File No. S370-37
Order No. SY20-0888-0

IBM Virtual Machine Facility/370: System Logic and Problem Determination Guide Volume 3

Systems

Remote Spooling Communication System (RSCS)

Release 5 PLC 1

This publication is intended for the IBM system hardware and software support personnel. It provides the following information for the RSCS component of VM/370:

- Description of program logic
- Module descriptions and cross-references

PREREQUISITE PUBLICATIONS

IBM Virtual Machine Facility/370:

Introduction, Order No. GC20-1800
Operator's Guide, Order No. GC20-1806
Terminal User's Guide, Order No. GC20-1810
Remote Spooling Communications Subsystem User's Guide, Order No. GC20-1816
CP Command Reference for General Users, Order No. GC20-1820

IBM

| First Edition (December 1977)

| This edition, with SY20-0886-0 and SY20-0887-0 makes obsolete
| SY20-0885-2. It corresponds to Release 5 PLC 1 (Program Level Change)
| of the IBM Virtual Machine Facility/370 and to all subsequent releases
| until otherwise indicated in new editions or Technical Newsletters.
Extensive changes have been made to this publication; therefore, the user should read it in its entirety.

Changes are periodically made to the specifications herein; before using this publication in connection with the operation of IBM systems, consult the latest IBM System/370 Bibliography, Order No. GC20-0001, for the editions that are applicable and current.

Technical changes and additions to text and illustrations are indicated by a vertical bar to the left of the change.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, VM/370 Publications, Dept. D58, Bldg. 706-2, P.O. Box 390, Poughkeepsie, New York 12602. Comments become the property of IBM.

This publication provides the IBM system hardware and software support personnel with the information needed to analyze problems that may occur on the IBM Virtual Machine Facility/370 (VM/370).

HOW THIS MANUAL IS ORGANIZED

This manual comprises three volumes:

"Volume 1. VM/370 Control Program (CP)," "Volume 2. Conversational Monitor System (CMS)," and "Volume 3. Remote Spooling Communications Subsystem (RSCS)" contain the logic description for each of the components. Each of these volumes is divided into four sections: Introduction, Method of Operation, Directory, and Diagnostic Aids.

The method of operation and program organization sections contain the functions and relationships of the program routines in VM/370. They indicate the program operation and organization in a general way to serve as a guide in understanding VM/370. They are not meant to be a detailed analysis of VM/370 programming and cannot be used as such.

The directories contain descriptions of all the assemble modules in CP, CMS, and RSCS. They also contain extensive cross-references between modules and labels within a VM/370 component.

The diagnostic aids sections contain additional information useful for determining the cause of a problem.

The Appendix -- which is in Volume 1 -- contains a description of VM/370 Extended Control-Program Support (ECPS).

HOW TO USE THIS MANUAL

- Isolate the component of VM/370 in which the problem occurred.
- Use the list of restrictions in VM/370 Planning and System Generation Guide to be certain that the operation that was being performed was valid.
- Use the directories and use the VM/370 Data Areas and Control Block Logic to help you to isolate the problem.
- Use the method of operation and program organization sections, if necessary, to understand the operation that was being performed.

DEVICE TERMINOLOGY

The following terms in this publication refer to the indicated support devices:

- "2305" refers to IBM 2305 Fixed Head Storage, Models 1 and 2.
- "270x" refers to IBM 2701, 2702, and 2703 Transmission Control Units or the Integrated Communications Adapter (ICA) on the System/370 Model 135.
- "3330" refers to the IBM 3330 Disk Storage, Models 1, 2, or 11; the IBM 3333 Disk Storage and Control, Models 1 or 11; and the 3350 Direct Access Storage operating in 3330/3333 Model 1 or 3330/3333 Model 11 compatibility mode.
- "3340" refers to the IBM 3340 Disk Storage, Models A2, B1, and B2, and the 3344 Direct Access Storage Model B2.
- "3350" refers to the IBM 3350 Direct Access Storage Models A2 and B2 in native mode.
- "3704", "3705", or "370X" refers to IBM 3704 and 3705 Communications Controllers.
- The term "3705" refers to the 3705 I and the 3705 II unless otherwise noted.
- "2741" refers to the IBM 2741 and the 3767, unless otherwise specified.
- "3270" refers to a series of display devices, namely the IBM 3275, 3276, 3277, 3278 Display Stations. A specific device type is used only when a distinction is required between device types.

Information about display terminal usage also applies to the IBM 3138, 3148, and 3158 Display Consoles when used in display mode, unless otherwise noted.

Any information pertaining to the IBM 3284 or 3286 also pertains to the IBM 3287, 3288 and the 3289 printers, unless otherwise noted.

RSCS COMPONENT

The Remote Spooling Communication Subsystem (RSCS) VM/370 component provides for the transmission of files across a teleprocessing network controlled by a VM/370 computer. Using RSCS, virtual machine users can transmit files to remote stations. Also, users at remote stations can transmit files to VM/370 virtual machines and to other remote stations using RSCS.

PREREQUISITE PUBLICATIONS

IBM Virtual Machine Facility/370

Introduction, Order No. GC20-1800

Terminal User's Guide, Order No. GC20-1810

Remote Spooling Communications Subsystem (RSCS) User's Guide, Order No. GC20-1816

CP Command Reference for General Users, Order No. GC20-1820

COREQUISITE PUBLICATIONS

IBM Virtual Machine Facility/370

Data Areas and Control Block Logic, Order No. SY20-0884

Operator's Guide, Order No. GC20-1806

System Messages, Order No. GC20-1808

OLTSEP and Error Recording Guide, Order No. GC20-1809

Operating Systems in a Virtual Machine, Order No. GC20-1821

Service Routines Program Logic, Order No. SY20-0882

SUPPLEMENTARY PUBLICATIONS

IBM System/360 Principles of Operation, Order No. GA22-6821

IBM System/370 Principles of Operation, Order No. GA22-7000

IBM OS/VS, DOS/VS, and VM/370 Assembler Language, Order No. GC33-4010

IBM OS/VS and VM/370 Assembler Programmer's Guide, Order No. GC33-4021

In addition, for EREP processing the following OS/VS Library publications are required:

OS/VS Environmental Recording Editing and Printing (EREP) Program, Order No. GC28-0772

OS/VS Environmental Recording Editing and Printing (EREP) Program Logic, Order No. SY28-0773

Contents

FIGURES

SYSTEM LOGIC AND PROBLEM DETERMINATION GUIDE HAS BEEN REORGANIZED

Changed: Documentation only

VM/370 System Logic and Problem Determination Guide has been split into three volumes. Volume 1 contains the CP component, Volume 2 the CMS component, and Volume 3 the RSCS component.

The following material has been removed from this publication:

- "Introduction to Debugging" and "Debugging with CMS." This information can be found in VM/370 System Programmer's Guide.
- "Appendix A. VM/370 Coding Conventions." This information can be found in VM/370 System Programmers Guide.
- "Appendix B. DASD Record Formats." This information can be found in VM/370 Service Routines Program Logic in the FORMAT section.
- "Appendix C. VM/370 Restrictions." This information can be found in VM/370 Planning and System Generation Guide.
- "Appendix D. Applying PTFs." This information can be found in VM/370 Planning and System Generation Guide.

The following sections have been removed from the "CMS Diagnostic Aids" section of this publication:

- ZAP Service Program. A complete description of ZAP can be found in VM/370 Operator's Guide.
- DDR. A complete description of DDR can be found in VM/370 Operator's Guide.
- CMS Return Codes. These can be found in VM/370 System Messages.
- Commands for Debugging. A complete description of DEBUG can be found in VM/370 CMS User's Guide.

The following topics have been removed from "CP Diagnostic Aids":

- CP Commands Used to Debug the Virtual Machine. These are contained in VM/370 CP Command Reference for General Users.
- CP Commands for System Programmers. These are contained in VM/370 Operator's Guide.

MISCELLANEOUS

Changed: Programming and Documentation

Minor technical and editorial changes have been made in order to clarify the text.

RSCS Introduction

The section provides the following information:

- Remote Spooling Communications Subsystem: Overview
- NPT Line Driver Program
- Synchronizing and Dispatching Tasks

Remote Spooling Communications Subsystem: Overview

The VM/370 Remote Spooling Communications Subsystem (RSCS) is the VM/370 component that provides for the transmission of files across a teleprocessing network controlled by the VM/370 computer. Using RSCS, virtual machine users can transmit files to remote stations. (Remote stations are I/O configurations attached to the VM/370 computer by communications lines.) Also, users at remote stations can transmit files to VM/370 virtual machines and to other remote stations using RSCS.

RSCS resides in a virtual machine dedicated to remote spooling. Using the RSCS command language, the RSCS operator manages the telecommunications facilities for the installation.

Operators at remote stations can manage their own configurations using a subset of the command language. Commands issued from remote stations can be entered either at a terminal or from a card reader.

You can find detailed descriptions of RSCS functions in the publication VM/370 Remote Spooling Communications Subsystem (RSCS) User's Guide.

The RSCS Virtual Machine and the VM/370 Control Program (CP)

Like the other VM/370 virtual machines, the RSCS virtual machine runs under the control of CP. In extending the VM/370 spooling system capability to include spooling to remote stations, RSCS interacts with the CP spooling system. Therefore, some of the information in this publication requires a knowledge of that area of CP.

The RSCS virtual machine consists of the virtual machine operator console, an RSCS system disk, and virtual telecommunications lines. During system generation, a virtual card reader is defined for the RSCS virtual machine, but this reader does not exist in the CP directory entry for the RSCS virtual machine.

Virtual printers, card punches, and readers are defined dynamically as they are needed. For example, when a file from a remote station is transmitted to RSCS, a virtual punch is defined to accept the file. Similarly, virtual readers are defined when RSCS receives a file to transmit. RSCS virtual storage also dumps onto a virtual printer when abnormal termination of the system occurs. Figure 1 shows the configuration of an RSCS virtual machine.

The minimum virtual storage required to run RSCS is 512K.

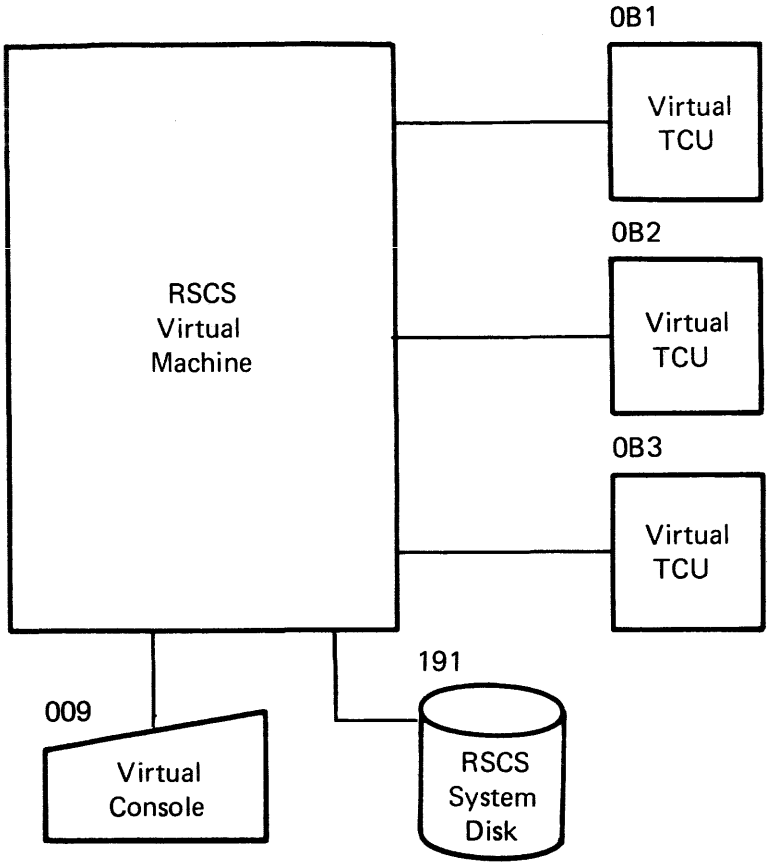


Figure 1. RSCS Virtual Machine Configuration

Locations and Links

At a local installation there are a number of transmission paths to remote stations. A unique location identifier (locid) is assigned to each of these remote stations.

For each transmission path (nonswitched line) or potential transmission path (switched line), a link must be defined at the local VM/370 installation. Each such link is given a name (linkid) that defines the location identifier of the remote station to which the transmission path leads. This link can be defined either at system generation or by means of the DEFINE command.

REMOTE STATIONS

Remote stations are configurations of I/O devices attached to the VM/370 computer by binary synchronous (BSC) switched or nonswitched lines. Two types of remote stations are supported by RSCS: programmable remote stations and nonprogrammable remote stations.

Programmable Remote Stations

Programmable remote stations, such as the IBM System/3 and System/370, are IBM processing systems with attached binary synchronous communications adapters. These systems must be programmed to provide the

MULTI-LEAVING line protocol necessary for their devices to function as remote stations. This programming support is provided by a remote terminal processor (RTP) program generated according to HASP workstation protocol and tailored to the system's hardware configuration. Certain programmable remote stations like the System/3 can only be programmed to function as remote terminals. Others, like the System/360 and System/370, can function either as remote terminals or as host batch systems using RSCS as a remote job entry workstation. Both of these types of remote stations are managed by the spool MULTI-LEAVING (SML) line driver of RSCS.

Nonprogrammable Remote Stations

Nonprogrammable remote stations are I/O configurations that cannot be programmed, but are hard-wired to provide the line protocol necessary for them to function as remote stations. They can receive, read, print, punch, and send files. An example of a nonprogrammable remote station is a 2780 Data Transmission Terminal. Nonprogrammable remote stations are managed by the NPT (Nonprogrammable Terminal) RSCS line driver.

The types of devices supported for all types of remote stations, programmable and nonprogrammable, are listed in the VM/370 Remote Spooling Communications Subsystem (RSCS) User's Guide.

Network Control: RSCS and VM/370 Commands

Both RSCS and VM/370 commands are used to control RSCS. The RSCS commands are used to control the RSCS network; VM/370 CP and CMS commands are used by virtual machine users who use the RSCS network.

RSCS COMMANDS

To manipulate the file being transmitted across the network and to communicate with the various network users, the RSCS control program provides a command language. Figure 2 is a list of RSCS commands and the functions they perform. You can find detailed descriptions of these commands in the publication VM/370 Remote Spooling Communications Subsystem (RSCS) User's Guide.

The operator may enter RSCS commands described in Figure 2 at the RSCS virtual machine console. A subset of the RSCS command language may be entered by operators of remote stations.

Command Name	Function
BACKSPAC	Restarts or repositions in a backward direction the file currently being transmitted.
CHANGE	Alters one or more attributes of a file owned by RSCS.
CMD	Controls certain functions performed by a remote system, or controls the logging of I/O activity on a specified link.
DEFINE	Temporarily adds a new link definition to the RSCS link table or temporarily redefines an existing link.
DELETE	Temporarily deletes a link definition from the RSCS link table.
DISCONN	Places RSCS in disconnect mode and optionally directs output to another virtual machine.
DRAIN	Deactivates an active communication link.
FLUSH	Discontinues processing the current file on the specified link.
FREE	Resumes transmission on a communication link previously in HOLD status.
FWDSPACE	Repositions the file currently being transmitted in a forward direction.
HOLD	Suspends file transmission on an active link without deactivating the line.
MSG	Sends a message to a local or remote station.
ORDER	Reorders files enqueued on a specific link.
PURGE	Removes all or specified files from a link.
QUERY	Requests system information for a link, a file, or for the system in general.
START	Activates a specified communication link.
TRACE	Monitors line activity on a specified link.

Figure 2. RSCS Commands and Functions

VM/370 CP AND CMS COMMANDS FOR RSCS

The VM/370 CP TAG and SPOOL commands specify a device to be spooled and to associate a destination location identifier (locid) with that device. SPOOL directs the file to the RSCS virtual machine. The CP CLOSE

command or the CMS PRINT or PUNCH commands close the file and transfer it to the RSCS virtual machine.

Data specified by the CP TAG command controls processing of files transmitted across the RSCS network. When a VM/370 user creates a file to be transmitted to a remote station via RSCS, the TAG command text operand takes the following format:

linkid [userid] [priority]

where:

linkid is the location identifier of the link on which the file is to be transmitted.

userid is the remote virtual machine that is to receive the file.

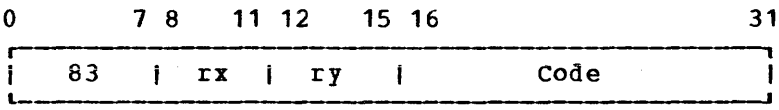
priority is the requested transmission priority (a decimal number 0-99, default 99). The lower numbers have higher priorities.

Also, the CP SPOOL command directs files to the RSCS virtual machine. See the publication For details on how to use the CP TAG and SPOOL commands to control RSCS network functions, see the VM/370 Remote Spooling Communications Subsystem (RSCS) User's Guide.

CP Instructions Used by the RSCS Control Program

When RSCS handles files being transmitted across the network, the RSCS control program (line driver tasks) issues CP DIAGNOSE instructions.

The DIAGNOSE instruction is the method of communication between a virtual machine and CP. In VM/370, the machine-coded format for the DIAGNOSE instruction is:



Content	Explanation
83	DIAGNOSE operation code
rx	User-specified register number
ry	User-specified register number
Code	Hexadecimal value that selects a particular CP function.

Figure 3 lists the DIAGNOSE function codes used by RSCS, the functions of those codes, and the RSCS modules from which they are issued.

The RSCS Control Program

RSCS is a control program composed of a multitasking supervisor and multiple tasks, which are controlled by the supervisor.

DIAGNOSE Code	Function	Issued by Module(s)
0008	Executes a CP command.	DMTAXS DMTREX DMTCMX DMTMGX DMTSML DMTNPT
000C	Gets the current time and date.	DMTSML DMINPT
0014	Manipulates input spool files.	DMTAXS DMTSML DMTNPT
0020	Performs general I/O without interrupt.	DMTINI
0024	Determines virtual device type information.	DMTREX DMTLAX DMTSML
005C	Edits error messages.	DMTREX

Figure 3. VM/370 DIAGNOSE Instructions Issued by the RSCS Program

The supervisor provides only those functions that cannot be consistently provided by the tasks themselves; that is, the supervisor provides only the support necessary to control and coordinate the execution of the tasks.

In RSCS, a task is a single program or set of subprograms that can run concurrently and autonomously with other such programs and subprograms, and which uses control functions provided by the Supervisor.

There are two types of tasks: system service tasks and line driver tasks. The system service tasks are those that provide the system support functions for the supervisor and for other tasks. The line driver tasks are those that manage the transmission paths to remote stations and that interact between the remote stations and the system service tasks and the Supervisor. Each line driver task manages the transmission of files to and from a single remote station.

Figures 12 and 13 in Section 2 show the communications paths between the supervisor, system service tasks, line driver tasks, remote stations, and VM/370 virtual machines.

The RSCS Supervisor

The RSCS supervisor is composed of a set of service routines that provide functions for the tasks that run under them. These service routines may be called by any task. In general, they provide four kinds of services:

- Task management
- I/O management
- Interrupt handling
- Virtual storage management

Task Management

The task management service routines provide three kinds of services: task execution control, task synchronization, and task-to-task communication.

Task execution control includes initiating and terminating tasks. In general, the only task to request these services is the REX system control task, which is described below. Task execution control also includes the dispatcher, DMTDSP, which activates task execution as soon as that task is initiated and while the task is active.

Task synchronization comprises a mechanism by which tasks are made ready or not ready for execution. When a task requests the services of another task, the requestor task may suspend its execution while the request is being processed. The synchronization mechanism that accomplishes this consists of two routines, DMTWAT and DMTPST. DMTWAT causes the requestor task to temporarily halt execution. DMTPST causes a temporarily-halted task to resume execution. For more information on task synchronization refer to the section "Synchronizing and Dispatching Tasks"

There are two types of task-to-task communications: (1) the DMTSIG routine (ALERT) and (2) the DMTGIV and DMTAKE routines (GIVE/TAKE).

The DMTSIG routine allows a task to immediately interrupt another task to pass it information. The interrupted task must have an asynchronous exit routine defined to handle the interruption. Functionally, DMTSIG performs a function analagous to an SVC instruction.

The DMTGIV and DMTAKE routines allow tasks to exchange information buffers with other tasks. The GIVE/TAKE function provides the means for organized enqueuing and delivery of requests for services or information from one task to another.

For more information on task-to-task communications, refer to the section "Task-to-Task Communications" in this section.

I/O Management

I/O management for tasks consists of the following functions:

- Handling requests for I/O operations
- Handling I/O interrupts
- Starting an I/O operation
- Completing an I/O request

Whenever a task requests the services of the I/O manager, that task builds an I/O request table to be passed to the I/O manager. This table consists of the following information:

- A synchronization lock for signaling I/O completion
- The address of the device on which the I/O operation is to take place
- The number of SENSE bytes to be returned, when applicable
- The address of the channel program to be executed

The following information is returned to the task by the I/O manager, in the I/O request table:

- The condition code for the SIO issued for the I/O operation
- The compcsite CSW
- The SENSE bytes returned by the operation (if any)

Using the information in this table, the I/O manager enqueues the request on the specified subchannel, starts the I/O operation, assembles the return information in the requestor's I/O request table, and posts the synchronization lock in the I/O request table signalling that the I/O operation is complete.

Interruption Handling

Supervisor service routines handle three kinds of interruptions: external interruptions, SVC interruptions, and I/O interruptions.

In RSCS, supervisor routines use the SVC (SUPERVISOR CALL) to suspend the execution or dispatching of a task when that supervisor routine received control. On an SVC interruption in RSCS, DMTSVC is entered. DMTSVC saves the status of the executing task and passes control to the calling supervisor routine in supervisor execution mode.

RSCS handles external interruptions from tasks by searching for asynchronous exit requests supplied by tasks. When a request with a code matching the external interruption code is found, its asynchronous exit is taken; otherwise, the external interruption is ignored.

I/O interruptions are handled by the RSCS I/O manager. When an active I/O request causes an I/O interruption, the status of the I/O request is updated to reflect the new information. Otherwise, a search is made for an asynchronous exit request for the interrupting device. When one is found, the asynchronous exit is taken. Otherwise, the interruption is ignored.

Virtual Storage Management

The supervisor virtual storage service routine DMTSTO handles requests by tasks for main storage. When a task requests main storage, DMTSTO reserves page(s) of storage for it. Main storage is freed directly by task programs.

DMTQRQ manages requests for free elements of the supervisor status queue. Supervisor routines call DMTQRQ to reserve and release supervisor status queue elements.

RSCS Task Structure

As described in the previous section, the RSCS supervisor comprises a set of routines that function together to manage RSCS system processing. The supervisor provides a base for many system programs called tasks. (These tasks are not to be confused with user-application programs.)

The RSCS system service tasks perform less generalized functions for the system than those functions performed by the supervisor. For example, the AXS system service task is designed specifically to access the VM/370 spool file system.

The supervisor identically manages all tasks in RSCS; the supervisor makes no distinction between system service tasks and line driver tasks. Figure 4 is a list of the RSCS tasks and a brief statement of the service each performs.

Task Name	Module Name	Function
REX	DMTREX	Handles console I/O; accepts requests for services passed by other system service tasks or line driver tasks; terminates a task; handled program check interruptions.
	DMTCRE	Creates a system service or line driver task.
	DMTCMX	Monitors processing of commands in RSCS; executes the DEFINE, DELETE, DISCONN, QUERY, and START commands.
	DMTMGX	Builds a message element and passes the element to the appropriate tasks for transmission or printing.
	DMTCOM	Performs common task functions.
AXS	DMTAXS	Communicates with the spool file system.
LAX	DMTLAX	Manages telecommunications line allocation.
Line Driver	DMTSML	Manages a telecommunications line for a programmable remote station using RTAM.
	DMTNPT	Manages a telecommunications line for a nonprogrammable remote station terminal.

Figure 4. RSCS Tasks

CREATE SYSTEM TASKS: DMTCRE

The main system service task, REX, is loaded with the supervisor during RSCS initialization. The REX task, in turn, creates other tasks required by the system. DMTCRE reads these other tasks from a CMS disk by means of a CMS read access method. The task is then started as a new active task under RSCS.

PROCESS COMMANDS: DMTCMX

DMTCMX receives commands by means of either GIVE request elements passed by line driver tasks or in the form of a console input line resulting from a console read by DMTREX.

The commands DEFINE, DELETE, DISCONN, QUERY, and START (for inactive links) are executed by DMTCMX. Execution of these commands generally involves referencing and modification of system status tables (SVECTORS, TTAGQ, TLINKS, etc.).

If the command is not one that DMTCMX executes within its own code, the command line is examined for syntax errors and then passed to the appropriate task for execution. To do this, DMTCMX generates a formatted table called a command element to be passed to another active task for execution via an ALERT asynchronous exit.

The commands CHANGE, ORDER, and PURGE are executed by DMTAXS; the commands BACKSPAC, CMD, DRAIN, FLUSH, FREE, FWDSPACE, HOLD, MSG, TRACE and START (for active links) are executed by the line driver task for the specified link.

PROCESS MESSAGES: DMTMGX

DMTMGX manages distribution of all RSCS messages, which may be generated by REX or by any other RSCS task. Each message to be issued is presented to DMTMGX (via GIVE/TAKE for tasks other than REX) along with an internal routing code and an internal severity code.

Messages may be addressed to the local RSCS operator console, to the local VM/370 operator, to a local VM/370 user console, to a remote station operator, or to any combination of these destinations, by means of the routing code. The severity code is defined for each message, and is an indication of the importance of the message.

Messages for the RSCS local operator console are enqueued for output on the RSCS virtual machine console. Messages for the local VM/370 system operator and for local virtual machine consoles are issued by means of execution of a VM/370 MESSAGE command (through the DIAGNOSE interface). Messages for remote RSCS operators are presented to the line drivers for the associated links by means of the RSCS MSG command element interface. This method of message handling simplifies RSCS message routing, tracing, and recording.

TERMINATE SYSTEM TASKS AND HANDLE PROGRAM CHECKS: DMTREX

When a line driver task requests termination, a TAKE request is passed to DMTREX specifying that function. DMTREX marks the task as terminated, then searches for active I/O associated with the task. If active I/O is found, it is terminated. To ensure that system integrity is maintained during the termination of the I/O, a mechanism (at label QUIESE) is set up to handle situations in which an HIO (Halt I/O instruction) does not take effect immediately.

All RSCS program checks are handled by a routine in DMTREX. Program check diagnostic information is dumped, a message to the operator is issued, and the RSCS system status is modified, depending on the nature of the program check.

COMMUNICATE WITH THE VM/370 SPOOL FILE SYSTEM: DMTAXS

DMTAXS is responsible for the maintenance of the total RSCS interface to the VM/370 spool system. When a spool file arrives at the RSCS virtual machine, AXS receives the associated asynchronous interrupt, reads and interprets the file's VM/370 spool file block (SFBLOK) and TAG, enqueues the file for transmission as appropriate, and notifies the appropriate line driver of the new file's availability. AXS provides a GIVE/TAKE request interface to line driver tasks for spool file data input and output, and defines and detaches virtual spool I/O devices as necessary. Also, AXS provides an interface to DMTCMX for second-level command execution support.

AXS maintains a queue of a fixed number of virtual storage elements (called tag slots) that describe files currently owned by the RSCS virtual machine. To maintain RSCS integrity in a simple way when a very large number of files is enqueued on the RSCS virtual machine, the virtual storage tag queue is not extended during execution.

When a new file arrives at the RSCS virtual machine, its destination locid is examined, and it is accepted only if there is a matching linkid for which there is a free tag slot available. If the file's destination locid is not defined as a linkid, the file is purged and the originating user is notified of the action. If there is no free tag slot available for a defined linkid, the file is left "pending", and is accepted when a TAG slot becomes free. While a file is pending, it is not recognized by the RSCS command processors, and cannot be referenced through RSCS functions.

To prevent links from being totally locked out by an exhausted (and stagnant) virtual storage tag queue, a minimum number of tag slots is reserved for each link. This guarantees that a minimum number of files is accepted for each associated link. The number of reserved slots is defined during system generation or in the DEFINE command for each link to be defined in RSCS. The appropriate number of slots to be reserved for each link may depend on the expected file traffic, the link's line speed, the expected time the link is to be active, and the desired level of service to be provided to the link. This number for each link may be arrived at through actual operational experience in each location.

MANAGE TELECOMMUNICATION LINE ALLOCATION: DMTLAX

DMTLAX is responsible for line port resource allocation to line driver tasks. DMTLAX allocates available switched ports (when a link is activated without a specified line address) through an ALERT request interface. When a line port is specifically requested (by virtual address), DMTLAX checks the device for validity as a line port.

LINE DRIVER TASKS: DMTNPT AND DMTSML

As part of the link activation process, REX (module DMTCRE) loads and starts a line driver task to service the remote location.

The general functions of line driver tasks are:

- Manage I/O on the BSC line

- Manage transmission of spool file data via a GIVE/TAKE request to the AXS task
- Provide GIVE/TAKE requests to the REX task command module (DMTCMX)

The precise functional requirements vary from line driver to line driver, depending on the type of remote station the line driver supports.

Each line driver is responsible for maintenance of its link status and line activity (TRACE) records in the RSCS system status tables.

Two line drivers are provided, one to support remote 2770, 2780, 3770 (in 2770 mode), and 3780 terminals, and another to interface to remote HASP- and ASP-type systems or work stations.

The SML Line Driver Program

The SML line driver program is composed of four general types of routines:

- Processors, which are routines that execute the functions required by the HOST and RJE processing modes.
- An input/output routine that accepts and transmits data on the BSC line.
- A function selector routine that dispatches one of the processors when a request for services is received.
- Buffer blocking and deblocking routines.

The SML line driver supports programmable remote stations (in both HOST and RJE modes) for HASP- and ASP-type systems. HOST mode is that processing mode in which a remote station may submit jobs to VM/370 and receive print and punch output from VM/370. RJE mode is that processing mode in which VM/370 may send jobs to a remote batch system for processing and receive print and punch output from the remote batch system.

Figure 5 shows the types of data flowing to and from RSCS via the SML line driver program.

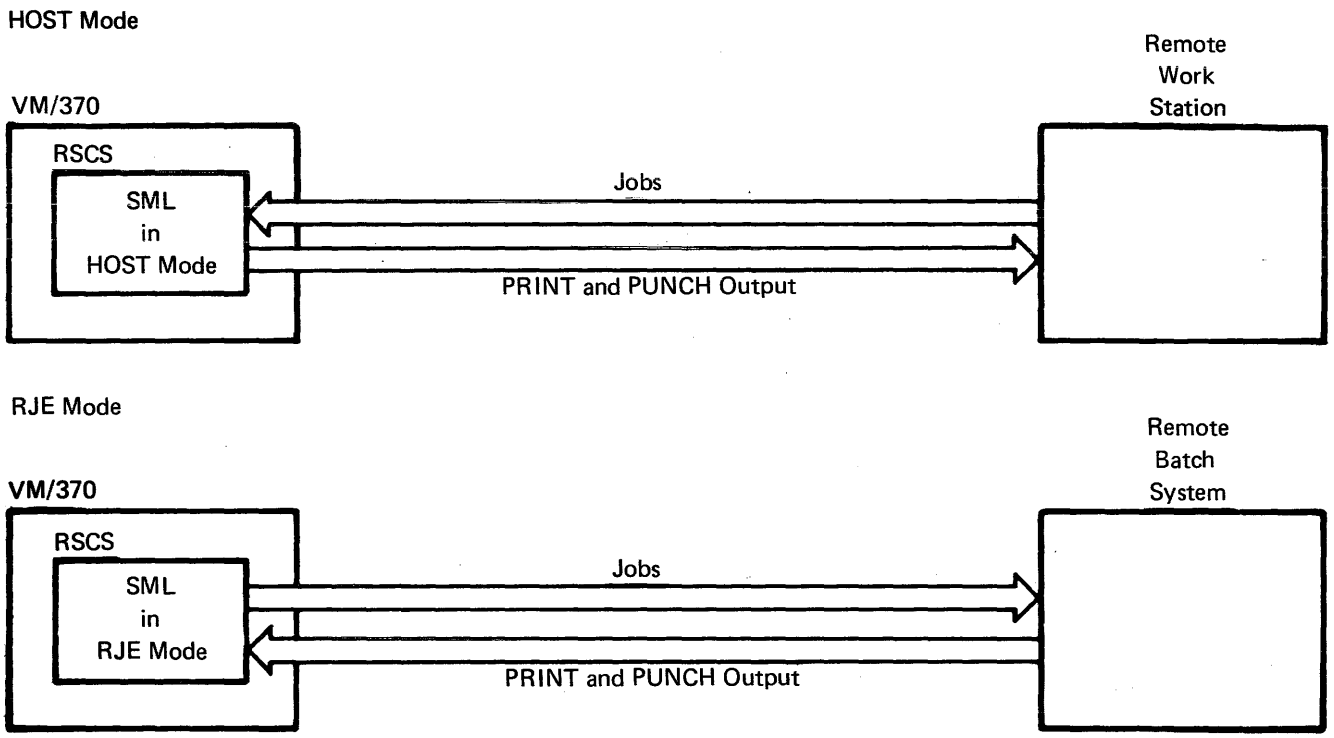


Figure 5. Data Flow between RSCS and Remote Stations via the SML Line Driver

SML PROCESSORS

To support the HOST and RJE processing modes, the SML program provides seven "processors," or routines, that handle the seven functions required to support the two processing modes. Figure 6 is a list of the SML processors, the processing modes they support, and a brief statement of their function.

Command Processing

When a command is transmitted from a remote station to RSCS, SML receives the command and coordinates processing of the command with supervisor routines and the REX task command module DMTCMX.

The SML processor, $WRTN1, processes a command request from a remote station by passing a command request element to the REX task (module DMTCMX) via a GIVE request. DMTCMX then determines whether the command should be executed by DMTCMX, DMTAXS, or by the line driver. If the command is to be executed by the line driver, it is passed back to SML via an ALERT request. The SML routine CMDPROC then executes the command.

Processor	Mode	Function
$CRTN1	HOST/RJE	Processes the following MULTI-LEAVING control records: permission to transmit, request to transmit, and SIGNON control records.
$PRTN1	RJE	Processes print file records received from remote stations and passes them to the VM/370 spool system.
$URTN1	RJE	Processes punch file records received from remote stations and passes them to the VM/370 spool system.
$JRTN1	HOST	Processes job file records received from the remote station and passes them to the VM/370 spool system.
$WRTN1	HOST/RJE	In HOST mode, passes command request elements, via DMTMGX, to DMTCMX for processing. In RJE mode, passes message request elements to the RSCS operator's console.
$RRTN1	HOST/RJE	Receives records from the VM/370 spool system for transmission to remote stations.
CMDPROC		Executes local commands passed by DMTCMX, and passes messages and commands to remote stations.

Figure 6. SML Function Processors

THE SML LINE I/O HANDLER ROUTINE: COMSUP

The SML line I/O handler routine, COMSUP, controls communications on the BSC line for SML. This routine receives data from the BSC line and passes the data to the deblocker routine ($TPGET). COMSUP also sends data (which has been blocked by the blocker routine, $TPPUT) to a remote station. COMSUP is also responsible for acknowledging receipt of data over the line using the standard BSC line control characters.

THE SML FUNCTION SELECTOR ROUTINE: $START

The $START routine is entered when SML is required (by either a remote station or a virtual machine) to perform a function. The purpose of this routine is to select a function to execute. The routine performs this function by using a commutator table, a list of synch locks, and task control tables.

The SML commutator table is a branch table consisting of branch (B) and no-operation (NOP) instructions. The targets of the branch instructions are the seven processor routines, each of which performs a specific function. When the service of a processor is not required, the Commutator Table entry for that processor is a NOP instruction. When the function of the processor is required, the NOP instruction in the commutator table entry for that processor is replaced with a B instruction, thereby opening a gate in the commutator table.

The $START routine cycles through the commutator table, falling through any NOP instructions and taking any branches. Control is passed in this way to any processor whose gate in the commutator table is open.

When the processor completes the function requested, it closes its gate in the commutator table by replacing the B instructions with a NOP instruction. $START continues cycling through the commutator table taking any open branches.

When the bottom of the commutator table is reached, $START tests a series of synch locks to see if any have been posted, signifying a request for an SML function. If any synch locks are posted, $START opens the commutator table gate for the requested processor and goes to the top of the commutator table to start cycling through it again.

If the bottom of the commutator table is reached and there are no posted synch locks, SML discontinues processing by issuing a wait request via a call to the supervisor module DMTWAT, waiting on a list of the synch locks. When any of the synch locks is posted, $START receives control, opens the appropriate gate, and starts cycling through the commutator table.

The task control table (TCT) is a DSECT defining data required by each of the processors. There is a TCT for each of the processors. Also, contained within the TCT is a branch instruction to the appropriate processor.

BLOCK AND DEBLOCK SML TELEPROCESSING BUFFERS: $TPPUT AND $TPGET

Data received over the BSC line is placed in a teleprocessing (TP) buffer. The size of TP buffers is specified by a START command parameter and can be up to 1024 bytes.

Data contained in TP buffers is deblocked into tanks, which are unit buffers of a specific size used to deblock the larger TP buffers. There are 15 tanks; these are allocated as they are needed by processors. The size of tanks is determined by MULTI-LEAVING control bytes.

When an SML function has been requested, the data must be either blocked for transmission (if it is data for a remote station) or deblocked for processing (if it has been received from a remote station).

$TPGET receives data from a BSC line (via the COMSUP routine) and allocates tanks to output processors as they are needed.

$TPPUT receives tanks from input processors, blocks the data in these tanks into TP buffers, and gives control to COMSUP to transmit the buffers over the line.

The NPT Line Driver Program

The NPT line driver program processes only one file at a time; it can either receive a file as input from the remote station or transmit an output file to a remote station. These two processes execute under control of a line monitor that reads and writes data over the BSC line and a function selector routine that determines whether an input or output function has been requested.

THE NPT LINE MONITOR ROUTINE: LINEIO

The NPT line monitor routine, LINEIO, controls communications on the BSC line. This routine sends and receives data over the BSC line.

When the data is received from remote stations, that data is received in the LINEINB buffer. When data is transmitted to a remote station, it is transitted using the LINEBUFF buffer. The NPT buffers are a fixed size, defined by terminal type and buffer size specified on the SIGNCN card.

THE NPT FUNCTION SELECTOR ROUTINE: NPTGET

When the NPT line driver program has been loaded and initialized, the NPTGET program begins a cycle in which it checks every three seconds for one of three functions to perform:

- Process a command
- Read a file from a remote station
- Write a file to a remote station

When a function is requested, a branch is taken to the appropriate routine.

NPT INPUT FILE PROCESSING

For files being received from remote stations, two processing routines are executed: PUTVRFY and PUTBLOCK. PUTVRFY reads the data contained in the input buffer (LINEINB) and verifies the BSC control characters for that data. PUTBLOCK deblocks the data in LINEINB, formats it for use by VM/370, and then writes the data to the VM/370 spool system.

NPT OUTPUT PROCESSING ROUTINES

For files being transmitted to a remote station, three processing routines are executed: MAKEBLOC, GETBLOCK, and GETVRFY.

MAKEBLOC accepts a block of data from the VM/370 spool system and passes control to GETBLOCK. GETBLOCK then builds a buffer with which to transmit the data and transmits the data to the remote station. The response received from that transmission is analyzed by GETVRFY.

Major Data Areas

The major data areas used by RSCS are:

- SVECTORS
- RSCS supervisor queue elements
- MAINMAP
- TAREA
- LINKTABL
- TAG
- RSCS request elements
- VM/370 data areas referenced by RSCS

The data areas discussed below give a brief functional overview of each data area and its relationship to other data areas in the system. This is not meant to be a comprehensive description of the RSCS data areas. Rather, it is meant as an introduction to the types of data used by RSCS in performing its various functions.

SVECTORS: SUPERVISOR CONTROL QUEUES AND SUPERVISOR ROUTINE ADDRESSES

The SVECTORS DSECT contains:

- The PSW for the last task dispatched
- The RSCS System Save area
- The task ID and task element address for the last task dispatched
- Pointers to the RSCS supervisor subqueues
- Entry addresses for all supervisor service routines

This data area is updated dynamically as tasks execute and is used by RSCS to monitor the execution status of the system.

RSCS SUPERVISOR QUEUE ELEMENTS

All supervisor status information pertaining to tasks and task requests is maintained in Supervisor storage defined by the SVECTORS DSECT. There are various queues defined in this DSECT, each pertaining to a particular Supervisor function, and composed of elements of similar format. The heads of these queues are defined in a portion of SVECTORS from FREEQ through GIVEQ. The DSECTS defining the elements chained on these queues are: FREEE, TASKE, IOE, ASYNE, and GIVEE.

MAINMAP: STORAGE AVAILABLE TO RSCS PROGRAMS AND TASKS

The MAINMAP DSECT is a grid of a fixed number of bytes, each of which represents a page of virtual storage. When a task (or the Supervisor) requests storage, the byte is filled with the TASKID (generated by the Supervisor) of the requestor, thus marking the storage page as taken by that task. When a page is free, its map entry is cleared to zero by the task owning the storage.

TAREA: THE SAVE AREA FOR AN INTERRUPTED TASK

The TAREA DSECT contains the PSW at which a task is to resume execution, the contents of the task general registers when it was interrupted, and the task's request synchronization lock. This area is used to maintain the status of a task when it is interrupted by another task.

LINKTABL: LINK DESCRIPTION DATA

The LINKTABL DSECT describes control data associated with each link in the system. The control data includes such information as the linkid of the link, the task name for the link's line driver (that is, the name by

which RSCS knows the task), the address of the line which is used by the link, and so on. The link table (a chain of LINKTABL DSECTs) is built during system generation and may be updated by the DEFINE, DELETE, START, and DRAIN commands.

TAG: THE RSCS FILE DESCRIPTOR

The TAG DSECT defines the attributes and status of a file being processed by RSCS. The TAG is built from information passed via the CP TAG command (or its counterpart for remote stations) and from the CP Spool File Block (SFBLOK) that describes the file.

RSCS REQUEST ELEMENTS

Request elements are data tables built by task programs when a service is to be requested by the task.

For example, when a command is processed by DMTCMX, the command line may be formatted into a command element, which gives the following types of information:

- Length of the command element
- The unique code identifying the command element
- The linkid to which command response is to be returned
- Modifiers that specify options for a given command
- A variable length buffer field containing the command line

This command element is then passed (via DMTSIG) to another task for processing.

Other types of request elements are built to process individual commands and messages, to create and terminate tasks, to process console I/O, and so on.

In many cases, elements are contained in a generalized control area used when processing a system function, for example, monitoring requests for DMTAXS module to open or close a VM/370 spool file.

VM/370 DATA AREAS REFERENCED BY RSCS

There are two VM/370 CP data areas referenced by RSCS when VM/370 spool files are processed:

- SFBLOK The VM/370 spool file block that contains control information and describes attributes of a VM/370 spool file.
- SPLINK The data block that links pages of a VM/370 spool file buffer.

RSCS Storage Requirements

Figure 7 shows the storage used by the RSCS control program and how the parts of the system (the Supervisor, the tasks, and the data areas) fit together in storage.

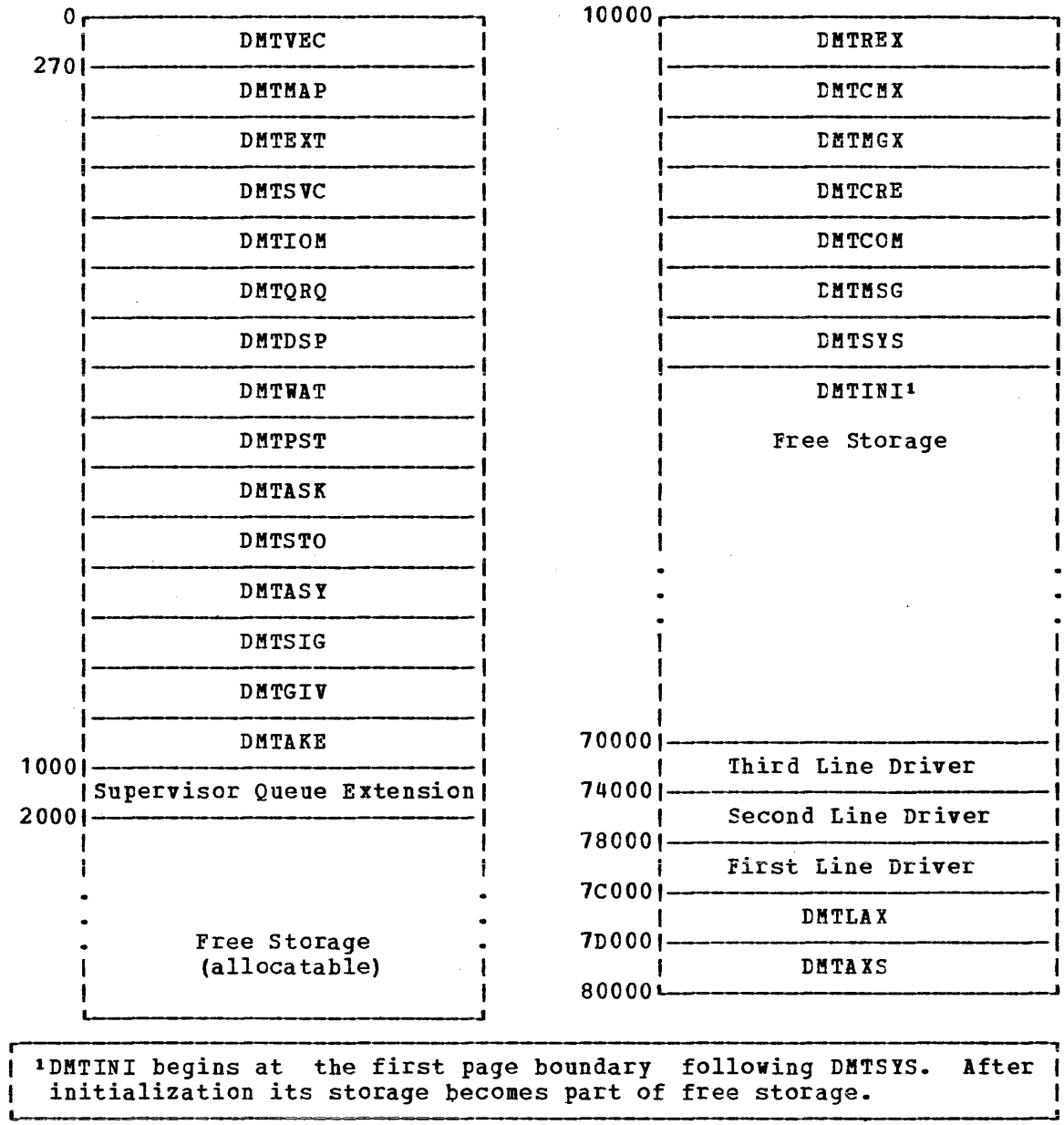


Figure 7. RSCS Storage Allocation

Synchronizing and Dispatching Tasks

The means by which RSCS synchronizes and dispatches tasks are the WAIT/POST routines (DMTWAT and DMTPST), synchronization locks, asynchronous requests and exits, and the dispatcher routine (DMTDSP).

The WAIT/POST method of task synchronization (Supervisor modules DMTWAT and DMTPST) is used when an executing task requires the services of another task. When this situation occurs, the requesting task must suspend its execution while it waits for the requested service to be performed. In conjunction with the dispatcher, WAIT/POST allows tasks to temporarily suspend execution until they receive a signal (via the synch lock) that they can resume execution.

THE WAIT/POST ROUTINES

To suspend its execution, the requesting task calls DMTWAT, which inspects the synchronization locks RSCS uses to synchronize task execution. Completion of a service is signaled by means of a synch lock, which is set (or "posted") by DMTPST.

SYNCHRONIZATION LOCKS

Synchronization locks (or "synch locks") are fullwords contained in task save areas or control tables (such as TAREA or IOTABLE). Synch locks are also found in control areas in function selector routines such as REXCYCLE in module DMTREX.

The synch lock must be set to zero before the request for services is made. Setting the synch lock to zero prepares it for processing by the WAIT routine.

The first byte of the fullword may contain either a zero or a "post code." If the first byte is zero, the task is nondispatchable, because the requested service has not yet been performed. A post code is a code which sets to one any bit in the first byte of the synch lock. DMTPST sets such a bit to specify that a requested service has been completed.

The requesting task, that is, the caller of DMTWAT, may specify the address of a single synch lock (as in the case of a GIVE Table or an IOTABLE) or the address of a list of synch locks (as in the case of REXCYCLE), one of which must be posted by DMTPST before dispatching of the requesting task can resume. Figure 8 shows the contents of Register 1 on a call.

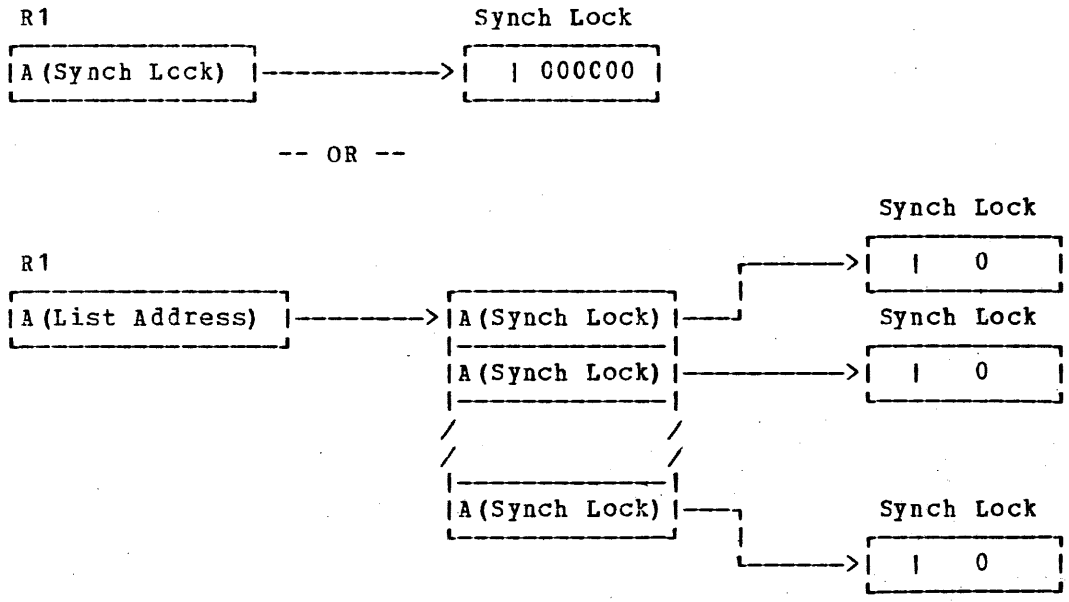


Figure 8. Input to the DMTWAT Routine

ASYNCHRONOUS INTERRUPTIONS AND EXITS

Asynchronous interruptions result from processes external to RSCS. For example, during REX task execution, the RSCS operator may press the ATTN key on the RSCS console, thereby asynchronously interrupting execution of the REX task.

To handle asychronous interruptions, RSCS tasks contain asynchronous exit routines. These asychronous exit routines are set up during initialization without dispatching the task being requested to perform the requested service. Asynchronous exits are provided for external interruptions, for certain I/O interruptions, and for ALERT requests that occur during execution of another task.

Asynchronous exits are taken after a task calls DMTASY specifying the requested exit conditions and the entry address of the asynchronous exit routine.

DMTASY also handles external interruptions reqeusted for the clock comparator. The request element is queued on the asynchronous exit queue and processed by DMTEXT. The DMTASY clock comparator provides a time delay mechanism by using the CPU hardware clock comparator.

Asynchronous exit routines perform limited function, often enqueueing requests for further processing at a later time by dispatched tasks. When the asynchronous exit routine completes processing, it returns control to the Supervisor, which then resumes dispatching tasks via a call to the dispatcher (DMTDSP).

USING ASYNCHRONOUSLY REQUESTED SERVICES: DMTWAT

Before a task can use the results of an asynchronously requested service, it must ensure that the service has been performed. To ensure that the service has been performed, the calling task signals that it is waiting for completion of a service via a call to the supervisor routine DMTWAT, specifying the synch lock associated with the requested service.

If the high-order byte of the task's synch lock is nonzero when DMTWAT inspects it, control is returned directly to the calling task. If the high-crder byte of the synch lock is zero, DMTWAT marks the calling task nondispatchable (via the task's request element), stores the address of the task's request element in the low-order bytes of the synch lock, and resumes dispatching for other tasks.

POSTING A SYNCHRONOUS LOCK

When the requested service is complete the REX Task signals completion by calling the POST routine (DMTPST), specifying the requesting task's associated synchronization lock. The POST routine sets the high-order byte of the synch lock to nonzero. This is referrred to as "posting" that synch lock, and indicates that the requested service is complete.

The supervisor functions return control to the tasks by means of the dispatcher (DMTDSP). The dispatcher scans the queue of tasks to be executed (TASKE in SVECTORS), selects the first dispatchable task element (that is, one that is not marked nondispatchable by DMTWAT), moves this task element to the end of the task queue, and restarts its execution. If no task element is marked "nondispatchable," a masked-on wait state PSW is loaded by the dispatcher.

In addition to posting a synch lock, DMTPST inspects the synch lock to determine whether DMTWAT has stored the address of a task element in that synch lock, implying that the task is nondispatchable. If this is the case, DMTPST marks the task's task element dispatchable and clears the last three bytes of the synch lock to zero.

Tasks may call DMTWAT specifying multiple synch locks. When this is the case, each synch lock is inspected and, if any synch lock is posted, task execution resumes immediately. If no synch locks are posted, the task element for the calling task is marked nondispatchable, its address is stored in each of the synchronization locks, and dispatching is resumed for other tasks.

When any synch lock in the list is posted, the task element is marked dispatchable. The dispatcher clears the low-order three bytes of each of the task's synchronization locks (pointed to in the task element before task execution is resumed).

Task-to-Task Communications

There are situations when a task requires the services of another task in order to complete a function. For example, SML may require that AXS open a file for input before processing of that file can continue. RSCS task communicate with each other to request these kinds of services using two methods: ALERT task-to-task communication and GIVE/TAKE communication.

Both methods use an element, which is a table of information that describes the nature of the request. In general, these elements are referred to as request elements and ALERT elements.

ALERT TASK-TO-TASK COMMUNICATION

The ALERT method of task-to-task communication allows a task to interrupt another task to request an immediate service. The type of request is described by an ALERT element, the address of which is specified by the requesting task in a call to DMTASY.

The supervisor responds by giving control to the asynchronous exit routine defined by the request task and by passing to that task the address of the ALERT element that describes the requested service.

The requested task's (that is, the task receiving the request) asynchronous exit routine responds immediately and may copy the ALERT element into its own storage for further processing. The receiving task's asynchronous exit routine then returns control to the supervisor, which allows the dispatched task to resume execution.

The ALERT routine (DMTSIG) also notifies another task that an asynchronous event has taken place. In this case, DMTSIG is not used with an ALERT request element.

GIVE/TAKE TASK-TO-TASK COMMUNICATION

While the ALERT method of task-to-task communication demands immediate response from the alerted task, the GIVE/TAKE method provides a means for ordered enqueueing of requests for services. These requests are handled when the servicing task is free to handle it, rather than upon immediate demand.

Request and Response Elements

Generally, request and response elements are formatted tables of information that reside in the storage of both the requesting task and the task providing the service. During task-to-task communication, these elements are passed from one task to another, containing either requests for services or responses to requests.

GIVE Tables

When a task requests services of another task via GIVE/TAKE, it builds a GIVE table in its storage. The GIVE request buffer and a GIVE response buffer. (The request and response buffers may be at the same location in storage.)

The GIVE request buffer contains a GIVE request element, which is a table of information describing the service being requested. Once the GIVE request element is built, the requesting task clears the synch lock in its address of the GIVE table to zero (in preparation for a call to DMTWAT) and specifies the address of the GIVE table in a call to DMTGIV.

Supervisor Handling of GIVE Requests

The supervisor then enqueues a supervisor GIVE element containing a pointer to the GIVE table, so that the request can be forwarded to the receiving task when that task is ready to accept the request.

Taking a GIVE Request

When the receiving task signals that it can process a GIVE request, the receiving task builds a TAKE table in its own storage. The TAKE table consists of a field to receive the task name of the requesting task and the addresses and the lengths of a TAKE request buffer and a TAKE response buffer. Functionally, these buffers complement the GIVE request and response buffers and, like the GIVE buffers, may be at the same location in storage.

Once the TAKE table is built, the receiving task specifies the address of the TAKE table in a call to DMTAKE. The supervisor then moves the GIVE request buffer (containing the GIVE request element) to the receiving task's TAKE request buffer.

Responding to a GIVE Request: DMTAKE Processing

The receiving task performs the requested service and updates the GIVE request element and places it in its TAKE response buffer. This modified GIVE request element contains information on results of request processing to be returned to the requesting task.

When all request processing is complete, the receiving task again calls DMTAKE, specifying the address of the TAKE table. The supervisor responds by immediately moving the contents of the receiving task's TAKE reponse buffer to the requesting task's GIVE response buffer, and posting the synch lock in the requesting task's GIVE table.

Multiple GIVE Requests for the Same Task

If another GIVE request addressed to the receiving task has been enqueued, it is given to the receiving task as described above, and dispatched task execution is resumed. On each call to it, DMTAKE first responds to a previously accepted GIVE request (if one exists) and then gives another modified GIVE request element back to the calling task (if one exists).

Waiting for Request Completion

The requesting task waits for request completion by specifying the address of the synch lock in its GIVE table in a call to the WAIT routine (DMTWAT).

The receiving task waits for request availability by calling DMTWAT and specifying the address of its "task request synch lock," which is located in its Task Save Area. The task request synch lock is cleared to zero by DMTAKE when no GIVE request address to the calling task remains enqueued. It is posted by DMTGIV when such a request is enqueued as a result of DMTGIV processing for another task.

Figure 9 shows the movement of data during a GIVE/TAKE transaction.

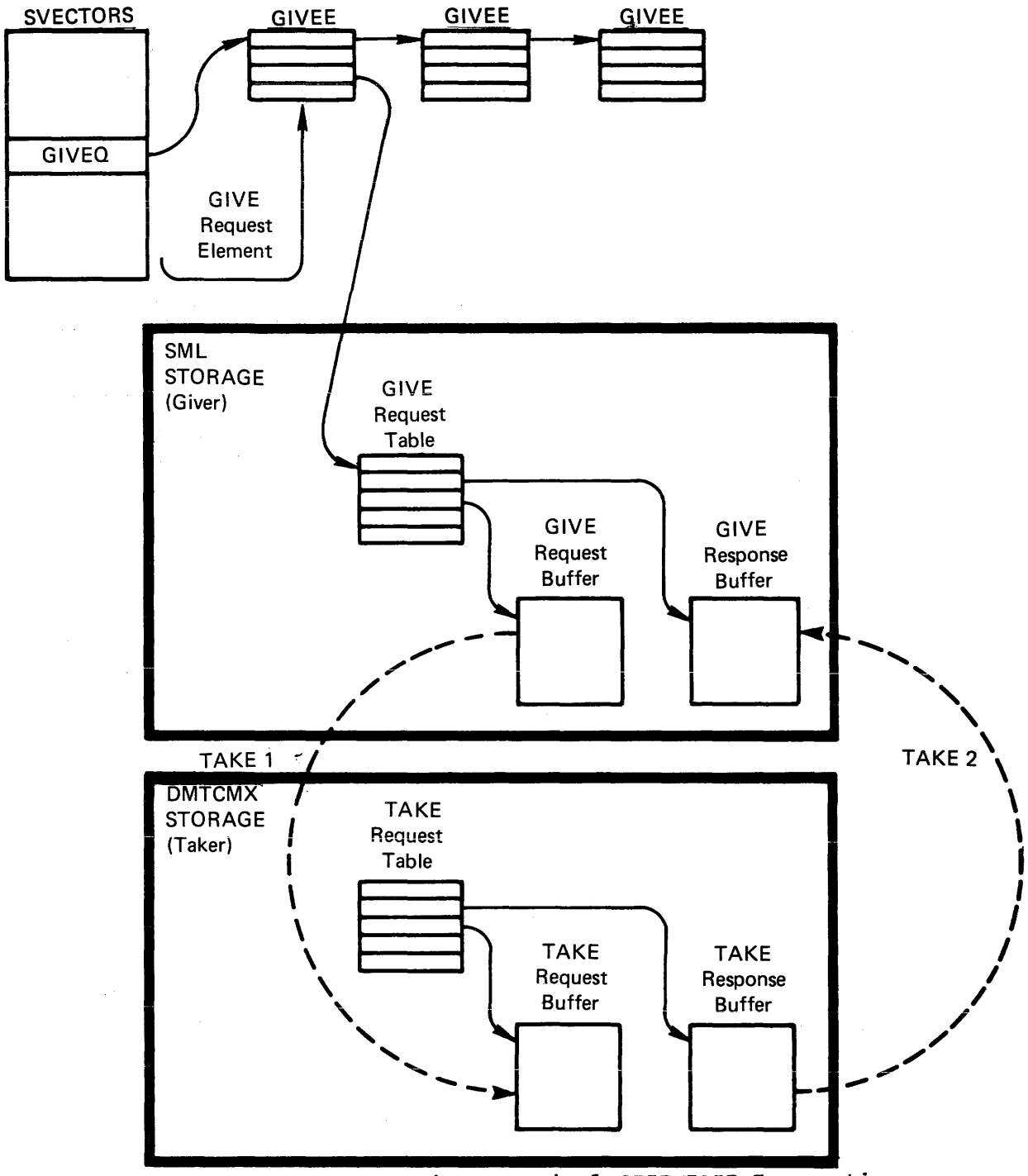


Figure 9. Movement of Data During a Typical GIVE/TAKE Transaction

Input/Output Methods and Techniques

Two data structures are created when RSCS performs an I/O operation: an I/O element and an I/O table.

The I/O table (defined by DSECT IOTABLE) is built by the requesting task and describes specific information required to perform the requested I/O operation.

The I/O element (defined by DSECT IOE) is built by the I/O request manager (DMTIOM) and consists of items of system information describing a request for an I/O operation.

I/O elements are placed on queues pointed to in SVECTORS: MPXIOQ (for multiplexer I/O requests) and SELIOQ (for Selector I/O requests). The elements in these two queues are in ascending subchannel order. Queue elements may also contain pointers to subqueues, which represent requests for use of the same nonshared subchannel. Each I/O element points to an I/O table.

Also, there is a queue of I/O asynchronous exit request elements pointed to in the SVECTORS data area. Figure 10 shows the relationships between these various data areas.

ACTIVE AND PENDING I/O QUEUES

The supervisor I/O queues (MPXIOQ and SELIOQ) include an active queue and a number of inactive or "pending" subqueues. Each element in the active I/O queue represents an I/O operation which is active on a particular nonshared I/O subchannel. The active I/O queue is ordered according to ascending numerical I/O subchannel address.

When an I/O operation is requested on an idle I/O subchannel, an I/O element representing the request is built and enqueued on the active I/O queue in its I/O subchannel's numerical address position. The I/O operation is then started.

When an I/O operation is requested on an I/O subchannel for which an I/O element is enqueued on the active I/O queue, the nonshared subchannel is busy and, therefore cannot be started immediately. In this case, an I/O element representing the request is built and enqueued on the subchannel's inactive I/O subqueue. The head of this subqueue is contained in the active I/O element enqueued on the active I/O queue.

When the nonshared subchannel's active I/O completes and the subchannel becomes available, the first element on the inactive I/O subqueue is enqueued on the active I/O queue and its I/O operation is started.

HANDLING LINK ACTIVITY: LINKTABLS AND TAGS

When the RSCS system is generated, a number of TAG slots are generated and enqueued on the free TAG queue. TAG slots are storage areas defined by the TAG DSECT; TAG slots describe the files being transmitted via RSCS; the free TAG queue comprises those TAG slots available for a given RSCS system.

The Free TAG Queue is defined in the DSECT TAGAREA, which also defines the status of TAG slots in the RSCS system. TAGAREA is pointed to by TTAGQ in SVECTORS.

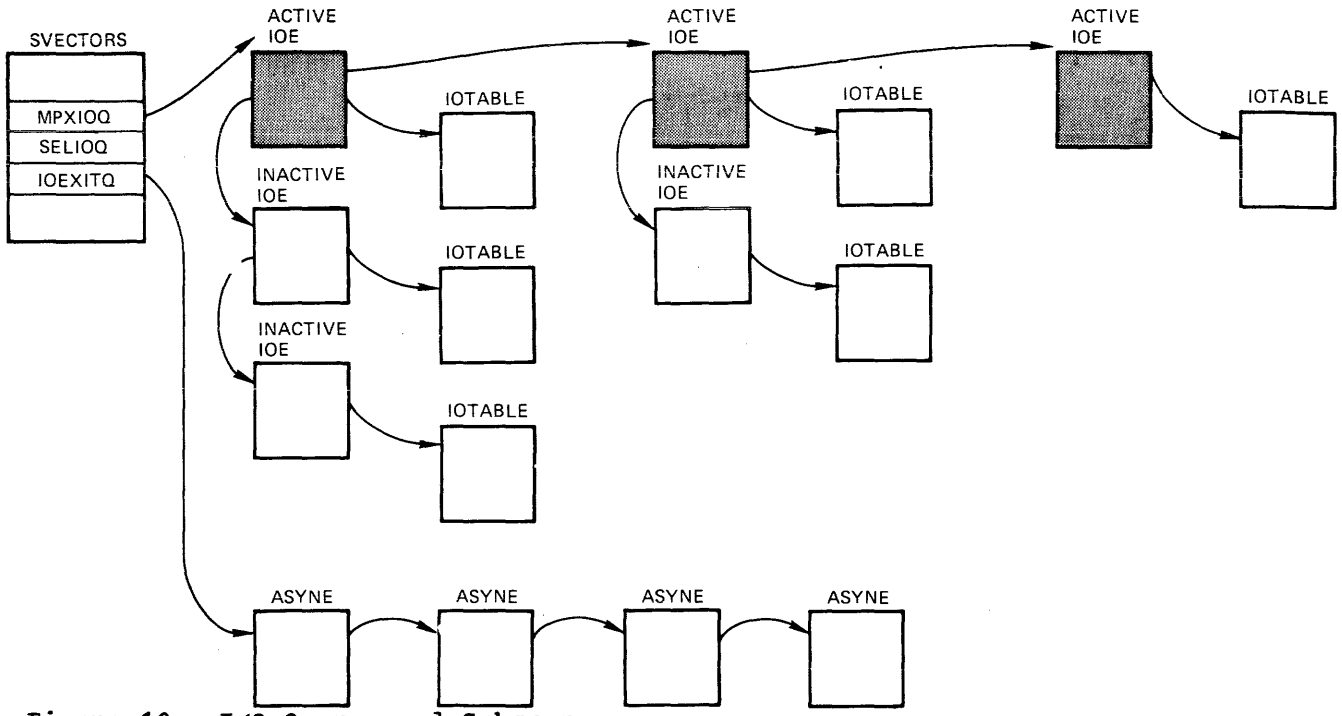


Figure 10. I/O Queues and Subqueues

How Links Handle Files

Each link in RSCS is defined by a LINKTABL DSECT. The LPOINTER field of the LINKTABL DSECT points to the link's inactive TAG queue. This queue comprises those TAGs describing files that RSCS has not yet transmitted. Only one TAG per link can be active at a time.

The queue of LINKTABLs (called the link table) is pointed to by the TLINKS field in SVECTORS.

Transmitting VM/370 Files to an RSCS Link

When a VM/370 file is spooled to RSCS for a specific link, RSCS accepts the file and:

- Obtains a free TAG slot for the file.
- Builds a description of the file in the TAG slot.
- Enqueues the new TAG on the link's inactive TAG queue.

When transmission to the remote station begins, the file's TAG is dequeued from the inactive TAG queue and enqueued on the active input file queue (TAGACIN in TAGAREA). When transmission of the file is complete, the TAG is dequeued from the active input queue and its slot is returned to the Free TAG Queue.

As in the case of VM/370 spool files, when files are received from remote stations, RSCS obtains a TAG slot and builds a description of the file in that slot. However, files from remote stations are enqueued on the active output queue (TAGACOUT in TAGAREA).

When the file is completely transmitted, its TAG is dequeued from the active output queue, closed to the VM/370 spool system, and its freed slot returned to the free TAG queue.

Figure 11 shows the relationships between the DSECTs described above.

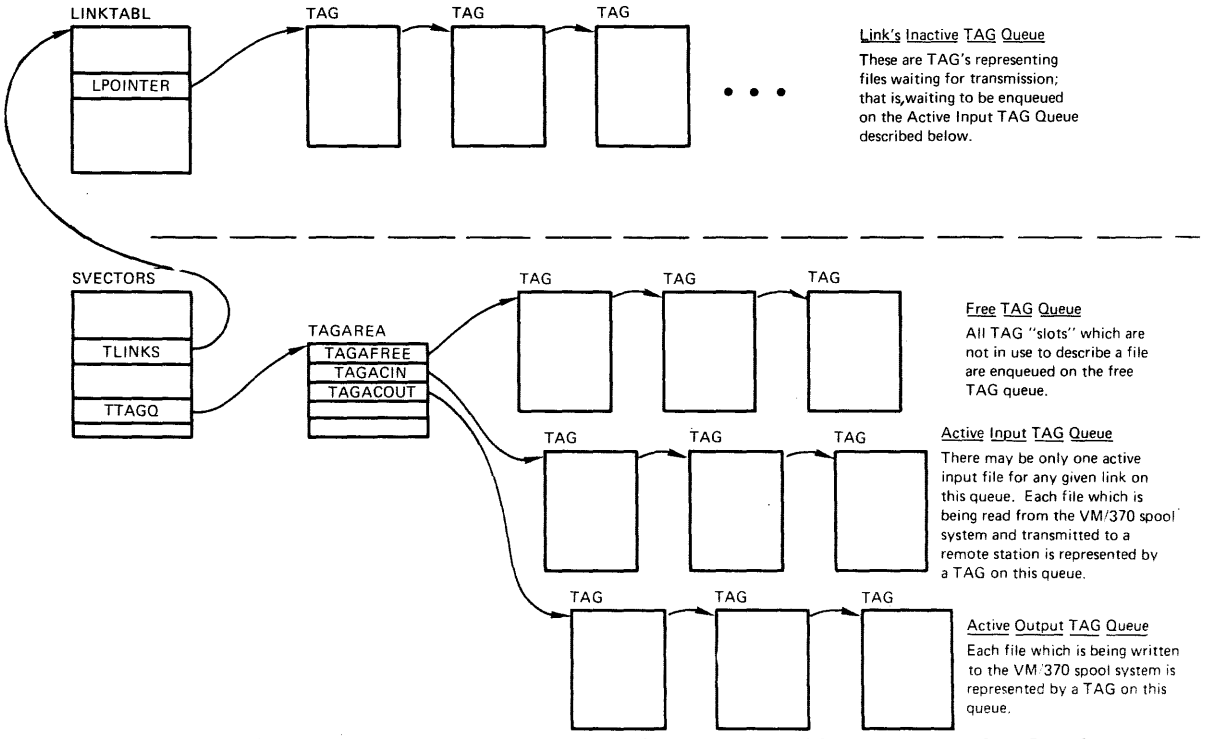


Figure 11. Chaining of Data Areas Required for File TAG Manipulation

RSCS Method of Operation and Program Organization

This section contains the following figures:

- Figure 12 through 17 show how the RSCS routines interact with each other functionally.
- FIgure 12 shows all of the RSCS components at an overview level.
- Figures 13 through show the parts of the individual components.

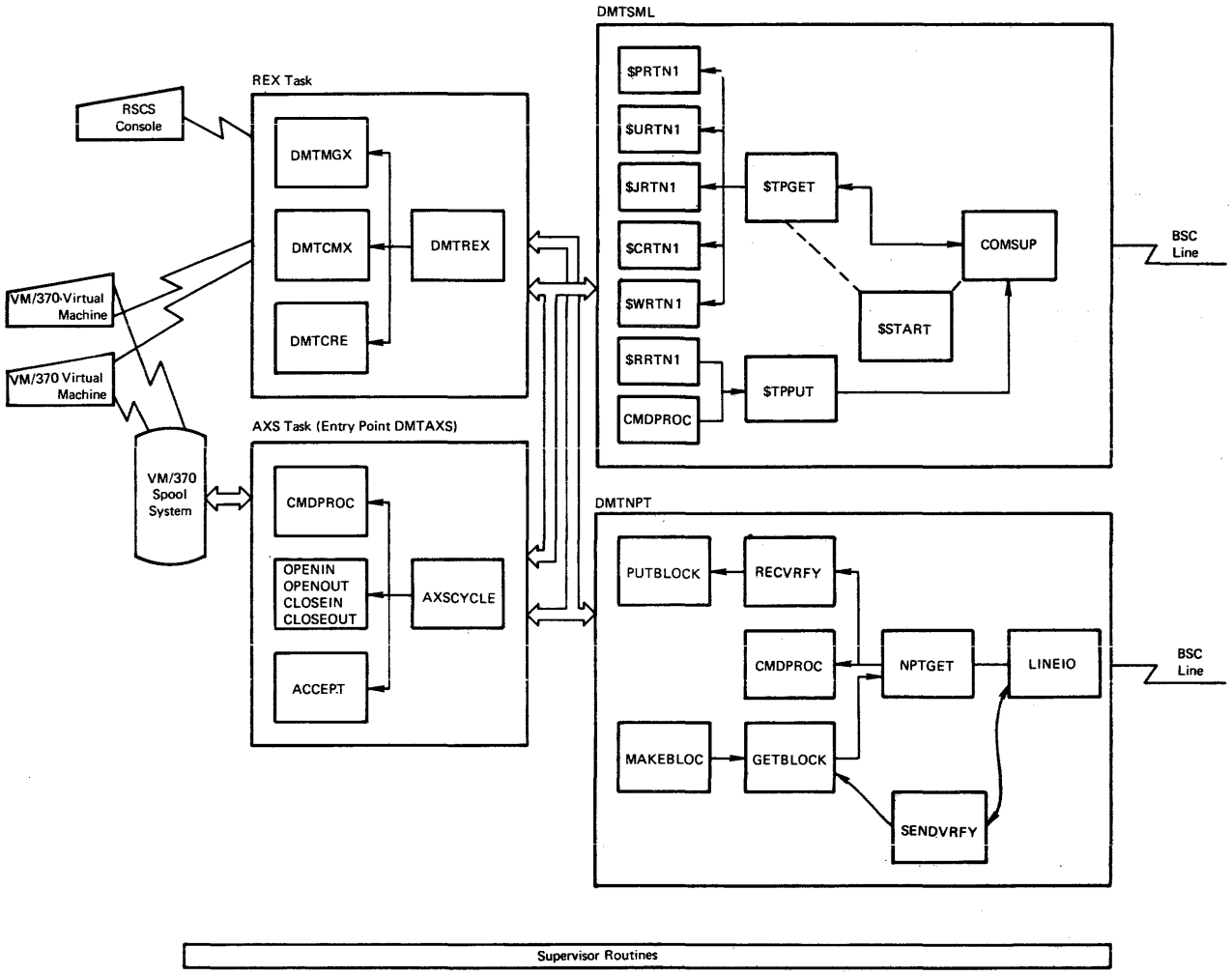


Figure 12. Overview of RSCS Program Organization

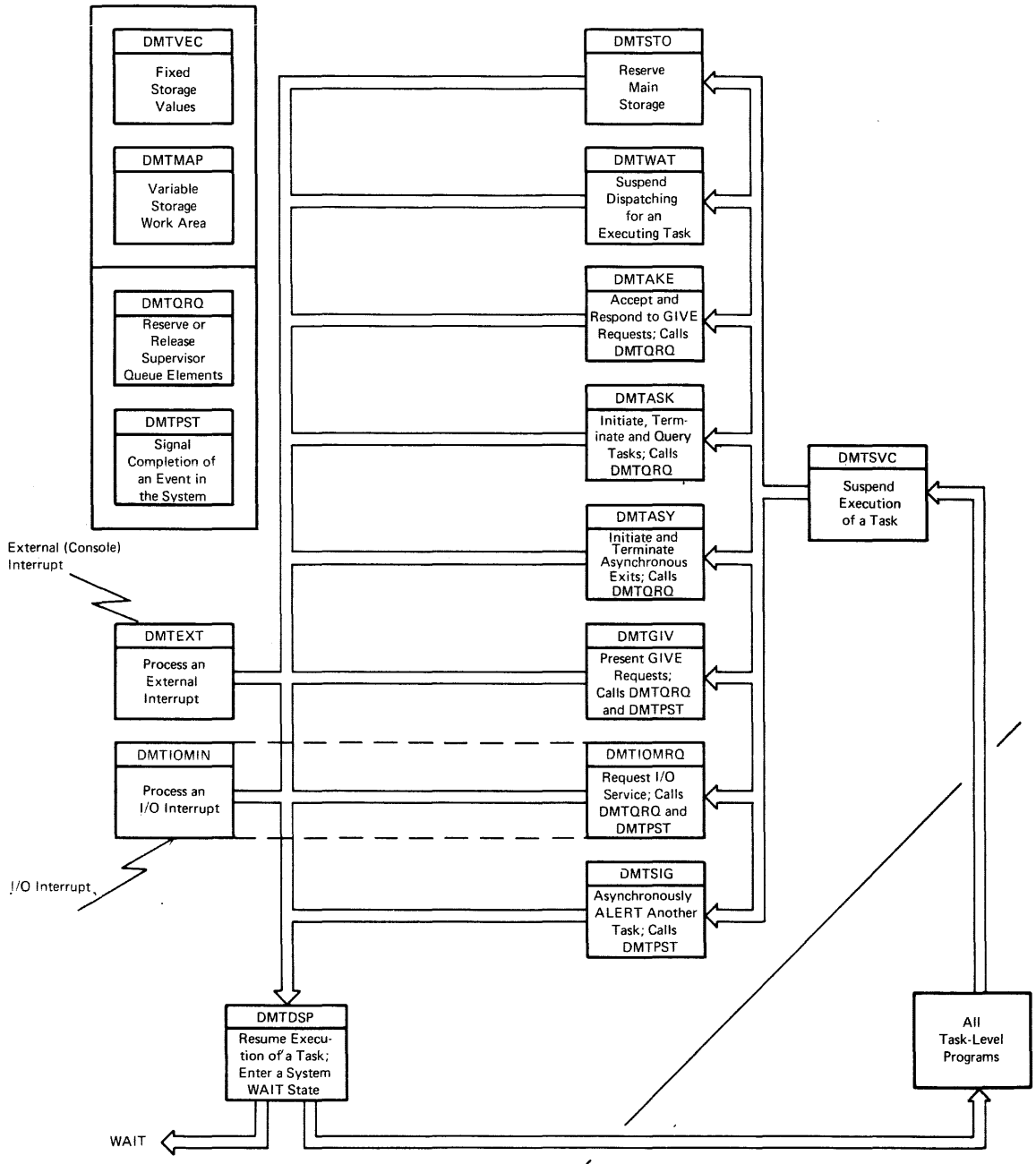


Figure 13. Program Organization for the Multitasking Supervisor

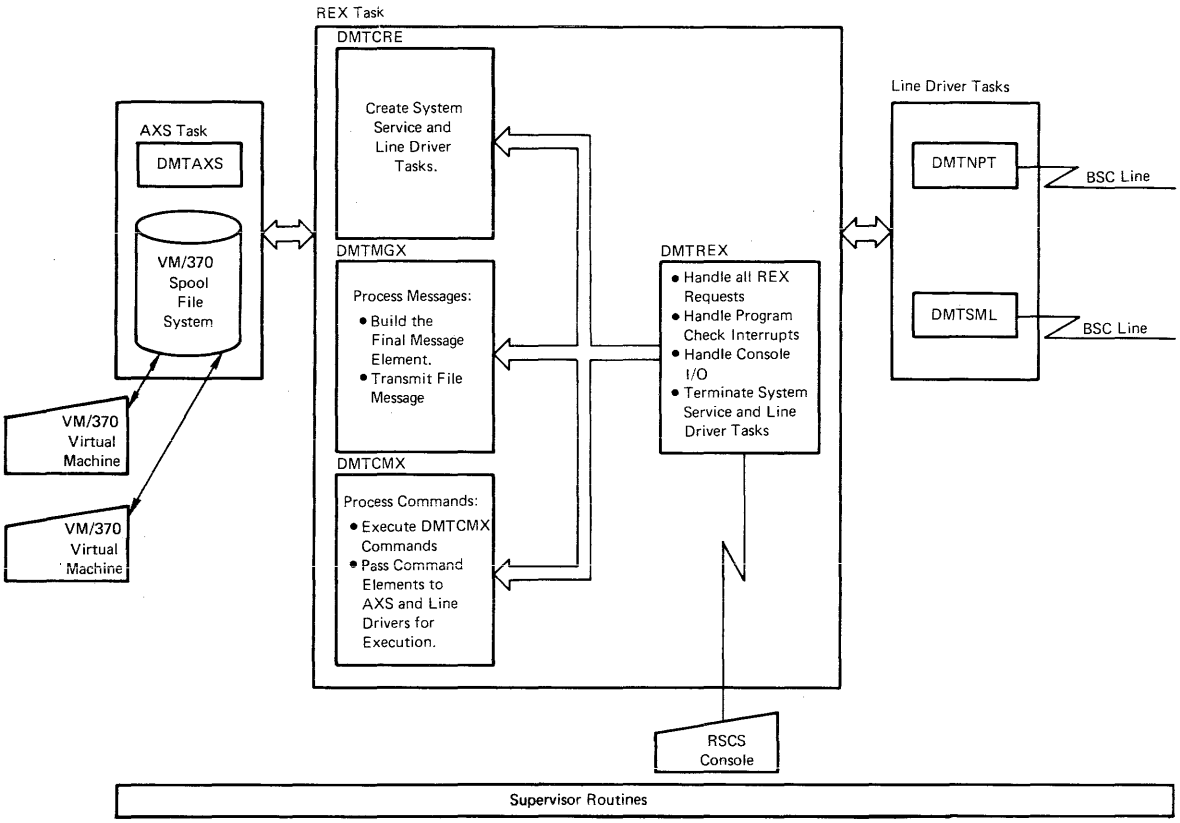


Figure 14. Program Organization for REX System Service Tasks

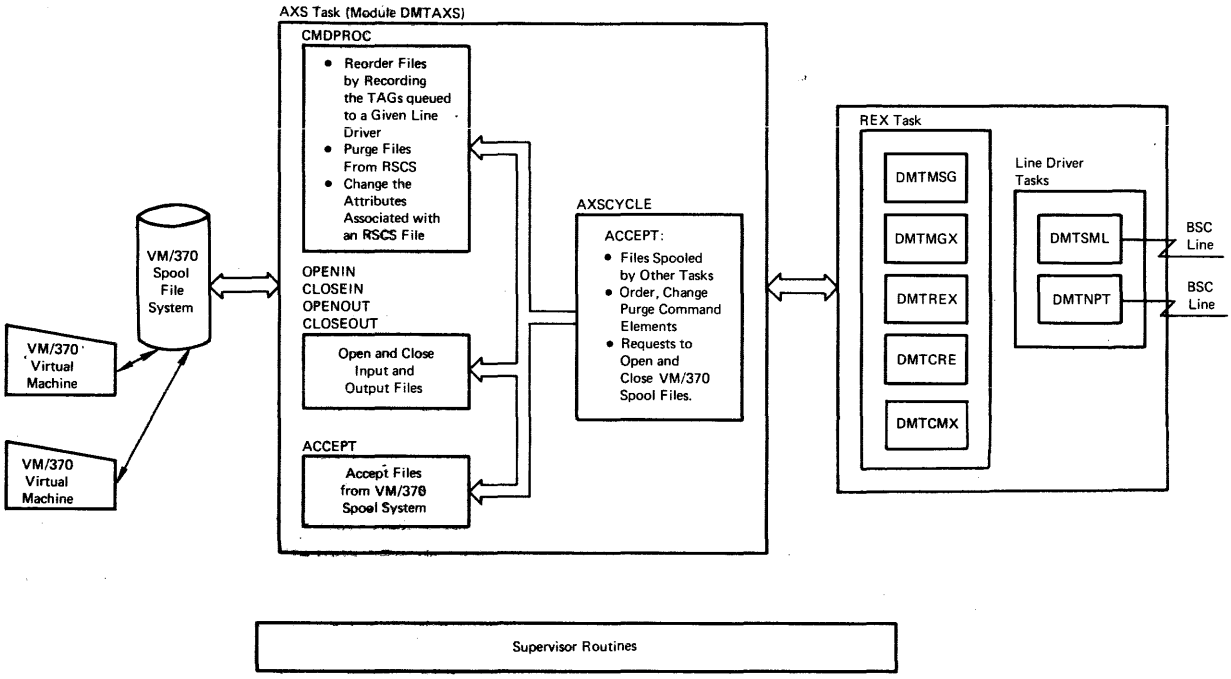


Figure 15. Program Organization for the AXS System Service Task

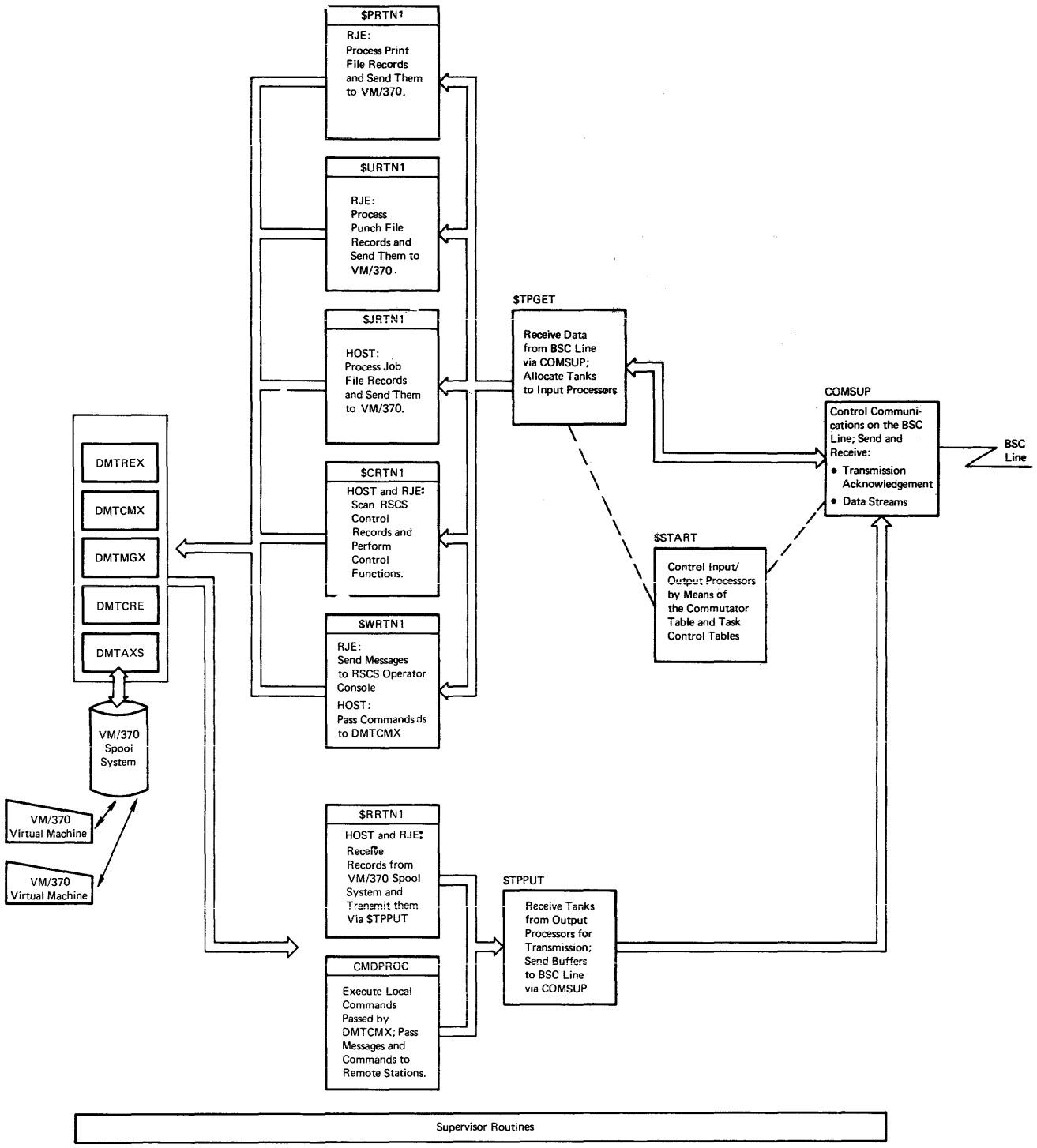


Figure 16. Program Organization for the SNL Line Driver Task

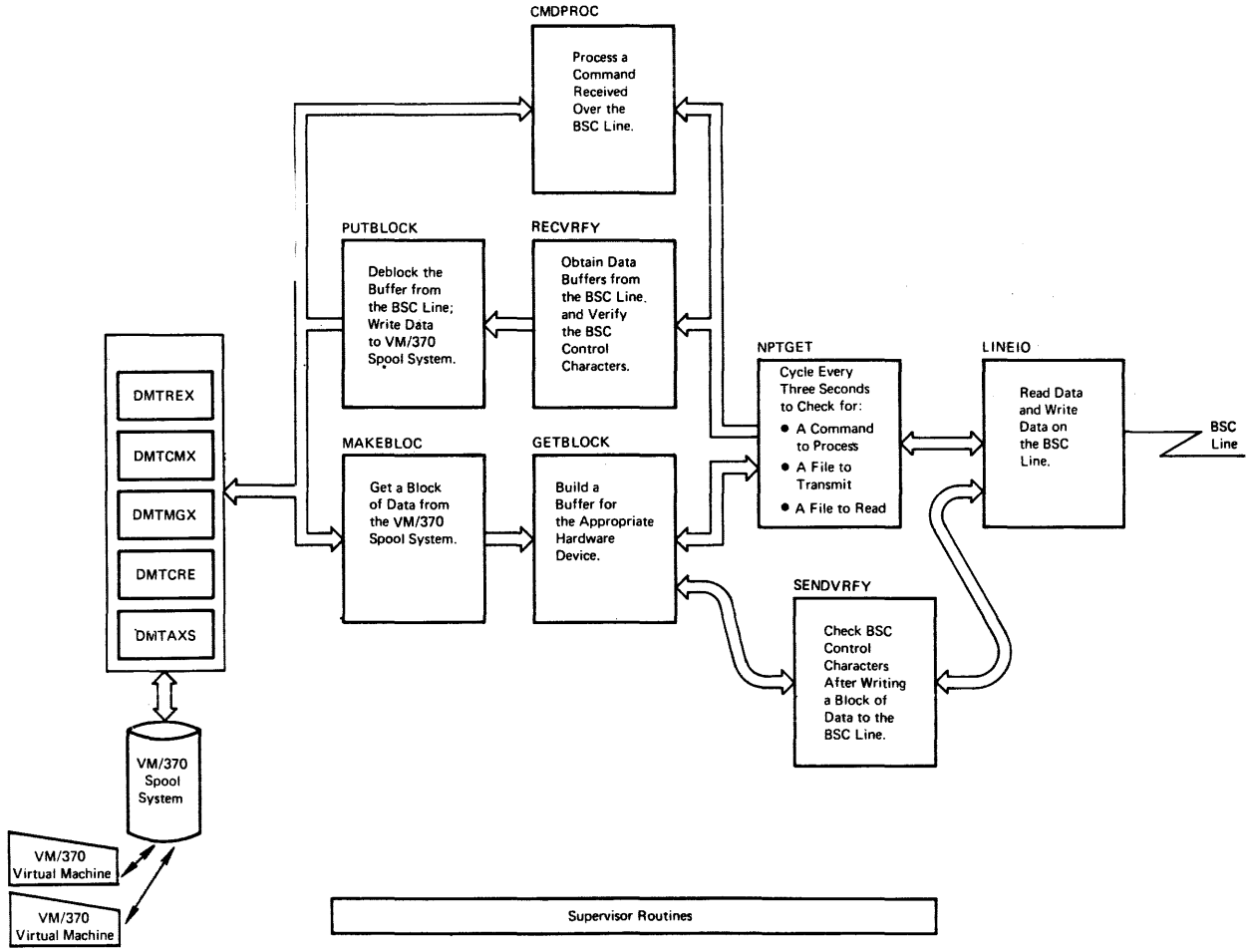


Figure 17. Program Organization for the NPT Line Driver Task

RSCS Directories

The following directories are contained in this section:

- RSCS Module Directory
- RSCS Module Entry Point Directory
- RSCS Module-to-Label Cross Reference
- RSCS Label-to-Module Cross Reference

RSCS Module Directory

RSCS Module	BALR to Module	At Label	Comments
DMTAKE	DMTDSP	TAKEXIT	Resumes dispatching; processing of a TAKE request is complete.
	DMTPST	TAKEMUTE	Signals a task that it must process a TAKE request.
	DMTQRQ	TAKEMUTE	Frees a GIVE element.
DMTASK	DMTDSP	TAEXIT	Resumes dispatching; processing of a task request has completed.
	DMTPST	TAGPURGE	Signals the termination of a task.
	DMTQRQ	TAFREEOK	Frees a terminated task element.
	DMTQRQ	TAGPURGE	Frees a terminated GIVE element.
	DMTQRQ	TAMAKE	Gets a queue element for a new task.
	DMTQRQ	TAQPTEST	Frees requested elements for a terminated task.
	DMTQRQ	TASQTEST	Frees an I/O element associated with a task being purged.
DMTASY	DMTDSP	ASEXIT	Resumes dispatching; processing of an asynchronous exit request has completed.
	DMTQRQ	ASQEND	Gets a free queue element; free a terminated queue element.
	DMTQRQ	ASQGOT	Gets a free queue element; free a terminated queue element.
DMTAXS	DMTAKE	AXSACCPT	Takes a request for DMTAXS services from another task.
	DMTASY	AXSIGSET	Requests an asynchrcnous exit for task asynchronous alerts.
	DMTASY	AXSIGSET	Requests an asynchrcnous exit for reader X'001'.
	DMTCOM	GETLINK	Gets a link table entry.
	DMTCOM	OPENIRTY	Gets a page of main storage.
	DMTCOM	OPENOLNK	Gets a page of main storage.
	DMTCOM	TODEBCD	Converts a S/370 format TOD to EBCDIC date and time.
	DMTGIV	MSGDO	Gives a message element to DMTMGX for processing.
	DMTPST	AXSALRT1	Signals acceptance of a command to process.
	DMTPST	AXSASYIO	Signals arrival of a request for an asynchronous exit.
DMTAXS	DMTSIG	ACCEFIND	Alerts a line driver task that a newly arrived file has been accepted.
	DMTSIG	CHANDONE	Alerts a line driver task.
	DMTWAT	AXSCYCLE	Waits for a request for DMTAXS services.
	DMTWAT	MSGDO	Waits until processing by DMTGIV has completed.
DMTCMX	DMTCOM	QYOLINK	Finds a link table entry.
	DMTCOM	TODEBCD	Converts a S/370 TOD to EBCDIC date and time.
	DMTCRE	STALNGOT	Creates a line driver task, as specified in the START command.
	DMTMGX	CMXDOIT	Writes a message resulting from command processing.
	DMTMGX	CMXM001	Writes a message showing the number of free pages in storage.
	DMTMGX	CMXM003B	Writes a message showing the command currently being executed by RSCS.
	DMTMGX	DISCHARG	Writes a message resulting from DISCONN command processing.

RSCS Module	BALR to Module	At Label	Comments
DMTCMX (cont)	DMTMGX	QYM654	Writes a message resulting from QUERY command processing.
	DMTMGX	QYM655	Writes a message resulting from QUERY command processing.
	DMTMGX	QYSYMSG	Writes a message resulting from command processing.
	DMTREX	DISCONN	DIAGNOSE instruction entry to CP console function.
	DMTREX	DISCHARG	DIAGNOSE instruction entry to CP console function.
	DMTSIG	CMXALRDY	Alerts a task for command processing.
	DMTSIG	STACREAT	Alerts DMTLAX to validate a line address used in a START command.
DMTCOM	DMTDSP	MFIXIT	Requests dispatching of a task for which a message has been stacked for transmission.
	DMTDSP	MFOXIT	Requests dispatching of a task for which a message has been unstacked for transmission.
	DMTSTO	GETPTRY	Requests main storage.
DMTCRE	DMTASK	CREQTASK	Requests the supervisor to start a new task.
	DMTIOM	CFILDOIO	Requests the I/O manager to read one DASD block from a file on a CMS-type system disk.
	DMTSTO	CRETRYIT	Requests main storage for the creation of a task.
	DMTWAT	CFILDOIO	Waits for a read I/C request to complete.
DMTEXT	DMTDSP	EXTGO	Resumes dispatching; processing of an external interruption is complete.
DMTGIV	DMTDSP	GIVEXIT	Resumes dispatching; processing of a GIVE request is complete.
	DMTPST	GIVESNIF	Signals a task tc begin processing a GIVE request.
	DMTQRQ	GIVESCAN	Gets a free queue element.
DMTINI	DMTDSP	INIQDONE	Dispatches the first task.
	DMTQRQ	INIQDONE	Initializes the queue of free elements.
DMTIOM	DMTDSP	IODISPCH	Resumes dispatching; processing of an I/O request is complete.
	DMTPST	IONORMAL	Signals completion of an I/O event.
	DMTPST	IOPUNT	Signals an error on a request for a queue element.
	DMTQRQ	DMTIOMRQ	Gets an element for an I/O request.
	DMTQRQ	IODISMIS	Frees an element used for a SENSE request.
	DMTQRQ	IONORMAL	Frees an element used in an I/O request.
	DMTQRQ	IOUNITCK	Gets an element for a SENSE request.
DMTLAX	DMTASY	LAXINIT	Sets up an asynchronous exit for DMTLAX.
	DMTWAT	LAXHANG	Terminates DMTLAX.
DMTMGX	DMTCOM	MGXBUILT	Gets a link table entry.
	DMTCOM	MGXTOLOC	Stacks a message.
	DMTREX	MGXNOPR	Writes a message to a local VM/370 userid.
	DMTREX	MGXNOVM	Writes a message to the VM/370 operator.
	DMTSIG	MGXBUILT	Alerts an originating task that a message has been handled.

RSCS Module	BALR to Module	At Label	Comments
DMTNPT	DMTASY	NPTNOPAS	Sets up an asynchronous interrupt for DMTNPT.
	DMTCOM	AXSMENQ	Enqueues a message on the message stack for processing by DMTMGX.
	DMTCOM	MSG2780	Unstacks a message for transmission to a remote station.
	DMTCOM	NPTNOPAS	Gets a page of storage for use as DMTNPT buffers.
	DMTCOM	TODEBCD	Converts S/370 TOD to EBCDIC date and time.
	DMTGIV	AXSGET	Requests DMTAXS to open a file.
	DMTGIV	AXSPURGE	Requests DMTAXS to purge a file.
	DMTGIV	COMMANDS	Passes a command element to DMTREX for processing by DMTCMX.
	DMTGIV	KLOGIT	Requests DMTAXS to open the LOG file for output.
	DMTGIV	LINEDROP	Requests DMTAXS to close a file.
	DMTGIV	LOGCLOSE	Requests DMTAXS to close the LOG file for output.
	DMTGIV	MSG1	Passes a message element to DMTMGX for processing.
	DMTGIV	PUTCLS1	Requests DMTAXS to close a file for output.
	DMTGIV	PUTOPEN	Requests DMTAXS to open a file for output.
	DMTGIV	TASKILL	Requests DMTREX to terminate the requesting NPT line driver.
	DMTIOM	LOGCONT1	Requests an I/O operation for the LOG routine.
	DMTIOM	LOGPRINT	Prints a LOG message.
	DMTIOM	XECUTE	Requests an I/O operation (general usage by DMTNPT).
	DMTPST	AXSALRT1	Signals that DMTNPT accepted a command.
	DMTWAT	AXSGET	Waits for a request to open a file to complete processing.
	DMTWAT	AXSPURGE	Waits for a request to purge a file to complete processing.
	DMTWAT	COMMANDS	Waits for DMTCMX to process a command.
	DMTWAT	KLOGIT	Waits for completion of a request to open the LOG file for processing.
	DMTWAT	LINEDROP	Waits for a request to close a file to complete processing.
	DMTWAT	LOGCLOSE	Waits for a request to close the LOG file when processing is complete.
	DMTWAT	LOGCONT1	Waits for an I/O operation to complete logging processing.
	DMTWAT	MSG1	Waits for message processing to complete.
	DMTWAT	PUTCLS1	Waits for a request to close a file to complete processing.
	DMTWAT	PUTOPEN	Waits for completion of a request to open a file for processing.
	DMTWAT	TASKILL	Waits for task termination processing to complete.
	DMTWAT	XECQWAIT	Waits for an I/O operation to complete.
DMTREX	DMTAKE	REXACCPT	Accepts a request to process a VM/370 file.
	DMTASK	QUIESE	Requests task termination.
	DMTASK	TERTKILL	Requests task termination.
	DMTASY	REXICGOT	Initializes an asynchronous exit.
	DMTCOM	REXFLUSH	Requests DMTMGX to write any queued messages.
	DMTCOM	REXOUTRY	Removes a message for the message stack and write it to the console.
	DMTCRE	REXICGOT	Creates the tasks DMTAXS and DMTLAX.

RSCS Module	BALR to Module	At Label	Comments
DMTREX	DMTDSP	REXDQUIT	Terminates dispatching due to program check.
(cont)	DMTDSP	REXHEXIT	Resumes dispatching after program check processing.
	DMTIOM	REXCONON	Requests an I/O operation (console write).
	DMTIOM	REXFCONF	Requests an I/O operation (console write).
	DMTIOM	REXQUERY	Requests an I/O operation (console read).
	DMTMGX	MSG	Passes a message element to DMTMGX for processing.
	DMTMGX	TERMSET	Writes a task terminated message.
	DMTPST	REXASYN	Signals a console attention.
	DMTPST	REXHALT	Signals that DMTREX is undispatchable due to program check.
	DMTWAT	QUIESE	Waits for a task to terminate.
	DMTWAT	QUICK	Waits for task I/O to terminate.
	DMTWAT	REXSWAIT	Waits for a console write to complete.
	DMTWAT	REXWAIT	Waits for completion of an event.
DMTSIG	DMTDSP	ALSCAN	Resumes dispatching; processing of an alerted task has completed.
		ALNOGO	
DMTSML	DMTASY	SETNOBUF	Sets up an asynchronous exit for DMTSML.
	DMTCOM	ASYNENQ	Stacks a message to be transmitted by DMTSML.
	DMTCOM	BUFSDONE	Gets a page of storage for DMTSML I/O tasks.
	DMTCOM	IBLDBUFS	Gets a page of storage for DMTSML TP buffers.
DMTSML	DMTCOM	MSGPROC1	Unstacks a message for transmission to a remote station.
	DMTCOM	TODEBCD	Converts S/370 TOD to EBCDIC date and time.
	DMTGIV	AXS	Requests services of DMTAXS for the SML line driver task.
DMTGIV		KLOGIT	Requests DMTAXS to open a LOG printer.
DMTGIV		LOGCLOSE	Requests DMTAXS to close the LOG printer.
	DMTGIV	AXSGET	Requests DMTAXS to give a file for transmission.
	DMTGIV	AXSPURGE	Requests DMTAXS to purge a file.
	DMTGIV	EOJ	Requests termination of the SML line driver task.
	DMTGIV	MSG1	Gives a message to DMTMGX for processing.
	DMTGIV	WGET1A	Requests that a message be written to the RSCS console; pass a command to DMTREX.
	DMTIOM	I27XXIO	Performs the initial I/O operation for the SML line driver task.
	DMTIOM	JOUT1	Requests an I/O operation; set up job processing controls.
	DMTIOM	PCONT2	Requests an I/O operation (set up printer
		PLINE	controls.
	DMTIOM	RSIO	Requests a start I/O for the DMTSML TRACE function.
	DMTIOM	UOUT2	Requests an I/O operation (sets up punch controls).
	DMTIOM	WRLOG1	Requests an I/O operation (log an I/O operation.
	DMTPST	ASYNRET	Posts the reader synch lock.
	DMTWAT	ALLCHK	Waits for the DMTSML synch lock to be posted (waits for a request to process).
	DMTWAT	AXS	Waits for completion of an event by DMTAXS.
	DMTWAT	AXSGET	Waits for DMTAXS to GIVE a file for transmission.

RSCS Module	BALR to Module	At Label	Comments
DMTGIV (cont)	DMTWAT	AXSPURGE	Waits for DMTAXS to purge a file.
	DMTWAT	EOJ	Terminates the SML line driver task by issuing a terminal WAIT request.
		KLOGIT	Waits for DMTAXS to open a LOG printer.
		LOGCLOSE	Waits for DMTAXS to close a LOG printer.
	DMTWAT	MSG1	Waits until GIVE to DMTMGX is complete.
	DMTWAT	RISIO1	Waits for initial SIO for the DMTSML line driver to complete.
	DMTWAT	WGET1A	Waits until message processing has completed.
	DMTWAT	WRLOG1	Waits for I/O logging to complete.
DMTSTO	DMTDSP	MAINDONE	Resumes dispatching; a request for a page of storage has been processed.
DMTWAT	DMTDSP	WAITGO	Resumes dispatching; processing of a WAIT request has completed.

RSCS Module Entry Point Directory

Module Name	Entry Points	Function
DMTAKE	DMTAKE	Contains the supervisor service that supplies task programs with the receiver interface to GIVE requests issued by other tasks. A single CALL causes DMTAKE to first respond to the previously supplied GIVE request and then supply a new GIVE request to the task for its processing.
DMTASK	DMTASK	A service routine that creates new tasks and deletes existing tasks executed by the MSUP dispatcher. The entry to DMTASK is via a BAL instruction from task programming. Any entry into DMTASK causes the calling task's execution to be suspended through the freeze SVC function.
DMTASY	DMTASY	A supervisor service module that starts and ends asynchronous exit requests for task programs. This routine handles asynchronous exit requests for asynchronous exit requests for I/O interruptions, and ALERT exit requests.
DMTAXS	DMTAXS	Controls the interface of the line drivers to the VM/370 spool file system, enqueues files for transmission and processes commands that manipulate spool files.
	AXSINIT	Initializes the AXS task.
	AXSCYCLE	Looks for work to do by examining the synch locks associated with the AXS task.
	REQXEQ	Scans the request table for a match and branches to the to the appropriate subroutine, depending on the request code.
	CMDPROC	Executes AXS commands from the command buffer passed on by an ALERT exit from DMTREX.
	OPENIN	Starts spool file processing.
	CLOSEOUT	Ends processing for output files.
	MSG	Sets the MSG request element. A CALL GIVE instruction passes the MSG request element to the message manager. The code associated with other entry points in this module format the MSG element variable areas in various ways and exit finally to MSG.

Module Name	Entry Points	Function
DMTAXS (cont.)	HEXGET	Converts and validates a hex string.
	DECGET	Converts and validates a decimal string.
	DECPUT	Converts a hex fullword to decimal and generates an EBCDIC representation of it, suppresses leading zeroes to a minimum count, which is optionally supplied by the caller.
	TODS370	Converts EBCDIC to the System/370 TOD value.
	TODEBCD	Converts System/370 TOD to an EBCDIC date and time.
	GSUCCESS	Gets inactive successor spool file.
	ACCEPT	Inspects newly arrived files.
	UNPEND	Brings in a link's pending tags.
	GETROUTE	Gets a routing table entry.
	GETLINK	Gets link table entry.
	GETSLOT	Gets a free tag queue element.
	FREESLOT	Returns a tag queue element.
	TAGGEN	Builds a file tag from hypervisor information.
	TAGPLACE	Sets a file tag into a link queue immediately before the first tag of numerically higher priority (lower real priority).
	FILSELEC	Selects a file to be read from a link queue.
	TAGFIND	Locates a file with spoolid matching the one supplied by the caller, within the internal file tag queues.
	DEFINE	Gets a virtual spool device.
	DETACH	Undefines a virtual spool device.
	VCHANGE	Changes VM/370 file attributes.
	VCLOSE	Issues the VM/370 CLOSE command for a device.
	VPURGE	Purges an inactive reader file from the VM/370 spool.
	VSPOOL	Sets VM/370 virtual spool device options.
	VTAGD	Sets a VM/370 tag for a virtual spool device.
	VTAGF	Sets a VM/370 tag for an inactive spool file.
DMTCMX	DMTCMX	This module is part of the REX system control task. DMTCMX is called in several places in DMTREX, which is the main REX control routine. DMTCMX accepts an EBCDIC string and executes the RSCS command that the string represents.
	CMXHIT	Calls the necessary individual command proccessing routine.
	CMSALERT	Passes a command element to another task via the ALERT task-to-task communications interface.
	KEYWDGET	Decodes the next keyword on the input command line.

Module Name	Entry Points	Function
DMTCMX (cont.)	LTABGET	Finds the link table entry implied by the first keyword in the command line described by the calling routine's register parameters.
	HEXGET	Converts and validates a hex string.
	DECPUT	Converts a hex fullword to decimal and generates an EBCDIC representation of it. It suppresses leading zeros to a minimum count, which is optionally supplied by the calling routine.
	FILGET	Locates a file, within the internal file tag queues, with a spoolid matching that supplied by the calling routine.
	TODEBCD	Converts a System/370 format TOD to EBCDIC data and time.
	PARMGET	Scans an EBCDIC line and frames the next parameter on the line.
DMTCOM	DMTCOM	Contains various reentrant routines used by RSCS tasks.
	GETLINK	Scans the link table chain and returns a link table address.
	GETPAGE	Gets a free page of main storage.
	FREEPAGE	Returns a page of main storage.
	MFI	Stacks message elements in a LIFO stack for later processing. If no room is available in the current page, a new page is fetched if there are at least five free pages remaining. If five free pages are not remaining, an error condition is returned. All tasks except REX are allowed only three pages of storage to stack messages.
	MFO	Unstacks message elements from the message queue for this task. If none are queued an error condition is returned.
	GTODEBCD	Converts a System/370 format TOD to EBCDIC data and time.
DMTCRE	DMTCRE	Creates new tasks under MSUP.
	CMSFILCH	Reads one dASD block from a CMS disk.
	CMSOPEN	Does initial work prior to reading a CMS file.
	CMSGET	Gets the next CMS file item.

Module Name	Entry Points	Function
DMTDSP	DMTDSP	This module is the MSUP dispatcher. It is entered when an exit occurs from supervisor functions that were entered following an interruption or that issued the freeze SVC function. DMTDSP must be entered with all PSW masks off (except for the machine check mask).
DMTEXT	DMTEXT	This module is the MSUP external interruption handler. DMTEXT receives control directly on an external interrupt and saves the status of the executing task if one was interrupted.
DMTGIV	DMTGIV	This is a supervisor service routine that enqueues GIVE requests from tasks to be delivered to other tasks by DMTAKE.
DMTINI	DMTINI	Receives control after initial loading of RSCS, and performs general initialization functions that are common to all parts of RSCS. DMTINI writes a copy of the initial load to DASD, according to operator instructions, when RSCS is initial program loaded from the generation IPL deck. When ititial program loaded from disk, DMTINI finishes reading the saved RSCS load. When IPL disk reading or writing is comlete, DMTINI initializes RSCS storage areas.
DMTION	DMTION	This module contains both the MSUP I/O interrupt handler and the task I/O service routine. The I/O service provided by DMTIOM to the task programs includes sequential subchannel scheduling, channel program execution, automatic sense executionon unit check when requested, retrun of all pertinent information re-the execution of the channel program, and notification via a POST upon completion of the channel program.
DMTLAX	DMTLAX	This routine is the line allocation task for RSCS. The major part of this routine functions as an asynchronous exit being alerted by DMTREX.

Module Name	Entry Points	Function
DMTMAP	DMMAP	Describes the non-fixed address MSUP status storage areas in main storage. This module contains no executable code.
DMTMGX	DMTMGX	Takes a message request buffer and constructs the message from the information in that buffer and the message text found in DMTMSG.
DMTMSG	DMTMSG	Contains a list of error messages to be used externally by DMTMGX. This module contains no executable code.
DMTNPT	DMTNPT	This module is a line driver that provides support for the 2770, 2780, 3770, and 3780 nonprogrammable terminals.
	NPTGET	Maintains a cyclic control of the DMTPT task on both sending and receiving operations.
	SENDOFF	Sends the BSC end-of-transmission character (EOT) on the line to the remote terminal.
	BUFFINIT	Initializes the line output buffer with the correct BSC character set, depending on the type of output file and and features available at the terminal.
	XECUTE	Requests the supervisor to execute I/O operations. After starting the I/O operations, XECUTE waits for either a command to be entered or the completion of the requested I/O operation.
	LINEIO	Executes (by calling XECUTE) I/O operations on the BSC line and checks the final state. LINEIO then sets the IOERR flag in the DEVFLAG byte.
	GETBLOCK	prepares the line output buffer to be transmitted to the remote terminal.
	GETVRFY	Analyses the response obtained from each buffer transmission and takes the appropriate error action.
	PUTBLOCK	Deblocks received TP buffers and writes the deblocked record to the VM/370 spool file system.
	PUTVRFY	Verifies the content of each received TP buffer and constructs an appropriate reply if the buffer is found in error.

Module Name	Entry Points	Function
DMTNPT (cont.)	COMMANDS	Passes commands received from the remote card reader to the RSCS command processor for execution.
	CMDPROC	Executes commands passed to it in the CMDRESP buffer after an ALERT from DMTREX indicates a command has been entered.
	MSGPROC	Unstacks messages from the task MSG queue and transmits them to the remote terminal printer. Prepares and sends requests to the specialized task REX to write console messages.
	MSG	Prepares and sends requests to the specialized task REX to write console messages.
	HEADPREP	Provides, one record after the other, the separator and header for print files and the header card for punch files.
	MAKEBLOC	Saves the caller's registers for a call to VMSB2CP. Upon return from VMSB2CP, it sets the return code and returns to the original caller.
	VMSB2CP	Deblocks the VM/370 spool page buffers into an unpacked buffer (PACKBLK).
	AXSGET	Requests the specialized task AXS to open, close, and delete the spool files that the NPT task is processing.
	TODEBCD	Converts System/370 TOD to EBCDIC date and time.
	PARMGET	Scans character strings to find delimiter characters.
	NPTINIT	Initialization routine for NPT.
	NPTLINK	NPT sign-on routine.
	NPTERROR	Writes the terminal I/O error message and terminates the task.
	NPTTERM	Terminates the NPT task.
DMTPST	DMTPST	A service routine that may be called from anywhere in RSCS. DMTPST signals the completion of an event by posting the event's associated synch lock. This routine is entirely reentrant and does not change the state of running PSW.
DMTQRQ	DMTQRQ	Manages the MSUP supervisor status queue for other MSUP functions. DMTQRQ is for use within the supervisor and be entered with all PSW masks off (except machine check).

Module Name	Entry Points	Function
DMTREX	DMTREX	This routine is the controlling supervisor task and together with DMTCMX, DMTMGX, DMTSYS, DMTCOM, DMTMSG, and DMTCRE make up the REX supervisor task.
	REXINIT	Performs the initialization for the DMTREX task.
	REXCYCLE	Monitors a list of synch locks when looking for work for DMTREX to perform.
	REXPCHEX	Processes program checks.
	REXITERM	Entered when RSCS initialization fails. Issues the initialization failure message, dumps the contents of main storage, types any remaining messages, and loads a disabled wait state PSW.
	REQXEQ	Scans the function table and calls the appropriate routine based on that code (either DMTCMX or DMTMGX).
	DEACT	Deactivates the link table entry.
	MSG	Writes messages.
	TERMINAT	Terminates a specified task.
	QUIESCE	Becomes the task code for a task in the process of termination. Looks for any outstanding I/O for the terminating task. If any outstanding I/O is found, issues HIO and waits for completion. When all I/O is completed, it terminates the task.
DMTSIG	DMTSIG	Performs a task alert exit for a requesting task.
DMTSML	DMTSML	Functions as an RJE work station into a remote system using the MULTI-LEAVING transmission protocol. It can also function as a host to a remote programmable work station supporting a System/370, System/3, Model 20, 1130, or a 2922.
	SMLINIT	Initializes various parameters needed by DMTSML. Saves the link table address, initializes output tags, and constructs the sign-on card from information in the operand field of the START command.
	ISIO	Performs the enable sequence on the communications line analyzes the response received. If the response is correct, writes the line connected message.
	ASYNEXIT	This is the alert exit entered by DMTSIG. Two tasks may alert this line driver: • DMTREX--When a command has been entered for processing by the DMTSML line driver. • DMTAXS--When DMTAXS must asynchronously notify DMTSML that a file has arrived for transmission.

Module Name	Entry Points	Function
DMTSML (cont.)	&START	This is the supervisor routine for DMTSML. The commutator cycles while looking for a routine to enter until all commutator entries are closed. It then waits for a synch lock list to be posted.
	&CTRN1	Dequeues tasks from its task queue and perfcrms the action requested by the control record in the dequeued task.
	&PRTN1	Dequeues tasks from its task queue, obtains a new output spool device, if needed, from DMTAXS, and sends the task to a virtual printer.
	&URTN1	Dequeues tasks from its task queue, obtains a new output spool device, if needed, from DMTAXS, and sends the task to a virtual punch.
	&JRTN1	Dequeues tasks from its task queue, obtains a new output spool device, if needed, from DMTAXS, and sends the task to a virtual device.
	&USREXIT	Validates the ID card in the front of decks read in from a remote card reader.
	&PRTN1	Reads in files from the VM/370 spool file system, deblocks the files into 132 byte records, and issues a call to PUT to block the record into a transmission buffer.
	AXSGET	This routine is the interface to DMTAXS. It gets files ready to transmit and purges those files when transmission is complete.
	VMDEBLOK	This is the deblock routine for the VM/370 page spool buffers. It returns the deblocked record in the RDTTDTA1 buffer.
	HEADPREP	Provides, one record after the other, the separator and header for print files and the header card fcr punch files.
	TODEBCD	Converts System/370 TOD tc EBCDIC data and time.
	&WRTN1	Writes received messages to the RSCS operator, if in RJE mode. Passes commands to DMTREX for execution, if in HOST mode. These commands or messages are dequeued from console TCT.
	CMDPROC	Executes commands passed to it in the CMDRESP buffer after an alert from DMTREX indicating a command was entered.
	MSGPROC	Entered when the MSGECB is posted by this task's asynchronous exit indicating messages are in the message queue for this task. These messages are unstacked from the message queue by repeated calls to GMSGREQ and queued for transmission.

Module Name	Entry Points	Function
DMTSML (cont.)	MSG	Prepares and sends requests to the specialized task REX to writes messages on the operator's console.
	PARMGET	Scans lines and tests for delimiter characters.
	&TPPUT	Takes a line and packs it into a teleprocessing buffer. When the buffer is filled, it is queued onto OUTBUF for processing by COMSUP.
	&TPGET	Deblocks received telecommunications buffers into tasks and queues the task ontc the appropriate prccessors TCTTASK queue.
	COMSUP	Processes all I/O on the communications line. It dequeues TP buffers from OUTBUF for transmission and queues received TP buffers onto the &INBUF queue for deblocking by TPGET.
	CERROR	Analyses all errors on the communications line. The appropriate corrective action is taken depending on the type of error.
DMTSTO	DMTSTO	Reserves pages of free storage for use by calling task programs. Task programs free storage pages by making the associated map byte zero in the main storage map.
DMTSVC	DMTSVC	This module is the MSUP interrupt handler and receives control directly when an SVC interrupt occurs.
DMTSYS	DMTSYS	The common system control information area that is shared by all task level functions of RSCS. All installation variable information used by an RSCS system is reflected in the assembly of this module. This module is the only module that must be assembled as part of an RSCS system generation.
DMTVEC	DMTVEC	Describes the fixed address storage utilization for MSUP, beginning at main storage address X'200'. System/370 architecture defies the first 512 bytes of main storage and MSUP uses this area as it is defined. This area is not included in the DMTVEC mcdule to facilitate initial system loading. This area is initialized by DMTINI at IPL time.
DMTWAT	DMTWAT	Task programs call this module by a BAL instruction. It synchronizes events by suspending the execution of a task until another process in the system signals that a specified event has completed.

RSCS Module-to-Label Cross Reference

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
DMTAKE	ACTIVE	DISPATCH	GIVEADDR	GIVEE	GIVENAME	GIVENEXT	GIVENID	GIVEQ	GIVERID	POSTREQ	QREQ	R1	R11
	R12	R13	R14	R15	R2	R3	R4	R5	R6	SVECTORS	TAREA	TASKE	TASKID
	TASKNAME	TASKNEXT	TASKQ	TGREG1	TGREG15	TREQLOCK							
DMTASK	ACTIVE	ALERTQ	DISPATCH	EXTQ	FREEE	FREEID	FREENEXT	GIVEADDR	GIVEE	GIVENEXT	GIVENID	GIVEQ	IOE
	IOEXITQ	IOID	IONEXT	IOSUBQ	LIMBO	MAINMAP	MAINSIZE	MPXIOQ	POSTREQ	QREQ	R0	R1	R12
	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9	SELIOQ	SVECTORS
	TAREA	TASKE	TASKID	TASKNAME	TASKNEXT	TASKQ	TASKSAVE	TASKSTAT	TGREG0	TGREG13	TGREG15		
DMTASY	ACTIVE	ALERTQ	ASYNCODE	ASYNE	ASYNEXIT	ASYNID	ASYNNEXT	ASYNTASK	DISPATCH	EXTQ	IOEXITQ	LACTIVE	LACTTNME
	LFLAG	LINKLEN	LINKTABL	LNKCLOCK	QREQ	R0	R1	R10	R12	R13	R14	R15	R2
	R3	R4	R5	R6	R7	R8	R9	SVECTORS	TAREA	TASKE	TASKID	TASKNAME	TGREG0
	TGREG15	TGREG2	TLINKS										
DMTAXS	ALERTREQ	ASYNREQ	COMDSECT	CSW	DE	DEVCODE	DEVCUU	GIVEREQ	GLINKREQ	GPAGEREQ	GTODEBCD	IOTABLE	LACTCLS1
	LACTIVE	LACTTNME	LALERT	LFLAG	LINKID	LINKLEN	LINKTABL	LPENDING	LPOINTER	LRESERVD	LSPARE	LTAKEN	MAINMAP
	POSTREQ	PROGADDR	ROUTDEST	ROUTE	ROUTNEXT	ROUTSIZE	R0	R1	R10	R11	R12	R13	R14
	R15	R2	R3	R4	R5	R6	R7	R8	R9	SFBCLAS	SFBCOPY	SFBDATE	SFBDIST
	SFBFILID	SFBFLAG	SFBFLAG2	SFBFNAME	SFBFTYPE	SFBINUSE	SFBLOK	SFBORIG	SFBRECNO	SFBRECSZ	SFBREQUE	SFBSHOLD	SFBTYPE
	SFBUHOLD	SVECTORS	TAG	TAGBLOCK	TAGCLASS	TAGCOPY	TAGDEV	TAGDIST	TAGFLAG	TAGFLAG2	TAGID	TAGINDEV	TAGINLOC
	TAGINTOD	TAGINVM	TAGLEN	TAGLINK	TAGNAME	TAGNEXT	TAGPRIOR	TAGRECLN	TAGRECNM	TAGTOLOC	TAGTOVM	TAGTYPE	TAKEREQ
	TASKE	TASKSAVE	TCOM	TLINKS	TROUTE	TTAGQ	TYPPRT	TYPPUN	TYP1403	TYP2540P	TYP3203	TYP3211	WAITREQ
DMTCMX	ALERTREQ	COMDSECT	DEVCODE	DEVCUU	DMTCRE	DMTCREDA	DMTMGX	DMTREXCN	DMTREXHC	DMTREXID	GLINKREQ	GTODEBCD	IOTABLE
	LACTCLS1	LACTDRVR	LACTIVE	LACTLINE	LACTTNME	LDEFCLS1	LDEFDRVR	LDEFLINE	LDEFTNME	LDRAIN	LFLAG	LHOLD	LINKID
	LINKLEN	LINKTABL	LPENDING	LPOINTER	LRESERVD	LTAKEN	LTRALL	LTRERR	MAINMAP	MAINSIZE	R0	R1	R10
	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9
	SFBSHOLD	SFBUHOLD	SVECTORS	TAG	TAGBLOCK	TAGCLASS	TAGCOPY	TAGDIST	TAGFLAG	TAGID	TAGINLOC	TAGINTOD	TAGINVM
	TAGLINK	TAGNAME	TAGNEXT	TAGPRIOR	TAGRECNM	TAGTOLOC	TAGTOVM	TCOM	TLINKS	TPORTS	TTAGQ		
DMTCOM	ACTIVE	DISPATCH	LACTTNME	LINKID	LINKLEN	LINKTABL	LMSGQ	MAINMAP	MAINREQ	MAINSIZE	R0	R1	R10
	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9
	SVECTORS	TAREA	TASKE	TASKID	TASKNAME	TASKNEXT	TASKQ	TGREG0	TGREG1	TGREG15	TGREG2	TLINKS	TPSW
DMTCRE	CC	CE	CUE	DE	DEVCODE	ENDCSW	IOREQ	IOTABLE	LACTDRVR	LACTTNME	LINKTABL	MAINMAP	MAINREQ
	MAINSIZE	R0	R1	R12	R14	R15	R2	R3	R4	R5	R6	R7	R9
	SILI	SIOCOND	SVECTORS	TAREA	TASKREQ	TGREG0	TGREG1	TGREG2	TYP2314	WAITREQ			
DMTDSP	ACTIVE	LIMBO	LOCKLIST	NEWPSW	R0	R1	R15	R2	R3	R4	SVECTORS	TAREA	TASKE
	TASKID	TASKNEXT	TASKQ	TASKSAVE	TASKSTAT	TGREG0	TGREG1	TPSW	WAITING				
DMTEXT	ACTIVE	ASYNCODE	ASYNE	ASYNEXIT	ASYNNEXT	ASYNTASK	DISPATCH	EXTQ	LACTIVE	LACTTNME	LFLAG	LINKLEN	LINKTABL
	LNKCLOCK	NEWEXT	OLDEXT	QREQ	R0	R1	R10	R13	R14	R15	R2	R3	R4
	R5	R8	R9	SSAVE	SVECTORS	TAREA	TASKE	TASKNAME	TASKSAVE	TGREG0	TGREG14	TLINKS	TPSW

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
DMTGIV	ACTIVE	DISPATCH	GIVEADDR	GIVEE	GIVENAME	GIVENEXT	GIVENID	GIVEQ	GIVERID	POSTREQ	QREQ	R0	R1
	R12	R13	R14	R15	R2	R3	R4	SVECTORS	TAREA	TASKE	TASKID	TASKNAME	TASKNEXT
	TASKQ	TASKSAVE	TGREG15	TREQLOCK									
DMTINI	CAW	CC	CE	CLASDASD	CLASTERM	CSW	DE	DEVCODE	DEVCUU	DISPATCH	DMTCREDA	DMTIOMIN	DMTMAPME
	DMTMAPQE	DMTREXVL	FREEE	FREENEXT	FREEQ	IOTABLE	IPLCCW1	IPLPSW	MAINMAP	MAINSIZE	MCHEK	NEWEXT	NEWIO
	OLDIO	QREQ	QUEUE	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3
	R4	R5	R6	R7	R8	R9	SILI	SVECTORS	TASKE	TASKID	TASKNAME	TASKNEXT	TASKQ
	TASKSAVE	TASKSTAT	TIMER	TYP2314	TYP3210	TYP3330	TYP3340	TYP3350	WAIT				
DMTIOM	ACTIVE	ASYNCODE	ASYNE	ASYNEXIT	ASYNNEXT	ASYNTASK	BUSY	CAW	CE	CHANDONE	CSW	DE	DEVCUU
	DISPATCH	ENDCSW	ENDSENSE	IOADDR	IOE	IOEXITQ	IOID	IONEXT	IOSBCHAN	IOSTAT	IOSUBQ	IOSYNCH	IOTABLE
	IOTABLEA	MPXIOQ	NEWIO	OLDIO	PCI	POSTREQ	PROGADDR	QREQ	R0	R1	R12	R13	R14
	R15	R2	R3	R4	R5	R6	SELIOQ	SENSING	SENSREQ	SIOCOND	SM	SSAVE	SVECTORS
	TAREA	TASKE	TASKID	TASKSAVE	TGREG0	TGREG14	TPSW	UC					
DMTLAX	ASYNREQ	CLASTERM	LACTIVE	LACTLINE	LFLAG	LINKID	LINKLEN	LINKTABL	R0	R1	R12	R14	R15
	R2	R3	R4	R5	R6	R7	R8	R9	SVECTORS	TLINKS	TPORTS	TYPBSC	TYP2700
	WAITREQ												
DMTMGX	ALERTREQ	COMDSECT	DMTMSG	DMTREXHC	GLINKREQ	LACTIVE	LACTTNME	LFLAG	LINKID	LINKTABL	PMSGREQ	R0	R1
	R10	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7	R8	R9
	SVECTORS	TCOM	TLINKS										
DMTNPT	ASYNREQ	BUSOUT	CC	CMDREJ	COMDSECT	EQCHK	GIVEREQ	GMSGREQ	GPAGEREQ	GTODEBCD	INTREQ	IOREQ	LACTLINE
	LDRAIN	LERRCNT	LFLAG	LHOLD	LINKID	LINKTABL	LTOCNT	LTRALL	LTRERR	LTRNSCNT	PMSGREQ	POSTREQ	R0
	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4	R5	R6	R7
	R8	R9	SILI	SKIP	SPLINK	SPRECNUM	SVECTORS	TAG	TAGDEV	TAGDIST	TAGID	TAGINDEV	TAGINLOC
	TAGINTOD	TAGINVM	TAGLINK	TAGNAME	TAGNEXT	TAGRECNM	TAGTOLOC	TAGTOVM	TASKE	TASKSAVE	TCOM	TLINKS	TYPPRT
	TYPPUN	TYP2700	TYP3210	UC	UE	WAITREQ							
DMTPST	R0	R1	R14	TASKE	TASKSTAT	WAITING							
DMTQRQ	FREEE	FREEID	FREENEXT	FREEQ	R1	R14	R15	SVECTORS					
DMTREX	ACTIVE	ASYNREQ	ATTN	COMDSECT	CSW	DEVCODE	DEVCUU	DISPATCH	DMTCMX	DMTCOMVC	DMTCRE	DMTMGX	DMTSYSLK
	DMTSYSND	DMTSYSPT	DMTSYSRT	DMTSYSTQ	ENDCSW	GMSGREQ	IOADDR	IOE	IOID	IONEXT	IOREQ	IOSYNCH	IOTABLE
	IOTABLEA	LACTDRVR	LACTIVE	LACTLINE	LACTTNME	LDEFDRVR	LFLAG	LHALT	LIMBO	LINKID	LINKLEN	LINKTABL	LMSGQ
	LOCKLIST	MAINMAP	MAINSIZE	MPXIOQ	NEWPROG	OLDPROG	POSTREQ	PROGADDR	R0	R1	R12	R13	R14
	R15	R2	R3	R4	R5	SELIOQ	SILI	SSAVE	SVECTORS	TAKEREQ	TAREA	TASKE	TASKID
	TASKNAME	TASKNEXT	TASKQ	TASKREQ	TASKSAVE	TASKSTAT	TCOM	TGREG0	TGREG12	TGREG13	TGREG2	TGREG4	TLINKS
	TPORTS	TPSW	TVECTOR0	TYP3210	UE	WAITING	WAITREQ						
DMTSIG	ACTIVE	ALERTQ	ASYNE	ASYNEXIT	ASYNNEXT	ASYNTASK	DISPATCH	R0	R13	R14	R15	R2	R3
	SVECTORS	TAREA	TASKE	TASKNAME	TGREG15	TGREG2							

MODULE	EXTERNAL REFERENCES (LABELS AND MODULES)												
DMTSML	ASYNREQ	CC	CCC	CD	COMDSECT	DEVCUU	ENDCSW	GIVEREQ	GMSGREQ	GPAGEREQ	GTODEBCD	IOREQ	IOSYNCH
	IOTABLE	LACTLINE	LDRAIN	LERRCNT	LFLAG	LHOLD	LINKID	LINKTABL	LTOCNT	LTRALL	LTRERR	LTRNSCNT	PMSGREQ
	POSTREQ	PROGADDR	R0	R1	R10	R11	R12	R13	R14	R15	R2	R3	R4
	R5	R6	R7	R8	R9	SILI	SKIP	SPLINK	SPRECNUM	SVECTORS	TAG	TAGDEV	TAGDIST
	TAGID	TAGINDEV	TAGINLOC	TAGINTOD	TAGINVM	TAGLINK	TAGNAME	TAGRECNM	TAGTOLOC	TAGTOVM	TASKE	TASKSAVE	TCOM
	TLINKS	TYPPRT	TYPPUN	TYP2700	TYP3210	UC	UE	WAITREQ					
DMTSTO	ACTIVE	DISPATCH	MAINMAP	R0	R1	R14	R15	R2	R3	R4	SVECTORS	TAREA	TASKE
	TASKID	TGREG1	TGREG15										
DMTSVC	ACTIVE	NEWPSW	NEWSVC	OLDSVC	R0	R13	R14	R15	SSAVE	SVECTORS	TAREA	TASKE	TASKSAVE
	TGREG0	TGREG13	TGREG14	TPSW									
DMTSYS	LINKLEN	ROUTSIZE	TAGLEN										
DMTVEC	DMTAKE	DMTASK	DMTASY	DMTDSP	DMTGIV	DMTIOMRQ	DMTMAPMS	DMTMAPQE	DMTMAPQU	DMTPST	DMTQRQ	DMTSIG	DMTSTO
	DMTWAT												
DMTWAT	ACTIVE	DISPATCH	LOCKLIST	R1	R14	R15	R2	R3	R4	R5	R6	SVECTORS	TASKE
	TASKSTAT	WAITING											

RSCS Label-to-Module Cross Reference

LABEL	COUNT	REFERENCES											
ACTIVE	000027	DMTAKE	DMTASK	DMTASY	DMTCCM	DMTDSP	DMTEXT	DMTGIV	DMTIOM	DMTREX	DMTSIG	DMTSTO	DMTSVC
		DMTWAT											
ALERTQ	000003	DMTASK	DMTASY	DMTSIG									
ALERTREQ	000005	DMTAXS	DMTCMX	DMTMGX									
ASYNCODE	000008	DMTASY	DMTEXT	DMTIOM									
ASYNE	000016	DMTASY	DMTEXT	DMTIOM	DMTSIG								
ASYNEXIT	000005	DMTASY	DMTEXT	DMTIOM	DMTSIG								
ASYNID	000003	DMTASY											
ASYNNEXT	000017	DMTASY	DMTEXT	DMTIOM	DMTSIG								
ASYNREQ	000007	DMTAXS	DMTLAX	DMTNPT	DMTREX	DMTSML							
ASYNTASK	000007	DMTASY	DMTEXT	DMTIOM	DMTSIG								
ATTN	000001	DMTREX											
BUSOUT	000001	DMTNPT											
BUSY	000001	DMTIOM											
CAW	000006	DMTINI	DMTIOM										
CC	000100	DMTCRE	DMTINI	DMTNPT	DMTSML								
CCC	000001	DMTSML											
CD	000001	DMTSML											
CE	000004	DMTCRE	DMTINI	DMTIOM									
CHANDONE	000004	DMTIOM											
CLASDASD	000001	DMTINI											
CLASTERM	000005	DMTINI	DMTLAX										
CMDREJ	000001	DMTNPT											
COMDSECT	000006	DMTAXS	DMTCMX	DMTMGX	DMTNPT	DMTREX	DMTSML						
CSW	000026	DMTAXS	DMTINI	DMTIOM	DMTREX								
CUE	000001	DMTCRE											
DE	000006	DMTAXS	DMTCRE	DMTINI	DMTICM								
DEVCODE	000014	DMTAXS	DMTCMX	DMTCRE	DMTINI	DMTREX							
DEVCUU	000009	DMTAXS	DMTCMX	DMTINI	DMTICM	DMTREX	DMTSML						
DISPATCH	000016	DMTAKE	DMTASK	DMTASY	DMTCOM	DMTEXT	DMTGIV	DMTINI	DMTIOM	DMTREX	DMTSIG	DMTSTO	DMTWAT
DMTAKE	000001	DMTVEC											
DMTASK	000001	DMTVEC											
DMTASY	000001	DMTVEC											
DMTCMX	000001	DMTREX											
DMTCOMVC	000001	DMTREX											
DMTCRE	000003	DMTCMX	DMTREX										
DMTCREDA	000003	DMTCMX	DMTINI										
DMTDSP	000001	DMTVEC											
DMTGIV	000001	DMTVEC											
DMTIOMIN	000001	DMTINI											
DMTIOMRQ	000001	DMTVEC											
DMTMAPME	000001	DMTINI											
DMTMAPMS	000001	DMTVEC											
DMTMAPQE	000002	DMTINI	DMTVEC										
DMTMAPQU	000001	DMTVEC											
DMTMGX	000010	DMTCMX	DMTREX										

LABEL	COUNT	REFERENCES			
DMTMSG	000001	DMTMGX			
DMTPST	000001	DMTVEC			
DMTQRQ	000001	DMTVEC			
DMTREXCN	000001	DMTCMX			
DMTREXHC	000004	DMTCMX	DMTMGX		
DMTREXID	000001	DMTCMX			
DMTREXVL	000001	DMTINI			
DMTSIG	000001	DMTVEC			
DMTSTO	000001	DMTVEC			
DMTSYSLK	000001	DMTREX			
DMTSYSND	000001	DMTREX			
DMTSYSPT	000001	DMTREX			
DMTSYSRT	000001	DMTREX			
DMTSYSTQ	000001	DMTREX			
DMTWAT	000001	DMTVEC			
ENDCSW	000014	DMTCRE	DMTIOM	DMTREX	DMTSML
ENDSENSE	000001	DMTIOM			
EQCHK	000001	DMTNPT			
EXTQ	000004	DMTASK	DMTASY	DMTEXT	
FREEE	000008	DMTASK	DMTINI	DMTQRQ	
FREEID	000002	DMTASK	DMTQRQ		
FREENEXT	000009	DMTASK	DMTINI	DMTQRQ	
FREEQ	000005	DMTINI	DMTQRQ		
GIVEADDR	000004	DMTAKE	DMTASK	DMTGIV	
GIVEE	000013	DMTAKE	DMTASK	DMTGIV	
GIVENAME	000003	DMTAKE	DMTGIV		
GIVENEXT	000014	DMTAKE	DMTASK	DMTGIV	
GIVENID	000005	DMTAKE	DMTASK	DMTGIV	
GIVEQ	000005	DMTAKE	DMTASK	DMTGIV	
GIVEREQ	000018	DMTAXS	DMTNPT	DMTSML	
GIVERID	000002	DMTAKE	DMTGIV		
GLINKREQ	000003	DMTAXS	DMTCMX	DMTMGX	
GMSGREQ	000004	DMTNPT	DMTREX	DMTSML	
GPAGEREQ	000005	DMTAXS	DMTNPT	DMTSML	
GTODEBCD	000004	DMTAXS	DMTCMX	DMTNPT	DMTSML
INTREQ	000001	DMTNPT			
IOADDR	000008	DMTIOM	DMTREX		
IOE	000024	DMTASK	DMTIOM	DMTREX	
IOEXITQ	000003	DMTASK	DMTASY	DMTIOM	
IOID	000004	DMTASK	DMTIOM	DMTREX	
IONEXT	000015	DMTASK	DMTIOM	DMTREX	
IOREQ	000013	DMTCRE	DMTNPT	DMTREX	DMTSML
IOSBCHAN	000006	DMTIOM			
IOSTAT	000009	DMTIOM			
IOSUBQ	000007	DMTASK	DMTIOM		
IOSYNCH	000021	DMTIOM	DMTREX	DMTSML	

LABEL	COUNT	REFERENCES										
IOTABLE	000036	DMTAXS	DMTCMX	DMTCRE	DMTINI	DMTIOM	DMTREX	DMTSML				
IOTABLEA	000009	DMTIOM	DMTREX									
IPLCCW1	000001	DMTINI										
IPLPSW	000005	DMTINI										
LACTCLS1	000005	DMTAXS	DMTCMX									
LACTDRVR	000008	DMTCMX	DMTCRE	DMTREX								
LACTIVE	000021	DMTASY	DMTAXS	DMTCMX	DMTEXT	DMTLAX	DMTMGX	DMTREX				
LACTLINE	000013	DMTCMX	DMTLAX	DMTNPT	DMTREX	DMTSML						
LACTTNME	000023	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTEXT	DMTMGX	DMTREX			
LALERT	000005	DMTAXS										
LDEFCLS1	000004	DMTCMX										
LDEFDRVR	000005	DMTCMX	DMTREX									
LDEFLINE	000004	DMTCMX										
LDEFTNME	000004	DMTCMX										
LDRAIN	000013	DMTCMX	DMTNPT	DMTSML								
LERRCNT	000008	DMTNPT	DMTSML									
LFLAG	000074	DMTASY	DMTAXS	DMTCMX	DMTEXT	DMTLAX	DMTMGX	DMTNPT	DMTREX	DMTSML		
LHALT	000003	DMTREX										
LHOLD	000017	DMTCMX	DMTNPT	DMTSML								
LIMBO	000005	DMTASK	DMTDSP	DMTREX								
LINKID	000045	DMTAXS	DMTCMX	DMTCOM	DMTLAX	DMTMGX	DMTNPT	DMTREX	DMTSML			
LINKLEN	000019	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTEXT	DMTLAX	DMTREX	DMTSYS			
LINKTABL	000017	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTEXT	DMTLAX	DMTMGX	DMTNPT	DMTREX	DMTSML
LMSGQ	000005	DMTCOM	DMTREX									
LNKCLOCK	000006	DMTASY	DMTEXT									
LOCKLIST	000004	DMTDSP	DMTREX	DMTWAT								
LPENDING	000018	DMTAXS	DMTCMX									
LPOINTER	000014	DMTAXS	DMTCMX									
LRESERVD	000006	DMTAXS	DMTCMX									
LSPARE	000002	DMTAXS										
LTAKEN	000006	DMTAXS	DMTCMX									
LTOCNT	000008	DMTNPT	DMTSML									
LTRALL	000016	DMTCMX	DMTNPT	DMTSML								
LTRERR	000013	DMTCMX	DMTNPT	DMTSML								
LTRNSCNT	000008	DMTNPT	DMTSML									
MAINMAP	000016	DMTASK	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTINI	DMTREX	DMTSTO			
MAINREQ	000002	DMTCOM	DMTCRE									
MAINSIZE	000007	DMTASK	DMTCMX	DMTCOM	DMTCRE	DMTINI	DMTREX					
MCHEK	000004	DMTINI										
MPXIOQ	000007	DMTASK	DMTIOM	DMTREX								
NEWEXT	000003	DMTEXT	DMTINI									
NEWIO	000004	DMTINI	DMTIOM									
NEWPROG	000004	DMTREX										
NEWPSW	000006	DMTDSP	DMTSVC									
NEWSVC	000001	DMTSVC										
OLDEXT	000002	DMTEXT										

LABEL	COUNT	REFERENCES											
OLDIO	000006	DMTINI	DMTIOM										
OLDPROG	000001	DMTREX											
OLDSVC	000004	DMTSVC											
PCI	000001	DMTIOM											
PMSGREQ	000003	DMTMGX	DMTNPT	DMTSML									
POSTREQ	000012	DMTAKE	DMTASK	DMTAXS	DMTGIV	DMTIOM	DMTNPT	DMTREX	DMTSML				
PROGADDR	000012	DMTAXS	DMTIOM	DMTREX	DMTSML								
QREQ	000015	DMTAKE	DMTASK	DMTASY	DMTEXT	DMTGIV	DMTINI	DMTIOM					
QUEUE	000001	DMTINI											
ROUTDEST	000001	DMTAXS											
ROUTE	000001	DMTAXS											
ROUTNEXT	000002	DMTAXS											
ROUTSIZE	000003	DMTAXS	DMTSYS										
R0	000549	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTDSP	DMTEXT	DMTGIV	DMTINI	DMTIOM	DMTLAX
		DMTMGX	DMTNPT	DMTPST	DMTREX	DMTSIG	DMTSML	DMTSTO	DMTSVC				
R1	001103	DMTAKE	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTDSP	DMTEXT	DMTGIV	DMTINI	DMTIOM
		DMTLAX	DMTMGX	DMTNPT	DMTPST	DMTQRQ	DMTREX	DMTSML	DMTSTO	DMTWAT			
R10	000062	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTEXT	DMTINI	DMTMGX	DMTNPT	DMTSML			
R11	000033	DMTAKE	DMTAXS	DMTCMX	DMTCOM	DMTINI	DMTNPT	DMTSML					
R12	000050	DMTAKE	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTGIV	DMTINI	DMTIOM	DMTLAX	DMTMGX
		DMTNPT	DMTREX	DMTSML									
R13	000190	DMTAKE	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTEXT	DMTGIV	DMTINI	DMTIOM	DMTMGX	DMTNPT
		DMTREX	DMTSIG	DMTSML	DMTSVC								
R14	001131	DMTAKE	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTEXT	DMTGIV	DMTINI	DMTIOM	DMTLAX
		DMTMGX	DMTNPT	DMTPST	DMTQRQ	DMTREX	DMTSIG	DMTSML	DMTSTO	DMTSVC	DMTWAT		
R15	000966	DMTAKE	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTDSP	DMTEXT	DMTGIV	DMTINI	DMTIOM
		DMTLAX	DMTMGX	DMTNPT	DMTQRQ	DMTREX	DMTSIG	DMTSML	DMTSTO	DMTSVC	DMTWAT		
R2	000744	DMTAKE	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTDSP	DMTEXT	DMTGIV	DMTINI	DMTIOM
		DMTLAX	DMTMGX	DMTNPT	DMTREX	DMTSIG	DMTSML	DMTSTO	DMTWAT				
R3	000728	DMTAKE	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTDSP	DMTEXT	DMTGIV	DMTINI	DMTIOM
		DMTLAX	DMTMGX	DMTNPT	DMTREX	DMTSIG	DMTSML	DMTSTO	DMTWAT				
R4	000627	DMTAKE	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTDSP	DMTEXT	DMTGIV	DMTINI	DMTIOM
		DMTLAX	DMTMGX	DMTNPT	DMTREX	DMTSML	DMTSTO	DMTWAT					
R5	000458	DMTAKE	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTEXT	DMTINI	DMTIOM	DMTLAX	DMTMGX
		DMTNPT	DMTREX	DMTSML	DMTWAT								
R6	000470	DMTAKE	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTINI	DMTIOM	DMTLAX	DMTMGX	DMTNPT
		DMTSML	DMTWAT										
R7	000335	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTINI	DMTLAX	DMTMGX	DMTNPT	DMTSML	
R8	000383	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTEXT	DMTINI	DMTLAX	DMTMGX	DMTNPT	DMTSML	
R9	000140	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTEXT	DMTINI	DMTLAX	DMTMGX	DMTNPT	DMTSML
SELIOQ	000007	DMTASK	DMTIOM	DMTREX									
SENSING	000003	DMTIOM											
SENSREQ	000002	DMTIOM											
SFBCLAS	000001	DMTAXS											
SFBCOPY	000001	DMTAXS											
SFBDATE	000001	DMTAXS											

LABEL	COUNT	REFERENCES											
SFBDIST	000001	DMTAXS											
SFBFILID	000010	DMTAXS											
SFBFLAG	000002	DMTAXS											
SFBFLAG2	000001	DMTAXS											
SFBFNAME	000001	DMTAXS											
SFBFTYPE	000001	DMTAXS											
SFBINUSE	000001	DMTAXS											
SFBLOK	000002	DMTAXS											
SFBORIG	000002	DMTAXS											
SFBRECNO	000001	DMTAXS											
SFBRECSZ	000001	DMTAXS											
SFBREQUE	000004	DMTAXS											
SFBSHOLD	000004	DMTAXS	DMTCMX										
SFBTYPE	000001	DMTAXS											
SFBUHOLD	000005	DMTAXS	DMTCMX										
SILI	000145	DMTCRE	DMTINI	DMTNPT	DMTREX	DMTSML							
SIOCOND	000005	DMTCRE	DMTIOM										
SKIP	000003	DMTNPT	DMTSML										
SM	000001	DMTIOM											
SPLINK	000006	DMTNPT	DMTSML										
SPRECNUM	000018	DMTNPT	DMTSML										
SSAVE	000011	DMTEXT	DMTIOM	DMTREX	DMTSVC								
SVECTORS	000022	DMTAKE	DMTASK	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTCRE	DMTDSP	DMTEXT	DMTGIV	DMTINI	DMTIOM
		DMTLAX	DMTMGX	DMTNPT	DMTQRQ	DMTREX	DMTSIG	DMTSML	DMTSTO	DMTSVC	DMTWAT		
TAG	000040	DMTAXS	DMTCMX	DMTNPT	DMTSML								
TAGBLOCK	000021	DMTAXS	DMTCMX										
TAGCLASS	000009	DMTAXS	DMTCMX										
TAGCOPY	000011	DMTAXS	DMTCMX										
TAGDEV	000016	DMTAXS	DMTNPT	DMTSML									
TAGDIST	000015	DMTAXS	DMTCMX	DMTNPT	DMTSML								
TAGFLAG	000008	DMTAXS	DMTCMX										
TAGFLAG2	000004	DMTAXS											
TAGID	000023	DMTAXS	DMTCMX	DMTNPT	DMTSML								
TAGINDEV	000022	DMTAXS	DMTNPT	DMTSML									
TAGINLOC	000012	DMTAXS	DMTCMX	DMTNPT	DMTSML								
TAGINTOD	000007	DMTAXS	DMTCMX	DMTNPT	DMTSML								
TAGINVM	000008	DMTAXS	DMTCMX	DMTNPT	DMTSML								
TAGLEN	000002	DMTAXS	DMTSYS										
TAGLINK	000020	DMTAXS	DMTCMX	DMTNPT	DMTSML								
TAGNAME	000009	DMTAXS	DMTCMX	DMTNPT	DMTSML								
TAGNEXT	000053	DMTAXS	DMTCMX	DMTNPT									
TAGPRIOR	000012	DMTAXS	DMTCMX										
TAGRECLN	000001	DMTAXS											
TAGRECNM	000004	DMTAXS	DMTCMX	DMTNPT	DMTSML								
TAGTOLOC	000013	DMTAXS	DMTCMX	DMTNPT	DMTSML								
TAGTOVM	000017	DMTAXS	DMTCMX	DMTNPT	DMTSML								

LABEL	COUNT	REFERENCES											
TAGTYPE	000001	DMTAXS											
TAKEREQ	000002	DMTAXS	DMTREX										
TAREA	000021	DMTAKE	DMTASK	DMTASY	DMTCOM	DMTCRE	DMTDSP	DMTEXT	DMTGIV	DMTIOM	DMTREX	DMTSIG	DMTSTO
		DMTSVC											
TASKE	000044	DMTAKE	DMTASK	DMTASY	DMTAXS	DMTCOM	DMTDSP	DMTEXT	DMTGIV	DMTINI	DMTIOM	DMTNPT	DMTPST
		DMTREX	DMTSIG	DMTSML	DMTSTO	DMTSVC	DMTWAT						
TASKID	000022	DMTAKE	DMTASK	DMTASY	DMTCOM	DMTDSP	DMTGIV	DMTINI	DMTIOM	DMTREX	DMTSTO		
TASKNAME	000018	DMTAKE	DMTASK	DMTASY	DMTCOM	DMTEXT	DMTGIV	DMTINI	DMTREX	DMTSIG			
TASKNEXT	000023	DMTAKE	DMTASK	DMTCOM	DMTDSP	DMTGIV	DMTINI	DMTREX					
TASKQ	000010	DMTAKE	DMTASK	DMTCOM	DMTDSP	DMTGIV	DMTINI	DMTREX					
TASKREQ	000003	DMTCRE	DMTREX										
TASKSAVE	000015	DMTASK	DMTAXS	DMTDSP	DMTEXT	DMTGIV	DMTINI	DMTIOM	DMTNPT	DMTREX	DMTSML	DMTSVC	
TASKSTAT	000014	DMTASK	DMTDSP	DMTINI	DMTPST	DMTREX	DMTWAT						
TCOM	000019	DMTAXS	DMTCMX	DMTMGX	DMTNPT	DMTREX	DMTSML						
TGREG0	000020	DMTASK	DMTASY	DMTCOM	DMTCRE	DMTDSP	DMTEXT	DMTIOM	DMTREX	DMTSVC			
TGREG1	000005	DMTAKE	DMTCOM	DMTCRE	DMTDSP	DMTSTO							
TGREG12	000001	DMTREX											
TGREG13	000005	DMTASK	DMTREX	DMTSVC									
TGREG14	000003	DMTEXT	DMTIOM	DMTSVC									
TGREG15	000020	DMTAKE	DMTASK	DMTASY	DMTCOM	DMTGIV	DMTSIG	DMTSTO					
TGREG2	000008	DMTASY	DMTCOM	DMTCRE	DMTREX	DMTSIG							
TGREG4	000001	DMTREX											
TIMER	000001	DMTINI											
TLINKS	000032	DMTASY	DMTAXS	DMTCMX	DMTCOM	DMTEXT	DMTLAX	DMTMGX	DMTNPT	DMTREX	DMTSML		
TPORTS	000003	DMTCMX	DMTLAX	DMTREX									
TPSW	000014	DMTCOM	DMTDSP	DMTEXT	DMTICM	DMTREX	DMTSVC						
TREQLOCK	000004	DMTAKE	DMTGIV										
TROUTE	000001	DMTAXS											
TTAGQ	000005	DMTAXS	DMTCMX										
TVECTOR0	000001	DMTREX											
TYPBSC	000002	DMTLAX											
TYPPRT	000009	DMTAXS	DMTNPT	DMTSML									
TYPPUN	000010	DMTAXS	DMTNPT	DMTSML									
TYP1403	000001	DMTAXS											
TYP2314	000005	DMTCRE	DMTINI										
TYP2540P	000001	DMTAXS											
TYP2700	000004	DMTLAX	DMTNPT	DMTSML									
TYP3203	000001	DMTAXS											
TYP3210	000009	DMTINI	DMTNPT	DMTREX	DMTSML								
TYP3211	000001	DMTAXS											
TYP3330	000002	DMTINI											
TYP3340	000002	DMTINI											
TYP3350	000002	DMTINI											
UC	000010	DMTIOM	DMTNPT	DMTSML									
UE	000003	DMTNPT	DMTREX	DMTSML									
WAIT	000002	DMTINI											

```
LABEL    COUNT     REFERENCES

WAITING  000005    DMTDSP   DMTPST   DMTREX   DMTWAT
WAITREQ  000030    DMTAXS   DMTCRE   DMTLAX   DMTNPT   DMTREX   DMTSML
```

RSCS Diagnostic Aids

This section contains the RSCS Message-to-Label Cross Reference

RSCS Message-to-Label Cross Reference

Message Code	Generated at Label	Message Text
DMTAXS101I	TAGPEND	FILE 'spoolid' ENQUEUED ON LINK 'linkid'
DMTAXS102I	ACCEPEND	FILE 'spoolid' PENDING FOR LINK 'linkid'
DMTAXS103E	ACCEPURG	FILE 'spoolid' REJECTED -- INVALID DESTINATION ADDRESS
DMTAXS104I	CLOOSCAN	FILE SPOOLED TO 'userid2' -- ORG 'locid1' ('userid1')
		mm/dd/yy hh:mm:ss
DMTAXS105I	CLOIPURG	FILE 'spoolid' PURGED
DMTAXS106I	FILSTRY	FILE 'spoolid'MISSING -- DEQUEUED FROM LINK 'linkid'
	OPENPOOF	
DMTAXS107I	UNPECHEK	nn PENDING FILES FOR LINK 'linkid' MISSING
DMTAXS108E	OPENRDER	SYSTEM ERROR READING SPOOL FILE 'spoolid'
DMTAXS520I	CHANGE	File 'spoolid' CHANGED
DMTAXS521I	CHANHO	FILE 'spoolid' HELD FOR LINK 'linkid'
DMTAXS522I	CHANNOH	FILE 'spoolid' RELEASED FOR LINK 'linkid'
DMTAXS523I	CHANSCAN	LINK 'linkid' QUEUE REORDERED
	ORDENEXT	
DMTAXS524E	CHANGE	FILE 'spoolid' ACTIVE -- NO ACTION TAKEN
	ORDECHEK	
	PURGCHEK	
DMTAXS525E	CHANGE	FILE 'spoolid' IS FOR LINK 'linkid' -- NO ACTION TAKEN
	ORDECHEK	
	PURGCHEK	
DMTAXS526E	CHANGE	FILE 'spoolid' NOT FOUND -- NO ACTION TAKEN
	ORDECHEK	
	PURGCHEK	
DMTAXS640I	PURGDONE	nn FILE(S) PURGED ON LINK 'linkid'
DMTCMX001I	CMXFINXT	FREE STORAGE = nn PAGES
DMTCMX003I	CMXM003	LINK 'linkid' EXECUTING: (command line)
DMTCMX200I	CMXLGOT	RSCS
DMTCMX201E	CMXHIT	INVALID COMMAND 'command'
	CMXLGOT	
	CMXMISS	
DMTCMX202E	DEFNOLNK	INVALID LINK 'linkid'
	MSGNOLNK	
DMTCMX203E	A1FLKGOT	INVALID SPOOL FILE ID 'spoolid'
	A1FSTOW	
	CHALKGOT	
	L2FLKGOT	
	QY0FILE	
	QY0FNULL	
DMTCMX204E	CHALKGOT	INVALID KEYWORD 'keyword'
	CHANTERM	
	CHASCAN	
	FLUMORE	
	LOTERM	
	L1TERM	
	QYTOOMCH	
	QY0FILE	
	QY0LINK	
	QY0SYSTM	
	R0SCAN	
DMTCMX205E	CHACLASS	CONFLICTING KEYWORD 'keyword'
	CHACOPY	

Message Code	Generated at Label	Message Text
DMTCMX205E (cont.)	CHAHOLD	
	CHANOHOL	
	CHAPRIOR	
	FLUKEYWD	
	LOTKEYWD	
	ROCLASS	
	ROKEEP	
	ROLINE	
	ROTASK	
	ROTYPE	
DMTCMX206E	CHACLASS	INVALID OPTION 'keyword' 'option'
	CHACOPY	
	CHADIST	
	CHANAME	
	CHAPRIOR	
	LOHOLD	
	LOTRACE	
	L1FLKGOT	
	QUERY	
	ROCLASS	
	ROCLMULT	
	ROKEEP	
	ROLINE	
	ROTASK	
	ROTYPE	
DMTCMX208E	DISCONN	INVALID USER ID 'userid'
	MSGNOLNK	
	MSGNOUSR	
DMTCMX300I	CMXALRDY	ACCEPTED BY TASK 'task'
DMTCMX301E	CMXALRDY	REJECTED BY TASK 'task' -- PREVIOUS COMMAND ACTIVE
DMTCMX302E	MSGNOLNK	LINK 'linkid' IS NOT DEFINED
DMTCMX303E	CMD	LINK 'linkid' IS NOT ACTIVE
	LOFLKGOT	
	L1FLKGOT	
	L2FLKGOT	
	MSG	
DMTCMX304E	CMXALRDY	REJECTED BY TASK 'task' NOT RECEIVING
DMTCMX540I	DEFLKNEW	NEW LINK 'linkid' DEFINED
DMTCMX541I	DEFLKNEW	LINK 'linkid' REDEFINED
DMTCMX542E	DEFINE	LINK 'linkid' ACTIVE -- NOT REDEFINED
DMTCMX543E	DEFNEXT	LINK 'linkid' NOT DEFINED -- LINK LIMIT REACHED
	DEFNOLNK	
DMTCMX544E	DEFLKNEW	LINK 'linkid' NOT DEFINED -- LIMIT REACHED
DMTCMX550I	DELDELET	LINK 'linkid' NOW DELETED
DMTCMX551E	DELETE	LINK 'linkid' ACTIVE -- NOT DELETED
DMTCMX552E	DELETE	LINK 'linkid' HAS A FILE QUEUE -- NOT DELETED
DMTCMX560I	DISCHARG	RSCS DISCONNECTING
DMTCMX561E	DISCONN	USERID 'userid' NOT RECEIVING
DMTCMX651I	QY1STAT	LINK 'linkid' INACTIVE
DMTCMX652I	QY1SNOD	LINK 'linkid' ACTIVE 'type' 'vaddr' c (HO|NOH) (DR|NCD) (TRA|TRE|NOT) Q=m P=n
DMTCMX653I	QY1DEF	LINK 'linkid' DEFAULT 'task' 'type' 'vaddr' c R=m
DMTCMX654I	QY1QUEUE	LINK 'linkid' Q=m P=n
DMTCMX655I	QY1INACT	FILE 'spoolid' 'locid' 'userid' CL a PR mm REC nnnnnn
DMTCMX660I	QY2STAT	FILE 'spoolid' INACTIVE ON LINK 'linkid'
DMTCMX661I	QY2STAT	FILE 'spoolid' ACTIVE ON LINK 'linkid'
DMTCMX662I	QY2RSS	FILE 'spoolid' ORG 'locid' 'userid' mm/dd/yy hh:mm:ss zzz TO 'locid' 'userid' VIA 'linkid'
DMTCMX663I	QY2VNOH	FILE 'spoolid' PR mm CL a CO nn (HO|NOH) DI 'distcode', NA ('fn ft'|'dsname')

Message Code	Generated at Label	Message Text
DMTCMX664E	QY2RSS QY2STAT QY2VM QY2VNOH	FILE 'spoolid' NOT FOUND
DMTCMX670I	QYSYACT	LINK 'linkid' ACTIVE -- LINE 'vaddr' (HO|NOH)
DMTCMX671I	QYM671	LINK 'linkid' INACTIVE
DMTCMX672I	QYSYNEXT	NO LINK ACTIVE
DMTCMX673I	QYM673	NO LINK DEFINED
DMTCMX700I	STALNGOT	ACTIVATING LINK 'linkid' 'task' 'type' 'vaddr'
DMTCMX701E	STACREAT	NO SWITCHED LINE AVAILABLE -- LINK 'linkid' NOT ACTIVATED
DMTCMX702E	STACREAT	LINE 'vaddr' IS IN USE BY LINK 'linkid1' -- LINK 'linkid2' NOT ACTIVATED
DMTCMX703E	STACREAT	DEV 'cuu' IS NOT A LINE PORT -- LINK 'linkid' NOT ACTIVATED
DMTCMX704E	STACREAT	LINE 'vaddr' CC=3 NOT OPERATIONAL -- LINK 'linkid' NOT ACTIVATED
DMTCMX705E	STACRERR	DRIVER 'type' NOT FOUND ON DISK 'vaddr' -- LINK 'linkid' NOT ACTIVATED
DMTCMX706E	STACRERR	FATAL ERROR LOADING FROM 'vaddr' -- LINK 'linkid' NOT ACTIVATED
DMTCMX707E	STACRERR	DRIVER 'type' FILE FORMAT INVALID -- LINK 'linkid' NOT ACTIVATED
DMTCMX708E	STACRERR	VIRTUAL STORAGE CAPACITY EXCEEDED -- LINK 'linkid' NOT ACTIVATED
DMTCMX709E	STACRERR	TASK NAME 'task' ALREADY IN USE -- LINK 'linkid' NOT ACTIVATED
DMTCMX710E	STAMAXER	MAX (nn) ACTIVE -- LINK 'linkid' NOT ACTIVATED
DMTCMX750E	STANOTCL	LINK 'linkid' ALREADY ACTIVE -- NO ACTION TAKEN
DMTCMX751I	CMXALRDY	LINK 'linkid' ALREADY ACTIVE -- NEW CLASS(ES) SET AS REQUESTED
DMTINI402T	INIEXIT	IPL DEVICE READ I/O ERROR
DMTINI406R	IPLDISK	SYSTEM DISK ADDRESS = cuu
DMTINI407R	ASKQUEST	REWRITE THE NUCLEUS? Y OR N
DMTINI409R	NUCCYLN	NUCLEUS CYL ADDRESS = nnn
DMTINI410R	IPLZERO	ALSO IPL CYLINDER 0? Y OR N
DMTINI431S	WRERROR	IPL DEVICE WRITE I/O ERROR
DMTINI479E	BINERR1	INVALID DEVICE ADDRESS - REENTER
DMTINI480E	DECERR1 RDORWRT	INVALID CYLINDER NUMBER - REENTER
DMTINI481E	IPLZERO	INVALID REPLY - ANSWER YES OR NO
DMTINI482E	BADIPLD	SYSTEM DISK ERROR - REENTER
DMTINI483E	NUCCYLN	NUCLEUS OVERLAYS CMS FILES - RECOMPUTE
DMTNPT070E	IOERRPRT	ERROR cuu SIOCC cc CSW csw SENSE sense CCW ccw
DMTNPT108E	VMSGET	SYSTEM ERROR READING SPOOL FILE 'spoolid'
DMTNPT141I	NPTEINIT	LINE 'vaddr' READY FOR CONNECTION TO LINK 'linkid'
DMTNPT142I	NPTEINIT	LINK 'linkid' LINE 'vaddr' CONNECTED
DMTNPT143I	LINEDIS2 LINEDROP	LINK 'linkid' LINE 'vaddr' DISCONNECTED
DMTNPT144I	PUTOPEN	RECEIVING: FILE FROM 'lccid1' ('name1') FOR 'locid2' ('name2')
DMTNPT145I	PUTCLS1	RECEIVED: FILE FROM 'locid1' ('name1') FOR 'lccid2' ('name1')
DMTNPT146I	GETGOT2	SENDING: FILE 'spoolid' ON LINK 'linkid', REC nnnnnn
DMTNPT147I	GETPURGE	SENT: FILE 'spoolid' ON LINK 'linkid'
DMTNPT149I	TRPRT	LINK 'linkid' LINE ACTIVITY: TOT= mmm; ERRS= nnn; TMOUTS= ppp
DMTNPT190E	VMSP1	INVALID SPOOL BLOCK FORMAT ON FILE 'spoolid'
DMTNPT510I	GTBKMSG	FILE 'spoolid' BACKSPACED
DMTNPT511E	SBKFWDN	NO FILE ACTIVE ON LINK 'linkid'

Message Code	Generated at Label	Message Text
DMTNPT570I	SETDRAIN	LINK 'linkid' NOW SET TO DEACTIVATE
DMTNPT571E	SETDRER1	LINK 'linkid' ALREADY SET TO DEACTIVATE
DMTNPT580I	GETFLUSH	FILE 'spoolid' PROCESSING TERMINATED
DMTNPT581E	SETFLUSH GETFLSHE	FILE 'spoolid' NOT ACTIVE
DMTNPT590I	SETFREE	LINK 'linkid' RESUMING FILE TRANSFER
DMTNPT591E	SETFRER1	LINK 'linkid' NOT IN HOLD STATUS
DMTNPT600I	GDGODNE	FILE 'spoolid' FORWARD SPACED
DMTNPT610I	SETHOLD GETFILE	LINK 'linkid' TO SUSPEND FILE TRANSMISSION
DMTNPT611I	SETHLDIM GETFILE	LINK 'linkid' FILE TRANSMISSION SUSPENDED
DMTNPT612E	SETHLDE1	LINK 'linkid' ALREADY IN HOLD STATUS
DMTNPT750E	SETSTRT1	LINK 'linkid' ALREADY ACTIVE -- NO ACTION TAKEN
DMTNPT752I	SETSTART	LINK 'linkid' STILL ACTIVE -- DRAIN STATUS RESET
DMTNPT801I	SETTR1	LINK 'linkid' ERROR TRACE STARTED
DMTNPT802I	SETTR2	LINK 'linkid' TRACE STARTED
DMTNPT803I	SETTRACE	LINK 'linkid' TRACE ENDED
DMTNPT810E	SETTRE1	LINK 'linkid' TRACE ALREADY ACTIVE
DMTNPT811E	SETTRE2	LINK 'linkid' TRACE NOT ACTIVE
DMTNPT902E	CONFCK1	NON-SIGNON CARD READ ON LINK (linkid)
DMTNPT903E	SPASS	PASSWORD (password) on LINK (linkid) IS INVALID
DMTNPT904E	SGNERR	SIGNON KEYWORD (keyword) INVALID
DMTNPT905I	NPTGETX	SIGNON OF LINK 'linkid' COMPLETE
DMTNPT934E	PUTCLOSE	ID MISSING ON LINK 'linkid' -- INPUT FILE PURGED
DMTNPT936E	GETGOT1	NO REMOTE PUNCH AVAILABLE ON LINK 'linkid' -- FILE 'spoolid' PURGED
DMTREX000I	REXICGOT	RSCS (VER v, LEV l, mm/dd/yy) READY
DMTREX002I	TERLHIT	LINK 'linkid' DEACTIVATED
DMTREX080E	TERLHIT	PROGRAM CHECK -- 'linkid' DEACTIVATED
DMTREX090T	REXPTERM	PROGRAM CHECK IN SUPERVISOR -- RSCS SHUTDOWN
DMTREX091T	REXITERM	INITIALIZATION FAILURE - RSCS SHUTDOWN
DMTSML070E	IOERRPRT	I/O ERROR -- SIOCC -- CSW -- SENSE -- CCW --
DMTSML108E	VMSPGET	SYSTEM ERROR READING SPOOL FILE 'spoolid'
DMTSML141I	ISIO	LINE 'vaddr' READY FOR CONNECTION TO LINK 'linkid'
DMTSML142I	SIGNOK	LINK 'linkid' LINE 'vaddr' CONNECTED
DMTSML143I	EOJ	LINK 'linkid' LINE 'vaddr' DISCONNECTED
DMTSML144I	JOUTPUT PCONT UOUTPUT	RECEIVING: FILE FROM 'locid1' ('name1') FOR 'locid2' ('name2')
DMTSML145I	JCLOSE1 PCLOSE UCLOSE	RECEIVED: FILE FROM 'locid1' ('name1') FOR 'lccid2' ('name2')
DMTSML146I	RLOC1	SENDING: FILE 'spoolid' ON LINK 'linkid', REC nnnnnn
DMTSML147I	RDEOF	SENT: FILE 'spoolid' ON LINK 'linkid'
DMTSML149I	TRPRT	LINK 'linkid' LINE ACTIVITY: TOT= mmm; ERRS= nnn; TMOUTS= ppp
DMTSML170I	WGET2	FROM 'linkid': (MSG message text)
DMTSML190E	VMSP1	INVALID SPOOL BLOCK FORMAT ON FILE 'spoolid'
DMTSML510I	RDBKMSG	FILE 'spoolid' BACKSPACED
DMTSML511E	SBKFWDN	NO FILE ACTIVE ON LINK 'linkid'
DMTSML530I	SETCMD	COMMAND FORWARDED ON LINK 'linkid'
DMTSML570I	SETDRAIN $USRNPUN	LINK 'linkid' NOW SET TO DEACTIVATE
DMTSML571E	SETDRER1	LINK 'linkid' ALREADY SET TO DEACTIVATE
DMTSML580I	RDFLUSH	FILE 'spoolid' PROCESSING TERMINATED
DMTSML581E	SETFLUSH RDFLSHER	FILE 'spoolid' NOT ACTIVE
DMTSML590I	SETFREE	LINK 'linkid' RESUMING FILE TRANSFER
DMTSML591E	SETFRER1	LINK 'linkid' NOT IN HOLD STATUS

Message Code	Generated at Label	Message Text
DMTSML600I	RDGODNE	FILE 'spoolid' FORWARD SPACED
DMTSML610I	SETHOLD	LINK 'linkid' TO SUSPEND FILE TRANSMISSION
DMTSML611I	ALLHLD SETHLDIM	LINK 'linkid' FILE TRANSMISSION SUSPENDED
DMTSML612E	SETHLDE1	LINK 'linkid' ALREADY IN HOLD STATUS
DMTSML750E	SETSTRT1	LINK 'linkid' ALREADY ACTIVE -- NO ACTION TAKEN
DMTSML752I	SETSTART	LINK 'linkid' STILL ACTIVE -- DRAIN STATUS RESET
DMTSML801I	SETTR1	LINK 'linkid' ERROR TRACE STARTED
DMTSML802I	SETTR2	LINK 'linkid' TRACE STARTED
DMTSML803I	SETTRACE	LINK 'linkid' TRACE ENDED
DMTSML810E	SETTRE1	LINK 'linkid' TRACE ALREADY ACTIVE
DMTSML811E	SETTRE2	LINK 'linkid' TRACE NOT ACTIVE
DMTSML901E	SMLIERR1	INVALID SML MODE SPECIFIED -- LINK 'linkid' NOT ACTIVATED
DMTSML902E	MC7ERR	NON-SIGNON CARD READ ON LINK (linkid)
DMTSML903E	MC7A	PASSWORD (password) ON LINK (linkid) IS INVALID
DMTSML905I	MC7B	SIGNON OF LINK 'linkid' COMPLETE
DMTSML906E	SMLIERR2	INVALID SML BUFFER PARAMETER -- LINK 'linkid' NOT ACTIVATED
DMTSML934E	JCLOSE	ID CARD MISSING ON LINK 'linkid' -- INPUT FILE PURGED
DMTSML935E	RDNOHLD	LINK 'linkid' IN RJE MODE -- PRINT FILE 'spoolid' PURGED

READER'S COMMENT FORM

Title: IBM Virutal Machine Facility/370: System Logic and Problem Determination Guide Volume 3

Order No. SY20-0888-0

Please check or fill in the items; adding explanations/comments in the space provided.

Which of the following terms best describes your job?

☐ Customer Engineer	☐ Manager	☐ Programmer	☐ Systems Analyst
☐ Engineer	☐ Mathematician	☐ Sales Representative	☐ Systems Engineer
☐ Instructor	☐ Operator	☐ Student/Trainee	☐ Other (explain below)

How did you use this publication?

☐ Introductory text ☐ Reference manual ☐ Student/ ☐ Instructor text

☐ Other (explain) _____

Did you find the material easy to read and understand? ☐ Yes ☐ No (explain below)

Did you find the material organized for convenient use? ☐ Yes ☐ No (explain below)

Specific criticisms (explain below)

Clarifications on pages _____

Additions on pages _____

Deletions on pages _____

Errors on pages _____

Explanations and other comments:

Note: Staples can cause problems with automated mail sorting equipment. Please use pressure sensitive or other gummed tape to seal this form.

Trim Along This Line

Thank you for your cooperation. No postage necessary if mailed in the U.S.A.

YOUR COMMENTS PLEASE . . .

Your views about this publication may help improve its usefulness; this form will be sent to the author's department for appropriate action. Using this form to request system assistance and/or additional publications or to suggest programming changes will delay response, however. *For more direct handling of such request, please contact your IBM representative or the IBM Branch Office serving your locality.* Your comments will be carefully reviewed by the person or persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Fold Fold

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department D58, Building 706-2
PO Box 390
Poughkeepsie, New York 12602

Attn: VM/370 Publications

Fold Fold

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601

Cut or Fold Along Line

IBM VM/370: System Logic and Problem Determination Guide Volume 3 Printed in U.S.A. SY20-0888-0