LY20-0714-2

Program Product

Customer Information Control System (CICS) Logic Manual

Program Number 5734-XX7 (OS-STANDARD V2)
Feature Number 8142

The IBM Customer Information Control System (CICS) is a transaction-oriented, multiapplication data base/data communication interface between a System/360 or System/370 operating system and user-written application programs. Applicable to most online systems, CICS provides many of the facilities necessary for standard terminal applications: message switching, inquiry, data collection, order entry, and conversational data entry.

CICS is available in three systems – two for DOS users and one for OS users. Because the two CICS/DOS systems are compatible with each other and with the CICS/OS system, it is possible to start with a small data base/data communication configuration and move up through DOS into OS.

The information contained in this manual is of interest to persons maintaining and modifying the operation of the CICS/OS–STANDARD V2 system.

IBM

PREFACE

This publication contains a detailed description of the logical structure of the CICS/OS-STANDARD V2 system and serves as a guide to the program listings. It provides system programmers with information needed to maintain and modify the operation of the system.

The "Methods of Operation" section contains information concerning the flowchart logic and is an expansion of the information available from the flowcharts and the program listings. The "Flowcharts" section contains flowcharts of the CICS management and service programs. These flowcharts include nonactive (dummy) labels that can be cross-referenced with the same labels contained in the program listings. The "Register Usage" section identifies the registers used and their content and function. The "Control Blocks" sections describe the contents of the three types of main storage areas (control areas, input/output areas, and work areas).

The words "transaction" and "task" have the same connotation in CICS and are used interchangeably throughout this publication; the processing of a transaction may involve the execution of one or more "programs".

For further information concerning the CICS/OS-STANDARD V2 system, see the following IBM publications:

General Information Manual (GH20-1028)
Application Programmer's Reference Manual (SH20-1047)
System Programmer's Reference Manual (SH20-1043)
Terminal Operator's Guide (SH20-1044)
Operations Guide (CICS/OS) (SH20-1048)

All references to CICS/OS and CICS/OS-STANDARD in this publication are references to the CICS/OS-STANDARD V2 system.

Third Edition (December 1972)

This edition is a major revision obsoleting LY20-0714-1.

This edition applies to Version 2, Modification Level 3, of the program product Customer Information Control System (CICS) (OS–STANDARD V2) (5734-XX7) and to all subsequent versions and modifications until otherwise indicated in new editions or Technical Newsletters.

Changes are continually made to the information herein. Therefore, before using this publication, consult the latest System/360 and System/370 SRL Newsletter (GN20-0360) for the editions that are applicable and current.

A form for readers' comments has been provided at the back of this publication. If this form has been removed, address comments to: IBM Corporation, Technical Publications Department, 1133 Westchester Avenue, White Plains, New York 10604. Comments become the property of IBM.

CONTENTS

INTRODUCTION

The IBM Customer Information Control System (CICS) is a multi-application data base/data communication interface between OS or DOS and user-written application programs. Applicable to most online systems, CICS provides many of the facilities for standard terminal applications: message switching, inquiry, data collection, order entry, and conversational data entry.

Functions performed by CICS include:

- Control of a mixed telecommunications network
- Concurrent management of a variety of programs
- Controlled access to the data base
- Management of resources for continous operation
- Prioritization of processing

By eliminating many of the development requirements for such functions of a real-time control system, CICS allows programmers to concentrate instead on implementing applications, dramatically reducing implementation time and cost.

Functions needed to support a data base/data communication system and standard terminal applications are provided by the following CICS management programs:

- Task Management - Provides the dynamic multitasking facilities necessary for effective, concurrent transaction processing. Functions associated with this facility include priority scheduling, transaction synchronization, and control of serially reusable resources.

- Storage Management - Controls main storage allocated to CICS. Storage acquisition, disposition, initialization, and request queuing are among the services and functions performed by this component of CICS.

- Program Management - Provides a multiprogramming capability through dynamic program management while offering a real-time program fetch capability.

- Program Interrupt Management - Provides for the interception of program interrupts by CICS to prevent total system termination. Individual transactions that program check are terminated by CICS with a dump (if Dump Management is used), thus preventing the entire CICS partition/region from terminating.

- Time Management - Provides control of various optional task functions (system stall detection, runaway task control, task synchronization, etc.) based on specified intervals of time or the time of day.

- Dump Management - Provides a facility to assist in analysis of programs and transactions undergoing development or modification. Specified areas of main storage are dumped onto a sequential data set, either tape or disk, for subsequent offline formatting and printing using a CICS utility program.

- Terminal Management - Provides polling according to user-specified line traffic control as well as user requested reading and writing. This facility supports automatic task initiation to process new transactions. The testing of application programs is accommodated by the simulation of terminals through sequential devices such as card readers, line printers, disk, tape, etc.

- File Management - Provides a data base facility using direct access and indexed sequential data management. This function supports updates, additions, random retrieval, and sequential retrieval (browsing) of logical data on the data base.
- Transient Data Management - Provides the optional queuing facility for the management of data in transit to and from user defined destinations. This function facilitates message switching, data collection, and logging.
- Temporary Storage Management - Provides the optional general purpose "scratch pad" facility. This facility is intended for video display paging, broadcasting, data collection suspension, conservation of main storage, retention of control information, etc.

In addition to these management functions, CICS provides system service programming to identify terminal operators, to give dynamic control of the entire system to a master terminal, to display real-time system statistics, to intercept abnormal conditions not handled directly by the operating system, and to end operation by gathering summary statistics, closing data sets, and returning control to the operating system.

METHODS OF OPERATION

SYSTEM INITIALIZATION PROGRAM (DFHSIP)

The System Initialization program is responsible for readying CICS for communication and inquiry by the user. It is invoked as an OS job by user job control statements.

The System Initialization program is a non-real-time component of CICS and is resident only long enough to start up CICS. The start-up sequence is as follows:

1. Acquire CICS nucleus storage
2. Parameter initialization
3. CICS nucleus build
4. Open and format CICS system data sets
5. Open user data sets
6. Acquire and build CICS dynamic storage pool
7. Load resident application modules
8. Issue SPIE and Start Time macros
9. Transfer control to CICS

Storage organization is depicted in the following charts. The first chart shows storage organization during initialization but prior to the opening of system and user data sets. The second chart shows storage organization during the real-time execution of CICS.

```
                 D U R I N G   I N I T I A L I Z A T I O N
High     ***************************************************
Storage  *                                                 *
         *                                                 *
         *                                                 *
         ***************************************************
         *                                                 *
         *  STORAGE ACQUIRED BY DFHSIP TO LOAD NUCLEUS     *  10K-190K
         *                                                 *
         ***************************************************
         *                                                 *
         *                                                 *
         *   STORAGE RETURNED TO THE OPERATING SYSTEM      *  10K
         *                                                 *
         *                                                 *
         ***************************************************
         *                                                 *
         *     SYSTEM INITIALIZATION PROGRAM (DFHSIP)      *  8K
Low      *                                                 *
Storage  ***************************************************
```

```
            D U R I N G   R E A L - T I M E   E X E C U T I O N
High        **************************************************
Storage     *                                                *
            *               OPERATING SYSTEM                 *
            *                                                *
            **************************************************
            *                                                *
            *                  CICS NUCLEUS                  *
            *                        (Includes Trace Table)  *
            *                                                *
            **************************************************
            *                                                *
            *             REQUIRED ACCESS METHODS            *
            *                                                *
            **************************************************
            *                                                *
            *        TEMPORARY AUXILIARY STORAGE TABLE       *
            *                                                *
            **************************************************
            *                                                *
            *          RESIDENT MODULES (If defined)         *
            *                                                *
            *------------------------------------------------*
            *                                                *
            *              DYNAMIC STORAGE POOL              *
            *                                                *
            *************************                        *
            * SUBPOOL BOUNDARY BOX  *                        *
            **************************************************
            *                                                *
            *             OSCOR SPECIFIED BY USER            *
Low         *                                                *
Storage     **************************************************
```

ACQUIRE CICS NUCLEUS STORAGE

The System Initialization program, after establishing addressability, issues an OS GETMAIN to acquire storage required to build the CICS nucleus. A conditional GETMAIN is issued for 200K bytes of main storage. If the request is successful, 10K of this storage is given back to the operating system and System Initialization continues. If 200K is not available, System Initialization decrements this amount by 10K and issues the GETMAIN again. This procedure continues until the storage is acquired in which to build the nucleus.

PARAMETER INITIALIZATION

The parameter initialization procedure is begun with the loading of the System Initialization Table. Data passed to the System Initialization program through use of the PARM field is examined to determine whether the user desires to override the standard System Initialization Table name (DFHSIT). If an override is supplied, the one-or-two-alphameric character suffix is appended to the name DFHSIT, and that table is loaded. If none is supplied, the standard table is loaded.

After loading the requested System Initialization Table, the System Initialization program scans the data passed through use of the PARM field to determine whether the user desires to override any of the values specified in the System Initialization Table loaded. Invalid data passed in the PARM field is logged by the System Initialization program on the console.

A list of keywords allowed as overrides and their maximum values may be found in the CICS/OS Operations Guide.

After all overrides have been applied to the loaded System Initialization Table, System Initialization constructs a CICS nucleus name list. This is accomplished by appending suffixes, if any, from the System Initialization Table to the corresponding nucleus module name. Exceptions are cases where a program and table combination is optional. In this case, if the program table module is optional and NO is supplied as the suffix, no table is loaded and the suffix DY is appended to the name of the optional nucleus module to cause a dummy module to be loaded.

CICS NUCLEUS BUILD

After the nucleus name list is built, the Processing Program Table (PPT) is loaded. The PPT is then scanned, and an OS BLDL is issued for each entry defined in the PPT. If the entry is found in the CICS Real-Time Relocatable Program Library, the TTRC, program size, and RLD are moved from the BLDL list to the PPT entry. If an entry in the PPT is not found, this condition is logged on the console preceded by the message 'DFH1596A APPLICATION MODULES NOT LOCATED'. The entire PPT is scanned in this fashion, and if application modules are not located, they are logged on the console.

At the completion of the PPT scan, the operator is given the choice to continue CICS initialization or to cancel. The Program Control Table (PCT) is then loaded. A check is made to determine whether any application modules were not located. If this is the case, the PPT is scanned to determine which applicaton modules were not located. Transaction codes that use modules not located are deleted in the loaded PCT and are logged on the system console.

After the PPT has been scanned completely, the PCT is scanned to ensure that the user has included a PCT entry for the transaction code CSAC associated with the Abnormal Condition program. If the transaction code CSAC cannot be found in the PCT, System Initialization abends with a dump.

Upon completion of the PPT and PCT scan and verify operations, System Initialization continues to load the rest of the CICS nucleus modules. This is done by the System Initialization Program Loader routine. If any nucleus module specified is not located, System Initialization abends with the message 'DFH1596 MODNAME NOT LOCATED'.

The final nucleus module to be loaded is the Common System Area (CSA). After the CSA is loaded, module entry points are resolved in the CSA. Upon completion of initializing the CSA, System Initialization then initializes and allocates the Trace Table specified by the user. Storage not being used by the CICS nucleus is then returned to the operating system.

OPEN AND FORMAT CICS SYSTEM DATA SETS

At this point, CICS system data sets are opened and formatted. The data set for the CICS Real-Time Relocatable Program Library is opened. The Temporary Storage data set, if specified by the user, is opened using a DCB specified in the System Initialization program. The entire extent allocated to this data set is formatted based on the block size supplied by the user. The count of the number of blocks that will fit in the extent is saved to build the Temporary Auxiliary Storage Table. The DCB in the System Initialization program is then closed, and the DCB in the loaded Temporary Storage program is opened.

Next the Intrapartition Transient Data data set, if specified, is opened. System Initialization uses a DCB in the System Initialization program to originally open this data set and formats the data set resetting the capacity record (R0). When this operation is completed, the DCB in System Initialization is closed and the DCB for the intrapartition data set in the Transient Data program is included in the open list for extrapartition data sets specified in the Destination Control Table. At this time, all Transient Data data sets are opened.

OPEN USER DATA SETS

System Initialization establishes addressability to the Terminal Control Table open list and issues an OS OPEN SVC. After opening the terminal data sets, System Initialization scans the DCB's specified in the Terminal Control Table for open failures. If the "not open" bit is on in any DCB, the DD name is extracted and logged on the console; correspondingly, that line is placed out of service. After opening and checking the DCB's, the 7770 DCB Processor routine is entered to initialize all 7770 DCB's, upon returning from the 7770 DCB Processor. a test is made to determine if Graphics was specified in the Terminal Control Table. If so, System Initialization issues the OS SPAR macro instruction.

System Initialization then issues an OPEN for all data sets defined as "initial open" in the File Control Table. Upon completion of the opening of the user's data base data sets, a test is made to determine whether Dump Control is included in the nucleus. If so, the Dump Control data set is opened.

ATTACH DL/I SUBTASK

If DL/I support was requested, system initialization issues an OS attach to create a subtask to support the DL/I data base. A dispatching priority one less than the CICS mother task, is attached to the subtask. System initialization then waits on the task communications ECB in the CICS-DL/I interface module until DL/I initialization is completed. If the DL/I subtask abends, system initialization aborts the start-up.

ESTABLISH CICS DYNAMIC STORAGE POOL

Upon completion of the opening of all required data sets, System Initialization acquires all storage left in the partition/region by issuing an OS variable conditional GETMAIN. After acquiring this storage, System Initialization determines the amount of main storage to be given back to OS, based on the amount specified by the user in the System Initialization Table (using the OSCOR operand). The amount of storage given back to OS at this time is calculated based on the size of the System Initialization program (approximately 8K), and this amount is always returned to OS. If the user's request exceeds 8K, the amount specified by the user is returned to OS.

After releasing the correct amount of main storage, System Initialization tests to determine whether a Temporary Auxiliary Storage Table is required. If so, the size of the table is calculated using the block count from the Initialization routine; the Temporary Auxiliary Storage Table is built, starting at the highest available storage address, working down. Upon the completion of the build of this table, the Subpool Boundary Box is established and built for the CICS dynamic storage pool.

LOAD RESIDENT APPLICATION MODULES

The Processing Program Table (PPT) loaded by System Initialization is scanned to determine if the user desires any resident application modules. If so, System Initialization, using its Program Loader routine and the Program Control program DCB for the CICS Real-Time Relocatable Program Library, loads modules defined to be resident. These modules are loaded at the top end of the dynamic storage pool. The amount of storage available is then adjusted. If the amount of storage remaining in the pool is not sufficient to execute the online system, the System Initialization program abends with the message 'DFH1599 PARTITION/REGION SIZE INSUFFICIENT TO INITIALIZE CICS'. Upon completion of the loading of resident application modules, the rest of the CICS storage pool is initialized to binary zeros.

ISSUE SPIE AND START TIME MACROS

At this point, System Initialization tests to determine whether the Program Interrupt program (PIP) has been included in this execution of CICS. If so, System Initialization branches to initialization code in PIP which issues the OS SPIE macro instruction.

Next System Initialization issues a CICS macro instruction which branches to initialization code in the loaded Internal Control program. If a dummy Interval Control program was loaded, control is returned directly to the System Initialization program. If a dummy Interval Control program was not loaded, the Interval Control program issues an OS Start Time macro instruction; control is then returned to the System Initialization program.

TRANSFER CONTROL TO CICS

Prior to transferring control to CICS, System Initialization ensures that there is enough storage to support the storage cushion specified by the user. If there is not enough storage, System Initialization abends. If there is enough storage, the System Initialization program transfers control to the CICS Dummy program, using an OS XCTL. This is done so that storage encumbered by the System Initialization program will be released.

The Dummy program loads the base registers of the Terminal Control program and records the entry point of the Terminal Control program in register 14. It then branches to the Storage Control program to cause Storage Control to acquire the storage cushion. Storage Control, after acquiring the storage cushion, then gives control to the Terminal Control program to begin the polling of terminals.

SYSTEM INITIALIZATION SUBROUTINES

Three primary subroutines are used by the System Initialization program:

1. System Initialization Program Loader
2. Parameter Scan routine
3. Console Put routine

System Initialization Program Loader

The System Initialization Program Loader uses the DCB defined in the System Initialization program to load CICS nucleus modules and tables. This routine loads modules and relocates them, using RLD information passed by OS. The amount of storage used is maintained in two registers, one register containing the highest address available to be used, and the other register containing the amount of storage remaining.

Three abends can occur in this routine: (1) if a nucleus module is not located, (2) if the partition/region size is insufficient to load the nucleus, and (3) if an I/O error is encountered while loading the Real-Time Relocatable Program Library.

The DCB in the System Initialization program, used to load the nucleus modules and tables, is closed after System Initialization loads the CSA. To load resident application programs, this routine uses the DCB specified in the Program Control program.

Parameter Scan Routine

The Parameter Scan routine is responsible for scanning and edit checking the data passed by the user in the PARM field. This routine is linked to by code in the System Initialization program; information is passed in a field defined in the System Initialization program. This routine ensures that information passed is syntactically correct, and determines whether data is valid numeric or alphameric.

Console Put Routine

The Console Put routine is used to write messages to the console from System Initialization that are purely informational in nature and are used only to trace the startup procedure. A test is made to determine the message level specified in the loaded System Initialization Table. If the message level is zero, the message is not written; if the message level is one, the message is logged on the console.

7770 DCB PROCESSOR

The 7770 DCB Processor scans the Terminal Control Table open list searching for 7770 DCB's. When a 7770 DCB is found, it is tested to determine if it was successfully opened. If unopen, it is bypassed and the scan continues with the next entry in the Terminal Control Table open list. The MACRF and DSORG fields are then checked to see if they are correct for a 7770 DCB. If the fields are incorrect, the DCB is bypassed and the DCB scan continues with the next entry in the Terminal Control Table open list.

When a 7770 DCB is found to be open and valid, it's address is passed to the 7770 DEB Processor (type four SVC). Upon returning from the 7770 DEB Processor, the number of lines allocated to the line group is obtained from the DEB and the amount of core required for an IOB for each allocated line is calculated and acquired from OS subpool zero.

The 7770 DCB is then modified to resemble a BTAM DCB and the address of the 7770 Read-Write Program (DFHRWP70), which was loaded during the CICS nucleus build, is placed in the DCBREAD field of the DCB. Next, each IOB is initialized and an EXCP operation which causes a NOP command to be issued, is initiated for each allocated line. The DCB scan then resumes with the next entry in the Terminal Control Table (TCT) open list.

If upon reaching the end of the TCT open list, no 7770 DCB's have been processed, control is returned to continue opening user data sets.

If a 7770 DCB has been processed, the 7770 DCB Processor enters a 15 second real-time wait state before testing the completion of the EXCP operations. Upon completion of the wait, the TCT open list is again scanned for 7770 DCB's. When one is found, the completion status of each IOB associated with that DCB is tested. If the completion status indicates that the operation has not yet completed, the request is purged, the DCB name is extracted and logged on the console, and the line is placed out of service. Also, if the status indicates line is not operational, the DCB name is extracted and logged on the console, and the line is placed out of service. If the status indicates an I/O hardware error, the DCB name is extracted and logged along with the error information. When the end of the Terminal Control Table open list is reached the second time, control is returned to continue opening the user data sets.

7770 DEB PROCESSOR (DFHDEB70)

Using the existing DEB (the DCB was opened for EXCP) and TIOT entry information for a 7770 DCB, this module constructs a new DEB containing UCB pointers for each device allocated, chains it into the task DEB chain, and then frees the old DEB.

When entered, the old DEB is first checked to see if it already contains multiple extents. If it does, no action is performed by this module. If the old DEB contains only one extent, the TIOT entry for the associated DCB is located and scanned to develop the count of the number of devices allocated to the line group. If the TIOT indicates that only one device was allocated, no further action is taken by this module.

If multiple devices have been allocated, the required size of the new DEB is calculated and the storage obtained from the Operating System (OS) subpool 254 (system queue space). The appendage table, prefix, and basic section of the old DEB are now copied to the new DEB storage, and the number of extents, size of the DEB, access method length, and appendage table address are updated in the new DEB. The extent scale in the new DEB is set to 2 and the TIOT entry is now scanned, moving the allocated UCB addresses to the new DEB extents. Any residual length (AM section) in the old DEB is now moved to the new DEB. The new DEB is finally inserted in the TCBDEB chain at the same place the old DEB occupied, the associated DCBDEBAD pointer is updated, and the old DEB storage is freed from subpool 254.

TASK CONTROL PROGRAM (DFHKCP) - CHART 2

Task Control is responsible for the origination, synchronization, and termination of all CICS and user-initiated tasks.

The facilities of Task Control are accessed by other CICS management programs and user-written programs through the use of macro instructions. Task Control supports the following types of macro requests:

- Task Origination - ATTACH
- Task Termination - DETACH
- Task Enqueue - ENQ
- Task Dequeue - DEQ
- Task Suspension - SUSPEND
- Task Resumption - RESUME*
- Priority Change - CHAP
- Task Synchronization - WAIT
- Resource Scheduling-SCHEDULE*
- Resource Availability-AVAIL*

- Conditional-ATTACH*

After servicing any of the above functional requests, other than those marked with an asterisk, Task Control dispatches ready CICS tasks on a priority basis. Control returns directly to the requesting task on those services marked with an asterisk.

The following topics contain information relating to CICS system design as it applies to Task Control. Included is a discussion of control areas, multitasking control, and task synchronization.

COMMON SYSTEM AREA (CSA)

The CSA is a main storage area provided as part of CICS. The CSA exists within the system from initialization of the system until the system is closed down. It is composed of areas of data necessary to the operation of CICS and an optional work area that may be used as temporary work storage by a processing program. The user temporary work storage area is available for operations that are performed between requests for CICS services. This work space is available to any task while it has control of the system.

Control system data contained in the CSA are module addresses, statistics, common system constants, CICS control data, and a general register storage save area.

TASK CONTROL AREA (TCA)

This area is created for each task that is currently within CICS. The TCA provides to its associated task:

1. Register storage areas
2. Unique core storage for the communication of requests to CICS
3. Address of the related Facility Control Area (FCA)
4. Transaction storage chain addresses

The TCA is in existence only during the time that work exists for a task. The TCA contains control addresses and data necessary for CICS to control the task, but provides no space for residual data such as statistics. The TCA's are chained together logically, sequenced first by priority and then within priority, in the order in which they were created.

The TCA contents is divided into three sections: CICS system control, application program communication, and an optional transaction work area. The control section contains control addresses and data necessary for CICS to control the task. Access to data in this area is limited to CICS management programs. The application program communication section is used primarily for communication between the task and the service modules. Access is provided for both the CICS programs and user-written application programs.

Appended to the TCA is the Transaction Work Area (TWA). The TWA is acquired at task initiation as part of the TCA and has the same base register as the TCA. The TWA provides the user-written program with unique storage for the duration of the task. This area may be used to pass data or address constants from one program to another within one task. The TWA must be used if parameters are passed up a logical level. The size of the TWA is specified by the user in the Program Control Table.

MULTITASKING CONTROL

In the CICS/OS-STANDARD system, TCA's are chained together in dispatching priority sequence at the time a new task is originated (attached). TCA storage is obtained from Storage Control, is formatted, and is inserted in the priority chain in dispatchable status. If equal in priority to the originating task, the originated task's TCA is placed into the priority chain lower in priority than the originating task.

The Task Dispatcher ultimately gives control to the highest priority dispatchable task by searching the TCA priority chain. Therefore, if the originated task is equal in priority to the originating task, the originating task retains control.

CICS has two system tasks--Terminal Control and Task Control. Both of these system tasks process independently of other processing tasks. Each system task has its own TCA. Since the TCA is the primary vehicle for communication between processing tasks and CICS management programs, the Terminal Control task and Task Control task can utilize the services of all CICS management programs.

Terminal Control is the highest priority task in the system. Its TCA is the first TCA found on the dispatching priority chain. The Task Control task is active primarily during the task dispatching functional processing of the Task Control program and is treated as a logical task only by other CICS management programs. Its TCA is not on the dispatching chain since the task is dynamically activated by the Task Control program rather than selectively activated by the Task Dispatcher.

TASK SYNCHRONIZATION

The following describes the task synchronization facilities of the Task Control program.

Task Control accepts the standard OS ECB format for a Wait Control Block. For example, the completion indicator is byte 0, bit 1 of the Wait Control Block.

Waits may be of two types:

1. Single: A field in the TCA of the task issuing the wait contains the address of the Wait Control Block.
2. Multiple: A field in the TCA of the task issuing the wait contains the address of a list of addresses of Wait Control Blocks. The high-order byte of each address in the list for Terminal Control indicates that the list entry points to a TCTLE.

Terminal Control executes as a task that waits on a list of events.

The following schematic shows the relationships for a Terminal Control wait list.

```
      TERMINAL CONTROL TCA
      ----------------------------------------------
      *       *  Addr of List *
      ----------------------------------------------
              |
              |
              |
              v
       WAIT LIST                     TCTLE
      -----------------------        -----------------
   1) *CODE*Event Cntl Addr * -------≥ * ECB *
      -----------------------        ------------------
   2) *CODE*Event Cntl Addr * ---┐
      -----------------------    |
                                 |
                                 |
                                 | TCTLE
                                 | -----------------
                                 └-->* ECB *     *
                                   -----------------
```

Example 1: BTAM. The event control address points to the ECB within the TCTLE which is to be posted upon event completion.

Example 2: SAM. Same as for BTAM.

Through the use of CICS macro facilities, tasks other than Terminal Control can wait on single events or on a list of events.

If a task is waiting for completion of a single event (DFHKC TYPE=WAIT,DCI=SINGLE), the address in the TCA points to the Wait Control Block (ECB). (See Example 1 below.)

```
      TCA Example 1
      ---------------------------------------------
      *       * Ctl / Event Ctl Addr *
      ---------------------------------------------
                               |
                               |
                               |
                               v
                      ----------------
                      *   E C B      *
                      ----------------
```

If a task is waiting for completion of one of multiple events (DFHKC TYPE=WAIT;DCI=LIST), the TCA points to a list of event control addresses. Each event control address points to the Wait Control Block (ECB). (See Example 2 below.)

```
TCA Example 2
--------------------------------------------
*       * Addr of List *
--------------------------------------------
            |
            |
            |
            v
     WAIT LIST
      -----------------------      ---------
      *    * Event Ctl Addr --------> * E C B *
      -----------------------      ----------
      *    * Event Ctl Addr --------> * E C B *
      -----------------------      ----------
```

The Task Control Dispatcher assumes responsibility for testing completion code postings. If no task is ready to resume processing, the Dispatcher issues an OS STIMER for the lesser of the:

1. Unexpired system polling time delay
2. Next time-ordered event interval (if any)

builds a wait list, and then issues an OS WAIT macro instruction, placing the CICS partition/region in a wait state and giving control to OS.

Upon expiration of the time interval or upon completion of any event referenced in the wait list, OS posts the appropriate OS Event Control Block and ultimately returns control to CICS.

ENTRY ANALYSIS

Upon entry to Task Control, this module saves the requesting task's registers and analyzes the type of request code placed in its TCA as a result of the macro expansion. Valid requests are directed to the proper CICS management programs. Invalid requests cause the requesting task to be abnormally terminated.

TASK ORIGINATION (ATTACH AND CONDITIONAL ATTACH)

The ATTACH and Conditional ATTACH macro servicer is used to start a new CICS task at the request of another CICS task and schedule the new task on the basis of its own priority.

To provide multitasking capabilities, Task Control initiates and dispatches the tasks under its control. Upon entry, the Task Origination module confirms that the Transaction Identification associated with the new task is valid through a search of the Program Control Table (PCT) for a matching code. The Task Origination module also confirms that the security protection key associated with the terminal operator agrees with that in the PCT entry. If either is not confirmed, the module alters the attaching request to that of a CICS invalid transaction task.

A Task Control Area (TCA) must then be obtained for the new (ATTACHED) task before it can be initiated. A conditional GETMAIN request is made for TCA storage, and if the storage is available, the new task's TCA is constructed and inserted in priority sequence in a chain of TCA's associated with other tasks currently under CICS control. The task accounting functions required when a task is initiated are performed by a subroutine.

Conditional ATTACH macro requests are issued only by CICS management modules and system service programs. The macro is not available to application programmers. A positive response is returned to the calling routine on a successful conditional ATTACH request, and a negative response when the request was unsuccessful (e.g. insufficient storage necessary for a TCA). Control is returned directly to the calling routine on a Conditional ATTACH request (rather than through the priority dispatching logic).

An unsuccessful unconditional ATTACH request causes the attaching task's TCA to be marked as non-dispatchable, and will be made dispatchable by the Task Dispatcher when the storage becomes available.

A successful unconditional ATTACH results in the attaching task and newly attached task competing for dispatching according to their own dispatching priorities.

TASK TERMINATION (DETACH)

The DETACH macro servicer is used to delete references to the task from various CICS control blocks, remove the task itself from the system, and make available any resources held by the task.

The Task Termination module first determines whether the task has enqueued on any resource (through the Task Enqueue facilities of CICS). If it has, the task is removed from all queues using the facilities of the DFHKCP Task Dequeue module. References to terminal facilities used by the task, any time-ordered request associated with the task, and automatic task initiation dependencies associated with the task are removed from the system. References associated with the task and the task's TCA are removed from the priority dispatching queue. The main storage area reserved for the task's TCA is released, as is all storage chained off the TCA.

The Task Termination module enters to the Task Accounting subroutine, and then exits to the Task Dispatcher.

TASK ENQUEUE (ENQ)

Through a system of queuing requests and giving control of a resource to only one task at a time, CICS permits independent tasks to obtain exclusive control of resources used in common. This is accomplished by the ENQ macro servicer.

All tasks enqueuing upon a given resource do so by referring to it by a specific name. There is a Task Queue Area (TQA) chained to the task's TCA for each resource upon which the task is enqueued. There is a Queue Element Area (QEA) chained to the CSA for each resource enqueued upon by any task currently processing under CICS.

Upon entry, the Task Enqueue module determines whether the task has previously enqueued on the named resource. This is accomplished by searching for an existing TQA for the resource. If this condition is found, a use count in the TQA is increased by one. The TQA use count prevents improper release of the resource. If the condition is not found, a new TQA is created in main storage (obtained from Storage Control), and the TQA is added to the task's chain of TQA's.

The system's chain of QEA's is searched for an entry for the resource requested by the task. If an entry exists, it indicates that another task currently has control of the resource; the task's TCA is then added to the queue of waiting tasks chained to that QEA, and the module exits to the DFHKCP Task Suspension module.

If a QEA does not exist for the requested resource, it indicates that the current task can be given control at that time. The Task Enqueue module builds a QEA for the resource in a main storage area (obtained from Storage Control) and exits to the Task Dispatcher.

TASK DEQUEUE (DEQ)

The DEQ macro servicer is used to remove a task's request for exclusive control of a resource from a queue, making the resource available to another enqueued task.

Upon entry, the Task Dequeue module decrements the use count in the TQA by one. If the result is positive (indicating the task is multiple-enqueued), it exits to Task Dispatcher. If the use count is not positive, this module removes the TQA associated with the named resource from the task's chain of TQA's and (through Storage Control) releases the main storage it occupied.

The module then determines whether another task is waiting for control of the resource. If another task is not waiting, the associated QEA is removed from the chain and the storage it occupied is released through Storage Control.

If other tasks are waiting for control of the resource and the task being dequeued had control of the resource, the first waiting task is made dispatchable through the Task Resumption module.

If the task being dequeued did not have control of the resource (this could occur during abnormal termination of a task), the task is merely removed from the QEA chain.

The Task Dequeue module exits either to the Task Dispatcher, or, if a task is being removed from the system, returns to the Task Termination module.

TASK SUSPENSION (SUSPEND)

The SUSPEND macro servicer is used to temporarily remove a task from the system, pending the occurrence of a CICS system event or upon request of another task.

Task Suspension permits a task to voluntarily suspend itself and relinquish control of the CPU to some other task. The suspend function is also used by CICS management programs, as well as other functional modules of Task Control, when a task's processing must be interrupted until system resources currently in use are released and available for the task.

Upon entry, this module removes the task's TCA from the active task chain and inserts it in priority sequence in the suspended task chain. It then exits to the Task Dispatcher.

TASK RESUMPTION (RESUME)

The RESUME macro servicer is used to reinstate another task that has been previously suspended through a SUSPEND macro instruction.

If the task indicated by the requesting task was suspended, the Task Resumption module removes the task's TCA from the suspended task chain, returns it to the active task chain (in priority sequence), and identifies that task's TCA as dispatchable. It exits directly to the requesting module.

PRIORITY CHANGE (CHAP)

The CHAP macro servicer is used to provide the user with the facility to change a task's dispatching priority.

Changing a task's own dispatching priority is one method used to synchronize tasks in a multitasking environment. The servicing of this request involves a repositioning of the task on the priority dispatching queue.

The Priority Change module employs two service submodules to accomplish its function. Through submodules, the task's TCA is deleted from its old position in the priority dispatching queue and inserted into the queue according to its new priority. Exit is then made to the Task Dispatcher.

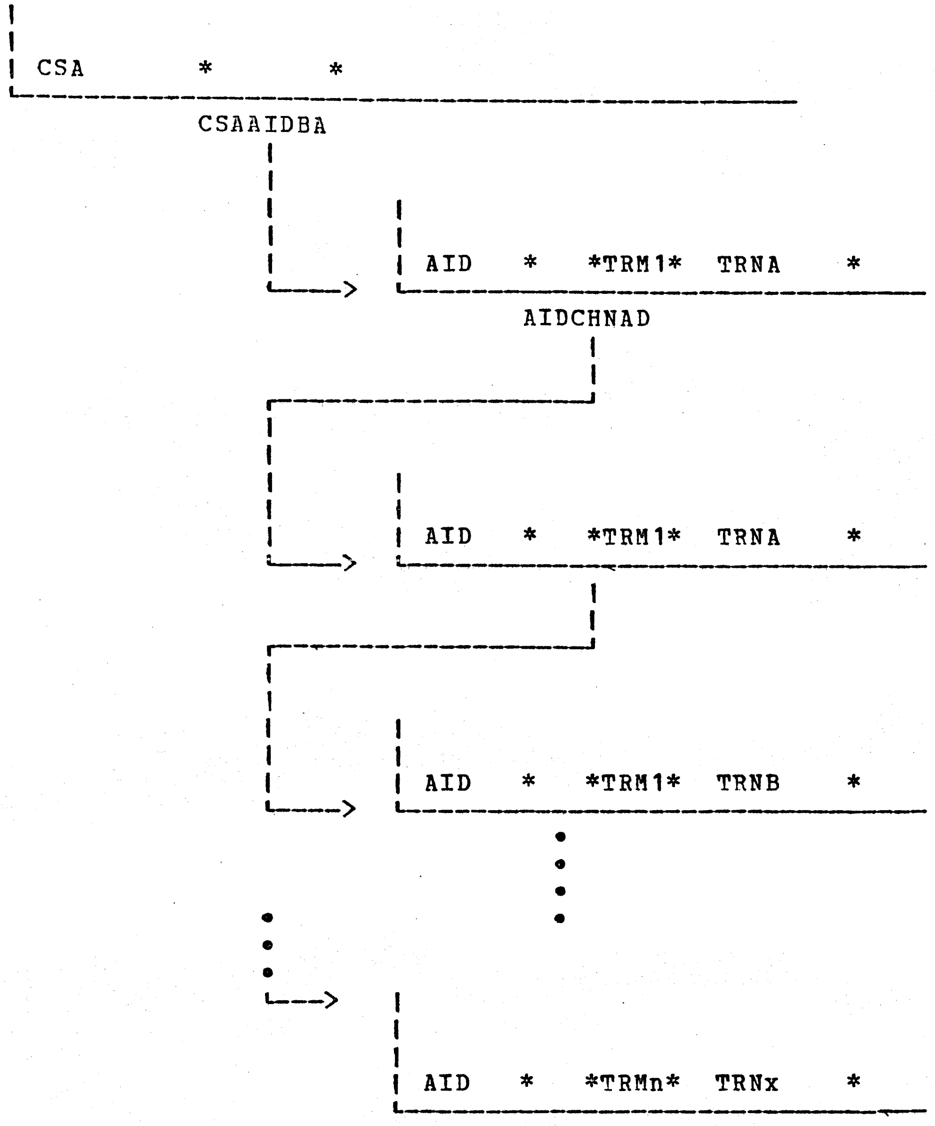
TASK SYNCHRONIZATION (WAIT)

The WAIT macro servicer is used to permit a task to synchronize its own processing with the completion of other events.

This module services the CICS WAIT macro instruction. A task can be waiting on a single event, on one of a list of events, or waiting in either a dispatchable or nondispatchable status. These options and pointers to Wait Control Blocks (ECB's) are identified by the user in the task's TCA upon entry to Task Control. This module passes the information on to the Task Dispatcher--directly in the case of tasks waiting on event completions and for wait-dispatchable tasks. In the case of a wait-nondispatchable task, it determines whether the task is associated with an Asynchronous Transaction Program terminal. If so, it performs a posting function; if not, an exit is taken to the Task Dispatcher through the Task Suspension module.

RESOURCE SCHEDULING (SCHEDULE)

The SCHEDULE macro servicer provides a queuing facility for initiating tasks from within CICS (as opposed to direct external requests from a user at a terminal), and synchronizing the initiation of the tasks with the availability of their respective terminal destination. Through the use of the system macro DFHKC TYPE=SCHEDULE (not available to application programmers) CICS modules request that the automatic initiating of a task be synchronized with the availability of a terminal. An Automatic Initiator Descriptor (AID) is created for each request for automatic task initiation dependent on the availability of a terminal, and is added to a chain of AIDs. Each AID contains the symbolic transaction identification of the task to be initiated, and the symbolic terminal identification of the terminal it is to be associated with. The entries in the AID chain are in sequence by symbolic transaction identification within symbolic terminal identification. The CSA contains the address of the top of the AID chain as illustrated in the following schematic.



When an AID is added to the chain, the Task Control program advises Terminal Control of an automatically initiated task depending on a particular terminal. This is done by setting an indicator in the associated Terminal Control Table terminal entry. Terminal Control advises the Task Control program when the particular terminal facility is available by issuing the AVAIL system macro instruction. The Task Control program initiates the ATTACH request for the new task. Interval Control program passes Interval Control Elements (ICE's) to Task Control. Task Control uses these for AID's and dynamically creates AID's when servicing other requests. Interval Control also passes AID's representing time-ordered data. If a time-ordered data record has been retained for the new task, the AID remains on the chain until the time-initiated task issues a request (GET) for the data record through Interval Control or terminates. The AID is removed from the chain at the time the task is initiated if no data record was associated with the original AID.

Upon entry to the Resource Scheduling module, it is determined whether an AID was provided by the calling module (for example, Interval Control program passing an ICE), bypassing the AID building logic if that was the case. Otherwise, the module issues a conditional Storage Control GETMAIN request for AID storage. If the request is not satisfied Task Control returns a response code to the calling module.

It is the calling module's responsiblity to queue the request and retry at some future time in its processing. If the storage is obtained an AID is built from parameters passed with the SCHEDULE macro request. These include the symbolic terminal and transaction identifications and a type classification for the AID itself.

The AID is then merged into the AID chain in Transaction Identification within Terminal Identification sequence. An indicator is set in the Terminal Control Table terminal entry corresponding to the symbolic terminal identification if the newly merged AID is the only one on the chain for that terminal. The purpose of the indicator is to signal to Terminal Control that a task is waiting to be initiated on a terminal. If the new AID fails to match an existing Terminal Control Table entry an error response code is returned to the calling module without merging the AID.

If the AID being merged already has an identical entry in the AID chain and does not have a time ordered data record associated with it, the redundant merge is not made and instead the AID storage is released.

This module returns control directly to the calling module (as opposed to priority dispatching).

RESOURCE AVAILABILITY (AVAIL)

Terminal Control advises the Task Control program when the terminal is available for automatic task initiation by issuing the CICS system macro instruction AVAIL. If a short-on-storage or maximum task condition exists, a response is returned to Terminal Control indicating that no task has been attached.

When the resources are available, the Task Control program searches the AID chain for the first AID for the available terminal. If none is found, the indicator in the Terminal Control Table is reset terminating automatic task initiation on that terminal. When the first matching AID is located, Task Control determines whether or not a task has already been initiated for that AID. If such is the case, it indicates that the task has terminated (normally or abnormally) without retrieving the data record associated with the original (Interval Control) PUT request. The logic unchains the AID, releases the Temporary Storage data area and the AID storage area, and then returns to search for the next matching AID on the chain.

When a matching AID not representing a previously initiated task is located, the Task Control program prepares to initiate the task. It first obtains a Terminal Input/Output Area from Storage Control (if an area is not already available for the new task). The new task is initiated through the conditional ATTACH routine. If unsuccessful, a negative response is returned to Terminal Control. If successful and the AID represents other than an original PUT request, the AID is unchained and its storage is released through Storage Control. Those representing PUT requests remain on the chain for the servicing of subsequent GET requests through Interval Control.

This module returns control directly to the calling module (as opposed to priority dispatching).

TASK INSERTION

Task Insertion is a service subroutine which places a task on either the active or suspended task priority chain according to the priority of other tasks currently under CICS control.

Upon entry, this subroutine scans the existing TCA's in the dispatcher priority chain and searches for the first TCA with a priority lower than the TCA to be added to the chain. At that point, the new TCA is inserted in the chain by adjusting the forward and backward chain addresses of the adjoining TCA's in the chain. The subroutine then returns to the calling module.

TASK DELETION

Task Deletion is a service subroutine which removes a task from either the active or suspended task priority chain.

Task Deletion removes reference to a given task by adjusting the forward and backward chain addresses in tasks' TCA's adjoining the given task's TCA. The subroutine then returns to the calling module.

TASK ACCOUNTING

The Task Accounting is a service subroutine. It provides the accounting functions related to the number of tasks under CICS control at a given time, the maximum number of tasks reached during CICS processing, and the maximum task level control desired by the user.

Through two entry points, this module accounts for all tasks originated, terminated, and in the system at a given time. It returns control to the calling module.

TASK DISPATCHER

The Task Dispatcher module gives control to the highest priority task under CICS control that is ready to execute.

There are two system tasks: a Terminal Control task and a Task Control task. Both have TCA's. The Terminal Control task is always active, always eligible tc resume control, and is highest in priority. The Task Control task is active only during selected times of Task Control processing.

The CICS/OS-STANDARD system maintains two types of task priority TCA chains: an active task chain which is maintained for task dispatching, and a suspended task chain containing the TCA's of tasks which have been temporarily suspended because of resource limitations or long duration input/output events. Tasks' TCA's are moved to the suspended chain at the direction of CICS modules performing services for the task. Tasks' TCA's are moved back to the active chain at the direction of CICS modules performing a service for some other task, the result of which makes available a resource the suspended task requires for future processing. The Task Dispatcher uses the active task chain in determining which task is to be given control of CICS.

Upon entry, the Task Dispatcher first performs certain Time Management functions. Through the Interval Control program, the Dispatcher initiates any time-dependent events that have expired since the last time the Dispatcher was in control. The actions taken depend cn the type of time event. (For further details, see the discussion of the Interval Control program in this manual.)

The Interval Control program returns to the Dispatcher the interval cf time remaining before expiration of the next time-dependent event. The Dispatcher then determines whether or not Terminal Control's dispatching time interval has expired; if it has, the Dispatcher resets the interval to the system's partition exit time interval defined by the user and posts the system Timer Event Control Block as completed.

The Task Dispatcher then initiates a scan of the active task TCA chain, locating the highest priority task identified as dispatchable, or that is ready to resume control because of the completion of an awaited event.

If no task qualifies as dispatchable, the Task Dispatcher issues an OS STIMER macro instruction for the smaller of the following time intervals:

1. Unexpired portion of the system partition/region exit time interval (Terminal Control's Dispatching Time Interval), or

2. Remaining time before expiration of the next time-dependent event.

The Dispatcher then repeats the active task TCA scan--this time building an OS WAIT list in an area acquired from the operating system, in preparation for releasing control to the operating system. If no task qualifies as dispatchable and the timer Event Control Block has not been posted as complete during the second scan of the TCA chain, the Dispatcher issues an OS WAIT on the list. Control is returned to the Task Dispatcher when any event in the list has been completed, and processing begins with the Time Management functions described earlier.

When the Task Dispatcher has determined that a task is dispatchable, it then makes several tests to determine whether or not it is a normal dispatch. Abnormal dispatches occur when a task has been requested to be abnormally terminated by another task, or in the course of performing system stall corrective action.

The stall detection and corrective action feature can be selected by the user for inclusion in CICS during system generation. Its purpose is to recognize when resources available to CICS become overloaded to the point where none of the tasks currently within CICS can continue and no new tasks can be started. The Task Dispatcher performs the detection function and initiates corrective action in the following manner:

- When dispatching Terminal Control, the Dispatcher determines whether the maximum task limit has been reached or main storage resources are in an overloaded state. If either condition exists, the Dispatcher initially sets a stall detection time interval and gives control to Terminal Control.

- If Terminal Control is the only task continually dispatched and CICS remains either at maximum tasks or with main storage in an overload condition during the stall time interval, it is assumed that a system stall exists and corrective action is initiated.

- The Dispatcher scans the suspended task chain, selects the lowest priority task in the system that is identified as "purgeable", and initiates its removal from the system. The "purgeability" of any task is under user control. When a task is started, it can initially have a purgeable or nonpurgeable status (defined in the Program Control Table entry). Dynamically, during execution, the task can alter this status via macro instructions. The act of removing a purgeable task from the system will normally relieve the stall condition and permit the remaining tasks to continue.

When dispatching any task other than Terminal Control, Runaway Task Detection is initiated, if applicable to the user's version of CICS.

This optional feature can be selected by the user for inclusion in CICS during system generation, and can be invoked or suspended during system initialization. The runaway task algorithm used by CICS is as follows:

- Any task which is given control by the Task Dispatcher will return to Task Control within a user-defined system interval of time. Control may be returned either directly (the application program issues a Task Control macro) or indirectly (the application program issues some other CICS service macro which in turn requests a Task Control service). Tasks not meeting this timed requirement are considered to be in a runaway (logical loop) state and are deleted from the system.

- Each time a task enters Task Control, the runaway task control for that task is terminated. Immediately prior to dispatching the task, Task Control sets the system timer for the system runaway task time interval specified by the user. Expiration of this time prior to returning to Task Control will cause a timer interrupt, giving control to the Interval Control program's runaway task routine where removal of the task is initiated.

DISPATCH CONTROLLER

The Dispatch Controller routine tests the dispatch control indicator in a given task's TCA to determine if the task is waiting on the completion of an event and/or whether completion of the event has occurred.

A task is in one of four states when being considered for dispatching: dispatchable, nondispatchable, waiting on a single event, or waiting on one of a series (list) of events. The task's status is identified in the dispatch control indicator of its TCA.

Upon entry, this routine analyzes the dispatch control indicator. If a task's status is none of the above, the routine abnormally terminates the task. If the task is waiting on a single event or one of a list of events, the routine tests for completion of the events. If the task is waiting on the completion of a single event, an entry in the task's TCA is the Event Control entry (a pointer to a completion posting medium). If the task is waiting on the completion of one of a list of events, the entries in the list are Event Control entries and the TCA points to the first list entry.

This routine passes Event Control entries to the Event Completed Test subroutine responsible for testing completion of events.

If an event associated with the task has been completed, or if the dispatch control indicator identifies the task as dispatchable, the routine branches directly to task dispatching logic in the Task Dispatcher.

If no event associated with the task has been completed, or if the dispatch control indicator identifies the task as nondispatchable, the routine analyzes the next task.

EVENT COMPLETED TEST

The Event Completed Test routine is a service subroutine that tests the completion code posting positions of the Event Control Blocks associated with the tasks.

Upon entry, this subroutine tests whether the Event Control entry is a TCTLE and whether an I/O event has been initiated on the line. If no I/O event has been initiated, the subroutine returns to the calling routine. If an I/O event has been initiated on the line, or if the Event Control entry is not a TCTLE, the subroutine tests the ECB for completion. If not posted as complete, the ECB is added to an OS WAIT list (providing the wait list indicator at KCOSWS is set).

If the completion code is posted in the selected Event Control Block, the subroutine exits directly to the Task Dispatcher Exit. If the completion code has not been posted, the subroutine returns to the calling routine.

REFRESH CSA TIME OF DAY

Two Time Management service subroutines are included in the Task Control program. One updates the packed decimal, binary and timer unit's representations of time-of-day maintained in the CSA. The other subroutine updates only the binary and timer unit's formats.

The packed form is only updated after control is returned to the Dispatcher from the operating system. Because the conversion routine involves considerable processing overhead, more frequent updating by CICS is performed only in response to Interval Control program macro requests issued by application programs.

The binary and timer unit forms are updated each time the Dispatcher is entered, as well as in response to application program macro requests.

A test is made to determine whether the current time of day obtained from the operating system is a value less then the previously obtained time of day. (Such is the case when OS resets the time of day to zero at midnight, or if the operating system's time of day is altered by the console operator.) If the current time of day is a value less than the previously obtained time of day, the difference between the two time values is recorded as an adjustment value in the CSA; two indicators are also set in the CSA.

If the optional Time Adjustment feature has been included in the system, a system task is initiated by the Interval Control program (ICP) which adjusts the time of day maintained by CICS to agree with the time of day maintained by the operating system. If the Time Adjustment feature has not been included, the time of day maintained by CICS is continually adjusted relative to the time CICS was initialized.

INTERVAL CONTROL PROGRAM (DFHICP) - CHART 3

The Interval Control program, along with the Task Control program, share the responsibility for the various Time Management functions supported by CICS. The Interval Control program logic contains the following functional routines:

- The optional CICS Time Management functions supporting macro service requests for time-of-day services, time-ordered task synchronization services, and time-ordered automatic task initiation services.

- The Timer Interrupt routine used in conjunction with the OS STIMER macro, including the optional CICS runaway task detection and corrective action support.

The Time Management macro support and runaway task support are logically independent within the Interval Control program and therefore will be described separately. The entry address of the Interval Control program in the CSA (CSAICNAC) points to a table assembled at the beginning of the program. The table contains entry addresses and address constants used in communication between Interval Control and other CICS system programs.

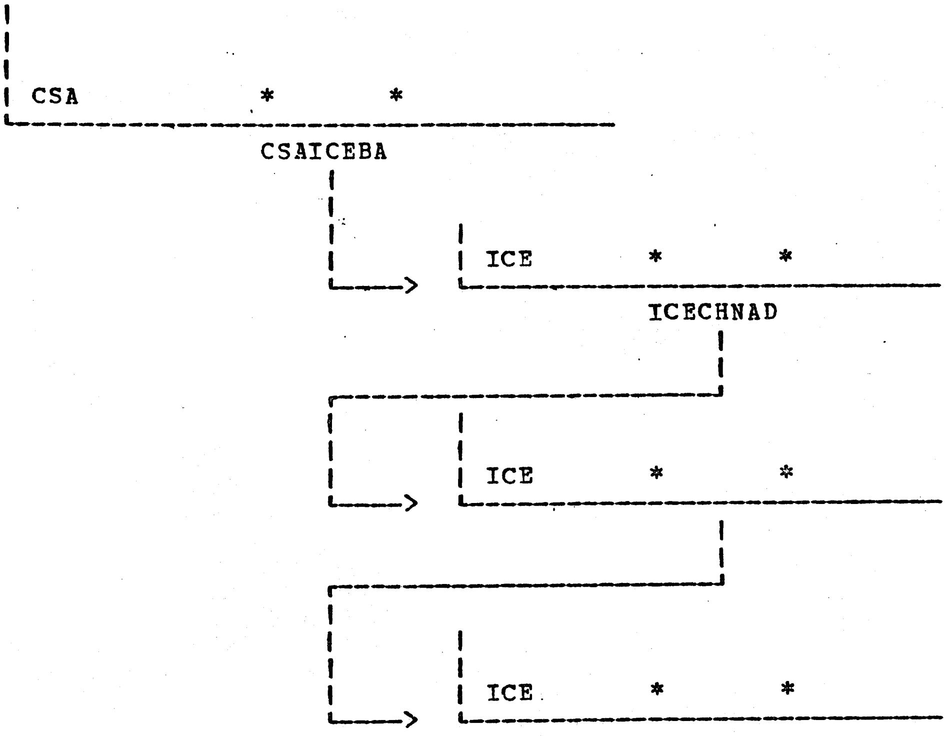
TIME MANAGEMENT MACRO SERVICE SUPPORT

User-written programs access the Time Management services of the Interval Control program through the use of the following types of macro requests:

- Time-of-day service - GETIME
- Task synchronization - WAIT
- Task synchronization - POST
- Automatic task initiation without data retention - INITIATE
- Automatic task initiation with data retention - PUT
- Retrieve time-ordered data - GET and RETRY
- Cancellation of a prior request - CANCEL

Services other than the time-of-day services are dependent upon the expiration of a timed event. CICS utilizes a time-ordered queuing technique in controlling these time-dependent service requests.

An Interval Control Element (ICE) is created for each time-dependent request received by the Interval Control program. These ICE's are logically chained off the CSA at CSAICEBA in expiration time-of-day sequence as illustrated in the following schematic.



Expiration of a time-ordered request is detected by the expired request logic of the Interval Control program running as a CICS system task whenever the Task Dispatcher gains control (see Task Control program). The type of service represented by the expired ICE is initiated, providing all resources required for the service are available, and the ICE is removed from the chain. If the resources are not available, the ICE remains on the chain and another attempt to initiate the requested service is made the next time the Task Dispatcher gains control.

The automatic task initiation services of the Interval Control program uses the resource scheduling facilities of the Task Control program in synchronizing expired requests with the availability of their respective terminal destinations, and in making time-ordered data records available to the tasks. Expired ICE's are moved from the ICE chain to a chain of Automatic Initiator Descriptors AID's (see Task Control program). This chain is sequenced by symbolic Transaction Identification within symbolic Terminal Identification. The CSA contains the address of the top of the AID chain as illustrated in the previous schematic.

Tasks associated with a terminal can retrieve sequential time-ordered records destined for the same Terminal Identification and symbolic Transaction Identification by issuing consecutive data record retrieval requests to the Interval Control program. When servicing a request for time-ordered data, the Interval Control program presents each record to the requesting task and removes its corresponding AID from the chain. When all data records represented by AID entries destined for the Terminal Identification and Transaction Identification are exhausted, the Interval Control program returns an end-of-data response to the requesting task.

The following describes the internal logic of the Interval Control program macro support modules and subroutines.

ENTRY ANALYSIS

Upon entry to Interval Control, this module saves the requesting task's registers in that task's TCA and analyzes the type of request code placed in its TCA as a result of the macro expansion. Valid requests are directed to the proper macro service modules. Invalid or non-supported types of requests result in an error response being returned to the requesting task.

TIME-OF-DAY SERVICES (GETIME)

The GETIME macro servicer uses one of two routines contained in the Task Control program to provide the time-of-day support. An address constant table within the Task Control program contains the entry addresses of the two time-of-day routines, as well as Task Control's own base register value. A pointer to the table is made available to the Interval Control program during CICS system initialization.

The GETIME macro servicer sets Task Control's base register, obtains either the packed decimal or binary form routine's entry address and branches directly to the Task Control program. If the user has provided a data field with his macro request, the Interval Control program moves the requested form of time to that field prior to returning control to the calling task.

TASK SYNCHRONIZATION (WAIT)

The WAIT macro servicer causes the requesting task to temporarily suspend its own processing and to resume control after the passage of time. An Interval Control Element for the WAIT request is built through a closed subroutine. Its address is placed in the requesting task's TCA, and the ICE is added to the ICE chain through another subroutine. The requesting task is then suspended through the Task Control program. When the task resumes processing (upon expiration of the time, or cancellation of the original request), it begins processing at the logic immediately following the SUSPEND macro. Here the ICE address and request identification in the task's TCA are cleared, the proper response code is set, and the ICE storage is released prior to returning to the user's program.

When the original WAIT request expires or is cancelled by another task, the Interval Control program unchains the ICE through a subroutine, requests Task Control to resume the WAITING task and returns to either the expiration analysis or cancellation logic of the Interval Control program.

TASK SYNCHRONIZATION (POST)

The POST macro servicer permits a requesting task to retain control of CICS, and CICS will indicate to the requesting task when a certain time has expired. An Interval Control Element for the POST request is built through a closed subroutine, and its address is placed in the requesting task's TCA. A four-byte field in the ICE, reserved for the Timer Event Control area, is set to binary zeros, its address is returned in the task's TCA, and the ICE is added to the chain. Control is given to Task Control to permit entry into the expiration analysis logic, ensuring the proper response code setting and posting of the Timer Event Control area had the time already expired.

When the original POST request expires or is cancelled by another task, the Interval Control program unchains the ICE through a subroutine, and posts the Timer Event Control Area as completed. The completion bits set are compatible with both OS and DOS operating systems. It then returns to either the expiration analysis or cancellation logic of the Interval Control program.

AUTOMATIC TASK INITIATION (INITIATE and PUT)

For the most part, common logic services both the original INITIATE and PUT macro requests. The request's symbolic Transaction Identification is verified against the Program Control Table and an error response is returned on unmatched conditions. If the task to be initiated is dependent upon the availability of a particular terminal facility, the symbolic Terminal Identification is verified against the Terminal Control Table (the TCT search is performed by a closed subroutine). An error response is returned on an unmatched condition.

An Interval Control Element for the INITIATE or PUT request is built through a closed subroutine, and the transaction and terminal identifications are moved to the ICE.

Finally, if it is a PUT macro service request, the Interval Control program stores the data record associated with the request using the Temporary Storage facilities of CICS. The symbolic data identification given to Temporary Storage is the same as the unique Request Identification provided by the user or developed by the Interval Control program.

Reference to the ICE is removed from the requesting task's TCA, the ICE is added to the chain through a subroutine, and control is returned to the requesting task.

If the original INITIATE or PUT request is cancelled prior to expiration, the ICE is unchained, the data area associated with a PUT request is released through Temporary Storage, the ICE storage is released through Storage Control, and control then returns to the cancellation logic of the Interval Control program.

When an INITIATE or PUT request not associated with a terminal expires, the new task is initiated immediately, provided neither a short-on-storage nor a maximum task condition exists. If either is the case, the expired ICE remains on the chain until such time as the system stress condition is relieved and the task can be started. If the resources are available, the ICE is unchained through a closed subroutine, and the ICE storage is released through Storage Control for other than an original PUT request. On an original PUT request, the ICE is retained and its address is used as the new task's Facility Control Address. The new task is initiated via an ATTACH macro request to Task Control.

The Task Dispatcher is in control and is using Task Control's TCA during this portion of the execution of the Interval Control program. However, since this TCA is not on the dispatching chain (and therefore is not dispatched back to this point in the logic), the ATTACH macro request acts like an unconditional branch to the task initiation logic of Task Control.

When an INITIATE or PUT request associated with a terminal expires, the ICE is unchained through a closed subroutine, and is merged into the Auto Initiate Descriptor (AID) chain in Transaction Identification within Terminal Identification sequence through the resource scheduling facilities of the Task Control program.

RETRIEVE TIME-ORDERED DATA (GET and RETRY)

The GET macro servicer permits a task to retrieve data records retained through the PUT macro facilities of the Interval Control program. A task not associated with a terminal can only retrieve the single data record associated with the task's originated PUT request. The task's Facility Control Address (in its TCA) is the address of the AID representing the original request.

A task associated with a terminal can retrieve one or more data records associated with original PUT requests. They must be destined for the same Terminal Identification and Transaction Identification, and must be on the AID chain during that task's execution. In response to a GET request, the Interval Control program searches the AID chain, returning a normal end-of-file response to the requesting task when no matching AID is found. The logic unchains the first matching AID that is found, releases the AID storage through Storage Control, and retrieves the associated data record from Temporary Storage. The data record is returned to the requesting task in either the data area provided by the task or an area obtained by the Temporary Storage program.

A response is returned to the requesting task in the event an I/O error occurs during the retrieval operation. The Interval Control program supports a RETRY macro request that can be issued in a user's error routine. The RETRY macro servicer executes the retrieval request to Temporary Storage using the parameters found in the TCA at the time the macro instruction was issued. These parameters are assumed to be the same as those returned to the requesting task at the time the error was detected.

CANCELLATION OF PRIOR REQUEST (CANCEL)

The CANCEL macro servicer permits a task to cancel a prior time-dependent request made by it or some other task. The cancellation request usually references a particular prior request by providing the unique Request Identification. A cancellation request without an accompanying unique Request Identification is normally used to cancel a prior POST request made by the same task, or when a task is being abnormally terminated with an unexpired request (for example, WAIT) still pending. In either case, the logic unchains any unexpired ICE through a closed subroutine, releases the ICE storage through Storage Control, and returns control to the requesting task.

When a cancellation request has an accompanying unique request identification, the Interval Control program scans the ICE chain for a matching Request Identification. If no matching ICE is found, a response indicating the condition is returned to the requesting task. If a matching ICE is found, the type of original request it represents is determined through a closed subroutine.

Depending on the type of original request, the Interval Control program processes the cancellation request as described previously in the appropriate macro servicer module description. Control is then returned to the requesting task.

EXPIRATION ANALYSIS

The Expiration Analysis routine is entered by the Task Dispatcher operating as a system task. Upon entry to this routine, the setting of two indicators in the CSA is tested. If both indicators are on, the Task Control program has detected a significant change in the operating system's time of day (for example, the time of day was reset to zero at midnight).

The Expiration Analysis routine then issues a Task Control ATTACH request that initiates the optional Time Adjustment feature (provided that neither a maximum task nor short-on-storage condition exists). The Time Adjustment program, operating as an independent CICS task, adjusts all expiration time-dependent controls (that is, expiration times of requests in the ICE chain) and then resets the time of day maintained by CICS to agree with the time of day maintained by the operating system. If no time adjustment action is required, the Task Dispatcher operating as a system task scans the ICE chain for expired elements.

The ICE chain is sequenced by expiration time of day. The Expiration Analysis routine scans the ICE chain, testing each element to determine whether or not its time has expired. Each expired request is serviced in turn through a closed subroutine that determines its type and then is processed by type as described previously in the appropriate macro servicer module description.

The remaining unexpired time interval for the first unexpired ICE detected is returned to Task Control. It is used by Task Control when setting the timer prior to relinquishing control of the CPU to the operating system. (Task Control presets the partition exit time interval value in its TCA prior to entering this routine. That value is returned if no unexpired ICE's are found.)

TYPE OF TIMED EVENT ANALYSIS

This is a closed subroutine used to support the cancellation and expiration analysis functions of the Interval Control program. It merely tests the type code of the ICE currently being addressed and branches to the appropriate macro servicer module. A response is returned to the calling routine if an unidentified ICE is detected.

CREATE INTERVAL CONTROL ELEMENT ROUTINE

This closed subroutine is used in support of the WAIT, POST, INITIATE, and PUT macro servicer modules. If the task issuing the request already has an ICE associated with it (as the result of a prior POST request), the ICE is unchained (if applicable) and reused for the current request. Otherwise, ICE storage is obtained through Storage Control. The type of request code is used to identify the new ICE type. A unique request identification is created by the Interval Control program if none was passed with the request. The interval or time-of-day value passed with the request is converted to an expiration time-of-day value expressed in 300ths of a second.

This routine exits to the calling macro servicer module.

TERMINAL CONTROL TABLE SEARCH ROUTINE

This closed subroutine is used to resolve the symbolic Terminal Identification (passed with requests for automatic task initiation) with an actual entry in the Terminal Control Table. The routine sets a negative condition code to indicate a no-match condition, or returns the address of the matching Terminal Control Table entry in a register when exiting to the calling module.

CHAIN AND UNCHAIN INTERVAL CONTROL ELEMENT ROUTINES

These closed subroutines are used in support of several other modules of the Interval Control program, providing common logic for the maintenance of the ICE chain. In addition to performing their appointed functions, they control the setting of two status indicators in the ICE's: the "expired on entry" indicator and the "on the chain" indicator. This routine exits back to the calling Interval Control program module.

RUNAWAY TASK SUPPORT

The Runaway Task feature of CICS is optional and can be selected for inclusion by the user during System Generation. If this feature is selected, any task which is given control by the Task Control program must return to Task Control within a user-defined interval of time. Control may be returned either directly (the application program issues a Task Control macro) or indirectly (the application program issues some other CICS service macro which in turn requests a Task Control service). Tasks not meeting this timed requirement are considered to be in a runaway state (logical loop) and will be deleted from the system.

If the user's generated version of CICS includes the Runaway Task feature, the feature can be invoked or suspended during System Initialization. It is during this phase of processing that the runaway task time interval value and corrective action linkage between CICS programs are established.

The Interval Control program contains an Initialization routine, Timer Interrupt routine, and Runaway Task Flush routine.

The System Initialization program enters the Initialization routine via a macro instruction. Upon entry, the Initialization routine stores the address of the Task Control program's address constant list and saves the CSA address. To support the Runaway Task feature, the Program Interrupt program (PIP) must also be operational. The Initialization routine then tests to make certain the PIP is loaded, exiting without initializing the Runaway Task feature if PIP has not been loaded. In support of the Runaway Task feature, the Initialization routine resolves the Flush routine address linkages in the CSA and the Interval Control program. The Initialization routine then returns control to the System Initialization program.

Most of the CICS management programs (for example, Storage Control and Program Control) contain system macros at their entry and exit points. The purpose of the macros is to indicate when a task is executing CICS management program logic or operating system logic as opposed to application program logic. This is done by setting and clearing the appropriate indicator bit in the task's TCA at TCASVMID.

One bit in the byte at TCASVMID is used by Runaway Task to signal whether or not the task's runaway task time interval has expired. Immediately prior to dispatching a task, Task Control turns on the control bit (sets it to 1) in the task's TCA, sets the system timer (via an OS macro) to the runaway task time interval, and establishes the runaway task linkage in the CSA.

When a CICS management program prepares to return control to a calling routine, it turns off (sets to zero) that program's assigned indicator bits in the task's TCA and tests the remaining bits. If any of the indicator bits at TCASVMID are still on, control passes back to the calling routine. If TCASVMID is zero, it indicates that the runaway task time interval has expired and the CICS management program branches to the runaway task linkage instructions in the CSA.

The first linkage instruction is a NOP branch instruction which is set to a branch status (turned off) each time the Task Control program is entered. It is set to a NOP status (turned on) only when a task is dispatched and the runaway task time interval is set. If the CSA runaway task linkage is executed with the first instruction in a branch status, the linkage causes control to pass directly back to the calling routine. If the first instruction is in a NOP status, the remaining linkage instructions are executed, causing entry to the Runaway Task Flush routine.

Entry to the Timer Interrupt routine of the Interval Control program occurs whenever a previously issued OS STIMER macro request expires. The Timer Interrupt routine determines whether or not the interrupt was the result of an expired runaway task time interval. If it was not, a Timer Event Control Area is posted and an exit occurs. If entry to the routine was because of a runaway task, the runaway task expiration bit in the task's TCA is set to zero and the TCASVMID byte is tested. The Timer Interrupt routine returns control to the operating system after issuing an OS POST request for the CICS Timer Event Control Block.

If the byte at TCASVMID is nonzero, the interrupt occurred while executing CICS management program logic or operating system logic. The runaway task is then deleted from the system when the CICS management program attempts to return control to the looping application program via the CSA linkage instructions.

If the byte at TCASVMID is zero, the looping application program itself was interrupted. In this case, a scan of the OS Request Block (RB) chain is initiated and the address portion of the PSW contained in the Program Request Block (PRB) is checked to make certain the address is within the CICS partition/region, exiting from the Timer Interrupt routine if the address is not within the partition/region. If the address is within the partition/region, the operation code byte at that location is saved in the CSA, and the operation code itself is set to zero. The address of the "zeroed" operation code is stored in the Register Save Area of the Program Interrupt Control program (DFHPIP).

The purpose of the program logic just described is to force a program interrupt to occur immediately upon returning control to the looping application program. The Program Interrupt Control program recognizes this type of interrupt, restores the operation code, and initiates the flushing of the runaway task.

The Runaway Task Flush routine in the Interval Control program moves a CICS abnormal condition code to the runaway task's TCA and abnormally terminates the task through that facility of the Program Control program.

STORAGE CONTROL PROGRAM (DFHSCP) - CHART 4

The Storage Control program is responsible for maintaining all main storage resources within the CICS partition. Its primary function is the acquisition and disposition of dynamic storage.

Storage Control first determines whether a GETMAIN or FREEMAIN function is requested and branches to the appropriate module. GETMAIN attempts to acquire storage from the chain of free storage areas. If unable to do so, the task is suspended or control is returned to the user with an indication that the storage area was not obtained. FREEMAIN returns a piece of storage to the free storage chain and combines it with any adjacent free storage. If any tasks are suspended, an attempt is made to get their storage and resume the tasks.

The major logical functions within Storage Control are:

- Entry Analysis
- Storage Acquisition
- Storage Disposition
- Storage Statistics
- Initialization
- Transaction Suspend
- Storage Cushion Management
- Allocation Protection

ENTRY ANALYSIS

The Entry Analysis module is responsible for determining whether a GETMAIN (conditional or unconditional) or FREEMAIN is requested. The RELEASED (RLSE) exit is taken by CICS at system initialization to cause Storage Control to obtain the initial storage cushion. The RELEASED exit is also used by Program Control (if the SOS indicator is on) to inform Storage Control that the use count of a temporarily resident program has been reduced to zero. Storage Control then restores the storage area occupied by the temporarily resident program to the subpool and attempts to satisfy any queued storage requests.

Upon entry, Parameter Validation ensures that the user has provided the necessary information to perform the Storage Control function.

Normally, all validation is done when the user expands the macro call to Storage Control; however, if the macro instruction is not used, validation must be available in Entry Analysis.

The following chart lists the storage classes and indicates whether the storage is chained (C) or not chained (NC).

STORAGE CLASS	STORAGE CHAINING
RSA	NC
QEA	NC
TQA	NC
1WD	NC
TCA	NC
FILE	C
USER	C
TRANSDATA	C
LCA	C
2WD	C
TERMINAL	C

STORAGE ACQUISITION (GETMAIN)

The Storage Acquisition module establishes a pointer to the Subpool Boundary Box. The Subpool Boundary Box controls the chain of available free areas within the dynamic storage area.

The GETMAIN subroutine is used to actually acquire the storage. If storage is acquired, its address is returned to the requesting task.

If storage is not available, various actions can be taken, depending on whether the request was conditional or unconditional. For each condition indicated in the following table, the corresponding action is determined by an "X" in the same vertical column.

TYPE OF REQUEST		
Unconditional	X	
Conditional		X
ACTION		
Free res pgms w/use count=0	X	X
Release storage cushion	X	X
Turn on SOS indicator	X	X
Suspend task	X	
Return to task, indicate no storage available		X

GETMAIN SUBROUTINE

This subroutine is used to locate a piece of dynamic storage which will satisfy a required length.

Upon entry to the Getmain subroutine, a pointer is established to the first free main storage area, the first eight bytes of which contain the length of this area and a chain address to the next free area.

The eight-byte control field is called a Free Area Queue Element (FAQE); it describes the contiguous free area.

Storage allocation is controlled by a chain of FAQE's which describe each non-adjacent free area. Each FAQE in the chain is checked until one equal to or greater than the requested size is found. If the free area is larger than the requested size, the residual area becomes a new entry in the Free Area Queue Element chain. If an area is found, its address is returned; otherwise, an address of zero is returned.

STORAGE DISPOSITION (FREEMAIN)

Storage Disposition calls the CICS FREEMAIN subroutine which returns the requested storage to the chain of free areas. The count of the number of FREEMAIN's is incremented and the SOS indicator is checked; if this indicator is on, an attempt is made to satisfy any storage requests for tasks which were suspended because storage was not available.

When a transaction storage request is satisfied, a Task Control RESUME macro instruction (DFHKC TYPE=RESUME) is issued so that the task's TCA is marked dispatchable and placed in the active task chain. If all storage requests are satisfied and the storage cushion acquired, the SOS flag is turned off, allowing Terminal Control to invite new transactions.

FRFEMAIN SUBROUTINE

The FREEMAIN subroutine searches the free area chain until the proper location for the area being freed is found. The free area is returned to the chain and is combined with any adjacent free areas to form a larger area. The chain consists of all non-adjacent free storage areas chained (regardless of size) in ascending order, from low to high main storage. Return is to the calling routine.

TRANSACTION SUSPEND

Whenever Storage Control receives an unconditional GETMAIN request and the request cannot be satisfied, a Task Control SUSPEND macro instruction (DFHKC TYPE=SUSPEND,DCI=SC) is issued. This causes Task Control to place the transaction's TCA in the suspended task chain and mark the TCA as "nondispatchable, waiting for storage". The Task Control RESUME macro instruction is issued by the Storage Disposition module when enough storage has been freed to satisfy the request of the suspended transaction.

STORAGE CUSHION MANAGEMENT

Storage cushion management is a technique used to reduce the occurrence of system overload.

The Storage Cushion is an area of dynamic main storage (size determined by user at system initialization) obtained by the system at system start-up time. The cushion is released by the Storage Acquistion module whenever a request for main storage cannot be satisfied. When the cushion is released, the SOS switch is turned on. The cushion is reacquired again by the Storage Disposition module whenever a FREEMAIN or RELEASED macro request results in enough free space to satisfy the cushion size. The SOS switch is turned off whenever the cushion is reacquired.

STORAGE ACCOUNTING

Storage Accounting prevents fragmentation of storage whenever a task fails to issue CICS FREEMAIN's for all storage it has acquired. It also ensures that the correct length and characteristics of the dynamic storage are preserved for storage disposition.

Once storage has been allocated because of a GETMAIN request, Storage Control must preserve certain control information concerning the request. For instance, the first four bytes of all storage requests are used by Storage Control to preserve the length and class of the storage. In addition, the second four bytes are used to chain all user-type storage onto the TCA of the requesting program. The user must be aware of these considerations when requesting dynamic storage. All terminal storage is chained from the TCTTE in the same manner.

STORAGE STATISTICS

Storage Statistics is not a physical module in the sense of the other Storage Control modules. Various statistics are gathered at many different points within the Storage Control program. These statistics, kept in the CSA, are:

- Number of GETMAIN's issued
- Number of FREEMAIN's issued
- Number of times storage cushion released
- Maximum number of suspended tasks
- Total number of suspended tasks, etc.

INITIALIZATION

Upon request from the user, the Storage Control Program will initialize every byte in the acquired storage to a user-specified byte configuration.

PROGRAM CONTROL PROGRAM (DFHPCP) - CHART 5

Program Control is responsible for the loading, deleting and managing of program modules controlled by CICS.

The facilities of Program Control are accessed by other CICS modules and user-written programs through the following types of macro requests:

- Program Link - LINK
- Program Transfer Control - XCTL
- Program Abnormal Termination - ABEND
- Module Load - LOAD
- Module Delete - DELETE
- Program Return - RETURN

ENTRY ANALYSIS

Entry Analysis determines the type of service to be performed for the requesting task.

Upon entry to Program Control, this routine saves the requesting task's registers in the TCA. The type of request code (placed in the TCA as a result of the macro expansion) is analyzed, and valid requests are directed to the proper CICS management programs. The Entry Analysis module abnormally terminates tasks whose request types are invalid.

PROGRAM LINK (LINK)

The program requesting a LINK service is considered to be of a higher logical level than the program being linked. The Program Link module provides the entry and return linkage for the requested program. Upon entry, this module obtains a register save area through Storage Control. It then saves the requesting program's registers and PPT and register save area addresses in the acquired area. The address of the new register save area is stored in the task's TCA before exiting to the Program Load module.

TRANSFER CONTROL (XCTL)

The program requesting an XCTL service is considered to be of a logical level equal to that of the program to which control is to be transferred. Therefore, the Transfer Control module does not support any return control linkage. It merely initiates the release of control of the task's currently active program through a support subroutine and exits to the Program Fetch module. The requested program is retrieved and control is passed to the entry point in that program. If the program issuing the XCTL is coded in ANS COBOL, the area acquired for the TGT is freed. If the program is coded in PL/I, the epilogue is executed so that any PL/I storage is freed.

ABNORMAL TERMINATION (ABEND)

The Program Control program services requests to abnormally terminate tasks. This includes obtaining a CICS system dump of the transaction, releasing control of all programs and storage associated with the task and passing control to the CICS Abnormal Condition program.

Upon entry, the Abnormal Termination module calls the CICS Dump Control program to obtain a transaction dump of the task being terminated. Each level of program associated with the task is released. This includes the currently active program, any linking programs, and any associated data storage.

The TCA is modified to cause a transfer of control to the CICS Abnormal Condition program. This program analyzes the abnormal condition codes and transmits the reason for the termination to the user. The Abnormal Condition program returns control to the Program Control program through a RETURN macro instruction, at which time the task is detached from the system.

MODULE LOAD (LOAD)

This module services LOAD macro requests. It loads the requested module into user storage by means of the Asynchronous Relocatable Loader. The location of the loaded module is then returned to the requesting program.

The first time a program is loaded, an entry is made in the load list and the use count is incremented by one. Any further loads given for that program merely increment the use count in the load list. This prevents multiple copies of a program from being loaded by multiple load requests for the same program.

MODULE DELETE (DELETE)

This module locates the Processing Program Table (PPT) entry associated with the program to be deleted. The use counter in the

PPT is decremented by one. When the use count reaches zero, the storage area is released if the SOS indicator is on; otherwise, the storage is released only when the SOS indicator is turned on.

PROGRAM RETURN (RETURN)

User-written programs terminate their processing by issuing a RETURN macro instruction. (The PL/I END or RETURN statement can be used if the program is written in PL/I.) If the program issuing the macro instruction is initially given control via a LINK macro instruction, return of control is to the linking program. If the program issuing the RETURN macro instruction is the highest level program associated with the task, the task itself is considered to be completed and is terminated and detached from the system via a DETACH request to Task Control.

The Program Return module releases control of the task's currently active program through the RETURN subroutine. This subroutine also reloads the registers for the next higher level program, if applicable.

If the program issuing the RETURN macro instruction is not the task's highest level program, control is returned to the next higher level program, if the program is coded in a high-level language, steps are taken (similar to those for an XCTL) to ensure that the appropriate storage areas are freed. Control is given back to the returned-to program at the entry point originally specified at the time that program relinquished control via the LINK request.

PROCESSING PROGRAM TABLE SEARCH ROUTINE

The Processing Program Table Search routine locates the first entry in the PPT from an address in the CSA. It then compares the program identification supplied by the calling routine against the program identification contained in each PPT entry. When a match is encountered, the PPT address of the entry is returned to the calling routine in a register. If no match is found, this routine abnormally terminates the task.

The PPT entry associated with each nonresident program has pertinent information such as the relative location of the program in the CICS Real-Time Relocatable Program Library and the amount of storage required to load the program. For a resident program, the PPT contains the entry address to the module.

PROGRAM FETCH

The Program Fetch module fetches a nonresident program identified in the task's TCA by the macro expansion of LINK, XCTL, or ATTACH. The fetched program's PPT address and program entry point (provided by the PPT Search routine) are stored in the task's TCA.

If the fetched program is identified as a COBOL program in its PPT entry, the module exits to the COBOL INIT1 routine after initializing registers 12, 13, and 14 to COBOL convention usage. If the fetched program is an Assembler language program, the Program Fetch module exits directly to the entry point of the fetched program.

ASYNCHRONOUS RELOCATABLE LOADER

Programs are loaded from the CICS Real-Time Relocatable Program Library. Programs are brought into storage by means of OS BSAM reads followed by CICS waits to allow asynchronous (concurrent) processing of other tasks.

Loaded programs are brought into the data area of main storage. As all records for the module are being read, the Relocatable Loader relocates all relocatable address constants identified and pointed to by the RLD entries.

Once loaded (and relocated, if applicable) the module loader routine analyzes the module's PPT entry and determines whether the module is from a COBOL Compiler. If so, addressability between the program and CICS must be established. Base addresses, such as the location of the CSA and the CICS/COBOL Interface routine entry address, are moved to appropriate base locator cells in the COBOL module.

The Module Load routine returns control to the calling PCP routine.

PROGRAM RELEASE

This subroutine removes reference from the task's TCA to the PPT associated with the identified program. The subroutine then returns to the calling PCP routine.

COBOL INTERFACE ROUTINE

The COBOL Interface routine is a service subroutine which is entered as the result of certain CICS macro instructions being issued by ANS COBOL programs. Upon entry, the subroutine saves the COBOL program's registers 14 through 12 in the COBOL save area. These registers are loaded with CICS information (base registers for the CSA and the task's TCA). Register 14 is loaded with the entry address of the appropriate CICS management program.

The CICS management program is given control through a branch and link. The program returns control to the interface. The COBOL program's register contents are restored, and control is returned to the COBOL program at the reentry point specified in register 14.

PL/I INTERFACE ROUTINE

The PL/I Interface routine is a service subroutine which is entered as the result of certain CICS macro instructions being issued by PL/I programs. Upon entry, the subroutine saves the PL/I program's registers 14 through 12 in the PL/I Dynamic Save Area (DSA) and saves register 13 in register 11.

If the routine was entered due to the first call from a PL/I program, the CSA address is passed to the PL/I program. Whenever the routine is entered subsequently for the same task, control is passed to the called CICS management program through a branch and link. The program returns control to the interface which then restores the PL/I program's registers and returns control at the reentry point specified in register 14.

PROGRAM INTERRUPT CONTROL PROGRAM (DFHPIP) - CHART 6

The function of the Program Interrupt Control program is to intercept control when a program interrupt occurs in the CICS partition/region. This avoids, where possible, abnormal termination of the entire partition/region. This program also supports the optional Runaway Task feature of CICS.

This program is optional to the user. It can be included in the system by specifying the appropriate code at System Initialization. (For details, see the discussion of System Initialization earlier in this section.)

The operating system passes control directly to the Program Interrupt Control program when a program check occurs in the partition/region. The program will not handle the error when:

1. The program check occurs with a system task (Terminal Control or Task Control's TCA) in Control.
2. The program-checked task has just previously been intercepted by the Interrupt program prior to the program check.

For these exceptional conditions, a message is issued to the console operator and the partition is terminated. Otherwise, only the interrupted task is terminated.

Program Interrupt Control intercepts program checks in the CICS partition and attempts to abend only the task causing the program check.

The OS SPIE macro instruction establishing the program check exit is issued during system initialization if the Program Interrupt option was selected.

Upon entry to Program Interrupt Control, the TCA of the interrupted task is compared to see if the interrupted task is either Terminal Control or Task Control. A check is then made to see if this is the second entry to Program Interrupt Control for the same task. If any of these conditions is true, CICS is abended with a dump after writing a message to the console operator informing him of the reason CICS is being terminated.

If none of the above conditions exist, a test is made to determine whether the program interrupt was initiated by the runaway task detection logic of the Interval Control program. If not a runaway task, the task is abnormally terminated by altering the PSW in the Program Interrupt Element (PIE) supplied by OS and giving control to the Program Control program. If a runaway task, the operation code byte that caused entry to the Program Interrupt program is restored and the PSW in the PIE is altered to cause entry to the Runaway Task Flush routine linkage instructions in the CSA at CSAICRNX.

DUMP CONTROL PROGRAM (DFHDCP) - CHART 7

The purpose of the Dump Control program is to dump specified areas of storage as indicated in the Dump Control macro instruction. The dump is written out to tape or disk (to be printed later by the Dump Utility program). The Dump Control program may be called by either an application program or CICS management program to write out the contents of main storage at any time. It does not terminate the requesting task, but merely services its request and returns control to it.

The Dump Control macro instruction sets the "type request" switch in the TCA. The Dump Control program uses this switch to determine which areas of main storage are requested to be dumped. The Common System Area (CSA) and Task Control Area (TCA) are always the first areas to be dumped, Additional areas dumped, according to the user's request, include:

1. All transaction storage
2. All CICS storage
3. Both transaction and CICS storage
4. Part of transaction storage

The output is written to tape or disk. Records have undefined length and are written out as a continuous string of data. That is, a single record may be a CICS table, one TCA, or an entire program.

Each record has an identification record preceding it to identify the record for the Dump Utility program to print the Dump.

The following areas may be dumped if requested:

1. Task Control Area
2. Common System Area
3. Transaction Storage
3. Trace Table
5. Terminal Storage
6. Program and Register Save Areas
7. System Control Tables

TERMINAL CONTROL PROGRAM (DFHTCP) CHART 8

Terminal Management provides the facility for routing data between the terminals and the processing programs. This is accomplished by scanning the lines and the terminals to service read and write requests issued by processing programs and initiating polling of the terminals for new activity.

To initiate activity in CICS, control is passed to the Terminal Control program from the System Initialization program.

The Line Control routine scans each line for action and selects the terminal dependent module (TDM) for servicing each of the terminals on the line. When all lines have been analyzed, the terminal events are synchronized with other CICS events by issuing a WAIT macro instruction to the Task Control program. Line Control returns to repeat the line scan when a terminal control initiated event is completed.

The terminal dependent modules (TDM) scan each terminal on the line, and a logic path is determined for servicing terminal responses to initiated events and terminal control macro requests.

The terminal dependent modules (TDM) are:

- Sequential module (DFHTCSAM)
- 7770 module (DFHTC77S)
- Local 2260 module (DFHTC60L)
- 1030 module (DFHTC30N)
- 1050 module (DFHTC50N)
- Remote video module (DFHTC60N)
- 2740 non-switched module (DFHTC40N)
- 2741 non-switched module (DFHTC41N)
- System/7 module (DFHTCS7N)
- 1050 dial module (DFHTC50S)
- 2740 dial module (DFHTC40S)
- 2741 dial module (DFHTC41S)
- TWX module (DFHTCTWX)
- Bisync non-switched Group A module (DFHTCN70)
  - System 360/370 non-switched
  - System 360/Model 20 non-switched
  - 2770 non-switched
  - 1130 non-switched
  - System/3 non-switched
- Bisync non-switched Group B module (DFHTCN80)
  - 2780 non-switched
- Bisync non-switched Group C module (DFHTCN29)
  - 2980 non-switched
- Bisync dial Group A module (DFHTCS70)
  - System 360/370 dial
  - System/360 Model 20 dial
  - 2770 dial
  - 1130 dial
  - System/3 dial
- Bisync dial Group B module (DFHTCS80)
  - 2780 dial
- 3270 common module (DFHTC70C)
- Local 3270 module (DFHTC70L)
- Remote 3270 module (DFHTC70R)
- 2260 compatibility module (DFHTCCP)
- 3735 dial module (DFHTCS35)

When a positive acknowledgement to an input event is received on a line, the responding terminal is identified, and the input area is attached to the corresponding Terminal Control Table. For an output event, the output area is released unless the user has requested the area to be saved.

The Activity Control routine analyzes each terminal for servicing Terminal Control functional requests and macro requests. The functional requests are task initiation, and automatic transaction initiation. The macro requests are READ, WRITE, WAIT, SAVE, DISC, ERASE, LINE ADDRESS, etc.

The Task Initiation routine creates tasks through the use of the Task Control ATTACH macro instruction.

The preparation of input and output events varies with the type of terminal. Output events are initiated by request only. Requested input events are serviced only if a task is attached to the terminal. When requested write and read events are completed, the waiting task is made dispatchable. The line is polled after all terminal macro requests on the line have been serviced. Requested reads and polling events are evenly distributed to prevent a terminal from seizing the line.

When automatic transaction initiation is requested and a task is not attached to the terminal, a task is created for initiating output events to the terminal from a destination queue.

When all terminals have been analyzed on a line, control is returned to Line Control to scan the next line in the Terminal Control Table.

LINE CONTROL

Module: DFHTCCLC
Cross reference: DOCTCP01-05

The Line control routine scans each line for required action and selects the terminal-dependent module for servicing each of the terminals on the line. Synchronization of terminal events with other CICS events is provided by this routine.

Control is initially passed to the Terminal Control program from the System Initialization program to initiate activity in the CICS partition.

When the Terminal Control program receives control from the Task Control program, each line in the Terminal Control Table is scanned for possible activity to be performed.

If the error pending indicator is on, control is passed to the error handling routine which attempts to process errors which were delayed at a previous time. The processing of an error can be delayed if either the short-on-storage or maximum task indicator is on.

If a bisynchronous line requires action, control is passed to bisynchronous line analysis module. The line analysis module synchronizes the required bisynchronous events and responses. It also identifies the terminal for which action is required and passes control to a bisynchronous terminal-dependent module for processing. See bisync detail design description for more information.

If a start stop line requires action, a terminal dependent module is selected for processing and control is passed to Terminal Event Analysis. Each terminal associated with a line is analyzed for activity to be performed. After the processing of a line is complete, the next line in the Terminal Control Table is selected for analysis.

When all lines in the Terminal Control Table have been processed a WAIT is issued to the Task Control program to pass control to other tasks in the CICS partition. If Asynchronous Transaction Processing is being used, terminal control first performs the logical post necessary to activate the Asynchronous Transaction Control program before issuing the WAIT. Control is returned to the Terminal Control program, either upon the expiration of the specified system time interval or upon completion of a terminal event.

EVENT ANALYSIS

Module(s): DFHTCCSS,DFHTCSNC, DFHTCSSC, and TDM
Cross-reference: See flowchart for appropriate TDM.

Event Analysis is the entry point for each of the Terminal Dependent modules. A logic path is determined in this routine for servicing terminal responses to initiated events.

Event Analysis receives control from Line Control when a line requires action.

Control is passed to the Activity Control routine if an initiated line event is not completed. If an I/O error is detected, control is passed to the error handling routine.

Upon completion of each input event, the related terminal is identified by linking to the Terminal search routine. The terminal entry search routine matches the responding terminal with the corresponding terminal entry table. If a polling event or an initial type read for a dial line has been completed, the terminal entry tables are scanned to find the corresponding terminal. For single dropped lines, the scan is not performed.

If a positive response to an initiated line event has been received, control is passed to the corresponding Event Completion routine.

If a negative response is detected, the action taken depends on the type of event. If the type of event is a requested read event, control is passed to the Activity Control routine. If it is a write event, the output area must be prepared for retransmission of the data. (For example, with BTAM, the data is translated from transmission code to EBCDIC).

For polling events and buffered terminal write events, a negative response time delay is calculated. A time delay factor is added to the time of day and saved for subsequent checking. The polling time delay prevents excessive nonproductive polling. The addressing time delay allows a buffer to empty before issuing the next write event. Control is passed to the Activity Control routine when a negative response to polling or addressing has been detected.

INPUT EVENT COMPLETION

Module(s): DFHTCCSS and TDM's
Cross reference: See flowchart for appropriate TDM.

This routine receives control from the Terminal Event Analysis routine when a positive response is detected.

When an input event is completed without error, storage management follows two distinct paths of logic for completed input events.

For a completed multidropped polling event, the polling type storage is changed to terminal type storage and is placed in the terminal entry storage chain. Input areas for single dropped events and requested read events are obtained as terminal type storage before initiation of the event; the input areas are already in the storage chains (TCTTE). The length of the data in the input area is calculated and placed in a field preceding the data. The read request indicator, set by the user's macro request, is turned off. Data in the input area is prepared for the user. For example, with BTAM, the data is translated from transmission code to EBCDIC.

Module(s): DFHTC70R
Cross reference: DOCTCPTC

The Remote 3270 Input Event Completion routine automatically handles the multiple reads normally required to obtain a complete remote 3270 message. When the first block of a 3270 message is received, the 3270 Storage Use Analysis routine is entered to obtain a TIOA of a size equal to or greater than the size specified in the TCTTEBDL field of the TCTTE for the terminal. If a TIOA is available, the AID character in EBCDIC and the cursor position in binary are saved at the corresponding TCTTEAID and TCTTECAD fields. The rest of the data is then moved from the line I/O area to the beginning of the TIOA. If the first block was not terminated with an ETX character, multiple read continues are issued until a block terminated by ETX is received.

As each block of the message is received, it is added, minus the line control characters, to the end of the existing 3270 message. If 3270 message exceeds the size of the current TIOA, 3270 Storage Use Analysis is again entered to obtain a TIOA 500 bytes larger than the current TIOA; the message is then reconstructed in the new TIOA. When the last block of the message is read, the total length of the message is calculated and placed in the TIOATDL field and normal input event completion posting can then take place.

If at any time during the read in process a TIOA cannot be obtained or facilities are not available to initiate a new task for an initial input, an RVI is sent to the terminal to terminate transmission of the message and turn off the read pending indicator (TCTTERPI) so that a read modified can be scheduled at a later time when resources become available.

If 2260 compatibility has been generated and if a complete 3270 message has been received, a test is made prior to the linkages to the 3270 Transcode lookup routine and the Read Completion Posting routine to determine whether the transaction currently attached to the TCTTE is a 2260 based transaction. If it is, a linkage is taken to the 2260 Compatibility Read routine to convert the data to a 2260 data stream.

If the message receive is a 3270 status message, a test for the presence of device end status is made to determine whether the device busy flag (TCTTEDBI) should be turned off. If a terminal error occurs, CSTE is initiated to handle the terminal abnormal condition.

OUTPUT EVENT COMPLETION

Module(s): DFHTCCSS and TDM's
Cross reference: See flowchart for appropriate TDM.

The Output Event Completion routine is entered upon a normal completion of an output event from Terminal Entry Analysis.

When write events are completed, the write request indicator, set by the user's macro request, is turned off. If a read or save indicator is on, control is passed to the Activity Control routine. When a read or a save request is not indicated, the output area storage is released by issuing a Storage Control FREEMAIN macro instruction. If the terminal is not buffered, control is passed to the Activity Control routine.

When the terminal is buffered, a time delay is calculated to allow the buffer to empty before another write event is initiated. The time delay factor, determined by the length of the data in the buffer, is added to the time of day and saved for subsequent checking. Control is passed to the Activity Control routine.

ACTIVITY CONTROL

Module(s): DFHTCCSS, DFHTCSNC, DFHTCSSC
Cross reference: See flowchart for appropriate TDM.

The Activity Control routine determines a logic path for servicing terminal requests for each of the terminals on the line.

The Activity Control routine receives control from either the Terminal Event Analysis routine or the Event Completion routines. Each terminal on the line indicates a path of logic based on existing requests. Control is returned to analyze the next terminal. This routine examines terminal request indicators by priority. For example, write requests have priority over read requests.

Each terminal is selected for analysis according to its position in the Terminal Control Table.

If the task-to-initiate indicator is on, control is passed to the Task Initiation routine. This routine attempts to service a request for a task which was delayed at a previous time. For example, a task cannot be created if the short-on-storage or the maximum-task indicator is on. From the Task Initiation routine, control returns to analyze the next terminal on the line.

If a write request is indicated, control is passed to the Output Event Preparation and Initiation routine.

When a task does not exist on the terminal, control is passed to the Automatic Transaction Initiation routine.

The read request indicator is checked only if a task exists on the terminal. When a read is requested by the user, the specific read indicator is turned on for subsequent checking in the Test Line and Input Event Preparation and Initiation routines. Control proceeds to analyze the next terminal on the line. If the read request indicator is <u>not</u> on, the wait request indicator is reset and the task is made dispatchable, and Activity Control proceeds to analyze the next terminal on the line.

After all terminals on the line have been scanned for requested activity, control is passed to the Input Event Preparation and Initiation routine.

INPUT EVENT PREPARATION AND INITIATION

Module(s): DFHTCSNC and TDM's
Cross reference: See flowchart for appropriate TDM.

The Input Event Preparation and Initiation routine provides storage for an input area and passes the necessary information for initiating an input event to the access method.

If a line has been initiated, the line is busy and control is returned to the line control routine.

When the negative poll time delay has not elapsed, control is returned to the Line Control routine.

When the specific read indicator is on, a poll is issued to the terminal requesting the read. If this indicator is off, a poll is issued to all of the active terminals in the polling list. With multidropped lines, equal consideration is given to the polling of specific terminals with read requests and the polling of terminals without attached tasks.

If the save request indicator is on, the application program has requested exclusive use of the terminal storage data area and it cannot be reused by the current read request. If the terminal storage data area is reusable, the data set control area is prepared for a specific poll event. A reusable storage area must be of sufficient length and the save indicator must be off.

When the terminal storage data area is not reusable and the short-on-storage indicator is on, control is returned to the Line Control routine. When the terminal storage data area is not reusable and the short-on-storage indicator is off, a new data area is obtained by issuing a Storage Control GETMAIN macro instruction. After the data area is acquired, the data set control area is prepared for a specific poll event.

If the specific read indicator is off and polling storage area is present, the data set control area is prepared for polling the active terminals in the polling list. When the short-on-storage indicator is on, the polling storage area is not used and the polling event is not initiated. The polling storage area is released by issuing a Storage Control FREEMAIN macro instruction. The release of this area makes storage available for existing CICS tasks. Control is returned to the Line Control routine. This prevents transaction initiation requests from entering the CICS partition.

When a polling storage area is not present and the short-on-storage indicator is on, control is returned to the Line Control routine. When a polling storage area is not present and the short-on-storage indicator is off, a polling storage area is obtained by issuing a Storage Control GETMAIN macro instruction. After storage is acquired, the data set control area is prepared for polling the active terminals in the polling list.

The data set control area is a table of information which the access method uses to initiate an event. For example, it contains the maximum length of the data to be read, the address of the input area, the operation code, and the address of the polling list. Control is passed to the Access Method Read/Write routine to initiate an input event. Upon return, control is given back to the Line Control routine.

OUTPUT EVENT PREPARATION AND INITIATION

Module(s): DFHTCCOM and TDM's
Cross reference: See flowchart for appropriate TDM

The output Event Preparation and Initiation routine passes the necessary information to the access method for initiating an output event.

The Output Event Preparation and Initiation routine receives control from the Activity Control routine when a write operation is requested. If an input event is initiated, the line is busy and the write request cannot be serviced immediately. A reset poll list macro instruction is issued to the access method to terminate the initiated polling operation, and control is returned to the Line Control routine.

If an output event is initiated, the line is busy and another write request cannot be serviced immediately. Control is returned to the Activity Control routine.

If the line is available for an output event, and the terminal is buffered and if the time delay has not elapsed, control is returned to the Activity Control routine. If the buffered terminal time delay has elapsed, or if the terminal is not buffered, the data set control area is prepared for addressing the terminal. The data set control area is a table of information which the access method uses to initiate an event. For example, it contains the length of the data to be written, the address of the output area, the operation code, and the location of the addressing list.

Data in the output area is prepared for transmission to the terminal. For example, with BTAM, the data is translated from EBCDIC to transmission code. Control is passed to the Access Method Read/Write routine to initiate an output event. Upon return, control is given back to the Activity Control routine.

AUTOMATIC TRANSACTION INITIATION

Module(s): DFHTCCOM
Cross reference: DOCTCP08

The Automatic Transaction Initiation routine provides the logic for creating tasks automatically within CICS.

The Automatic Transaction routine receives control from the Activity Control routine when a task is not currently attached to a terminal. If the automatic output indicator is off, control is returned to the Activity Control routine.

If either the short-on-storage indicator or maximum-task indicator is on, control is returned to the Activity Control routine.

If Task Control has indicated that a transaction needs to be initiated on the terminal, a DFHKC TYPE=AVAIL macro instruction is issued which causes Task Control to pass the identification of the transaction to be ATTACHED.

When an input event is initiated, the line is busy and a task which requests output events cannot be created immediately. A reset poll list macro instruction is issued to the access method to terminate the initiated polling operation, and control is returned to Line Control routine.

If an input event is not initiated, the Destination Control Table is scanned to find the table entry which corresponds to the terminal. This table entry contains the transaction code which is placed in the Task Control Area and passed to the Task Initiation routine. This routine attempts to create a task to satisfy the automatic transaction request. From this routine, control is returned to the Activity Control routine.

TASK INITIATION

Module(s): DFHTCTI
Cross reference: DOCTCP13

The Task Initiation routine manages the creation of tasks from terminal requests.

This routine receives control from the Activity Control routine, the input event completion routine or the Automatic Transaction Initiation routine. If a task currently exists on a terminal, control returns to the requesting routine.

If the short-on-storage indicator is on, or if the maximum-task indicator is on, the task-to-initiate indicator is turned on and control is returned to the requesting routine.

When the short-on-storage and the maximum-task indicators are off, a task is initiated by issuing a Task Control ATTACH macro instruction. After the task is created, the task-to-initiate indicator is turned off and control is returned to the requesting routine.

For a 3270 display which is in compatible mode, the input buffer is scanned for an SMI character and the four characters following the SMI are used as the transaction ID. When a 3270 display is not in compatible mode, the beginning of the TIOA data area is examined for the transaction code. If an SBA is found in the first position, three characters are skipped. If the compatible terminal flag is on, a check is made to determine if the first true data character is an SMI character. If present, it is skipped. The ensuing four characters are taken as the transaction code.

Before a task is ATTACHed, the TCTTETC field is checked to see if there is a default transaction code present. If there is anything but hexadecimal zeros in the field, the four characters are used as the transaction code. Otherwise, the transaction code from the data stream is used. The TCTTETC field is retained across transactions. If not, the field is reset to hexadecimal zeroes.

ERROR HANDLING

The error handling routine prepares error codes and information for passage to the terminal abnormal condition program. This routine receives control from numerous points in terminal control where error conditions have been detected. A terminal abnormal condition program task is initiated for processing of errors and extended user error recovery in the terminal error program.

For an output event, the data area must be prepared for retransmission. For example, with BTAM, the data is translated from transmission code to EBCDIC. This allows data to be transferred to another type of terminal when an unrecoverable I/O error occurs.

When the short-on-storage indicator or the maximum task indicator delays error handling preparation and task initiation, an error pending indicator is placed in the line entry table. This indicator gives top priority to processing errors before initiating new activity on the line.

GET TERMINAL STORAGE

Module(s): DFHTCORS
Cross reference: DOCTCP06

The Get Terminal Storage routine is entered when a read request has been issued, and is entered from Input Event Preparaton and Initiation routine.

This routine determines if the storage that is attached to the terminal can be reused or if new storage must be obtained. If storage is attached and the save indicator is on, new storage is obtained. If the storage that is attached is smaller than the requested line I/O area specified in the line entry, the storage is freed and a new area of storage is obtained. The starting location in the storage area into which the data is to be read is determined by device type. Control is returned to the requesting routine.

FREE TERMINAL STORAGE

Module: DFHTCORS
Cross reference: DOCTCP07

The Free Terminal Storage routine is entered upon a request from one of the other Terminal Control modules to free terminal on-line storage.

Storage can be freed one piece per request or all attached storage freed in one request. The Free Terminal Storage routine issues a Storage Control FREEMAIN to free attached storage. The Free Terminal Storage routine returns control to the requesting routine.

INPUT DATA LENGTH COMPUTATION

Module: DFHTCCOM
Cross Reference: DOCTCP09

The Input Data Length Computation routine is entered from the Input Event Completion routine.

The Input Data Length Computation routine computes the length of the data read and places the length of the actual data in the I/O area length field.

If a start symbol is not received from a 2260, the START SYMBOL MISSING message is sent to that device. If the data received is greater in length than the terminal I/O area, the MESSAGE TOO LONG message is sent to that device.

EVENT INITIATION

Module: DFHTCCOM
Cross Reference: DOCTCP10

The Event Initiation routine is entered from either the Input Event Initiation or the Output Event Initiation routine.

This routine sets up the appropriate access method read/write routine and posts the line as initiated. Upon completion of starting the event, control is returned to the requesting routine.

TRANSLATE

Module: DFHTCCOM
Cross reference: DOCTCP15

The Translate routine is entered from the (1) Input Event Completion routine to translate the received data from the transmission code to EBCDIC, (2) Output Event Preparation routine to translate the data to be sent from EBCDIC to the correct transmission code, and (3) Error handling routine to retranslate from the transmission code to EBCDIC on an error. Upon completion, control is returned to the requesting routine.

EVENT TERMINATION

Module(s): DFHTCCOM
Cross reference: DOCTCP11

The Event Termination routine is entered to terminate a polling event on a line.

If the interruptable read indicator is on, a RESETPL macro is issued and the interruptable read indicator is turned off. Control is then passed to the Line Control routine. If the RESETPL macro is not completed successfully, control is passed to the Line Control routine, leaving the interruptable read indicator unchanged. If the indicator is not on, control is returned to the requesting routine.

BINARY SYNCHRONOUS LINE ANALYSIS

Module(s): DFHTCCBS
Cross reference: DOCTCPJA

A determination of line type, switched or non-switched is made. Control is then passed to the appropriate module.

DIAL ENTRY ANALYSIS

Module(s): DFHTCSBS
Cross reference: DOCTCPKA

If an event for the line has completed, control is passed to event completion analysis. If an event has not completed or no event is outstanding, control is passed to the Terminal Scan routine.

DIAL EVENT COMPLETION ANALYSIS

Module(s): DFHTCSBS
Cross reference: DOCTCPKB

A determination of the last operation is made and if an initial type read, control is passed to terminal search. If not read initial, the address of terminal is known from line, and control is passed to dial device determination.

DIAL TERMINAL SCAN

Module(s): DFHTCSBS
Cross reference: DOCTCPKD

If line is connected to terminal, control is passed to dial device determination. If line is not connected to a terminal the first terminal is made addressable and control is passed to line device determination.

DIAL TERMINAL ANSWERBACK SEARCH

Module(s): DFHTCSBS
Cross reference: DOCTCPKC

The terminal address that is received is compared with entries in TCT to find terminal entry. If terminal is not found a disconnect is issued. Once the ID is found control is passed to dial device determination.

DIAL EXPANDED ID VERIFICATION

Module(s): DFHTCSBS

Cross reference: DOCTCPKF

The address of the entry in the terminal list containing the received identification sequence is passed by BTAM to CICS. If the user portion of this entry contains a valid TCTTE address, control is passed to Dial Device Determination. If it does not contain a TCTTE address, control is passed to Dial Terminal Answerback Search.

DIAL DEVICE DETERMINATION

Module(s): DFHTCSBS
Cross reference: DOCTCPKE

Terminal type is determined and control is passed to appropriate terminal module.

NON-DIAL ENTRY ANALYSIS

Module(s): DFHTCNBS
Cross reference: DOCTCPMA

If an event for the line has completed, control is passed to event completion analysis. If an event has not completed or no event outstanding, contrcl is passed to terminal scan routine.

NON-DIAL EVENT COMPLETION ANALYSIS

Module(s): DFHTCNBS
Cross reference: DOCTCPMB

A determination of last operation is made and if read initial, control is passed to terminal search. If not read initial the address of terminal is known from the line and is set up before passing control to non-dial device determination.

NON-DIAL TERMINAL SCAN

Module(s): DFHTCNBS
Cross reference: DOCTCPMD

If line is in use and event not completed control is passed to device determination, with terminal address in line entry that is connected to line. If no event outstanding the first terminal on line is set up (if in service) and control is passed to device determination.

NON-DIAL TERMINAL SEARCH

Module(s): DFHTCNBS
Cross reference: DOCTCPMC

A search of control unit on general poll devices and first terminal is set up and control is passed to device determination. A search of terminal on non-general poll devices is made and control is passed to device determination.

NON-DIAL DEVICE DETERMINATION

Module(s): DFHTCNBS
Cross reference: DOCTCPME

Terminal type is determined and control is passed to the appropriate terminal module.

READ AND WRITE ROUTINES

Module(s): DFHTCCBS
Cross reference: DOCTCPOA

These routines set up the requested BTAM operation.

LOGICAL READ ROUTINE

Module(s): DFHTCCBS
Cross reference: DOCTCPOB

An analysis of the line I/O area is made to determine if data still resides in this area. If data is present in the line I/O area, it is moved to the terminal I/O area. If no data is present, a READ is issued to request more data.

TERMINAL ADVANCE ROUTINE

Module(s): DFHTCCBS
Cross reference: DOCTCPOC

An analysis of the line activity and line request is made to determine the next action to be taken on that line.

DYNAMIC OPEN/CLOSE PROGRAM (DFHOCP) - CHART 9

The Dynamic Open/Close program provides open/close capabilities for Dump Control data sets, Transient Data extrapartition data sets, and File Control data sets.

ENTRY ANALYSIS

Entry Analysis determines the type of request (TRANSDATA, DATA BASE, or DUMP) and gives control to the proper Open/Close routine.

Entry Analysis also determines if the task giving control to the Open/Close program is nonpurgeable for stall detection and runaway task detection. If the task is purgeable, it is made nonpurgeable upon entry to Open/Close. Storage is acquired to save the type of response and to preserve the address of the open/close parameter list passed to Open/Close if the request was for DATA BASE or TRANSDATA.

DUMP CONTROL ENTRY

Dump Control Entry determines if the request is a valid open/close/switch request for the dump data set. If it is not a valid request, an invalid request response is returned to the task calling the Open/Close program. If the type request is Open, the Open routine is given control.

DUMP CONTROL CLOSE REQUEST

If the dump data set has been previously closed, no action is taken and control is given to the Switch routine. If the dump data set is currently open, it is closed and the "Dump Control data set closed" indicator in the Dump Control program is turned on. This prevents Dump Control from attempting to take any further dumps.

DUMP CONTROL SWITCH REQUEST

This code is entered after any close request to check for a switch type request. If the request is for a switch of the dump data set, the ddname in the DCB is altered to point to the alternate Dump Control data set. If the alternate Dump Control data set was opened previously, the ddname in the DCB is altered to point to the primary data set.

If the dump data set is contained on tape, the same switching technique is used. Two DD cards are needed, one containing the ddname DFHDMPA and the other containing the ddname DFHDMPB. DFHDMPA is considered the primary dump data set.

DUMP CONTROL OPEN REQUEST

The dump data set is opened, and the Note/Point information in Dump Control is zeroed to indicate Dump Control is to start writing at the beginning of the data set. The "Dump Control data set closed" indicator is turned off in the Dump Control program.

If the dump data set was previously open, no action is taken.

TRANSIENT DATA ENTRY

The Transient Data LOCATE macro instruction is issued to locate the Destination Control Table entry associated with each of the requests in the Open/Close parameter list. If a dummy program response is received from Transient Data, the Invalid Request response is set and control is returned to the calling program.

TRANSIENT DATA OPEN

The DCT is tested to see if the extrapartition entry to be opened is resident or nonresident. If the destination is resident, no control blocks need be loaded and the DCB address is inserted in the open list in the Destination Control Table.

If the destination is nonresident, the PPT is searched to ensure that the control blocks to be loaded exist in the PPT before a Program Control LOAD is issued.

The suffixed data set control block is loaded and the address and Open Option byte are inserted in the DCT open list.

If an override list is specified, the DFHTRNDY control block is loaded and initialized with the parameters specified in the override list.

When the list of data sets to be opened is complete, control is given to the Common STAE Open/Close routine.

TRANSIENT DATA CLOSE

A list of DCB's to be closed is constructed and control is given to the STAE Open/Close Interface routine. Upon completion of the close, the DCT open list is reconstructed to delete the DCB being closed. The destinations closed are then examined to see if they are nonresident. If any of the DCB's are nonresident, the DCB's are freed.

DATA BASE ENTRY

A File Control LOCATE is issued to locate the File Control Table entries for the Open/Close parameter list. If a dummy program response is returned from File Control, the Invalid Request response is set and control is returned to the calling program.

DATA BASE OPEN

The Dynamic Open/Close program locates the FCT entries for the requested opens via the locate function of the File Control program (FCP) and builds an open list of DCB's. Control is then given to the STAE Open/Close Interface routine.

Upon completion of OS OPEN, the FCT open list is reconstructed to reflect all open data sets. A File Control open request is then made to logically open the data sets for use by CICS application programs.

DATA BASE CLOSE

The Dynamic Open/Close program first issues a File Control logical close to close the data sets for CICS application programs. It then builds a list of DCB's to be closed and gives control to the STAE Open/Close Interface routine. Upon return, the Open/Close program rebuilds the FCT open list to reflect only open data sets.

COMMON EXIT

The Common Exit routine makes the calling task purgeable for system stall detection and for runaway task detection if it was purgeable when the Open/Close program was entered.

The Type response and Open/Close parameter list addresses are placed in the TCA and a Program Control RETURN issued to return to the calling program.

STAE OPEN/CLOSE INTERFACE ROUTINE

The STAE Open/Close Interface routine is entered by all open/close routines to issue an OS STAE SVC to establish a STAE exit to prevent abends. The issued STAE request calls for no purging of active I/O and allows asynchronous (concurrent) exits.

If no storage is available for a STAE control block, no open/close processing is done; a "no storage" response is placed in the TCA and control is returned to the calling program via a Program Control RETURN. If the STAE exit is successfully established, the type of request field is tested to see if the request was for open or close; the appropriate SVC is then issued.

Upon completion of OS OPEN/CLOSE processing, the STAE exit is cancelled.

STAE EXIT ROUTINE

The STAE Exit routine checks the completion code passed for the OS ABEND that has occurred and determines if a retry should be scheduled. If no retry is to be attempted, control is returned to the operating system to complete ABEND processing. If a retry is to be attempted, control is returned to the operating system with the address of the Retry routine and an indication that a retry is to be scheduled with a purge of the RB chain.

STAE RETRY ROUTINE

The STAE Retry routine reestablishes the SPIE exit for CICS to intercept program checks. It then returns control to the Open/Close routine that entered STAE processing.

LOAD ROUTINE

This routine issues the Program Control load and checks to ensure the table was loaded.

PPT SCAN ROUTINE

This routine scans the PPT to ensure the table or program to be loaded is in the PPT before a load is issued.

FILE CONTROL PROGRAM (DFHFCP) - CHART 10

The File Control program performs the logical processing for the control of data set (file) operations. File Control reads and writes user data sets, utilizes user-established indexing procedures (indirect accessing), gathers statistical data, and acquires and releases main storage for data set operations.

This program uses the OS Basic Direct Access Method (BDAM) and a Basic Indexed Sequential Access Method (BISAM). Interface to these access methods is established through standard OS DCB's generated from information supplied by the user when the File Control Table is generated.

A symbolic data set name is included in the data set dependent information required to generate a File Control Table entry (FCTE). The data set name is supplied to File Control by the user in each request for services and is used to find the related File Control Table entry. This symbolic name is the same as the ddname used in the job control DD statement which defines the data set.

ENTRY ANALYSIS

The Entry Analysis module of File Control performs housekeeping functions such as saving the requesting task's registers, establishing File Control addressability, etc. Depending upon the request, the validity of the request is verified and control is given to the appropriate servicing logic.

Upon entry to this submodule, the requester's registers are saved in the requester's TCA at TCAFCRS. Addressability for File Control is then established.

The request indicator (TCAFCTR) in the TCA is tested. A branch is made to the proper logic based on whether the request is for RELEASE, PUT, GET, GETAREA, BROWSE, or OPEN/CLOSE/LOCATE services. If the request is not for any of these services, an invalid request indicator is set in the requester's TCA and return is made to the requesting program.

If the request is for GET, GETAREA, or SETL services, the File Control Table (FCT) is searched for the File Control Table entry (FCTE) of the data set name specified in the request. The FCT contains one entry for each data set which File Control accesses. The entries in the FCT are created from user-supplied definition statements. Among other data set dependent information, the FCTE contains a data set identification (name) which is specified in requests involving the data set. The FCT search is made on the data set Identification available in the requester's TCA from the macro expansion. If the referenced FCTE cannot be found in the FCT, an invalid data set identification indicator is set in the requester's TCA and return is made to the requesting program.

If an FCTE is found for identification specified in the GET, GETAREA, or SETL request, a check is made to ensure the data set is open. If open, the address of the FCTE is placed in the FCTE base register and the appropriate service routine is entered. If the data set is not open, control is returned to the user with an error code indicating the data set is closed.

FILE CONTROL TABLE SEARCH

If a request for a GET, GETAREA, or SETL is made, the File Control Table (FCT) is searched for the data set name specified in the request. if the referenced File Control Table entry cannot be found in the FCT, an invalid data set identification indicator is set in the requestor's TCA and return is made to the requesting program.

RETRIEVE A RECORD FROM A DATA SET (GET)

The DFHFC TYPE=GET macro instruction conveys to File Control the need to obtain a record from a data set defined in the FCT. The request (user-issued macro instruction) identifies the data set by name and specifies an area in the requesting program which contains the Record Identification field (for example, key, relative track and key, actual address, etc.). An indication must be made in the request if this requested record is to be updated and rewritten. The update intention must be indicated to File Control for protection against concurrent update of a record by another transaction.

Event preparation consists of acquiring an I/O area in main storage of sufficient size (defined in the FCTE) to accommodate the maximum block size contained in the referenced data set plus some event-dependent data areas. This area also contains the DECB and is defined using the FIOA DSECT (symbolic storage definition).

Request-dependent information (for example, I/O area address, requester's key area, etc.) is placed into the DECB portion of the FIOA. If the request is a GET for update, and the user has specified the exclusive control feature for the data set in the File Control Table, the Exclusive Control routine is used to prevent concurrent update to the same record by another transaction.

The proper access method is entered to execute the requested read operation. The reading is performed at the basic (read/write) level. The CICS WAIT macro instruction is invoked to await completion of the operation.

After acquisition of the desired record, a File Work Area (FWA) may be acquired and the record, or requested portions (segments), placed into it. Whether or not a FWA is acquired depends upon the type of operation and the type of record being retrieved. An FWA is always used when the request is for a read-with-update operation. If the request is for read-only (that is, no update), the FWA is used if the records are blocked (and deblocking is requested) or if the records are segmented.

The following table summarizes the use of the FWA and the FIOA when returning records to the user. (For each condition indicated, the corresponding action is determined by an "X" in the same vertical column.)

```
*************************************************************
* CONDITION                                                 *
*   Update Request                  X                       *
*   Read-only/Blocked/Deblocking        X                     *
*   Read-only/Blocked/No deblocking      X                   *
*   Read-only/unblocked                   X                 *
*   Read-only/segmented                     X               *
*   Read only/unsegmented                     X             *
*************************************************************
* ACTION                                                    *
*   Data Returned in FIOA               X X   X             *
*   Data Returned in FWA            X X     X               *
*************************************************************
```

The File I/O Area (FIOA) is always released unless it is being used to pass the record back to the user, unless the request was for a read-for-update, or unless an I/O error occurred.

Retrieve a Segmented Record

One feature of the File Control program is the control of segmented records. The user may define data set records as collections of segments. A group of segments (one or more) is defined as a segment set. The definition of these segment sets is specified by the user at FCT generation time and is included as an appendage to the appropriate FCTE. Each segment set is symbolically named by the user and identified by its name in any GET request for which only the information of a defined segment set is desired. The record is retrieved from the data set and the designated segment set is located and moved to a File Work Area (FWA) to be passed back to the user.

Retrieve a Record Through Indirect Accessing

Another feature of the File Control program is the technique of indirect accessing. A File Control Table entry (FCTE) is identified as an index data set, and information is included to describe the index data set record which contains, among other items, a pointer to a record in the next data set to be read. A user requesting a record via indirect accessing identifies both the index data set and the ultimate (target) data set by name. The user also provides the Record Identification of the record required from the index data set (the initial record).

File Control reads the index data set for the record identified in the request, obtains the record, and extracts from it the Record Identification used to access the next data set. The symbolic identification of the next data set referenced is obtained from the FCTE of the index data set. This process continues until the data set name to be read (as contained in the index data set FCTE extension) is the same as the target data set name in the user's request. The next read then acquires the record desired by the requester. This feature provides the facility for a user to reference data sets sequenced by other than the request reference.

UPDATE OR ADD DATA TO A DATA SET (PUT)

A DFHFC TYPE=PUT macro instruction may be issued to write an updated record acquired by a previous GET request, or to write (add) a new record to an already existing data set. In the case of adding a new record to a data set, an I/O area is obtained by File Control. Request-dependent information (for example, I/O area address, work area address, etc.) is placed into the DECB portion of the FIOA.

If the request is for an update to an ISAM data set, File Control determines if another update occurred on the data set since the record to be updated was retrieved by a previous GET. If another update did occur, File Control rereads the specified record to ensure update integrity.

The updated record (provided by the user in the FWA) is then moved to its proper location in the FIOA, and the appropriate form of the WRITE macro instruction is issued. A CICS WAIT macro instruction is issued to await completion of the I/O event. The areas associated with this request (I/O areas, work areas) are released in the Storage Disposition routines, and exclusive control (if obtained) is released through a CICS DEQ macro instruction or an OS RELEX macro instruction.

Update or Add Data to Segmented Records

The Put Segment Services routine packs requested segments into a physical record and calculates the length for the output routines.

RELEASE FILE DATA (RELEASE)

The DFHFC TYPE=RELEASE macro instruction causes the Storage Disposition routine of File Control to release a file storage area and release exclusive control, if applicable. An example of the use of this macro instruction is a record obtained by a File Control GET request and identified as being for update. The GET request for update places the record under exclusive control (if the user so specifies) and does not release this control until the related PUT is executed. If a user obtains a record with a GET for update and then decides not to rewrite it, he must issue a RELEASE macro instruction identifying the FWA in which he received the record. Control information placed in a reserved section of the FWA is used to release the exclusive control. This frees the areas (that is, FIOA and FWA) obtained by the GET and normally released by the PUT.

OBTAIN A FILE WORK AREA (GETAREA)

When a user wishes to add a new record to an already existing data set, he must acquire a File Work Area (FWA) in which to build his new record via a DFHFC TYPE=GETAREA macro instruction. The user must identify the data set in his request. File Control, using information available in the referenced FCTE, acquires an area sufficient to contain the record and returns the address of this area to the requesting program. This area address is subsequently specified to File Control in a PUT request.

OBTAIN EXCLUSIVE CONTROL OF A RECORD DURING UPDATE

When a read-for-update request is made, the File Control program checks the FCTE for the data set specified to see if the user wants the records placed under exclusive control during the update operation. If so specified, File Control provides exclusive control at the logical record level for ISAM data sets and at the physical record level for DAM data sets. For ISAM data sets, the Exclusive Control module uses the CICS ENQ macro instruction to enqueue upon a unique name which is constructed by concatenating the symbolic data set name with the record identifier (for example, key, TTR, MBBCCHHR). For BDAM data sets, an OS READ EXCLUSIVE is issued to provide exclusive control.

This technique prevents concurrent updates of the same record by more than one user, yet allows simultaneous access to a data set by more than one transaction during a conversational update operation.

OPEN/CLOSE/LOCATE A DATA SET

If the request is for OPEN, CLOSE, or LOCATE, the File Control program utilizes a user-provided parameter list to perform the specified function. This list consists of any number of twelve-byte entries containing the eight-byte symbolic data set name and a four-byte address field which is filled in by File Control. Regardless of the function required, each data set specified in the parameter list is located in the File Control Table (FCT) and its File Control Table entry (FCTE) address is placed in the four-byte field of the parameter list. If the symbolic data set name does not exist in the FCT, binary zeros are placed in the four-byte field of the parameter list.

In addition, as each FCTE is located, File Control sets an open/close indicator if the request was for for an OPEN or CLOSE. The indicator logically opens or closes a data set, thus allowing or preventing access to it by the File Control program. No provision is made within File Control to physically OPEN or CLOSE the data set; the user must do this by using the CICS Open/Close service program through the Master Terminal function.

INITIATE BROWSING (SETL)

The DFHFC TYPE=SETL macro instruction is used to initiate a browse operation on any data set defined in the File Control Table. The FCT is searched to verify that the data set ID is valid. If a valid data set entry is found in the FCT, its address is loaded into the FCTE base register and control is given to the SETL Processing routine.

The SETL Processing routine first acquires a File I/O Area (FIOA) large enough to process the largest block on the data set. Next a File Browse Work Area (FBWA) is acquired, initialized, and chained to the FIOA. The initial block identification or key (as specified in the user's Record Identification field) is preserved in the FBWA for use on subsequent GETNEXT requests.

The SETL Processing routine next acquires a File Work Area (FWA) using the Segment Services routine to determine the length required if the data set is segmented. The FWA is flagged to indicate it is associated with a browse operation, the FIOA is chained to it, and the FWA address is placed in the TCA at TCAFCAA. Control is then returned to the user.

RETRIEVE NEXT SEQUENTIAL RECORD (GETNEXT)

The DFHFC TYPE=GETNEXT macro instruction is used to acquire the next sequential record in a browse operation. The user must have previously issued a SETL request and placed the FWA address in the TCA at TCAFCAA.

The GETNEXT routine first ensures that the address passed at TCAFCAA is a browse FWA by checking the appropriate browse flag set by the SETL request. If not a valid FWA, an "invalid request" condition code is returned to the user.

The GETNEXT routine then locates the FIOA and FBWA associated with the FWA (the FWA points to the FIOA which points to the FBWA).

Control information contained in the FBWA indicates whether a new block of records must be read or if the next sequential record can be extracted from the current block in the FIOA. If all the logical records in the current block (assuming blocked records) have not been presented to the application program, the next record is extracted from the block and either placed in the FWA or presented to Segment Control (discussed below).

If conditions indicate that a new block of records must be read from the data set (that is, current block exhausted, unblocked records, or first block to be read), the GETNEXT routine determines the data set organization.

If an ISAM data set is being browsed, the key of the last logical record presented to the user (maintained in the FBWA) is incremented by a binary 1. A basic ISAM (random) READ is then issued to retrieve the next block of records. Because of the technique of using basic ISAM (that is, random type read) instead of sequential ISAM (that is, queued) to perform I/O operations associated with ISAM browsing, certain restrictions are associated with the CICS browse feature. The addition of a binary 1 to the key of the last logical record processed may create a key which does not exist in the data set. It is therefore essential that the access method use a "search key high/equal" when performing an I/O operation on the data set. It is necessary for the user to define all ISAM data sets subject to browsing as "blocked". Therefore, all data sets to be browsed are automatically defined as blocked during FCT generation.

The necessity for a "search key high/equal" also exists when the access method retrieves blocks from the ISAM overflow area.

OS/360 Release 20.1 contains a change to the ISAM access method that will cause overflow records on a blocked data set to be retrieved with a "search key high/equal". This means that any OS user of CICS browse must be using an operating system at least as current as Release 20.1 of OS/360.

Since it is likely that a "no record found" indication will be returned by the access method, CICS File Control ignores this indication when in browse mode and does its own deblocking of blocked records. Because of this, all ISAM data sets that are to be browsed must have embedded keys within each logical record.

If it is necessary to read a new block of records and a DAM data set is being browsed, the block ID of the next block is extracted from the FBWA and a basic DAM read is issued.

Once the GETNEXT routines have located the next sequential logical record, a check is made to determine if the records are segmented. If the data set was defined as containing segmented records, CICS expands the segments into the FWA using the Segment Set Identification specified by the user in the GETNEXT request. If the user does not specify a Segment Set Identification in the GETNEXT request, the default Segment Set Identification as specified in the SETL request is used. If the user fails to specify any Segment Set Identification at SETL time and the data is segmented, the actual packed record is returned to the user.

If the user specifies a segment set name in a GETNEXT request different from the one specified in the SETL, and the new segment set requires more main storage than the old segment set, File Control releases the old FWA and acquires a new one large enough to process the new segment set. The address of the new FWA is passed back to the user.

Once the next logical record has been placed in the FWA, the user's Record Identification field is updated to show the key or block ID of the record. The user may use this updated Record Identification field to make a random read-for-update request when a desired record is located through browsing. Any random File Control request does not affect the browse operation.

Control is returned to the application program at the instruction following the DFHFC TYPE=GETNEXT macro instruction.

TERMINATE A BROWSE OPERATION (ESETL)

The ESETL routines first ensure that a file browse operation was previously initiated by a DFHFC TYPE=SETL macro instruction. It is done by verifying that the FWA provided as part of the ESETL request is really a browse FWA. If not, an invalid request indication is returned to the user.

If a valid FWA was specified, the browse operation associated with that FWA is terminated by freeing the FWA, the FIOA, and the FBWA. This is done by using the File Control Storage Disposition subroutine.

The termination of one browse operation does not affect other browse operations which have been initiated by the same transaction or by other transactions.

Control is returned to the application program at the next instruction following the DFHFC TYPE=ESETL macro instruction.

RESET A BROWSE OPERATION (RESETL)

A DFHFC TYPE=RESETL macro instruction causes an existing browse operation to be reset to some other location in the data set. It is functionally equivalent to issuing an ESETL and another SETL with new arguments. However, the RESETL accomplishes this without the overhead of freeing and reacquiring the FIOA and FBWA. The FWA is always freed and reacquired to ensure a correct size in case segmented records are being browsed.

TRANSIENT DATA CONTROL PROGRAM (DFHTDP) - CHART 11

The Transient Data Control program maintains the queues for intrapartition and extrapartition data. Requests for retrieval (GET) and disposition (PUT) of data to these queues are serviced by this program.

The transient data queues (destinations) are defined by the user in the Destination Control Table (DCT). The user includes a symbolic destination name for each destination defined, and all references to a transient data queue are made using the symbolic name. The destination is identified with intrapartition or extrapartition data.

ENTRY ANALYSIS

The Entry Analysis routine performs the initial housekeeping for Transient Data Control such as saving the calling program's registers and establishing addressability for Transient Data Control. Entry Analysis is entered via the Transient Data GET, PUT, FEOV, and LOCATE macro instructions.

A search of the DCT is made for the referenced destination. If no DCT entry can be found for that destination, an error indicator is set in the calling program's TCA and control is returned to the requesting program.

When a matching DCT entry is found, tests of an indicator within the DCT are made to determine whether the destination is extrapartition, intrapartition, or indirect.

If the destination is extrapartition, a branch is made to the Extrapartition Data routine.

If the destination is intrapartition, a branch is made to the Intrapartition Data routine.

If the destination is indirect, the new destination identification is moved to the TCA and the DCT Scan routine is reentered.

If the destination is not extrapartition, intrapartition, or indirect, the requesting task is terminated by a Task Control ABEND.

INTRAPARTITION

The Intrapartition submodule performs read and write requests for data in the referenced queue (destination). The queues are maintained on a direct access storage device. Retrieval and disposition of data involving an intrapartition destination is performed on a first-in/first-out basis. The location of the next record to be read and the next location available for a write are maintained in the DCT entry for each destination. Space for queues is obtained one track at a time from the Transient Data track pool. The chain record (on the track) points to the next track in the queue when retrieving data. Each destination has its own dynamic chain.

A count of the number of active records (records written but not retrieved) is maintained for each destination. When the count reaches the user-supplied trigger level (optional DCT entry by destination) and no task has already been initiated by this condition, Transient Data causes a user-specified task to be initiated (automatic transaction initiation) if the ultimate destination is not a terminal. If the ultimate destination is a terminal, the terminal's entry in the Terminal Control Table is flagged. This indicates that Terminal Control is to initiate the task when the required terminal is available (has no task attached).

Intrapartition queues are provided for the passing of data associated with a task and resource to some other task and resource. Message switching is a common application which makes use of this facility.

Read Intrapartition Data (GET)

A test is made of the DCT entry field which contains the count of the number of active records in this queue (destination). If the count is zero, the flag in the DCT entry for automatic transaction initiation is turned off, indicating that a task is not to be initiated. An indicator in the requesting program's TCA is set to indicate that the queue was empty, and return is made to the requesting program.

If the field containing the number of active records in the queue is not zero main storage for an I/O area is obtained from the Storage Control module by issuing a CICS GETMAIN macro instruction. The DECB is initialized with the I/O area address, and a direct access read is executed for the TTR of the next record to be read for this destination. The TTR of the next record to be read for each destination is maintained in each destination's DCT entry. A CICS WAIT macro instruction is executed to await completion of the read.

After successful completion of the read, the record just read is checked to determine if it is the chain record. If so, the TTR for forward chaining (next track address) and the record count for the current track are placed in the DCT entry. The TTR for the current input track is then incremented to the next record number (R portion of TTR). A read (direct access) is executed on that TTR and synchronized with a CICS WAIT. Upon completion of the WAIT, the count for the number of active records (records not read) in the destination queue and the number of active records on the current input track are decremented by one. The field containing the number of active records cn the track is checked for zero count before exiting, if it is zero, the reuse flag (DESTID Option) is checked. If on, a return is made to the application program. If off, the transient data space management submodule is entered which subsequently releases the exhausted track and returns it to the intrapartition track pool for reuse.

Intrapartition Space Management Routine

This submodule is responsible for controlling the intrapartition space allocated at system initialization time for Transient Data. The space is allocated or de-allocated in whole track increments as necessary. This routine is entered under the following conditions:

1. A write is issued for a particular destination queue when there is insufficient space on the current output track for the data record. In this case a new track is acquired and chained to the preceding track in the queue (DOCTDP16).

2. Upon reading a track to completion and the destination has reusable tracks, the space module is entered to release the track for re-use (DOCTDP14).

3. The Transient Data purge macro causes the space management routine to be entered for the purpose of returning all the intrapartition tracks associated with a particular destination.

The intrapartition space routine manages the Transient Data track space by employing a disk map, wherein, each intrapartition track is represented by a bit in the disk map. When the system is initialized the bits are set to zero, indicating available tracks. As tracks are allocated their corresponding bits are set on and the bits are turned off as the tracks are returned to the track pool.

Write Intrapartition Data (PUT)

A check is made to determine if this is the first record to be written to this destination. If it is, the intrapartition space management submodule is entered and a track is obtained for this destination if available. Otherwise a no-space response is returned to the user. After the track is obtained, the first record written is the chain record containing the chain ID, destination ID, and backward chain pointer.

A check is made to determine if there is adequate space available cn this destination's track to contain this record. The check is made between the length of the record and control information maintained in the DCT entry.

If there is insufficient space on the track, another track is obtained via the intrapartition space management submodule from the Transient Data track pool. If a track is not available, a no space response code is returned to the user. The chain record is updated with the forward chain pointer and the count for data records written on the track. The newly acquired track becomes the current output track, and is initialized with a chain record. The new tracks relative track and record number one (TIR=TT1) are used as the direct access address to write the record. The R portion of the TIR in the DCT entry for output is incremented by one and is used to write the record referenced in the PUT request.

The DECB is initialized to contain the proper output area address. A direct access write is executed, using the length contained in the first two bytes of the output record. A CICS WAIT is issued to await completion of the write operation.

Upon successful completion of the disk write, the referenced destination's DCT entry is updated to contain the next TTR to be used for the next PUT request to this destination. The DCT entries for space available on the track and the number of active records in the queue are updated.

If there is an error while writing any disk record the following action is taken:

1. If the error occurred while writing a new chain record, the track is left flagged as used and another track is allocated.
2. If the error occurred while writing a data record, the track is treated as being full, another track is allocated, and a new chain record and the data is written on it. The chain record on the previous track is updated to point to the new track.
3. If the error occurred while updating a chain record, a new track is allocated and the data from the error track is moved to the new track.

A test of the DCT entry is made to determine if the automatic transaction-initiation option (defined by user in DCT entry) is operative. If the option is not in effect for this destination, control is returned to the requesting program. If the option is in effect, the following conditions are tested to determine if the transaction (user-specified in DCT entry definition) can or should be automatically initiated.

1. Is an automatic transaction-initiated task from this destination already in operation? If so, control is returned to the requesting program.
2. Has the user-specified level of data in the queue for this destination been reached for automatic transaction initiation? If not, control is returned to the requesting program.

When the two conditions listed above are satisfied for automatic transaction initiation, a test is made of the DCT entry and the Terminal Control Table terminal entry to determine if a terminal is to be associated with the task. If a terminal is to be associated with the task to be initiated, the TCT terminal entry is flagged so that the CICS Terminal Control program can cause the automatic initiation. The DCT entry is flagged to indicate that a task is initiated, and return is made to the requesting program. If no terminal is to be associated with the automatic initiated task, a CICS ATTACH macro instruction is issued to initiate the task, the DCT entry is flagged to indicate that a task is initiated, and return is made to the requesting program.

EXTRAPARTITION

The extrapartition submodule reads or writes to the referenced destination's queue. An extrapartition destination may be either an input or an output queue, but not both. Retrieval and disposition of data involving an extrapartition destination is performed sequentially. Extrapartition queues may be maintained on direct access storage devices, magnetic tape, or unit record devices.

Extrapartition data queues are provided for data entering the CICS partition/region as input from outside of the partition/region, and for data leaving the CICS partition/region. For example, extrapartition destinations can be used for the logging of all transactions received by CICS for offline analysis.

Read Extrapartition Data (GET)

The DCB address is obtained from the DCT and an OS GET is issued against the DCB. If end of data is reached, an indicator is set in the TCA. The user must recognize this as end of data.

Write Extrapartition Data (PUT)

The DCB address is obtained from the DCT and an OS PUT is issued against the DCB.

Control Processing of Extrapartition Data (FEOV)

The DCB address is obtained from the DCT and an OS FEOV (Forced End of Volume) is issued against the DCB. Control is given back to the calling program with a normal return code.

Locate the Specified Destination (LOCATE)

A search is made of the DCT for the specified destination. The address of the DCT entry is returned to the caller. If no entry is found, an error indicator is set in the caller's TCA and control is returned to the calling program.

TEMPORARY STORAGE CONTROL PROGRAM (DFHTSP) - CHART 12

The Temporary Storage Control program provides the facility for storing data into a temporary location in main storage or direct access storage for the purpose of transferring information between nonconcurrent tasks.

ENTRY ANALYSIS

The Entry Analysis routine analyzes the type of request in the TCA for a PUT to main storage, PUT to auxiliary storage, GET, or RELEASE, and passes control to the appropriate temporary storage module.

When a PUT, GET, or RELEASE is not indicated, the request is invalid. The request error response code is placed in the TCA, and control is returned to the application program.

GET DATA OR RELEASE TABLE SEARCH

The Get Data or Release Table Search routine is given control from the Entry Analysis routine when a GET or a RELEASE has been requested.

The Main Storage Table is searched for the requested symbolic data identification. If the requested data identification is found in the Main Storage Table, control is passed to the Main Storage Get routine. If the requested data identification is not found in the Main Storage Table, the Auxiliary Storage Table is searched for the requested symbolic data identification. If the requested data identification is found in the Auxiliary Storage Table, control is passed to the Auxiliary Storage Get routine.

If the requested data identification is not found in either Temporary Storage Table, a data identification error response code is placed in the TCA, and control is returned to the application program.

MAIN STORAGE PUT (PUT)

The Main Storage Put routine receives control from the Entry Analysis routine when a Temporary Storage PUT to main storage is requested. When the data length of a PUT to Main Storage request is greater than 256 bytes, control is passed to the Auxiliary Storage Put routine by way of the Entry Analysis routine.

If the data length is less than or equal to 256 bytes, a Storage Control GETMAIN macro instruction is issued to acquire the requested main storage. The data is then placed in this storage along with the symbolic data identification. This storage is then added to the main storage chain of Temporary Storage and control is returned to the application program.

MAIN STORAGE GET (GET)

The Main Storage Get routine moves previously stored data from main storage to a user work area.

This routine receives control from the Get Data or Release Table Search routine when the requested data identification is found in the Temporary Storage Main Storage Table. If only a RELEASE has been requested, control is passed directly to the Main Storage Release routine.

If a GET has been requested, the user has the option of supplying a work area or allowing the temporary storage facility to obtain the area. If a work area is not supplied by the user, a work area is obtained by issuing a Storage Control GETMAIN macro instruction. The data is then moved from the Temporary Storage main storage area to the user work area.

When a RELEASE request is issued in conjunction with a GET request, control is passed to the Main Storage Release routine after the GET request has been serviced. If a RELEASE is not requested, control is returned to the application program.

MAIN STORAGE RELEASE (RELEASE)

The Main Storage Release routine removes main storage areas from the main storage chain of Temporary Storage and releases the storage.

This routine receives control from the Main Storage Get routine when a RELEASE is requested.

The requested area of main storage is deleted from the main storage chain of Temporary Storage and the main storage is released by issuing a Storage Control FREEMAIN macro instruction.

AUXILIARY STORAGE PUT (PUT)

The Auxiliary Storage Put routine receives control from the Entry Analysis routine when a Temporary Storage PUT to auxiliary storage is requested, or when data to be PUT to main storage has a data length greater than 256 bytes. When the data length of a PUT request is greater than the length of a direct access storage track, the PUT request cannot be serviced. An "invalid request" code is placed in the TCA, and control is returned to the application program.

If the data length is acceptable for direct access storage, the Auxiliary Storage Table is searched for a pointer to an unused track. If a track is not currently available on the temporary storage data set, the task is placed in a WAIT state until space becomes available by issuing a Task Control SUSPEND macro instruction. When a track becomes available, a Task Control RESUME macro instruction is issued in the Auxiliary Storage Release routine and the task proceeds to store the data.

The data identification name and the data length are placed in the Auxiliary Storage Table entry containing the address of the track. Control is passed to the Event Preparation and Synchronization routine to write the data on direct access storage. From this routine, control is returned to the application program.

AUXILIARY STORAGE GET (GET)

The Auxiliary Storage Get routine reads previously stored data from disk storage to a user work area.

This routine receives control from the Get Data or Release Table Search routine when the requested data identification is found in the Auxiliary Storage Table. If only a RELEASE has been requested, control is passed directly to the Auxiliary Storage Release routine. If a GET has been requested, the user has the option of supplying a work area or allowing the temporary storage facility to obtain the area. If a work area is not supplied by the user, a work area is obtained by issuing a Storage Control GETMAIN macro instruction.

Control is then passed to the Event Preparation and Synchronization routine to read the requested data into the work area. When a RELEASE request is issued in conjunction with a GET request, control is passed to the Auxiliary Storage Release routine after the GET request has been serviced. If a release is not requested, control is returned to the application program.

AUXILIARY STORAGE RELEASE (RELEASE)

The Auxiliary Storage Release routine marks unused disk space areas as available for use in the auxiliary storage table.

This routine receives control from the Auxiliary Storage GET routine when a RELEASE is requested.

If PUT requests have been suspended, the track being released is reserved for the first task in the suspended task chain that was suspended by the Temporary Storage program. The suspended task is made dispatchable by issuing a Task Control RESUME macro instruction. Upon return, control is given back to the application program.

When PUT requests have not been suspended, the track is marked available by initializing the data identification name to zeros in the Auxiliary Storage Table, and control is returned to the application program.

SIGN-ON/SIGN-OFF PROGRAM (DFHSNP/DFHSFP) - CHART 13

The Sign-on program logically attaches a terminal to the system and initializes it to some predefined status. The Sign-off program logically detaches the terminal from the system and may, at the operator's option, take the terminal off the poll list.

The Sign-on program is invoked as an application whenever a terminal operator issues a sign-on or sign-off request. If a sign-on request was issued, and the request is not all numeric, the syntax of the input message is checked to ensure the presence of the password and name keyword parameters. If missing or improperly placed, an appropriate message is returned to the terminal. If syntax is satisfactory, the Sign-on Table is used to verify that a legitimate password and operator name were used. If name and password verfication is positive, the terminal is logically connected by showing its status as "signed on."

If a sign-off request was issued, the terminal is logically disconnected and a check made to see if the operator specified removal from the polling list. If he did, the no-poll switch is set and a disposition message is returned to the terminal.

ENTRY ANALYSIS

The Entry Analysis routine determines whether a sign-on or a sign-off function has been requested by examining the first four characters of the terminal input data area.

If a sign-off function is requested, the Sign-off program is linked to by the Sign-on program.

If a sign-on function is requested, Entry Analysis first checks to ensure that the previous operator on the terminal issued a sign-off request. If not, control is given to the Sign-off program through a program link. Entry Analysis then exits to the Verify Syntax routine.

VERIFY SYNTAX

If the sign-on request is not numeric, the Verify Syntax routine verifies that the terminal operator has correctly supplied the password and operator name keyword parameters. If not present or invalid, sign on is terminated and a message returned to the terminal.

VALIDATE PASSWORD

The Validate Password routine ensures that the correct terminal operator has signed on using the correct password.

Once the syntax of the input line has been validated, the Sign-on Table is loaded via a call to Program Control. The keyed-in operator name is compared against authorized names in the table. If the name is not found, an error message is returned to the terminal. If found, the keyed-in password is compared with the authorized password for the operator name entry. If not equal, an error message is returned to the terminal. If password is satisfactory, the user is signed on.

LOGICAL CONNECT

The Logical Connect routine logically connects a terminal to the system when a sign-on command is issued from the terminal.

Once the keyed information has been validated, the operator identification, security key, and password are moved to the TCTTE from the Sign-on Table and the status of the terminal is set to signed on. This action constitutes the logical connection of the terminal to the system.

SIGN OFF

The Sign-off module performs the operations necessary to logically disconnect a terminal from the system whenever an operator keys in a sign-off command, or keys in a sign-on command and the previous operator has not signed off.

The terminal status is set to a "signed-off" condition, and dynamic storage is acquired for an output area for Transient Data Control. Accumulators within the TCTTE are reset to zero after placing their values in the output area. This journal record is written via Transient Data Control to a master log device. Control is then returned to the Sign-on program via a Program Control return.

DISPOSITION MESSAGE OUTPUT

The Disposition Message Output routine returns a message to the terminal which requested the sign on or sign off, indicating the status of the request (successful, or if not, why it was not successful).

Upon entry, a CICS GETMAIN is issued to get a Terminal I/O area. The message which was passed to this routine is moved to the Terminal I/O Area, and a CICS Terminal Control WRITE is issued. Exit is then made to Sign-on/Sign-off general exit.

MASTER TERMINAL PROGRAM (DFHMTP)

The Master Terminal program is an optional feature of CICS selected at System Initialization. This program consists of six modules: DFHMTPA, DFHMTPB, DFHMTPC, DFHMTPD, DFHMTPE, and DFHMTPF.

The Master Terminal program is a system service program that provides the user with the means of dynamically changing certain system parameters, the status of lines, control units, or terminals.

This program is invoked by keying the proper transaction identification at a master terminal, a supervisory terminal, or a single terminal. The transaction identification may optionally be followed by a series of abbreviated keywords in any order, describing the service to be performed, a numeric value, and/or a parameter list. Each abbreviated keyword, numeric value, and parameter must be separated by commas.

Immediately preceding the first parameter in a parameter list must be a parameter list keyword. The parameter list must be entered last.

If the keyword CANCEL is entered anywhere in the original or subsequent entries, the Master Terminal program is terminated immediately with no further processing. If, while trying to perform the requested service, the Master Terminal program discovers that insufficient information was entered in the original data entry, a response is solicited from the terminal providing the missing information. The response to a request for more information must be either an unabbreviated keyword, a numeric value, or a parameter list pertinent to the service requested.

The services provided by the Master Terminal program are:

1. Inquire about or change the partition exit time interval value.
2. Inquire about or change the runaway task interval value.
3. Inquire about or change the stall detection interval value.
4. Inquire about or change the storage cushion size.
5. Inquire about or change the maximum number of tasks value.
6. Inquire about or change the maximum number of batch tasks value.
7. Inquire about or change the maximum number of ATP tasks value.
8. Inquire about or change the negative poll delay for a terminal.
9. Inquire about or change the trigger level of a transient data intrapartition data set.
10. Turn the CICS Trace function on or off.
11. Inquire about or change the status of a single terminal.
12. Change the status of a list of terminals.
13. Change the status of a class of terminals.
14. Change the status of all the terminals in the system.
15. Inquire about or change the status of a line.
16. Inquire about or change the status of a control unit.
17. Inquire about or change the status of one or more data base data sets.
18. Open one or more data base data sets.
19. Open one or more transient data extrapartition data sets.
20. Open the dump data set.
21. Close one or more data base data sets.
22. Close one or more transient data extrapartition data sets.
23. Close the dump data set.
24. Switch the dump data set to the alternate dump data set.
25. Inquire about the status of a program.
26. Terminate a task.
27. Terminate CICS.

A master terminal may request any of the above services. A supervisory terminal may request only services 11 through 16 and service 26 for terminals, control units, and lines under the supervision of that operator. A single terminal may request only service 11 to inquire about or change its own status.

If the keyword INQUIRY is not entered, it is assumed that a change is requested. If the requested service is to change a numeric system parameter and computations must be performed on the new numeric value before it can be stored within CICS, the computations are performed on the new numeric value, the new numeric value is stored within CICS, and the new computed value appears in the "changed to" portion of the final message.

The response to a request for additional information from Modules B, C, D, E, and F of the Master Terminal program is validated against only the unabbreviated keywords that are meaningful to the routines contained in those modules. A numeric value response is accepted and replaces any numeric values previously entered. If a parameter list is entered, it is treated in the same way as described for the Scan Input routine of Module A.

MASTER TERMINAL PROGRAM MODULE A (DFHMTPA) - CHART 14

Module A of the Master Terminal program is invoked by keying the proper Transaction Identification at a master terminal, a supervisory terminal, or a single terminal.

Entry Analysis

The Entry Analysis routine turns on the "return to Module A" indicator and transfers control to Module F of the Master Terminal Program which locates the line entry in the Terminal Control Table to which the Master Terminal Program is connected and transfers control back to Module A. Upon returning from Module F, the "return to Module A" indicator is turned off, and the Entry Analysis routine determines from which type of terminal the original data entry was made. If the data entry was made from a single terminal, the keywords TERMINAL and SINGLE and a parameter list containing that terminal's symbolic Terminal Identification are assumed to have been entered. If one or more abbreviated keywords, a numeric value, and/or a parameter list have been entered, control is given to the Scan Input routine. At label MTWMSGA, the terminal operator is requested to enter the service he wants performed.

Module Selection And Initiation

If a service has not been requested, control is passed to label MTWMSGA. If the routine that is to perform the requested service is in Module A, control is given to the Test Module A Service Indicators routine which determines the service that has been requested and gives control to the proper routine. If the routine that is to perform the requested service is not in Module A, control is given via an XCTL to the Master Terminal program module that contains the routine.

Scan Input

The Scan Input routine scans the data entry from right to left (backwards), ignoring invalid keywords and accepting only one numeric value and/or parameter list. Incorrect information need simply be followed by a comma and the correct information. This routine builds a formatted parameter list from the information in the inputted parameter list. Formatted parameter lists identified by different parameter list keywords are chained together. If a parameter list is entered with a parameter list keyword that has been previously entered, the old parameter list is deleted from the parameter list chain and the new one is added to the end of the chain. When all the information in the data entry is analyzed, control is passed to the Module Selection and Initiation routine.

A reponse to a request for more information from Module A of the Master Terminal program is validated against the unabbreviated keywords. A numeric value is accepted and replaces any numeric values that were previously entered. If a parameter list is entered, control is given to the Scan Input routine.

Module A of the Master Terminal program provides system services as performed by the Time Interval and Runaway Task Interval routines.

Time Interval Routine

If the keyword INQUIRY has been entered, the current value of the partition exit time interval is displayed and the transaction is terminated. If a numeric value has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Time Interval routine. If the numeric value is less than 100 or greater than 27962020, greater than the runaway task interval, or greater than the stall detection time interval, an error message is displayed, a corrected numeric value is requested from the terminal operator, and control is returned to the beginning of the Time Interval routine. If the numeric value is within the above limits, the new partition exit time interval is placed in the CSA, the old and new values are displayed, and the transaction is terminated.

Runaway Task Interval Routine

If Runaway Task control is not supported, a message to that effect is displayed and the transaction is terminated. If the keyword INQUIRY has been entered, the current value of the runaway task interval is displayed and the transaction is terminated. If a numeric value has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Runaway Task Interval routine. If the numeric value is less than the partition exit time interval or greater than 27962020, an error message is displayed, a corrected numeric value is requested from the terminal operator, and control is returned to the beginning of the Runaway Task Interval routine. If the numeric value is within the above limits, the new runaway task interval is placed in the CSA, the old and new runaway task interval values are displayed, and the transaction is terminated. To make runaway task control inoperative, the value may be set to zero.

MASTER TERMINAL PROGRAM MODULE B (DFHMTPB) - CHART 15

Module B of the Master Terminal program is given control from Module A or D via an XCTL when any of the following routines are required:

- Storage Cushion Routine
- Maximum Number of Tasks Routine
- BATCH or ATP Maximum Number of Tasks Routine
- Negative Poll Delay Routine
- Trace Routine
- File Routine

Unless the request is entered from a master terminal, an error message is displayed and the transaction is terminated.

Entry Analysis determines the type of service requested and gives control to the proper routine.

Storage Cushion Routine

If the keyword INQUIRY has been entered, the current storage cushion size is displayed and the transaction is terminated. If a numeric value has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Storage Cushion routine. If the numeric value is less than 20 or greater than 65535, an error message is displayed, a corrected numeric value is requested from the terminal operator, and control is returned to the beginning of the Storage Cushion Routine. If the numeric value is within the above limits, the new storage cushion size is placed in the CSA, the old and new storage cushion size values are displayed, and the task is terminated.

Maximum Tasks Routine

If the keyword INQUIRY has been entered, the current maximum number of tasks is displayed and the transaction is terminated. If a numeric value has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Maximum Tasks routine. If the numeric value is less than 2 or greater than 999, an error message is displayed, a corrected numeric value is requested from the terminal operator, and control is returned to the beginning of the Maximum Tasks routine. If the numeric value is within the above limits, the new maximum number of tasks is placed in the CSA, the old and new maximum number of tasks are displayed, and the transaction is terminated.

Batch or ATP Max Task Routine

If the Asynchronous Transaction Control Program has not been included in the system, a message to that effect is displayed and the transaction is terminated. If the keywords INQUIRY and BATCH MAXIMUM TASKS have been entered, the current maximum number of batch tasks is displayed and the transaction is terminated. If the keywords INQUIRY and ATP MAXIMUM TASKS have been entered, the current maximum number of ATP tasks is displayed and the transaction is terminated. If the keyword INQUIRY has not been entered and a numeric value has also not been entered, a numeric value is requested from the terminal operator and control is returned to the beginning of the Batch or ATP Max Task Routine. If the keyword BATCH MAXIMUM TASKS has been entered, and the numeric value is not less than the maximum number of tasks value or less than the maximum number of ATP tasks value, an error message is displayed, a corrected numeric value is requested from the terminal operator, and control is returned to the beginning of the Batch or ATP Max Task routine. If the keyword ATP MAXIMUM TASKS has been entered, and the numeric value is greater than the maximum number of batch tasks value, an error message is displayed, a corrected numeric value is requested from the terminal operator, and control is returned to the beginning of the Batch or ATP Max Task routine. If the numeric value is within the above limits, the new maximum number of batch or ATP tasks is placed in the ATP control information area, the old and new maximum number of batch or ATP tasks are displayed, and the transaction is terminated.

Negative Poll Delay Routine

If a parameter list containing symbolic terminal identifications has not been entered, one is requested from the terminal operator and control is returned to the beginning of the Negative Poll Delay routine. The Terminal Control Table is then searched for a terminal entry containing the same symbolic Terminal Identification as was entered in the parameter list. If such an entry is not found, an error message is displayed, a corrected parameter list is requested from the terminal operator, and control is returned to the beginning of the Negative Poll Delay routine.

If the keyword INQUIRY has been entered, the negative poll delay for the line of the requested terminal is displayed and the transaction is terminated. If a numeric value has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Negative Poll Delay routine. If the numeric value is greater than 20000, an error message is displayed, a corrected numeric value is requested from the terminal operator, and control is returned to the beginning of the Negative Poll Delay routine. If the numeric value is within the above limit, the new negative poll delay value for the requested terminal's line is placed in that terminal's terminal entry in the Terminal Control Table, the old and new negative poll delay values for that terminal are displayed, and the transaction is terminated.

Trace Routine

If the trace facility is not currently operative in the system, a message to that effect is displayed and the transaction is terminated. If the new status of the trace facility (ON or OFF) has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Trace routine. If the keyword ON has been entered, the trace facility is turned on, a message to that effect is displayed, and the transaction is terminated. If the keyword OFF has been entered, the trace facility is turned off, a message to that effect is displayed, and the transaction is terminated.

File Routine

If File Control is not currently in the system, a message to that effect is displayed and the transaction is terminated.

If the keyword ALL has been entered, a parameter list is built comprised of all symbolic data base data set names currently in the system. If a parameter list containing symbolic data set names is not present, one is requested from the terminal operator and control is returned to the beginning of the File Routine. Each symbolic data set name is then used as an argument against the File Control Table via a DFHFC TYPE=LOCATE macro instruction.

If the keyword INQUIRY has not been entered, control is given to the File Status Change routine; otherwise, a display is generated containing all of the symbolic data set identifications in the parameter list, and the transaction is terminated. If a data set has not been found in the File Control Table, the words "DOES NOT EXIST" appear beside its symbolic identification in the display. If a data set has been found in the File Control Table, the status of that data set appears beside its symbolic identification in the display.

In the File Status Change subroutine, if the keywords OPEN or CLOSE have been entered, control is given to DFHMTPD via an XCTL; if the action to be taken (keywords ON or OFF) has not been entered, it is requested from the terminal operator. If the function(s) to be changed (READ, UPDATE, ADD, or EXCLUSIVE CONTROL) has not been entered, it is requested from the terminal operator. The File Control Table entry for each data set in the parameter list which has been found in the File Control Table is then modified according to the terminal operator request, the "keyword INQUIRY has been entered" indicator is turned on, and control is returned to the beginning of the File Routine.

MASTER TERMINAL PROGRAM MODULE C (DFHMTPC) - CHART 16

Module C of the Master Terminal program is given control from Module A via an XCTL for any type of terminal status request.

Entry Analysis determines if the transaction has been initiated by a supervisory terminal. If so, and the supervisor's identification has not been entered, it is requested from the terminal operator and that supervisor's Terminal List Table is loaded into main storage via a DFHPC TYPE=LOAD macro instruction. If the type of terminal status request (SINGLE, LIST, CLASS, or ALL) has not been entered, it is requested from the terminal operator. The type of request is then determined and control is given to the proper routine.

Single Routine

If a parameter list containing symbolic Terminal Identifications has not been entered, one is requested from the terminal operator and control is returned to the beginning of the Single routine. If entered from a supervisory terminal and the symbolic Terminal Identification in the parameter list is not in that supervisor's Terminal List Table, an error message is displayed, a corrected parameter list is requested, and control is returned to the beginning of the Single routine.

If the keyword INQUIRY has been entered, control is passed to the label MTSNGBAL. If the new status of the terminal (IN SERVICE, OUT OF SERVICE, RECEIVE, TRANSCEIVE, or TRANSACTION from a master or a supervisory terminal; RECEIVE, TRANSCEIVE, or TRANSACTION from a single terminal) has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Single Routine.

If the new status is not OUT OF SERVICE, control is passed to the label MTSNGBAL. If the action to be taken in the event a task is attached to the terminal (DISPLAY, INTERCEPT, TERMINATE, or SUSPEND) has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Single routine.

At label MTSNGBAL, control is given to the Minor Terminal routine. If the return from the Minor Terminal routine indicates that the symbolic Terminal Identification in the parameter list cannot be found in the Terminal Control Table, an error message is displayed, a corrected parameter list is requested, and control is returned to the beginning of the Single routine.

If the new status is not OUT OF SERVICE, the transaction is terminated. If the request was to put the requesting terminal OUT OF SERVICE, an error message is displayed and the transaction is terminated.

If there was no task attached to the terminal, a message to that effect is displayed and the transaction is terminated. If the action was not to display the task attached to the terminal, a message indicating the action taken is displayed and the transaction is terminated. If the action was to display the task, the Transaction Identification of the task attached to the terminal is displayed, the "keyword DISPLAY was entered" indicator is turned off, the action INTERCEPT, TERMINATE, or SUSPEND to be taken with the task is requested from the terminal operator, and control is returned to the beginning of the Single routine.

List Routine

If the keyword INQUIRY was entered, an error message is displayed and the transaction is terminated. If a parameter list containing symbolic Terminal Identifications has not been entered, one is requested from the terminal operator and control is returned to the beginning of the List routine. The symbolic Terminal Identifications in the parameter list are then validated against entries in the Terminal Control Table.

Each symbolic Terminal Identification which cannot be found in the Terminal Control Table is displayed in an error message and removed from the parameter list. If entered from a supervisory terminal, the symbolic Terminal Identifications that remain in the parameter list are validated against that supervisor's Terminal List Table. Each symbolic Terminal Identification which cannot be found in the Terminal List Table is displayed in an error message and removed from the parameter list.

If the new status for the list of terminals (IN SERVICE, OUT OF SERVICE, RECEIVE, TRANSCEIVE, or TRANSACTION) has not been entered, it is requested from the terminal operator and control is returned to the beginning of the List routine. If the new status is not OUT OF SERVICE, control is passed to label MTTENLPL3. If the action to be taken in the event there is a task attached to any of the terminals (TERMINATE or SUSPEND) has not been entered, it is requested from the terminal operator and control is returned to the beginning of the List routine.

At label MTTENLPL3, control is passed to the Minor Terminal routine for each of the symbolic Terminal Identifications which remain in the parameter list.

Control is then given to the Write Task Statistics routine. If the requesting terminal's symbolic Terminal Identification is in the parameter list and the new status is OUT OF SERVICE, the status of that terminal remains unchanged.

Class Routine

If the keyword INQUIRY has been entered, an error message is displayed and the transaction is terminated. If a parameter list containing the terminal class identification has not been entered, one is requested from the terminal operator and control is returned to the beginning of the Class routine.

If the new status for the class of terminals (IN SERVICE or OUT OF SERVICE) has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Class routine. If the new status is not OUT OF SERVICE, control is passed to label MTSTUPCL. If the action to be taken in the event a task is attached to any of the terminals (TERMINATE or SUSPEND) has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Class routine.

At label MTSTUPCL, the Terminal List Table for the requested class of terminals is loaded into main storage via a DFHPC TYPE=LOAD macro instruction. Control is then given to the Minor Terminal routine for each symbolic Terminal Identification in the Terminal List Table for the requested class of terminals. If entered from a supervisory terminal, control is given to the Minor Terminal routine only for the symbolic Terminal Identifications that are in both the supervisor's Terminal List Table and the Terminal List Table for the requested class of terminals. Control is then given to the Write Task Statistics routine. If the requesting terminal's symbolic Terminal Identification is in the Terminal List Table for the requested class of terminals and the new status is OUT OF SERVICE, the status of that terminal remains unchanged.

All Routine

If the keyword INQUIRY has been entered, an error message is displayed and the transaction is terminated. If the new status for the terminals (IN SERVICE or OUT OF SERVICE) has not been entered, it is requested from the terminal operator and control is returned to the beginning of the All routine. If the new status is not OUT OF SERVICE, control is passed to the label MTSTUPAL. If the action to be taken in the event a task is attached to any of the terminals (TERMINATE OR SUSPEND) has not been entered, it is requested from the terminal operator and control is returned to the beginning of the All routine.

At label MTSTUPAL, control is given to the Minor Terminal routine for each terminal entry in the Terminal Control Table. If entered from a supervisory terminal, control is given to the Minor Terminal routine only for the terminal entries that contain symbolic Terminal Identifications which can be found in that supervisor's Terminal List Table. Control is then given to the Write Task Statistics routine. If the new status is OUT OF SERVICE, the status of the requesting terminal remains unchanged.

The following routines support the system service routines of Module C of the Master Terminal program.

Minor Terminal Routine

If the symbolic Terminal Identification cannot be found in the Terminal Control Table, control is returned to the calling routine at the next sequential instruction indicating that the identification cannot be found.

If the keyword INQUIRY has been entered, the status of the requested terminal is displayed and control is passed to the label MTRTNTRM.

If the keyword RECEIVE has been entered, the requested terminal is put in a RECEIVE status (able to receive messages only; no input) and control is passed to label MTTRMSTC.

If the keyword TRANSCEIVE has been entered, the requested terminal is put in TRANSCEIVE status (able to initiate transactions and receive messages automatically or on request) and control is passed to the label MTTRMSTC.

If the keyword TRANSACTION has been entered, the requested terminal is put in TRANSACTION status (able to initiate transactions and receive messages on request) and control is passed to the label MTTRMSTC.

If the keyword IN SERVICE has been entered, the requested terminal is put IN SERVICE and control is passed to the label MTTRMSTC.

If the requested terminal is also the requesting terminal, control is passed to label MTRTNTRM. If the terminals are not the same, the requested terminal is put OUT OF SERVICE, automatically suspending any task which is attached to that terminal.

If the keywords TERMINATE or INTERCEPT have not been entered, control is passed to label MTTTEXST.

If the requested terminal is part of a pool of terminals and is not connected to a line, the terminal storage for that terminal is released and control is passed to label MTTRMSTC. If a line event has been initiated to the requested terminal, the task cannot be terminated or intercepted, therefore control is passed to the label MTTRMSTC.

1. If a task has not been attached to the requested terminal and either the keyword TERMINATE or INTERCEPT has been entered, all terminal storage for the requested terminal is freed and control is passed to label MTTRMSTC.

2. If a task has been attached to the requested terminal and the keyword TERMINATE has not been entered, control is passed to the label MTTRMSTC.

3. If a task has been attached to the requested terminal and the keyword TERMINATE has been entered, the task is terminated by freeing all of the terminal storage for the requested terminal, and setting an indicator in that task's Task Control Area marking that task for abnormal termination by the Task Control program. Control is then passed to the label MTTRMSTC.

4. If a task has been attached to the requested terminal and the keyword INTERCEPT has been entered, at the requesting terminal, the task is intercepted by removing the task from association with the requested terminal and associating it with the requesting terminal. The Master Terminal program then associates itself with the requested terminal and does a normal termination.

At label MTTRMSTC, if the keyword SINGLE has been entered, the new status of the terminal is displayed.

At label MTRTNTRM, control is returned to the calling routine at the next sequential instruction plus four bytes, indicating that the symbolic Terminal Identification was found in the Terminal Control Table.

Write Task Statistics

A message is displayed indicating the requested terminal's status has been changed. If the keyword OUT OF SERVICE has not been entered, the transaction is terminated. If the keyword OUT OF SERVICE has been entered, the number of tasks that were attached to the requested terminals, the number of these tasks that were terminated, if any, and the number of these tasks that were suspended, if any, are displayed and the transaction is terminated.

MASTER TERMINAL PROGRAM MODULE D (DFHMTPD) - CHART 17

Module D of the Master Terminal program is given control from Module A or B via an XCTL when any of the following services are requested:

1. Open one or more data base data sets.
2. Open one or more transient data extrapartition data sets.
3. Open a dump data set.
4. Close one or more data base data sets.
5. Close one or more transient data extrapartition data sets.
6. Close the dump data set.
7. Switch the dump data set to the alternate dump data set.

If not entered from a master terminal, an error message is displayed and the transaction is terminated. Entry Analysis sets the open, close, or switch request code for the Open/Close program (DFHOCP) in the Common Communication area of the Task Control Area. If the requested service is an open or a close, control is given to the Common Open/Close routine. If the requested service is to switch the dump data set, control is given to the Dump Data Set Open/Close Switch routine.

Common Open/Close Routine

If the type of data set to be opened or closed (DATA BASE, TRANSIENT DATA, or DUMP) has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Common Open/Close routine. The proper request code for data base, transient data, or dump data sets is placed in the Common Communication area of the Task Control Area for the Open/Close program. If the keyword DATA BASE has been entered, control is given to the Data Base Open/Close routine. If the keyword TRANSIENT DATA has been entered, control is given to the Transient Data Open/Close routine. If the keyword DUMP has been entered, control is given to the Dump Data Set Open/Close Switch routine.

Dump Data Set Open/Close Switch Routine

If the Dump facility is not active, a message to that effect is displayed and the transaction is terminated. If the Dump facility is active, control is given to the Open/Close program via a DFHPC TYPE=LINK macro instruction to perform the requested dump data set service. A message is then displayed indicating that the requested service has been completed, and the transaction is terminated.

Data Base Open/Close Routine

If the File Control facility is not active, a message to that effect is displayed and the transaction is terminated.

If the keyword ALL has been entered, a parameter list is built containing all of the symbolic data base data set names currently in the system. If a parameter list containing symbolic data set names is not present, one is requested from the terminal operator and control is returned to the beginning of the Data Base Open/Close routine. If symbolic Data Base data set names are present, the address of the parameter list is placed in the Common Communication area of the Task Control Area and control is given to the Open/Close program via a DFHPC TYPE=LINK macro instruction to perform the requested data base data set service. Upon return from the Open/Close program, the "keyword INQUIRY has been entered" indicator is turned on and control is given to Module B of the Master Terminal program via an XCTL.

Transient Data Open/Close Routine

If the Transient Data facility is not active, a message to that effect is displayed and the transaction is terminated. If a parameter list containing symbolic transient data extrapartition destinations has not been entered, one is requested from the terminal operator and control is returned to the beginning of the Transient Data Open/Close routine.

If any of the destinations require special DCB parameters (indicated by supplying the suffix 'DY' with the destination identification), the parameters are requested from the terminal operator and a list of override parameters is built.

If symbolic transient data extrapartition destinations have been entered, the address of the parameter list is placed in the Common Communication area of the Task Control Area and control is given to the Open/Close program via a DFHPC TYPE=LINK macro instruction to perform the requested transient data extrapartition data set service. A message is then generated containing all of the symbolic transient data extrapartition destination names.

If the service has been performed for a destination, the words "HAS BEEN OPENED" appear beside that name in the message if the request was to open the data set; otherwise, the words "HAS BEEN CLOSED" appear.

If for any reason the service could not be performed for a destination, the words "CANNOT BE OPENED" appear beside that name in the display if the request was to open the data set; otherwise, the words "CANNOT BE CLOSED" appear. The transaction is then terminated.

MASTER TERMINAL PROGRAM MODULE E (DFHMTPE) - CHART 18

Module E of the Master Terminal program is given control from Module A via an XCTL when any of the following services are requested. If not entered from a master terminal, an error message is displayed and the transaction is terminated. Entry Analysis determines which of these services has been requested and gives control to the proper routine.

Trigger Level Routine

If the Transient Data facility is not active, a message to that effect is displayed and the transaction is terminated. If a parameter list containing symbolic Transient Data Destination Identifications has not been entered, one is requested from the terminal operator and control is returned to the beginning of the Trigger Level routine. The symbolic Destination Identification in the parameter list is used as an argument against the Destination Control Table via a DFHTD TYPE=LOCATE macro instruction.

If the symbolic Destination Identification cannot be found, or the destination is not intrapartition, an error message is displayed, a corrected parameter list is requested from the terminal operator, and control is returned to the beginning of the Trigger Level routine. If the keyword INQUIRY has been entered, the trigger level for the requested destination is displayed and the transaction is terminated.

If a numeric value has not been entered, one is requested from the terminal operator and control is returned to the beginning of the Trigger Level routine. If the numeric value is greater than 255, an error message is displayed, a corrected numeric value is requested from the terminal operator, and control is returned to the beginning of the Trigger Level routine. If the numeric value is within the above limit, the new trigger level value is placed in the Destination Control Table entry for the requested destination, the old and new trigger level values for the requested destination are displayed, and the transaction is terminated.

Program Routine

If a parameter list containing Program Identifications has not been entered, one is requested from the terminal operator and control is returned to the beginning of the Program routine. If the Program Identification in the parameter list cannot be found in the Processing Program Table, an error message is displayed, a corrected parameter list is requested from the terminal operator, and control is returned to the beginning of the Program routine. If the Program Identification is found in the Processing Program Table, a message is generated containing the program name, the programming language in which it was written, the size of the program in bytes, whether or not it is permanently resident in main storage, whether or not it is in main storage, its cumulative use count, and its current use count; the transaction is then terminated.

Stall Detection Interval Routine

If the keyword INQUIRY has been entered, the current value of the stall detection interval is displayed and the transaction is terminated. If a numeric value has not been entered, one is requested from the terminal operator and control is returned to the beginning of the Stall Detection Interval routine. If the numeric value is less than the system time interval, or greater than 32767, an error message is displayed, a corrected numeric value is requested from the terminal operator, and control is returned to the beginning of the Stall Detection Interval routine. If the numeric value is within the above limits, the new stall detection interval value is placed in the CSA, the old and new stall detection interval values are displayed, and the transaction is terminated.

MASTER TERMINAL PROGRAM MODULE F (DFHMTPF) - CHART 19

Module F of the Master Terminal program is given control from Module A via an XCTL upon entry to the Master Terminal Program or for a line cr control unit request or a request to terminate a task. If the "return to Module A" indicator is on, control is passed to Find Master Terminal Line Entry Address routine. If the indicator is not on, entry Analysis determines if the transaction has been entered by a supervisory terminal. If so, and the supervisor's identification has not been entered, it is requested from the terminal operator and that supervisor's Terminal List Table is loaded into main storage via a DFHPC TYPE=LOAD macro instruction. The requested service is then determined and control is given to the proper routine.

Find Master Terminal Line Entry Address Routine

The Terminal Control Table is scanned for the line which supports the terminal associated with the Master Terminal Program. If the line is a pooled line, the pool is searched for the line to which the Master Terminal is connected. If the Master Terminal is a 3270, the pool is scanned for the last line in the pool.

When the Line Entry in the Terminal Control Table to which Master Terminal is connected is located, its address is placed in the TWA and control is returned to Module A via an XCTL.

Line And Control Unit Common Routine

If a parameter list containing symbolic Terminal Identifications has not been entered, one is requested from the terminal operator and control is returned to the beginning of the Line and Control Unit Common routine. If entered from a supervisory terminal, the Terminal List Table is scanned for the symbolic Terminal Identification in the parameter list. If the symbolic Terminal Identification cannot be found, an error message is displayed, a corrected parameter list is requested from the terminal operator, and control is returned to the beginning of the Line and Control Unit Common routine.

If the keyword INQUIRY has been entered, control is given to the Line routine if the keyword LINE has been entered, or to the Control Unit routine if the keyword CONTROL UNIT has been entered. If the new status of the line or control unit (IN SERVICE or OUT OF SERVICE) has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Line and Control Unit Common routine. If the keyword OUT OF SERVICE has not been entered, control is given to the Line routine or the Control Unit routine depending upon what service has been requested.

If a task is attached to any of the terminals connected to the line cr control unit and the appropriate response (TERMINATE or SUSPEND) has not been entered, it is requested from the terminal operator and control is returned to the beginning of the Line and Control Unit Common routine.

If the keyword LINE has been entered, control is given to the Line routine; otherwise, contrcl is given to the Control Unit routine.

Line Routine

If the requested line is a pooled line, and a numeric value has not been entered, one is requested from the terminal operator and control is returned to the beginning of the Line and Control Unit Common routine. If the requested line cannot be found by using the numeric value as a relative line number, an error message is displayed, a corrected numeric value is requested from the terminal operator, and control is returned to the beginning of the Line and Control Unit Common routine.

If the requested line supports 3270 terminals, the line pool is scanned for the last line in the pool.

If the keyword INQUIRY has been entered, the status of the requested line is displayed and the transaction is terminated.

If the keyword IN SERVICE has been entered and the request was entered from a supervisory terminal, the symbolic Terminal Identification in the parameter list is validated against that supervisor's Terminal List Table.

If the symbolic Terminal Identification cannot be found, an error message is displayed, a corrected parameter list is requested from the terminal operator, and control is returned to the beginning of the Line and Control Unit Common routine. If the symbolic Terminal Identification can be found or if not entered from a supervisory terminal, the requested line is put IN SERVICE, a message to that effect is displayed, and the transaction is terminated.

If the keyword OUT OF SERVICE has been entered and the requested line is the same as the requesting line, an error message is displayed and the transaction is terminated. If the request was entered from a supervisory terminal, the symbolic Terminal Identification of each terminal connected to the requested line is validated against that supervisor's Terminal List Table. If any symbolic Terminal Identification cannot be found, an error message is displayed, a corrected parameter list is requested from the terminal operator, and control is returned to the Line and Control Unit Common routine. The requested line is then put OUT OF SERVICE, control is given to the Minor Terminal routine (discussed below) for each terminal connected to the line, an OUT OF SERVICE message is displayed, and control is given to the Write Task Statistics routine.

Control Unit Routine

If the terminal identified by the symbolic Terminal Identification in the parameter list is not connected to a control unit, or if the control unit cannot be found in the polling list, an error message is displayed, a corrected parameter list is requested from

If the keyword OUT OF SERVICE has been entered and the terminal operator, and control is returned to the beginning of the Line and Control Unit Common routine.

If the keyword INQUIRY has been entered, the status of the control unit is displayed and the transaction is terminated. the requested control unit is also the requesting control unit, an error message to that effect is displayed and the transaction is terminated.

If the keyword OUT OF SERVICE has been entered and the request was entered from a supervisory terminal, the symbolic Terminal Identification of each terminal connected to the requested control unit is validated against that supervisor's Terminal List Table. If any symbolic Terminal Identification cannot be found, an error message is displayed, a corrected parameter list is requested from the terminal operator, and control is returned to the beginning of the Line and Control Unit Common routine.

If all symbolic Terminal Identifications are valid, control is given to the Minor Terminal routine for each terminal connected to the requested control unit, the status of the control unit is changed, a message to that effect is displayed, and control is given to the Write Task Statistics routine.

Note: When a control unit is put out of service, it is removed from the polling list and all the terminals attached to that list are put out of service. Conversely, when the control unit is put in service, it is inserted into the polling list and all the terminals attached to that list are put in service.

Terminate Task Routine

If a parameter list containing symbolic Terminal Identifications has not been entered, one is requested from the terminal operator and control is returned to the beginning of the Terminate Task routine. The symbolic Terminal Identification in the parameter list is then validated against the Terminal Control Table. If the identification cannot be found, an error message is displayed, a corrected parameter list is requested from the terminal operator, and control is returned to the beginning of the Terminate Task routine.

If the request was entered from a supervisory terminal, the symbolic Terminal Identification in the parameter list is then validated against that supervisor's Terminal List Table. If the identification cannot be found, an error message is displayed, a corrected parameter list is requested from the terminal operator, and control is returned to the beginning of the Terminate Task routine. If the requested terminal is also the requesting terminal, an error message to that effect is displayed and the transaction is terminated. Control is then given to the Out of Service routine, a message is displayed indicating what action was taken, and the transaction is terminated.

The following routines support the system service routines of Module F of the Master Terminal program.

Minor Terminal Routine

If the keywords IN SERVICE and LINE have been entered, control is returned to the calling routine at the next sequential instruction. If the keywords IN SERVICE and CONTROL UNIT have been entered, the terminal is put in service, and control is returned to the calling routine at the next sequential instruction. At this point it is established that the keyword OUT OF SERVICE has been entered. If the keyword CONTROL UNIT has been entered, the terminal is put out of service. Control is then given to the Out of Service routine.

Out Of Service Routine

If the keyword TERMINATE has not been entered, control is passed to label MTTTEXST. If the requested terminal is part of a pool of terminals and is not connected to a line, the terminal storage for that terminal is freed and control is returned to the calling routine at the next sequential instruction. If a line event has been initiated to the requested terminal, the task cannot be terminated; therefore, control is returned to the calling routine at the next sequential instruction.

At the label MTTTEXST, if a task has not been attached to the requested terminal, control is returned to the calling routine at the next sequential instructions; if the keyword TERMINATE has been entered, all terminal storage for the requested terminal is freed before returning control. If a task is attached to the requested terminal and the keyword TERMINATE has not been entered, control is returned to the calling routine at the next sequential instruction. If a task is attached and the keyword TERMINATE has been entered, the task is terminated by freeing all the terminal storage for the requested terminal and setting an indicator in that task's Task Control Area marking that task for abnormal termination by the Task Control program. Control is then returned to the calling routine at the next sequential instruction.

Write Task Statistics

If the keyword OUT OF SERVICE has not been entered, the transaction is terminated. If the keyword OUT OF SERVICE has been entered, the number of tasks attached to the requested terminals, the number of these tasks which were terminated, if any, and the number of these tasks which were suspended, if any, are displayed and the transaction is terminated.

SYSTEM STATISTICS PROGRAM (DFHSTKC) - CHART 20

System Statistics is a system service program which provides the terminal operator with the capability of logging, any time during the day, all or selected statistics maintained by the various CICS management programs.

This program consists of three independent but logically connected modules---Supervisory Statistics, File and Terminal Statistics, and Transient Data and Temporary Storage Statistics.

Supervisory Statistics gains control from Program Control whenever a request for system statistics is keyed from a terminal. When finished, it passes control to File and Terminal Statistics via a Program Control XCTL request. This second module passes control to Transient Data and Temporary Storage Statistics via an XCTL. When finished, this module then returns control to CICS Program Control via a RETURN macro instruction, thus terminating the transaction.

Raw statistics are kept in the CSA and are formatted by the respective programs and sent via Transient Data Control to the symbolic destination CSSL.

All statistics can be requested, or any combination of the following:

1. Task
2. Storage
3. Program
4. Dump

5. Terminal
6. File
7. Transient Data
8. Temporary Storage

SUPERVISORY STATISTICS

This module edits and logs statistics maintained by the following CICS management programs:

1. Task Control
2. Storage Control
3. Program Control
4. Dump Control

Upon entry, Supervisory Statistics determines whether all statistics or just selected ones are wanted. A byte in the Transaction Work Area contains a unique bit for each set of statistics required. Entry Anaylsis examines the parameters keyed in by the terminal operator and sets the appropriate bit. Each routine responsible for formatting and outputting a particular set of statistics first checks to see if that set of statistics has been requested. If it has, the values are fetched from the Common System Area, formatted and placed on a symbolic Transient Data destination.

FILE AND TERMINAL STATISTICS

This module edits and logs statistics maintained by the following CICS control modules:

1. Terminal Control
2. File Control

Upon entry, File and Terminal Statistics checks the selection bits in the transaction work area (set by Supervisory Statistics) to determine if Terminal Control Statistics were requested. If they were, an output area is acquired and the statistics are formatted and sent through transient data for output. A check is made to see if File Control statistics were requested. If requested, these statistics are written in the same manner. A Task Control CHAP is issued to ensure a low-dispatching priority, and control is passed to Transient Data and Temporary Storage Statistics via a Program Control XCTL.

TRANSIENT DATA AND TEMPORARY STORAGE STATISTICS

The Transient Data and Temporary Storage Statistics module edits and logs the statistics maintained by the following CICS control modules:

1. Transient Data Control
2. Temporary Storage Control

Upon entry, Transient Data and Temporary Storage Statistics checks the selection bits in the Transaction Work Area, (set by Supervisory Statistics) to determine if Transient Data statistics were requested. If they were, an output area is acquired and the statistics are formatted and sent through Transient Data Control for output. A check is then made to see if Temporary Storage statistics were requested. If so, these statistics are written in the same manner. The transaction is then terminated by issuing a Program Control RETURN macro instruction.

ABNORMAL CONDITION PROGRAM (DFHACP) - CHART 21

The Abnormal Condition program is a system service program that is used to analyze abnormal conditions which occur within the system, and to inform the terminal operator of the problem.

Entry to the Abnormal Condition program is normally from the Program Control program when an abnormal dump is requested by the system. Errors are classed as one of two broad categories: (1) task abnormal conditions, and (2) operator errors.

1. Task Abnormal Conditions

   Whenever a CICS management program detects a problem, it issues an ABEND request with a unique code. This is often caused by the application program destroying system control information. When this happens, the task is terminated, the terminal operator is informed of the error, and the error is logged at destination CSMT.

2. Operator Errors

   Operator errors occur during interaction with the system terminals. Some of the errors which might occur in this category are invalid transaction ID, security key violation, and operator not signed on. In addition to the operator being notified, the errors are also logged at destination CSMT.

TERMINAL TEST PROGRAM (DFHFEP) - CHART 22

The primary purpose of the Terminal Test program is to help the IBM Field Engineer when he is testing the 2260 Display Station (Local Attachment) and the Common Carrier Teletypewriter Exchange Terminal Station (TWX Model 33/35). However, the program is operational on all terminals supported by the system. It will send all the characters that are printable on that terminal upon request. It will also send back to a terminal the message just entered from the terminal.

Upon entry, the user is requested to specify what action is to be taken by entering "end", "print", or any desired message. If "end" is specified, the transaction is terminated. If "print" is specified, all characters printable at that terminal are sent. If neither "print" nor "end" are specified, the keyed input is returned exactly as entered.

DUMP UTILITY PROGRAM (DFHDUP) - CHART 23

The Dump Utility program formats the dump data set for printing and prints out the data in both hexadecimal and alphameric format.

The dump data set resides on disk or tape as created by the Dump Control program during the execution of CICS. It consists of those areas of storage that were requested for dumping. Each significant area is identified by a heading, with its actual starting and ending addresses. It is then printed (in both hexadecimal and decimal format) starting with a relative address of zero.

SYSTEM TERMINATION PROGRAM (DFHSTP) - CHART 24

The purpose of System Termination is to provide for an orderly shutdown of CICS.

System Termination involves the following phases in the shutdown process:

1. Terminal Quiesce
2. Print Statistics
3. Close Data Sets

TERMINAL QUIESCE

Terminal Quiesce ensures an orderly cessation of terminal activity whenever system termination is requested.

Upon entry to Terminal Quiesce, a check is made to see if an immediate termination is requested. If it is requested, Terminal Quiesce is ignored. If the request is not immediate, all other tasks are allowed to terminate before system shutdown is completed. At entry to the Quiesce routine, System Termination is detached from the terminal entering the shutdown request to allow other activity on that terminal.

PRINT STATISTICS

The Statistics Print module logs system statistics which are maintained by various CICS management programs.

Statistics Print issues a Program Control LINK macro instruction to the System Statistics program which places all statistics onto a symbolic Transient Data destination (CSSL). (See the discussion of the System Statistics program.)

CLOSE DATA SETS

This modules closes all CICS system data sets before terminating CICS.

The following data sets are closed, providing they were opened by System Initialization or through the Dynamic Open/Close facility:

1. Program Control
1. Dump Control
2. Terminal Control
3. File Control
4. Transient Data Control
5. Temporary Storage Control

The address of the open/close list is acquired from the Common System Area. It is loaded into the appropriate register and an OS CLOSE macro instruction is issued.

TRACE CONTROL PROGRAM (DFHTRP) - CHART 25

The Trace Control program provides CICS and the user with an easy and convenient method of tracing significant system activity. Through the use of this program, CICS has the capability of creating standard Trace Table entries each time a CICS macro instruction is issued. In addition, the CICS user is provided with a special macro instruction which may be used to create a Trace Table entry.

ENTRY ANALYSIS

Upon entry to the Trace Control program, a check is made to determine if trace is active for the type of request issued. If trace is active, basic initialization of the Trace Table entry is completed. When the end of the table is reached, entries are again made at the beginning of the table, thus creating a wrap-around effect. After basic initialization is complete, control is passed to the appropriate Trace Request module to build the trace entry. Each Trace module can be enabled or disabled via the turn-on/turn-off trace request.

TRACE ENTRY DEPENDENT ROUTINES

The Trace Entry Dependent routines complete the Trace Table entry with the necessary information for the type trace being taken. Possible types of entries include:

- User-Supplied
- Task Control
- Storage Control
- Program Control
- File Control
- Transient Data Control
- Dump Control
- Interval Control
- Temporary Storage Control
- Turn on/Turn off

See the section "Control Blocks - Control Tables and Control Areas" for the format and contents of each type of trace entry.

COMMON EXIT

After the trace entry is complete, Common Exit checks the trace entry just completed against the previous entry to see if it is a duplicate. If the entry is a duplicate, a duplicate entry indicator is turned on in the previous entry and the Trace Table pointers are backed up one entry. This prevents wiping out the Trace Table with duplicate entries in case of a loop. The duplicate entry contains a count of the number of times the preceding entry was duplicated.

<u>HIGH-LEVEL LANGUAGE PREPROCESSOR PROGRAM</u> (DFHPRPR) - CHART 26

The High-Level Language Preprocessor prepares a high-level language program for input to the Assembler. The Assembler then generates the high-level language statements for CICS macros for input to the High-Level Language Compiler.

The output from the High-Level Language Preprocessor to be used as input to the Assembler can be on punched cards, tape, or direct access storage.

OPEN OUTPUT DEVICE

The Open Output Device module opens the DCB.

PROCESS STATEMENTS

The Process Statements module determines for each statement whether it is a CICS macro instruction or high-level language statement.

If the statement is a CICS macro instruction, this module leaves it untouched, with the following exception. If the statement is a Storage Control or File Control macro instruction with an initialization byte specified, it converts the single byte to a zoned decimal halfword.

If the statement is not a CICS macro instruction, this module inserts a REPRO statement before the source statement and writes both statements so that the Assembler will write out the statement to be processed later by the High-Level Language Compiler.

DEVICE DEPENDENT MODULE (DFHDDM) - CHART 27

The Device Dependent module formats a message and obtains the proper amount of terminal storage for a 1030 device. This is accomplished as follows:

- Entry Analysis tests the Terminal Control Table entries for a 1030 device type. If this is not a 1030 device, an immediate return is made to the calling routine.
- Compute Terminal Area calculates the amount of terminal storage needed to contain the message to be written.
- Obtain Terminal Area issues a Storage Control GETMAIN request for terminal storage.
- Move Data Routine moves the message to be sent into the terminal storage area inserting the necessary idle characters.
- Common Exit issues a Program Control RETURN to exit to the requesting routine.

TERMINAL ERROR PROGRAM (DFHTEP)

The Terminal Error Program is a CICS provided module intended for user alteration or replacement when used, this program contains only a DFHPC TYPE=RETURN macro instruction. The user should refer to the Terminal Abnormal Condition Program (DFHTACP) for details concerning the interface between DFHTEP and DFHTACP.

TIME ADJUSTMENT PROGRAM (DFHTAJP) - CHART 28

The Time Adjustment program is a system service program whose purpose is to adjust CICS-maintained expiration times of day to reflect significant changes to the current time of day maintained by the operating system. The Time Adjustment program is executed as a system task (automatically initiated by the Interval Control program), based on changes in the operating system's time of day detected by the Task Control program.

Upon entry, the Time Adjustment program tests a time adjustment indicator at CSAICIND to ensure that the task was automatically initiated by the system. If the indicator is not set, the program refreshes the julian date at CSAJYDP and exits without any adjustment processing.

If the indicator is set, the program logically determines whether or not the task was initiated because midnight occurred, setting an internal indicator accordingly. It then develops an adjustment value in two forms (binary and timer units) which are used to adjust CICS internal expiration times.

The Time Adjustment program first performs the necessary adjustments to any unexpired Interval Control Elements (ICE's) found on the ICE chain. If the program was invoked because midnight occurred, all unexpired ICE's are adjusted. If it was invoked due to some other change in the operating system time of day, only ICE's whose expiration times are dependent on the passage of intervals of time are adjusted.

The ICE's are first removed from the ICE chain, their expiration times are reduced by the adjustment value, and the ICE's are finally remerged into the ICE chain in expiration time of day sequence. When the end of the ICE chain has been reached, the program adjusts the negative poll delay expiration times in the Terminal Control Table.

Terminal Control controls the time intervals between line polls when a negative response to a poll is detected. It does so by adding the user-defined negative poll time delay value in a given Terminal Control Table line entry to the current time of day maintained by CICS. The next poll will be made after this calculated expiration time has been reached. The Time Adjustment program scans the Terminal Control Table, reducing the calculated "next poll" expiration times in applicable entries by the adjustment value.

The Time Adjustment program finally adjusts the Terminal Control program's next dispatching time of day (CSATCNDT), clears the time of day adjustment value total (CSATADJT), clears the current timer units time of day, and resets the time adjustment indicator. The program then refreshes the time of day formats maintained by CICS in the CSA to be the same as the operating system's time of day, and prints an informative message for the console operator prior to completing the task with a normal Program Control RETURN request.

DUMMY CSA PROGRAM (DFHDCSA) - CHART 29

The Dummy CSA program is the module which is given control via an OS XCTL macro instruction at the completion of system initialization. The program issues a Storage Control macro instruction to obtain the storage cushion and gives control to Terminal Control to begin processing. This module should be link edited with the RENT parameter and put into SYS1.LINKLIB. In an MVT system, it will then be loaded by OS into Subpool 252, where it will fit into an already existing fragment.

PL/I STORAGE ALLOCATION PROGRAM (DFHSAP) - CHART 30

The PL/I Storage Allocation Program is composed of a CICS module which routes the storage request to the proper entry point in a modified version of the standard PL/I module IHESA. The IHESA module has been altered to:

1. Omit any issuance of SPIE and STAE
2. Replace OS GETMAIN's with CICS GETMAIN's

PL/I INTERFACE PROGRAM (DFHPL1I) - CHART 31

The PL/I Interface program, which is link edited to the front of every PL/I application program, serves as a bridge to either the PL/I interface in Program Control or the PL/I Storage Allocation Program (DFHSAP). It also serves as the entry point for the PL/I application program and replaces IHENTRY in a normal PL/I program.

An entry point exists in this program for every call that a PL/I program will make to the normal PL/I module IHESA. All calls for IHESA are intercepted here and are routed to DFHSAP which contains the necessary parts of the standard IHESA module.

TERMINAL ABNORMAL CONDITION PROGRAM (DFHTACP) - CHART 32

The Terminal Abnormal Condition program (DFHTACP) is a system service program used to analyze terminal errors and/or line errors and take appropriate action with regard to the terminal and/or line being placed in service or out of service.

DFHTACP is attached by the Terminal Control program (DFHTCP). DFHTCP then obtains storage and places a copy of the line entry in this storage for DFHTACP to analyze. DFHTCP chains up to ten of these storage areas (in a push down list) and then places the line out of service.

For every error encountered, a message is created and written to the master terminal log (destination CSMT), the terminal log (destination CSTL), or to the terminal itself. If the message goes to a terminal that is on a switched line but disconnected, the message is written to the terminal log (destination CSTL).

After the message has been created and written, an analysis of the error is made and, where appropriate, the line status and/or terminal status are modified.

If the terminal error is a BTAM return code, a second message is created and written to the terminal log (destination CSTL) and the line is placed out of service.

In the case of all other errors, control is passed to the Terminal Error program (DFHTEP).

ASYNCHRONOUS TRANSACTION CONTROL PROGRAM (DFHATP) - CHART 33

The Asynchronous Transaction Control Program (part of the Asynchronous Transaction Processing facility) controls the initiation, and data handling of all asynchronous task's which are submitted as part of a batch. ATP is executed as a unique CICS task, with its own TCA, and may only be resident and active whenever one or more batches exist within CICS.

A batch is one or more CICS transactions, along with any associated data, which have entered the system through an Asynchronous Transaction Input Processor (RDR). When the 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 may be automatically transmitted back to a terminal, depending on how the batch was entered. If not automatically transmitted, the output remains queued until it is requested by the originating or alternate terminal.

When a batch is created by DFHRDR, a Batch Control Area (BCA), containing batch status information, is placed on a BCA chain and ATP is either ATTACH'ed or marked "ready to run" by the RDR. Once active, ATP remains active until there are no further BCA's to be processed, or a special CATP STOP transaction is entered from an authorized terminal. When all services have been performed for all existing batches, ATP issues a CICS type WAIT, causing the ATP task to enter the WAIT state. The ECB on which ATP is waiting may be POST'ed by DFHRDR, DFHWTR, the Task Control Program or the Terminal Control

Program, depending upon circumstances. This ECB is in the first byte of field CSAATP which is in the CSA optional features list.

Each time ATP is activated (that is POST'ed), it scans the BCA chain looking for possible services to perform. The services performed depend upon the status of the batch (as determined by the BCA Analysis routine) and whether there is an asynchronous task currently processing the batch. Each transaction initiated by ATP is given the address of the dummy TCTTE in the BCA. The application program sees this dummy TCTTE exactly as a real TCTTE and performs his terminal requests as if he were attached to a real terminal. Any DFHTC TYPE=READ request causes ATP to extract data off the input queue and pass it to the transaction. Likewise, any DFHTC TYPE=WRITE request causes ATP to place the transactions output data onto an output queue for later processing be an Asynchronous Transaction Output Processor (DFHWTR). All rules which apply to the handling of Terminal I/O Areas when a task is connected directly to a terminal, also apply to transactions being run asynchronously.

ENTRY ANALYSIS (Cross reference label: DOCATP01)

The Entry Analysis routine is executed only when DFHATP is ATTACH'ed. It's initial function is to determine if CATP was ATTACH'ed by a CRDR transaction or by Terminal Control Program in response to a terminal command. If the ATTACH was issued by a CRDR transaction, the TCA facility address field (TCAFCAAA) will be zero and control is passed directly to the BCA Scan Initialize routine.

If the TCA facility control address field is non-zero, it is assumed to be the address of the TCTTE for the terminal which issued a CATP command. Entry Analysis determines whether a CATP START or a CATP STOP command was issued. If neither was issued, an "INVALID REQUEST" message is returned to the terminal.

If CATP STOP was issued, the stop flag (BCASTOP) is turned on in the ATP CSA extension area field named CSABCAI. This flag will cause ATP to terminate itself whenever the status of all existing batches is conducive to termination. The message "ATP Termination Scheduled" is returned to the terminal, and the ATP task attached to the terminal terminates. If CATP START was issued, Entry Analysis first issures that no other ATP task is currently active by checking the use count for PPT entry DFHATP. If another ATP task is active, the message "ATP Already Active" is returned to the terminal and the ATP task attached to the terminal terminates. If another ATP task is not active, the attached terminal is released and control is given to the BCA Scan Initialize routine in preparation for servicing any existing batches.

BCA SCAN INITIALIZED (Cross reference label: DOCATP02)

The BCA Scan Initialize routine is entered upon initial execution of ATP and each time ATP is activated (i.e. POST'ed). Its basic function is to initialize pointers to the Lead of the BCA processing chain and pass control to BCA Analysis routine. If there are no BCA's on the chain, the ATP task terminates.

BCA ADVANCE (Cross reference label: DOCATP03)

The BCA Advance routine has two basic functions: 1) advance pointers to the next BCA in the chain to be analyzed and pass control to the BCA Analysis routine and 2) to issue a DFHKC TYPE=WAIT on a pseudo ECB when the end of the BCA chain has been reached. Whenever control is again given to ATP via a POST of the pseudo ECB, the Scan Initialize routine is entered. Depending upon various conditions, ATP may insure re-activation at the next timer interval by placing a X'80' at CSAATP before issuing the WAIT. This flag causes the Terminal Control Program to POST the ECB (CSAATP) whenever it gains control to service terminals; at most, one time interval. Unless the X'80' is first placed in the CSAATP field, ATP will only be made active (POST'ed) under the following conditions:

1. By Task Control Program whenever an asynchronous task issues a DFHTC WAIT macro instruction. When this happens, Task Control determines if the task is under control of a dummy TCTTE and, if so, places a X'40' (POST) into CSAATP.

2. By Task Control Program whenever an asynchronous task either normally or abnormally terminates.

3. By DFHRDR when a new batch is created and placed onto the BCA chain.

4. By DFHWTR when a user requests the release of a batch previously being held.

BCA ANALYSIS (Cross reference label: DOCATP04)

BCA Analysis is entered from either Scan Initialize or BCA Advance and its functions is to determine the status of the BCA being examined and give control to the appropriate routine. If the BCA is in "hold" status, control is given to BCA Advance to get the next BCA. If the status is "input complete" (BCARDYIN) control is given to the Initiator to initiate the processing of tasks. If the status is "in progress" (BCANPROC) control is given to Service Analysis and Control to examine the dummy TCTTE for possible service requests. If the status is "ready for output" (BCARDYOT), control is given to the Output Scheduler so that a DFHWTR application may be scheduled. Any other status causes control to be passed to BCA Advance.

BCA PURGE LINKAGE (Cross reference label: DOCATP05)

The BCA Purge Linkage routine is entered whenever ATP detects a BCA marked "to be deleted" (BCADELTQ). This is accomplished by the user through a CWTR command statement. If an asynchronous task is currently processing the batch, it is abnormally terminated, the BCA is unchained and passed, as a facility, to the Asynchronous Queue Purge Program (CAQP). CAQP purges the transient data queues of all associated input and output data, then frees the BCA storage.

INITIATOR (Cross reference label: DOCATP06)

The Initiator routine is responsible for initiating (ATTACHing) CICS tasks which have been submitted through the Asynchronous Transaction Input processor (CRDR). There are two entry points into the Initiator. ATPINIT is the initial entry point whenever a batch is first marked "ready for processing". Entry at this point assumes that no other task has been previously initated. ATPINITP is used to enter the initiator whenever one asynchronous task terminates and another may be ready to start. One of the first functions performed before initiating a task is to insure that the system is not under stress or that the number of active asynchronous tasks is with limits defined by the user. If the first task is being initiated, the Initiator acquires a dynamic Transient Data Input buffer, which is used to read data off the input queue, and a TIOA which used to pass data records to the transaction. After all I/O areas have been acquired and properly initialized, ATP places the first (or next) logical record into the TIOA. The Initiator assumes that the first four characters of this record contain a valid CICS transaction code and uses it to issue a CICS DFHKC TYPE=ATTACH macro instruction. The Initiator then exits to BCA Advance to set up pointers to next BCA.

ATP SERVICE ANALYSIS (Cross reference label: DOCATP08)

The Service Analysis routine is entered each time a BCA indicates an "in process" status. In this routine, the dummy TCTTE is examined to determine what services, if any, have been requested by the asynchronous task which is processing the batch. Request bits are placed in dummy TCTTE when the asynchronous application program issues a DFHTC macro instruction indicating data is to be read or written. If the request is for a READ, ATP uses the Fetch subroutine to acquire the next logical record, places it in the TIOA, marks the applications TCA as "ready" to run, and exits to BCA advance. If the request is for a WRITE, a Transient Data Output buffer is dynamically acquired if one does not already exist. The data in the TIOA is moved into the output buffer, and, if the buffer is full, it is written to the output queue through use of Transient Data PUT macro instruction.

The size of the Input and Output buffers are defined by the user at system generator. If an application program issues a DFHTC TYPE=WRITE request, and presents an output record larger than the output buffer size, the record will be truncated on the right to fit in the buffer.

OUTPUT SCHEDULER (Cross reference label: DOCATP07)

The Output Scheduler routine is entered when a BCA indicates that all processing for a batch is complete, and the BCA is marked "ready for output". The Output Scheduler first links to the Batch Termination Subroutine to perform clean-up for the last task, if any. It then examines the Write Request Element (WRE) chain in the BCA to determine if any Asynchronous Transaction Output Processors (CWTR) need to be scheduled. If there are any WRE's attached to the BCA which have not had writers scheduled, the Output Scheduler issues a DFHKC TYPE=SCHEDULE macro instruction. This causes a CWTR transaction to be initiated on the specified terminal whenever the terminal is ready to receive output.

FETCH LOGICAL RECORD SUBROUTINE (Cross reference label: DOCATP09)

The basic function of this subroutine is to acquire the next logical input record and, if necessary, read a new block of data from the input queue. Depending upon special indicators set prior to entry, this subroutne will

1. fetch the next logical record and pass it back to caller, disregarding its content.

2. ignore all logical records until the next flush delimiter is detected.

3. fetch next logical record, ignoring the delimiter, if found.

BATCH TERMINATION SUBROUTINE (Cross reference label: DOCATP10)

The Batch Termination subroutine is entered whenever ATP detects that a batch's input queue has been exhausted. Its basic function is to insure that the last output buffer is written to the output queue and to release all storage associated with the batch, such as TIOA's.

ASYNCHRONOUS TRANSACTION INPUT PROCESSOR (DFHRD1,DFHRD2) - CHART 33

The Asynchronous Transaction Input Processor (CRDR) reads groups of data from a terminal and queues them on Transient Data Intrapartition queues for later handling by the Asynchronous Transaction Control Program (DFHATP) and subsequent transmission to a terminal using the Asynchronous Transaction Output Processor (CWTR).

The data read by CRDR is called a batch and consists of transaction initiating records, input data records, and one or more delimiters. Each batch is maintained and controlled through the use of a Batch Control Area (BCA). (Refer to "Control Blocks - Control Tables and Control Areas" for details on the contents of a BCA).

CRDR is a two phase program. Phase 1, DFHRD1, interprets the CRDR record, builds and chains the BCA, and returns any messages to the terminal operator. Phase 2, DFHRD2, reads the batch data, interacts with any exit routines, and queues the data onto a Transient Data Intrapartition queue. Phase 2 XCTL's to Phase 1 to terminate CRDR.

TRANSACTION ANALYSIS (DFHRD1)

Entry Analysis (DOCRD101)

Upon entry to Phase 1, a test is made to determine whether reentry has been made from Phase 2 to return error messages and/or terminate CRDR. If such is the case, control is passed to the Message Processor. If not, initial entry is assumed and the CRDR message is interpreted.

CRDR Message Interpreter (DOCRD102)

Any keywords in the CRDR message are verified for correctness and meaning. When presence of a legitimate keyword is determined, control is passed to the appropriate parameter handling subroutine. After all keywords have been processed, control is given to the BCA Build section.

Parameter Extraction (DOCRD103)

The parameter for each keyword is verified for presence and length and put into its corresponding BCA field. The exit routine suffix, if present, is saved in the TCA Work Area.

BCA Build (DOCRD104)

After the name, password, and delimiter fields have been built from possible keyword parameters, the BCA is completed by BCA Build.

The CRDR TCTTE is copied to the BCA and freed from the TCA and terminal storage. If a batch name was specified it is checked for uniqueness. Otherwise a name is constructed from the TCTTETI anc TCAKCTTA fields. The input and output dynamic DCTs are built and initialized.

Finally, the exit routine, if requested, is loaded and the Transient Data buffer is acquired.

Phase 1 Termination (DOCRD105)

If no errors have been recognized, the BCA is entered into the BCA chain. A XCTL is issued to enter DFHRD2.

MESSAGE PROCESSOR (DFHRD1)

Message Build (DOCRD106)

The TWAERROR byte is scanned to determine which message is to be sent to the terminal operator. The messages may be in one of two groups which are selected by determining if the message request originated in Phase 1 or Phase 2 (Queue Build). If an error occurred and CRDR must be terminated, the operator is requested to enter STOP to overtly authorize termination. Two successive user defined delimiters will also terminate CRDR.

BCA Purge (DOCRD107)

If the error was detected in Queue Build or the operator requested that the batch be deleted in the delimiter statement, the BCA is unchained and the space is returned to CICS.

QUEUE BUILD (DFHRD2)

Terminal Read (DOCRD201)

After initializing the buffer for blocking, a simple GET is issued to obtain the first terminal data message. Subsequent entries to this routine bypass the initialization.

Exit Routine Entry (DOCRD202)

This routine is bypassed if an exit routine was not requested.

If an exit routine is present, TWAREC is zeroed, registers 0-11 are stored and a BAL 14,4(15) transfers control to the exit. Upon return, the registers are restored and a test is made (TWAREC≠0) to determine if a record is to be inserted. If not, the last terminal message is used. If a zero length message has been inserted, control is passed back to Terminal Read. If a record is to be inserted, control passes to Input Blocking. The last terminal message will be lost if a subsequent exit action doesn't save it.

Input Blocking (DOCRD203)

All messages, either read or inserted, are blocked to form a standard blocked variable length record. The size of the physical record will be no greater than the INBUFF= parameter defined during system generation. When a block is full, it is written to a Transient Data Intrapartition queue.

After each record is blocked, it is checked to see if it is a delimiter. If a delimiter was specified in the CRDR message, a check is made to see if this logical record is the second delimiter in a row. If the batch is complete, control is passed to CRDR Shutdown. If the batch is not complete a test is made to see if the exit routine should be re-entered. If such is not the case, control is passed back to Terminal Read.

CRDR Shutdown (DOCRD204)

If there are still data records in the buffer when CRDR Shutdown is entered the buffer is purged. A check is then made to see if the batch is to be deleted or held. If neither of these conditions apply, the batch is staged for processing and ATP is posted to let it begin processing. If ATP is not in the system it is attached.

Control is then returned (via XCTL) to Phase 1 to print the end of job messages.

ASYNCHRONOUS TRANSACTION OUTPUT PROCESSOR (DFHWT1,DFHWT2) - CHART 33

After batches have been built by CRDR they are processed under the supervision of the Asynchronous Transaction Processor (ATP). The operation is analogous to transaction/terminal processing except that two Transient Data queues simulate the terminal. CRDR builds the queue that contains all input from the terminals and ATP, under the direction of the transaction program Terminal Control macros, builds the output queue. When the output queue is complete, the batch is ready for output.

Output is scheduled in response to a CWTR statement and actually occurs if CWTR is attached to a real terminal. This means that output can be scheduled by a CWTR statement in the batch input stream and transmission will take place as soon as the terminal can be acquired and the output batch is complete.

WRE BUILD (DFHWT1)

Keyword Verify (DOCWT101)

Initial entry to CWTR is made to Keyword Verify in Phase 1. A check is made to determine if entry was from CSMT. If so, the initiating message is shifted to overlay the CSMT characters and a flag is set on to bypass password checking.

Next a check is made to verify that all keywords in the message are valid.

Parameter Extract (DOCWT102)

An attempt is made to locate all parameters and their lists. The NAME= and TERMID= parameters are extracted during BCA SCAN.

BCA Scan (DOCWT103)

The BCA chain is searched to locate batches that have been requested by the CWTR statement.

If the NAME= keyword is present, those batches named that originated from the SOURCE= terminal are located. If no names were specified, all batches that originated from the SOURCE= terminal are located. If SOURCE= ALL is present, no check of the originating terminal is made.

Once a BCA is located a check is made to see if the action can be performed now. If SAVE, DELETE, or RELEASE was requested, the action is performed immediately (note: DFHWT2 is LINKed to to perform the DELETE). If STATUS or output was requested a Write Request Element (WRE) must be built.

WRE Build (DOCWT104)

Once a BCA whose output or status is to be sent is located, a WRE is built for each terminal that is to receive the output. Before the new WRE is added to the BCA's WRE chain, a check is made to assure that a duplicate WRE is not already on the chain.

A WRE must be constructed for each TERMID applying to each batch. For example, CWTR NAME=(B1,B2),TERMID=(T1,T2,T3) will cause six WREs to be built.

Phase 1 Termination (DOCWT105)

After all WREs are built or all immediate operations (e.g. SAVE) have completed, a test is made to see if CWTR is operating without a real terminal. If this is the case, CWTR terminates. Otherwise an XCTL is effected to give control to Phase 2 to try to satisfy any outstanding WREs for this terminal.

OUTPUT SECTION (DFHWT2)

Entry Analysis (DOCWT201)

Upon entry, Phase 2 tests to see if entry was to delete a BCA. If so, control is passed to the WRE unchain routine.

WRE Search (DOCTW202)

Each WRE on each BCA chain is examined to see if it applies to the users terminal. If one is found and is ready for output or status was request, control is passed to the output module. Preparatory to output, a TIOA that will handle the largest record is acquired in lieu of the maximum TIOA furnished with the input message. The optional user exit is also loaded.

Get Logical Record (DOCWT203)

The Output DCT located in the BCA is copied to the TCA Work Area. The TCA DCT is used to retrieve all records to be sent to the terminal. As each logical record is extracted from the block, it is passed to the user exit. After the last record has been processed, control is given to the Unchain WRE routine to remove the WRE that initiated this operation.

Exit Routine (DOCWT204)

Prior to entering the optional user exit routine, TWAREC is zeroed and TWANXREC is stuffed with the address of the logical record just extracted from the buffer. Registers 0-11 are then saved and the user's routine is entered with a BAL 14, 4(15). Upon return, the registers are restored.

If a record has been inserted the record from the queue is held for later processing: the inserted record is considered the next record for transmission. If the inserted record has a zero length the next record from the queue is retrieved and the user routine is re-entered.

Terminal Write (DOCWT205)

The Terminal Control operation code is extracted from the TIOAWCI field to build the DFHTC macro. Recognized operations are write, optical image unit request, line address, and erase. If at least one of these operations is not present, an error message is written and a PUT is performed.

Depending on the state of TWAXTRTN, control is returned to either the user routine or Get Logical Record.

Unchain WRE (DOCWT206)

At the end of a CWTR operation that has been initiated by a WRE, the WRE is removed from the chain and its storage is released. If this WRE is the last one on the chain and was not a status request WRE, the Unchain BCA routine is entered. Otherwise control is passed to WRE Search.

If the BCA is to be deleted because of a user request, as many WRE's as possible are unchained (note: a WRE cannot be unchained if another CWTR is currently using it).

Unchain BCA (DOCWT207)

After all WREs have been removed from a BCA and SAVE doesn't prevent it, the BCA is located on the BCA chain and is removed from it. The BCA storage is released.

If entry to Phase 2 was just to delete the BCA, control is returned to Phase 1 otherwise WRE Search is once again entered to look for more work to do.

ASYNCHRONOUS QUEUE PURGE PROGRAM (DFHAQP)

The Asynchronous Queue Purge Program (AQP) is a CICS System Service program which is part of the Asynchronous Transaction Processing facility. Its function is to perform the purging of data from Transient Data queues used to process batches.

AQP can be ATTACH'ed by either the Asynchronous Transaction Control Program (ATP) or a Asynchronous Transaction Output Processor (WTR) when a batch has been completely processed and all output transmitted to a terminal. When it is ATTACH'ed, a Batch Control Area (BCA) is passed as a facility representing the queues to be purged.

AQP establishes a pointer to the input queue DCT in the BCA and issues a DFHTD TYPE=PURGE macro instruction. This causes Transient Data program to release all associated direct access storage assigned to the queue.

AQP then establishes a pointer to the output DCT in the BCA and, if output exists, issues another DFHTD TYPE=PURGE.

When all queues have been purged, AQP releases the BCA and terminates.

BASIC MAPPING SUPPORT (DFHBMSMM) - CHART 34

The Basic Mapping Support module is linked to as a result of DFHBMS being used. Entry analysis checks that the terminal is within 3270 range. When a map is specified by name, a copy is loaded into main storage. If a terminal read is required, a Terminal Control read is issued. A work area is obtained into which data is mapped. The size of the work area is defined in the first half word of every map.

The input mapping operation analyzes the native 3270 data stream to determine fields which match position with those defined in the input map. Any data entered is moved across into the work area (left justified and blank padded); the length of the input is moved into the work area. This data is positioned down the work area according to the DSECT used by the user to reference an input TIOA.

For a pen detectable field, the flag in the DSECT is set to FF, when the field is selected. End of the map or end of the 3270 native data stream terminates the mapping operation. The 3270 native TIOA is freed and work area passed back to the user as a TIOA.

The output mapping operation checks for a data request of NO, YES or ONLY. For NO, the existing map default data is mapped out. With YES and ONLY, the user must have provided the data to be mapped in a TIOA.. If ONLY was requested, only the user supplied data is mapped, and no default data is sent from the map.

The required data is mapped into the work area to form a 3270 native data stream. Mapping completes when the end of the output map is reached. A Terminal Control write is scheduled; wait and erase are also scheduled if requested.

If the data request was YES or ONLY, the TIOA sent by the user is freed, unless a SAVE request has been made. However, TCTTEDA does not point to the user supplied TIOA when the Basic Mapping Support module returns control to the user application program. The TIOA containing the 3270 native data stream is freed by Terminal Control program upon completion of the requested write.

DL/I INTERFACE (DFHDLI) - CHARTS 35-38

The CICS DL/I Interface consists of four modules:

1. DFHDLI - A CICS management module which services DL/I requests from user application programs.
2. DFHDLA - Called by DFHSIP to bring IMS/360 and DL/I into storage.
3. DFHDLQ - Serves as the application program to DL/I Batch Program Controller (DFSPCC30).

4. DFHDLE - Passes DL/I calls to the DL/I Language Interface (DFSLI000) for processing.

The interface is based on a DL/I batch program executing as an OS subtask of CICS. In this manner, the user of the interface need only have CICS and DL/I Data Base System. (Note: the interface requires the installation of IMS Data Base System, Version 2, Modification Level 2, or later.) The interface passes DL/I requests, one at a time, from CICS transactions to the DL/I action modules for processing, and returns any retrieved data to the calling program. The pre-built blocks feature is used, which uses ACBLIB instead of PSBLIB and DBDLIB.

DFHDLI MODULE

This module is entered via the following statements: DFHFC TYPE=(DL/I,...), CALLDLI in Assembler programs, a CALL 'CBLTDLI' in COBOL programs or CALL PLITDLI in PL/I programs. It is also entered from the Program Control Program during normal or abnormal termination of a program which has made DL/I requests; and from System Termination at CICS shutdown.

The first time this module is entered it initializes the Interface Scheduling Blocks (ISBs) used to simulate message regions to DL/I action modules. Each one contains a protect key for the simulated or pseudo-region, a pointer to the Partition Specifications Table (PST) assigned to it, the OSAM free space management value, and a unique task ID (TCAKCTTA) for the task scheduled into the pseudo-region. The protect key is not a hardware protect key, but is a four-bit identifier used by DL/I routines to differentiate concurrently executed transactions.

For problem determination, the following addresses are stored in the beginning of the program DFHDLI along with their names in EBCDIC.

- Common System Area (CSA)
- Task Control Area (TCA) of transaction currently using DFHDLE or which last used DFHDLE
- SCA (Saved Control Area) of transaction currently using DFHDLE or which last used DFHDLE. The SCA is a copy of TCA+X'80' through TCA+X'DB' which is the TCA Common Communication Area and associated register save area
- The Contents Directory Entry (MVT) or Entry Point (MFT) of DFHDLQ
- DL/I's System Contents Directory (SCD) address
- DL/I's Partition Specification Table (PST) address from DL/I Batch Nucleus (DFSBNUC0).
- The address of the PST acquired for the task currently using DFHDLE or which last used DFHDLE

The System Contents Directory fields SCDLOWID and SCDNAVID for DL/I OSAM free space management are initialized.

Each time this module is entered, storage is acquired to save the TCA's Common Communication Area, register save area, plus some working storage. The area from TCA+X'80' to TCA+X'D8' is moved to the Saved Control Area (SCA). This allows those fields in the TCA to be used for other CICS calls by DFHDLI. The SCA is released when the interface returns to the calling program.

An entry is made in the CICS trace table using code X'F8'.

TCADLFUN is checked for the function desired. The function may be one of the following:

- 'PCB' - schedule a PSB and return PCB addresses
- 'T' - unschedule and free the PSB
- 'TERM'- terminate the CICS-DL/I subtask
- All other codes

'PCB' FUNCTION - SCHEDULE PSB AND LOCATE PCB ADDRESSES

If the function is 'PCB', the interface performs the actions required to locate and bring the required PSB into main storage, returning a list of PCB addresses to the calling program. In effect it performs most of the functions of the DL/I Data Communcation System Application Scheduler. Up to 15 DL/I requesting transactions (the number of available storage protect keys, which differentiate log records) may be passing calls through the interface. These transactions are considered to be operating in different regions (pseudo-regions) and are identified by the pseudo-protect key assigned during scheduling. Information is saved from the 'PCB' call to the transaction termination call and retained in the Interface Scheduling Block (ISB). There are 15 ISBs, each of which contains the following; pseudo-protect key, the unique task identification TCAKCTTA (assigned to the task currently executing in the pseudo-region, or binary 0 if none), starting time of the transaction, and the SCDNAVID number assigned to the task for free space management.

Whenever the scheduled PSB name is not supplied by the user, the interface moves the Program Control Table entry for this transaction into TCADLPSB. Since only one transaction at a time may use a PSB, the program attempts a CICS enqueue on the PSB name. Once this task has control of the PSB, the program searches for an unused ISB. If all are in use, the task is put into a CICS wait state, waiting on DLISBECB until a transaction terminates and releases an ISB. Storage is then acquired, a DL/I application scheduling log record is built, and the address is stored in TCADLIO.

PST copy storage is acquired, and then copied from the batch nucleus. The protect key of the ISB assigned to this pseudo-region is entered in the PST and the address of the PST is stored in the ISB.

To ensure that only one transaction at a time goes to DFHDLE for scheduling, DLSCHECB is checked. If the WAIT bit is on, transactions are locked out from going to DFHDLE.

When the program passes control to DFHDLE, and it determines that this is a 'PCB' call, register 13 is loaded with the address of a different save area set than the one used by all other calls. Control is passed to the DL/I ACB Block Loader (DFSDBLM0) which attempts to load the PSB and DMB's required for this transaction. If DFHDBLM0 and its subprograms can locate (and load if necessary) the required blocks, it returns to DFHDLE with the PSTSCHED bit on in the PSTCODE1 byte in the PST. This indicates that the transaction may be scheduled. The address of the PSB list is placed in TCADLPCB and the scheduling log record is written to the DL/I log. When control returns from DFHDLE, the wait bit in DLSCHECB is set off, allowing other transactions to use the scheduling code. The TCAOFDLI bit is set on in TCAOFDI to indicate that this transaction is scheduled to use DL/I.

When not enough room exists in the PSB or DMB pool for the required blocks, DFSDPDM0 (the DL/I Pool Manager) secures the address of a retry routine from PSTSMB. The address was placed there by DFHDLQ. The retry routine is part of DFHDLE. It directs the pointer to the special save area set, places a -1 in TCADLPCB to indicate retry and returns to DFHDLI. The transaction is put into a CICS WAIT state waiting on DLPSBECB ("waiting for pool space") until another transaction using DL/I terminates. At this time the 'PCB' call is re-issued. Upon finding a -1 in TCADLPCB, DFHDLE refers to the retry routine which returns to DFSDPDM0. Again, the Pool Manager attempts to get storage. This process is repeated until the storage is acquired or no other tasks are left. In the latter case the Pool Manager returns with a return code of 4, indicating insufficient pool space. The transaction is terminated with a 992 or 993 pseudo-abend (depending on which pool was too small), and the directory of the PSB or DMB which was too large is stopped.

DL/I TRANSACTION TERMINATION AND BLOCK UNSCHEDULING - 'T' CALL

In the processing of ordinary DL/I functions, the interface sets the TCAOFDLI indicator on in TCAOFDI indicating to the Program Control Program (PCP) that it must pass control to the interface at normal or abnormal termination of transactions. PCP does this by coding a DFHFC TYPE=(DL/I,T) statement when a transaction terminates and the TCAOFDLI bit is on. The interface sets the TCAOFDLI bit off, and builds a DL/I application termination log record. The log record is filled with call counts and completion codes from the PST and the unique task ID from the ISB. The address of the log record is stored in TCADLIO. The ISB for this pseudo-region is cleared. The program branches to pass control to the DL/I subtask (DFHDLE) which writes the log record and marks as unscheduled the PDIR for the PSB which is now unused. Upon return from the subtask, the program does a CICS dequeue of the PSB used in the transaction so that another transaction can use it. The "ISB available" ECB (DLISBECB) and "waiting for pool space" ECB (DLPSBECB) are posted.

If any transactions are waiting for a free ISB, the top priority one is made dispatchable. When it is dispatched, it searches for a free ISB. If a transaction is waiting for pool space, it is made dispatchable. When it is dispatched, it goes to DFHDLE again to get pool space.

A 'T' call can also be issued by an application program. The effect is to free the PSB from the transaction and set up associated blocks available to other transactions.

TERMINATION OF DL/I SUBTASKS - 'TERM' CALL

When the CICS system terminates, a special 'TERM' call is made to DFHDLI. The call is passed to DFHDLE normally terminating it. OS will POST DFHIMECB (the task termination ECB specified by DFHDLQ when it attached DFHDLE). DFHDLQ resumes and after finding a zero return code, is aware termination is in progress. It detaches the TCB of DFHDLE, clears DFHDLTCA, posts TCADLECB with a zero return code and terminates normally. DFHDLI resumes and checks DFHCIECB which will be posted when DFHDLQ terminates. If the ECB is not already posted the program does a CICS WAIT on it. When posted, the completion code is checked. If it is zero, a console message is issued indicating DL/I terminated normally. If it is not zero, a message with the completion code included is issued indicating abnormal termination. In either case the program returns to the System Termination routine.

ORDINARY DL/I CALLS

If a call is not 'PCB', 'T' or 'TERM' then it is assumed that an ordinary DL/I call exists. The following items are checked:

- The calling language
- Whether it was a CALL, CALLDLI, or DFHFC TYPE=(DL/I)
- Whether there were Segment Search Arguments (SSAs) in the call

All calls are reformatted if necessary into an assembler variable-length parameter list. The address of the list is placed in SCADLPAR. The TCA Common Communication Area is moved from the SCA to the TCA and the address of the TCA is stored in DFHDLTCA in the interface parameter list. If no other transaction is calling DL/I at this time, the interface assumes control of DL/I. This is accomplished by setting on the first bit in DLDLIECB which single-threads control to DFHDLE. If the bit is already on, the interface issues an internal CICS WAIT (DCI=CICS) on DLDLIECB until DFHDLE is free. The ECB for DFHDLE (DFHEXECB) is posted and the transaction is put into a wait state for two ECBs; TCADLECB - the transaction's ECB, and DFHIMECB - the ECB posted if DFHDLQ ABENDS. This allows DFHDLE to assume control and process the call. When DFHDLE completes the call, the interface regains control. DLDLIECB is posted, DFHEXECB is waited on, and the next transaction in priority can pass control to DL/I.

Any storage which was acquired to build parameter lists or SSA lists by the calling macro is released. DFHCIECB is checked. If it is posted and the return code is zero, return is made to system termination since this was a 'TERM' call. If the return code is not zero, the message "DFH3900 - DL/I INTERFACE ABENDED" is sent to destination CSMT and the interface entrance is altered to pass control to the dummy program. The interface then returns to the calling transaction with an invalid request return code.

If DFHCIECB is not posted, TCADLECB is checked. If it is 0, the message slots pointed to by DFHDLMSG are checked. The Message Generation in DFHDLE places the address of any DL/I messages (up to 7) in the message slots. If there are any messages, they are sent to destination CSMT and the slots are cleared. If the field TCADLECB is not 0, then DL/I pseudo-ABEND occurred or DFHDLE ended with a system ABEND code. The completion code is reduced to system and user ABEND codes, and are placed in message "DL/I PSEUDO-ABEND--Snnn--Unnn" which is sent to destination CSMT. The message slots are processed as above and the transaction is abended with a DLPA ABEND code.

If the call did not ABEND, the DL/I function is checked to determine whether it was a creative call. If so, storage is acquired for a DL/I work-area, based on the size contained in the PST at the halfword location PSTSEGL+2. Data is moved from DL/I's storage to the work-area and its address is placed in TCADLIO. The TCAOFDLI bit is set on. Control is then returned to the calling transaction.

DFHDLIDY - CICS-DL/I Interface Dummy Program

This program is generated as part of the control system dummy group. It is also included at the end of DFHDLI if the interface program is generated. The purpose is to return a FCP INVREQ (X'08') indicator if the interface program was not loaded or error caused the interface to fail and terminate DL/I processing. Initial program entry sends a message to the operator's console and destination CSMT to indicate that a DL/I call was processed by the dummy program. Each time the dummy program is entered the TCAFCTR (TCAFCRC if the language is ANS

COBOL) field is set to X'08' to indicate INVREQ (Invalid Request) and returns to the calling program.

DFHDLE - Call Executor

This program is a subtask of DFHDLQ, operating as the application program to DL/I. When it is attached, it is passed the address of EXPARML, a parameter list containing the address of DFHDLI (which is used as the base of the interface parameter list DSECT). It also contains addresses of DL/I PXPARMS and DL/I LIPARMS, parameter lists used by the DL/I Language Interface (DFSLI000). This address is stored in the register1 word of the top problem program save area so that DFSLI000 can find the SCD and the PST when called.

This program can handle two calls at once - a 'PCB' call, and if that call is waiting for PSB or DMB pool space, any other type of call. Two save area sets are therefore needed. One, which services all calls except 'PCB' calls, is contained in the batch nucleus. The other is provided in this program.

For 'PCB' processing, register 13 is loaded with the address of a special save area, to which the special save area set is chained. For any other call register 13 contains the address of the save area provided to it upon entry, and DFSLI000 chains the batch nucleus save area set to it. Control is then passed to DFSDBLM0, the ACB Block Loader and its subroutines, which attempt to load and schedule the PSB and DMB's. If the attempt is successful, the PSTSCHED bit in PSTCODE1 is set on. The address of the PCB is left blank in the TCA and control is returned to DFHDLE by posting TCADLECB and waiting on DFHEXECB.

DFHDLQ - DL/I Application Program

During CICS initialization, the SIP attaches DFHDLA which transfers control to DFSRRC00 (the DL/I Region Controller), passing a parameter list containing the program name DFHDLQ. DFSPCC30 (the DL/I Batch Region Program Controller) links to DFHDLQ. This program is identified to DL/I as the batch application program. Its functions are:

- Locate the interface parameter list in DFHDLI
- Locate the SCD and PST in the batch nucleus
- Fill in entry points of ACB scheduling, ENQ/DEQ storage management modules, PDIR list and DDIR list in the SCD
- Set up the control data for DFSISMN0 (Storage Management)
- Call DFSIIND0 to initialize PSB and DMB directories
- Attach DFHDLE, the DL/I call executor, indicating to OS to POST DFHIMECB when DFHDLE terminates.
- Acquire and initialize storage for the PSB and DMB pools.

The primary function of this program is to establish communication with DFHDLI by locating the interface parameter list in DFHDLI. The first word of the highest level save area for this task contains the address of the CSA. The address was placed there by DFHDLA. Using the CSA, the program locates the interface parameter list.

Using the first PCB in the initialization PSB, the program performs a GSCD call to locate the SCD and the PST in the batch nucleus and places these addresses in the interface parameter list in DFHDLI.

Using PSB and DMB pool sizes passed to DFHDLA by the SIP, the program determines the amount of storage required for these pools, including overhead. It stores these values in the pool initialization lists in DFHDLI. DFSISMN0 is then called to obtain the storage and initialize the pools.

The program then attaches DFHDLE which will actually perform all DL/I calls. In the ATTACH, it indicates to OS to POST DFHIMECB when DFHDLE terminates. This is similar to the DL/I online system where DFSRRC10 (online region controller) attaches DFSPCC10 (online program controller). If any DL/I module ABENDS, DFHDLE will terminate. Program DFHDLQ attaches DFHDLE, stores its TCB address in DFHSTTCB, posts DFHCIECB, and waits on DFHIMECB, which will be posted whenever DFHDLE terminates.

When DFHDLQ regains control, it checks the completion code in DFHIMECB. If it is zero, it assumes DFHDLE terminated normally as a result of a 'TERM' call issued by CICS System Termination to terminate DL/I processing. It detaches the TCB of DFHDLE and posts TCADLECB causing control to return to System Termination issuing a RETURN (14,12),RC=0 command causing the DL/I subtask to terminate. As a result all OS subtasks of CICS are terminated normally.

If the completion code in DFHIMECB is not zero, the program assumes that DFHDLE abended. It checks to determine if this is the result of a DL/I module ABEND or pseudo-abend. If either abend occurred, it detaches the TCB of DFHDLE, re-attaches DFHDLE and posts TCADLECB with the pseudo-abend code. The program waits again for DFHIMECB. If OS cannot successfully ATTACH or re-attach DFHDLE, DFHDLQ ABENDS with an OS return code in register 1. When DFHDLI detects there is a non-zero return code in DFHCIECB, it will issue a DLIA ABEND against the transaction, send message DFH3900 - DL/I INTERFACE FAILED to destination CSMT, and cause all further entries to be sent to the dummy program.

DFHDLA - Attach DL/I Region Control Program (DFSRRC00)

When the System Initialization Program (SIP) is processing initiation parameters, it checks whether or not DL/I is to be used in this session. If so, SIP processes the BUFPL (DL/I Buffer Pool), PSBPL (PSB pool), DMBPL (DMB pool) and PSB (initiation PSB name) in the execution parameters and/or SIT loaded. The defaults are BUFPL=8 (expressed in 1024 byte blocks), PSBPL=4, DMBPL=4, PSB=CICSPSB.

SIP builds a parameter list containing the following:

/H(size)/A(CSA)/DBB,DFHDLQ,psbname,bufpool,00,psbpool,dmbpool/

SIP attaches program DFHDLA with register1 pointing to the parameter list.

DFHDLA places the address of the CSA in the first word of the first save area associated with the TCB. This enables DFHDLQ to later access the CSA. Next, the PSB and DMB values are placed in the pool initialization lists in DFHDLI. The length of the field is adjusted and placed in front of the word DLI. The final list is as follows:

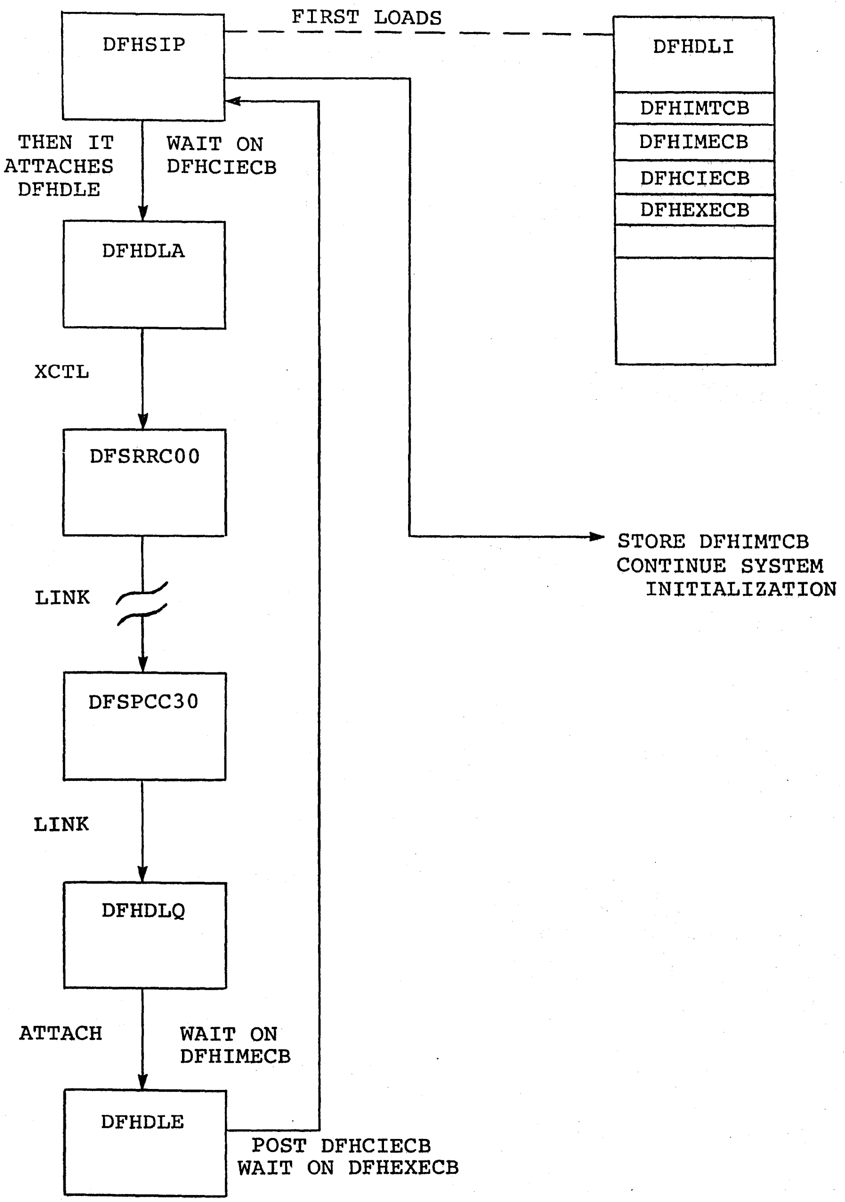
/H(size)/DBB,DFHDLQ,psbname,bufpool,00

The address of this list is placed in register 1. Registers 2-12 are restored and DFHDLA transfers control to DFSRRC00 by means of the XCTL macro instruction. The usual course of DL/I batch initiation then begins.

CICS-DL/I CHARTS

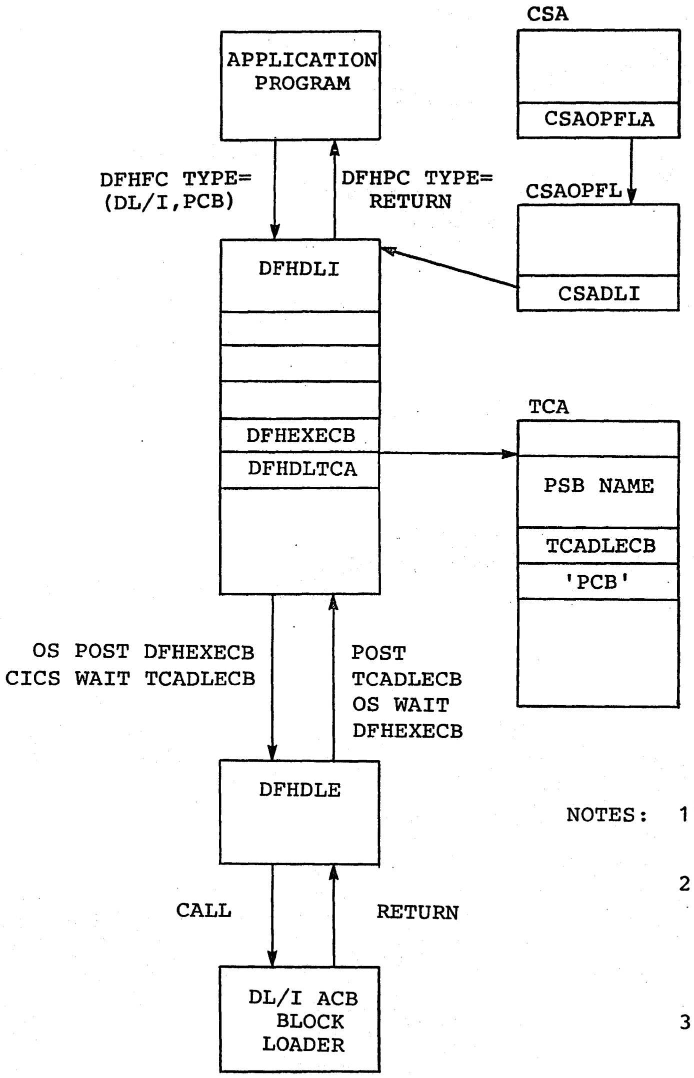
The following charts give additional information on the flow of control through CICS and DL/I modules indicating important fields in modules and control blocks.

INITIATION OF DL/I SUBTASKS



x

'PCB' (SCHEDULING) CALL



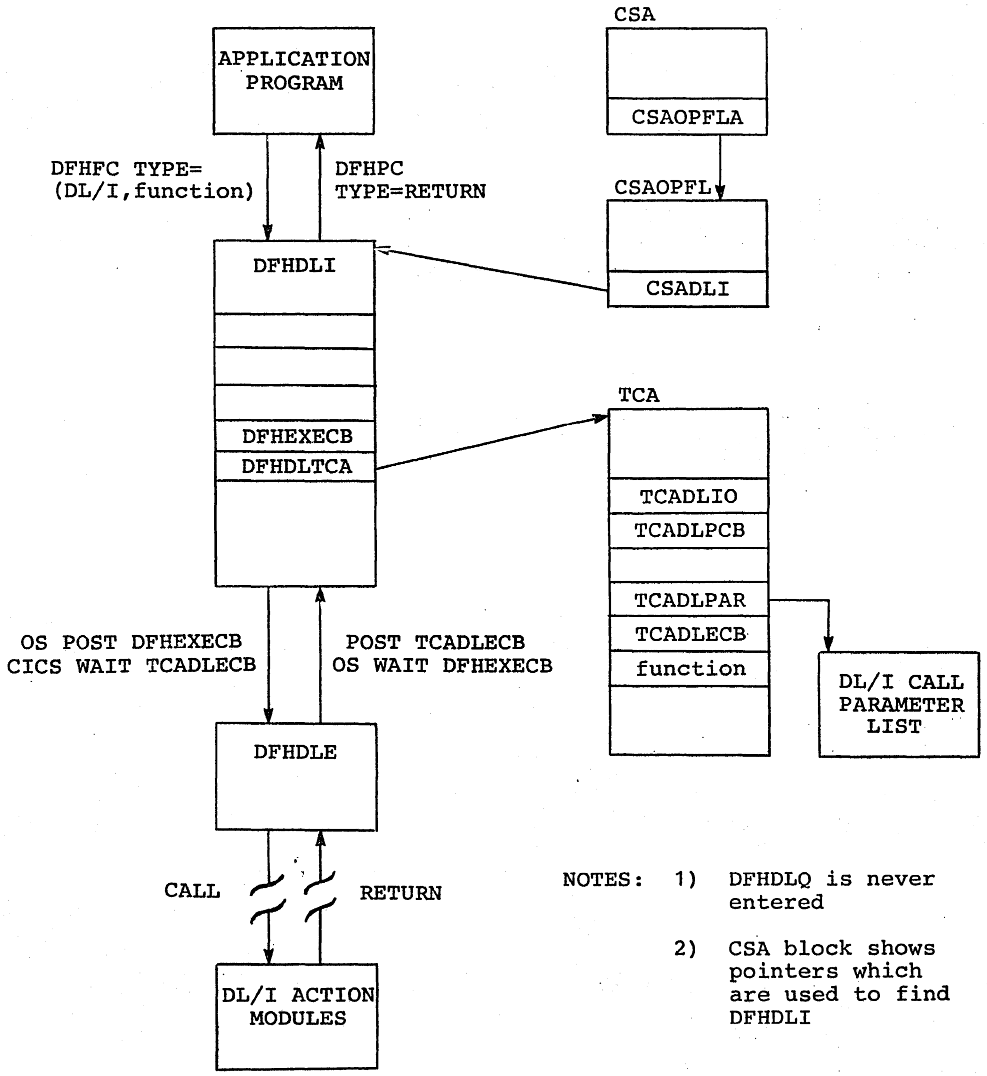
NOTES:

1) DFHDLQ is never entered

2) DFHDLI and DFHDLE may exchange control many times before scheduling is successful

3) CSA block shows pointers which are used to find DFHDLI

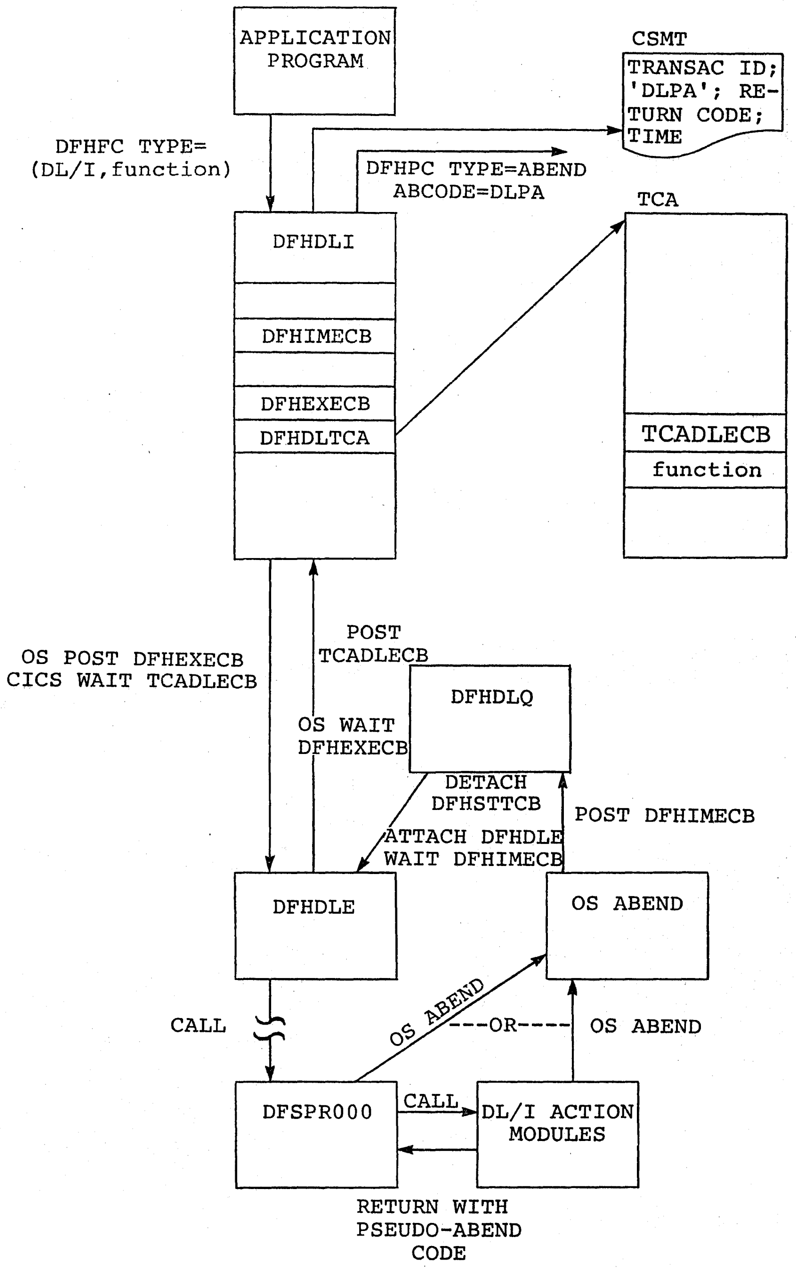
ORDINARY DL/I CALL (INCLUDING CICS PSEUDO-CALLS EXCEPT 'PCB')



NOTES: 1) DFHDLQ is never entered

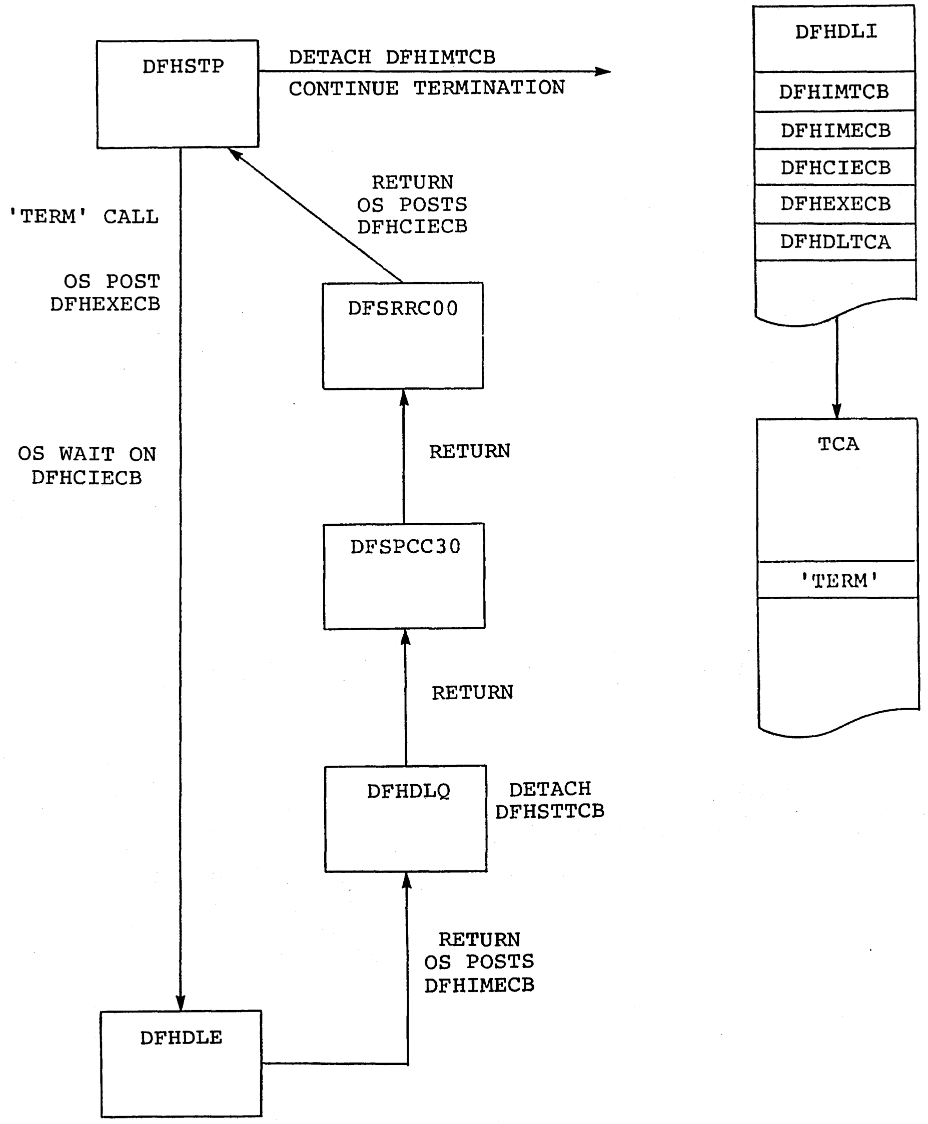
2) CSA block shows pointers which are used to find DFHDLI

PROCESSING OF DL/I ABEND OR PSEUDO-ABEND



X

TERMINATION OF DL/I SUBTASKS



CICS/TCAM INTERFACE (DFHTCAM) - CHART 39

QUEUE TYPE (DOCTCPY1)

The TACP error code is zeroed and the type of queue is determined (Input or Output).

INPUT PROCESSING (DOCTCPY2)

Storage is obtained prior to reading from the TCAM partition/region. A branch and link is made to the Read/Write routine. After completion, the appropriate internal indicators are set and a branch and link is made to the TCTTE Search routine. If a TCTTE is found and processed, a normal exit is made to Terminal Control Line Advance.

OUTPUT PROCESSING (DOCTCPY3)

The output group of TCTTE's is scanned for write requests. A branch and link is made to the Read/Write routine to accomplish the transfer of data to the TCAM partition/region. A check is made for a good completion and a branch is made to the common exit. If an abnormal completion results, the appropriate TACP error code is set and an exit is made to TACP.

TCTTE SEARCH (DOCTCPY4)

The pointer to the output queue is obtained from the input queue and a check is made for POOL=YES.

FIND SPECIFIC TCTTE (DOCTCPY5)

The input destination header is matached to a specific TCTTE. Exit is made to TACP if no match is made.

FIND AVAILABLE TCTTE (DOCTCPY6)

The scan is made here if POOL=YES was specified for the group of TCTTE's attached to the output queue. After the input TIOA is attached to a TCTTE, the user exit XTCMIN is taken and/or the destination header is removed. Input device-dependent processing takes place at this point.

TASK INITIATION (DOCTCPY7)

A link is made to the Task Initiation routine and if the system has sufficient storage, the task is initiated.

READ/WRITE ROUTINE (DOCTCPY8)

If a read is being processed, the branch and link is made to transfer the data across partition/region boundaries. If a write is being processed, the user exit XTCMOUT is taken and/or the TCAM destination header is built. Some device-dependent processing is done before data transfer takes place.

SEGMENT PROCESSING (DOCTCPY9)

If OPTCD=C is specified for the input queue DCB, the user furnished segment identifier is moved from TCTTETCM and prefixed to the output data.

INPUT STORAGE CONTROL (DOCTCPYA)

Read in storage is obtained here for the input operation. It is attached to the input queue TCTTE and if OPTCD=C is specified for the input queue DCB the data pointer is set to the segment identifier which is later moved to TCTTETCM.

INPUT EDIT ROUTINE (DOCTCPYB)

The TCAM Destination header is removed and the data pointers are adjusted.

ERROR PROCESSING (DOCTCPYC)

Various error conditions are processed here. TACP error codes are set and proper exit is made to TACP. When input data cannot be disposed of readily, the input queue is temporarily suspended for the amount of time indicated in NPDELAY.

If an unsolicited input error occurs, the unsolicited data is placed on the Input Queue TCTTE. Then the input TCTTE is placed out of service and a pointer to the offending output Terminal Entry is placed at the label TCTTECA on the Input Queue TCTTE.

CICS/TCAM COMMUNICATIONS PROCESSING (DOCTCPYD)

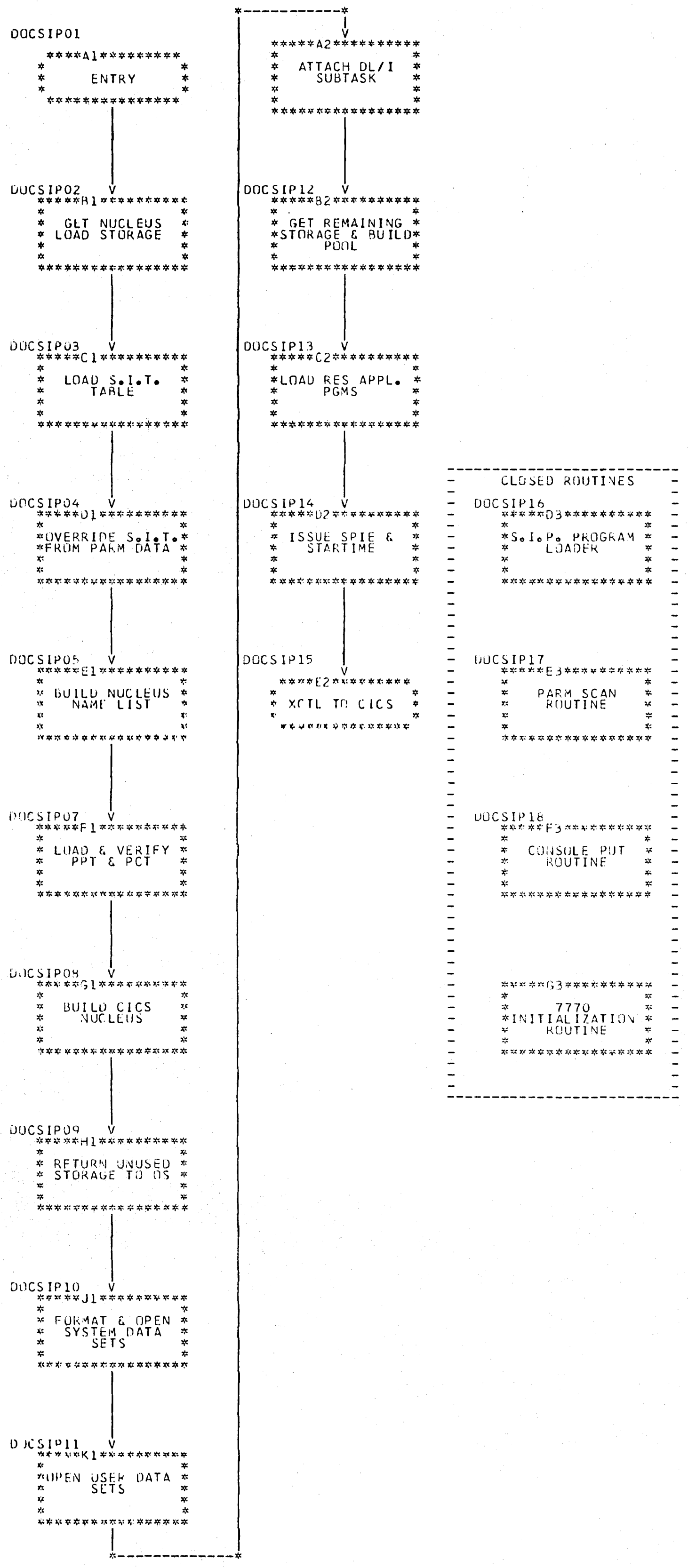
The device dependent communication byte is set up to pass to the MCP requests that cannot be handled in CICS (for example, 2260L Write - Lock).

FLOWCHARTS

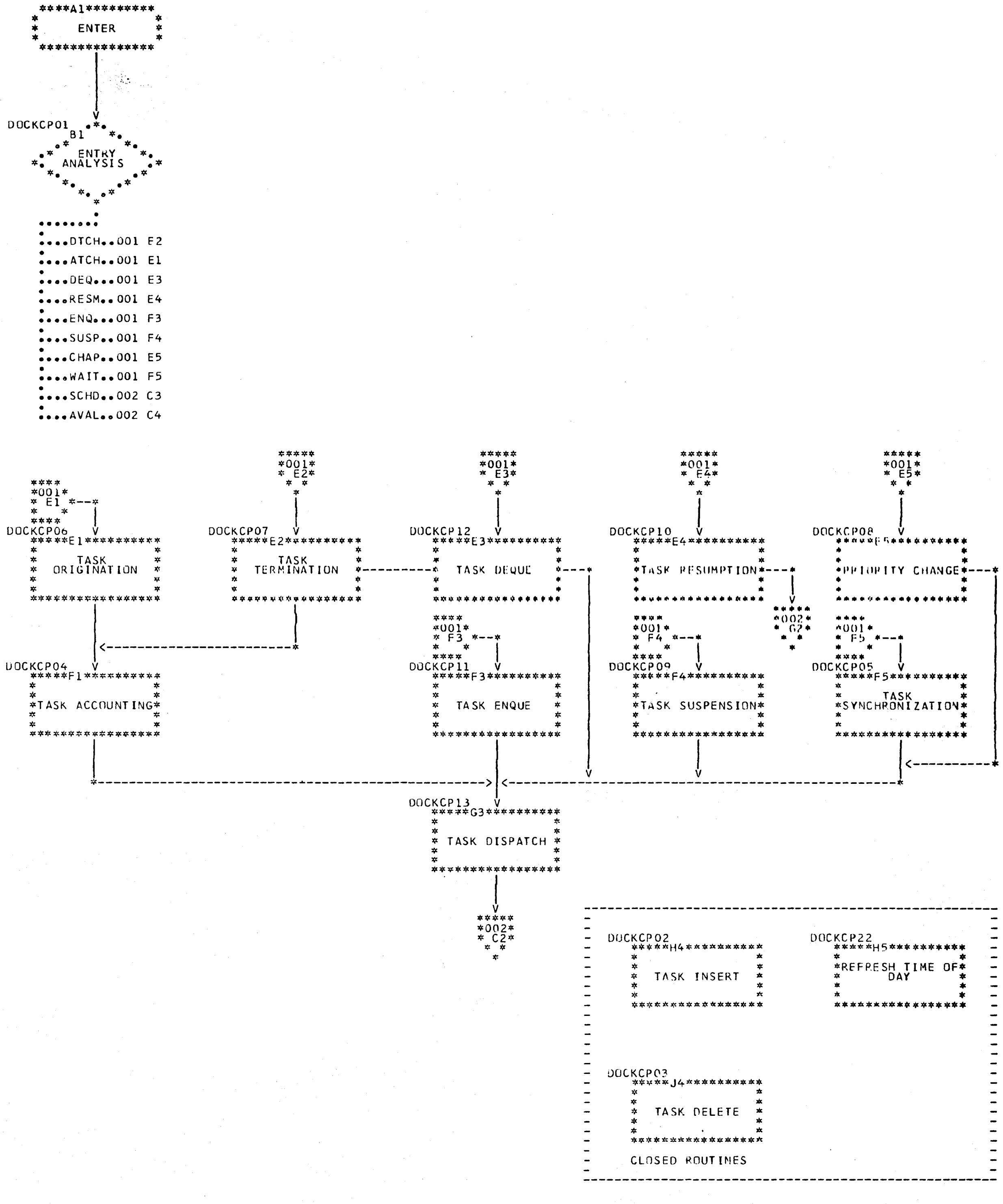
The following flowcharts are provided in this section:

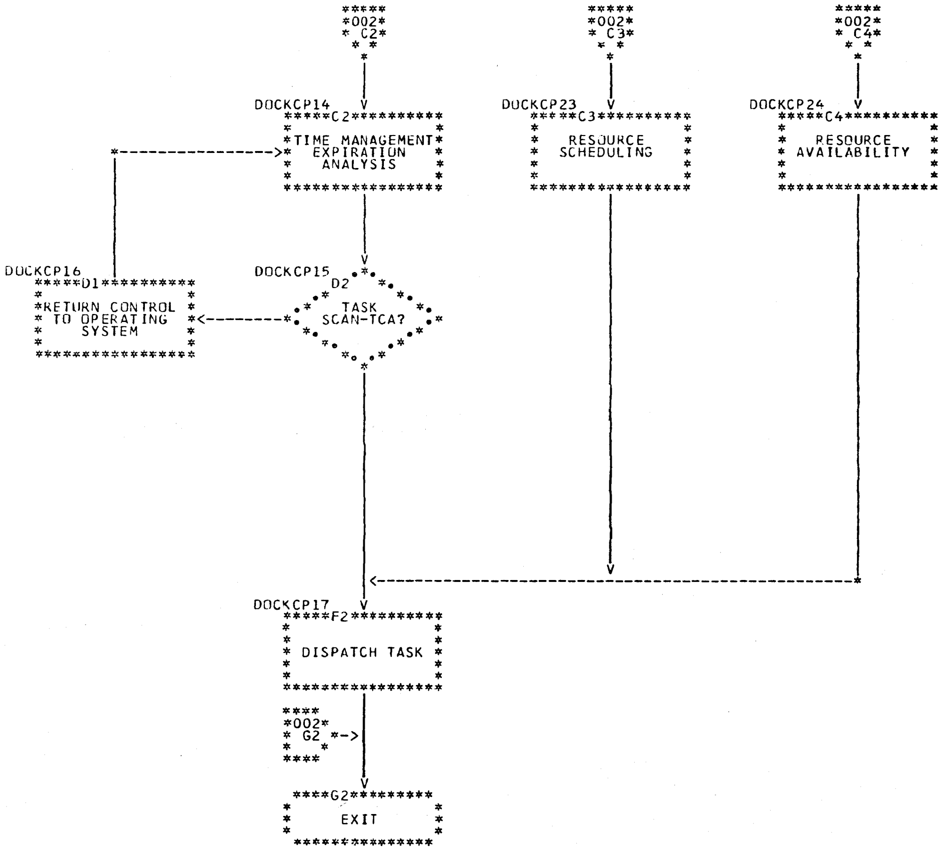
- System Initialization Program (DFHSIP)
- Task Control Program (DFHKCP)
- Interval Control Program (DFHICP)
- Storage Control Program (DFHSCP)
- Program Control Program (DFHPCP)
- Program Interrupt Program (DFHPIP)
- Dump Control Program (DFHDCP)
- Terminal Control Program (DFHTCP)
    - Start/Stop Common Routines
    - Binary Synchronous Common Routines
    - Sequential Terminal Dependent Module (DFHTCSAM)
    - Local 2260 Terminal Dependent Module (DFHTC60L)
    - 1030 Terminal Dependent Module (DFHTC30N)
    - 1050 Terminal Dependent Module (DFHTC50N)
    - Remote 2260/2265 Terminal Dependent Module (DFHTC60N)
    - 2740 Terminal Dependent Module (DFHTC40N)
    - 2741 Non-switched Terminal Dependent Module (DFHTC41N)
    - 1050 Dial-up Terminal Dependent Module (DFHTC50S)
    - 2740 Dial-up Terminal Dependent Module (DFHTC40S)
    - 2741 Dial-up Terminal Dependent Module (DFHTC41S)
    - TWX Terminal Dependent Module (DFHTCTWX)
    - System/7 Dial-up Terminal Dependent Module (DFHTCS7S)
    - System/7 Non-switched Terminal Dependent Module (DFHTCS7N)
    - 7770 Terminal Dependent Module (DFHTC77S)
    - 2770 and Programmable Terminal Dial-up Module (DFHTCS70)
    - 2780 Dial-up Terminal Dependent Module (DFHTCS80)
    - 2770 and Programmable Non-switched Terminal Dependent Module (DFHTCN70)
    - 2780 Non-switched Terminal Dependent Module (DFHTCN80)
    - 2980 Terminal Dependent Module (DFHTCN29)
    - 3735 Dial-up Terminal Dependent Module (DFHTCS35)
    - Local 3270 Terminal Dependent Module (DFHTC70L)
    - Remote 3270 Terminal Dependent Module (DFHTC70R)
    - 2260 Compatibility Terminal Dependent Module (DFHTCCP)
- Dynamic Open/Close Program (DFHOCP)
- File Control Program (DFHFCP)
- Transient Data Control Program (DFHTDP)
- Temporary Storage Control Program (DFHTSP)
- Sign-on/Sign-off Program (DFHSNP/DFHSFP)
- Master Terminal Program Module A (DFHMTPA)
- Master Terminal Program Module B (DFHMTPB)
- Master Terminal Program Module C (DFHMTPC)
- Master Terminal Program Module D (DFHMTPD)
- Master Terminal Program Module E (DFHMTPE)
- Master Terminal Program Module F (DFHMTPF)
- System Statistics Program (DFHSTKC)
    - Terminal Statistics (DFHSTTR)
    - File Statistics (DFHSTTD)
- Abnormal Condition Program (DFHACP)
- Terminal Test Program (DFHFEP)
- Dump Utility Program (DFHDUP)
- System Termination Program (DFHSTP)
- Trace Control Program (DFHTRP)
- High-Level Language Preprocessor (DFHPRPR)
- Device Dependent Module (DFHDDM)
- Terminal Error Program (DFHTEP)
- Time Adjustment Program (DFHTAJP)

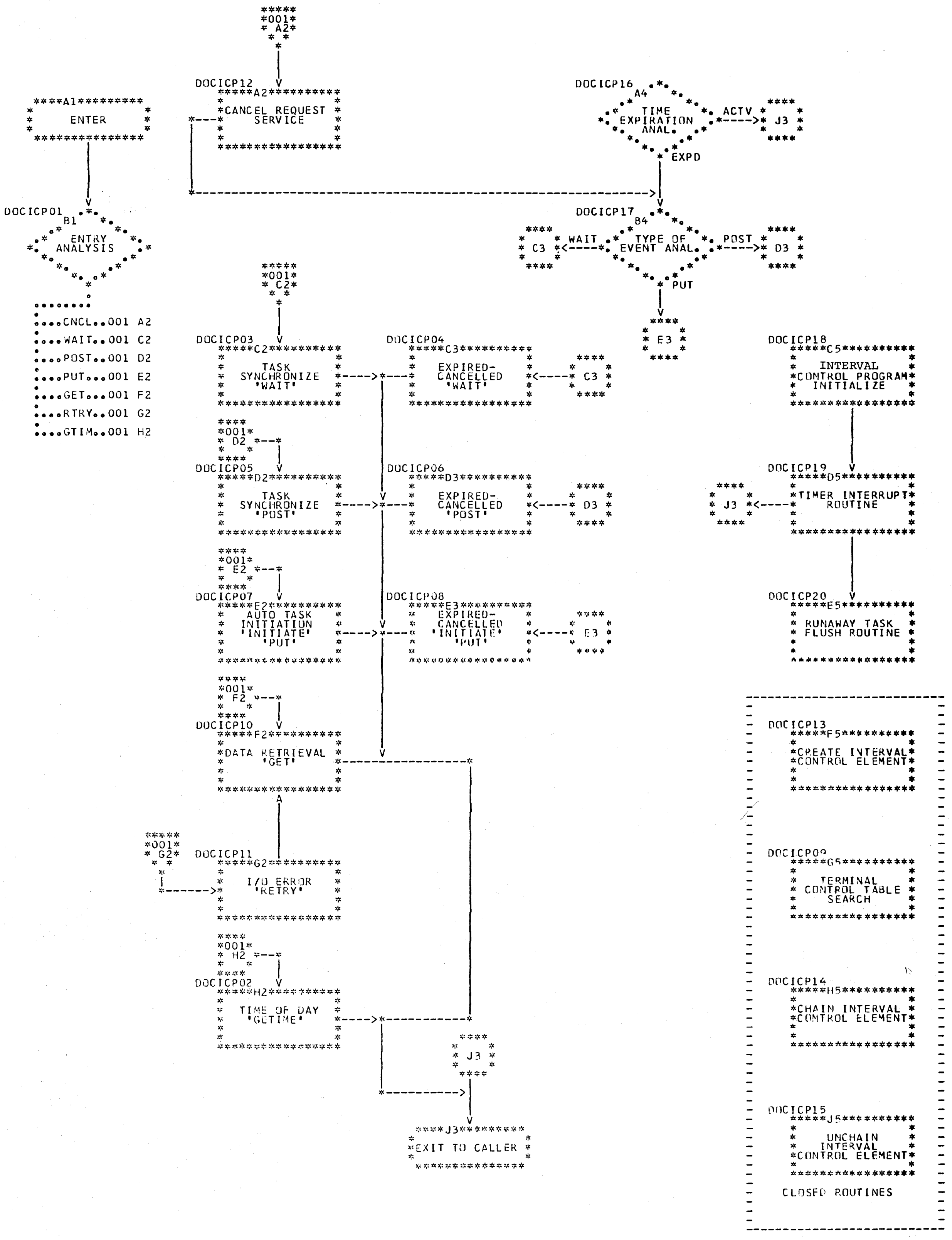
- Dummy CSA Program (DFHDCSA)
- PL/I Storage Allocation Program (DFHSAP)
- PL/I Interface Program (DFHPL1I)
- Terminal Abnormal Condition Program (DFHTACP)
- Asynchronous Transaction Control Program (DFHATP)
    - Asynchronous Transaction Input Processor (DFHRD1, DFHRD2)
    - Asynchronous Transaction Output Processor (DFHWT1, DFHWT2)
    - Asynchronous Transaction Queue Purge Program (DFHAQP)
- 3270 Basic Mapping Support (DFHBMSMM)
- CICS-DL/I Interface Initialization Program (DFHDLA)
- CICS-DL/I Interface Application Program (DFHDLQ)
- CICS-DL/I Interface CALL Execution Program (DFHDLE)
- CICS-DL/I Interface CALL Initiation Program (DFHDLI)
- CICS/TCAM Interface Program (DFHTCAM)


DOCSIP01
A1
ENTRY
DOCSIP02
B1
GET NUCLEUS LOAD STORAGE
DOCSIP03
C1
LOAD S.I.T. TABLE
DOCSIP04
D1
OVERRIDE S.I.T. FROM PARM DATA
DOCSIP05
E1
BUILD NUCLEUS NAME LIST
DOCSIP07
F1
LOAD & VERIFY PPT & PCT
DOCSIP08
G1
BUILD CICS NUCLEUS
DOCSIP09
H1
RETURN UNUSED STORAGE TO OS
DOCSIP10
J1
FORMAT & OPEN SYSTEM DATA SETS
DOCSIP11
K1
OPEN USER DATA SETS
A2
ATTACH DL/I SUBTASK
DOCSIP12
B2
GET REMAINING STORAGE & BUILD POOL
DOCSIP13
C2
LOAD RES APPL. PGMS
DOCSIP14
D2
ISSUE SPIE & STARTIME
DOCSIP15
E2
XCTL TO CICS
CLOSED ROUTINES
DOCSIP16
D3
S.I.P. PROGRAM LOADER
DOCSIP17
E3
PARM SCAN ROUTINE
DOCSIP18
F3
CONSOLE PUT ROUTINE
G3
7770 INITIALIZATION ROUTINE

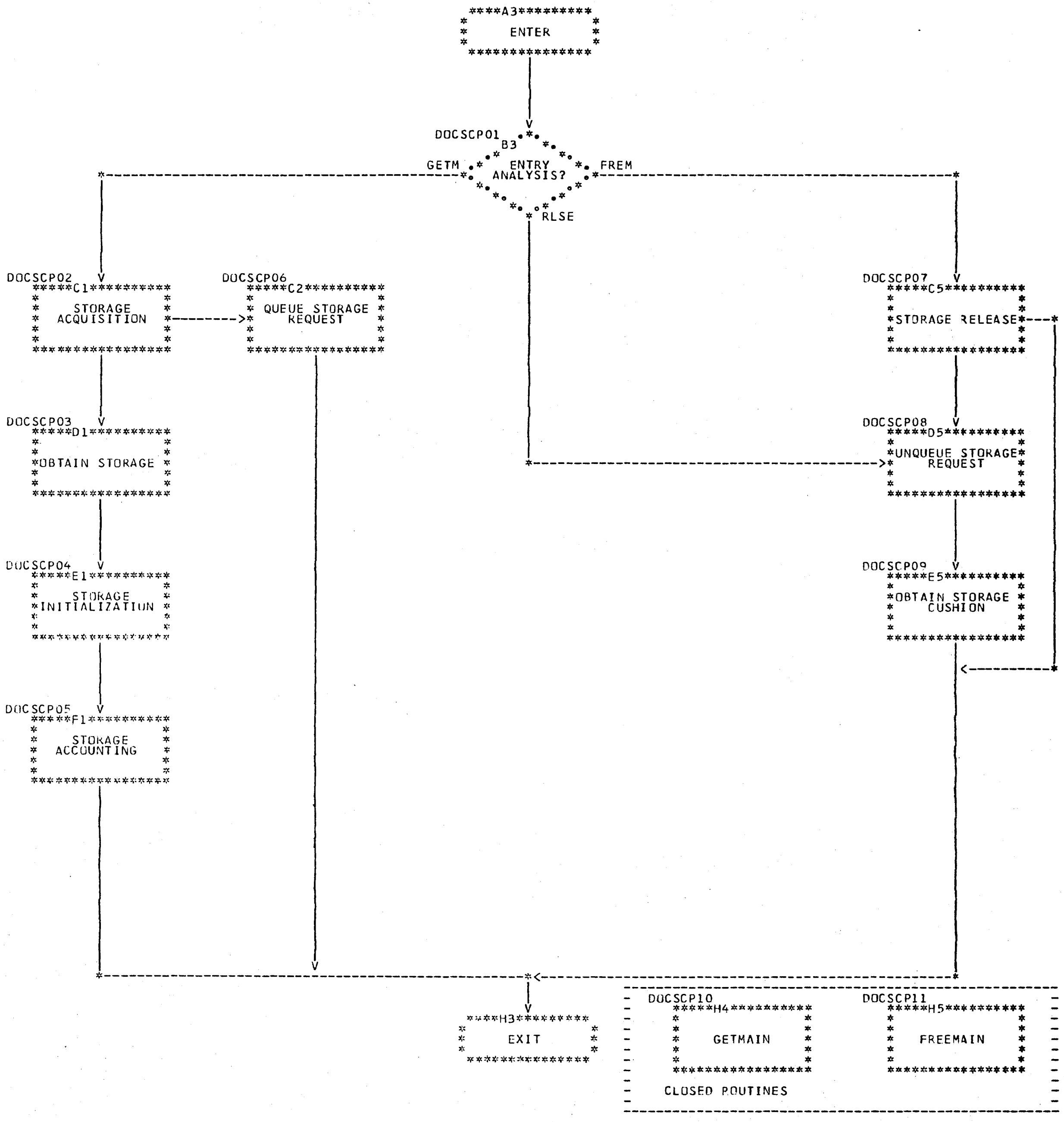
TASK CONTROL PROGRAM
DFHKCP
CHART 2



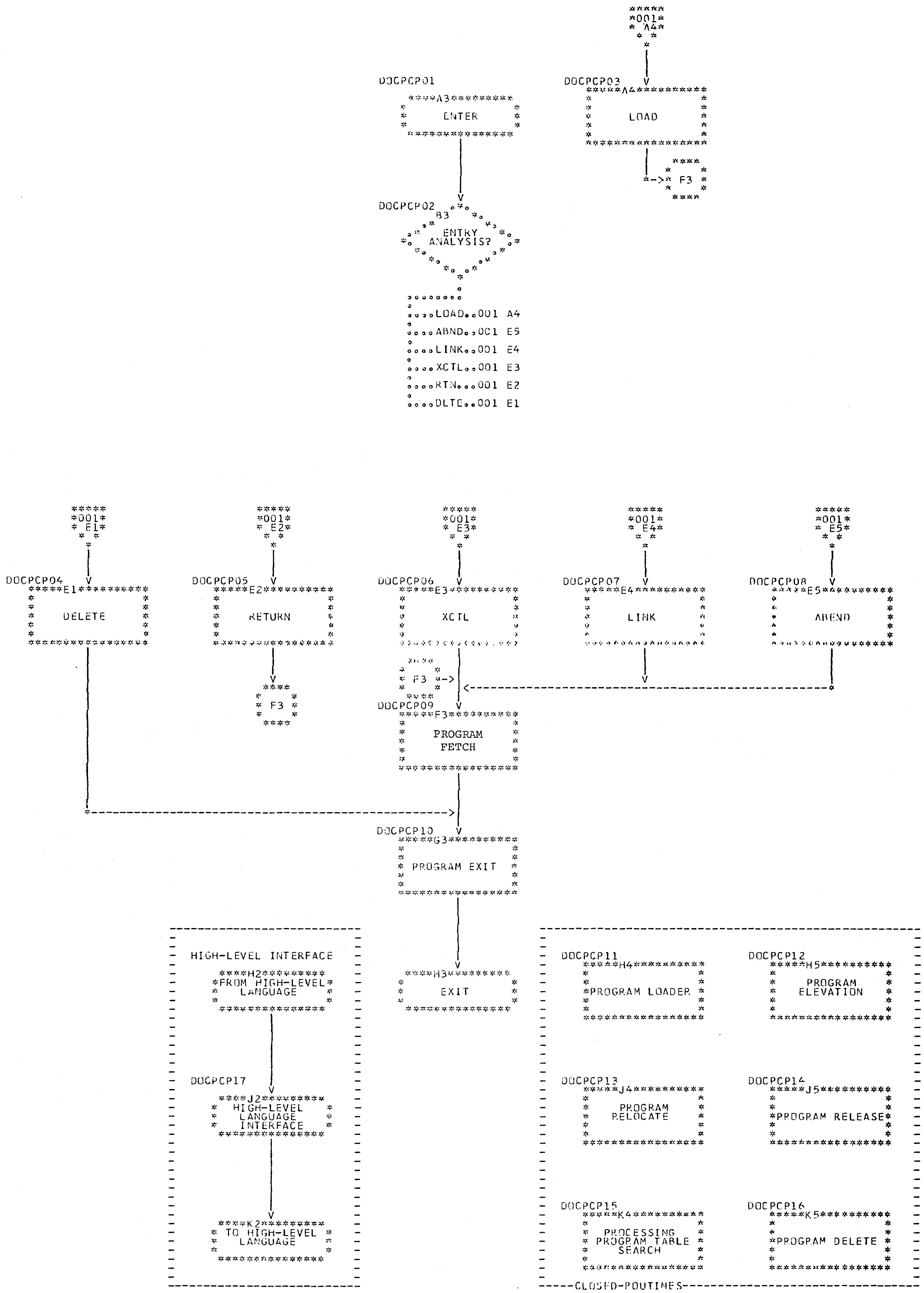

002 C2
002 C3
002 C4
DOCKCP14
C2
TIME MANAGEMENT EXPIRATION ANALYSIS
DOCKCP23
C3
RESOURCE SCHEDULING
DOCKCP24
C4
RESOURCE AVAILABILITY
DOCKCP16
D1
RETURN CONTROL TO OPERATING SYSTEM
DOCKCP15
D2
TASK SCAN-TCA?
DOCKCP17
F2
DISPATCH TASK
002 G2
G2
EXIT


001 A2
A1 ENTER
DOCICP12 A2 CANCEL REQUEST SERVICE
DOCICP16 A4 TIME EXPIRATION ANAL.
ACTV
J3
EXPD
DOCICP01 B1 ENTRY ANALYSIS
CNCL..001 A2
WAIT..001 C2
POST..001 D2
PUT...001 E2
GET...001 F2
RTRY..001 G2
GTIM..001 H2
DOCICP17 B4 TYPE OF EVENT ANAL.
C3
WAIT
POST
D3
PUT
E3
001 C2
DOCICP03 C2 TASK SYNCHRONIZE 'WAIT'
DOCICP04 C3 EXPIRED-CANCELLED 'WAIT'
C3
DOCICP18 C5 INTERVAL CONTROL PROGRAM INITIALIZE
001 D2
DOCICP05 D2 TASK SYNCHRONIZE 'POST'
DOCICP06 D3 EXPIRED-CANCELLED 'POST'
D3
DOCICP19 D5 TIMER INTERRUPT ROUTINE
J3
001 E2
DOCICP07 E2 AUTO TASK INITIATION 'INITIATE' 'PUT'
DOCICP08 E3 EXPIRED-CANCELLED 'INITIATE' 'PUT'
E3
DOCICP20 E5 RUNAWAY TASK FLUSH ROUTINE
001 F2
DOCICP10 F2 DATA RETRIEVAL 'GET'
DOCICP13 F5 CREATE INTERVAL CONTROL ELEMENT
001 G2
DOCICP11 G2 I/O ERROR 'RETRY'
DOCICP09 G5 TERMINAL CONTROL TABLE SEARCH
001 H2
DOCICP02 H2 TIME OF DAY 'GETIME'
DOCICP14 H5 CHAIN INTERVAL CONTROL ELEMENT
J3
J3 EXIT TO CALLER
DOCICP15 J5 UNCHAIN INTERVAL CONTROL ELEMENT
CLOSED ROUTINES

STORAGE CONTROL PROGRAM
DFHSCP
CHART 4



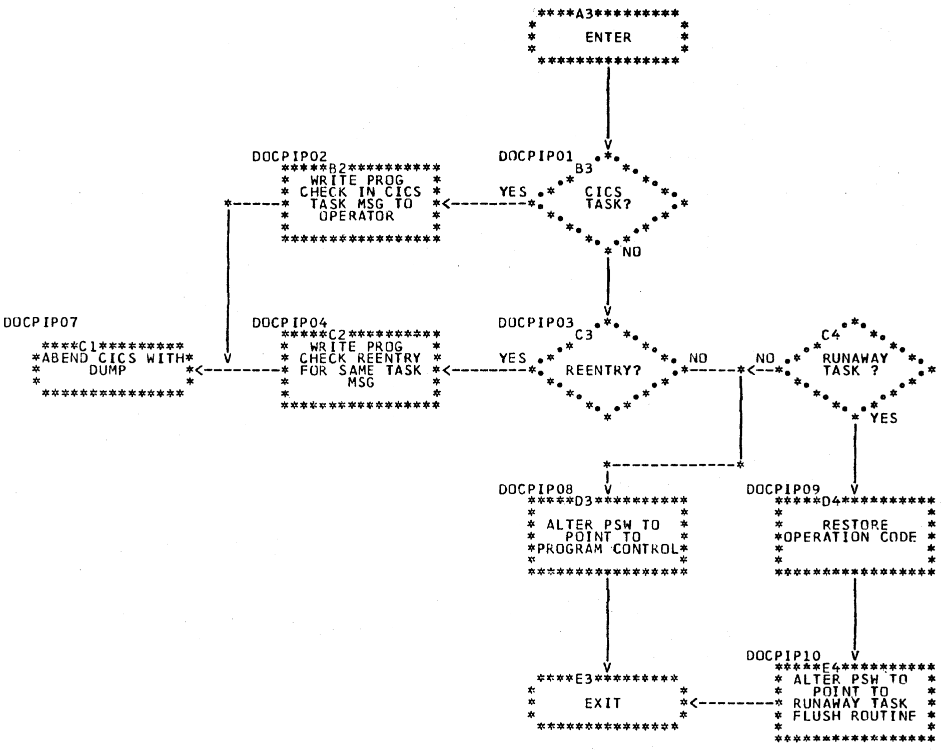
PROGRAM CONTROL PROGRAM
DFHPCP
CHART 5



PROGRAM INTERRUPT PROGRAM

DFHPIP

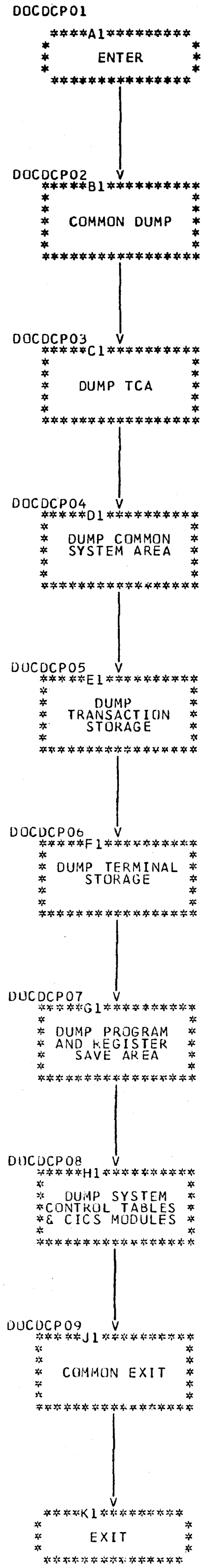
CHART 6

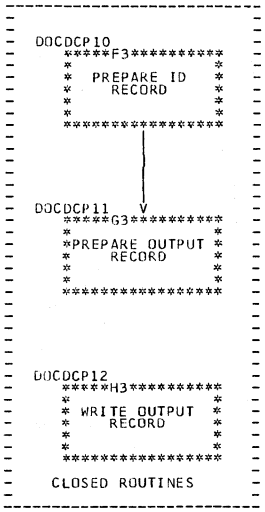


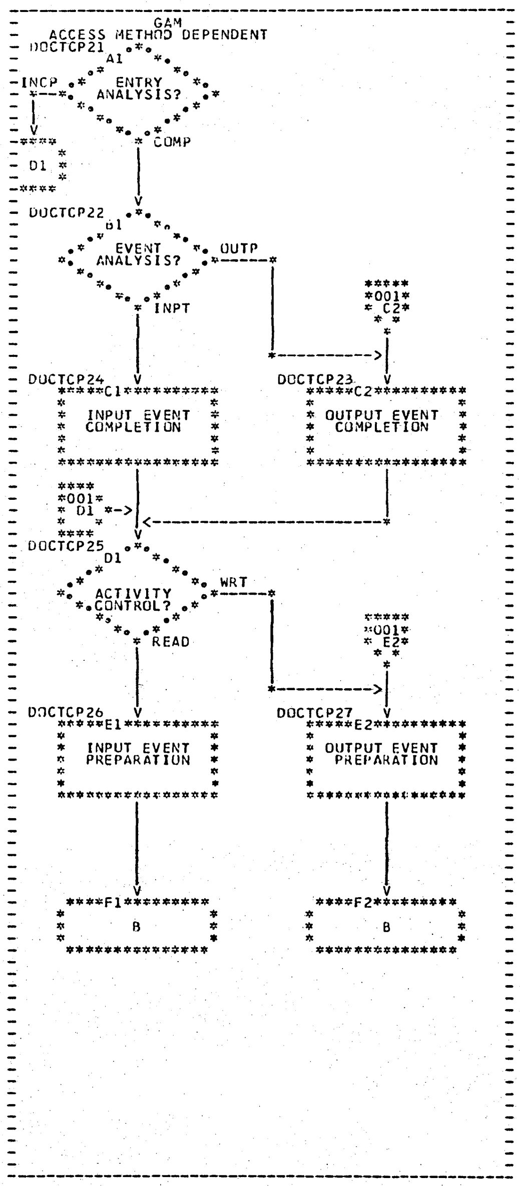
DUMP CONTROL PROGRAM

DFHDCP

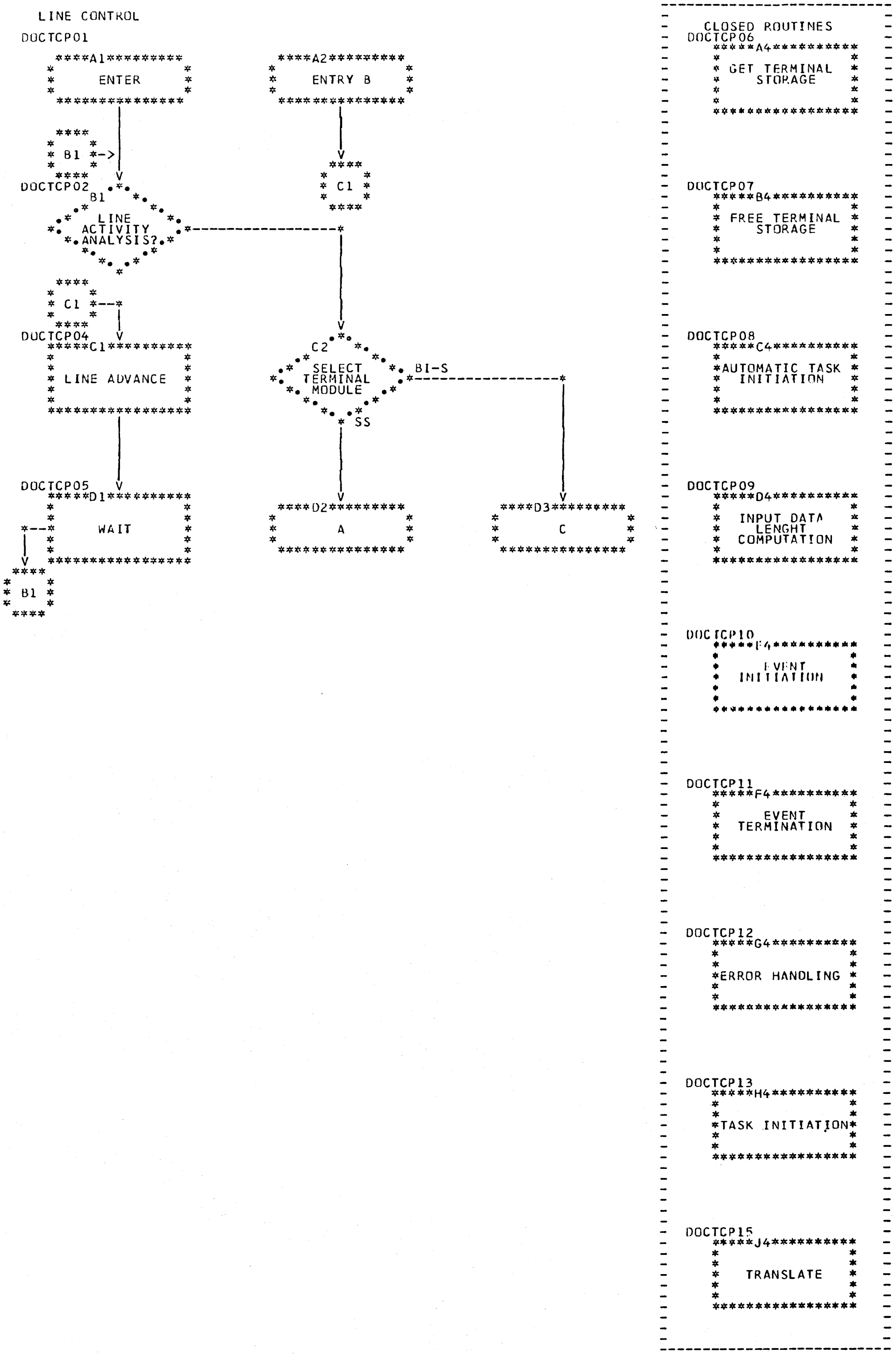
CHART 7

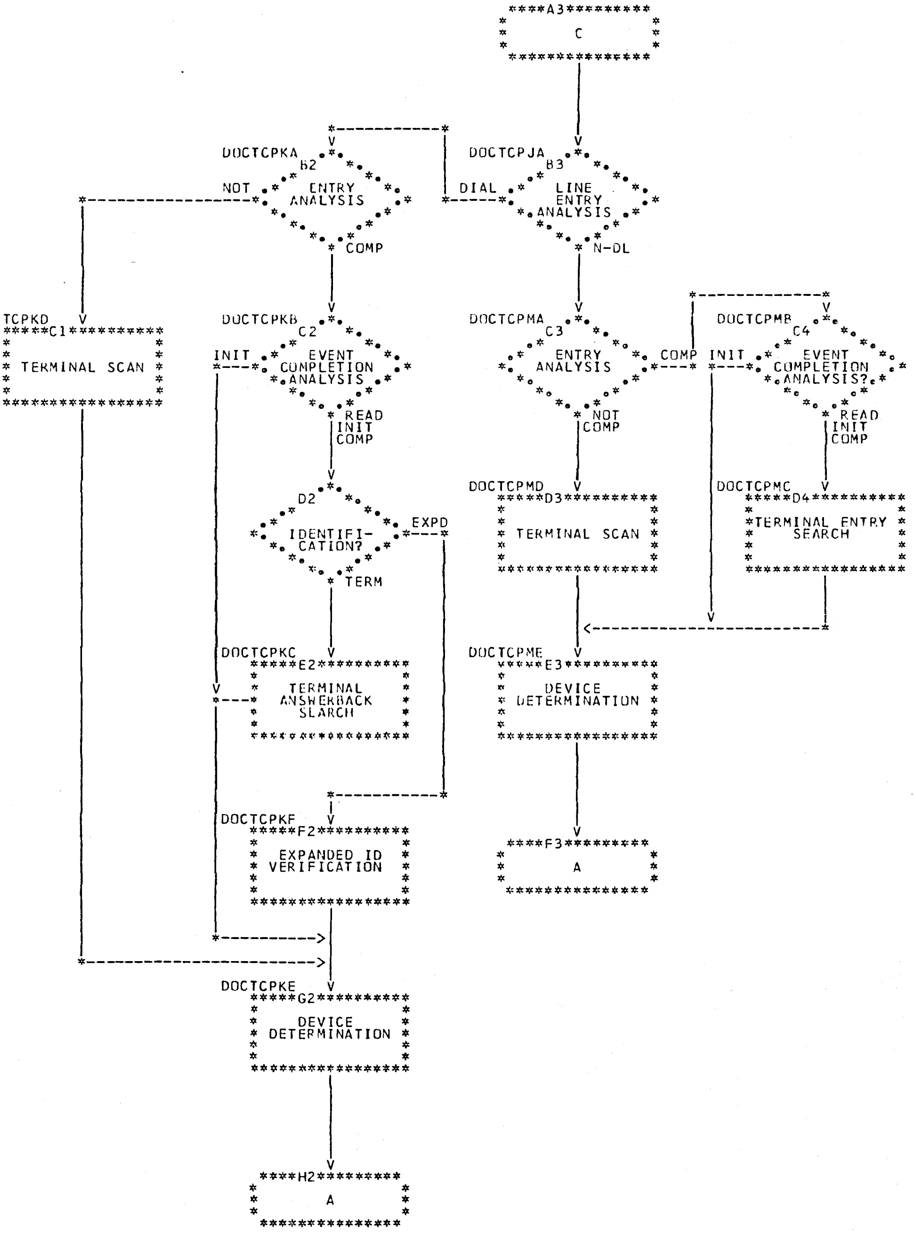





GAM
ACCESS METHOD DEPENDENT
DOCTCP21
A1
INCP
ENTRY ANALYSIS?
COMP
D1
DOCTCP22
B1
EVENT ANALYSIS?
OUTP
INPT
001 C2
DOCTCP24
C1
INPUT EVENT COMPLETION
DOCTCP23
C2
OUTPUT EVENT COMPLETION
001 D1
DOCTCP25
D1
ACTIVITY CONTROL?
WRT
READ
001 E2
DOCTCP26
E1
INPUT EVENT PREPARATION
DOCTCP27
E2
OUTPUT EVENT PREPARATION
F1
B
F2
B

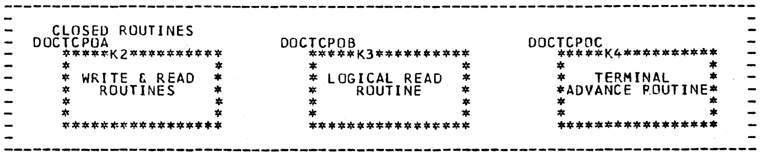
TERMINAL CONTROL PROGRAM
START/STOP COMMON ROUTINES
CHART B



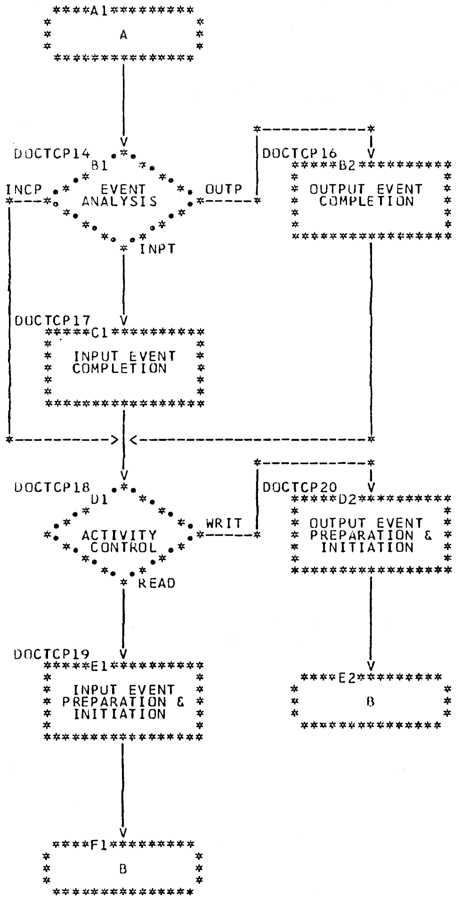


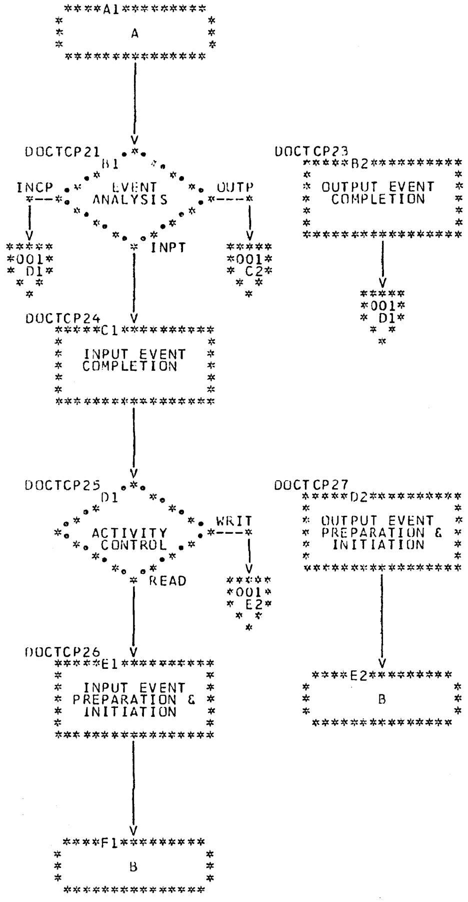
MODULES:

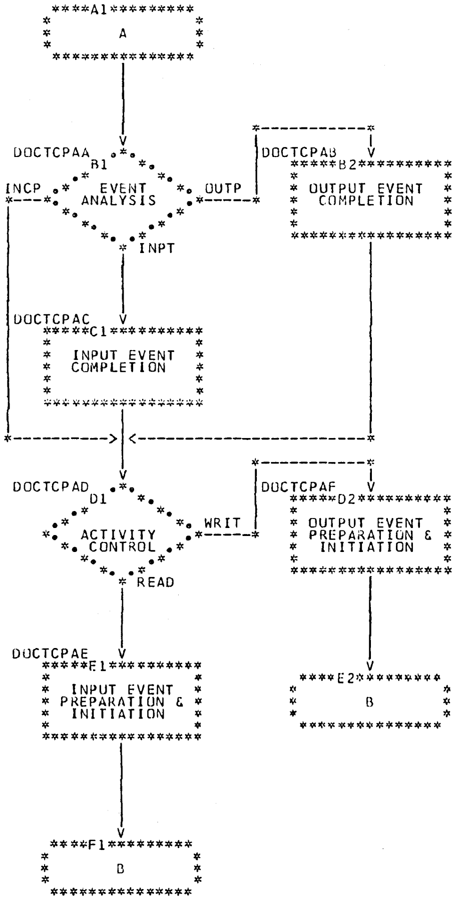
DFHTCCBS
DFHTCNBS
DFHTCSBS



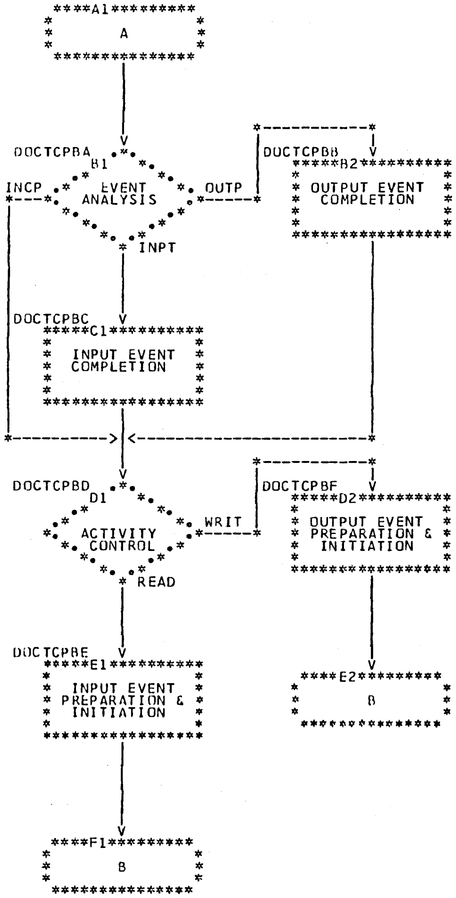
TERMINAL CONTROL PROGRAM
DFHTCSAM - SEQUENTIAL TERMINAL DEPENDENT MODULE
CHART 8



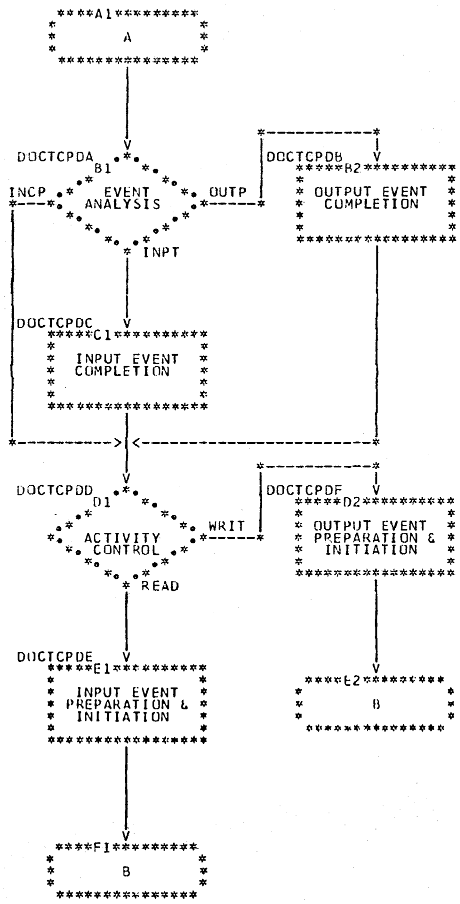

A1
A
DOCTCP21
B1
EVENT ANALYSIS
INCP
OUTP
INPT
001 D1
001 C2
DOCTCP23
B2
OUTPUT EVENT COMPLETION
001 D1
DOCTCP24
C1
INPUT EVENT COMPLETION
DOCTCP25
D1
ACTIVITY CONTROL
WRIT
READ
001 E2
DOCTCP27
D2
OUTPUT EVENT PREPARATION & INITIATION
DOCTCP26
E1
INPUT EVENT PREPARATION & INITIATION
E2
B
F1
B


A1
A
DOCTCPAA
B1
EVENT ANALYSIS
INCP
OUTP
INPT
DOCTCPAB
B2
OUTPUT EVENT COMPLETION
DOCTCPAC
C1
INPUT EVENT COMPLETION
DOCTCPAD
D1
ACTIVITY CONTROL
WRIT
READ
DOCTCPAF
D2
OUTPUT EVENT PREPARATION & INITIATION
DOCTCPAE
E1
INPUT EVENT PREPARATION & INITIATION
E2
B
F1
B

TERMINAL CONTROL PROGRAM
DFHTC50N - 1050 TERMINAL DEPENDENT MODULE
CHART 8

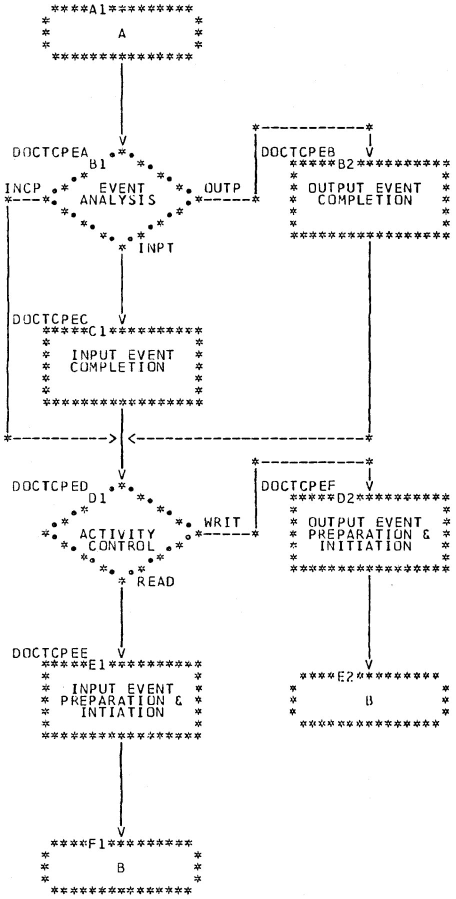


```
     ****A1*********
    *               *
    *       A       *
    *               *
     ***************
            |
            |
            |
            |             *------------*
            V             |            |
 DOCTCPCA  .*.            |DOCTCPCB    V
         B1   *.             *****B2***********
       .*        *.          *                *
INCP .*   EVENT   *. OUTP |   * OUTPUT EVENT   *
*---*.  ANALYSIS   .*-----*   *  COMPLETION    *
|     *.         .*           *                *
|       *.     .*             *                *
|         *. .*               ******************
|           * INPT                    |
|           |                          |
|           |                          |
|           |                          |
|DOCTCPCC   V                          |
|   *****C1***********                 |
|   *                *                 |
|   *  INPUT EVENT   *                 |
|   *  COMPLETION    *                 |
|   *                *                 |
|   *                *                 |
|   ******************                 |
|           |                          |
*---------->|<-------------------------*
            |             *------------*
            V             |            |
 DOCTCPCD  .*.            |DOCTCPCF    V
         D1   *.             *****D2***********
       .*        *.          *                *
     .*            *. WRIT |   * OUTPUT EVENT   *
    *.  ACTIVITY   .*-----*   * PREPARATION &  *
      *. CONTROL  .*           *  INITIATION    *
        *.     .*             *                *
          *. .*               ******************
            * READ                      |
            |                          |
            |                          |
            |                          |
 DOCTCPCE   V                          |
    *****E1***********                 V
    *                *        ****F2*********
    *  INPUT EVENT   *       *               *
    * PREPARATION &  *       *       B       *
    *  INITIATION    *       *               *
    *                *        ***************
    ******************
            |
            |
            |
            |
            V
     ****F1*********
    *               *
    *       B       *
    *               *
     ***************
```


A1
A
DOCTCPDA
B1
EVENT ANALYSIS
INCP
OUTP
INPT
DOCTCPDB
B2
OUTPUT EVENT COMPLETION
DOCTCPDC
C1
INPUT EVENT COMPLETION
DOCTCPDD
D1
ACTIVITY CONTROL
WRIT
READ
DOCTCPDF
D2
OUTPUT EVENT PREPARATION & INITIATION
DOCTCPDE
E1
INPUT EVENT PREPARATION & INITIATION
E2
B
F1
B

DFHTC41N - 2741 NON-SWITCHED TERMINAL DEPENDENT MODULE

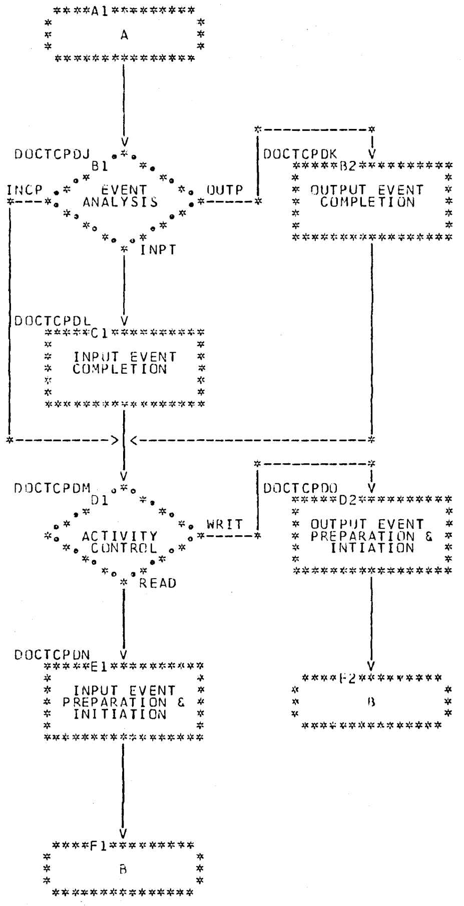
CHART 8

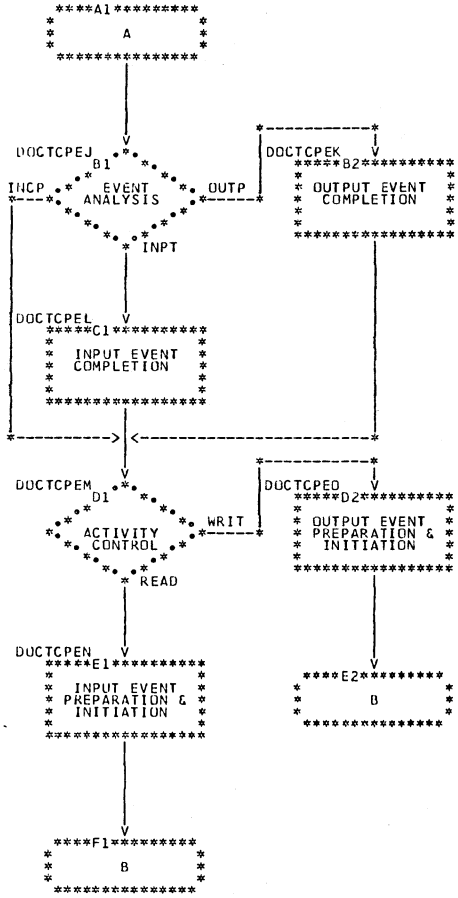


```
     ****A1*********
    *               *
    *       A       *
    *               *
     ***************
            |
            |
            |
            |
            |             *-----------*
            V             |           |
 DOCTCPBJ  .*.            |DOCTCPBK   V
         B1   *.          |   *****B2**********
       .*        *.       |   *               *
INCP .*   EVENT   *. OUTP |   * OUTPUT EVENT  *
*---*.  ANALYSIS   .*-----*   *  COMPLETION   *
|     *.         .*           *               *
|       *.     .*             *               *
|         *. .*               *****************
|           *  INPT                    |
|           |                         |
|           |                         |
|           |                         |
|           |                         |
|DOCTCPBL   V                         |
|   *****C1**********                 |
|   *               *                 |
|   *  INPUT EVENT  *                 |
|   *  COMPLETION   *                 |
|   *               *                 |
|   *               *                 |
|   *****************                 |
|           |                         |
|           |                         |
*---------->|<------------------------*
            |
            |             *-----------*
            V             |           |
 DOCTCPBM  .*.            |DOCTCPBO   V
         D1   *.          |   *****D2**********
       .*        *.       |   *               *
     .*          *. WRIT  |   * OUTPUT EVENT  *
    *.  ACTIVITY   .*-----*   * PREPARATION & *
      *. CONTROL .*           *  INITIATION   *
        *.     .*             *               *
          *. .*               *****************
            *  READ                    |
            |                         |
            |                         |
            |                         |
            |                         |
 DOCTCPBN   V                         |
    *****E1**********                 |
    *               *                 |
    *  INPUT EVENT  *                 |
    * PREPARATION & *                 |
    *  INITIATION   *                 |
    *               *                 |
    *****************                 |
            |                         |
    ****    |                         |
    *011*   |                         |
    * F1 *->|                         |
    *   *   |<------------------------*
    ****    |
            V
     ****F1*********
    *               *
    *       B       *
    *               *
     ***************
```

DFHTC40S - 2740 DIAL-UP TERMINAL DEPENDENT MODULE

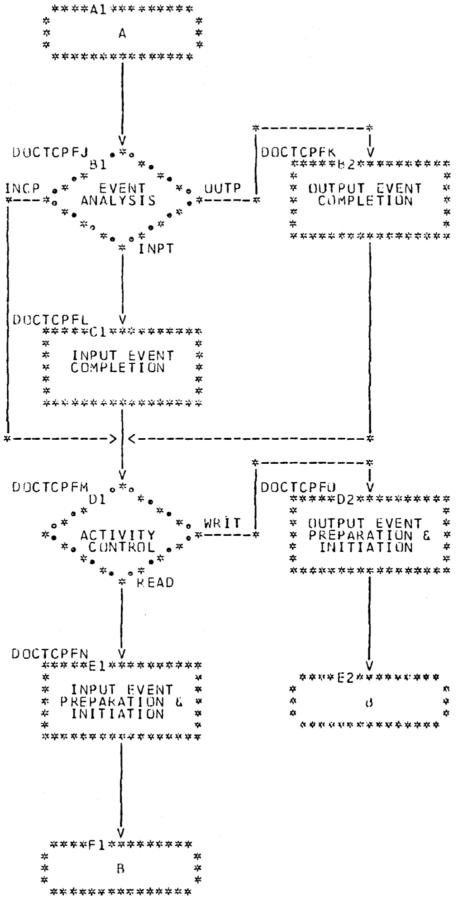
CHART 8




A1
A
DOCTCPEJ
B1
EVENT ANALYSIS
INCP
OUTP
INPT
DOCTCPEK
B2
OUTPUT EVENT COMPLETION
DOCTCPEL
C1
INPUT EVENT COMPLETION
DOCTCPEM
D1
ACTIVITY CONTROL
WRIT
READ
DOCTCPEO
D2
OUTPUT EVENT PREPARATION & INITIATION
DOCTCPEN
E1
INPUT EVENT PREPARATION & INITIATION
E2
B
F1
B

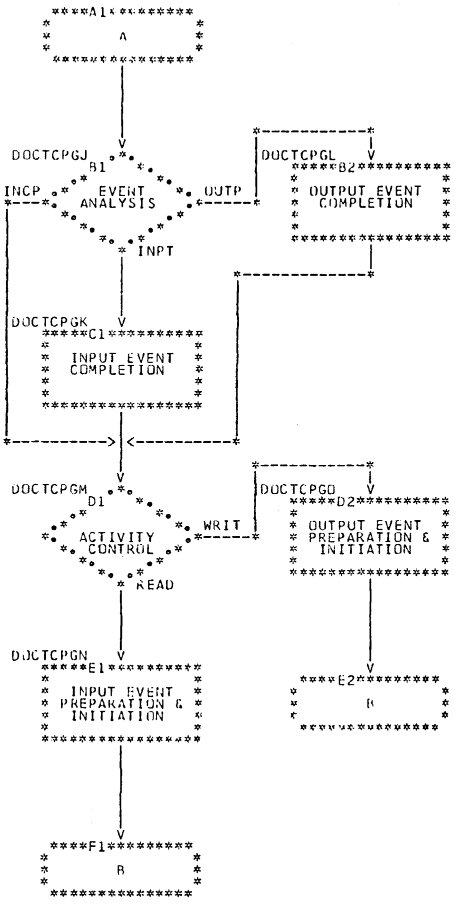
DFHTCTWX - TWX TERMINAL DEPENDENT MODULE

CHART 8



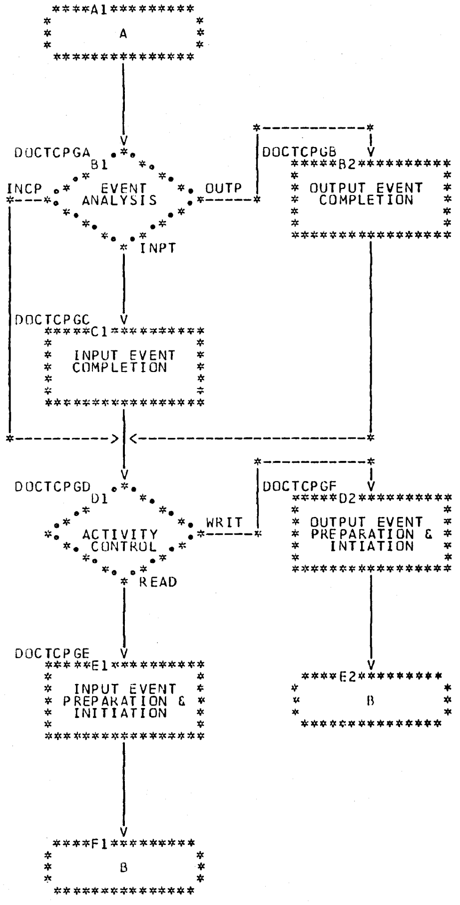
DFHTCS7S - SYSTEM/7 DIAL-UP TERMINAL DEPENDENT MODULE

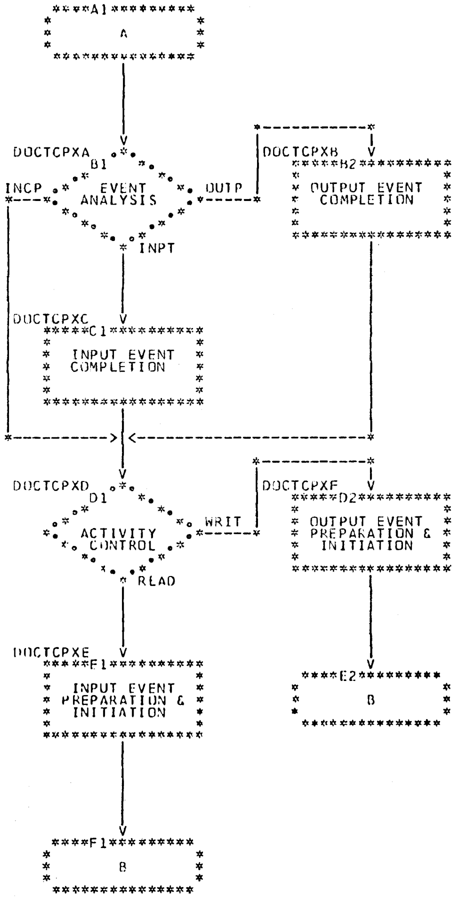
CHART 8



DFHTCS7N - SYSTEM/7 NON-SWITCHED TERMINAL DEPENDENT MODULE

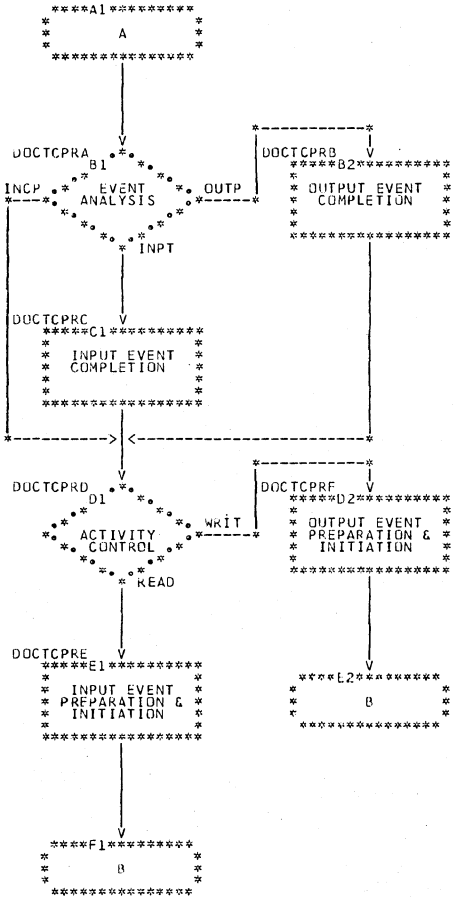
CHART 8

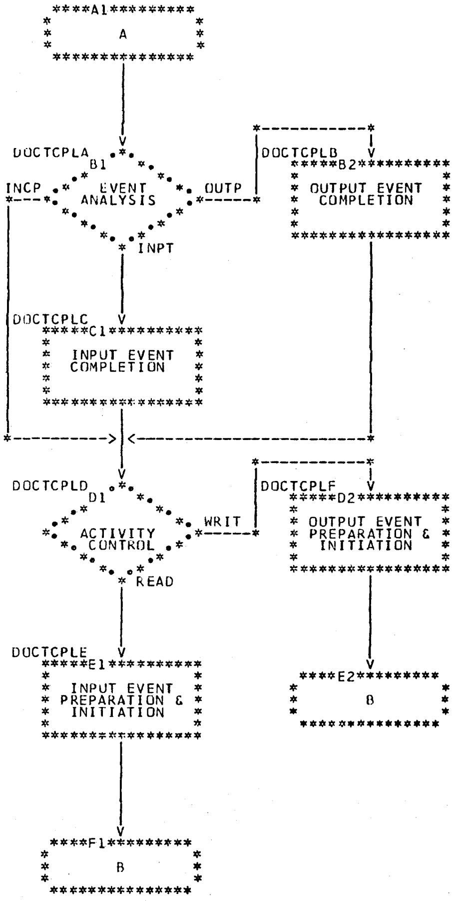


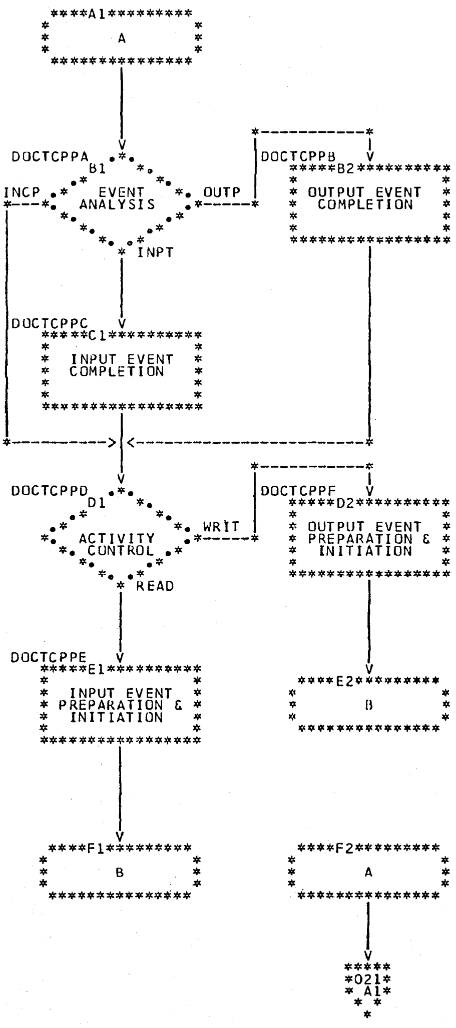

A1
A
DOCTCPXA
B1
EVENT ANALYSIS
INCP
OUTP
INPT
DOCTCPXB
B2
OUTPUT EVENT COMPLETION
DOCTCPXC
C1
INPUT EVENT COMPLETION
DOCTCPXD
D1
ACTIVITY CONTROL
WRIT
READ
DOCTCPXF
D2
OUTPUT EVENT PREPARATION & INITIATION
DOCTCPXE
F1
INPUT EVENT PREPARATION & INITIATION
E2
B
F1
B

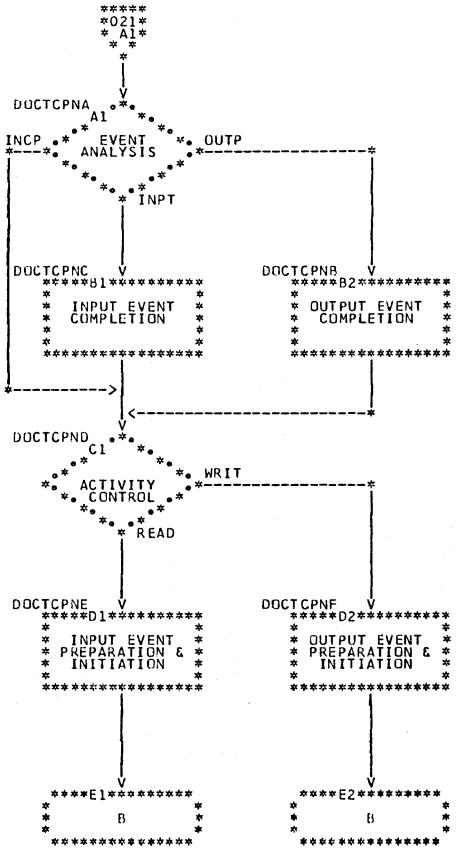
DFHTCS70 - 2770 & PROGRAMMABLE TERMINAL DIAL-UP MODULE

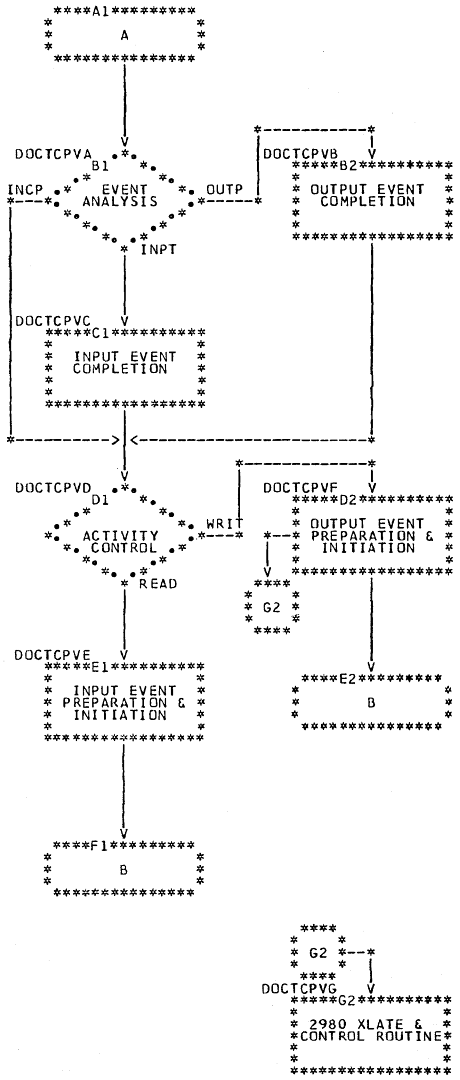
CHART 8




A1
A
DOCTCPLA
B1
EVENT ANALYSIS
INCP
OUTP
INPT
DOCTCPLB
B2
OUTPUT EVENT COMPLETION
DOCTCPLC
C1
INPUT EVENT COMPLETION
DOCTCPLD
D1
ACTIVITY CONTROL
WRIT
READ
DOCTCPLF
D2
OUTPUT EVENT PREPARATION & INITIATION
DOCTCPLE
E1
INPUT EVENT PREPARATION & INITIATION
E2
B
F1
B

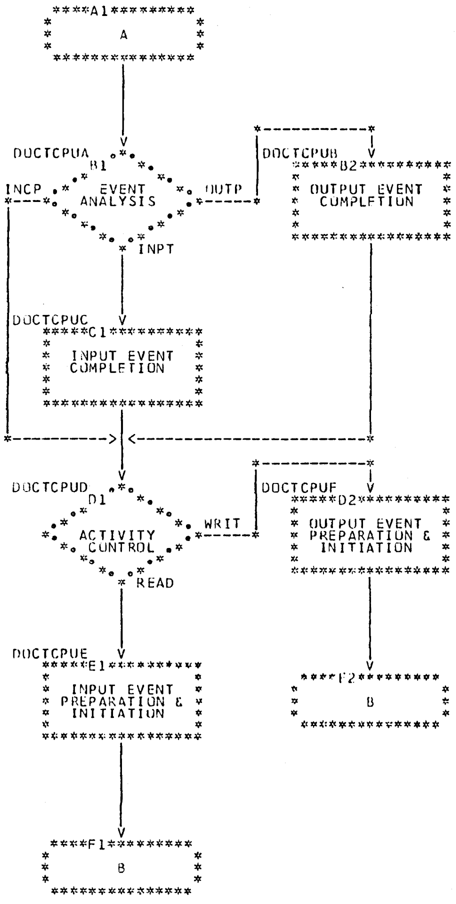

A1
A
DOCTCPPA
B1
EVENT ANALYSIS
INCP
OUTP
INPT
DOCTCPPB
B2
OUTPUT EVENT COMPLETION
DOCTCPPC
C1
INPUT EVENT COMPLETION
DOCTCPPD
D1
ACTIVITY CONTROL
WRIT
READ
DOCTCPPF
D2
OUTPUT EVENT PREPARATION & INITIATION
DOCTCPPE
E1
INPUT EVENT PREPARATION & INITIATION
E2
B
F1
B
F2
A
021
A1


021
A1
DOCTCPNA
A1
EVENT ANALYSIS
INCP
OUTP
INPT
DOCTCPNC
B1
INPUT EVENT COMPLETION
DOCTCPNB
B2
OUTPUT EVENT COMPLETION
DOCTCPND
C1
ACTIVITY CONTROL
WRIT
READ
DOCTCPNE
D1
INPUT EVENT PREPARATION & INITIATION
DOCTCPNF
D2
OUTPUT EVENT PREPARATION & INITIATION
E1
B
E2
B


A1
A
DOCTCPVA
B1
EVENT ANALYSIS
INCP
OUTP
INPT
DOCTCPVB
B2
OUTPUT EVENT COMPLETION
DOCTCPVC
C1
INPUT EVENT COMPLETION
DOCTCPVD
D1
ACTIVITY CONTROL
WRIT
READ
DOCTCPVF
D2
OUTPUT EVENT PREPARATION & INITIATION
G2
DOCTCPVE
E1
INPUT EVENT PREPARATION & INITIATION
E2
B
F1
B
G2
DOCTCPVG
G2
2980 XLATE & CONTROL ROUTINE

DFHTCS35 - 3735 DIAL-UP TERMINAL DEPENDENT MODULE

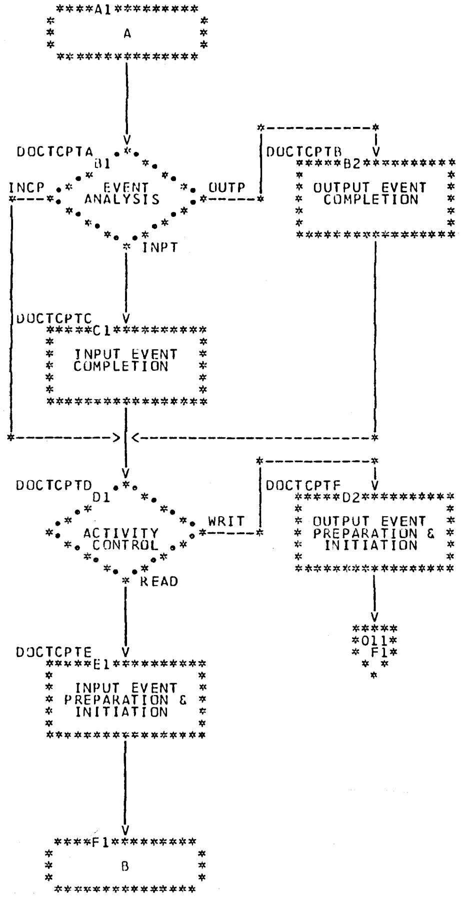
CHART 8



```
        ****A1**********
        *              *
        *       A      *
        *              *
        ****************
               |
               |
               |
               |                 *-----------*
               V                 |           |
  DOCTCPSA    .*.                |DOCTCPSB   V
            B1   *.              |  *****B2**********
           .*       *.           |  *               *
 INCP    .*   EVENT   *.  OUTP   |  * OUTPUT EVENT  *
 *---*.     ANALYSIS   .*--------*  *  COMPLETION   *
 |      *.           .*             *               *
 |        *.       .*               *               *
 |          *.   .*                 *****************
 |             *  INPT                      |
 |             |                            |
 |             |                            |
 |             V                            |
 |DOCTCPSC     V                            |
 |    *****C1**********                     |
 |    *               *                     |
 |    *  INPUT EVENT  *                     |
 |    *  COMPLETION   *                     |
 |    *               *                     |
 |    *               *                     |
 |    *****************                     |
 |             |                            |
 *------------>|<---------------------------*
               |                 *-----------*
               V                 |           |
  DOCTCPSD    .*.                |DOCTCPSF   V
            D1   *.              |  *****D2**********
           .*       *.           |  *               *
         .*           *.  WRIT   |  * OUTPUT EVENT  *
       *.    ACTIVITY   .*-------*  * PREPARATION & *
         *.  CONTROL  .*            *   INITIATION  *
           *.       .*              *               *
             *.   .*                *****************
               *  READ                      |
               |                            V
               |                          *****
               |                          *011*
  DOCTCPSE     V                          * F1*
      *****E1**********                    * *
      *               *                     *
      *  INPUT EVENT  *
      * PREPARATION & *
      *   INITIATION  *
      *               *
      *****************
               |
               |
               |
               V
        ****F1**********
        *              *
        *       B      *
        *              *
        ****************
```

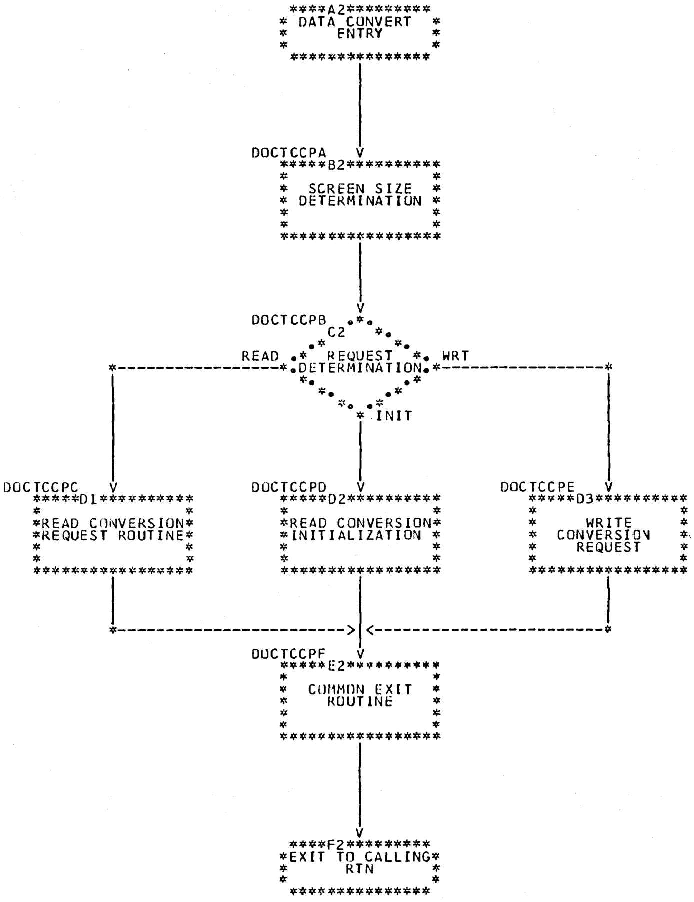
DFHTC70R - REMOTE 3270 TERMINAL DEPENDENT MODULE

CHART 8

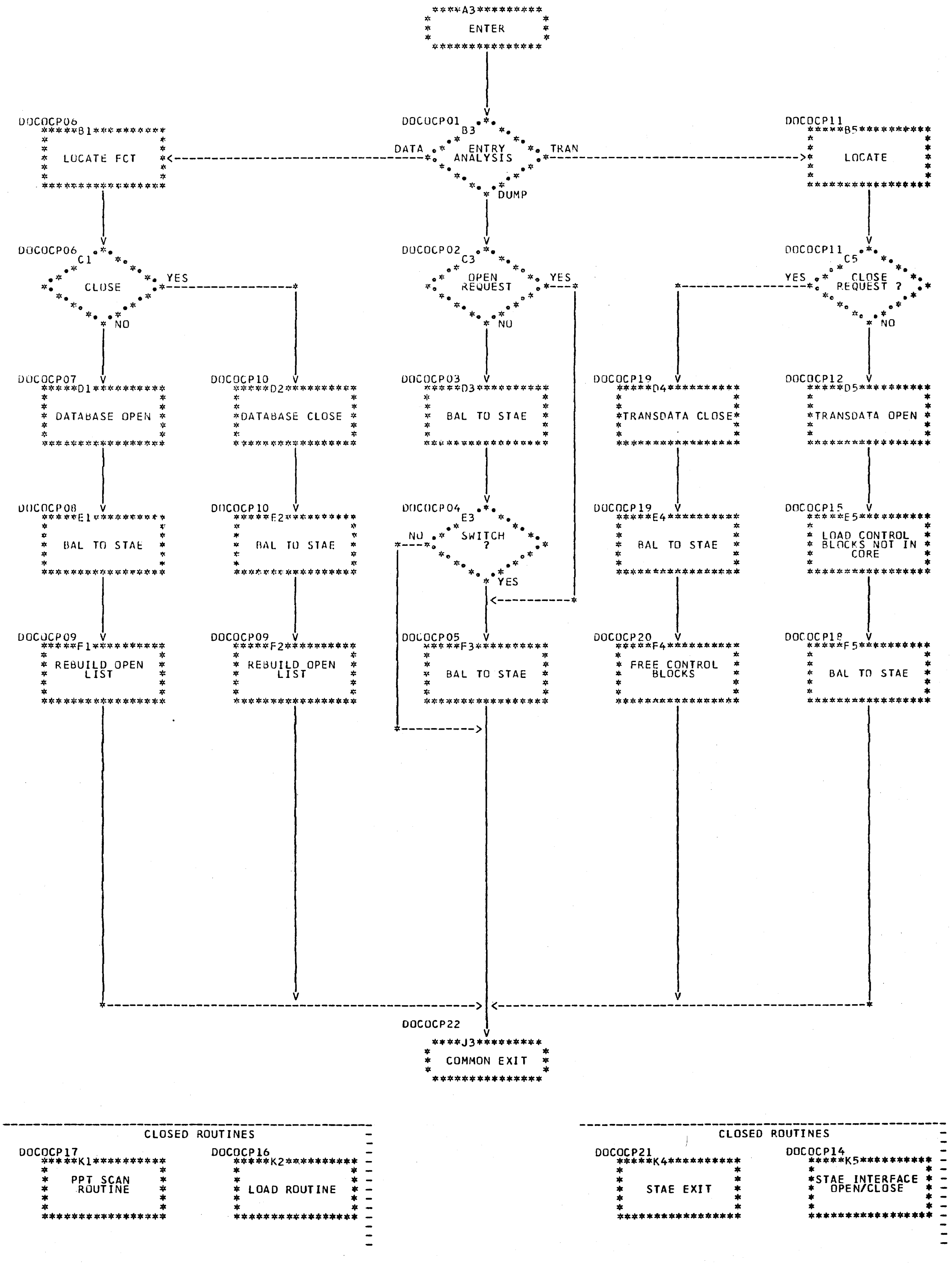


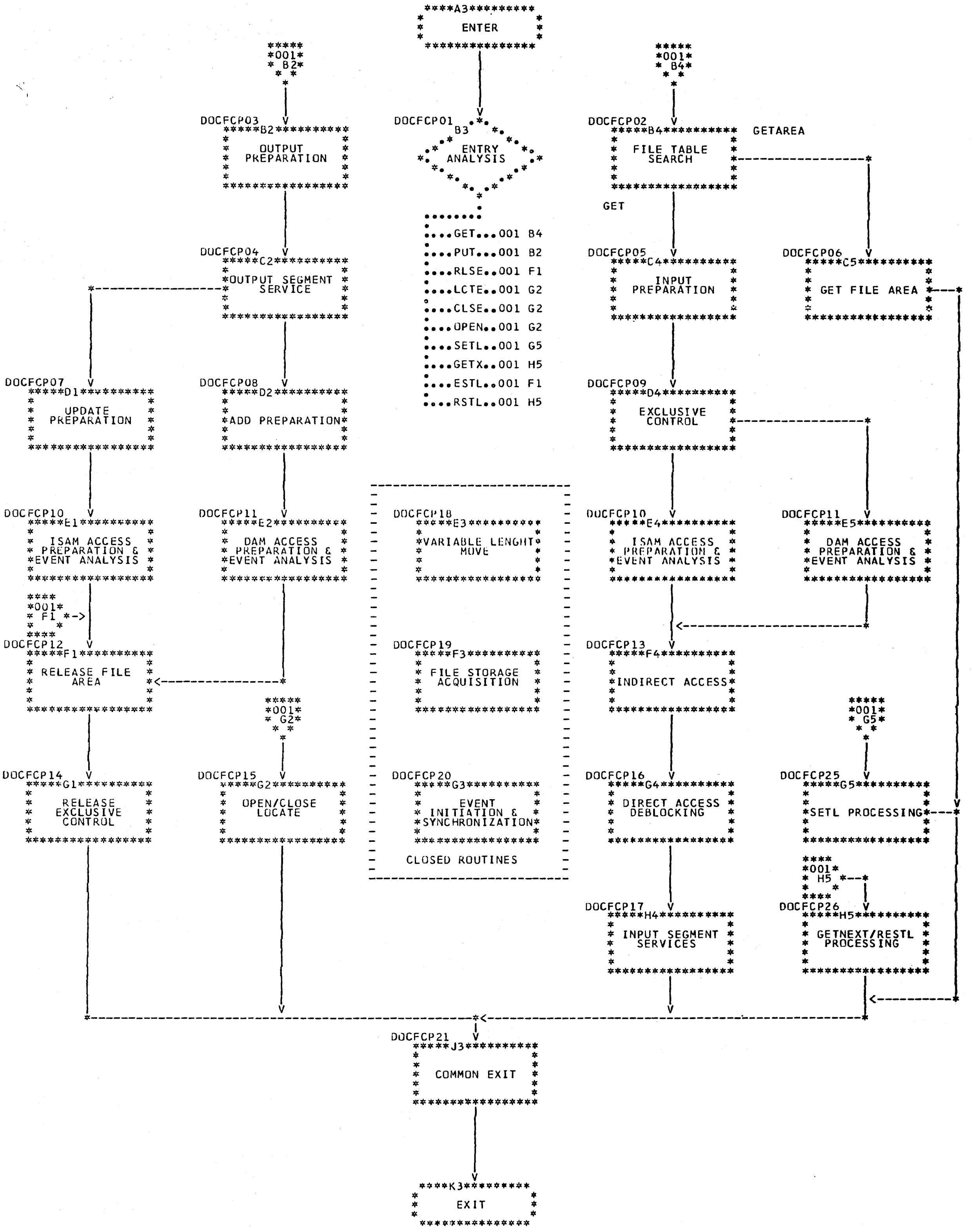
DFHTCCP - 2260 COMPATIBILITY TERMINAL DEPENDENT MODULE

CHART 8

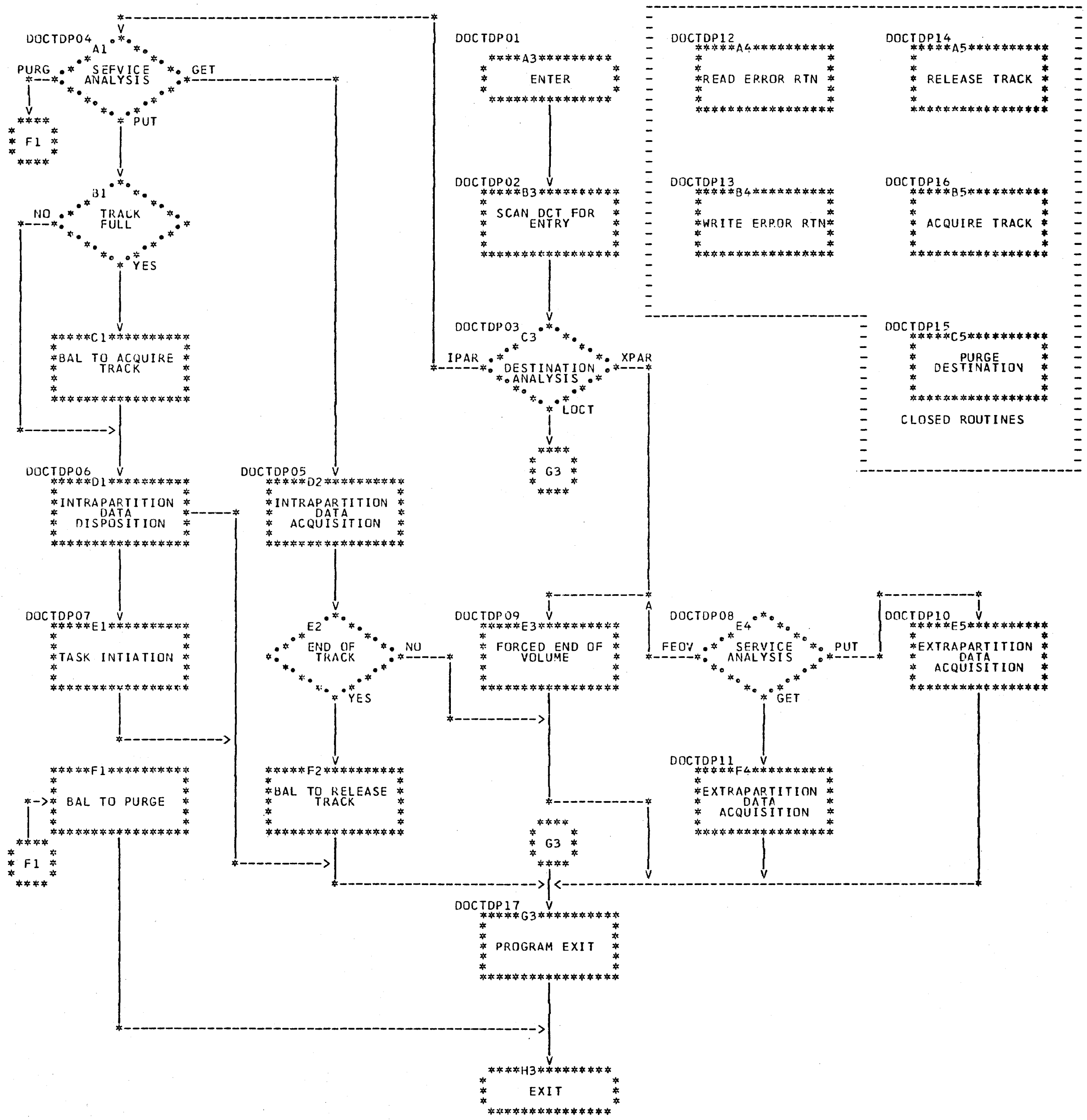


DYNAMIC OPEN/CLOSE PROGRAM
DFHOCP
CHART 9

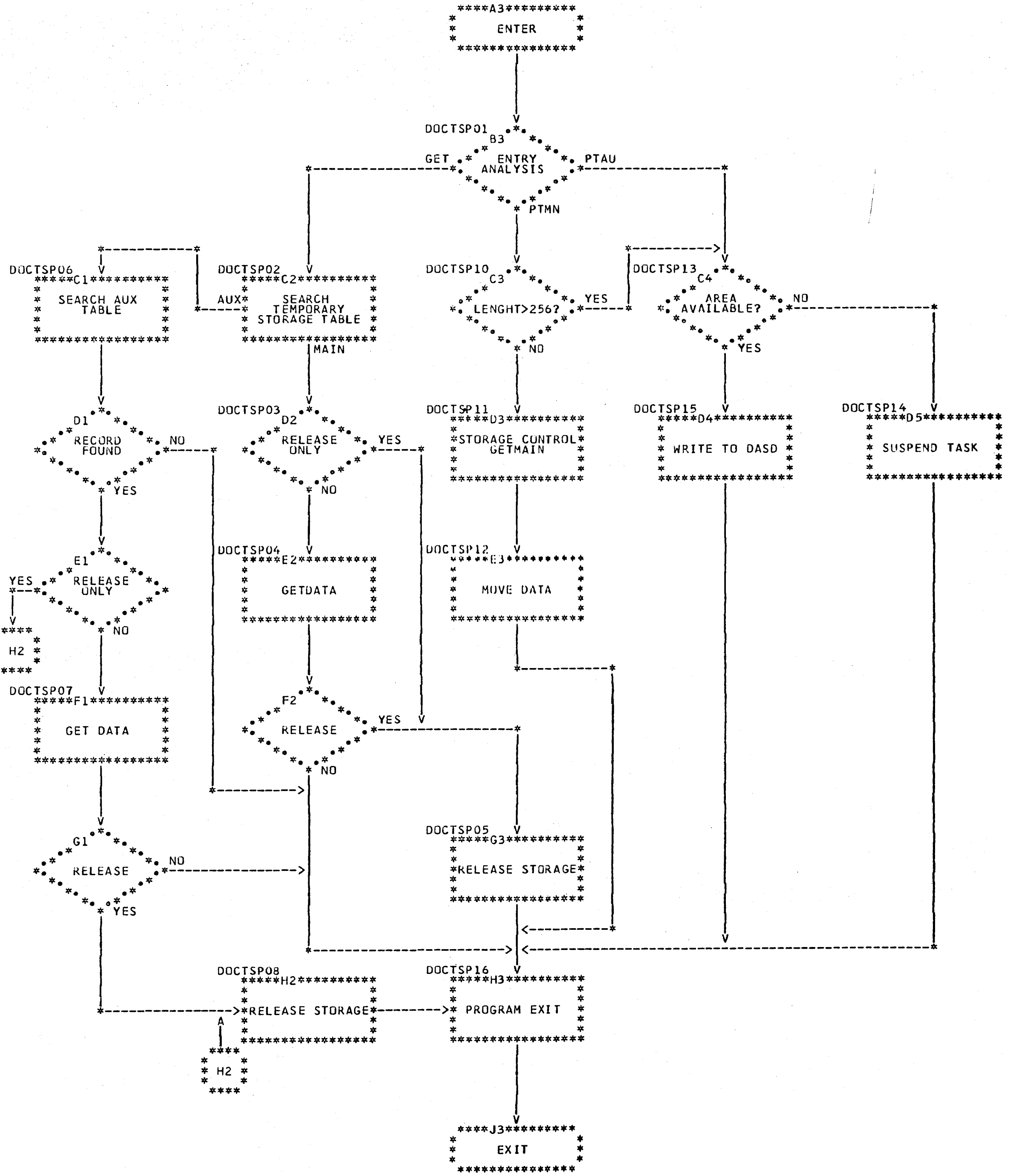



A3 ENTER
001 B2
001 B4
DOCFCP03 B2 OUTPUT PREPARATION
DOCFCP01 B3 ENTRY ANALYSIS
DOCFCP02 B4 FILE TABLE SEARCH
GETAREA
GET
....GET...001 B4
....PUT...001 B2
....RLSE..001 F1
....LCTE..001 G2
....CLSE..001 G2
....OPEN..001 G2
....SETL..001 G5
....GETX..001 H5
....ESTL..001 F1
....RSTL..001 H5
DOCFCP04 C2 OUTPUT SEGMENT SERVICE
DOCFCP05 C4 INPUT PREPARATION
DOCFCP06 C5 GET FILE AREA
DOCFCP07 D1 UPDATE PREPARATION
DOCFCP08 D2 ADD PREPARATION
DOCFCP09 D4 EXCLUSIVE CONTROL
DOCFCP10 E1 ISAM ACCESS PREPARATION & EVENT ANALYSIS
DOCFCP11 E2 DAM ACCESS PREPARATION & EVENT ANALYSIS
DOCFCP18 E3 VARIABLE LENGHT MOVE
DOCFCP10 E4 ISAM ACCESS PREPARATION & EVENT ANALYSIS
DOCFCP11 E5 DAM ACCESS PREPARATION & EVENT ANALYSIS
001 F1
DOCFCP12 F1 RELEASE FILE AREA
DOCFCP19 F3 FILE STORAGE ACQUISITION
DOCFCP13 F4 INDIRECT ACCESS
001 G2
001 G5
DOCFCP14 G1 RELEASE EXCLUSIVE CONTROL
DOCFCP15 G2 OPEN/CLOSE LOCATE
DOCFCP20 G3 EVENT INITIATION & SYNCHRONIZATION
DOCFCP16 G4 DIRECT ACCESS DEBLOCKING
DOCFCP25 G5 SETL PROCESSING
CLOSED ROUTINES
001 H5
DOCFCP17 H4 INPUT SEGMENT SERVICES
DOCFCP26 H5 GETNEXT/RESTL PROCESSING
DOCFCP21 J3 COMMON EXIT
K3 EXIT

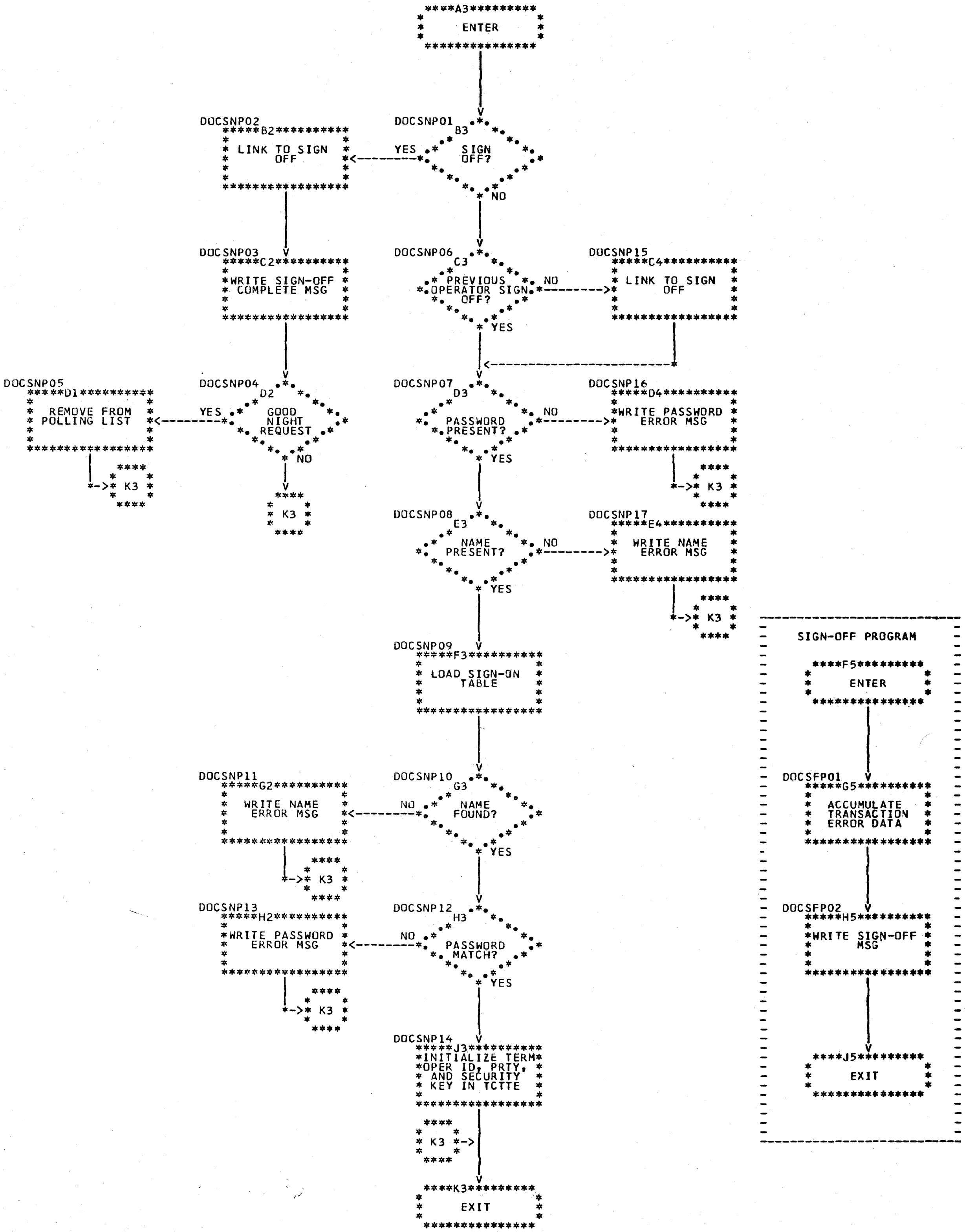
TRANSIENT DATA CONTROL PROGRAM
DFHTDP
CHART 11



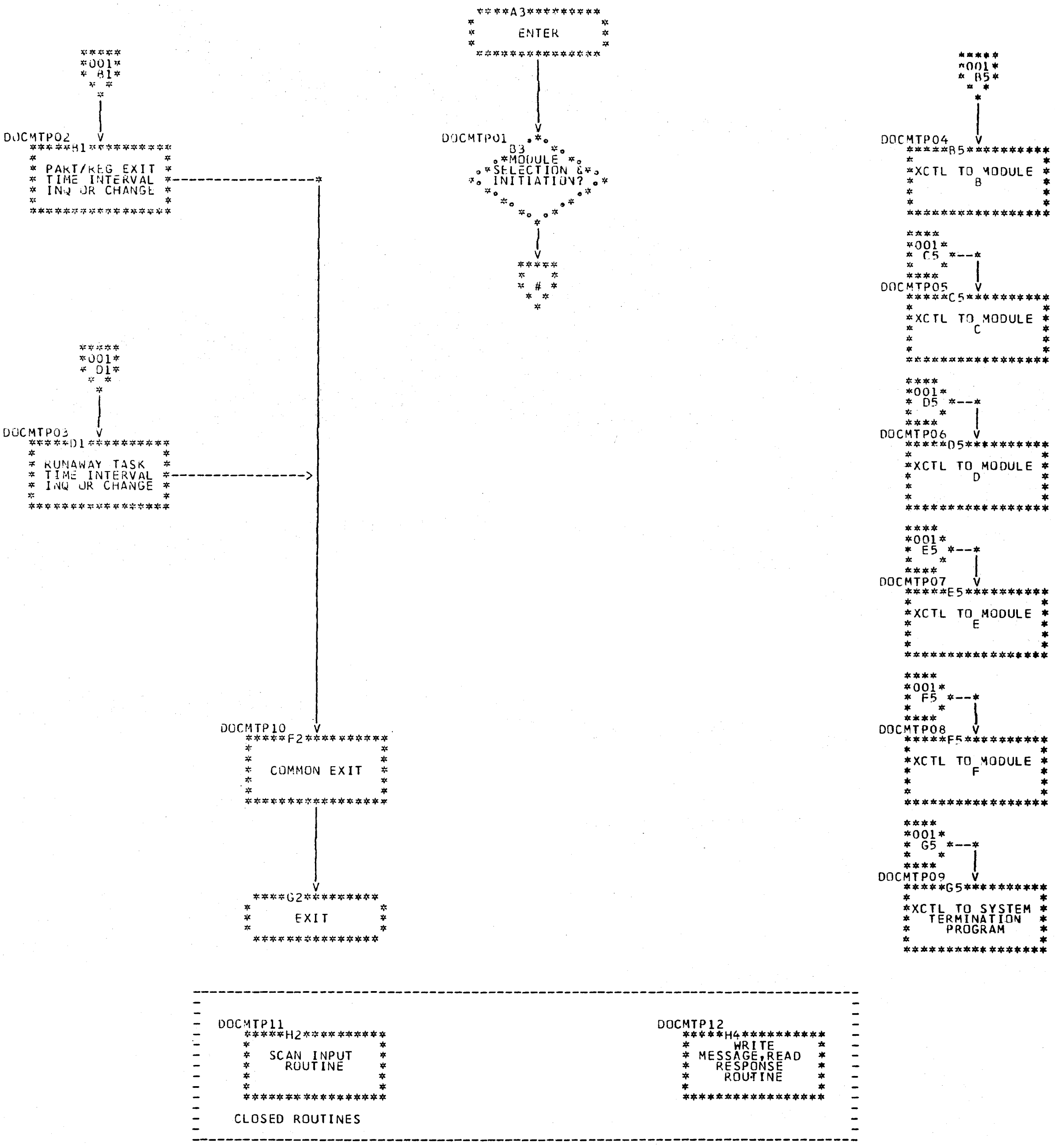
TEMPORARY STORAGE CONTROL PROGRAM
DFHTSP
CHART 12



SIGN-ON/SIGN-OFF PROGRAM
DFHSNP/DFHSFP
CHART 13



MASTER TERMINAL PROGRAM MODULE A
DFHMTPA
CHART 14

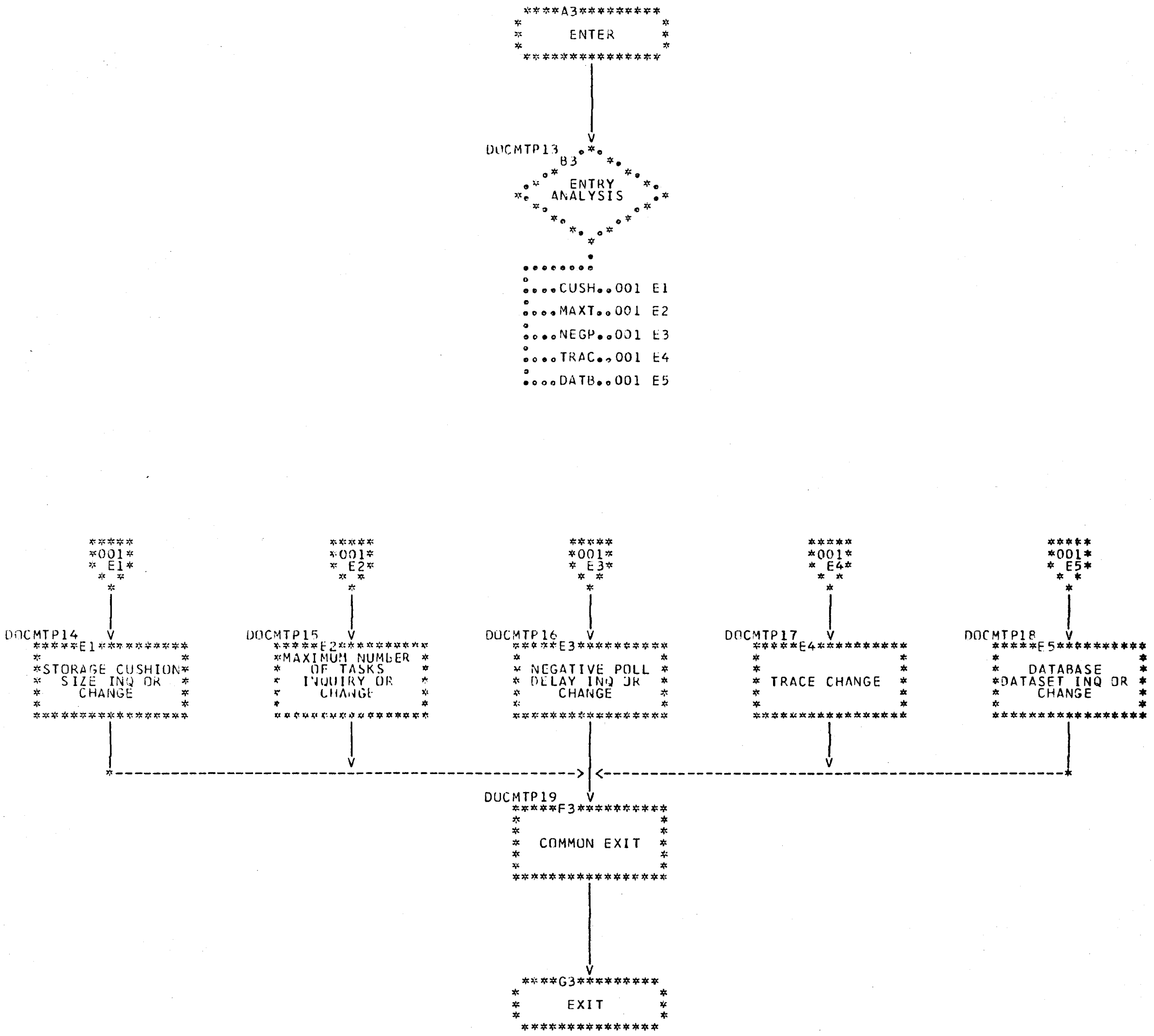


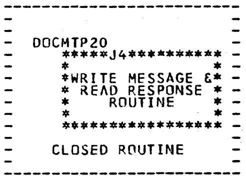
```
B3. ...TIME..001 B1
    ...RNAW..001 D1
    ...CUSH..001 B5
    ...MAXT..001 B5
    ...NEGP..001 B5
    ...TRAC..001 B5
    ...DATB..001 B5
    ...TERM..001 C5
    ...OPEN..001 D5
    ...CLOS..001 D5
    ...SWIT..001 D5
    ...TRIG..001 E5
    ...PGRM..001 E5
    ...STAL..001 E5
    ...LINE..001 F5
    ...CNTL..001 F5
    ...TRMN..001 F5
    ...SHUT..001 G5
```

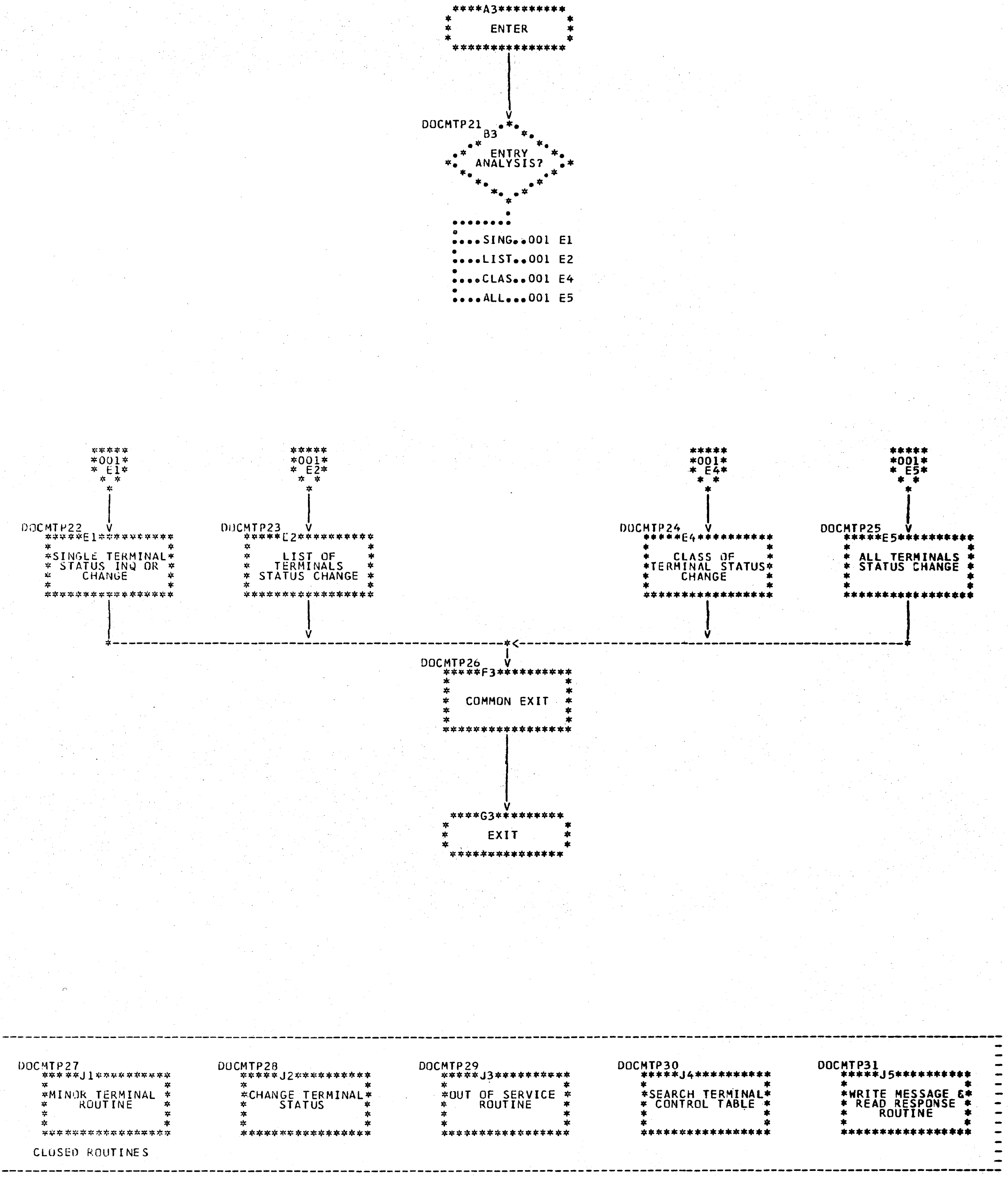
MASTER TERMINAL PROGRAM MODULE B

DFHMTPB

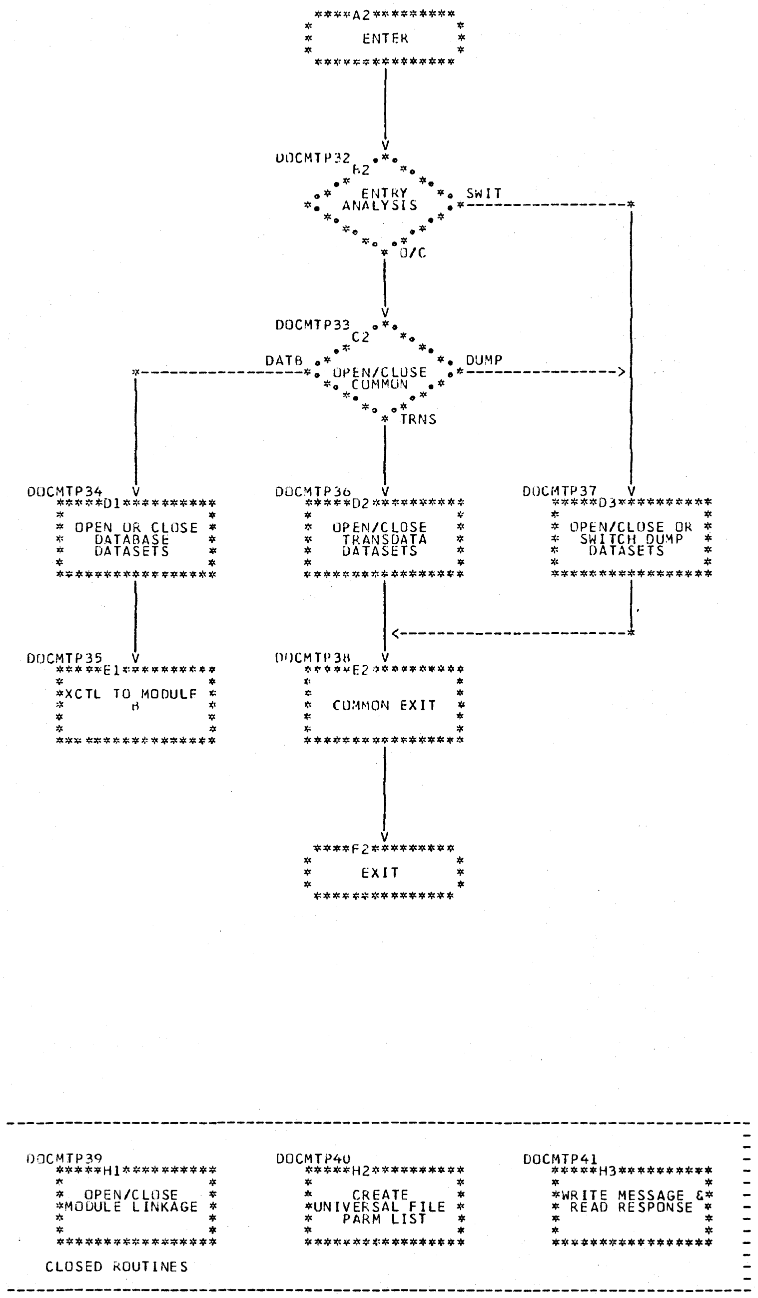
CHART 15

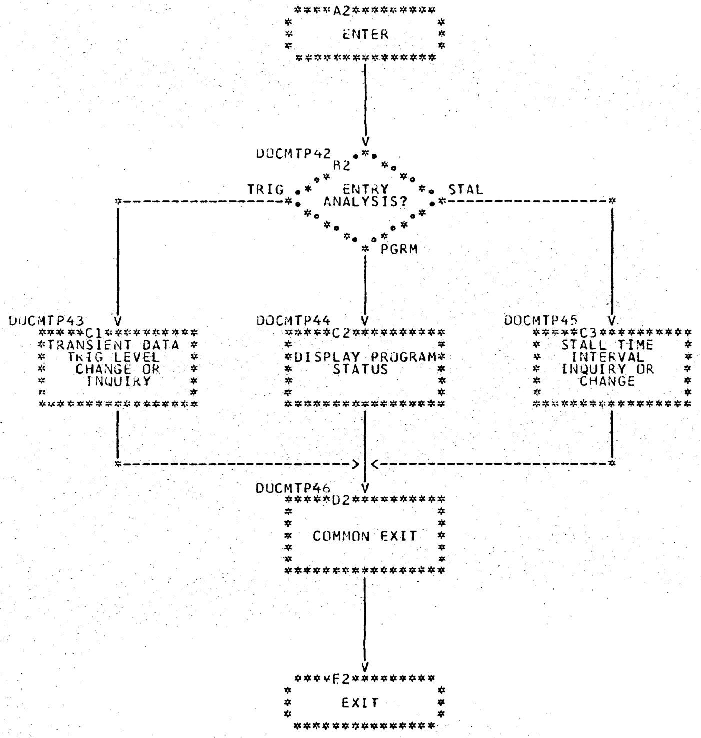


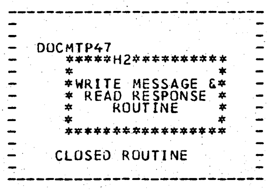



A3
ENTER
DOCMTP21
B3
ENTRY ANALYSIS?
SING..001 E1
LIST..001 E2
CLAS..001 E4
ALL...001 E5
001 E1
001 E2
001 E4
001 E5
DOCMTP22
E1
SINGLE TERMINAL STATUS INQ OR CHANGE
DOCMTP23
E2
LIST OF TERMINALS STATUS CHANGE
DOCMTP24
E4
CLASS OF TERMINAL STATUS CHANGE
DOCMTP25
E5
ALL TERMINALS STATUS CHANGE
DOCMTP26
F3
COMMON EXIT
G3
EXIT
DOCMTP27
J1
MINOR TERMINAL ROUTINE
DOCMTP28
J2
CHANGE TERMINAL STATUS
DOCMTP29
J3
OUT OF SERVICE ROUTINE
DOCMTP30
J4
SEARCH TERMINAL CONTROL TABLE
DOCMTP31
J5
WRITE MESSAGE & READ RESPONSE ROUTINE
CLOSED ROUTINES

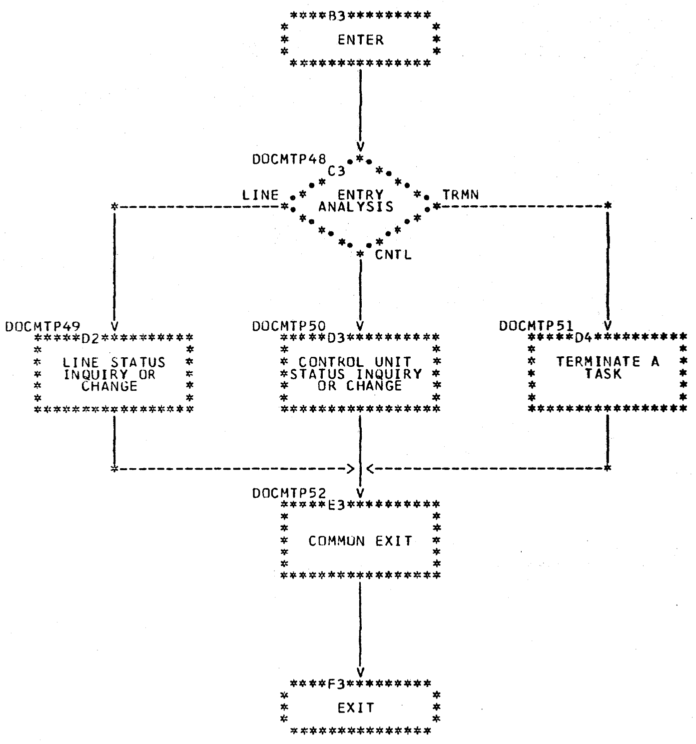
MASTER TERMINAL PROGRAM MODULE D
DFHMTPD
CHART 17

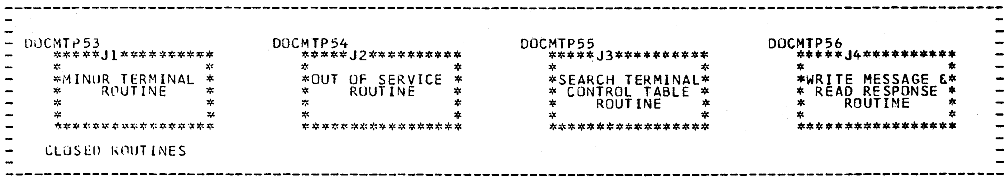



A2
ENTER
DOCMTP42
B2
ENTRY ANALYSIS?
TRIG
STAL
PGRM
DOCMTP43
C1
TRANSIENT DATA TRIG LEVEL CHANGE OR INQUIRY
DOCMTP44
C2
DISPLAY PROGRAM STATUS
DOCMTP45
C3
STALL TIME INTERVAL INQUIRY OR CHANGE
DOCMTP46
D2
COMMON EXIT
E2
EXIT


DOCMTP47
H2
WRITE MESSAGE & READ RESPONSE ROUTINE
CLOSED ROUTINE

MASTER TERMINAL PROGRAM MODULE F
DFHMTPF
CHART 19

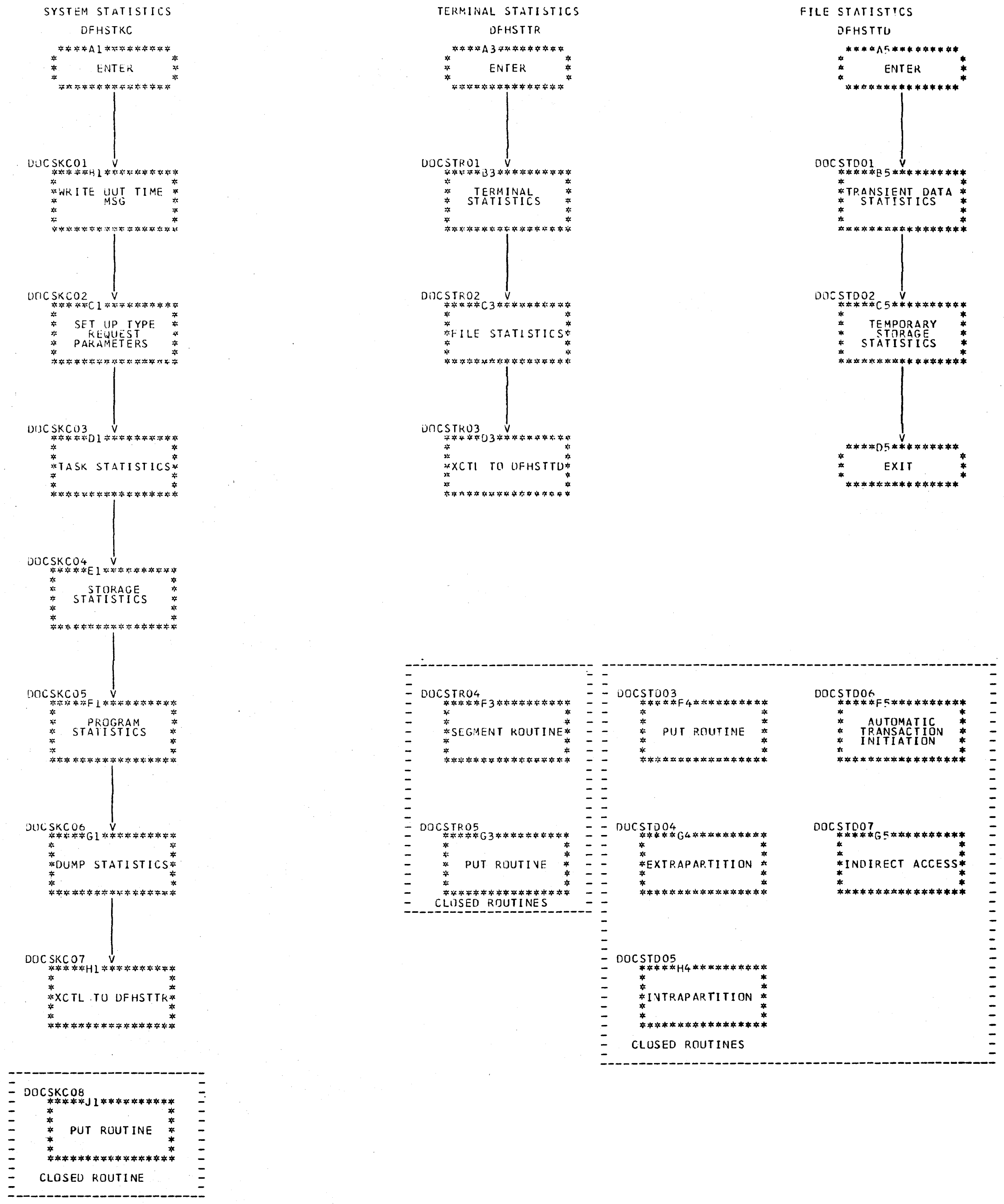




SYSTEM STATISTICS PROGRAM - TERMINAL STATISTICS PROGRAM - FILE STATISTICS PROGRAM

DFHSTKC - DFHSTTR - DFHSTTD

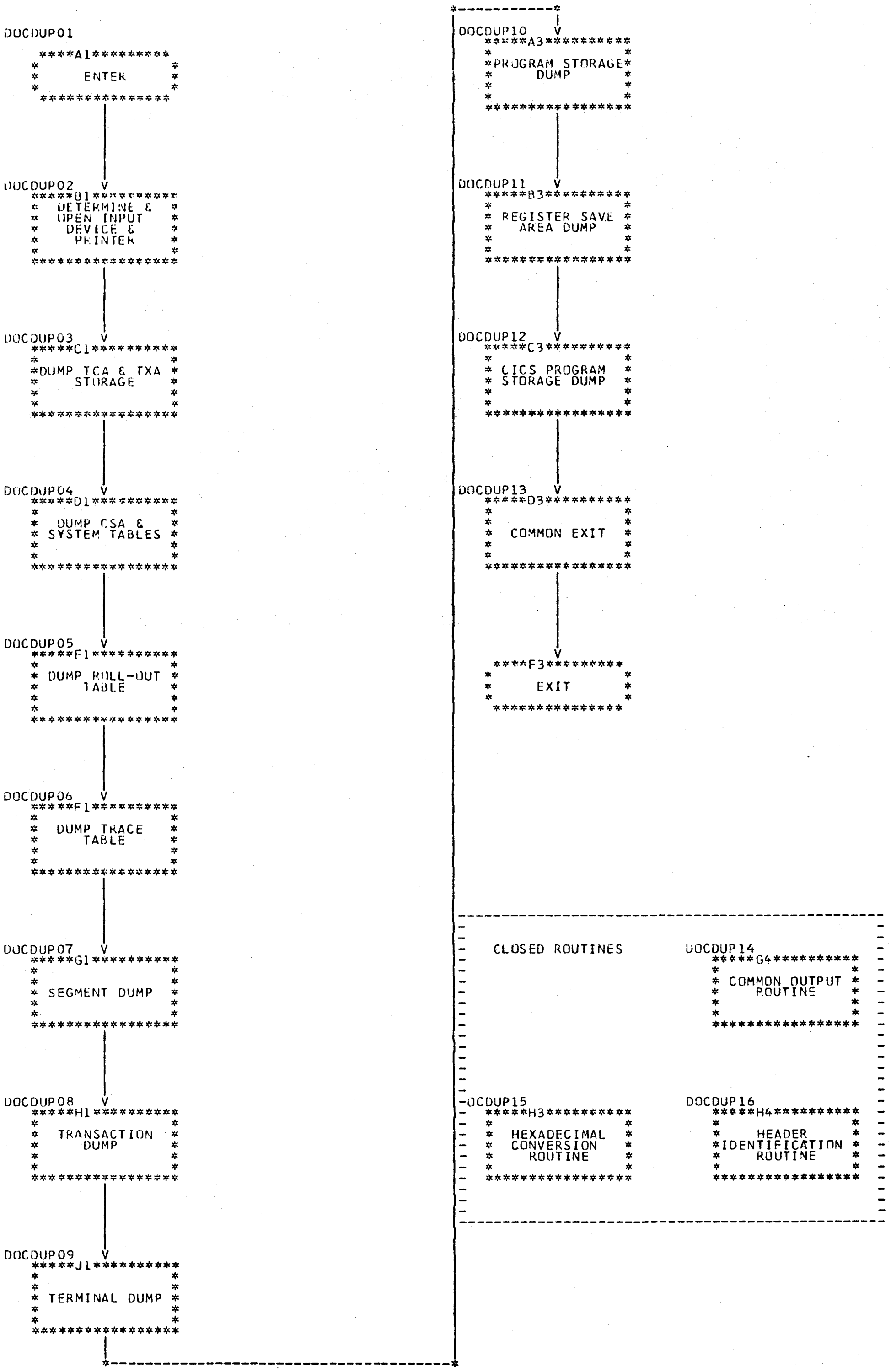
CHART 20



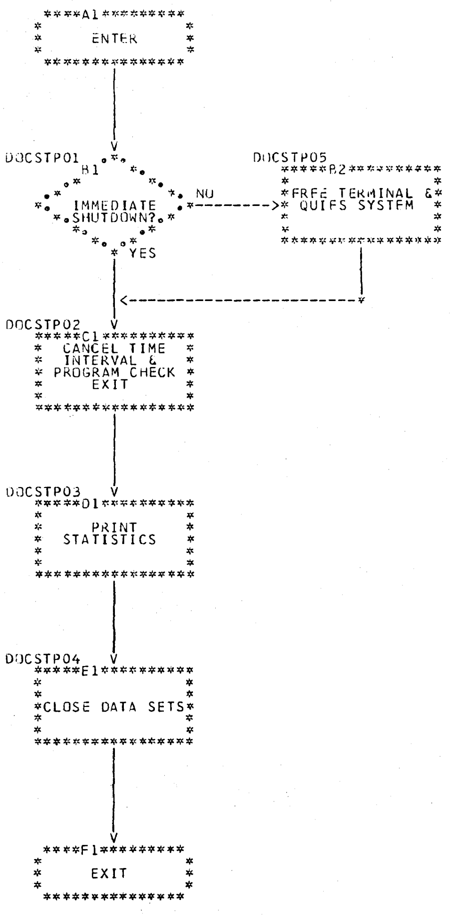
```
     *****A1**********
    *                 *
    *      ENTER      *
    *                 *
     *****************
             |
             |
             |
             |
             V
DOCACP01    .*.                   DOCACP09    .*.                DOCACP13
        B1     *.                         B2     *.                *****B3**********
        .*       *.                      .*        *.              *                 *
     .*    ENTRY    *.  ABND          .*             *.  YES       *WRITE ABEND MSG*
    *.   ANALYSIS   .*--------->*.    TERMINAL      .*---------->*  TO OPERATOR   *
      *.          .*                 *.FACILITY?.*                 *                 *
        *.      .*                     *.      .*                  *                 *
          *.  .*                         *.  .*                    *****************
            *  ERR                         *  NO                           |
            |                              |                               |
            |                              |                               |
            |                              *------------------------------>|
DOCACP04    V                                                DOCACP14      V
    *****C1**********                                          *****C3**********
    *                 *                                        *                 *
    *WRITE ERROR MSG*                                          *WRITE TO MASTER*
    *  TO OPERATOR    *                                        *    TERMINAL     *
    *                 *                                        *                 *
    *                 *                                        *                 *
    *****************                                          *****************
            |                                                          |     ****
            |                                                          |    *    *
            |                                                          *->* F1  *
            |                                                               *    *
            |                                                                ****
DOCACP05    V
    *****D1**********
    *                 *
    *WRITE ERROR MSG*
    *   TO MASTER     *
    *   TERMINAL      *
    *                 *
    *****************
            |
            |
            |
            |
            V
DOCACP07   .*.
       E1     *.
       .*       *.
    .*   WRITE    *.  YES
   *.  DISCONNECT .*--------------------*
     *.         .*                      |
       *.     .*                        |
         *. .*                          |
          *  NO                         |
   ****   |                             |
  *    *  |                             |
  * F1 *->|                             |
  *    *  |                             |
   ****   |                             |
          |                  DOCACP11   V
          V                     *****F2**********
   ****F1**********             *                 *
  *                *            *      WRITE      *
  *      EXIT      *<--*        *DISCONNECT PUT *
  *                *   |        *OUT OF SERVICE *
   ****************    |        *                 *
                       |        *****************
                      ****              |     ****
                     *    *             |    *    *
                     * F1 *             *->* F1  *
                     *    *                  *    *
                      ****                    ****



                         ----------------------------------------------------------------
                         -                                                              -
                         -  DOCACP15                                                    -
                         -     *****H3**********                                        -
                         -     *                 *                                      -
                         -     * SWITCHED LINE   *                                      -
                         -     *    ROUTINE      *                                      -
                         -     *                 *                                      -
                         -     *                 *                                      -
                         -     *****************                                        -
                         -                                                              -
                         -                                                              -
                         -                                                              -
                         -                                                              -
                         -  DOCACP16                                                    -
                         -     *****J3**********                                        -
                         -     *                 *                                      -
                         -     *    WRITE TO     *                                      -
                         -     *    TERMINAL     *                                      -
                         -     *    OPERATOR     *                                      -
                         -     *                 *                                      -
                         -     *****************                                        -
                         -   CLOSED ROUTINE                                             -
                         -                                                              -
                         ----------------------------------------------------------------
```

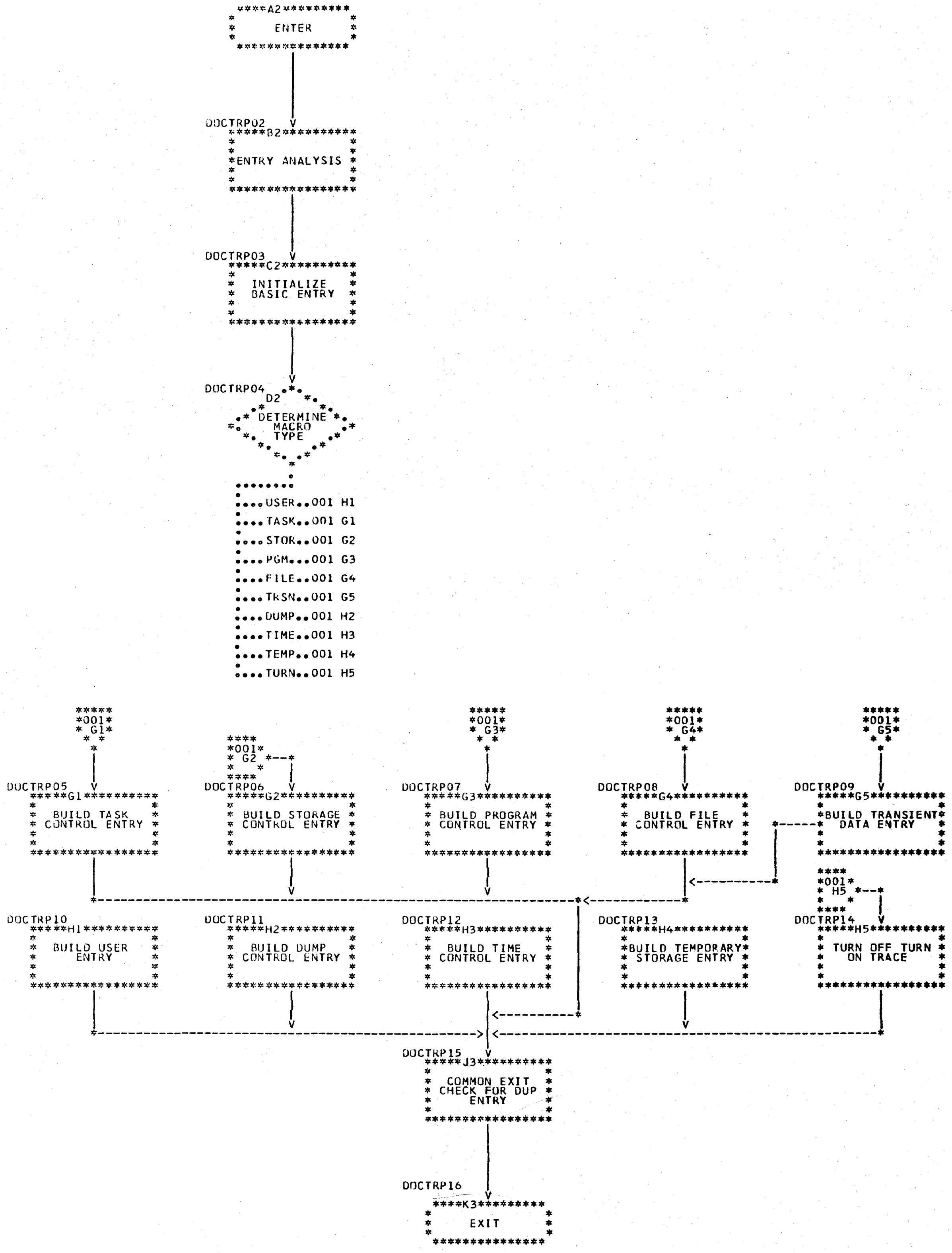
```
                                  TERMINAL TEST PROGRAM
                                         DFHFEP
                                        CHART  22



                           *****A2**********
                           *               *
                           *     ENTER     *
                           *               *
                           *****************
                                   |
                                   |
                                   |
                                   |
                         DOCFEP01  V
                           *****B2**********
                           *               *
                           *               *
                           *WRITE ENTRY MSG*
                           *               *
                           *               *
                           *****************
                                   |
                                   |
          *----------------------->|<-----------------------*
          |                        V                        |
DOCFEP02  |              DOCFEP03 .*.            DOCFEP04   |
 *****C1**********              .*  C2 *.            *****C3**********
 *               *            .*         *.          *               *
 *   WRITE ALL   *     PRNT .*    READ     *.        *   WRITE SAME  *
 *   PRINTABLE   *<--------*.  RESPONSE?  .*-------->*CHARACTERS SENT*
 *  CHARACTERS   *            *.         .*          *               *
 *               *              *.     .*            *               *
 *****************                *. .*              *****************
                                    *  END
                                    |
                                    |
                                    |
                                    V
                           *****D2**********
                           *               *
                           *     EXIT      *
                           *               *
                           *****************
```

DUMP UTILITY PROGRAM
DFHDUP
CHART 23

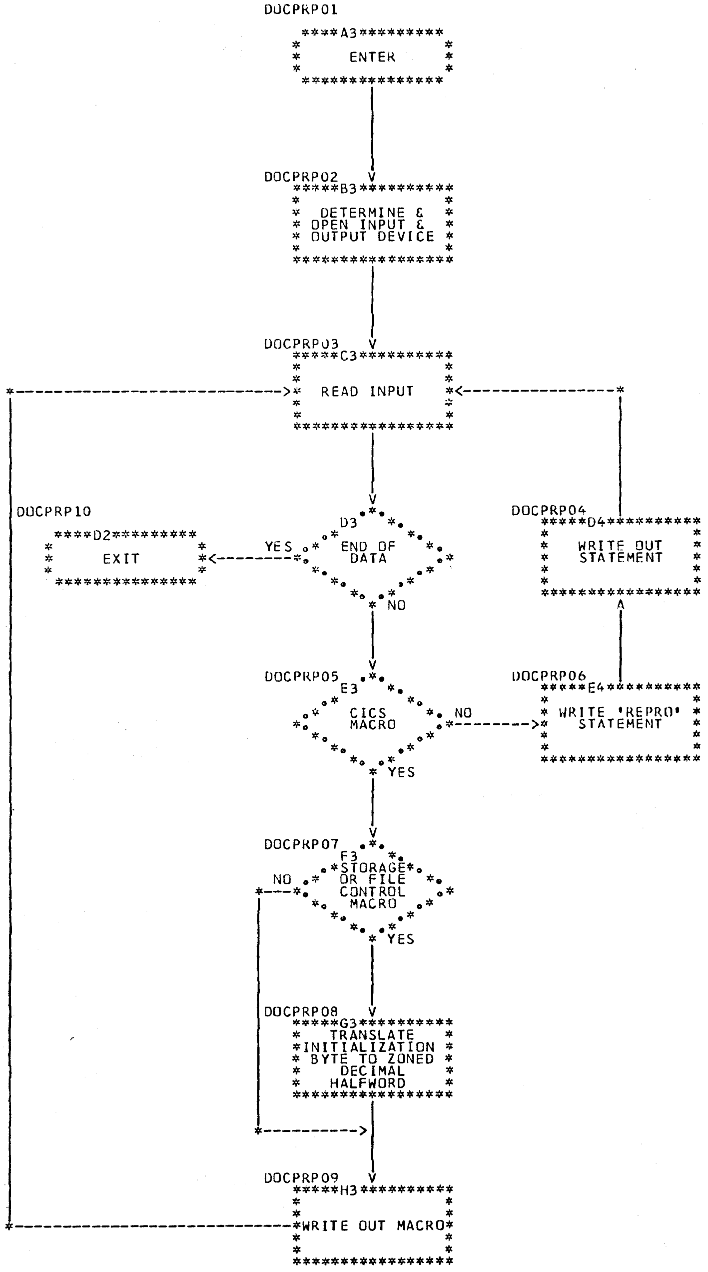


SYSTEM TERMINATION PROGRAM
DFHSTP
CHART 24

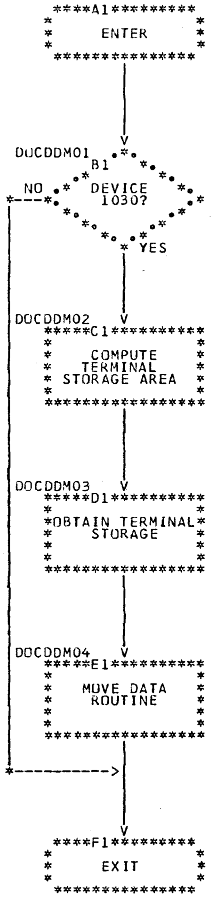



A2
ENTER
DDCTRP02
B2
ENTRY ANALYSIS
DDCTRP03
C2
INITIALIZE BASIC ENTRY
DDCTRP04
D2
DETERMINE MACRO TYPE
USER..001 H1
TASK..001 G1
STOR..001 G2
PGM...001 G3
FILE..001 G4
TRSN..001 G5
DUMP..001 H2
TIME..001 H3
TEMP..001 H4
TURN..001 H5
001 G1
001 G2
001 G3
001 G4
001 G5
DDCTRP05
G1
BUILD TASK CONTROL ENTRY
DDCTRP06
G2
BUILD STORAGE CONTROL ENTRY
DDCTRP07
G3
BUILD PROGRAM CONTROL ENTRY
DDCTRP08
G4
BUILD FILE CONTROL ENTRY
DDCTRP09
G5
BUILD TRANSIENT DATA ENTRY
001 H5
DDCTRP10
H1
BUILD USER ENTRY
DDCTRP11
H2
BUILD DUMP CONTROL ENTRY
DDCTRP12
H3
BUILD TIME CONTROL ENTRY
DDCTRP13
H4
BUILD TEMPORARY STORAGE ENTRY
DDCTRP14
H5
TURN OFF TURN ON TRACE
DDCTRP15
J3
COMMON EXIT CHECK FOR DUP ENTRY
DDCTRP16
K3
EXIT

HIGH-LEVEL LANGUAGE PREPROCESSOR
DFHPRPR
CHART 26



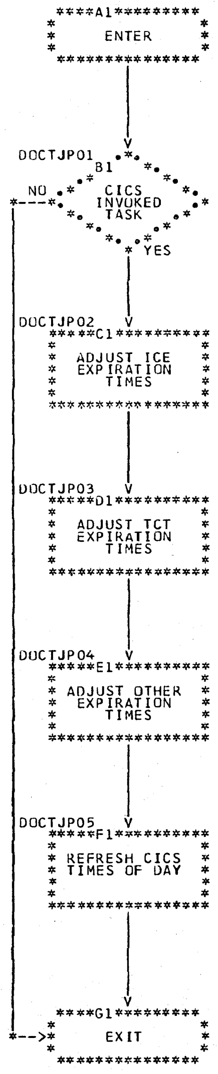
DEVICE DEPENDENT MODULE
DFHDDM
CHART 27



TIME ADJUSTMENT PROGRAM

DFHTAJP

CHART 28



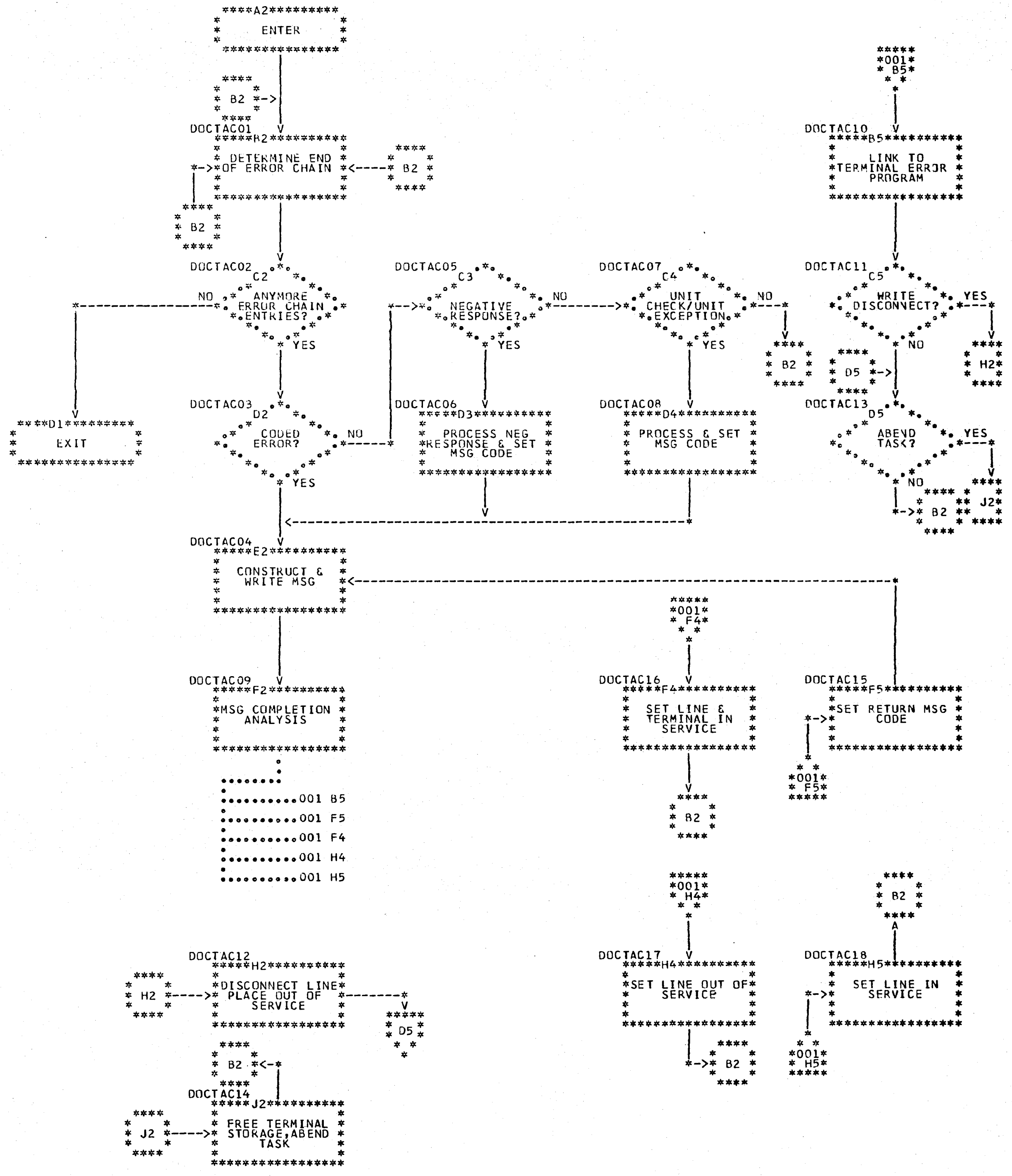
```
                                        DUMMY CSA PROGRAM                                    PAGE   001
                                             DFHDCSA
                                            CHART  29



DOCDCS01
    *****A1**********
   *                 *
   *      START      *
   *                 *
    *****************
            |
            |
            |
            |
DOCDCS02    V
   *****B1**********
   *               *
   *   SAVE O/S    *
   *   REGISTERS   *
   *               *
   *               *
   *****************
            |
            |
            |
            |
DOCDCS03    V
   *****C1**********
   *               *
   *ESTABLISH CSA &*
   * TERM CTL TCA  *
   *               *
   *               *
   *****************
            |
            |
            |
            |
DOCDCS04    V
   *****D1**********
   *               *
   *STORAGE CONTROL*
   *MACRO TO OBTAIN*
   *    CUSHION    *
   *               *
   *****************
            |
            |
            |
            |
DOCDCS05    V
   *****E1**********
   *               *
   *  SET UP BASE  *
   * REGS FOR TERM *
   *      CTL      *
   *               *
   *****************
            |
            |
            |
            |
DOCDCS06    V
   *****F1**********
   *               *
   *    EXIT TO    *
   *   TERMINAL    *
   *    CONTROL    *
   *               *
   *****************
```

PL/I STORAGE ALLOCATION PROGRAM
DFHSAP
CHART 30

```
DOCSAP01
     ****A1**********
    *                *
    *     START      *
    *                *
     ****************
            |
            |
            |
            |
DOCSAP02    V
    *****B1**********
    *                *
    *   BRANCH TO    *
    *REQUESTED ENTRY*
    *      PT.       *
    *                *
    *****************
            |
            |
            |
            |
DOCSAP03    V
    *****C1**********
    *                *
    *                *
    *MODIFIED IHESA  *
    *                *
    *                *
    *****************
            |
            |
            |
            |
            V
    *****D1**********
    *                *
    *   RETURN TO    *
    *  APPLICATION   *
    *    PROGRAM     *
    *                *
    *****************
```

```
DOCPLI01
   ****A1**********
   *              *
   *     START    *
   *              *
   ****************
          |
          |
          |
DOCPLI02  V
  *****B1***********
  *               *
  *    TURN ON    *
  *   PROLOGUE    *----*
  *  REQUEST BIT  *    |
  *               *    |
  *****************    |
                       |
                       |
DOCPLI03               |
   ****C1**********    |
   *              *    |
   *     START    *    |
   *              *    |
   ****************    |
          |            |
          |            |
          |            |
DOCPLI04  V            |
  *****D1***********   |
  *               *    |
  *   GOTO PL/I   *    |
  * INTERFACE IN  *    |
  *PROGRAM CONTROL*    |
  *               *    |
  *****************    |
                       |
                       |
DOCPLI05               |
   ****E1**********    |
   *              *    |
   *     START    *    |
   *              *    |
   ****************    |
          |            |
          |            |
          |            |
DOCPLI06  V            |
  *****F1***********   |
  *               *    |
  *TURN ON PROPER *    |
  *  REQUEST BIT  *    |
  *               *    |
  *               *    |
  *****************    |
          |            |
          |<-----------*
          |
DOCPLI07  V
  *****G1***********
  *               *
  * HOUSEKEEPING  *
  *  FOR DFHSAP   *
  *               *
  *               *
  *****************
          |
          |
          |
DOCPLI08  V
   ****H1**********
   *              *
   *EXIT TO DFHSAP *
   *              *
   ****************
```

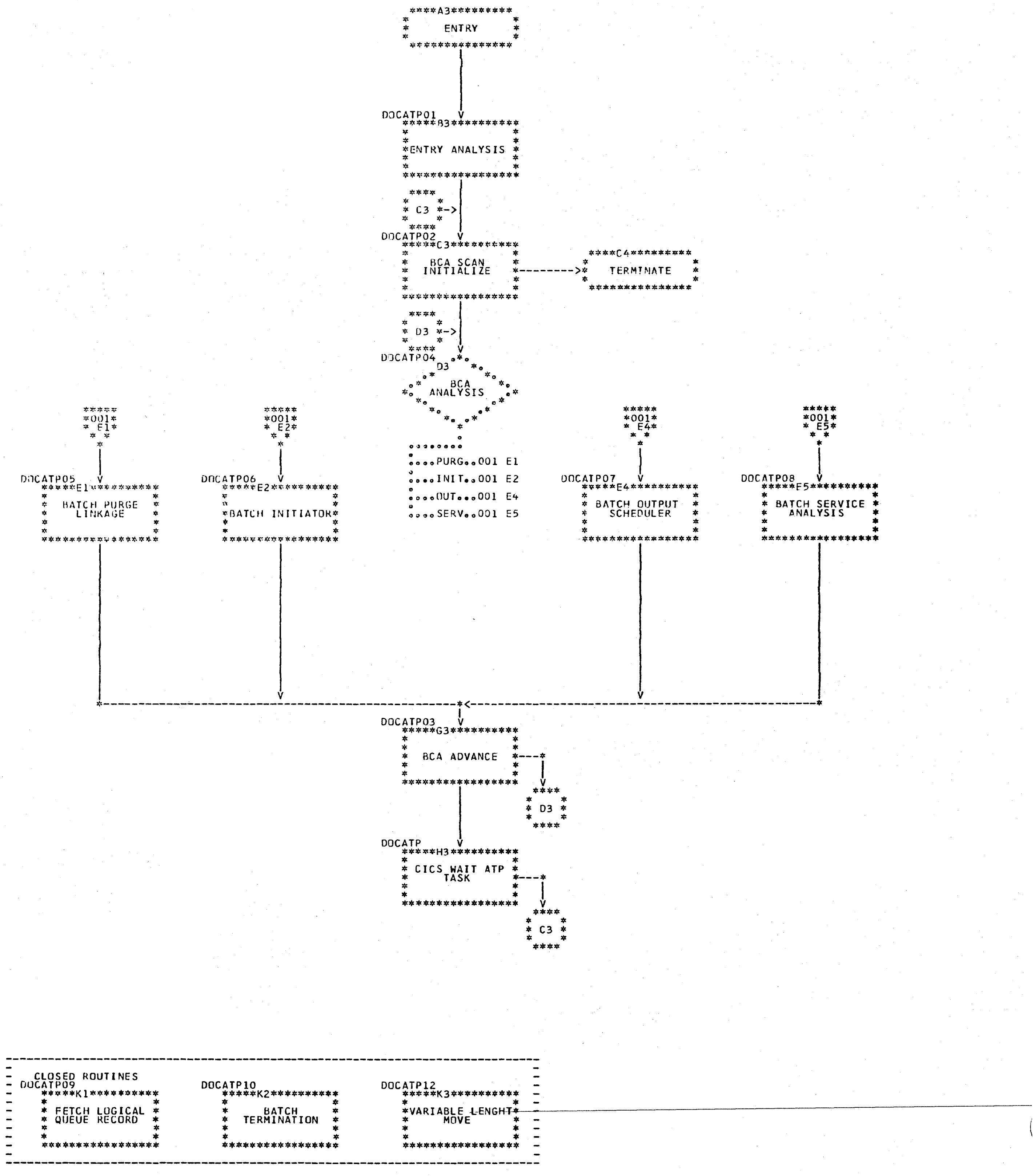
TERMINAL ABNORMAL CONDITION PROGRAM
DFHTACP
CHART 32



ASY JUS TRANSACTION CONTROL PROGRAM

DFHATP

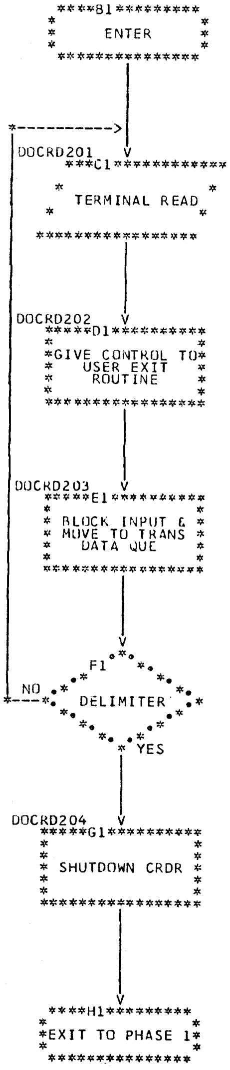
CHART 33



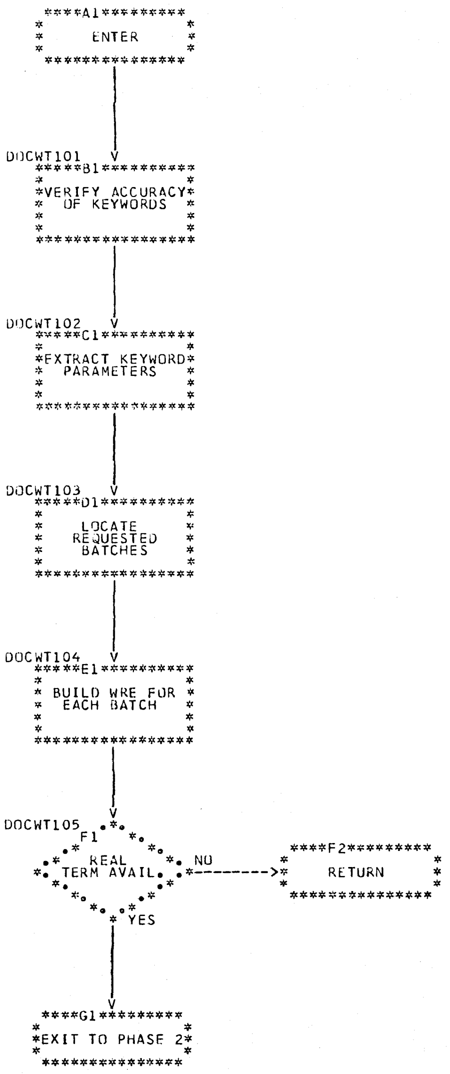
```
                       ASYNCHRONOUS TRANSACTION INPUT PROCESSOR
                                 PHASE 1 (DFHRD1)
                                    CHART 33
```

```
    ****C1*********
   *               *
   *     ENTER     *
   *               *
    ***************
           |
           |
           |
           |
DOCRD101   V
   *****D1**********
   *               *
   *               *
   *ENTRY ANALYSIS *------------------*
   *               *                  |
   *               *                  |
   *****************                  |
           |                          |
           |                          |
           |                          |
DOCRD102   V              DOCRD106    V
   *****E1**********          *****E2**********
   *               *          *               *
   *INTERPRET CRDR *          *DETERMINE TYPE *
   *    MESSAGE    *          *& BUILD MESSAGE*
   *               *          *               *
   *               *          *               *
   *****************          *****************
           |                          |
           |                          |
           |                          |
DOCRD103   V              DOCRD107    V
   *****F1**********          *****F2**********
   *               *          *               *
   *EXTRACT KEYWORD*          * PURGE BCA IF  *
   *  PARAMETERS   *          *   NECESSARY   *
   *               *          *               *
   *               *          *               *
   *****************          *****************
           |                          |
           |                          |
           |                          |
DOCRD104   V                          V
   *****G1**********           ****G2*********
   *               *          *               *
   *               *          *               *
   *   BUILD BCA   *          *     EXIT      *
   *               *          *               *
   *               *           ***************
   *****************
           |
           |
           |
DOCRD105   V
   *****H1**********
   *               *
   *   TRANSFER    *
   *  CONTROL TO   *
   *    PHASE 2    *
   *               *
   *****************
```

ASYNCHRONOUS TRANSACTION INPUT PROCESSOR
PHASE 2 (DFHRD2)
CHART 33



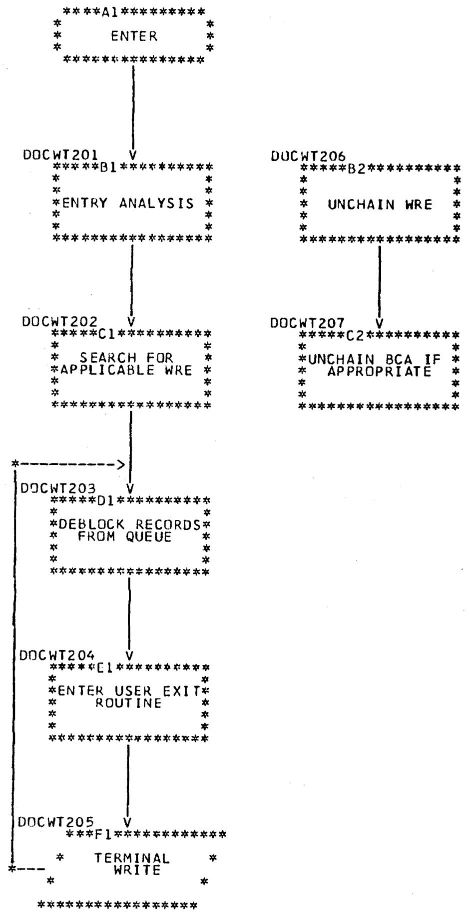
ASYNCHRONOUS TRANSACTION OUTPUT PROCESSOR
WRE BUILD (DFHWT1)
CHART 33

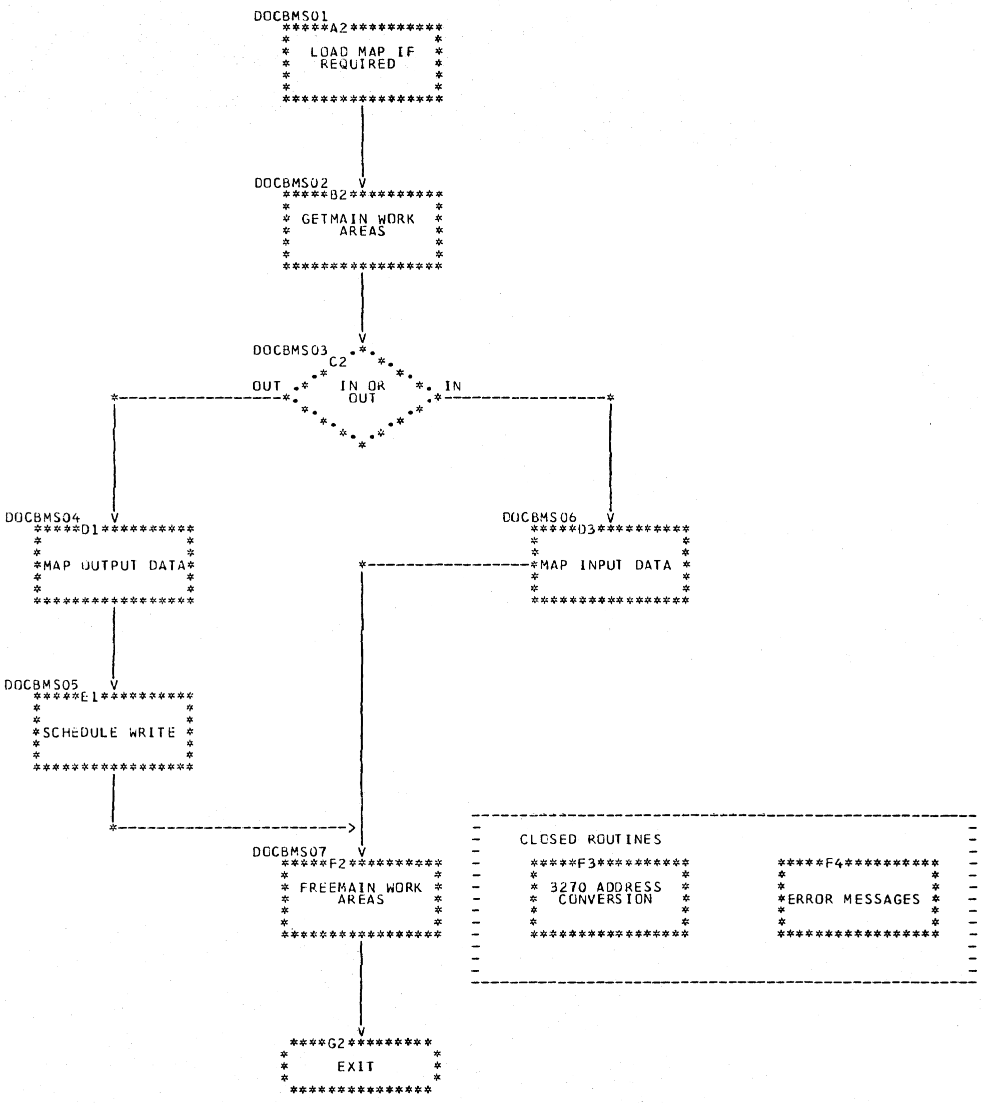


ASYNCHRONOUS TRANSACTION OUTPUT PROCESSOR

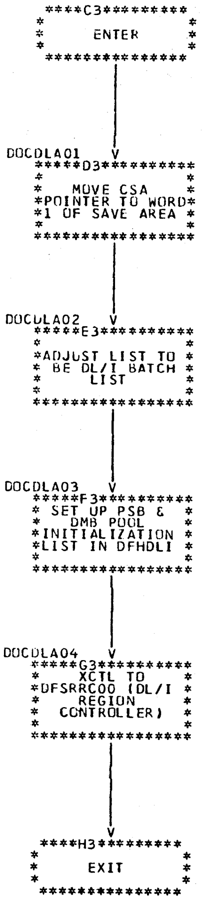
OUTPUT SECTION (DFHWT2)

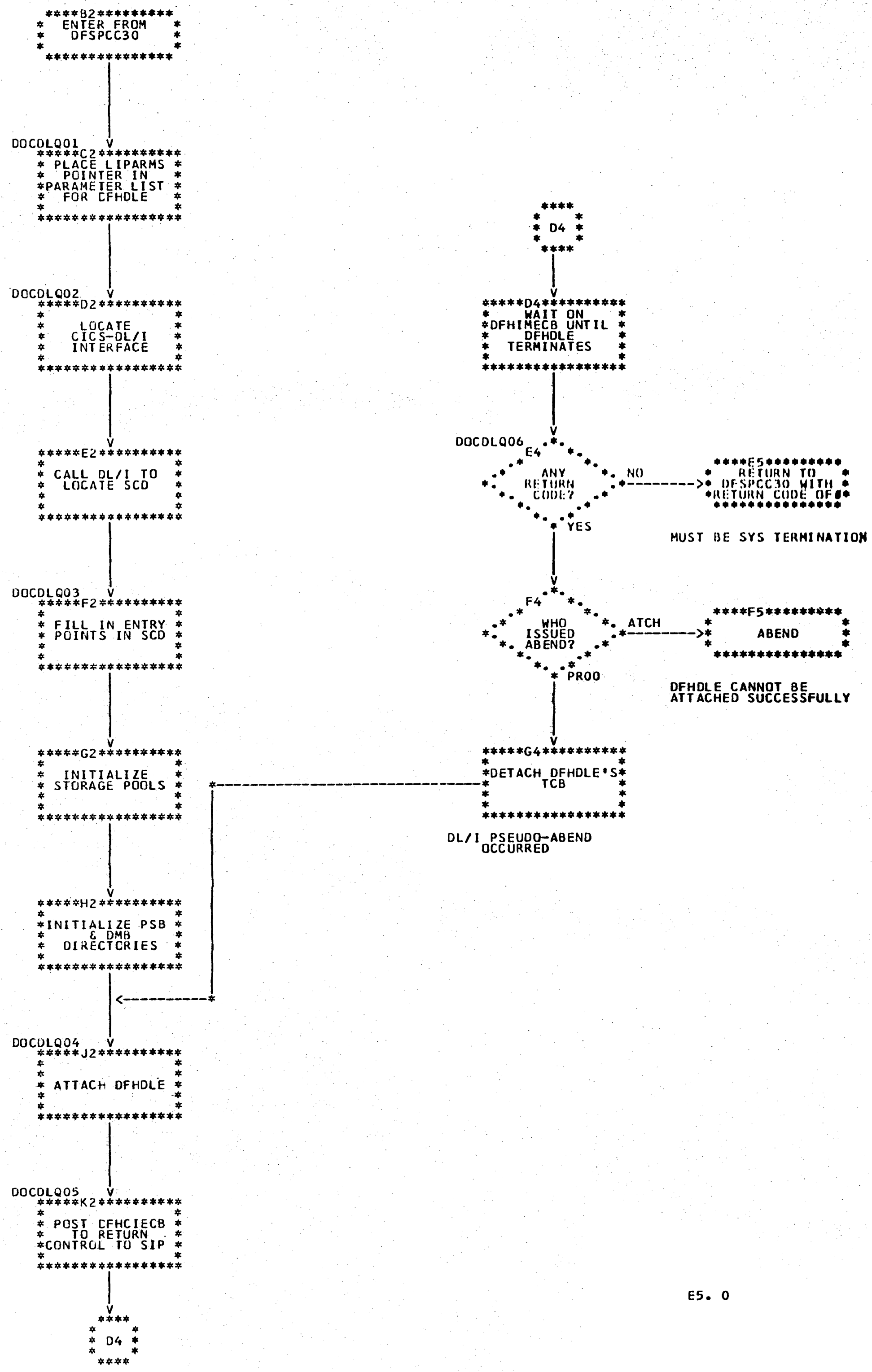
CHART 33




DOCBMS01
A2
LOAD MAP IF REQUIRED
DOCBMS02
B2
GETMAIN WORK AREAS
DOCBMS03
C2
IN OR OUT
OUT
IN
DOCBMS04
D1
MAP OUTPUT DATA
DOCBMS06
D3
MAP INPUT DATA
DOCBMS05
E1
SCHEDULE WRITE
DOCBMS07
F2
FREEMAIN WORK AREAS
CLOSED ROUTINES
F3
3270 ADDRESS CONVERSION
F4
ERROR MESSAGES
G2
EXIT

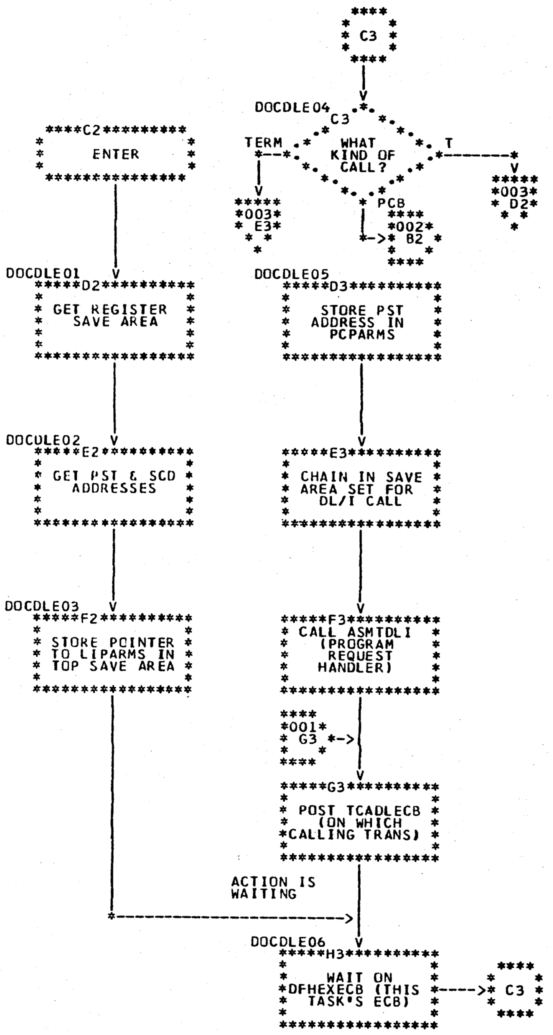
DFHDLA
INITIALIZE AND ATTACH DL/I SUBTASK
CHART 35



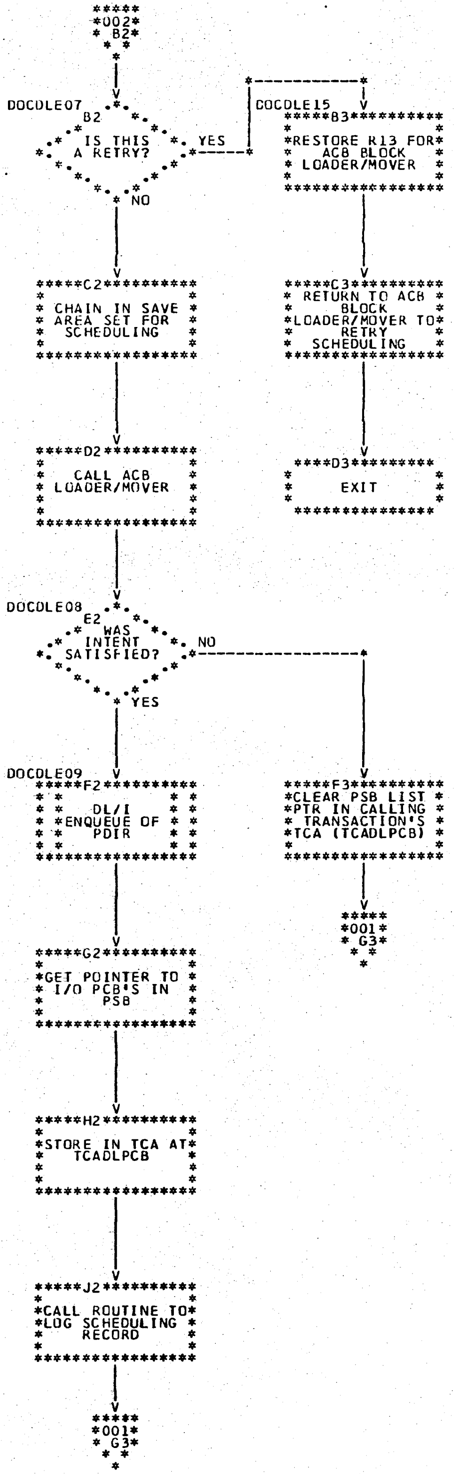

B2
ENTER FROM DFSPCC30
DOCDLQ01
C2
PLACE LIPARMS POINTER IN PARAMETER LIST FOR DFHDLE
DOCDLQ02
D2
LOCATE CICS-DL/I INTERFACE
E2
CALL DL/I TO LOCATE SCD
DOCDLQ03
F2
FILL IN ENTRY POINTS IN SCD
G2
INITIALIZE STORAGE POOLS
H2
INITIALIZE PSB & DMB DIRECTORIES
DOCDLQ04
J2
ATTACH DFHDLE
DOCDLQ05
K2
POST DFHCIECB TO RETURN CONTROL TO SIP
D4
D4
D4
WAIT ON DFHIMECB UNTIL DFHDLE TERMINATES
DOCDLQ06
E4
ANY RETURN CODE?
NO
YES
E5
RETURN TO DFSPCC30 WITH RETURN CODE OF 0
MUST BE SYS TERMINATION
F4
WHO ISSUED ABEND?
ATCH
PROO
F5
ABEND
DFHDLE CANNOT BE ATTACHED SUCCESSFULLY
G4
DETACH DFHDLE'S TCB
DL/I PSEUDO-ABEND OCCURRED

E5. 0

DFHDLE
DL/I CALL EXECUTOR SUBTASK
CHART 37

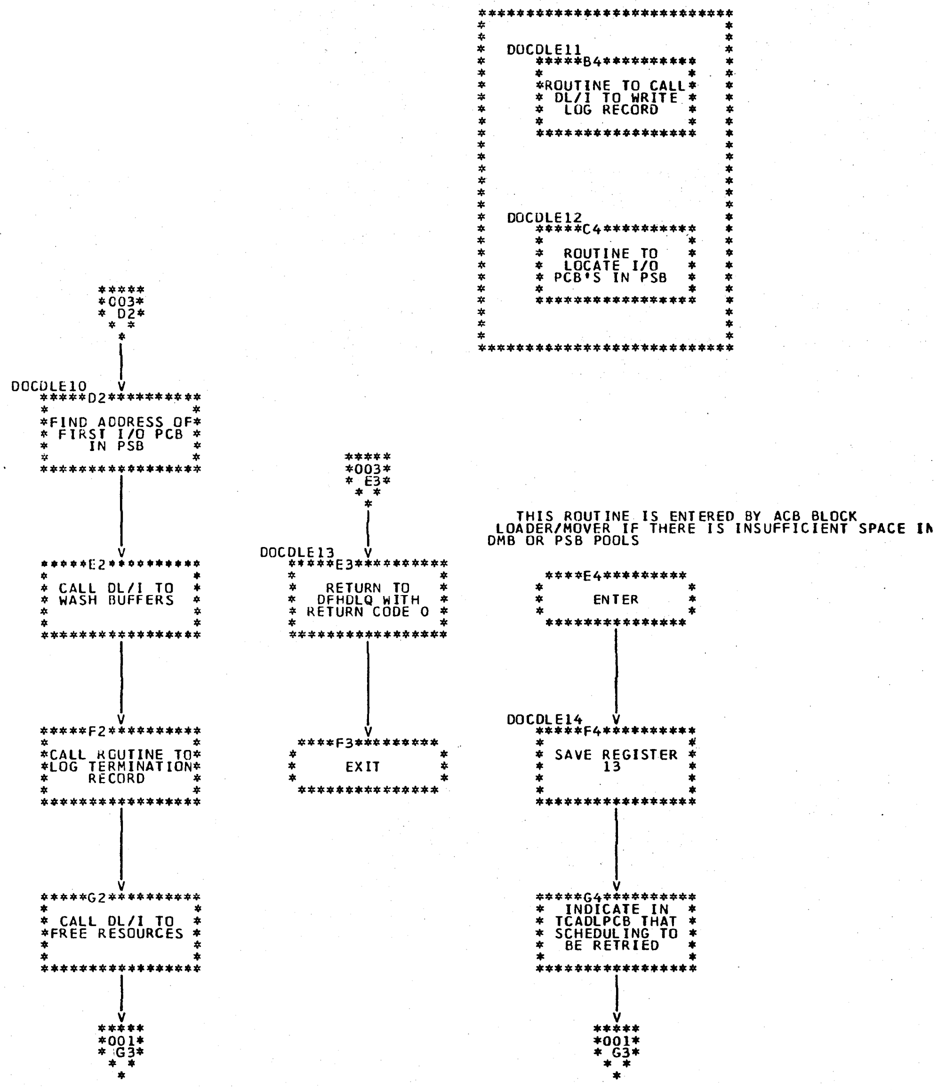


DFHDLE
DL/I CALL EXECUTOR SUBTASK - 'PCB'(SCHEDULING) CALL
CHART 37

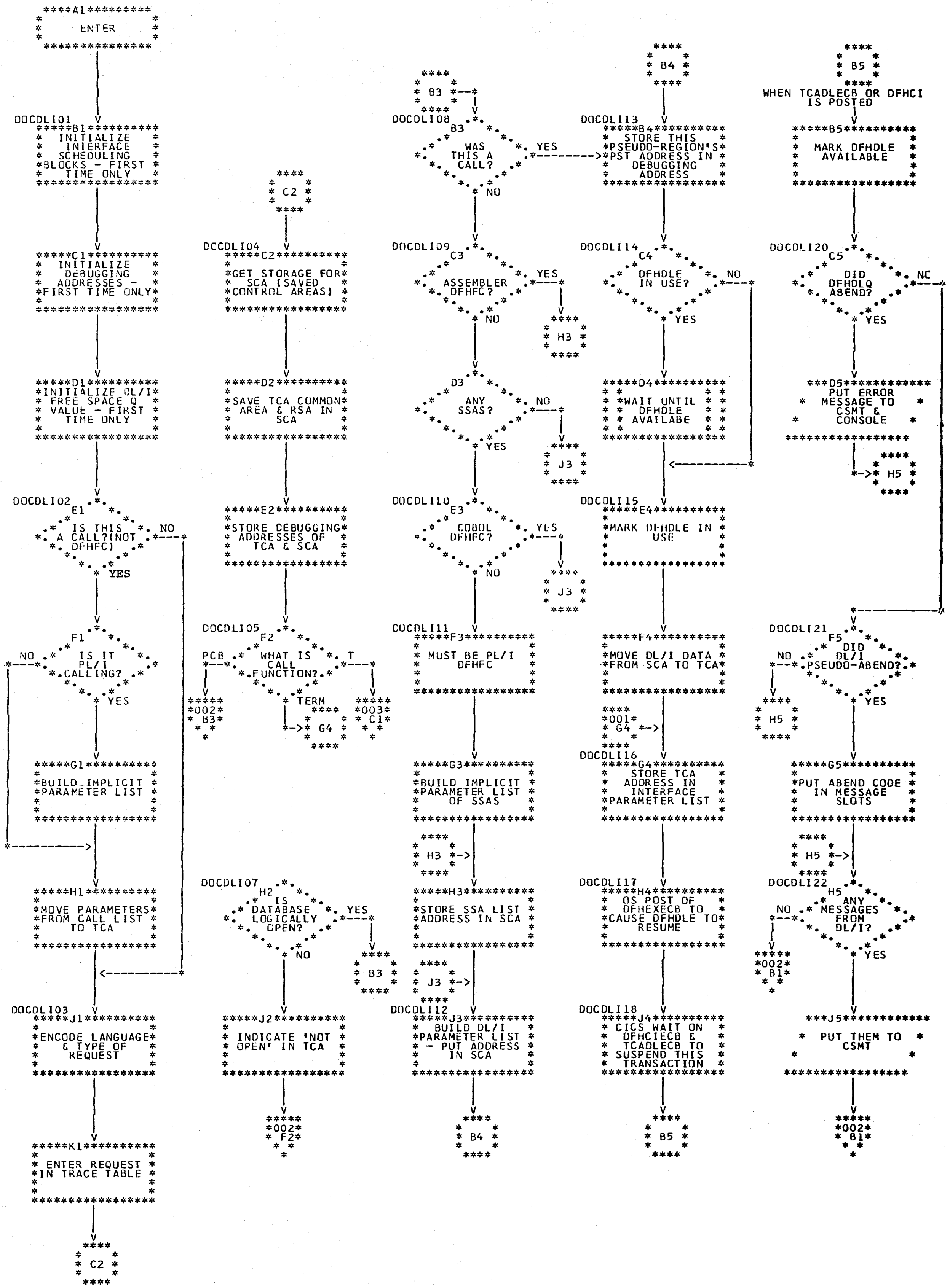


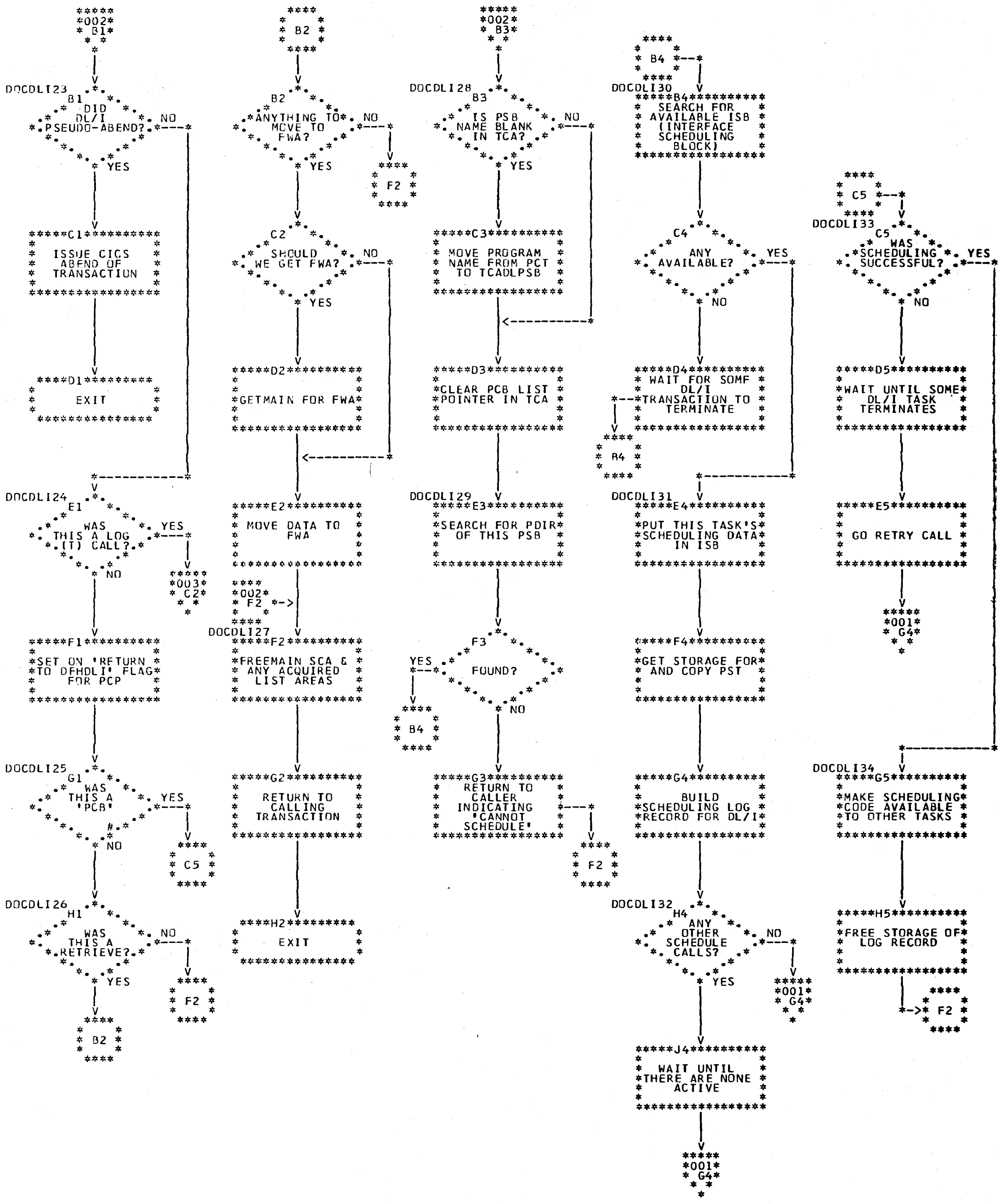
DL/I CALL EXECUTOR SUBTASK – 'T' (TRANSACTION TERMINATION CALL

CHART 37



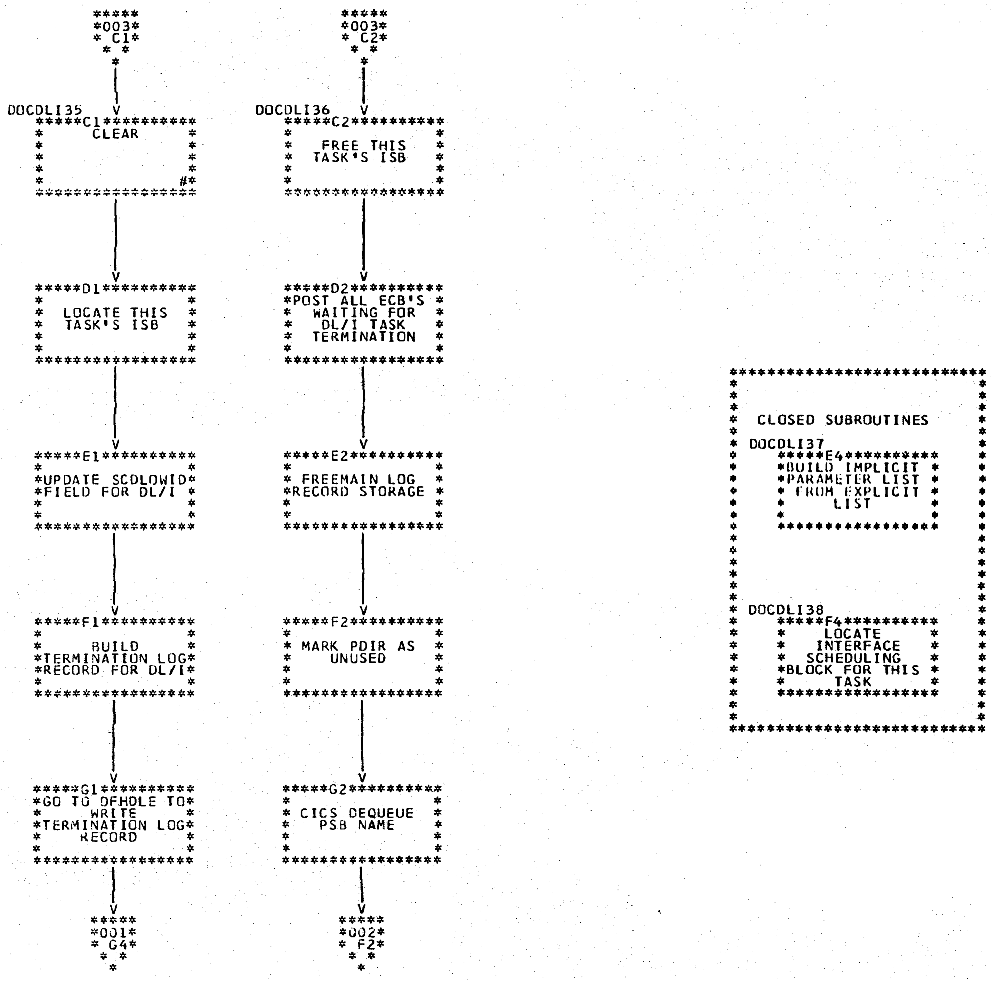
THIS ROUTINE IS ENTERED BY ACB BLOCK LOADER/MOVER IF THERE IS INSUFFICIENT SPACE IN DMB OR PSB POOLS


A1
ENTER
DOCDLI01
B1
INITIALIZE INTERFACE SCHEDULING BLOCKS - FIRST TIME ONLY
C1
INITIALIZE DEBUGGING ADDRESSES - FIRST TIME ONLY
D1
INITIALIZE DL/I FREE SPACE Q VALUE - FIRST TIME ONLY
DOCDLI02
E1
IS THIS A CALL?(NOT DFHFC)
NO
YES
F1
IS IT PL/I CALLING?
NO
YES
G1
BUILD IMPLICIT PARAMETER LIST
H1
MOVE PARAMETERS FROM CALL LIST TO TCA
DOCDLI03
J1
ENCODE LANGUAGE & TYPE OF REQUEST
K1
ENTER REQUEST IN TRACE TABLE
C2
C2
DOCDLI04
C2
GET STORAGE FOR SCA (SAVED CONTROL AREAS)
D2
SAVE TCA COMMON AREA & RSA IN SCA
E2
STORE DEBUGGING ADDRESSES OF TCA & SCA
DOCDLI05
F2
WHAT IS CALL FUNCTION?
PCB
T
TERM
002 B3
003 C1
G4
DOCDLI07
H2
IS DATABASE LOGICALLY OPEN?
YES
NO
B3
J2
INDICATE 'NOT OPEN' IN TCA
002 F2
B3
DOCDLI08
B3
WAS THIS A CALL?
YES
NO
DOCDLI09
C3
ASSEMBLER DFHFC?
YES
NO
H3
D3
ANY SSAS?
NO
YES
J3
DOCDLI10
E3
COBOL DFHFC?
YES
NO
J3
DOCDLI11
F3
MUST BE PL/I DFHFC
G3
BUILD IMPLICIT PARAMETER LIST OF SSAS
H3
H3
STORE SSA LIST ADDRESS IN SCA
J3
DOCDLI12
J3
BUILD DL/I PARAMETER LIST - PUT ADDRESS IN SCA
B4
B4
DOCDLI13
B4
STORE THIS PSEUDO-REGION'S PST ADDRESS IN DEBUGGING ADDRESS
DOCDLI14
C4
DFHDLE IN USE?
NO
YES
D4
WAIT UNTIL DFHDLE AVAILABE
DOCDLI15
E4
MARK DFHDLE IN USE
F4
MOVE DL/I DATA FROM SCA TO TCA
001 G4
DOCDLI16
G4
STORE TCA ADDRESS IN INTERFACE PARAMETER LIST
DOCDLI17
H4
OS POST OF DFHEXECB TO CAUSE DFHDLE TO RESUME
DOCDLI18
J4
CICS WAIT ON DFHCIECB & TCADLECB TO SUSPEND THIS TRANSACTION
B5
B5
WHEN TCADLECB OR DFHCI IS POSTED
B5
MARK DFHDLE AVAILABLE
DOCDLI20
C5
DID DFHDLQ ABEND?
NO
YES
D5
PUT ERROR MESSAGE TO CSMT & CONSOLE
H5
DOCDLI21
F5
DID DL/I PSEUDO-ABEND?
NO
YES
H5
G5
PUT ABEND CODE IN MESSAGE SLOTS
H5
DOCDLI22
H5
ANY MESSAGES FROM DL/I?
NO
YES
002 B1
J5
PUT THEM TO CSMT
002 B1

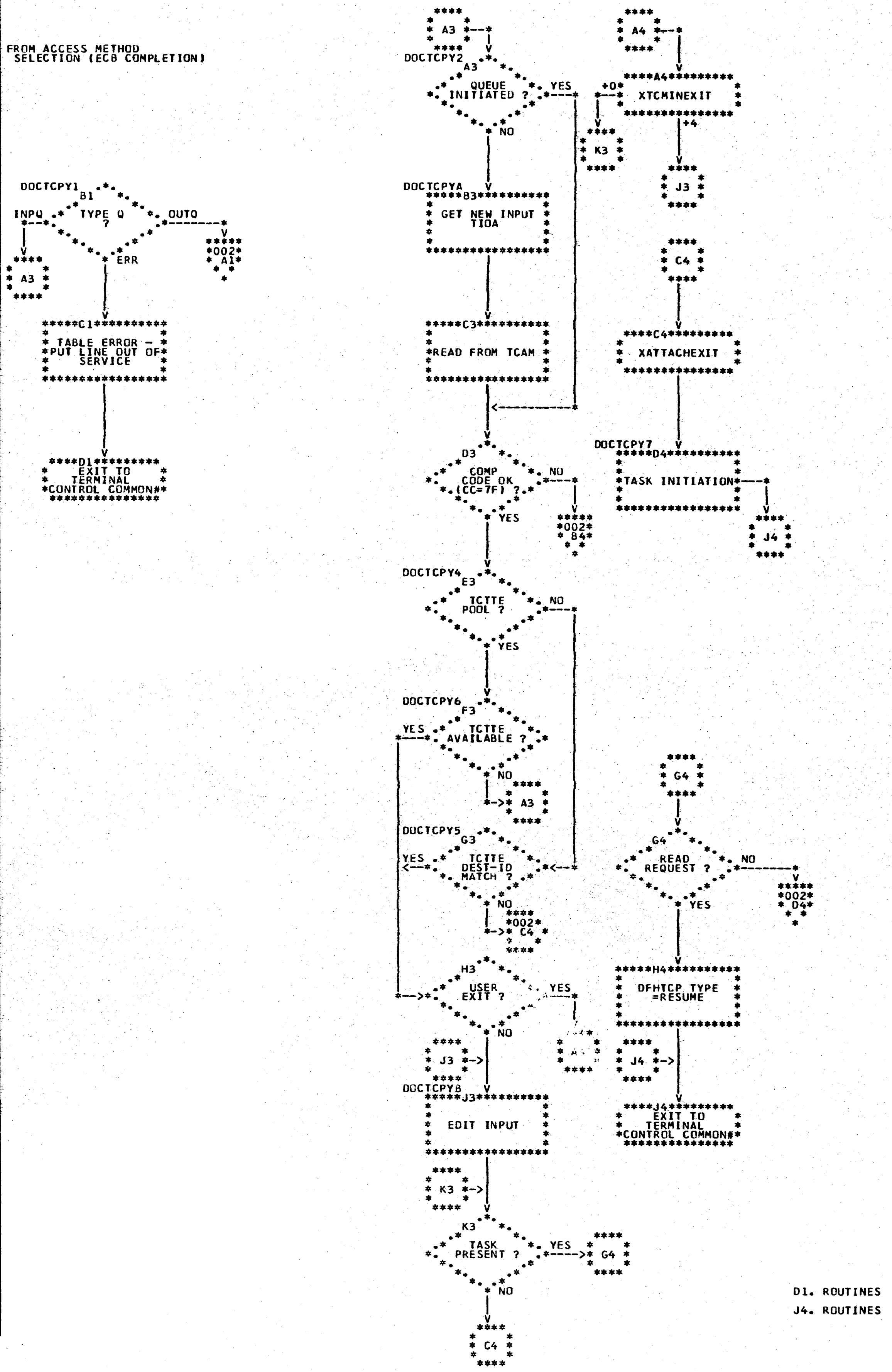


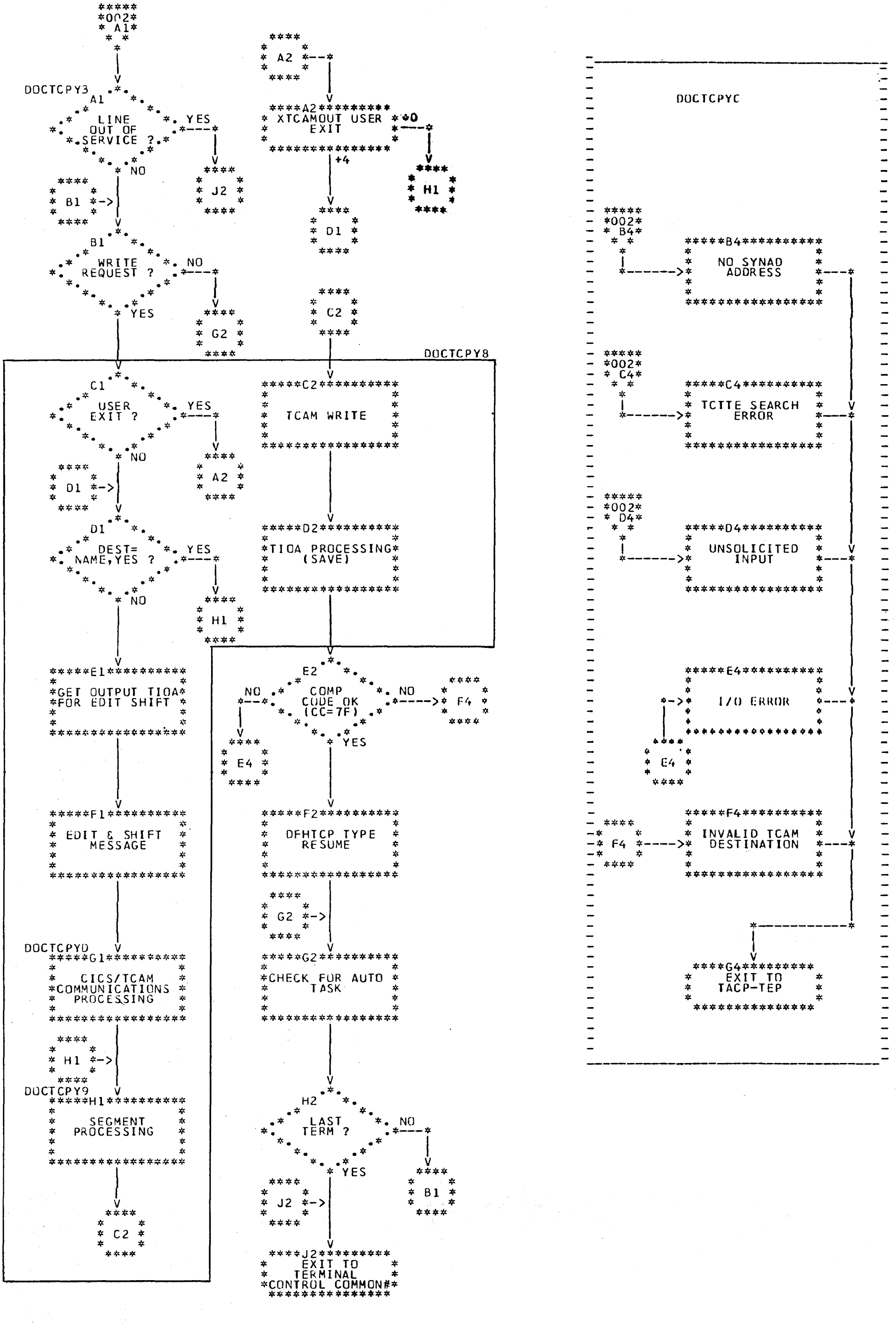
G1. (SCHEDULE) CALL?

DFHDLI / CICS-DL/I INTERFACE
TERMINATION ('T') CALL, SUBROUTINES
CHART 38



C1. 'RETURN-TO-DFHDLI'
FLAG ('DL/I WAS HERE')


FROM ACCESS METHOD SELECTION (ECB COMPLETION)
DOCTCPY1
B1
TYPE Q ?
INPQ
OUTQ
ERR
A3
002 A1
C1
TABLE ERROR - PUT LINE OUT OF SERVICE
D1
EXIT TO TERMINAL CONTROL COMMON
A3
DOCTCPY2
A3
QUEUE INITIATED ?
YES
NO
DOCTCPYA
B3
GET NEW INPUT TIOA
C3
READ FROM TCAM
D3
COMP CODE OK (CC=7F) ?
NO
YES
002 B4
DOCTCPY4
E3
TCTTE POOL ?
NO
YES
DOCTCPY6
F3
TCTTE AVAILABLE ?
YES
NO
A3
DOCTCPY5
G3
TCTTE DEST-ID MATCH ?
YES
NO
002 C4
H3
USER EXIT ?
YES
NO
J3
DOCTCPYB
J3
EDIT INPUT
K3
K3
TASK PRESENT ?
YES
G4
NO
C4
A4
A4
XTCMINEXIT
+0
+4
K3
J3
C4
C4
XATTACHEXIT
DOCTCPY7
D4
TASK INITIATION
J4
G4
G4
READ REQUEST ?
NO
002 D4
YES
H4
DFHTCP TYPE =RESUME
J4
J4
EXIT TO TERMINAL CONTROL COMMON
D1. ROUTINES
J4. ROUTINES


002
A1
DOCTCPY3
A1
LINE OUT OF SERVICE ?
YES
NO
J2
B1
B1
WRITE REQUEST ?
NO
YES
G2
A2
A2
XTCAMOUT USER EXIT
+0
+4
H1
D1
C2
DOCTCPY8
C1
USER EXIT ?
YES
NO
A2
C2
TCAM WRITE
D1
D1
DEST= NAME,YES ?
YES
NO
H1
D2
TIOA PROCESSING (SAVE)
E1
GET OUTPUT TIOA FOR EDIT SHIFT
E2
COMP CODE OK (CC=7F)
NO
NO
F4
E4
YES
F1
EDIT & SHIFT MESSAGE
F2
DFHTCP TYPE RESUME
G2
DOCTCPYD
G1
CICS/TCAM COMMUNICATIONS PROCESSING
G2
CHECK FOR AUTO TASK
H1
DOCTCPY9
H1
SEGMENT PROCESSING
C2
H2
LAST TERM ?
NO
YES
B1
J2
J2
EXIT TO TERMINAL CONTROL COMMON
DOCTCPYC
002
B4
B4
NO SYNAD ADDRESS
002
C4
C4
TCTTE SEARCH ERROR
002
D4
D4
UNSOLICITED INPUT
E4
I/O ERROR
E4
F4
F4
INVALID TCAM DESTINATION
G4
EXIT TO TACP-TEP

REGISTER USAGE

USER'S REGISTERS

The following registers are available to the user throughout his entire program:

Registers 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, and 15.

Registers 12 and 13 are used by all user-written programs as defined below:

Register 12	Base register for Task Control Area and Transaction Work Area (TCA/TWA)
Register 13	Base register for Common System Area (CSA)

Register 14 may be used by user-written programs. However, before any CICS macro instruction is issued, the user value that is in register 14 must be saved by the user program if that value is of any significance to the user program. If not saved by the user program, the macro expansion will destroy the value that was in register 14.

CICS CONTROL REGISTERS

Label	Register	Function
TCACBAR	12	Base register for TCA/TWA. Register 12 contains the address of the TCA associated with the controlling task in CICS.
	13	Base register for CSA. Register 13 contains the address of the CSA throughout the execution of CICS.

DSECT REGISTER USAGE

Each DSECT within CICS has either a label associated with a register or a label that is to be associated with a user program register. Listed below are the DSECTS:

DSECT FOR:	LABEL	REGISTER
Common System Area		13
Destination Control Table	DCTCBAR	*
File Control Table	FCTDSBAR	*
File Control Table Indirect Access	FCTIABAR	*
File Control Table Segment Definition	FCTSDBAR	*
File Control Table Segment Header	FCTSHBAR	*
File Control Table Segment Set	FCTSSBAR	*
File Input/Output Area	FIOABAR	*
File Work Area	FWACBAR	*
Intrapartition Data Input Area	TDIABAR	*
Processing Program Table	PPTCBAR	*
Program Control Table	PCTCBAR	*
Sign-on Table	SNNTBAR	*
Storage Accounting Area	SAACBAR	*

DSECT FOR:	LABEL	REGISTER
Task Control Area	TCACBAR	12
Temporary Storage Input/Output Area	TSIOABAR	*
Temporary Storage Tables	TSATBAR	*
Terminal Control Table Line Entry	TCTLEAR	1
Terminal Control Table Terminal Entry	TCTTEAR	*
Terminal Input/Output Area	TIOABAR	*
Transient Data Output Area	TDOABAR	*

* User's choice

CICS MANAGEMENT PROGRAM REGISTER USAGE

All CICS management programs and system service programs provide symbolic definition for registers used within the individual program. The register definitions and assignments, as well as all equated symbols, are located at the beginning of every program assembly listing.

CICS is a modular, table-controlled system. To understand the operation of the system, knowledge of the CICS Control Tables and Areas and their contents is required. The following are functional descriptions and the general contents of the various control areas and tables encountered in the CICS/OS-STANDARD system.

COMMON SYSTEM AREA

DSECT NAME: DFHCSADS

The Common System Area (CSA) is a main storage control area provided as a part of CICS. 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 necessary to the operation of CICS and an optional work area that may be used as temporary work storage by a processing program. The user temporary work storage is available for operations that are performed between requests to CICS. This work space is available to any task while it has control of the system.

```
                     C O M M O N   S Y S T E M   A R E A
               ------------------------------------------------------------
               *                                                          *
Dec.   Hex.    * <------------------4 BYTES-----------------------------> *
               *                                                          *
| -112   -70   ************************************************************
               *                                                          *
               *          COPYRIGHT INFORMATION IBM CORP                  *
               *                                                          *
   0      0    *----------------------------------------------------------*
               *                       CSAOSRSA                           *
               *                                                          *
               *          COMMON SYSTEM REGISTER STORAGE AREA             *
               *                                                          *
  72     48    *----------------------------------------------------------*
               *  CSASOSI    *   CSAKCMI  *          CSAKCMT              *
               *             *            *                               *
               * SHORT ON    * MAX NO OF  *   MAX NUMBER OF TASKS CTL     *
               * STRG IND    * TASKS IND  *                               *
  76     4C    *----------------------------------------------------------*
               *                       CSACDTA                            *
               *                                                          *
               *          CURRENTLY DISPATCHED TASK ADDRESS               *
  80     50    *----------------------------------------------------------*
               *                       CSATODP                            *
               *                                                          *
               *                  TIME OF DAY PACKED                      *
  84     54    *----------------------------------------------------------*
               *                       CSAICEBA                           *
               *                                                          *
               *          INTERVAL CTL ELEMENT CHAIN BEGIN ADDR           *
  88     58    *----------------------------------------------------------*
               *         CSAICSIC          *               *  CSAICIND    *
               *                           *               *              *
               *                           *  RESERVED     *  INTERVAL    *
               *   STALL TIME INTERVAL     *               * CONTROL IND  *
  92     5C    *----------------------------------------------------------*
```

COMMON SYSTEM AREA

```
           *                                                           *
Dec.  Hex. * <--------------------4 BYTES----------------------------> *
           *                                                           *
           *************************************************************
           *                         CSATADJT                          *
           *                                                           *
           *               TIME OF DAY ADJUSTMENT VALUE                *
96    60   *-----------------------------------------------------------*
           *                         CSACTODB                          *
           *                         (CSACSCC)                         *
           *                                                           *
           *               CURRENT TIME OF DAY - BINARY                *
100   64   *-----------------------------------------------------------*
           *                          CSASBTI                          *
           *                                                           *
           *               SYSTEM BINARY TIMER INTERVAL                *
104   68   *-----------------------------------------------------------*
           *                         CSATTECB                          *
           *                                                           *
           *             TERMINAL TIME EVENT CONTROL BLOCK             *
108   6C   *-----------------------------------------------------------*
           *                         CSASITOD                          *
           *                                                           *
           *            TIME OF DAY AT SYSTEM INIT - BINARY            *
112   70   *-----------------------------------------------------------*
           *                          CSASCNB                          *
           *                                                           *
           *              STORAGE CUSHION NUMBER OF BYTES              *
116   74   *-----------------------------------------------------------*
           *                          CSAPLBA                          *
           *                                                           *
           *             PARTITION LOWER BOUNDARY ADDRESS              *
120   78   *-----------------------------------------------------------*
           *                          CSAPUBA                          *
           *                                                           *
           *             PARTITION UPPER BOUNDARY ADDRESS              *
124   7C   *-----------------------------------------------------------*
           *                          CSAJYDP                          *
           *                                                           *
           *             JULIAN DATE - YEAR AND DAY PACKED             *
128   80   *-----------------------------------------------------------*
           *                         CSATDTCA                          *
           *                                                           *
           *                TASK DISPATCHER TCA ADDRESS                *
132   84   *-----------------------------------------------------------*
           *   CSATRMF1   *  CSATRMF2   *                              *
           *              *             *         RESERVED             *
           * TRACE MASTER * TRACE SYSTEM*                              *
           *     FLAGS    *   FLAGS     *                              *
136   88   *----------------------------*                              *
           *                                                           *
           *                                                           *
144   90   *-----------------------------------------------------------*
           *                         CSAUNQID                          *
           *                                                           *
           *               UNIQUE IDENTIFICATION COUNTER               *
148   94   *-----------------------------------------------------------*
           *                         CSAAIDBA                          *
           *                                                           *
           *     AUTO INITIATE DESCRIPTOR CHAIN BEGINNING ADDRESS      *
152   98   *-----------------------------------------------------------*
```

COMMON SYSTEM AREA

```
             ┌──────────────────────────────────────────────────────────┐
             *                                                          *
Dec.  Hex.   * <─────────────────────4 BYTES─────────────────────────>  *
             *                                                          *
             ************************************************************
             *                        CSASP0AD                          *
             *                                                          *
             *               SUBPOOL 0 BOUNDARY ADDRESS                 *
156    9C    *----------------------------------------------------------*
             *                        CSASTCA                           *
             *                                                          *
             *           DUMMY SUSPENDED TCA STARTING ADDRESS           *
160    A0    *----------------------------------------------------------*
             *                                                          *
             *                        RESERVED                          *
164    A4    *----------------------------------------------------------*
             *                        CSATCA                            *
             *                                                          *
             *             DUMMY ACTIVE TCA STARTING ADDRESS            *
168    A8    *----------------------------------------------------------*
             *                        CSASUSFA                          *
             *                                                          *
             *           SUSPENDED TCA FORWARD CHAIN ADDRESS            *
172    AC    *----------------------------------------------------------*
             *                        CSASUSBA                          *
             *                                                          *
             *           SUSPENDED TCA BACKWARD CHAIN ADDRESS           *
176    B0    *----------------------------------------------------------*
             *                        CSATCAFA                          *
             *                                                          *
             *                TCA FORWARD CHAIN ADDRESS                 *
180    B4    *----------------------------------------------------------*
             *                        CSATCABA                          *
             *                                                          *
             *                TCA BACKWARD CHAIN ADDRESS                *
184    B8    *----------------------------------------------------------*
             *                        CSATCTCA                          *
             *                                                          *
             *             ADDRESS OF TERMINAL CONTROL TCA              *
188    BC    *----------------------------------------------------------*
             *                                                          *
             *                        RESERVED                          *
             *                                                          *
200    C8    *----------------------------------------------------------*
             *                        CSAOPFLA                          *
             *                                                          *
             *               OPTIONAL FEATURE LIST ADDRESS              *
204    CC    *----------------------------------------------------------*
             *                                                          *
             *                        RESERVED                          *
             *                                                          *
224    E0    *----------------------------------------------------------*
             *                        CSAKCNAC                          *
             *                                                          *
             *                TASK CONTROL ENTRY ADDRESS                *
228    E4    *----------------------------------------------------------*
             *                        CSASCNAC                          *
             *                                                          *
             *              STORAGE CONTROL ENTRY ADDRESS               *
232    E8    *----------------------------------------------------------*
```

COMMON SYSTEM AREA

```
              *                                                          *
Dec.   Hex.   * <----------------------4 BYTES-------------------------> *
              *                                                          *
              ************************************************************
              *                        CSAPCNAC                          *
              *                                                          *
              *             PROGRAM CONTROL ENTRY ADDRESS                *
236     EC    *----------------------------------------------------------*
              *                        CSAICNAC                          *
              *                                                          *
              *             INTERVAL CONTROL ENTRY ADDRESS               *
240     F0    *----------------------------------------------------------*
              *                        CSADCNAC                          *
              *                                                          *
              *               DUMP CONTROL ENTRY ADDRESS                 *
244     F4    *----------------------------------------------------------*
              *                        CSATCNAC                          *
              *                                                          *
              *             TERMINAL CONTROL ENTRY ADDRESS               *
248     F8    *----------------------------------------------------------*
              *                        CSAFCNAC                          *
              *                                                          *
              *               FILE CONTROL ENTRY ADDRESS                 *
252     FC    *----------------------------------------------------------*
              *                        CSATDNAC                          *
              *                                                          *
              *         TRANSIENT DATA CONTROL ENTRY ADDRESS             *
256    100    *----------------------------------------------------------*
              *                        CSATSNAC                          *
              *                                                          *
              *        TEMPORARY STORAGE CONTROL ENTRY ADDRESS           *
260    104    *----------------------------------------------------------*
              *                        CSASANAC                          *
              *                                                          *
              *        PL/I STORAGE ALLOCATION MODULE ADDRESS            *
264    108    *----------------------------------------------------------*
              *                        CSATRNAC                          *
              *                                                          *
              *               TRACE CONTROL ENTRY ADDRESS                *
268    10C    *----------------------------------------------------------*
              *                        CSAPINAC                          *
              *                       (CSAPIPSW)                         *
              *                                                          *
              *      PGM INTERRUPT ENTRY ADDRESS (PSW SAVE AREA)         *
272    110    *----------------------------------------------------------*
              *                        CSASNNAC                          *
              *                                                          *
              *              SNAP SHOT PROGRAM ENTRY ADDRESS             *
276    114    *----------------------------------------------------------*
              *                                                          *
              *                        RESERVED                          *
              *                                                          *
284    11C    *----------------------------------------------------------*
              *                        CSATRTBA                          *
              *                                                          *
              *              TRACE TABLE BEGINNING ADDRESS               *
288    120    *----------------------------------------------------------*
              *                        CSAPCTBA                          *
              *                                                          *
              *        PROGRAM CONTROL TABLE BEGINNING ADDRESS           *
292    124    *----------------------------------------------------------*
```

COMMON SYSTEM AREA

```
              ┌────────────────────────────────────────────────────────┐
              *                                                        *
Dec.   Hex.   * <──────────────────4 BYTES─────────────────────────>  *
              *                                                        *
              **********************************************************
              *                       CSAPPTBA                         *
              *                                                        *
              *      PROCESSING PROGRAM TABLE BEGINNING ADDRESS        *
296    128    *--------------------------------------------------------*
              *                       CSATCTBA                         *
              *                                                        *
              *    TERMINAL CONTROL WAIT LIST/LINE ENTRY ADDRESS       *
300    12C    *--------------------------------------------------------*
              *                       CSAFCTBA                         *
              *                                                        *
              *        FILE CONTROL TABLE BEGINNING ADDRESS            *
304    130    *--------------------------------------------------------*
              *                       CSADCTBA                         *
              *                                                        *
              *     DESTINATION CONTROL TABLE BEGINNING ADDRESS        *
308    134    *--------------------------------------------------------*
              *                       CSATSATA                         *
              *                                                        *
              *    TEMPORARY AUXILIARY STORAGE TABLE ADDRESS           *
312    138    *--------------------------------------------------------*
              *                       CSATSMTA                         *
              *                                                        *
              *        TEMPORARY MAIN STORAGE TABLE ADDRESS            *
316    13C    *--------------------------------------------------------*
              *                       CSAQETBA                         *
              *                                                        *
              *        QUEUE ELEMENT TABLE BEGINNING ADDRESS           *
320    140    *--------------------------------------------------------*
              *                       CSAPOLA                          *
              *                                                        *
              *         PROGRAM DATA SET OPEN LIST ADDRESS             *
324    144    *--------------------------------------------------------*
              *                       CSADOLA                          *
              *                                                        *
              *          DUMP DATA SET OPEN LIST ADDRESS               *
328    148    *--------------------------------------------------------*
              *                       CSATOLA                          *
              *                                                        *
              *      TERMINAL DATA SET OPEN LIST ADDRESS               *
332    14C    *--------------------------------------------------------*
              *                       CSAFOLA                          *
              *                                                        *
              *          FILE DATA SET OPEN LIST ADDRESS               *
336    150    *--------------------------------------------------------*
              *                       CSATDOLA                         *
              *                                                        *
              *        TRANSIENT DATA SET OPEN LIST ADDRESS            *
340    154    *--------------------------------------------------------*
              *                       CSATSOLA                         *
              *                                                        *
              *        TEMPORARY STORAGE DATA SET OPEN LIST ADDR       *
344    158    *--------------------------------------------------------*
              *                                                        *
              *                       RESERVED                         *
              *                                                        *
352    160    *--------------------------------------------------------*
```

COMMON SYSTEM AREA

```
             ------------------------------------------------------------
             *                                                          *
Dec.  Hex.   * <--------------------4 BYTES---------------------------> *
             *                                                          *
             ************************************************************
             *                        CSAPICA                           *
             *                                                          *
             *                 PROGRAM INTERRUPT            *-----------*
             *                   CONTROL AREA               *           *
             *                                              *  RESERVED *
             *                                              *           *
360   168    *----------------------------------------------------------*
             *                        CSAPIEA                           *
             *                                                          *
             *            PROGRAM INTERRUPT ELEMENT AREA                *
384   180    *----------------------------------------------------------*
             *                        CSASOL                            *
             *                                                          *
             *            SNAP DATA SET OPEN LIST ADDRESS               *
388   184    *----------------------------------------------------------*
             *                        CSASPLAC                          *
             *                                                          *
             *            SNAP PARAMETER LIST ADDRESS                   *
392   188    *----------------------------------------------------------*
             *  CSASLID    *                                            *
             *             *               RESERVED                     *
             * SNAP ID NO  *                                            *
396   18C    *----------------------------------------------------------*
             *                        CSASDCB                           *
             *                                                          *
             *            SNAP DATA CONTROL BLOCK ADDRESS               *
400   190    *----------------------------------------------------------*
             *                                                          *
             *                        RESERVED                          *
             *                                                          *
412   19C    *----------------------------------------------------------*
             *                        CSAICFNA                          *
             *                                                          *
             *        RUNAWAY TASK FLUSH ROUTINE ENTRY ADDR             *
416   1A0    *----------------------------------------------------------*
             *                        CSAICRNX                          *
             *                                                          *
             *            RUNAWAY TASK FLUSH ROUTINE                    *
             *                  LINKAGE CODE                            *
424   1A8    *----------------------------------------------------------*
             *                        CSATODTU                          *
             *                                                          *
             *                TIME OF DAY IN TIMER UNITS                *
428   1AC    *----------------------------------------------------------*
             *                        CSATCNDT                          *
             *                                                          *
             *            TERM CONT NEXT DISPATCH TIME OF DAY           *
432   1B0    *----------------------------------------------------------*
             *                        CSAICRIC                          *
             *                                                          *
             *                RUNAWAY TASK TIME INTERVAL                *
436   1B4    *----------------------------------------------------------*
             *      CSAICRUN         *                                  *
             *     RUNAWAY TASK      *                                  *
             *     ACCUMULATOR       *          RESERVED                *
             *                                                          *
448   1C0    *----------------------------------------------------------*
```

COMMON SYSTEM AREA

```
           --------------------------------------------------------------
           *                                                            *
Dec.  Hex. * <--------------------4 BYTES---------------------------->  *
           *                                                            *
           **************************************************************
           *                       CSAKCMTC                             *
           *                                                            *
           *               NUMBER OF TIMES AT MAX TASK                  *
452   1C4  *------------------------------------------------------------*
           *                                                            *
           *                       RESERVED                             *
           *                                                            *
456   1C8  *------------------------------------------------------------*
           *     CSAKCCT                 *        CSAKCMTA              *
           *                             *                              *
           * CURRENT TASK ACCUMULATOR    *  MAX NUMBER OF TASK          *
460   1CC  *------------------------------------------------------------*
           *                 CSAKCTTA                    *    CSASCAR   *
           *                                             *              *
           *       TASK ORIGINATED ACCUMULATOR           *              *
464   1D0  *------------------------------------------------------------*
           *                 CSASCAR                     *    CSASCFI   *
           *                 (CONT)                      *              *
           *       ACQUISITION REQUEST ACCUMULATOR       *              *
468   1D4  *------------------------------------------------------------*
           *                 CSASCFI                     *    CSASCCR   *
           *                 (CONT)                      *              *
           *                                             * CUSHION RELSE*
           *       FREEMAIN ISSUED ACCUMULATOR           *  ACCUMULATOR *
472   1D8  *------------------------------------------------------------*
           *   CSASCCR    *        CSASCRQ               * CSASCMQ      *
           *              *                              *              *
           *              * STORAGE REQUESTS QUEUED      * MAX STORAGE  *
           *   (CONT)     *     ACCUMULATOR             *REQUESTS QUEUED*
476   1DC  *------------------------------------------------------------*
           *   CSASCMQ    *       CSASCQZ                * CSAPINI      *
           *              *                              *              *
           *    (CONT)    * STORAGE REQUESTS QUEUED      *   NUMBER OF  *
           *              *  FROM ZERO                   *  INTERRUPTS  *
480   1E0  *------------------------------------------------------------*
           *   CSAPINI    *           CSADCND            *  CSATSMSA    *
           *              *                              *              *
           * RESERVED     *  NUMBER OF STORAGE DUMPS     *              *
484   1E4  *------------------------------------------------------------*
           *             CSATSMSA      *         CSATSASA               *
           *       TEMP STORAGE (CONT) *                                *
           * MAIN STRG USE ACCUMULATOR *  TEMP AUX STORAGE USE ACCUM    *
488   1E8  *------------------------------------------------------------*
           *    CSATSASA  *        CSASPA1               *  CSASPA2     *
           *              *                              *              *
           *  (CONT)      *    SERVICE PROGRAM ACCUM     * SERVICE PROG *
           *              *                              *    ACCUM     *
           *------------------------------------------------------------*
492   1EC  *    CSASPA2   *           CSASPA3                           *
           *              *                                             *
           *  (CONT)      *    SERVICE PROGRAM ACCUMULATOR              *
496   1F0  *------------------------------------------------------------*
           *                   CSATDNT                   *  CSAUTA1     *
           *                                             *              *
           *            NUMBER INTRAPARTITION TRACKS     *              *
500   1F4  *------------------------------------------------------------*
```

```
                C O M M O N   S Y S T E M   A R E A
             ------------------------------------------------------------
             *                                                          *
Dec.   Hex.  * <--------------------4 BYTES----------------------------> *
             *                                                          *
             ************************************************************
             *          CSAUTA1          *          CSAUTA2             *
             *          (CONT)           *                              *
             *                           *                              *
             *   USER TRANSACTION ACCUM  *  USER TRANSACTION ACCUM      *
504    1F8   *----------------------------------------------------------*
             *  CSAUTA2      *          CSAUTA3                         *
             *               *                                          *
             *  (CONT)       *     USER TRANSACTION ACCUM               *
508    1FC   *----------------------------------------------------------*
             *          CSAUTA4                     *                   *
             *                                      *   RESERVED        *
             *      USER TRANSACTION ACCUM          *                   *
512    200   *----------------------------------------------------------*
             *                       CSAWABA                            *
             *                                                          *
             *                      WORK AREA                           *
4096   1000  ************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
-112	-70		112	Copyright information IBM Corporation
0	0	CSAOSRSA	72	Common system register storage area
72	48	CSASOSI	1	Short-on-storage indicator
73	49	CSAKCMI	1	Maximum number of tasks indicator
74	4A	CSAKCMT	2	Maximum number of tasks control
76	4C	CSACDTA	4	Currently dispatched task address
80	50	CSATODP	4	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.
84	54	CSAICEBA	4	Interval Control Element chain beginning address
88	58	CSAICSIC	2	Stall time interval. A binary integer of which the least significant bit represents one one hundredth of a second.
90	5A		1	Reserved
91	5B	CSAICIND	1	Interval Control indicator

Bits	Setting	Function
0-7	0000 0001	Time of day adjustment mask
0-7	0000 0010	Adjustment task pending mask

<table>
<tr><th colspan="2">Displacement</th><th></th><th></th><th></th></tr>
<tr><th>Dec.</th><th>Hex.</th><th>Field</th><th>Bytes</th><th>Function</th></tr>
<tr><td>92</td><td>5C</td><td>CSATADJT</td><td>4</td><td>Time of day adjustment value. Difference between operating system time of day and CICS time of day expressed in 300ths of a second (timer units).</td></tr>
<tr><td>96</td><td>60</td><td>CSACTODB</td><td>4</td><td>Current time of day. A binary integer of which the least significant bit represents one one-hundredth of a second.</td></tr>
<tr><td>96</td><td>60</td><td>CSACSCC</td><td>4</td><td>Common system control clock</td></tr>
<tr><td>100</td><td>64</td><td>CSASBTI</td><td>4</td><td>System binary timer interval</td></tr>
<tr><td>104</td><td>68</td><td>CSATTECB</td><td>4</td><td>Terminal Time Event Control Block</td></tr>
<tr><td>108</td><td>6C</td><td>CSASITOD</td><td>4</td><td>Time of day at System Initialization. A binary integer of which the least significant bit represents one second.</td></tr>
<tr><td>112</td><td>70</td><td>CSASCNB</td><td>4</td><td>Storage cushion number of bytes</td></tr>
<tr><td>116</td><td>74</td><td>CSAPLBA</td><td>4</td><td>Partition lower boundary address</td></tr>
<tr><td>120</td><td>78</td><td>CSAPUBA</td><td>4</td><td>Partition upper boundary address</td></tr>
<tr><td>124</td><td>7C</td><td>CSAJYDP</td><td>4</td><td>Julian date. A packed integer of the form 00YYDDDC where YY is years, DDD is days, and C is a positive sign.</td></tr>
<tr><td>128</td><td>80</td><td>CSATDTCA</td><td>4</td><td>Task Dispatcher TCA address</td></tr>
<tr><td>132</td><td>84</td><td>CSATRMF1</td><td>1</td><td>Trace system master flag
<table>
<tr><th>Bits</th><th>Setting</th><th>Function</th></tr>
<tr><td>0-7</td><td>1000 0000</td><td>Master flag; if on, tracing occurs</td></tr>
<tr><td>0-7</td><td>0100 0000</td><td>System master flag; if on, system entries (ID 200-239) are traced</td></tr>
<tr><td>0-7</td><td>0010 0000</td><td>User master flag; if on, user entries (ID 0-199) are traced</td></tr>
</table></td></tr>
<tr><td>133</td><td>85</td><td>CSATRMF2</td><td>1</td><td>Trace system flag
<table>
<tr><th>Bits</th><th>Setting</th><th>Function</th></tr>
<tr><td>0-7</td><td>1000 0000</td><td>Trace Task Control</td></tr>
<tr><td>0-7</td><td>0100 0000</td><td>Trace Storage Control</td></tr>
<tr><td>0-7</td><td>0010 0000</td><td>Trace Program Control</td></tr>
</table></td></tr>
</table>

Displacement				
Dec.	Hex.	Field	Bytes	Function
				0-7 0001 0000 Trace Interval Control
				0-7 0000 1000 Trace Dump Control
				0-7 0000 0100 Trace File Control
				0-7 0000 0010 Trace Transient Data Control
				0-7 0000 0001 Trace Temporary Storage Control
134	86		10	Reserved
144	90	CSAUNQID	4	Unique identification counter
148	94	CSAAIDBA	4	Automatic Initiate Descriptor chain beginning address
152	98	CSASP0AD	4	Subpool 0 boundary box address
156	9C	CSASTCA	4	Dummy Suspended TCA starting address
160	A0		4	Reserved
164	A4	CSATCA	4	Dummy Active TCA starting address
168	A8	CSASUSFA	4	Suspended TCA forward chain address
172	AC	CSASUSBA	4	Suspended TCA backward chain address
176	B0	CSATCAFA	4	Active TCA forward chain address
180	B4	CSATCABA	4	Active TCA backward chain address
184	B8	CSATCTCA	4	Terminal Control TCA address
188	BC		12	Reserved
200	C8	CSAOPFLA	4	Optional Feature List address
204	CC		20	Reserved
224	E0	CSAKCNAC	4	Task Control entry address
228	E4	CSASCNAC	4	Storage Control entry address
232	E8	CSAPCNAC	4	Program Control entry address
236	EC	CSAICNAC	4	Interval Control entry address
240	F0	CSADCNAC	4	Dump Control entry address
244	F4	CSATCNAC	4	Terminal Control entry address
248	F8	CSAFCNAC	4	File Control entry address
252	FC	CSATDNAC	4	Transient Data Control entry address
256	100	CSATSNAC	4	Temporary Storage Control entry address

Displacement

Dec.	Hex.	Field	Bytes	Function
260	104	CSASANAC	4	PL/I Storage Allocation module entry
264	108	CSATRNAC	4	Trace Control entry address
268	10C	CSAPINAC	4	Program Interrupt entry address
272	110	CSASNNAC	4	Snap Shot program entry address
276	114		8	Reserved
284	11C	CSATRTBA	4	Trace Table beginning address
288	120	CSAPCTBA	4	Program Control Table beginning address
292	124	CSAPPTBA	4	Processing Program Table beginning address
296	128	CSATCTBA	4	Terminal Control Table line entry address
300	12C	CSAFCTBA	4	File Control Wait List/Line Entry beginning address
304	130	CSADCTBA	4	Destination Control Table beginning address
308	134	CSATSATA	4	Temporary Auxiliary Storage Table beginning address
312	138	CSATSMTA	4	Temporary Main Storage Table beginning address
316	13C	CSAQETBA	4	Queue Element Table beginning address
320	140	CSAPOLA	4	Program data set open list address
324	144	CSADOLA	4	Dump data set open list address
328	148	CSATOLA	4	Terminal data set open list address
332	14C	CSAFOLA	4	File open list
336	150	CSATDOLA	4	Transient data set open list address
340	154	CSATSOLA	4	Temporary storage data set open list address
344	158		8	Reserved
352	160	CSAPICA	6	Program Interrupt Control area
358	166		2	Reserved
360	168	CSAPIEA	24	Program Interrupt Element area
384	180	CSASOL	4	Snap Data Set Open List address

Displacement				
Dec.	Hex.	Field	Bytes	Function
388	184	CSASPLAC	4	Snap Parameter List address
392	188	CSASLID	1	Snap Identification number
393	189		3	Reserved
396	18C	CSASDCB	4	Snap Data Control Block address
400	190		12	Reserved
412	19C	CSAICFNA	4	Runaway Task Flush routine entry address
416	1A0	CSAICRNX	8	Runaway Task Flush routine linkage instructions
424	1A8	CSATODTU	4	Time of day in timer units
428	1AC	CSATCNDT	4	Terminal Control's next dispatch time of day
432	1B0	CSAICRIC	4	Runaway task time interval
436	1B4	CSAICRUN	2	Runaway task accumulator
438	1B6		10	Reserved
448	1C0	CSAKCMTC	4	Number of times at maximum number of tasks
452	1C4		4	Reserved
456	1C8	CSAKCCT	2	Current task accumulator
458	1CA	CSAKCMTA	2	Maximum number of task
460	1CC	CSAKCTTA	3	Task orignated accumulator; total number of tasks CICS has originated
463	1CF	CSASCAR	4	Number of Storage Control GETMAIN's that have been requested
467	1D3	CSASCFI	4	Number of Storage Control FREEMAIN's that have been requested
471	1D7	CSASCCR	2	Number of times cushion released
473	1D9	CSASCRQ	2	Number of storage requests queued because storage not available
475	1DB	CSASCMQ	2	Maximum number of storage requests queued at any one time because storage not available
477	1DD	CSASCQZ	2	Number of times the storage queued chain started from zero
479	1DF	CSAPINI	2	Number of program interrupts
481	1E1	CSADCND	2	Number of storage dumps

Displacement				
Dec.	Hex.	Field	Bytes	Function
483	1E3	CSATSMTA	3	Temporary main storage use accumulator
486	1E6	CSATSASA	3	Temporary auxiliary storage use accumulator
489	1E9	CSASPA1	2	Service program accumulator 1
491	1EB	CSASPA2	2	Service program accumulator 2
493	1ED	CSASPA3	3	Service program accumulator 3
496	1F0	CSATDNT	3	Number of intrapartition tracks
499	1F3	CSAUTA1	3	User transaction accumulator 1
502	1F6	CSAUTA2	3	User transaction accumulator 2
505	1F9	CSAUTA3	3	User transaction accumulator 3
508	1FC	CSAUTA4	3	User transaction accumulator 4
511	1FF		1	Reserved
512	200	CSAWABA	0-3584	User-specified work area

CSA OPTIONAL FEATURE LIST

DSECT NAME: CSAOPFL

The CSA Optional Feature List is an extension area of the CSA which is used to support optional CICS features; it contains the control addresses of related optional features. The address of the Optional Feature List is contained in the CSA at CSAOPFLA.

```
                    O P T I O N A L   F E A T U R E   L I S T
               ┌──────────────────────────────────────────────────────────────┐
               *                                                              *
Dec.    Hex.   * <--------------------4 BYTES---------------------------->    *
               *                                                              *
 0       0     ****************************************************************
               *                       CSAATP                                 *
               *                                                              *
               *        ASYNCHRONOUS TRANS PROCESSOR CSA EXT AREA ADDR        *
 4       4     *--------------------------------------------------------------*
               *                     CSAATTCH                                 *
               *                                                              *
               *                 ATTACH LIST ADDRESS                          *
 8       8     *--------------------------------------------------------------*
               *                       CSADLI                                 *
               *                                                              *
               *             DL/I INTERFACE CTL ADDRESS                       *
12       C     ****************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	CSAATP	4	Asynchronous Transaction Processor CSA extension area address
4	4	CSAATTCH	4	OS ATTACH macro list address
8	8	CSADLI	4	DL/I Interface control area address

ATP CSA EXTENSION AREA

DSECT NAME: DFHATPDS
REGISTER: CSAATBAR

The ATP CSA extension area is a block of main storage containing various statistics accumulated by the Asynchronous Transaction Control Program. If ATP is not active in the system, this area will not exist. If it does exist, it's address is contained in the CSA optional feature list field labeled CSAATP.

```
                   A T P   C S A   E X T E N S I O N   A R E A
           ┌────────────────────────────────────────────────────────────┐
           *                                                            *
Dec.  Hex. * <------------------4 BYTES----------------------------->   *
           *                                                            *
 0     0   **************************************************************
           *                   CSABCABA                                 *
           *                                                            *
           *           BCA CHAIN BEGINNING ADDRESS                      *
 4     4   *------------------------------------------------------------*
           *       CSABCMXT            *           CSABCMXB             *
           *                           *                                *
           *   MAX BATCH TASK          *       BATCH INHIBITOR          *
 8     8   *------------------------------------------------------------*
           *       BCACTIVE            *           BCATOTAL             *
           *                           *                                *
           *   NO OF ACTIVE BATCHES    *   NO OF BATCH TRANSACTIONS     *
12     C   *------------------------------------------------------------*
           *   BCATOTAL    *               BCAJOBS                      *
           *               *                                            *
           *    (CONT)     *       NO OF BATCH TRANSACTIONS             *
16    10   *------------------------------------------------------------*
           *   CSABCAI     *                                            *
           *               *               NOT USED                     *
           *    FLAGS      *                                            *
20    14   **************************************************************
```

Displacement

Dec.	Hex	Field	Bytes	Function
0	0	CSABCABA	4	Beginning address of the BCA chain
4	4	CSABCMXT	2	Maximum batch tasks
6	6	CSABCMXB	2	Batch initiation inhibit level
8	8	BCACTIVE	2	Number of currently active tasks processed
10	A	BCATOTAL	3	Total number of batch transactions processed
13	D	BCAJOBS	3	Total of batch processed
16	10	CSABCAI	1	Batch control flags

			Bits	Setting	Function
16	10	BCASTOP	0-7	1000 0000	Terminate ATP

TASK CONTROL AREA (TCA)

DSECT NAME: DFHTCADS
REGISTER: TCACBAR

The Task Control Area (TCA) is created for each task that is currently within CICS and is in existence only during the time that work exists for a task. The TCA provides no space for any residual data such as statistics. It contains control addresses and data necessary for CICS to control the task.

The TCA contents are divided into three logical sections, a CICS system control section, an application program communication section, and the optional transaction work area. The system control section contains control address and data necessary for CICS to control the task. Access to data in this area is limited to CICS management programs, and to system type user-developed programs. The application program communication section is used primarily for communication between the task and the CICS service modules. Access to this section is provided to both the CICS programs and the user-written application programs.

Appended to the TCA is the Transaction Work Area (TWA). The TWA is acquired at task initiation as part of the TCA and has the same base register as the TCA. The TWA provides the user-written program with unique storage for the duration of the task. This area may be used to pass data or address constants from one program to another within one task. The TWA must be used if parameters are passed up a logical level. The size of the TWA is specified by the user in the Program Control Table.

CICS SYSTEM CONTROL SECTION OF TCA

DSECT NAME: DFHTCADY

```
                      T A S K   C O N T R O L   A R E A
              ┌─────────────────────────────────────────────────────────┐
              *                                                         *
Dec.   Hex.   * <──────────────────4 BYTES────────────────────────────> *
              *                                                         *
-64    -40    ***********************************************************
              *                                                         *
              *   TCASAACL    *   TCASAAFI *          TCASAAD           *
              *               *            *                            *
              *    CLASS      *    FORMAT  *          STORAGE           *
              *   STORAGE     *      ID    *        DISPLACEMENT        *
-60    -3C    *---------------------------------------------------------*
              *                    TCASCCA                              *
              *                                                         *
              *       TRANSACTION STORAGE CHAIN ADDRESS                 *
-56    -38    *---------------------------------------------------------*
              *   TCAOFDI     *                                         *
              *   OPTIONAL    *             TCAKCTTA                    *
              *   FEATURE     *                                         *
              *  DEPENDENT    *        TASK IDENTIFICATION              *
              *  INDICATOR    *              NUMBER                     *
-52    -34    *---------------------------------------------------------*
```

TASK CONTROL AREA

```
            ┌─────────────────────────────────────────────────────────────┐
            *                                                             *
Dec.  Hex.  * <--------------------4 BYTES------------------------------> *
            *                                                             *
            ***************************************************************
            *                          TCATCPC                            *
            *                                                             *
            *          PROGRAM CONTROL TABLE ENTRY ADDRESS                *
-48   -30   *-------------------------------------------------------------*
            *                          TCATCQC                            *
            *                                                             *
            *          TASK CONTROL TASK QUEUE CHAIN                      *
-44   -2C   *-------------------------------------------------------------*
            *                          TCAKCQC                            *
            *                                                             *
            *          TASK QUEUE ELEMENT CHAIN ADDRESS                   *
-40   -28   *-------------------------------------------------------------*
            *                          TCARSTSK                           *
            *                                                             *
            *          RESUMED TASK CONTROL ADDRESS                       *
-36   -24   *-------------------------------------------------------------*
            *                                                             *
            *                          RESERVED                           *
            *                                                             *
-32   -20   *-------------------------------------------------------------*
            *                          TCAICEAD                           *
            *                                                             *
            *          INTERVAL CONTROL ELEMENT ADDRESS                   *
-28   -1C   *-------------------------------------------------------------*
            *                                                             *
            *                          RESERVED                           *
            *                                                             *
-24   -18   *-------------------------------------------------------------*
            *                          TCAPCTA                            *
            *                                                             *
            *          PROCESSING PROGRAM TABLE ADDRESS                   *
-20   -14   *-------------------------------------------------------------*
            *                          TCAPCSA                            *
            *                                                             *
            *          PROGRAM REGISTER STORAGE ADDRESS                   *
-16   -10   *-------------------------------------------------------------*
            *                          TCAPCPA                            *
            *                                                             *
            *          PL/I ACQUIRED AREA ADDRESS                         *
-12   -C    *-------------------------------------------------------------*

-16   -10   *-------------------------------------------------------------*
            *                          TCAPCCA                            *
            *                                                             *
            *          COBOL ACQUIRED AREA ADDRESS                        *
-12   -C    *-------------------------------------------------------------*
            *                          TCAPCLC                            *
            *                                                             *
            *          LOADED PROGRAM CHAIN ADDRESS                       *
-8    -8    *-------------------------------------------------------------*
            *                          TCAIDAA                            *
            *                                                             *
            *              INTRAPARTITION DATA AREA ADDRESS               *
-4    -4    *-------------------------------------------------------------*
            *                                                             *
            *                          RESERVED                           *
            *                                                             *
0     0     ***************************************************************
```

```
COMMUNICATION SECTION OF TCA

DSECT NAME:    DFHTCADY
REGISTER:  TCACBAR

                       T A S K   C O N T R O L   A R E A
           ┌-----------------------------------------------------------┐
           *                                                           *
Dec.  Hex. * <-------------------4 BYTES-----------------------------> *
           *                                                           *
           *************************************************************


                           COMMUNICATION SECTION
 0     0   *-----------------------------------------------------------*
           *                        TCASYAA                            *
           *                                                           *
           *                   TCA SYSTEM ADDRESS                      *
 4     4   *-----------------------------------------------------------*
           *                                                           *
           *                        RESERVED                           *
           *                                                           *
 8     8   *-----------------------------------------------------------*
           *   TCAFCI                     TCAFCAAA                     *
           *               *                                           *
           * FACILITY      *         FACILITY CONTROL                  *
           * CONTROL IND   *           AREA ADDRESS                    *
12     C   *-----------------------------------------------------------*
           *                        TCATCFA                            *
           *                                                           *
           *            PRIORITY CHAIN FORWARD ADDRESS                 *
16    10   *-----------------------------------------------------------*
           *                        TCATCBA                            *
           *                                                           *
           *               PRIORITY CHAIN BACKWARD ADDRESS             *
20    14   *-----------------------------------------------------------*
           *                        TCATCQA                            *
           *                   QUEUE NAME ADDRESS                      *
           *                                                           *
           *                        TCATCEA                            *
           *               TASK CONTROL EVENT CONTROL                  *
           *                        ADDRESS                            *
24    18   *-----------------------------------------------------------*
           *   TCATCEI    *  TCATCTR    *   TCATCDP    *   TCAPCABR    *
           *              *             *              *               *
           *EVENT COND IND*  TYPE OF    *TASK DISPTCHG *PROG CNT ABEND *
           *              *  REQUEST    *   PRIORITY   *   REQUEST     *
           *              *             *              *               *
           *   TCATCDC    *             *              *   TCAPCDMP    *
           *              *             *              *               *
           * DISPATCH IND *             *              * PGM CNT TASK  *
           *              *             *              *  DUMP IND     *
28    1C   *-----------------------------------------------------------*
           *   TCAPURGI   *   TCASVMID  *                              *
           *              *             *                              *
           *   TASK PURGE * SERVICEMOD  *           RESERVED           *
           * INDICATOR    * CONTROL ID  *                              *
32    20   *-----------------------------------------------------------*
           *                        TCATCRS                            *
           *                                                           *
           *           TASK CONTROL REGISTER STORAGE                   *
88    58   *-----------------------------------------------------------*
```

T A S K   C O N T R O L   A R E A

```
           *                                                          *
Dec. Hex.  * <-------------------4 BYTES---------------------------->  *
           *                                                          *
           ************************************************************
           *                      TCARTNSV                            *
           *                                                          *
           *              INTERNAL RETURN REG SAVE AREA               *
 92   5C   *----------------------------------------------------------*
           *                      TCASCSA                             *
           *                                                          *
           *              STORAGE CONTROL STORAGE ADDRESS             *
           *                                                          *
           *   TCASCTR     *  TCASCIB    *          TCASCNB           *
           *               *             *                            *
           * STORAGE CNTL  * STORAGE CNTL*                            *
           * TYPE REQUEST  * INIT BYTE   *  STORAGE CONTROL NUMBER BYTES*
 96   60   *----------------------------------------------------------*
           *                      TCASCRS                             *
           *                                                          *
           *              STORAGE CONTROL REGISTER STORAGE            *
128   80   *----------------------------------------------------------*


                      TASK CONTROL COMMUNICATION AREA
128   80   *----------------------------------------------------------*
           *   TCAKCRC     *                                          *
           *  TASK CONTROL *          RESERVED                        *
           *  RETURN CODE  *                                          *
152   98   *----------------------------------------------------------*
           *                      TCAKCTI                             *
           *                                                          *
           *                  TRANSACTION IDENTIFICATION              *
156   9C   *----------------------------------------------------------*
           *                      TCAKCFA                             *
           *                                                          *
           *                  FACILITY CONTROL ADDRESS                *
160   A0   *----------------------------------------------------------*


                      INTERVAL CONTROL COMMUNICATION AREA
128   80   *----------------------------------------------------------*
           *                          TCAICDA                         *
           *                      DATA ADDRESS                        *
           *              *                                           *
           *   TCAICTR    *           TCAICTEC                        *
           * TYPE REQUEST *                                           *
           *   RESPONSE   *   TIMER EVENT CONTROL AREA ADDRESS        *
132   84   *----------------------------------------------------------*
           *                      TCAICQID                            *
           *                                                          *
           *                  REQUEST IDENTIFICATION                  *
140   8C   *----------------------------------------------------------*
           *                      TCAICRT                             *
           *          REQUESTED TIME OF DAY, TIME INTERVAL            *
           *                OR EXPIRATION TIME                        *
144   90   *----------------------------------------------------------*
           *                      TCAICFA                             *
           *                  FACILITY CONTROL ADDRESS                *
           *                                                          *
           *                      TCAICTI                             *
           *                  TRANSACTION IDENTIFICATION              *
148   94   *----------------------------------------------------------*
```

TASK CONTROL AREA

```
            *                                                          *
Dec.  Hex.  * <-------------------4 BYTES--------------------------->  *
            *                                                          *
            ************************************************************
            *                         TCAICTID                         *
            *                                                          *
            *                  TERMINAL IDENTIFICATION                 *
152   98    *----------------------------------------------------------*
            *                                                          *
            *                         RESERVED                         *
            *                                                          *
160   A0    *----------------------------------------------------------*
            *                         TCAICRS                          *
            *                                                          *
            *           INTERVAL CONTROL REGISTER SAVE AREA            *
216   D8    *----------------------------------------------------------*


                     PROGRAM CONTROL COMMUNICATION AREA
128   80    *----------------------------------------------------------*
            *   TCAPCTR    *         TCAPCLA                           *
            *              *                                           *
            *   TYPE OF    *                                           *
            *REQUEST/RESP  *  LOADED PROGRAM BEGINNING ADDRESS         *
132   84    *----------------------------------------------------------*
            *                         TCAPCPI                          *
            *                                                          *
            *                  PROGRAM IDENTIFICATION                  *
140   8C    *----------------------------------------------------------*
            *                         TCAPCAC                          *
            *                  ABNORMAL TERMINAL CODE                  *
144   90    *----------------------------------------------------------*
            *                         TCAPCPSW                         *
            *                                                          *
            *           PROG INTERRUPT PROGRAM STATUS WORD             *
152   98    *----------------------------------------------------------*
            *                                                          *
            *                         RESERVED                         *
            *                                                          *
160   A0    *----------------------------------------------------------*
            *                         TCAPCRS                          *
            *                                                          *
            *           PROGRAM CONTROL REGISTER SAVE AREA             *
216   D8    *----------------------------------------------------------*


                              OPEN/CLOSE AREA
140   8C    *----------------------------------------------------------*
            *   TCAOCTR    *               TCAOCLA                     *
            *  OPEN/CLOSE  *                                           *
            *    TYPE OF   *            OPEN/CLOSE LIST                *
            *   REQUEST    *               ADDRESS                     *
144   90    *----------------------------------------------------------*
```

TASK CONTROL AREA

```
             *                                                            *
Dec.   Hex.  * <--------------------4 BYTES--------------------------->   *
             *                                                            *
             **************************************************************
```

BASIC MAPPING SUPPORT

```
 140    8C   *------------------------------------------------------------*
             *   TCABMSFB   *             *  TCABMSWC     *               *
             *              *             *               *               *
             *     FLAG     *  RESERVED   *    WRITE      *   RESERVED    *
             *     BYTE     *             *   CONTROL     *               *
             *              *             *  CHARACTER    *               *
 144    90   *------------------------------------------------------------*
             *                         TCABMSMA                           *
             *                                                            *
             *                         MAP ADDRESS                        *
|148    94   *----                                                    ----*
             *                         TCABMSMN                           *
             *                                                            *
             *                         MAP NAME                           *
 152    98   *------------------------------------------------------------*
```

DUMP CONTROL COMMUNICATION AREA

```
 128    80   *------------------------------------------------------------*
             *             TCADCTR          *           TCADCNB           *
             *                              *                             *
             *         TYPE OF REQUEST      *       NUMBER OF BYTES       *
 132    84   *------------------------------------------------------------*
             *                         TCADCSA                            *
             *                                                            *
             *                     STORAGE ADDRESS                        *
 136    88   *------------------------------------------------------------*
             *                                                            *
             *                         RESERVED                           *
             *                                                            *
 140    8C   *------------------------------------------------------------*
             *                         TCADCDC                            *
             *                                                            *
             *                    IDENTIFICATION CODE                     *
 144    90   *------------------------------------------------------------*
             *                                                            *
             *                         RESERVED                           *
             *                                                            *
 160    A0   *------------------------------------------------------------*
             *                         TCADCRS                            *
             *                                                            *
             *             DUMP CCNTROL REGISTER SAVE AREA                *
 216    D8   *------------------------------------------------------------*
```

FILE CONTROL COMMUNICATION AREA

```
 128    80   **************************************************************
             *                             TCAFCAA                        *
             *           *                                                *
             *  TCAFCTR  *          FILE AREA ADDRESS                     *
             *TYPE REQUEST *                                              *
 132    84   *------------------------------------------------------------*
```

T A S K   C O N T R O L   A R E A

```
             ┌─────────────────────────────────────────────────────────┐
             *                                                         *
Dec.   Hex.  * <-------------------4 BYTES---------------------------> *
             *                                                         *
             ***********************************************************
             *                        TCAFCDI                          *
             *                                                         *
             *               DATA SET IDENTIFICATION                   *
 140    8C   *---------------------------------------------------------*
             *                        TCAFCAI                          *
             *                                                         *
             *               INDIRECT ACCESS IDENTIFICATION            *
             *                            *                            *
             *       TCAFCURL             *                            *
             *                            *                            *
             *  UNDEFINED RECORD LENGTH   *                            *
 148    94   *---------------------------------------------------------*
             *                        TCAFCSI                          *
             *                                                         *
             *               SEGMENT SET IDENTIFICATION                *
 156    9C   *---------------------------------------------------------*
             *                        TCAFCRI                          *
             *                                                         *
             *               RECORD IDENTIFICATION ADDRESS             *
 160    A0   *---------------------------------------------------------*
             *                        TCAFCRS                          *
             *                                                         *
             *           FILE CONTROL REGISTER SAVE AREA               *
 216    D8   *---------------------------------------------------------*


                            DL/I COMMUNICATION AREA
 128    80   *---------------------------------------------------------*
             *                        TCADLIO                          *
             *                                                         *
             *                   WORK AREA ADDRESS                     *
 132    84   *---------------------------------------------------------*
             *                        TCADLPCB                         *
             *                                                         *
             *                     PCB ADDRESS                         *
 136    88   *---------------------------------------------------------*
             *                        TCADLPSB                         *
             *                                                         *
             *                        PSB NAME                         *
 144    90   *---------------------------------------------------------*
             *                        TCADLSSA                         *
             *                                                         *
             *                   SSA LIST ADDRESS                      *
 148    94   *---------------------------------------------------------*
             *                        TCADLPAR                         *
             *                                                         *
             *                   PARAMETER LIST                        *
 152    98   *---------------------------------------------------------*
             *                        TCADLLAN                         *
             *                   CALLING MODULE LANGUAGE               *
             *                        TCADLECB                         *
             *                   CICS SUBTASK ECB                      *
 156    9C   *---------------------------------------------------------*
             *                        TCADLFUN                         *
             *                                                         *
             *                     DL/I FUNCTION                       *
 160    A0   *---------------------------------------------------------*
```

TASK CONTROL AREA

```
            ┌──────────────────────────────────────────────────────────┐
            *                                                          *
Dec.  Hex.  * <-------------------4 BYTES--------------------------->  *
            *                                                          *
            ************************************************************
            *                       TCADLRS                            *
            *                                                          *
            *               DL/I REGISTER SAVE AREA                    *
216    D8   *----------------------------------------------------------*


                 TRANSIENT DATA CONTROL COMMUNICATION AREA
128    80   *----------------------------------------------------------*
            *   TCATDTR                     TCATDAA                    *
            *              *                                           *
            * TYPE REQUEST *    TRANSIENT DATA/ DATA ADDRESS           *
132    84   *----------------------------------------------------------*
            *                       TCATDDI                            *
            *                                                          *
            *               DESTINATION IDENTIFICATION                 *
136    88   *----------------------------------------------------------*
            *                                                          *
            *                       RESERVED                           *
            *                                                          *
160    A0   *----------------------------------------------------------*
            *                       TCATDRS                            *
            *                                                          *
            *           TRANSIENT DATA REGISTER STORAGE AREA           *
216    D8   *----------------------------------------------------------*


                 TEMPORARY STORAGE CONTROL COMMUNICATION AREA
128    80   *----------------------------------------------------------*
            *   TCATSTR                     TCATSDA                    *
            *              *                                           *
            * TYPE REQUEST *    TEMPORARY STORAGE DATA AREA ADDRESS    *
132    84   *----------------------------------------------------------*
            *                       TCATSDI                            *
            *                                                          *
            *           TEMPORARY DATA IDENTIFICATION                  *
140    8C   *----------------------------------------------------------*
            *                                                          *
            *                       RESERVED                           *
            *                                                          *
160    A0   *----------------------------------------------------------*
            *                       TCATSRS                            *
            *                                                          *
            *       TEMPORARY STORAGE REGISTER SAVE AREA               *
216    D8   *----------------------------------------------------------*
            *                                                          *
            *                       RESERVED                           *
            *                                                          *
224    E0   *----------------------------------------------------------*
            *                       TCATRF1                            *
            *                                                          *
            *               TRACE ENTRY DATA AREA 1                    *
228    E4   *----------------------------------------------------------*
            *                       TCATRF2                            *
            *                                                          *
            *               TRACE ENTRY DATA AREA 2                    *
232    E8   *----------------------------------------------------------*
```

TASK CONTROL AREA

```
            *                                                          *
Dec.  Hex.  * <-------------------4 BYTES-----------------------------> *
            *                                                          *
            ************************************************************
            *   TCATRTR    *  TCATRID    *   TCATRMF    *              *
            *              *             *              *              *
            *   TYPE OF    *   TRACE     *    TRACE     *   RESERVED   *
            * TRACE REQUEST* ENTRY ID    * CONTROL FLAGS*              *
 236   EC   *----------------------------------------------------------*
            *                                                          *
            *                       RESERVED                           *
            *                                                          *
 248   F8   *----------------------------------------------------------*
            *                       TCACSPE                            *
            *                                                          *
            *             HIGH LEVEL LANGUAGE CONTROL                  *
            *                 SYSTEM PROGRAM ENTRY                     *
 252   FC   *----------------------------------------------------------*
            *                       TCANXTID                           *
            *                                                          *
            *             TRANS ID OF NEXT TRANSACTION                 *
 256   100  *----------------------------------------------------------*
            *                       TWACOBA                            *
            *                                                          *
            *                TRANSACTION WORK AREA                     *
4096  1000  ************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
-64	-40	TCASAACL	1	Class of storage
-63	-3F	TCASAFFI	1	Format identification
-62	-3E	TCASADD	2	Storage displacement
-60	-3C	TCASCCA	4	Address of the first transaction storage area in the chain
-56	-38	TCAOFDI	1	Optional feature dependent indicator Code — Function 01 — DL/I dependent
-55	-37	TCAKCTTA	3	Task identification number
-52	-34	TCATCPC	4	Program Control table entry address
-48	-30	TCATCQC	4	Task Control task queue chain address
-44	-2C	TCAKCQC	4	Task Queue element chain address
-40	-28	TCARSTSK	4	Resumed task's control address
-36	-24		4	Reserved
-32	-20	TCAICEAD	4	Interval Control element address
-28	-1C		4	Reserved
-24	-18	TCAPCTA	4	Processing Program Table (PPT) address.
-20	-14	TCAPCSA	4	Program register storage address where the registers are saved on execution of a LINK macro instruction.
-16	-10	TCAPCPA	4	PL/I acquired area address
-16	-10	TCAPCCA	4	COBOL acquired area address
-12	-C	TCAPCLC	4	Loaded program chain address.

Displacement				
Dec.	Hex.	Field	Bytes	Function
				Address of load list area associated with this TCA.
-8	-8	TCAIDAA	4	Intrapartition data area address
-4	-4		4	Reserved
0	0	TCASYAA	4	TCA system area address
4	4		4	Reserved
8	8	TCAFCI	1	Facility Control indicator

Bits	Setting	Function
0-7	0000 0000	Task-dependent facility
0-7	0000 0001	Terminal facility
0-7	0000 0010	File facility
0-7	0000 1000	DCT facility

Dec.	Hex.	Field	Bytes	Function
8	8	TCAFCAAA	4	Facility Control Area address; contents related to the system facility associated with the task • Terminal-dependent task and address of associated TCTTE • Non-terminal-dependent task initiated by Transient Data; address of associated Destination Control Table • Non-terminal-dependent task initiated by Interval Control; address of associated Automatic Initiate Descriptor
12	C	TCATCFA	4	Priority chain forward address. Address of next higher priority task's TCA on active or suspended task chain.
16	10	TCATCBA	4	Priority chain backward address. Address of next lower priority task's TCA on active or suspended task chain.
20	14	TCATCQA	4	Enqueued resource name address
20	14	TCATCEA	4	Address of the Event Control Byte
24	18	TCATCEI	1	Event control indicator

Bits	Setting	Function
0-7	0001 0000	Not dispatchable
0-7	0001 0001	...ATTACH
0-7	0001 0010	...ENQ
0-7	0001 0011	...ENQ
0-7	0001 1000	Suspended by Storage Control
0-7	0001 1100	Suspended by Temp Storage
0-7	0010 0000	Dispatchable
0-7	0010 0001	Dispatchable -

Displacement

Dec.	Hex.	Field	Bytes	Function		
						Auto abend requested
				0-7	0010 0010	Dispatchable - Stall purge
				0-7	0100 0000	ECB list
				0-7	1000 0000	Single ECB
24	18	TCATCDC	1	Dispatch control indicator		
				Same bit settings as for TCATCEI		
25	19	TCATCTR	1	Type of request		
				Bits	Setting	Function
				0-7	0001 0000	ATTACH
				0-7	0010 0000	CHAP
				0-7	0100 0000	WAIT
				0-7	1000 0000	DETACH
				0-7	0000 0100	SUSPEND
				0-7	0000 1000	RESUME
				0-7	0000 0001	ENQUEUE
				0-7	0000 0010	DEQUEUE
				0-7	0001 0001	Conditional ATTACH
				0-7	0001 0010	SCHEDULE
				0-7	0001 0100	AVAIL
26	1A	TCATCDP	1	Task dispatching priority		
				Bits	Setting	Function
				0-7	0000 0000 to 1111 1111	Priority
27	1B	TCAPCABR TCAPCDMP	1	Program Control task abend request Program Control task dump request		
				Bits	Setting	Function
				0-7	1000 0000	Program to be abended
				0-7	0100 0000	Program dumped
28	1C	TCAPURGI	1	Task purge indicator		
				Bits	Setting	Function
				0-7	1000 0000	Term error purge
				0-7	0100 0000	Stall purge
29	1D	TCASVMID	1	Service module control ID and runaway task control		
				Bits	Setting	Function
				0-7	0000 0001	Runaway task time not expired
				0-7	0000 0010	System task mask
				0-7	0000 0100	Storage Control program mask

Displacement

Dec.	Hex.	Field	Bytes	Function		
				0-7	0000 1000	Trace Program
				0-7	0001 0000	Program Control program mask
				0-7	0010 0000	Dump Control program mask
				0-7	0011 0000	File Control program mask
				0-7	0100 0000	Transient Data program mask
				0-7	0101 0000	Temporary Storage program mask
				0-7	0110 0000	Interval Control program mask
30	1E		2	Reserved		
32	20	TCATCRS	56	Task Control program register storage area stores registers 14 through 11		
88	58	TCARTNSV	4	Internal return register save area		
92	5C	TCASCSA	4	Address of storage after it has been obtained by Storage Control and has been initialized to requested configuration		

Before the address is placed in field named TCASCSA, the same field contains the following information:

Displacement

Dec.	Hex.	Field	Bytes	Function		
92	5C	TCASCTR	1	Type request/response		
				Bits	Setting	Function
				0-7	1000 0000	Acquire storage
				0-7	0100 0000	Release storage
				0-7	0010 0000	Storage has been freed outside of Storage Control
				0-7	0001 0000	Initialize strg
				0-7	0101 0000	Release terminal storage
				0-7	0000 1000	Subpool indicator
				0-7	0000 0100	Chained storage
				0-7	0000 0010	TCA type storage
				0-7	0000 0001	Terminal type storage
93	5D	TCASCIB	1	Value to which storage is to be initialized; zero, blanks, etc.		
94	5E	TCASCNB	2	Number of bytes of main storage requested		
96	60	TCASCRS	32	Storage Control register storage area; stores registers 14 through 5.		

Displacement

Dec.	Hex.	Field	Bytes	Function
128	80	TCACCCA	32	Common control communication area. This segment of the TCA is overlaid by the communication area of one of the following programs when that program has control of the CPU: Task Control, Program Control, File Control, Transient Data Control, Dump Control. (See each separate overlay area on following pages.)
160	A0	TCACCRS	56	Common control register storage area. This segment of the TCA is overlaid by the register storage area of one of the following modules when that module has control of the CPU: Interval Control, Program Control, File Control, Dump Control, Temporary Storage Control, or Transient Data Control. (See each separate overlay area on following pages.)
216	D8		8	Reserved
224	E0	TCATRF1	4	Trace entry data area one
228	E4	TCATRF2	4	Trace entry data area two
232	E8	TCATRTR	1	Type of trace request
233	E9	TCATRID	1	Trace entry identification
234	EA	TCATRMF	1	TCA Trace Control flags

Bits	Setting	Function
0-7	1000 0000	Trace user request even if user master flag is off (CSA)

235	EB		1	Reserved
236	EC		12	Reserved
248	F8	TCACSPE	4	High-level language system module entry address
252	FC	TCANXTID	4	Transaction identification of next transaction on facility
256	100	TWACOBA	*	Beginning of the Transaction Work Area (TWA)

* The length of the TWA is determined by the user when he defines the appropriate PCT entry.

The following are the overlays related to the common control areas of the TCA.

TASK CONTROL COMMUNICATION AREA

Displacement

Dec.	Hex.	Field	Bytes	Function
128	80	TCAKCRC	1	Task Control return code
129	81		23	Reserved
152	98	TCAKCTI	4	Transaction identification
156	9C	TCAKCFA	4	Facility control address

INTERVAL CONTROL COMMUNICATION AREA

Displacement

Dec.	Hex.	Field	Bytes	Function
128	80	TCAICTR	1	Type of request/response

Bits	Setting	Function
0-7	0001 00xy where:	GETIME request
6	x=0	Refresh CSA only
6	x=1	Return to user-defined area
7	y=0	Binary format
7	y=1	Packed format
0-7	0010 ab00	WAIT request
0-7	0011 ab00	POST request
0-7	0100 ab0d	INITIATE request
0-7	0101 ab0d	PUT request
0-7	1000 00c0	GET request
0-7	1001 0000	RETRY - GET request
0-7	1111 0b00 where:	CANCEL
4	a=0	INTRVAL parameter given
4	a=1	TIME parameter given
5	b=0	No request ID given
5	b=1	Request ID parameter given
6	c=0	User-defined input area
6	c=1	System to provide input area
7	d=0	Task not terminal dependent
7	d=1	Terminal-dependent task

Dec.	Hex.	Field	Bytes	Function
128	80	TCAICDA	4	Data area address
128	80	TCAICTEC	4	Time event control area address

Displacement

Dec.	Hex.	Field	Bytes	Function
132	84	TCAICQID	8	Unique Request Identification
140	8C	TCAICRT	4	Requested time interval or expiration time of day Requested time-of-day response
144	90	TCAICFA	4	Facility control address
144	90	TCAICTI	4	Symbolic Transaction Identification
148	94	TCAICTID	4	Symbolic Terminal Identification
152	98		8	Reserved
160	A0	TCAICRS	56	Interval Control program register storage area; stores registers 14 through 11

PROGRAM CONTROL COMMUNICATION AREA

Displacement

Dec.	Hex.	Field	Bytes	Function
128	80	TCAPCLA	4	Beginning address of the loaded program

This field is also used prior to loading the beginning address of the load program with the following:

128	80	TCAPCTR	1	Type of request/response

Bits	Setting	Function
0-7	0000 0001	LINK
0-7	0000 0010	XCTL
0-7	0000 0100	LOAD
0-7	0000 1000	DELETE
0-7	0001 0000	RETURN
0-7	0110 0000	ABEND and DUMP
0-7	0100 0000	ABEND
0-7	1001 0000	Task Control refresh load

132	84	TCAPCPI	8	Program identification
140	8C	TCAPCAC	4	Abnormal termination code
144	90	TCAPCPSW	8	Program Interrupt Program Status Word
152	98		8	Reserved
160	A0	TCAPCRS	56	Program Control program register strg area; stores registers 14 through 11

OPEN/CLOSE COMMUNICATION AREA

Displacement

Dec.	Hex.	Field	Bytes	Function
140	8C	TCAOCTR	1	Open/close type of request/response

Type Request:

Bits	Setting	Function
0-7	1000 0000	Open
0-7	0100 0000	Close

Dec.	Hex.	Field	Bytes	Function

0-7	0010 0000	Switch
0-7	0000 0100	Data Base
0-7	0000 0010	Trans Data
0-7	0000 0001	Dump

Type Response:

Bits	Setting	Function
0-7	1111 1111	Invalid request
0-7	1000 0000	Open error
0-7	0100 0000	Close error
0-7	0010 0000	No storage available
0-7	0001 0000	Invalid table ID

140	8C	TCAOCLA	4	Open/close list address; stores registers 14 through 11

BASIC MAPPING SUPPORT COMMUNICATION AREA

Displacement

Dec.	Hex.	Field	Bytes	Function
140	8C	TCABMSFB	1	Flag byte
141	8D		1	Reserved
142	8E	TCABMSWC	1	Write control character
143	8F		1	Reserved
144	90	TCABMSMA	4	Map address
148	94	TCABMSMN	4	Map name

DUMP CONTROL COMMUNICATION AREA

Displacement

Dec.	Hex.	Field	Bytes	Function
128	80	TCADCTR	2	Type of request

Byte 1:

Bits	Setting	Function
0-7	0010 0000	Dump program storage and register storage area
	0000 1000	Dump terminal storage
0-7	0000 0100	Dump transaction storage
0-7	0000 0001	Dump segment

Byte 2:

Bits	Setting	Function
0-7	1000 0000	Dump CICS program modules
0-7	0010 0000	Dump Processing Program Table

Dec.	Hex.	Field	Bytes	Function
				0-7 0000 1000 Dump Program Control Table
				0-7 0000 0100 Dump Terminal Control Table

Bits	Setting	Function
0-7	0000 0010	Dump File Control Table
0-7	0000 0001	Dump Destination Control Table

130	82	TCADCNB	2	Number of bytes
132	84	TCADCSA	4	Dump Control storage address
136	88		4	Reserved
140	8C	TCADCDC	4	Dump identification code
144	90		16	Reserved
160	A0	TCADCRS	56	Dump Control program register save area; stores registers 14 through 11

FILE CONTROL COMMUNICATION AREA

Displacement

Dec.	Hex.	Field	Bytes	Function
128	80	TCAFCAA	4	File area address

This field is also used prior to loading the file area address.

128	80	TCAFCTR	1	Type of request/response

Bits	Setting	Function
Request codes:		
0-7	1000 0000	Read (inquiry only)
0-7	1000 1000	Read segmented record (inquiry only)
0-7	1000 0100	Read update
0-7	1000 1100	Read segmented record with update
0-7	1000 0010	Read/indirect accessing (inquiry only)
0-7	1000 1110	Read indirectly a segmented record for update
0-7	0100 0100	Write new record
0-7	0100 1100	Write new segmented record
0-7	0010 0000	Get an area
0-7	0001 0000	Release an area
0-7	0000 0001	DAM blocked records
0-7	1100 0000	OPEN request

Displacement

Dec.	Hex.	Field	Bytes	Function		
				0-7	1110 0000	CLOSE request
				0-7	1111 0000	LOCATE request
				0-7	1010 0000	SETL request
				0-7	1011 0000	GETNEXT request
				0-7	1010 0100	RESETL request
				Response codes:		
				Bits	Setting	Function
					0000 0000	Normal response
					0000 0001	No data set ID in FCT
					0000 0100	No segment cntrl entry in FCT
					0000 1000	Invalid request code
					0000 1010	Record being returned is from duplicates data set
					0000 1100	File not open
					0000 1111	End-of-file ind
					1000 0000	I/O error (refer to field FCIOERR in FIOA)
					1000 0001	No record found
					1000 0010	Duplicate record (occurs only during the write of a new record to ISAM data set)
					1000 0011	No space to add new record
129	81	TCAFCTR+1	1	Type request		
				Bits	Setting	Function
				0-7	1000 0000	Deblocking by key
				0-7	0100 0000	Deblocking by relative record
132	84	TCAFCDI	8	Data set ID		
140	8C	TCAFCAI	8	Indirect access ID		
140	8C	TCAFCURL	2	Undefined record length		
148	94	TCAFCSI	8	Segment set ID		
156	9C	TCAFCRI	4	Record ID address		
160	A0	TCAFCRS	56	File Control program register save area; stores registers 14 through 11		

DL/I COMMUNICATION AREA

Displacement				
Dec.	Hex.	Field	Bytes	Function
128	80	TCADLIO	4	Work area address
132	84	TCADLPCB	4	PCB address
136	88	TCADLPSB	8	PSB name
144	90	TCADLSSA	4	SSA list address
148	94	TCADLPAR	4	Parameter list address
152	98	TCADLLAN	4	Calling module language
152	98	TCADLECB	4	CICS subtask ECB
156	9C	TCADLFUN	4	DL/I function
160	A0	TCADLRS	56	DL/I support register save area

TRANSIENT DATA CONTROL COMMUNICATION AREA

Displacement				
Dec.	Hex.	Field	Bytes	Function
128	80	TCATDAA	4	Transient data area address

This field is also used prior to loading the transient data area address.

128	80	TCATDTR	1	Type request/response

Bits	Setting	Function
Request codes:		
0-7	1000 0000	GET
0-7	0100 0000	PUT
0-7	0010 0000	FEOV
0-7	0001 0000	LOCATE
0-7	0000 0100	PURGE
0-7	0000 0100	DCT ENTRY passed
Response codes:		
0-7	0001 0000	No space
0-7	0000 0000	Normal response
0-7	0000 0001	Queue empty
0-7	0000 0010	Dest ID error
0-7	0000 0100	I/O error

132	84	TCATDDI	4	Destination ID
136	88		24	Reserved
160	A0	TCATDRS	56	Transient data register storage area

TEMPORARY STORAGE CONTROL COMMUNICATION AREA

Displacement				
Dec.	Hex.	Field	Bytes	Function
128	80	TCATSDA	4	Temporary storage data area address

The first byte of this field is also used as follows:

128	80	TCATSTR	1	Type request/response

Bits	Setting	Function
Request codes:		
0-7	1000 0000	Get
0-7	0100 0000	Put on aux stg
0-7	0010 0000	Release
0-7	0001 0000	Address supplied on Get
Response codes:		
0-7	0000 0000	Normal response
0-7	0000 0010	Data ID not found
0-7	0000 0100	I/O error
0-7	0010 0000	Invalid request

132	84	TCATSDI	8	Temporary storage data ID
140	8C		20	Reserved
160	A0	TCATSRS	56	Temporary storage register storage area

TERMINAL CONTROL TRANSACTION WORK AREA

```
                    T R A N S A C T I O N   W O R K   A R E A
               ┌──────────────────────────────────────────────────────────────┐
               *                                                              *
Dec.   Hex.    * <-----------------------4 BYTES------------------------------> *
               *                                                              *
256    100     ****************************************************************
               *                          TCSPTA                              *
               *                                                              *
               *          SPECIFIC READ TERMINAL ENTRY ADDRESS                *
260    104     *--------------------------------------------------------------*
               *       TCPIND        *              TCERRSA                   *
               *                     *                                        *
               * POLLING INDICATOR   *      TERMINAL ERROR CODE SAVE AREA     *
264    108     *--------------------------------------------------------------*
               *                          TCTXTPA                             *
               *                                                              *
               *                     TERMINAL POOL ADDRESS                    *
268    10C     *--------------------------------------------------------------*
               *      TCTXLPAF       *              TCTXLPA                   *
               *                     *                                        *
               *    POOLED LINE      *      FIRST LINE IN POOL POINTER        *
               *   AVAILABLE FLAG    *              SAVE AREA                 *
272    110     *--------------------------------------------------------------*
               *                          TCTRNTA                             *
               *                                                              *
               *                   TRANSLATE TABLE ADDRESS                    *
276    114     *--------------------------------------------------------------*
               *                          TCL3PTSV                            *
               *                                                              *
               *       LOCAL 3270  POLLED TERMINAL   SAVE AREA                *
280    118     *--------------------------------------------------------------*
               *                          TCTSPRA                             *
               *                                                              *
               *                  SPECIFIC POLL RETURN ADDRESS                *
284    11C     *--------------------------------------------------------------*
```

TRANSACTION WORK AREA

```
               ┌──────────────────────────────────────────────────────────────┐
               *                                                              *
Dec.   Hex.    * <-------------------4 BYTES----------------------------->    *
               *                                                              *
               ****************************************************************
               *                        TCTWLA                                *
               *                                                              *
               *                ACTIVE WAIT LIST ADDRESS                      *
288    120     *--------------------------------------------------------------*
               *                                                              *
               *                   TWO RESERVED WORDS                         *
               *                                                              *
296    128     *--------------------------------------------------------------*
               *                       TWACFWD1                               *
               *                                                              *
               *                COMPATIBILITY WORK AREA 1                     *
300    12C     *--------------------------------------------------------------*
               *                       TWACFWD2                               *
               *                                                              *
               *                COMPATIBILITY WORK AREA 2                     *
304    130     *--------------------------------------------------------------*
               *                       TWACFWD3                               *
               *                                                              *
               *                COMPATIBILITY WORK AREA 3                     *
308    134     *--------------------------------------------------------------*
               *                       TWACFWD4                               *
               *                                                              *
               *                COMPATIBILITY WORK AREA 4                     *
312    138     *--------------------------------------------------------------*
               *  TWACFLAG   *              *   2260 COMPATIBILITY FIELD      *
               *  CCNTROL    *   RESERVED   *          TWAC2260               *
               *   FLAGS     *              *    NUMBER OF CHARACTERS         *
               *             *              *       PER 2260 LINE             *
316    13C     *--------------------------------------------------------------*
               *      TWAC3270              *          TWAFDLBA               *
               *  NUMBER OF CHARACTERS      *      ADDRESS OF FIRST           *
               *     PER 3270 LINE          *   COMPATIBILITY LINES           *
320    140     *--------------------------------------------------------------*
               *      TWALDLBA              *          TWAIBDL                *
               *  ADDRESS OF LAST           *    INCREMENT BETWEEN            *
               *  COMPATIBILITY LINE        *  COMPATIBILITY LINES            *
324    144     *--------------------------------------------------------------*
               *    TWACNBEO                *         TWACBAP                 *
               *  ERASE SCREEN LINE         * CURRENT COMPATIBILITY           *
               *      DATA LENGTH           *     BUFFER ADDRESS              *
328    148     *--------------------------------------------------------------*
               *      TWACLSA               *                                 *
               *  START OF CURRENT          *                                 *
               *  COMPATIBILITY LINE        *                                 *
332    14C     *----------------------------*                                 *
               *                       TCTTT                                  *
               *                                                              *
               *                TRANSLATE AND TEST TABLE                      *
586    24A     ****************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
256	100	TCSPTA	4	Specific read terminal entry address
260	104	TCPIND	1	Polling indicator

Displacement						
Dec.	Hex.	Field	Bytes	Function		
				Bits	Setting	Function
				0-7	0000 0001	Requested read indicator
				0-7	0000 0010	Polling indicator
				0-7	0000 0100	Task present indicator
261	105	TCERRSA	3	Terminal error code save area		
264	108	TCTXTPA	4	Terminal pool address		
268	10C	TCTXLPA	4	First line in pool address save area		
268	10C	TCTXLPAF	1	Line in pool available flag		
				Bits	Setting	Function
				0-7	1000 0000	A line is available in the pool
272	110	TCTRNTA	4	Address of TRT table in TWA		
276	114	TCT3PTSV	4	Polled terminal save area for local 3270		
280	118	TCTSPRA	4	Specific poll return address		
284	11C	TCTWLA	4	Active wait list address		
288	120		4	Two reserved fullwords		
296	128	TWACFWDI	4	Compatibility work area		
300	12C	TWACFWD2	4	Compatibility work area		
304	130	TWACFWD3	4	Compatibility work area		
308	134	TWACFWD4	4	Compatibility work area		
312	138	TWACFLAG	4	Compatibility control flags		
				Bits	Setting	Function
				0-7	0000 0001	Data scan complete
				0-7	0000 0010	Wrapped screen
				0-7	0000 0100	Short line found
				0-7	0000 1000	SMI found
313	139		1	Reserved		
314	13A	TWAC2260	2	Number of characters per 2260 line for compatibility		
316	13C	TWAC3270	2	Number of characters per 3270 line for compatiblity		
318	13E	TWAFDLBA	2	3270 buffer address of start of first compatibility line		
320	140	TWALDLBA	2	3270 buffer address of start of last compatibility line		
322	142	TWAIBDL	2	Number of characters between compatibility display lines		
324	144	TWACNBEO	2	Number of characters to erase a compatibility line		
326	146	TWACBAP	2	Current compatibility buffer address		
328	148	TWACLSA	2	3210 buffer address of current compatibility line		
330	14A	TCTTT	256	Translate and test table		

INTERVAL CONTROL ELEMENT (ICE)

DSECT NAME: DFHICEDS
REGISTER: ICECBAR

An Interval Control Element (ICE) is created for each time-dependent request received by the Interval Control program. These ICE's are logically chained to the CSA in expiration time-of-day sequence.

Expiration of a time-ordered request is detected by the expired request logic of the Interval Control program running as a CICS system task whenever the Task Dispatcher gains control (see Task Control program). The type of service represented by the expired ICE is initiated, providing all resources required for the service are available, and the ICE is removed from the chain. If the resources are not available, the ICE remains on the chain and another attempt to initiate the requested service is made the next time the Task Dispatcher gains control.

```
                   I N T E R V A L   C O N T R O L   E L E M E N T
              ┌──────────────────────────────────────────────────────────────┐
              *                                                              *
Dec.    Hex.  * <----------------------4 BYTES-----------------------------> *
              *                                                              *
 0       0    ****************************************************************
              *                                                              *
              *                     STORAGE ACCOUNTING                       *
              *                                                              *
 4       4    *--------------------------------------------------------------*
              *                          ICECHNAD                            *
              *                                                              *
              *                  NEXT CHAINED ICE ADDRESS                    *
 8       8    *--------------------------------------------------------------*
              *                          ICETRMID                            *
              *                  TERMINAL IDENTIFICATION                     *
              *                                                              *
              *                          ICETCAAD                            *
              *                     TASK'S TCA ADDRESS                       *
12       C    *--------------------------------------------------------------*
              *                          ICETRNID                            *
              *                 TRANSACTION IDENTIFICATION                   *
              *                                                              *
              *                          ICETECA                             *
              *                  TIMER EVENT CONTROL AREA                    *
16      10    *--------------------------------------------------------------*
              *  ICETYPE    *  ICESTATI  *                                   *
              *  TYPE OF    *   STATUS   *            RESERVED               *
              *    ICE      *  INDICATOR *                                   *
20      14    *--------------------------------------------------------------*
              *                          ICEXTOD                             *
              *                                                              *
              *                  EXPIRATION TIME OF DAY                      *
24      18    *--------------------------------------------------------------*
              *                          ICERQID                             *
              *                                                              *
              *             REQUEST (AND DATA) IDENTIFICATION                *
32      20    ****************************************************************
```

Displacement				
Dec.	Hex.	Field	Bytes	Function
0	0		4	Storage Accounting Area
4	4	ICECHNAD	4	Address of next ICE on chain
8	8	ICETRMID	4	Symbolic Terminal Identification
8	8	ICETCAAD	4	Requesting task's TCA address
12	C	ICETRNID	4	Symbolic Transaction Identification
12	C	ICETECA	4	Task's assigned Timer Event Control Area
16	10	ICETYPE	1	Type of Interval Control Element

Bits	Setting	Type of ICE
0-7	0010 0000	WAIT
0-7	0011 0000	POST
0-7	0100 0000	INITIATE
0-7	0101 0000	PUT

17	11	ICESTATI	1	Status indicator

Bits	Setting	Status
0-7	0000 0001	Currently on chain
0-7	0000 1000	Expiration time dependent
0-7	0001 0000	Cancelled by another task
0-7	0010 0000	Expired at time of original request
0-7	1000 0000	Normal expiration

18	12		2	Reserved
20	14	ICEXTOD	4	Expiration time of day
24	18	ICERQID	8	Unique Request Identification

AUTOMATIC INITIATE DESCRIPTOR (AID)

DSECT NAME: DFHAIDDS
REGISTER: AIDCBAR

The automatic task initiation services of CICS employs a queuing technique in synchronizing the tasks initiation with the availability of their respective terminal destinations. An Automatic Initiator Descriptor (AID) is created for each request for automatic task initiation and is added to a chain of AIDs. This request is dependent upon the availability of a terminal. This chain is sequenced by symbolic Transaction Identfication within symbolic Terminal Identification. The CSA contains the address of the top of the AID chain.

When an AID is added to the chain, the Task Control program advises Terminal Control of an automatically initiated task pending on a particular terminal. This is done by setting an indicator in the associated Terminal Control Table terminal entry. Terminal Control advises the Task Control program when the particular terminal facility is available by issuing an AVAIL system macro instruction. The Task Control program initiates the ATTACH request for the new task. The Interval Control program passes expired Interval Control Elements (ICEs) that represent either time-ordered task initiation requests or time-ordered data records. Task Control uses these for AIDs. If a time-ordered data record has been retained for the new task, the AID remains on the chain until the time-initiated task issues a request (GET) for the data record or terminates. The AID is removed from the chain at the time the task is initiated if no time-ordered data record was associated with the original request (INITIATE).

AUTOMATIC INITIATE DESCRIPTOR

```
                  ┌─────────────────────────────────────────────────────────┐
                  *                                                         *
Dec.      Hex.    * <---------------------4 BYTES---------------------------> *
                  *                                                         *
 0         0      ***********************************************************
                  *                                                         *
                  *                   STORAGE ACCOUNTING                    *
 4         4      *---------------------------------------------------------*
                  *                       AIDCHNAD                          *
                  *                                                         *
                  *                NEXT CHAINED AID ADDRESS                 *
 8         8      *---------------------------------------------------------*
                  *                       AIDTRMID                          *
                  *                                                         *
                  *                TERMINAL IDENTIFICATION                  *
12         C      *---------------------------------------------------------*
                  *                       AIDTRNID                          *
                  *                                                         *
                  *               TRANSACTION IDENTIFICATION                *
16        10      *---------------------------------------------------------*
                  *   AIDTYPE     *  AIDSTATI    *                          *
                  *               *              *         RESERVED         *
                  * TYPE OF AID   * STATUS IND   *                          *
20        14      *---------------------------------------------------------*
                  *                       AIDICTOD                          *
                  *                                                         *
                  *                 EXPIRATION TIME OF DAY                  *
24        18      *---------------------------------------------------------*
                  *                       AIDICDID                          *
                  *                                                         *
                  *                  DATA IDENTIFICATION                    *
32        20      ***********************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		4	Storage Accounting Area
4	4	AIDCHNAD	4	Address of next AID on chain
8	8	AIDTRMID	4	Symbolic Terminal Identification
12	C	AIDTRNID	4	Symbolic Transaction Identification
16	10	AIDTYPE	1	Type of AID

Bits	Setting	Type of AID
0-7	0100 0000	INITIATE request
0-7	0101 0000	PUT data request

17	11	AIDSTATI	1	Status indicator

Bits	Setting	Status
0-7	0000 0001	Task initiated

18	12		2	Reserved
20	14	AIDICTOD	4	Expiration time of day

24   18   AIDICDID   8   Data identification

PROGRAM CONTROL TABLE (PCT)

DSECT NAME: DFHPCTDS
REGISTER: PCTCBAR

The Program Control Table (PCT) provides the facility for the user to describe control information to be used by Terminal Control for identifying and initializing a new transaction. A portion of each entry is used to accumulate transaction statistics.

The PCT is generated and maintained by the user. This table is required by CICS to verify and control each transaction. The PCT is used to verify the incoming transaction and supply initial transaction information.

```
                   P R O G R A M   C O N T R O L   T A B L E

           *                                                           *
Dec.   Hex.* <--------------------4 BYTES----------------------------> *
           *                                                           *
 0      0  *************************************************************
           *                       PCTTI                               *
           *                                                           *
           *              TRANSACTION IDENTIFICATION                   *
 4      4  *-----------------------------------------------------------*
           *                   PCTTIA                  *   PCTTA       *
           *                                           *               *
           **                 RESERVED                 *               *
 8      8  *-----------------------------------------------------------*
           *       PCTTA                *  PCTTPA      *   PCTTSKA     *
           *                            *              *               *
           *       (CONT)               *-             *               *
           *   TRANSACTION ACCUMULATOR  *TRANS PRIORITY*               *
12      C  *-----------------------------------------------------------*
           *       PCTTSKA              *          PCTTWA              *
           *                            *                              *
           *       (CONT)               *                              *
           *  TRANSACTION SECURITY KEY  * TRANSACTION WORK AREA SIZE   *
16     10  *-----------------------------------------------------------*
           *                       PCTIPIA                             *
           *                                                           *
           *               INITIAL PROGRAM IDENTIFICATION              *
24     18  *-----------------------------------------------------------*
           *   PCTFLAG    *           PCTSPA           *               *
           * TRANSACTION  *                            *               *
           * FLAG IND     *    STALL PURGE ACCUMULATOR *               *
28     1C  *--------------*----------------------------*               *
           *                                                           *
           *                       RESERVED                            *
           *                                                           *
32     20  *************************************************************
```

Displacement				
Dec.	Hex.	Field	Bytes	Function
0	0	PCTTI	4	Transaction identification
4	4	PCTTIA	3	Reserved
7	7	PCTTA	3	Transaction accumulator
10	A	PCTTPA	1	Transaction priority
11	B	PCTTSKA	3	Transaction security key
14	E	PCTTWA	2	Transaction Work Area size
16	10	PCTIPIA	8	Initial program identification
24	18	PCTFLAG	1	Transaction flag indicator

Bits	Setting	Function
0-7	1000 0000	Terminal error purge
0-7	0100 0000	Stall purge
0-7	0010 0000	Format mode of compatibility
0-7	0011 0000	Fullbuf mode of compatibility

25	17	PCTSPA	2	Stall purge accumulator
27	19		5	Reserved

PROCESSING PROGRAM TABLE (PPT)

DSECT NAME: DFHPPTDS
REGISTER: PPTCBAR

The Processing Program Table (PPT) provides a means for the user to describe the control information concerning his processing programs to Program Control. In addition, Program Control will use portions of each table entry to retain certain information used to maintain control of the user's programs and to capture specified program statistics.

The PPT is generated and maintained by the user. This table is required by CICS to verify and control each program as it is loaded and released from main storage. The PPT is used to verify the program identification and to retain information relative to the program's location in the library and in main storage.

```
                     P R O C E S S I N G   P R O G R A M   T A B L E
                  ┌─────────────────────────────────────────────────────────────┐
                  *                                                             *
Dec.     Hex.     * <-------------------4 BYTES-----------------------------> *
                  *                                                             *
 0        0       ***************************************************************
                  *                         PPTPI                               *
                  *                                                             *
                  *               PROGRAM IDENTIFICATION                        *
 8        8       *-------------------------------------------------------------*
                  *                        PPTDASA                              *
                  *                                                             *
                  *                      DASD ADDRESS                           *
12        C       *-------------------------------------------------------------*
                  *                        PPTCSA                               *
                  *                                                             *
                  *                  MAIN STORAGE ADDRESS                       *
16       10       *-------------------------------------------------------------*
```

```
                          P R O C E S S I N G   P R O G R A M   T A B L E
                     ┌──────────────────────────────────────────────────────────────┐
                     *                                                              *
Dec.      Hex.       * <-------------------------4 BYTES----------------------------> *
                     *                                                              *
                     ****************************************************************
                     *          PPTSAR             *                                *
                     *                             *         RESERVED               *
                     *STORAGE AREA REQUIRED BY PGM*                                 *
20        14         *--------------------------------------------------------------*
                     *   PPTTLR    *            PPTUCC                              *
                     *             *                                                *
                     * TYPE PROGRAM *                                               *
                     *    IND      *         PROGRAM STATISTICS                     *
24        18         *--------------------------------------------------------------*
                     *       PPTENTD               *           PPTRCC               *
                     *                             *                                *
                     *   ENTRY POINT DISPLACEMENT *    RESIDENT CONTROL COUNTER    *
28        1C         *--------------------------------------------------------------*


                                          COBOL EXTENSION
28        1C         *--------------------------------------------------------------*
                     *       PPTCCR                *           SAVE AREA            *
                     *                             *                                *
                     *    SIZE OF TGT              *   SIZE OF TGT +16              *
                     *                             *           SAVE AREA            *
32        20         *--------------------------------------------------------------*
                     *                                       PPTCOTGT               *
                     *             *                                                *
                     * TYPE PROGRAM *                                               *
                     *   IND        * TGT ADDRESS IN ORIGINAL COBOL PROGRAM         *
36        24         *--------------------------------------------------------------*
                     *       PPTCCBLL              *                                *
                     *                             *           RESERVED             *
                     * DISPLACEMENT TO FIRST BLL   *                                *
                     *         CELL                *                                *
40        28         ****************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	PPTPI	8	Program ID
8	8	PPTDASA	4	DASD address
12	C	PPTCSA	4	Main storage address
16	10	PPTSAR	2	Storage area required by program
18	12	PPTRLDSR	2	Storage area required by RLD
20	14	PPTTLR	1	Type program indicator

Bits	Setting	Function
0-7	0100 0000	COBOL program initialized
0-7	0010 0000	PL/I program
0-7	0001 0000	COBOL program
0-7	0000 1000	Permanently core resident
0-7	0000 0100	Temporarily core resident
0-7	0000 0001	Non-Reusable program

Dec.	Hex.	Field	Bytes	Function
21	15	PPTUCC	3	Program statistics

24	18	PPTENTD	2	Entry point displacement
26	1A	PPTRCC	2	Resident control counter

COBOL EXTENSION

Displacement

Dec.	Hex.	Field	Bytes	Function
28	1C	PPTCCR	2	Size of TGT
30	1E	PPTCOTP	2	Size of TGT+PGT+16 save area
32	20	PPTCOTGT	4	TGT address in original COBOL program
36	24	PPTCOBLL	2	Displacement to first BLL cell
38	26		2	Reserved

TERMINAL CONTROL TABLE (TCT)

The Terminal Control Table (TCT) contains line and terminal information to provide the necessary control information for Terminal Control, plus the space to retain certain desired statistical information.

Note:

Line Entry Lengths	Hex
Start/stop non-switched lines	54
Start/stop first switched lines	58
Start/stop all switched lines (except first line)	54
Bisynchronous lines (all)	6C

Terminal Entry Lengths	
Start/stop	44
Bisynchronous terminals:	
Basic	54
3270	+14
3270 Compatibility	+ C
3735	+ 4

TERMINAL CONTROL TABLE LINE ENTRY (TCTLE)

DSECT NAME: DFHTCTLE
REGISTER: TCTLEAR

```
                         T E R M I N A L   C O N T R O L   T A B L E
                                     L I N E   E N T R Y
                   ┌─────────────────────────────────────────────────────────────┐
                   *                                                             *
Dec.     Hex.      * <--------------------4 BYTES---------------------------->   *
                   *                                                             *
-12      -C        ***************************************************************
                   *                                                             *
                   *                     BISYNC INLIST AREA 1                    *
                   *                                                             *
-8       -8        *-------------------------------------------------------------*
                   *                                                             *
                   *                     BISYNC INLIST AREA 2                    *
                   *                                                             *
-4       -4        *-------------------------------------------------------------*
```

TERMINAL CONTROL TABLE
LINE ENTRY

```
                 ┌───────────────────────────────────────────────────────────┐
                 *                                                           *
Dec.     Hex.    * <-----------------------4 BYTES-----------------------> *
                 *                                                           *
                 *************************************************************
                 *                                                           *
                 *                   BISYNC INLIST AREA 3                    *
                 *                                                           *
  0        0     *-----------------------------------------------------------*


                              7770 MESSAGE ADDRESS LIST
 -8       -8     *-----------------------------------------------------------*
                 *                                                           *
                 *                 7770 READY MESSAGE ADDRESS                *
                 *                                                           *
 -4       -4     *-----------------------------------------------------------*
                 *                                                           *
                 *                 7770 ERROR MESSAGE ADDRESS                *
                 *                                                           *
  0        0     *************************************************************
                 *                         TCTLEECB                          *
                 *                                                           *
                 *                    EVENT CONTROL BLOCK                    *
  4        4     *-----------------------------------------------------------*
                 *          TCTLETOP          *          TCTLEIOL            *
                 *                            *                              *
                 *     TYPE OF OPERATION      *   INPUT/OUTPUT DATA LENGTH   *
  8        8     *-----------------------------------------------------------*
                 *                         TCTLEDCB                          *
                 *                                                           *
                 *                        DCB ADDRESS                        *
 12        C     *-----------------------------------------------------------*
                 *                         TCTLEIOA                          *
                 *                                                           *
                 *                INPUT/OUTPUT AREA ADDRESS                  *
 16       10     *-----------------------------------------------------------*
                 *                                                           *
                 *           ACCESS METHOD EXTENSION OVERLAY AREA            *
                 *                                                           *
                 *                (SEE DESCRIPTIONS BELOW)                   *
 48       30     *-----------------------------------------------------------*
                 *   TCTLESI     *    TCTLEMI   *         TCTLEAL            *
                 *               *              *                            *
                 * LINE STATUS   *  MULTIPLE    *                            *
                 * INDICATOR     *  INDICATOR   *          INPUT             *
                 *               *    BYTE      *     DATA AREA LENGTH       *
 52       34     *-----------------------------------------------------------*
                 *                         TCTLERA                           *
                 *                                                           *
                 *               INPUT AREA ADDRESS RETENTION                *
 56       38     *-----------------------------------------------------------*
                 *                         TCTLENP                           *
                 *                                                           *
                 *                  NUMBER OF POLLS ISSUED                   *
 60       3C     *-----------------------------------------------------------*
                 *                         TCTLEBC                           *
                 *                                                           *
                 *NEGATIVE POLL *    BYPASS CONTROL COUNTER                  *
                 *  TIME DELAY  *                                            *
 64       40     *-----------------------------------------------------------*
```

TERMINAL CONTROL TABLE
LINE ENTRY

```
            ┌─────────────────────────────────────────────────────────────┐
            *                                                             *
Dec.  Hex.  * <--------------------4 BYTES-----------------------------> *
            *                                                             *
            ***************************************************************
            *   TCTLELF                        TCTLEPLA                   *
            *              *                                              *
            * LINE FEATURES*            POLLING LIST ADDRESS              *
 68    44   *-------------------------------------------------------------*
            *                          TCTLETEA                           *
            *                                                             *
            *           ACTIVE TERMINAL TABLE ENTRY ADDRESSES             *
 72    48   *-------------------------------------------------------------*
            *   TCTLETT      *   TCTLECL     *       TCTLELE              *
            *                *               *                            *
            * TYPE TRANS-    * LINE CLASS    *   NUMBER OF TRANSMISSION   *
            *   LATION       *               *          ERRORS            *
 76    4C   *-------------------------------------------------------------*
            *                          TCTLEECA                           *
            *                                                             *
            *               LINE ERROR CHAIN ADDRESS                      *
 80    50   *-------------------------------------------------------------*
            *   TCTLELEC     *                TCTLEPP                     *
            *                *                                            *
            *  LINE ERROR    *        PREVIOUS POLLING LIST PRINTER       *
            *     COUNT      *                                            *
 84    54   *-------------------------------------------------------------*
            *   TCTLEAB      *                 TCTLEPA                    *
            * LINE ENTRY     *                                            *
            * ANSWER BACK    *            TERMINAL POOL ADDRESS           *
            *  INDICATOR     *                                            *
 88    58   *-------------------------------------------------------------*
            *                          TCTLEBAA                           *
            *                                                             *
            *                    BI-SYNC AUXILIARY AREA                   *
 96    60   *-------------------------------------------------------------*
            *       TCTLEBRA         *  TCTLEBTO         *   TCTLEBEI     *
            *                        *                   *                *
            *   BISYNC RESPONSE      * LAST BISYNC       *   BISYNC EVENT*
            *       I/O AREA         * TYPE OF OPERATION *   INDICATORS   *
100    64   *-------------------------------------------------------------*
            *   TCTLEDI   *  TCTLEIBS*                                    *
            *                                       RESERVED              *
            * TEMPORARY   *INDEX BYTE*                                    *
            * DELAY IND   * SAVE AREA*                                    *
104    68   *-------------------------------------------------------------*
            *                                                             *
            *                          RESERVED                           *
            *                                                             *
108    6C   *-------------------------------------------------------------*


                           BASIC SEQUENTIAL ACCESS METHOD
 16    10   *-------------------------------------------------------------*
            *                          TCTLEIOB                           *
            *                                                             *
            *                          RESERVED                           *
 20    14   *-------------------------------------------------------------*
            *                          TCTLESID                           *
            *                                                             *
            *                  BSAM INPUT DCB ADDRESS                     *
 24    18   *-------------------------------------------------------------*
```

TERMINAL CONTROL TABLE
LINE ENTRY

```
                 *-------------------------------------------------------------*
                 *                                                             *
Dec.     Hex.    * <--------------------4 BYTES------------------------------> *
                 *                                                             *
                 ***************************************************************
                 *                         TCTLESOD                            *
                 *                                                             *
                 *                 BSAM OUTPUT DCB ADDRESS                     *
28       1C      *-------------------------------------------------------------*
                 *                                                             *
                 *                         RESERVED                            *
                 *                                                             *
48       30      *-------------------------------------------------------------*


                            TELECOMMUNICATION ACCESS METHOD
16       10      *-------------------------------------------------------------*
                 *   TCTLESB     *               *          TCTLETRC           *
                 *               *               *                             *
                 *    FIRST      *    SECOND     *                             *
                 *  SENSE BYTE   *  SENSE BYTE   *       RESIDUAL COUNT        *
20       14      *-------------------------------------------------------------*
                 *   TCTLECC     *         TCTLETLA                            *
                 *               *                                             *
                 * COMMAND CODE  *         TERMINAL LIST ADDR                  *
24       18      *-------------------------------------------------------------*
                 *   TCTLESF     *   TCTLERLN    *   TCTLERSP    *             *
                 *               *               *               *             *
                 *               *   RELATIVE    *  RESPONSE TO  * RESPONSE TO *
                 * STATUS FLAGS  * LINE NUMBER   *  ADDRESSING   * VRC/LRC     *
28       1C      *-------------------------------------------------------------*
                 *   TCTLETPO    *   TCTLEES     *          TCTLECSW           *
                 *               *               *                             *
                 *  TP OP CODE   * ERROR STATUS*            CSW STATUS         *
32       20      *-------------------------------------------------------------*
                 *                         TCTLEALP                            *
                 *                                                             *
                 *                CURRENT ADDRESSING LIST POINTER              *
36       24      *-------------------------------------------------------------*
                 *                         TCTLEPLP                            *
                 *                                                             *
                 *                CURRENT POLLING LIST POINTER                 *
40       28      *-------------------------------------------------------------*
                 *                               *          TCTLEOL            *
                 *           RESERVED            *                             *
                 *                               *       OUTPUT LENGTH         *
44       2C      *-------------------------------------------------------------*
                 *                         TCTLEOA                             *
                 *                                                             *
                 *                OUTPUT AREA ADDRESS                          *
48       30      *-------------------------------------------------------------*


               G R A P H I C S   A C C E S S   M E T H O D   E X T E N S I O N
16       10      *-------------------------------------------------------------*
                 *   TCTLEEGC    *                 *        TCTLEGRC           *
                 * LENGTH OR READ*    RESERVED     *   RESIDUAL COUNT IF       *
                 *   ERROR CODE  *                 *        READ ERROR         *
20       14      *-------------------------------------------------------------*
```

TERMINAL CONTROL TABLE
LINE ENTRY

```
                 ┌──────────────────────────────────────────────────────────┐
                 *                                                          *
Dec.    Hex.     * <-------------------4 BYTES---------------------------> *
                 *                                                          *
                 ************************************************************
                 *                        TCTLELGC                          *
                 *                                                          *
                 *                INPUT/OUTPUT DATA LENGTH                  *
24      18       *----------------------------------------------------------*
                 *                                                          *
                 *                        RESERVED                          *
                 *                                                          *
28      1C       *----------------------------------------------------------*
                 *   TCTLEDGC     *  TCTLEGLR    *                          *
                 *                *              *                          *
                 *   INDEX TO     *  LOCK OPTION *         RESERVED         *
                 *   DEB TABLE    *  REQUEST     *                          *
                 *   ADDR PTR     *              *                          *
32      20       *----------------------------------------------------------*
                 *                                                          *
                 *                        RESERVED                          *
                 *                                                          *
48      30       ************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
-12	-C		4	Bisync inlist area 1 Input area address for message response
-8	-8		4	Bisync inlist area 2 Input area length for message response
-4	-4		4	Bisync inlist area 3 Input area for message response
				7770 Message Address List
-8	-8		4	7770 Ready message address
-4	-4		4	7770 Error message address
0	0	TCTLEECB	4	Event control block
4	4	TCTLETOP	2	Type of operation

	Bits	Setting	Function
TCBOTRTI	0-7 8-15	0000 0000 0000 0001	Read initial
TCBOTRIR	0-7 8-15	0000 0000 1000 0001	Read initial with reset
TCBOTWTI	0-7 8-15	0000 0000 0000 0010	Write initial op code
TCBOTWIR	0-7 8-15	0000 0000 1000 0010	Write initial with reset
TCBOTRTT	0-7	0000 0000 0000 0011	Read continue
TCBOTWTT	0-7 8-15	0000 0000 0000 0100	Write continue

Displacement

Dec.	Hex.	Field	Bytes	Function		
		TCBOTRTV		0-7 8-15	0000 0000 0000 0101	Read conversational
		TCBOTWTV		0-7 8-15	0000 0000 0000 0110	Write conversational
		TCBOTTA		0-7 8-15	0000 0000 0000 1000	Write positive acknowledgement
		TCBOTTN		0-7 8-15	0000 0000 0000 1010	Write negative acknowledgement
		TCBOTTL		0-7 8-15	0000 0000 0000 1100	Write at line address
		TCBOTTS		0-7 8-15	0000 0000 0000 1110	Write erase
		TCBOTTRV		0-7 8-15	0000 0000 0001 0001	Read interrupt
		TCBOTRTC		0-7 8-15	0000 0000 0001 0001	Read connect
		TCBOTTIO		0-7 8-15	0000 0000 0000 1100	Write initial optical
		TCBOTTR		0-7 8-15	0000 0000 0000 1010	Write end of transmission
		TCBOTTVO		0-7 8-15	0000 0000 0000 0010	Write conversational optical
		TCBOTTQ		0-7 8-15	0000 0000 0000 0110	Write inquiry
		TCBOTTD		0-7 8-15	0000 0000 0001 0000	Write disconnect
		TCBOTWTC		0-7 8-15	0000 0000 0001 1100	Write connect
6	6	TCTLEIOL	2	Input/output data length		
8	8	TCTLEDCB	4	DCB address		
12	C	TCTLEIOA	4	Terminal input/output area address		

Terminal Control Table Line Entry Extension

Displacement

Dec.	Hex.	Field	Bytes	Function		
16	10		32	Access method extension overlay area		
48	30	TCTLESI	1	Line status indicator		
				Bits	Setting	Function
		TCTLESOS		0-7	0000 0001	Line out of service
		TCTLESLI		0-7	0000 0010	Line initiated

Displacement						
Dec.	Hex.	Field	Bytes	Function		
		TCTLESTR		0-7	0000 0100	Terminal read initiated
		TCTLESLC		0-7	0000 1000	Switched line connected
		TCTLESIR		0-7	0001 0000	Interruptable read initiated
		TCTLESAK		0-7	0010 0000	Dial line acknowledge ind
		TCTLESER		0-7	0100 0000	Error pending retry indicator
		TCTLESEP		0-7	1000 0000	Error pending ind
49	31		1	Multiple indicator byte		
				Bits	Setting	Function
		TCTLEMI				Access method ind
		TCTLEASA		0-7	0000 0001	Sequential Access Method
		TCTLEAGA		0-7	0000 0010	Local 2260 line
		TCTLEATA		0-7	0000 0100	Telecommunication Access Method
		TCTLEETI				Error Task
		TCTLEMET		0-7	0000 1000	initiate indicator
		TCTLEPI				First pool line
		TCTLEMFP		0-7	0001 0000	indicator
		TCTLEBUI				Bisync line in
		TCTLEMLU		0-7	0010 0000	use indicator
		TCTLEWL				Wrap list
		TCTLEMWL		0-7	0100 0000	
		TCTLELPB				Last in pool
		TCTLELPI		0-7	1000 0000	indicator
50	32	TCTLEAL	2	Input data area length		
52	34	TCTLERA	4	Input area address retention		
56	38	TCTLENP	4	Number of polls issued		
60	3C	TCTLEBC	1	Negative poll time delay value		
			3	Bypass control counter		
64	40	TCTLELF	1	Line features		
				Bits	Setting	Function
		TCTLEFAA		0-7	0000 0001	Auto answer feature
		TCTLEFAC		0-7	0000 0010	Auto call feature
		TCTLEFAP		0-7	0000 0100	Auto poll feature
		TCTLEFBR		0-7	0000 1000	Buffer receive feature
		TCTLEFCK		0-7	0001 0000	Checking feature
		TCTLEFSC		0-7	0010 0000	Station control feature
		TCTLEFWL		0-7	0100 0000	Wrap list feature
		TCTLEFLO		0-7	1000 0000	Lock option feature

Displacement				
Dec.	Hex.	Field	Bytes	Function
64	40	TCTLEPLA	4	Polling list address
68	44	TCTLETEA	4	Active terminal table entry address
72	48	TCTLETT	1	Type translation

	Bits	Setting	Function
TCTLETXT	0-7	0000 0001	Text translation indicator
TCTLECOR	0-7	0000 0010	Correspondence translation indicator
TCTLEBTC			Bisync translate
TCTLEMTT	0-7	0000 1000	Code indicator
TCTLETTB	0-7	0001 0000	7770 ABB'
TCTLETTC	0-7	0010 0000	7770 ABC
TCTLEBAC			Bisync ASCII
TCTLEMUT	0-7	1000 0000	Translate code indicator

73	49	TCTLECL	1	Line Class

	Bits	Settings	Function
TCTLECCV	0-7	0000 0001	Conversational
TCTLECB	0-7	0000 0010	Batch
TCTLECV	0-7	0000 0100	Video
TCTLECHC	0-7	0000 1000	Hard copy
TCTLECBS	0-7	0001 0000	Bisynchronous
TCTLECA	0-7	0100 0000	Audio

74	4A	TCTLELE	2	Number of transmission error (packed decimal)
76	4C	TCTLEECA	4	Line error chain address
80	50	TCTLELEC	1	Line error count (packed decimal)
81	51	TCTLEPP	3	Previous polling list pointer
84	54	TCTLEAB	1	Line entry answerback indicator

	Bits	Setting	Function
TCTLEAAB	0-7	0000 0001	TWX automatic answerback
TCTLETAB	0-7	0000 0010	Terminal answerback
TCTLEXIV	0-7	0000 0100	Expanded ID verification

85	55	TCTLEPA	3	Terminal pool address
88	58	TCTLEBAA	8	Bisync auxiliary area
96	60	TCTLEBRA	2	Bisync response I/O area
98	62	TCTLEBTO	1	Last Bisync type of operation
99	63	TCTLEBEI	1	Bisync event indicators

	Bits	Setting	Function
TCTLEBRI	0-7	0000 0001	Bisync RUI sent indicator
TCTLEBBI	0-7	0000 0010	Bisync blocked input indicator
TCTLEBET	0-7	0000 0100	Bisync EOT received indicator
TCTLEBTQ	0-7	0000 1000	Bisync Write inquiry

Displacement

Dec.	Hex.	Field	Bytes	Function		
						indicator
		TCTLEBTU		0-7	0001 0000	Bisync Time out ind
		TCTLEBWS		0-7	0010 0000	Bisync Wack sent indicator
		TCTLEBEP		0-7	0100 0000	Bisync error pending indicator
		TCTLEBEM		0-7	1000 0000	Bisync error message indicator
100	64	TCTLEDI	1	Temporary Delay Indicator		
				Bits	Setting	Function
		TCBSTDI		0-7	0000 0001	Delay indicator
		TCBSDR		0-7	0000 0010	Error disconnect request
		TCBSWB		0-7	0000 0100	Error write break request
		TCBSFDI		0-7	0000 1000	Error disconnect indicator
101	65	TCTLEIBS	1	Index byte save area		
102	66		6	Reserved		

For further details, see the publication "IBM System/360 Disk Operating System Basic Telecommunication Access Method", GC30-5001.

Basic Sequential Access Method Extension

Displacement

Dec.	Hex.	Field	Bytes	Function
16	10	TCTLEIOB	4	IOB address
20	14	TCTLESID	4	BSAM input DCB address
24	18	TCTLESOD	4	BSAM output DCB address
28	1C		20	Reserved

Telecommunication Access Method Extension

Displacement

Dec.	Hex.	Field	Bytes	Function
16	10	TCTLESB	1	First sense byte
17	11		1	Second sense byte
18	12	TCTLETRC	2	Residual count
20	14	TCTLECC	1	Command code
21	15	TCTLETLA	3	Terminal list address
24	18	TCTLESF	1	Status flags
25	19	TCTLERLN	1	Relative line number
26	1A	TCTLERSP	1	Response to addressing
27	1B	TCTLELRC	1	Response to VRC/LRC
28	1C	TCTLETPO	1	TP op code
29	1D	TCTLEES	1	Error status
30	1E	TCTLECSW	2	CSW status
32	20	TCTLEALP	4	Current addressing list pointer
36	24	TCTLEPLP	4	Current polling list pointer
40	28		2	Reserved

42	2A	TCTLEOL	2	Output length
44	2C	TCTLEOA	4	Output area address

See the publication "IBM System/360 Operating System Control Blocks", GC28-6628.

Graphics Access Method Extension

Displacement

Dec.	Hex.	Field	Bytes	Function
16	10	TCTLEEGC	4	Length error or read error code
17	11		1	Reserved
18	12	TCTLEGRC	2	Residual count if length error
20	14	TCTLELGC	4	Input/output data length
24	18		4	Reserved
28	1C	TCTLEDGC	1	Index to DEB table address pointer
29	1D	TCTLEGLR	1	Lock option request
30	1E		2	Zeros
32	20		16	Reserved

Note: Refer to IBM System/360 Operating System, Graphic Programming Services for IBM 2260 Display Station (Local Attachment), GC27-6912.

TERMINAL CONTROL TABLE TERMINAL ENTRY (TCTTE)

DSECT NAME: DFHTCTTE
REGISTER: TCTTEAR

```
               T E R M I N A L   C O N T R O L   T A B L E
                         T E R M I N A L   E N T R Y
           ┌────────────────────────────────────────────────────────────┐
           *                                                            *
Dec.   Hex.* <--------------------4 BYTES-----------------------------> *
           *                                                            *
 0      0  **************************************************************
           *                       TCTTETI                              *
           *                                                            *
           *               TERMINAL IDENTIFICATION                      *
 4      4  *------------------------------------------------------------*
           *   TCTTELT     *               TCTTETA                      *
           *               *                                            *
           *LAST TERMINAL  *                                            *
           *   INDICATOR   *           TERMINAL ADDRESS                 *
 8      8  *------------------------------------------------------------*
           *   TCTTETT     *  TCTTETM    *  TCTTETP     *   TCTTETS     *
           *               *             *              *               *
           *  TERMINAL     *  TERMINAL   *  TERMINAL    *   TERMINAL    *
           *    TYPE       *   MODEL     *  PRIORITY    *    STATUS     *
12      C  **************************************************************
           *               TCTTENI                      *   TCTTENO     *
           *                                            *               *
           *           NUMBER OF INPUTS                 *               *
16     10  *------------------------------------------------------------*
           *           TCTTENO           *         TCTTETE              *
           *                             *                              *
           *           (CONT)            *                              *
           *       NUMBER OF OUTPUTS     * NUMBER OF TRANSMISSION ERRORS*
20     14  *------------------------------------------------------------*
```

TERMINAL CONTROL TABLE
TERMINAL ENTRY

```
           ┌-----------------------------------------------------------┐
           *                                                           *
Dec.   Hex.* <--------------------4 BYTES----------------------------> *
           *                                                           *
           *************************************************************
           *                   TCTTEOI                 *  TCTTESK      *
           *                                           *               *
           *           OPERATOR IDENTIFICATION         *               *
24     18  *-----------------------------------------------------------*
           *           TCTTESK          *   TCTTEOP    *  TCTTEOT      *
           *           (CONT)           *              *               *
           *       OPERATOR SECURITY    *   OPERATOR   *               *
           *              KEY           *   PRIORITY   *               *
28     1C  *-----------------------------------------------------------*
           *           TCTTEOT          *         TCTTEOE              *
           *           (CONT)           *                              *
           *  NUMBER OF TRANSACTIONS    * NUMBER OF TRANSACTION ERRORS *
32     20  *-----------------------------------------------------------*
           *                       TCTTESC                             *
           *                                                           *
           *           TERMINAL STORAGE CHAIN ADDRESS                  *
36     24  *-----------------------------------------------------------*
           *                       TCTTEDA                             *
           *                                                           *
           *           ACTIVE TERMINAL DATA AREA ADDRESS               *
40     28  *-----------------------------------------------------------*
           *                       TCTTECA                             *
           *                                                           *
           *               TASK CONTROL AREA ADDRESS                   *
44     2C  *-----------------------------------------------------------*
           *   TCTTEOS     *   TCTTECS  * TCTTETEC     *  TCTTECL      *
           *               *            *              *               *
           *   EXTERNAL    *   EXTERNAL * TERMINAL     *  OPERATION    *
           *  OPERATION    *   CONTROL  * ERROR COUNT  *   CLASS       *
           *    STATUS     *    STATUS  *              *               *
48     30  *-----------------------------------------------------------*
           *                       TCTTETC                             *
           *                                                           *
           *           TERMINAL TRANSACTION CODE                       *
52     34  *-----------------------------------------------------------*
           *                       TCTTEBC                             *
           *                                                           *
           *           TERMINAL BYPASS CONTROL COUNTER                 *
56     38  *-----------------------------------------------------------*
           *       TCTTEVSS             *         TCTTETEL             *
           *                            *                              *
           *   VIDEO SCREEN SIZE        *      TABLE ENTRY LENGTH      *
60     3C  *-----------------------------------------------------------*
           *   TCTTEIO     *   TCTTEEN  * TCTTERC      *  TCTTEURC     *
           *               *            *              *               *
           *   INTERNAL    *  POLL LIST * TERMINAL     * USER RETURN   *
           *   OPERATION   *   ENTRY    *  ERROR       *    CODE       *
           *     STATUS    *   NUMBER   *RETRY COUNTER *               *
64     40  *-----------------------------------------------------------*
           *                       TCTTEDES                            *
           *                                                           *
           *                  TCAM DESTINATION NAME                    *
68     44  *-----------------------------------------------------------*
           *                       TCTTECIA                            *
           *                                                           *
           *                 POINTER TO USER AREA                      *
72     48  *-----------------------------------------------------------*
```

TERMINAL CONTROL TABLE
TERMINAL ENTRY

```
           *                                                            *
Dec.   Hex.* <--------------------4 BYTES-----------------------------> *
           *                                                            *
           **************************************************************
           *                       TCTTEBIA                             *
           *                                                            *
           *              BLOCKED INPUT RECORD ADDRESS                  *
76     4C  *------------------------------------------------------------*
           *           TCTTEBDL          *  TCTTEBES    *   TCTTEBWA    *
           *                             *              *               *
           *       BISYNC DATA AREA      * BISYNC EVENT *NUMBER OF WACKS*
           *            LENGTH           *  INDICATORS  *  TO ABEND     *
80     50  *------------------------------------------------------------*
           *   TCTTEBNA    *   TCTTETAB  *  TCTTEPCF    *   TCTTESID    *
           *               *             *              *               *
           *NUMBER STORAGE *  2980 TAB   *  2980 PASS   * 2980 STATION  *
           *AREAS PER TRANS*   FACTOR    * BOOK CONTROL *      ID       *
84     54  *------------------------------------------------------------*
           *   TCTTEBAA    *   TCTTENSA  *  TCTTETID    *   TCTTEFLG    *
           *               *             *              *               *
           *2980 ALTERNATE *  2980 NORMAL*  2980 TELLER *  2980 CONTROL *
           *   ADDRESS     *   ADDRESS   *     ID       *     FLAGS     *
88     58  *------------------------------------------------------------*
           *                       TCTTEBDA                             *
           *                                                            *
           *               BLOCKING DATA AREA ADDRESS                   *
92     5C  *------------------------------------------------------------*
           *   TCTTEDOS    *   TCTTEAID  *         TCTTECAD             *
           *               *             *                              *
           *   DISPLAY     *  ATTENTION  *                              *
           *  OPERATION    * IDENTIFIER  *   CURSOR ADDRESS IN BINARY   *
           *   STATUS      *             *                              *
96     60  *------------------------------------------------------------*
           *   TCTTEFIB    *             *         TCTTELSV             *
           *               *             *                              *
           *   TERMINAL    *   RESERVED  *  TERMINAL DATA LENGTH        *
           *   FEATURE     *             *      RETENTION               *
           *   INDICATOR   *             *                              *
100    64  *------------------------------------------------------------*
           *                       TCTTEBMN                             *
           *                                                            *
           *           NAME OF FORMAT IMAGE IN BUFFER                   *
108    6C  *------------------------------------------------------------*
           *   TCTTECTT    *   TCTTECTM  *  TCTTECFG    *   TCTTECSS    *
           *               *             *              *               *
           *  COMPATIBLE   * COMPATIBLE  * COMPATIBLE   *  COMPATIBLE   *
           *   TERMINAL    *  TERMINAL   *   FLAGS      *     SCREEN    *
           *     TYPE      *    MODEL    *              *      SIZE     *
112    70  *------------------------------------------------------------*
           *       TCTTECSM              *  TCTTERTT    *   TCTTERMN    *
           *                             *              *               *
           *   SMI BINARY POSITION       *REAL TERMINAL *REAL TERMINAL  *
           *                             *   MODEL      *    MODEL      *
116    74  *------------------------------------------------------------*
           *                       TCTTECPI                             *
           *                                                            *
           *           COMPATIBLE PRINTER IDENTIFICATION                *
120    78  *------------------------------------------------------------*
```

TERMINAL CONTROL TABLE
TERMINAL ENTRY

```
              ┌─────────────────────────────────────────────────────────────┐
              *                                                             *
Dec.    Hex.  * <------------------4 BYTES-------------------------------> *
              *                                                             *
              ***************************************************************


                                  3735 EXTENSION
88      58    *-----------------------------------------------------------*
              *   TCTTEMCI     *              TCTTEDMP                      *
              *                *                                            *
              *  3735 MODE     *         DATA RETENTION AREA                *
              * CONTROL IND    *                                            *
              *-----------------------------------------------------------*
```

Displacement						
Dec.	Hex.	Field	Bytes	Function		
0	0	TCTTETI	4	Terminal Identification; user-specified physical destination		
4	4	TCTTELT	1	Last terminal indicator		
				Bits	Setting	Function
		TCTTESF		0-7	0000 0001	Skip flag
		TCTTECSF		0-7	0000 0001	status indicator
		TCTTECRS		0-7	0000 0010	Terminal read skip indicator
		TCTTECTC		0-7	0000 0100	Terminal connected indicator
				Bits	Setting	Function
		TCTTECSP		0-7	0000 1000	Specific poll
		TCTTECPB				Compatible terminal flag
		TCTTECPF		0-7	0001 0000	Compatible terminal indicator
		TCTTEWCB				Control character supplied flag
		TCTTEWCI		0-7	0100 0000	Control character supplied indicator
		TCTTECLT		0-7	1000 0000	Last terminal in group indicator
5	5	TCTTETA	3	Terminal address; defines the physical address and terminal device specification neccessary for the WRITE macro instruction		
8	8	TCTTETT	1	Terminal type		
				Bits	Setting	Function
		TCTTETSD		0-7	0001 0010	Sequential disk
		TCTTET77		0-7	0000 0001	7770
		TCTTES7		0-7	0000 0010	System 7
		TCTTETMT		0-7	0001 0100	Magnetic tape
		TCTTETCR		0-7	0001 1000	Card reader/line printer
		TCTTETHC		0-7	0010 0000	Hard copy terminals
		TCTTETWX		0-7	0010 0001	Model 33/35 TWX

Displacement						
Dec.	Hex.	Field	Bytes	Function		
		TCTTET50		0-7	0010 0100	1050
		TCTTET30		0-7	0010 0110	1030
		TCTTET40		0-7	0010 1000	2740
		TCTTET4C		0-7	0010 1010	2741 Correspondence
		TCTTET4E		0-7	0010 1011	2741 EBCDIC
		TCTTETV0		0-7	0100 0000	Video terminals
		TCTTET6L		0-7	0100 0001	Local 2260
		TCTTET6R		0-7	0100 1000	Remote 2260
		TCTTET53		0-7	0100 1010	1053
		TCTTET65		0-7	0100 1100	2265
		TCTTETBI		0-7	1000 0000	Bisync
		TCTTET80		0-7	1000 0100	2780
		TCTTET70		0-7	1000 0010	2770
		TCTTE298		0-7	1000 0110	2980
		TCTTET35		0-7	1000 1000	3735
		TCTTET37		0-7	1001 0001	Remote 3277
		TCTTET75		0-7	1001 0010	Remote 3275
		TCTTET84		0-7	1001 0011	Remote 3284
		TCTTET86		0-7	1001 0100	Remote 3286
		TCTTETL7		0-7	1001 1001	Local 3277
		TCTTETL4		0-7	1001 1011	Local 3284
		TCTTETL6		0-7	1001 1100	Local 3286
		TCTTETPD		0-7	1010 0000	Bisync-programmable
		TCTTES3		0-7	1010 0001	System 3
		TCTTE20		0-7	1010 0010	Model 20
		TCTTE360		0-7	1010 0011	System 360
		TCTTE370		0-7	1010 0100	System 370
		TCTTE113		0-7	1010 0101	1130
9	9	TCTTETM	1	Terminal model number		
				Bits	Setting	Function
		TCTTESCN		0-7	0010 0000	2980 shift scan flag
10	A	TCTTETP	1	Terminal priority (hex 00 to FF)		
11	B	TCTTETS	1	Terminal status		
				Bits	Setting	Function
		TCTTESOS		0-7	0000 0001	Out of service
		TCTTESTA		0-7	0000 0010	Terminal attended
		TCTTESAT		0-7	0000 0100	Automatic transaction initiate
		TCTTESNP		0-7	0000 1000	No polling
		TCTTESRP		0-7	0001 0000	Reduced polling
		TCTTESPO		0-7	0010 0000	Permanent out of service
		TCTTESRO		0-7	0100 0000	Read only Dummy
		TCTTEATP		0-7	1000 0000	TCTTE Indicator (ATP)
12	C	TCTTENI	3	Number of inputs for terminal (packed decimal)		
15	F	TCTTENO	3	Number of outputs for terminal (packed decimal)		
18	12	TCTTETE	2	Number of transmission errors for terminal (packed decimal)		
20	14	TCTTEOI	3	Operator ID (user specified)		
23	17	TCTTESK	3	Operator security key		

Displacement

Dec.	Hex.	Field	Bytes	Function
26	1A	TCTTEOP	1	Operator priority
27	1B	TCTTEOT	3	Number of valid transactions for terminal (packed decimal)
30	1E	TCTTEOE	2	Number of transaction errors for terminal (packed decimal)
32	20	TCTTESC	4	Terminal storage chain address of first terminal type storage area for any one task
36	24	TCTTEDA	4	Active terminal data area address
40	28	TCTTECA	4	Task Control Area (TCA) address TCAM Input Queue TCTTE may have a pointer here to an output TCTTE that has had an unsolicited input error associated with it
44	2C	TCTTEOS	1	External operation status

Field	Bits	Setting	Function
TCTTEWR TCTTEOWR	0-7	0000 0001	Write request
TCTTEOIU TCTTEOOI	0-7	0000 0010	Optical image unit request
TCTTESR TCTTEOSR	0-7	0000 0100	Synchronization request
TCTTEDC TCTTEODR	0-7	0000 1000	Disconnect request
TCTTERR TCTTEORR	0-7	0001 0000	Read request
TCTTELA TCTTEOLA	0-7	0010 0000	Line addressing request
TCTTESTS TCTTEOSS	0-7	0100 0000	Save terminal storage request
TCTTEER TCTTEOER	0-7	1000 0000	Erase request

Dec.	Hex.	Field	Bytes	Function
45	2D	TCTTECS	1	External Control Request Byte

Field	Bits	Setting	Function
TCTTERLO TCTTEORL	0-7	0001 0000	Read lock request byte and indicator
TCTTEWLO TCTTEOWL	0-7	0010 0000	Write lock request byte and indicator
TCTTEEUB TCTTEEUI	0-7	0100 0000	Erase all unprotected bytes and indicators
TCTTERBB TCTTERBI	0-7	1000 0000	Read buffer request bytes and indicators
TCTTECYB			Copy request flag

Displacement						
Dec.	Hex.	Field	Bytes	Function		
		TCTTECYI		0-7	0000 1000	Copy request ind
		TCTTEPBM		0-7	0000 0001	Pseudo-binary mode
		TCTTETRM		0-7	0000 0010	Transparent mode
		TCTTENTR		0-7	0000 0010	Notranslate request
46	2E	TCTTETEC	1	Terminal error count (packed decimal)		
47	2F	TCTTECL	1	Operation Class Codes		
				Bits	Setting	Function
		TCTTECCV		0-7	0000 0001	Conversational term
		TCTTECB		0-7	0000 0010	Batch terminal
		TCTTECV		0-7	0000 0100	Video terminal
		TCTTECHC		0-7	0000 1000	Hard copy terminal
		TCTTECBS		0-7	0001 0000	Bisynchronous term
		TCTTECAI		0-7	0010 0000	Auto input transaction iniate
		TCTTECAU		0-7	0100 0000	Audio terminal
48	30	TCTTETC	4	Terminal transaction code		
52	34	TCTTELEA	4	Line Entry Address		
52	34	TCTTEBC	4	Terminal Bypass Control counter		
56	38	TCTTEVSS	2	Video screen size (binary)		
58	3A	TCTTETEL	2	Length of this TCTTE in bytes		
60	3C	TCTTEIO	1	Internal operation status		
				Bits	Setting	Function
		TCTTEAK TCTTEOAK		0-7	0000 0001	Write acknowledgement
		TCTTEKI TCTTEOTI		0-7	0000 0010	Task to be initiated
		TCTTEIC TCTTEOIC		0-7	0000 0100	Time control transaction initiation ind
		TCTTEGA TCTTEOGA		0-7	0000 1000	Graphics attention
		TCTTEAT TCTTEOAT		0-7	0010 0000	Automatic transaction initiate
		TCTTEAO TCTTEOAO		0-7	0100 0000	Automatic output message
		TCTTENS TCTTEONR		0-7	1000 0000	Negative response
61	3D	TCTTEEN	1	Poll list entry number		
62	3E	TCTTERC	1	Terminal retry count (packed decimal)		
		TCTTETCM		TCAM OPTCD flag		
63	3F	TCTTEURC	1	User return code		
		TCTTECR		Conditional return request		

Displacement						
Dec.	Hex.	Field	Bytes	Function		
				Bits	Setting	Function
		TCTTEOFR		0-7	0010 0000	End of File request ind
		TCTTEOFC		0-7	0000 0010	End of File Condition ind
64	40	TCTTEDES	4	TCAM destination name		
68	44	TCTTECIA	4	Pointer to user area		
		TCTTECIL		Length of user area		
72	48	TCTTEBIA	4	Blocked input record address		
76	4C	TCTTEBDL	2	Bisync data area length		
78	4E	TCTTEBES	1	Bisync event indicators		
				Bits	Setting	Function
		TCTTEBIB		0-7	0000 0001	Bisync incomplete batch
		TCTTEBRI		0-7	0000 0010	Bisync retry
		TCTTEBBI		0-7	0000 0100	Bisync blocked input
		TCTTEBUB		0-7	0000 1000	Bisync user deblocking
		TCTTEBEI		0-7	0001 0000	Bisync end of input
		TCTTEBIW		0-7	0010 0000	Bisync immed WACK feature
		TCTTEBAI		0-7	0100 0000	Bisync Read or Write Abort ind
79	4F	TCTTEBWA	1	Number of WACKS to ABEND (packed decimal)		
80	50	TCTTEBNA	1	Number of storage areas per transaction		
81	51	TCTTETAB	1	2980 Tab factor		
82	52	TCTTEPCF	1	2980 Pass book control		
				Bits	Setting	Function
		TCTTEPCR		0-7	1000 0000	Passbook present on read
		TCTTEPCW		0-7	0100 0000	2980 passbook control
83	53	TCTTESID	1	2980 Station identification		
84	54	TCTTEBAA	1	2980 Alternate address		
85	55	TCTTENSA	1	2980 Normal address		
86	56	TCTTETID	1	2980 Teller identification		
87	57	TCTTEFLG	1	2980 Control flags		
				Bits	Setting	Function
		TCTTEXLT		0-7	0000 0001	Data translate flag
		TCTTEAAI		0-7	0000 0010	2980 station addr in use
		TCTTEPBI		0-7	0000 0100	Passbook inserted on poll
		TCTTESEG		0-7	0000 1000	2980 Segmented write
		TCTTEB96		0-7	0001 0000	Buffer expansion flag
		TCTTEPBK		0-7	0010 0000	2980 Passbook request
		TCTTEWKF		0-7	0100 0000	Work factor
		TCTTECBW		0-7	1000 0000	Common buffer write request
88	58	TCTTEBDA	4	Blocking data area address		
92	5C	TCTTEDOS	1	Display operation status		

Displacement						
Dec.	Hex.	Field	Bytes	Function		
				Bits	Setting	Function
		TCTTEDBB				Device busy flag
		TCTTEDBI		0-7	1000 0000	Device busy indicator
		TCTTEPSB				Pending status message flag
		TCTTEPSI		0-7	0100 0000	Pending status ind
		TCTTERLB				Read length
		TCTTERLI		0-7	0010 0000	Saved indicators
		TCTTEICB				Incomplete message flag
		TCTTEICI		0-7	0001 0000	Incomplete message indicator
		TCTTERKB				Keyboard restore request
		TCTTERKI		0-7	0000 1000	Keyboard restore indicator
		TCTTEWLB				Write length saved
		TCTTEWLI		0-7	0000 0100	Write length saved
93	5D	TCTTEAID	1	Attention identifier		
94	5E	TCTTECAD	2	Cursor address in binary		
96	60	TCTTEFIB		Terminal feature indicator byte		
				Bits	Setting	Function
		TCTTEFPA		0-7	0000 0001	Print adapter feature
		TCTTEFSP		0-7	1000 0000	Selector pen feature
		TCTTEFAA		0-7	0000 0100	Audible alarm feature
		TCTTEFCV		0-7	0000 1000	Copy valid feature
97	61		1	Reserved		
98	62	TCTTELSV	2	Terminal data length retention		
100	64	TCTTEBMN	8	Name of format image in buffer		
108	6C	TCTTECTT	1	Compatible terminal type		
109	6D	TCTTECTM	1	Compatible terminal model		
110	6E	TCTTECFG	1	Compatibility flags		
				Bits	Setting	Function
		TCTTECMF		0-7	1000 0000	Compatible mode flag
		TCTTECPZ		0-7	0010 0000	Print flag
111	6F	TCTTECSS	1	Compatible screen size		
				Bits	Setting	Function
		TCTTEC24		0-7	1000 0000	6X40/240/2260
		TCTTEC48		0-7	0100 0000	12X40/480/2260
		TCTTEC96		0-7	0010 0000	12X80/960/2260
		TCTTEC15		0-7	0001 0000	15X64/960/2265
		TCTTEC12		0-7	0000 1000	12X40/480/3270
		TCTTEC19		0-7	0000 0100	24X80/1920/3270
		TCTTECFB		0-7	0000 0001	Fullbuff mode flag
112	70	TCTTECSM	2	SMI binary position		
114	72	TCTTERTT	1	Real terminal type		
115	73	TCTTERMN	1	Real terminal model		

Displacement

Dec.	Hex.	Field	Bytes	Function
116	74	TCTTECPI	4	Compatible printer identification

3735 Extension

Displacement

Dec.	Hex.	Field	Bytes	Function		
88	58	TCTTEMCI	1	3735 mode control indicator		
				Bits	Setting	Function
		TCTTEMCO		0-7	0000 0000	Initialization image
		TCTTEMBR		0-7	0000 0001	Batch Read
		TCTTEMBW		0-7	0000 0010	Batch Write
		TCTTEMTC		0-7	0000 0100	Transmission complete
		TCTTEMEF		0-7	0000 1000	End of File
		TCTTEMSF		0-7	0001 0000	Error status
		TCTTEMGI		0-7	0010 0000	Getmain
		TCTTEMIQ		0-7	0100 0000	Inquiry
89	59	TCTTEDMP	3	Data retention area		

FILE CONTROL TABLE (FCT)

DSECT NAME: DFHFCTDS
REGISTER: FCTDSBAR

The File Control Table (FCT) is used to describe the data sets accessed by the File Control program. Each entry specifies the types of services which are to be allowed for a data set (file), indicates the kind of access method used to get or put a record, and describes the record. Included as an appendage to each FCT entry is the DCB for that data set and the indirect access or segment control extensions where applicable. The File Control Table is created during System Generation. It can be recreated any time an entry is added to the table or when an entry in the table is modified.

```
                        F I L E   C O N T R O L   T A B L E
            ┌─────────────────────────────────────────────────────────────┐
            *                                                             *
Dec.  Hex.* <--------------------4 BYTES-----------------------------> *
            *                                                             *
  0     0   ***************************************************************
            *                       FCTDSID                               *
            *                                                             *
            *                DATA SET IDENTIFICATION                      *
  8     8   *-------------------------------------------------------------*
            *   FCTDSVR1    *   FCTDSVR2   *          FCTDSTEL             *
            *               *              *                               *
            *DATA SET CNTL  *DATA SET CNTL*      TABLE ENTRY LENGTH       *
 12     C   *-------------------------------------------------------------*
```

FILE CONTROL TABLE

```
           ┌───────────────────────────────────────────────────────────┐
           *                                                           *
Dec.   Hex.* <--------------------4 BYTES---------------------------> *
           *                                                           *
           *************************************************************
12     C   *----------------------------*
           *          FCTDSSCD          *
           *                            *
           * SEGMENT TABLE DISPLACEMENT *
16     10  *----------------------------*

12     C   *----------------------------*
           *          FCTDSIAD          *
           *                            *
           *      INDIRECT ACCESS       *
           *    TABLE DISPLACEMENT      *
16     10  *----------------------------*

12     C                                *------------------------------*
                                        *          FCTDSBLK            *
                                        *                              *
                                        *          BLOCK SIZE          *
16     10  *-----------------------------------------------------------*
           *          FCTDSREC          *          FCTDSRKP            *
           *                            *                              *
           *        RECORD LENGTH       *    RELATIVE KEY POSITION     *
20     14  *-----------------------------------------------------------*
           *  FCTDSKL     *             *  FCTDSTB1    *   FCTDSTB2    *
           *              *             *              *               *
           * KEY LENGTH   *  RESERVED   *PSEUDO TRAFFIC*PSEUDO TRAFFIC *
           *              *             *    BYTE 1    *    BYTE 2     *
24     18  *-----------------------------------------------------------*
           *                                                           *
           *                         RESERVED                          *
           *                                                           *
40     28  *-----------------------------------------------------------*
           *                    FCTDSRD                  *   FCTDSWRA  *
           *                                             *             *
           *           STATISTICS-READ REQUESTS          *             *
44     2C  *-----------------------------------------------------------*
           *      FCTDSWRA (CONT)       *          FCTDSWRU            *
           *         STATISTICS-        *                              *
           *      WRITE ADD REQUESTS    * STATISTICS - UPDATE REQUESTS *
48     30  *-----------------------------------------------------------*
           *  FCTDSWRU    *               FCTDSOFL                     *
           *  (CONT)      *                                            *
           *              *      STATISTICS OVERFLOW RECORDS           *
52     34  *-----------------------------------------------------------*
           *                         FCTDSDCB                          *
           *                                                           *
           *                     BEGINNING OF DCB                      *
           *************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	FCTDSID	8	Data set ID
8	8	FCTDSVR1	1	Data set control indicator 1

Displacement						
Dec.	Hex.	Field	Bytes	Function		
				Bits	Setting	Function
		FCTDSRI	1	0-7	1000 0000	Read valid
		FCTDSWI	1	0-7	0100 0000	Write valid
		FCTDSUPD	1	0-7	0010 0000	Update valid
		FCTDSADD	1	0-7	0001 0000	Add valid
		FCTDSISM	1	0-7	0000 1000	ISAM data set
		FCTDSBDM	1	0-7	0000 0100	BDAM data set
		FCTBRWSE	1	0-7	0000 0010	Browsing valid
		FCTDSOPN	1	0-7	0000 0001	Open/Close ind
9	9	FCTDSVR2	2	Data set control indicator 2		
				Bits	Setting	Function
		FCTDSEXC	1	0-7	1000 0000	Exclusive cntrl
		FCTDSIAI	1	0-7	0100 0000	Indirect accessing
		FCTDSVLI	1	0-7	0000 1000	Variable length records
		FCTSFLI	1	0-7	0000 0100	Fixed length records
		FCTDSNBK	1	0-7	0000 0010	Record blocking
		FCTDSKEY	1	0-7	0000 0001	DAM keyed records
10	A	FCTDSTEL	2	Table entry length		
12	C	FCTDSSCD	2	Segment table displacement		
12	C	FCTDSIAD	2	Indirect access table displacement		
14	E	FCTDSBLK	2	Block size		
16	10	FCTDSREC	2	Record length		
18	12	FCTDSRKP	2	Relative key position		
20	14	FCTDSKL	1	Key length		
21	15		1	Reserved		
22	16	FCTDSTB1	1	Pseudo traffic byte 1		
23	17	FCTDSTB2	1	Pseudo traffic byte 2		
24	18		16	Reserved		
40	28	FCTDSRD	3	Statistics Read requests		
43	2B	FCTDSWRA	3	Statistics Write add requests		
46	2E	FCTDSWRU	3	Statistics Update requests		
49	31	FCTDSOFL	3	Statistics Overflow records		
52	34	FCTDSDCB	Variable	Beginning of the DCB		

SEGMENT CONTROL TABLES

The Segment Control Tables provide the capability to define for File Control the necessary control information to render the segment control service. This requires that the user define through these tables only the record segments required to execute a given transaction. In this way, storage not needed during processing can be saved and, if desired, operating with fixed format data from a variable format data set can be accomplished.

File Control Table Segment Header (FCTSH)

DSECT NAME: DFHFCTSH
REGISTER: FCTSHBAR

```
               F I L E   C O N T R O L   T A B L E   S E G M E N T
                               H E A D E R
           ┌------------------------------------------------------------┐
           *                                                            *
Dec.   Hex.* <--------------------4 BYTES-----------------------------> *
           *                                                            *
 0      0  **************************************************************
           *           FCTSHSSD          *         FCTSHSSL             *
           *                             *                              *
           *  SEGMENT SET BEGIN DISPL    *  SEGMENT SET ENTRY LENGTH    *
 4      4  *--------------------------------------------*****************
           *   FCTSHHD     *   FCTSHID   *   FCTSHIFT   *
           *               *             *              *
           *HEADER SEGMENT*  IND DISPL   *  TYPE OF IND *
           *   LENGTH      * IN HEADER   *   INDICATOR  *
 8      8  **********************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	FCTSHSSD	2	Segment set begin displacement
2	2	FCTSHSSL	2	Segment set entry length
4	4	FCTSHHD	1	Header segment length
5	5	FCTSHID	1	Indicator displacement in header
6	6	FCTSHIFT	1	Type of segment indicators being used

Bits	Setting	Function
0-7	0000 0000	Bit type segment indicator
0-7	1000 0000	Displacement type segment indicator

File Control Table Segment Definition (FCTSD)

DSECT NAME: DFHFCTSD
REGISTER: FCTSDBAR

```
                   F I L E   C O N T R O L   T A B L E
                   S E G M E N T   D E F I N I T I O N
           ┌------------------------------------------------------------┐
           *                                                            *
Dec.   Hex.* <--------------------4 BYTES-----------------------------> *
           *                                                            *
 0      0  **************************************************************
           *   FCTSDST     *   FCTSDSL   *
           *               *             *
           *  SEG CHARS    * SEG LENGTH  *
 4      4  *******************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	FCTSDST	1	Segment characteristics

Bits	Setting	Function
0-3	1000 0000	Fixed-length segment
0-3	0100 0000	Variable-length segment
4-7	0000 0000	Byte alignment
4-7	0000 0001	Halfword alignment
4-7	0000 0011	Fullword alignment
4-7	0000 0111	Doubleword alignment

Dec.	Hex.	Field	Bytes	Function
1	1	FCTSDSL	1	Segment length

File Control Table Segment Set (FCTSS)

DSECT NAME: DFHFCTSS
REGISTER: FCTSSBAR

```
                       F I L E   C O N T R O L   T A B L E
                               S E G M E N T   S E T
          ┌───────────────────────────────────────────────────────────┐
          *                                                           *
Dec. Hex. * <--------------------4 BYTES----------------------------> *
          *                                                           *
 0    0   *************************************************************
          *                          FCTSSN                           *
          *                                                           *
          *                     SEGMENT SET NAME                      *
 8    8   *-----------------------------------------------------------*
          *      FCTSSLGH              *          FCTSSS              *
          *                            *                              *
          *   LENGTH OF SEGMENT AREA   * SEGMENT SET USE ACCUMULATOR  *
12    C   *----------------------------*********************************
          *  FCTSSS      *  FCTSSIND   *
          *  (CONT)      *             *
          *              * SEGMENT SET *
          *              *  IND BYTE   *
16   10   *****************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	FCTSSN	8	Segment set name
8	8	FCTSSLGH	2	Length of segment area
10	A	FCTSSS	3	Segment set use accumulator
13	D	FCTSSIND	1	Segment set indicator byte

FILE CONTROL TABLE INDIRECT ACCESS EXTENSION

DSECT NAME: DFHFCTIA
REGISTER: FCTIABAR

The indirect access portion of the File Control Table contains information regarding levels of index. It is included as an extension of the FCT entry for the data set which is the index. It contains the name of the File Control Table entry for the next data set to be read and the location of the key or block reference information to be used in the next data set read. It also indicates whether a duplicate data set is associated with this index level and, if so, what its File Control Table entry is named. The indirect access information is generated by the assembly which creates the entire File Control Table in response to the DFHFCT TYPE=INDACC macro instruction.

```
                              F I L E   C O N T R O L
                         T A B L E   I N D I R E C T   A C C E S S
                ┌──────────────────────────────────────────────────────────┐
                *                                                          *
Dec.    Hex.    * <--------------------4 BYTES-------------------------->  *
                *                                                          *
  0       0     ************************************************************
                *                      FCIACIAD                            *
                *                                                          *
                *          NEXT LEVEL DATA SET IDENTIFICATION              *
  8       8     *----------------------------------------------------------*
                *         FCIACRDP          *   FCIACDL    *   FCIASI      *
                *                           *              *               *
                *   RELATIVE DATA POSITION  * DATA LENGTH  * DIRECT ACCESS *
                *                           *              *  SEARCH IND   *
 12       C     *----------------------------------------------------------*
                *   FCIACDUP    *              FCIACDID                    *
                *               *                                          *
                *  DUPLICATE    *                                          *
                * REC IND MASK  *                                          *
 16      10     *---------------*                                          *
                *                    DUPLICATE DATA SET IDENTIFICATION     *
                *                                                          *
 20      14     *               ********************************************
                *               *
 24      18     *****************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	FCIACIAD	8	Next level data set ID
8	8	FCIACRDP	2	Relative position within the logical record where the record ID of the next record is located
10	A	FCIACDL	1	Length of record ID field
11	B	FCIASI	1	Type of deblocking argument for blocked DAM

Bits	Setting	Function
0-3	1000 0000	Deblock by key
0-3	0100 0000	Deblock by rel record number

12	C	FCTACDUP	1	Duplicator record indicator mask
13	D	FCTACDID	8	Duplicate data set ID

DESTINATION CONTROL TABLE (DCT)

DSECT NAME: DFHDCTDS
REGISTER: DCTCBAR

The Destination Control Table contains entries which define the symbolic destinations, to and from which transient data is routed.

Intrapartition data is data coming to Transient Data Control from within CICS and being directed to a facility which is allocated to the same partition/region. Extrapartition data is used to define data which is either coming from a source outside of the partition/region or being directed from a source within the partition/region to a destination which is outside the partition/region.

```
                 D E S T I N A T I O N   C O N T R O L   T A B L E
           ┌------------------------------------------------------------┐
           *                                                            *
Dec.   Hex.* <-------------------4 BYTES------------------------------> *
           *                                                            *
 0      0  **************************************************************
           *                       TDDCTDID                             *
           *                                                            *
           *                     DESTINATION ID                         *
 4      4  *------------------------------------------------------------*
           *   TDDCTDT     *               TDDCTDS                      *
           *               *                                            *
           * DESTINATION   *    DESTINATION STATISTICS                  *
           *     TYPE      *          PACKED DECIMAL                    *
 8      8  *------------------------------------------------------------*


                               INDIRECT DESTINATION
 8      8  *------------------------------------------------------------*
           *                       TDDCTIDI                             *
           *                                                            *
           *            INDIRECT DESTINATION IDENTIFICATION             *
12      C  *------------------------------------------------------------*


                               EXTRAPARTITION ENTRY
 8      8  *------------------------------------------------------------*
           *   TDDCTCTL                    TDDCTCBA                     *
           *               *                                            *
           *    CONTROL    *                                            *
           *  INFORMATION  *           DCB ADDRESS                      *
12      C  *------------------------------------------------------------*
```

```
                     D E S T I N A T I O N   C O N T R O L   T A B L E
                   --------------------------------------------------------------
                   *                                                            *
Dec.      Hex.     * <--------------------4 BYTES----------------------------> *
                   *                                                            *
                   **************************************************************


                                  INTRAPARTITION ENTRY
  8         8      *------------------------------------------------------------*
                   *                         TDDCTTQC                           *
                   *                                                            *
                   *                   TOTAL QUEUE COUNTER                      *
 12         C      *------------------------------------------------------------*
                   *   TDDCTDQL             *   TDDCTTRC          *   TDDCTRQC  *
                   *                        *                     *             *
                   *                        *                     *   RESIDUAL  *
                   *   TRIGGER LEVEL        * TRACK REC COUNT     *  QUEUE COUNT *
 16        10      *------------------------------------------------------------*
                   *   TDDCTDEL  * TDDCTECB*                                    *
                   *             *         *             RESERVED               *
                   *   LOCK      *DUMMY ECB*                                    *
 20        14      *------------------------------------------------------------*
                   *       TDDCTOSA            *           TDDCTTROL            *
                   *                           *                                *
                   * OUTPUT TRACK SPACE USED   *        MAX DATA LENGTH         *
 24        18      *------------------------------------------------------------*
                   *                         TDDCTODA                           *
                   *                                                            *
                   *               OUTPUT DASD ADDRESS TTR                 *
 28        1C      *------------------------------------------------------------*
                   *                         TDDCTQSA                           *
                   *                                                            *
                   *               QUEUE STARTING ADDRESS TTR                   *
 32        20      *------------------------------------------------------------*
                   *                         TDDCTIDA                           *
                   *                                                            *
                   *                INPUT DASD ADDRESS TTR                      *
 36        24      *------------------------------------------------------------*
                   *                         TDDCTPTA                           *
                   *                                                            *
                   *              PREVIOUS TRACK ADDRESS TTR                    *
 40        28      *------------------------------------------------------------*
                   *                         TDDCTFAC                           *
                   *                                                            *
                   *              FORWARD CHAIN ADDRESS TTR                     *
 44        2C      *------------------------------------------------------------*


                           AUTOMATIC TRANSACTION INITIATION EXTENSION
 44        2C      *------------------------------------------------------------*
                   *                         TDDCTTID                           *
                   *                                                            *
                   *               TRANSACTION IDENTIFICATION                   *
 48        30      **************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	TDDCTDID	4	Symbolic destination ID
4	4	TDDCTDT	1	Destination type

Bits	Setting	Function
0-7	1000 0000	Intrapartition
0-7	0100 0000	Extrapartition
0-7	0010 0000	Indirect
0-7	1000 0001	Not reusable
0-7	1000 1000	Task Initiated
0-7	1001 0000	Terminal DEST, auto transaction
0-7	1001 0010	Non-terminal DEST, auto transaction

5	5	TDDCTDS	3	Destination statistics

INDIRECT DESTINATION

8	8	TDDCTIDI	4	Indirect destination ID if indirect destination

EXTRAPARTITION

8	8	TDDCTCTL	1	Open and resident indication

Bits	Setting	Function
0-7	1000 0000	Open
0-7	0000 1000	Non-resident

9	9	TDDCTCBA	3	DTF/DCB address if extrapartition

INTRAPARTITION

8	8	TDDCTTQC	4	Total Queue Counter
12	C	TDDCTDQL	2	Destination Queue trigger level
14	E	TDDCTTRC	1	Number of records on track
15	F	TDDCTTQC	1	Number of records left on track to be read
16	10	TDDCTDEL	1	Destination entry lock
17	11	TDDCTECB	1	Dummy ECB
18	12		2	Dummy ECB
20	14	TDDCTOSA	2	Output track space used
22	16	TDDCTIOL	2	Maximum data length
24	18	TDDCTODA	4	Output DASD address TTR
28	1C	TDDCTQSA	4	Queue starting address
32	20	TDDCTIDA	4	Input DASD address TTR
36	24	TDDCTPTA	4	Previous track address TTR
40	28	TDDCTFCA	4	Forward chain address
44	2C	TDDCTTID	4	Automatic transaction initiation transaction ID

DESTINATION CONTROL CHAIN RECORD

The first record of every track of the Intrapartition data set in use is a chain record.

C H A I N   R E C O R D   F O R M A T

```
                 *                                                        *
Dec.   Hex.*  <-------------------4 BYTES------------------------------>  *
                 *                                                        *
  0      0   **********************************************************
                 *                                                        *
                 *                        TDCRLH                          *
                 *                                                        *
                 *                  CHAIN RECORD LENGTH                   *
  4      4   *--------------------------------------------------------*
                 *                 TDCRIC                *     TDCRRC      *
                 *                                       *                 *
                 *           IDENTIFICATION CODE         *   REC COUNT     *
  8      8   *--------------------------------------------------------*
                 *                        TDCRDI                          *
                 *                                                        *
                 *               DESTINATION IDENTIFICATION               *
 12      C   *--------------------------------------------------------*
                 *                        TDCRBC                          *
                 *                                                        *
                 *                    BACKWARD POINTER                    *
 16     10   *--------------------------------------------------------*
                 *                        TDCRFC                          *
                 *                                                        *
                 *                    FORWARD POINTER                     *
 20     14   *--------------------------------------------------------*
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	TDCRLH	4	Chain record length
4	4	TDCRIC	3	Chain record identification code X'FDFEFF'
7	7	TDCRRC	1	Record count zero unless track is full
8	8	TDCRDI	4	Destination identification
12	C	TDCRBC	4	Chain record backward pointer Zero for first track of a queue
16	10	TDCRFC	4	Chain record forward pointer Zero unless track is full

SIGN-ON TABLE (SNT)

DSECT NAME: DFHSNNT
REGISTER: SNNTBAR

The Sign-on Table provides the user with the means for permanently retaining terminal operator data. Each entry in the table contains data used by CICS to verify an operator name and to establish a priority and a security key for the transactions which the operator enters.

```
                              S I G N - O N   T A B L E   E N T R Y
                  ┌────────────────────────────────────────────────────┐
                  *                                                    *
Dec.    Hex.      * <------------------4 BYTES-----------------------> *
                  *                                                    *
  0       0       ******************************************************
                  *                    SNNTN                           *
                  *                                                    *
                  *                OPERATOR NAME                       *
 20      14       *----------------------------------------------------*
                  *   SNNTNL      *             SNNTPS                 *
                  *               *                                    *
                  * NAME LENGTH   *            PASSWORD                *
 24      18       *----------------------------------------------------*
                  *   SNNTPS      *             SNNTID                 *
                  *               *                                    *
                  *    (CONT)     *       OPERATOR IDENTIFICATION      *
 28      1C       *----------------------------------------------------*
                  *             SNNTSK                  *   SNNTOP     *
                  *                                     *              *
                  *       OPERATOR SECURITY KEY         *  OPERATOR    *
                  *                                     *  PRIORITY    *
 32      20       ******************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	SNNTN	20	Operator name
20	14	SNNTNL	1	Actual length of operator name
21	15	SNNTPS	4	Password
25	19	SNNTID	3	Operator ID
28	1C	SNNTSK	3	Operator security key
31	1F	SNNTOP	1	Operator priority

TRACE TABLE (TRT)

DSECT NAME: DFHTRNTY

The Trace Table is a collection of entries representing events recorded by the Trace Control program. The number of entries in the table is specified at System Generation and is alterable at System Initialization. The Trace Control program records events as they occur, in consecutive entries in the table, in a wrap-around fashion (the most current event entry replacing the oldest event entry).

```
                                T R A C E   T A B L E   E N T R Y
                  ┌────────────────────────────────────────────────────┐
                  *                                                    *
Dec.    Hex.      * <------------------4 BYTES-----------------------> *
                  *                                                    *
  0       0       ******************************************************
                  *     TRID      *             TRRETAD                *
                  *               *                                    *
                  * TRACE ENTRY   *                                    *
                  *IDENTIFICATION*       CONTENTS OF REGISTER 14       *
  4       4       *----------------------------------------------------*
```

```
                          TRACE TABLE ENTRY
             ┌------------------------------------------------------------┐
             *                                                            *
Dec.   Hex.  * <-------------------4 BYTES--------------------------->   *
             *                                                            *
             **************************************************************
             *     TRTR       *              TRTCAID                      *
             *                *                                           *
             *   TYPE OF      *                                           *
             * REQ FROM TCA   *      TASK IDENTIFICATION NUMBER           *
  8      8   *------------------------------------------------------------*
             *                       TRDATA1                              *
             *                                                            *
             *               DATA FIELD 1 (FIELD A)                       *
 12      C   *------------------------------------------------------------*
             *                       TRDATA2                              *
             *                                                            *
             *               DATA FIELD 2 (FIELD B)                       *
 16     10   **************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	TRID	1	Trace entry ID
1	1	TRRETAD	3	Contents of register 14 at entry to the CICS management program
4	4	TRTR	1	Type of request from TCA
5	5	TRTCAID	3	Task ID number
8	8	TRDATA1	4	Data field 1
12	C	TRDATA2	4	Data field 2

If the full trace function is operative, execution of each CICS service macro instruction causes an entry to be made in the Trace Table. For example, if the application issues a CICS Storage Control GETMAIN, an entry is placed in the table indicating that request.

For further details, see the discussion of "Program Testing and Debugging" in the CICS Application Programmer's Reference Manual.

TEMPORARY STORAGE TABLE (TST)

DSECT NAME: DFHTSAT
REGISTER: TSATBAR

```
                     TEMPORARY STORAGE TABLE
             ┌------------------------------------------------------------┐
             *                                                            *
Dec.   Hex.  * <-------------------4 BYTES-------------------------->    *
             *                                                            *
  0      0   **************************************************************
             *                        TSATDI                              *
             *                                                            *
             *                TST DATA IDENTIFICATION                     *
  8      8   *----------------------------*********************************
             *          TSATDL            *
             *                            *
             *        DATA LENGTH         *
 12      C   ******************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	TSATDI	8	TST data ID
8	8	TSATDL	2	TST data length

SYSTEM INITIALIZATION TABLE

DSECT NAME: DFHSITDS

```
               S Y S T E M   I N I T I A L I Z A T I C N   T A B L E
           ┌-----------------------------------------------------------┐
           *                                                           *
Dec.   Hex.* <--------------------4 BYTES----------------------------> *
           *                                                           *
 0      0  *************************************************************
           *                       DFHSITDS                            *
           *                                                           *
           *                   HEADER INFORMATION                      *
16     10  *-----------------------------------------------------------*
           *                       SITOSCOR                            *
           *                                                           *
           *               MAIN STORAGE RESERVED FOR OS                *
20     14  *-----------------------------------------------------------*
           *                       SITSCSZ                             *
           *                                                           *
           *                 STORAGE CUSHION SIZE                      *
24     18  *-----------------------------------------------------------*
           *                       SITTRTSZ                            *
           *                                                           *
           *               TRACE TABLE NUMBER OF ENTRIES               *
28     1C  *-----------------------------------------------------------*
           *                        SITICVAL                           *
           *                                                           *
           *                  SYSTEM TIME INTERVAL                     *
32     20  *-----------------------------------------------------------*
           *                       SITRICVL                            *
           *                                                           *
           *               RUNAWAY TASK TIME INTERVAL                  *
36     24  *-----------------------------------------------------------*
           *       SITSICVL             *           SITMXTSK           *
           *                            *                              *
           *   STALL TIME INTERVAL      *       MAXIMUM TASK COUNT     *
40     28  *-----------------------------------------------------------*
           *   SITMSGLVL  *  SITSNAP    *  SITPL1      *    SITDL1     *
           *              *             *              *               *
           * CONSOLE MSG  *  SNAP DATA  *  PL/I IND    *    DL/I IND   *
           *  LEVEL IND   *   SET IND   *              *               *
44     2C  *-----------------------------------------------------------*
           *   SITATP     *        SITTCTSF            *    SITFCTSF   *
           *    ATP       *     TERMINAL CONTROL       *    FCT SUFFIX *
           *    IND       *       TABLE SUFFIX         *               *
48     30  *-----------------------------------------------------------*
           *   SITFCTSF   *        SITDCTSF            *    SITPPTSF   *
           *    (CONT)    *                            *               *
           *              *      DCT SUFFIX            *   PPT SUFFIX  *
52     34  *-----------------------------------------------------------*
           *              *                            *               *
           *   SITPPTSF   *        SITPCTSF            *    SITCSASF   *
           *    (CONT)    *                            *      CSA      *
           *              *      PCT SUFFIX            *    SUFFIX     *
56     38  *-----------------------------------------------------------*
```

SYSTEM INITIALIZATION TABLE

```
              ┌─────────────────────────────────────────────────────────────┐
              *                                                             *
Dec.   Hex.   * <-------------------4 BYTES-------------------------------> *
              *                                                             *
              ***************************************************************
              *   SITCSASF   *        SITKCPSF             *   SITSCPSF     *
              *    (CONT)    *                             *   STORAGE      *
              *              *   TASK CONTROL SUFFIX       *   CONTROL      *
              *              *                             *   SUFFIX       *
60     3C     *-------------------------------------------------------------*
              *   SITSCPSF   *        SITPCPSF             *   SITDCPSF     *
              *    (CONT)    *                             * DUMP CONTROL   *
              *              * PROGRAM CONTROL SUFFIX      *   SUFFIX       *
64     40     *-------------------------------------------------------------*
              *   SITDCPSF   *        SITICPSF             *   SITTCPSF     *
              *    (CONT)    *                             *   TERMINAL     *
              *              * INTERVAL CONTROL SUFFIX     *   CONTROL      *
              *              *                             *   SUFFIX       *
68     44     *-------------------------------------------------------------*
              *   SITTCPSF   *        SITFCPSF             *   SITTDPSF     *
              *    (CONT)    *                             *   TRANSIENT    *
              *              *   FILE CONTROL SUFFIX       *   DATA SUFFIX  *
72     48     *-------------------------------------------------------------*
              *   SITTDPSF   *        SITTSPSF             *   SITTRPSF     *
              *    (CONT)    *                             *    TRACE       *
              *              *   TEMP STORAGE SUFFIX       *    SUFFIX      *
76     4C     *-------------------------------------------------------------*
              *   SITTRP     *        SITPIPSF             *                *
              *  (CONT)      *                             *                *
              *              * PROGRAM INTERRUPT SUFFIX    *                *
80     50     ***************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	DFHSITDS	16	Header information
16	10	SITOSCOR	4	Main storage reserved for the operating system
20	14	SITSCSZ	4	Storage cushion size
24	18	SITTRTSZ	4	Trace Table number of entries
28	1C	SITICVAL	4	System time interval
32	20	SITRICVL	4	Runaway task time interval
36	24	SITSICVL	2	Stall detection time interval
38	26	SITMXTSK	2	Maximum number of tasks count
40	28	SITMSGLV	1	Console message level indicator
41	29	SITSNAP	1	SNAP data set indicator
42	2A	SITPL1	1	PL/I indicator
43	2B	SITDL1	1	DL/1 indicator
44	2C	SITATP	1	Asynchronous Transaction Process
45	2D	SITTCTSF	2	Terminal Control Table suffix char(s)
47	2F	SITFCTSF	2	File Control Table suffix char(s)
49	31	SITDCTSF	2	Destination Control Table suffix char(s)
51	33	SITPPTSF	2	Processing Program Table suffx char(s)
53	35	SITPCTSF	2	Program Control Table suffix char(s)
55	37	SITCSASF	2	Common System Area suffix char(s)
57	39	SITKCPSF	2	Task Control program suffix char(s)
59	3B	SITSCPSF	2	Storage Control program suffx char(s)
61	3D	SITPCPSF	2	Program Control program suffx char(s)
63	3F	SITDCPSF	2	Dump Control program suffix character
65	41	SITICPSF	2	Interval Control program suffix char(s)

67	43	SITTCPSF	2	Terminal Control program suffix char(s)
69	45	SITFCPSF	2	File Control program suffix char(s)
71	47	SITTDPSF	2	Transient Data Control program suffix char(s)
73	49	SITTSPSF	2	Temporary Storage Control program suffix char(s)
75	4B	SITTRPSF	2	Trace Control program suffix char(s)
77	4D	SITPIPSF	2	Program Interrupt program suffix

SIP ATTACH LIST

DESCT NAME: DFHDLL

```
                               S I P   A T T A C H   L I S T
                   ┌──────────────────────────────────────────────────────────────┐
                   *                                                              *
Dec.    Hex.       * <------------------------4 BYTES---------------------------> *
                   *                                                              *
 0       0         ****************************************************************
                   *                                                              *
                   *            POINTER TO SYMBOLIC NAME                          *
 4       4         *--------------------------------------------------------------*
                   *   HIEARCHY    *                                              *
                   *   SUPPORT     *           DCB ADDRESS                        *
                   *   IND         *                                              *
 8       8         *--------------------------------------------------------------*
                   *               *                                              *
                   *   RESERVED    *           ECB ADDRESS                        *
                   *               *                                              *
12       C         *--------------------------------------------------------------*
                   *                                                              *
                   *                    GSPL OR GSPV ADDRESS                      *
                   *                                                              *
16      10         *--------------------------------------------------------------*
                   *               *                                              *
                   *  RESERVED     *           SHSPL ADDRESS                      *
                   *               *                                              *
20      14         *--------------------------------------------------------------*
                   *   ROLLIN/     *                                              *
                   *   ROLLOUT     *       EXIT ROUTINE ADDRESS                   *
                   *   INDICATOR   *                                              *
24      18         *--------------------------------------------------------------*
                   *                          *                *                  *
                   *         DPMOD            *     LPMOD      *   RESERVED       *
                   *                          *                *                  *
28      1C         *--------------------------------------------------------------*
                   *                                                              *
                   *                       RESERVED                               *
                   *                                                              *
32      20         *--------------------------------------------------------------*
                   *                                                              *
                   *                       RESERVED                               *
                   *                                                              *
36      24         *--------------------------------------------------------------*
                   *                                                              *
                   *                       RESERVED                               *
                   *                                                              *
40      28         *--------------------------------------------------------------*
```

S I P   A T T A C H   L I S T

```
                *-----------------------------------------------------------*
                *                                                           *
Dec.    Hex.    * <--------------------4 BYTES-------------------------->   *
                *                                                           *
                *************************************************************
                *                                                           *
                *           STAI PARM + EXIT ADDRESS                        *
                *                                                           *
 44      26     *-----------------------------------------------------------*
                *                                                           *
                *                   TASK LIB                                *
                *                                                           *
 48      30     *-----------------------------------------------------------*
                *                   SIPDLIIT                                *
                *                                                           *
                *       DL/I INITIALIZATION MODULE NAME                     *
 52      34     *-----------------------------------------------------------*
                *                   SIPDLIIT                                *
                *                                                           *
                *                    (CONT)                                 *
 56      38     *-----------------------------------------------------------*
                *                   DLILISTA                                *
                *                                                           *
                *           ADDRESS OF DL/I PARM LIST                       *
 60      3C     *-----------------------------------------------------------*
                *                           *         DLILIST               *
                *       RESERVED            *                               *
                *                           *     LENGTH OF PARMLIST        *
 64      40     *-----------------------------------------------------------*
                *                   DLICSA                                  *
                *                                                           *
                *                 CSA ADDRESS                               *
 68      44     *-----------------------------------------------------------*
                *                                                           *
                *               DL/I IDENTIFIER                             *
                *                                                           *
 72      48     *-----------------------------------------------------------*
                *                                                           *
                *            CICS DL/I TASK NAME                            *
                *                                                           *
 76      4C     *                                       *-------------------*
                *                                       *                   *
                *                                       *     DLIPSBN       *
                *                                       *                   *
 80      50     *---------------------------------------*                   *
                *                   DLIPSBN                                 *
                *                                                           *
                *                   PSB NAME                                *
 88      58     *-----------------------------------------------------------*
                *           DLIPSBL                     *     DLIDMBL       |
                *                                       *                   *
                *       PSB BUFFER LENGTH               * DMB BUFFER LENGTH *
 92      5C     *-----------------------------------------------------------*
                *       DLIDMBL             *         DLIBUFL               *
                *                           *                               *
                *       (CCNT)              *     BUFFER LENGTH             *
 96      60     *-----------------------------------------------------------*
                *   DLIBUFL  *              *                               *
                *            *    COMMA     *         RESERVED              *
                *   (CONT)   *              *                               *
100      64     *-----------------------------------------------------------*
```

S I P ATTACH LIST

```
              ┌──────────────────────────────────────────────────────────────┐
              *                                                              *
Dec.   Hex.   * <-------------------4 BYTES--------------------------------> *
              *                                                              *
              ****************************************************************
              *                           SHRSPLST                           *
              *                                                              *
              *                       SHARED SUBPOOL LIST                    *
228    E4     ****************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		4	Pointer to symbolic name
4	4		1	Hierarchy support indicator
5	5		3	DCB address
8	8		1	Reserved
9	9		3	ECB address
12	C		4	GSPL or GSPY address
16	10		1	Reserved
17	11		3	SHSPL address
20	14		1	Rollin/rollout indicator
21	15		3	Exit routine address
24	18		2	DPMOD
26	1A		1	LPMOD
27	1B		13	Reserved
40	28		4	STAI PARM and exit address
44	2C		4	Task LIB
48	30	SIPDLIIT	8	DL/I initialization name
56	38	DLILISTA	4	Pointer to PARM LIST
60	3C	Reserved	2	Reserved
62	3E	DLILIST	2	Length of PARM LIST
64	40	DLICSA	4	CSA address
68	44		4	DL/I identifier
72	48		7	CICS DL/I task name
79	4E	DLIPSBN	9	PSB name
88	58	DLIPSBL	3	PSB buffer length
91	5B	DLIDMBL	3	DMB buffer length
94	5E	DLIBUFL	3	Buffer length
97	61		1	Comma
98	62		2	Reserved
100	64	SHRSPLST	128	Shared subpool list

BATCH CONTROL AREA

DSECT NAME: DFHBCADS
REGISTER: BCABAR

The Batch Control Area (BCA) (part of main storage) is acquired by the Asynchronous Transaction Input Processor (DFHRDR) when a user submits a batch of transactions from a remote terminal using the CRDR transaction code. Data contained in the BCA is used to control the processing of the batched transactions. The BCA is placed on the BCA chain, the head of which is located in the ATP CSA extension area (DFHATPDS).

BATCH CONTROL AREA

```
                  *                                                            *
 Dec.     Hex.    *  <--------------------------4 BYTES--------------------------> *
                  *                                                            *
   0         0    **************************************************************
                  *                                                            *
                  *                    STORAGE ACCOUNTING                      *
                  *                                                            *
   4         4    *------------------------------------------------------------*
                  *                          BCANAME                           *
                  *                                                            *
                  *                        BATCH NAME                          *
  12         C    *------------------------------------------------------------*
                  *              *                *                *           *
                  *  BCASTAGE    *    BCAINDA     *    BCAINDB     *  BCADLML  *
                  *    FLAGS     *     FLAGS      *     FLAGS      * DELIM LGTH*
  16        10    *------------------------------------------------------------*
                  *                          BCADELIM                          *
                  *                                                            *
                  *                   BATCH FLUSH DELIMITER                    *
  20        14    *------------------------------------------------------------*
                  *                          BCAPASSW                          *
                  *                                                            *
                  *                          PASSWORD                          *
  28        1C    *------------------------------------------------------------*
                  *                          BCACHAIN                          *
                  *                                                            *
                  *                       CHAINING FIELD                       *
  32        20    *------------------------------------------------------------*
                  *                           BCAWRE                           *
                  *                                                            *
                  *                   WRE CHAIN FOR THIS BCA                   *
  36        24    *------------------------------------------------------------*
                  *        BCANLREC               *           BCAOLREC         *
                  *                               *                            *
                  *   LARGEST INPUT REC           *      LARGEST OUTPUT REC    *
  40        28    *------------------------------------------------------------*
                  *                          BCAOBUF                           *
                  *                                                            *
                  *                 ADDR OF OUTPUT QUEUE BUFFER                *
  44        2C    *------------------------------------------------------------*
                  *                          BCAIBUF                           *
                  *                                                            *
                  *                 ADDR OF INPUT QUEUE BUFFER                 *
  48        30    *------------------------------------------------------------*
                  *                          BCADCTIN                          *
                  *                                                            *
                  *                 INPUT QUEUE CONTROL BLOCK                  *
  92        5C    *------------------------------------------------------------*
                  *                          BCADCTOT                          *
                  *                                                            *
                  *                 OUTPUT QUEUE CONTROL BLOCK                 *
 136        88    *------------------------------------------------------------*
                  *                          BCADTCTE                          *
                  *                                                            *
                  *                  COPY OF ORIGINAL TCTTE                    *
 220        DC    **************************************************************
```

Displacement				
Dec.	Hex.	Field	Bytes	Function
0	0		4	Storage accounting field
4	4	BCANAME	8	Batch name
12	C	BCASTAGE	1	Batch processing control flags

	Bits	Setting	Function
BCAQFULL	0-7	1000 0000	Output queue full
BCARDYIN	0-7	0100 0000	Input complete - ready for processing
BCANPROC	0-7	0010 0000	Batch being processed
BCAHOLD	0-7	0001 0000	Batch in "hold" status
BCARDYOT	0-7	0000 1000	Processing complete - ready for output
BCAOPROC	0-7	0000 0100	Output in progress
BCASAVEQ	0-7	0000 0010	Output data to be saved
BCADELTQ	0-7	0000 0001	Batch to be terminated and data deleted

13	D	BCAINDA	1	Batch processing control flags

	Bits	Setting	Function
BCAPWPTI	0-7	1000 0000	Output is password protected
BCAABEND	0-7	0100 0000	Batch task is abnormally terminated
BCAABTRM	0-7	0010 0000	Last task abnormally terminated
BCAATCHP	0-7	0001 0000	Attach pending
BCATRUNC	0-7	0000 1000	Output truncated

14	E	BCAINDB	1	Processing control flags
15	F	BCADLML	1	Length of flush delimiter character string minus one
16	10	BCADELIM	4	Batch flush delimiter character string
20	14	BCAPASSW	8	Batch password if BCAPWPTI bit in BCAINDA is on
28	1C	BCACHAIN	4	Address of next BCA on BCA chain
32	20	BCAWRE	4	Address of first WRE for this batch
36	24	BCANLREC	2	Largest input record (binary)
38	26	BCAOLREC	2	Largest output record (binary)
40	28	BCAOBUF	4	Address of the Transient Data output buffer used to queue batch output
44	2C	BCAIBUF	4	Address of the Transient Data input buffer used to acquire queued input data for batch processing
48	30	BCADCTIN	44	DCT used to control queued batch input (for description, see DFHDCTDS DSECT)
92	5C	BCADCTOT	44	DCT used to control queued batch output (for description, see DFHDCTDS DSECT)
136	88	BCADTCTE	84	Copy of originating terminal's TCTTE (for description, see DFHTCTTE DSECT)

CONTROL BLOCKS - INPUT/OUTPUT AREAS AND WORK AREAS

The following areas contain control fields which precede the actual data areas acquired by CICS. They contain information required by CICS and, in some instances, useful to the application programs.

TERMINAL INPUT/OUTPUT AREA (TIOA)

DSECT NAME: DFHTIOA
REGISTER: TIOABAR

```
                          T E R M I N A L   I / O   A R E A
            ┌────────────────────────────────────────────────────────────┐
            *                                                            *
Dec.  Hex.  * <--------------------4 BYTES----------------------------> *
            *                                                            *
 0     0    **************************************************************
            *                            *          TIOASAL             *
            *                            *                              *
            *    STORAGE ACCOUNTING      *        AREA LENGTH           *
 4     4    *------------------------------------------------------------*
            *                         TIOASCA                            *
            *                                                            *
            *              TERMINAL STORAGE CHAIN ADDRESS                *
 8     8    *------------------------------------------------------------*
            *          TIOATDL           *   TIOAWCI    *   TIOALAC      *
            *                            *              *                *
            *   TERMINAL DATA LENGTH     *  WRITE CONTR *  LINE ADDR     *
            *                            *   INDICATOR  *   CONTROL      *
12     C    *------------------------------------------------------------*
            *                         TIOADBA                            *
            *                                                            *
            *                    TERMINAL DATA AREA                      *
            **************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		2	Storage accounting
2	2	TIOASAL	2	Area length
4	4	TIOASCA	4	Chain address of next terminal storage related to the same task
8	8	TIOATDL	2	Length of terminal data area
10	A	TIOAWCI	1	Write control indicator
11	B	TIOALAC	1	Line address control
12	C	TIOADBA		Terminal data area

FILE INPUT/OUTPUT AREA (FIOA)

DSECT NAME: DFHFIOA
REGISTER: FIOABAR

The FIOA is acquired dynamically from main storage by the File Control program whenever a request is made for file services. The data area, beginning at field FIOADBA, is used as the true I/O area from/to which records are read/written.

FILE INPUT/OUTPUT AREA

```
               ┌────────────────────────────────────────────────────────────┐
               *                                                            *
Dec.    Hex.   * <--------------------4 BYTES----------------------------> *
               *                                                            *
 0       0     **************************************************************
               *                                                            *
               *           STORAGE ACCOUNTING CONTRCL INFORMATION           *
               *                                                            *
 4       4     *------------------------------------------------------------*
               *   FCIOEXB    *                                             *
               *              *                                             *
               *EXCLUSIVE CNTL*     STORAGE ACCOUNTING CHAIN ADDRESS        *
               *  INDICATOR   *                                             *
 8       8     *------------------------------------------------------------*


                                  DATA EVENT CONTROL BLOCK
 8       8     *------------------------------------------------------------*
               *                          FCFIOECB                          *
               *                                                            *
               *                    OS EVENT CONTROL BLOCK                  *
12       C     *------------------------------------------------------------*
               *           FCFIOTYP            *          FCFIOLNG          *
               *                               *                            *
               *TYPE OF OPERATION INDICATOR *    LENGTH OF DATA RECORD      *
16      10     *------------------------------------------------------------*
               *                          FCFIODCB                          *
               *                                                            *
               *           ADDRESS OF DATA CONTROL BLCCK (DCB)              *
20      14     *------------------------------------------------------------*
               *                          FCFIOAA                           *
               *                                                            *
               *                      I/O AREA ADDRESS                      *
24      18     *------------------------------------------------------------*
               *                          FCFIOLRS                          *
               *                                                            *
               *                 ADDRESS OF LOGICAL RECORD                  *
28      1C     *------------------------------------------------------------*
               *                          FCFIOKA                           *
               *                                                            *
               *                      ADDRESS OF KEY                        *
32      20     *------------------------------------------------------------*
               *                          FCFIOBRF                          *
               *                                                            *
               *           ADDRESS OF BLOCK REFERENCE FIELD                 *
36      24     *------------------------------------------------------------*
               *                          FCFNXADR                          *
               *                                                            *
               *           ADDRESS OF BDAM FEEDBACK FIELD                   *
40      28     *------------------------------------------------------------*
               *           FCFIOVRL            *          FCUPHOLD          *
               *                               *                            *
               *   VARIABLE RECORD LENGTH      *        UPDATE COUNT        *
44      2C     *------------------------------------------------------------*
               *                          FCFECADR                          *
               *                                                            *
               *          ADDRESS OF EXCLUSIVE CONTRCL ARGUMENT             *
48      30     *------------------------------------------------------------*
               *                          FCFIOLRA                          *
               *                                                            *
               *                 ADDRESS OF LOGICAL RECORD                  *
52      34     *------------------------------------------------------------*
```

FILE INPUT/OUTPUT AREA

```
                  ┌──────────────────────────────────────────────────────────────┐
                  *                                                              *
Dec.     Hex.     * <--------------------4 BYTES------------------------------> *
                  *                                                              *
                  ****************************************************************
                  *                         FCFIOFCT                             *
                  *                                                              *
                  *              FIIE CCNTROL TABLE ENTRY ADDRESS                *
56       38       *--------------------------------------------------------------*
                  *                         FCFBWA                               *
                  *                                                              *
                  *              ADDRESS OF FILE BROWSE WORK AREA                *
60       3C       *--------------------------------------------------------------*
                  *                         FCFIOICA                             *
                  *                                                              *
                  *                 FILE I/O AREA CHAIN ADDRESS                  *
64       40       *--------------------------------------------------------------*
                  *                         FCDSO1D                              *
                  *                                                              *
                  *                 BEGINNING OF DATA AREA                       *
                  ****************************************************************
```

Displacement

Hex.	Dec.	Field	Bytes	Function
0	0		4	Storage accounting control data
4	4	FCIOEXB	1	Exclusive control indicator

Bits	Setting	Function
0-7	0100 0000	Record under exclusive control
0-7	0001 0000	Record being added to ISAM data set

Hex.	Dec.	Field	Bytes	Function
5	5		3	Storage accounting chain address
8	8	FCFIOECB	4	Event Control Block (ECB)
12	C	FCFIOTYP	2	Type of operation codes - Refer to <u>OS/360 System Control Blocks</u> (GC28-6628) for specific bit meanings of ISAM and BDAM DECB type of I/O requests.
14	E	FCFIOLNG	2	Length of data
16	10	FCFIODCB	4	DCB address associated with this I/O operation.
20	14	FCFIOAA	4	I/O area address. Normally this address of field FCDSO1 below.
24	18	FCFIOLRS	4	Address of logical record after an ISAM I/O request.
28	1C	FCFIOKA	4	Address of key
32	20	FCFIOBRF	4	Address of block reference field (BDAM) or ISAM exception codes
36	24	FCFNXADR	4	Address of feedback area for BDAM
40	28	FCFIOVRL	2	Variable record length of ISAM record being updated.
42	2A	FCUPHOLD	2	Update count from FCTE at beginning of update operation.
44	2C	FCFECADR	4	Address of exclusive control argument
48	30	FCFIOLRA	4	Logical record address
52	34	FCFIOFCT	4	Address of FCTE for this I/O request
56	38	FCFBWA	4	Address of File Browse Work Area during browse operation

Displacement

Hex.	Dec.	Field	Bytes	Function
60	3C	FCFIOICA	4	File I/O Area chain address
64	40	FCDSO1D	Variable	Beginning of data area

TRANSIENT DATA INPUT AREA (TDIA)

DSECT NAME: DFHTDIA
REGISTER: TDIABAR

```
                         T R A N S I E N T   D A T A   I N P U T   A R E A
                   ┌─────────────────────────────────────────────────────────────┐
                   *                                                             *
Dec.      Hex.     * <---------------------4 BYTES---------------------------> *
                   *                                                             *
 0         0       ***************************************************************
                   *                           *          TDIASAL                *
                   *                           *                                 *
                   *    STORAGE ACCOUNTING     *    STORAGE ACCOUNTING           *
                   *                           *        AREA LENGTH              *
 4         4       *-------------------------------------------------------------*
                   *                        TDIASCA                              *
                   *                                                             *
                   *         TRANSACTION STORAGE CHAIN ADDRESS                   *
 8         8       *-------------------------------------------------------------*
                   *                        TDIADECB                             *
                   *                                                             *
                   *         BDAM DATA EVENT CONTROL BLOCK                       *
36        24       *-------------------------------------------------------------*
                   *         TDIAIRL           *                                 *
                   *                           *                                 *
                   *       INTRAPARTITION      *           RESERVED              *
                   *       RECORD LENGTH       *                                 *
40        28       *-------------------------------------------------------------*
                   *                        TDIADBA                              *
                   *                                                             *
                   *                  BEGINNING OF DATA                          *
                   ***************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		2	Storage accounting
2	2	TDIASAL	2	Storage accounting area length
4	4	TDIASCA	4	Transaction storage chain address
8	8	TDIADECB	28	BDAM Data Event Control Block
36	24	TDIAIRL	2	Intrapartition record length
38	26		2	Reserved
40	28	TDIADBA		Beginning of data area

TRANSIENT DATA OUTPUT AREA (TDOA)

DSECT NAME: DFHTDOA
REGISTER: TDOABAR

```
                T R A N S I E N T   D A T A   O U T P U T   A R E A
              ┌─────────────────────────────────────────────────────────────┐
              *                                                             *
Dec.    Hex.  * <--------------------4 BYTES------------------------------> *
              *                                                             *
 0       0    ***************************************************************
              *                             *          TDOASAL              *
              *     STORAGE ACCOUNTING      *    STORAGE ACCOUNTING         *
              *                             *         AREA LENGTH           *
 4       4    *-------------------------------------------------------------*
              *                          TDOASCA                            *
              *                                                             *
              *            TRANSACTION STORAGE CHAIN ADDRESS                *
 8       8    *-------------------------------------------------------------*
              *            TDOAVRL          *                               *
              *                             *                               *
              *   VARIABLE RECORD LENGTH    *           RESERVED            *
12       C    *-------------------------------------------------------------*
              *                          TDOADBA                            *
              *                                                             *
              *                      BEGINNING OF DATA                      *
              ***************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		2	Storage accounting
2	2	TDOASAL	2	Storage accounting area length
4	4	TDOASCA	4	Transaction storage chain address
8	8	TDOAVRL	2	Variable record length
10	A		2	Reserved
12	C	TDOADBA		Beginning of data area

TEMPORARY STORAGE INPUT/OUTPUT AREA (TSIOA)

DSECT NAME: DFHTSIOA
REGISTER: TSIOABAR

```
          T E M P O R A R Y   S T O R A G E   I N P U T / O U T   A R E A
              ┌─────────────────────────────────────────────────────────────┐
              *                                                             *
Dec.    Hex.  * <--------------------4 BYTES------------------------------> *
              *                                                             *
 0       0    ***************************************************************
              *                             *          TSIOASAL             *
              *                             *                               *
              *     STORAGE ACCOUNTING      *     STORAGE ACCOUNTING        *
              *                             *         AREA LENGTH           *
 4       4    *-------------------------------------------------------------*
              *                          TSIOASCA                           *
              *                                                             *
              *          TRANSACTION STORAGE CHAIN ADDRESS                  *
 8       8    *-------------------------------------------------------------*
              *            TSIOAVRL         *                               *
              *                             *                               *
              *   VARIABLE RECORD LENGTH    *           RESERVED            *
12       C    *-------------------------------------------------------------*
              *                          TSIOADBA                           *
              *                                                             *
              *                      BEGINNING OF DATA                      *
              ***************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		2	Storage accounting
2	2	TSIOASAL	2	Storage accounting area length
4	4	TSIOASCA	4	Transaction storage chain address
8	8	TSIOAVRL	2	Variable record length
10	A		2	Reserved
12	C	TSIOADBA		Beginning of data area

FILE WORK AREA (FWA)

DSECT NAME: DFHFWADS
REGISTER: FWACBAR

```
                              F I L E   W O R K   A R E A
                ┌─────────────────────────────────────────────────────────────┐
                *                                                             *
Dec.    Hex.    * <──────────────────────4 BYTES───────────────────────────> *
                *                                                             *
 0       0      ***************************************************************
                *                                                             *
                *              STORAGE ACCOUNTING INFORMATION                 *
                *                                                             *
 8       8      *-------------------------------------------------------------*
                *                          FCUPDRA                            *
                *                                                             *
                *                   FILE I/O AREA ADDRESS                     *
12       C      *-------------------------------------------------------------*
                *                          FCUFCTA                            *
                *                                                             *
                *            ADDRESS OF FILE CONTROL TABLE ENTRY              *
16      10      *-------------------------------------------------------------*
                *                           FCUWA                             *
                *                                                             *
                *             BEGINNING ADDRESS WORK AREA                     *
                ***************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		8	Storage accounting
8	8	FCUPDRA	4	Address of File Input/Output Area
12	C	FCUFCTA	4	Address of File Control Table entry
16	10	FCUWA		Beginning of data area

FILE BROWSE WORK AREA (FBWA)

DSECT NAME: DFHFBWA
REGISTER: FBWABAR

```
                              F I L E   B R O W S E   W O R K   A R E A
                 ┌───────────────────────────────────────────────────────────────┐
                 *                                                               *
Dec.    Hex.     * <────────────────────4 BYTES────────────────────────────────> *
                 *                                                               *
  0       0      *****************************************************************
                 *                                                               *
                 *                   STORAGE ACCOUNTING DATA                     *
                 *                                                               *
  4       4      *---------------------------------------------------------------*
                 *                                                               *
                 *               STORAGE ACCOUNTING CHAIN FIELD                  *
                 *                                                               *
  8       8      *---------------------------------------------------------------*
                 *         FBWAWK1          *           FBWASEGL                 *
                 *                          *                                    *
                 *        WORK AREA         *  LENGTH OF LAST SEGMENT SET        *
 12       C      *---------------------------------------------------------------*
                 *                        FBWARI                                 *
                 *                                                               *
                 *      ADDRESS OF USER'S RECORD IDENTIFICATION FIELD            *
 16      10      *---------------------------------------------------------------*
                 *                        FBWASEG                                *
                 *                                                               *
                 *         SEGMENT SET NAME SPECIFIED AT SETL                    *
 24      18      *---------------------------------------------------------------*
                 *                        FBWABLKE                               *
                 *                                                               *
                 *        ADDRESS OF LAST DATA POSITION IN FIOA                  *
 28      1C      *---------------------------------------------------------------*
                 *                        FBWASSBR                               *
                 *                                                               *
                 *       ADDRESS OF FCTE FOR DEFAULT SEGMENT SET                 *
 32      20      *---------------------------------------------------------------*
                 *         FBWADRRN         *           FBWAFCTR                 *
                 *                          *                                    *
                 *   BDAM REL RECORD NUMBER *  SETL TYPE OF REQUEST IND          *
 36      24      *---------------------------------------------------------------*
                 *         FBWAIDLN         *           FBWAKEYE                 *
                 *                          *                                    *
                 *   LENGTH OF BDAM BLK REF *      INDEX INTO FBWAKEY            *
 40      28      *---------------------------------------------------------------*
                 *                        FBWAKEY                                *
                 *                                                               *
                 *              ID OF LAST BLOCK RETRIEVED                       *
                 *****************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		8	Storage accounting field (DFHSAADS)
8	8	FBWAWK1	2	Work area used to increment by one the key of the last block retrieved.
10	A	FBWASEGL	2	Length of last segment set retrieved
12	C	FBWARI	4	Address of user's record ident field as provided in SETL request.

Displacement				
Dec.	Hex.	Field	Bytes	Function
16	10	FBWASEG	8	Segment set name as provided in SETL request
24	18	FBWABLKE	4	Address of the last data position in the FIOA.
28	1C	FBWASSBR	4	Address of the segment set entry in FCT.
32	20	FBWADRRN	2	BDAM relative record number of next logical record to be presented to user.
34	22	FBWAFCTR	2	Type of request as specified in SETL request. See TCA DSECT, field TCAFCTR for bit definitions.
36	24	FBWAIDLN	2	Length of block reference being used in browse operation
38	26	FBWAKEYE	2	Index into FBWAKEY field. For ISAM, relative pointer (from beginning of this field) to next-to-last position of key. For BDAM, points to next record ID.
40	28	FBWAKEY	Note	For ISAM, contains key of last record given to user in GETNEXT request. For BDAM, contains three fields as follows: ID of current block Key of current block (if any) ID of next block to be read

Note: For ISAM, length of FBWAKEY is equal to key length. For BDAM, length is equal to 2 (block reference length) plus the key length (if any).

STORAGE ACCOUNTING AREA (SAA)

DSECT NAME: DFHSAADS
REGISTER: SAACBAR

```
                        S T O R A G E   A C C O U N T I N G   A R E A
              ┌──────────────────────────────────────────────────────────────┐
              *                                                              *
Dec.    Hex.  * <---------------------4 BYTES-------------------------------> *
              *                                                              *
 0       0    ****************************************************************
              *                       SAACFDC                                *
              *   SAASCI      *   SAASFI     *          SAASAD               *
              *               *              *                               *
              *  STORAGE      *  STORAGE     *       STORAGE AREA            *
              *  CLASS ID     * FORMAT ID    *       DISPLACEMENT            *
 4       4    *--------------------------------------------------------------*
              *                       SAASACA                                *
              *                                                              *
              *            STORAGE ACCOUNTING CHAIN ADDRESS                  *
 8       8    ****************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	SAACFDC	4	Storage accounting
0	0	SAASCI	1	Storage class ID

Bits	Setting	Function
0-7	1000 1100	Chained storage (CLASS=USER,TEMPSTRG, TRANSDATA)
0-7	1000 1010	TCA storage
0-7	1000 0101	Chained terminal storage
0-7	1000 0000	Unchained storage

Dec.	Hex.	Field	Bytes	Function
1	1	SAASFI	1	Storage format ID
2	2	SAASAD	2	Storage area displacement
4	4	SAASACA	4	Storage accounting chain address

REGISTER STORAGE AREA (RSA)

DSECT NAME: DFHRSADS

The Register Storage Area (RSA) is obtained dynamically and is used by Program Control to store the user's registers (14 through 11) on a CICS LINK. The data stored in the Register Storage Area is restored when the user's program issues a CICS RETURN. If a RETURN is issued and there is no Register Storage Area, this indicates that the RETURN is from the highest level program and Program Control will terminate the transaction.

```
                       R E G I S T E R   S T O R A G E   A R E A
               ┌-------------------------------------------------------------┐
               *                                                             *
Dec.     Hex.  * <-----------------------4 BYTES---------------------------> *
               *                                                             *
 0        0    ***************************************************************
               *                                                             *
               *           STORAGE ACCOUNTING INFORMATION                    *
               *                                                             *
 4        4    *-------------------------------------------------------------*
               *                       RSAPCTA                               *
               *                                                             *
               *           PROCESSING PROGRAM TABLE ADDRESS                  *
 8        8    *-------------------------------------------------------------*
               *                       RSAPCSA                               *
               *                                                             *
               *           PROGRAM REGISTER STORAGE ADDRESS                  *
12        C    *-------------------------------------------------------------*
               *                                                             *
               *                       RESERVED                              *
               *                                                             *
16        10   *-------------------------------------------------------------*
               *                       PCSAR                                 *
               *                                                             *
               *              REGISTER STORAGE AREA (14-11)                  *
72        48   ***************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		4	Storage accounting information
4	4	RSAPCTA	4	Processing Program Table address
8	8	RSAPCSA	4	Program register storage address
12	C		4	Reserved
16	10	PCSAR	56	User registers (14-11)

LOAD LIST AREA (LLA)

DSECT NAME: DFHLLADS

The Load List Area (LLA) is obtained dynamically and is used by Program Control to store the PPT entry addresses of programs and/or tables that are loaded by a transaction. As DELETE's are issued for the previously loaded programs/tables, the entries are cleared in the Load List Area.

```
                          L O A D   L I S T   A R E A
                ┌──────────────────────────────────────────────────────────┐
                *                                                          *
Dec.    Hex.    * <---------------------4 BYTES---------------------------> *
                *                                                          *
 0       0      ************************************************************
                *                                                          *
                *              STORAGE ACCOUNTING INFORMATION              *
                *                                                          *
 8       8      *----------------------------------------------------------*
                *                          PCLLCA                          *
                *                                                          *
                *                 LOAD LIST CHAIN ADDRESS                  *
12       C      *----------------------------------------------------------*
                *                          PCLLPPT                         *
                *                                                          *
                *             PPT ENTRY ADDRESSES (ROOM FOR 5)             *
32      20      *----------------------------------------------------------*
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		8	Storage accounting information
8	8	PCLLCA	4	Pointer to next load list on chain
12	C	PCLLPPT	20	Space for five PPT entry addresses

SUBPOOL BOUNDARY BOX (SBB)

DSECT NAME: DFHSBBDS

```
                         S U B P O O L   B O U N D A R Y   B O X
                ┌────────────────────────────────────────────────────────────┐
                *                                                            *
Dec.    Hex.    * <──────────────────────4 BYTES──────────────────────────> *
                *                                                            *
  0       0     **************************************************************
                *                       SBBFIRST                             *
                *                                                            *
                *             ADDRESS OF FIRST FAQE IN CHAIN                 *
  4       4     *------------------------------------------------------------*
                *                       SBBLAST                              *
                *                                                            *
                *             ADDRESS OF LAST FAQE IN CHAIN                  *
  8       8     *------------------------------------------------------------*
                *                       SBBLOW                               *
                *                                                            *
                *             SUBPOOL LOWER BOUNDARY ADDRESS                 *
 12       C     *------------------------------------------------------------*
                *                       SBBSIZE                              *
                *                                                            *
                *                   SIZE OF SUBPOOL                          *
 16      10     **************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	SBBFIRST	4	Address of first free area queue element in chain
4	4	SBBLAST	4	Address of last free area queue element in chain
8	8	SBBLOW	4	Subpool lower boundary address
12	C	SBBSIZE	4	Size of subpool

FREE AREA QUEUE ELEMENT (FAQE)

```
                     F R E E   A R E A   Q U E U E   E L E M E N T
                ┌────────────────────────────────────────────────────────────┐
                *                                                            *
Dec.    Hex.    * <──────────────────────4 BYTES──────────────────────────> *
                *                                                            *
  0       0     **************************************************************
                *                       FAQESIZE                             *
                *                                                            *
                *                   SIZE OF FREE AREA                        *
  4       4     *------------------------------------------------------------*
                *                       FAQEADDR                             *
                *                                                            *
                *                  ADDRESS OF NEXT FAQE                      *
                **************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	FAQESIZE	4	Length in bytes of the free area
4	4	FAQEADDR	4	Address of next FAQE; zero if the last FAQE

TASK QUEUE AREA (TQA)

DSECT NAME: DFHTQADS

```
                         T A S K   Q U E U E   A R E A
                ┌─────────────────────────────────────────────────────────────┐
                *                                                             *
Dec.     Hex.   * <---------------------4 BYTES-----------------------------> *
                *                                                             *
  0        0    ***************************************************************
                *                                                             *
                *          STORAGE ACCOUNTING CONTROL DATA                    *
  4        4    *-------------------------------------------------------------*
                *                    TQANECA                                  *
                *                                                             *
                *          NEXT TASK QUEUE AREA CHAIN ADDRESS                 *
  8        8    *-------------------------------------------------------------*
                *                    TQAQEAA                                  *
                *                                                             *
                *          QUEUE ELEMENT AREA ADDRESS                         *
 12        C    *-------------------------------------------------------------*
                *                    TQAUSECT                                 *
                *                                                             *
                *          MULTIPLE ENQUEUING USE COUNT                       *
 16       10    ***************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		4	Storage accounting control data
4	4	TQANECA	4	Next task queue area chain address
8	8	TQAQEAA	4	Queue element area address
12	C	TQAUSECT	4	Multiple enqueuing use count

QUEUE ELEMENT AREA (QEA)

DSECT NAME: DFHQEADS

```
                      Q U E U E   E L E M E N T   A R E A
              ┌──────────────────────────────────────────────────────────────┐
              *                                                              *
Dec.   Hex.*  <-------------------4 BYTES---------------------------->       *
              *                                                              *
 0      0     ****************************************************************
              *                                                              *
              *            STORAGE ACCOUNTING CONTROL DATA                   *
 4      4     *--------------------------------------------------------------*
              *                        QEANECA                               *
              *                                                              *
              *             NEXT QUEUE ELEMENT CHAIN ADDRESS                 *
 8      8     *--------------------------------------------------------------*
              *                        QEAFQTA                               *
              *                                                              *
              *                 FIRST QUEUED TASK ADDRESS                    *
12      C     *--------------------------------------------------------------*
              *   QEAQARGL    *                QEAQARG                       *
              *               *                                              *
              *   ENQUEUED    *                                              *
              *   ARGUMENT    *            ENQUEUED ARGUMENT                 *
              *   LENGTH      *                                              *
              *  ATTRIBUTE    *                                              *
16     10     *---------------*                                              *
              *                                                              *
              *                                                              *
              ****************************************************************
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		4	Storage accounting control data
4	4	QEANECA	4	Next queue element chain address
8	8	QEAFQTA	4	First queued task address
12	C	QEAQARGL	1	Enqueued argument length attribute
12	C	QEAQARG	*	Enqueued argument

* Number of bytes are as indicated in QEAQARGL.
  If QEAQARGL is zero, the number of bytes is four.

WRITE REQUEST ELEMENT (WRE)

DSECT NAME: DFHWREDS
BASE REGISTER: WREBAR

The Write Request Element is a dynamic piece of main storage acquired by the Asynchronous Transaction Output Scheduler (CWTR) and chained onto the Batch Control Area at BCAWRE. One WRE is created for each valid, non-duplicate, output request for a batch. It contains information necessary for the transmission of batch data to a terminal destination.

WRITE REQUEST ELEMENT

```
              ┌───────────────────────────────────────────────────────────┐
              *                                                           *
Dec.   Hex.   * <-------------------4 BYTES-------------------------->   *
              *                                                           *
  0     0     *************************************************************
              *                                                           *
              *                    STORAGE ACCOUNTING                     *
  4     4     *-----------------------------------------------------------*
              *                        WRETRMID                           *
              *                                                           *
              *            OUTPUT TERMINAL INDENTIFICATION                *
  8     8     *-----------------------------------------------------------*
              *   WRECOPY      *       WREFLAG     *       WREXIT         *
              *                *                   *                      *
              *  NUMBER OF     *   CONTROL FLAGS   * USER EXIT PROGRAM    *
              *  COPIES        *                   * SUFFIX CHARACTERS    *
 12     C     *-----------------------------------------------------------*
              *                        WRECHAIN                           *
              *                                                           *
              *            ADDR OF NEXT WRE ON CHAIN                      *
 16    10     *************************************************************
```

Displacement						
Dec.	Hex.	Field	Bytes	Function		
0	0		4	Storage accounting field		
4	4	WRETRMID	4	Terminal ID to which output is to be be transmitted.		
8	8	WRECOPY	1	Number of copies to be transmitted		
9	9	WREFLAG	1	Processing flags		
				Bits	Settings	Function
		WRESCHED		0-7	1000 0000	Output has been Scheduled
		WREACT		0-7	0100 0000	CWTR is working on WRE
		WRESTAT		0-7	0010 0000	Status requested
10	A	WREXIT	2	User exit program suffix character		
12	C	WRECHAIN	4	Address of next WRE on chain		

COBOL AREA

DSECT NAME: DFHCRADS

```
                           C O B O L   A R E A
                 ┌─────────────────────────────────────────────────────────┐
                 *                                                         *
Dec.     Hex.    * <───────────────────4 BYTES──────────────────────────>  *
                 *                                                         *
 0        0      ***********************************************************
                 *                                                         *
                 *           STORAGE ACCOUNTING INFORMATION                *
                 *                                                         *
 8        8      *---------------------------------------------------------*
                 *                      PCTGTLOC                           *
                 *                                                         *
                 *                      COBOL TGT                          *
 n        n      *---------------------------------------------------------*
                 *                                                         *
                 *                   COBOL BLL CELLS                       *
                 *                                                         *
                 ***********************************************************
```

Dec.	Hex.	Field	Bytes	Function
0	0		8	Storage accounting information
8	8	PCTGTLOC		Beginning of task copy of ANS COBOL Task Global Table
n	n			First BLL cell (n = displacement found in PPTCOBLL of PPT + 8)

CICS-DL/I INTERFACE - PARAMETER LIST

DSECT NAME: DFHDLINT

The CICS-DL/I Interface Parameter List is used to pass data from DFHDLI to DFHDLQ and DFHDLE.

```
                         CICS-DL/I INTERFACE - PARAMETER LIST
              -------------------------------------------------------------
              *                                                           *
Dec.   Hex.   * <----------------------4 BYTES--------------------------> *
              *                                                           *
  0      0    *************************************************************
              *                                                           *
              *       HOUSEKEEPING CODE AT BEGINNING OF DFHDLI            *
              *                                                           *
 12      C    *-----------------------------------------------------------*
              *                         DFHIMTCB                          *
              *                                                           *
              *                   TCB ADDRESS OF DFHDLQ                   *
 16     10    *-----------------------------------------------------------*
              *                         DFHIMECB                          *
              *                                                           *
              *                   ECB THAT CONTROLS DFHDLQ                *
 20     14    *-----------------------------------------------------------*
              *                         DFHCIECB                          *
              *                                                           *
              * ECB CONTROLS CICS AT SYSTEM INITIALIZATION OR TERMINATION *
 24     18    *-----------------------------------------------------------*
              *                         DFHEXECB                          *
              *                                                           *
              *                   ECB THAT CONTROLS DFHDLE                *
 28     1C    *-----------------------------------------------------------*
              *                         DFHDLTCA                          *
              *                                                           *
              *       TCA ADDRESS OF TRANSACTION MAKING DL/I CALL         *
 32     20    *-----------------------------------------------------------*
              *                         DFHDLPST                          *
              *                                                           *
              *                   ADDRESS OF PST IN DFSBNUCO              *
 36     24    *-----------------------------------------------------------*
              *                         DFHDLPLS                          *
              *                                                           *
              *       ADDRESS OF PSB AND DMB POOL INITIALIZATION LISTS    *
 40     28    *-----------------------------------------------------------*
              *                         DFHDLMSG                          *
              *                                                           *
              *            ADDRESS OF MESSAGES RETURNED BY DL/I           *
 44     2C    *-----------------------------------------------------------*
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		12	Housekeeping code at beginning of DFHDLI
12	C	DFHIMTCB	4	TCB address of DFHDLQ
16	10	DFHIMECB	4	ECB that controls DFHDLQ
20	14	DFHCIECB	4	ECB that controls CICS at system initialization or termination
24	18	DFHEXECB	4	ECB that controls DFHDLE
28	1C	DFHDLTCA	4	TCA address of transaction making DL/I CALL

Displacement

Dec.	Hex.	Field	Bytes	Function
32	20	DFHDLPST	4	Address of PST in DFSBNUCO
36	24	DFHDLPLS	4	Address of PSB and DMB pool initialization lists
40	28	DFHDLMSG	4	Address of messages returned
44	2C			

CICS-DL/I INTERFACE - SCHEDULING BLOCK

DSECT NAME: ISB

The CICS-DL/I Interface Scheduling Block is used by DFHDLI during the execution of transactions using DL/I.

```
                         CICS-DL/I INTERFACE - SCHEDULING BLOCK
                  ┌──────────────────────────────────────────────────────────────┐
                  *                                                              *
Dec.      Hex.    * <-----------------------4 BYTES-----------------------------> *
                  *                                                              *
 0         0      ****************************************************************
                  *   ISBPROT      *                    ISBTSKID                 *
                  *                *                                             *
                  *  PROTECT KEY   *            TASK ID FROM TCAKCTTA            *
 4         4      *--------------------------------------------------------------*
                  *                          ISBFSQID                            *
                  *                                                              *
                  *                  FREE SPACE QUEUE ID OF TASK                 *
 8         8      *--------------------------------------------------------------*
                  *                          ISBPST                              *
                  *                                                              *
                  *                  ADDRESS OF PST FOR TASK                     *
12         C      *--------------------------------------------------------------*
                  *                          ISBTIME                             *
                  *                                                              *
                  *                  START TIME OF TASK                          *
16        10      *--------------------------------------------------------------*
                  *                          ISBPCBDV                            *
                  *                                                              *
                  *          ADDRESS OF STORAGE ACQUIRED FOR IMPLICIT PCB        *
                  *                      LIST IF PL/I PSB                        *
20        14      *--------------------------------------------------------------*
                  *                                                              *
                  *                          NOT USED                            *
                  *                                                              *
24        18      *--------------------------------------------------------------*
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0	ISBPROT	1	Protect key
1	1	ISBTSKID	3	Task ID from TCAKCTTA
4	4	ISBFSQID	4	Free space Queue ID of this task
8	8	ISBPST	4	Address of PST for this task
12	C	ISBTIME	4	Start time of task
16	10	ISBPCBDV	4	Address of storage acquired for Implicit PCB list if PL/I PSB
20	14		4	Not used

CICS-DL/I INTERFACE - DFHDLE ATTACH PARAMETER LIST

DSECT NAME: EXPARML

The DFHDLE Attach Parameter List is passed to DFHDLE when it is attached by DFHDLQ.

```
                    CICS-DL/I INTERFACE - DFHDLE ATTACH PARAMETER LIST
                 ┌────────────────────────────────────────────────────────────┐
                 *                                                            *
Dec.    Hex.     * <--------------------4 BYTES--------------------------->   *
                 *                                                            *
 0       0       **************************************************************
                 *                                                            *
                 *                         NOT USED                           *
                 *                                                            *
 4       4       *------------------------------------------------------------*
                 *                         DFHCSADL                           *
                 *                                                            *
                 *             ADDRESS OF CICS-DL/I INTERFACE                 *
 8       8       *------------------------------------------------------------*
                 *                         DFHSTPXP                           *
                 *                                                            *
                 *                 ADDRESS OF DL/I PXPARMS                    *
12       C       *------------------------------------------------------------*
                 *                         DFHSTLIP                           *
                 *                                                            *
                 *                 ADDRESS OF DL/I LIPARMS                    *
16      10       *------------------------------------------------------------*
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		4	Not used
4	4	DFHCSADL	4	Address of CICS - DL/I Interface
8	8	DFHSIPXP	4	Address of DL/I PXPARMS
12	C	DFHSTLIP	4	Address of DL/I LIPARMS

CICS-DL/I INTERFACE - LAYOUT OF I/O WORK AREA

The I/O Work Area is returned to the user after a retrieve call.

```
               CICS-DL/I INTERFACE - LAYOUT OF I/O WORK AREA
           -------------------------------------------------------------
           *                                                           *
Dec.   Hex.* <---------------------4 BYTES---------------------------> *
           *                                                           *
 0      0  *************************************************************
           *                                                           *
           *                   STORAGE ACCOUNTING                      *
           *                                                           *
 8      8  *-----------------------------------------------------------*
           *                                                           *
           *           BEGINNING OF RETRIEVED SEGMENT(S)               *
           *                                                           *
           *-----------------------------------------------------------*
```

Displacement

Dec.	Hex.	Field	Bytes	Function
0	0		8	Storage Accounting
8	8		Variable	Beginning of retrieved segment(s)

READER'S COMMENT FORM

Customer Information Control System (CICS)
Logic Manual

LY20-0714-2

Please comment on the usefulness and readability of this publication, suggest additions and deletions, and list specific errors and omissions (give page numbers). All comments and suggestions become the property of IBM. If you wish a reply, be sure to include your name and address.

COMMENTS

fold

fold

fold

fold

- Thank you for your cooperation. No postage necessary if mailed in the U.S.A.
FOLD ON TWO LINES, STAPLE AND MAIL.

YOUR COMMENTS PLEASE...

Your comments on the other side of this form will help us improve future editions of this publication. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material.

Please note that requests for copies of publications and for assistance in utilizing your IBM system should be directed to your IBM representative or the IBM branch office serving your locality.

fold fold

FIRST CLASS
PERMIT NO. 1359
WHITE PLAINS, N. Y.

BUSINESS REPLY MAIL

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY...

IBM Corporation
1133 Westchester Avenue
White Plains, N.Y. 10604

Attention: Technical Publications

fold fold

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

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

LY20-0714-2

Customer Information Control System LM Printed in U.S.A. LY20-0714-2

IBM

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

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