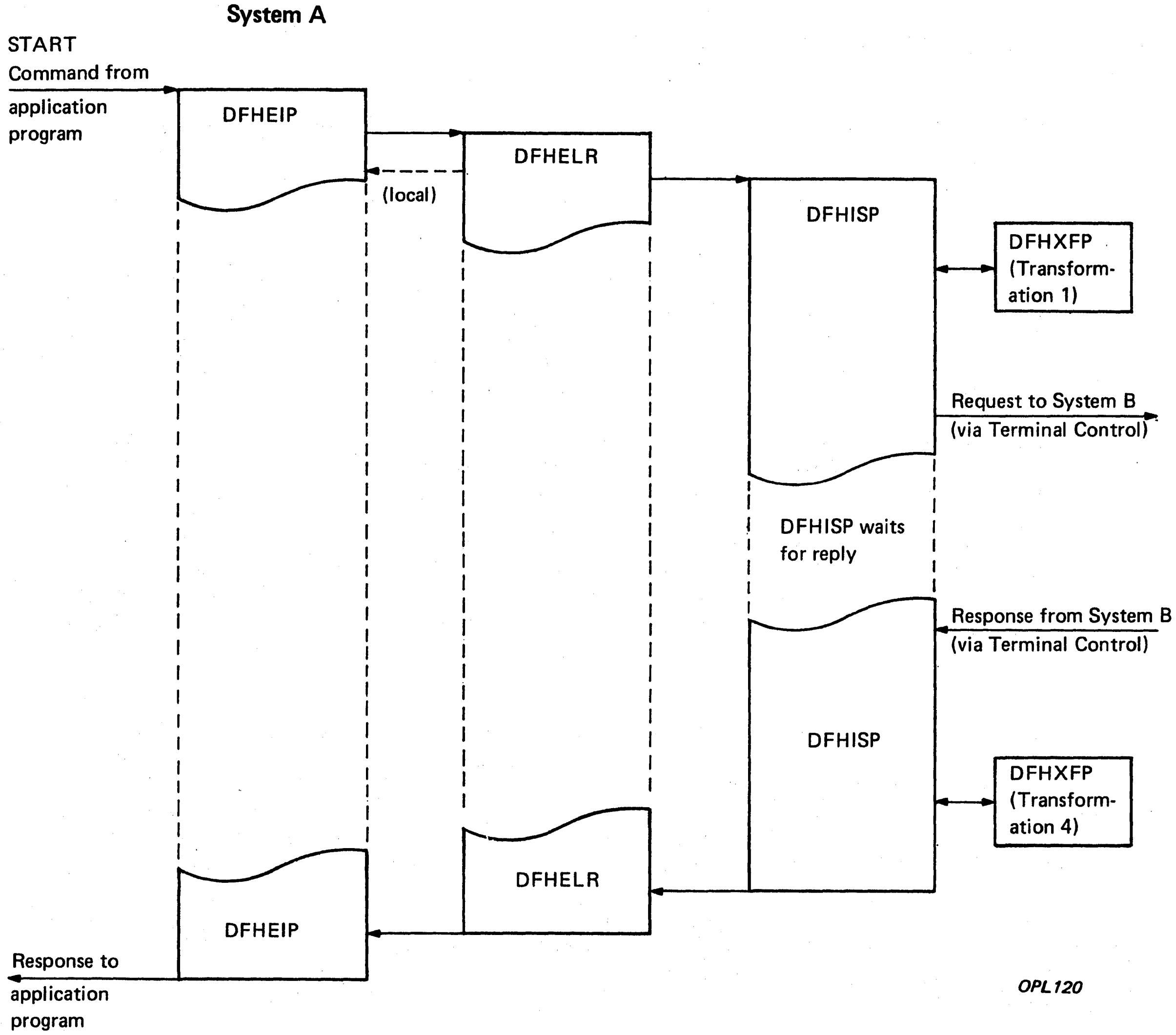
DFHEIP invokes DFHELR which determines whether the request is for a remote resource on yet another system or for a local resource. If the resource is remote, DFHELR allocates a new and separate storage area and invokes DFHISP etc.

If the resource is local, DFHELR returns control to DFHEIP which processes the command for the mirror program in the normal way.

When DFHEIP has successfully completed execution of the command, control is returned to the mirror program with the results of the execution. If the mirror program was attached due to a DFHTC TYPE=(WRITE,LAST) macro request, at this point the macro request has been completed and the mirror program terminates without sending a reply. Otherwise, for requests that do not specify the NOCHECK option, the mirror program invokes DFHXFP to transform the command response into a suitable form for the transmission of the reply. (This is known as "transformation 3".)

DFHXFP returns control to the mirror program which then invokes terminal control to transmit the reply to DFHISP. The mirror program does this by issuing either a DFHTC TYPE=(WRITE, LAST) or a DFHTC TYPE=(WRITE, WAIT, READ) macro.

On receiving the reply, DFHISP invokes DFHXFP in order to transform the reply into a form expected by the application program. (This is known as "transformation 4".) DFHISP then returns control via DFHELR to DFHEIP which raises any error conditions associated with return codes. Finally, DFHEIP returns control to the application program.



|Figure 3-27. (Part 1 of 2). Overview of Intersystem Communication (ISC)

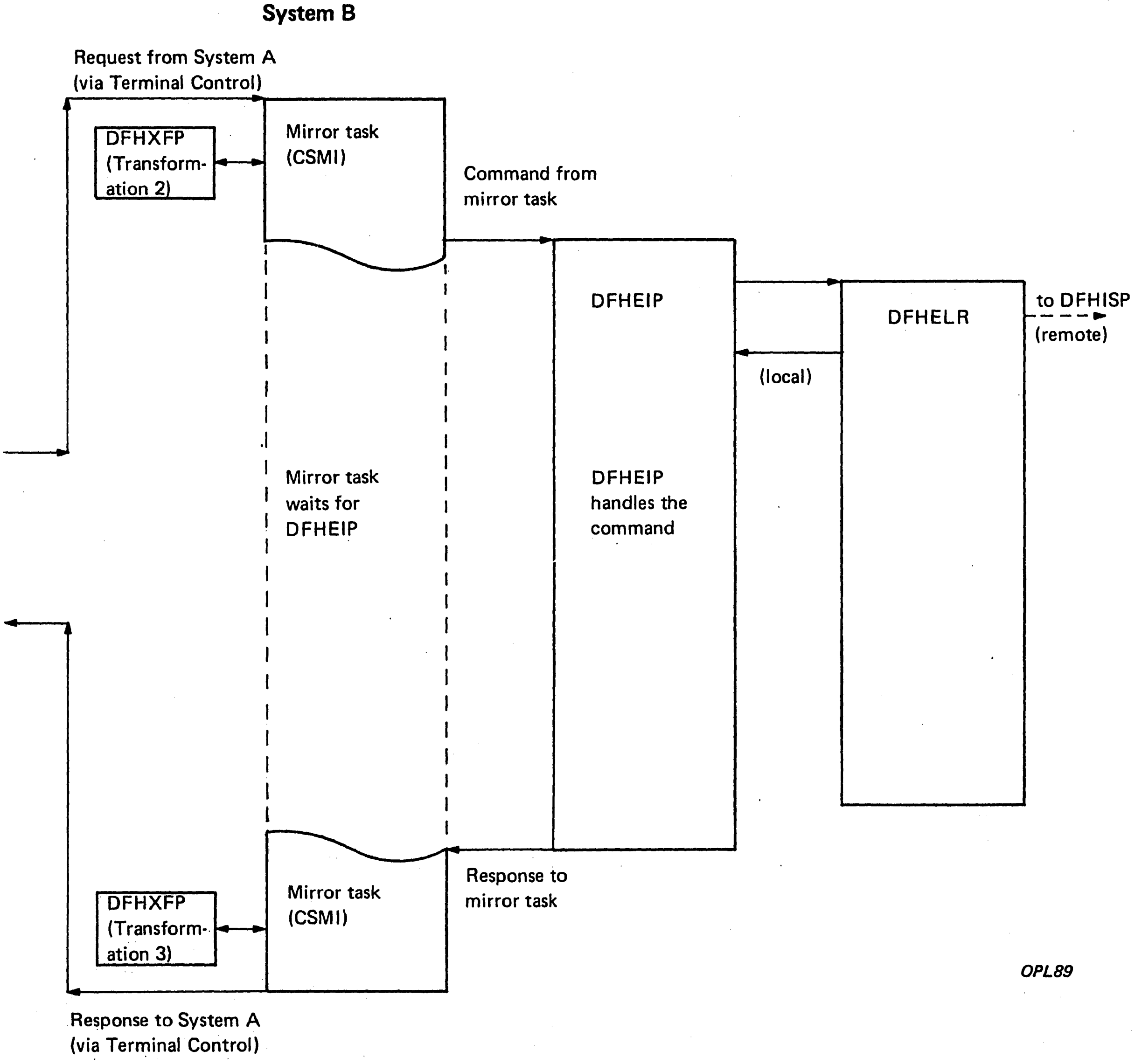


Figure 3-27. (Part 2 of 2). Overview of Intersystem Communication (ISC)

## ISC HANDLING OF CICS/VS EXEC NOCHECK REQUESTS

The command-level NOCHECK requests (see Figure 3-28) are handled in the same manner as other command-level requests, except that:

- There is a single message transmitted to the remote system, and there is no reply transmitted from the remote system.
- DFHISP issues a DFHTC TYPE=(WRITE,LAST) request instead of a DFHTC TYPE=(WRITE,WAIT,READ) request.
- DFHMIR does not issue any terminal control requests.
- Transformations 3 and 4 normally performed by DFHXFP are avoided. (Errors are not returned to the requesting system.)

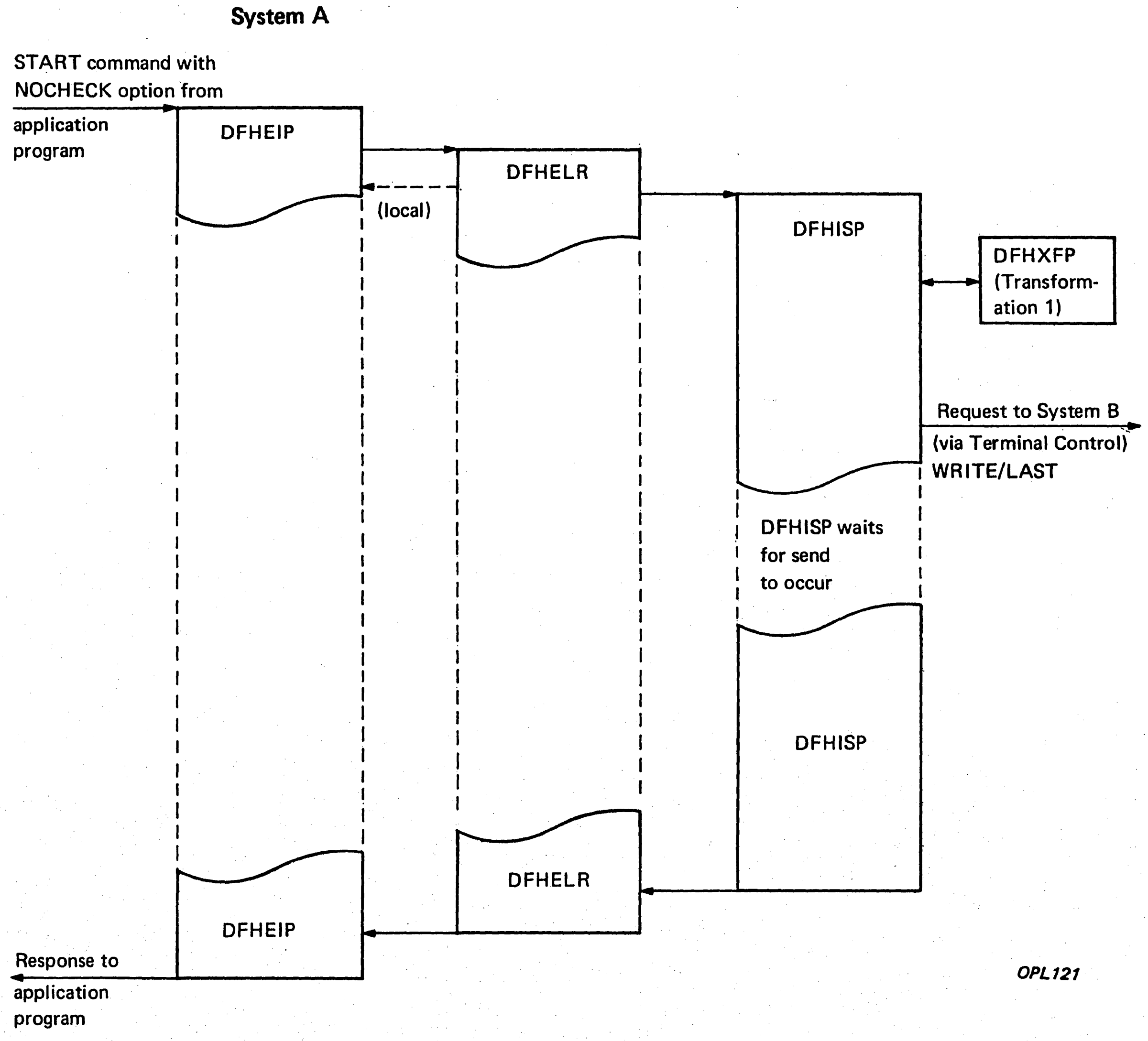

System A
START command with NOCHECK option from application program
DFHEIP
DFHELR
(local)
DFHISP
DFHXFP (Transform-ation 1)
Request to System B (via Terminal Control) WRITE/LAST
DFHISP waits for send to occur
DFHISP
DFHELR
DFHEIP
Response to application program
OPL121

Figure 3-28. (Part 1 of 2). Overview of Intersystem Communication (with NOCHECK Option)

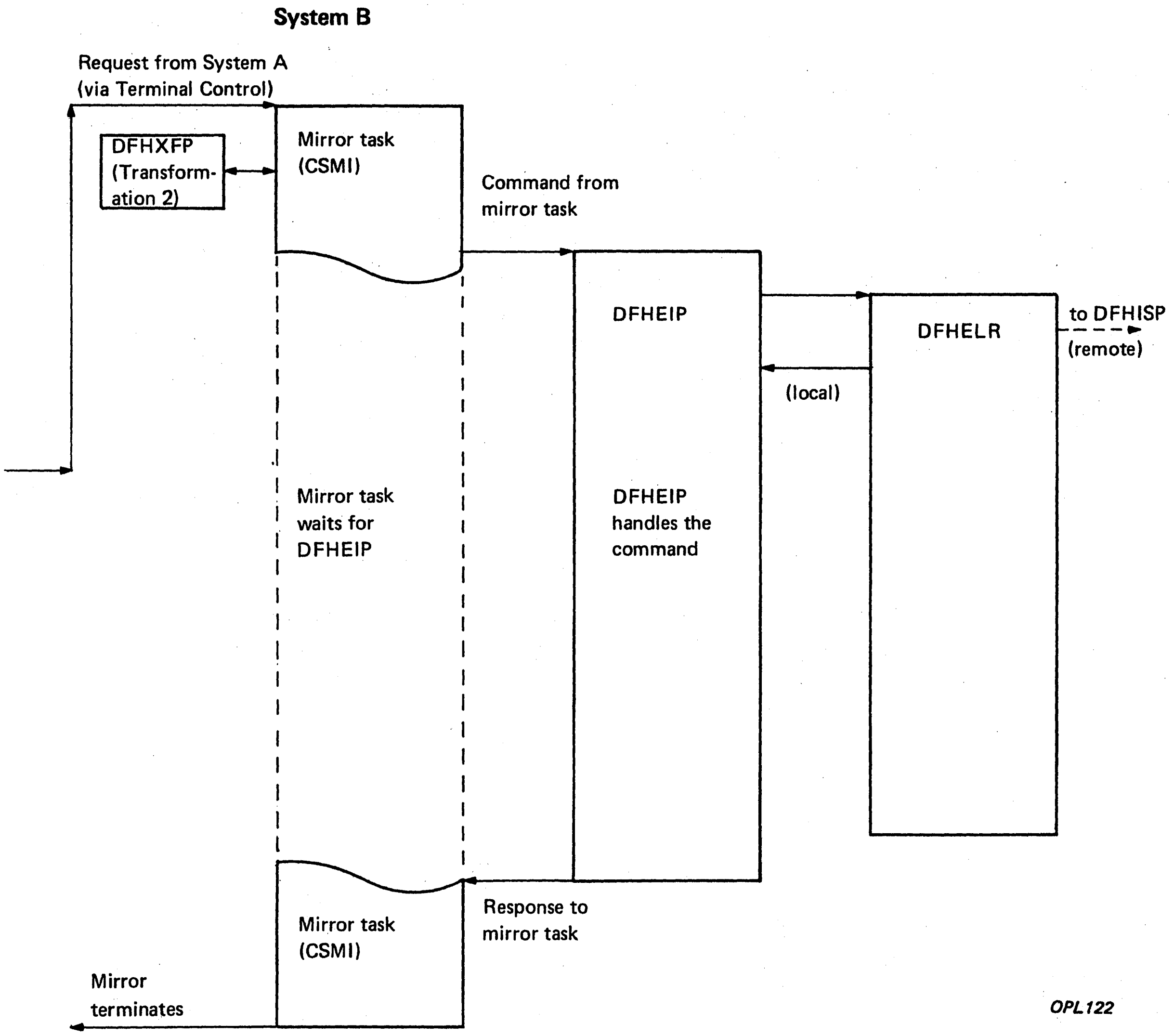


Figure 3-28. (Part 2 of 2). Overview of Intersystem Communication (with NOCHECK Option)

## ISC HANDLING OF DL/I REQUESTS

Whereas all CICS/VS commands (for a local or remote system) are handled by DFHEIP, all DL/I requests are handled by the CICS/VS module DFHDLI (See Figure 3-29). DFHDLI checks whether the request is for a local or remote data base. If local, the request is handled normally. If remote, control is passed to DFHISP. On receiving control, DFHISP invokes DFHXFP to transform the request into a form suitable for transmission, and then invokes terminal control to transmit.

As for a CICS/VS request, the mirror transaction (CSMI) is attached and then invokes DFHXFP to transform the received request into a form suitable for execution by DFHDLI. The mirror program (DFHMIR) then passes the request to DFHDLI in the same way as any other application program would. DFHDLI in the receiving system then checks if the request is for yet another remote system.

When DFHDLI has successfully completed the request, control is returned to the mirror program with the results in the user interface block (UIB). DFHMIR then invokes DFHXFP to transform the results into a form suitable for transmission, and then invokes terminal control to transmit the reply.

On receipt of the reply, terminal control returns to DFHISP, which has been waiting for the reply; DFHISP then invokes DFHXFP to transform the reply into a form that can be used by DFHDLI. DFHXFP sets the return codes in an intermediate control block DFHDRX so that they may ultimately be copied to the UIB or the TCA for the application program. Control is then returned via DFHISP and DFHDLI to the application program.

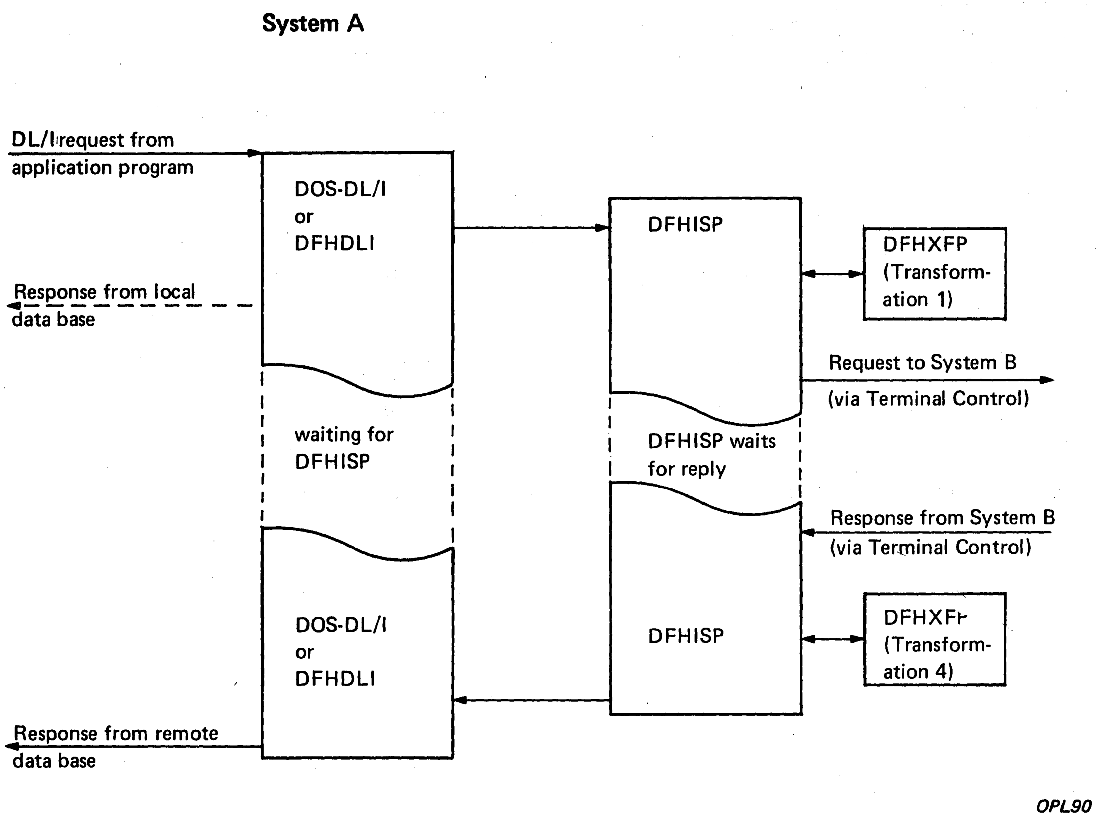

System A
DL/I request from application program
DOS-DL/I or DFHDLI
Response from local data base
waiting for DFHISP
DOS-DL/I or DFHDLI
Response from remote data base
DFHISP
DFHISP waits for reply
DFHISP
DFHXFP (Transformation 1)
Request to System B (via Terminal Control)
Response from System B (via Terminal Control)
DFHXFP (Transformation 4)
OPL90

Figure 3-29. (Part 1 of 2). Overview of ISC Handling of DL/I Requests

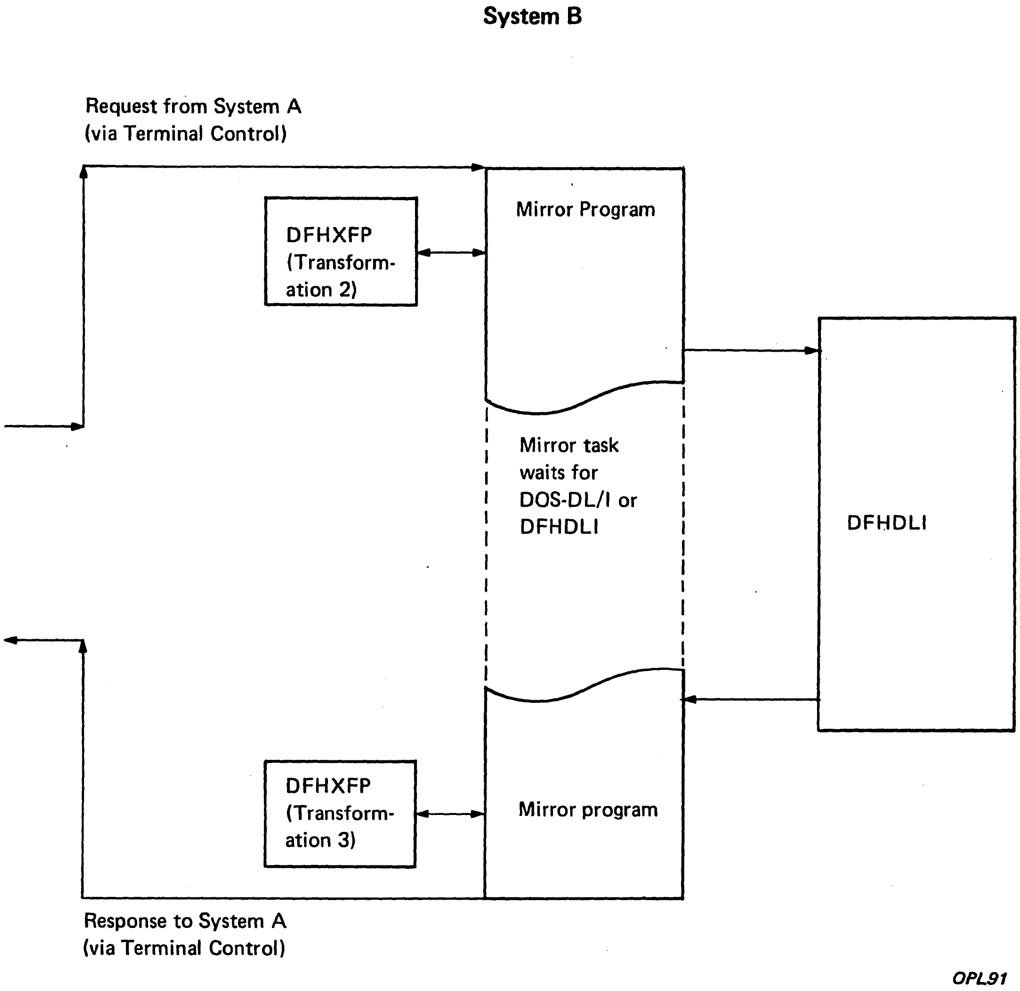


Figure 3—29. (Part 2 of 2). Overview of ISC Handling of DL/I Requests

## LOCAL/REMOTE DETERMINATION PROGRAM (DFHELR)

The local/remote determination program (DFHELR) is called by the EXEC interface program (DFHEIP) and inspects the appopriate CICS/VS table to determine if the request is for a local or a remote resource. The following tests are made to determine if a resource is local or remote:

1. If the function group is not FC, TD, TS, IC, then the resource is local.

2. If the function group is IC, but the function is not START or CANCEL, then the resource is local.

3. If SYSID is specified then the resource is remote.

4. If the relevant CICS table entry specifies REMOTE, then the resource is remote. The relevant tables are:

| | |
|---|---|
| FC | FCT |
| TD | DCT |
| TS | TST |
| IC | PCT |

The notes below refer to Figure 3-30.

Notes:

1. DFHELR is called by DFHEIP to determine if the resource associated with the command-level request is on a local or a remote system.

2. If the resource is local, DFHELR returns control to DFHEIP which invokes the appropriate CICS/VS modules locally.

3. If the resource is remote, DFHELR then:

   a. Allocates a transformer storage area (XFSTG) chained off the EXEC interface storage EIS. XFSTG provides a central area in which all information about processing of the request can be accessed.

   b. Places the following data in XFSTG:

      (1) Name of remote system, for subsequent use by DFHISP.

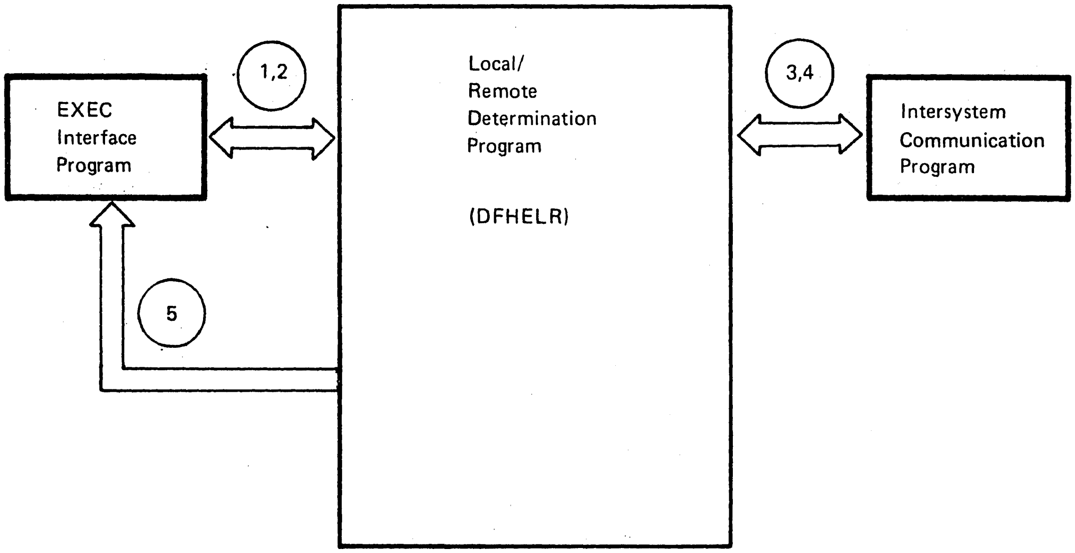
      (2) Address of application's list of parameters (EXEC parameter list) associated with the command being executed.

      (3) Address of the table (FCT, DCT etc.,) for the requested resource.

   c. Issues a DFHIS TYPE=CONVERSE macro which passes control to the intersystem program DFHISP.

4. The reply from the remote system is passed to DFHELR by DFHISP.

5. Control is returned to DFHEIP to examine the reply and invoke the EIP exception handler if necessary.



OPL92

Figure 3–30. Local/Remote Determination Program Interfaces

## INTERSYSTEM COMMUNICATION PROGRAM (DFHISP)

The macro DFHIS TYPE=CONVERSE expands into a call to the intersystem communication program (DFHISP). On receiving control, DFHISP is responsible for allocating a session TCTTE, sending requested command to remote location, receiving reply from remote location, returning result to application program, and if necessary, freeing the session.

The notes below refer to Figure 3-31.

Notes:

1. The local/remote determination program (DFHELR) issues a DFHIS TYPE=CONVERSE macro request.

2. Many ISC sessions can exist at the same time and consequently a task can own more than one TCTTE. Ownership of a TCTTE by a TCA is represented by a chain that starts at the TCA and links all the owned TCTTEs. DFHZISP manipulates this chain and actions requests for session TCTTEs.

   DFHZISP contains three separate routines, POINT, ALLOCATE, and FREE. DFHISP obtains the address of the TCTTE that controls the session with the remote system and places it in XFSTG. The address of the TCTTE is obtained by DFHISP issuing a DFHTC TYPE=POINT macro, the processing being carried out by DFHZISP.

3. If no session is established, there will be no TCTTE; in this case DFHISP issues a DFHTC TYPE=ALLOCATE macro and DFHZISP establishes the session TCTTE. If all the session TCTTEs are in use, the request is queued.

4. The transformation program (DFHXFP) is then invoked to transform the requested command and parameter list into a form suitable for transmission.

5. DFHISP then issues a DFHTC TYPE=(WRITE, WAIT, READ) macro and the terminal control program transmits the request to the remote terminal. If a NOCHECK macro request is being transmitted then a DFHTC TYPE=(WRITE,LAST) macro is issued. No reply is given and the session is immediately freed by issuing DFHTC TYPE=FREE. Control then returns to the caller.

6. Following processing in the remote system the resulting reply is transformed by the transformation program (DFHXFP) into a form expected by the application program.

7. Session TCTTEs can be freed either explicitly or implicitly. If the mirror program detects that the request has completely terminated, it will issue a DFHTC TYPE=(WRITE, LAST) macro and then issue a DFHPC TYPE=RETURN macro. Since DFHISP receives the end bracket indication, it will explicitly issue a DFHTC TYPE=FREE macro, processing being carried out by DFHZISP, in order to release the session TCTTE.

8. Control is then returned to DFHEIP via the local/remote determination program (DFHELR).

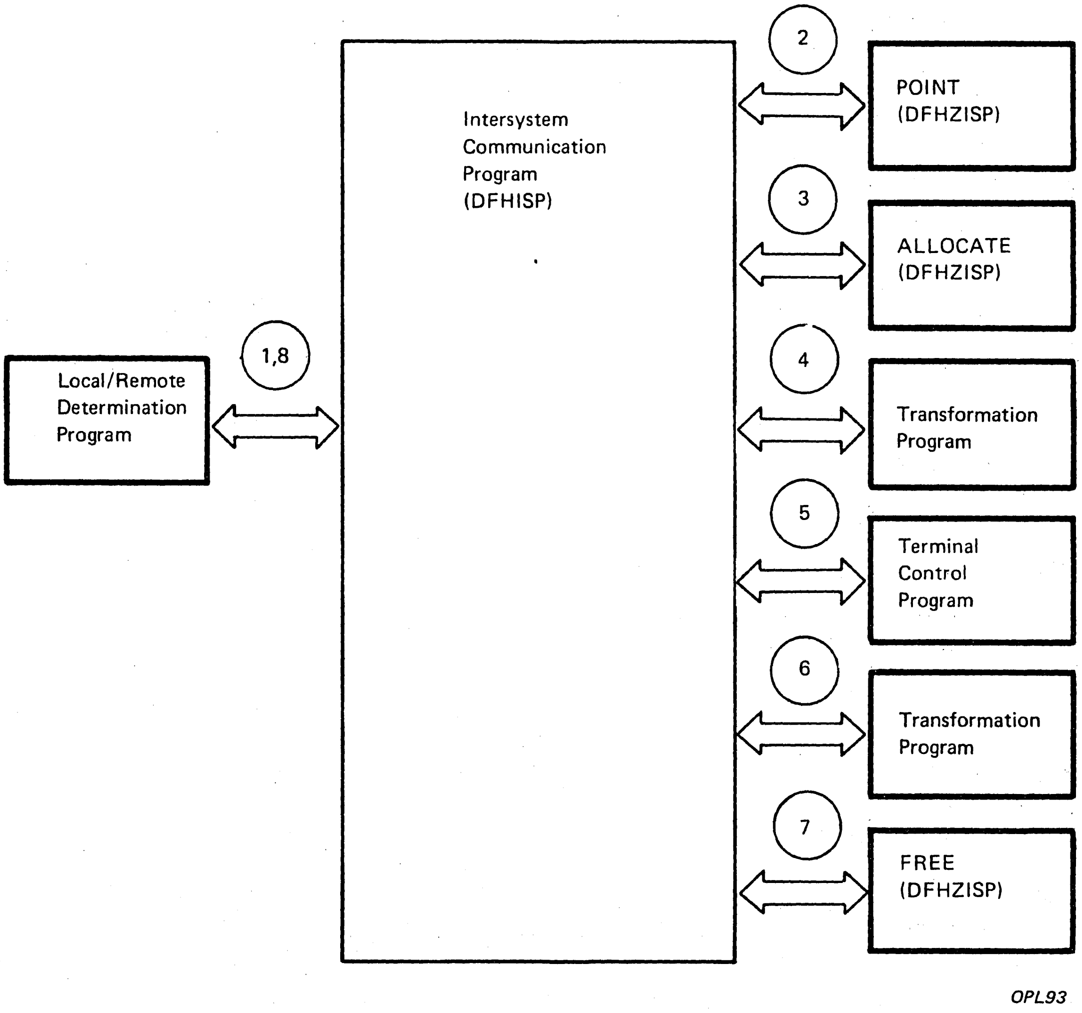


Figure 3–31. Intersystem Communication Program Interfaces

## TRANSFORMATION PROGRAM (DFHXFP)

The transformation program (DFHXFP) is responsible for encoding data that is to be transmitted during intersystem communication. For any request (and reply), the data has to go through four transformations:

1. From standard parameter list form to address free form for the request.

2. From address free form to standard parameter list form for the request.

3. From standard parameter list form to address free form for the reply.

4. From address free form to standard parameter list form for the reply.

Whilst each of the transformations is logically distinct from the others, the code for all four transformations is held in the one module DFHXFP.

The requests (and replies) that can be handled by DFHXFP include:

- CICS file control requests

- CICS interval control requests

- CICS transient data requests

- CICS temporary storage requests

- DL/I schedule and data base requests

For each request, the address free form consists of a function management header (FMH) and (possibly) following data. CICS file control requests are represented by FMHs of type 43 which are architected privately by CICS. All other requests are represented by FMHs of type 6 which form part of the overall LU6 architecture.

The notes below refer to Figure 3-32.

Notes:

1. Transformation 1

   The intersystem communication program (DFHISP) invokes DFHXFP to transform the request into a suitable form for transmission.

   DFHXFP establishes the group and function codes for the request. The group code identifies, for example, a transient data request. It is used to locate the tables and code for transient data requests. The function code is then used to locate the routine for a specific request, for example, a WRITEQ-TD request. Finally, the transformation number, in this case 1, is used to locate the routine for the specific transformation; control is then passed to this routine.

   The standard parameter list is scanned twice, the first time to determine how large a TIOA is required, and the second time to complete the TIOA. In determining the TIOA size, attempts are made to reduce the amount of data that is actually transmitted. For example, trailing blanks are removed from queue names.

During transformation 1 an attach function mangement header (AFMH) is generated when a change of architected model occurs, for example from DL/I model to CICS private model. The generated AFMH will cause the mirror table CSMI to be attached in the remote system.

When processing is complete, transformation 1 passes control back to DFHISP.

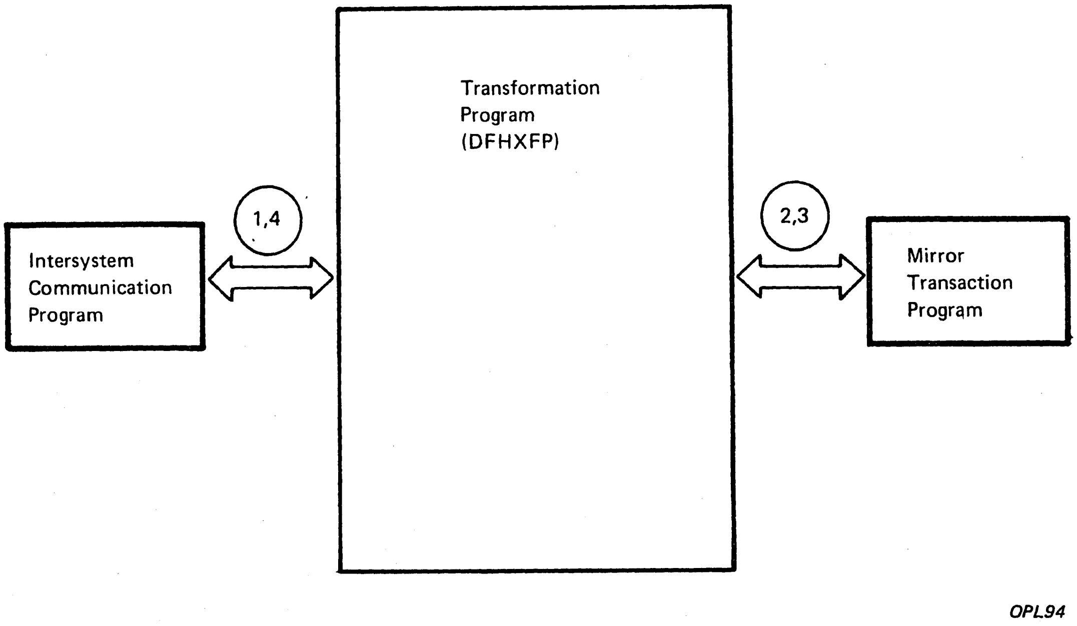


Figure 3-32. Transformation Program Interfaces

2. Transformation 2

   DFHMIR, the mirror transaction program, passes control to DFHXFP which during transformation 2 constructs a standard parameter list from the data contained in the input TIOA.

   During transformation 2 the attach function management header, if present, is ignored in favour of the FMH of interest. The group and function codes, and the type carried in the FMH are used to select the relevant processing code. For CICS private model (file control) requests and DL/I model requests, the processing code will be that which constructs the standard parameter list. For queue and schedule model requests the processing code distinguishes between, for example, a transient data request and a temporary storage request; control is then passed to the routine that constructs the standard parameter list.

   Working storage is then obtained to hold the standard parameter list and values that were compressed for transmission. The storage will also hold the POINTER and LENGTH variables that are to be set as a result of executing a READ SET command. In the case of file control requests, working storage also holds the transmitted key. All remaining values are left in the input TIOA.

   Before finally passing control back to DFHMIR transformation 2 places the address of the parameter list into the XF control block, and also indicates whether DFHHEIP or DL/I is to be called.

3. Transformation 3

   DFHMIR invokes DFHXFP to prepare the reply for transmission. The processing done during transformation 3 is very similar to that carried out during transformation 1. The main difference is that the EIB and UIB control blocks have to be considered in addition to the standard parameter list.

   The final actions taken during transformation 3 are to free the working storage obtained during transformation 2 (note that this does not include storage for a file control request) , and to determine whether or not the mirror task can be terminated. This test is complementary to that performed by DFHMIR. Control is then returned to the mirror transaction program.

4. Transformation 4

The reply is received by DFHISP and then DFHXFP is invoked to set up data areas and status fields from the data contained in the input TIOA.

The selection of the specific processing code is very similar to that done during transformation 2. During transformation 4 execution of the request is completed except for the raising of any error condition; this is done by DFHEIP or DL/I.

Where the request was READ INTO, the reply is moved from the TIOA into the user's designated area. For a READ SET request storage is allocated to hold the reply. For all requests, status information is moved into the EIB control block for CICS requests or into TCAFCTR and TCADLTR for DL/I requests. (DL/I schedule requests require special processing because much of the control block structure on the remote system has to be reconstructed in the local system.)

After freeing the input TIOA, DFHXFP returns control to DFHISP.

## MIRROR TRANSACTION PROGRAM (DFHMIR)

The mirror transaction program (DFHMIR) executes under the transaction identification CSMI. It is attached by terminal control on receipt of a message on an intersystem communication TCTTE. Its function is to call DFHEIP to execute the request, and then to return the response to the requesting system. Depending on certain conditions the mirror transaction will then be terminated.

The notes below refer to Figure 3-33.

Notes:

1. Terminal control receives the request from DFHISP and attaches the mirror program (DFHMIR).

2. The transformation program (DFHXFP) is invoked to convert the request into a form suitable for execution by the EXEC interface program (DFHEIP).

3. The mirror program calls DFHEIP to execute the request. If the mirror program is attached due to a DFHTC TYPE=(WRITE,LAST) request, then since no reply can be given, the mirror program issues a DFHPC TYPE=RETURN macro to terminate immediately.

4. Otherwise, on receipt of the reply from DFHEIP, the mirror program invokes DFHXFP to convert the reply into a suitable form for transmission.

5. The mirror program then issues a DFHTC TYPE=(WRITE, LAST) macro if it detects that the request has completely terminated. If the request has not terminated, DFHMIR issues a DFHTC TYPE=(WRITE, WAIT, READ) macro. This macro is issued in the following circumstances:

   a. If the mirror program has altered a recoverable resource.

   b. If the mirror program has executed a request that gives the mirror program access to a file or positional information in browsing or updating a file.

   In the two cases above the termination of the mirror task will be initiated, not by DFHMIR, but by the user issuing a sync point. On receiving a PREPARE or SPR message as a result of the sync point, DFHMIR will issue a DFHPC TYPE=RETURN macro. The mirror task is thus terminated and return is then made via terminal control to DFHISP.

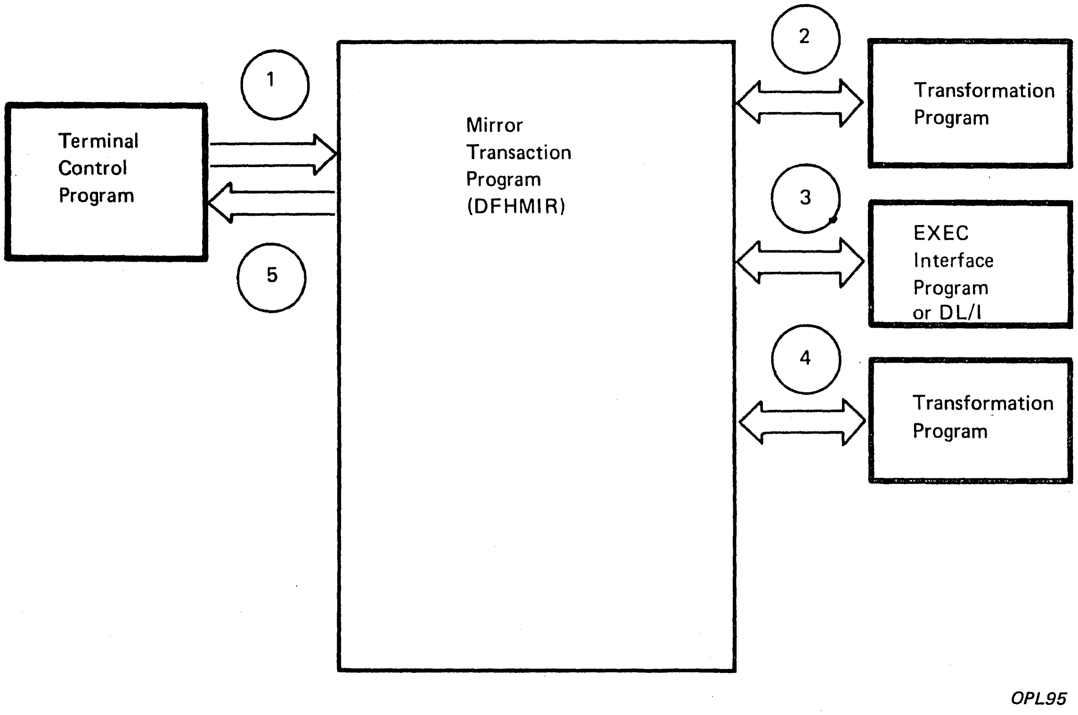


Figure 3—33. Mirror Transaction Program Interfaces

## JOURNAL MANAGEMENT (DFHJCP)

The Journal Control program (DFHJCP) communicates directly with three other CICS/VS management modules: Task Control, Program Control, and Interval Control. In addition, it communicates with the operating system and standard access methods, and within its component parts.

Not all portions of the Journal Control program need be resident in main storage. Some seldom-used Journal Control macro services are performed by separate modules (transients) that are loaded into main storage when required.

CICS/VS provides optional Automatic Journaling facilities for records that are updated, deleted from, or added to a file control data set. Automatic Journaling is specified in the File Control Table, by the user, for each data set affected. For a specified data set, a record read for update, a new record added, or an existing record deleted is automatically written to the specified journal.

The notes below refer to Figure 3-34.

Notes:

1. Journal Control (DFHJC) macro instructions are communicated to Journal Control through a special area, the Journal Control Area (JCA). This area must be acquired for a task by means of a DFHJC macro instruction before any other DFHJC macro instructions are issued by the task. The TCAJCAAD field of the TCA then points to the JCA for the duration of the task.

2. The normal DFHJC macro expansion sets indicators and addresses in the JCA, before branching to Journal Control.

3. Program Control services are requested by Journal Control when a request for a seldom-used macro service (for example, OPEN, CLOSE, or input) performed by a nonresident module (transient) is received. A DFHPC TYPE=LINK macro instruction is used to request that Program Control load the program. All output and wait requests are serviced by resident code.

4. The Journal Control Open and Close transients use the Journal Control Open/Close List to communicate requests to an operating system subtask, which then issues the actual Open or Close. The main CICS/VS task remains dispatchable even while a journal open or close is outstanding. Communication is via OS/VS POST plus CICS/VS Task Control DFHKC TYPE=WAIT. The Journal Control Open/Close List is located at the start of the Journal Control program and is mapped via COPY DFHJCOCL.

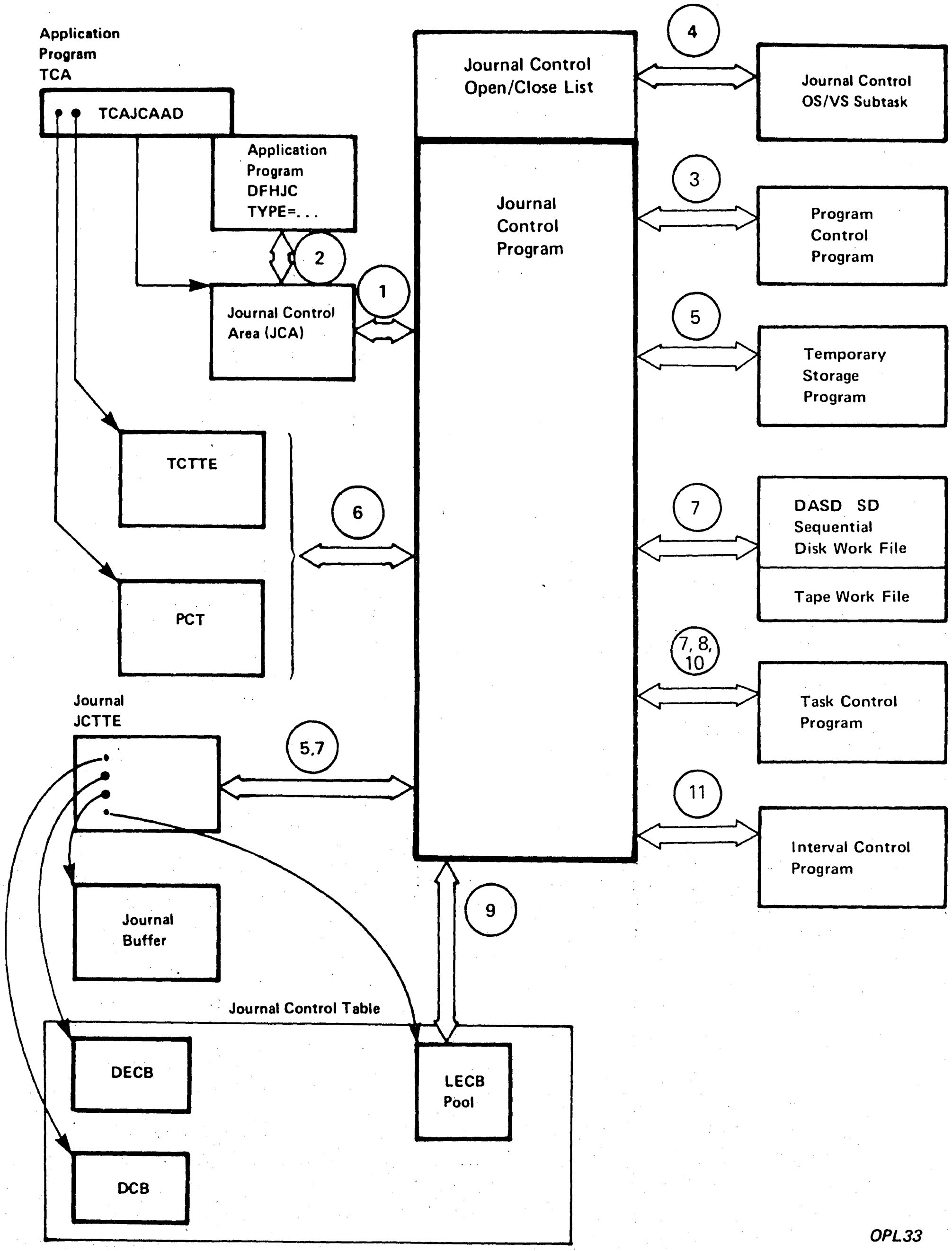


Figure 3-34. Journal Control Interfaces

5. If this task requires logging for Dynamic Transaction Backout, a decision is taken whether to log this record or not. If a record is logged, user storage may be required for a dynamic buffer. If the record will not fit into the buffer, the record or the buffer will spill to temporary storage. The queue used to spill to is idenitified by a byte X'FF' followed by the characters DTB followed by the task number.

   Output requests cause journal records to be built in the appropriate journal's buffer. Journal Control accesses the buffer through pointers in the journal's Journal Control Table Table-Entry (JCTTE). The JCTTE is the repository for all pointers associated with the journal, thus enabling the main Journal Control program to remain truly reentrant. The JCTTE is mapped via COPY DFHJCTTE.

   The journal buffer was acquired at system initialization and is owned as transaction storage by a separate, never-ending CICS/VS journal task. Journal tasks, like CICS/VS Terminal Control, run at high priority and enable I/O to be scheduled rapidly and efficiently for common high-usage resources—in this case, journal data sets. There is one CICS/VS journal task for each journal data set.

6. While building a journal record, Journal Control accesses the system area of the caller's TCA for task number and to locate the terminal identification and transaction identification, for inclusion in the system prefix of the journal record. Journal records are mapped via COPY DFHJCR.

7. If I/O initiation is necessary, Journal Control uses the pointers in the JCTTE to access and set up the journal's DTF. The OS/VS DTF is mapped via COPY DFHJCICA. Journal Control then links to the Basic Sequential Access Method (BSAM) to initiate I/O. If necessary, Journal Control issues a DFHKC TYPE=RESUME macro instruction (a Task Control macro instruction documented only for system programming) for the journal task, which in turn issues a DFHKC TYPE=WAIT on the I/O event.

8. If the user task wishes to wait (synchronize), Journal Control issues a DFHKC TYPE=WAIT macro instruction on a special logical ECB (LECB). The journal task will POST that LECB when the I/O completes successfully.

9. The Journal Control Table (JCT) contains a pool of LECBs, from which LECBs are allocated and deallocated by Journal Control as required. The LECB pool is controlled through pointers at the beginning of the JCT, which is mapped via COPY DFHJCTDS.

10. When there is no outstanding output I/O event for a journal, Journal Control issues a DFHKC TYPE=SUSPEND macro instruction (another Task Control macro instruction intended only for system programming) to suspend the journal task.

11. In the particular case when the user task waits on a journal output request for which STARTIO=NO was specified, then Interval Control is called to provide a Timer Event Control Area which will limit the user wait to a maximum of one second.

This page is left blank intentionally

## TASK MANAGEMENT (DFHKCP, DFHKCSP, DFHHPSVC)

The Task Control program (DFHKCP) can be divided into three functional areas. The first of these supports the Task Control (DFHKC) macro instruction by performing or initiating a particular service requested. As for other CICS/VS service modules, the requesting task's TCA is the communication vehicle and is in control during the execution of the requested service. Unlike other CICS/VS service modules, however, control is not returned directly to the requesting program. Instead control passes to the second functional area of Task Control, known as the Task Dispatcher. The CICS/VS system function of selecting the task to be given control is performed. The third functional area comprises the time-of-day and timer interruption routines that support CICS/VS operations.

The notes below refer to Figure 3-35.

Notes:

1. Issuing a DFHKC macro instruction causes a type of request code to be set in TCATCTR in the requesting task's TCA. Processing enters the Task Control macro instruction service logic. The register contents are saved in TCATCRS, or in CSAOSRSA for RESUME and ATTACH HTA services, and the requested service is performed or initiated.

   While processing the DFHKC macro request other modules may be called. DFHKCP also interfaces with DFHPCP to terminate tasks in certain situations. During ATTACH macro processing the search code in the DFHPCT table is used to find the PCT entry for the new transaction ID. DFHSPP is called during DETACH macro processing for DWE and sync point processing. If the task owned a terminal, then during DETACH macro processing DFHTCP/DFHZCP are called to free the terminal, DFHTSP is called to free data associated with an automatically initiated task, DFHDIP is called to terminate outstanding requests and free the DIB, and DFHICP is called to cancel any ICEs associated with the task.

2. Storage Control services are used by Task Control macro instruction service logic. With the exception of task termination service, the requesting task's TCA remains in control and is the communication vehicle during Storage Control processing.

3. Upon entry to the Task Dispatcher, Task Control's TCA is activated as an intermodule communication vehicle.

4. If the expiry time of any ICE (describing a time-ordered event) has passed, the Task Dispatcher calls a section of the Interval Control program where such events are initiated. The time remaining until expiration of the next time-ordered event is returned to the Task Dispatcher of Task Control. This value is used when operating system services are requested (see note 8 below).

5. Two task chains are maintained by the Task Control program: an active task chain and a suspended task chain. The Task Dispatcher scans the active task chain when attempting to locate a dispatchable task. The suspended task chain is scanned only during exception processing (for example, stall corrective action). The elements on the chains are Dispatch Control Areas (DCAs), each of which contains pointers to the previous and next elements on the chain. In the case of the active task chain, the pointers address the next higher priority DCA and next lower priority DCA (DCAKCFA and DCAKCBA, respectively). Each DCA also contains a pointer to the associated task's TCA (DCATCAA). The CSA contains pointers to the highest and lowest priority DCAs on the active chain (CSAACTBA and CSAACTFA, respectively). The dispatchability of a task is determined by examining the setting of the dispatch control indicator (DCATCDC) in the DCA. If it indicates that the task is waiting on the completion of some event, the ECB associated with the event (pointed to by DCATCEA) is tested for completion posting.

   The suspended task chain is not a priority-ordered chain. If a task has a timeout value associated with it (TCATRTO is non-zero), then task control will add the timeout task to the DCA chain in timeout sequence. CSASUSFA addresses the DCA with the temporarily most remote timeout value. KCP maintains an internal pointer (KCLTPTR) to the most imminent. Tasks with no timeout value (TCATRTO is zero) are placed at the other end of the suspended chain (addressed by CSASUSBA). Placement in timeout sequence helps to minimize the overhead for tasks that have timed-out.

6. If the Task Dispatcher determines that a task is dispatchable, its TCA is activated, the task's register contents are restored, and control passes back to the module or program that requested the Task Control service. During this processing, operating system timer facilities may be used, as described in note 8 below, in support of the runaway task control facility of CICS/VS.

7. If the Task Dispatcher finds no CICS/VS task that is currently dispatchable, it releases control of the CPU through operating system WAIT facilities, requesting that control be returned to CICS/VS upon completion of the next event. One of the events represents the expiration of a time interval as noted in 8, below. Others may be associated with completion of I/O requests, completion of SRB mode activity (see Note 13), etc.

8. The Task Dispatcher interfaces with OS/VS timer facilities. When dispatching a task, the OS/VS timer is set at the runaway task interval value defined by the user. Its purpose is to interrupt any task that is apparently looping and initiate its removal from the system. When relinquishing control to the operating system, the timer is set to the time interval remaining until expiration of the next time-controlled event. This, in effect, requests that control be returned to CICS/VS at a time no later than the expiration of that interval. The dispatcher continually updates the current time-of-day values in the CSA (CSATODP and CSACTODB), but does not use the OS/VS timer facilities for this process.

9. The Task Dispatcher either sets the operating system timer to a maximum time that CICS/VS wishes to relinquish control to the operating system via a WAIT, or uses the timer as a means of interrupting an apparently looping (runaway) CICS/VS task. When a time interval expires, the operating system gives control to the Interval Control Timer Interrupt routine. Based on the setting of an indicator in the CSA (CSAICRIN), the Timer Interrupt routine determines which type of interval has expired. In a non-runaway case, the Timer Interrupt routine merely posts the CICS/VS timer ECB (CSATTECB) and returns control to the operating system.

10. If a runaway task interval has expired, the Timer Interrupt routine takes steps to purge (abnormally terminate) the apparently looping task. The setting of bits in a system-maintained field in a task's TCA (TCASVMID) indicate whether a task is currently executing in a CICS/VS management module or the application program itself. In the latter case, the Timer Interrupt routine ensures that an immediate purge will be initiated when CICS/OS/VS is again given control, then returns control to the operating system. This is done by changing the next instruction operation code to X'00' to force a program check. The SPIE exit then sets the instruction back to its former value and purges the task. If the runaway task interval expired while a CICS/VS management module was executing, the abnormal termination is deferred until control returns to the application program. The Timer Interrupt routine then returns control to the operating system.

11. Occurrence of a timer interrupt when the application program is in control causes the Timer Interrupt routine to replace the "next instruction" (addressed by the interrupt PSW) by an invalid operation code. The consequent operation exception interrupt is recognized by CICS/VS as indicating a runaway task situation. The application code is restored by the program check routine, and the offending task purged by resetting the interrupt PSW to address the abnormal termination linkage in the CSA, see Note 12.

12. When exiting from a CICS/VS management module, the bits representing the module in TCASVMID are set off. If all bits in this indicator are off, the module exits to the task abnormal termination linkage in the CSA.

13. The linkage in the CSA causes entry into the Task Abnormal Termination routine, which requests abnormal termination by means of a Program Control DFHPC TYPE=ABEND macro instruction. The CICS/VS Program Control program terminates the task.

14. If SRB support generated, tasks on the active chain may be running under SRBs, rather than the CICS/VS TCB, during execution of certain CICS/VS services. Such tasks must not, of course, be dispatched by the normal (TCB mode) CICS/VS task dispatcher. Upon completion of the SRB mode service, a task becomes eligible for normal TCB mode dispatch again. To indicate that a task has become eligible in this way, the SRB mode service posts an ECB (DCAPECB in the task's DCA), and places the HTA of the task on the stage chain (SRASTGCH). The task dispatcher dispatches from the stage chain in preference to the active chain except when the "next dispatch time" for the terminal control task (CSATCNDT) expires.

15. In response to an ENTER SRB mode or ATTACH HTA service, KCP issues an SVC SRBSVC (HPO only). This causes control to pass to KCSP running as an SRB. KCP itself continues in TCB mode.

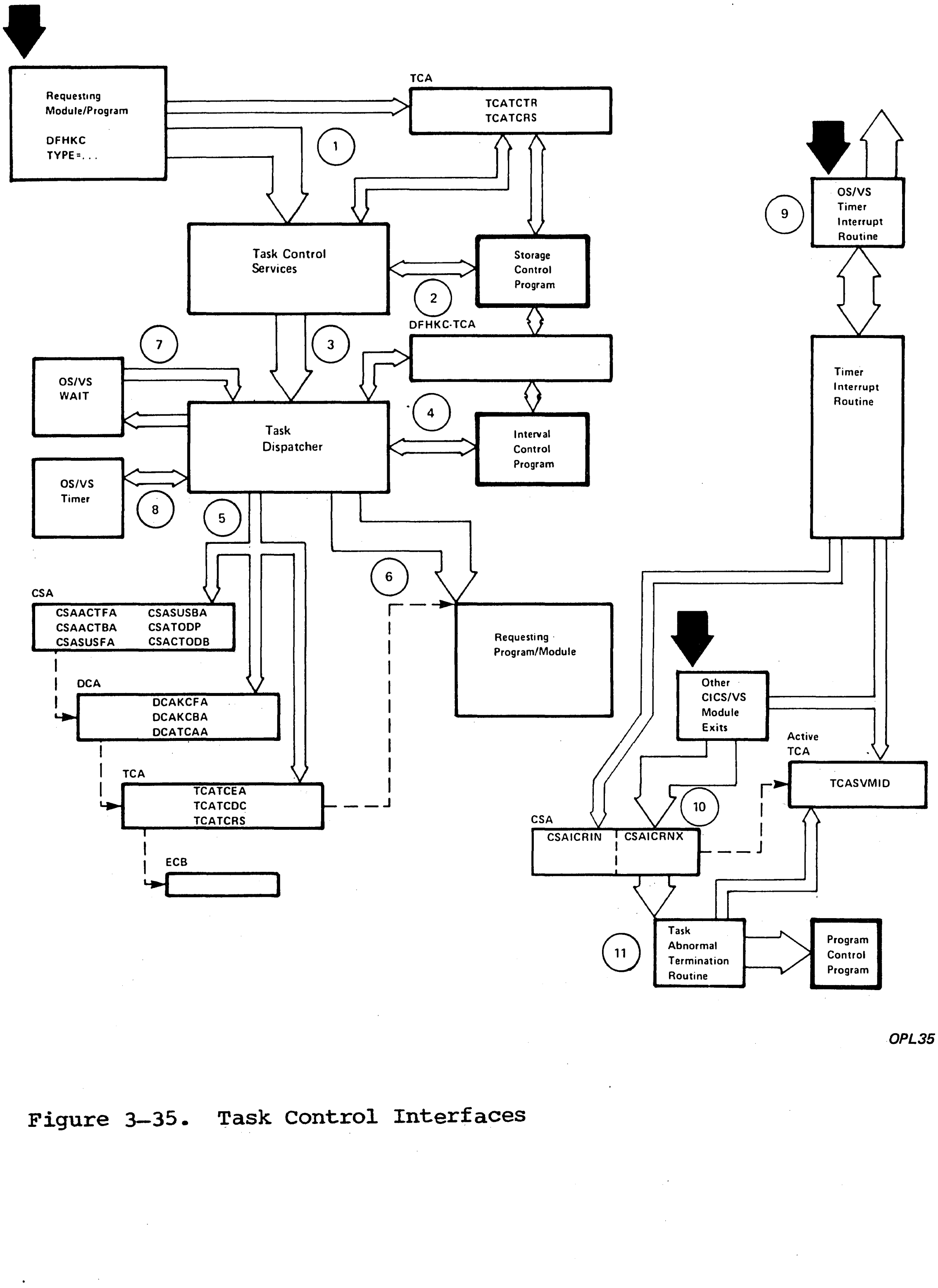


Figure 3-35. Task Control Interfaces

## KEYPOINT PROGRAM (DFHKPP)

The Keypoint program (DFHKPP) is activated for one of three purposes:

- To collect and record data from system tables and control blocks, and write that information to the Restart data set for use by the System Initialization program (DFHSIP) in a subsequent warm start of CICS/VS
- To collect and record data from system tables and control blocks, and write that information to the System Log for use by the Recovery Utility program (DFHRUP) in a subsequent emergency restart of CICS/VS
- To record and retrieve records written to the Restart data set in support of emergency restart. DFHRUP interfaces with DFHKPP to cause records to be written to the Restart data set and the Transaction Backout program (DFHTBP) interfaces with DFHKPP to retrieve these records for recovery purposes.

The first of these functions is known as warm keypointing; the second is called activity keypointing, and the third is called emergency restart support. System interfaces for warm keypointing and activity keypointing are shown in Figures 3-36 and 3-37, respectively.

### WARM KEYPOINTING

The notes below refer to Figure 3-36.

Notes:

1. The Keypoint program is linked to by the System Termination program (DFHSTP) when the system is terminated in response to a user request for termination.
2. The System Recovery program (DFHSRP) may generate a link to the Keypoint program when an unrecoverable error condition precludes further execution of CICS/VS.
3. Information collected by DFHKPP is written on the Restart data set (DFHRSD), which is a BDAM data set with user-specified block size preformatted by DFHSIP.
4. The collected information consists of:
   - PPT, PCT, and FCT – the entire tables
   - TCT – the nonswitched TCTTEs
   - DCT – the intrapartition entries and the bit map
   - TSUT – the auxiliary destination identifications, queue counters, RBAs, and the bit map
   - Interval Control Elements (ICEs) and Automatic Initiate Descriptors (AIDs) – the entire control blocks

- Batch Control Areas (BCAs) – the entire control blocks and their associated Write Request Elements (WREs)

- CSA – certain fields, such as time intervals and maximum task values

When all data has been recorded, a time-stamped control record is written to DFHRSD. This record contains DASD addresses of the data and is used by DFHSIP at warm-start time.

5. Control is then returned to the requesting module, either DFHSTP or DFHSRP.

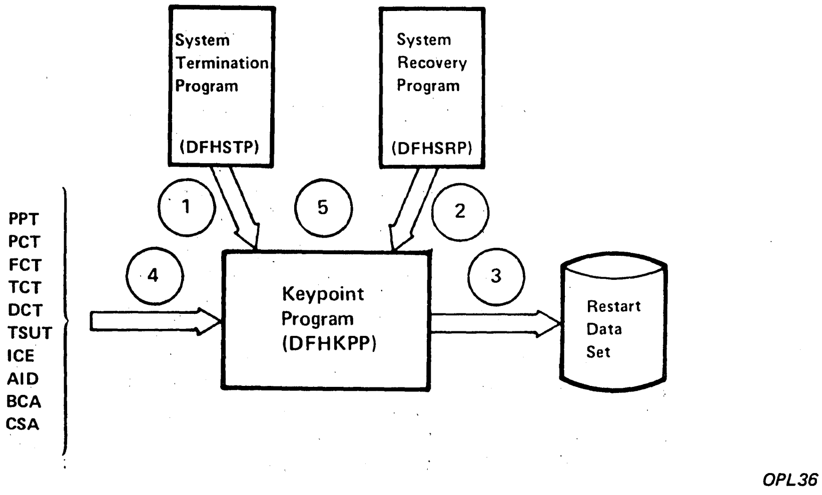

System Termination Program (DFHSTP)
System Recovery Program (DFHSRP)
1
5
2
4
3
Keypoint Program (DFHKPP)
Restart Data Set
PPT
PCT
FCT
TCT
DCT
TSUT
ICE
AID
BCA
CSA
OPL36

Figure 3-36. Keypoint Program Interfaces During Warm Keypointing

## ACTIVITY KEYPOINTING

The notes below refer to Figure 3-37.

Notes:

1. The need for activity keypointing is signaled when an activity keypoint frequency count is reached during Journal Control (DFHJCP) logging of activity on the System Log (JID1).

2. Journal Control attaches the task associated with transaction identification CSKP. CSKP invokes the Activity Keypoint program (DFHAKP).

3. DFHAKP passes control to the Keypoint program (DFHKPP).

4. DFHKPP gathers the TCA, DCT, and TCT information in buffers and interfaces with DFHJCP to log this information on the System Log.

5. DFHKPP returns to DFHAKP, which writes a time stamp to the master terminal (CSMT).

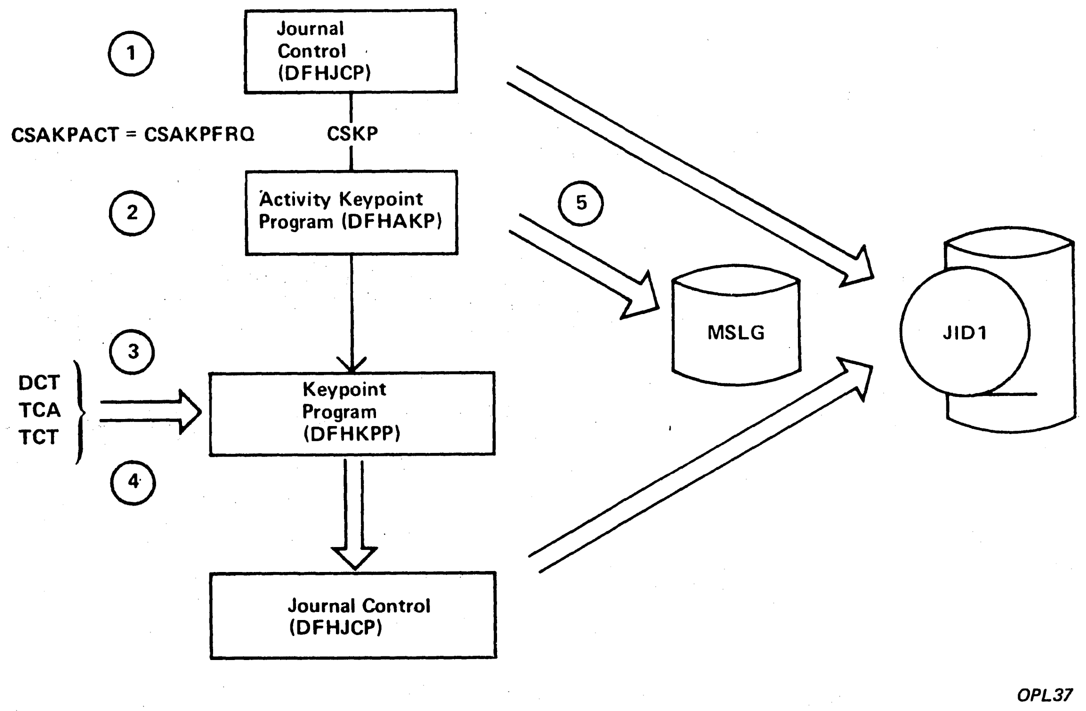


Figure 3-37. Keypoint Program Interfaces During Activity Keypointing

## MESSAGE SWITCHING (DFHMSP)

Message Switching runs as a task under CICS/VS. A terminal operator requests activation of this task by entry of the transaction identification CMSG (or another installation-defined four-character transaction identification), followed by appropriate parameters. Once initiated, Message Switching interfaces with CICS/VS Basic Mapping Support (BMS) and CICS/VS management functions. Although Message Switching appears conversational to the terminal operator, the message switching task is terminated with each terminal response. Through the entry in the PCT which references program DFHMSP, the CICS/VS user can define the transaction identification (which must be four characters long) for message switching. Conversation is forced, if continuation is possible, by effectively terminating the transaction with a DFHPC TYPE=RETURN,TRANSID=XXXX, where XXXX is the transaction identification taken from the task's PCT entry. Actually, XXXX is dynamically moved into TCANXTID prior to issuing DFHPC TYPE=RETURN.

The notes below refer to Figure 3-38.

Notes:

1. If the first four characters of the TIOA (not including a possible SBA sequence from a 3270 terminal) do not match the transaction identification in the task's PCT entry, then this task must have started as part of a conversation, because a previous task set up the next transaction identification. A 'C' immediately following the transaction identification is also a force continuation. In such a case, information has been stored in and has to be retrieved from temporary storage (using a record key of one-byte X'FC', four-byte terminal identification, and three-byte C'MSG') to allow the task to resume where it left off.

2. The operands in the input TIOA are processed and their values and status are stored in the TWA.

3. If a ROUTE operand specifies Terminal List Table(s) (TLTs) for a standard routing list, a DFHPC TYPE=LOAD macro instruction is issued to request the Program Control program to load the requested TLT(s).

4. Message Switching requests storage areas for:

   a. Building route lists (one or more segments, each of which has room for the number of destinations specified by MSRTELNG, an EQU within the program)

   b. Constructing a record to be placed in temporary storage

   c. Providing the message text to BMS if message parts from previous inputs exceed the current TIOA size, a message is completed in the current TIOA but has parts from previous inputs, or a heading has been requested but the message in the current TIOA is too close to TIOADBA to allow the header to be inserted.

5. Message Switching requests BMS routing functions by means of the DFHBMS TYPE=ROUTE macro instruction. The message text is sent via DFHBMS TYPE=TEXTBLD, and completion of the message is indicated by DFHBMS TYPE=PAGEOUT. BMS returns the status of destinations and any error indications in response to the DFHBMS TYPE=CHECK macro instruction.

6. Message Switching interfaces with BMS via DFHBMS TYPE=(EDIT,OUT) and with CICS/VS Terminal Control via DFHTC TYPE=WRITE (for 3270 only) , in providing responses to terminals. These may indicate normal completion, signal that input is to continue, or provide notification of input error.

7. Like any other task, Message Switching has a Task Control Area (TCA) in which values may be placed prior to issuing CICS/VS macro instructions and from which any returned values can be retrieved after an operation. All values for the DFHBMS TYPE=ROUTE macro instruction are placed in the TCA because they are created at execution time. The TWA is used for storing status information (partly saved in Temporary Storage across conversations) and space for work area. The DFHMSP module is coded reentrant.

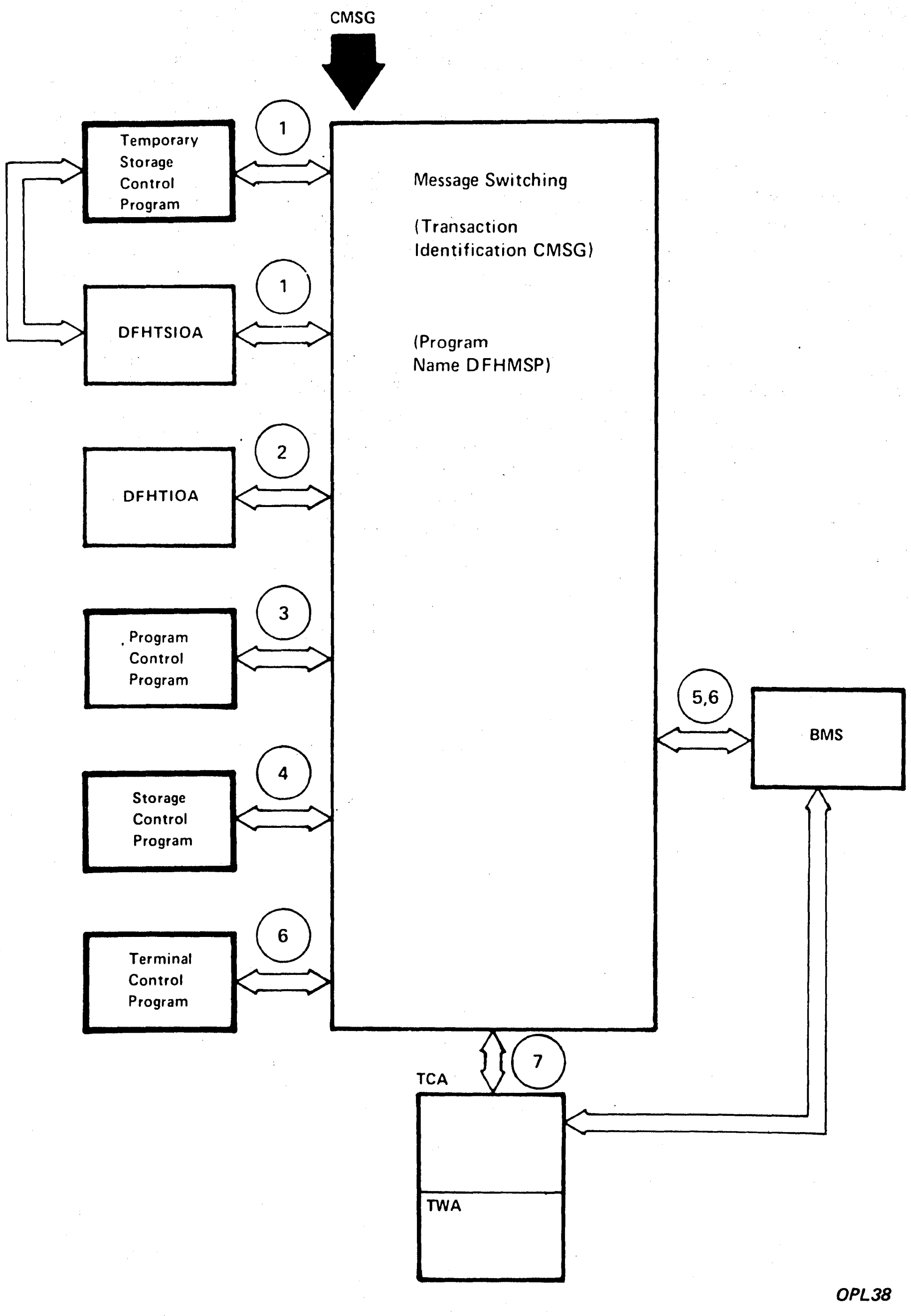


Figure 3-38. Message Switching Interfaces

This page is left blank intentionally

## MASTER TERMINAL PROGRAM (DFHMTP)

The Master Terminal program (DFHMTP) is an optional feature of CICS/VS selected at system initialization. This program consists of seven modules: DFHMTPA, DFHMTPB, DFHMTPC, DFHMTPD, DFHMTPE, DFHMTPF, and DFHMTPG. It is a service program that provides the user with the means of dynamically changing certain system parameters, the status of lines, control units, or terminals.

The Master Terminal program is invoked by operator keying of the proper transaction identification at a master terminal, a supervisory terminal, or a single terminal. The transaction identification may be followed by a series of keywords describing the services to be performed. If the keyword CANCEL is entered anywhere in the original message or subsequent entries, the Master Terminal program is terminated immediately. If, while trying to perform a requested service, the Master Terminal program discovers that insufficient information has been entered, additional information is solicited from the requesting terminal.

The notes below refer to Figure 3-39.

Notes:

1. All transaction input under transaction identification CSMT, CSST, or CSOT is placed in a TIOA accessible to DFHMTPA. The input is scanned to determine whether sufficient keywords are present, to fully specify the requested service. Keyword indicators are set in the requesting task's TWA for all keywords entered. Control is transferred to the DFHMTP module that will provide the requested service. If time or runaway task services are requested, DFHMTPA responds to the request. Otherwise, additional information is requested from the input terminal as needed. DFHMTPA interfaces with Terminal Control for reads and writes and with Storage Control to obtain necessary storage, as do all other DFHMTP modules.

2. DFHMTPB responds to all requests for file, cushion, maximum task, negative poll delay, and trace services. The File Control program (DFHFCP) is used to locate file entries in the File Control Table (FCT).

3. DFHMTPC responds to all requests for terminal services.

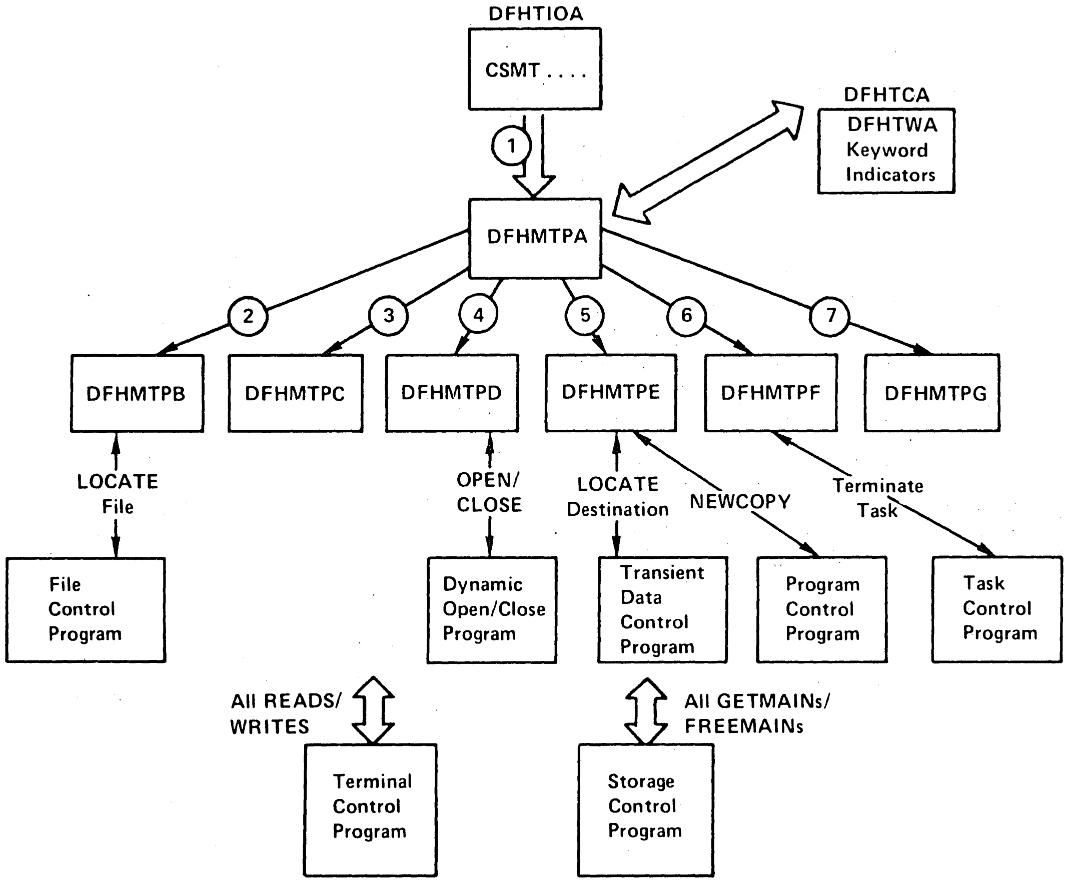
4. DFHMTPD responds to all requests for open, close, and switch of Dump data sets. The Dynamic Open/Close program (DFHOCP) is used to perform the actual file open or close.

5. DFHMTPE responds to all requests for Transient Data destinations, stall, trigger level, and copy services. The Transient Data Control program (DFHTDP) is used to locate destinations in the Destination Control Table (DCT). Program Control (DFHPCP) services are requested by means of a DFHPC TYPE=BLDL macro instruction when responding to a NEWCOPY request.

6. DFHMTPF responds to all requests for line, control unit, terminate task, and task list services. Task Control is used to schedule a task for termination.

7. DFHMTPG responds to all requests for transaction and program services.

8. DFHSTP responds to requests for system shutdown.



OPL39

Figure 3-39. Master Terminal Program (DFHMTP) Interfaces

## DYNAMIC OPEN/CLOSE PROGRAM (DFHOCP)

The Dynamic Open/Close program provides open/close capabilities for Dump data sets, Transient Data extrapartition data sets, and File Control data base data sets. These facilities may be invoked via the CICS/VS Master Terminal program or through the use of DFHOC macro instructions in an Assembler-language application program.

The notes below refer to Figure 3-40.

Notes:

1. The DFHOC macro expansion issues a Program Control DFHPC TYPE=LINK macro instruction to transfer control to the Dynamic Open/Close program. This transfer is transparent to the program issuing the DFHOC macro instruction.

2. Dynamic Open/Close communicates with

   a. Storage Control, in response to all requests for services

   b. Dump Control in response to requests for open, close, or switch of Dump data sets

   c. File Control for open or close of data base data sets and initialization of indices

   d. Transient Data Control and Program Control for open or close of extrapartition data sets

3. Return to the requesting program is achieved by means of a Program Control DFHPC TYPE=RETURN macro instruction, which is transparent to the requesting program.

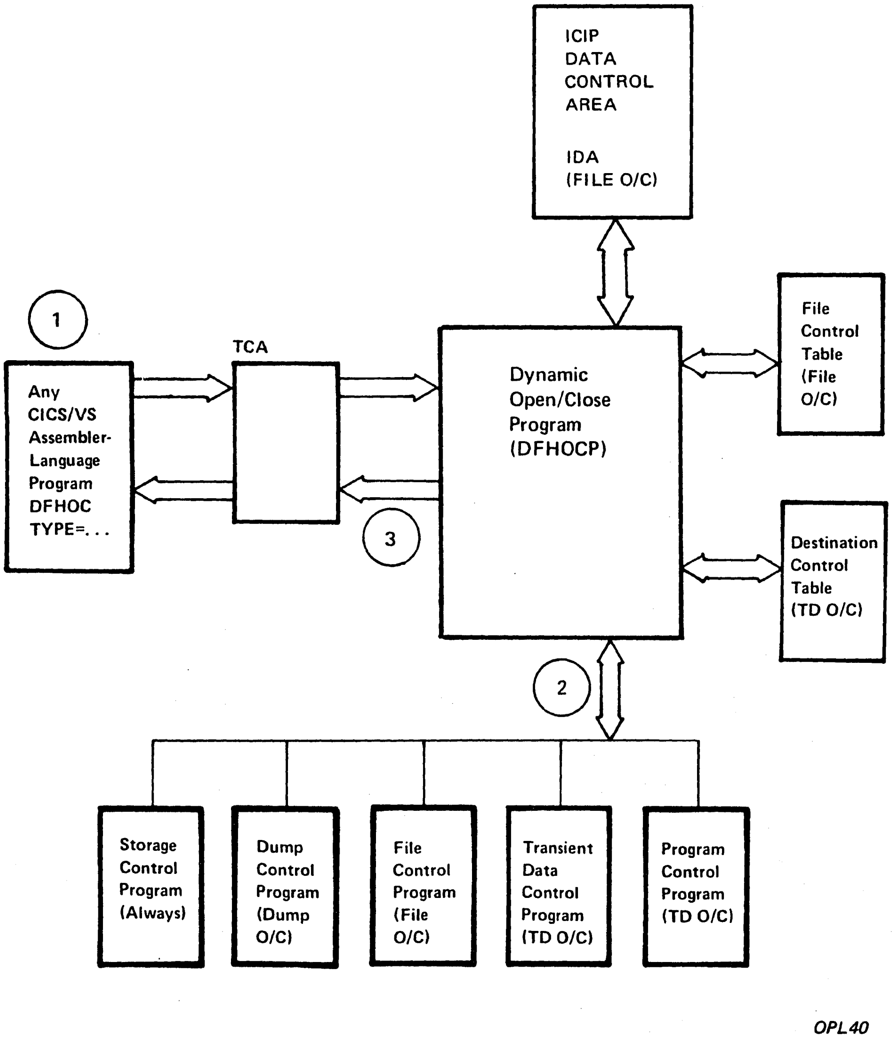

ICIP
DATA
CONTROL
AREA
IDA
(FILE O/C)
1
TCA
Any CICS/VS Assembler-Language Program DFHOC TYPE=...
Dynamic Open/Close Program (DFHOCP)
File Control Table (File O/C)
3
Destination Control Table (TD O/C)
2
Storage Control Program (Always)
Dump Control Program (Dump O/C)
File Control Program (File O/C)
Transient Data Control Program (TD O/C)
Program Control Program (TD O/C)
OPL40

Figure 3-40. Dynamic Open/Close Program Interface

## PROGRAM MANAGEMENT (DFHPCP)

The Program Control program (DFHPCP) may communicate with any CICS/VS function or any user-written application program. In addition, CICS/VS-supported high-level language compilers interface with CICS/VS through Program Control when they use the macro interface to CICS/VS.

The notes below refer to Figure 3-41.

Notes:

1. Program Control (DFHPC) macro instructions establish fields in the TCA to communicate with the Program Control program.

2. ANS COBOL and PL/I modules that use the macro interface establish fields in the TCA and communicate with Program Control en route to other required CICS/VS functions.

3. Any CICS/VS or user-written module may issue a Program Control DFHPC TYPE=ABEND macro instruction to request that a task be abnormally terminated.

4. Control is passed to the calling program or the called program, on the successful completion of a LINK, XCTL, LOAD, SETXIT, or RETURN routine.

   If the called program uses the Command Interface, DFHPCP invokes DFHEIP to initialize the Command Interface environment.

5. Calls from ANS COBOL and PL/I modules using the macro interface may result in an exit to any CICS/VS management function.

6. In the event of an ABEND, control may be passed to a user-specified SETXIT module or subroutine. If, however, there is no SETXIT or the SETXIT routine(s) returns abnormally, the ABEND continues. In this situation DFHACP will be invoked unless Dynamic Transaction Backout is required, in which case DFHDBP will be invoked.

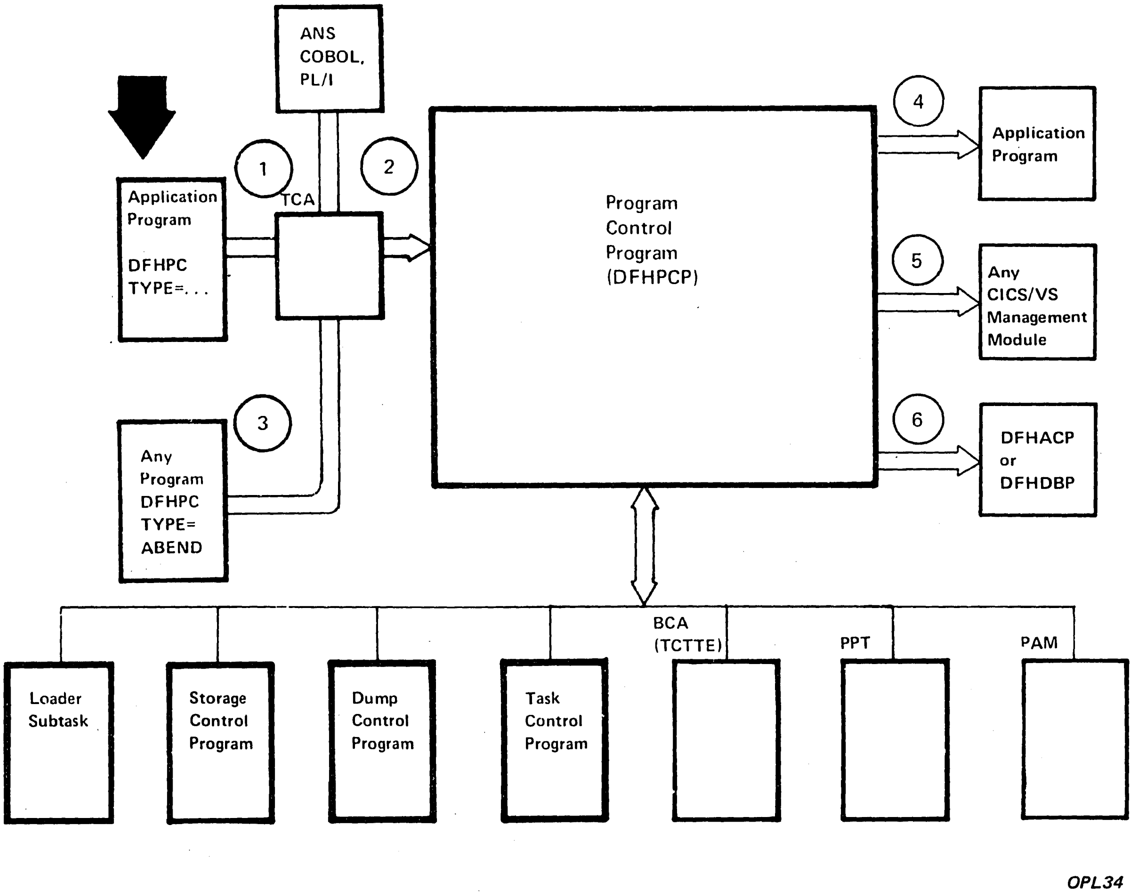


Figure 3-41. Program Control Interfaces

## RECOVERY UTILITY PROGRAM (DFHRUP)

The Recovery Utility program (DFHRUP) is invoked by the System Initialization program (DFHSIP) in the event of an Emergency Restart. The object is to restore the system to the point at which it was when the breakdown occurred. Prior to calling DFHRUP, DFHSIP restarts the PPT, PCT, TCT, FCT, DCT, and CSA. (This is a cold start.) DFHSIP also provides a cold start for Transient Data and Temporary Storage and builds their maps to indicate that all tracks or control intervals are empty.

The notes below refer to Figure 3-42.

Notes:

1. During an Emergency Restart, the System Log is automatically repositioned after the last record written during the previous execution. DFHRUP interfaces with DFHJCP to read this data set backwards in order to process System recovery data and to collect user recovery backout data. The backward scan is completed when the two following conditions are met:

   a. At least one complete Activity Keypoint (delimited by end and start of keypoint records) has been retrieved.

   b. The start-of-task record for all Logical Units of Work (LUWs), in-flight at system termination, has been reached.

2. During the backward scan, DFHRUP uses DFHKPP to output to the Restart data set the following data:

   a. Records output to the System Log by tasks (LUWs) that did not complete processing before the system abnormally terminated (in-flight tasks). These records follow the standard Journal Control Record Layout, they have the flag JCSPRRIF set ON in field JCSPP1, and they are as follows:

      (1) Records automatically logged by the File Control program for data sets with the specification LOG=YES in the FCT.

      (2) Records automatically journaled to the System Log by the File Control program (FCP), according to the user-specified option in the FCT.

      (3) User-journaled records written to the System Log that were output by in-flight tasks.

      Note: User-journaled records with the high-order bit set ON in the JTYPEID which are encountered during the backward scan, are copied over to the Restart data set regardless of the status of the task (in-flight or complete). If the task was completed, the flag JCSPRRIF is OFF in field JCSPF1. User-written activity keypoint records should have an identification as stated above in order to be accessible from the Restart data set.

   b. Initial input and final output message per LUW logged by Terminal Control program for terminals with the PROTECT option group specified in the PCT.

c. All input/output messages for in-flight tasks journaled by the Journal Control program as specified in the MSGJRNL=operand in the PCT.

d. All DL/I records logged to the system log that did not complete processing before the system abnormally terminated.

e. All update/replace records entered on the system log by the Temporary Storage Control program, as specified in the Temporary Storage Table (TST).

3. The following tables are created by DFHRUP and later written to the Restart data set using DFHKPP.

   a. The Transaction Backout Table (TBO) contains an entry for each task in-flight at the time the system abnormally terminated.

      There are two types of entries in the Transaction Backout Control Record:

      (1) In-flight tasks. These are tasks that caused records to be written to the System Log, but failed to complete before system failure. No special start-of-task record is written to the System Log, but the first record logged for the task is flagged as being start-of-task. When DFHRUP reads the log backwards, and the first record found for a task is one <u>other than</u> an end-of-task record, this task is considered in-flight. DFHRUP must then find the corresponding start-of-task indication to complete the collection of recovery backout data for this task.

      (2) Active tasks. These are tasks that completed a LUW and started another, but did not cause any records to be written to the System Log during this LUW. Thus, during DFHRUP processing, a completion of a LUW was found, but no <u>physical</u> end-of-task (that is, task DETACH) was found.

   b. The File Backout Table (FBO) contains an entry for each data set for which a logged or journaled record was written to the Restart data set.

   c. The Message Backout Table (MBO) contains an entry for each terminal for which logged or journaled message or message resynchronization records were written to the Restart data set.

   d. The DL/I Backout Table (DBO) contains an entry for each in-flight transaction that was scheduled to alter a DL/I data base.

4. DFHRUP links to the Transient Data Recovery program to create the Transient Data bit map.

5. DFHRUP links to the Temporary Storage Recovery Program, which restores the Temporary Storage byte map (to reflect the recoverable data records) and reschedules the recoverable ICEs.

6. DFHRUP interfaces with the Transient Data program to write the recovery statistics to the CSSL destination.

7. DFHRUP schedules any terminal dependent AID's for Transient Data destinations whose trigger level has been reached or had been scheduled prior to system abnormal termination.

8. After building all recovery information on the Restart data set, DFHRUP returns control to the System Initialization program which passes control to the Transaction Backout program to perform recovery processing.

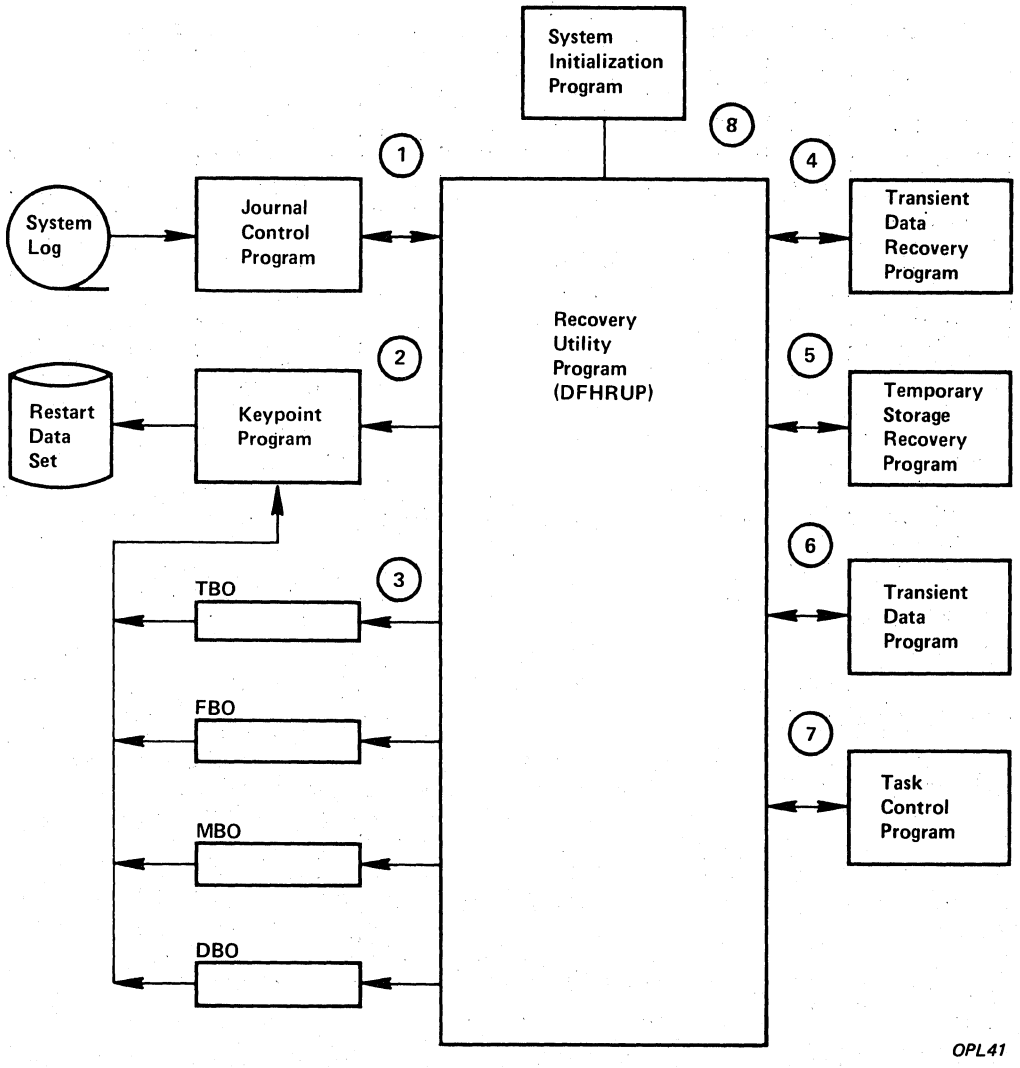


Figure 3-42. Recovery Utility Program Interfaces

## STORAGE MANAGEMENT (DFHSCP)

The Storage Control program (DFHSCP) communicates with other CICS/VS functions and user-written application programs to satisfy their storage requirements. It makes extensive use of CICS/VS control blocks and interfaces with other CICS/VS functions for special processing when exception conditions arise.

The notes below refer to Figure 3-43.

Notes:

1. All CICS/VS and user-written programs communicate their requests for Storage Control services through the TCA. The address of acquired storage is returned to the requesting program in the TCA.

2. Storage Control determines the type of request by referring to the TCA and returns the address of acquired storage in the TCA. Storage associated with a task is chained off the TCA. When working with terminal storage, the address of the TCTTE is found in the TCA. Storage chained off the TCA can be frozen to aid problem determination (for more information refer to the CICS/VS Problem Determination Guide). Requests to free frozen storage are not honored. The storage is only freed when the transaction that acquired it is terminated.

3. Storage Control maintains a chain of terminal storage through a field in the TCTTE. A count of storage violations associated with a terminal is also kept in the TCTTE. Terminal storage chained off the TCTTE can also be frozen, as explained in note 2 above.

4. Storage Control manipulates the short-on-storage (SOS) indicator in the CSA. The CSA is used to find the Page Allocation Map (PAM) and the suspended Dispatch Control Area (DCA) chain. Statistics related to storage control are kept in the CSA.

5. The chain of suspended DCAs is searched by Storage Control when an SOS condition occurs. Suspended requests for storage are retried whenever storage becomes available for allocation.

6. When the system is approaching a SOS condition, Storage Control searches the Processing Program Table (PPT) for programs residing in dynamic storage that are not currently in use. Program storage for programs not in use is freed, and the PPT is changed to reflect the fact that the program is no longer in storage.

7. A count of the number of storage violations associated with a transaction identification is kept in the Program Control Table (PCT).

8. The Page Allocation Map (PAM) contains dynamic values relating to the allocating and freeing of storage. It also contains a map of all pages in the dynamic storage area that indicates the current disposition of each page.

9. If an unconditional request for storage cannot be satisfied, Storage Control issues a DFHKC TYPE=SUSPEND macro instruction, which is a Task Control macro instruction used only by CICS/VS management modules. The requesting task is suspended until the storage request can be satisfied, at which time a DFHKC TYPE=RESUME macro instruction (also used only by CICS/VS management modules) is issued to request Task Control to start the task.

   If an invalid request is issued, or an address specified in a request is invalid, Storage Control issues a Program Control DFHPC TYPE=ABEND macro instruction to request the Program Control program to terminate the task.

10. If the Storage Control program program-checks while attempting to service a DFHSC TYPE=GETMAIN or DFHSC TYPE=FREEMAIN request, the System Recovery program intercepts and passes control to the Storage Control Recovery routine.

11. If Storage Control detects a storage violation while servicing a DFHSC TYPE=GETMAIN or DFHSC TYPE=FREEMAIN request, control is passed to the Storage Control Recovery routine. The Storage Control Recovery routine optionally generates a formatted dump, then fixes the storage problem and returns to Storage Control so that the request that revealed the problem can be retried.

12. Storage items with task-related lifetime are normally put on a chain suspended from TCASCCA, so that they can be freed automatically at task termination or abend. If the task is designated 'primed' in the PCT then these task-related areas are allocated contiguously from within the primed allocation (PRA). PRAs are normal storage areas (CLASS=SHARED) which are allocated, one per primed task, as needed, as tasks of a particular type become active. When a primed task terminates, its PRA is not returned to the common pool but placed instead on a chain of free PRAs hanging off the PCT, to be reused when a new task of the same type is attached. Free PRAs are only returned to the common pool at short-on-storage.

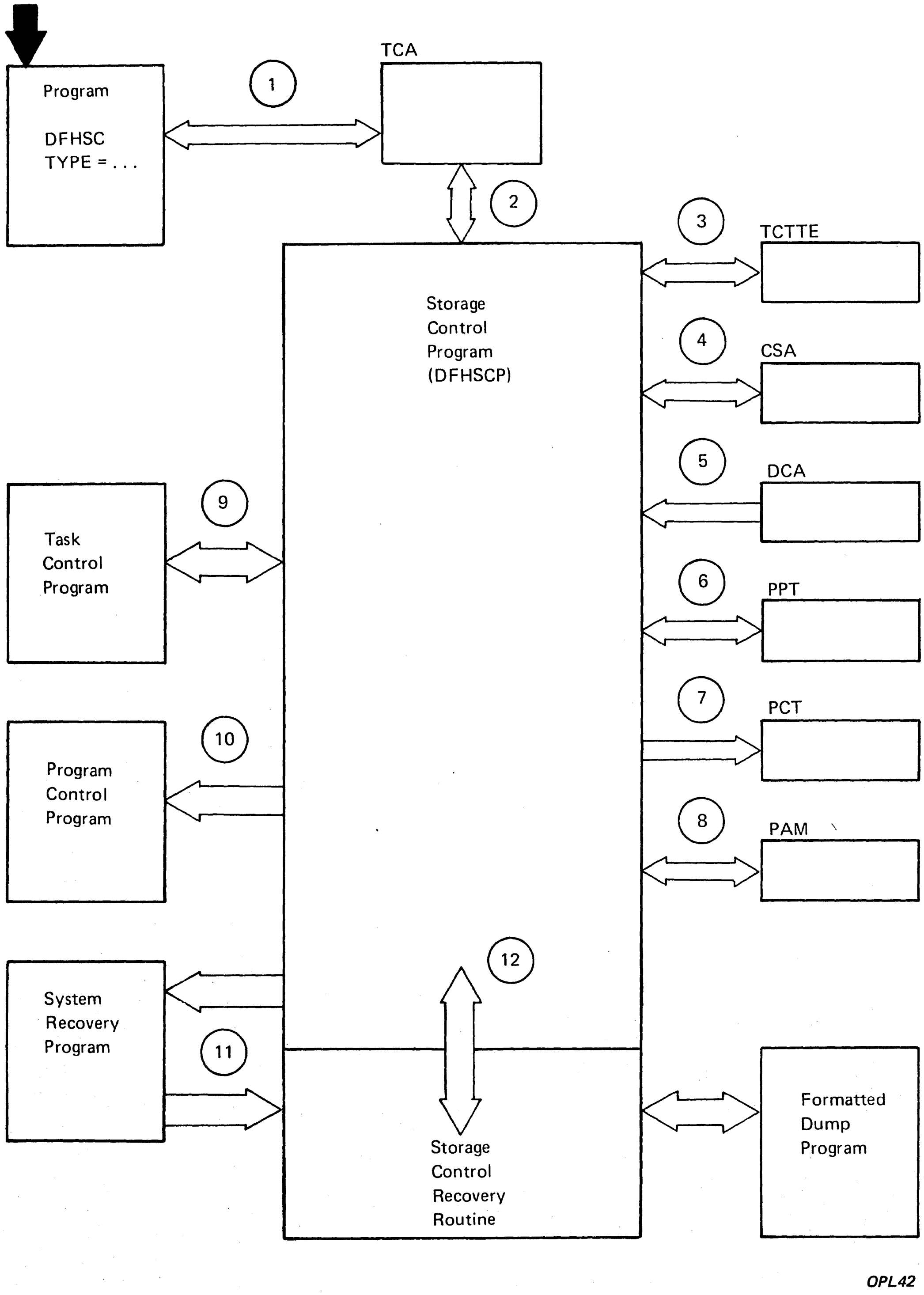


Figure 3-43. Storage Control Interfaces

## SYSTEM INITIALIZATION PROGRAM (DFHSIP)

The System Initialization program (DFHSIP) is a non-real-time component of CICS/VS and is resident only long enough to start up CICS/VS. It is responsible for readying CICS/VS for communication and inquiry by the user.

System initialization provides three classes of restart which are:

COLD
: Complete reinitialization of CICS/VS and system data sets without regard for any previous system activity.

WARM
: This optional restart process reinitializes CICS/VS to the status that existed at the previous system termination. This type of restart assumes that the previous termination was normal, that the system was quiesced prior to termination, and that a warm keypoint was taken during that termination of CICS/VS.

EMERGENCY
: This optional restart process restores the system using information recorded during the previous execution of the system to a predefined point which existed prior to the interruption.

Selection of startup options can be made in the System Initialization Table specification (COLD or WARM) or in the override parameters (COLD, WARM, or EMER).

### SYSTEM INITIALIZATION OVERLAYS

User-written overlays can be added to the System Initialization program; however, they must conform to CICS/VS naming conventions. All system initialization overlays are seven-character names in the format, DFHSIxy, where x is a letter from A to Z and y is a number from 1 to 9. CICS/VS development reserves suffixes that end in 1 (A1, B1,...Z1). User overlays may use any other two-character suffix.

Overlay processing in system initialization is driven from the System Initialization Table SIMODS=parameter. User overlays can be inserted at any point in system initialization, but the sequence of CICS/VS overlays must not be disturbed. CICS/VS is responsible for common subroutine and overlay linkage (assuming that these routines are not modified), and normal system initialization functions.

A list of System Initialization subroutines, and the conventions for calling them, is in the CICS/VS System Programmer's Reference Manual. The following areas are always addressable to System Initialization overlays at entry and must be addressable at exit:

| Register | Area |
|---|---|
| 13 | Common System Area (CSA) |
| 5 (SIPPLAR) | System Initialization Table (SIT) |
| 4 (SIPBAR1) | System Initialization Common Area (SIPCOM)<br>System Initialization Common Routines |

In addition, at entry to an overlay, registers 3 (SIPBAR) and 9 (SIPBAR2) contain the entry point address of the overlay and that address plus 4096, respectively, for addressability purposes.

The following fullword fields are supplied as parameter-passing fields between user overlays of System Initialization. These fields are not used by CICS/VS overlays.

SIPARMP6
SIPARMP7

The notes below refer to Figure 3-44.

Notes:

1. System Initialization receives control from the OS/VS Initiator. Parameters may be passed to System Initialization through the PARM operand of the EXEC statement by which it is invoked. (These parameters are documented in the CICS/VS System Programmer's Reference Manual.)

2. The Restart Data Set is a DAM file, used by the Keypoint program (DFHKPP) to save certain system information at system termination time so that a warm start or emergency restart can be initiated later. System Initialization can warm start the following CICS/VS control information:

   Program Control Table (PCT)
   Processing Program Table (PPT)
   Terminal Entries (nonswitched)
   File Control Table (FCT)
   Selected areas from the Common System Area (CSA)
   Destination Control Table (DCT) Intrapartition Entries
   Transient Data Intrapartition space allocation bit map
   Identifications and Relative Byte Addresses for Temporary Storage auxiliary destinations/queues
   Temporary Storage space allocation bit map
   Interval Control Elements (ICE) and Automatic Initiate Descriptors (AID)
   Batch Control Areas (BCA) and Write Request Elements (WRE) for ATP

3. When emergency restart is invoked via the START=EMER keyword, the System Initialization program will take the following action:

   a. Reposition the system log.

   b. COLD start the PPT, PCT, TCT, FCT, DCT, CSA, Transient data bit map, and the Temporary Storage maps.

   c. Link to the Recovery Utility program which reads the system log and builds recovery data and tables which are written to the Restart data set.

   d. Link to the Transaction Backout program which reads the recovery data and backs out the effects of transactions in process prior to system termination.

4. The System Initialization program does the following:

   a. Builds the CICS/VS nucleus (for High-Performance Option (HPO) certain modules are loaded in protected storage and control blocks required for Service Request Block (SRB) processing are constructed in the System Queue Area (SQA). Authorization is relinquished after this has been completed).

   b. Initializes data sets

   c. Opens system and user data sets (for ICIP data sets page fixed buffers are allocated and in-core indexes are constructed if specified in the File Control Table (FCT)).

   d. Constructs and initializes tables.

   e. Builds the CICS/VS dynamic storage pool.

5. The CICS/VS Program Library is accessed by means of BSAM READs to build the CICS/VS nucleus, load tables, load resident application programs, and initialize the Processing Program Table (PPT).

6. Interface to CICS/VS nucleus modules is required during post-initialization processing, both by System Initialization and by application programs running at this time. System Initialization always interfaces with Storage Control, Task Control, Interval Control, and Program Control and may interface with Temporary Storage Control, Transient Data Control, File Control, and System Recovery. All interface to CICS/VS nucleus modules is done under Terminal Control's Task Control Area (TCA), which is "borrowed" temporarily as a communication vehicle.

7. System Initialization first passes control to the dummy CSA module via an OS/VS XCTL. This frees the storage area occupied by System Initialization for use by OS/VS. The dummy CSA module branches to the CICS/VS Terminal Control program, which begins polling.

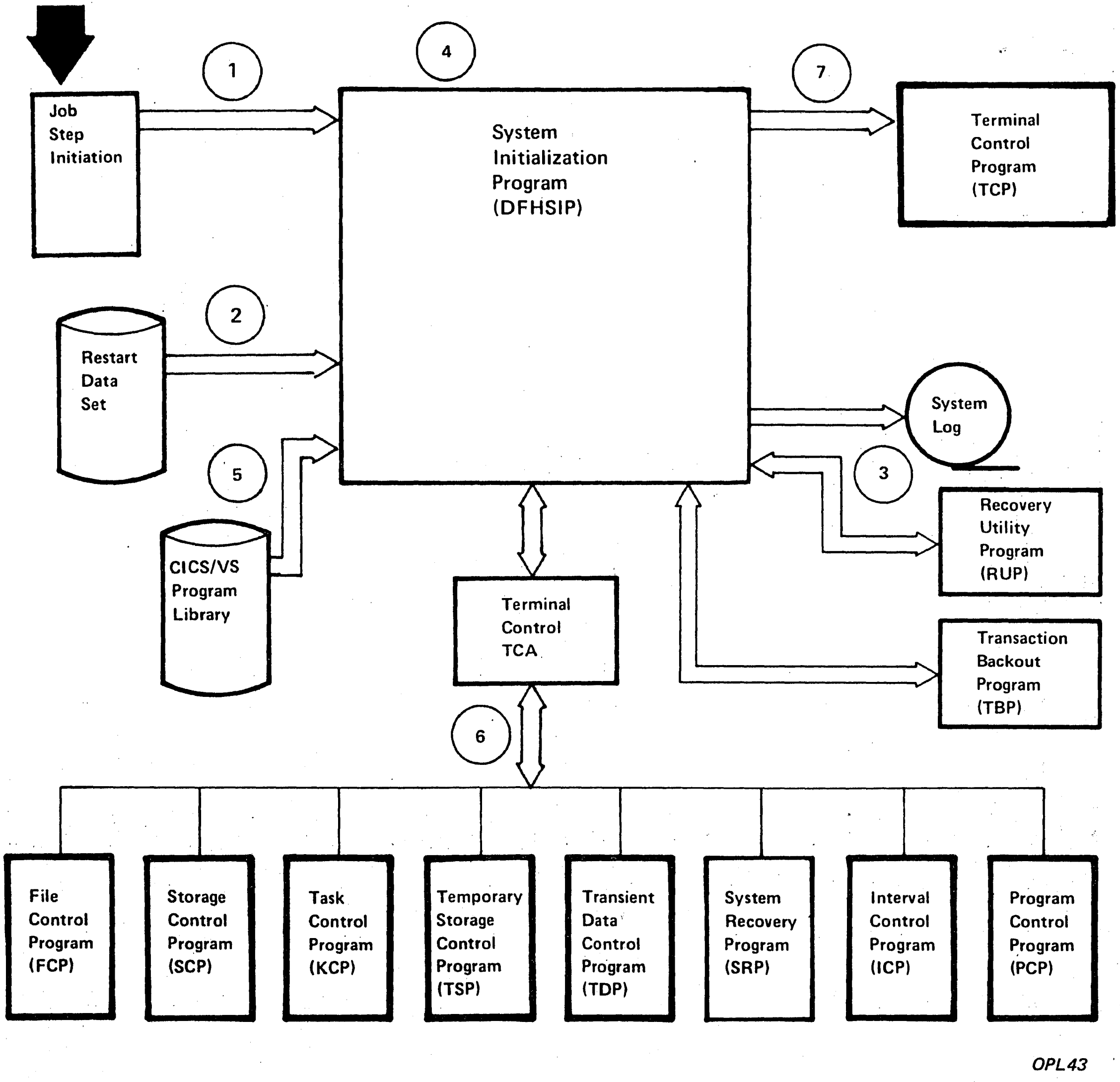


Figure 3-44. System Initialization Interfaces

## SYNC POINT PROGRAM (DFHSPP)

The Sync Point program (DFHSPP) is one of the modules within CICS/OS/VS enabling the user to participate in recovery/restart processing. The recovery/restart features are available as an option to CICS/OS/VS users. DFHSPP is invoked by the Task Control program (DFHKCP) whenever a task is detached (provided DFHSPP has some work to do — DFHKCP checks first). It can be invoked by an application program. Generally, DFHSPP services need be invoked by application programs only when tasks are long-running tasks. In such a case, it is used to divide the task into shorter units, referred to as logical units of work (LUWs), which better fit recovery requirements.

Deferred Work Elements (DWEs) are created by CICS/VS management modules, chained off the task's TCA through the TCADWLBA field, and represent deferred processing to be done upon completion of a logical unit of work or changes to be backed out if the LUW is abandoned by the user. The module that creates a DWE can insert an entry address of a DWE processor in that DWE. Control is passed to this DWE processor at the end of the task or LUW by the Sync Point program.

Three types of DWEs exist:

- work is to be done (or previous work backed out)
- data is to be logged
- some work is to be done and data is to be logged

A DWE processor pointed to by a work-only DWE can create another DWE, indicating that logging is required.

The Sync Point program examines the DWE chain in the following way:

- If DFHPC TYPE=ROLLBACK was specified, DFHDBP is invoked to scan the DWE chain and dynamic log, and to back out changes as required.
- Scan the chain and pass control to a DWE processor if work is to be done only.
- Scan the chain and log data as required.
- Scan the chain and pass control to the DWE processor required for a DWE requiring both work and logging (the logging was completed on the previous scan).

A DWE indicating both work and logging to be done can be looked upon as saying:

> The data that is being written to the System Log represents the intention of the DWE processor. If the system terminates abnormally before the DWE processor has finished its work, the System Log tells CICS/VS modules involved in recovery what to do during emergency restart.

The notes below refer to Figure 3-45.

Notes:

1. DFHSPP is entered from either the Task Control program or an application program.
2. For a rollback request, DWEs and dynamically logged records from the current LUW are dealt with by the dynamic backout program (DFHDBP).
3. The DWE chain is scanned, and control is passed to DWE processors. Upon return from a DWE processor, the DWE is freed.
4. The DWE chain is scanned, and data is logged via Journal Control.
5. An end-of-task record is written to the System Log via Journal Control.
6. Scan intersystem TCTTEs issuing 'REQUEST COMMIT' and 'COMMIT' commands (via DFHTC CTYPE=PREPARE and CTYPE=SPR respectively) to synchronize associated work on remote systems.
7. The DWE chain is scanned, and control is passed to DWE processors. Upon return from a DWE processor, the DWE is freed.
8. All resources enqueued upon through the Task Control DFHKC TYPE=ENQ facility are dequeued.
9. Reinitialize the dynamic log ready for the next logical unit of work.

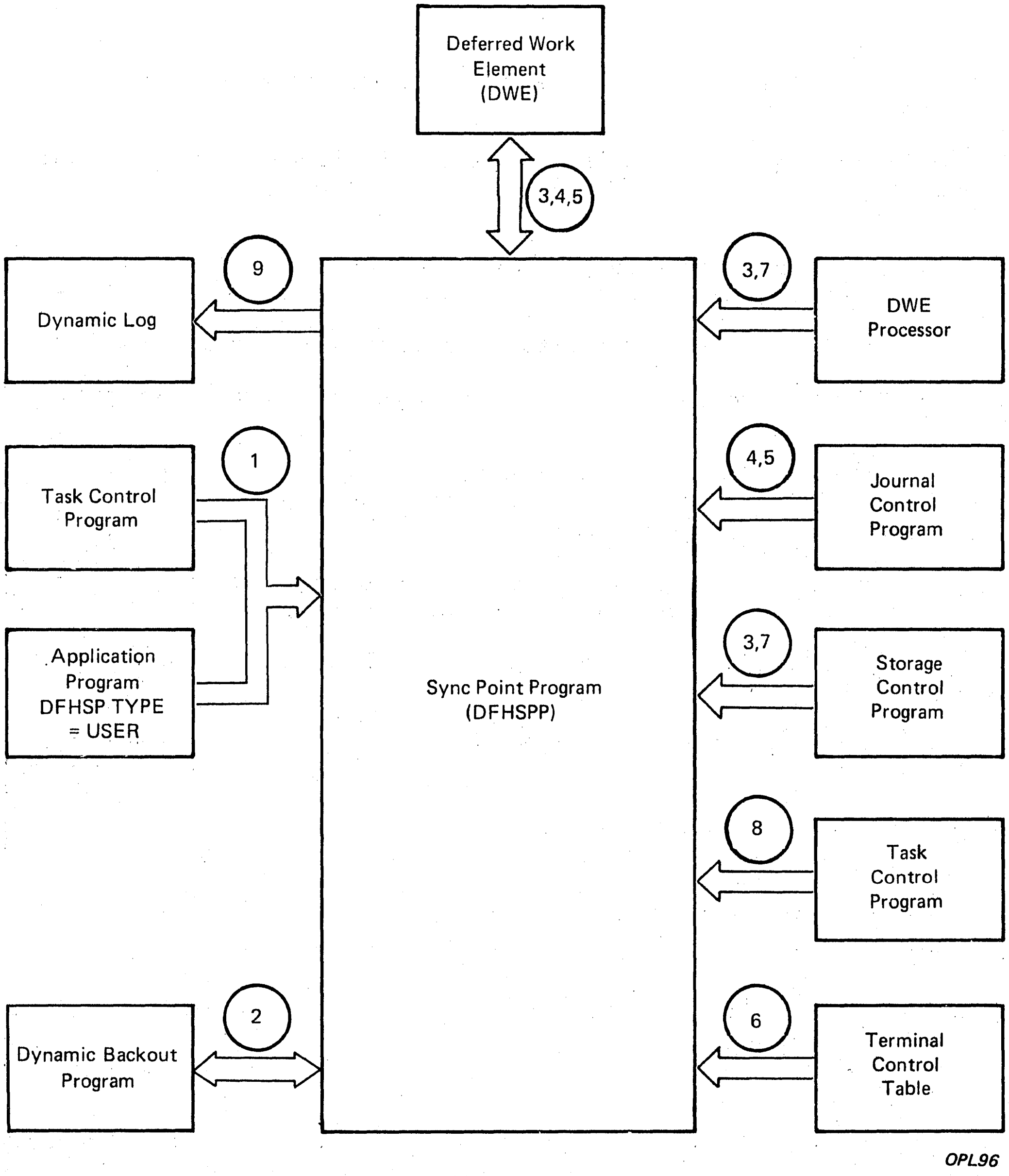


Figure 3-45. Sync Point Program Interfaces

This page is left blank intentionally

## SYSTEM RECOVERY MANAGEMENT (DFHSRP)

The System Recovery program (DFHSRP) is a generalized abnormal termination handler that receives control from the operating system when a program check or ABEND condition is recognized. It provides program interrupt logic for the capture and recovery of program-check interrupts and system ABENDs. A program check interrupt normally causes DFHSRP to invoke a DFHPC ABEND of the task in which it arose. A system ABEND condition may be handled by user code or IBM-supplied code, after which CICS/VS will attempt to continue to run.

The notes below refer to Figure 3-46.

Notes:

1. The System Recovery Program is given control as a SPIE exit.

2. If the System Recovery program determines that a program check or ABEND condition has occurred during an error condition, or that the current task is a system task, it issues a message, forces a dump, then terminates CICS/VS. Otherwise, it returns control to DFHPCP, via the operating system, to terminate one task with abend code ASRA.

3. Invocation of the System Recovery program as a STAE exit (OS/VS1 and OS/VS2 Release 1) or an ESTAE exit (OS/VS2 Release 2 onwards) causes it to search the System Recovery Table for an entry containing the actual system AB code.

4. If a match is found, the routine specified is invoked or the named program LINKed to. The routine may be that supplied by IBM as part of DFHSRT.

5. Following the recovery logic, a DFHPC TYPE=ABEND macro instruction is issued to abnormally terminate the offending task. If recovery has been achieved successfully, execution of CICS/VS can continue. If a recovery cannot be effected, or if no recovery was attempted, a keypoint is taken (optionally) and CICS/VS is terminated.

   For Systems with High-Performance Option (HPO) the Service Request Block (SRB) in the System Queue Area (SQA) is freed by using a CICS SVC (DFHCSVC).

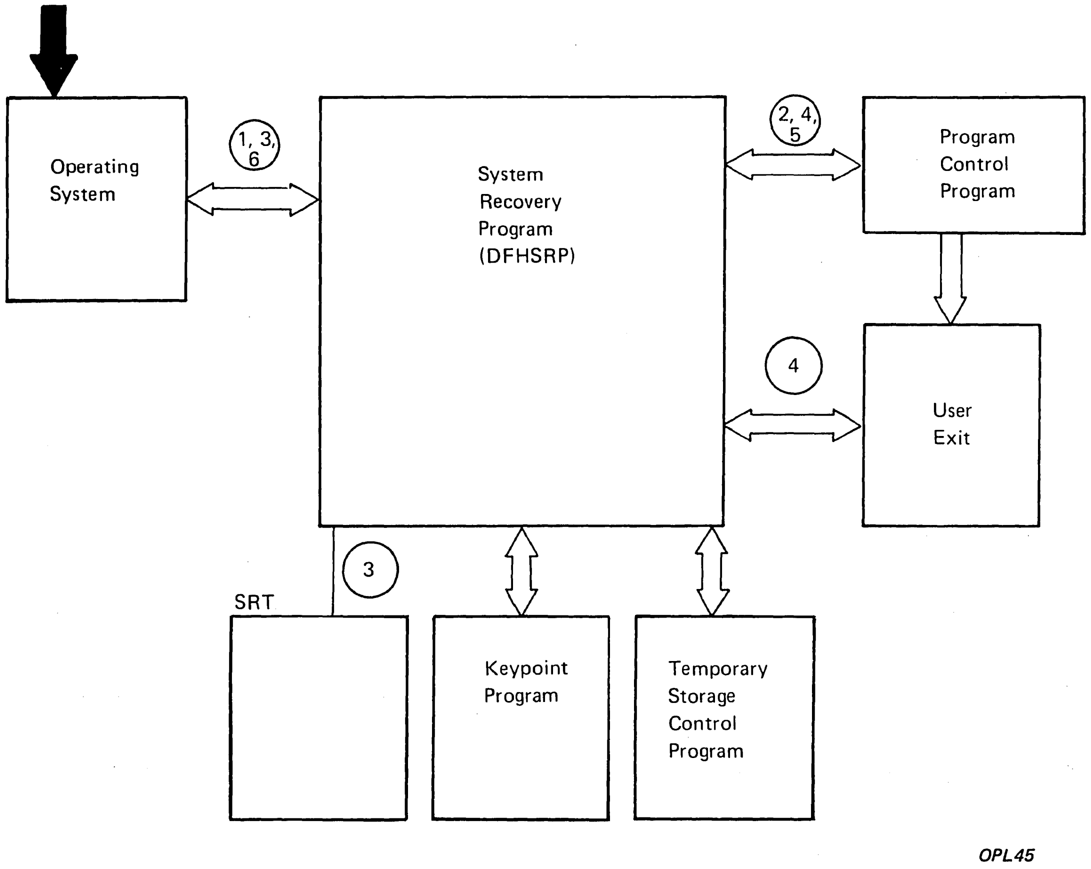


Figure 3–46. System Recovery Interfaces

## SYSTEM TERMINATION PROGRAM (DFHSTP)

The purpose of System Termination (DFHSTP) is to provide for an orderly shutdown of CICS/VS. It is activated by the Master Terminal program (DFHMTPA) when that program is responding to a shutdown request entered by the CICS/VS master terminal operator.

The notes below refer to Figure 3-47.

Notes:

1. The Master Terminal program (DFHMTPA) transfers control to DFHSTP by means of a Program Control DFHPC TYPE=XCTL macro instruction when a CSMT SHUTDOWN request has been entered by the CICS/VS master terminal operator.

2. The Transaction List Table (XLT) and Program List Table (PLT) are loaded via Program Control from the CICS/VS Program Library (DFHRPL).

3. Terminal activity is quiesced via an indicator in the CSA. This tells Terminal Control not to attach any transactions other than those specified in the XLT. The termination task logically disconnects itself from the physical terminal to allow other activity on that terminal.

4. The termination task allows all other tasks (except any journal tasks) to complete before linking to the first program specified in the first portion of the PLT.

5. When all programs in the first portion of the PLT have executed, terminal activity is quiesced completely via an indicator in the CSA. The ICE, AID, and BCA chains are broken (addresses saved in the TWA), and the programs specified in the second portion of the PLT are executed.

6. CICS/VS-DL/I Interface and Journal Control (subtasks) are terminated; Temporary Storage Control (DFHTSP) is requested to output its buffer; statistics are taken by means of a link to the System Statistics program (DFHSTKC); and a keypoint is taken by the Keypoint program (DFHKPP).

7. Control is returned to the operating system, with or without a dump (depending upon the parameters specified in the shutdown request causing termination).

   For Systems with High-Performance Option (HPO) the Service Request Block (SRB) in the System Queue Area (SQA) is freed by using a CICS SVC (DFHCSVC).

   If an immediate shutdown is requested, no load of tables, terminal quiescing, or execution or programs specified in the PLT (see notes 2, 3, 4, and 5) occurs.

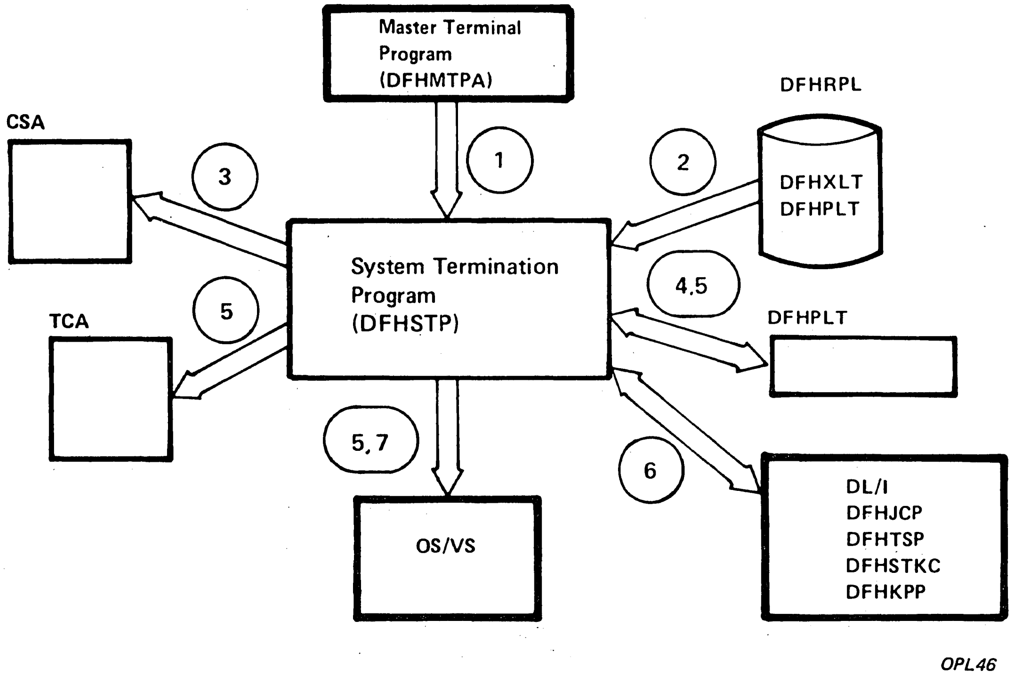


Figure 3-47. System Termination Interfaces

## TERMINAL MANAGEMENT (DFHTCP, DFHZCP)

The Terminal Management function consists of seven CICS control modules, DFHTCP, DFHZCP, DFHZCX, DFHZCA, DFHZCB, DFHZCY, and DFHZCZ. DFHZCP and DFHZCX provide the common functions (VTAM and non-VTAM) while DFHZCA, DFHZCB, DFHZCY, and DFHZCZ provide VTAM only support. DFHTCP provides the non-VTAM support. The Terminal Management function communicates with user-written application programs, CICS/VS System Management functions (Task Control, Storage Control), CICS/VS Application Services (Basic Mapping Support), System Reliability functions (abnormal condition handling), and with operating system access methods (BTAM, GAM, SAM, or VTAM). Requests for Terminal Management functions made by application programs, BMS, or Task Control, are processed through the common interface of DFHZCP. Generally Terminal Management requests for other CICS/VS or operating system functions are issued by any one of the six DFHZC modules (VTAM support) or by DFHTCP, depending upon the terminal being serviced.

The notes below refer to Figure 3-48.

Notes:

### Common Interface

1. When a Terminal Management (DFHTC) macro instruction is issued by an application program or by the Basic Mapping Support program (BMS), request bits are set in the user's TCA (Transaction Control Area) and control is passed to the common interface (VTAM, non-VTAM) routines of DFHZCP.

2. If the DFHTC macro instruction includes a WAIT request and the IMMED option is not in effect, control is passed to Task Control to place the requesting program (task) in a suspended state. If a WAIT request is not included, control is returned to the requesting task.

3. The task's TCA contains either a field named TCAFCAAA (Facility Control Area Associated Address) a pointer to the terminal with which the task is associated, or a field named TCATPTA in which the address of the TCTTE to be used may be passed.

4. Task Control dispatches Terminal Management through the common interface for one of the following reasons:

   a. The system partition/region exit time interval (specified in the ICV operand of the DFHSIT macro instruction by which the System Initialization Table is generated) has elapsed.

   b. The Terminal Management event initiated by the DFHTC macro instruction has been posted complete (non-VTAM ECB posted or exit scheduled in the case of VTAM).

   c. Neither a nor b, above, has occurred, but one second has elapsed since the last time Terminal Management was dispatched with a pending request to be serviced.

5. Terminal Management through its common interface, requests Task Control to perform a CICS/VS WAIT when Terminal Management has processed through the terminal network and has no further work that it can do.

## Access Method Dependent Interface

6. Terminal Management communicates with Storage Management by means of DFHSC TYPE=GETMAIN or FREEMAIN macro instructions to obtain and release storage as follows:

   Non-VTAM

   DFHTCP issues DFHSC macro instructions to obtain and release terminal and line storage.

   VTAM

   Any one of the six DFHZC modules issues DFHSC macro instructions to obtain and release terminal, line (line class is used for the Receive-Any I/O areas), and RPL storage.

7. Terminal Management communicates with Task Management by means of the DFHKC macro instruction. The macro instruction is issued by any one of the seven CICS control modules, depending upon the terminal being serviced. Terminal Management may request Task Management to perform one of the following:

   a. attach a task upon receipt of a transaction identification from a terminal

   b. respond to a DFHKC TYPE=AVAIL request (a Task Control macro instruction documented only for system programming) when a time-initiated task is indicated for a terminal and that facility is available.

8. Terminal Control communicates with operating system access methods in either of the following ways, depending upon the terminal being serviced:

   Non-VTAM (DFHTCP)

   DFHTCP builds access method requests in the DECB, which is part of the TCTLE. The DECB portion of the TCTLE is passed to the access method by Terminal Management to request a service of that access method. The access method notifies Terminal Management of the completion of the service through the DECB. Terminal Management analyzes the contents of the DECB upon completion to determine the type of completion and to check for error information.

VTAM (DFHZCP)

DFHZCP builds VTAM request information in the RPL which is then passed to VTAM for servicing. VTAM notifies Terminal Management of completion by placing completion information in the RPL. DFHZCP analyzes the contents of the RPL upon completion to determine the type of completion and the presence of error information. Communication with VTAM also occurs by VTAM scheduling exits, for example, LOGON or LOSTERM. VTAM passes parameter lists and does not always use the RPL.

When authorized path VTAM has been generated, communication with VTAM also occurs in SRB mode (using DFHZHPRX); DFHZCP uses the RPL with an extension to communicate with its SRB mode code. When an SRB mode RPL request is complete, DFHZCP will call the relevant exit of post the ECB, as indicated by the RPL extension.

9. Terminal Management communicates with the CICS/VS abnormal condition functions in either of the following ways, depending upon the terminal being serviced:

Non-VTAM

DFHTCP attaches the Terminal Abnormal Condition program (TACP) and passes a Terminal Abnormal Condition line entry (TACLE) when an error occurs. The TACLE is a copy of the DECB portion of the TCTLE and contains all information necessary for proper evaluation of the error, plus special action indicators that can be manipulated to alter the error correction procedure. After the DECB has been analyzed, it is passed to the user's error recovery program (DFHTEP).

VTAM

DFHZCA attaches the Node Abnormal Condition Program (NACP) when an error occurs. DFHNACP does some preliminary error processing and then passes control to the user's Node Error Program (DFHNEP). Upon the completion of the user's error processing, control is returned to DFHNACP.

10. Terminal Management executes either under the user's TCA or its own TCA as follows:

User's TCA

a. During the application program interface

b. During the interface with Basic Mapping Support

c. While performing non-chained VTAM terminal requests

Terminal Control's TCA

a. When Task Control dispatches Terminal Control

b. When Terminal Control issues a request to Task Control (DFHKC)

c. When Terminal Control issues a request to Storage Control (DFHSC)

d. While performing non-VTAM terminal I/O or chained VTAM terminal I/O

Since many devices are supported by CICS/VS Terminal Management, the number of modules required to provide this support is significant. Figure 3-49 through 3-53 give an overview of the interrelationships and functions within Terminal Management.

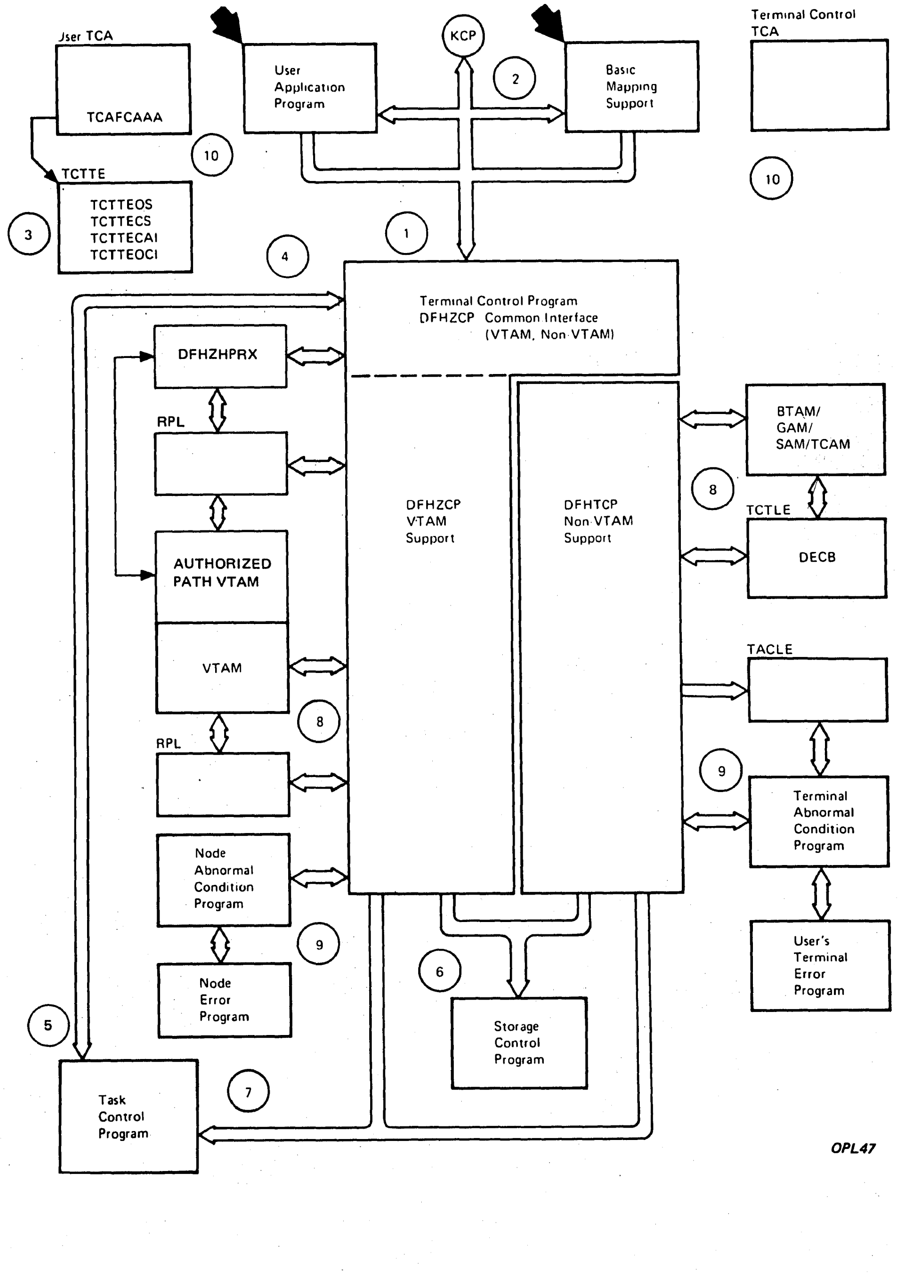


Figure 3-48. Terminal Management Interfaces

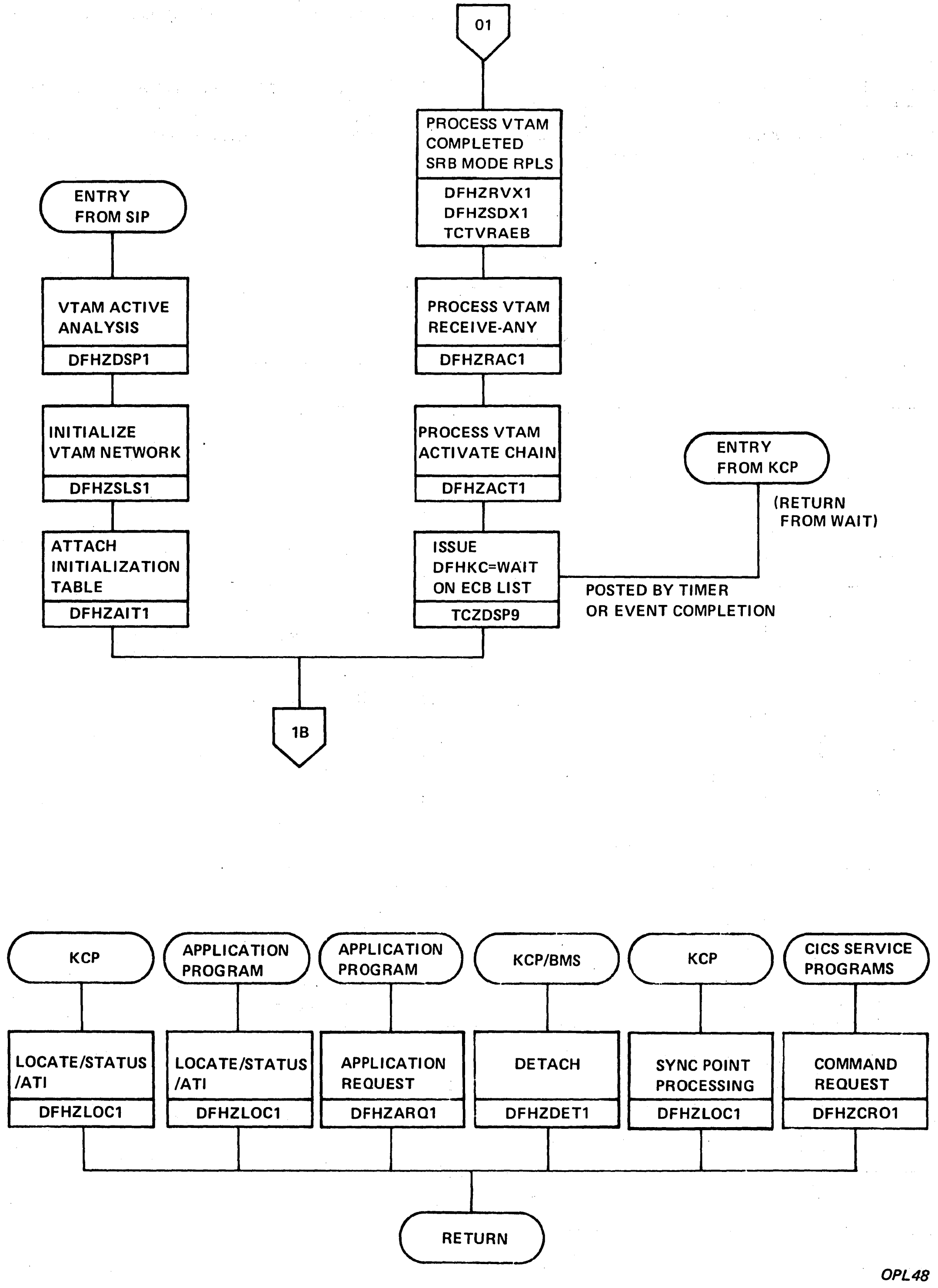


Figure 3-49. (Part 1 of 2). Terminal Management Common Control Routine

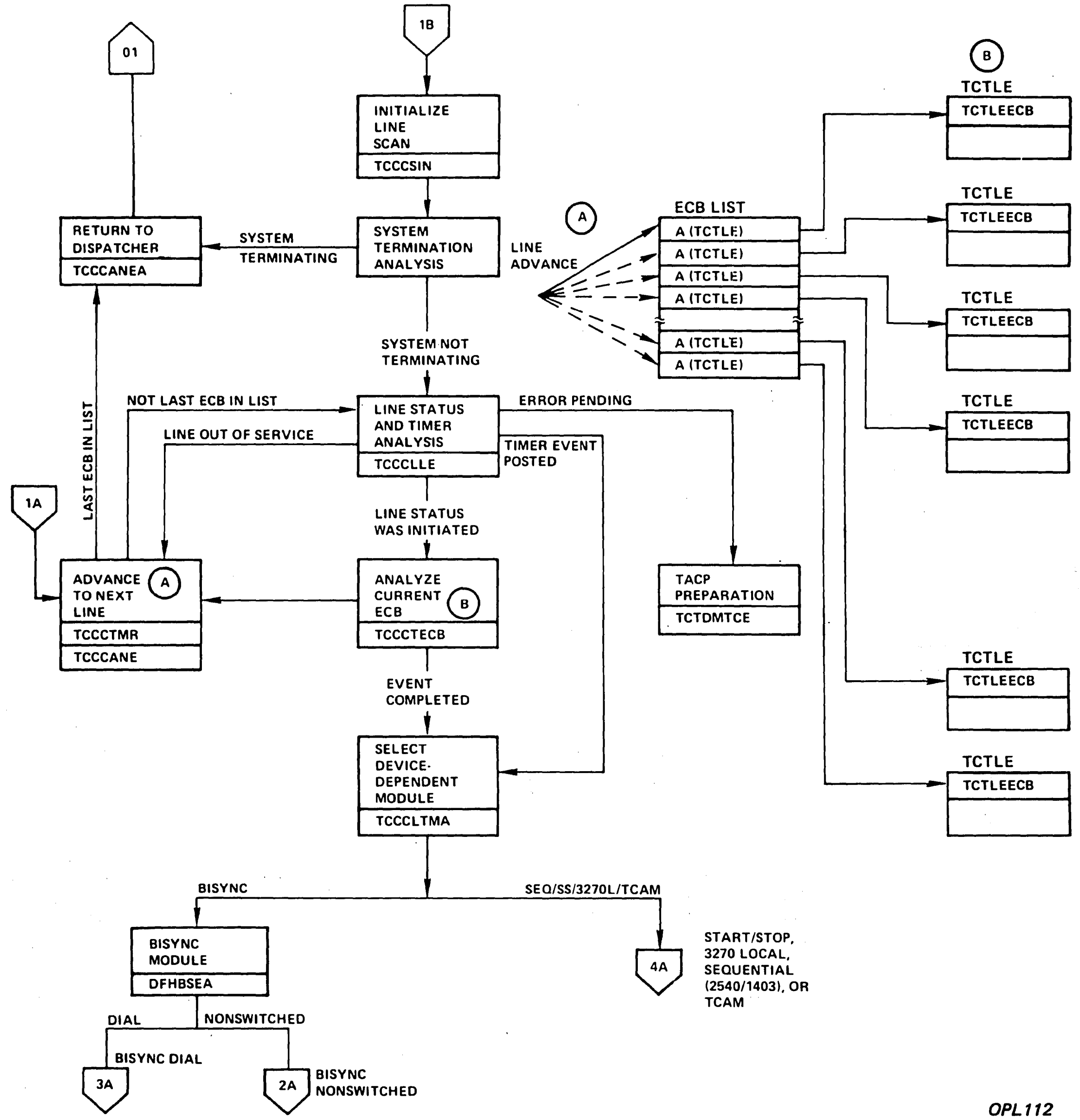


Figure 3-49. (Part 2 of 2). Terminal Management Common Control Routine

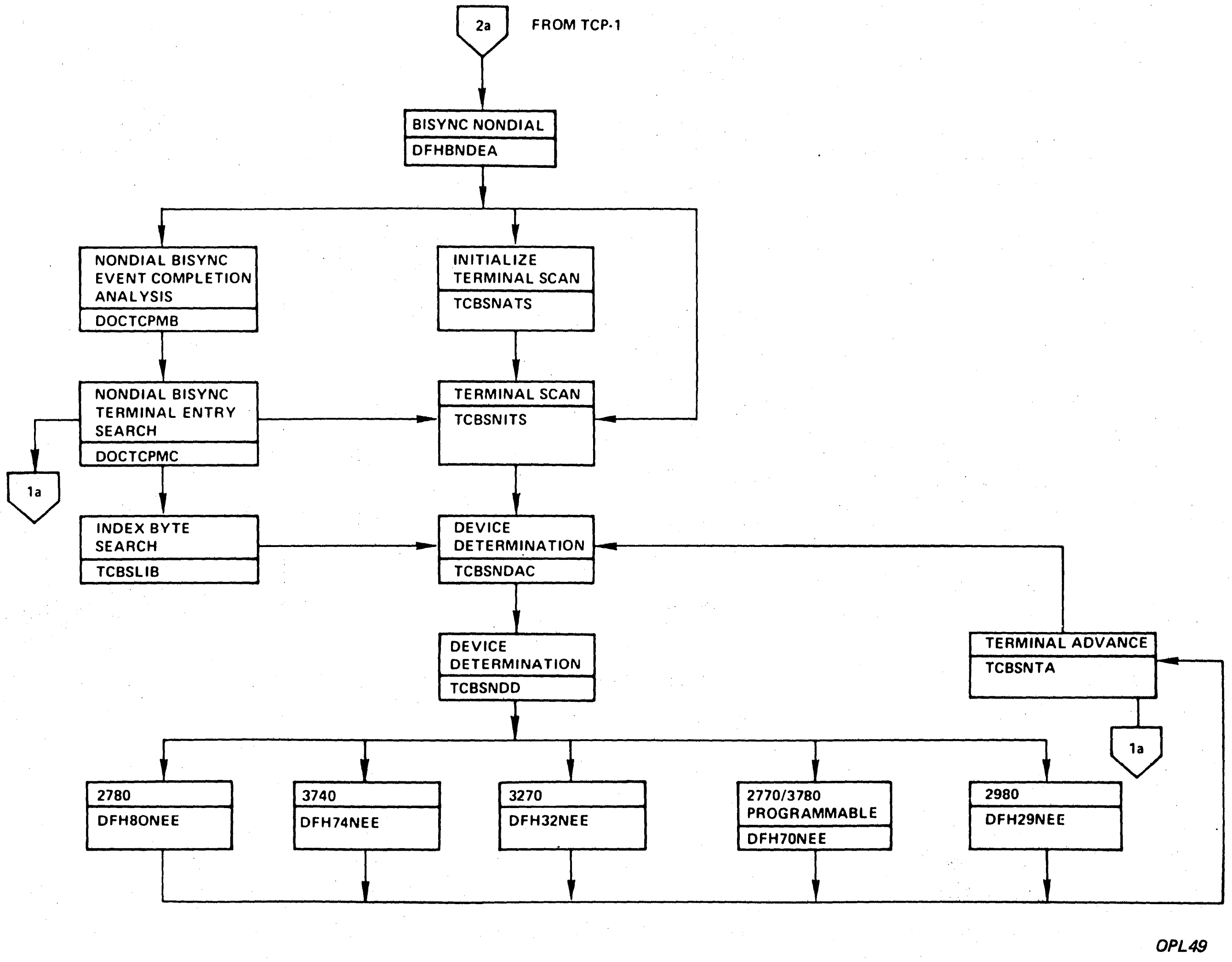


Figure 3-50. Terminal Management Bisync Nonswitched Routine

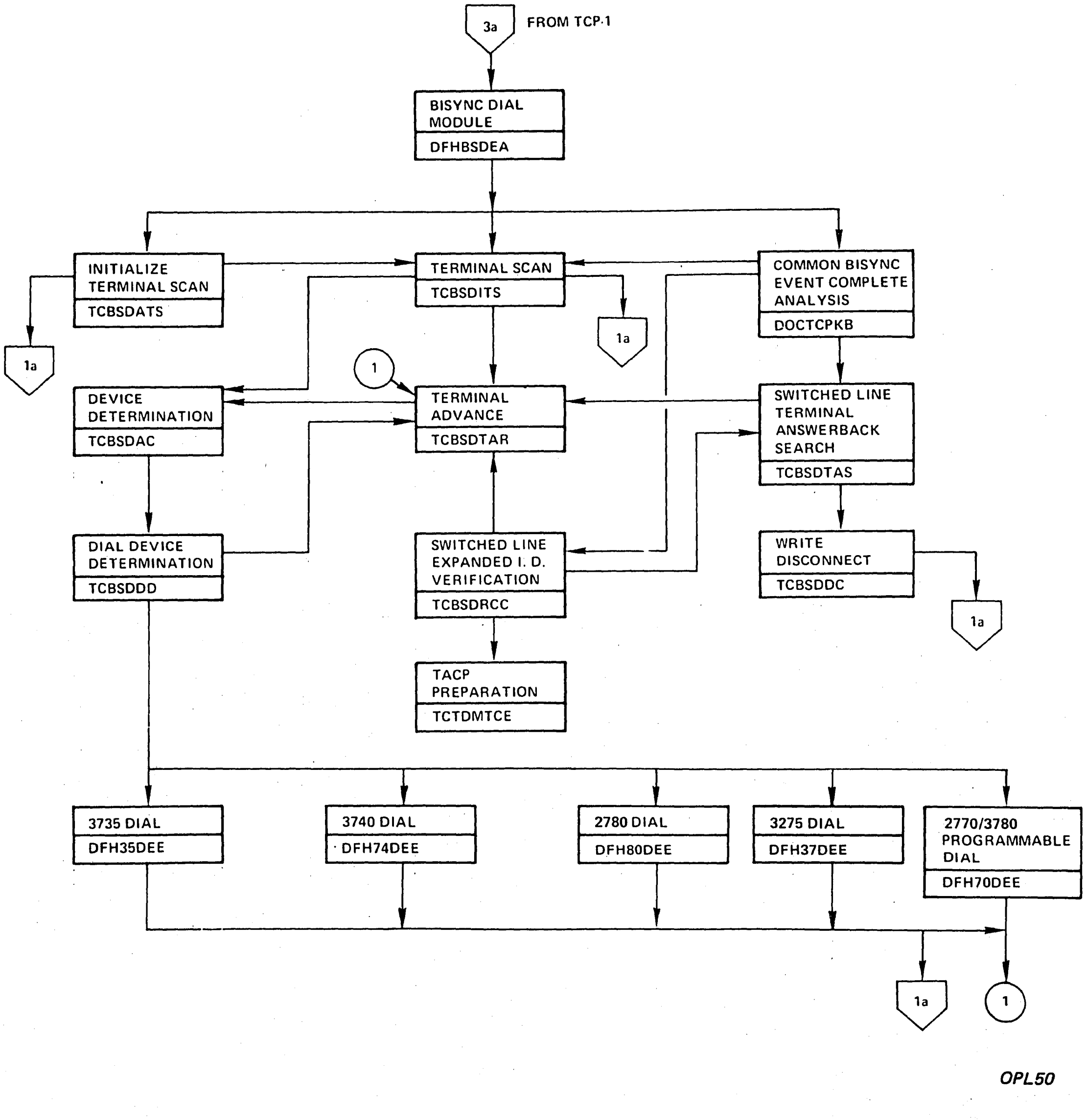


Figure 3–51. Terminal Management Bisync Dial Routine

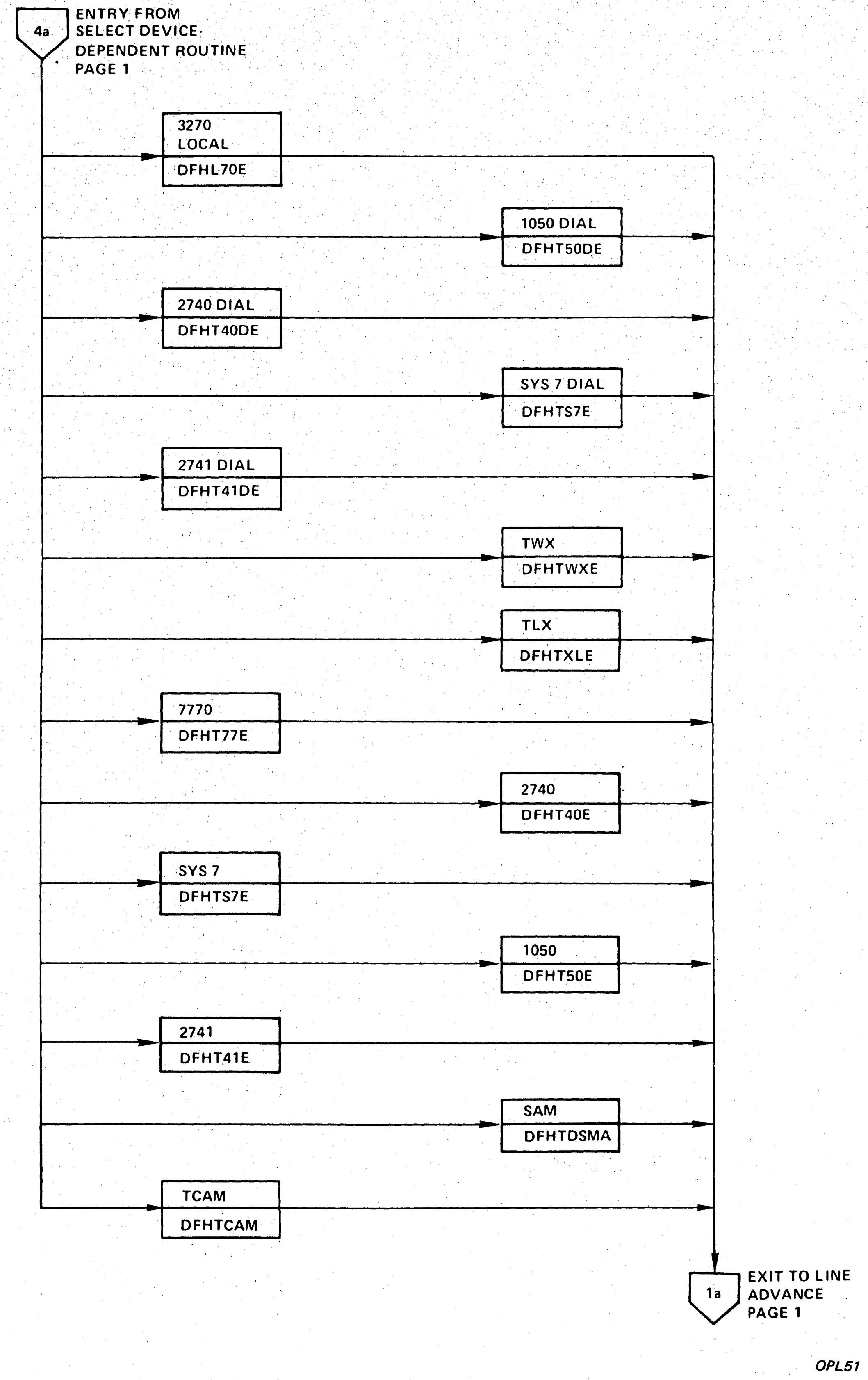
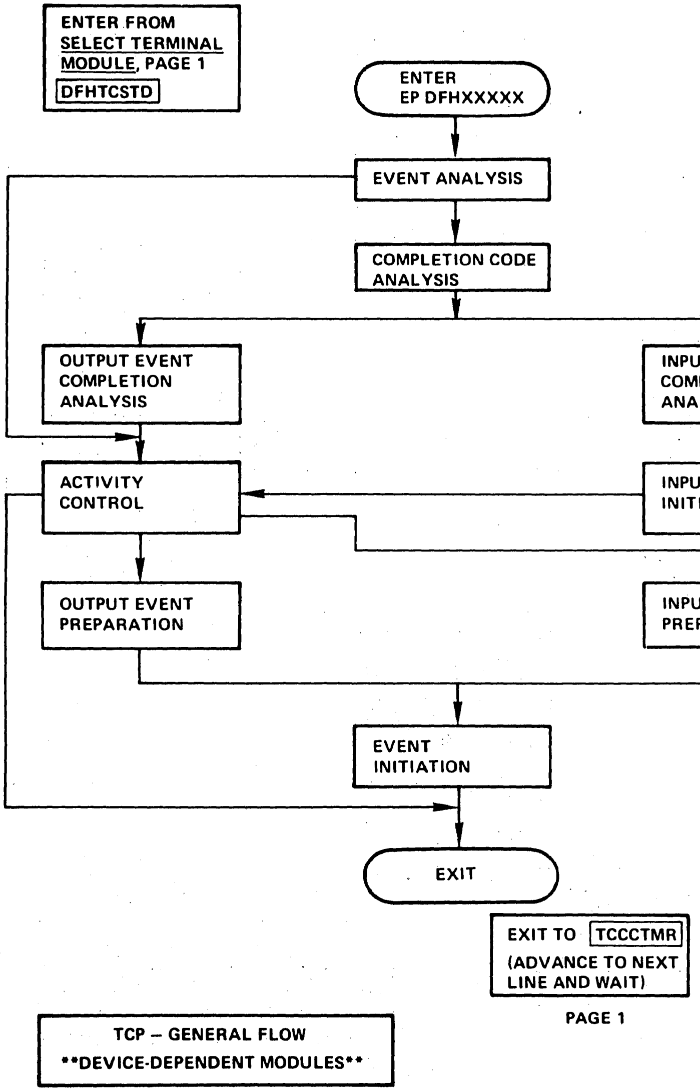


Figure 3-52. Terminal Management 3270/SS Routines



DFHXXXXX

| DEVICE-DEPENDENT ROUTINE ENTRY POINT NAMES | |
|---|---|
| DEVICE TYPE | EP NAME XXXXX |
| 3735 DIAL | 35DEE |
| 3740 DIAL | 74DEE |
| 3740 | 74NEE |
| 2780 DIAL | 80DEE |
| 2770 DIAL | 70DEE |
| 3270 | 32NEE |
| 3275 DIAL | 37DEE |
| 2770/3780 | 70NEE |
| 2980 | 29NEE |
| 3270 LOCAL | L70E |
| 1050 DIAL | T50DE |
| 2740 DIAL | T40DE |
| SYS7 DIAL | TS7E |
| 2741 DIAL | T41DE |
| TXL | TXLE |
| TWX | TWXG |
| 7770 | T77E |
| 2740 | T40E |
| SYS7 | TS7E |
| 1050 | T50E |
| 2741 | T41E |
| SEQUENTIAL | TDMSA |
| TCAM | TCAMM |

OPL52

Figure 3–53. Terminal Management General Flow through Device–dependent Modules

## TRANSACTION BACKOUT PROGRAM (DFHTBP)

The Transaction Backout program is invoked by the System Initialization program in the event of an emergency restart. It reads recovery data from the restart data set, previously written there by the Recovery Utility program. It passes control to user exits and constructs the Temporary Storage message cache and resend slot, which are used in message resynchronization and message representation.

The following notes refer to Figure 3-54.

Notes:

1. DFHTBP is invoked by DFHSIP in an emergency restart, via a conditional program control link.
2. DFHTBP reads data from the restart data set. This data comprises the Transaction Backout Table, File Backout Table, Message Backout Table, DL/I Backout Table, and transaction backout data.
3. DFHTBP saves messages for inflight tasks and unresponded-to output messages in a Temporary Storage user message cache and resend slot.
4. Records are read via requests to the Keypoint program.
5. The Storage Control program is invoked to acquire and free storage areas.
6. User-written exits, if present, are given control.
7. The Transient Data program is used to write messages, which also go to the operator console.
8. The File Control program is invoked to backout the effects that inflight tasks had on data bases under its control.
9. DFHTBP restores sequence numbers in VTAM terminal entries (including entries for terminals involved in intersystem communication). If in doubt when intersystem links failed, the description of the original failure is reconstructed (as a list of message inserts) and chained off the terminal control table entry for use during session recovery.

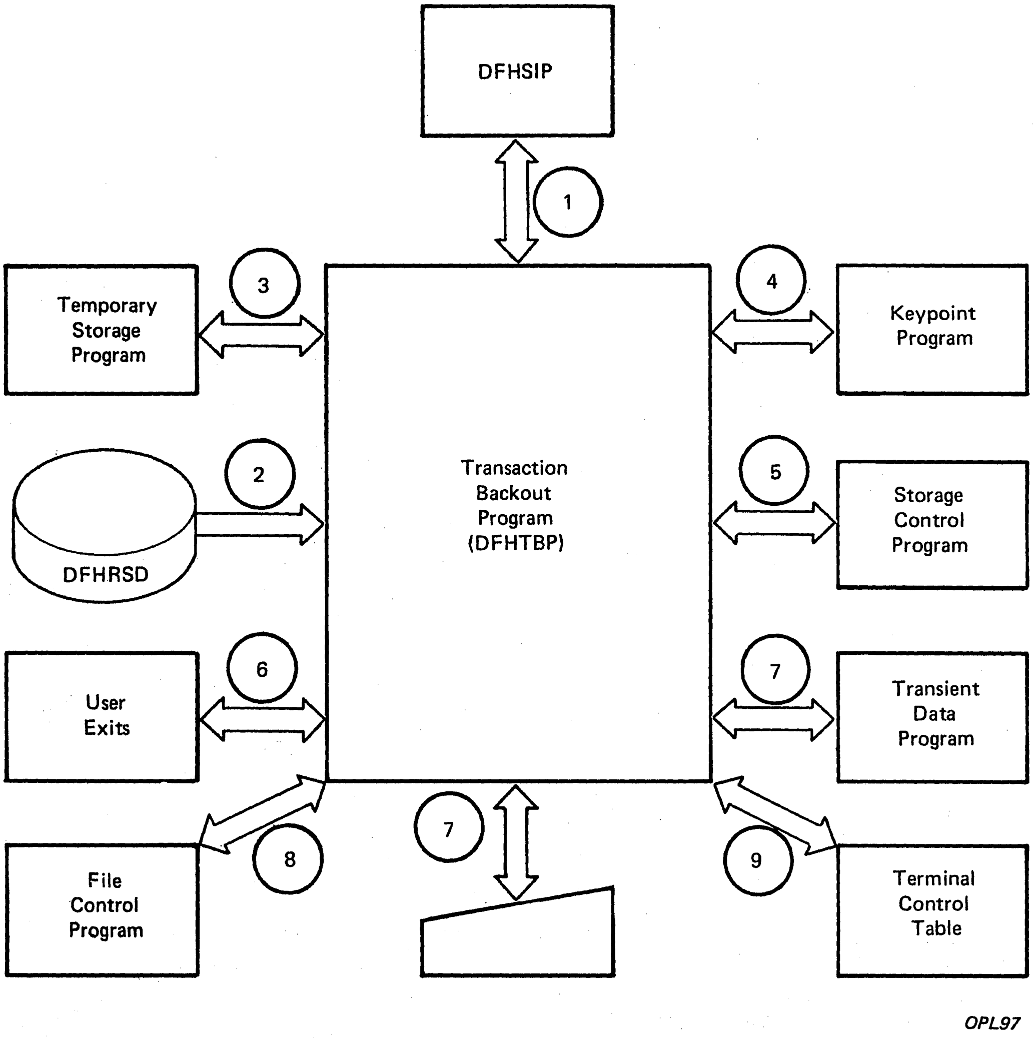


Figure 3–54. Transaction Backout Program Interfaces

## TRANSIENT DATA MANAGEMENT (DFHTDP)

The Transient Data Control program (DFHTDP) communicates directly with three other functions of the CICS/VS System Management component. These are Task Management, Storage Management, and Program Management (the Task Control program, Storage Control program, and Program Control program, respectively). The application program communicates with DFHTDP through use of DFHTD macro instructions.

The notes below refer to Figure 3-55.

Notes:

### Intrapartition

1. An application program request for Transient Data services causes a request code to be set in TCATDTR, signifying GET, PUT, PURGE, LOCATE. The destination identification is placed in TCATDDI. Control is passed to the Transient Data Control program.

2. The field TDDCTDID is used to search the DCT for the destination requested.

3. If an invalid request for Transient Data services is received, control is passed to Program Control to terminate the task issuing the request.

4. Task control is used to:

   a. lock a DCT entry, to prevent simultaneous update of the entry (BDAM). Task control is used to lock the DCT, to obtain control of the entry, and the access method control blocks (VSAM).

   b. (with logical recovery) acquire ownership of an end of a queue until the end of the LUW.

   c. wait for access to the disk map, to prevent simultaneous update.

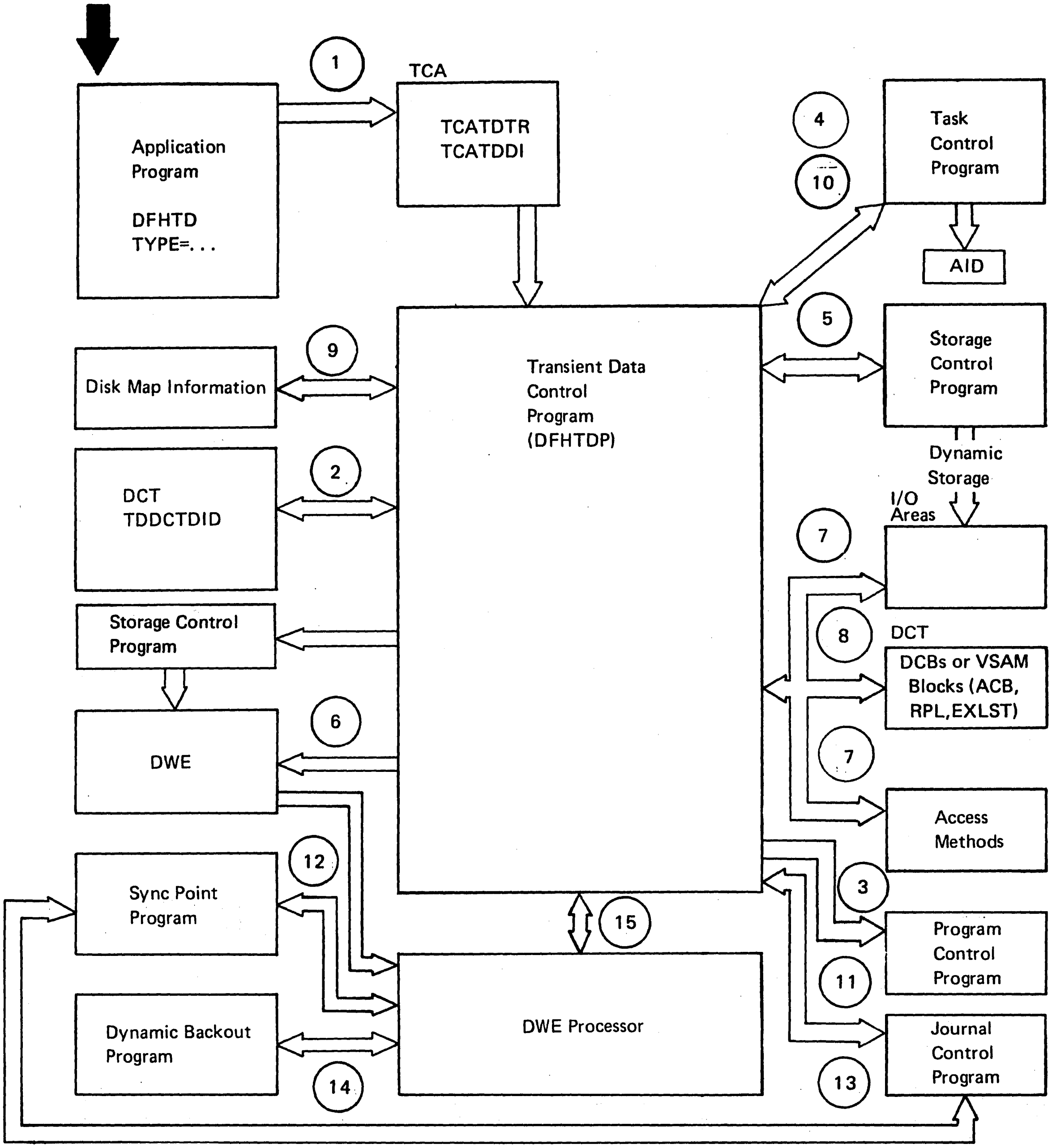
   d. wait for I/O completion.

5. Transient Data Control uses Storage Control to obtain storage for an intrapartition input area, containing a DECB.

6. If logical recovery is supported by DFHTDP and requested for the destination, storage control or RPL creates a Deferred Work Element (DWE) for each logically recoverable destination when it is accessed.

7. Transient Data passes control and the address of the DECB to the read/write routines of the access methods to perform data set I/O.

8. The DECB or RPL status bytes are tested for event completion and I/O errors.



| Figure 3-55. Transient Data Control Program Interfaces for
| Intrapartition

9. Transient Data Control obtains intrapartition disk map information from the main storage area addressed through the Destination Control Table (DCT).

10. If an automatically initiated task is to be associated with a terminal, task control is used to schedule an AID. If an automatically initiated task is not to be associated with a terminal, task control is used to attach the task.

11. For physically recoverable destinations, the Journal Control program is used to log the DCT entry.

12. At the normal end of a logical unit of work (LUW), defined by either an application program DFHSP TYPE=USER request or by Task Control at task termination, the CICS/VS Sync Point program gives control to the DWE processor of the Transient Data Control program. This DWE processor performs the logical update of the DCT entry and/or the disk map. A separate DWE is used for GET, PUT, and PURGE for each logically recoverable destination accessed by the task. The CICS/VS Sync Point program or the Dynamic Backout program (if dynamic transaction backout required) presents each DWE separately to the DFHTDP DWE processor.

13. For logically recoverable (DWE processor) destinations, the Journal Control program is used to log the DCT entry.

14. If the task abends, DFHDBP invokes the DWE processor to perform Transient Data backout of the DCT and disk map when the Dynamic Backout program encounters a Transient Data DWE.

15. The DWE processor (part of DFHTDP) uses common routines in DFHTDP.

The notes below refer to Figure 3-56.

## Extrapartition

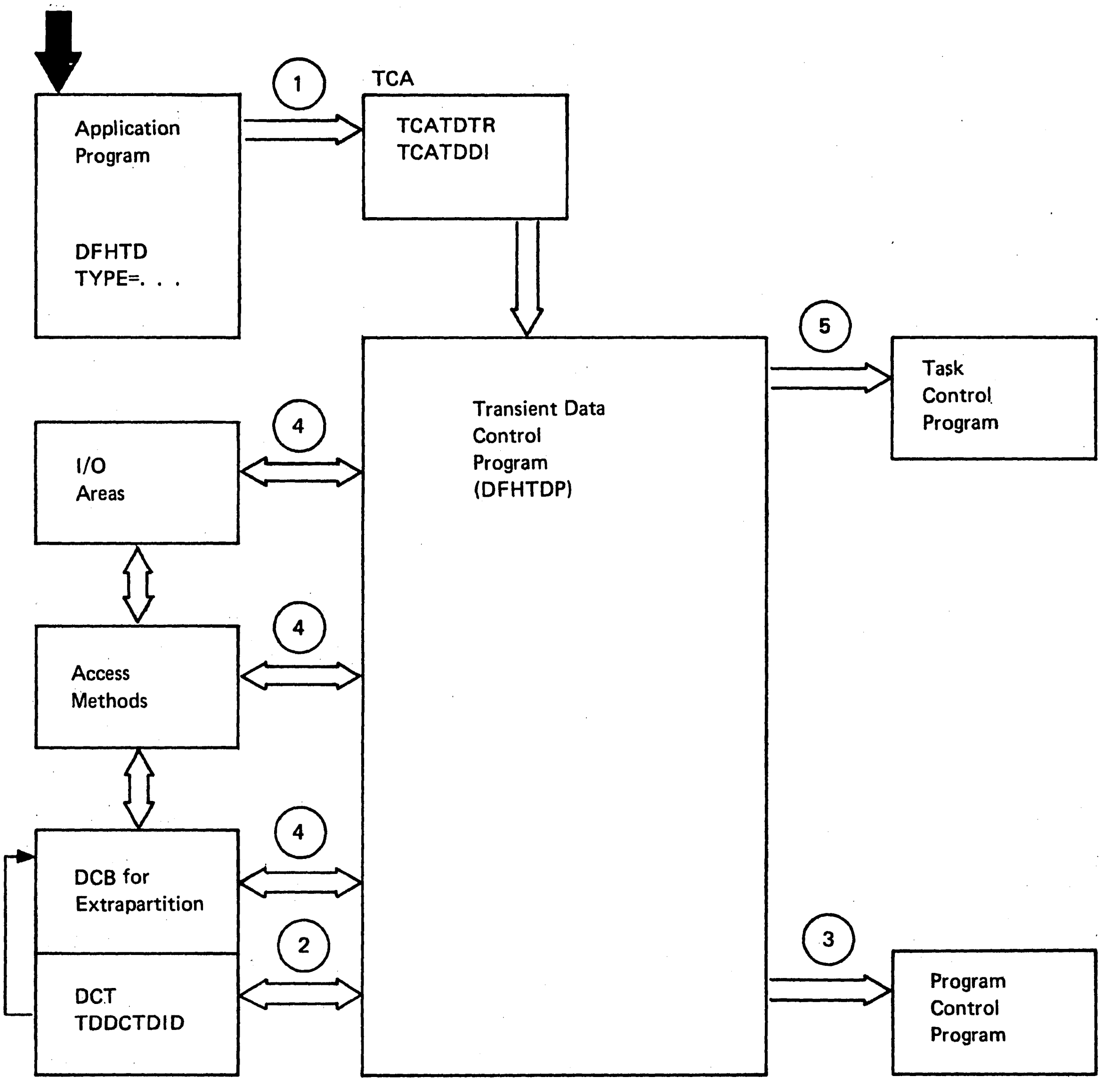
1. An application program request for Transient Data services causes a request code to be set in TCATDTR, signifying GET, PUT, FEOV, or LOCATE. The destination identification is placed in TCATDDI. Control is passed to the Transient Data Control program.

2. The DCT is searched by field TDDCTDID for the destination requested.

3. If an invalid request for Transient Data services is received, control is passed to Program Control to terminate the task issuing the request.

4. The DCB is checked for OPEN/CLOSE. Transient Data passes control and the address of the DCB to the read/write routines of the access methods to perform data set I/O.

5. The Task Control program is used to enter a dispatchable WAIT, before returning to application program.



OPL124

Figure 3-56. Transient Data Control Program Interfaces for Extrapartition

## TAPE END OF FILE (DFHTEOF) -6EH2

This program is run as a standalone program or attached by the System Initialization program (DFHSIP) during the initialization phase of an emergency restart. It performs the following functions:

- Verification of tape volumes
- Verification of log records collected as part of CICS/VS run prior to system failure
- Writing end of file

The log volume is opened and verified. If an incorrect volume is mounted, volume swapping takes place until eithe1 the correct volume is mounted or swapping is discontinued without finding the correct volume. In the latter case, the program is terminated.

As the file is processed, the label records of blocks on the file are checked to verify that they are in ascending sequence. Verification of these label records is performed as follows:

- Creation date equal to or greater than that specified on the volume label.
- Volume sequence number equal to or greater than that specified on the volume label.
- Run start time equal to that specified on the volume label.

The end of valid log data is assumed under either of two conditions:

- Negative verification occurs during validation of label records.
- Two consecutive I/O errors are encountered.

If either of these conditions occurs, the tape volume is backspaced over the appropriate number of records and an end-of-file record is written.

This page is left blank intentionally

## TRACE MANAGEMENT (DFHTRP)

The Trace Control program (DFHTRP) is designed as a debugging aid for the programmer. It provides an easy and convenient means of tracing system activity. (The content and format of entries in the Trace Table are explained in detail in the CICS/VS Application Programmer's Reference Manual.)

The notes below refer to Figure 3-57.

Notes:

1. The application program can issue Trace Control (DFHTR) macro instructions to turn on/off the trace facility for specific types of entries, to turn on/off or close the auxiliary trace function, or to cause a specified entry to be logged into the Trace Table if the trace facility is on for that type of entry.

2. Standard entries may be recorded in the Trace Table whenever CICS/VS macro instructions of the following types are issued by an application program or a CICS/VS system program:

   a. DFHKC (Task Control)
   b. DFHSC (Storage Control)
   c. DFHPC (Program Control)
   d. DFHIC (Interval Control)
   e. DFHDC (Dump Control)
   f. DFHFC (File Control)
   g. DFHTD (Transient Data Control)
   h. DFHTS (Temporary Storage Control)
   i. DFHJC (Journal Control)
   j. DFHBMS (Basic Mapping Support)
   k. DFHBIF (Built-In Functions)
   l. CICS/VS-DL/I Interface
   m. DFHDI (Data Interchange)
   n. DFHSP (Sync Point)
   o. DFHZC (VTAM Terminals)
   p. DFHXFP (Online Transformer Flow)
   q. DFHISP (ISC Request Shipping)

   These standard entries in the Trace Table are generated by special DFHTRACE macro instructions included in the CICS/VS management modules that provide these macro capabilities, or by using the DFHLFM macro.

3. Control is returned to the requesting program. Bit manipulation for on/off requests is performed in the application program itself (the inserted macro expansion) in the case of an Assembler-language program, or in Program Control in the case of an ANS COBOL or PL/I program.

4. When the Auxiliary Trace facility is active, trace table entries along with a time storage value are written to the Auxiliary Trace data set (DFHAUXTR) which can be printed offline using the Trace Utility Program (DFHTUP).

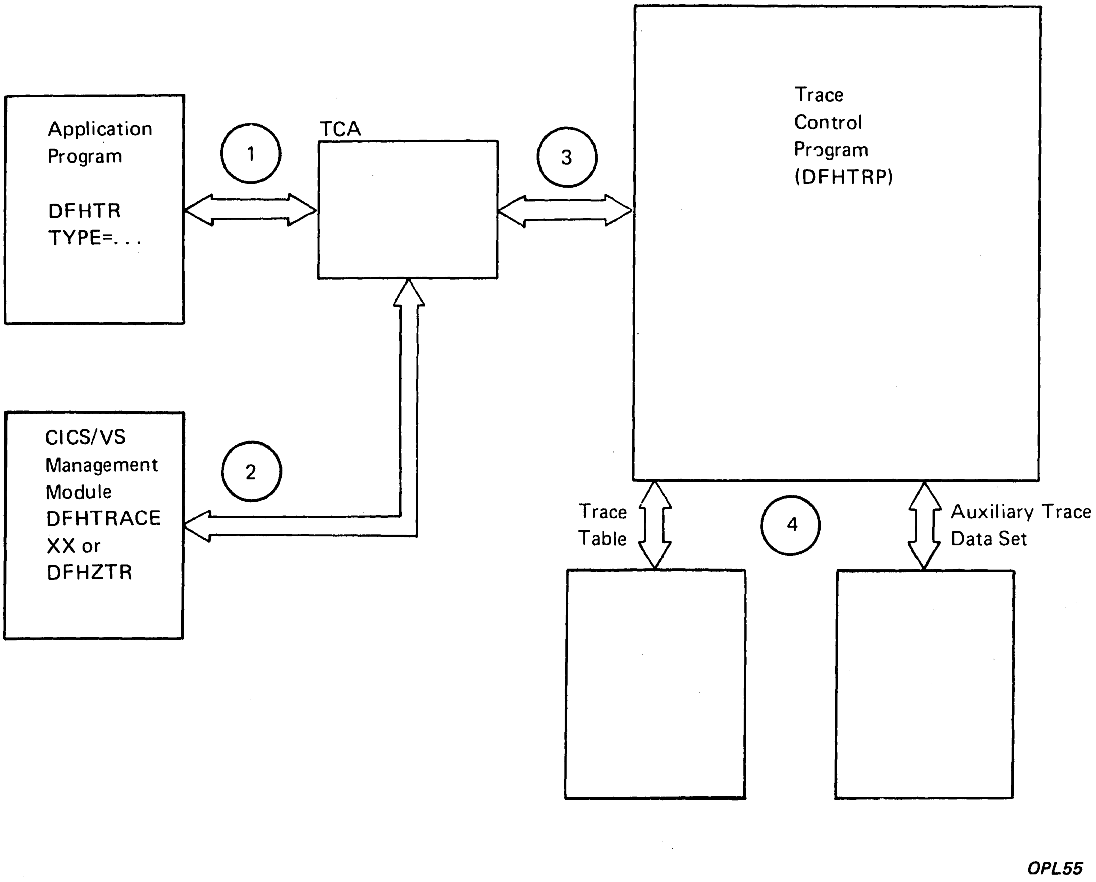


Figure 3-57. Trace Control Interfaces

## TEMPORARY STORAGE MANAGEMENT (DFHTSP)

The Temporary Storage Control program (DFHTSP) communicates directly with two other functions of the CICS/VS System Management component. These are Storage Management and Task Management (the Storage Control program and Task Control program, respectively). This interface is achieved through use of the application program's (task's) TCA.

The notes below refer to Figure 3-58.

Notes:

1. An application program requests Temporary Storage services by means of a DFHTS macro instruction.
2. Temporary Storage Control communicates with Storage Control to request Temporary Storage unit table extensions, Temporary Storage Group Identifications (TSGIDs), and main storage for DFHTS TYPE=PUT or TYPE=PUTQ requests to main storage and DFHTS TYPE=GET or TYPE=GETQ requests with no address supplied.
3. Temporary Storage Control communicates with Task Control to perform a CICS/VS WAIT pending completion of I/O (and suspending and resuming tasks).
4. Temporary Storage Control interfaces indirectly with VSAM (the access method) through the Temporary Storage Bit Map. A Request Parameter List (RPL) points to fields in this control block which are passed to VSAM.
5. VSAM communicates with Temporary Storage Control (through Task Control) when the I/O is complete.
6. Temporary Storage Control analyzes the contents of the RPL to determine the type of completion and any error information.

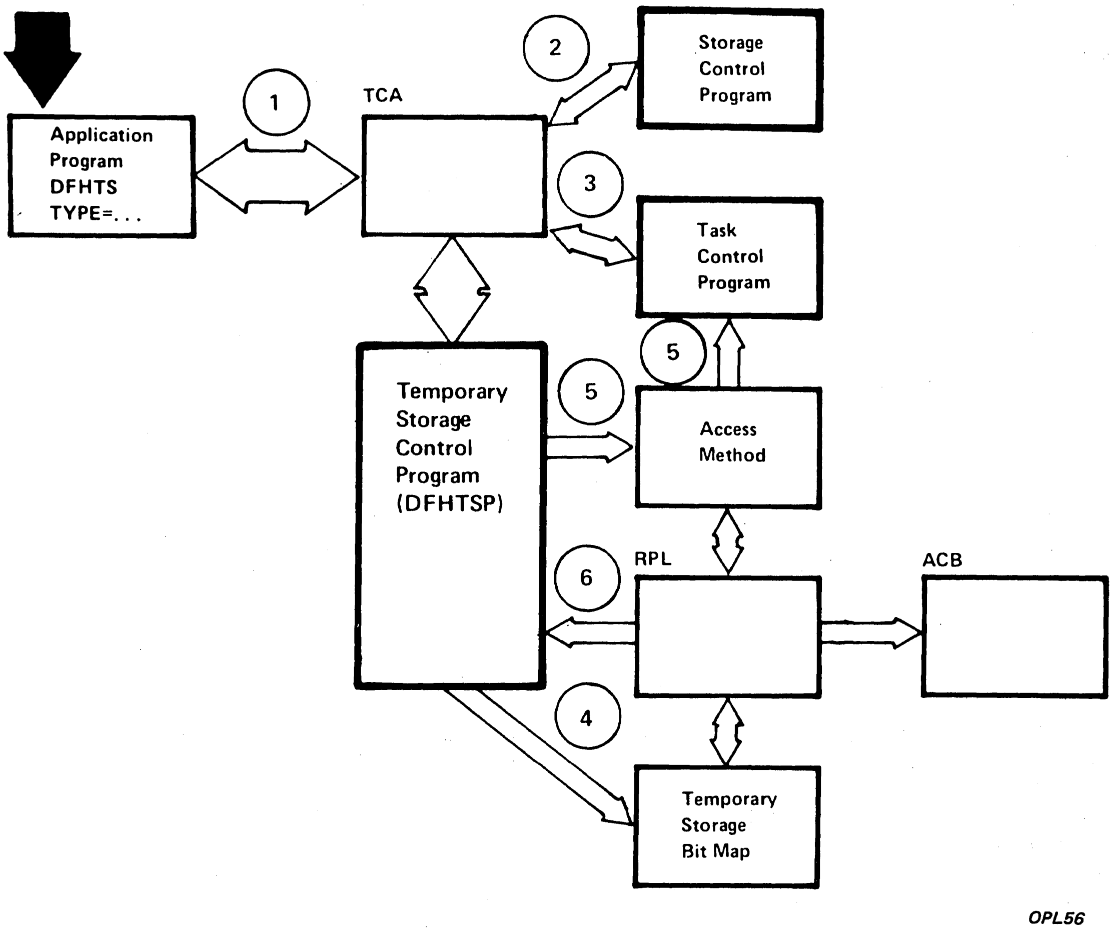


Figure 3—58. Temporary Storage Control Interfaces

## LIFO STORAGE ROUTINES (DFHLFA, DFHLFO)

The LIFO storage routines (DFHLFA and DFHLFO) provide LIFO storage space to be used as work space and to save registers for those CICS/VS modules invoked by the task. When such a module is invoked, a portion of LIFO storage space (called a LIFO stack entry) is allocated by DFHLFA as a data area for use by that particular module. If DFHLFA is unable to allocate the requested length of LIFO storage, DFHLFO is invoked to obtain another segment. The module to which LIFO storage is allocated may in turn link to another CICS/VS module, in which case a further stack entry is allocated.

Stacking continues in this way until a linked-to module completes execution. At this time the LIFO stack entry for that module is released, and the stack entries for the other uncompleted modules remain intact.

The notes below refer to Figure 3-59.

Notes:

1. Invocation of a CICS/VS module is made by a CICS/VS application program.
2. When the DFHLFM macro is executed from within a CICS/VS module, a parameter list is passed to the LIFO prologue routine (DFHLFA in module DFHCSA). DFHLFA attempts to allocate the requested length of LIFO storage from the current LIFO segment.
3. If DFHLFA is unable to allocate sufficient storage, the LIFO overflow routine DFHLFO is invoked to obtain another segment.
4. DFHLFO invokes storage control (DFHSCP) to obtain storage for the new LIFO segment and then returns control to DFHLFA.
5. DFHLFA optionally makes a trace entry for the invoking CICS/VS module.
6. DFHLFA returns control to the invoking CICS/VS module.

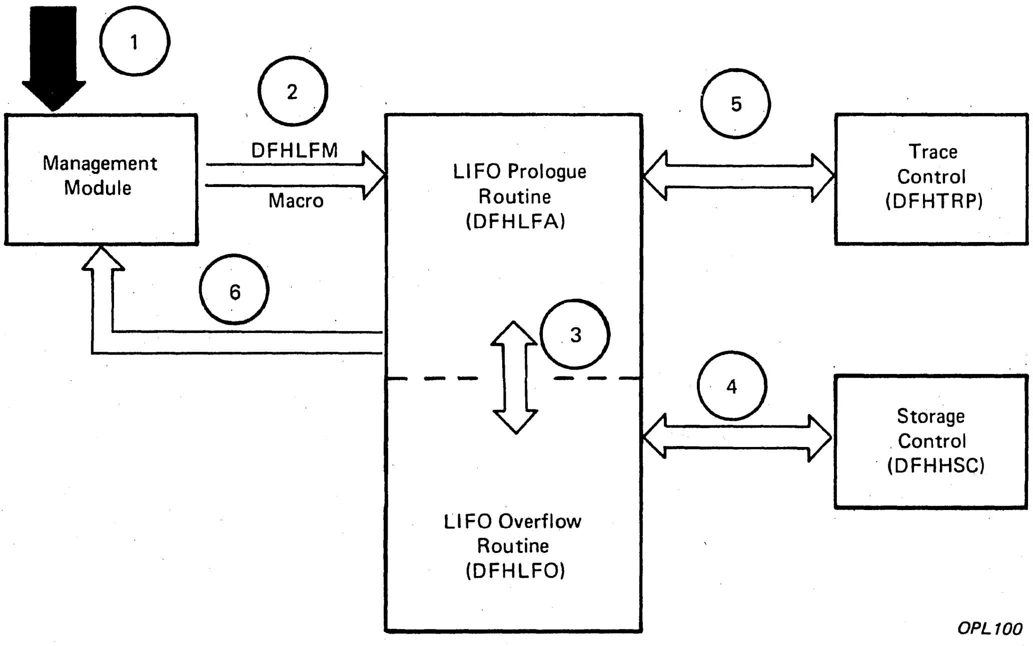


Figure 3-59. LIFO Storage Interfaces

# BIBLIOGRAPHY

Additional information about CICS/OS/VS is available in the following IBM CICS/VS publications:

Customer Information Control System/Virtual Storage (CICS/VS):

Master Index (SC33-0095)
General Information (GC33-0066)
Introduction to Program Logic (SC33-0067)
Program Logic: Data Areas (LY33-6035)
System/Application Design Guide (SC33-0068)
Application Programmer's Reference Manual (Command Level) (SC33-0077)
Application Programmer's Reference Manual (Macro Level) (SC33-0079)
Application Programmer's Reference Summary (Command Level) (GC33-6012)
System Programmer's Reference Manual (SC33-0069)
System Programmer's Guide (OS/VS) (SC33-0071)
Operator's Guide (SC33-0080)
Messages and Codes (SC33-0081)
Problem Determination Guide (SC33-0089)
Master Terminal Operator Reference Summary (SX33-6011)
Program Debugging Reference Summary (SX33-6010)

There are also a number of manuals dealing with the various terminal systems with which CICS/OS/VS interfaces:

IBM 3600/3630 Guide (SC33-0072)
IBM 3650 Guide (SC33-0073)
IBM 3767,3770, and 6670 Guide (SC33-0074)
IBM 3790 Guide (SC33-0075)

As a System/370 user, you should be generally familiar with one or both of the following documents:

- OS/VS1 Planning and Use Guide (GC24-5090)
- OS/VS2 Planning and Use Guide (GC28-0600)

For reference information on space allocation, see:

OS/VS Data Management Services Guide (GC26-3783)

For reference and "how-to" information on allocating, extending, and releasing space, see:

- OS/VS JCL Services (GC28-0617)
- OS/VS JCL Reference (GC28-0618)

For reference and "how-to" information on deleting data sets, renaming data sets, and protecting data sets, see:

OS/VS Data Management for System Programmers (GC28-0631)

For access method logic:

- OS/VS SAM Logic (SY26-3788)
- OS/VS ISAM Logic (SY26-3786)
- OS/VS BDAM Logic (SY26-3789)
- OS/VS BTAM Logic (SY27-7246)
- OS/VS Virtual Storage Access Method (VSAM) Logic (SY26-3818)

Users of the CICS/VS-DL/I Interface should see:

- IMS/VS Program Logic Manual (LY20-8004 and LY20-8005)
- IMS/VS Application Programmer's Reference Manual (SH20-9026)
- IMS/VS Utilities Reference Manual (SH20-9029)

| Information on Systems Network Architecture (SNA) can be found in the
| following two Systems Network Architecture publications:

| - Functional Description of Logical Unit Types (GC20-1868)

| - Types of Logical Unit to Logical Unit Sessions (GC20-1869)

Other OS/VS publications are listed in theIBM System/370 Bibliography (GC20-0001). The IBM System/370 Advanced Function Bibliography (GC20-1763) is a valuable supplement for System/370 VS users.

## AVAILABILITY OF PUBLICATIONS

The availability of a publication is indicated by its use key, which is the first letter in the order number. The use keys and their meanings are:

G — Generally available: Provided to users of IBM systems, products, and services without charge,in quantities to meet their normal requirements. Can also be purchased by anyone through IBM branch offices.

S — Sold: Can be purchased by anyone through IBM offices.

L — Licensed Material, property of IBM: Available only to licensees of the related program products under the terms of the license agreements.

Each page number in this index refers to the start of the paragraph containing the indexed item. Where more than one page reference is given, the major reference is first.

Customer Information Control System/Virtual Storage (CICS/VS)
Program Logic: Description (OS/VS)

LY33-6034-0

**READER'S COMMENT FORM**

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. 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.*

Number of your latest Technical Newsletter for this publication. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

**Note:** *Staples can cause problems with automated mail sorting equipment. Please use pressure-sensitive or other gummed tape to seal this form.*

— **Cut Along Dotted Line** —

If you want an acknowledgement, give your name and address below.

Name . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Job Title . . . . . . . . . . . . . . Company . . . . . . . . . . . . . . . . .

Address . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

. . . . . . . . . . . . . . . . . . . . . . . . Zip . . . . . . . . . .

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

LY33-6034-0

Reader's Comment Form

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 40 ARMONK, NEW YORK

Postage will be paid by addressee:

International Business Machines Corporation
Department 812H
1133 Westchester Avenue
White Plains, New York 10604

Fold and tape Please do not staple Fold and tape

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

CICS/VS Program Logic: Description (OS/VS) Printed in U.S.A. LY33-6034-0

IBM Technical Newsletter

This Newsletter No. LN33-6252
Date December 1978

Base Publication No. LY33-6028-1

Previous Newsletters None

**Customer Information Control System/Virtual Storage (CICS/VS) Version 1, Release 4 Program Logic (DOS/VS)**

This technical newsletter, a part of Version 1, Release 4 of Customer Information Control System/Virtual Storage (CICS/VS), provides replacement pages for the subject publication. These replacement pages remain in effect for subsequent versions and releases unless specifically altered. Pages to be inserted or removed are:

895,896
899,900
919,920
923,924
997–1008
1008.1,1008.2(added)
1225,1226
1226.1,1226.2(added)

A change to the text or to an illustration is indicated by a virtical line to the left of the change.

**Summary of Amendments**

The main changes introduced by this TNL are the addition of the Transient Data VSAM Control Interval Map, and corrections to the Function Management Header, both in the Data Areas section of the publication.

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

**IBM United Kingdom Laboratories Ltd, Technical Documentation Department, Hursley Park, Winchester, Hampshire SO21 2JN, England.**

## TABLE OF CONTENTS

## COMMON SYSTEM AREA (CSA)

DSECT NAME: DFHCSADS

The Common System Area (CSA) is a main storage control area provided for each system as a part of CICS/VS. The CSA exists within the system from initialization of the system until the system is closed down. The CSA is composed of areas of data essential to the operation of CICS/VS and an optional work area that may be used as temporary working storage by processing programs. The user temporary working storage is available to any task while it has control of the system (that is, for operations performed between requests to CICS/VS).

ALPHABETIC INDEX TO FIELD NAMES

| | Offset | | Offset | | Offset |
|---|---|---|---|---|---|
| | or | | or | | or |
| Name | Reference | Name | Reference | Name | Reference |
| CSAACTBA | B4 | | | CSASUSBA | AC |
| CSAACTFA | B0 | | | CSASUSFA | A8 |
| | | | | CSASUSOF | 9C |
| | | CSANXDCA | A4 | | |
| CSAAIDBA | 94 | CSAOPFLA | C8 | | |
| | | CSAOPREL | 9D | | |
| | | CSAOPSYS | 9C | | |
| CSABASCL | 16C | CSAOSRSA | 0 | CSATADJT | 5C |
| CSABASTU | 170 | CSAPAMA | 98 | CSATCNAC | F4 |
| CSABTCCB | D8 | | | | |
| CSACDTA | 4C | CSAPCDTF | 190 | CSATCNDT | 1AC |
| CSACICS | 9E | | | | |
| CSACIREL | 9F | | | | |
| CSACSCC | 60 | CSAPCNAC | E8 | | |
| CSACTODB | 60 | CSAPCTBA | 120 | CSATCTBA | 128 |
| | | CSAPINI | 1DF | CSATCTCA | B8 |
| CSADATFT | 5A | CSAPIPSW | 10C | CSATDNAC | FC |
| | | CSAPLBA | 74 | CSATDNT | 1F0 |
| | | CSAPOLA | 140 | CSATDOLA | 150 |
| CSADCNAC | F0 | CSAPPTBA | 124 | CSATDTCA | 80 |
| CSADCND | 1E1 | CSAPUBA | 78 | CSATODB | 6C |
| | | CSAQCAA | 13C | | |
| CSADCTBA | 130 | CSAQETBA | 13C | CSATODP | 50 |
| CSADOLA | 144 | CSARDATC | 1B6 | CSATODTU | 1A8 |
| CSAEXTM | 194 | | | | |
| CSAFCNAC | F8 | CSASANAC | 104 | CSATOLA | 148 |
| CSAFCTBA | 12C | CSASBTI | 64 | CSATRMF1 | 84 |
| | | CSASCAR | 1CF | CSATRNAC | 108 |
| CSAFOLA | 14C | CSASCCR | 1D7 | CSATRMF2 | 85 |
| CSAICEBA | 54 | CSASCFI | 1D3 | CSATRMF3 | 86 |
| | | | | CSATRMF4 | 87 |
| CSAICFNA | 19C | CSASCMQ | 1DB | CSATRNAC | 108 |
| | | CSASCNAC | E4 | CSATRTBA | 11C |
| CSAICIND | 5B | CSASCNB | 70 | CSATSASA | 1E6 |
| | | CSASCPXM | E0 | CSATSATA | 134 |
| CSAICNAC | EC | CSASCQZ | 1DD | CSATSDTI | 8A |
| CSAICRIC | 1B0 | CSASCRQ | 1D9 | CSATSMSA | 1E3 |
| CSAICRIN | 1A1 | | | | |
| CSAICRNX | 1A0 | CSASITBA | 8C | CSATSMTA | 138 |
| CSAICRUN | 1B4 | CSASITOD | 6C | CSATSNAC | 100 |
| CSAICSIC | 58 | CSASNNAC | 110 | CSATSOLA | 154 |
| CSAIOCP | 1C6 | | | | |
| CSAJYDP | 7C | CSASOSI | 48 | CSATTECB | 68 |

| Name | Offset or Reference | Name | Offset or Reference | Name | Offset or Reference |
|---|---|---|---|---|---|
| CSAKCCT | 1C8 | | | CSAUNQID | 90 |
| CSAKCMI | 49 | CSASPA2 | 1EB | CSAUTA1 | 1F3 |
| CSAKCMT | 4A | CSASPA3 | 1ED | CSAUTA2 | 1F6 |
| CSAKCMTA | 1CA | CSASPNAC | 114 | CSAUTA3 | 1F9 |
| CSAKCMTC | 1C0 | CSASSI1 | 48 | CSAUTA4 | 1FC |
| CSAKCNAC | E0 | CSASSI2 | 49 | CSAWABA | 200 |
| | | | | CSAWAL | C6 |
| CSAKCTTA | 1CC | | | CSAWTOAD | 18C |
| CSAKPACT | CE | | | | |
| CSAKPCLK | D0 | | | | |
| CSAKPCNT | 1C4 | | | | |
| CSAKPFRQ | CC | | | | |
| CSAMAXT | C0 | | | | |
| CSAMAXTC | C2 | | | | |

STORAGE LAYOUT

| Offset | Length | Name | Description | | |
|---|---|---|---|---|---|
| 0 | 72 | CSAOSRSA | Common system register storage area | | |
| 48 | 1 | CSASOSI | Short-on-storage indicator, or | | |
| | | CSASSI1 | System signal indicator 1 | | |
| | | | | Value | Function |
| | | CSASOSON | | X'01' | Short on storage condition exists |
| 49 | 1 | CSAKCMI | Maximum number of tasks indicator, or | | |
| | | CSAECI | CICS subset indicator, or | | |
| | | CSASSI2 | System signal indicator 2 | | |
| | | | | Value | Function |
| | | CSAMXTON | | X'02' | Maximum task indicator, on condition |
| | | CSATQIM | | X'04' | Transaction quiesce indicator mask |
| | | CSATCPQM | | X'08' | Terminal Control quiesce mask |
| | | CSAPLTPI | | X'10' | PLTPI phase has completed |
| | | CSAECICS | | X'20' | CICS Subset Indicator |
| | | CSAFNLTM | | X'40' | Final termination phase posting mask |
| | | CSASTIM | | X'80' | System termination indicator mask |
| 4A | 2 | CSAKCMT | Maximum number of tasks | | |
| 4C | 4 | CSACDTA | Currently dispatched TCA task address | | |
| 50 | 4 | CSATODP | Time of day. A packed integer of the form HHMMSSTC where HH is hours, MM is minutes, SS is seconds, T is tenths of a second, and C is a positive sign. | | |

| Offset | Length | Name | Description | |
|---|---|---|---|---|
| 9C | 1 | CSAOPSYS | Operating System | |
| | | | Value | Function |
| | | CSADOS | C'D' | DOS/VS |
| | | CSAVS1 | C'1' | OS/VS1 |
| | | CSASVS | C'S' | OS/SVS |
| | | CSAMVS | C'M' | OS/MVS |
| 9D | 1 | CSAOPREL | Operating System release | |
| | | | Value | Function |
| | | CSAD34 | X'34' | DOS/VS Release 34 |
| | | CSA160 | X'60' | OS/VS1 Release 6.0 |
| | | CSAS17 | X'17' | OS/SVS Release 1.7 |
| | | CSAM37 | X'37' | OS/MVS Release 3.7 |
| 9E | 1 | CSACICS | CICS/VS System | |
| | | | Value | Function |
| | | CSAELS | C'E' | Entry Level System (ELS) |
| | | CSAFULL | C'F' | Full CICS/VS |
| 9F | 1 | CSACIREL | CICS/VS Release | |
| | | | Value | Function |
| | | CSAC14 | X'14' | Version 1, Release 4 |
| A0 | 4 | | Zero. Dummy suspended task DCA has DCATRIO set to zero. | |
| A4 | 4 | CSANXDCA | Next available Dispatch Control Area (DCA) | |
| A8 | 4 | CSASUSFA | Lowest priority suspended task DCA | |
| AC | 4 | CSASUSBA | Highest priority suspended task DCA | |
| B0 | 4 | CSAACTFA | Lowest priority active task DCA | |
| B4 | 4 | CSAACTBA | Highest priority active task DCA | |
| B8 | 4 | CSATCTCA | Terminal Control TCA address | |
| BC | 4 | | Dummy dispatch control, priority, etc | |
| C0 | 2 | CSAMAXT | Maximum active task value | |
| C2 | 2 | CSAMAXTC | Maximum active tasks, accumulated | |
| C4 | 2 | | Reserved | |
| C6 | 2 | CSAWAL | Length, in bytes, of CSA + CSA work area + system ID | |
| C8 | 4 | CSAOPFLA | CSA Optional Features List address | |
| CC | 2 | CSAKPFRQ | Activity keypoint frequency counter | |
| CE | 2 | CSAKPACT | Activity keypoint attach counter | |
| D0 | 8 | CSAKPCLK | Activity keypoint, store clock | |
| D8 | 4 | CSABTCCB | BTAM master CCB address | |

| Offset | Length | Name | Description |
|---|---|---|---|
| DC | 4 | | Memory constant – CNST |
| E0 | 4 | CSAKCNAC | Task Control entry address |
| E4 | 4 | CSASCNAC | Storage Control entry address<br>Value Function |
| | | CSASCPXM | X'80' Storage Control program–check exit active |
| E8 | 4 | CSAPCNAC | Program Control entry address |
| EC | 4 | CSAICNAC | Interval Control entry address |
| F0 | 4 | CSADCNAC | Dump Control entry address |
| F4 | 4 | CSATCNAC | Terminal Control entry address |
| F8 | 4 | CSAFCNAC | File Control entry address |
| FC | 4 | CSATDNAC | Transient Data Control entry address |
| 100 | 4 | CSATSNAC | Temporary Storage Control entry address |
| 104 | 4 | CSASANAC | PL/I Storage Allocation module entry |
| 108 | 4 | CSATRNAC | Trace Control entry address |
| 10C | 4 | CSAPIPSW | Program Interrupt PSW save area address, or |
| | 4 | CSAPINAC | Program Interrupt entry address (during System Initialization only) |
| 110 | 4 | CSASNNAC | Snapshot program entry address |
| 114 | 4 | CSASPNAC | Sync point program entry address |
| 118 | 4 | CSATCRWE | Terminal Control program entry address |
| 11C | 4 | CSATRTBA | Trace Table beginning address |
| 120 | 4 | CSAPCTBA | Program Control Table beginning address |
| 124 | 4 | CSAPPTBA | Processing Program Table beginning address |
| 128 | 4 | CSATCTBA | Terminal Control Table beginning address |
| 12C | 4 | CSAFCTBA | File Control Table beginning address |
| 130 | 4 | CSADCTBA | Destination Control Table beginning address |

| Offset | Length | Name | Description |
|---|---|---|---|
| A | 6 | | Reserved |
| 10 | 4 | FCIOAECI | Event control information field, or |
| | 10 | FCIDLOC | ID location for DAM |
| 1A | 1 | FCFIOEX | ISAM exception code, or |
| | 2 | FCIOERR | DAM error bytes |
| 1C | 4 | FCRSAVE | Save area for record size |
| 20 | 4 | FCFECADR | Address of exclusive control data |
| 24 | 8 | FCNREPSV | File Control subroutine save area |
| 2C | 2 | FCIDTYPI | DAM block reference indicator |
| 2E | 1 | FCDUMISR | Dummy record search indicator, or ISAM embedded read indicator |
| 2F | 1 | FCFIOTYP | Type of operation (i.e., READKEY) |
| 30 | 4 | FCFIOVAR | Variable area beginning address, or |
| 30 | 2 | FCFIOVRL | Variable record length, and |
| 32 | 2 | FCMVDISP | Block reference displacement in ID location |
| 34 | 4 | FCFIOLRA | Logical record address |
| 38 | 4 | FCFIOFCT | File Control Table entry address |
| 3C | 4 | FCFBWA | Address of File Browse Work Area (FBWA) |
| 40 | 4 | FCFIOICA | File I/O Area (FIOA) chain address |
| 48 | 2 | FIOACAE | Control area ending address |
| 48 | 2 | FCFCOUNT | Count field (CCHHRKDD) |
| 50 | 4 | FCDS01D | Beginning address of data area |
| | 4 | FIOADBA | |

## FUNCTION MANAGEMENT HEADER (FMH)

The Function Management Header (FMH) is a data string containing SNA information. The FMH is optionally added to the data sent or received by SNA devices, according to requirements.

DSECT NAME: DFHFMHDS
REGISTER: FMHBAR

ALPHABETIC INDEX TO FIELD NAMES

| | Name | Offset or Reference |
|---|---|---|
| \| | FMHARDPN | CD |
| \| | FMHARPRN | CD |
| | FMHATDBA | 7 |
| \| | FMHATDPL | CD |
| \| | FMHATDPN | CD |
| | FMHATDPV | 1 |
| \| | FMHATDQN | CD |
| \| | FMHATPRN | CD |
| | FMHATSEC | 6 |
| \| | FMHCARGL | CD |
| | FMHCARGN | 2 |
| | FMHCARGV | 3 |
| \| | FMHCOPTS | CD |
| \| | FMHCRCDE | CD |
| | FMHCT | 1 |
| | FMHDAREA | 3 |
| \| | FMHDDBDN | CD |
| | FMHDDRCD | 6 |
| | FMHDDSTS | 7 |
| | FMHDESEL | 4 |
| \| | FMHDIOMX | CD |
| \| | FMHDLENG | CD |
| | FMHDLOCK | 6 |
| | FMHDMKYL | 7 |
| | FMHDNTNT | 3 |
| | FMHDPARM | 3 |
| | FMHDPCBI | 6 |
| | FMHDPKEY | 3 |
| \| | FMHDPSBN | CD |
| | FMHDPSSA | 3 |
| | FMHDRCDS | 6 |
| | FMHDRCD1 | 6 |
| | FMHDRCD2 | 7 |
| \| | FMHDSAMX | CD |
| | FMHDESGL | 8 |
| \| | FMHDSEGN | CD |
| | FMHDSEGS | B |
| | FMHDSNH | 9 |
| | FMHDSNL | 8 |
| | FMHDSRCS | 6 |
| | FMHDSRC1 | 6 |
| | FMHDSRC2 | 7 |
| | FMHDSTCD | A |
| | FMHDTYPE | 2 |
| | FMHERCI | 5 |
| | FMHFLAGS | 4 |
| | FMHFLAG3 | 3 |

| Name | Offset or Reference |
|---|---|
| FMHFN | 3 |
| FMHFORG | 6 |
| FNHGROUP | 2 |
| FMHHD | 1 |
| FMHL | 0 |
| FMHLDC | 2 |
| FMHLEN | 0 |
| FMHLENG | 0 |
| FMHMEDIA | 2 |
| FMHQCURS | CD |
| FMHQNAME | CD |
| FMHQQORG | 6 |
| FMHQRCLN | CD |
| FMHQRCNT | CD |
| FMHQSENS | CD |
| FMHQSTAT | 7 |
| FMHQSTA1 | 7 |
| FMHQSTA2 | 8 |
| FMHQTRSZ | CD |
| FMHQXFST | 7 |
| FMHSCDPN | CD |
| FMHRESV2 | 6 |
| FMHSDELY | CD |
| FMHSIREQ | CD |
| FMHSMCCD | 8 |
| FMHSMDCD | C |
| FMHSMNUM | 6 |
| FMHSPPRN | CD |
| FMHSPRN | CD |
| FMHSPSTS | 6 |
| FMHSQNME | CD |
| FMHSRDPN | CD |
| FMHSREQN | CD |
| FMHSRQST | 6 |
| FMHSSDPN | CD |
| FMHSSSTS | 6 |
| FMHTYPE | 1 |
| FMHXCMD | 2 |
| FMHXFXCT | 5 |
| FMHXMOD | 4 |
| FMH2DAT1 | 3 |
| FMH2DAT2 | 4 |
| FMH2NURC | 3 |
| FMH2OPCD | 2 |
| FMH2RITY | 3 |

STORAGE LAYOUT

** COMMON SECTION **

| Offset | Length | Name | Description | |
|---|---|---|---|---|
| 0 | 1 | FMHLENG | Length of FMH | |
| | | | Value | Function |
| | | FMHL3600 | X'03' | Length of 3600 FMH |
| | | FMHLBLU | X'06' | Length of Batch LU FMH |

| Offset | Length | Name | Description | |
|---|---|---|---|---|
| 1 | 1 | FMHHD | Header description | |
| | | | Value | Function |
| | | FMHFD | X'40' | Message has formatted data |
| | | FMHALARM | X'20' | Trigger alarm at device |
| | | FMHTBLU | X'01' | Batch LU is type X'01' |
| 2 | 1 | FMHLDC | Logical device code<br>Same values as in DFHSLDC, except: | |
| | | | Value | Function |
| | | FMHBLUIN | X'80' | Input indicator for Batch LU |
| 3 | 1 | | Reserved | |

** BATCH LU EXTENSION **

| Offset | Length | Name | Description | |
|---|---|---|---|---|
| 4 | 1 | FMHFLAGS | Batch LU flags | |
| | | | Value | Function |
| | | FMHSUSP | X'80' | Suspend data set |
| | | FMHBODS | X'40' | Beginning of data set |
| | | FMHEODS | X'20' | End of data set |
| 5 | 1 | | Reserved | |

**TYPE 1 FMH FORMAT**

| Offset | Length | Name | Description | |
|---|---|---|---|---|
| 0 | 1 | FMHLEN | Length of complete FMH | |
| 1 | 1 | FMHTYPE | Type of FMH | |
| | | | Value | Function |
| | | FMHFTYP1 | X'01' | Type 1 FMH |
| | | FMHFTYP2 | X'02' | Type 2 FMH |
| | | FMHFTYP3 | X'03' | Type 3 FMH |
| | | FMHFCONC | X'80' | Concatenated FMH |
| 2 | 1 | FMHMEDIA | Media selection byte | |
| | | | Value | Function |
| | | FMHMEFCN | X'00' | Console |
| | | FMHMEFEX | X'10' | Exchange media |
| | | FMHMEFCD | X'20' | Card reader |
| | | FMHMEFPR | X'30' | Print |
| | | FMHMEFDI | X'40' | Disk |
| | | FMHMEFPD | X'60' | PDS |
| | | FMHMEFAN | X'7F' | Any |
| | | | Only bits 1–3 used<br>Bit 0 reserved<br>Bits 4–7 = logical subaddress | |
| 3 | 1 | FMHFLAG3 | Flag byte | |
| | | | Value | Function |
| | | FMHT1STK | X'80' | 'your' stack indicator |
| | | | Bit 1–7 reserved | |

| Offset | Length | Name | Description | |
|---|---|---|---|---|
| 4 | 1 | FMHDESEL | Destination select field | |
| | | | Value | Function |
| | | | Bit 0-2 | only |
| | | FMHDEFRE | X'00' | Resume data set |
| | | FMHDEFEN | X'20' | End data set |
| | | FMHDEFBG | X'40' | Begin data set |
| | | FMHDEFBD | X'60' | Begin and end data set |
| | | FMHDEFSU | X'80' | Suspend data set |
| | | FMHDEFAB | X'A0' | Abort data set |
| | | | Bits 3-7 reserved | |
| 5 | 1 | FMHERCI | Exchange record length | |
| 6 | 2 | FMHRESV2 | Reserved | |
| 8 | 1 | FMHDSNL | Length of destination name | |
| 9 | 1 | FMHDSNH | Actual DSN name | |

** TYPE 2 FMH OVERLAY **

| Offset | Length | Name | Description | |
|---|---|---|---|---|
| 2 | 1 | FMH2OPCD | Type of operation | |
| | | | Value | Function |
| | | FMH2FADD | X'24' | ADD operation |
| | | FMH2FREP | X'25' | REPLACE operation |
| | | FMH2FQUE | X'28' | QUERY operation |
| | | FMH2FNOT | X'29' | NOTE operation |
| | | FMH2NTRY | X'2A' | NOTE REPLY operation |
| | | FMH2FRID | X'2B' | RECID operation |
| | | FMH2FERA | X'2C' | ERASE operation |
| | | FMH2FVOL | X'2E' | VOLID operation |
| 3 | | FMH2NURC | Number of records affected | |
| 3 | | FMH2RITY | Type of key for RECID type | |
| | | | Value | Function |
| | | FMH2RIAK | X'00' | Addressed direct |
| | | FMH2RID1 | X'01' | Key direct KEY1 |
| | | FMH2RID2 | X'02' | Key direct KEY2 |
| | | FMH2RIAP | X'03' | Application definition |
| | | FMH2RICC | X'04' | Control definition |
| 3 | | FMH2DAT1 | Start of data - first type | |
| 3 | 1 | | Overlayed byte | |
| 4 | 1 | FMH2DAT2 | Start of data - second type | |

The following part of the DSECT describes Function Management Headers, and in some cases the data that can follow the header.

The organization of the definitions given below is as follows:

1. The standard part of a Function Management Header. These definitions apply whatever type, group, and function code the header may carry.

2. Definitions for Function Management Headers of type 5; that is, attach headers. These may be identified by the prefix 'FMHA'.

3. Definitions for Function Management Headers of type 43; that is, CICS private headers. These may be identified by the prefix 'FMH'.

4. Definitions for Function Management Headers of type 6; that is, scheduler model, queue model, and DL/I model headers. These may be identified by the prefixes 'FMHS', 'FMHQ', and 'FMHD' respectively.

Within the definitions for each FMH, definitions for fixed length parameters appear before the definitions for variable length parameters. The sequence of the parameters in this DSECT is the same as that in which parameters can appear in the FMH.

Variable length parameters have a length of 1 or 2 bytes and are followed by a value. A variable length parameter is omitted if the length is 0, and need not appear if it is omitted and also the last parameter in the FMH.

Thus, for variable length parameters, the offset value is generally context dependent and no absolute offset value can be given. (In these cases the offset value is replaced by CD; CD indicating 'context dependent'). For convenience of processing by the transformation program, the FMH pointer is assumed to address the length byte immediately preceeding the value.

| Offset | Length | Name | Description | |
|---|---|---|---|---|
| 0 | 1 | FMHL | Length of FMH | |
| 1 | 1 | FMHCT | Concatenation flag and FMH type bits set as follows: | |
| | | | Value | Function |
| | | FMHCAT | X'80' | A second FMH comes after this one |
| | | | Bits 1–7 FMH type values set as follows | |
| | | | Value | Function |
| | | FMHT05 | X'05' | IBM architected attach FMH |
| | | FMHT06 | X'06' | IBM architected model FMH |
| | | FMHT07 | X'07' | IBM architected system message FMH |
| | | FMHT43 | X'43' | CICS architected model FMH |
| 2 | 2 | FMHXCMD | Group and function codes | |
| 2 | 1 | FMHGROUP | Group code | |
| 3 | 1 | FMHFN | Function code | |
| 4 | 1 | FMHXMOD | Modifier | |
| | | | Value | Function |
| | | FMHXLNSZ | X'80' | '0' for 1 byte FMH Length fields |
| | | | Bits 1–7 reserved | |
| 5 | 1 | FMHXFXCT | Length of fixed length parameters in FMH | |
| 6 | 1 | FMHFORG | Origin for the type, group, and function dependent fixed length parameters | |

TYPE 5 FMH – The Function Management Header for an ATTACH

| Offset | Length | Name | Description |
|---|---|---|---|
| | | | FMHGROUP values set as follows: |
| | | | Value Function |
| | | FMHT5ATT | X'02' Group is ATTACH |
| | | | FMHFN values set as follows: |
| | | FMHATTFN | X'02' Function is ATTACH |
| 6 | 1 | FMHATSEC | Security algorithm value |
| 7 | 1 | FMHATDBA | Data algorithm value |
| | | | Value Function |
| | | FMHAU | X'00' Undefined |
| | | FMHAV | X'01' Variable length |
| | | FMHASCSD | X'02' Document subset of SCS |
| | | FMHASCSC | X'03' Card subset of SCS |
| | | FMHARUC | X'04' Chain of request units |
| | | FMHARU | X'05' Request unit |
| CD | | FMHATDPN | Process to be initiated |
| CD | | FMHATDPL | Process name length |
| | | | Value Function |
| | | FMHARLEN | 1 Length of an architected process name |
| 1 | 4 | FMHATDPV | Process name up to four characters |
| | | | Value Function |
| | | FMHARMAX | X'3F' Maximum possible value for an architected process name (defined as non-graphic values) |
| CD | | FMHATPRN | Resource for initiated process |
| CD | | FMHARDPN | Return process name |
| CD | | FMHARPRN | Resource for return process |
| CD | | FMHATDQN | Queue to be associated with initiated process |

CICS PRIVATE (TYPE X'43') FMH – The Function Management Header for a CICS request or reply. Since this is a private FMH, the direction of transmission determines whether it represents a request or a reply.

| Offset | Length | Name | Description |
|---|---|---|---|
| CD | | FMHCOPTS | For outbound requests – the existence and TCA bits from ARGO |
| CD | | FMHCRCDE | For inbound replies – the error codes from EIBRCODES |

The data that follows this FMH represents the argument(s) to an EIP call that has to be serviced remotely.

Not all arguments will be sent; furthermore the value that is sent will depend on the CICS function and on the direction of transmission.

| Offset | Length | Name | Description |
|---|---|---|---|
| CD | | FMHCARGL | Length of parameter; includes length and ARGNO fields |
| 2 | 1 | FMHCARGN | Argument number; ARG3 is represented by value X'06' |
| 3 | 256 | FMHCARGV | The argument itself, it may be, for example, a key |

TYPE 6 FMH – The Function Management Header for a SCHEDULE request

| | | | |
|---|---|---|---|
| 6 | 1 | FMHSRQST | Details of sched request.<br>Bits set as follows:<br>Value Function |
| | | FMHSTIME | X'80' Time delay specified<br>Bits 1-7 reserved |
| CD | | FMHSSDPN | Name of process that is to be initiated |
| CD | | FMHSPRN | Name of primary resource for process being initiated |
| CD | | FMHSRDPN | Suggested name for return process |
| CD | | FMHSRPRN | Suggested name for primary resource for return process |
| CD | | FMHSQNME | Name of queue associated with process being initiated |
| CD | | FMHSREQN | Name of request instance associated with process |
| CD | | FMHSDELY | The interval or time initiation delay field |

The Function Management Header for a SCHEDULE STATUS Reply

| | | | |
|---|---|---|---|
| 6 | 1 | FMHSSSTS | Status of schedule request<br>Bits set as follows:<br>Bits 0-3 reserved<br>Value Function |
| | | FMHSIDPN | X'08' Invalid process name |
| | | FMHSIPRN | X'04' Invalid resource name |
| | | FMHSERR | X'02' Unable to schedule due to processing error |
| | | FMHSINV | X'01' Invalid request |
| CD | | FMHSIREQ | Request name generated by receiving system |

The Function Management Header for a PURGREQ request

| | | | |
|---|---|---|---|
| CD | | FMHSREQN | |
| CD | | FMHSCDPN | Name of process that is to be cancelled |

| Offset | Length | Name | Description |
|---|---|---|---|
| The Function Management Header for a PURGSTAT reply | | | |
| 6 | 1 | FMHSPSTS | Status of purge request<br>Bits set as follows:<br>Bits 0-6 reserved<br>Value Function |
| | | FMHSNFD | X'01' Named request not found |
| The Function Management Header for a QPUT request | | | |
| 6 | 1 | FMHQQORG | Type of queue<br>Value Function |
| | | FMHQNSPE | X'00' Queue type not specified |
| | | FMHQSEQL | X'01' Queue type is sequential |
| | | FMHQLINE | X'02' Queue type is linear |
| | | FMHQHIER | X'03' Queue type is hierarchical |
| CD | | FMHQNAME | The queue name is from 1 to 8 characters |
| The Function Management Header for a QGET request | | | |
| 6 | 1 | | FMHQQORG |
| CD | | | FMHQNAME |
| CD | | FMHQCURS | The cursor is held as two byte binary |
| CD | | FMHQTRSZ | The maximum record length is held as two byte binary |
| The Function Management Header for a QPURGE request | | | |
| 6 | 1 | | FMHQQORG |
| CD | | | FMHQNAME |
| The Function Management Header for a QXFR reply | | | |
| 7 | 1 | FMHQXFST | Status byte<br>Bits set as follows:<br>Bits 0-4 reserved<br>Value Function |
| | | FMHQDISP | X'04' Disposition of queue<br>Bit 6 reserved |
| | | FMHQEMSG | X'01' End of message |
| CD | | | FMHQCURS |
| CD | | FMHQRCNT | Number of occurrences of records at lowest level of cursor |
| CD | | FMHQRCLN | Record length before truncation |
| The Function Management Header for a QSTATUS reply | | | |
| 6 | 1 | | FMHQQORG |
| 7 | 2 | FMHQSTAT | Status of request |

| Offset | Length | Name | Description | | |
|---|---|---|---|---|---|
| 7 | 1 | FMHQSTA1 | First status byte | | |
| | | | | Value | Function |
| | | FMHQINVL | | X'80' | Invalid length for request |
| | | FMHQINVN | | X'40' | Invalid queue name |
| | | FMHQNAUT | | X'20' | Unauthorized request |
| | | FMHQNAVL | | X'10' | Queue name not available |
| | | FMHQSPAC | | X'08' | No space left on queue |
| | | FMHQINVC | | X'04' | Invalid cursor |
| | | FMHQERRO | | x'02' | I/O error when queue accessed |
| | | FMHQEMPT | | X'01' | Queue is empty |
| 8 | 1 | FMHQSTA2 | Reserved | | |
| CD | | | FMHQCURS | | |
| CD | | FMHQSENS | Sense data (could be access method data) | | |
| CD | | | FMHQNAME | | |

The Function Management Header for a QREPL request

| | | | |
|---|---|---|---|
| 6 | 1 | | FMHQQORG |
| CD | | | FMHQNAME |
| CD | | | FMHQCURS |

The Function Management Header for a QGETN request

| | | | |
|---|---|---|---|
| 6 | 1 | | FMHQQORG |
| CD | | | FMHQNAME |
| CD | | | FMHQTRSZ |

The Function Management Header for a DL/I SCHEDULE request

| | | | |
|---|---|---|---|
| CD | | FMHDPSBN | The PSB name is from 1 to 8 characters |

The Function Management Header for a DL/I schedule reply

| | | | | | |
|---|---|---|---|---|---|
| 6 | 2 | FMHDSRCS | DL/I return codes | | |
| 6 | 1 | FMHDSRC1 | DL/I return code with bits set as follows | | |
| | | | | Value | Function |
| | | FMHDNOPN | | X'80' | Data base not open |
| | | FMHDNFND | | X'40' | PSB not found |
| | | FMHDNACT | | X'20' | DL/I not active |
| | | FMHDFAIL | | X'10' | PSB initialization failed |
| | | FMHDNAUT | | X'08' | Unauthorized access to PSB |
| | | FMHDCONF | | X'04' | Intent schedule conflict |
| | | | | Bit 6 and Bit 7 reserved | |
| 7 | 1 | FMHDSRC2 | Reserved | | |

The data following this and other FMHs for DL/I, only one I/O area or key can follow a FMH; however, several SSAs or PCBs can follow a FMH.

| Offset | Length | Name | Description |
|---|---|---|---|
| CD | | FMHDLENG | Length of parameter; includes length and type fields |
| 2 | 1 | FMHDTYPE | Parameter type – values set as follows<br>Value Function |
| | | FMHDIOA | X'01' Flag set to show that parameter is an I/O area |
| | | FMHDSSA | X'02' Flag set to show that parameter is a SSA |
| | | FMHDPCB | X'03' Flag set to show that parameter is a PCB |
| | | FMHDKEY | X'04' Flag set to show that parameter is a key |
| 3 | 256 | FMHDPARM | The parameter itself; it may be for example, a PCB |
| 3 | 256 | FMHDAREA | The I/O area – maximum length rather than actual length |
| 3 | 256 | FMHDPSSA | The segment search argument – maximum length rather than actual length |
| 3 | 4 | FMHDNTNT | Processing intent for this data base |
| 7 | 4 | FMHDMKYL | Maximum key length for this PCB (binary) |
| B | 4 | FMHDSEGS | Number of sensitive segments (binary) |
| CD | | FMHDDBDN | The DBDNAME is from 1 to 8 characters |
| CD | | FMHDSAMX | The maximum possible size for a SAA |
| CD | | FMHDIOMX | The maximum possible size for an I/O area |
| 3 | 256 | FMHDPKEY | The fully concatenated key for this operation |

The Function Management Header for a DL/I REPL request

| | | | |
|---|---|---|---|
| 6 | 2 | FMHDPCBI | The index for this PCB |

The Function Management Header for a DL/I ISRT request

| | | | |
|---|---|---|---|
| 6 | 3 | | FMHDPCBI |

The Function Management Header for a DL/I DLET request

| | | | |
|---|---|---|---|
| 6 | 3 | | FMHDPCBI |

The Function Managament Header for a DL/I GU request

| | | | |
|---|---|---|---|
| 6 | 3 | | FMHDPCBI |

| Offset | Length | Name | Description |
|---|---|---|---|
| The Function Managament Header for a DL/I GHU request | | | |
| 6 | 3 | | FMHDPCBI |
| The Function Management Header for a DL/I GN request | | | |
| 6 | 3 | | FMHDPCBI |
| The Function Management Header for a DL/I GNH request | | | |
| 6 | 3 | | FMHDPCBI |
| The Function Management Header for a DL/I GNP request | | | |
| 6 | 3 | | FMHDPCBI |
| The Function Management Header for a DL/I GHNP request | | | |
| 6 | 3 | | FMHDPCBI |
| The Function Management Header for a DL/I DLIDBXFR reply | | | |
| 6 | 2 | FMHDRCDS | DL/I return codes |
| 6 | 1 | FMHDRCD1 | DL/I return code with bits set as follows<br>Value function |
| | | FMHDNOPN | X'80' Data base not open<br>Bits 1-4 reserved |
| | | FMHDNVRQ | X'04' Invalid PCB index<br>Bits 6 and 7 reserved |
| 7 | 1 | FMHDRCD2 | Reserved |
| 8 | 2 | FMHDSEGL | Segment level (binary) |
| A | 2 | FMHDSTCD | Status codes |
| CD | | FMHDSEGN | The segment name is from 1 to 8 characters |
| The Function Management Header for a DL/I DLIDBSTS reply | | | |
| 6 | 2 | | FMHDRCDS |
| 6 | 1 | | FMHDRCD1 |
| 7 | 1 | | FMHDRCD2 |
| 8 | 2 | | FMHDSEGL |
| A | 2 | | FMHDSTCD |
| CD | | | FMHDSEGN |

| Offset | Length | Name | Description |
|---|---|---|---|
| The Function Management Header for a DL/I DLIDEQ request | | | |
| 6 | 2 | FMHDLOCK | LOCK class assigned to segments by prior retrieval or insertion request |
| The Function Management Header for a DL/I DLIDEQR reply | | | |
| 6 | 1 | FMHDDRCD | Status of request with bits set as follows<br>Bits 0-7 reserved |
| 7 | 2 | FMHDDSTS | Status code |
| TYPE 7 FMH | | | |
| 6 | 2 | FMHSMNUM | Message number |
| 8 | 4 | FMHSMCCD | CICS abend code |
| C | 5 | FMHSMDCD | DL/I abend code |

## FUNCTION AND MODULE IDENTIFIERS

DSECT NAME: DFHFMIDS

This control block is a collection of symbolic labels used in connection with the following DSECTs: DFHDWEDS, DFHJCADS, and DFHJCR. It does not overlay an area of storage, as does a typical DSECT, but rather provides symbolic labels equated to constants, for reference in Assembler-language instructions, such as MVI and CLI. The symbolic labels are listed in groups as function identifiers and module identifiers below, first in alphabetic order, then in a logical order according to their use.

## FUNCTION IDENTIFIERS

### ALPHABETIC REFERENCE TO FUNCTION IDENTIFIERS

| Name | Value |
|---|---|
| FIDAJRN | X'20' |
| FIDAKE | X'81' |
| FIDAKS | X'80' |
| FIDALOG | X'40' |
| FIDBEOSP | X'F6' |
| FIDBEOTK | X'F4' |
| FIDBMPM | X'F1' |
| FIDDBCHN | X'81' |
| FIDDBOFL | X'80' |
| FIDFCRO | X'80' |
| FIDFCRU | X'81' |
| FIDFCWA | X'83' |
| FIDFCWU | X'82' |
| FIDICDB | X'08' |
| FIDICPUT | X'50' |
| FIDJCLAB | X'80' |
| FIDJCLOK | X'81' |
| FIDKPDCT | X'83' |
| FIDKPTCA | X'82' |
| FIDKPTCR | X'84' |
| FIDKPTST | X'85' |
| FIDLEOTK | X'F2' |
| FIDLSOSP | X'F1' |
| FIDPEOTK | X'F3' |
| FIDSPR | X'F5' |
| FIDTCAJ | X'20' |
| FIDTCAL | X'40' |
| FIDTCDB | X'08' |
| FIDTCDIP | X'04' |
| FIDTCDWL | X'01' |
| FIDTCEIB | X'8A' |
| FIDTCFMH | X'03' |
| FIDTCIM | X'81' |
| FIDTCIMF | X'85' |
| FIDTCML | X'F0' |
| FIDTCOM | X'82' |
| FIDTCOMN | X'86' |
| FIDTCON | X'87' |
| FIDTCONN | X'88' |

## TRANSIENT DATA OUTPUT AREA (TDOA)

DSECT NAME: DFHTDOA
REGISTER: TDOABAR

The TDOA is used to describe an output area, acquired via a Storage Control request, from which intrapartition transient data is moved by a PUT macro instruction.

ALPHABETIC INDEX TO FIELD NAMES

| Name | Offset or Reference |
|---|---|
| TDOADBA | C |
| TDOASAL | 2 |
| TDOASCA | 4 |
| TDOAVRL | 8 |

STORAGE LAYOUT

| Offset | Length | Name | Description |
|---|---|---|---|
| 0 | 2 | | Storage accounting information; storage class=Transient Data (see "Storage Accounting Area (SAA)") |
| 2 | 2 | TDOASAL | Storage accounting area length |
| 4 | 4 | TDOASCA | Transaction storage chain address |
| 8 | 2 | TDOAVRL | Variable record length |
| A | 2 | | Reserved |
| C | Var | TDOADBA | Beginning of data area |

## TRANSIENT DATA - VSAM CONTROL INTERVAL MAP

DSECT NAME: DFHTDCI

The TDCI is a map of the VSAM control interval as used by VSAM transient data.

### ALPHABETIC INDEX TO FIELD NAMES

| Name | Offset or Reference |
|---|---|
| TDCF | 0 |
| TDCHBC | 4 |
| TDCHDI | 0 |
| TDCHFC | 8 |
| TDCIDF | 0 |
| TDDATED | C |
| TDID | 0 |
| TDLENREC | 1 |
| TDLUS | 2 |
| TDNUMCI | A |
| TDNUMREC | 1 |
| TDOUS | 0 |
| TDRDF | 0 |
| TDRESRV | 10 |

### STORAGE LAYOUT

| Offset | Length | Name | Description |
|---|---|---|---|
| Map of first control interval | | | |
| 0 | 10 | TDID | ID to be checked when restarting |
| A | 2 | TDNUMCI | Number of control intervals used to size control interval bit map |
| C | 4 | TDDATED | Date information copied from CSAJYDP |
| 10 | 12 | TDRESRV | Reserved (not used) |

Map of chain records at start of subsequent control intervals

| Offset | Length | Name | Description |
|---|---|---|---|
| 0 | 4 | TDCHDI | Chain record destination ID |
| 4 | 4 | TDCHBC | Chain record backward chain |
| 8 | 4 | TDCHFC | Chain record forward chain |

## DATA RECORDS AND FREE SPACE

These fields are positioned at negative offsets relative to the end of the control interval

### RECORD DEFINITION FIELD

| Offset | Length | Name | Description | |
|---|---|---|---|---|
| 0 | 3 | TDRDF | Record definition field | |
| 0 | 1 | TDCF | Control field (flag byte) | |
| | | | Value | Function |
| | | TDRSINGL | X'00' | Right record definition field (RDF) gives length of single record. No left RDF |
| | | TDRLPAIR | X'40' | Right RDF says paired left RDF exists |
| | | TDLNUM | X'08' | Left RDF gives number of records in TDNUMREC |
| 1 | 2 | TDNUMREC | Number of records of same length | |
| 1 | 2 | TDLENREC | Length of record (s) | |

### CONTROL INTERVAL DEFINITION FIELD

| Offset | Length | Name | Description |
|---|---|---|---|
| 0 | 4 | TDCIDF | Control interval definition field |
| 0 | 2 | TDOUS | Offset of unused space |
| 2 | 2 | TDLUS | Length of unused space |

## TEMPORARY STORAGE TABLE

DSECT NAME: DFHTSTDS

The Temporary Storage Table (TST) is a list of generic mnemonics used (1) to identify temporary storage DATAIDs for which CICS/VS is to provide recoverability in the event of abnormal termination of CICS/VS and subsequent emergency restart (2) to identify DATAIDs on a remote system. Each entry in the table specifies the leading characters of user-defined DATAIDs for which CICS/VS will provide protection (enqueuing) during a logical unit of work by an applciation program and automatic logging of the status of the data at task termintion (or sync point).

ALPHABETIC INDEX TO FIELD NAMES

| Name | Offset or Reference |
|---|---|
| TSTFL | 2 |
| TSTLL | 0 |
| TSTPL | 3 |
| TSTPRFX | 4 |
| TSTRPFX | F |
| TSTSYS | C |

STORAGE LAYOUT

COMMON PART

| Offset | Length | Name | Description | |
|---|---|---|---|---|
| 0 | 2 | TSTLL | Length of entry | |
| 2 | 1 | TSTFL | Flag describing entry | |
| | | | Value | Function |
| | | TSTRCVBM | X'80' | Recoverable |
| | | TSTRMTBM | X'40' | Remote |
| | | TSTRNMBM | X'20' | Remote prefix given |
| | | TSTLSTBM | X'01' | =1 for last entry |
| 3 | 1 | TSTPL | Prefix length -1 | |
| 4 | 0 | TSTPRFX | Prefix | |
| 4 | 4 | | First four bytes | |
| 8 | 4 | | Last four - included only when prefix greater than four bytes, or remote | |

IBM Technical Newsletter

| | |
|---|---|
| This Newsletter No. | LN33-6253 |
| Date | December 1978 |
| Base Publication No. | LY33-6029-1 |
| Previous Newsletters | None |

**Customer Information Control**
**System/Virtual Storage (CICS/VS)**
**Version 1, Release 4**
**Program Logic (OS/VS)**

This technical newsletter, a part of Version 1, Release 4 of Customer Information Control System/Virtual Storage (CICS/VS), provides replacement pages for the subject publication. These replacement pages remain in effect for subsequent versions and releases unless specifically altered. Pages to be inserted or removed are:

927, 928
965, 966
999, 1000
1201, 1202
1346.1, 1346.2 (added)

A change to the text or to an illustration is indicated by a vertical line to the left of the change.

**Summary of Amendments**

The main change introduced by this TNL is the addition of the Transient Data VSAM Control Interval Map in the Data Areas section of the manual.

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

**IBM United Kingdom Laboratories Ltd, Technical Documentation Department, Hursley Park, Winchester, Hampshire SO21 2JN, England.**

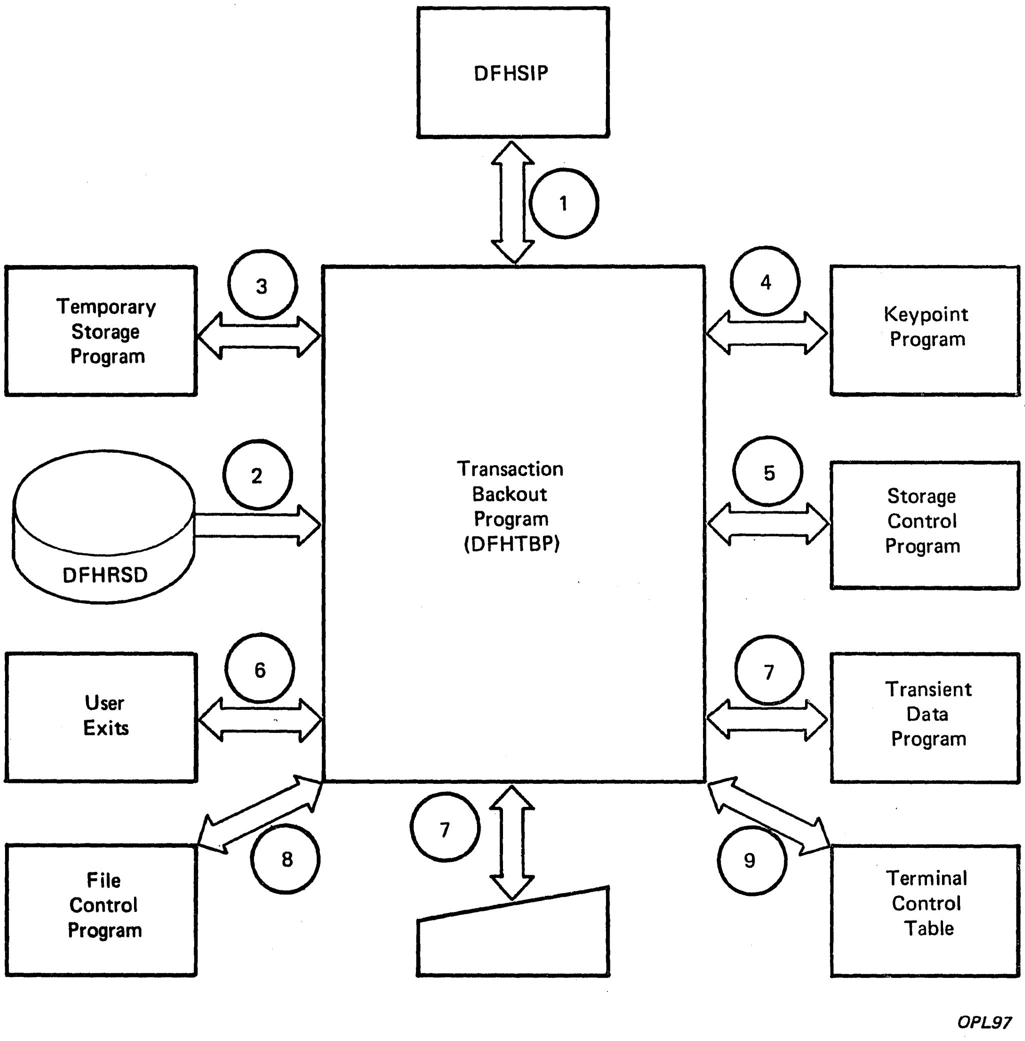


Figure 3-53. Transaction Backout Program Interfaces

## TRANSIENT DATA MANAGEMENT (DFHTDP)

The Transient Data Control program (DFHTDP) communicates directly with three other functions of the CICS/VS System Management component. These are Task Management, Storage Management, and Program Management (the Task Control program, Storage Control program, and Program Control program, respectively). The application program communicates with DFHTDP through use of DFHTD macro instructions.

The notes below refer to Figure 3-54.

Notes:

### Intrapartition

1. An application program request for Transient Data services causes a request code to be set in TCATDTR, signifying GET, PUT, PURGE, FEOV, LOCATE, or CHECK. The destination identification is placed in TCATDDI. Control is passed to the Transient Data Control program.

2. Transient Data Control obtains intrapartition disk map information from the main storage area addressed through the Destination Control Table (DCT).

3. The field TDDCTDID is used to search the DCT for the destination requested. The Data Control Block (DCB) in the DCT is checked for OPEN/CLOSE when the reference is to an extrapartition destination.

4. Task Control is used to wait for access to the DCT and to attach an automatically initiated task not associated with a terminal. If a task is associated with a terminal, an Automatic Initiate Descriptor (AID) is scheduled.

5. Transient Data Control communicates with Storage Control to obtain storage for the DECB for an intrapartition data set and for intrapartition I/O areas.

6. Transient Data passes control and the address of the DECB to the read/write routines of the access methods to perform data set I/O.

7. The DECB status bytes are tested for event completion and I/O errors.

8. If an invalid request for Transient Data services is received, control is passed to Program Control to terminate the task issuing the request.

9. If restart/recovery is supported within the system, a Deferred Work Element (DWE) is created for each logically recoverable destination when it is accessed. At the end of a logical unit of work (LUW), defined by either an application program DFHSP TYPE=USER request or by Task Control at task termination, the CICS/VS Sync Point program gives control to the DWE processor of the Transient Data Control program. This DWE processor performs the logical update of the DCT entry and/or the track (or control interval-VSAM) bit map.

| Offset | Length | Name | Description |
|---|---|---|---|
| 98 | 4 | CSAPAMA | Address of Page Allocation Map (PAM) |
| 9C | 1 | CSAOPSYS | Operating system |
| | | | Value  Function |
| | | CSADOS | C'D'  DOS/VS |
| | | CSAVS1 | C'1'  OS/VS1 |
| | | CSASVS | C'S'  OS/SVS |
| | | CSAMVS | C'M'  OS/MVS |
| 9D | 1 | CSAOPREL | Operating system release |
| | | | Value  Function |
| | | CSAD34 | X'34'  DOS/VS Release 34 |
| | | CSA160 | X'60'  OS/VS1 Release 6.0 |
| | | CSAS17 | X'17'  OS/SVS Release 1.7 |
| | | CSAM37 | X'37'  OS/MVS Release 3.7 |
| 9E | 1 | CSACICS | CICS/VS system |
| | | | Value  Function |
| | | CSAELS | C'E'  Entry Level System |
| | | CSAFULL | C'F'  Full CICS/VS |
| 9F | 1 | CSACIREL | CICS/VS Release |
| | | | Value  Function |
| | | CSAC14 | X'14'  Version 1, Release 4 |
| A0 | 4 | | Zero. Dummy suspended task DCA has DCATRTO set to 0. |
| A4 | 4 | CSANXDCA | Next available Dispatch Control Area (DCA) |
| A8 | 4 | CSASUSFA | Lowest priority suspended task DCA |
| AC | 4 | CSASUSBA | Highest priority suspended task DCA |
| B0 | 4 | CSAACTFA | Lowest priority active task DCA |
| B4 | 4 | CSAACTBA | Highest priority active task DCA |
| B8 | 4 | CSATCTCA | Terminal Control TCA address |
| BC | 4 | | Dummy dispatch, control, priority, etc |
| C0 | 2 | CSAMAXT | Maximum active task value |
| C2 | 2 | CSAMAXTC | Maximum active task accumulate |
| C4 | 2 | | Reserved |
| C6 | 2 | CSAWAL | Length of (CSA + workarea + system ID) |
| C8 | 4 | CSAOPFLA | CSA Optional Features List address |
| CC | 2 | CSAKPFRQ | Activity keypoint frequency count |
| CE | 2 | CSAKPACT | Activity keypoint attach counter |

| Offset | Length | Name | Description |
|---|---|---|---|
| D0 | 8 | CSAKPCLK | Activity keypoint store clock |
| D8 | 4 | | Reserved |
| DC | 4 | | Memory constant |
| E0 | 4 | CSAKCNAC | Task Control entry address |
| E4 | 4 | CSASCNAC | Storage Control entry address<br>Value Function |
| | | CSASCPXM | X'80' Storage Control program—check exit active |
| E8 | 4 | CSAPCNAC | Program Control entry address |
| EC | 4 | CSAICNAC | Interval Control entry address |
| F0 | 4 | CSADCNAC | Dump Control entry address |
| F4 | 4 | CSATCNAC | Terminal Control entry address |
| F8 | 4 | CSAFCNAC | File Control entry address |
| FC | 4 | CSATDNAC | Transient Data Control entry address |
| 100 | 4 | CSATSNAC | Temporary Storage Control entry address |
| 104 | 4 | CSASANAC | PL/I Storage Allocation module entry |
| 108 | 4 | CSATRNAC | Trace Control entry address |
| 10C | 4 | CSAPIPSW | Program Interrupt PSW save area address, or |
| | 4 | CSAPINAC | Program Interrupt entry address (during System Initialization only) |
| 110 | 4 | CSASNNAC | Snapshot program entry address |
| 114 | 4 | CSASPNAC | Sync Point program entry address |
| 118 | 4 | CSATCRWE | Terminal Control Program Read/Write request program entry |
| 11C | 4 | CSATRTBA | Trace Table beginning address |
| 120 | 4 | CSAPCTBA | Program Control Table beginning address |
| 124 | 4 | CSAPPTBA | Processing Program Table beginning address |
| 128 | 4 | CSATCTBA | Terminal Control Table Prefix address |
| 12C | 4 | CSAFCTBA | File Control Table beginning address |

| Name | Offset or Reference | Name | Offset or Reference |
|---|---|---|---|
| SITFDPSF | F2 | SITPLTPI | AC |
| SITFLDSP | 60 | SITPLTSD | AE |
| SITFLDST | 64 | SITPL1 | 6C |
| SITHPO | 72 | | |
| SITICPOP | 121 | SITPPTOP | 11E |
| SITICPSF | CA | SITPPTSF | B0 |
| SITICVAL | 28 | SITPRINT | 6A |
| SITIIPSF | E8 | SITRICVL | 2C |
| SITIMODA | 18 | SITRLRSF | E0 |
| SITIND | 11A | | |
| SITIOCP | 34 | SITSCPSF | C4 |
| SITIRCS | 71 | | |
| SITISCSF | FA | SITSCSZ | 3C |
| SITISPSF | 100 | | |
| SITJCPSF | DA | SITSICVL | 30 |
| SITJCTSF | A6 | | |

STORAGE LAYOUT

| Offset | Length | Name | Description | |
|---|---|---|---|---|
| 0 | 8 | DFHSITBA | Table entry point | |

## OPERATING SYSTEM AND CICS/VS LEVELS

| Offset | Length | Name | Description | |
|---|---|---|---|---|
| 0 | 1 | SITOPSYS | Operating system | |
| | | | Value | Function |
| | | SITDOS | C'D' | DOS/VS |
| | | SITVS1 | C'1' | OS/VS1 |
| | | SITSVS | C'S' | OS/SVS |
| | | SITMVS | C'M' | OS/MVS |
| 1 | 1 | SITOPREL | Operating system release | |
| | | | Value | Function |
| | | SITD34 | X'34' | DOS/VS Release 34 |
| | | SIT160 | X'60' | OS/VS1 Release 6.0 |
| | | SITS17 | X'17' | OS/SVS Release 1.7 |
| | | SITM37 | X'37' | OS/MVS Release 3.7 |
| 2 | 1 | SITCICS | CICS/VS system | |
| | | | Value | Function |
| | | SITFULL | C'F' | Full CICS/VS |
| 3 | 1 | SITCIREL | CICS/VS release | |
| | | | Value | Function |
| | | SITC14 | X'14' | Version 1, Release 4 |

| Offset | Length | Name | Description |
|---|---|---|---|
| LENGTHS OF SIT AND CWA | | | |
| 4 | 2 | SITLEN | Length of SIT |
| 6 | 2 | SITCWA | Required CWA size |
| ADDRESS CONSTANTS | | | |
| 8 | 4 | DFHDLL | Address of DL/I link list |
| C | 4 | DFHAPT | Address of ATP CSA extension |
| 10 | 4 | SITCOMA | Reserved |
| 14 | 4 | SITOVRPM | Address of override parameters |
| 18 | 4 | SITIMODA | Initialization module suffix list |
| 1C | 4 | SITSRPAE | Reserved |
| 20 | 8 | | Reserved |
| TIME CONTROL VALUES | | | |
| 28 | 4 | SITICVAL | System time interval |
| 2C | 4 | SITRICVL | Runaway task time interval |
| 30 | 2 | SITSICVL | Stall time interval |
| 32 | 2 | SITTSDTI | Terminal scan delay |
| 34 | 2 | SITIOCP | Wait count percentage |
| 36 | 2 | | Reserved |
| MISCELLANEOUS SIZES AND COUNTERS | | | |
| 38 | 4 | SITOSCOR | Address space reserved for OS |
| 3C | 4 | SITSCSZ | Storage cushion size |
| 40 | 4 | SITPGSIZ | Pagesize |
| 44 | 4 | SITTRTSZ | Trace table number of entries |
| 48 | 2 | SITTSMSG | Number of messages per ID or group |

## DATA RECORDS AND FREE SPACE

These fields are positioned at negative offsets relative to the end of the control interval

### RECORD DEFINITION FIELD

| Offset | Length | Name | Description | | |
|---|---|---|---|---|---|
| 0 | 3 | TDRDF | Record definition field | | |
| 0 | 1 | TDCF | Control field (flag byte) | | |
| | | | | Value | Function |
| | | TDRSINGL | | X'00' | Right record definition field (RDF) gives length of single record. No left RDF |
| | | TDRLPAIR | | X'40' | Right RDF says paired left RDF exists |
| | | TDLNUM | | X'08' | Left RDF gives number of records in TDNUMREC |
| 1 | 2 | TDNUMREC | Number of records of same length | | |
| 1 | 2 | TDLENREC | Length of record(s) | | |

### CONTROL INTERVAL DEFINITION FIELD

| Offset | Length | Name | Description |
|---|---|---|---|
| 0 | 4 | TDCIDF | Control interval definition field |
| 0 | 2 | TDOUS | Offset of unused space |
| 2 | 2 | TDLUS | Length of unused space |

## TRANSIENT DATA - VSAM CONTROL INTERVAL MAP

DSECT NAME: DFHTDCI

The TDCI is a map of the VSAM control interval as used by VSAM transient data.

ALPHABETIC INDEX TO FIELD NAMES

| Name | Offset or Reference |
|---|---|
| TDCF | 0 |
| TDCHBC | 4 |
| TDCHDI | 0 |
| TDCHFC | 8 |
| TDCIDF | 0 |
| TDDATED | C |
| TDID | 0 |
| TDLENREC | 1 |
| TDLUS | 2 |
| TDNUMCI | A |
| TDNUMREC | 1 |
| TDOUS | 0 |
| TDRDF | 0 |
| TDRESRV | 10 |

STORAGE LAYOUT

Map of first control interval

| Offset | Length | Name | Description |
|---|---|---|---|
| 0 | 10 | TDID | ID to be checked when restarting |
| A | 2 | TDNUMCI | Number of control intervals used to size control interval bit map |
| C | 4 | TDDATED | Date information copied from CSAJYDP |
| 10 | 12 | TDRESRV | Reserved (not used) |

Map of chain records at start of subsequent control intervals

| Offset | Length | Name | Description |
|---|---|---|---|
| 0 | 4 | TDCHDI | Chain record destination ID |
| 4 | 4 | TDCHBC | Chain record backward chain |
| 8 | 4 | TDCHFC | Chain record forward chain |

LY33-6034-0

CICS/VS Program Logic: Description (OS/VS) Printed in U.S.A. LY33-6034-0

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