MERVA for ESA

Operations Guide

Version 4 Release 1

SH12-6375-01

MERVA for ESA

Operations Guide

Version 4 Release 1

SH12-6375-01

Note!
Before using this information and the product it supports, be sure to read the general information under "Appendix B. Notices" on page 385.

Second Edition, May 2001

This edition applies to Version 4 Release 1 of IBM MERVA for ESA (5648-B29) and to all subsequent releases and modifications until otherwise indicated in new editions.

Changes to this edition are marked with a vertical bar.

Contents

About This Book

This book is written for the operators of the IBM licensed program Message Entry and Routing with Interfaces to Various Applications for ESA Version 4 Release 1 (abbreviated to MERVA ESA in this book). The book describes the facilities available to you to carry out the tasks of:

- Starting and stopping MERVA ESA and its components
- Controlling and monitoring MERVA ESA and its components
- Using the MERVA ESA utility programs
- Dealing with problems arising from the use of MERVA ESA and its components.

Before using this book you should be familiar with *MERVA for ESA Concepts and Components*, which describes the functions, services, and utilities supplied. It is aimed at readers who want to get a general idea of the message concept, queues, routing, message handling, and the network links.

It is assumed that you are familiar with the operating system under which MERVA ESA is run.

If the SWIFT Link is installed, you should be familiar with SWIFT terminology as defined in the *S.W.I.F.T. User Handbook*, published by the Society for Worldwide Interbank Financial Telecommunication s.c. in La Hulpe, Belgium.

If the Telex Link is installed, you should be familiar with telex terminology as defined in the documentation provided by your local telecommunications authority.

Notation conventions are described in "Chapter 1. Introduction to the Commands" on page 3. All responses to the commands of MERVA ESA and its components are described in *MERVA for ESA Messages and Codes*.

Note: In this book the examples of job-control statements (JCLs) show only the main components as seen from MERVA ESA. Adapt these examples to meet the requirements of your installation.

The job control statements for VSE are based on the JCL specifications for VSE/ESA™ Version 2.

The term *CICS* is used to refer to CICS/ESA®, CICS TS, and CICS/VSE®. The term *IMS* is used to refer to IMS/ESA®.

Summary of Changes

This edition reflects the following changes:

FMT/ESA can now use MQI Attachment

Financial Message Transfer/ESA (FMT/ESA) can now use MQI Attachment as well as MERVA Link ESA to transfer SWIFT messages between two MERVA ESA systems (see "Chapter 9. Operating Financial Message Transfer/ESA (FMT/ESA)" on page 189).

Part 1. MERVA ESA Operator Commands

The commands in these chapters are presented in alphabetical order.

Chapter 1. Introduction to the Commands

In MERVA ESA, there are the following types of command:

- User commands, which are described in detail in the *MERVA for ESA User's Guide*.
- Operator commands, which are described in this book.

Entering Operator Commands

You can enter operator commands at either the system console, or a 327x display station. You can, however, enter the MERVA Link operator commands only at a 327x display station. The following explains how you enter commands at a console or a display station.

Entering Commands at the System Console

Entering commands at the operating system console requires the MERVA ESA console program DSLNMOP to be started automatically during the MERVA ESA startup, or manually by an authorized MERVA ESA operator using a display station.

The MERVA ESA console program issues the message `DSL001A Enter a MERVA command` at the console, indicating that you can now enter MERVA ESA commands at the system console:

- If you are working under VSE, enter the CICS transaction code defined for the program DSLCMO in your installation, followed by the operator command.

  The program DSLCMO always returns to an inactive state after an operator command has been processed. Therefore, each MERVA ESA operator command must be preceded by the transaction code.
- If you are working under MVS™, enter the MERVA ESA operator commands as replies to the `DSL001A Enter a MERVA command` message. Use the MVS **reply** command, followed by the appropriate MERVA ESA operator command.

You cannot use the operating system console to enter MERVA Link operator commands.

Entering Commands at a Display Station

You can enter operator commands at an IBM 327x display station when you are authorized to use the MERVA ESA Operator Command function (CMD) or the MERVA System Control function (MSC). To do so, you must sign on to MERVA ESA at any display station connected to IMS or CICS that has been authorized for signing on to MERVA ESA. See the *MERVA for ESA User's Guide* for information on signing on to MERVA ESA. See "MERVA System Control" on page 6 for information on the MERVA System Control function.

The Command Format

A MERVA ESA command can consist of either a command word with one or more parameters, or just a command word.

If a command contains any parameters, you must separate the command word and each of its parameters with a blank or a comma (,). MERVA ESA supports only positional parameters; keyword parameters are not supported. Therefore, when a parameter is not specified in a command and one of the parameters that follows the omitted parameter is specified, the omission must be indicated by specifying all the necessary commas or blanks. For example, the omission of the first parameter must be entered:

comword,,parm2

or

comword ,parm2

The following shows the possible formats that you can use to enter commands:

comword	
comword	parm1 parm2 parm3 ..... parm20
comword	parm1,parm2,parm3,.....,parm20

Command Words

A command word (**comword**) can have 1 to 8 characters. You can abbreviate the command words with more than 4 characters to their first 4 characters. The format description shows the shortest acceptable abbreviation. For example, the following is the format of the **reshut** command:

reshut

This means that **resh**, **reshu**, and **reshut** are all acceptable forms of this command. Some special abbreviations are defined for some of the command words of MERVA ESA. These are given on a separate line in the command description. For example, the abbreviation **rs** is defined for the command word **reshut**.

You can enter the command words and the abbreviations in either uppercase or lowercase.

Parameters

A parameter (*parm*) can have 1 to 63 characters. If the parameter contains a blank, comma, or any other special character, enclose the parameter in single quotation marks. For example, if you use the parameter CH,ICE, enter it as **'CH,ICE'**. MERVA ESA uses the single quotation marks to identify the beginning and end of the parameter.

If a parameter contains a single quotation mark, enter two single quotation marks instead. For example, enter the parameter CH'ICE as **'CH''ICE'**, where two single quotation marks replace the quotation mark that is part of the parameter. The beginning and end of the parameter are marked by single quotation marks ('). Do not use double quotation marks (").

Do not include the single quotation marks enclosing the parameter when determining the length of a parameter. If parameters are shown in *italics* in the format description, they can be substituted by an appropriate value. You can enter parameters in either uppercase or lowercase.

Optional Parameters

Parameters for some of the commands are optional. You can enter one, a combination, or none of the optional parameters. Brackets ([]) show that the

parameter is optional. Braces ({}) show that you must select one option from several. A bar (|) is used to separate a list of parameters of which one can be selected. You must not enter the brackets, braces, or bars.

An underlined parameter in the format description shows a default option. If no parameter is entered for the command, MERVA ESA assumes the underlined default.

diva	*line* [,Buffers | **States** | Vtambnd]

Mandatory Parameters: You *must* enter a parameter for some of the commands. Mandatory parameters are shown without brackets or braces.

force	*userid*

Help for Command Words

You can get help information for all MERVA ESA command codes when working at a display station in a MERVA ESA end user session. The help panel is displayed by entering the command:

help comword

The full command word must be specified as parameter, abbreviations or synonyms will not work. For example,

help login

will display the page with the explanation of the *login* command. This page is defined in the MCB for the SWIFT Link operator commands.

Program Function Keys

You can use the predefined program function (PF) keys instead of entering a command. For details on the PF-key settings refer to the *MERVA for ESA Customization Guide*.

Authorized Users

To use MERVA ESA operator commands you must be authorized to use the CMD or the MSC function. To use the MERVA Link operator commands you must be authorized to use the MSC function. The authorization to use a MERVA ESA function is defined in the User File record of the user.

The use of some MERVA ESA operator commands is restricted to authorized users. Such restrictions are indicated in the command description of this book. The authorization for using restricted operator commands is determined as follows:

- By the OPID parameter in the MERVA ESA customization parameter module DSLPRM.
- By the MQI parameter in the MERVA ESA function table DSLFNTT.
- By the User Type in the User File record.
- When using commands for the Telex Link via a fault-tolerant system the user ID must start with one of the following:
  - The 3 characters MAS.
  - The 3 characters specified with the OPNAM parameter in the Telex Link customization parameter module ENLPRM.

- By the restrictions set up during the customization of MERVA ESA in the user exit DSLNCU01.

MERVA System Control

The MERVA System Control Facility (MSC) provides a user interface to enter both MERVA ESA operator commands and MERVA Link operator commands. The MSC function uses several panels. The name of a panel is displayed in the upper left corner. All commands can be entered in any panel of the MSC function. Examples of the panels are in "Chapter 7. Operating MERVA Link ESA" on page 133. A short description of each panel follows.

MSC Main Menu and Local Help (ACMM)

When you have selected the MSC function in the MERVA ESA Function Selection menu, the MSC Main Menu (ACMM) is displayed. This panel contains a short description of the MERVA System Control Facility, and offers local help for the commands of all MERVA components.

The local help consists of several pages, and you can use the program function keys (PF keys) 7 and 8 to page backward and forward. You can select a MERVA component directly by positioning the cursor and pressing the ENTER key. You can enter commands on any page of these help panels.

When you enter the first command, one of the other panels is displayed with the command response. To redisplay the ACMM panel, press PF 9 (SWAP) once or twice, depending on the panel currently displayed, or enter the command **cmd** or **cmdu**.

If you enter the command **explain** or **xpl** on the ACMM panel, the abbreviations of all MERVA Base, SWIFT Link commands, and Telex Link commands when using a fault-tolerant system are displayed. You can enter any command on this panel. Press the ENTER key to return to the ACMM panel.

MERVA Operator Command Response Display (AC00)

The responses to MERVA Base and SWIFT Link commands, or commands for Telex Link are displayed on the panel AC00. The panel shows the command response and the current time. The last command remains displayed on the command line.

If a command response is longer than one page, you can use the PF 7 and PF 8 keys to page backward and forward in the response. For example, when you use the test command **jrn** (refer to the *MERVA for ESA Installation Guide* for details), the response can be more than one page, depending on the journal record. A maximum of five pages is supported by the AC00 panel. The continuation indicator (three periods at the end of the last command response data line) tells you that more data is available and that it can be displayed in the next page. Up to 89 command response lines can be displayed in these five pages. There is no continuation indicator at the end of line 89.

If you enter the command **explain** or **xpl** on the AC00 panel, the abbreviations of all MERVA Base and SWIFT Link commands and Telex Link commands are displayed.

You can enter any command or press any PF key on this panel. If you press the ENTER key, the ACMM panel is displayed. You cannot return to the AC00 panel and redisplay the command response.

MERVA Link ASP List Display (AC01)

The MSC panel AC01 shows a list of the *application support process* (ASP) entries defined in the MERVA Link Partner Table (see "Chapter 7. Operating MERVA Link ESA" on page 133 for details). Each ASP is displayed in one line.

If you enter the command **explain** or **xpl** on the AC01 panel, a subset of the MERVA Link commands is displayed. You can enter any command or press any PF key on this panel. If you press the ENTER key, the AC01 panel is redisplayed.

MERVA Link Specific ASP/MTP Display (AC02)

The MSC panel AC02 shows detailed information of a specific ASP and its associated *message transfer process* (MTP). See "Chapter 7. Operating MERVA Link ESA" on page 133 for details.

If you enter the command **explain** or **xpl** on the AC02 panel, an explanation of the data displayed on the AC02 panel is shown. You can enter any command or press any PF key on this panel. If you press the ENTER key, the AC02 panel is redisplayed.

MERVA Link SCP List Display (AC03)

The MSC panel AC03 is used by the Partner MERVA System Control Function to display the list of partner MERVA systems. For more information, refer to "Partner MERVA System Control".

MERVA Link PT Header Information Display (AC04)

The MSC panel AC04 is used to display information of a MERVA Link system:

- MERVA Link Partner Table Header information.
- The MERVA ESA identifier from DSLPRM.
- An indicator of the applicable data communication subsystem (CICS/MVS, CICS/ESA, or CICS/VSE for MERVA ESA CICS, or APPC/MVS for MERVA ESA IMS).
- In the MERVA ESA CICS environment, parameters and status information of the ASP Monitor. The ASP Monitor is not supported in the MERVA ESA IMS environment.

If you enter the command **explain** or **xpl** on the AC04 panel, an explanation of the data displayed on the AC04 panel is shown. You can enter any command or press any PF key on this panel. If you press the ENTER key, the AC04 panel is redisplayed.

Keeping the Command on the Command Line

The MERVA System Control Facility keeps the command in the command line for the majority of the commands. This is helpful if the same command is needed again, or if the command was incorrect and needs correction.

Partner MERVA System Control

A Partner MERVA System Control function is part of the MERVA System Control Facility. It provides the necessary functionality to operate a partner MERVA system within the local MSC function. The partner MERVA system must be connected to the local system through MERVA Link.

When the target MERVA system has been selected from a partner MERVA system list (SCP list) or has been directly specified, the MSC main menu of the partner MERVA system appears, and then all commands are executed in the partner MERVA system. Note that the user must have a user ID on the remote system.

Some elements of the Partner MERVA System Control Function are described in the following sections.

The MERVA Link System Control Process List

The panel AC03 is used by the Partner MERVA System Control Function to display the list of the MERVA Link System Control Processes (SCPs). It is the set of partner MERVA systems that can be operated from the local MSC function. Enter the command **node** or **ps** (for partner system) to display this list.

If this list does not fit on one page of the panel, you can use the scrolling keys (PF 6, PF 7, and PF 8) to display any partner MERVA system defined in the MERVA Link Partner Table (SCP entries). You can select a partner MERVA system in this list and switch to it by placing the cursor at the required entry and pressing PF 4 (SELECT).

When the command **explain** (or **xpl**) is entered on the AC03 panel, an environment explanation is displayed that contains a short description of the Partner MERVA System Control Function. Enter any command or press any PF key on this explanation panel to execute a function, or press the ENTER key to return to the SCP List Display.

Connecting to a Partner MERVA System

MERVA ESA provides methods by which you can connect to other MERVA systems, execute commands there, and switch back to the local system.

Direct Partner System Selection

You can directly switch from any panel in any partner MERVA system to a specific other partner MERVA system using the **node** or **ps** commands together with a valid partner MERVA system name. The MERVA Link node name of your local MERVA system is always a valid partner MERVA system name. It can be used to switch from a partner MERVA system back to the local MERVA system.

Switching to Another Partner MERVA System

The facilities to switch to a partner MERVA system are available at any time, regardless of whether you operate the local MERVA system or whether you have switched to a partner MERVA system. You can therefore directly switch from one partner system to another partner system with the command **node ps_name** or **ps ps_name**.

Partner System Indication

After a switch to a partner MERVA system, the MERVA Link node name of the partner MERVA system is displayed in the upper right corner of all MSC panels (as in the local MSC environment) in the color red to remind you that all commands apply to a partner MERVA system. The name of your local MERVA system is displayed in white.

Returning to the Local MERVA System

There are several ways of returning to the local MERVA system after a switch to a partner MERVA system:

- The commands **ps** or **node** without a parameter return to the local MERVA system and display the SCP list.

- The commands **ps *** or **node *** return to the local MERVA system and display the ASP list.
- The commands **ps local_node_name** or **node local_node_name** return to the local MERVA system and display the MSC main menu.
- The PF 3 key (**return** command) returns to the Function Selection Menu in the local MERVA system.

Commands Executed in the Local System

Some commands are always executed in the local system, that is, the partner MERVA system is not involved. These commands are:

- **node** and **ps**. If a partner system switch was active, these commands switch back to the local MERVA system.
- **explain** or **xpl**, and **qhelp** or **hlp**. The command is executed in the local system and you get the explanations for your local system. The MSC switch to a partner MERVA system is kept if a partner system switch was active.
- If you press the ENTER key on the panels ACMM and AC00 without a command in the command line, no empty command is transferred to the partner MERVA system. The current panel is redisplayed using information available in the local MERVA system. However, if you press the ENTER key without a command in one of the panels AC01, AC02, or AC04, an empty command is transferred to the partner MERVA system to request an update of the current display with the most recent information.

Chapter 2. Starting MERVA ESA

This chapter describes the actions used to start and stop MERVA ESA under the following operating systems:

- CICS/VSE
- MVS and CICS
- MVS and IMS

Note: Before starting MERVA ESA for the first time, you must have run all jobs distributed with the JCL file and referenced in the program directory.

MERVA ESA needs the following VSAM data sets:

- Journal A
- Journal B
- User File
- Queue Data Set (formatted)
- Large message cluster (LMC)
- Message counter log
- Nicknames File (optional)
- Authenticator-Key File (formatted or loaded)
- SWIFT Correspondents File (optional, initialized)
- SWIFT Currency Code File (optional, initialized)
- Telex Correspondents File (optional, initialized).

In addition, MERVA ESA-IMS needs the SPA File (non-VSAM).

If you run MERVA ESA with queue management using DB2®, MERVA ESA needs DB2 tables for the queue data set on DB2.

Starting MERVA ESA under CICS/VSE

To start MERVA ESA under CICS/VSE, do the following:

1. Start CICS/VSE.
2. Sign on to CICS/VSE by entering the transaction code **cssn**. You can do this at:
   - An IBM 327x Display Station
   - A VSE system console, used as a CICS terminal.

   The code **cssn** should be entered on a clear display panel.
3. Enter the MERVA ESA transaction code. Ask your system administrator for the transaction code defined for your installation. The code is defined in the CICS transaction definitions for the startup and command program DSLCMO.

If MERVA ESA is already running, a new start attempt is refused. If, because of an error, the status information contained in the CICS common work area (CWA) shows that MERVA ESA is starting but is not really active, an operator can start MERVA ESA using the NEW parameter together with the transaction name (for example DSL NEW). NEW must not be used if MERVA ESA is still active.

On successful completion of the MERVA ESA initialization, the message `DSL000A MERVA is ready` is displayed at the terminal (if a terminal was used for entering the transaction code) and at the VSE system console in either case.

Alternatively, the program DSLCAS is provided, which starts MERVA ESA automatically when CICS is started. This saves having to sign on to CICS to enter the DSL transaction. If you wish to use this program, it must be added to the CICS member DFHPLTPI as follows:

```
DFHPLT TYPE=ENTRY,PROGRAM=DSLCAS
```

You must also:

- Make the following definition in CICS RDO:
  ```
  DEFINE PROGRAM(DFHPLTPI) GROUP(....)
  ```
- Specify in the system initialization table (SIT):
  ```
  PLTPI=PI
  ```

Note: The suffix PI may be replaced by a different suffix in both places.

Starting MERVA ESA under CICS in MVS

To start MERVA ESA under CICS in MVS, do the following:

1. Start CICS.
2. Sign on to CICS by entering the transaction code **cesn**. Only an IBM 327x Display Station that has been defined as a CICS terminal can be used. CICS/MVS® does not support the MVS system console as a CICS terminal.

   The code **cesn** should be entered on a clear display screen.
3. Enter the MERVA ESA transaction code. Ask your system administrator for the transaction code defined for your installation. The transaction code is defined in the CICS transaction definitions for the startup and command program DSLCMO.

If MERVA ESA is already running, a new start attempt is refused. If, because of an error, the status information contained in the CICS common work area (CWA) shows that MERVA ESA is starting but is not really active, an operator can start MERVA ESA using the NEW parameter together with the transaction name (for example DSL NEW). NEW must not be used if MERVA ESA is still active.

On successful completion of the MERVA ESA initialization, the message `DSL000A MERVA is ready` is displayed at the screen terminal and at the MVS system console.

Alternatively, the program DSLCAS is provided to start MERVA ESA automatically when CICS is started. This saves having to sign on to CICS to enter the DSL transaction.

If you wish to use this program, DSLCAS must be added to the CICS member DFHPLTPI as follows:

```
DFHPLT TYPE=ENTRY,PROGRAM=DSLCAS
```

You must also:

- Make the following definition in CICS RDO:
  ```
  DEFINE PROGRAM(DFHPLTPI) GROUP(....)
  ```
- Specify in the system initialization table (SIT):
  ```
  PLTPI=PI
  ```

Note: The suffix PI may be replaced by a different suffix in both places.

Alternatively, MERVA ESA can run as a native MVS batch program or started task.

Starting MERVA ESA under IMS/ESA in MVS

To start MERVA ESA under IMS/ESA, do the following:

1. Start the IMS control region.
2. Start the MERVA ESA batch message program (BMP) region by using either an MVS start command, or the MVS job-control statements (via the MVS reader).

The message `DSL000A MERVA is ready` is displayed at the MVS system console, when the initialization is successfully completed.

Alternatively, MERVA ESA can run as a native MVS batch program or started task.

Starting MERVA ESA in a Multisystem Environment

If your MERVA ESA is defined to run in a multisystem environment under CICS/ESA or IMS/ESA where multiple MERVA ESA instances are distributed among multiple systems with shared resources, you have to consider the following:

- MERVA ESA V4 can consist of multiple MERVA ESA instances, each running on a different system or address space.
- There is only one primary MERVA ESA instance. Only the primary MERVA ESA instance accepts operator commands and MERVA ESA commands.
- The primary MERVA ESA instance should be started before any secondary MERVA ESA instance is started. To relief from starting secondary MERVA ESA instance(s) via the operator interface, parameters can be defined in DSLPRM to allow them to be started automatically by the primary MERVA ESA instance. However, this requires definition of a procedure to be started as a started task.

It is recommended to run all MERVA ESA instances as native MVS batch programs or started tasks.

Chapter 3. Stopping MERVA ESA

This chapter applies to all installations, irrespective of the operating system in use. To avoid losing data and disrupting user sessions and network connections, ensure that no session or connection is active before you stop MERVA ESA.

The commands used to stop MERVA ESA are:

- **shutdown**
- **cancel** or **terminat**

The normal sequence in which these commands are used is:

1. End the connections between MERVA ESA and the communication networks. For example, to end the connection to the SWIFT network, you use the SWIFT Link **logout** command.
2. Stop all the user sessions and hard-copy printer tasks. You use the MERVA ESA **shutdown** command to do this.

   The command automatically stops the printer tasks when they have finished processing the current message. User sessions return to the Function Selection menu. Users can then only select the Operator Command function (CMD), if this function is on the user's Function Selection menu.
3. When you have completed the first two steps, use the MERVA ESA **cancel** or **terminat** command to stop MERVA ESA.

   If you use the **cancel** or **terminat** command without completing the first two steps, the users may have to enter some data again, or some messages may be printed again on hard-copy printers.

If you have entered the **shutdown** command, but decide not to stop MERVA ESA, enter the command **reshut**. The **reshut** command resets the SHUTDOWN status so that users can sign on to MERVA ESA again or can select a function from the Function Selection menu. Printer tasks can be started.

If your MERVA ESA is defined to run in a multisystem environment where multiple MERVA ESA instances are distributed among multiple systems with shared resources, the command to stop MERVA ESA must be entered on the primary MERVA ESA instance. This command is propagated to all active MERVA ESA instances.

Chapter 4. Operating the Base Functions

The commands in this chapter are presented in alphabetical order.

Stopping MERVA ESA (CANCEL and TERMINAT)

Use the command **cancel** or **terminat** to stop MERVA ESA. These commands are identical and therefore interchangeable. If your installation has added programs to MERVA ESA via DSLNPTT that require special processing, you should ensure that this special processing has been carried out before you enter the **cancel** or **terminat** command. For example, when the SWIFT Link is running, you must log out all master logical terminals before terminating MERVA ESA, as the SWIFT Link cannot process **logout** during the termination.

You should also use the **shutdown** command before the **cancel** or **terminat** command to allow the users and hard-copy printer tasks to complete the message being processed.

Notes:

1. You must be authorized to use these commands.
2. Your system programmer will tell you when to use the parameters for these commands.
3. Under CICS: After having entered the **terminat** command, you should verify that the termination is complete before terminating CICS (for example, using CEMT INQ), otherwise, you may have a restart of the MERVA ESA queue management, or, when using the SWIFT network, you may have an abnormal end of MVS with system abend code A03, as a line subtask may still be attached.

Command Formats

The formats of the **terminat** and **cancel** commands are:

cancel **c**	[{DUMP | ABDUMP }]
terminat **t**	[{DUMP | ABDUMP }]

Parameter Descriptions

The parameters for these commands have the following meanings:

DUMP

A dump is taken before MERVA ESA is stopped. This dump is useful, for example, when an error message was issued but the program issuing the error message did not take a dump. The dump is taken before the message **TERMINATE/CANCEL accepted** is issued, and a normal end of MERVA ESA takes place after the dump.

In MERVA ESA running under CICS, this is a CICS transaction dump with dump code 0015. In MERVA ESA running under IMS, this is an MVS SNAP dump with ID=015.

ABDUMP

A dump is taken before MERVA ESA is stopped. MERVA ESA first removes the CICS or MVS abend exit and then causes a program check to happen. No MERVA ESA termination is processed. When MERVA ESA is started the next time, the MERVA ESA queue management processes a restart. In MERVA ESA running under CICS, it may be necessary to restart CICS.

The ABDUMP parameter can provide a dump that may be more helpful than the dump provided with the DUMP parameter, especially when problems of accesses to the operating system have to be analyzed. In CICS, the kind of dump depends on the abend recovery options used with CICS.

Command Example

You can use the abbreviation **t** instead of the **terminat** command:

t

Example of the Display from a CANCEL/TERMINAT Command

Figure 1 shows the display that would result if you entered the command shown in the command example. Instead of the **terminat** command, the **cancel** command can be entered. Message DSL051I returns the information resulting from the execution of this command.

```
                        Operator Command Processing

> T
  DSL051I TERMINAT accepted

  115033   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 1. Stopping MERVA ESA with TERMINAT

Figure 2 shows the unsolicited operator messages that are displayed only on the operating system console during the stopping of MERVA ESA. Each active program of the MERVA ESA program table is stopped individually. Message DWS771I is issued by the Authenticator-Key File program when it is terminated by the SWIFT Link program SWIFTAUT. Message DSL012I shows the MERVA ESA termination.

```
DSL061I RTCOMM 1 stop successful
DSL061I CONSOLE 2 stop successful
DSL061I BATCH 3 stop successful
DSL061I TRANSACT 4 stop successful
DSL061I MSGCOUNT 5 stop successful
DWS771I Keys found: 00000000 in aging table, 000000 on disk
DSL061I SWIFTAUT 6 stop successful
DWS690I VNDEBET2A GPA SN=0515, ISN=000002, OSN=000000, SESS=0437
DWS690I VNDEBET2A FIN SN=0511, ISN=009932, OSN=014264, SESS=0437
DWS690I VNDEBET3A GPA SN=0000, ISN=000000, OSN=000000, SESS=0000
DWS690I VNDEBET3A FIN SN=0000, ISN=000000, OSN=000000, SESS=0000
DWS690I VNDPBET2A GPA SN=0000, ISN=000000, OSN=000000, SESS=0000
DWS690I VNDPBET2A FIN SN=0000, ISN=000000, OSN=000000, SESS=0000
DSL061I SWIFTII 7 stop successful
DSL061I SWLOADSK 8 stop successful
ENL902I Telex Link has been terminated
DSL061I TELEX 9 stop successful
DSL012I MERVA has been terminated
```

Figure 2. Termination of MERVA ESA

Changing Logical Terminal Names (CF)

Some message-processing functions can be associated with a transaction and a logical terminal (LT) name. The transaction processes the messages written to the queue of the message-processing function, for example, prints them on a terminal printer with the logical terminal name defined for the function.

Use the **cf** command to change the logical terminal name to another logical terminal name. For example, if a function normally uses a specific printer, but this printer is out of order, you can assign another printer to the function by the logical terminal name.

Note: If you enter the **cf** command while the function is in ACTIVTD status, MERVA ESA sets the function in HOLD status to prevent processing of the rest of the queue with the old logical terminal name.

In MERVA IMS, remember that the IMS message queue may still contain start records for the old logical terminal name.

You must enter an **sf** command with the function to resume processing with the new logical terminal name. If related functions are defined, they all get the new logical terminal name when the **cf** command is used for one of them.

Note: You must be authorized to enter the **cf** command for a function which is defined in the MERVA ESA function table DSLFNTT using the parameter MQI=YES.

The transaction code can be a logical name pointing to an entry in the MERVA ESA transaction table DSLTXTT. In this case, the specification of a logical terminal in the transaction table entry gains precedence over the value in the function table entry and the logical terminal cannot be changed with the **cf** command.

Command Format

CF Command

The format of the **cf** command is:

cf	*function,ltname*

Parameter Descriptions

The parameters for this command have the following meanings:

function
The name of the message-processing function for which you want to change the logical terminal name.

ltname
The logical terminal name of the new device.

Note: *ltname* is checked against all the logical terminal names generated within the MERVA ESA Function Table, in the CICS terminal definitions, or in the IMS nucleus. The change is carried out if a match is found. If no match is found, a check is made against the DC system. When the terminal name is defined to the DC system, the change is carried out, otherwise the error message DSL114I is shown.

Command Example

This example shows the command you use to change the logical terminal used for the function L1PR0:

```
cf,l1pr0,l86a
```

Example of the Display from a CF Command

Figure 3 on page 21 shows the display that would result if you entered the command shown in the command example. Message DSL115I confirms the change of the terminal name, for more information refer to *MERVA for ESA Messages and Codes*.

```
                        Operator Command Processing

> CF L1PR0 L86A
  DSL115I L1PR0    CHANGED, TRAN=DSLH    , LT1=L86A    , LT2=L84A
```

```
  115343   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 3. Changing the Logical Terminal Name

This message shows the following information:

- The name of the function: L1PR0
- The confirmation that the command was accepted: CHANGED
- The name of the transaction: DSLH
- The logical terminal name to be used from now on for the transaction: L86A
- The original logical terminal name: L84A.

Displaying the Message Counter Log (DCLOG)

Use the **dclog** command to see online information about the current status of the MERVA ESA message counter data set.

Command Format

The format of the **dclog** command is:

dclog	[{ LAST | DETAIL }]

Parameter Descriptions

The parameters for this command have the following meanings:

LAST
Displays the status of the current month. A detailed report for message counters is given. When the parameter is not specified, the message counter log status for the last 12 months is shown.

DETAIL
Displays the status of all individual counters for the last 12 months. When the parameter is not specified, only the total number of messages is shown, but not the status of the individual counters.

Example of the Display from a DCLOG Command

Figure 4 shows an example of the information displayed when you enter the **dclog** command.

```
                        Operator Command Processing

> DCLOG
  DSL170I Message counter LOG status from 19980401 to 19990401 (12 months)
  DSL175I Licensed message usage                                      1000
  DSL171I Average monthly usage                                      14742
  DSL180I Usage exceeds the licensed tier, 14742 is required









  142411   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 4. Displaying the Message Counter Log Status

When you enter the **dclog** command the following information is displayed on the command panel to inform you about the current MERVA ESA usage. The messages are explained in *MERVA for ESA Messages and Codes*.

DSL170I Shows the reporting period. If MERVA ESA has been used for more than 12 months, only the last 12 months are taken into account.

DSL175I Shows the current licensed messages of the installation.

DSL171I Shows the average monthly usage during the reporting period.

DSL180I This message is shown only when the average monthly usage of MERVA ESA exceeds the licensed messages. In this case, contact your IBM marketing representative to license the additional messages.

Displaying the Function Status (DF)

Use the **df** command to monitor the status of either one message-processing function, or all message-processing functions for which a transaction name is defined.

Command Format

The format of the **df** command is:

df	[{ FIRST | *function* [,FIRST]}]

DF Command

The first time you enter the **df** command without parameters, the status of the first 13 message-processing functions is displayed. If you enter the same command again, the status of the next 13 functions is displayed.

Parameter Descriptions

The parameters for this command have the following meanings:

FIRST
Displays the status of the first 13 message-processing functions for which transaction codes are specified in the MERVA ESA Function Table.

function
Is the name, or part of the name, of a message-processing function defined in the MERVA ESA Function Table. There are two methods of entering part of a function name:

- You can enter 1 to 7 characters without using the substitution characters asterisk (*) or percent (%). This method displays the status of the first 13 functions beginning with these characters.
- You can enter 1 to 8 characters, including the substitution characters asterisk (*) and percent (%).
  - The substitution character * represents any number of characters. As many asterisks as required can appear anywhere in the function name.
  - The substitution character % is a place-holding character, representing any single character. As many percent symbols as necessary may appear anywhere in a function name.

If you enter the same command again, and there are more than 13 functions matching these characters, the next 13 functions are displayed. If you then want to see again the first 13 functions matching these characters, you must enter the **df** command with the function parameter followed by the FIRST parameter.

Command Examples

The following section shows some examples of how to enter the **df** command.

Example 1: To display a function enter the **df** command with a function name for the *function* parameter:

```
df,l1pr0
```

Example 2: To display the functions starting with the same characters, such as L1P, enter the following command:

```
df,l1p
```

Example 3: Use the FIRST parameter to display the first 13 functions:

```
df,first
```

Example 4: Use the substitution character * to display the first 13 functions whose names contain the characters “sw” anywhere in the name:

```
df,*sw*
```

Example 5: Use the substitution character % to display the first 13 functions whose names contain the characters “pr” in positions 3 and 4 of the function name:

```
df,%%pr*
```

Example of the Display from a DF Command

"Displaying the Function Status (DF)" on page 22 shows an example of the information that is returned when you enter the **df** command without parameters.

```
                        Operator Command Processing

> DF
  DSL118I Display Functions
   Function Status  Transact LTERM1   LTERM2
   DMPR0    HOLD    DSLH     PRT1
   DMPR1    HOLD    DSLH     PRT1
   DMPR2    HOLD    DSLH     PRT1
   DMPR3    HOLD    DSLH     PRT1
   DMPR4    HOLD    DSLH     PRT1
   DMPR89   HOLD    DSLH     PRT1
   L1PR0    NOHOLD  DSLH     PRT1
   L1PR1    NOHOLD  DSLH     PRT1
   L2PR0    NOHOLD  DSLH     PRT1
   L2PR1    NOHOLD  DSLH     PRT1
   L2PR1S   NOHOLD  DSLH     PRT1
   L3PR0    NOHOLD  DSLH     PRT1
   L3PR1    NOHOLD  DSLH     PRT1

  124642   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 5. Displaying the Status of Message-Processing Functions

Message DSL118I is displayed in response to the command, for details refer to *MERVA for ESA Messages and Codes*. The message contains the following information:

Function The name of the function, such as DMPR0.

Status The status of the message-processing function, such as HOLD.

Transact The transaction code associated with the function, such as DSLH.

Note: Transaction codes have 4 characters in CICS and up to 8 characters in IMS.

When the transaction table DSLTXTT is installed, the transaction code refers to an entry in the transaction table.

LTERM1 The logical terminal name currently used by the transaction, such as PRT1. This is either the logical terminal originally defined for the transaction, or the logical terminal name assigned with a **cf** command.

Note: Logical terminal names have 4 characters in CICS and up to 8 characters in IMS.

LTERM2 The logical terminal name as originally defined for the transaction, after a **cf** command has changed the currently-used terminal name.

Displaying the Large Message Cluster Status (DLMC)

Use the **dlmc** command to monitor the status of the large message cluster (LMC) and see information about the usage of space. The information helps you maintain the large message cluster with respect to size and performance.

Command Format

The format of the **dlmc** command is:

dlmc	

The **dlmc** command has no parameters.

Example of the Display from a DLMC Command

Figure 6 shows an example of the information displayed when you enter the **dlmc** command.

```
                        Operator Command Processing

>DLMC
 DSL414I Statistics for Large Message Cluster - Maintenance
 Lower limit for a large message           10.000 Bytes
 Number of large messages                     198
 Current average message length           105.162 Bytes
 Historical minimum message length         27.000 Bytes
 Historical maximum message length        498.196 Bytes
 Space allocated                       25.927.424 Bytes
 Space used                                    73 %
 Reorganization recommended, processing continues




 142411   is the time of this display

Command =====>
PF 1=Help     2=Repeat    3=Return     4=DF     5=DU          6=DM Last
PF 7=         8=          9=Hardcopy  10=DP    11=DQ filled  12=DL
```

Figure 6. Displaying Status of Large Message Cluster for Maintenance

When you enter the **dlmc** command the following information is displayed:

Lower limit for a large message
The value entered in DSLPRM or the default value of 31900 bytes, which is the maximum value of a small message.

Number of large messages
The number of large messages in the LMC at the time the **dlmc** command was entered.

Current average message length
The current average length of TOF data of all large messages in the LMC at the time the **dlmc** command was entered.

Historical minimum message length
The minimum TOF data length historically observed for all large messages in the LMC since its last reorganization or definition.

Historical maximum message length
The maximum TOF data length historically observed for all large messages in the LMC since its last reorganization or definition.

Space allocated
The high allocated relative byte address (RBA) value for the LMC, that is, the maximum number of bytes available as disk space for the large message cluster.

Space used
The filling status of the LMC, that is, the percentage calculated from stored bytes divided by high allocated RBA at the time the **dlmc** command was entered.

Reorganization recommended, processing continues
This text is displayed when the LMC is processed in insert mode and the LMC is more than 70% filled.

Reorganization recommended, LMC is full
This text is displayed when the LMC is processed in insert mode and the LMC is considered full, that is, the maximum number of put retries has been reached for inserting a record.

LMC not closed since date/time
This text is displayed when MERVA ESA terminated abnormally and no large message cluster reorganization was done before MERVA ESA restart.

LMC processing is done in insert mode.

LMC not closed since date/time, LMC is full
This text is displayed when MERVA ESA terminated abnormally and no large message cluster reorganization was done before MERVA ESA restart. Also, the LMC is considered full, that is, the maximum number of put retries has been reached for inserting a record.

Blank line
This appears when none of the above conditions apply.

Displaying the Large Message Cluster Statistics (DLMCT)

Use the **dlmct** command to monitor statistics of the large message cluster (LMC), in order to tune large message processing and thereby improve LMC input/output (I/O) performance and space usage.

For information on how to interpret the information displayed refer to the *MERVA for ESA Installation Guide*.

Command Format

The format of the **dlmct** command is:

dlmct	

The **dlmct** command has no parameters.

Example of the Display from a DLMCT Command

Figure 7 shows an example of the information displayed when you enter the **dlmct** command.

```
                        Operator Command Processing

>DLMCT
 DSL410I Statistics for Large Message Cluster - Tuning

 Current average message length           105.162 Bytes
 Maximum record length                    114.000 Bytes
 Control interval size                     16.384 Bytes
 Avg. number of segments per message         1,47
 Number of allocated extents                    1
 Processing mode                           insert
 Space used                                    71 %
 Reorganization recommended, processing continues




 142442   is the time of this display

Command =====>
PF 1=Help     2=Repeat    3=Return     4=DF     5=DU          6=DM Last
PF 7=         8=          9=Hardcopy  10=DP    11=DQ filled  12=DL
```

Figure 7. Displaying Statistics of Large Message Cluster for Tuning

When you enter the **dlmct** command the following information is displayed:

Current average message length
: The current average message length of all large messages in the LMC at the time the **dlmct** command was entered.

Maximum record length
: The maximum logical record length defined for the LMC.

Control interval size
: The control interval size specified for the LMC.

Avg. number of segments per message
The ratio of number of records to number of large messages. This field contains the average number of records (or segments) per large message, which is an indication of the degree of segmentation.

A '1' in this field means no segmentation.

Number of allocated extents
The number of extents allocated for the large message cluster at the time the **dlmct** command was entered.

Processing mode
The text **load** means records are loaded with a growing key into the LMC, which is a VSAM KSDS cluster.

The text **insert** means records are inserted into the LMC with a randomly-selected key.

Space used
The percentage calculated from stored bytes divided by high allocated relative byte address (RBA).

Reorganization recommended, processing continues
This text is displayed when the LMC is processed in insert mode and the LMC is more than 70% filled. (Space used > 70%.)

Reorganization recommended, LMC is full
This text is displayed when the LMC is processed in insert mode and the LMC is considered full, that is, the maximum number of put retries has been reached for inserting a record.

LMC not closed since date/time
This text is displayed when MERVA ESA terminated abnormally and no large message cluster reorganization was done before MERVA ESA restart.

LMC processing will be done in insert mode.

LMC not closed since date/time, LMC is full
This text is displayed when MERVA ESA terminated abnormally and no large message cluster reorganization was done before MERVA ESA restart. Also, the LMC is considered full, that is, the maximum number of put retries has been reached for inserting a record.

Blank line
This appears when none of the above conditions apply.

Displaying Unsolicited Messages (DM)

Use the **dm** (display messages) command to display messages that are not solicited by the operator. These messages are written to the operating system console and to the MERVA ESA Journal.

Command Format

The format of the **dm** command is:

dm	[{ *prefix* }] [{ *date* [,*prefix*] }] [{ *date* ,*time* [,*prefix*]}] [{ FIRST [,*prefix*]}] [{ LAST [,*prefix*]}]

When you enter the **dm** command without parameters, the first 14 messages are displayed. If you enter the FIRST, *date* or *time* parameter, 14 messages are displayed according to the specified parameter. If you then enter the **dm** command again without a parameter, the display continues with the next 14 messages.

Parameter Descriptions

The parameters for this command have the following meanings:

prefix
A 1- to 8-character prefix. Only messages starting with the prefix appear in the display.

date
The date in the format *yymmmdd*, where:

yy Are the lower two digits of the year, for example, 00 to indicate the year 2000.

mmm Is the month. It must be one of the following: JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, or DEC.

dd Is the day.

The display starts with the first message for that specified date. If this date is not found, no messages are displayed.

Note: It is possible to enter invalid dates (for example, 99FEB31) as a parameter for this command. This date will not be found.

time
The time of day in the format *hhmmss*, where:

hh Is the hour

mm Is the minute

ss Is the second.

You can specify the *time* parameter without seconds or minutes, such as *hhmms*, *hhmm*, *hhm*, or *hh*. If you enter a time of 121, all messages with a time indication of 12:10:00 or later are displayed. The display starts with the message containing this time or the next available time.

FIRST
The display starts with the first available message and displays up to the next 13 messages.

LAST
The last message and up to 13 preceding messages are displayed.

Command Examples

The following section shows some examples of how to enter the **dm** command.

Example 1: Use the **dm** command to display the first messages issued on or after the specified date and time:

```
dm,99jul01,1030
```

This displays the first 14 messages created on the first of July 1999 at 10.30 a.m. or later.

Example 2: Use the **dm** command to display the first messages issued at, or after, the specified time. The commas show the omission of the *date* parameter:

```
dm,,1030
```

This displays the first 14 messages created today at 10.30 a.m. or later.

Example 3: Use the FIRST parameter to display the first 14 messages that are available:

```
dm,first
```

Example 4: Use the prefix parameter to limit the type of messages displayed. The following command shows the last 14 messages issued by the SWIFT Link modules:

```
dm,last,dws
```

Example of the Display from a DM Command

Displaying Unsolicited Messages (DM) shows an example of the information displayed when you enter the **dm** command.

DM Command

```
                        Operator Command Processing

> DM
  DSL075I Display Message 1999JUN07
  113845 DSL045I Journal data set A open, 1820 records, 38% used in 1 ex
  113846 DSL360I Status of QDS DSLQDS   is NORMAL   at 19990607/102651
  113846 DSL151I Last UMR is MERVAESA 00012345 at 19990607/102517
  113847 DSL060I RTCOMM 1 start successful
  113847 DSL060I CONSOLE 2 start successful
  113848 DSL060I BATCH 3 start successful
  113848 DSL060I TRANSACT 4 start successful
  113848 DSL060I MSGCOUNT 5 start successful
  113848 DSL060I SWIFTAUT 6 start successful
  113849 DSL060I SWIFTII 7 start successful
  113849 DSL060I SWLOADSK 8 start successful
  113849 ENL900I Telex Link startup successful
  113849 DSL060I TELEX 9 start successful
  113849 DSL000A MERVA is ready

  114232   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 8. Displaying Unsolicited Operator Messages

Displaying the Nucleus Servers (DNS)

Use the **dns** (display nucleus servers) command to monitor the status either of one particular nucleus server or of all nucleus servers defined in the DSLNSVT.

Command Format

The format of the **dns** command is:

dns	[{ TASK | ALL | FIRST | *servername* }]

When you enter the **dns** command without parameters, the status of the first 26 nucleus servers running as subtasks is displayed. If you enter the same command again, the status of the next 26 nucleus servers is displayed.

Parameter Descriptions

The parameters for this command have the following meanings:

none
Displays the next 26 nucleus servers of the group displayed before. If there was no previous display, this command is equivalent to a **dns task** command.

FIRST
Displays, in two columns, the status of the first 26 nucleus servers of the group displayed before. If there was no previous display, this command is equivalent to a **dns task** command.

TASK
Displays, in two columns, the status of the nucleus servers defined in DSLNSVT, which are running as subtasks.

ALL
Displays, in two columns, the status of all nucleus servers defined in MERVA ESA. These nucleus servers are defined in the DSLNSVT, or in one of the other nucleus tables DSLNPTT, DSLNTRT, or DSLNCMT. The nucleus servers can run under direct control of DSLNUC or running as subtasks.

servername
The name of the nucleus server (up to 8 characters) of which the status is to be displayed. The name of this nucleus server must be defined in DSLNSVT, or in one of the other tables DSLNPTT, DSLNTRT, or DSLNCMT.

Command Examples

This section shows some examples of how to enter the **dns** command.

Example 1: Use the TASK parameter to display the status of the nucleus servers running as subtasks:

```
dns,task
```

Example 2: You can also specify a nucleus server to display the status of this specific server:

```
dns SWIFTII
```

Example of the Display from a DNS Command

DNS Command

"Displaying the Nucleus Servers (DNS)" on page 33 shows an example of the information displayed when you enter the **dns** command.

```
                        Operator Command Processing

> DNS ALL
  DSL160I Display Nucleus Servers
   Server   Type     Status           Server   Type     Status
   DSLNUC   NUCT                     DSLNCMD  SUBR NCM   DSLNUC
   DSLNMOP  NUCT NTR                 CONSOLE  SUBR NPT   DSLNMOP
   DSLNDM   SUBR NCM   DSLNMOP       APPCSRV1 MVS  NPT I
   RTCOMM   NUCT NPT A               BATCH    NUCT NPT A
   TRANSACT NUCT NPT A               MSGCOUNT NUCT NPT A 0296.322
   CICSSRV  NUCT NPT A 0000.000      SWIFTAUT NUCT NPT A
   SWIFTII  NUCT NPT A 5695.566      SWLOADSK NUCT NPT A 5695.566
   TELEX    NUCT NPT I               DSLQMGT  CICS NTR   00000432
   DSLJRNP  CICS NTR   00000436      DSLNCS   NUCT NTR
   DSLNUSR  NUCT NTR                 DSLNRTCP NUCT NTR
   DWSAUTP  NUCT NTR                 DSLNSHU  NUCT NCM
   DSLNDU   NUCT NCM                 DSLQMGTR NUCT NCM
   DSLRTRSW NUCT NCM                 DSLQLRGC NUCT NCM

  122412   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 9. Displaying the Nucleus Server Status

Message DSL160I shows the following information, for details refer to *MERVA for ESA Messages and Codes*.

Server The name of the nucleus server as defined in the DSLNSVT. If a nucleus server is defined in the DSLNPTT only, it is the descriptive name defined by DSLNPT. If a nucleus server is defined in the DSLNTRT only, it is the name of the central service defined by DSLNTR. If a nucleus server is defined in the DSLNCMT only, it is the name of the command execution routine.

Type The nucleus server type consists of two components. The first item indicates the mode in which the program is executing:

- **CICS** The nucleus server runs as a CICS task, the transaction code is defined in the DSLNSVT.
- **MVS** The nucleus server runs as an MVS task.
- **NUCT** The nucleus server runs under direct control of DSLNUC.
- **SUBR** The program runs as a subroutine of a main nucleus server.

The second item indicates the type of the program:

- **NCM** The nucleus server is a command execution routine.
- **NPT** The nucleus server is a nucleus (NPT) program.
- **NTR** The nucleus server is a central service.

Status The status information is dependent on the type of program. The letter 'A' indicates an active NPT program. The letter 'I' indicates an inactive NPT program. The letter 'W' indicates a nucleus server waiting on some other service. For a CICS task, the task number is displayed for diagnosis

purposes. For active NPT programs running under direct control of DSLNUC, the elapsed time in seconds since the last scheduling is displayed.

Displaying the Program Status (DP)

Use the **dp** (display program) command to monitor the status either of one particular program or of all programs defined in DSLNPTT.

Command Format

The format of the **dp** command is:

dp	[{ FIRST | *progname* | *pid* }]

When you enter the **dp** command without parameters, the status of the first 26 programs is displayed. If you enter the same command again, the status of the next 26 programs is displayed.

Parameter Descriptions

The parameters for this command have the following meanings:

FIRST
Displays, in two columns, the status of the first 26 programs defined in DSLNPTT.

progname
The descriptive name of the program (up to 8 characters) of which the status is to be displayed. The name of this program must be defined in DSLNPTT.

pid
The program identification (up to 3 characters) of the program. This program identification is generated in DSLNPTT, and is used instead of the program name.

Command Examples

This section shows some examples of how to enter the **dp** command.

Example 1: Use the FIRST parameter to display the status of the first 26 programs defined in DSLNPTT:

```
dp,first
```

Example 2: You can also specify the program identification (here: 3) to display the status of a program:

```
dp,3
```

Example of the Display from a DP Command

"Displaying the Program Status (DP)" shows an example of the information displayed when you enter the **dp** command.

```
                      Operator Command Processing

> DP
  DSL102I Display Programs
   PROGNAME PID Y S P A STATUS LRC    PROGNAME PID Y S P A STATUS LRC
   RTCOMM   1   9 N N Y ACTIVE  00    CONSOLE  2   9 Y Y Y ACTIVE  00
   BATCH    3   4 N N Y ACTIVE  00    TRANSACT 4   5 N N Y ACTIVE  00
   MSGCOUNT 5   6 N N Y ACTIVE  00    APPCSRV1 6   5 Y Y N INACTV  00
   CICSSRV  7   5 Y Y Y ACTIVE  00    SWIFTAUT 8   5 Y Y Y ACTIVE  00
   SWIFTII  9   8 Y Y Y ACTIVE  00    SWLOADSK 10  2 Y Y Y ACTIVE  00
   TELEX    11  7 Y Y N INACTV  00









  115235   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 10. Displaying the Program Status

Message DSL102I shows the following information, for details refer to *MERVA for ESA Messages and Codes*.

PROGNAME
The descriptive name of the program in DSLNPTT.

PID The program identification contained in DSLNPTT.

Y The priority of the program with a value of A (lowest) to Z and 0 to 9 (highest).

S Shows whether the **start** command is allowed for this program (Y), or not (N).

P Shows whether the **stop** command is allowed for this program (Y), or not (N).

A Shows whether the program is started automatically during the MERVA ESA startup (Y), or not (N).

STATUS
Shows the status of the program as described in *MERVA for ESA Messages and Codes*.

LRC The last return code issued by the program when called for a **start** or **stop** command, or after one of its event control blocks (ECBs) was posted.

Displaying the Queue Status (DQ)

Use the **dq** command to display the status of MERVA ESA queues.

Command Format

The format of the **dq** command is:

dq	[*function*],[FIRST][,FILLED] STATUS SQLERROR

The first time you enter the **dq** command without parameters, the status is displayed for the first 26 queues that are associated with message-processing functions. If you enter the same command again, the next 26 queues are displayed.

Parameter Descriptions

The parameters for this command have the following meanings:

function
Is the name, or part of the name, of a message-processing function defined in the MERVA ESA Function Table. There are two methods for entering part of a function name:

- You can enter 1 to 7 characters without using the substitution characters asterisk (*) or percent (%). This method displays the status of the first 26 function queues whose names begin with these characters.
- You can enter 1 to 8 characters including the substitution characters asterisk (*) and percent (%).
  - The substitution character * represents any number of characters. As many asterisks as required can appear anywhere in the function name.
  - The substitution character % is a place-holding character, representing any single character. As many percent symbols as are necessary may appear anywhere in a function name.

If you enter the same command again, and there are more than 26 function queues matching these characters, the next 26 function queues are displayed. If you want to see the first 26 function queues again, you must enter the **dq** command with the function parameter followed by the FIRST parameter. You can also use the FILLED parameter to display only those queues matching these characters and containing messages.

Note: The FIRST and FILLED parameters can be used in any order. However, if the *function* parameter is used, it must be the first in the sequence.

FIRST
Displays the status of the first 26 queues. You can enter the parameter FILLED as a second parameter if you want to display the status of the first 26 queues that contain messages.

FILLED
Displays the status of the queues that contain messages. However, queues that have been defined with DQFILL=NO in DSLFNTT are not displayed. Such queues can be displayed with a **dq** command without a parameter or **dq** *function* command.

You can enter the parameter FIRST as a second parameter if you want to display the status of the first 26 queues that contain messages.

STATUS
The status of the MERVA ESA queue data set is displayed. The response shows the number of available and used entries in the Queue Key Table, the number of DSLQDS system and data blocks, and how they are used. Furthermore, it shows the number of large messages and the last UMR.

If MERVA ESA is customized for queue management using DB2, the response shows the number of messages, the last message table number, the I/O module used, and the last UMR.

SQLERROR
If MERVA ESA is customized for queue management using DB2, error information for the last SQL error is displayed.

Command Examples

The following section shows some examples of how to enter the **dq** command.

Example 1: You can use the FIRST parameter to display the status of the first 26 queues associated with message-processing functions:

```
dq,first
```

Example 2: To display the status of a certain function you can use the *function* parameter so that only the status of the specified function is displayed:

```
dq,l1ai0
```

Example 3: Enter the following command to display the first 26 queues whose names start with the characters L1A:

```
dq,l1a
```

If you enter this command again, the next 26 queues whose names start with L1A are displayed. If you wish to display again the first 26 queues whose names start with the characters L1A, you enter the following command:

```
dq,l1a,first
```

Example 4: You can use the substitution character * to display the first 26 queues whose names contain the characters “sw” anywhere in the name:

```
dq,*sw*
```

Example 5: You can use the substitution character % to display the first 26 queues whose names contain the characters “pr” in positions 3 and 4 of the function name:

```
dq,%%pr*
```

Example 6: Use the following command to display the status of the queue data set:

```
dq,status
```

Example of the Display from a DQ Command

“Displaying the Queue Status (DQ)” on page 38 shows an example of the display resulting from a **dq** command without parameters. Message DSL143I shows the information for up to 26 queues.

```
                      Operator Command Processing

> DQ
  DSL143I Display Queues
   Function    K USR WAIT  THRSH      Function    K USR WAIT  THRSH
   DMPR0     H 0 00 000036 00020 T    DMPR1     H 0 00 000000 00020
   DMPR2     H 0 00 000000 00020      DMPR3     H 0 00 000000 00020
   DMPR4     H 0 00 000000 00020      DMPR89    H 0 00 000098 00020 T
   DMSDI       0 00 000000 00000      DMSO0       0 00 000000 00000
   DMSO1       0 00 000000 00000      DMERR       0 00 000000 00000
   DMSY0       0 00 000000 00000      DMSY1       0 00 000000 00000
   DMSY2       0 00 000000 00000      DMSY3       0 00 000000 00000
   DMSY4       0 00 000000 00000      DUMMY       0 00 000000 00000
   DMTST       3 00 000000 00030      L1DE0       1 00 000017 00050
   L1AI0       1 00 000049 00050      L1VE0       3 00 000000 00020
   L1PR0     N 0 00 000000 00100      L1SWU       0 00 000004 00030
   L1SWN       0 00 000000 00030      L1ACK       3 00 000023 00100
   L1ERROR     0 00 000000 00100      L1FREE      0 00 000000 00100

  115237   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 11. Displaying the Queue Status

Message DSL143I is described in *MERVA for ESA Messages and Codes*. Here is a short explanation of the queue-status displayed:

Function
The name of the MERVA ESA queue. The column after the queue name shows the status of the MERVA ESA function queue:

H Hold

N Nohold

A Activated

***** NOTIFY=YES is specified in the function-table entry.

Otherwise it is left blank.

K The keys that are defined for this queue. As an operator, you do not need this information, but, if you are interested, the queue keys are explained in the book *MERVA for ESA Messages and Codes*.

USR The number of users who are currently processing this queue.

WAIT The number of messages stored in this queue.

THRSH
The threshold number of messages defined in the MERVA ESA function-table entry. "T" indicates that the threshold number has been reached. If the threshold number is exceeded, you should notify the user responsible for that queue.

Figure 12 on page 41 shows an example of the information that results when you enter the STATUS parameter of the **dq** command. Message DSL146I shows the status of the queue data set.

```
                      Operator Command Processing

> DQ,STATUS
  DSL146I Queue status
          Key tables: Maximum= 00003000
                      Used   = 00002999
          QDS blocks: System = 000007
                      Data   = 000323  (43% used)
          # of large messages= 00000052
          Last UMR  : ID     = MERVAESA
                      Number = 00005876
                      Date   = 19990419
                      Time   = 192311
          I/O Module -Method = DSLQMCNV -VSB
          Active Data Sets   = QDS1


   115333   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 12. Displaying the Status of the Queue Data Set

Message DSL146I is explained in *MERVA for ESA Messages and Codes*. The above example shows that, with 2999 used of 3000 available entries in the Queue Key Table, the queue data set is only filled by 43 percent. As there are no free entries in the Queue Key Table, the remaining space in the queue data set cannot be used. Therefore the number of entries in the Queue Key Table should be increased to 6000 or 7000. This can be done by your system programmer in the MERVA ESA customizing parameter module DSLPRM by the NQE parameter of the DSLPARM macro.

Displaying the Queues Sorted (DQSORTED)

Use the **dqsorted** command to display the status of MERVA ESA queues sorted according to the number of messages. The queues with the highest number of messages are shown first. Only filled queues are shown. The command may be abbreviated to **dqso**. The display is very similar to the display of the **dq** command. The numbers are shown without leading zeros to improve the readability.

Command Format

The format of the **dqsorted** command is:

dqsorted	[*function*][,FIRST]

The first time you enter the **dqsorted** command without parameters, the status is displayed for the first 26 queues with the highest message count. If you enter the same command again, the next 26 queues are displayed.

Parameter Description

The parameters for this command have the following meanings:

function
This parameter is identical to the respective parameter of the **dq** command.

FIRST
Displays the status of the first 26 queues, the queues with the highest message count.

Command Examples

The following section shows an example of how to enter the **dqsorted** command.

Example 1

You can use the FIRST parameter to display the status of the 26 queues with the highest message count associated with message processing functions:

```
dqso first
```

Example 2

Enter the following command to display up to 26 queues whose names start with the characters L1A and sorted according to the message count:

```
dqso l1a
```

If you enter this command again, the next 26 queues whose names start with L1A are displayed. If you wish to display again the first 26 queues whose names start with the characters L1A, you enter the following command:

```
dqso l1a,first
```

Example of the Display from a DQSORTED Command

Figure 13 on page 43 shows an example of the display resulting from a **dqsorted** command without parameters. Message DSL143I shows the information for up to 26 queues.

```
                      Operator Command Processing

> DQSO
  DSL143I Display Queues
   Function    K USR WAIT  THRSH      Function    K USR WAIT  THRSH
   L1SDO               432   100 T    L1PR1     N         142   100 T
   L3PR0     N         124   100 T    L2PR1     N         113   100 T
   L2VE0       3       108    20 T    L1ACK       3       100   100 T
   L2DE0       1       100    50 T    L2ACK       3       100   100 T
   L3ACKF      3       100   100 T    SLPT1ACK    3       100
   SLPT1SDO            100            SLPT2ACK    3       100
   SLPT2SDO            100            L1DO0       1        51   100
   L1PR0     N          50   100      L2SDO                10   100
   L3PR1     N          10   100      SLPT1SYS             10
   L2PR0     N           7   100      L3VE0       3         5    20
   SLPT2SYS              4            L1VE0       3         2    20
   L2DO0       3         1   100      L3DO0       3         1   100
   LRGDE0      3         1

  150312   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 13. Displaying the Queue Status

Message DSL143I is described in *MERVA for ESA Messages and Codes*. A short explanation can be found in the chapter for the **dq** command.

Displaying the User Status (DU)

Use the **du** command to display the status of a single user or all active MERVA ESA users who are signed on.

Command Format

The format of the **du** command is:

du	[{ FIRST | *userid* [,FIRST]}]

If you enter the **du** command without parameters, the status of the first 26 users is displayed. If you enter the command a second time without parameters, the status of the next 26 users is displayed.

Parameter Descriptions

The parameters for this command have the following meanings:

FIRST
Displays the sign-on status of the first 26 users.

userid
The name, or part of a name, of a user. The two methods for entering part of a user name are:

- You can enter 1 to 7 characters without using the substitution characters asterisk (*) or percent (%). This method displays the first 26 user identifications beginning with these characters.
- You can enter 1 to 8 characters including the substitution characters asterisk (*) and percent (%).
  - The substitution character * represents any number of characters. As many asterisks as are required can appear anywhere in the user identification name.
  - The substitution character % is a place-holding character, representing any single character. As many percent symbols as are necessary may appear anywhere in the user identification name.

  If you enter the same command again and there are more than 26 users matching these characters, the next 26 users are displayed. If you want to see the first 26 users again, you must enter the **du** command with the *userid* followed by the FIRST parameter.

If you want to display the status of a single user, you must enter the complete user identification, or enough characters to make the user identification unique.

Command Examples

This section shows some examples of how to enter the **du** command.

Example 1: Use the **du** command without any parameters to display the status of the first 26 users. If you repeat this command, the next 26 users are displayed:

```
du
```

Example 2: Enter the following command to display the status of users with user identifications that start with the same characters, such as MIL:

```
du,mil
```

Example 3: Use the following command to redisplay the status of the first 26 users, if you have already used the **du** command without parameters more than once:

```
du,first
```

Example 4: You can use the substitution character * to display the first 26 users whose names contain the characters "ab" anywhere in the name:

```
du,*ab*
```

Example 5: You can use the substitution character % to display the first 26 users whose names contain the characters "cd" in positions 3 and 4 of the user name:

```
du,%%cd*
```

Example of the Display from a DU Command

"Displaying the User Status (DU)" on page 44 shows the information that is displayed when you enter the **du** command. Message DSL132I is explained in *MERVA for ESA Messages and Codes*.

```
                      Operator Command Processing

 > DU
   DSL132I Display Users
      User-ID Function Origin-ID         User-ID Function Origin-ID
      MAS     CMD      TIBMBEAAAX00      USER1   L1DE0    TIBMBEAAAX00
      USER2   L1VE0    TIBMBEAAAX00      USER3   L1AI0    TIBMBEAAAX02
      USER4   L1VE9    TIBMBEAAAX01      USER5   L1AK0    TIBMBEAAAX00
      USER6   L1DA0    TIBMBEAAAX00      USER7   L1AO0    TIBMBEAAAX02









   115431   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 14. Displaying the User Status

The message in "Displaying the User Status (DU)" on page 44 contains user-status information under the following headings:

User-ID The identification of the user who is signed on

Function The name of the function being used

Origin-ID The origin identifier of the user.

Forcing Off Users (FORCE)

Use the **force** command to remove a user from the system (that is, to force a sign-off). The **force** command should only be used when the task of the user has ended abnormally, without signing him off. Using the **force** command allows the user to sign on again.

If this user was processing a MERVA ESA User File record, other users cannot access this record. The **force** command also allows this User File record to be accessed by other users or by the forced user.

Notes:

1. You must be authorized to use the **force** command.
2. When the **force** command is used on an active user, the user is signed off only when returning to the Function Selection menu.

Command Format

The format of the **force** command is:

force	*userid*

Parameter Description

userid
Is the user identification of the user who is to be forced off.

Command Example

To force off a user with the user identification of TEST06 enter the following command:

```
force,test06
```

Example of the Display from a FORCE Command

Figure 15 on page 47 shows the display resulting from the command shown in the command example. Message DSL133I returns the information resulting from the execution of this command.

```
                        Operator Command Processing

> FORCE,TEST06
  DSL133I TEST06 has been forced









  115135   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 15. Forcing a User Off

Setting Functions to HOLD (HF)

Some message-processing functions can be associated with a transaction. The transaction processes the messages written to the queue of the message-processing function.

Use the **hf** command to set a message-processing function to HOLD status or to stop an active transaction after completion of the message currently being processed. HOLD status means that the associated transaction is *not* started automatically when a message is written to the function queue. This status can be reset with the **sf** command. If related functions are defined, they are all set to HOLD status when the **hf** command is used for one of them.

Note: You must be authorized to enter the **hf** command for a function which is defined in the MERVA ESA function table DSLFNTT using the parameter MQI=YES.

Command Format

The format of the **hf** command is:

hf	{*function* | ALL}

Parameter Description

The parameters for this command have the following meanings:

function
The name of the message-processing function that you want to set to HOLD status.

ALL
Specifies that you want to set all queues that are associated with a transaction to HOLD status.

Note: You must be authorized to use the **hf** command with the ALL parameter.

Command Example

This example shows the command you use to set function L1PR0 to HOLD status:

```
hf,l1pr0
```

Example of the Display from an HF Command

Figure 16 on page 49 shows the display that would result if you entered the command shown in the command example. Message DSL115I confirms that the function is in HOLD status, for details refer to *MERVA for ESA Messages and Codes*.

```
                        Operator Command Processing

> HF,L1PR0
  DSL115I L1PR0    HELD   , TRAN=DSLH    , LT1=L84A    , LT2=









  115345   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 16. Setting a Function to HOLD Status

This message shows the following information:

- The name of the function: L1PR0
- The confirmation that the function is in HOLD status: HELD

  A transaction associated with this function is not started automatically when a message is written to the queue of this message-processing function. Users cannot process the queue; messages cannot be retrieved from a held queue.
- The name of the transaction associated with the function: DSLH
- The logical terminal name defined for the function: L84A.

Transaction codes and logical terminal names have 4 characters in CICS and up to 8 characters in IMS.

Note: If the associated transaction is running, the HOLD status will take effect when the program requests the next message from the queue.

Set the Journal Switch Status (JSET)

Use the **jset** command to display the current switch status of the journal or to modify the current switch status. There are three possible values for the switch status:

ONCE The switch is performed only once. The journal data set A is switched to B when A becomes full. This is the method used in previous MERVA ESA versions.

MANUAL
The switch is performed only when the **jswitch** command is issued by an operator. No automatic switching is performed; when the current journal data set becomes full MERVA ESA cannot continue processing.

CYCLE
The switch is performed automatically each time a journal data set becomes unavailable, for example, because it is full.

Note:

You must be authorized to use the **jswitch** command to modify the journal switch status.

Command Format

The format of the **jset** command is:

jset	[{ ONCE | MANUAL | CYCLE }]

Parameter Description

ONCE
Set the journal switch status to ONCE.

MANUAL
Set the journal switch status to MANUAL.

CYCLE
Set the journal switch status to CYCLE.

Example of the Display from a JSET Command

Figure 17 on page 51 shows the display resulting from the command **jset**. Messages DSL055I or DSL056I return the information resulting from the execution of this command.

```
                        Operator Command Processing

> JSET MANUAL
   DSL056I Journal switch status is changed from CYCLE to MANUAL

  173648   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 17. Setting the Journal Switch Status

The message DSL056I shows the previous journal switch status and the new switch status. Entering the command without parameter shows the message DSL055I with the current journal switch status.

Display the Status of the Journal Data Sets (JSTAT)

Use the **jstat** command to display the current status of the journal data set and a list of the journal switch activities which occurred since MERVA ESA was started. The list is limited to the most recent 12 entries.

Command Format

The format of the **jstat** command is:

jstat	

Example of the Display from a JSTAT Command

Figure 18 shows the display resulting from the command **jstat**. Message DSL045I returns the information resulting from the execution of this command.

```
                        Operator Command Processing

 > JSTAT
   DSL045I Journal data set A open, 19669 records, 14% used in 7 extent(s

    Open Time and Status  Added    Fill     Journal Switch Log
A>  19990303 170309 used       1  14% /007
B>  19990303 151409 used    1407   4% /001  SWITCH MAS      VNDEBET2
A>  19990303 145944 used      20  14% /007  SWITCH MAS      VNDEBET2









   173620   is the time of this display

 Command =====>
 PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
 PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 18. Switching the Journal Data Set

The message DSL045I shows the status of the currently active journal data set. This is followed by a table showing one row for each journal switch event. The first character in each row, A or B, indicates the journal data set.

The open time gives the date and time when the journal data set was opened or switched to. The format of the timestamp is YYYYMMDD HHMMDD. The status can be either ‘used’ or ‘reset’. The indication ‘used’ means that records were found in the data set when it was opened; new records are added after the existing records. The ‘Added’ column indicates the number of journal records added since the data set was opened. The ‘Fill’ status shows the same information as in the message DSL045I. ‘Journal Switch Log’ shows the reason for the switching; normally either an operator entered the **jswitch** command as in the example above, or the switch happens just because the first data set is full.

Switch the Journal Data Sets (JSWITCH)

Use the **jswitch** command to switch from using the journal data set A to journal data set B or vice versa. After the **jswitch** command has been used, the records in the first data set can be processed by a utility program, for example REPRO to copy the records to an archive file. After the records have been saved, the data set can be reset to an initial state (empty) and the **jswitch** command can be used to switch back to the first data set.

The **jswitch** allows to specify **reset** to indicate that the new data set is cleared before being used. This **reset** parameter works only when the VSAM cluster is defined with the option REUSE.

Be aware that having specified the REUSE option on the cluster definition, there is a risk of losing the journal records. If the data set is reset by the operator without a preceding backup, the records in the data set are lost.

Note: You must be authorized to use the **jswitch** command.

Command Format

The format of the **jswitch** command is:

jswitch	[TO] [{A | B }] [AND] [RESET]

Parameter Description

TO
For documentation purposes only.

A or B
The journal data set which is to be used next. If this data set is already the active data set, the switch is not performed.

AND
For documentation purposes only.

RESET
The journal data set which will be used next is to be reset to an initial state. This works only when the cluster definition used the option REUSE. When the cluster was not defined with the option REUSE, the parameter RESET is ignored. When RESET is used, the parameter A or B must be specified also for security reasons.

Command Examples

Example 1

To switch the journal from the current data set to the other:

```
jswitch
```

Example 2

To switch the journal from using data set B to using data set A and reset data set A to the initial (empty) state. It is assumed that the current journal data set is B:

```
jswitch A reset
```

Example of the Display from a JSWITCH Command

Figure 19 shows the display resulting from the command shown in the command example 1. Message DSL045I returns the information resulting from the execution of this command.

```
                        Operator Command Processing

> JSWITCH
  DSL045I Journal data set B open, 1409 records, 4% used in 1 extent(s)









  173620   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 19. Switching the Journal Data Set

Setting Priorities for Programs Defined in DSLNPTT (PRIORITY)

Use the **priority** command to set the priority of a program that has been defined in DSLNPTT. As the **priority** command shows only an acceptance or rejection message, use the **dp** (Display Program) command to see the new priorities of the programs.

Only programs running under direct control of DSLNUC are affected by the priority settings. The priority of NPT programs running as subtasks is determined by the specification in the nucleus server table entry DSLNSV.

Note: You must be authorized to use the **priority** command.

Command Format

The format of the **priority** command is:

priority **y**	{ *progname1,priority1, ... ,progname8,priority8* } { *pid1,priority1, ... ,pid8,priority8* }

Parameter Descriptions

The parameters for this command have the following meanings:

progname1 **-** *progname8*
The descriptive names of up to eight programs that are defined in the DSLNPTT. The **priority** command can be used for any of the defined programs whether they are running or stopped.

pid1 **-** *pid8*
Instead of the program name, the program identification, generated by DSLNPTT, can be used. It is a 1- to 3-digit identifier.

priority1 **-** *priority8*
The 1-character priority value. The priority value must be in the range A to Z, or 0 to 9:

A Is the lowest priority

9 Is the highest priority.

If the same priority is defined for several programs, the dispatching order within this priority is determined in a way that all programs get an equal amount of service. A program in the group is not called again until all the programs in the group that need to be called have been called.

Note: The parameters *progname* and *priority*, or *pid* and *priority*, must always be specified in pairs. The priority of up to eight programs can be set with one **priority** command.

Note: The dynamic dispatching group DYN does no longer exist in MERVA ESA. This concept is not needed, because the scheduling order for all NPT programs of equal priority works the same way as the dynamic dispatching group worked in the past. If a program was serviced once, it is serviced a second time only if no other program of the group needs to be serviced. The order in which the programs are defined in the DSLNPTT is irrelevant. The number of priority values has been increased from 10 to 36 to allow for more granularity in defining the priorities of programs.

Command Examples

The following section shows some examples of how to enter the **priority** command.

Example 1: You can enter the **priority** command with three sets of *progname* and *priority* parameters:

```
priority,batch,8,transact,7,cicssrv,5
```

Example 2: This example is the same as example 1, but the abbreviation for the **priority** command is used:

```
y,batch,8,transact,7,cicssrv,5
```

Example of a Display from a PRIORITY Command

Figure 20 shows an example of the **priority** command. Message DSL070I confirms that the command has successfully completed.

```
                      Operator Command Processing

> PRIORITY CONSOLE 9 BATCH 8 TRANSACT 5
  DSL070I Priority values accepted









  115133   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 20. Setting the Priorities of Programs

Switching the Queue Trace (QSWITCH)

Use the **qswitch** command to define whether or not the Queue trace is written to the MERVA ESA Journal. You use it to change or query the mode of the queue trace facility in general or for up to 20 individual queues. *MERVA for ESA Concepts and Components* contains an explanation of the queue-trace entries.

Each response to a correct **qswitch** command shows always the states of the general queue trace and of the individual queues independent of the parameters used.

Notes:

1. You must be authorized to use the **qswitch** command.
2. You should only use this command when your system administrator has instructed you to do so.

Command Format

The format of the **qswitch** command is:

qswitch **qw**	[{*state*}] [{*queue-name*[,*state*]}] [{[*state*],*queue-name*}]

Parameter Descriptions

If you enter the **qswitch** command without parameters, the actual setting is displayed.

The parameters for this command have the following meanings:

state
Is the state of the queue trace. If the *queue-name* parameter is not specified, the state applies to the general queue trace. *state* can have the following values:

LARGE
The large queue trace is active, that is, the queue parameter list and the complete queue element are traced. You can abbreviate this parameter to L.

If MERVA ESA is customized for queue management using DB2, the queue parameter list, DB2 return information, queue descriptors, and the message are traced.

SMALL
The small queue trace is active, that is, the queue parameter list and the queue element prefix are traced. You can abbreviate this parameter to S.

If MERVA ESA is customized for queue management using DB2, only the queue parameter list is traced.

OFF
The queue trace is not active. You can abbreviate this parameter to O.

queue-name
Is the name of a queue for which you want to define an individual queue trace state. Any 1 to 8 characters that are different from the state parameters and that are a queue or dummy queue defined in the MERVA ESA function table are considered to be the name of a queue.

If you do not specify the *state* parameter with the *queue-name* parameter, the state LARGE is assumed as default.

If you specify the state OFF for a queue, an individual state is not used for this queue anymore.

If the general queue trace and an individual queue have a different trace state, the higher state is taken, LARGE being the highest state and SMALL being the lowest state.

If, after a queue management multiple put or route operation, a queue element is written to more than one and up to 12 queues, and one or more of these queues have an individual queue trace state, also the highest state is taken.

Command Examples

This section shows some examples of how to enter the **qswitch** command.

Example 1: Enter the following command to request the large queue trace:

```
qswitch,large
```

Example 2: Use the SMALL parameter to not trace the data of the queue elements:

```
qw,small
```

Example 3: If you specify the command without any parameters the actual setting of the queue trace facility is displayed:

```
qw
```

Example 4: Specify the name of a queue to request that the queue facility traces all activities of this queue. The following formats of the command can be used:

```
qw,l1ve0
qw,l1ve0,large
qw,large,l1ve0
qw,,l1ve0
```

Example of the Display from a QSWITCH Command

Figure 21 on page 59 shows an example of the message returned when you enter the **qswitch,l2ve0,small** command.

```
                        Operator Command Processing

> QSWITCH,L2VE0,SMALL
  DSL038I Queue trace status is OFF
     Queue Name   Status                Queue Name   Status
     L1DE0        LARGE                 L1VE0        LARGE
     L2VE0        SMALL

  115033   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 21. Using the QSWITCH Command

After the general trace state, the following information is shown:

Queue Name
The name of the queue for which an individual trace state is displayed

Status
Indicates the queue trace state of the queue.

This example shows that the general queue trace state is OFF, and three queues have an individual queue trace state.

Resetting Shutdown (RESHUT)

The **reshut** command allows the reactivation of the user sessions and Hard-Copy Printer functions after **shutdown** has been entered. This command can be entered at any time *before* the **cancel** or **terminat** command is entered.

Note: You must be authorized to use the **reshut** command.

Command Format

The format of the **reshut** command is:

reshut **rs**	

The **reshut** command has no parameters.

Command Example

You can use the abbreviation **rs** instead of the **reshut** command:

```
rs
```

Example of the Display from a RESHUT Command

Figure 22 shows the display resulting from the command shown in the command example. Message DSL052I returns the information resulting from the execution of this command.

```
                        Operator Command Processing

> RS
  DSL052I SHUTDOWN reset, MERVA end users can sign on again
  115236   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 22. Resetting SHUTDOWN with RESHUT

Switching the Routing Trace (RSWITCH)

Use the **rswitch** command to define whether or not the routing trace is written to the MERVA ESA Journal. You use it to change or query the mode of the routing trace facility in general or for up to 20 individual routing tables. *MERVA for ESA Concepts and Components* contains an explanation of the routing-trace table entries.

Each response to a correct **rswitch** command shows always the states of the general routing trace and of the individual routing tables independent of the parameters used.

Notes:

1. You must be authorized to use the **rswitch** command.
2. You should only use this command when your system administrator has instructed you to do so.

Command Format

The format of the **rswitch** command is:

rswitch **rw**	[{*state* }] [{*rt-name*[,*state*]}] [{[*state*],*rt-name*}]

Parameter Descriptions

If you enter the **rswitch** command without parameters, the actual setting is displayed.

The parameters for this command have the following meanings:

state
: Is the state of the routing trace. If the *rt-name* parameter is not specified, the state applies to the general routing trace. *state* can have the following values:

  ALL
  : All routing activities based on a routing table are traced.

  WARNING
  : Only warnings and severe errors detected by the routing are traced. Successful routing operations are not traced. You can abbreviate this parameter to W.

  SEVERE
  : Only severe errors detected by the routing are traced. Routing operations that are completed successfully or with a warning are not traced. You can abbreviate this parameter to S.

  OFF
  : No routing activities are traced. You can abbreviate this parameter to O.

rt-name
: Is the name of a routing table for which you want to define an individual routing trace state. Any 1 to 8 characters that are different from the state parameters are considered to be the name of a routing table.

  If you do not specify the *state* parameter with the *rt-name* parameter, the state ALL is assumed as default.

If you specify the state OFF for a routing table, an individual state is not used for this routing table anymore.

If the general routing trace and an individual routing table have a different trace state, the higher state is taken, ALL being the highest state and SEVERE being the lowest state.

Command Examples

This section shows some examples of how to enter the **rswitch** command.

Example 1: Enter the following command to request that the routing facility traces all activities:

```
rswitch,all
```

Example 2: Use the SEVERE parameter to trace only severe routing errors:

```
rw,severe
```

Example 3: If you specify the command without any parameters the actual setting of the mode of the routing trace facility is displayed:

```
rw
```

Example 4: Specify the name of a routing table to request that the routing facility traces all activities of this routing table. The following formats of the command can be used:

```
rw,dwsl2out
rw,dwsl2out,all
rw,all,dwsl2out
rw,,dwsl2out
```

Example of the Display from a RSWITCH Command

Figure 23 on page 63 shows an example of the message returned when you enter the **rswitch,dwsl2out,severe** command.

```
                      Operator Command Processing

> RSWITCH,DWSL2OUT,S
  DSL030I Routing trace status is OFF
     R-Table  Location    Status        R-Table  Location    Status
     DWSL2AO0 DSLFNTT  OK ALL          DWSL2DO0 DSLFNTT  OK ALL
     DWSL2OUT DWSLTT   OK SEVERE









   115033   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 23. Confirming the RSWITCH Command

After the general trace state, the following information is shown:

R-Table
The name of the routing table for which an individual trace state is displayed

Location
The name of the MERVA ESA function table or the SWIFT Link logical terminal table where the routing table was found. This name is followed by the availability indicator:

OK
Indicates that the routing table could be loaded during the MERVA ESA or SWIFT Link initialization

NA
(Not available) indicates that the routing table could **not** be loaded during the MERVA ESA or SWIFT Link initialization.

If the routing table was not found, **NOTFOUND** is shown as location, and the availability indicator is not used. Even in this case, MERVA ESA considers the trace state when this routing table is used in a routing operation.

Status
Indicates the routing trace state of the routing table.

This example shows that the general routing trace state is OFF, and three routing tables, that were found in DSLFNTT or DWSLTT, have an individual routing trace state.

Starting Functions (SF)

Some message-processing functions can be associated with a transaction. The transaction processes the messages written to the queue of the message-processing function.

Use the **sf** command to set a message-processing function to NOHOLD status and start the transaction associated with the function. The **sf** command can be used with any function status (see description of the **df** command in Topic "Displaying the Function Status (DF)" on page 22). Starting the transaction is requested from CICS or IMS. When the function cannot be started because CICS or IMS did not accept the start, the function is set to HOLD status. If related functions are defined, the transaction is started once for the two or three related functions, and they are all set to the ACTIVTD status.

Note: You must be authorized to enter the **sf** command for a function which is defined in the MERVA ESA function table DSLFNTT using the parameter MQI=YES.

Command Format

The format of the **sf** command is:

sf	*function*

Parameter Descriptions

The parameter for this command has the following meaning:

function
The name of the MERVA ESA message-processing function whose transaction is to be started.

The transaction name and the (optional) logical terminal name must have been specified in the function-table entry during MERVA ESA generation.

Command Example

This example shows the command you use to start the function L1PR0:

```
sf,l1pr0
```

Example of the Display from an SF Command

"Starting Functions (SF)" shows the display that would result if you entered the command shown in the command example. Message DSL115I confirms the start of the transaction, for details refer to *MERVA for ESA Messages and Codes*.

```
                        Operator Command Processing

> SF,L1PR0
  DSL115I L1PR0    STARTED, TRAN=DSLH    , LT1=L84A    , LT2=
```

```
  115353   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 24. Starting a Message-Processing Function

This message shows the following information:

- The name of the function: L1PR0
- Confirmation that the function is started: STARTED
- The transaction associated with the function: DSLH
- The logical terminal name defined for the function: L84A.

Transaction codes and logical terminal names have 4 characters in CICS and up to 8 characters in IMS.

Shutting Down User Sessions (SHUTDOWN)

The **shutdown** command causes user sessions and MERVA ESA hard-copy-printer tasks to stop when the current message processing has finished.

The **shutdown** command has no effect on:

- The MERVA ESA batch programs, or on MERVA ESA users working under the CMD function (operator command processing).
- The MERVA ESA users working under the MSC function (MERVA ESA System Control).

You can sign on for both functions after a **shutdown** command has been issued.

Note: You must be authorized to use the **shutdown** command.

Command Format

The format of the **shutdown** command is:

shutdown **sh**	

The **shutdown** command has no parameters.

Command Example

You can use the abbreviation **sh** instead of the **shutdown** command:

```
sh
```

Example of the Display from a SHUTDOWN Command

Figure 25 on page 67 shows the display that would result if you entered the command shown in the command example. Message DSL050I confirms that the command has been accepted.

```
                        Operator Command Processing

> SH
  DSL050I SHUTDOWN accepted









  115438   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 25. Shutting Down MERVA ESA

Starting a Program Defined in DSLNPTT (START)

Use the **start** command to start a program that has been defined in DSLNPTT.

Note: You must be authorized to use the **start** command.

Command Format

The format of the **start** command is:

start **s**	{ *progname* | *pid* } [,*parameter*]

Parameter Descriptions

The parameters for this command have the following meanings:

progname
The descriptive name of the program to be started. The name must be defined in DSLNPTT, and a **start** command must be allowed for it.

The program must not have already been started.

pid
A 1-to-3 character identification of the program you want to start. This program identification is generated in DSLNPTT. This parameter can be used instead of the *progname* parameter.

parameter
A 1- to 8-byte *parameter* value for the program to be started. This *parameter* value is moved to the field NPTPARM for use by the started program. For example, the program DSLISYNP (descriptive name SYNPOINT) used under IMS uses a start parameter: a 1- or 2-digit time interval in minutes.

Command Examples

The following section shows some examples of how to enter the **start** command.

Example 1: To start a program, such as the CONSOLE program, you can enter the **start** command with the *progname* parameter:

```
start,console
```

Example 2: You can also specify a program identification (*pid* parameter) to start a program. In this example the abbreviation of the **start** command is used:

```
s,1
```

Example 3: Use a start parameter for the syncpoint program SYNPOINT to write an IMS syncpoint every 5 minutes:

```
s,synpoint,5
```

Example of the Display for a START Command

Figure 26 on page 69 shows an example of the **start console** command as it appears on the panel after it has been entered. Message DSL060I confirms the start of the program.

```
                        Operator Command Processing

> START,CONSOLE
  DSL060I CONSOLE  1   start successful









  115135   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 26. Starting a Program

Stopping a Program Defined in DSLNPTT (STOP)

Use the **stop** command to stop a program defined in DSLNPTT.

Note: You must be authorized to use the **stop** command.

Command Format

The format of the **stop** command is:

stop **p**	{ *progname* | *pid* }

Parameter Descriptions

The parameters for this command have the following meanings:

progname
The descriptive name of the program to be stopped. This name must be defined in DSLNPTT and a **stop** command must be allowed for it. The program must have been started.

pid
An identification, 1 to 3 characters in length, of the program you want to stop. This program identification is generated in DSLNPTT. It can be used instead of the program name.

Command Examples

The following section shows some examples of how to enter the **stop** command.

Example 1: You can specify the program name in the **stop** command:

```
stop,console
```

Example 2: You can also enter a program identification. In this example the abbreviation of the **stop** command is used:

```
p,1
```

Example of the Display from a STOP Command

Figure 27 on page 71 shows an example of the **stop console** command. Message DSL061I confirms that the program has stopped.

```
                        Operator Command Processing

> STOP,CONSOLE
  DSL061I CONSOLE  1   stop successful









  115233   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 27. Stopping a Program

Stopping MERVA ESA (TERMINAT)

The **terminat** command is described in "Stopping MERVA ESA (CANCEL and TERMINAT)" on page 17.

Chapter 5. Operating the SWIFT Link

Before reading this chapter, you should be familiar with the SWIFT network, as described in *S.W.I.F.T. User Handbook*. The SWIFT link communicates with the SWIFT II network.

The SWIFT link can be started automatically when MERVA ESA is started, or it can be started using the MERVA ESA operator command **start swiftii**.

The SWIFT link can be stopped using the MERVA ESA operator command **stop swiftii**. During the MERVA ESA termination, the SWIFT link is stopped automatically.

The SWIFT Link commands described in this chapter cause the generation of the SWIFT LOGIN, SELECT, QUIT, LOGOUT and ABORT messages, and enable you to monitor and control the status of the connection to the SWIFT network.

The commands in this chapter are presented in alphabetical order.

The synonym command codes with the prefix '2' are no longer defined. If required, you can add the required definitions in the DWSNCMTC copy book.

Aborting the FIN Application (ABORTAP)

Use the command **abortap** to abort the connection of the FIN applications (which must have been defined in the Logical Terminal Table DWSLTT) with the SWIFT network. The **abortap** command is accepted in any case, no matter which status the FIN application has. However, if the FIN application is in CLOSED status, the **abortap** command has no effect.

You should use the **abortap** command only in cases where the **quit** command does not work, for example, when the SWIFT network does not send the ISN acknowledgments, and you want to select the FIN application again to resume its processing.

If necessary, the **abortap** command generates an abort AP message (APDU 33) and sends it to the SWIFT network.

When the abort process is complete:

- The association of the FIN application is released (disassociation of the AI layer) if it exists.
- The transport connection of the FIN application is released (T-disconnect of the transport layer) if it exists.

Use the command to abort:

- One FIN application
- All FIN applications on one line to the SWIFT network (if you are authorized by your installation)
- All FIN applications on all lines to the SWIFT network (if you are authorized by your installation).

When one line of the SWIFT link is shared by several financial institutions with several FIN applications, an **abortap** command must be entered for each FIN application that wants to abort the communication with the SWIFT network. Alternatively, you can use the **abortlt** command to abort the master logical terminals or the **abortli** command to abort the line.

Command Format

The format of the **abortap** command is:

abortap **aap**	{[*ltname* | ALL] [,*line* | **ALL**]}

Parameter Descriptions

Note: You must be authorized to use the **abortap** command with an ALL parameter.

The parameters for this command have the following meanings:

ltname | **ALL**
Is the 9-character name of the master logical terminal (LT) whose FIN application is to be aborted.

This name must be:

- A valid logical terminal name that has been given to the financial institution by S.W.I.F.T.
- Defined in the Logical Terminal Table (DWSLTT) as a master LT.
- A FIN application must be defined for this logical terminal in DWSLTT.

If you have the appropriate authorization level, you can use any master logical terminal; otherwise you can use only the master logical terminal that matches the first 9 characters of the origin identification in your User File record.

The default for this parameter depends on whether you use the system console to enter the **abortap** command:

- If you use the system console, the first master logical terminal in the Logical Terminal Table DWSLTT is taken.
- If you do not use the system console, the first 9 characters of the origin identification of your User File record are taken.

If you use the parameter ALL in the place of the *ltname* parameter, the FIN applications of all master logical terminals of all lines or of the specified line are treated depending on their status as follows (the parentheses show the status in response to the **dl** command):

Session key pending (SK PND)
The FIN application is immediately set to CLOSED status.

Session key proceeding (SK PRC)
The FIN application is immediately set to CLOSED status.

Select pending (SEL PND)
The FIN application is immediately set to CLOSED status.

Select proceeding (SEL PRC)
The FIN application is set to ABORT PENDING status.

Open (OPEN)
The FIN application is set to ABORT PENDING status.

Quit pending (QUI PND)
The FIN application is set to ABORT PENDING status.

Quit proceeding (QUI PRC)
The FIN application is set to ABORT PENDING status.

For any other status, there is no change for the FIN application.

If the ABORT PENDING status was set, the abort AP message (APDU 33) is eventually generated and sent to the SWIFT network.

If you use the parameter ALL in the place of the *ltname* parameter, the FIN applications of all master logical terminals of all lines or of the specified line are treated according to their status (the parentheses show the status in response to the **dl** command):

Session key pending (SK PND)
The FIN application is immediately set to CLOSED status.

Session key proceeding (SK PRC)
The FIN application is immediately set to CLOSED status.

Select pending (SEL PND)
The FIN application is immediately set to CLOSED status.

Select proceeding (SEL PRC)
The FIN application is set to ABORT PENDING status.

Open (OPEN)
The FIN application is set to ABORT PENDING status.

Quit pending (QUI PND)
The FIN application is set to ABORT PENDING status.

Quit proceeding (QUI PRC)
The FIN application is set to ABORT PENDING status.

For any other status, there is no change for the FIN application.

If the ABORT PENDING status was set, the abort AP message (APDU 33) is eventually generated and sent to the SWIFT network.

line | **ALL**
If you use the *line* parameter, you must also use the ALL parameter in place of the *ltname*. All FIN applications of the specified line are then treated according to their status. If you specify ALL in place of the *line* parameter, the FIN applications of all lines are treated according to their status. If the second parameter is not specified, all lines are processed as default.

Command Examples

This section shows examples of how to enter the **abortap** command.

Example 1: This example shows the **abortap** command for the FIN application of a specific master logical terminal:

```
abortap,vndebet2a
```

Example 2: This example shows the **abortap** command for the FIN application of all master logical terminals on line 1:

```
abortap,all,1
```

Example 3: Enter one of the following commands to abort all FIN applications on all lines:

```
aap,all
aap,all,all
```

Example of the Display from an ABORTAP Command

Figure 28 shows an example of a panel displayed in response to the **abortap** command for all FIN applications on all lines. Figure 29 shows how the FIN applications are aborted with a **dm last** command.

```
                        Operator Command Processing

> AAP,ALL
  DWS554I ABORTAP ALL FIN accepted for all lines
```

```
  115032   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 28. Abort of All FIN Applications on All Lines

```
                        Operator Command Processing

> DM LAST
  DSL075I Display Message 1999MAY28
          ...
          ...
          ...
          ...
          ...
          ...
          ...
          ...
          ...
          ...
  092915 DWS621I VNDEBET2A FIN ABORT proceeding on line 1
  092916 DWS621I VNDEBET2B FIN ABORT proceeding on line 2
  092930 DWS622I VNDEBET2A FIN ABORT successful on line 1
  092932 DWS622I VNDEBET2B FIN ABORT successful on line 2

  115034   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 29. Completion of Abort of All FIN Applications on All Lines

Aborting Lines to the SWIFT Network (ABORTLI)

Use the command **abortli** to abort lines to the SWIFT network, that is, to abort the computer based terminal (CBT).

The **abortli** command is accepted in any case, no matter which status the master logical terminals and FIN applications of this line have.

You should use the **abortli** command only in cases where the **quit**, **logout**, **abortap**, and **abortlt** commands do not work, for example, when the SWIFT network does not send the acknowledgments and you do not want to wait for the time-out on these acknowledgments as you want to resume processing using **login** and **select**.

The **abortli** command performs the same functions as the **close,***line***,imm** command.

The **abortli** command does not generate any message for sending to the SWIFT network. Instead:

- The traffic on the line stops.
- On an X.25 line, the session to MERVA Extended Connectivity is closed.
- The line subtask is detached.
- All storage allocated for the line is freed.
- The line definition module is deleted.

Note: You must be authorized to use the **abortli** command, and you can use it to abort one line or all lines to the SWIFT network.

Command Format

The format of the **abortli** command is:

abortli **ali**	{*line* | ALL}

Parameter Descriptions

The parameter of this command has the following meaning:

line | **ALL**
Specifies the number of the line you want to abort, or ALL, if you want to abort all lines.

Command Examples

This section shows some examples of how to enter the **abortli** command.

Example 1: Enter the following command to abort line 2:

```
abortli,2
```

Example 2: This example shows the abbreviation of the **abortli** command with the ALL parameter:

```
ali,all
```

Example of the Display from an ABORTLI Command

Figure 30 shows an example of a panel displayed in response to the **abortli** command for line 2.

```
                        Operator Command Processing

> ALI 2
  DWS561I ABORTLI accepted for line 2









  115036   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 30. Aborting Line 2 to the SWIFT Network

Aborting Master Logical Terminals (ABORTLT)

Use the command **abortlt** to abort the connection of SWIFT master logical terminals (which must have been defined in the Logical Terminal Table DWSLTT) with the SWIFT network. The **abortlt** command is accepted in any case, no matter which status the master logical terminals have. However, if a master logical terminal is in LOGOUT status, the **abortlt** command has no effect. Aborting a master logical terminal also aborts the FIN application of this master logical terminal.

You should use the **abortlt** command only in cases where the **logout** command does not work, for example, when the SWIFT network does not send the ISN acknowledgments, and you want to log in the master logical terminal again for resuming its processing.

If necessary, the **abortlt** command generates an abort LT message (APDU 35) and sends it to the SWIFT network. If FIN applications are also aborted, the abort LT message is sent only after the abort process of these FIN applications is complete and these applications are closed.

When the abort process is complete, the following is done for this master logical terminal:

- The association of the application control (APC) is released (disassociation of the AI layer), if it exists.
- The transport connection of the APC is released (T-disconnect of the transport layer), if it exists.

In addition, when the abort process for the last master logical terminal is complete:

- The association of the logical terminal control (LTC) is released (disassociation of the AI layer).
- The transport connection of the LTC is released (T-disconnect of the transport layer).
- The network connection is released (N-disconnect of the link layer). For a switched line, the physical connection to the SWIFT network is released.

Use the command to abort one master logical terminal. If you are authorized by your installation, you can use the command to abort all master logical terminals on one line or on all lines to the SWIFT network.

When one line of the SWIFT link is shared by several financial institutions with several master logical terminals, an **abortlt** command must be entered for each master logical terminal that wants to abort the communication with the SWIFT network. Alternatively, you can use the **abortli** command to abort the line.

Command Format

The format of the **abortlt** command is:

abortlt **alt**	{[*ltname* | ALL] [,*line* | **ALL**]}

Parameter Descriptions

Note: You must be authorized to use the **abortlt** command with an ALL parameter.

The parameters for this command have the following meanings:

ltname
Is the 9-character name of a master logical terminal (LT).

This name must be:

- A valid logical terminal name that has been given to the financial institution by S.W.I.F.T.
- Defined in the Logical Terminal Table DWSLTT as a master LT.

If you have the appropriate authorization level, you can use any master logical terminal; otherwise you can use only the master logical terminal that matches the first 9 characters of the origin identification in your User File record.

The default for this parameter depends on whether you use the system console to enter the **abortlt** command:

- If you use the system console, the first master logical terminal in the Logical Terminal Table DWSLTT is taken.
- If you do not use the system console, the first 9 characters of the origin identification of your User File record are taken.

The master logical terminal is treated depending on its status (the parentheses show the status in response to the **dl** command):

Session key pending (SK PND)
The master logical terminal is immediately set to LOGOUT status.

Session key proceeding (SK PRC)
The master logical terminal is immediately set to LOGOUT status.

Login pending (LIN PND)
The master logical terminal is immediately set to LOGOUT status.

Login proceeding (LIN PRC)
The master logical terminal is set to ABORT PENDING status.

Logged in (LOGIN)
The master logical terminal is set to ABORT PENDING status.

Logout pending (OUT PND)
The master logical terminal is set to ABORT PENDING status.

Logout proceeding (OUT PRC)
The master logical terminal is set to ABORT PENDING status.

For any other status, there is no change for the master logical terminal.

If the ABORT PENDING status was set, the abort LT message (APDU 35) is eventually generated and sent to the SWIFT network.

ALL[,*line* **| ALL]**
If you use the parameter ALL,*line*, all master logical terminals assigned to the specified line are treated according to their status. If you specify ALL,ALL, all master logical terminals of all lines are treated according to their status as listed below. If the second parameter is not specified, all lines are used as default (the parentheses show the status in response to the **dl** command):

Session key pending (SK PND)
The master logical terminal is immediately set to LOGOUT status.

Session key proceeding (SK PRC)
The master logical terminal is immediately set to LOGOUT status.

Login pending (LIN PND)
The master logical terminal is immediately set to LOGOUT status.

Login proceeding (LIN PRC)
The master logical terminal is set to ABORT PENDING status.

Logged in (LOGIN)
The master logical terminal is set to ABORT PENDING status.

Logout pending (OUT PND)
The master logical terminal is set to ABORT PENDING status.

Logout proceeding (OUT PRC)
The master logical terminal is set to ABORT PENDING status.

For any other status, there is no change for the master logical terminal.

If the ABORT PENDING status was set, the abort LT message (APDU 35) is eventually generated and sent to the SWIFT network.

If a master logical terminal is set to ABORT PENDING status, its FIN applications that are not in CLOSED status are treated as described in "Aborting the FIN Application (ABORTAP)" on page 74.

Command Examples

This section shows some examples of how to enter the **abortlt** command.

Example 1: This example shows the **abortlt** command for a specific master logical terminal:

```
abortlt,vndebet2a
```

Example 2: Enter the following command to set the status of all master logical terminals assigned to line 2 to LOGOUT or ABORT PENDING, depending on their current settings. The FIN applications of these master logical terminals are also set to status CLOSED or ABORT PENDING, depending on their current settings:

```
alt,all,2
```

Example of the Display from an ABORTLT Command

Figure 31 shows an example of a panel displayed in response to the **abortlt** command with the ALL parameter for all lines. Figure 32 shows, by a **dm last** command, how the master logical terminals and their FIN applications are aborted.

```
                        Operator Command Processing

> ALT ALL,ALL
  DWS544I ABORTLT ALL accepted for all lines









  115133   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 31. Abort of All Master Logical Terminals on All Lines

```
                        Operator Command Processing

> DM LAST
  DSL075I Display Message 1999MAY28
         ...
         ...
  091915 DWS621I VNDEBET2A FIN ABORT proceeding on line 1
  091916 DWS621I VNDEBET2B FIN ABORT proceeding on line 2
  091930 DWS622I VNDEBET2A FIN ABORT successful on line 1
  091931 DWS621I VNDEBET2A ABORT proceeding on line 1
  091932 DWS622I VNDEBET2B FIN ABORT successful on line 2
  091933 DWS621I VNDEBET2B ABORT proceeding on line 2
  091940 DWS622I VNDEBET2A ABORT successful on line 1
  091942 DWS622I VNDEBET2B ABORT successful on line 2
  091955 DWS651I CBT disconnect is complete on line 1
  091957 DWS651I CBT disconnect is complete on line 2
  092001 DWS643I Network connection terminated (T,0) on line 1
  092002 DWS643I Network connection terminated (T,0) on line 2

  115043   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 32. Completion of Abort of All Master Logical Terminals on All Lines

Closing a Line (CLOSE)

Use the command **close** to close a line to the SWIFT network. Closing a line:

- Detaches the line subtask
- Closes the session to MERVA Extended Connectivity on the 37xx Communications Controller (X.25 line)
- Frees all storage allocated for the line
- Deletes the line definition module.

The command is useful if you want to use the storage of an unused line for other purposes.

Under normal circumstances, you use the **close** command only if the line is not active, that is, all master logical terminals assigned to this line are logged out and all their FIN applications are closed. However, after errors on the line, you may want to close the line and open it for a new login. In this case, you can use the **close** command with the IMM or DUMP parameter.

Note: You must be authorized to use the **close** command.

Command Format

The format of the **close** command is:

close **cl**	*line*[,{IMM | DUMP}]

Parameter Descriptions

The parameters for this command have the following meaning:

line
Is the number of the line to be closed. It must be a number from 1 to 30. This number refers to the name of a line definition module, for example, DWSLIN1 for line 1 or DWSLIN15 for line 15. If the number you enter is wrong (that is, there exists no line definition module for the number), the command is rejected.

There is no default for the *line* parameter.

IMM | DUMP
The IMM or DUMP parameters close the line whether the line is active or not. This can be helpful after errors on this line that prevent the line from becoming inactive. The DUMP parameter provides also a dump of the line subtask with the dump code U102.

Command Examples

This section shows some examples of how to enter the **close** command.

Example 1: Enter the following command to close line 2:

```
close,2
```

Example 2: Enter the following command abbreviation to close line 3 even if line 3 is active:

cl,3,imm

Example of the Display from a CLOSE Command

Closing a Line (CLOSE) shows an example of the display in response to the **close** command in Example 1.

```
                        Operator Command Processing

> CLOSE,2
  DWS581I CLOSE command accepted for line 2

  115038   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 33. Close a Line to the SWIFT Network

Displaying and Updating the Delivery Subset Mnemonics (DDS)

You use the **dds** command:

- To display the delivery subset mnemonics of a FIN application
- To update the delivery subset mnemonics of a FIN application before you use the **select** command for this FIN application.

The display shows:

- The status of the FIN application
- The actual delivery subset mnemonics contained in the pertinent entry of the Logical Terminal Table (DWSLTT)
- Instructions on how to update the delivery subset mnemonics if the status of the FIN application is CLOSED.

If you use the system console to enter the **dds** command, you cannot update the delivery subset mnemonics.

Command Format

The format of the **dds** command is:

dds	[*ltname*]

Parameter Descriptions

The parameter for this command has the following meaning:

ltname
Is the 9-character name of the master logical terminal for whose FIN application you want to display or update the delivery subset mnemonics.

This name must be:

- A valid logical terminal name that has been given to the financial institution by S.W.I.F.T.
- Defined in the Logical Terminal Table DWSLTT as a master LT. A FIN application must be defined for this logical terminal in DWSLTT.

If you have the appropriate authorization level, you can use any master logical terminal; otherwise you can use only the master logical terminal that matches the first 9 characters of the origin identification in your User File record.

The default for this parameter depends on whether you use the system console to enter the **login** command:

- If you use the system console, the first master logical terminal in the Logical Terminal Table DWSLTT is taken.
- If you do not use the system console, the first 9 characters of the origin identification of your User File record are taken.

The display shows up to 30 delivery subset mnemonics of the FIN application. If the status is CLOSED, you can add, delete, and change the delivery subset mnemonics, and you can sort them in a different order by means of a sequence field. The sequence fields are renumbered in ascending order in the next display.

After having determined the new delivery subset mnemonics, you update the DWSLTT entry by entering the **dds** command again.

Command Example

This section shows an example of how to enter the **dds** command.

Enter the following command to display the delivery subset mnemonics of the FIN application of a master logical terminal.

```
dds,vndebet2a
```

Example of the Display from A DDS Command

Displaying and Updating the Delivery Subset Mnemonics (DDS) shows an example of a panel displayed in response to a **dds** command.

```
                      Operator Command Processing

> DDS VNDEBET2A
  DWS575I VNDEBET2A FIN delivery subsets          Status=CLOSED
  SQ  subset    SQ  subset    SQ  subset    SQ  subset    SQ  subset
  01  SYSTEM    02  URGENT    03  NORMAL    __  ______    __  ______
  __  ______    __  ______    __  ______    __  ______    __  ______
  __  ______    __  ______    __  ______    __  ______    __  ______
  __  ______    __  ______    __  ______    __  ______    __  ______
  __  ______    __  ______    __  ______    __  ______    __  ______
  __  ______    __  ______    __  ______    __  ______    __  ______

To update the delivery subset mnemonics:
   Change one by overtyping, delete one by overtyping with blanks
   Add one in a free field with a sequence number
   Change the sequence using the SQ fields
   Redisplay by pressing ENTER
   Update DWSLTT by using the DDS command again

  115433   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 34. Displaying the Delivery Subset Mnemonics

In this example, the input areas for the sequence fields and the delivery subset mnemonics are indicated by underline characters (_). Underlining input areas depends on the type of display station used and the setting of the **ul** command (see the *MERVA for ESA User's Guide* for details).

Displaying X.25 Interface Information (DIVA)

You use the **diva** command:

- To monitor the status of the MERVA ESA X.25 send and receive buffers
- To monitor the status of the SWIFT connection via X.25
- To list the values of your VTAM® bind customization (DWSVLINE macro).

The display shows only:

- Lines that have been successfully initialized
- Lines that are assigned to a master logical terminal in the Logical Terminal Table (DWSLTT)
- Lines that are used for connection to SWIFT via X.25.

Command Format

The format of the **diva** command is:

diva	*line* [,Buffers | **States** | Vtambnd]

Parameter Description

The parameters for this command have the following meanings:

line
Denotes the number of the line whose X.25 interface information you want to display. The value entered must be a number from 1 to 30. The number refers to the name of a line definition module, for example, DWSLIN1 for line 1, DWSLIN15 for line 15. If the line is not initialized or is not an X.25 line, the command is rejected.

BUFFERS
Displays the status of the MERVA ESA X.25 send and receive buffers. There are four send buffers and three receive buffers defined for each line. You can abbreviate BUFFERS with any of B, BU, BUF, BUFF, BUFFE, and BUFFER.

STATES
Displays the status of the SWIFT connection via X.25, such as:

- VTAM Bind information, such as the PLU and SLU name, and used telephone numbers
- Current state of the network layer and the VTAM interface layer
- Last request to MERVA Extended Connectivity
- Last disconnect (disco) reason and last reset reason from MERVA Extended Connectivity
- Last messages from the MERVA ESA VTAM interface program

You can abbreviate STATES with any of S, ST, STA, STAT, and STATE, or omit it because it is the default.

VTAMBND
Displays most VTAM bind values. These values are either hard-coded or you have customized them via the DWSVLINE macro. You can abbreviate VTAMBND with any of V, VT, VTA, VTAM, VTAMB, and VTAMBN.

Command Examples

This section shows some examples of how to enter the **diva** command.

Example 1: Enter the following command to display the status of the send and receive buffers of line 3:

```
diva 3,buffers
```

Example 2: Enter the following command to display the status of the MERVA ESA VTAM interface program for line 7:

```
diva 7,states
```

Example 3: Enter the following command to display most values of the VTAM bind customization for line 9:

```
diva 9,vtambnd
```

Example of the Command from a DIVA Command

Figure 35 shows an example of a panel displayed in response to a **diva***line,*BUFFERS command.

```
                        Operator Command Processing

> DIVA 3,BUFFERS
  DWS460I Line 3. Display BUFFERS of VTAM X.25 interface
  o SEND (Network Layer to VTAM)      |    o RECEIVE (VTAM to Network L.)
                                      |
  total sent: 23,456                  |    total received: 13,579
                                      |
  Status 0 - free send buffer: 2      |    Status 0 - free recv buffer: 1
  Status 2 - to be sent      : 1      |    Status 6 - waiting for data: 1
  Status 3 - VTAM accepted   : 0      |    Status 7 - receive complete: 0
  Status 4 - send scheduled  : 1      |    Status 8 - recv in process : 1
  Status 5 - send complete   : 0      |    Status 9 - receive error   : 0
  Status 9 - send error      : 0      |
                                      |
                                      |
                                      |
                                      |
                                      |
  115031   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 35. Displaying the Status of X.25 Send and Receive Buffers

When you enter the **diva** command with the BUFFERS parameter the following information is shown:

total sent Number of elements sent since the session was established

Status 0 - free send buffer
Number of send buffer elements with status 0 = free

Status 2 - to be sent
Number of send buffer elements with status 2 = data to be sent

Status 3 - VTAM accepted
Number of send buffer elements with status 3 = send started

Status 4 - send scheduled
Number of send buffer elements with status 4 = send scheduled

Status 5 - send complete
Number of send buffer elements with status 5 = send complete

Status 9 - send error
Number of send buffer elements with status 9 = send error

total received Number of elements received since the session was established

Status 0 - free recv buffer
Number of receive buffer elements with status 0 = free

Status 6 - waiting for data
Number of receive buffer elements with status 6 = receive initiated

Status 7 - receive complete
Number of receive buffer elements with status 7 = receive complete

Status 8 - recv in process
Number of receive buffer elements with status 8 = receive in process

Status 9 - receive error
Number of receive buffer elements with status 9 = receive error

Figure 36 shows an example of a panel displayed in response to a **diva***line*,STATES command.

```
                        Operator Command Processing

> DIVA 7,STATES
  DWS461I Line 7. Display STATES of VTAM X.25 interface
  Network link state  : 1           Last disco reason: X'00 00 00'
  Last Connection req.: X'00'       Last reset reason: X'00 00 00'

  VTAM interface state: 1
  PLU-name .......... : ID0AC384    SLU-name: F39VU21
  Last VTAM message   : DWS423I Line=7 Session established, PLU=ID0AC384
                         SLU=F39VU21
  Phone number used   : 00497111234567890
  Local DTE address   : 1234567890123
  Remote DTE address  :







  115037   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 36. Displaying the Status of MERVA ESA VTAM Interface

When you enter the **diva** command with the STATES parameter the following information is shown:

Network link state
The Network link state is set by the network layer program DWSNLNKV to one of:

0 Not initialized

1 Initialized, no network connection

2 Outgoing connection pending

3 Incoming connection pending

4 Data transfer ready

5 Provider initiated RESET pending.

Last disco reason
Last disconnect reason. The format is 'origin - cause - diagnostic'. You can find a detailed description of these fields in *MERVA Extended Connectivity Installation and User's Guide*. You can use the command SHOW X25REAS to get an explanation of the reason code.

Last Connection req.
Last Extended Connectivity request. It can be one of the following:

X'00' Data request

X'01' Data indication

X'02' Connect request

X'03' Connect indication

X'04' Connect response

X'05' Connect confirm

X'08' Reset request

X'09' Reset indication

X'0A' Reset response

X'0B' Reset confirm

X'10' Disconnect request

X'11' Disconnect indication.

Last reset reason
Last reset reason. The format is 'origin - cause - diagnostic'. You can find a detailed description of these fields in *MERVA Extended Connectivity Installation and User's Guide*.

VTAM interface state
The VTAM interface state is set by the MERVA ESA VTAM interface program DWSVTMLC to one of:

0 VTAM ACB closed / no session

1 Session is between brackets

2 Session can receive data

3 Session can send data

4 Session is waiting for a definite response (DR)

5 BID rejected, awaiting data from SLU

6 Error.

PLU-name
Primary logical unit name.

SLU-name
Secondary logical unit name.

Last VTAM message
Last VTAM message. It is displayed in up to two lines.

Phone number used
Telephone number used for dial to SWIFT (switched line).

Local DTE address
The local data terminal equipment (DTE) address. Can be null, a two-digit subaddress, or a telephone number.

Remote DTE address
The remote data terminal equipment (DTE) address. This represents the address an X.25 connection is established to, the called DTE address. Must not have been specified for a leased line nor for a switched line.

Figure 37 shows an example of a panel displayed in response to a **diva***line*,VTAMBND command.

```
                        Operator Command Processing

> DIVA 9,VTAMBND
  DWS462I Line 9. Display VTAM BIND parameters of VTAM X.25 interface
  VTAM logon mode name ..... :
  VTAM bind parameters ..... : X'0103 03B1 B030 8000 0085 85'
                               X'0000 0100 0000 0000 0000 0000 0000'
  RU size receiving / sending: 256   / 256
  Local  NSAP name ......... : BANKUSAA01000
  Remote NSAP name ......... : FIN
  Line type (Leased/Switched): L
  Call user data (CUD) ..... : X'4D45525645333230'
  Phone number for call back : (n/a for leased line)









  115039   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 37. Displaying the VTAM Bind Values of a Line

When you enter the **diva** command with the VTAMBND parameter the following information is shown:

VTAM logon mode name
VTAM logon mode name. Customizable via DWSVLINE macro. If specified, the bind parameter of this logon mode name will overwrite the bind parameter generated via the DWSVLINE macro, which are shown below.

VTAM bind parameters
VTAM bind parameter byte 1 to 25. Byte 1 to 9 and 12 to 25 are hard-coded in the DWSVLINE macro. Via the DWSVLINE macro you can customize byte 10 (RU size receiving) and byte 11 (RU size sending). If a VTAM logon mode name is specified, those bind parameter will overwrite the bind parameter listed here.

RU size receiving / sending
Request unit size for receiving and sending. Customizable via DWSVLINE macro. The RU sizes are coded in bytes 10 and 11 of the bind parameter.

Local NSAP name
Local network service access point name. Customizable via DWSVLINE macro.

Remote NSAP name
Remote network service access point name - always FIN. Hard-coded in the DWSVLINE macro.

Line type (Leased/Switched)
Line type (Leased or Switched). Customizable via DWSVLINE macro.

Call user data (CUD)
The Call user data consists of two parts. The first part is fixed as defined by SWIFT. It is 'MERVE320' in ASCII format, which is X'4D45525645333230'.

The second part defines additional Call user data and is customizable via DWSVLINE macro in hexadecimal notation. Must not have been specified for a leased line nor for a switched line.

Phone number for call back
The local telephone number for call back. Customizable via DWSVLINE macro. Applicable only for switched lines (see Line type).

Displaying the Line and Link Status (DL)

You use the **dl** command:

- To monitor the status of the lines to the SWIFT network
- To monitor the status of the links of the master logical terminals and their FIN applications with:
  - Login sequence numbers (LSN) or Select sequence numbers (SSN)
  - Session numbers (SN)
  - Input sequence numbers (ISN)
  - Output sequence numbers (OSN)
  - Status.

The display shows only:

- Lines that have been successfully initialized
- Lines that are assigned to a master logical terminal in the Logical Terminal Table (DWSLTT).

Command Format

The format of the **dl** command is:

dl	[**ALL**[,FIRST]] FIRST LINES *line*[,FIRST] *ltname*

Parameter Description

The parameters for this command have the following meanings:

ALL

If you enter the **dl all** command (or just **dl**, as ALL is the default if you do not enter a parameter) the status of all lines and their master logical terminals and their FIN applications is displayed:

- The status of the line is displayed first.
- The first master logical terminal (that is assigned to this line) is displayed next with its ISN, OSN, LSN, and LOGIN status.
- The FIN application defined for this logical terminal with its status.
- The second master logical terminal that is assigned to this line and its FIN application follows, then the third one and so on.

If all LTs and applications of this line are displayed, the status of the next line with its LTs and FIN applications are displayed.

The following information is displayed for each line:

- Line number
- Line status.

If the display does not fit on one panel, you can request the continuation of the display by entering the same command again (**dl** or **dl all**). If you want to see the first part of the display again, enter **dl first**.

FIRST
Starts the display with the first available line, for example, after a **dl** or **dl all** command.

LINES
Displays only the status of the lines, but not the status of the master logical terminals and the FIN applications.

ltname
Is the name of a master logical terminal whose link status you want to display. It must consist of 1 to 9 characters.

If less than 9 characters are specified, the parameter is used as a generic name and all master logical terminals starting with the specified characters are displayed.

The display shows the status of the line that is currently assigned to this master logical terminal, followed by the status of this master logical terminal and its FIN application.

If 9 characters are specified, the display shows the same information as in the response to the **setlt** command shown in "Setting Parameters for a Master Logical Terminal (SETLT)" on page 117.

line[**,FIRST**]
Is a number from 1 to 30 that specifies the line whose status and master logical terminals and FIN applications you want to display.

If there is more information to display for this line than fits into one response, you can request the continuation of the display by entering the same command again. If you want to see the first display again, enter:

```
dl,line,first
```

You can also use the *line*[,FIRST] parameter after the *ltname* parameter to display only the status of these master logical terminals of one line.

Command Examples

This section shows some examples of how to enter the **dl** command.

Example 1: Enter the following command to display the status of a specific master logical terminal. The status of the line currently assigned to this master logical terminal and of its FIN application is also displayed:

```
dl,vndebet2a
```

Example 2: Enter the following command to display the status of all lines followed by their master logical terminals and FIN applications:

```
dl
```

Example 3: Enter the following command to display the status of all lines without the status of their master logical terminals and FIN applications:

```
dl,lines
```

Example of the Display from a DL Command

Figure 38 shows an example of a panel displayed in response to a **dl** command.

```
                        Operator Command Processing

> DL
  DWS570I LT and AP display
  Line=1 Line is initialized (X.25)
  LT name   AP       LSN  SESS  ISN    OSN   MXW ISW OSW status  SS Q AS
  VNDEBET2A GPA      0013 0012 000002 000001 001 000 000 LOGIN        AC
  VNDEBET2A FIN      0006 0004 000044 000023 010 000 001 OPEN    YY Y AC
  VNDEBET2B GPA      0000 0000 000000 000000 000 000 000 LOGOUT       DC
  VNDEBET2B FIN      0000 0000 000000 000000 000 000 000 CLOSED       DC

  Line=3 Line is initialized (X.25)
  LT name   AP       LSN  SESS  ISN    OSN   MXW ISW OSW status  SS Q AS
  VNDEBET2C GPA      0013 0012 000002 000001 001 000 000 LOGIN        AC
  VNDEBET2C FIN      0006 0004 000044 000023 010 000 001 OPEN    YY Y AC

  Line=4 Line not initialized


  115433   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 38. Displaying the Line Status

Line 1 and line 3 are connections to the

LT name
Name of the master logical terminal whose link status is displayed

AP
Name of the application whose status is displayed

LSN
Last login sequence number (LSN) for the master logical terminal or the last select sequence number (SSN) for the FIN application. If secure login/select (SLS) is used and the session key for this LSN or SSN is available, an asterisk (*) precedes the LSN or SSN.

SESS
Session number

ISN
Last input sequence number

OSN
Last output sequence number

MXW
Maximum window

ISW
Actual ISN window

OSW
Actual OSN window

status
Login or select status. If automatic repetition of login or select is active, an asterisk (*) precedes the login or select status.

SS Application select state

Q LT-directed queue specification

AS
Association status.

This example shows that line 2 is not initialized, nor is there any master logical terminal assigned to it in the Logical Terminal Table DWSLTT. This example also shows that line 4 could be closed (using the **close** command), as there is no master logical terminal assigned to it.

Displaying the Active Lines and Links (DLA)

You use the **dla** command in a similar way as the **dl** command; only the logical terminals which are not logged out are displayed. The line number is displayed together with the status of the GPA application rather than using a separate display line. This results usually in a compacter display when a multitude of lines and links are active.

Refer to the description of the **dl** command for details about the usage of the command and its parameters.

Command Format

The format of the **dla** command is:

dla	*ltname*[,FIRST]

Parameter Description

The parameters for this command have the following meanings:

ltname
Is the name of a master logical terminal whose link status you want to display. It must consist of 1 to 9 characters.

If less than 9 characters are specified, the parameter is used as a generic name and all master logical terminals starting with the specified characters are displayed.

The display shows the status of a master logical terminal and its FIN application, if it is not logged out.

FIRST
If there is more information to display than fits into one response, you can request the continuation of the display by entering the same command again. If you want to see the first display again, enter:

```
dla ltname,first
```

Command Example

Enter the following command to display the status of all active master logical terminals.

```
dla
```

Example of the Display from a DLA Command

Figure 39 shows an example of a panel displayed in response to a **dla** command.

```
                      Operator Command Processing

> DLA
  DWS570I LT and AP display
  LT name   AP       LSN  SESS  ISN    OSN   MXW ISW OSW status  SS Q AS
  VNDEBET2A GPA   1  0241 1488 000001 000005 001 000 000 LOGIN        SC
  VNDEBET2A FIN      0188 1358 163143 183576 012 012 000 OPEN    YY Y AC
  VNDEBET2C GPA   3  0352 1514 000002 000001 010 000 000 LIN PRC      DC
  VNDEBET2C FIN      0141 1219 065447 071475 000 000 000 CLOSED       DC



  154913   is the time of this display









Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 39. Displaying the Line Status for Active Logical Terminals

The information shown for both applications of the master logical terminal, GPA and FIN, is identical to the display shown in response to the **dl** command. The number on the right of the indication GPA shows the line number used by this logical terminal.

Starting the Connection to the SWIFT Network (LOGIN)

Use the **login** command to start the connection of a SWIFT master logical terminal (which must be defined in the Logical Terminal Table DWSLTT) to the SWIFT network.

The **login** command for a master logical terminal is only accepted if this master logical terminal is in LOGOUT status.

If the line is not initialized when the **login** command is entered:

- The line definition module is loaded.
- The subtask for processing this line is attached.
- The line is initialized.

  The **login** command generates a SWIFT login message (APDU 02) and sends it to the SWIFT network. If it is the first login message of this CBT, the following actions are also performed:

  1. The network connection is established (link layer). For a switched line that means also connecting physically to the SWIFT network.
  2. The transport connection of the logical terminal control (LTC) is established (transport layer).
  3. The association of the LTC is established (AI layer).

If the login is accepted by the SWIFT network, the transport connection and the association of the application control (APC) of this master logical terminal are established.

When this line of the SWIFT link is shared by several financial institutions with several master logical terminals, a **login** command must be entered for each master logical terminal that wants to communicate with the SWIFT network.

Note: You must be authorized to use the **login** command.

Command Format

The format of the **login** command is:

login **li**	[*ltname*],[*lsn*],[*sk1*],[*sk2*][,*window*]

Parameter Descriptions

Note: When one of the parameters shown in the command format is omitted, but one of them to the right of it is specified, all the commas must be specified up to the last parameter used.

The parameters for this command have the following meanings:

ltname
Is the 9-character name of the master logical terminal that you want to log in to the SWIFT network.

This name must be:

- A valid logical terminal name that has been given to the financial institution by S.W.I.F.T.

- Defined in the Logical Terminal Table DWSLTT as a master LT.

If you have the appropriate authorization level, you can use any master logical terminal; otherwise you can use only the master logical terminal that matches the first 9 characters of the origin identification in your User File record.

The default for this parameter depends on whether you use the system console to enter the **login** command:

- If you use the system console, the first master logical terminal in the Logical Terminal Table DWSLTT is taken.
- If you do not use the system console, the first 9 characters of the origin identification of your User File record are taken.

lsn Is the login sequence number (LSN) required by the SWIFT network for the login message. The number you use must be in the range from 0 to 9999. The current login sequence number is used as the default for this parameter.

It is only necessary to enter a value for this parameter after the first MERVA ESA startup or when the SWIFT network has reset the number. The default can be used at all other times.

If SLS is used and you enter the *lsn* parameter, and there is a session key available in the DWSLTT entry for this master logical terminal, this session key is erased even if you enter the same LSN as already contained in the DWSLTT entry. This forces the SWIFT Link to get a new session key, for example, after a change of the ICC parameters.

sk1

Is one of the following:

- If paper tables are used for login, this parameter is the first 4 digits of the session key required for the login sequence number. It is used to calculate the user authentication field (field 501) for the login message.

  You need only enter this parameter if the Login Authorization Table (DWSLOG2) is not used in your installation of the SWIFT link. If DWSLOG2 is included, and you enter a value for this parameter, it overrides the value supplied by DWSLOG2. This parameter is mandatory, if you do not have DWSLOG2 or the actual LSN is not defined in DWSLOG2.
- If secure login/select (SLS) is used, this parameter is the 32 characters session key from the card reader when you use unconnected mode. If connected mode or preloaded session keys are used, you need not enter the session key here. Refer to "Maintenance of Session Keys for Login and Select" on page 122 for details on preloading session keys.
- You can specify AUTO for this parameter if you want an automatic repetition of login after failures. This requires preloaded session keys or a connection to the workstation for USE (User Security Enhancements) when SLS is used, or DWSLOG2 containing the session keys when paper tables are used.

sk2

Is one of the following:

- If paper tables are used for login, this parameter is the second 4 digits of the session key required for the login sequence number. It is used to calculate the user authentication field (field 501) for the login message. The same rules apply as for the *sk1* parameter for using paper tables.

- If secure login/select (SLS) is used, this parameter is the 4 characters check value for the session key specified with the *sk1* parameter from the card reader when you use unconnected mode.
- If you have specified AUTO for the *sk1* parameter, you can specify a retry count from 1 to 9 for the automatic repetition of login after failures with the *sk2* parameter. If you enter a value greater than 9, only the first digit is used. If you do not specify this parameter or you specify an incorrect value, the default of 3 is used.

window
Is the window for this session of the master logical terminal for the field 110 of the login message. The number entered must be in the range from 1 to 999. The default for this parameter is the window size specified in the Logical Terminal Table DWSLTT.

Usually the SWIFT network accepts only a value of 1 for the window in the login acknowledgment (APDU 22).

Command Examples

This section shows some examples of how to enter the **login** command.

Example 1: This example shows the **login** command with the following parameters:

- *ltname* - VNDEBET2A
- *lsn* - 111
- *sk1* - is omitted
- *sk2* - is omitted
- *window* - 1

The format of the command is:

```
login,vndebet2a,111,,,1
```

The three commas after the *lsn* parameter (111) indicate that the parameters *sk1* and *sk2* are not entered in the command.

Example 2: This example shows the abbreviation for the **login** command (**li**) with the following parameters when using paper tables:

- *ltname* - VNDEBET2A
- *lsn* - 44
- *sk1* - 0555
- *sk2* - 0666
- *window* - is omitted

The format of the command is:

```
li,vndebet2a,44,0555,0666
```

Example 3: If you enter the command without any parameters, the default values are assumed:

```
li
```

Example 4: This example shows the abbreviation for the **login** command with the *window* parameter only:

```
li,,,,,1
```

The five commas after the command show that the *ltname*, the *lsn*, the *sk1*, and the *sk2* parameters have not been entered, and the defaults are used.

Example 5: This example shows, when SLS unconnected mode is used, how to enter the **login** command with the 32 character session key in the *sk1* parameter and the 4 character check value in the *sk2* parameter:

```
li,,,967532247e7cd4d3b815844002ec262f,1668
```

The three commas after the command show that the *ltname* and the *lsn* parameters have not been entered, and the defaults are used.

Example of the Display from a LOGIN Command

Starting the Connection to the SWIFT Network (LOGIN) shows an example of a panel displayed in response to the **login** command.

```
                        Operator Command Processing

> LOGIN,VNDEBET2A
  DWS501I VNDEBET2A LOGIN accepted for line 1, login pending
```

```
  115531   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 40. Login of a Master Logical Terminal

Ending the Connection to the SWIFT Network (LOGOUT)

Use the command **logout** to end the connection of a SWIFT master logical terminal (which must have been defined in the Logical Terminal Table DWSLTT) with the SWIFT network. The master logical terminal must be successfully logged in to the SWIFT network after a **login** command. The functions of the **quit** command are executed for the FIN application of this master logical terminal together with the **logout** command (see "Quitting FIN Applications (QUIT)" on page 108 for details). The **logout** command generates a SWIFT logout message (APDU 06) and sends it to the SWIFT network, when there are no acknowledgment messages outstanding for messages sent to the SWIFT network for this master logical terminal (ISN acknowledgments), and the FIN application of this master logical terminal is in closed status.

When the logout process is complete:

- The association of the application control (APC) is released (disassociation of the AI layer).
- The transport connection of the APC is released (T-disconnect of the transport layer).

In addition, when the logout process for the last master logical terminal is complete:

- The association of the logical terminal control (LTC) is released (disassociation of the AI layer).
- The transport connection of the LTC is released (T-disconnect of the transport layer).
- The network connection is released (N-disconnect of the link layer). For a switched line, the physical connection to the SWIFT network is released.

Use the command to log out one master logical terminal, or, if you are authorized by your installation, all master logical terminals on one line or on all lines to the SWIFT network.

When one line of the SWIFT link is shared by several financial institutions with several master logical terminals, a **logout** command must be entered for each master logical terminal that wants to end the communication with the SWIFT network.

Note: You must be authorized to use the **logout** command.

Command Format

The format of the **logout** command is:

logout **lo**	{[*ltname* | ALL], [*line* | **ALL**] [,*timeday*]}

Parameter Descriptions

The parameters for this command have the following meanings:

ltname
Is the 9-character name of a master logical terminal.

This name must be:

- A valid logical terminal name that has been given to the financial institution by S.W.I.F.T.
- Defined in the Logical Terminal Table DWSLTT as a master LT.

If you have the appropriate authorization level, you can use any master logical terminal; otherwise you can use only the master logical terminal that matches the first 9 characters of the origin identification in your User File record.

The default for this parameter depends on whether you use the system console to enter the **logout** command:

- If you use the system console, the first master logical terminal in the Logical Terminal Table DWSLTT is taken.
- If you do not use the system console, the first 9 characters of the origin identification of your User File record are taken.

The master logical terminal's status must be one of the following (the parentheses show the status in response to the **dl** command):

Session key pending (SK PND)
The master logical terminal is immediately set to LOGOUT status.

Session key proceeding (SK PRC)
The master logical terminal is immediately set to LOGOUT status.

Login pending (LIN PND)
The master logical terminal is immediately set to LOGOUT status.

Logged in (LOGIN)
The master logical terminal is set to LOGOUT PENDING status, and the logout message (APDU 06) is eventually generated and sent to the SWIFT network.

For any other status, the **logout** command is rejected.

ALL,[*line* **| ALL]**
If you use the parameter ALL,*line* or ALL,ALL (ALL is used as default if the second parameter is not specified), all master logical terminal assigned to all lines or to the specified line are treated according to their status (the parentheses show the status in response to the **dl** command):

Session key pending (SK PND)
The master logical terminal is immediately set to LOGOUT status.

Session key proceeding (SK PRC)
The master logical terminal is immediately set to LOGOUT status.

Login pending (LIN PND)
The master logical terminal is immediately set to LOGOUT status.

Logged in (LOGIN)
The master logical terminal is set to LOGOUT PENDING status, and the logout message (APDU 06) is eventually generated and sent to the SWIFT network.

For any other status, there is no change for the master logical terminal.

timeday
Specifies the *timeday* value for the field 173 of the logout message in the format *ddhhmm*:

dd Day from 1 to 31

hh Hour from 0 to 23

mm Minute from 0 to 59.

If you do not specify the *timeday* parameter, the logout message is sent without the text block (that contains field 173 as the only field). If you specify the *timeday* parameter, the logout message is sent with the text block and field 173.

If the FIN application of a master logical terminal is processed as for a **quit** command, the *timeday* parameter of the **logout** command is also used for the quit.

Command Examples

This section shows some examples of how to enter the **logout** command.

Example 1: This example shows the **logout** command for a specific master logical terminal:

```
logout,vndebet2a
```

Example 2: Enter the following command to set the status of all master logical terminals assigned to line 2 to LOGOUT or LOGOUT PENDING, depending on their current settings:

```
lo,all,2
```

Example 3: This example shows the abbreviation of the command, where all master logical terminals of all lines (the default is used) are set to LOGOUT or LOGOUT PENDING status, and with the *timeday* parameter to allow the next login only after the day 12, 8 o'clock in the morning:

```
lo,all,,120800
```

Example of the Display from a LOGOUT Command

Ending the Connection to the SWIFT Network (LOGOUT) shows an example of a panel displayed in response to a **logout** command with the ALL parameter for all lines. Use the **dm last** command to display how the master logical terminals are logged out (refer to Figure 42).

```
                        Operator Command Processing

> LO ALL
  DWS516I LOGOUT ALL accepted for all lines









  115131   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 41. Logout of All Master Logical Terminals on All Lines

```
                        Operator Command Processing

> DM LAST
   DSL075I Display Message 1999MAY28
          ...
          ...
          ...
          ...
          ...
          ...
   082015 DWS606I VNDEBET2A LOGOUT proceeding on line 1
   082016 DWS606I VNDEBET2B LOGOUT proceeding on line 2
   082030 DWS607I VNDEBET2A LOGOUT successful on line 1
   082031 DWS607I VNDEBET2B LOGOUT successful on line 2
   082037 DWS651I CBT disconnect is complete on line 1
   082038 DWS651I CBT disconnect is complete on line 2
   082041 DWS643I Network connection terminated (T,0) on line 1
   082042 DWS643I Network connection terminated (T,0) on line 2

  115333   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 42. Completion of Logout of All Master Logical Terminals on All Lines

Quitting FIN Applications (QUIT)

Use the command **quit** to quit SWIFT FIN applications (which must have been defined in the Logical Terminal Table DWSLTT). When the FIN application is successfully selected after a **select** command, and there are no acknowledgment messages outstanding for messages sent to the SWIFT network for this FIN application (ISN acknowledgments), the **quit** command generates a SWIFT quit message (APDU 05) and sends it to the SWIFT network.

When the quit process is complete, the following is done for this FIN application:

- The association of the FIN application is released (disassociation of the AI layer).
- The transport connection of the FIN application is released (T-disconnect of the transport layer).

You use the command to quit one FIN application, or, if you are authorized by your installation, all FIN applications on one line or on all lines to the SWIFT network.

When one line of the SWIFT link is shared by several financial institutions with several FIN applications, a **quit** command must be entered for each FIN application that wants to end the communication with the SWIFT network.

The functions of the **quit** command are included in the functions of a **logout** command.

Note: You must be authorized to use the **quit** command.

Command Format

The format of the **quit** command is:

quit **q**	{[*ltname* | ALL], [*line* | **ALL**] [,*timeday*]}

Parameter Descriptions

The parameters for this command have the following meanings:

ltname | **ALL**
Is the 9-character name of the master logical terminal whose FIN application is to be quit.

This name must be:

- A valid logical terminal name that has been given to the financial institution by S.W.I.F.T.
- Defined in the Logical Terminal Table DWSLTT as a master LT.
- A FIN application must be defined for this logical terminal in DWSLTT.

If you have the appropriate authorization level, you can use any master logical terminal; otherwise you can use only the master logical terminal that matches the first 9 characters of the origin identification in your User File record.

The default for this parameter depends on whether you use the system console to enter the **quit** command:

- If you use the system console, the first master logical terminal in the Logical Terminal Table DWSLTT is taken.
- If you do not use the system console, the first 9 characters of the origin identification of your User File record are taken.

The FIN application's status must be one of the following (the parentheses show the status in response to the **dl** command):

Session key pending (SK PND)
The FIN application is immediately set to CLOSED status.

Session key proceeding (SK PRC)
The FIN application is immediately set to CLOSED status.

Select pending (SEL PND)
The FIN application is immediately set to CLOSED status.

Open (OPEN)
The FIN application is set to QUIT PENDING status, and the quit message (APDU 05) is eventually generated and sent to the SWIFT network.

For any other status, the **quit** command is rejected.

If you use the parameter ALL in the place of the *ltname* parameter, the FIN applications of all master logical terminals of all lines or of the specified line are treated depending on their status (the parentheses show the status in response to the **dl** command):

Session key pending (SK PND)
The FIN application is immediately set to CLOSED status.

Session key proceeding (SK PRC)
The FIN application is immediately set to CLOSED status.

Select pending (SEL PND)
The FIN application is immediately set to CLOSED status.

Open (OPEN)
The FIN application is set to QUIT PENDING status, and the quit message (APDU 05) is eventually generated and sent to the SWIFT network.

For any other status, there is no change for the FIN application.

line | **ALL**
If you use the parameter ALL,*line* or ALL,ALL, all FIN applications of the master logical terminals assigned to all lines or to the specified line are treated according to their status. If the second parameter is not specified, all lines are processed as default.

timeday
Specifies the *timeday* value for the field 173 of the quit message in the format *ddhhmm*:

dd Day from 1 to 31

hh Hour from 0 to 23

mm Minute from 0 to 59.

If you do not specify the *timeday* parameter, the quit message is sent without the text block (that contains field 173 as the only field). If you specify the *timeday* parameter, the quit message is sent with the text block and field 173.

Command Examples

This section shows some examples of how to enter the **quit** command.

Example 1: This example shows the **quit** command for the FIN application of a specific master logical terminal:

```
quit,vndebet2a
```

Example 2: This example shows the **quit** command for the FIN applications of all master logical terminals on line 1:

```
quit,all,1
```

Example 3: This example shows the abbreviation of the **quit** command and how to quit all FIN applications on all lines:

```
q,all
```

Example 4: This example shows the abbreviation of the command without the *ltname*, the *line* parameters, but with the *timeday* parameter to allow the next select only after the day 12, 8 o'clock in the morning:

```
q,,,120800
```

Example of the Display from a QUIT Command

Figure 43 shows an example of a panel displayed in response to the **quit** command. Figure 44 displays the response of a **dm last** command to show how the FIN applications are closed.

```
                        Operator Command Processing

> Q,ALL
  DWS534I QUIT ALL FIN accepted for all lines









  115738   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 43. Quit All FIN Applications on All Lines

```
                        Operator Command Processing

> DM LAST
  DSL075I Display Message 1999MAY28
          ...
          ...
          ...
          ...
          ...
          ...
          ...
  081915 DWS616I VNDEBET2A FIN QUIT proceeding on line 1
  081916 DWS616I VNDEBET2B FIN QUIT proceeding on line 2
  081930 DWS617I VNDEBET2A FIN QUIT successful on line 1
  081937 DWS617I VNDEBET2B FIN QUIT successful on line 2

  115733   is the time of this display



Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 44. Completion of Quit of All FIN Applications on All Lines

Selecting a FIN Application (SELECT)

Use the **select** command to select the FIN application for a master logical terminal. Both the master logical terminal and the FIN application must be defined in the Logical Terminal Table DWSLTT.

The **select** command for a FIN application is only accepted if its master logical terminal is in session key pending; in session key proceeding; in LOGIN pending; in LOGIN proceeding; or in LOGIN status, and the FIN application must be in CLOSED status.

The **select** command generates a SWIFT select message (APDU 03) and sends it to the SWIFT network.

If the select message is accepted by the SWIFT network, the transport connection and the association of the FIN application of this master logical terminal are established.

When this line of the SWIFT link is shared by several financial institutions with several master logical terminals, a **select** command must be entered for each FIN application that wants to communicate with the SWIFT network.

Note: You must be authorized to use the **select** command.

Command Format

The format of the **select** command is:

select **se**	[*ltname*],[*ssn*],[*sk1*], [*sk2*],[*window*],[*state*][,*delivery*]

Parameter Descriptions

Note: If one of the above parameters is omitted, but one of them to the right of it is specified, all the commas must be specified up to the last parameter used.

The parameters for this command have the following meanings:

ltname
Is the 9-character name of a master logical terminal whose FIN application you want to select.

This name must be:

- A valid logical terminal name that has been given to the financial institution by S.W.I.F.T.
- Defined in the Logical Terminal Table DWSLTT as a master LT.
- A FIN application must be defined for this logical terminal in DWSLTT.

If you have the appropriate authorization level, you can use any master logical terminal; otherwise you can use only the master logical terminal that matches the first 9 characters of the origin identification in your User File record.

The default for this parameter depends on whether you use the system console to enter the **select** command:

- If you use the system console, the first master logical terminal in the Logical Terminal Table DWSLTT is taken.
- If you do not use the system console, the first 9 characters of the origin identification of your User File record are taken.

ssn

Is the select sequence number (SSN) required by the SWIFT network for the select message. The number you use must be in the range from 0 to 9999. The current select sequence number is used as the default for this parameter.

It is only necessary to enter a value for this parameter after the first MERVA ESA startup or when S.W.I.F.T. has reset the number. The default can be used at all other times.

If SLS is used and you enter the *ssn* parameter, and there is a session key available in the DWSLTT entry for this FIN application, this session key is erased even if you enter the same SSN as already contained in the DWSLTT entry. This forces the SWIFT Link to get a new session key, for example, after a change of the ICC parameters.

sk1

Is one of the following:

- If paper tables are used for select, this parameter is the first 4 digits of the session key required for the select sequence number. It is used to calculate the user authentication field (field 501) for the select message.

  You need only enter this parameter if the Login Authorization Table (DWSLOG2) is not used in your installation of the SWIFT link. If DWSLOG2 is included, and you enter a value for this parameter, it overrides the value supplied by DWSLOG2. This parameter is mandatory, if you do not have DWSLOG2 or the actual SSN is not defined in DWSLOG2.
- If secure login/select (SLS) is used, this parameter is the 32 characters session key from the card reader when you use unconnected mode. If connected mode or preloaded session keys are used, you need not enter the session key here. Refer to "Maintenance of Session Keys for Login and Select" on page 122 for details on preloading session keys.
- You can specify AUTO for this parameter if you want an automatic repetition of select after failures. This requires preloaded session keys or a connection to the USE workstation when SLS is used, or DWSLOG2 containing the session keys when paper tables are used.

sk2

Is one of the following:

- If paper tables are used for select, this parameter is the second 4 digits of the session key required for the select sequence number. It is used to calculate the user authentication field (field 501) for the select message. The same rules apply as for the *sk1* parameter for using paper tables.
- If secure login/select (SLS) is used, this parameter is the 4-character check value for the session key specified with the *sk1* parameter from the card reader when you use unconnected mode.
- If you have specified AUTO for the *sk1* parameter, you can specify a retry count from 1 to 9 for the automatic repetition of select after failures with the *sk2* parameter. If you enter a value greater than 9, only the first digit is used. If you do not specify this parameter or you specify an incorrect value, the default of 3 is used.

window

Is the window for this session of the FIN application for the field 110 of the

select message. The number entered must be in the range from 1 to 999. The default for this parameter is the window size specified in the Logical Terminal Table DWSLTT.

Usually the SWIFT network accepts only a value of 10 for the window in the select acknowledgment (APDU 23).

state

Specifies the select state and the LT-directed queue value. You can specify 3 characters, each of them being either N or Y:

- The first 2 characters indicate the select state for field 204 as defined by S.W.I.F.T.
- The third character indicates the LT-directed queue value for field 208.

The default for the parameter is YYY.

A specification of YNY is incorrect for this parameter (input only and Yes for LT-directed queue, see the *S.W.I.F.T. User Handbook*), it is changed to YNN by MERVA ESA.

delivery

Specifies the delivery subset lines for field 338. Each delivery subset specification must be 6 characters long, and you can specify up to 10 delivery subset mnemonics separated by commas (like the other parameters). If you need more than 10 delivery subset mnemonics or more than fit into the command line, you must use the **dds** command to set the delivery subset mnemonics before you use the **select** command.

Each delivery subset must have been specified by a message type 047 before you can use it in a **select** command. The SWIFT link cannot check that you have defined the delivery subsets properly.

The delivery subset mnemonics specified in the **select** command replace the specification done in DWSLTT for this FIN application, or replace the specifications done in an earlier **dds** or **select** command and saved in DWSLTT. Replacing means that the specification of only one delivery subset mnemonic removes **all** saved (that is, up to 30) delivery subset mnemonics in DWSLTT and saves only the one (or more) specified in this **select** command.

This technique allows for:

- Generating the delivery subset mnemonics in DWSLTT if you use the same mnemonics every day. Then you need not specify the mnemonics in the **select** command.
- Changing them in a **dds** or **select** command if necessary, and using the changed mnemonics in subsequent **select** commands, without specifying them again until another change is necessary.

After a change, the delivery subset mnemonics generated in DWSLTT are available again after restarting CICS, or MERVA ESA in IMS.

The field 338 of the select message is generated with the delivery subset mnemonics available in DWSLTT (either generated or saved from a **dds** command, or from an earlier or from this **select** command). If no delivery subset mnemonics are found, the select message is generated without field 338.

Note: You must not specify two consecutive commas between two delivery subset names. If you do so, the evaluation of these parameters stops at the second comma, and all delivery subset names after the second comma are ignored.

Command Examples

This section shows some examples of how to enter the **select** command.

Example 1: This example shows the **select** command with the following parameters:

- *ltname* - VNDEBET2A
- *ssn* - 111
- *sk1* - is omitted
- *sk2* - is omitted
- *window* - 10
- *state* - YYY
- *delivery* - URGENT,SYSTEM,NORMAL

The format of the command is:

```
select,vndebet2a,111,,,10,yyy,urgent,system,normal
```

The tree commas after the *ssn* parameter (111) mean that the parameters *sk1* and *sk2* are not entered in the command.

Example 2: This example shows the abbreviation for the **select** command (**se**) with the following parameters when using paper tables:

- *ltname* - VNDEBET2A
- *ssn* - 44
- *sk1* - 0555
- *sk2* - 0666
- *window* - is omitted
- *state* - is omitted
- *delivery* - is omitted

The format of the command is:

```
se,vndebet2a,44,0555,0666
```

The parameters after *sk2* are omitted. As they are all at the end of the parameter list, no commas are needed. The defaults are taken.

Example 3: This example shows the abbreviation for the command without any parameters. The defaults for all the parameters are taken for this command:

```
se
```

Example 4: This example shows the abbreviation for the command with only the *delivery* parameter specified. The 7 commas show the omission of all the other parameters, and that the defaults are to be used:

```
se,,,,,,,urgent
```

Example 5: This example shows, when SLS unconnected mode is used, how to enter the **select** command with the 32 character session key in the *sk1* parameter and the 4 character check value in the *sk2* parameter:

```
se,,,967532247e7cd4d3b815844002ec262f,1668
```

The three commas after the command show that the *ltname* and the *ssn* parameters have not been entered, and the defaults are used.

Example of the Display from a SELECT Command

Selecting a FIN Application (SELECT) shows an example of a panel displayed in response to the **select** command.

```
                        Operator Command Processing

> SELECT VNDEBET2A
  DWS521I SELECT VNDEBET2A FIN accepted, select pending









  115932   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 45. Select the FIN Application of a Master Logical Terminal

Setting Parameters for a Master Logical Terminal (SETLT)

Use the command **setlt** to display or set the following for a master logical terminal and its FIN application:

- The line number
- The technology flag for using paper tables or secure login/select (SLS)
- The integrated circuit card (ICC) parameters whitelist flag, kernel version, and ICC set number
- The name of the USE workstation used for SLS.

Notes:

1. You must be authorized to use the **setlt** command.
2. This command can be used to set the above parameters only if the master logical terminal is in LOGOUT status.

Command Format

The format of the **setlt** command is:

setlt **slt**	*ltname*[,*line*[,*tflag*[,*iccparm*[,*usename*]]]]

Parameter Descriptions

The parameters for this command have the following meanings:

ltname
Is the 9-character name of a master logical terminal. This name must be defined in the Logical Terminal Table DWSLTT as a master logical terminal.

line
Defines the number of the line to be used for the next login of the specified master logical terminal to the SWIFT network. The value entered is used instead of the line originally specified in the Logical Terminal Table DWSLTT for this master logical terminal.

The value entered must be a number from 1 to 30. The number refers to the name of a line definition module, for example, DWSLIN1 for line 1, DWSLIN15 for line 15. If the number you enter is wrong (that is, no line definition module for the number exists), the command is rejected.

tflag
Defines the technology flag for the login and select.

PT specifies to use paper tables.

SLS specifies to use secure login/select.

You must have agreed the change of the technology with S.W.I.F.T.

iccparm
Sets the ICC parameters when SLS is used. You must always specify 10 digits in the following order:

- 2 digits for the whitelist flag
- 2 digits for the kernel version
- 4 digits padding zeros ('0000')
- 2 digits for the ICC set number.

This is the same format as received from SWIFT in the field 502 of the login acknowledgment (LAK or LNK) or select acknowledgment (SAK or SNK).

The ICC parameters are set automatically when you use SLS connected mode for the first time, or when the session keys are pregenerated in the USE workstation and sent to MERVA ESA for loading into the session key queues defined in DWSLTT.

You change the ICC parameters after having received a negative login acknowledgment (APDU 42) with the reason code L34 (incorrect MAC), and after having agreed with S.W.I.F.T. which parameters to use.

usename
Defines the name of the USE workstation when SLS is used. You specify 1 to 9 characters that follow the rules defined for S.W.I.F.T. addresses. If you specify less than 9 characters, the name is padded with characters X.

This name is used when using SLS connected mode for the routing of a single session key request to a MERVA Link send queue for sending to the USE workstation. A single session key request is a SWIFT message type 999 with a special format handled by the SWIFT Link programs. The name of the USE workstation is used in the application header as destination address and can be evaluated during the routing of this message.

The parameters *line*, *tflag*, *iccparm* and *usename* can be specified in any order in the parameter positions 2 to 5 of the **setlt** command. They are evaluated according to the following rules:

- One or two digits from 1 to 30 is a line number of the *line* parameter.
- The characters PT or SLS are the technology flag of the *tflag* parameter.
- Ten digits are the ICC parameters of the *iccparm* parameter.
- Up to nine characters that follow the rules of a SWIFT address are the *usename* parameter.
- Anything else is a parameter error which causes rejection of the whole command.

If a parameter is not specified, the value in DWSLTT is not changed.

If the technology flag PT is set, the parameters referring to SLS are not used during the login and select process.

A change of the technology flag, the ICC parameters and the USE workstation name is maintained until it is changed again with a **setlt** command, even when the SWIFT Link is stopped and started again.

Command Examples

This section shows examples of how to enter the **setlt** command.

Example 1: The command is entered with its abbreviation and sets the line 2 for the master logical terminal VNDEBET2A:

```
slt,vndebet2a,2
```

Example 2: The command is entered with the following parameters:

- *ltname* - VNDEBET2A
- *line* - 3

- *tflag* - SLS
- *iccparm* - 0101000001
- *usename* - USEMERVA2

The following formats of the command can be used (or any other order of the last 4 parameters):

```
slt,vndebet2a,3,SLS,0101000001,usemerva2
slt,vndebet2a,usemerva2,0101000001,sls,3
```

Example 3: The command is entered to set the technology flag:

- *ltname* - VNDEBET2A
- *line* - is omitted
- *tflag* - SLS
- *iccparm* - is omitted
- *usename* - is omitted

The format of the command is:

```
slt,vndebet2a,SLS
```

Example of the Display from a SETLT Command

Setting Parameters for a Master Logical Terminal (SETLT) shows an example of a panel displayed in response to a **setlt** command.

```
                      Operator Command Processing

> SLT VNDEBET2A,SLS
  DWS572 SETLT VNDEBET2A accepted
  Line=1 Line not initialized
  LT name   AP name  LSN  SESS  ISN    OSN   MXW ISW OSW status  SS Q AS
  VNDEBET2A GPA      0000 0000 000000 000000 000 000 000 LOGOUT       DC
  VNDEBET2A FIN      0000 0000 000000 000000 000 000 000 CLOSED       DC

                      actual            original
  Technology flag    :  SLS               PT
  USE WorkstationName:  USEMERVA2         USEMERVA2

                      in login/select   from SWIFT
  Whitelist flag     :  01                01
  ICC kernel version :  01                01
  ICC set number     :  000001            000001


  115133   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 46. Setting or Displaying Parameters for a Master Logical Terminal

Note: If the **setlt** command is entered at the operating system console, the response does not show the lines starting with LT name and VNDEBET2A. To see the information contained in these lines, you must enter the **dl** command.

Routing SWIFT Link Commands for Parallel Processing (SWIFTII)

Use the command **swiftii** only when parallel processing is used for the MERVA ESA SWIFT Link. Parallel processing allows the use of multiple SWIFT Link servers under a single MERVA ESA nucleus. The alternative SWIFT Link servers are identified in the nucleus program table as SWIFTIIx, where the x stands for a letter from A to Z. Each SWIFT Link server runs separately from the other servers. Each server has its own logical terminal table (DWSLTTx); its name is specified as the program start parameter.

For details, refer to the *MERVA for ESA Customization Guide*.

The operator commands issued to control the other servers must be routed to the appropriate subtask. The command **swiftii** or its alias **sw** is used to enter a command that is directed to another SWIFT Link server.

The first parameter is a letter indicating the requested server. Alternatively, MERVA ESA defines several alias commands to address the servers directly. The command code **swa** directs a SWIFT Link command to the SWIFTIIA server, the command code **swb** uses the SWIFTIIB server.

Note:

You must be authorized to use the authorized SWIFT Link commands.

Command Format

The format of the **swiftii** command is:

swiftii **sw**	*x,SWIFT Link command string*

Parameter Descriptions

The parameters for this command have the following meanings:

x The letter indicates the descriptive name of the requested server. The descriptive name is SWIFTIIx, where the suffix x stands for the letter entered as the first parameter. This descriptive name must be defined in the nucleus program table (DSLNPTT) and in the nucleus server table (DSLNSVT). The program must be active.

SWIFT Link *command string*
The command code and the command parameters that should be executed by the indicated SWIFT Link server. All SWIFT Link operator commands can be routed using this command, except the **diva** and the **xtrace** command. The commands **diva** and **xtrace** are not dependent on a specific SWIFT Link server, they process information about X.25 line numbers.

Command Examples

This section shows examples of how to enter the **swiftii** command.

Example 1:

The command is used to control the SWIFT Link server with the descriptive name SWIFTIIA. In this example the logical terminal parameters are set. The line number to be used for this LT is 11.

```
sw a,slt,vndebet2a,11,SLS,0101000001,usemerva2
```

Alternatively, the command can be entered as

```
swa slt,vndebet2a,11,SLS,0101000001,usemerva2
```

The command code **swa** is used to indicate the SWIFT Link server with the descriptive name SWIFTIIA.

Example 2:

A login command for the master logical terminal VNDEBET2A is executed. The command code **swa** is used to indicate the SWIFT Link server with the descriptive name SWIFTIIA.

```
sw a,li,vndebet2a
```

Alternatively, the command can be entered as:

```
swa li,vndebet2a
```

Example 3:

To display the line and logical terminal status for a specific SWIFT Link server, the appropriate command parameter must be used. All active logical terminals of this SWIFT Link server are displayed.

```
sw a,dla
```

Alternatively, the command can be entered as:

```
swa dla
```

Examples of the display from the **swiftii** command are not provided, because the display is identical to the display of the executed command.

Maintenance of Session Keys for Login and Select

When secure login/select (SLS) is used, you can pregenerate session keys at the USE workstation and send them to MERVA ESA for preloading into session key queues. The session key queues are defined in the Logical Terminal Table (DWSLTT) and in the MERVA ESA Function Table (DSLFNTT). Refer to the *MERVA for ESA Macro Reference* for details on these definitions.

You can pregenerate as many session keys at the USE workstation at one time as you want. For example, you pregenerate as many as you need for one week, one month, or one year, or you pregenerate the 10000 that are available for one set of whitelist flag, kernel version, and set number of the integrated circuit cards (ICCs) used for SLS.

After the pregeneration is finished, you can send the session keys from the USE workstation to MERVA ESA with MERVA Link. If necessary, you can send the same session keys more than once as MERVA ESA ensures that the same session key is stored once only.

Refer to the *MERVA USE Administration Guide* for details on how to pregenerate session keys and send them to MERVA ESA.

The session keys are received in MERVA ESA and routed to the queue defined with the LSKQUE parameter of the DWSPARM macro (see *MERVA for ESA Macro Reference* for details). The program DWSDLSK gets the session keys from this queue and puts them into the session key queues defined in DWSLTT.

During the transmission and the storage in the queues, the session keys are encrypted with the SWIFT ST-1 algorithm.

The session keys in the session key queues cannot be displayed as they are not stored in the MERVA ESA TOF format. Instead, the maintenance of the session keys is done using the queue keys and the queue key list.

The queue keys of the session keys contain the following information:

- Key 1:
  - 9 characters for the logical terminal name
  - 1 character for the application name (G for GPA and F for FIN)
  - 1 blank for separator
  - 2 digits for the whitelist flag
  - 2 digits for the ICC kernel version
  - 2 digits for the ICC set number. The 4 padding zeros left of the ICC set number are not used.
  - 1 blank for separator
  - 4 digits for the LSN or SSN.
- Key 2 contains the same information in a different order to allow for the retrieval of session keys when the ICC parameters whitelist flag, kernel version, and set number are not known to the SWIFT Link (these parameters are not specified during the retrieval but are nevertheless part of key 2):
  - 9 characters for the logical terminal name
  - 1 character for the application name (G for GPA and F for FIN)
  - 1 blank for separator

- 4 digits for the LSN or SSN
- 1 blank for separator
- 2 digits for the whitelist flag
- 2 digits for the ICC kernel version
- 2 digits for the ICC set number. The four padding zeros to the left of the ICC set number are not used.

With this information in the queue keys, you can maintain the session keys by selecting the session key queue from the MERVA ESA function selection panel and using the MERVA ESA queue list (refer to the *MERVA for ESA User's Guide* for details). You can:

- Display the keys to see for which LSNs and SSNs the session keys are available, and you can pregenerate more session keys at the USE workstation and send them to MERVA ESA in time.
- Delete session keys that are no longer needed using the user command **delete** described in the *MERVA for ESA User's Guide*. Under normal circumstances, session keys are deleted automatically after login or select is successful, as the session key for this LSN/SSN and the previous ones are never used again. If this automatic deletion does not work for any reason, you must delete them.

  You can delete one session key with the **delete** *qsn* command, where *qsn* is the queue sequence number (QSN) of the session key that you want to delete. The QSN is shown in the queue list.

  You can delete a range of session keys with the **delete** *qsn1*,*qsn2* command, where *qsn1* is the QSN of the first session key of the range you want to delete, and *qsn2* is the QSN of the last session key of the range you want to delete. After the deletion, you are informed how many session keys have been deleted.

Figure 47 shows an example of a queue key list of the session key queue SLSFIN for the FIN application of the master logical terminal VNDEBET2A.

```
                          Queue Key List                    Func SLSFIN
                                                            Wait 00000200

  Key 1:                  Key 2:                        QSN        RBN
> VNDEBET2AF 010101 0031  VNDEBET2A 0031 010101        0000000166 00015
> VNDEBET2AF 010101 0032  VNDEBET2A 0032 010101        0000000167 00015
> VNDEBET2AF 010101 0033  VNDEBET2A 0033 010101        0000000168 00015
> VNDEBET2AF 010101 0034  VNDEBET2A 0034 010101        0000000169 00015
> VNDEBET2AF 010101 0035  VNDEBET2A 0035 010101        0000000170 00015
> VNDEBET2AF 010101 0036  VNDEBET2A 0036 010101        0000000171 00015
> VNDEBET2AF 010101 0037  VNDEBET2A 0037 010101        0000000172 00015
> VNDEBET2AF 010101 0038  VNDEBET2A 0038 010101        0000000173 00015
> VNDEBET2AF 010101 0039  VNDEBET2A 0039 010101        0000000174 00015
> VNDEBET2AF 010101 0040  VNDEBET2A 0040 010101        0000000175 00015



To select a message, move cursor to ">" and press PF4 (Get QSN)


Select only  KEY 1: ____________________________ KEY 2: ____________________________

Command =====>
PF 1=Help      2=Retrieve  3=Return     4=Get QSN    5=Get Next    6=Get First
PF 7=List Back 8=List Fwd  9=Hardcopy  10=List Last 11=List First 12=List Off
```

Figure 47. Displaying the Queue Keys of Preloaded Session Keys

This example shows under the heading Key 1:

- The 9-character logical terminal name VNDEBET2A
- The 1-character application name F indicating that the session keys are for the select sequence numbers (SSNs) of the FIN application
- After the separating blank, 6 digits 010101 for the ICC parameters: 01 for the whitelist flag, 01 for the ICC kernel version, and 01 for the ICC set number
- After the separating blank, 4 digits for the select sequence numbers 0031 to 0040 that are stored in this queue.

Under the heading Key 2, you see the same information with the select sequence numbers shown before the ICC parameters.

In the second line on the right, Wait 00000200 indicates that there are 200 session keys stored in this queue, and you can page forward and backward in the queue list to see for which SSNs they are.

Chapter 6. Operating the Telex Link

In MERVA ESA there are two ways to communicate with the public telex network:

- The Telex Link via a fault-tolerant system (for example, the Series/1) using the Telex Interface Program.
- The Telex Link via a workstation using the Telex Link attachment of MERVA ESA components.

The Telex Link via a workstation is controlled by the MERVA Link in order to transfer the telex messages from MERVA ESA to the MERVA ESA components.

The commands available for managing the Telex Link via a fault-tolerant system described in this chapter do not apply to the workstation based telex functions. The following similar functions are provided:

Telex Link via a Fault-Tolerant System	**workstation based telex functions Alternative**
TXDISP	The MERVA System Control (MSC) function provides information on the state of the MERVA Link connection to the workstation.
TXON	Initiating the TelexBox connection can be done only from the workstation. In MERVA ESA, the MERVA Link AOPEN and ASTART commands can be used to initiate the transfer of messages to a workstation.
TXOFF	Terminating the TelexBox connection can be done only from the workstation. In MERVA ESA, the MERVA Link HOLD command can be used to suspend transfer of messages to a workstation. The MERVA Link RECOVER command can recover Telexes from a stalled MERVA Link application.

The rest of this chapter applies only to the operation of the Telex Link via a fault-tolerant system.

Operating the Telex Link via a Fault-Tolerant System

The Telex Link always runs under the control of, and together with, MERVA ESA. There are two ways to start the Telex Link:

- Automatically, during the startup of MERVA ESA
- Using the MERVA ESA command **start telex** (**telex** is the descriptive name of ENLSTP, the main program of the Telex Link).

After successful startup, the Telex Link issues the message:

```
ENL900I Telex Link startup successful.
```

In case of a startup failure, an error message is issued.

There are also two ways to stop the Telex Link:

- Using the MERVA ESA command **stop telex**

- Automatically, during the termination of MERVA ESA.

To find out whether the Telex Link is active, enter the MERVA ESA command **dp** (refer to "Displaying the Program Status (DP)" on page 36).

The transmission of telex messages between the Telex Link and the Telex Interface Program requires the establishment of a session between them. For this purpose, the Telex Link provides the following operator commands:

txon To sign on a session between the Telex Link and the Telex Interface Program

txdisp [recover]
To monitor the status of the session and to request that the last outgoing telex message to the Telex Interface Program be resent if this message is not acknowledged by the Telex Interface Program within the time specified by the RTIM parameter of the ENLPARM macro

txoff To sign off the current session.

The Telex Link operator commands can be used only if the Telex Link is active.

The commands in this chapter are presented in alphabetical order.

Monitoring the Telex Link Session (TXDISP)

Use the command **txdisp** to display the status of the session between the Telex Link and the Telex Interface Program.

Command Format

The format of the **txdisp** command is:

txdisp	[RECOVER]

Parameter Descriptions

The parameter for this command has the following meaning:

RECOVER
If the response to the command **txdisp** shows that the Telex Link has already waited too long for the logical acknowledgment of a telex message sent to the Telex Interface Program, you can use the command **txdisp recover** to make the Telex Link send the message to the Telex Interface Program again. The Telex Link does so only if the waiting time for the logical acknowledgment is greater than the time specified in the RTIM parameter of the ENLPARM macro. Resending can help to resume normal message traffic in the current session with the Telex Interface Program.

Example of the Display from a TXDISP Command

Figure 48 shows an example of the response to the **txdisp** command.

```
                      Operator Command Processing

> TXDISP
  ENL944I Session Status Information
     Name     SESS SENT RCVD Pending  Status
     TELEX    0329 0001 0020          ACTIVE
     TXIP     0329 0021 0001 00:00:39 0320 CHECKING OUTGOING TELEX









   115033   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 48. Displaying the Session Status

Message ENL944I provides the following information (for details refer to *MERVA for ESA Messages and Codes*):

Name The response line starting with TELEX shows the status of the

session as seen by the Telex Link. The response line starting with TXIP shows the latest status of the session as seen by the Telex Interface Program, and reported to the Telex Link. The Telex Interface Program sends a status report to the Telex Link every two minutes.

SESS Gives the 4-digit session number of the current session.

SENT The number of telex messages that have been sent from the Telex Link to the Telex Interface Program (in the TELEX line), or that have been sent from the Telex Interface Program to the Telex Link (in the TXIP line) during the current session.

RCVD In the TELEX line, the number of messages sent from the Telex Interface Program to the Telex Link. Usually this number is equal to the number under the heading SENT in the TXIP line. Differences in these numbers are caused by the delayed status of the Telex Interface Program or because a message is still on the line. In the TXIP line, the number of messages sent from the Telex Link to the Telex Interface Program.

Pending Is the time in *hhmmss* format when one of the following is pending:

- In the **TELEX** line, if the time is present, it shows how long the Telex Link has been waiting for a logical acknowledgment from the Telex Interface Program.
- In the **TXIP** line, it shows how much time has passed since this status report was received from the Telex Interface Program.

Status Refer to *MERVA for ESA Messages and Codes* for details on the status. The Telex Link status is described in message ENL944I, the Telex Interface Program status is described in the *MERVA for ESA Installation Guide*.

Signing Off the Session with the Telex Interface Program (TXOFF)

Signing off the session with the Telex Interface Program can be done in one of the following ways:

- Automatic sign-off when the Telex Link is stopped.
- Sign-off by the command **txoff**.

Command Format

The format of the command is:

txoff	

There are no parameters for this command.

After you have entered the **txoff** command, the Telex Link:

- Stops sending messages (outgoing telex messages or acknowledgments) to the Telex Interface Program.
- Processes a message received before the sign-off acknowledgment, but does not acknowledge it. The Telex Interface Program sends this message again at the beginning of the next session, but the Telex Link recognizes it as being a duplicate.
- Does not process a received transmission acknowledgment. This acknowledgment is sent again by the Telex Interface Program in the next session.
- Processes any invalid data received as described in the *MERVA for ESA User's Guide*.

The Telex Link session status first changes to SIGNOFF PENDING and, upon receiving the sign-off acknowledgment from the Telex Interface Program, then changes to NOT ACTIVE.

The Telex Link issues the following message to the operators informing them of the sign-off:

```
ENL905I TELEX   signoff, sess=0329, sent=00011, rcvd=00012
```

Example of the Display from a TXOFF Command

Figure 49 on page 130 shows an example of the display in response to a **txoff** command.

TXOFF Command

```
                        Operator Command Processing

> TXOFF
  ENL941I TELEX   signoff accepted









  115033   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF          5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP         11=DQ         12=DL
```

Figure 49. Signing Off the Session with the Telex Interface Program

Signing On the Session with the Telex Interface Program (TXON)

Signing on the session with the Telex Interface Program can be done in one of the following ways:

- Automatic sign-on when the Telex Link has been started, if the AUTO=YES parameter is specified in the ENLPARM macro.
- Sign-on by the command **txon**.

Command Format

The format of the **txon** command is:

txon	

There are no parameters for this command.

Signing on the session with the Telex Interface Program is only successful when:

- The Telex Substation on which the Telex Interface Program operates is switched on.
- The communication line to this Telex Substation is active.
- The Telex Link has been started.

Example of the Display from a TXON Command

Figure 50 shows the panel after the Telex Link has accepted your sign-on request. The message ENL940I confirms that the Telex Link has accepted the command.

```
                        Operator Command Processing

> TXON
  ENL940I TELEX    signon accepted

  115033   is the time of this display

Command =====>
PF 1=Help      2=Repeat    3=Return     4=DF         5=DU          6=DM Last
PF 7=          8=          9=Hardcopy  10=DP        11=DQ filled  12=DL
```

Figure 50. Signing On the Session with the Telex Interface Program

To check whether the sign-on to the Telex Interface Program is successful:

- Enter the MERVA ESA command **dm** or **dm last** to find out if the following message has been issued by the Telex Link:

```
ENL903I TELEX   signon successful, session=0329
```

- Enter the command **txdisp** to display the session status.

Chapter 7. Operating MERVA Link ESA

The MERVA System Control Facility (MSC) provides commands to operate MERVA Link ESA.

How to operate MERVA Link USS and the corresponding commands is described in “Chapter 8. Operating MERVA Link USS” on page 173.

You can use the unrestricted MSC operator commands if you are authorized to use the MERVA System Control Facility function. You can use the restricted MERVA Link operator commands if you are authorized to use the function MSC and the restricted MERVA ESA operator commands.

Overview of the MERVA Link Control within MSC

To monitor and operate the MERVA Link, select the MERVA System Control function (MSC) on the Function Selection menu, and switch to the MERVA Link environment by pressing the PF 9 key. You can use the MSC function, for example, to display the parameters and status of the MERVA Link Application Support Processes (ASPs) defined in the MERVA Link Partner Table (EKAPT), also referred to as PT. You can also use it to display the parameters and the status of the partner MERVA systems which can be operated at the local MERVA system, and to display information in the PT header.

Displaying the ASP List

Figure 51 shows a sample display of the MERVA Link message transfer applications (ASP list).

```
AC01       MERVA Link  List of Message Transfer Applications               N1
================================================================================
ASP Name   Partner      Date       Time     LR MSN  LC MSN   Status Diag    AI
================================================================================
A1I        BTB       1999/05/21  17:47:02    1603    1603    ON 04 SUB RQ   SR
--------------------------------------------------------------------------------
A2A        CA02      1999/05/21  17:47:10            0016    ON 00 CNFRMD
================================================================================









Command =====>
PF 1=Help      2=Retrieve   3=Return     4=SELECT      5=START       6=Nextgrp
PF 7=Backward  8=Forward    9=SWAP      10=Kickoff    11=Lstinop    12=Lstall
```

Figure 51. ASP List Sample

In this example only two ASPs are defined in the MERVA Link Partner Table. The ASPs are displayed in the sequence they are defined in the Partner Table. They are not displayed in alphabetical order.

Panel AC01 Contents

AC01 Is the panel identifier. This panel shows up to 16 ASP entries or ASP group separators defined in the PT.

N1 Is the node name of the applicable MERVA Link system. It is either the node name of the local MERVA Link system or the node name of a partner MERVA Link system.

ASP Name ASP name.

Partner Identifier of the partner system.

Date Date when the status information became applicable.

Time Time when the status information became applicable.

LR MSN Sequence number of the last received message.

LC MSN Sequence number of the last confirmed message.

Status Status information consists of two elements:

- The Application Support status identifier that can be:

  ON ASP in OPEN-NOHOLD status

  OH ASP in OPEN-HOLD status

  CN ASP in CLOSED-NOHOLD status

  CH ASP in CLOSED-HOLD status.
- One of the following:
  - The Message Transfer status code that applies to a sending ASP. It can be 00, 04, 08, 09, 12, or 13.
  - The receiving process error diagnostic information type that can be:

    MT Receiving process error detected at the MERVA Link MT layer (MTSP or MTP).

    AS Receiving process error associated with a specific ASP.
  - The character string '--', indicating that MERVA System Control Facility command processing error information is displayed in the diagnostic code field.

Diag

The diagnostic code provides additional information regarding the sending ASP status. For receiving process errors it is always RP ERR. Use the PF key defined for the **display** command (PF 4, described as SELECT in the PF-key lines) to display the receiving process error diagnostic information.

If receiving process error information is available and the applicable resource (ASP or MTP) is unknown, all ASP list lines indicate the receiving process error. The error information is deleted as soon as any receiving process is successful.

For details about these codes refer to *MERVA for ESA Messages and Codes*.

AI The ASP activity indicators can be:

S For sending ASP active

R For receiving ASP active

SR For sending and receiving ASP active.

The ASP activity indicators do not apply in an IMS environment because only one task can be active in an IMS Message Processing Region (MPR) at a time. The ASP activity indicator is displayed when the MERVA ESA End-User Driver task is active, the MERVA Link processing task in this MPR cannot be active then.

- - - Shows the ASP group structure defined in the PT.

=== Shows the beginning and end of the ASP list.

Displaying the Latest Status Information

If an ASP list is displayed and you press ENTER without any command data, the list is redisplayed with the most up-to-date information that is available in the PT.

In an IMS environment, the automatic PT-status update is performed only by tasks that run in the IMS MPR that owns this PT. It is the IMS MPR in which the MERVA ESA End User Driver (DSLEUD) is scheduled. If MERVA Link does not always run in the same IMS MPR as DSLEUD, you must use the PF key defined for the **display** command (PF 4, described as SELECT in the PF-key lines) to get the status updates resulting from tasks in other IMS MPRs.

ASP Groups

The ASP entries in the PT can be divided into ASP groups. An ASP group boundary is defined in the PT by an EKAPT TYPE=DUMMY macro. The PT dummy entry generated from this macro is represented in the ASP list as a line of dashes or any other character defined in the EKAPT TYPE=DUMMY macro. This line is displayed to show the group structure of the PT.

ASP group definitions are optional and have no effect on MERVA Link processing.

ASP List Subsets

When there are more than 16 ASPs or ASP group separators defined in the PT, the ASP list is displayed on more than one page with adjacent pages overlapping by one line. The last display line is followed by a row of equal signs in the color specified during customization (see the *MERVA for ESA Customization Guide*).

Use the **display** command to specify a generic ASP name subset. Only ASPs starting with the name of this subset are displayed. To reset the ASP name subset to the full set of ASPs, enter the **display** command without a parameter on the command line or press PF 4 (SELECT) with the cursor on the command line. If you press PF 4 with the cursor on an ASP list line, more details are displayed for the ASP and its associated MTP.

Use the **lstinop** command to specify the ASP status subset. Only inoperable ASPs are displayed. To reset the ASP status subset to the full set of ASPs, enter the **lstall** command.

Special Usage of PF Keys

The MERVA Link commands that have an ASP name as parameter can be entered as follows:

- You can enter the command and the ASP name in the command line.

- You can move the cursor to the ASP entry you want use and press the related PF key instead of entering the command and the ASP name.

For example, if your cursor is on the line for A2A and you press PF 10, the ASP A2A is kicked off.

Displaying Specific ASP/MTP Parameters

To display the details of a specific ASP entry and its associated MTP entry:

- Place the cursor at the ASP list line in panel AC01 describing that ASP, and press the PF key defined for the **display** command (PF 4, described as SELECT in the PF-key lines).
- Enter the command **dsa** with a full or a generic ASP name.

A sample display of specific ASP and Message Transfer Process (MTP) parameters is shown in Figure 52.

```
AC02       MERVA Link  Display Specific ASP / MTP Information               N1
================================================================================
ASP Name   MTP Name     Date       Time     LR MSN  LC MSN   Status Diag    AI
================================================================================
A2A        T2A       1999/05/21  17:47:10    0018    0016    ON 00 CNFRMD
================================================================================
--------------------------------------------------------------------------------
Free Form Name : APPC CONNECTION FROM N1 TO N2                   Recov .: M0
Partner ASP Name     Partner Node : A1A       N2                 Format.: N*
Queue Names of Send Queue Cluster : EKA2AS1   --------  --------  Start .: AE
AS Filter Names, User Exit Number : --------  --------  7010      Secur .: E
Control Queue Name    Window Size : EKA2ACQ   010       Journal : N N N
Local Routing Control Information : EKA2ACQ   EKAAWQ    EKA2ACQ   R C R
--------------------------------------------------------------------------------
Local System Information . . . . .: EKATS10   EKAS      X12A      Connect: QA
Partner System Information . . . .: CA02      EKAR      X21A      APPC
--------------------------------------------------------------------------------

Status Note in MERVA Link LC Control Message . . . . :
EKA714I Confirmation received for message with MSN = 0016
Command =====>
PF 1=Help      2=Retrieve   3=Return     4=SELECT      5=START       6=Nextgrp
PF 7=Backward  8=Forward    9=SWAP      10=Kickoff    11=Lstinop    12=Lstall
```

Figure 52. Specific ASP/MTP Display Sample

Panel AC02 Contents

Refer to Figure 51 on page 133 for more information on the lines starting with the panel identifier AC02 and ASP Name. This panel displays the name of the MTP (MTP Name) instead of the name of the partner system.

Free Form Name

Shows the first 40 characters of the free form name specified in the NAME parameter of the EKAPT.

Recov

Shows the values of the IPRECOV and ACCEPT parameters defined in the PT. It can be:

M0 For IPRECOV=MANUAL and ACCEPT=OK

A0 For IPRECOV=AUTO and ACCEPT=OK

M4 For IPRECOV=MANUAL and ACCEPT=WARNING

A4 For IPRECOV=AUTO and ACCEPT=WARNING

Partner ASP Name at Partner Node
Shows the address of the partner ASP, consisting of its ASP name and its node name.

Format
Shows the message format defined in the PT. It can be:

Q For MERVA ESA queue format.

N For line (network) format. If the message format is N, the line-format identifier is also specified, it can be:

***** Indicates that the first character of the message type must be used as line-format identifier. For example, S is the line-format identifier for a SWIFT message S100.

n Identifies a specific line format, such as T for telex.

# Indicates that the message type defined in MERVA Link control field EKANETID must be used.

Queue Names of Send Queue Cluster
Shows the queue names of the send-queue cluster. Three send-queue names can be specified as sub-parameters of the **SENDQC** parameter of an ASP definition. An omitted sub-parameter is indicated by eight minus signs in the display of a specific ASP.

Start
Shows the ASP start indicator and the receiving ASP status.

The ASP start indicator can be:

A For automatic start

O For operator start

R For automatic start with retry.

The receiving ASP status can be:

E For receiving ASP enabled

D For receiving ASP disabled.

AS Filter Names, User Exit Number
Shows the names of the first two Application Support Filters and the number of the MFS User Exit associated with this ASP. All three filter names are displayed if an MFS User Exit is not associated with this ASP, or if the User Exit number does not begin with **7**. The range of 7000 to 7999 is reserved by MERVA ESA for MERVA Link MFS User Exit numbers. The title of this information reads **Application Support Filter Names** if an MFS User Exit Number is not shown.

Up to three filter names can be specified as subparameters of the **FILTER** parameter of an ASP definition. "--------" is displayed if a subparameter is omitted.

Secur
Shows the message security defined for this ASP:

A For message authentication

E For message authentication and encryption.

Control Queue Name Window Size
Shows the application control queue name and the MIP window size. The window size in an ISC-based message transfer is 1, regardless of the window size specified in the PT ASP entry.

Journal
Consists of 3 characters that specify whether messages of the corresponding category are journaled or not. Each of these characters is Y for YES, or N for NO:

- The first character refers to the outgoing messages.
- The second character refers to the incoming reports.
- The third character refers to the incoming application messages.

Local Routing Control Information
Consists of three MERVA ESA queue names and three routing type identifiers. The first queue name and the first routing type identifier apply to local outgoing message routing when the message has been delivered to the partner ASP.

There are the following routing types:

P Specifies the MERVA ESA queue that stores the outgoing message.

R Specifies that an outgoing message must be routed according to the MERVA ESA Routing Table associated with the specified MERVA ESA queue.

D Specifies that an outgoing message must be discarded.

The second queue name and the second routing type identifier apply to local incoming report routing.

There are the following routing types:

R Indicates that an incoming report must be routed according to the MERVA ESA Routing Table associated with the specified MERVA ESA queue.

Q Indicates that an incoming report must be correlated with a message in the specified MERVA ESA queue, and the reported message must be routed according to the MERVA ESA Routing Table associated with the specified MERVA ESA queue.

C Indicates that an incoming report must be correlated with a message in the specified MERVA ESA queue, and the reported message must be routed according to the MERVA ESA Routing Table associated with the application control queue.

The third queue name and the third routing type identifier apply to local incoming application message routing.

There is only one routing type:

R Indicates that an incoming application message must be routed according to the MERVA ESA Routing Table associated with the specified MERVA ESA queue.

Local System Information
Consists of the:

- Sending Message Transfer Processor name, here: EKATS10

- Sending ASP transaction identifier, here: EKAS
- External MTP name, here: X12A

Connect
Shows the partner system connection requirements for this MTP.

A CONNECT=ACQ has been specified for this MTP. The MTP must try to acquire an APPC connection to the partner system before the process ends if an ALLOCATE session command was not successful.

Q CONNECT=QUEUE has been specified for this MTP. The MTP must issue ALLOCATE requests without the NOQUEUE option. If CONNECT=ACQ is also specified, the MTP must try to acquire an APPC connection at the begin of the process, if a session could not be allocated successfully.

R CONNECT=REL has been specified for this MTP. The MTP must release the APPC connection to the partner system at the end of the conversation.

ACQ, QUEUE, and REL may be specified in the CONNECT parameter. Connect: QA is shown in this case. Connect: QA, QR, or AR is shown if only two subparameters are specified.

The partner system connection requirements may be specified for any MTP, and shown in this panel. They are, however, honored only by an APPC-type MTP in the CICS environment.

Partner System Information
Consists of the:

- Partner system identifier, here CA02
- Partner receiving process identifier, here EKAR
- Partner's external MTP name, here X21A
- Partner system type, here APPC. It can be BTB or APPC.

The partner system identifier (the first item of the partner system information) is specified by the second subparameter of the **LINK** parameter or the first subparameter of the **DEST** parameter. These parameters are part of the MTP definition in the Partner Table. Its meaning depends on the local system environment.

- The keyword **BTB** is used as the partner system identifier of a Back-to-Back MTP.
- In the MERVA CICS environment it is the identifier of the APPC connection to the partner system as defined in CICS.
- In the MERVA IMS environment it is either the partner LU name (specified in the LINK parameter), or the APPC/MVS Symbolic Destination Name of the partner system (specified in the DEST parameter).

If external MTP names are not used by this MTP, the Profile or Logmode Name (specified in the LINK or DEST parameter) is shown as the third partner system information item.

Status Note in MERVA Link LC Control Message
The last confirmed control message (LC MSG) gives the status of the sending ASP. In this example message EKA714I is displayed.

The control queue of a specific ASP contains the LR and LC control messages. It can also contain in-process (IP) messages. In this example, the IP messages have been submitted to the partner. If the number of IP messages in the control queue is not zero, the number is displayed above the Free Form Name field as part of the sentence *Number of In Process messages in the application control queue is nnn*.

Displaying ASP/MTP Error Codes

Figure 53 displays specific ASP and MTP parameters with sending and receiving process error diagnostic information. The errors in this example were generated by disabling the MCB for the transmitted message in the local and in the partner system (in two separate steps).

```
AC02       MERVA Link  Display Specific ASP / MTP Information              N1
================================================================================
ASP Name   MTP Name     Date       Time     LR MSN  LC MSN   Status Diag    AI
================================================================================
A2A        T2A       1999/05/21  17:51:25    0018    0016    ON 09 MP ERR
================================================================================
Receiving process error diagnostic information type AS :  001A 001C  0004 008E
Free Form Name : APPC CONNECTION FROM N1 TO N2                   Recov .: M0
Partner ASP Name     Partner Node : A1A       N2                 Format.: N*
Queue Names of Send Queue Cluster : EKA2AS1   --------  --------  Start .: AE
AS Filter Names, User Exit Number : --------  --------  7010      Secur .: E
Control Queue Name    Window Size : EKA2ACQ   010       Journal : N N N
Local Routing Control Information : EKA2ACQ   EKAAWQ    EKA2ACQ   R C R
--------------------------------------------------------------------------------
Local System Information . . . . .: EKATS10   EKAS      X12A      Connect: QA
Partner System Information . . . .: CA02      EKAR      X21A      APPC
--------------------------------------------------------------------------------

Error Code Vector and Status Note in LC Ctrl Message :  0008  0020  0004  0889
EKA721E Error reported for message with MSN = 0017

Command =====>
PF 1=Help      2=Retrieve   3=Return     4=SELECT      5=START       6=Nextgrp
PF 7=Backward  8=Forward    9=SWAP      10=Kickoff    11=Lstinop    12=Lstall
```

Figure 53. ASP Error Diagnostic Information Display Sample

The line starting with A2A displays the error information '09 MP ERR' (Message Processing ERRor). This information has been provided by the partner system. It indicates that the partner system could not successfully process the incoming message. This error was generated by disabling the MCB for the transmitted message in the partner system.

The Error Code Vector in line 19 of this panel shows the corresponding error codes of the local MTP.

The receiving process error diagnostic information indicates that the local system could not successfully process an incoming message. This error was generated by disabling the MCB for the incoming message in the local system.

Note: Line 5 (ASP list line), line 7 (receiving process error diagnostic information), and line 20 (status note EKA721E) are displayed in red on the panel. The color of these lines is defined by the sample color customization.

Refer to Figure 52 on page 136 for more information on the panel fields displayed. This panel shows an additional field in the line above the Free Form Name field.

Receiving process error diagnostic information type AS
Refer to *MERVA for ESA Messages and Codes* for details on MERVA Link receiving process error diagnostic information (type AS, MT, US).

Error Code Vector and Status Note in LC Ctrl Message
The error code vector consists of four hexadecimal codes that apply to the local sending MTP. It shows the MTSP return code, the MTP return code, the MTP reason code, and an additional code which may have been provided by the data communication subsystem. In this example, the error code vector tells that an error has been reported by the sending MTP EKATS10 (0008), that a request or data was not accepted by the partner process (0020), that a MERVA Link error report was received from the partner process (0004). **0889** is the VTAM sense code for an APPC Send_Error verb issued by the partner MTP.

If the local DC system is APPC/MVS, the error code vector reads:

```
000A  0020  0004 0889
```

000A indicates that an error has been reported by the sending MTP EKATPO1. All other codes are the same.

The ASP status note in the last confirmed control message (LC MSG) explains the status of the sending ASP. In this example message EKA721E is displayed.

Interpreting ASP/MTP Error Codes

MERVA Link USS provides an application that can interpret error information shown in panel AC02. This application is called the MERVA Link USS Application Control Command application EKAACC, abbreviated to ACC. It can be used, for example, in an interactive OS/390® USS shell if MERVA Link USS is installed in the OS/390 USS environment. The ACC commands that interpret the error information shown in Figure 53 on page 140 are explained in the following. The ACC command entered by the operator is shown in boldface, the command response follows in normal face.

For more information about the MERVA Link USS control command application ACC, refer to "Overview of the MERVA Link Control within ACC" on page 173.

Explain a Diagnostic Code

Use the ACC command **dxd** to explain a MERVA Link ESA diagnostic code, for example:

```
--------------------------------------
ekaacc dxd mp err
EKAACC dxd mp err

EKAACC_xdc MP ERR is reported by MERVA ESA or MERVA OS/2
EKAACC_xdc Application message processing error
--------------------------------------
```

Explain Receiving Process Error Type AS

Use the ACC command **dxra** to explain a MERVA Link ESA receiving process error code vector of the type AS, for example:

```
--------------------------------------
ekaacc dxra 1a1c048e
EKAACC dxra 1a1c048e

EKAACC_xrn The normalized error code vector is 0008 0008 001A 001C 0004 008E

EKAACC_xrn Error reported by the MERVA Link ESA Inbound AS Program
EKAACC_xrn 001A: Error found while handling an Application Message
```

```
EKAACC_xrn 001C: Copy message from net buffer to TOF failed
EKAACC_xrn 0004: DSLMFS request completed with restrictions
EKAACC_xrn 008E: Format NET to TOF failed, message type 0DSL is used
-------------------------------------
```

Explain Receiving Process Error Type MT

The receiving process error diagnostic information shown in panel AC02 is of the type MT if an error was found in the lower MERVA Link layers. Use the ACC command **dxrm** to explain a MERVA Link ESA receiving process error code vector of the type MT, for example:

```
-------------------------------------
ekaacc dxrm 00080070E7F1F2C1
ekaacc dxrm 00080070E7F1F2C1

EKAACC_xrn The normalized error code vector is 0008 0070 E7F1 F2E7 0000 0000

EKAACC_xrn Error reported by the MERVA Link ESA Inbound MTSP
EKAACC_xrn 0070: Sending MTP name is invalid
-------------------------------------
```

The first four characters of the invalid MTP name are shown in error codes 3 and 4 in hexadecimal representation. The beginning of the invalid MTP name is **X12X**. The correct name in this example is X12A.

Explain Error Code Vector in LC Control Message

The error code vector that is stored in the LC Control Message is a MERVA Link ESA sending process error code vector of the type MT. Use the ACC command **dxsm** to explain this type of a MERVA Link ESA error code vector, for example:

```
-------------------------------------
ekaacc dxsm 08200489
EKAACC dxsm 08200489

EKAACC_xsm Error reported by the MERVA Link ESA Outbound TP EKATS10
EKAACC_xsm 0020: Request or data not accepted by the partner process
EKAACC_xsm 0004: MERVA Link error report received from the partner system
EKAACC_xsm 0089: 0889 The APPC transaction program has detected an error
-------------------------------------
```

or

```
-------------------------------------
ekaacc dxsm 000A002000040889
EKAACC dxsm 000A002000040889

EKAACC_xsm Error reported by the MERVA Link ESA Outbound TP EKATPO1
EKAACC_xsm 0020: Request or data not accepted by the partner process
EKAACC_xsm 0004: MERVA Link error report received from the partner system
EKAACC_xsm 0889: 0889 The APPC transaction program has detected an error
-------------------------------------
```

Displaying the SCP List

Enter the command **node** or **ps** without a parameter in the command line of any MSC panel to display the list of System Control Processes (SCPs) in panel AC03. Figure 54 on page 143 shows a sample display of a MERVA Link System Control Process (SCP) list.

```
AC03       MERVA Link  List of Partner MERVA Systems                        N1
================================================================================
Partner    Node Name  System ID   L-Date     L-Time    P-Time     Status  Diag
================================================================================
LONDON     N2         CL01      1999/03/15  14:46:19  13:46:52    00  CMD SUCC
--------------------------------------------------------------------------------
PARIS      N3         CP01      1999/03/15  14:48:59  14:48:23    09  DSL CONN
================================================================================









Command =====>
PF 1=Help      2=Retrieve   3=Return     4=SELECT      5=START       6=Nextgrp
PF 7=Backward  8=Forward    9=SWAP      10=Kickoff    11=Lstinop    12=Lstall
```

Figure 54. SCP List Sample

Two partner MERVA systems are defined in this example. The partner MERVA systems are located in London and in Paris. A command has successfully been sent to the MERVA system in London. The MERVA system in Paris is not active.

Panel AC03 Contents

Partner
Shows the local nickname for the partner MERVA system. You can use this name as the parameter of the **node** or **ps** command to switch to that partner MERVA system.

Node Name
Shows the MERVA Link Node Name of the partner MERVA system. You can use this name as the parameter of the **node** or **ps** command to switch to that partner MERVA system.

System ID
Shows the local identifier of the partner data communication (DC) subsystem (CICS or APPC/MVS). If your local DC system is CICS, it is either the identifier of a CICS Connection (1-4 characters) or a CICS Partner name (5-8 characters) defined in your local CICS system. If your local DC system is APPC/MVS, it is either the VTAM application name of the partner LU, or the APPC/MVS Symbolic Destination Name of the partner system.

L-Date
Shows the local date when a command has been sent to the partner system.

L-Time
Shows the local time when a command has been sent to the partner system.

P-Time
Shows the partner system time when a command response has been returned from the partner system.

Status
Shows the partner system status when a command has been sent to the partner system. The status code has the following meanings:

00 Command request successfully processed

08 Error detected in the local system

09 Error reported by the partner system

Diag
Shows an eight-character diagnostic code which provides additional information if an error is found by either, the local system or the partner system. For details refer to *MERVA for ESA Messages and Codes*.

Switch to a Partner MERVA System

The request to switch to a partner MERVA system can be entered as follows:

- You can enter the command **node** or **ps** with the nickname or the MERVA Link node name of the partner MERVA system as parameter in the command line of any MSC panel.
- You can move the cursor to the SCP list line in AC03 describing the partner MERVA system, and press the PF key defined for the **display** command (PF 4, described as SELECT in the PF-key lines).

Displaying PT Header Information

Enter the command **dpth** in the command line of any MSC panel to display PT header information in panel AC04. You get this display also if you enter the command **backward** or **bw** (or press PF 7) when the first ASP is displayed in panel AC02.

Figure 55 shows a sample display of MERVA Link Partner Table Header information.

```
AC04       MERVA Link  Display Partner Table Header Information             N1
================================================================================
PT Hdr ID  Version     Date    Time    #ASP  #MTP  #SCP    DC System  MERVA ID
================================================================================
* EKAPT *   4.1.0   1999/03/03 17:00    002   002   002    C520/MVS       DSL1

Test Environment . . . . . . . . . . .  OFF              (ON, OFF, EXCEPTION)
Conversation Trace ID,  Ctrc Type  . .  EKAT      FULL    (N/A,    FULL, WEAK)
--------------------------------------------------------------------------------

ASP Monitor  Transaction Identifier  .  EKAM             (N/A :  AM disabled)
AM Execution Count,  ASP Start Count .  00005     00001   (AM Tasks   SF Cmds)
AM Start Time, Restart Time Interval .  03307432  01:00   (Interval  in mm:ss)
Current Time,  Wait for Restart  . . .  03307440  00:52   (Wait Time in mm:ss)
--------------------------------------------------------------------------------




Command =====>
PF 1=Help      2=Retrieve   3=Return     4=SELECT      5=START       6=Nextgrp
PF 7=Backward  8=Forward    9=SWAP      10=Kickoff    11=Lstinop    12=Lstall
```

Figure 55. PT Header Display Sample

Panel AC04 Contents

PT Hdr ID
Shows the mandatory PT header identifier.

Version
Shows the version of the Partner Table. In this example, it is Version 4, Release 1, Modification Level 0.

Date Time
Shows the date and time when the Partner Table was generated.

#ASP #MTP #SCP
Shows the numbers of ASP, MTP, and SCP entries in the PT. There are two entries of each type in this sample.

DC System
Shows an identifier for the type and version of the local data communication subsystem. It can be, for example, C410/MVS, C520/MVS, C220/VSE, or APPC/MVS.

MERVA ID
Shows the MERVA system identifier defined in the MERVA Parameter Table (DSLPARM).

Test Environment
Shows the MERVA Link test environment status as it is generated in the PT header (EKAPT TYPE=INITIAL,TEST=...), or as it is set by a **set** or **reset** command. It can be ON, OFF, or EXCEPTION.

Conversation Trace ID
Shows the four-character identifier of the conversation trace destination in the CICS environment, if the conversation trace is enabled. If it is disabled in the CICS environment or if MERVA Link executes in the IMS environment, N/A is displayed instead of a conversation trace identifier. This indicates that the conversation trace is not applicable.

Conversation Trace Type
Shows whether a weak or a full conversation trace is written if the conversation trace is enabled. If the conversation trace is disabled, blanks are displayed instead of a conversation trace type.

ASP Monitor Transaction Identifier
Shows the four-character transaction identifier of the ASP Monitor in the CICS environment, if the ASP Monitor is enabled. If it is disabled in the CICS environment or if MERVA Link executes in the IMS environment, N/A is displayed instead of a transaction identifier. This indicates that the ASP Monitor is not applicable. All follow-on information of the ASP Monitor group is missing in that case.

AM Execution Count
Shows the number of ASP Monitor task instances within the current CICS session. Re-loading the PT from the program library during a CICS session resets this counter.

ASP Start Count
Shows the number of **sf** commands (MERVA Start Function commands) issued by the ASP Monitor when it was recently active.

AM Start Time Stamp
Shows a relative number in seconds with an undefined origin representing the time when the ASP Monitor task was recently started. This number is zero if

the ASP Monitor has not yet been started in this CICS session. The dummy time stamp '00000255' indicates that an ASP Monitor start request has been issued by a sending or receiving ASP, or by an MSC operator via the **set am** command. The ASP Monitor task will be started by CICS in the latter case, when the applicable restart time interval has elapsed.

Restart Time Interval
Shows the ASP Monitor restart time interval in minutes and seconds. The seconds are always zero. The restart time interval can be set to multiples of one minute up to 59 minutes. A restart time interval of 00:00 means that the ASP Monitor must not be restarted.

Current Time
Shows a relative number in seconds which represents the current time. This number corresponds to the AM Start Time Stamp.

Wait for Restart
Shows the time interval until the ASP Monitor will execute the next time. This time interval may be blank or invalid if the ASP Monitor was not yet active, or if the restart time interval was modified by a terminal operator.

The MERVA Link Commands of the MERVA System Control Facility

The commands for operating MERVA Link are explained in the following sections in alphabetical order. These explanations are preceded by some general considerations concerning the MERVA Link commands.

MERVA Link Command Considerations

ASP Operating Commands

The **lstall** and **lstinop** commands provide for displaying a list of all ASPs defined in the partner table and the subset of inoperable ASPs, respectively. Detailed information of a specific ASP and its associated MTP can be displayed using a PF key or using the command **dsa**.

A sending ASP that is in HOLD status can be opened and closed using the commands **aopen** and **aclose**. These commands require an ASP name as parameter.

A sending ASP can be started, kicked off, or put in HOLD status using the commands **astart**, **kickoff**, and **hold**. These commands require an ASP name as parameter.

The **kickoff *** command kicks off all ASPs defined in the Partner Table.

The **hold *** command holds all messages of all ASPs defined in the Partner Table in their send queue cluster. ASPs must be started individually.

The commands **lcreset** and **lrreset** resynchronize cooperating ASPs when an MIP violation has been encountered. These commands, however, should not be used without investigating the reason for the MIP violation.

The **recover** (or **ipcopy**) command provides a means to copy in-process messages of an inoperable or permanently waiting ASP in CLOSED status to a send queue of another transmission medium.

The **iprecov** (or **ipmove**) command provides a means to route a specific in-process message, which was not successfully delivered, to an error queue, to delete it from

the set of in-process messages, and to resume message transmission. The functionality of this command corresponds to the functionality of the automatic IP message recovery function which can be requested by the PT parameter IPRECOV=AUTO.

In the CICS environment, the commands **set** and **reset** allow you to modify temporarily the TRACE, TEST, ACCEPT, and IPRECOV parameters defined in the Partner Table.

Note: The commands **set** and **reset** are not (fully) supported in the IMS environment. They are therefore disabled by default in the IMS environment. If you want to use the SNAP, PSNAP, and WSZ options, which are supported in the IMS environment, you can enable the **set** and **reset** commands with the command **enable set**. You must, however, be aware of the fact, that the other **set** or **reset** command options may not have the desired effect.

In the CICS environment, the commands **enable** and **disable** allow you to temporarily modify the status of a specific receiving ASP or all receiving ASPs. These commands may not have the desired effect in the IMS environment.

Privileged Commands

The MSC commands which modify or manipulate the MERVA Link processing environment are characterized as privileged commands.

If the OPID parameter in the DSLPARM defines a master operator prefix, a privileged command can be issued only by operators with an operator identifier starting with that master operator prefix. Message EKA779E is issued when an unauthorized operator issues a privileged command.

A privileged command can be issued by all operators if the OPID parameter in the DSLPARM is defined as '***'.

Command Processing Errors

An error can be encountered by the MERVA System Control Facility while:

- Processing the IP messages in the control queue of an ASP
- Processing the LC control message in the control queue of an ASP
- Entering a MERVA ESA command.

MERVA ESA command and message handling errors are reported in the applicable ASP list line. These error codes are explained in *MERVA for ESA Messages and Codes*.

Setting the Status of an ASP to CLOSED (ACLOSE)

The **aclose** command sets the status of an ASP to CLOSED, which means that the ASP cannot transmit messages to its partner ASP. This command updates the AS status information in the last confirmed control message (LC MSG) in the ASP Control queue.

The **aclose** command is a privileged command. It can be issued only by an authorized operator.

Command Format

The format of the **aclose** command is:

aclose	*aspname*

Parameter Descriptions

aspname
The name of the ASP.

Note: The ASP specified must be in HOLD status. An **aclose** command for an ASP in NOHOLD status is not accepted.

Command Example

If the sending ASP named AS1 is in HOLD status, enter the following command to close this ASP.

```
aclose as1
```

Setting the Status of an ASP to OPEN (AOPEN)

The **aopen** command sets the status of a specific ASP to OPEN so that it can transmit messages to its partner ASP. This command updates the AS status information in the last confirmed control message (LC MSG) in the ASP control queue.

The **aopen** command is a privileged command. It can be issued only by an authorized operator.

Command Format

The format of the **aopen** command is:

aopen	*aspname*

Parameter Descriptions

aspname
The name of a specific ASP.

Note: The ASP specified must be in HOLD status. An **aopen** command for an ASP in NOHOLD status is not accepted.

Command Example

To open an ASP, named A1I, enter:

```
aopen a1i
```

Starting a Sending ASP (ASTART)

This command starts a sending ASP and sets it to NOHOLD status if it is currently in HOLD status.

The **astart** command is a privileged command. It can be issued only by an authorized operator.

The **astart** command sets all members of a send queue cluster to NOHOLD status if they are related via the RELATED parameter of the DSLFNT macro. If this parameter was not specified, the **astart** command applies only to the queue that is specified as the first send queue in the SENDQC parameter of the EKAPT TYPE=ASP macro. The other queues of the send queue cluster are not changed to NOHOLD status. However, all send queues in NOHOLD status are processed in response to the **astart** command, regardless of the RELATED parameter.

If you position the cursor on an ASP line displayed in the MERVA Link Display Specific Message Transfer Application panel, and press the PF key defined for **astart** (PF 5), you can set the send queue cluster of this ASP and the ASP itself to NOHOLD status, start the ASP, and display the ASP list.

Command Format

The format of the **astart** command is:

astart **sa**	*aspname*

Parameter Descriptions

aspname
The name of a specific ASP.

Note: You can also move the cursor to the ASP to be started and press PF 5.

Command Example

To start a specific ASP, named A1, enter:

```
astart a1
```

Scrolling through a List of ASPs or SCPs (BACKWARD and FORWARD)

Use these commands to scroll backward (**backward**) and forward (**forward**) if there are more than 15 elements in the current ASP or SCP list. There is an overlap of one line from one page to the next.

The **backward** command has no effect if the first part of the list is already displayed. The **forward** command has no effect if the last part of the list is already displayed.

If you press the PF key defined for the **backward** command (PF 7):

- On the MERVA Link Display Specific ASP/MTP panel (AC02), you can display the parameters of the previous ASP in the current ASP subset list.
- When the parameters of the first ASP in the current ASP subset list are displayed, PT header information is displayed in panel AC04.
- In panel AC04, the ASP subset list is displayed again.

If you press the PF key defined for the **forward** command (PF 8):

- On the MERVA Link Display Specific ASP/MTP panel, you can display the parameters of the next ASP in the current ASP subset list.
- When the parameters of the last ASP in the current ASP subset list are displayed, that ASP subset list is displayed again.
- In panel AC04, the details of the first ASP/MTP are displayed in panel AC02.

Command Format

The format of the **forward** and **backward** commands is:

backward **bw**	
forward **fw**	

There are no parameters for these commands.

Disabling a Receiving ASP (DISABLE)

This command disables a specific ASP or all ASPs for receiving messages in the CICS environment. It may not have the desired effect in the IMS environment.

The **disable** command is a privileged command. It can be issued only by an authorized operator.

When an incoming message is rejected because the receiving ASP is disabled, the sender gets an error indication. This error indication consists of the Return Code 08 (shown as 09 in MSC screens AC01 and AC02) and the diagnostic code RA DIS for Receiving Application DISabled.

Command Format

The format of the **disable** command is:

disable	{*aspname* | *}

Parameter Descriptions

aspname
The name of a specific ASP that must be disabled for receiving messages.

* Specifies that all ASPs defined in the PT must be disabled for receiving messages.

Command Example

To disable all receiving ASPs enter:

```
disable *
```

Displaying an ASP (DISPLAY)

This command allows you to specify a generic ASP name subset and to list only the ASPs of this subset. Only ASPs starting with the name of this subset, and all ASP group separators, are displayed.

Note: To reset the ASP name subset to the full set of ASPs, enter the **display** command without a parameter, or position the cursor on the command line and press the PF key defined for the **display** command (PF 4, described as SELECT in the PF-key lines).

To obtain more detailed information on a particular ASP, position the cursor on the line for that ASP and press the PF key defined for the **display** command (PF 4, described as SELECT in the PF-key lines).

Command Format

The format of the **display** command is:

display **da**	[*aspname* | *generic aspname*]

Parameter Descriptions

aspname | *generic aspname*
To display an ASP list, you must specify a particular ASP or a generic ASP name subset as a parameter. Press ENTER to redisplay the list with the latest information in the PT.

Displaying PT Header Information (DPTH)

This command allows you to display data contained in the header of the MERVA Link Partner Table in panel AC04. This data displayed in panel AC04 includes information about the status of the MERVA Link CICS ASP Monitor.

Command Format

The format of the **dpth** command is:

dpth	

There are no parameters for this command.

Displaying Information of a Specific ASP (DSA)

This command allows you to specify a full ASP name or a generic ASP name subset and to display detailed information of the ASPs of this subset in panel AC02. The ASPs starting with the name of this subset are displayed in wrap-around mode when the **dsa** command is entered repeatedly.

Command Format

The format of the **dsa** command is:

dsa	[*aspname* | *generic aspname*]

Parameter Descriptions

aspname | *generic aspname*

To display detailed information of an ASP or a set of ASPs, you must specify a particular ASP or a generic ASP name subset as a parameter. Press ENTER with the **dsa** command in the AC02 command line to display information of the next ASP of the applicable ASP subset.

Command Examples

To display detailed information of an ASP, named A2A, enter:

```
dsa a2a
```

To display detailed information of all ASPs with names starting with A, enter:

```
dsa a
```

and press ENTER in panel AC02 to display the next ASP of that subset.

Enabling MERVA Link Resources (ENABLE)

This command allows you to enable the **set** command in the IMS environment for specific purposes. The **set** command is disabled by default in the IMS environment.

This command allows you also to enable a specific ASP or all ASPs for receiving messages in the CICS environment. The **enable** command for receiving ASPs may not have the desired effect in the IMS environment.

The **enable** command is a privileged command. It can be issued only by an authorized operator.

Command Format

The format of the **enable** command is:

enable	{*aspname* | * | set}

Parameter Descriptions

aspname
The name of a specific ASP that must be enabled for receiving messages.

* Specifies that all ASPs defined in the PT must be enabled for receiving messages.

set
Specifies that the **set** command must be enabled in the IMS environment for specific purposes.

Command Example

To enable the **set** command in the IMS environment enter:

```
enable set
```

Obtaining Help Information (EXPLAIN)

You can display MERVA ESA help information by pressing the PF key defined for the **help** command.

MERVA Link provides the **explain** command to display environment dependent explanations. Explanations are available for:

- MERVA Base, SWIFT Link, and Telex Link commands when you enter the **explain** command in the MSC Main Menu (panel ACMM) or in the MSC MERVA Operator Command panel (AC00).
- MERVA Link commands, when you enter the **explain** command in the Display List of Message Transfer Applications panel AC01 (Figure 51 on page 133).
- Specific ASP/MTP control information, when you enter the **explain** command in the Display Specific ASP/MTP panel AC02 (Figure 53 on page 140).
- The Partner MERVA System Control Facility when you enter the **explain** command in the List of Partner MERVA Systems panel AC03 (Figure 54 on page 143).
- The PT Header information, when you enter the **explain** command in the Display Partner Table Header Information panel AC04 (Figure 55 on page 144).

To return from an explanation panel to the previous display, press the ENTER key, or enter any other command. Pressing the ENTER key in the AC00 panel without a command, however, results in the display of the MSC main menu. You cannot redisplay a MERVA command response in the CMD environment.

The **explain** command is always executed in your local MERVA system. This means, you get the explanation in the language of your local MERVA system. If you are in a partner MERVA system control session, the follow-on commands are again executed in the partner MERVA system.

Command Format

The format of the **explain** command is:

explain **xpl**	

There are no parameters for this command.

Setting the Status of an ASP to HOLD (HOLD)

This command sets the AS status of an ASP and its send queue cluster to HOLD. Message processing is completed, and no new message in the send queue cluster is processed.

The **hold** command is a privileged command. It can be issued only by an authorized operator.

The **hold** command sets all members of a send queue cluster to HOLD status if they are related via the RELATED parameter of the DSLFNT macro. If this parameter is not specified, this command applies only to the queue that is specified as the first send queue in the SENDQC parameter of the EKAPT TYPE=ASP macro.

The **hold** command is rejected and the operator message EKA771E is issued if the MERVA Link LC Control Message is currently being updated by a MERVA Link sending or receiving process. Enter the **hold** command again to set the ASP into HOLD status in this situation.

Command Format

The format of the **hold** command is:

hold **ha**	{*aspname* | *}

Parameter Descriptions

aspname
The name of a specific ASP.

* Specifies that all ASPs defined in the PT must be set to HOLD status.

Command Example

To set the status of all ASPs to HOLD enter:

```
hold *
```

Recovering from a Delivery Error (IPRECOV)

The **iprecov** (or **ipmove**) command can be used to recover an inoperable ASP from an IP message that cannot be delivered to the receiving application. It can also be used to recover an inoperable ASP from a local MIP violation.

The **iprecov** command includes the functions of the **kickoff** command. This means, message transmission is automatically resumed as soon as the ASP has been successfully recovered. Message transmission is resumed with the first IP message in the control queue. This message may have already been delivered to the receiving application.

The **iprecov** command is a privileged command. It can be issued only by an authorized operator.

Automatic recovery of an ASP from an undeliverable message can be requested by an ASP customization option (IPRECOV=AUTO). If you specify IPRECOV=MANUAL, an ASP must be recovered manually from that message using the **iprecov** command specifying the MIP message sequence number of that message.

The **iprecov nnnn** command is accepted only if you specify it in the AC02 panel (Display Specific ASP / MTP) displaying an inoperable ASP. This ensures that the applicable ASP and the control queue which contains the subject IP message are clearly identified. This panel may show an LC control message note which contains the MIP message sequence number of the IP message which is undeliverable (operator message EKA721E or EKA732E). You must specify the MIP message sequence number of the IP message which is undeliverable as a parameter of the **iprecov** command with four digits. Specify leading zeros, if applicable.

The MIP message sequence number must be entered with great care in the **iprecov** command. If you enter the MSN of an existing IP message that is not the one that is reported as undeliverable, you may duplicate a message and loose another message.

When MERVA Link executes the **iprecov nnnn** command, it assigns message class **RI** to the identified IP message, and moves the message out of the control queue using the MERVA ESA routing table associated with that control queue. If you use the **iprecov nnnn** command, this routing table must be able to handle messages of the class **RI**.

The recovery of an ASP from an undeliverable message may be unsuccessful and may cause a local MIP violation in a subsequent sending process. Use the **iprecov *** command to recover an ASP from any type of local MIP violation.

The **iprecov *** command is accepted only if you specify it in the AC02 panel (Display Specific ASP / MTP) displaying an inoperable ASP. This ensures that the applicable ASP and the control queue that contains the IP messages which must be resequenced are clearly identified.

Command Format

The format of the **iprecov** command is:

iprecov **ipmo**ve	{*mip msn* | *}

Parameter Descriptions

mip msn
The MIP sequence number of an IP message in the control queue of the identified ASP.

* Checks the message sequence numbers of all IP messages in the applicable control queue, and corrects the message sequence numbers if they are not in ascending order.

Note: This command can be specified only in the AC02 panel which identifies the applicable ASP.

Command Example

To recover an ASP from the undeliverable message with MIP message sequence number 385, enter:

```
iprecov 0385
```

in the AC02 panel.

Starting an ASP (KICKOFF)

Similar to the **start** command the **kickoff** command starts an ASP. However, the **kickoff** command does *not* change the HOLD or NOHOLD status of an ASP; for this reason, a **kickoff** command for an ASP whose status is HOLD only starts the processing of unconfirmed messages in the control queue.

The **kickoff** command is a privileged command. It can be issued only by an authorized operator.

If you position the cursor on an ASP line and press the PF key defined for the **kickoff** command (PF 10), you set the send queue cluster of this ASP to NOHOLD status, request the start of the ASP, and return to the ASP list display (shown in Figure 51 on page 133).

Command Format

The format of the **kickoff** command is:

kickoff **ka**	{*aspname* | *}

Parameter Descriptions

aspname
The name of a specific ASP.

Note: You can also move the cursor to a specific ASP you want to kick off, and press PF 10.

* Starts all ASPs defined in the PT.

Command Example

To kick off all ASPs defined in the PT, enter:

```
kickoff *
```

Resynchronizing Partner ASPs (LCRESET and LRRESET)

The **lcreset** command resynchronizes partner ASPs at the sending side after a MERVA Link message integrity protocol (MIP) violation.

If the ASP status is inoperable, the **lcreset** command resets the last confirmed message sequence number (LC MSN) in the LC control message of the control queue to 0. The command has no effect on an ASP that is in operable status.

The **lcreset** command is a privileged command. It can be issued only by an authorized operator.

When the message sequence number (MSN) in the LC control message is 0, an MIP reset indicator is transmitted with the next message. The MSN of this message is 1.

The **lrreset** command resynchronizes partner ASPs at the receiving side after a MERVA Link MIP violation.

The **lrreset** command deletes the last received control message (LR MSG) in the applicable control queue.

The **lrreset** command is a privileged command. It can be issued only by an authorized operator.

If an LR MSG is missing in the application control queue, the MSN of a received message is not checked. Any message is accepted, and the LR MSN is set to the MSN in the received message. The MSN of a received message is then checked again, as any received message is saved as LR MSG in the application control queue.

Command Format

The format of the **lcreset** command is:

lcreset **lcrs**	*aspname*
lrreset **lrrs**	*aspname*

Parameter Descriptions

aspname
The name of a specific ASP.

Command Example

If you want to reset the MIP of a specific sending ASP, named A2A, enter:

```
lcrs a2a
```

Resetting the ASP Status Subset (LSTALL)

The **lstall** command resets the ASP status subset so that all ASPs of the current ASP name subset are listed.

Command Format

The format of the **lstall** command is:

lstall **la**	

There are no parameters for this command.

Setting the ASP Status Subset (LSTINOP)

The **lstinop** command sets the ASP status subset so that the list contains only those inoperable ASPs that are members of the current ASP name subset. ASP group separators are also listed.

Command Format

The format of the **lstinop** command is:

lstinop	

There are no parameters for this command.

Displaying the Next ASP or SCP Group (NEXTGRP)

The **nextgrp** command sets the ASP or SCP list position to the first entry of the next ASP group if the command is entered in panel AC01, and to the first entry of the next SCP group if the command is entered in panel AC03. The beginning of an ASP group is defined by the beginning of the PT or by an EKAPT TYP=ASPGS (or TYPE=DUMMY) macro. The beginning of an SCP group is defined by the beginning of the PT or by an EKAPT TYP=SCPGS macro.

If you enter the **nextgrp** command when the last ASP or SCP group is listed, the first ASP or SCP group is listed. This command does not affect the ASP name subset, or the ASP status subset.

Command Format

The format of the **nextgrp** command is:

nextgrp **ng**	

There are no parameters for this command.

Switching to a Partner MERVA System (NODE and PS)

You can use the **node** and **ps** commands to select a partner MERVA Link system for Partner MERVA System Control (PMSC), and to switch to a specific partner MERVA system. The PMSC function is fully supported by all MERVA ESA partner systems. A subset of the MSC functions is supported by the PMSC function of MERVA AIX®. The PMSC function is not supported by the MERVA Link implementations in the OS/390 USS, OS/2®, and Windows NT® environments.

You can switch directly to a partner MERVA system if you enter a valid partner MERVA system name as parameter of the **node** command. If you enter the **node** command without a parameter or with an incorrect partner MERVA system name, a list of valid partner MERVA systems is displayed.

If the list of partner MERVA systems does not fit into one display panel, you can scroll through it using the scrolling commands. To select a specific partner MERVA system place the cursor at the required entry in the partner MERVA system list and press PF 4 (SELECT).

A connection to the applicable partner system must be available for a successful switch to the partner MERVA system. In the **CICS** environment, the connection status is checked as part of each PMSC interaction (PMSC command). If the connection is released, the PMSC client in the local system acquires the connection, and waits two seconds before it tries to allocate a session. In the **IMS** environment, intersystem connections are dynamically established by APPC/MVS.

The **node** command is a privileged command. It can be issued only by an authorized operator. The switch to a specific partner MERVA system may be further restricted to a subset of these operators. This is done in the respective EKAPT TYPE=SCP entries of the local and the partner MERVA Link systems.

The MSC user in the local MERVA system must have an entry in the User File of the partner MERVA system. This User File Entry can authorize the (remote) MSC user to issue restricted MSC commands, for example, kickoff ASP.

In the **CICS** environment, CICS System Programming Interface (SPI) commands are used by the PMSC client program. CICS command security checking applies as specified by the CICS XCMD and CMDSEC system initialization parameters. For more information, refer to the CICS documentation.

When the **node** or **ps** command is entered, all information about the position in a command sequence (for example, the sequence of JRN or DF commands) is reset. The response to a command will be the same as if the MSC function was entered from the MERVA Function Selection menu. This rule applies in any case, whether you switch to a partner MERVA system or whether you stay in the local MERVA system after the **node** command.

The **node** or **ps** command is always executed in your local MERVA system. This means, you cannot display the SCP list of a partner MERVA system, and you cannot switch from one of your partner MERVA systems to one of its partner MERVA systems.

Command Format

The format of the **node** command is:

node **ps**	[*local_partner_system_name* | *partner_system_node_name* | *local_system_node_name* | *]

Parameter Descriptions

local_partner_system_name
A local nickname for the partner MERVA system defined in an SCP entry of the partner table. This parameter switches to the specified partner MERVA system.

partner_system_node_name
The MERVA Link node name of the partner MERVA system. This parameter switches to the specified partner MERVA system.

local_system_node_name
The name of the local MERVA system. This parameter switches back to the local MERVA system and displays the MSC main menu (ACMM).

*****
This parameter switches back to the local MERVA system and displays the beginning of the local MERVA Link ASP list.

The set of partner systems which can be operated from a MERVA system is defined in the MERVA Link Partner Table of that MERVA system.

Recovering In-Process Messages (RECOVER)

The **recover** (or **ipcopy**) command is used to copy in-process (IP) messages of a specific inoperable or permanently waiting ASP to a send queue of another message transmission medium (for example, for the telex or the SWIFT network). The command is not accepted for an ASP in OPEN status.

The **recover** command is a privileged command. It can be issued only by an authorized operator.

The **recover** command modifies information in the IP messages in the applicable control queue (set PDM indicator), and processes a copy of each IP message:

- Set message class to RC
- Reset MERVA Link control fields
- Route message as specified by the MERVA ESA Routing Table associated with the applicable control queue.

Any error in this process is reported in the ASP list line.

Successful processing of a **recover** command is indicated by an information message that contains the number of IP messages that were recovered. If the command was not successful, command error diagnostic information can be contained in the applicable ASP list line (MT status code "--" and error codes in the format xx-yyy in the diagnostic code field). This command error diagnostic information is explained in *MERVA for ESA Messages and Codes*.

When the ASP has become operational again, the recovered IP messages are processed by MERVA Link. They are delivered to the receiving application with the MERVA Link PDM indicator. This means, the MERVA Link control field EKAPDUPM contains the characters *PDM*. This indicator can be checked in a routing table to handle recovered messages (that may be duplicate) differently from other messages.

Command Format

The format of the **recover** command is:

recover **ipco**py	*aspname*

Parameter Descriptions

aspname
The name of a specific ASP.

Command Example

To recover an ASP, named AP3, enter:

```
recover ap3
```

Refreshing ASP List in the IMS Environment (REFRESH)

This command allows you to refresh the data displayed in panel AC01 with the most up to date information in the Last Confirmed Control Messages of all ASPs. In the CICS environment, this update is performed automatically. In the IMS environment this update is not performed automatically if MERVA Link send tasks and the DSLEUD are scheduled in different IMS Message Processing Regions (MPRs).

The MSN of the last received message is not updated as response to this command. It is updated when information of the specific ASP is displayed in panel AC02.

Command Format

The format of the **refresh** command is:

refresh **ra**	

The **refresh** command has no parameter.

Modifying the Processing Environment Parameters (SET and RESET)

Use **set** and **reset** commands to modify temporarily a subset of the MERVA Link processing environment parameters specified in the EKAPT TYPE=INITIAL macro in the CICS environment. This command is also used to modify temporarily a subset of the MERVA Link ASP definition parameters specified in an EKAPT TYPE=ASP macro in the CICS environment.

Note: As the default, these commands are *not* supported in the IMS environment. The command **enable set** enables the **set** command in the IMS environment for specific purposes only.

The **set** and **reset** commands are privileged commands. They can be issued only by an authorized operator.

The **set** and **reset** commands can be entered in any panel of the MERVA System Control Facility. The command response is an updated display of the current panel (AC01, AC02, AC03, or AC04), or an unchanged redisplay of the current panel (ACMM, AC00).

An incorrect command option or other **set** or **reset** command processing errors result in the display of an error message in the current panel. The error message is displayed in panel AC01, in spite of the panel in which the command was entered, if the command applies to an ASP.

Command Format

The format of the **set** and **reset** commands is:

set	{am | test | xc | ctrcf | ctrcw} | {snap | psnap | accept | iprecov | wsz}
reset	{test | ctrc | accept | iprecov | am}

Parameter Descriptions

Both commands have a keyword as a mandatory parameter. You can enter the following commands:

set test
Corresponds to the parameter TEST=ON in the EKAPT TYPE=INITIAL macro.

set xc Corresponds to the parameter TEST=EXCEPTION in the EKAPT TYPE=INITIAL macro.

set ctrcf
Corresponds to the parameter TRACE=(,FULL) in the EKAPT TYPE=INITIAL macro, and enables the conversation trace identifier specified in the TRACE parameter of the EKAPT TYPE=INITIAL macro.

set ctrcw
Corresponds to the parameter TRACE=(,WEAK) in the EKAPT TYPE=INITIAL macro, and enables the conversation trace identifier specified in the TRACE parameter of the EKAPT TYPE=INITIAL macro.

set snap
Asks for a SNAP dump of the MSC task in your local MERVA system

when the following MSC command has been processed. Only one SNAP dump is written as response to a **set snap** command.

When this command is entered after an MSC switch to a partner MERVA system, the SNAP dump still applies to the local MERVA system.

This command is supported in the CICS and IMS environments. Issue the command **enable set** to enable the **set snap** command in the IMS environment.

set psnap
Asks for a SNAP dump of the MSC task in the currently applicable partner MERVA system when the following MSC command has been processed. It has no effect if MSC has not switched to a partner MERVA system. Only one SNAP dump is written as response to a **set psnap** command.

This command is supported in the CICS and IMS environments. Issue the command **enable set** to enable the **set psnap** command in the IMS environment.

set accept
Corresponds to the parameter ACCEPT=WARNING in the EKAPT TYPE=ASP macro. This command applies to the first ASP displayed in an ASP list (if entered in panel AC01) or to the identified ASP (if entered in panel AC02).

set iprecov
Corresponds to the parameter IPRECOV=AUTOMATIC in the EKAPT TYPE=ASP macro. This command applies to the first ASP displayed in an ASP list (if entered in panel AC01) or to the identified ASP (if entered in panel AC02).

set am Starts the ASP Monitor if the ASP Monitor transaction ID and a nonzero restart time interval are contained in the PT header.

The ASP Monitor restart time interval (00 .. 59) can be added to the command parameter **am**. The command **set am05** means, for example, that the ASP Monitor restart time interval is set to five minutes and the ASP Monitor must be started. The command **set am00** means that the ASP Monitor restart time interval is set to zero and the ASP Monitor must be stopped.

set wsz
Scans through all ASPs defined in the PT. For all ASPs in HOLD status, the window size in the LC control message and the window size specified in the PT ASP entry are compared. If the window sizes do not match, the LC control message is updated to match the window size in the PT ASP entry.

An ASP in NOHOLD status is not affected by this command. The same applies if no LC control message is found in the control queue of an ASP.

This command is supported in the CICS and IMS environments. Issue the command **enable set** to enable the **set wsz** command in the IMS environment.

reset test
Corresponds to the parameter TEST=OFF in the EKAPT TYPE=INITIAL macro.

reset ctrc
Corresponds to a missing TRACE parameter in the EKAPT TYPE=INITIAL

macro, and disables the conversation trace identifier specified in the TRACE parameter of the EKAPT TYPE=INITIAL macro.

Note: You cannot disable the conversation trace identifier that is specified in an EKAPT TYPE=MTP macro and therefore associated with a specific sending MTP only. The conversation trace types FULL and WEAK, however, also apply to that specific sending MTP trace.

reset accept
Corresponds to the parameter ACCEPT=OK in the EKAPT TYPE=ASP macro. This command applies to the first ASP displayed in an ASP list (if entered in panel AC01) or to the identified ASP (if entered in panel AC02).

reset iprecov
Corresponds to the parameter IPRECOV=MANUAL in the EKAPT TYPE=ASP macro. This command applies to the first ASP displayed in an ASP list (if entered in panel AC01) or to the identified ASP (if entered in panel AC02).

reset am
Sets the ASP Monitor restart time interval to zero and stops the ASP Monitor.

Chapter 8. Operating MERVA Link USS

The MERVA Link USS Application Control Command (ACC) application provides commands to operate MERVA Link USS in an OS/390 USS shell environment. A subset of these commands can also be used in an USS or MVS batch environment.

Overview of the MERVA Link Control within ACC

The MERVA Link USS Application Control Command application can be used by any authorized user to display MERVA Link resources and to modify MERVA Link processing parameters in a USS command shell environment. The ACC functions are, for example:

- Display ACC help information
- Display ACT header information
- Display a list of the partner nodes (ISC parameter sets) defined in the ACT
- Display static and dynamic information of an intersystem connection to a specific partner node
- Analyze and explain error information in the ACT
- Explain MERVA Link ESA error information
- Request ACD termination.

The requested ACC function is entered as the first parameter of the USS shell command **ekaacc**. This parameter is called the ACC command name. Additional command parameters may be applicable.

ACC provides more than 75 commands. Only the subset of the most important ACC commands is described in the following. Enter **ekaacc** without a parameter to get a list of the full set of ACC commands.

ACC Execution Environment

The MERVA Link USS Application Control Command (ACC) application can be used in an OS/390 USS interactive command shell environment, in an USS batch environment (BPXBATCH), and in an MVS batch environment. The command or program **ekaacc** calls ACC in the OS/390 USS environments. Program **EKAACC** calls ACC in the MVS environment.

An ACC interactive command begins with an ACC command name that must be entered in lowercase letters. A subset of the ACC command names is followed by a resource name. Resource names can be, for example, an ASP name, a partner system node name, a keyword, or a MERVA Link diagnostic code. MERVA Link resource names are normally made of uppercase characters. They can, however, be entered with lowercase letters. An uppercase translation is performed automatically (where applicable). Help information is displayed as response to an invalid ACC command.

A conversation mode is supported by ACC. The USS command **ekaacc sc** starts the ACC conversation mode in a window. All data entered in a window in ACC conversation mode is interpreted as an ACC command. The command prompt **ekaacc** is a reminder of the fact that the window is in ACC conversation mode.

A batch input mode is supported by ACC. The program calls **ekaacc si** (BPXBATCH) and **EKAACC si** (MVS) start ACC in batch input mode. The ACC commands are retrieved from **stdin**, and the command output is written to **stdout**.

ACC cannot be used when the MERVA Link USS Daemon EKAACD is not active. EKAACD generates and owns the MERVA Link Application Control Table (EKAACT) which is the main resource of the MERVA Link USS Control Facility.

The ACC commands and other subjects related to the MERVA Link USS Application Control Command application are described in the following.

Note: The support of ASP related functions has not been removed from the ACC implementation in the OS/390 USS environment. ASPs are, however, supported by the MERVA Link USS Gateway for installation verification and test purposes only. All functions related to ASPs are therefore not applicable in a production environment.

ACC Command Format

An ACC command consists of a command name and a resource name. The resource name is empty in a subset of the ACC commands. An ACC command name consists of up to four characters.

The majority of the ACC command names is structured according to the following rules:

- The first letter identifies a command activity, for example, **a**nalyze, **c**hange, **d**isplay, **h**elp, **l**ist, send **p**robe, **r**eset, **s**et, **s**tart, and **t**erminate.
- A command activity can also be identified by two characters of a command name, for example, **cx** (configuration export) and **dx** (display explanation).
- The character behind the activity indicator identifies a data class (if it is not the last character of the command name). Data classes are, for example, **e**rror code vector, **p**arameter, **r**eceiving process, **r**outing process, **s**ending process, **s**tatus information, and **t**race information.
- The last character of the command name often identifies an object or an object class, for example, **a** (ACT ASP entry), **c** (ACT ISC entry), **h** (ACT header), **d** (diagnostic code), and **e** (error information or error class).

ACC Command Groups

The set of ACC commands is divided into ACC command groups for documentation purposes. The commands of a command group are functionally related. There is an ACC **help** command for each command group that describes the commands in that group at medium detail level.

The ACC commands are listed and explained in the following in alphabetic sequence of the ACC command group names.

The ACC Commands

Perform an Error Analysis

The **analyze** commands provide a means to get an explanation of error information stored in the ACT. This explanation is supposed to be related to the application process rather than to the MERVA Link internal processing structure.

Analyze MERVA Link Process Error Information (a)

The ACC command **a** scans the ACT for error information that is associated with a

- MERVA Link unidentified receiving process
- MERVA Link sending process
- MERVA Link receiving process
- MERVA Link routing process.

It displays a list of the processes that have failed. Each line of that list applies to a MERVA Link process and contains the ACC command that can be used to analyze that specific MERVA Link process error.

Analyze Unidentified Receiving Process (aih)

The ACC command **aih** analyzes MERVA Link receiving process error information in the ACT header. The error information is associated with an inbound conversation that failed before the identity of the originator or the recipient could be determined.

Analyze Receiving Process (ara asp_name)

The ACC command **ara asp_name**, with the name of a local MERVA Link ASP as the ACC resource name, analyzes a MERVA Link receiving process error. It displays the explanation of a number of receiving process error codes contained in the specified ACT ASP entry and in the ACT ISC entry that is associated with the specified ASP entry.

Analyze Routing Process (arc pnode_name)

The ACC command **arc pnode_name**, with the name of a partner MERVA Link node as the ACC resource name, analyzes a MERVA Link routing process error. The partner node name specifies the destination of the routing process. The **arc** command displays the explanation of a number of routing process error codes contained in the specified ACT ISC entry.

Analyze Sending Process (asa asp_name)

The ACC command **asa asp_name**, with the name of a local MERVA Link ASP as the ACC resource name, analyzes a MERVA Link sending process error. It displays the explanation of a number of sending process error codes contained in the specified ACT ASP entry, and its associated ACT ISC entry (if applicable).

Export MERVA Link USS Configuration

The ACC commands **cxf path_name** (ACT configuration export to a single file) and **cxs path_name** (ACT configuration export to separate files) export the customization parameters that are currently contained in the ACT to one or more HFS files. The exported data has a format that is accepted by the configuration import functions used by the ACD.

The fully qualified path name (beginning with '/') of the output file can be specified as a command parameter. A relative path name specified as output file name is based on the configuration subdirectory of the MERVA instance directory, for example, **/u/merva1/cfg/**. If a file name is not specified as a command parameter, the file **ekaact.cfg** is the default. The default name for the sample output file is **/u/merva1/cfg/ekaact.cfg**.

You must be the ACD process owner or a root user with uid=0 to execute the configuration export commands. Read and write permissions are granted for the file owner. Members of the owning group and other USS users are authorized to read that file.

A configuration export command is rejected if this configuration exists already. To replace an existing configuration file, you must use the **replace** versions **cxfr** and **cxfs** of the configuration export command.

Export Configuration to a Single File (cxf cfg_file_path)

The ACT configuration is exported to a single file that contains the parameters of the ACT header and all ACT entries. The default file name is **ekaact.cfg** in the subdirectory **/cfg** of the applicable MERVA USS instance directory. The sample configuration export file name is **/u/merva1/cfg/ekaact.cfg**.

Export Configuration to Separate Files (cxs cfg_file_path)

The ACT configuration is exported to a main file and to a set of configuration include files, one for each ACT entry. The main file contains the parameters of the ACT header and includes statements for all ACT entry parameter files. A configuration include file is generated for each ACT entry. It contains the parameters of one ACT ASP or one ACT ISC entry.

The default main file name is **ekaact.cfg** in the subdirectory **/cfg** of the applicable MERVA USS instance directory. The sample main configuration export file name is **/u/merva1/cfg/ekaact.cfg**.

The configuration include files are written to the same directory as the main configuration file, for example, **/u/merva1/cfg/**.

- The name of an include file for an ACT ASP entry consists of **cfga.** followed by the ASP name, for example, **/u/merva1/cfg/cfga.ASP1**.
- The name of an include file for an ACT ISC entry consists of **cfgc.** followed by the partner node name, for example, **/u/merva1/cfg/cfgc.PNODE1**.

Display Export Configuration Help Information

An invalid ACC command starting with **cx** causes Configuration Export Help information to be displayed. The ACC command **hcx** displays the same help information.

Display Parameters and Resource Status Information

The **display** commands provide a means to display parameters of a resource and to display status information associated with a resource.

Display ASP Parameters (dpa asp_name)

The ACC command **dpa asp_name**, with the name of a local MERVA Link ASP as the ACC resource name, displays a set of customization parameters associated with the specified ASP. This parameter set includes the following ASP parameters:

- Local ASP name
- Partner ASP name
- Partner node name
- Send queue name(s).

The ACC command **dpa *** displays the customization parameters of all ASPs defined in the ACT.

Display ISC Parameters (dpc partner_node_name)

The ACC command **dpc partner_node_name**, with the name of a partner MERVA Link node as the ACC resource name, displays the set of inter-system communication (ISC) parameters that is used to connect to the specified partner MERVA Link node. This parameter set can include the following parameters:

- Partner MERVA Link node name

- Partner SNA APPC symbolic destination name
- Partner SNA APPC TP name
- Partner TCP/IP host name
- Partner TCP/IP port number
- Conversation security user ID
- Conversation security user password information.

The ACC command **dpc *** displays the customization parameters of all partner nodes defined in the ACT.

Display ACT Header Parameters (dph)

The ACC command **dph** displays static information that is contained in the header of the MERVA Link Application Control Table (ACT). This information includes:

- The local MERVA Link node name
- The ACT generation date/time stamp
- The number of ASP entries in this ACT
- The number of ISC entries in this ACT.

Display ASP Status Information (dsa asp_name)

The ACC command **dsa asp_name**, with the name of a local MERVA Link ASP as the ACC resource name, displays status information that applies to the specified ASP. Error codes are shown and explained.

Display ISC Status Information (dsc partner_node_name)

The ACC command **dsc partner_node_name**, with the name of a partner MERVA Link node as the ACC resource name, displays information related to the intersystem communication with the specified partner MERVA Link node. This information is divided in four groups that are associated with the four MERVA Link USS TPs EKATPO, EKATCO, EKATPI, and EKATCI.

The ISC status information for each TP can include:

- The TP's last activity date/time stamp
- The TP return code
- The TP reason code
- The USS or SNA error number.

Error codes are shown and explained.

Display MERVA Link Daemon Status and Activity Trace (dsd)

The ACC command **dsd** displays information related to the MERVA Link daemon EKAACD. This information is contained in the header of the MERVA Link Application Control Table (ACT). This information includes:

- The EKAACD process identifier
- The time of the daemon's latest activity
- The EKAACD activity trace.

The EKAACD activity trace consists of up to 16 operator messages. Each of these operator messages is preceded by the applicable time stamp in the format HH:MM:SS. The message text explains a specific activity performed by the daemon.

The EKAACD activity trace area in the ACT header is used in wrap-around mode. This means, you can see 16 of the most recent EKAACD activity messages if the

last message issued by EKAACD is stored at the end of the EKAACD activity trace area. If the last message issued by EKAACD is not stored at the end of the EKAACD activity trace area, you can see up to 19 activity messages. A trace area wrap is indicated by a separator line (<----------->).

Display ACT Header Status Information (dsh)

The ACC command **dsh** displays status information that is contained in the header of the MERVA Link Application Control Table (ACT). This information can include:

- The local MERVA Link node name
- The ACT generation date and time stamp
- The unidentified EKATPI error information
- The unidentified EKATCI error information
- The unidentified EKAP1I error information.

Error codes are shown and explained.

Unidentified error information is stored by a MERVA Link receiving process in the ACT header when an error is found before the identity of the partner could be determined. As soon as the originator's identity is known and the corresponding ACT ISC entry is identified, receiving process error information is stored in that ACT ISC entry.

Explain Error Information

The **display explanation** commands (**dx..**) provide a means to display an explanation for error information. These commands can be grouped as follows:

- The **dxc** and **dxd** commands that try to explain error information in an ACT ISC entry and a diagnostic code, respectively
- The **dxm** commands that can explain a subset of the return and reason codes of MERVA ESA V4 services
- The **dxr** and **dxs** commands that try to explain MERVA Link error code vectors for sending and receiving processes at the MT and AS layers.

Note: The display explanation commands can explain all MERVA Link ESA error codes, but only a subset of the MERVA ESA V4 service return and reason codes. Refer to the *MERVA for ESA Messages and Codes* manual for an interpretation of the full set of codes.

Display Error Code Vector Explanation (dxc partner_node_name)

The ACC command **dxc partner_node_name** explains an internal MERVA Link ESA error code vector that may be stored in the ACT ISC entry for the specified partner node. It can explain a major subset of the error code vectors that can be included by MERVA Link ESA receiving processes in an error report.

A MERVA Link ESA inbound process error code vector consists of a sequence of six halfword (unsigned short int) error codes EC1 - EC6:

1. If EC1 is not 0008, the error was found in the MERVA Link inbound TP (EKATR10 or EKATPI1). EC1 is the return code of program EKATR10 or EKATPI1. The meaning of EC2 to EC4 is specified by the latter programs. EC1 = 0008 means that the error is reported by the MTSP.
2. If EC1 is 0008 and EC2 is not 0008, the error was found in the MERVA Link inbound MTSP (EKASP10). EC2 is the return code of program EKASP10. The meaning of EC3 to EC5 is specified by the latter program. EC1 = 0008 and EC2 = 0008 means that the error is reported by the inbound AS program (EKAAR10).

3. If both EC1 and EC2 are 0008, the error was found in the MERVA Link inbound AS Program (EKAAR10). EC3 is the return code of program EKAAR10. The meaning of EC4 to EC6 is specified by the latter program.

There are always four possible error codes associated with the MERVA Link programs at the three layers TP, MT, and AS. A subset of these error codes may, however, be zero.

The **dxc** command cannot explain all possible error code vectors. The complete set of MERVA ESA and MERVA Link ESA error codes is explained in the *MERVA for ESA Messages and Codes* manual.

Display Diagnostic Code Explanation (dxd diag_code)

The ACC command **dxd diag_code** tries to explain the specified MERVA Link diagnostic code. It can explain all diagnostic codes generated by the MERVA Link USS programs, and a major subset of the diagnostic codes generated by MERVA Link AIX and MERVA Link NT. These diagnostic codes start with an origin identifier of two alphabetic characters. The diagnostic information follows in two codes of two hexadecimal characters each. A MERVA Link USS diagnostic code entered with less than 6 characters is automatically padded with zeros up to the length of 6 characters.

The **dxd** command can also explain all alphabetic diagnostic codes generated by MERVA Link ESA sending and receiving processes, parts of all other MERVA Link ESA sending process diagnostic codes, and some alphabetic MERVA USE Workstation programs diagnostic codes. It cannot explain MERVA Link ESA receiving process diagnostic codes originating from the MTP and MTSP.

A large set of diagnostic codes generated by MERVA Link AIX and MERVA Link NT matches the MERVA Link USS diagnostic codes.

Operating system error numbers may, however, be incorrectly interpreted by the MERVA Link USS **dxd** command.

A MERVA Link USS diagnostic code consists of:

1. Two letters identifying the MERVA Link USS program that generated the diagnostic code (DC origin).
2. Two hexadecimal characters (0 .. 9, A .. F) identifying the MERVA Link program return code.
3. Two hexadecimal characters (0 .. 9, A .. F) identifying the MERVA Link program reason code.

The meaning of the return code is determined by the first two letters of the diagnostic code (MERVA Link USS error class). The meaning of the reason code depends on the value of the return code.

A diagnostic code generated by MERVA Link ESA or by MERVA USE Workstation programs consists of six alphanumeric characters. It can be a mnemonic for a processing failure (for example, **DSL IN** if MERVA ESA is not operational in the partner system), or a sequence of three hexadecimal error codes in character representation (for example, **0872E4** if the recipient node name in the PROBE envelope does not match the local node name of the MERVA ESA system).

The diagnostic code that is entered as the parameter of the **dxd** command can contain one blank. It can consist of four characters if the last two characters have no effect on the explanation. The diagnostic codes **0872E4** and **0872**, for example, have the same explanation.

The explanation of a diagnostic code consists of one line explaining the source of the diagnostic code, and one or two lines explaining the meaning of the diagnostic code. The source of a diagnostic code can be one of the following:

- One of the MERVA Link USS, AIX, or NT message processing programs
- MERVA Link of MERVA ESA or MERVA USE Workstation programs
- MERVA Link of MERVA ESA
- MERVA Link of MERVA USE Workstation programs

The diagnostic codes that are defined in MERVA Link USS, and MERVA USE Workstation programs, are explained as diagnostic codes from a MERVA Link program **EKA**xxx. The diagnostic codes that are defined in both MERVA Link ESA and MERVA Link OS/2, are explained as diagnostic codes from MERVA ESA or MERVA USE Workstation programs.

Display Explanation of DSLMMFS RC and RS (dxmm rcrs)

The ACC command **dxmm** explains return and reason codes of the MERVA ESA MFS (DSLMMFS RC and RS). The RC and RS must be specified as the command parameter in one token.

If the command parameter starts with the characters **00**, its format must be the sequence of one or two codes, each consisting of four hexadecimal digits (0 - F). Otherwise, the codes must consist of two hexadecimal digits each.

Display Explanation of DSLNICP or DSLNICT RC (dxmn rc)

The ACC command **dxmn** explains return codes of the MERVA ESA NICP or NICT Services (DSLNICP and DSLNICT RC) and associated DSLRTNSC reason codes. The DSLNIC RC must be specified as two or four hexadecimal digits.

Display Explanation of DSLQMGT RC (dxmq rcrs)

The ACC command **dxmq** explains return codes of the MERVA ESA QMGT Services (DSLQMGT RC) and associated DSLRTNSC reason codes. The RC and RS must be specified as the command parameter in one token.

If the command parameter starts with the characters **00**, its format must be the sequence of one or two codes, each consisting of four hexadecimal digits (0 - F). Otherwise, the codes must consist of two hexadecimal digits each.

Display Explanation of DSLRTNSC RC and RS (dxmr rcrs)

The ACC command **dxmr** explains return and reason codes of the MERVA ESA Routing Scanner (DSLRTNSC RC and RS). The RC and RS must be specified as the command parameter in one token.

If the command parameter starts with the characters **00**, its format must be the sequence of one or two codes, each consisting of four hexadecimal digits (0 - F). Otherwise, the codes must consist of two hexadecimal digits each.

Display Explanation of DSLTOFSV RC and RS (dxmt rcrs)

The ACC command **dxmt** explains return and reason codes of the MERVA ESA TOF Supervisor Services (DSLTOFSV RC and RS). The RC and RS must be specified as the command parameter in one token.

If the command parameter starts with the characters **00**, its format must be the sequence of one or two codes, each consisting of four hexadecimal digits (0 - F). Otherwise, the codes must consist of two hexadecimal digits each.

Display Explanation of RP ECV Type AS (dxra ar_ecv)

The ACC command **dxra** explains an Error Code Vector (ECV) that is associated with a MERVA Link ESA receiving process error of the type AS. The ECV must be specified as the command parameter in one token.

If the command parameter starts with the characters **00**, the format of the ECV must be the sequence of up to four error codes each consisting of four hexadecimal digits (0 - F). Otherwise, all error codes in the ECV must consist of two hexadecimal digits.

For more information about MERVA Link ESA receiving process ECVs, refer to the description of the ACC command **dxc**.

Display Explanation of RP ECV Type MT (dxrm mt_ecv)

The ACC command **dxrm** explains an Error Code Vector (ECV) that is associated with a MERVA Link ESA receiving process error of the type MT. The ECV must be specified as the command parameter in one token.

If the command parameter starts with the characters **00**, the format of the ECV must be the sequence of up to six error codes each consisting of four hexadecimal digits (0 - F). Otherwise, all error codes in the ECV must consist of two hexadecimal digits.

For more information about MERVA Link ESA receiving process ECVs, refer to the description of the ACC command **dxc**.

Display Explanation of SP ECV Type AS (dxsa as_ecv)

The ACC command **dxsa** explains an Error Code Vector (ECV) that is associated with a MERVA Link ESA sending process at the AS layer. This type of ECV can, for example, be displayed as the sending ASP diagnostic code in the MERVA Link ESA MSC screens AC01 and AC02. It can also be part of the MERVA Link operator message EKA701E. The ECV must be specified as the command parameter in one token.

If the command parameter starts with the characters **00**, the format of the ECV must be the sequence of up to four error codes each consisting of four hexadecimal digits (0 - F). Otherwise, all error codes in the ECV must consist of two hexadecimal digits.

Display Explanation of SP ECV Type MT (dxsm mt_ecv)

The ACC command **dxsm** explains an Error Code Vector (ECV) that is associated with a MERVA Link ESA sending process at the MT layer. This type of ECV can, for example, be displayed as **Error Code Vector ... in LC Ctrl Message** in a MERVA Link ESA MSC screen AC02. The ECV must be specified as the command parameter in one token.

If the command parameter starts with the characters **00**, the format of the ECV must be the sequence of up to four error codes each consisting of four hexadecimal digits (0 - F). Otherwise, all error codes in the ECV must consist of two hexadecimal digits.

An ECV that contains the first two bytes of a CICS EIBRCODE, for example, **D008**, must be entered as a sequence of 4-character error codes. ECV **00080022D008**, for

example, means that the partner system is not available. The CICS EIBRCODE cannot be interpreted in the short form of this ECV (**082208**).

An ECV that contains APPC/MVS return and reason codes that are identified by **AC..** and **AE..**, respectively, can, however, be entered as 2-character error codes. The APPC/MVS return and reason code indicators are not required for the interpretation of these codes. An ECV that contains an SNA sense code starting with **08** can also be entered as a sequence of 2-byte error codes. The two hexadecimal digits 08 are added in this case before the SNA sense code is interpreted. ECV **000A0010AC020857**, for example, means that all available sessions to the partner system are in use and the allocation of a new session failed because the required SSCP-LU session is not active. The short form of this ECV (**0A100257**) is also valid and has the same explanation.

The **dxsm** command can explain only a subset of all possible MERVA Link ESA sending MTP error code vectors. Sending MTP error code vectors are considered as MERVA Link ESA internal information for problem determination, and not described in MERVA Link documentation.

Display Explanation Help Information

An invalid ACC command starting with **dx** causes Display Explanation Help information to be displayed. The ACC command **hdx** displays the same help information.

Show ACC Command Help Information

The **help** commands provide a means to display information regarding the ACC commands. One of the help screens is displayed when an invalid ACC command is entered.

Show ACC Command Summary Help Screen (h)

The ACC command **h** displays a summary of ACC commands. This help information is automatically displayed if you enter an invalid ACC command, and no specific help screen is associated with that invalid ACC command.

Show Analyze Command Help Screen (ha)

The ACC command **ha** displays an explanation of the ACC **analyze** commands. This help information is automatically displayed if you enter an invalid ACC command that starts with **a**.

Show Change Command Help Screen (hc)

The ACC command **hc** displays an explanation of the ACC **change** commands. This help information is automatically displayed if you enter an invalid ACC command that starts with **c**.

Show Configuration Export Command Help Screen (hcx)

The ACC command **hcx** displays an explanation of the ACC **configuration export** commands. This help information is automatically displayed if you enter an invalid ACC command that starts with **cx**.

Show Display Command Help Screen (hd)

The ACC command **hd** displays an explanation of the ACC **display** commands. This help information is automatically displayed if you enter an invalid ACC command that starts with **d**.

Show Display Explanation Command Help Screen (hdx)

The ACC command **hdx** displays an explanation of the ACC **display explanation** commands. This help information is automatically displayed if you enter an invalid ACC command that starts with **dx**.

Show List Command Help Screen (hl)

The ACC command **hl** displays an explanation of the ACC **list** commands. This help information is automatically displayed if you enter an invalid ACC command that starts with **l**.

Show ASP Operating Command Help Screen (hm)

The ACC command **hm** displays an explanation of the local and partner ASP operating commands. This help information is automatically displayed if you enter an incomplete partner ASP operating command, for example, **pda** without the ASP name, or an invalid ACC command that starts with **m** or **k**.

Show Send Probe Command Help Screen (hp)

The ACC command **hp** displays an explanation of the ACC **send probe** commands. This help information is automatically displayed if you enter an invalid ACC command that starts with **p**.

Show Reset Command Help Screen (hr)

The ACC command **hr** displays an explanation of the ACC **reset** commands. This help information is automatically displayed if you enter an invalid ACC command that starts with **r**.

Show Set Command Help Screen (hs)

The ACC command **hs** displays an explanation of the ACC **set** and **start** commands. This help information is automatically displayed if you enter an invalid ACC command that starts with **s**.

Show Extended ACC Command Help Screen (hx)

The ACC command **hx** displays an explanation of all ACC commands in a list. The size of this list is most probably larger than the size of the display window. You cannot see the beginning of that list if you enter the **hx** command in ACC conversation mode. To get the complete list displayed in separate pages, issue the command **ekaacc hx | pg** in the USS command shell environment.

List Resource Groups

The **list** commands provide a means to display lists of resource groups in the ACT. Resource groups are, for example, all ACT resources, the ACT ASP entries, the ACT ISC entries, and the message transfer rates of sending ASPs.

The size of an ASP or Partner Node list can be limited by specifying a resource name subset as **list** command parameter. An ASP list is, for example, limited to the set of ASPs that start with the characters specified as list command parameter. The same applies for a list of partner nodes (ACT ISC entries).

The resource name subset can be specified in lowercase characters as the list command parameter. It is automatically translated to uppercase characters before it is interpreted as an ASP or Partner Node name subset. A list command parameter is reset after handling a list command in ACC conversation mode. It is not used as default parameter of a following ACC command.

List ACT Entries (l)

The ACC command **l** displays the ACT name and the local MERVA Link node name from the ACT header, and lists all ASP and ISC entries defined in the ACT.

If a listed ACT resource (ACT header or ACT entry) does not contain error information, the ACC command that displays the parameters of that ACT is added at the end of the list line. If an ACT resource contains error information, the ACC command that explains the error information is added at the end of the list line. An alert indicator (==>) is contained in the list line of each ASP or ISC entry that contains error information.

List ASP Entries (la)

The ACC command **la** lists all ASP entries defined in the ACT. This list is a subset of the list obtained via the ACC command **l**.

List Active ASPs (laa)

The ACC command **laa** lists all sending and receiving ASPs defined in the ACT that are currently active as specified by the ASP active flag in the ACT.

List ISC Entries (lc)

The ACC command **lc** lists all ISC entries defined in the ACT. This list is a subset of the list obtained via the ACC command **l**.

List Routing Processes (lr)

The ACC command **lr** lists all ISC entries defined in the ACT, and tells whether a routing process is active with the subject destination node.

List Active Routing Processes (lra)

The ACC command **lra** lists all ISC entries defined in the ACT that are associated with an active routing process. A list line shows the partner node name, the total number of routing processes to the subject destination node, and the numbers of routing processes that use SNA and TCP/IP for outbound communication, respectively.

List MERVA Link Error Classes (le)

The ACC command **le** lists all MERVA Link USS internal error classes. Each error class is associated with a MERVA Link program or a subsystem that provides error codes of that class. The meaning of an error code can be determined only if its error class is known.

An error code that cannot be explained in any of the ACC **display status** and **analyze** commands, is identified in the command response with its error class. You can then refer to documentation of the applicable subsystem (for example, Communications Server or USS Kernel services) for an explanation of the error code.

List Message Transfer Rates (lx)

The ACC command **lx** displays the average message transfer rates of all three MERVA Link message transfer process subgroups:

- Receive message process
- Send message process
- Route message process.

The commands **lxi**, **lxo**, and **lxr** are provided to display a list concerning one of these groups only. These commands and the information in each sublist are described in the following sections.

All information in the ACT concerning message transfer rates can be reset to zero by the ACC command **rxi**. The time frame that applies to information concerning message transfer rates starts either when the current ACD process has started, or when the last **rxi** command has been entered.

List Receiving ASP Transfer Rates (lxi)

The ACC command **lxi** lists the inbound message transfer rates for all ASPs defined in the ACT. The indicator **no message received** appears in the list line of any ASP that has not received messages since the last startup of the ACD.

The calculation of the message transfer rate displayed for a particular ASP is based on the total number of messages received and the accumulated duration of the receiving processes since the last start of the ACD. The transfer rate is shown in units of messages per hour. The total number of messages received since the last ACD start is displayed as part of the ASP status information (see command **dsa**).

List Sending ASP Transfer Rates (lxo)

The ACC command **lxo** lists the outbound message transfer rates for all ASPs defined in the ACT. The indicator **no message sent** appears in the list line of any ASP that has not sent messages since the last startup of the ACD.

The calculation of the message transfer rate displayed for a particular ASP is based on the total number of messages sent and the accumulated duration of the sending processes since the last start of the ACD. The transfer rate is shown in units of messages per hour. The total number of messages sent since the last ACD start is displayed as part of the ASP status information (see command **dsa**).

List Routing Process Transfer Rates (lxr)

The ACC command **lxr** lists the message transfer rates of the MERVA Link processes that route messages (from all source nodes) to the partner nodes defined in the ACT ISC entries. The indicator **no message routed** appears in the list line of any ISC entry (partner node) that has not received messages in a routing process since the last startup of the ACD.

The calculation of the message transfer rate displayed for a particular partner node is based on the total number of messages routed and the accumulated duration of the routing processes since the last start of the ACD. The transfer rate is shown in units of messages per hour. The total number of messages routed since the last ACD start is displayed as part of the ISC status information (see command **dsc**).

Reset Resource Characteristics

The **reset** commands provide a means to reset information in the ACT or to stop MERVA Link facilities. A subset of these commands has the opposite effect as the corresponding **set** commands.

Reset Error Information (re)

The ACC command **re** sets the error information in the ACT header and in all ACT ASP and ISC entries to zero. This command resets also the sending and receiving ASP active flags in all ASP entries, as well as the routing process active flags in all ISC entries.

Reset Sending ASP Trace (rta)

The ACC command **rta asp_name** requests that no processing trace is written by the sending MERVA Link processes that are associated with the specified ASP. The ASP name can be entered in lowercase characters.

Reset Receiving TCP/IP Process Trace (rtc)

The ACC command **rtc** requests that no processing trace is written by the receiving MERVA Link processes that are associated with an inbound TCP/IP conversation. This command has no command parameter.

Reset Trace Directory Path Name (rtd)

The ACC command **rtd** erases the trace directory path name in the ACT header. It provides a means to disable all MERVA Link processing traces in one place. This command has no command parameter. The ACC command **std** must be used to enable the MERVA Link trace facility again.

Reset Receiving SNA APPC Process Trace (rtp)

The ACC command **rtp** requests that no processing trace is written by the receiving MERVA Link processes that are associated with an inbound SNA APPC conversation. This command has no command parameter.

Reset Message Transfer Rate Information (rxi)

The ACC command **rxi** requests that all information in the ACT that is related to the message transfer statistics is erased. This command has no command parameter.

When this command has been entered, message transfer information (number of messages and duration of the corresponding MERVA Link processes) is collected and accumulated from zero values. Zero values apply also when the ACD has been started and a new ACT has been created by the ACD.

Start ACC in Special Mode

The **start** commands provide a means to start ACC in specific modes.

Start EKAACC Conversation Mode (sc)

The USS command **ekaacc sc** places the terminal in ACC conversation mode. In ACC conversation mode you are prompted by **ekaacc** to enter an ACC command (without typing the name of the command processor **ekaacc**). The ACC commands **x** and **end** end the ACC conversation mode. The Ctrl_C key that generates an interrupt signal (SIGINT) for the ACC process can be used in an rlogin shell to end ACC. The **sc** and **si** commands are invalid in the ACC conversation mode. Command **sc** terminates the ACC conversation mode. Command **si** starts and terminates the ACC batch input mode.

ACC supports up to three periods (.) in its conversation mode to refer to the command name and two parameters of the previous command. If the last command was **cmd1 p1 p2**, you can enter `..x2` to issue the command **cmd1 p1 x2** or enter `.x1 .` to issue the command **cmd1 x1 p2**. Enter `cmd2 .` to issue the command **cmd2 p1 p2**. The last command is repeated if you press the Enter key without data.

Start EKAACC Batch Input Mode (si)

The USS command **ekaacc si** starts an ekaacc process in ACC batch input mode. In batch input mode, the ACC commands are read from stdin without prompting. EOF on stdin and the ACC commands **x** and **end** end the ACC batch input mode. The **sc** and **si** commands are invalid in ACC batch input mode. Help information for the **set** commands is displayed as response to these commands if they were entered in batch input mode.

Set Resource Characteristics

The **set** commands provide a means to set information in the ACT. A subset of these commands has the opposite effect as the corresponding **reset** commands.

Set EKAACD Retry Time Interval (srt)

The ACC command **srt time_iv** sets the retry time interval of the MERVA Link daemon to the value specified in the command parameter **time_iv**. Retry time

interval values of one second to 3600 seconds (one hour) are accepted. A zero retry time interval means that the ACD must not kick off inoperable sending ASPs automatically.

Set EKAACD Sleep Time Interval (sst)

The ACC command **sst time_iv** sets the sleep time interval of the MERVA Link daemon to the value specified in the command parameter **time_iv**, and sends an alarm to the MERVA Link daemon. Sleep time interval values of one second to 3600 seconds (one hour) are accepted.

Set Sending ASP Trace Level (sta)

The ACC commands **sta asp_name**, **sta1 asp_name**, **sta2 asp_name**, **sta3 asp_name**, and **sta9 asp_name** request that a processing trace is written by the sending MERVA Link processes that are associated with the specified ASP. The ASP name can be entered in lowercase characters.

The trace level that can be specified as the fourth character of the command name can be 0, 1, 2, 3, or 9. The default trace level is 1. With trace level 3, all information that can be written to a trace file is actually included in the trace file (general control information, message envelopes, message headings, status reports, and message text). With trace level 2, all level-3 information except the message text is included in the trace file. With trace level 1, all level-2 information except the message envelopes, message headings, and status reports is included in the trace file.

Trace level 9 requests a trace for performance analysis. A level-9 trace contains the initial and final process information that is also contained in a level-1 trace. The process activity information consists only of service primitive and processing time information at the P2 layer.

The ACC command **sta0 asp_name** requests that no processing trace is written by the specified sending ASP.

Set Receiving TCP/IP Process Trace Level (stc)

The ACC command **stc** requests that a processing trace is written by the receiving MERVA Link processes that are associated with an inbound TCP/IP conversation. This command has no command parameter.

The trace level that can be specified as the fourth character of the command name can be 0, 1, 2, 3, or 9. The default trace level is 1. For more information about the trace level refer to the description of the ACC command **sta asp_name**.

The ACC command **stc0** requests that no processing trace is written by a receiving TCP/IP process.

Set Trace Directory Path Name (std)

The ACC command **std dir_path_name** specifies the trace directory path name that is stored in the ACT header. The trace directory path name must start and end with a forward slash (/), and must not exceed 20 characters.

Trace files generated by MERVA Link programs are filed in the USS directory that is named in the ACT header. No trace files are generated if this directory path name is empty.

Set Receiving SNA APPC Process Trace Level (stp)

The ACC command **stp** requests that a processing trace is written by the receiving MERVA Link processes that are associated with an inbound SNA APPC conversation. This command has no command parameter.

The trace level that can be specified as the fourth character of the command name can be 0, 1, 2, 3, or 9. The default trace level is 1. For more information about the trace level refer to the description of the ACC command **sta asp_name**.

The ACC command **stp0** requests that no processing trace is written by a receiving SNA APPC process.

Set External EKAACD Trace Level (stx)

The ACC command **stx** controls the external trace written by the MERVA Link daemon EKAACD. It has no effect if the MERVA Link daemon is started without the external trace option.

The ACC command **stx0** requests that the external ACD trace is temporarily disabled. The ACC command **stx** with a fourth character different from 0 requests that the external ACD trace is reactivated.

The ACC command **stx** has no effect on the internal trace written by the MERVA Link daemon EKAACD. A trace level switch is, however, recorded in the internal ACD trace.

Set Trace File Wrap Limit (swa swc swp)

The ACC commands **swa**, **swc**, and **swp** set the processing trace-file wrap limit for an ASP, for inbound TCP/IP processes, and for inbound SNA APPC processes. The value of the wrap limit is specified like the trace level in the **st***x* commands. It can be 0, 1, 2, 3, or 9.

The trace file names contain date and time information if the wrap limit is zero. Trace files written by MERVA Link processes are saved without limits in MERVA Link in this case.

If the wrap limit is 1 or greater than 1, trace file names are generated in wrap-around mode. Trace file names contain a wrap index rather than date and time information. The maximum number of trace files for a trace class (a, c, p) matches the trace file wrap limit. Trace files written by MERVA Link processes are overwritten when the trace file wrap index exceeds the specified limit.

Terminate the MERVA Link USS Daemon

The ACC commands **trm EKAACD** and **trm daemon** request the termination of the MERVA Link USS Daemon. Only the owner of the ACT, the ACD process owner, or a root user can issue these commands and terminate the ACD.

Chapter 9. Operating Financial Message Transfer/ESA (FMT/ESA)

MERVA-to-MERVA Financial Message Transfer/ESA (FMT/ESA) uses the capabilities of MERVA Link or MQI Attachment to transfer SWIFT messages between two MERVA ESA systems in a way similar to the way MERVA ESA transfers messages via the SWIFT network.

According to the SWIFT message protocol, sent messages contain an input sequence number (ISN), and received messages contain an output sequence number (OSN). The FMT/ESA automatically assigns an ISN or an OSN to a message.

ISNs and OSNs for the FMT/ESA function are maintained by means of two dedicated queues:

ISN control queue
This queue has the name EKAISNCQ (for MERVA Link) or DSLISNCQ (for MQI Attachment), and contains one or more ISNs. A separate ISN is counted for each:

- MERVA Link ASP in which FMT/ESA processes SWIFT input messages (when using FMT/ESA with MERVA Link)
- MQI Attachment send process in which FMT/ESA processes SWIFT input messages (when using FMT/ESA with MQI Attachment)

OSN control queue
This queue has the name EKAOSNCQ (for MERVA Link) or DSLOSNCQ (for MQI Attachment), and contains one OSN. The OSN is counted globally across all:

- MERVA Link ASPs in which the FMT/ESA generates SWIFT output messages (when using FMT/ESA with MERVA Link)
- MQI Attachment receive processes in which the FMT/ESA generates SWIFT output messages (when using FMT/ESA with MQI Attachment)

FMT/ESA handles both ISNs and OSNs:

- If a sequence number does not yet exist, it initializes it with '000001'.
- It increments a sequence number up to '999999'.
- When an overflow occurs, it continues counting with '000001'.

You can do the following things with or to the sequence numbers used:

- You can initialize each ISN.
- For existing ISNs and the OSN, you can perform the following actions:
  - Display the sequence number
  - Modify the sequence number
  - Print the sequence number
  - Delete the sequence number

Note: Do not modify or delete a sequence number while FMT/ESA is running, as this can cause unpredictable results.

The following examples show the panels you use when operating FMT/ESA with MERVA Link. The panels used when operating FMT/ESA with MQI Attachment are not shown, but they are similar.

Initializing an Input Sequence Number

To initialize an ISN:

1. Assign the ISN Control Message function EKAISNCQ (for FMT/ESA with MERVA Link) or DSLISNCQ (for FMT/ESA with MQI Attachment) to a MERVA ESA user using the MERVA ESA user file maintenance. Sign on to MERVA ESA with this user identification.
2. From the MERVA ESA Function Selection menu, select the ISN Control Message function EKAISNCQ (for FMT/ESA with MERVA Link) or DSLISNCQ (for FMT/ESA with MQI Attachment). The Message Selection panel is displayed.
3. Enter the message type of the ISN control message:
   - For FMT/ESA with MERVA Link: **mt isnsim**
   - For FMT/ESA with MQI Attachment: **mt kisn**

   Figure 56 shows how the message type is entered on the Message Selection panel for FMT/ESA with MERVA Link.

```
                          Message Selection                Func EKAISNCQ

       Message Selection Commands

       Create New Message (Message Type)      MT nxxxxxxx
       Select Next Message                    GET NEXT or GET N
       Select First Message                   GET FIRST or GET F
       Select Message by Key 1 EKAASP         GET <K1> x...x
       Select Message by Key 2 EKAAMSID       GET K2 x...x

       Message Processing Commands (for use on message panels)

       Escape From Current Message            ESCAPE
       Requeue Current Message                REQUEUE
       Complete and Store Message             EOM
       Delete Current Message                 DELETE




Command =====> mt isnsim
PF 1=Help      2=Retrieve  3=Return     4=           5=Get Next  6=Get First
PF 7=          8=          9=Hardcopy  10=          11=List     12=
```

Figure 56. Entering Message Type for ISN Control Message

4. Press the ENTER key. Figure 57 on page 191 shows the ISN Control Message panel that is displayed for FMT/ESA with MERVA Link.

```
MT ISNSIM    ISN Control Message for Financial Message Transfer  Page 00001
                                                                 Func EKAISNCQ

             ASP Name: ________   MERVA Link Application Support Process

             ISN . . : ______     Input Sequence Number









Command =====>
PF 1=Help      2=Retrieve  3=EOM        4=Repeat     5=Get Next  6=Requeue
PF 7=Page -1   8=Page +1   9=Hardcopy  10=Pro Line  11=Prompt   12=Escape
```

Figure 57. Input Sequence Number Data Entry Panel

5. For FMT/ESA with MERVA Link, enter in the ASP Name field the name of the ASP. The name of the ASP must be the same as the name specified for this ASP in the MERVA Link partner table. In this example, the name of the ASP is **asp1**.

   For FMT/ESA with MQI Attachment, enter the name of the send process in the Process Name field. The name of the process must be the same as the name specified in the MQI Attachment process table.
6. Enter a value that is one less than the ISN you want to use. In this example, you want the initial ISN to be 1000, so enter **999** in the ISN field (you do not need to enter leading zeros).
7. Press the ENTER key. Figure 58 on page 192 shows the ISN Control Message panel that is redisplayed for FMT/ESA with MERVA Link.

```
MT ISNSIM    ISN Control Message for Financial Message Transfer  Page 00001
                                                                 Func EKAISNCQ

             ASP Name: ASP1____   MERVA Link Application Support Process

             ISN . . : 000999     Input Sequence Number









Command =====>
PF 1=Help      2=Retrieve  3=EOM        4=Repeat     5=Get Next  6=Requeue
PF 7=Page -1   8=Page +1   9=Hardcopy  10=Pro Line  11=Prompt   12=Escape
```

Figure 58. Display of Input Sequence Number after Data Entry

You initialized the ISN to a value of '000999'. For the next SWIFT input message that is transmitted on the ASP named 'ASP1', the FMT/ESA increments the ISN and inserts '001000' into the message.

Displaying and Modifying a Sequence Number

You can display and modify the OSN after it is created by FMT/ESA. To do this:

1. Assign the OSN Control Message function EKAOSNCQ (for FMT/ESA with MERVA Link) or DSLOSNCQ (for FMT/ESA with MQI Attachment) to a MERVA ESA user using the MERVA ESA user file maintenance. Sign on to MERVA ESA with this user identification.
2. From the MERVA ESA Function Selection menu, select the OSN Control Message function EKAOSNCQ (for FMT/ESA with MERVA Link) or DSLOSNCQ (for FMT/ESA with MQI Attachment). The Message Selection panel is displayed.
3. Press the PF key defined for the **get first** command (PF6).

   Figure 59 shows how the OSN is displayed. The last used OSN is shown and can be modified.

```
MT OSNSIM    OSN Control Message for Financial Message Transfer  Page 00001
                                                                 Func EKAOSNCQ

             OSN: 000128  Output Sequence Number









Command =====>
PF 1=Help      2=Retrieve  3=EOM        4=Repeat     5=Get Next  6=Requeue
PF 7=Page -1   8=Page +1   9=Hardcopy  10=Pro Line  11=Prompt   12=Escape
```

Figure 59. Display of Output Sequence Number

Deleting a Sequence Number

To delete a specific ISN or the OSN:

1. Display the ISN (as shown in Figure 58 on page 192) or OSN (as shown in Figure 59) to be deleted.
2. Enter:

   del
3. Press the ENTER key.

The message is deleted from its control queue.

Setting the Output Sequence Number

The OSN is intended to be initialized by the FMT/ESA only. However, you can set the OSN to an initial value.

As preparation, you must customize definitions made for the FMT/ESA in two MERVA ESA tables:

1. Change the parameter MTGEN in the message type table entry by specifying one of the following:
   - For FMT/ESA with MERVA Link: **DSLMTT MTYPE=OSNSIM,MTGEN=YES,...**
   - For FMT/ESA with MQI Attachment: **DSLMTT MTYPE=KOSN,MTGEN=YES,...**
2. Assemble and link-edit table DSLMTTT.
3. Change the parameter DE in the function table entry by specifying one of the following:
   - For FMT/ESA with MERVA Link: **DSLFNT NAME=EKAOSNCQ,DE=YES,...**
   - For FMT/ESA with MQI Attachment: **DSLFNT NAME=DSLOSNCQ,DE=YES,...**
4. Assemble and link-edit table DSLFNTT.

5. Restart MERVA ESA to activate the modified tables.

To initialize the OSN:

1. From the MERVA ESA Function Selection menu, select the OSN Control Message function EKAOSNCQ (for FMT/ESA with MERVA Link) or DSLOSNCQ (for FMT/ESA with MQI Attachment). The Message Selection panel is displayed.
2. Enter the message type of the OSN control message by specifying one of the following:
   - For FMT/ESA with MERVA Link: **mt osnsim**
   - For FMT/ESA with MQI Attachment: **mt kosn**

   The OSN Control Message panel is displayed (see Figure 59 on page 193).
3. Enter the initial value for the OSN.

Note: After changing in the Message Type Table and Function Table, it is your responsibility to ensure that you do not create a second control message in queue EKAOSNCQ (for FMT/ESA with MERVA Link) or DSLOSNCQ (for FMT/ESA with MQI Attachment). The FMT/ESA always processes the first message in the queue.

Part 2. Running Batch Programs

This part describes the batch programs provided to allow the transfer of data between MERVA ESA queues and sequential data sets.

Chapter 10. Running the MERVA Message Processing Client/Server

MERVA ESA supports MERVA Message Processing Client workstations using either APPC or TCP/IP. The MERVA Message Processing Client/Server programs are implemented as batch programs, except when using APPC under CICS, in which case the CICS APPC server is implemented as a CICS transaction. This section describes the JCL required to run the batch servers.

The JCL must include the MERVA ESA base load module library and the data sets defined in the MERVA ESA File Table. These are standardly the SWIFT and Telex correspondents files, the nickname file, and the currency file. In the figures, the lowercase parameters have the following meaning:

loadlib
The library containing the MERVA ESA base load modules

nickname
Your data set of correspondents nicknames

swift-corr
Your data set of SWIFT correspondents

telex-corr
Your data set of Telex correspondents

currency
Your data set of currency codes.

Figure 60 shows the JCL required to run the APPC/MVS workstation server.

```
//.......  JOB .........
//RUN      EXEC PGM=DSLAFA01,REGION=2M
//STEPLIB  DD DSN=loadlib,DISP=SHR
//DSLCORN  DD DSN=nickname,DISP=SHR
//DWSCOR   DD DSN=swift-corr,DISP=SHR
//ENLCOR   DD DSN=telex-corr,DISP=SHR
//DWSCUR   DD DSN=currency,DISP=SHR
//SYSPRINT DD SYSOUT=*
//DSLSNAP  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
```

Figure 60. Running the MERVA ESA APPC/MVS Workstation Server

Figure 61 on page 198 shows the JCL required to run the TCP/IP workstation server under MVS.

```
//.......  JOB .........
//RUN      EXEC PGM=DSLAFATM,REGION=2M
//STEPLIB  DD DSN=loadlib,DISP=SHR
//PROFILE  DD DSN=SYS1.TCPPARMS(PROFILE),DISP=SHR   TCP/IP
//DSLCORN  DD DSN=nickname,DISP=SHR
//DWSCOR   DD DSN=swift-corr,DISP=SHR
//ENLCOR   DD DSN=telex-corr,DISP=SHR
//DWSCUR   DD DSN=currency,DISP=SHR
//SYSPRINT DD SYSOUT=*
//DSLSNAP  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
```

Figure 61. Running the TCP/IP MERVA Message Processing Client/Server under MVS

Figure 62 shows an example of JCL to run the TCP/IP workstation server under VSE.

```
// JOB .........
// DLBL PRODL,'MERVA410.PRODLIB.BASE',99/365,SD
// EXTENT ,DOSRES
LIBDEF *,SEARCH=(PRODL.MCUST,PRODL.MBASE)
// DLBL IJSYSUC,'VSESP.USER.CATALOG',,VSAM
// DLBL DSLCORN,'MERVA410.NAMES',,VSAM,CAT=IJSYSUC
// DLBL DWSAUTD,'MERVA410.AUTHKF',,VSAM,CAT=IJSYSUC
// DLBL DWSCOR,'MERVA410.SCOR',,VSAM,CAT=IJSYSUC
// DLBL DWSCUR,'MERVA410.SCUR',,VSAM,CAT=IJSYSUC
// DLBL ENLCOR,'MERVA410.TCOR',,VSAM,CAT=IJSYSUC
// OPTION DUMP
// EXEC DSLAFA04,SIZE=600K
// EXEC LISTLOG
/&
```

Figure 62. Running the TCP/IP MERVA Message Processing Client/Server under VSE

Chapter 11. Sequential Data Set Input (DSLSDI)

DSLSDI reads a batch of messages from a sequential data set and writes them to MERVA ESA queues. In one run of DSLSDI, you can process only one sequential data set containing messages of the same format. These formats can be:

- SWIFT messages, either in SWIFT I or in SWIFT II format
- Free-format telex messages
- SWIFT messages prepared for telex transmission
- Messages in MERVA ESA queue format including all internal fields
- EDIFACT messages to be converted to SWIFT message types 105 and 106
- User-defined messages.

The record format must be VB (blocked variable-length) or VBS (spanned blocked variable-length). When VBS is used a logical record can be larger than 32KB.

Usually, one logical record of the MERVA ESA sequential data set contains one message. In addition to this standard format, MERVA ESA supports a segmentation scheme for messages. One message may span more than one logical record of the sequential data set. The first byte of each record is a segment indicator showing which logical records belong together. This byte may contain one of the following four values:

0 (X'F0') Only segment of a message

1 (X'F1') First segment of a message

2 (X'F2') Last segment of a message

3 (X'F3') Middle segment of a message.

DSLSDI assembles the logical records belonging to one message internally and stores the complete message in MERVA ESA queues. This technique allows messages larger than 32KB to be processed, using record format VB.

For MERVA ESA messages up to 2MB are supported.

VBS record format and the MERVA ESA segmentation scheme are mutually exclusive. If both are specified, MERVA ESA segmentation is assumed.

When the segmentation scheme is used it applies to all records in a sequential data set.

DSLSDI requires at least 2048KB of virtual storage for execution. When a batch of messages contains one or more messages larger than 32KB, the region size must be increased by at least twice the size of the largest message in the batch of messages.

When the parameter SDDB2 is set to YES in your MERVA ESA customization-parameter module DSLPRM, DSLSDI runs with direct queue management (DB2 MVS only).

DSLSDI - Input Program under MVS

MERVA ESA operates in one region; DSLSDI is executed in another region. The input data set DSLSDSI is assigned to the region where DSLSDI is executed.

Figure 63 shows the JCL to run the batch program DSLSDI under MVS.

```
//.......  JOB .........
//RUN      EXEC PGM=DSLSDI,PARM='parm1,parm2,parm3,parm4,parm5'
//STEPLIB  DD DSN=loadlib,DISP=SHR
//SYSUDUMP DD SYSOUT=L
//DSLSNAP  DD SYSOUT=L
//DSLSDSI  DD DISP=OLD,DSN=dsname
```

Figure 63. Writing Messages from a Data Set to MERVA ESA Queues in MVS

In the JCL, the lowercase parameters have the following meanings:

loadlib The library containing the MERVA ESA load modules

dsname The name of the cataloged data set containing the messages (input data set).

DSLSDI - Input Program under VSE

MERVA ESA operates in one partition; DSLSDI is executed in another partition. The sequential file DSLxxxx (residing on disk or tape) must be assigned to SYS025 in the partition where DSLSDI is executed. The values xxxx in the file name must have the same value as *parm0*.

Figure 64 shows the JCL to run the batch input program DSLSDI under VSE.

```
// JOB  .........
// ASSGN SYS025,DISK,VOL=volid,SHR
// DLBL DSLxxxx,infile,0,SD,BLKSIZE=blksize
// EXTENT SYS025
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION PARTDUMP
// EXEC DSLSDI,SIZE=300K,PARM='parm0,parm1,parm2,parm3,parm4,parm5'
/*
/&
```

Figure 64. Writing Messages from a Data Set to MERVA ESA Queues in VSE

In the JCL, the lowercase parameters have the following meanings:

volid The volume identification of the volume containing the input file or the program library.

infile The data set name of the input file.

blksize The block size of the input file. It must be at least the length of the longest logical record in the input file plus 8.

program library
The name of the library containing the MERVA ESA programs.

library.sublib The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibraries.

DSLSDI - Parameters

The following parameters can be specified in the PARM field of the EXEC statement:

parm0 For VSE only, the parameter is the input device, which is:

DISK For a DASD as input device, using record format VB

TAPN For input from an unlabeled tape, using record format VB

TAPL For input from a labeled tape, using record format VB

SPDI For a DASD as input device, using record format VBS

SPTN For input from an unlabeled tape, using record format VBS

SPTL For input from a labeled tape, using record format VBS.

parm1 If only 1 character is specified, it represents the format of the message. These formats can be:

W For SWIFT II messages

S For SWIFT I messages

T For free-format telex messages

Z For EDIFACT messages

Q For messages in MERVA ESA queue format

x User-defined format; it is required that for the specified character there is a DSLLDEV TYPE=NET section in all MCBs used for mapping the messages.

The message type is determined by a message type determination exit individually for each message in the sequential data set.

If 2 to 8 characters are specified, the name represents a message identification. If a message identification is specified representing an MCB, all messages are processed with this MCB. For example, telex messages can be processed with the message identification TCOV.

The parameter can also be coded as follows:

(parm11,parm12)

parm11 Specifies the message identification

parm12 Specifies the line-format identification.

If the message identification is omitted, the default value S applies, and the message type is determined automatically.

If the line-format identification is omitted, the network identification (the first character of *parm11*) is used as the default.

Note: In one batch run, DSLSDI can process only messages for the same value of *parm1* or *parm11/parm12*.

parm2 Specifies what happens when an incorrect message is encountered:

ACCEPT The incorrect message is queued if possible. When a routing table is used for the SDI function, incorrect messages can be routed to different function queues from those used for correct messages. Processing is continued with the next message.

DROP The incorrect message is dropped. Processing is continued with the next message.

CANCEL The job is terminated.

Messages in the queue format are not checked for correctness. In this case, *parm2* is ignored.

parm3 The name (up to 8 characters) of the intermediate queue that holds the messages before they are routed to the target queues. This queue must never be the target of a routing operation, and must be reserved for exclusive use by one DSLSDI job.

parm4 Reserved. Do not specify this parameter.

parm5 When specified the value must be SEGMENT or SEG. This parameter indicates that the input data set contains logical records which represent segmented messages. You can generate a data set with this format by using the DSLSDO program and specifying SEGMENT as an EXEC PARM parameter.

When the parameter is not specified no segmentation is done; each logical record of the input file represents one message.

Chapter 12. Sequential Data Set Output (DSLSDO)

DSLSDO reads a batch of messages from a MERVA ESA queue and writes them to a sequential data set.

In one run of DSLSDO, the input queue must contain only messages that can be formatted to the same line format. You can create an output data set containing:

- SWIFT messages, either in SWIFT I or in SWIFT II format.
- Free-format telex messages. You can process a mixture of SWIFT messages and free-format telex messages, provided that you have defined a common line format for both types of message. If you have not defined a common line format, the result in the sequential data set is unpredictable. This is also true if you use this data set as input for DSLSDI.
- SWIFT messages prepared for telex transmission.
- Messages in MERVA ESA queue format including all internal fields.
- EDIFACT messages converted from SWIFT message types 105 and 106.
- User-defined messages.

The record format must be VB (blocked variable-length) or VBS (spanned blocked variable-length). When VBS is used a logical record can be larger than 32KB.

Usually, one logical record of the MERVA ESA sequential data set contains one message. In addition to this standard format MERVA ESA supports a segmentation scheme for messages. One message may span more than one logical record of the sequential data set. The first byte of each record is a segment indicator showing which logical records belong together. This byte may contain one of the following four values:

0 (X'F0') Only segment of a message

1 (X'F1') First segment of a message

2 (X'F2') Last segment of a message

3 (X'F3') Middle segment of a message.

DSLSDO disassembles each message into one or more logical records and stores the records together with the segment indicator prefix in the output data set. This technique allows messages larger than 32KB to be processed, using record format VB.

For MERVA ESA messages up to 2MB are supported.

VBS record format and the MERVA ESA segmentation scheme are mutually exclusive. If both are specified, MERVA ESA segmentation is assumed.

When the segmentation scheme is used it applies to all records in a sequential data set.

DSLSDO requires at least 2048KB of virtual storage for execution. When a batch of messages contains one or more messages larger than 32KB, the region size must be increased by at least twice the size of the largest message in the batch of messages.

When the parameter SDDB2 is set to YES in your MERVA ESA customization-parameter module DSLPRM, DSLSDO runs with direct queue management (DB2 MVS only).

DSLSDO - Output Program under MVS

MERVA ESA operates in one region; DSLSDO is executed in another region. The output data set DSLSDSO is assigned to the region where DSLSDO is executed. When neither segmentation nor VBS is used, the record length must be defined so that the largest message (including a 4-byte prefix) fits in the record. Usually, LRECL=11000 is enough for SWIFT messages.

Figure 65 shows the JCL to run the batch output program DSLSDO under MVS.

```
//.......  JOB  ...........
//RUN      EXEC PGM=DSLSDO,PARM='parm1,parm2,parm3,parm4,parm5,parm6'
//STEPLIB  DD DSN=loadlib,DISP=SHR
//SYSUDUMP DD SYSOUT=L
//DSLSNAP  DD SYSOUT=L
//DSLSDSO  DD DISP=OLD,DSN=dsname
```

Figure 65. Writing Messages from a MERVA ESA Queue to a Data Set in MVS

In the JCL, the lowercase parameters have the following meanings:

loadlib The library containing the MERVA ESA load modules.

dsname The name of the data set to which the messages are written (output data set). If you create a new data set or if you use a tape, then you must complete the JCL parameters as required.

DSLSDO - Output Program under VSE

MERVA ESA operates in one partition; DSLSDO is executed in another partition. The sequential file DSLxxxx (residing on disk or tape) must be assigned to SYS025 in the partition where DSLSDO is executed. The values xxxx in the file name must have the same value as *parm0*.

Figure 66 shows the JCL to run the batch output program DSLSDO under VSE.

```
// JOB  ............
// ASSGN SYS025,DISK,VOL=volid,SHR
// DLBL DSLxxxx,outfile,0,SD,BLKSIZE=blksize
// EXTENT SYS025,volid,extent information
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION PARTDUMP
// EXEC DSLSDO,SIZE=300K,PARM='parm0,parm1,parm2,parm3,parm4,parm5,parm6'
/*
/&
```

Figure 66. Writing Messages from a MERVA ESA Queue to a Data Set in VSE

In the JCL, the lowercase parameters have the following meanings:

volid The volume identification of the volume containing the output file or the program library.

outfile The data set name of the output file.

blksize The block size of the output file. When neither segmentation nor VBS is used, the block size must be at least the length of the longest message in the output file plus 8. For SWIFT messages, a block size of 11000 is usually enough.

extent information
The extent information of the output file, when it is created with this job and is a disk file.

program library
The name of the library containing the MERVA ESA product.

library names The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibraries.

DSLSDO - Parameters

The following parameters can be specified in the PARM field of the EXEC statement:

parm0 For VSE only, this parameter is the output device, which is:

- **DISK** For a DASD as output device, using record format VB
- **TAPN** For output to an unlabeled tape, using record format VB
- **TAPL** For output to a labeled tape, using record format VB
- **SPDI** For a DASD as output device, using record format VBS
- **SPTN** For output to an unlabeled tape, using record format VBS
- **SPTL** For output to a labeled tape, using record format VBS.

parm1 Specifies a 1-character line-format identification. These formats can be:

- **W** For SWIFT messages in SWIFT II format.
- **S** For SWIFT messages in SWIFT I format.
- **T** For free-format telex messages.
- **Z** For EDIFACT messages.
- **Q** For messages in MERVA ESA queue format.
- **0** MERVA ESA base format, for example, to be used for the external line format (ELF).
- **x** User-defined format; it is required that for the specified character there is a DSLLDEV TYPE=NET section in all MCBs used for mapping the messages.

If the line-format identification is omitted, the default value of S is used.

Note: During the DSLSDO processing, the actual message identification of each message is used by MERVA ESA to format the message. Therefore, the message identifications can be different in the input queue for DSLSDO. However, the line format identification, which is used for the formatting for the sequential output file, is the same for all messages in one run of DSLSDO.

parm2 Specifies what happens when an incorrect message is encountered:

ACCEPT The incorrect message is written to the sequential data set if, despite the error, formatting was possible. If the formatting for the sequential data set is not possible, the message is routed (see ROUTE parameter). Processing is continued with the next message.

ROUTE The incorrect message is routed to the designated error queue, and processing is continued with the next message.

CANCEL A dump of the message is taken and the job is terminated.

Messages in the queue format are not checked for correctness. In this case, ROUTE is not applicable.

parm3 The name (up to 8 characters) of the queue from which messages are to be retrieved. This queue must not be used by another program. In particular, a hard-copy queue must not be specified here.

parm4 The *msgid* (up to 8 characters), which defines the formatting of the messages. If this parameter is used, all messages of the queue are formatted according to the same *msgid*. If this parameter is omitted, the messages are formatted according to the *msgid* stored in the DSLEXIT field in the TOF. For SWIFT messages prepared for telex transmission and for free-format telex messages, the *msgid* parameter TCOV is mandatory.

parm5 Reserved. Do not specify this parameter.

parm6 When specified the value must be SEGMENT or SEG. This parameter indicates that the output data set should contain logical records which represent segmented messages. Each of these logical records start with a one-byte segment indicator prefix. You can process a data set created in this format by using the DSLSDI program and specifying SEGMENT as an EXEC PARM parameter. When the parameter is not specified no segmentation is done; each logical record of the output file represents one message.

Chapter 13. System Printer Output (DSLSDY)

DSLSDY reads a batch of messages from a MERVA ESA queue and prints them on a line (SYSOUT) printer. In one run of DSLSDY, the input queue can contain any mixture of messages. The messages are printed according to the *msgid* definition of the function-table entry for the input queue. For example, the *msgid* defined for the function can cover only the telex part of a message, or only the SWIFT part, or both.

DSLSDY requires at least 2048KB of virtual storage for execution.

When the parameter SDDB2 is set to YES in your MERVA ESA customization-parameter module DSLPRM, DSLSDY runs with direct queue management (DB2 MVS only).

DSLSDY - Print Program under MVS

MERVA ESA operates in one region; DSLSDY is executed in another region. The print data set DSLSDSY is assigned to a user-defined SYSOUT class.

Figure 67 shows the JCL to run the batch print program DSLSDY under MVS.

```
//.......  JOB  .............
//RUN      EXEC PGM=DSLSDY,PARM='parm1,parm2,parm3,parm4,parm5'
//STEPLIB  DD DSN=loadlib,DISP=SHR
//SYSUDUMP DD SYSOUT=L
//DSLSNAP  DD SYSOUT=L
//DSLSDSY  DD SYSOUT=L,DCB=(LRECL=125,BLKSIZE=1254)    LINE PRINTER
```

Figure 67. Printing Messages on a Line Printer in MVS

In the JCL, the lowercase parameter has the following meaning:

loadlib The library containing the MERVA ESA load modules.

DSLSDY - Print Program under VSE

MERVA ESA operates in one partition; DSLSDY is executed in another partition. The print file DSLSDSY is assigned to SYSLST.

Figure 68 shows the JCL to run the batch print program DSLSDY under VSE.

```
// JOB  .........
// ASSGN SYSLST,00E
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION PARTDUMP
// EXEC DSLSDY,SIZE=300K,PARM='parm1,parm2,parm3,parm4,parm5'
/*
/&
```

Figure 68. Printing Messages on a Line Printer in VSE

In the JCL, the lowercase parameters have the following meanings:

volid The volume identification of the volume containing the program library.

program library
The name of the library containing the MERVA ESA product.

library.sublib The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibraries.

DSLSDY - Parameters

Five parameters must be specified in the PARM field of the EXEC statement:

parm1 The name (up to 8 characters long) of the queue from which the messages are to be printed.

parm2 If *parm3* is 0 to 3, this parameter is used as a 1-character language identification for the printer MCB. If *parm3* is 4, this parameter is used as a 1-character line-format identifier for the printer MCB.

This parameter can also be coded as *x*U, where *x* specifies the language or line format, and U specifies that lowercase characters in the printout are converted to uppercase characters.

Note: If this parameter is omitted, the value specified in the function-table entry for the specified input queue is taken.

parm3 The 1-character compression format. It must be a value from 0 to 4. See the PRFORM parameter of the DSLFNT macro described in *MERVA for ESA Macro Reference*.

Messages in external line format should be printed with compression format 4.

Note: If this parameter is omitted, the value specified in the function-table entry for the specified input queue is taken.

parm4 Specifies whether the messages in the input queue are to be deleted or kept after printing. The permitted values are:

KEEP Keeps the message

DELETE Deletes the message.

If this parameter is not specified, the value of the KEEPMSG parameter of the DSLFNT macro is taken. See *MERVA for ESA Macro Reference*.

parm5 The name (up to 8 characters long) of a message identification or a Message Control Block (MCB) name to be printed off-line. The message identification must be contained in the MERVA ESA Message Type table. This facility can be used to print MCBs without data, for example, for testing purposes. Only one message identification or MCB can be printed in one invocation of DSLSDY. MERVA ESA need not be active at that time, and the queue name in *parm1* is ignored.

Chapter 14. Sequential Data Set Input (DSLSDIR)

DSLSDIR reads as DSLSDI a batch of messages from a sequential data set and writes them to MERVA ESA queues. For restart and recovery reasons the messages are first written to an intermediate queue, before they are put or routed to the target queue(s).

You can invoke DSLSDIR either by:

- The front-end exec DSLSDIRF, which displays an ISPF panel where you can enter the runtime parameters and data set names (MVS only)
- Creating and submitting a batch job yourself.

Invoke DSLSDIR via ISPF Panel (MVS only)

Menu

To invoke DSLSDIR via an ISPF panel, enter on an ISPF command line 'TSO %DSLSDIRF', or on ISPF option 6 '%DSLSDIRF'. You will see the following panel.

```
---------------------------- MERVA ESA V4.1 DSLSDIR ----------------------------
COMMAND   ===>

Action    ===> _         ( C = create JCL, D = DD statements, H = print Help )
                         ( J = Jobcard, L = edit List, S = create and Submit )
Function  ===> _         ( B = ROUTB, C = CHECK, D = ROUTD, P = PUT )
Msg Form  ===> _         ( Q = MERVA queue format, S = SWIFT I, W = SWIFT II )
Queue 1   ===> ________  ( Intermediate queue name, may be blank with CHECK )
Queue 2   ===> ________  ( Target queue name for PUT )
Fld Name  ===> _         ( 0 = none, 1 = MSGDST, 2 = MSGNET, 3 = SWBHLT )
Fld Value ===> __________ ( Field value that must be matched for Fld Name )
Inco. Msg ===> ___       ( ACC = accept incorrect msg's, CAN = cancel DSLSDIR )
                         ( DRO = drop incorr. msg's, INC = incorr. msg's only )
Log Level ===> _         ( Log level 1 .. 4 , use 4 only in case of problems )
Perf Info ===> _         ( N = do not measure performance, Y = do measure )
Segmented ===> _         ( A = auto, N = input not segmented, Y = segmented )

Input DS  ===> ____________________________________________ ( Input data set )
List DS   ===> ____________________________________________ ( Listing data set)

JCL DS    ===> ____________________________________________ ( JCL data set )

PF1 = HELP                                                 (C) IBM Corp. 1999
```

Figure 69. DSLSDIR (Sequential Data Set Input) Menu

To exit press PF3.

Note: The installation of the ISPF panel is described in the *MERVA for ESA Installation Guide*.

Runtime Parameters

The following parameters can be specified:

No.	Field	Description
1	Action	**C** Create JCL in data set *JCL DS*, member DSLSDIR **D** Open pop-up window to enter DD statements **H** Print help information to data set *List DS* **J** Open pop-up window to enter JCL cards **L** Edit listing data set *List DS* **S** Create JCL in data set *JCL DS*, member DSLSDIR, and submit it. This parameter is required.
2	Function	**B** Route the messages in the input data set *Input DS* via the routing (table) of the queue *Queue 1*. The used API function is ROUB - route with back reference. **C** Check the messages in the input data set *Input DS*. **D** Route the messages in the input data set *Input DS* via the routing (table) of the queue *Queue 1*. The used API function is ROUN - route directly. ROUN is faster than ROUB. It requires MERVA ESA V4 (or higher). **P** Put the messages in the input data set *Input DS* to the queue *Queue 2*. This parameter is required.
3	Msg Form	Message format: **Q** MERVA ESA queue format **S** SWIFT I **W** SWIFT II. This parameter is required.
4	Queue 1	Name of the intermediate queue. This parameter is required with *Function* PUT, ROUTB, and ROUTD. It is ignored with *Function* CHECK.
5	Queue 2	Name of the target queue. This parameter is required with *Function* PUT. It is ignored with *Function* CHECK, ROUTB, and ROUTD.
6	Fld Name	The identifier for a MERVA field: **0** All messages should be processed **1** Only messages with the SWIFT master destination (MSGDST) specified in *Fld Value* will be processed **2** Only messages with the message type (MSGNET) specified in *Fld Value* will be processed **3** Only messages with the SWIFT basic header logical terminal address (SWBHLT) specified in *Fld Value* will be processed. This parameter is optional, the default value used is 0.

No.	Field	Description
7	Fld Value	Value that must be matched for the field specified in *Fld Name*: • If *Fld Name* is 1, an up to 9 character SWIFT master destination, for example, VNDEBET2A. • If *Fld Name* is 2, an up to 8 character MERVA message type. Should start with the letter 'S', for example, S100. • If *Fld Name* is 3, an up to 12 character S.W.I.F.T. basic header logical terminal address, for example, VNDEBET2AXXX. The wildcard '*' can be specified as last (or only) character. This parameter is required when *Fld Name* is 1, 2, or 3.
8	Inco. Msg	Specifies what happens when an incorrect message is encountered: **ACC** Incorrect messages are accepted **CAN** DSLSDIR is terminated **DRO** Incorrect messages are dropped **INC** Only incorrect messages are processed, the correct messages are dropped. This parameter is optional, the default value used is ACC.
9	Log Level	**1** Only overview data is shown in the listing **2** Detailed data for each message is shown **3** Checking errors are more detailed. Fields SW20 (TRN) and SW108 (Msg. user ref.) are shown. **4** Should be used in case of problems only. This parameter is optional, the default value used is 2.
10	Performance Information	**N** No MERVA ESA API performance information is gathered **Y** The time spent in the MERVA ESA API functions GETU, MSGP, PUT, PUTB, ROUB, and ROUN is gathered. This parameter is optional, the default value used is N.
11	Segmented	Indicates whether the input data set *Input DS* contains segmented messages: **A** DSLSDIR determines whether the input messages are segmented or not segmented (auto) **N** The input messages are not segmented **Y** The input messages are segmented. This parameter is optional, the default value used is A.
12	Input DS	Input data set This parameter is required with *Action* C and S.
13	List DS	Listing data set This parameter is required with *Action* C, H, L, and S.
14	JCL DS	JCL data set. The created (and submitted) JCL is written as member DSLSDIR to this data set. This parameter is required with *Action* C, H, and S.

MVS DD Statements

DSLSDIR requires the following DD statements to be defined:

STEPLIB The load library containing the MERVA ESA programs

DWSCUR The currency code file. DWSCUR is required only when you specify runtime parameter *Msg Form* S or W and CURCODE=FILE is specified in your DSLPRM.

SYSEXEC The library containing the program DSLSDIR

DSLSDSI The input data set

SYSTSPRT The listing data set. Must be preallocated, record format VB, logical record length 136 is recommended, or 'SYSOUT=*'.

SYSTSIN The specified DSLSDIR runtime parameters.

- Enter STEPLIB, DWSCUR, and SYSEXEC with *Action* D.
- DSLSDSI, the input data set, and SYSTSPRT, the listing data set, can be specified on the panel.
- SYSTSIN, the DSLSDIR runtime parameters, is generated by DSLSDIRF.

First-Time Users

If you are a first-time user, you must allocate two data sets in TSO, provide with *Action* D the required DD statements for DWSCUR, STEPLIB, and SYSEXEC, and with *Action* J the required job statement information.

Allocate two data sets in TSO:

- The data set to which DSLSDIRF writes the generated JCL: Should be PO, FB80. Name, for example, uid.MERVA.JCL
- The data set to which DSLSDIR writes the listing: Should be PS, VB136. Name, for example, uid.MERVA.LIST or use 'SYSOUT=*'.

Note: You can also use existing data sets.

The job statement information could look like this:

Edit Jobcards in the window below, PF3 to end

```
//HEGSDIR JOB (DE05020),'S. HEGENAUER',MSGLEVEL=(1,1),MSGCLASS=X,
//            CLASS=A,NOTIFY=HEG,USER=HEG
//*
//*
```

Figure 70. DSLSDIR (Sequential Data Set Input) Job Statement Window

The DD statements could look like this:

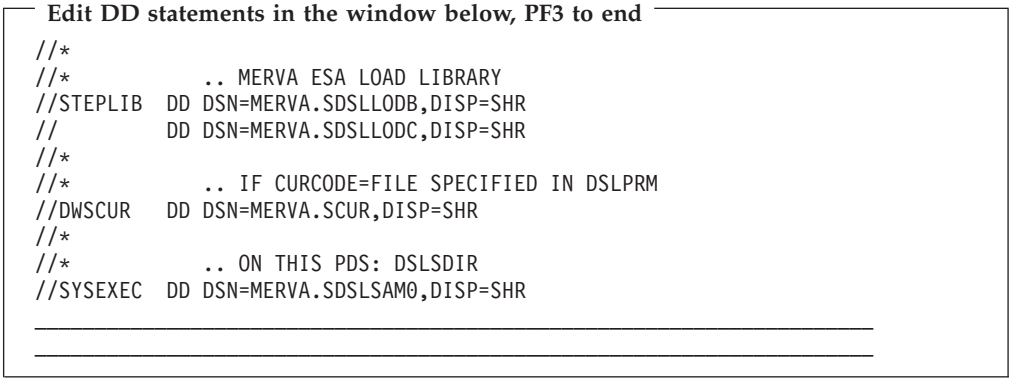

Figure 71. DSLSDIR (Sequential Data Set Input) DD Statements Window

All entered values will be remembered in ISPF profile variables.

Invoke DSLSDIR via JCL

You can invoke DSLSDIR also via JCL. If you are running MERVA ESA under VSE you have to do so.

Job Control Statements for MVS

The following figure shows the MVS JCL to run DSLSDIR.

```
//.......  JOB  ............
//REXXB    EXEC PGM=DSLAREXX,REGION=0K,PARM=DSLSDIR
//*
//*           .. RUNTIME PARAMETERS
//SYSTSIN  DD  *
  * Comments start with '*' and ';'
  * HELP
  * -- Required parameters --
  FUNCTION  = ccccc        ; (Queue management) Function
  MSGFORMAT = c            ; Message format, Q, S, or W
  QUEUE1    = cccccccc     ; Intermediate queue name
  QUEUE2    = cccccccc     ; Target queue name for PUT
  * -- Optional parameters --
  FLDNAME1  = cccccc       ; MERVA field name
  FLDVALUE1 = ccccccccc    ; MERVA field value
  INCORRECT = ccccccccccc  ; Incorrect message disposition
  LOGLEVEL  = n            ; Log level 1 .. 4
  PERFORM   = ccc          ; Performance info, YES or No
  SEGMENT   = cccc         ; Are input messages segmented?
  USERPARM  = ccccccccc..  ; User specific parameter
/*
//*
//*           .. MERVA ESA LOAD LIBRARY
//STEPLIB  DD DSN=loadlib,DISP=SHR
//*
//*           .. IF CURCODE=FILE SPECIFIED IN DSLPRM
//DWSCUR   DD DSN=curds,DISP=SHR
//*
//*           .. INPUT DATASET
//DSLSDSI  DD DSN=inputds,DISP=SHR
//*
//*           .. ON THIS PDS: DSLSDIR
//SYSEXEC  DD DSN=samplib,DISP=SHR
//*
//*           .. LISTING DATASET
//SYSTSPRT DD DSN=listds,DISP=OLD
//
//
```

Figure 72. DSLSDIR (Sequential Data Set Input) Sample JCL (MVS)

Data Set Names

In the JCL, the lowercase data set names have the following meanings:

loadlib The name of the load library containing the MERVA ESA programs.

curds The name of the currency code file. curds is required only when you specify runtime parameter MSGFORMAT = S or W and CURCODE=FILE is specified in your DSLPRM.

inputds The name of the input data set.

samplib The name of the library containing the program DSLSDIR.

listds The name of the listing data set. Must be preallocated, record format VB, logical record length 136 recommended.

Job Control Statements for VSE

The following figure shows the VSE JCL to run DSLSDIR.

```
// JOB DSLSDIR ...
// ASSGN SYS025,DISK,VOL=volid,SHR
// DLBL DSLSDSI,infile,0,SD
// EXTENT SYS025
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION NODUMP
// EXEC DSLAREXX,SIZE=300K,PARM='DSLSDIR'
   * Comments start with '*' and ';'
   * HELP
   * -- Required parameters --
   FUNCTION  = ccccc        ; Queue management function
   MSGFORMAT = c            ; Message format, Q, S, or W
   QUEUE1    = cccccccc     ; Intermediate queue name
   QUEUE2    = cccccccc     ; Target queue name for PUT
   * -- Optional parameters --
   FLDNAME1  = cccccc       ; MERVA field name
   FLDVALUE1 = ccccccccc    ; MERVA field value
   INCORRECT = ccccccccccc  ; Incorrect message disposition
   LOGLEVEL  = n            ; Log level 1 .. 4
   PERFORM   = ccc          ; Performance info, YES or No
   SEGMENT   = cccc         ; Are input messages segmented?
   USERPARM  = ccccccccc..  ; User specific parameter
/*
/&
```

Figure 73. DSLSDIR (Sequential Data Set Input) Sample JCL (VSE)

Data Set Names

In the JCL, the lowercase parameters have the following meanings:

volid The volume identification of the volume containing the input file or the program library

infile The data set name of the input file

program library
The name of the library containing the MERVA ESA programs

library.sublib The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibraries.

Runtime Parameters

The runtime parameters are passed to DSLSDIR via SYSTSIN under MVS and via SYSIPT under VSE. They have the form KEYWORD = VALUE. Each pair must be coded on a separate line. The input is folded to uppercase and leading and trailing blanks are stripped off from the specified keyword value. Lines starting with an asterisk '*' are treated as comments, a semicolon ';' starts a line comment.

HELP as the only parameter prints a description of the runtime parameters.

Required Parameters

FUNCTION
Specifies the (queue management) function performed:

CHECK Check the messages in the input data set. Do not write any message to a MERVA ESA queue.

PUT Put the messages in the input data set to the target queue QUEUE2.

ROUTB Route the messages in the input data set via the routing (table) of the queue QUEUE1. The used API function is ROUB - route with automatic delete.

ROUTD Route the messages in the input data set via the routing (table) of the queue QUEUE1. The used API function is ROUN - route directly. ROUN is faster than ROUB. It requires MERVA ESA V4 (or higher).

This parameter is required.

MSGFORMAT
Specifies the format code used to format the messages:

Q MERVA ESA queue format

S SWIFT I

W SWIFT II.

This parameter is required.

QUEUE1
Intermediate queue name. The name of the intermediate queue that holds the messages before they are put or routed. The routing (table) of this queue also determines the routing for FUNCTION = ROUTB and ROUTD.

Can be checked in a user exit of DSLSDIR. This parameter is required with FUNCTION = PUT, ROUTB, and ROUTD. It is ignored with FUNCTION = CHECK.

QUEUE2
Target queue name. The name of the target queue for PUT. Can be checked in a user exit of DSLSDIR.

This parameter is required with FUNCTION = PUT. It is ignored with FUNCTION = CHECK, ROUTB, and ROUTD.

Optional Parameters

FLDNAME1
MERVA field name:

NONE Specifies that no field should be used to select messages, all messages are processed.

MSGDST Specifies that only messages with the SWIFT master destination specified in parameter FLDVALUE1 should be processed.

MSGNET Specifies that only messages with the message type specified in parameter FLDVALUE1 should be processed.

SWBHLT Specifies that only messages with the SWIFT basic header logical terminal address specified in parameter FLDVALUE1 should be processed.

This parameter is optional, the default value used is FLDNAME1 = NONE.

FLDVALUE1
MERVA field value:

- If parameter FLDNAME1 = MSGDST, an up to 9 character SWIFT master destination
- If parameter FLDNAME1 = MSGNET, an up to 8 character MERVA message type. Should start with 'S', for example, S100 or S1*.
- If parameter FLDNAME1 = SWBHLT, an up to 12 character SWIFT basic header logical terminal address.

The wildcard '*' is allowed as last (or only) character.

This parameter is required when FLDNAME1 = MSGDST, MSGNET, or SWBHLT.

INCORRECT
Incorrect message disposition. Specifies what happens when an incorrect message is encountered. If specified, the keyword value must be one of:

ACCept	Incorrect messages are accepted
CANcel	DSLSDIR is terminated
DROp	Incorrect messages are dropped
INCorronly	Only incorrect messages are processed, correct messages are dropped.

This parameter is optional, the default value used is INCORRECT = ACCEPT.

Note: If the parameters FLDNAME1 and FLDVALUE1 are specified so that only selected messages should be processed, the parameter INCORRECT applies only to messages that passed this filter. If, for example, the parameter FLDNAME1 = MSGNET, the parameter FLDVALUE = S1*, and parameter INCORRECT = CANCEL, an erroneous S200 message will **not** cancel DSLSDIR, as only S1* messages are processed.

LOGLEVEL
Log level:

1	Only overview data is shown in the listing.
2	Detailed data for each message is shown.
3	Checking errors are more detailed. Fields SW20 (TRN) and SW108 (Msg. user ref.) are shown.
4	Should be used in case of problems only.

This parameter is optional, the default value used is LOGLEVEL = 2.

PERFORM
Performance information. Indicates whether performance information about the API commands GETU, MSGP, PUT, PUTB, and ROUB should be gathered. If specified, the keyword value must be Yes or No.

This parameter is optional, the default value used is PERFORM = NO.

SEGMENT
Segmented input messages. Indicates whether the input data set contains segmented messages. If specified, the keyword value must be one of:

Auto DSLSDIR determines itself whether the messages in the input data set are segmented or not segmented

No The messages in the input data set are not segmented

Yes The messages in the input data set are segmented.

This parameter is optional, the default value used is SEGMENT = AUTO.

With the following parameter you can pass a user-specific parameter to DSLSDIR:

USERPARM
User-specific parameter. The specified data is accepted by DSLSDIR and available in the variable *parm_userparm*.

This parameter is optional.

Required Parameters for VSE

VSEBLKSIZE
DSLSDSI BLKSIZE. BLKSIZE of input SAM file DSLSDSI, maximum 32761.

This parameter is required under VSE.

VSERECFORM
DSLSDSI RECFORM. RECFORM of input SAM file DSLSDSI: FIXUNB, FIXBLK, VARUNB, or VARBLK.

This parameter is required under VSE.

VSERECSIZE
DSLSDSI RECSIZE. RECSIZE of input SAM file DSLSDSI, maximum. 32761.

This parameter is required under VSE when VSERECFORM = FIXUNB or FIXBLK.

EDIFACT FINPAY Conversion

DSLSDIR supports basic conversion of EDIFACT FINPAY messages to SWIFT MT121 messages. It can process input FINPAY messages, created, for example, by a bank application, and append the necessary SWIFT headers and tags to make them SWIFT MT121 input messages. DSLSDIR FINPAY conversion assumes that the input FINPAY message is for one single recipient.

Runtime Parameters

Specify the runtime parameter USERPARM = XFINPAY. The parameter MSGFORMAT must be W, and FLDNAME1 and FLDVALUE1 must not be specified.

Customization

For the DSLSDIR FINPAY conversion you must customize some settings in the DSLSDIR `FINPAY CUSTOMIZATION SECTION`:

1. Variable *yourlt*

   Your 12-character SWIFT LT address, for example, VNDEBET2AXXX. If set, the value is used as Logical Terminal address in the SWIFT Basic Header (swbhlt) and overwrites the sender's address found in the FINPAY message. If blank, the sender's address found in the FINPAY message is used to look up the Logical Terminal address in list *lta.*

2. Stem variable (array) *lta*

   The bank address in FINPAY can be different from the SWIFT LT address. Therefore it is necessary to specify pairs of:

name The bank's address as used in the FINPAY message.

lta.name The bank's SWIFT LT address.

The number of pairs that you can specify is unlimited.

Example: The FINPAY address used for the Bluebank is BLUEBANK-ED. To assign the LT address BLUEDEFFAXXX, you would specify the following pair:

```
name = 'BLUEBANK-ED' ; lta.name = 'BLUEDEFFAXXX'
```

When DSLSDIR, then it finds the bank address BLUEBANK-ED in the FINPAY message, it uses BLUEDEFFAXXX in the generated MT121 message. When DSLSDIR finds an undefined bank address, it issues a message and sets the LT field (SWBHLT or SWAHILT) to FPUNKNOWNADR. This can be used for routing purposes.

MT121 Fields

- The field **SWBHLT** (sender's LT address) is set in DSLSDIR routine `XFinpay_set_SWBHLT`.
- The field **SWAHILT** (recipient's LT address) is set in DSLSDIR routine `XFinpay_set_SWAHILT`.
- The field **SW108** (message user reference) is set in DSLSDIR routine `XFinpay_set_SW108`. To omit the User Header, specify sw108 = ' '.

Customization

The following DSLPRM parameters affect DSLSDIR:

PRTNAME Your institution name as it is to appear in the printout of (most) REXX batch utilities.

SDDB2 When this parameter is set to YES, direct queue management is enabled (DB2 MVS only).

You can use the following routines in DSLSDIR to reject entered runtime parameters:

1. USEREXIT_Q1 can be used to reject the entered value for runtime parameter QUEUE1, intermediate queue.
2. USEREXIT_Q2 can be used to reject the entered value for runtime parameter QUEUE2, target queue.

Sample Printout

The following figure shows the information printed after the execution of the DSLSDIR program.

```
MERVA ESA V4.1 DSLSDIR                          21. Apr. 1999  18:00:15
(C) Copyright IBM Corp. 1999

+ ------------------------------------------------------------ +
|         S A M P L E   B A N K   B o e b l i n g e n          |
+ ------------------------------------------------------------ +


DDDDD    SSSSSS   LLL      SSSSSS   DDDDD    III   RRRRR
DDDDDD   SSSSSS   LLL      SSSSSS   DDDDDD   III   RRRRRR
DD  DD   SS       LLL      SS       DD  DD   III   RR  RR
DD  DD   SS       LLL      SS       DD  DD   III   RR  RR
DD  DD   SSSSSS   LLL      SSSSSS   DD  DD   III   RRRRR
DD  DD   SSSSSS   LLL      SSSSSS   DD  DD   III   RRRRRR
DD  DD       SS   LLL          SS   DD  DD   III   RR  RR
DD  DD       SS   LLL          SS   DD  DD   III   RR  RR
DDDDDD   SSSSSS   LLLLLL   SSSSSS   DDDDDD   III   RR  RR
DDDDD    SSSSSS   LLLLLL   SSSSSS   DDDDD    III   RR  RR


REXX version of DSLSDI - Sequential Data Set Input.
HELP as the only parameter prints a description.


DSLSDIR_001I : DSLSDIR started by user HEG at 21. Apr. 1999 18:00:15

DSLSDIR_003I : Runtime parameters:
                1. FUNCTION   - Queue mgmt function  : PUT
                2. MSGFORMAT  - Message format ..... : W
                3. QUEUE1     - Intermed. queue name : DMSDI
                4. QUEUE2     - Target queue name .. : L1DE0
                5. FLDNAME1   - MERVA field name ... : NONE
                6. FLDVALUE1  - MERVA field value .. :
                7. INCORRECT  - Incorrect msg disp.  : ACCEPT
                8. LOGLEVEL   - Log level .......... : 3
                9. PERFORM    - Performance info ... : YES
               10. SEGMENT    - Segmentation ind.    : AUTO
               11. USERPARM   - User specific data   : MYPARM

DSLSDIR_005I : MERVA ID is MHEG, MERVA name is MERVAESA.

DSLSDIR_006I : Queue management module ..... : DSLQMCNV
               Access method / features used : VXBD

DSLSDIR_009I : Check for DSLSDIR restart situation.

DSLSDIR_053I : MERVA API command QLL ended with INTRC 09.
               INTQUEUE: DMSDI
               The intermediate queue is empty (OK).

DSLSDIR_013I : Now at start of STAGE 1 processing.
```

Figure 74. DSLSDIR (Sequential Data Set Input) Sample Printout (Part 1 of 3)

```
DSLSDIR_014I : The input dataset used is HEG.DSLSDI.FILE4.
               Note: This info was obtained by using internal control
                     blocks.

DSLSDIR_015I : Read the input dataset.

DSLSDIR_016I : Read of input dataset DSLSDSI with EXECIO was ok.
               Number of read input lines: 5

DSLSDIR_018I : AUTO determined segmentation indicator: NOSEGMENT

DSLSDIR_112W : MERVA API command MSGP for input message 4
               ended with INTRC 00.
               MFS has detected checking errors.
               INTERMF1: DWS3529 Field SW32A DATE must have format YYMMDD
               INTERMF2: DWS3513 Field SW57 option must be A B D

DSLSDIR_027I : Now at start of STAGE 2 processing.

DSLSDIR_028I : The DSLSDIR restart message has been deleted successfully.
               Queue DMSDI, QSN 1026.

DSLSDIR_032I : Detailed statistical data for stage 1
               (Input dataset to intermediate queue DMSDI):

               E  M
               R  A
               R  T
               O  C  Record  MSGP  PUT
               R  H  number   rc   rc   MT     SW20              SW108
               -  -  ------   --   --   -----  ----------------  ----------------
                          1   ok   ok   S100   100
                          2   ok   ok   S100   100
                          3   ok   ok   S100   100
               >          4   00   ok   S100   100
                          5   ok   ok   S100   100

DSLSDIR_033I : Performance info for stage 1
               (Input dataset to intermediate queue DMSDI):

               E
               R
               R
               O   Record   +- MSGP -+   +- PUT --+
               R   number   rc   time    rc   time    MT
               -   ------   --  ------   --  ------   -----
                        1   ok  0.8141   ok  0.0041   S100
                        2   ok  0.0363   ok  0.0039   S100
                        3   ok  0.0304   ok  0.0033   S100
               >        4   00  0.0344   ok  0.0037   S100
                        5   ok  0.0298   ok  0.0192   S100

              Sum:              0.9450       0.0342
```

Figure 74. DSLSDIR (Sequential Data Set Input) Sample Printout (Part 2 of 3)

```
DSLSDIR_034I : Detailed statistical data for stage 2
               (Intermediate queue DMSDI to final queue L1DE0):

                     Queue-  Record   +- GETU -+   +- PUTB -+
               Err   element number   rc   time    rc   time    QSN
               ---   ------  ------   --  ------   --  ------   ----------
                          1       1   ok  0.0194   ok  0.0053   0000002078
                          2       2   ok  0.0043   ok  0.0044   0000002079
                          3       3   ok  0.0044   ok  0.0044   0000002080
                          4       4   ok  0.0039   ok  0.0053   0000002081
                          5       5   ok  0.0198   ok  0.0059   0000002082

               Sum:                       0.0518       0.0253

               Note: The restart message is not contained in this list.
DSLSDIR_035I : Overview statistical data

               GETU = Read next message from a MERVA queue
               MSGP = Map a message from external format to internal buffer
               PUT  = Copy a message to another MERVA queue
               PUTB = Move a message to another MERVA queue
               ROUB = Route a message to a MERVA queue with automatic delete
               ROUN = Route next message directly to a MERVA queue

               ---  Stage 1  ---

               No. of input records ...... : 5
               No. of input messages ..... : 5

               No. of messages MSGP ok ... : 4
               No. of MSGPs with intrc 00  : 1
               No. of messages MSGP failed : 0

               No. of messages PUT ok .... : 5
               No. of messages PUT failed  : 0

               ---  Stage 2  ---

               No. of messages GETU ok ... : 5
               No. of messages GETU failed : 0

               No. of messages PUTB ok ... : 5
               No. of messages PUTB failed : 0

DSLSDIR_036I : Number of input messages ................... : 5
               Number of input messages in error .......... : 1
                 The messages have been ACCEPTed.
               Number of messages PUT to queue L1DE0 ...... : 5

DSLSDIR_030I : DSLSDIR ended with return code 4 - Warning.

DSLSDIR_037I : DSLSDIR ended at 21. Apr. 1999 18:00:17
```

Figure 74. DSLSDIR (Sequential Data Set Input) Sample Printout (Part 3 of 3)

Listing Fields

The 'Detailed statistical data' of the listing contains the following information:

For stage 1:

ERROR A '>' indicates an error with the message, for example, a checking error, or the PUT to the intermediate queue failed

MATCH 'N' indicates that this message does not match a specified MSGDST, MSGNET, or SWBHLT value

Record number
Running record number

MSGP rc Return code of API function MSGP - 'ok' indicates that the message was successfully translated from the external format to the internal MERVA ESA format

PUT rc Return code of API function PUT - 'ok' indicates that the message was successfully put to the intermediate queue

MT Message type (MSGNET)

MSGDST SWIFT master destination. Is printed only when specified as selection criterion

SWBHLT SWIFT basic header logical terminal address. Is printed only when specified as selection criterion

SW20 Transaction reference number (TRN). Is printed only with log level ≥ 3

SW108 Message user reference. Is printed only with log level ≥ 3

QSN QSN of the intermediate queue. Is printed only with function PUT, ROUTB, and ROUTD, and log level ≥ 4

For performance information of stage 1:

ERROR As above

Record Number
As above

MSGP rc As above

MSGP time Time in seconds spent in API function MSGP

PUT rc As above

PUT time Time in seconds spent in API function PUT.

For stage 2:

Err A '>' indicates an error with the message, for example, the PUT to the MERVA target queue failed.

Queue Element
Running queue element number

Record Number
Running record number

GETU rc Return code of API function GETU, read a message from the intermediate queue

GETU time Time in seconds spent in API function GETU

PUTB rc Return code of API function PUTB - move a message from the intermediate queue to the target queue

PUTB time Time in seconds spent in API function PUTB

ROUB rc Return code of API function ROUB, route a message via the routing (table) of the intermediate queue

ROUB time Time in seconds spent in API function ROUB

ROUN rc	Return code of API function ROUN, route a message via the routing (table) of the intermediate queue
ROUN time	Time in seconds spent in API function ROUB
QSN	PUT and ROUTB: QSN of the target queue. ROUTD: QSN of the intermediate queue

Notes:

1. With function CHECK there is no stage-2 processing.
2. `rc <= -2` indicates an error in DSLSDIR or the REXX host command environment. Refer to the *MERVA for ESA Application Programming Interface Guide* for a description of the return code.

Messages and Codes

DSLSDIR ends with the return codes:

0	Successful
4	Warning
8	Error
12	Severe error.

The messages have the following structure:

DSLSDIR_nnnl Message text

Where:

DSLSDIR	Identifies the message as a message of the DSLSDIR utility.
_	Underscore
nnn	Is a 3-digit number used to identify the message.
l	Is a single character used to show the type of message. The following characters are used:
	I An information message
	W A warning message
	E An error message
	S A severe error message.

The messages are not further documented. Note that the message numbers are subject of change at any time.

Comparison of DSLSDIR with DSLSDI

- **Capabilities provided by both, DSLSDI and DSLSDIR**

  DSLSDI and DSLSDIR both provide:

  1. Capable of restart (same mechanism as DSLSDI).
  2. Input data set can be segmented or unsegmented.
  3. ENQ and DEQ have the same resource name as DSLSDI.
  4. Add own MSGTRACE entry to the message.
  5. Write WTO messages about processing.

Note: WTO messages are also written to the journal.

- **Advantages of DSLSDIR**

  DSLSDIR has the following advantages over DSLSDI:

  1. The input messages can either be:
     - Routed via the routing (table) of the intermediate queue (this does DSLSDI)
     - Put to the target queue
     - Checked only.
  2. You can specify that only input messages are to be processed that match a specified:
     - MSGDST (SWIFT master destination)
     - MSGNET (message type)
     - SWBHLT (SWIFT basic header logical terminal address).
  3. Additional Incorrect message disposition INCORRONLY - process incorrect messages only.
  4. Segmentation indicator can be auto determined.
  5. Detailed error message for each input message.
  6. Overview and detailed statistical data.
  7. The message type (MT), SW20 (TRN), and SW108 (Msg. user ref.) of each input message can be shown.
  8. Performance information can be gathered.
  9. User exits to check intermediate and target queue name.
  10. Tells the name of input data set (by using internal control blocks, MVS only).
  11. ISPF front-end panel DSLSDIRP (MVS only).

- **Restrictions of DSLSDIR**

  DSLSDIR has the following restrictions compared with DSLSDI:

  1. The input data set must be on disk (not on tape).
  2. Does not support VBS records.
  3. Supports only message format Q, S, and W.
  4. UMR is also written for the restart message.
  5. User exit DSLMU020 is not supported.

Chapter 15. Sequential Data Set Load in REXX (DSLSDLR)

DSLSDLR loads the messages unloaded with the DSLSDUR utilities back to their MERVA queues preserving the queue name, QSN, key values, and the *write back* indicator.

It is checked that on the MERVA ESA queue no queue elements with the same or a higher QSN exist.

Dependencies: MERVA ESA must be active.

Input Data Set Layout

See "Unload/Reload Data Set Layout" on page 251 for the expected layout of the input data set. The input data set must have record format FB and logical record length 1024.

Job Control Statements for MVS

The following figure shows the MVS JCL to load MERVA queues in batch.

```
//.......  JOB  ............
//REXXB    EXEC PGM=DSLAREXX,REGION=8M,PARM=DSLSDLR
//*
//SYSTSIN  DD *
  * Comments start with '*' and ';'
  * HELP
  * -- Required parameters --
  MSGFORMAT   = c           ; Message format, Q, S, or W
  QUEUE       = cccccccc    ; Queue pattern (mult.)
  QUEUE       = cccccccc
  QUEUE       = cccccccc
  * -- Optional parameters --
  ACTIVEUSERS = ccccccc     ; Active users allowed?
  LOGLEVEL    = n           ; Log level 1 .. 4
  QSNERROR    = cccccccc    ; QSN errors, CONTINUE or STOP
/*
//*
//*           .. MERVA ESA LOAD LIBRARY
//STEPLIB  DD DSN=loadlib,DISP=SHR
//*
//*           .. IF MSGFORMAT=S/W AND CURCODE=FILE SPEC. IN DSLPRM
//DWSCUR   DD DSN=curds,DISP=SHR
//*
//*           .. INPUT DATA SET (FB1024)
//DSLSDSU  DD DSN=inputds,DISP=OLD
//*
//*           .. ON THIS PDS: DSLSDLR
//SYSEXEC  DD DSN=samplib,DISP=SHR
//*
//*           .. LISTING DATASET (VB136)
//SYSTSPRT DD DSN=listds,DISP=OLD
//
```

Figure 75. DSLSDLR (Sequential Data Set Load) Sample JCL (MVS)

Data Set Names

In the JCL, the lowercase data set names have the following meanings:

loadlib The name of the load library containing the MERVA ESA programs.

curds The name of the currency code file. curds is required only when you specify runtime parameter MSGFORMAT = S or W and CURCODE=FILE is specified in your DSLPRM.

inputds The name of the input data set. Must be record format FB, logical record length 1024, and must have **DISP=OLD**.

samplib The name of the library containing the program DSLSDLR.

listds The name of the listing data set. Must be preallocated, record format VB, logical record length 136 recommended.

Job Control Statements for VSE

The following figure shows the VSE JCL to run DSLSDLR.

```
// JOB DSLSDLR ...
// ASSGN SYS025,DISK,VOL=volid,SHR
// DLBL DSLSDSL,infile,0,SD
// EXTENT SYS025
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION NODUMP
// EXEC DSLAREXX,SIZE=300K,PARM='DSLSDLR'
   * Comments start with '*' and ';'
   * HELP
   * -- Required parameters --
   MSGFORMAT   = c           ; Message format, Q, S, or W
   QUEUE       = cccccccc    ; Queue pattern (mult.)
   QUEUE       = cccccccc
   QUEUE       = cccccccc
   * -- Optional parameters --
   ACTIVEUSERS = ccccccc     ; Active users allowed?
   LOGLEVEL    = n           ; Log level 1 .. 4
   QSNERROR    = cccccccc    ; QSN errors, CONTINUE or STOP
/*
/&
```

Figure 76. DSLSDLR (Sequential Data Set Load) Sample JCL (VSE)

Data Set Names

In the JCL, the lowercase parameters have the following meanings:

volid The volume identification of the volume containing the input file or the program library.

infile The data set name of the input file.

program library
The name of the library containing the MERVA ESA programs.

library.sublib The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibraries.

Runtime Parameters

The runtime parameters are passed to DSLSDLR via SYSTSIN under MVS and via SYSIPT under VSE. They have the form KEYWORD = VALUE. Each pair must be coded on a separate line. The input is folded to uppercase and leading and trailing blanks are stripped off from the specified keyword value. Lines starting with '*' are treated as comments, a ';' starts a line comment.

HELP as the only parameter prints a description of the runtime parameters.

Required Parameters

MSGFORMAT
Specifies the format code used to format the input messages:

Q MERVA ESA queue format (was recommended)

S SWIFT I

W SWIFT II.

This parameter is required.

QUEUE
Queue name or queue name pattern. The messages of these queues will be processed. To reload all queues, specify '*'. The entered queue names can be checked in a user exit of DSLSDLR.

This parameter can be specified multiple times. It must be specified at least once.

Optional Parameters

ACTIVEUSERS
Active users. Indicates whether active users are allowed:

CMDONLY Active users in function CMD, FLM, MSC, and USRx are allowed

NO Active users are not allowed

YES Active users are allowed.

This parameter is optional. If omitted, the default value used is NO.

LOGLEVEL
Log level:

1 Only overview data is shown in the listing.

2 Detailed statistical data of the API calls MSGP and PUTR are shown.

3 A WTO message 'Processing ..' is issued for each queue. The first and the last loaded QSN are printed in the listing. When a checking error is found, the MERVA messages are printed.

4 Should be used in case of problems only.

This parameter is optional. If omitted, the default value used is 2.

QSNERROR
Specifies how DSLSDLR should proceed when a QSN to be inserted is already there or a higher QSN exists:

CONTinue DSLSDLR continues.

STOP DSLSDLR stops processing.

This parameter is optional. If omitted, the default value used is STOP. CONTINUE should be used when a DSLSDLR must be performed.

Customization

The following DSLPRM parameters affect DSLSDLR:

PRTNAME Your institution name as it should appear in the printout of (most) REXX batch utilities.

SDDB2 When this parameter is set to YES, direct queue management is enabled (DB2 MVS only).

You can use the following routine in DSLSDLR to reject entered runtime parameters:

USEREXIT_Q1 can be used to reject the entered value for runtime parameter QUEUE, queue pattern.

Sample Printout

The following figure shows the information printed after the execution of the DSLSDLR program.

```
MERVA ESA V4.1 DSLSDLR                          1. Apr. 1999  15:17:23
(C) Copyright IBM Corp. 1999

+ ---------------------------------------------------------- +
|         S A M P L E   B A N K   B o e b l i n g e n        |
+ ---------------------------------------------------------- +


DDDDD    SSSSSS   LLL      SSSSSS   DDDDD    LL       RRRRR
DDDDDD   SSSSSS   LLL      SSSSSS   DDDDDD   LL       RRRRRR
DD  DD   SS       LLL      SS       DD  DD   LL       RR  RR
DD  DD   SS       LLL      SS       DD  DD   LL       RR  RR
DD  DD   SSSSSS   LLL      SSSSSS   DD  DD   LL       RRRRR
DD  DD   SSSSSS   LLL      SSSSSS   DD  DD   LL       RRRRRR
DD  DD       SS   LLL          SS   DD  DD   LL       RR  RR
DD  DD       SS   LLL          SS   DD  DD   LL       RR  RR
DDDDDD   SSSSSS   LLLLLL   SSSSSS   DDDDDD   LLLLLL   RR  RR
DDDDD    SSSSSS   LLLLLL   SSSSSS   DDDDD    LLLLLL   RR  RR
```

Figure 77. DSLSDLR (Sequential Data Set Load) Sample Printout (Part 1 of 2)

```
Load the messages unloaded with DSLSDUR back to MERVA queues.


DSLSDLR_001I : DSLSDLR started by user HEG at 1. Apr. 1999 15:17:23

DSLSDLR_003I : Runtime parameters:
               1. MSGFORMAT   - Message format ..... : Q
               2. QUEUE       - Queue pattern ...... : *
               3. ACTIVEUSERS - Active users allowed : CMDONLY
               4. LOGLEVEL    - Log level .......... : 1
               5. QSNERROR    - QSN error .......... : STOP

DSLSDLR_005I : MERVA ID is MHEG, MERVA name is MERVAESA.

DSLSDLR_009I : The input dataset used is HEG.DSLSDU.FILE.
               Note: This info was obtained by using internal control
                     blocks.

DSLSDLR_013I : EOF of input data set reached (OK).

DSLSDLR_024I : Overview statistical data of the load.

               E                                                  C  O
               R                                                  H  T
               R                                              Q   E  H
               O                   Records   +--- Messages ---+  S  C  E
               R  Number  Queue     Read      Read      Written   N  K  R
               -  ------  --------  --------  --------  --------  -  -  -
                      1  L1ACK         4000      1000      1000
                      2  L1AI0         4000      1000      1000
                      3  L1DE0         4000      1000      1000
                      4  L1VE0         4000      1000      1000
                      5  L2ACK         4000      1000      1000
                      6  L2DE0         4000      1000      1000

DSLSDLR_025I : No. of input records processed : 24026  (incl. comment lines ...
               No. of queue names processed   :     6
               No. of messages read ......... :  6000  (incl. invalid / ...
               No. of messages written ...... :  6000

DSLSDLR_022I : DSLSDLR ended with return code 0 - successful.

DSLSDLR_026I : DSLSDLR ended at 1. Apr. 1999 15:20:29
```

Figure 77. DSLSDLR (Sequential Data Set Load) Sample Printout (Part 2 of 2)

Listing Fields

The 'Overview statistical data' of the listing contains the following information:

ERROR '>' indicates an error with the queue, for example, input records are invalid or a QSN already exists.

Number Running queue number.

Queue Queue name.

Records Read Number of records read from the input data set. The data set is read with EXECIO from the DD name DSLSDSL.

Messages Read
Number of messages read from the input data set.

Messages Written
Number of messages written to MERVA ESA queues.

QSN 'X' indicates that one or more QSNs of the queue already exist or that a higher QSN exists on that queue.

CHECK 'X' indicates that one or more messages of the queue have checking errors (MSGFORMAT = S and W only).

OTHER 'X' indicates that one or more messages of the queue have other errors.

First QSN First loaded QSN. Only printed with log level ≥ 3.

Last QSN Last loaded QSN. Only printed with log level ≥ 3.

Messages and Codes

DSLSDLR ends with the usual return code:

0 Successful

4 Warning

8 Error

12 Severe error.

The messages have the following structure:

DSLSDLR_nnnl Message text

Where:

DSLSDLR Identifies the message as a message of the DSLSDLR utility.

_ Underscore.

nnn Is a 3-digit number used to identify the message.

l Is a single character used to show the type of message. The following characters are used:

- **I** Information message
- **W** Warning message
- **E** Error message
- **S** Severe error message.

The messages are not further documented. Note that the message numbers are subject to change at any time.

Chapter 16. Sequential Data Set Output in REXX (DSLSDOR)

DSLSDOR reads as DSLSDO a batch of messages from a MERVA ESA queue and writes them to a sequential data set.

You can invoke DSLSDOR:

- By the front-end exec DSLSDORF, which displays an ISPF panel where you can enter the runtime parameters and data set names (MVS only)
- By creating and submitting a batch job yourself.

Invoke DSLSDOR via ISPF Panel (MVS only)

Menu

To invoke DSLSDOR via an ISPF panel, enter on an ISPF command line **'TSO %DSLSDORF'**, or on ISPF option 6 **'%DSLSDORF'**. The following panel is displayed:

```
---------------------------- MERVA ESA V4.1 DSLSDOR ----------------------------
COMMAND    ===>

Action     ===>           ( C = create JCL, D = DD statements, H = print Help )
                          ( J = Jobcard, L = edit List, S = create and Submit )
Function   ===> _         ( C = CHECK, D = DELETE, K = KEEP )
Queue Name ===> ________  Message Format ===> _ ( Q = queue fmt, S = SWIFT I ,
                                                          W = SWIFT II )
Fr/To QSN  ===> __________ / __________  ( leave blank for all queue elements )
KEY1 -or-  ===> ____________________
Fr/To KEY1 ===> ____________________ / ____________________
KEY2 -or-  ===> ____________________
Fr/To KEY2 ===> ____________________ / ____________________
Field Name ===> _    ( 1=MSGDST, 2=MSGNET, 3=SWBHLT )  Value ===> ___________
Incor. Msg ===> ___  ( ACC,ALT,CAN,DRO,INC,PUT,ROU )   Queue ===> ________
Log Level  ===> _    ( 1 = basic .. 4 = all )     Perf. Info ===> _  ( No/Yes )
Segmented  ===> _    ( N = output not segmented, Y = output segmented )

Output DS  ===> ________________________________________________ (Output data set)
Altout DS  ===> ________________________________________________ (Alternate Output)
List DS    ===> ________________________________________________ (Listing data set)
JCL DS     ===> ________________________________________________ (JCL data set)

PF1 = HELP                                                 (C) IBM Corp. 1999
```

Figure 78. DSLSDOR (Sequential Data Set Output) Menu

To exit press PF 3.

Note: The installation of the ISPF panel is described in the *MERVA for ESA Installation Guide*.

Runtime Parameters

The following parameters can be specified:

No.	Field	Description
1	Action	**C** Create JCL in data set *JCL DS*, member DSLSDOR **D** Open pop-up window to enter DD statements **H** Print help information to data set *List DS* **J** Open pop-up window to enter JCL cards **L** Edit listing data set *List DS* **S** Create JCL in data set *JCL DS*, member DSLSDOR, and submit it. This parameter is required.
2	Function	**C** Only check the messages of the queue *Queue* **D** Write the messages to the sequential data set *Output DS* (and optionally to the alternate sequential data set *Altout DS*) and delete them from the queue *Queue* **K** Write the messages to the sequential data set *Output DS* (and optionally to the alternate sequential data set *Altout DS*), but do not delete them from the queue *Queue* (keep them). This parameter is required.
3	Queue	The messages of this queue will be processed. This parameter is required.
4	Message Format	Message format **Q** MERVA ESA queue format **S** SWIFT I **W** SWIFT II. This parameter is required.
5	Fr(om) QSN	Only messages in the range From .. To QSN will be processed.
6	To QSN	See From QSN.
7	KEY1	Only messages with a key 1 matching the specified value will be processed. The specified string may contain the wildcards '%' and '*'. **Note:** You can specify either KEY1 or Fr(om)/To KEY1, not both.
8	Fr(om) KEY1	Only messages with a key 1 greater than or equal to the specified value will be processed. The specified string must not contain the wildcards '%' and '*'. See also To KEY1 and the note at KEY1.
9	To KEY1	Only messages with a key 1 less than or equal to the specified value will be processed. The specified string must not contain the wildcards '%' and '*'. See also Fr(om) KEY1 and the note at KEY1.
10	KEY2	Only messages with a key 2 matching the specified value will be processed. The specified string may contain the wildcards '%' and '*'. **Note:** You can specify either KEY2 or Fr(om)/To KEY2, not both.
11	Fr(om) KEY2	Only messages with a key 2 greater than or equal to the specified value will be processed. The specified string must not contain the wildcards '%' and '*'. See also To KEY2 and the note at KEY2.

No.	Field	Description
12	To KEY2	Only messages with a key 2 less than or equal to the specified value will be processed. The specified string must not contain the wildcards '%' and '*'. See also Fr(om) KEY2 and the note at KEY2.
13	Field Name	The identifier for a MERVA field. **1** Only messages with the SWIFT master destination (MSGDST) specified in *(Field) Value* will be processed. **2** Only messages with the message type (MSGNET) specified in *(Field) Value* will be processed. **3** Only messages with the SWIFT basic header logical terminal address (SWBHLT) specified in *(Field) Value* will be processed. This parameter is optional. It can be specified only with *Function* Check or Keep, but not with *Function* Delete.
14	(Field) Value	Value that must be matched for the field specified in *Field Name*: • If *Field Name* is 1, an up to 9 character SWIFT master destination, for example, VNDEBET2A. • If *Field Name* is 2, an up to 8 character MERVA message type. Should start with the letter 'S', for example, S100. • If *Field Name* is 3, an up to 12 character SWIFT basic header logical terminal address, for example, VNDEBET2AXXX. The wildcard '*' can be specified as last (or only) character. This parameter is required when *Field Name* is 1, 2, or 3.
15	Incor. Msg	Specifies what happens when an incorrect message is encountered: **ACC** Incorrect messages are accepted **ALT** Incorrect messages are accepted, but written to the alternate output data set *Altout DS* **CAN** DSLSDOR is terminated **DRO** Incorrect messages are dropped **INC** Only incorrect messages are processed, the correct messages are dropped. **PUT** Incorrect messages are put to the error queue *(Error) Queue* **ROU** Incorrect messages are routed according to the routing (table) of the queue *(Error) Queue*. This parameter is optional, the default value used is ACC. You can specify PUT and ROU only with *Function* Delete. With *Message Format* Q you can specify only ACCept.
16	(Error) Queue	Error queue. This parameter is required with *Incor. Msg* PUT and ROU.

No.	Field	Description
17	Log Level	**1** Only overview data is shown in the listing **2** Detailed data for each message is shown **3** Checking errors are more detailed. Key 1, key 2, and MSGDST are shown even if not specified as selection criterion. **4** Should be used in case of problems only. This parameter is optional. If omitted, the default value used is 2.
18	Perf. Info	**N** No MERVA ESA API performance information is gathered. **Y** The time spent in the MERVA ESA API functions GET, MSGG, PUTB, ROUB, and DELE is gathered. This parameter is optional. If omitted, the default value used is N.
19	Segmented	Indicates whether the output data sets *Output DS* and *Altout DS* should contain segmented messages. **N** The messages in the output data sets are not segmented **Y** The messages in the output data sets are segmented. This parameter is optional. If omitted, the default value used is N.
20	Output DS	Output data set This parameter is required when: • *Action* is C or S • *Function* is D or K.
21	Altout DS	Alternate output data set This parameter is required when: • *Action* is C or S • *Function* is D or K • *Incor. Msg* is ALT.
22	List DS	Listing data set This parameter is required with *Action* C, H, L, and S.
23	JCL DS	JCL data set. The created (and submitted) JCL will be written as member DSLSDOR to this data set. This parameter is required with *Action* C, H, and S.

MVS DD Statements

DSLSDOR requires the following DD statements to be defined:

STEPLIB The load library containing the MERVA ESA programs.

DWSCUR The currency code file, which is required only when you specify runtime parameter *Message Format* S or W and CURCODE=FILE is specified in your DSLPRM.

SYSEXEC The library containing the program DSLSDOR.

DSLSDSA The alternate output data set. DSLSDSA is required only if parameter *Incor. Msg* is ALT.

DSLSDSO The output data set.

SYSTSIN The specified DSLSDOR runtime parameters.

SYSTSPRT The listing data set. Must be preallocated, record format VB, logical record length 136 is recommended, or 'SYSOUT=*'.

- Enter STEPLIB, DWSCUR, and SYSEXEC with *Action* D.
- DSLSDSA, the alternate output data set, DSLSDO, the output data set, and SYSTSPRT, the listing data set, can be specified on the panel.
- SYSTSIN, the DSLSDOR runtime parameters, is generated by DSLSDORF.

First-Time Users

If you are a first-time user, you must allocate three or four data sets in TSO, provide with *Action* D the required DD statements for DWSCUR, STEPLIB, and SYSEXEC, and with *Action* J the required job statement information.

Allocate three or four data sets in TSO:

- The output data set *Output DS*. If you prefer unsegmented messages in the output data set, the LRECL of the output data set must be large enough to hold the largest input message.
- If you plan to use *Incor. Msg* ALT, allocate also the alternate output data set *Altout DS*.
- The data set to which DSLSDORF writes the generated JCL. Should be PO, FB80. Name, for example, uid.MERVA.JCL
- The data set to which DSLSDOR writes its listing: Should be PS, VB136. Name, for example, uid.MERVA.LIST or use 'SYSOUT=*'.

Note: You can also use existing data sets.

The job statement information could look like this:

Edit Jobcards in the window below, PF3 to end

```
//HEGSDOR JOB (DE05020),'S. HEGENAUER',MSGLEVEL=(1,1),MSGCLASS=X,
//            CLASS=A,NOTIFY=HEG,USER=HEG
//*
//*
```

Figure 79. DSLSDOR (Sequential Data Set Output) Job Statement Window

The DD statements could look like this:

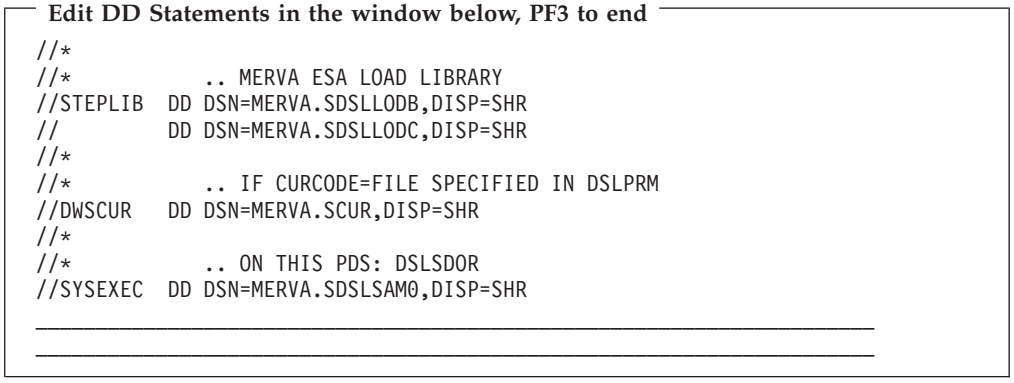

Figure 80. DSLSDOR (Sequential Data Set Output) DD Statements Window

All entered values with the exception of the values entered for KEY fields are remembered in ISPF profile variables.

Invoke DSLSDOR via JCL

You can invoke DSLSDOR also via JCL. If you are running MERVA ESA under VSE, you have to do so.

Job Control Statements for MVS

The following figure shows the MVS JCL to run DSLSDOR.

```
//.......  JOB  ............
//REXXB    EXEC PGM=DSLAREXX,REGION=0K,PARM=DSLSDOR
//*
//*           .. RUNTIME PARAMETERS
//SYSTSIN  DD  *
  * Comments start with '*' and ';'
  * HELP
  * -- Required parameters --
  FUNCTION  = cccccc       ; Queue management function
  MSGFORMAT = c            ; Message format, Q, S, or W
  QUEUE     = cccccccc     ; Queue name
  * -- Optional parameters --
  FROMQSN   = nnnnnnnnnn   ; From QSN
  TOQSN     = nnnnnnnnnn   ; To QSN
  KEY1      = ccccccccc..  ; Key 1 -or-
  FROMKEY1  = ccccccccc..  ; From Key 1
  TOKEY1    = ccccccccc..  ; To Key 1
  KEY2      = ccccccccc..  ; Key 2 -or-
  FROMKEY2  = ccccccccc..  ; From Key 2
  TOKEY2    = ccccccccc..  ; To Key 2
  FLDNAME1  = cccccc       ; MERVA field name
  FLDVALUE1 = ccccccccc..  ; MERVA field value
  INCORRECT = ccccccccc    ; Incorrect message disposition
  ERRQUEUE  = cccccccc     ; Error queue name
  LOGLEVEL  = n            ; Log level 1 .. 4
  PERFORM   = ccc          ; Performance info, YES or No
  SEGMENT   = ccc          ; Output messages segmented?
  USERPARM  = ccccccccc..  ; User specific parameter
/*
//*
//*           .. MERVA ESA LOAD LIBRARY
//STEPLIB  DD DSN=loadlib,DISP=SHR
//*
//*           .. IF CURCODE=FILE SPECIFIED IN DSLPRM
//DWSCUR   DD DSN=curds,DISP=SHR
//*
//*           .. OUTPUT DATASET
//DSLSDSO  DD DSN=outputds,DISP=OLD
//*
//*           .. ALTERNATE OUTPUT DATASET
//DSLSDSA  DD DSN=altoutds,DISP=OLD
//*
//*           .. ON THIS PDS: DSLSDOR
//SYSEXEC  DD DSN=samplib,DISP=SHR
//*
//*           .. LISTING DATASET
//SYSTSPRT DD DSN=listds,DISP=OLD
//
//
```

Figure 81. DSLSDOR (Sequential Data Set Output) Sample JCL (MVS)

Data Set Names

In the JCL, the lowercase data set names have the following meanings:

loadlib The name of the load library containing the MERVA ESA programs.

curds The name of the currency code file. curds is required only when you specify runtime parameter MSGFORMAT = S or W and CURCODE=FILE is specified in your DSLPRM.

outputds The name of the output data set. Must be **DISP=OLD** or **DISP=NEW**.

altoutds The name of the alternate output data set. Must be **DISP=OLD** or **DISP=NEW**.

samplib The name of the library containing the program DSLSDOR.

listds The name of the listing data set. Must be preallocated, record format VB, logical record length 136 recommended.

Job Control Statements for VSE

The following figure shows the VSE JCL to run DSLSDOR.

```
// JOB DSLSDOR ...
// ASSGN SYS025,DISK,VOL=volid,SHR
// DLBL DSLSDSO,outfile,0,SD
// EXTENT SYS025,volid,extent information
// ASSGN SYS026,DISK,VOL=volid,SHR
// DLBL DSLSDSA,altout,0,SD
// EXTENT SYS026,volid,extent information
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION NODUMP
// EXEC DSLAREXX,SIZE=300K,PARM='DSLSDOR'
   * Comments start with '*' and ';'
   * HELP
   * -- Required parameters --
   FUNCTION  = cccccc       ; Queue management function
   MSGFORMAT = c            ; Message format, Q, S, or W
   QUEUE     = cccccccc     ; Queue name
   * -- Optional parameters --
   FROMQSN   = nnnnnnnnnn   ; From QSN
   ...  (see MVS JCL)
/*
/&
```

Figure 82. DSLSDOR (Sequential Data Set Output) Sample JCL (VSE)

Data Set Names

In the JCL, the lowercase parameters have the following meanings:

volid The volume identification of the volume containing the output files or the program library.

outfile The data set name of the output file.

altout The data set name of the alternate output file.

extent information
The extent information of the output files.

program library
The name of the library containing the MERVA ESA programs.

library.sublib The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibraries.

Runtime Parameters

The runtime parameters are passed to DSLSDOR via SYSTSIN under MVS and via SYSIPT under VSE. They have the form KEYWORD = VALUE. Each pair must be coded on a separate line. The input is folded to uppercase with the exception of the entered KEY value and leading and trailing blanks are stripped off from the specified keyword value. Lines starting with an asterisk (*) are treated as comments, a semicolon (;) starts a line comment.

Notes:

1. **HELP** as the only parameter prints a description of the runtime parameters.
2. The entered values for the following keywords are not translated to uppercase: KEY1, FROMKEY1, TOKEY1, KEY2, FROMKEY2, and TOKEY2.

Required Parameters

FUNCTION

Queue Management Function:

CHECK Only check the messages in the MERVA ESA queue QUEUE.

DELete Write the messages to the sequential data set DSLSDSO (and optionally to the alternate sequential data set DSLSDSA), and delete them from the MERVA ESA queue QUEUE

KEEP Write the messages to the sequential data set DSLSDSO (and optionally to the alternate sequential data set DSLSDSA), but do not delete them from the MERVA ESA queue QUEUE (keep them).

This parameter is required.

MSGFORMAT

Specifies the format code used to format the messages:

Q MERVA ESA queue format

S SWIFT I

W SWIFT II.

This parameter is required.

QUEUE

Queue name. The messages of this queue will be processed.

This parameter is required.

Optional Parameters

With the following parameters you can select the queue elements to be processed. If not specified, all are processed.

FROMQSN

From QSN. Only messages with a QSN greater than or equal to this QSN will be processed.

This parameter is optional, the default value used is FROMQSN = 0. See also TOQSN.

TOQSN
To QSN. Only messages with a QSN less than or equal to this QSN will be processed.

This parameter is optional, the default value used is that all messages are processed. See also FROMQSN.

KEY1
Key 1. Only messages with a key 1 matching the specified value will be processed. The specified string may contain the wildcards '%' and '*'.

Note: You can specify either KEY1 or FROMKEY1 .. TOKEY1, not both.

FROMKEY1
From key 1. Only messages with a key 1 greater than or equal to the specified value will be processed. The specified string must not contain the wildcards '%' and '*'. See also To KEY1 and the note at KEY1.

TOKEY1
To key 1. Only messages with a key 1 less than or equal to the specified value will be processed. The specified string must not contain the wildcards '%' and '*'. See also FROMKEY1 and the note at KEY1.

KEY2
Key 2. Only messages with a key 2 matching the specified value will be processed. The specified string may contain the wildcards '%' and '*'.

Note: You can specify either KEY2 or FROMKEY2 .. TOKEY2, not both.

FROMKEY2
From key 2. Only messages with a key 2 greater than or equal to the specified value will be processed. The specified string must not contain the wildcards '%' and '*'. See also To KEY2 and the note at KEY2.

TOKEY2
To key 2. Only messages with a key 2 less than or equal to the specified value will be processed. The specified string must not contain the wildcards '%' and '*'. See also FROMKEY2 and the note at KEY2.

FLDNAME1
MERVA field name:

MSGDST	Specifies that only messages with the SWIFT master destination specified in parm FLDVALUE1 should be processed.
MSGNET	Specifies that only messages with the message type specified in parm FLDVALUE1 should be processed.
SWBHLT	Specifies that only messages with the SWIFT basic header logical terminal address specified in parm FLDVALUE1 should be processed.

This parameter is optional. It can be specified only with FUNCTION = CHECK or KEEP, but not with FUNCTION = DELETE.

FLDVALUE1
MERVA field value:

- If parameter FLDNAME1 = MSGDST, an up to 9 character SWIFT master destination

- If parameter FLDNAME1 = MSGNET, an up to 8 character MERVA message type. Should start with 'S', for example, S100 or S1*.
- If parameter FLDNAME1 = SWBHLT, an up to 12 character SWIFT basic header logical terminal address.

The wildcard '*' is allowed as last (or only) character.

This parameter is required when FLDNAME1 = MSGDST, MSGNET, or SWBHLT.

Other Parameters

With the following parameters you can specify what is to happen with incorrect messages, how detailed the information given is, and the output format:

INCORRECT

Incorrect message disposition. Specifies what happens when an incorrect message is encountered. If specified, the keyword value must be one of the following:

ACCept	Incorrect messages are accepted.
acceptALT	Incorrect messages are accepted, but written to the alternate output data set DSLSDSA.
CANcel	DSLSDOR is terminated.
DROp	Incorrect messages are dropped.
INCorronly	Only incorrect messages are processed, correct messages are dropped.
PUT	Incorrect messages are put to the error queue ERRQUEUE.
ROUte	Incorrect messages are routed according to the routing (table) of the queue ERRQUEUE.

This parameter is optional, the default value used is INCORRECT = ACCEPT.

Notes:

1. You can specify PUT and ROUTE only with FUNCTION = DELETE, but not with FUNCTION = CHECK or KEEP. This is because CHECK and KEEP should not change the MERVA ESA queue data set in any way.
2. With MSGFORMAT = Q you cannot specify an INCORRECT message disposition other than ACCEPT. This is because messages in the queue format are not checked for correctness.
3. If the parameters FLDNAME1 and FLDVALUE1 are specified so that only selected messages should be processed, the parameter INCORRECT applies only to messages that passed this filter. If, for example, the following parameters are specified:
   a. FLDNAME1 = MSGNET
   b. FLDVALUE = S1*
   c. INCORRECT = CANCEL.

   Now an erroneous S200 message will *not* cancel DSLSDOR, as only S1* messages are processed.

ERRQUEUE

Error queue. The name of the error queue. Can be checked in a user exit of DSLSDOR.

This parameter is required with INCORRECT = PUT and ROUTE.

LOGLEVEL
Log level:

1 Only overview data is shown in the listing.

2 Detailed data for each message is shown.

3 Checking errors are more detailed. Key 1, key 2, and MSGDST are shown even if not specified as selection criterion.

4 Should be used in case of problems only.

This parameter is optional, the default value used is LOGLEVEL = 2.

PERFORM
Performance information. Indicates whether performance information about the API commands GET, MSGG, PUTB, ROUB, and DELE is to be gathered. If specified, the keyword value must be Yes or No.

This parameter is optional, the default value used is PERFORM = NO.

SEGMENT
Segmented output messages. Indicates whether the output data sets DSLSDSO and DSLSDSA are to contain segmented messages:

No The messages in the output data sets are not segmented.

Yes The messages in the output data sets are segmented.

This parameter is optional, the default value used is SEGMENT = NO.

With the following parameter you can pass a user-specific parameter to DSLSDOR:

USERPARM
User-specific parameter. The specified data is accepted by DSLSDOR and available in the variable *parm_userparm*.

This parameter is optional.

Required Parameters for VSE

VSEBLKSIZE
DSLSDSO BLKSIZE. BLKSIZE of output SAM file DSLSDSO, maximum 32761.

This parameter is required under VSE.

VSERECFORM
DSLSDSO RECFORM. RECFORM of output SAM file DSLSDSO: FIXUNB, FIXBLK, VARUNB, or VARBLK.

This parameter is required under VSE.

VSERECSIZE
DSLSDSO RECSIZE. RECSIZE of output SAM file DSLSDSO, maximum 32761.

This parameter is required under VSE, when VSERECFORM = FIXUNB or FIXBLK.

EDIFACT FINPAY Conversion

DSLSDOR supports basic convertion of SWIFT MT121 messages to EDIFACT FINPAY messages. Only the bank message part of MT121 messages between '{4:' and '//-}' is written then to the data set (// = X'0D25' (crlf), carriage return/line feed).

Runtime Parameters

Specify the runtime parameter USERPARM = XFINPAY. Additionally, other parameters must be specified as follows:

1. MSGFORMAT = W
2. FLDNAME1 = MSGNET
3. FLDVALUE1 = S121

Customization

The following DSLPRM parameters affect DSLSDOR:

PRTNAME Your institution name as it is to appear in the printout of (most) REXX batch utilities.

SDDB2 When this parameter is set to YES, direct queue management is enabled (DB2 MVS only).

You can use the following routines in DSLSDOR to reject entered runtime parameters:

1. USEREXIT_Q1 can be used to reject the entered value for runtime parameter QUEUE1, input queue.
2. USEREXIT_Q2 can be used to reject the entered value for runtime parameter QUEUE2, error queue.

Sample Printout

```
MERVA ESA V4.1 DSLSDOR                          9. Apr. 1999  18:38:09
(C) Copyright IBM Corp. 1999

+ ---------------------------------------------------------- +
|        S A M P L E   B A N K   B o e b l i n g e n         |
+ ---------------------------------------------------------- +


DDDDD    SSSSSS   LLL      SSSSSS   DDDDD     OOOO    RRRRR
DDDDDD   SSSSSS   LLL      SSSSSS   DDDDDD   OOOOOO   RRRRRR
DD  DD   SS       LLL      SS       DD  DD   OO  OO   RR  RR
DD  DD   SS       LLL      SS       DD  DD   OO  OO   RR  RR
DD  DD   SSSSSS   LLL      SSSSSS   DD  DD   OO  OO   RRRRR
DD  DD   SSSSSS   LLL      SSSSSS   DD  DD   OO  OO   RRRRRR
DD  DD       SS   LLL          SS   DD  DD   OO  OO   RR  RR
DD  DD       SS   LLL          SS   DD  DD   OO  OO   RR  RR
DDDDDD   SSSSSS   LLLLLL   SSSSSS   DDDDDD   OOOOOO   RR  RR
DDDDD    SSSSSS   LLLLLL   SSSSSS   DDDDD     OOOO    RR  RR


REXX version of DSLSDO - Sequential Data Set Output.
HELP as the only parameter prints a description.


DSLSDOR_001I : DSLSDOR started by user HEG at 9. Apr. 1999 18:38:09

DSLSDOR_003I : Runtime parameters:
                1. FUNCTION   - Queue mgmt function : KEEP
                2. MSGFORMAT  - Message format .... : W
                3. QUEUE      - Queue name ........ : L3DE0
                4. ERRQUEUE   - Error queue name .. :
                5. FLDNAME1   - MERVA field name .. :
                6. FLDVALUE1  - MERVA field value   :
                7. FROMKEY1   - From KEY1 ......... :
                8. FROMKEY2   - From KEY2 ......... :
                9. FROMQSN    - From QSN .......... :
               10. INCORRECT  - Incorrect msg disp. : ACCEPT
               11. KEY1       - KEY1 .............. :
               12. KEY2       - KEY2 .............. :
               13. LOGLEVEL   - Log level ......... : 2
               14. PERFORM    - Performance info .. : NO
               15. SEGMENT    - Segmentation ind.   : NOSEGMENT
               16. TOKEY1     - To KEY1 ........... :
               17. TOKEY2     - To KEY2 ........... :
               18. TOQSN      - To QSN ............ :
               19. USERPARM   - User specific data  : MYDATA

DSLSDOR_006I : MERVA ID is MHEG, MERVA name is MERVAESA.

DSLSDOR_007I : Queue management module ..... : DSLQMCNV
               Access method / features used : VXBD

DSLSDOR_011I : The output dataset used is HEG.DSLSDO.FILE.
               Note: This info was obtained by using internal control
                     blocks.

DSLSDOR_012I : The record length of output data set DSLSDSO is 32752.

DSLSDOR_115W : Message no. 4 with QSN 4:
               MERVA API command MSGG ended with INTRC 00.
               MFS has detected checking errors.
```

Figure 83. DSLSDOR (Sequential Data Set Output) Sample Printout (Part 1 of 2)

```
DSLSDOR_025I : Detailed statistical data for stage 1
               (Input queue L3DE0 to output dataset)

               E  M
               R  A
               R  T
               O  C                                GET MSGG                Length
               R  H  Number   Input QSN    MT      rc  rc   Output    rc  Output QSN
               -  -  ------   ----------   -----   --  --   --------  --  ----------
                          1   0000000001   S100    ok  ok   DSLSDSO   ok     170 char.
                          2   0000000002   S100    ok  ok   DSLSDSO   ok     130 char.
                          3   0000000003   S100    ok  ok   DSLSDSO   ok     134 char.
               >          4   0000000004   S100    ok  00   DSLSDSO   ok     138 char.
                          5   0000000005   S100    ok  ok   DSLSDSO   ok     134 char.

DSLSDOR_029I : Overview statistical data

               DELE = Delete a message from a MERVA queue
               GET  = Read a message from a MERVA queue
               MSGG = Map a message from internal buffer to external format
               PUTB = Move a message to another MERVA queue
               ROUB = Route a message to a MERVA queue with automatic delete

               ---  Stage 1  ---

               No. of elements matching QSN and KEY .. : 5

               No. of messages GET ok ................ : 5
               No. of messages GET failed ............ : 0

               No. of messages MSGG ok ............... : 4
               No. of messages MSGG intrc 00 (checking): 1
               No. of messages MSGG failed ........... : 0

               No. of messages written to dataset .... : 5
               No. of records written to dataset ..... : 5
               No. of messages write to dataset failed : 0

DSLSDOR_030I : Number of (matching) input messages read .. : 5
               Number of messages ok ..................... : 4
               Number of messages in error ............... : 1
               Number of messages written to DSLSDSO ..... : 5

DSLSDOR_023I : DSLSDOR ended with return code 4 - Warning.

DSLSDOR_031I : DSLSDOR ended at 9. Apr. 1999 18:38:11
```

Figure 83. DSLSDOR (Sequential Data Set Output) Sample Printout (Part 2 of 2)

Listing Fields

The 'Detailed statistical data' of the listing contains the following information:

For stage 1:

ERROR '>' indicates an error with the message, for example, a checking error, or the PUT to the intermediate queue failed.

MATCH 'N' indicates that this message does not match a specified MSGDST, MSGNET, or SWBHLT value.

Number Running number.

Input QSN QSN of the message in the input queue.

MT Message type (MSGNET).

GET rc Return code of API function GET - 'ok' indicates that the message was successfully read from the input queue.

MSGG rc Return code of API function MSGG - 'ok' indicates that the message was successfully translated from the internal MERVA ESA format to the external format.

Output Output medium.

- **DSLSDSO** The normal sequential output data set.
- **DSLSDSA** The alternate sequential output data set.
- **queue name** The error queue name.

rc Return code of the Output operation, either EXECIO to a sequential data set, or PUT or ROUTE of erroneous messages to an error queue.

Length Length of external format of the message in characters.

Output QSN QSN of the erroneous message in the error queue.

Key 1 Key-1 value. Is printed in the second line, but only if specified as selection criterion or log level ≥ 3.

Key 2 Key-2 value. Is printed in the second line, but only if specified as selection criterion or log level ≥ 3.

MSGDST SWIFT master destination. Is printed in the second line, but only if specified as selection criterion or log level ≥ 3.

SWBHLT SWIFT basic header logical terminal address. Is printed in the second line, but only if specified as selection criterion or log level ≥ 3.

For performance information of stage 1:

Number Running number.

Input QSN QSN of the message in the input queue.

MT Message type (MSGNET).

Length Length of external format of the message in characters.

GET rc Return code of API function GET – 'ok' indicates that the message was successfully read from the input queue.

GET time Time in seconds spent in API function GET.

MSGG rc Return code of API function MSGG – 'ok' indicates that the message was successfully translated from the internal MERVA ESA format to the external format.

MSGG time Time in seconds spent in API function MSGG.

PUTB rc Return code of API function PUTB, move an erroneous message from the input queue to the error queue. Is printed only with Incorrect message disposition PUT.

PUTB time Time in seconds spent in API function PUTB.

ROUB rc Return code of API function ROUB, route an erroneous message from the input queue to the error queue. Is printed only with Incorrect message disposition ROUTE.

ROUB time Time in seconds spent in API function ROUB.

For stage 2:

Err	'>' indicates an error with the message, for example, the DELETE from the input queue failed.
Number	Running number.
Input QSN	QSN of the message in the input queue.
DELE rc	Return code of API function DELE, delete a message from the input queue.
DELE time	Time in seconds spent in API function DELE.

Notes:

1. With functions CHECK and KEEP there is no stage-2 processing.
2. `rc <= -2` indicates an error in DSLSDOR or the REXX host command environment, refer to the *MERVA for ESA Application Programming Interface Guide* for a description of the return codes.

Messages and Codes

DSLSDOR ends with the return code:

0	Successful
4	Warning
8	Error
12	Severe error.

The messages have the following structure:

DSLSDOR_nnnl Message text

Where:

DSLSDOR	Identifies the message as a message of the DSLSDOR utility.
_	Underscore.
nnn	Is a 3-digit number used to identify the message.
l	Is a single character used to show the type of message. The following characters are used:

I	Iinformation message.
W	Warning message.
E	Error message.
S	Severe error message.

The messages are not further documented. Note that the message numbers are subject to change at any time.

Comparison of DSLSDOR with DSLSDO

- **Capabilities provided by both, DSLSDO and DSLSDOR**

  DSLSDO and DSLSDOR both provide the following:

  1. Restart capable (same mechanism as DSLSDO).
  2. The messages in the output data sets can be segmented or unsegmented.

3. Write WTO messages about processing.

   Note: WTO messages are also written to the journal.

- **Advantages of DSLSDOR**

  DSLSDOR has the following advantages over DSLSDO:

  1. The messages of the input queue can be:
     - Written to a sequential data set and then be deleted from the queue (this does DSLSDO)
     - Written to a sequential data set but kept in the queue
     - Checked only.
  2. A FROM and TO QSN can be specified.
  3. A KEY1 or a FROM and TO KEY1 can be specified.
  4. A KEY2 or a FROM and TO KEY2 can be specified.
  5. You specify that only messages are processed that match a specified:
     - MSGDST (SWIFT master destination)
     - MSGNET (message type)
     - SWBHLT (SWIFT basic header logical terminal address).
  6. Additional incorrect message dispositions:

     ACCEPTALT Write incorrect messages to an alternate output data set.

     DROP Drop incorrect messages.

     INCORRONLY Only incorrect messages are processed.

     PUT Incorrect messages are put to an error queue.
  7. Detailed error message for each message.
  8. Overview and detailed statistical data.
  9. The message type and key values of each message can be shown.
  10. Performance information can be gathered.
  11. User exits to check the queue names.
  12. Tells name of output data sets (by using internal control blocks, MVS only).
  13. ISPF front-end panel DSLSDORP (MVS only).

- **Restrictions of DSLSDOR**

  DSLSDOR has the following restrictions compared to DSLSDO:

  1. For the output data sets only DISP=OLD and DISP=NEW are possible, DISP=MOD is not supported.
  2. The output data sets must be on disk (not on tape).
  3. Does not support VBS records.
  4. Supports only message format Q, S, and W.
  5. UMR is also written for the restart message.
  6. User exit DSLMU022 is not supported.

Chapter 17. Sequential Data Set Unload in REXX (DSLSDUR)

DSLSDUR unloads the messages from all or specified MERVA ESA queues and writes them together with the following information to a sequential data set:

- Queue name
- QSN
- Key 1 and key 2
- *Write back* indicator
- Date and time of unload
- Length of the message in the unloaded format.

The messages are kept in their queues, that is, they are not deleted. You can use DSLSDLR to load the messages back to their MERVA ESA queues, preserving the queue name, QSN, key values, and the *write back* indicator.

Dependencies: MERVA ESA must be active.

Unload/Reload Data Set Layout

DSLSDUR writes the messages in the same segmented format as DSLSDO and DSLSDOR with a record length of 1024. The queue name, QSN, .. are written in an extra line ahead the message lines. This extra line is identified by the character 'A' in column 1.

The layout of 'A' lines is as follows:

No.	Field	Offset	Len	Description
1	Line identifier	0	1	Character 'A'
2		1	1	One blank
3	Queue name	2	8	Queue name
4		10	1	One blank
5	QSN	11	10	QSN
6		21	1	One blank
7	Date	22	8	Unload date YYYYMMDD
8		30	1	One blank
9	Time	31	6	Unload time HHMMSS
10		37	1	One blank
11	Message length	38	8	Length of the message in the unloaded format as written to the data set
12		46	1	One blank
13	Write back ind.	47	1	*Write back* indicator: X = set, blank = not set
14		48	1	One blank
15	Key 1	49	24	Key 1 value
16		73	1	One blank
17	Key 2	74	24	Key 2 value

For example:

```
*** Queue L1DE0
A L1DE0    0000002831 19990916 111258 00000136   MYKEY1A
0{1:F01VNDEBET2AXXX0000000000}{2:I100VNDOBET2AXXXN}{4:  :20:001  :32A:980101DEM..
A L1DE0    0000002832 19990916 111258 00000128   MYKEY1B
0{1:F01VNDEBET2AXXX0000000000}{2:I100VNDOBET2AXXXN}{4:  :20:002  :32A:980101USD..
```

Figure 84. DSLSDUR (Sequential Data Set Unload) Sample Unload Data Set

The output data set must have record format FB and logical record length 1024. Under VSE, the data set is written with BLKSIZE 6144, RECFORM FIXBLK, and RECSIZE 1024.

Job Control Statements for MVS

The following figure shows the MVS JCL to unload MERVA ESA queues in batch.

```
//.......  JOB  ............
//REXXB    EXEC PGM=DSLAREXX,REGION=8M,PARM=DSLSDUR
//*
//SYSTSIN  DD *
  * Comments start with '*' and ';'
  * HELP
  * -- Required parameters --
  MSGFORMAT   = c           ; Message format, Q, S, or W
  QUEUE       = cccccccc    ; Queue pattern (mult.)
  QUEUE       = cccccccc
  QUEUE       = cccccccc
  * -- Optional parameters --
  ACTIVEUSERS = ccccccc     ; Active users allowed?
  LOGLEVEL    = n           ; Log level 1 .. 4
/*
//*
//*           .. MERVA ESA LOAD LIBRARY
//STEPLIB  DD DSN=loadlib,DISP=SHR
//*
//*           .. IF MSGFORMAT=S/W AND CURCODE=FILE SPEC. IN DSLPRM
//DWSCUR   DD DSN=curds,DISP=SHR
//*
//*           .. OUTPUT DATA SET (FB1024)
//DSLSDSU  DD DSN=outputds,DISP=OLD
//*
//*           .. ON THIS PDS: DSLSDUR
//SYSEXEC  DD DSN=samplib,DISP=SHR
//*
//*           .. LISTING DATASET (VB136)
//SYSTSPRT DD DSN=listds,DISP=OLD
//
```

Figure 85. DSLSDUR (Sequential Data Set Unload) Sample JCL (MVS)

Data Set Names

In the JCL, the lowercase data set names have the following meanings:

loadlib The name of the load library containing the MERVA ESA programs.

curds The name of the currency code file. curds is required only when you specify runtime parameter MSGFORMAT = S or W and CURCODE=FILE is specified in your DSLPRM. Must be record format FB, logical record length 1024, and have **DISP=OLD** or **DISP=NEW**.

outputds The name of the output data set.

samplib The name of the library containing the program DSLSDUR.

listds The name of the listing data set. The data set must be preallocated, record format VB, logical record length 136 recommended.

Job Control Statements for VSE

The following figure shows the VSE JCL to run DSLSDUR.

```
// JOB DSLSDUR ...
// ASSGN SYS025,DISK,VOL=volid,SHR
// DLBL DSLSDSU,outfile,0,SD
// EXTENT SYS025,volid,extent information
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION NODUMP
// EXEC DSLAREXX,SIZE=300K,PARM='DSLSDUR'
   * Comments start with '*' and ';'
   * HELP
   * -- Required parameters --
   MSGFORMAT   = c           ; Message format, Q, S, or W
   QUEUE       = cccccccc    ; Queue pattern (mult.)
   QUEUE       = cccccccc
   QUEUE       = cccccccc
   * -- Optional parameters --
   ACTIVEUSERS = ccccccc     ; Active users allowed?
   LOGLEVEL    = n           ; Log level 1 .. 4
/*
/&
```

Figure 86. DSLSDUR (Sequential Data Set Unload) Sample JCL (VSE)

Data Set Names

In the JCL, the lowercase parameters have the following meanings:

volid The volume identification of the volume containing the output file or the program library.

outfile The data set name of the output file.

extent information
The extent information of the output file.

program library
The name of the library containing the MERVA ESA programs.

library.sublib The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibraries.

Runtime Parameters

The runtime parameters are passed to DSLSDUR via SYSTSIN under MVS and via SYSIPT under VSE. They have the form KEYWORD = VALUE. Each pair must be coded on a separate line. The input is folded to uppercase, and leading and trailing blanks are stripped off from the specified keyword value. Lines starting with '*' are treated as comments, a ';' starts a line comment.

HELP as the only parameter prints a description of the runtime parameters.

Required Parameters

MSGFORMAT
Specifies the format code used to format the messages:

Q MERVA ESA queue format (recommended)

S SWIFT I

W SWIFT II.

This parameter is required.

QUEUE
Queue name or queue name pattern. The messages of these queues will be processed. The entered queue name can be checked in a user exit of DSLSDUR.

This parameter can be specified multiple times. It must be specified at least once. To unload, for example, all queues which names start with L1, and the queues L2DE0 and L3DE0, you would specify:

```
QUEUE = L1*
QUEUE = L2DE0
QUEUE = L3DE0
```

Optional Parameters

ACTIVEUSERS
Indicates whether active users are allowed:

CMDONLY Active users in function CMD, FLM, MSC, and USRx are allowed.

NO Active users are not allowed.

YES Active users are allowed.

This parameter is optional. If omitted, the default value used is NO.

LOGLEVEL
Log level:

1 Only overview data is shown in the listing.

2 Detailed statistical data of the API calls GETU and MSGG is shown.

3 A WTO message 'Unloading ··' is issued for each queue. The first and last unloaded QSN are printed in the listing. When a checking error is found, the MERVA messages are printed.

4 Should be used in case of problems only.

This parameter is optional. If omitted, the default value used is 2.

Customization

The following DSLPRM parameters affect DSLSDUR:

PRTNAME Your institution name as it is to appear in the printout of (most) REXX batch utilities.

SDDB2 When this parameter is set to YES, direct queue management is enabled (DB2 MVS only).

You can use the following routines in DSLSDUR to reject entered runtime parameters:

USEREXIT_Q1 can be used to reject the entered value for runtime parameter QUEUE, queue pattern.

Sample Printout

The following figure shows the information printed after the execution of the DSLSDUR program.

```
MERVA ESA V4.1 DSLSDUR                          1. Apr. 1999  15:08:40
(C) Copyright IBM Corp. 1999

+ ---------------------------------------------------------- +
|         S A M P L E   B A N K   B o e b l i n g e n        |
+ ---------------------------------------------------------- +


DDDDD    SSSSSS   LLL      SSSSSS   DDDDD    UU  UU   RRRRR
DDDDDD   SSSSSS   LLL      SSSSSS   DDDDDD   UU  UU   RRRRRR
DD  DD   SS       LLL      SS       DD  DD   UU  UU   RR  RR
DD  DD   SS       LLL      SS       DD  DD   UU  UU   RR  RR
DD  DD   SSSSSS   LLL      SSSSSS   DD  DD   UU  UU   RRRRR
DD  DD   SSSSSS   LLL      SSSSSS   DD  DD   UU  UU   RRRRRR
DD  DD       SS   LLL          SS   DD  DD   UU  UU   RR  RR
DD  DD       SS   LLL          SS   DD  DD   UU  UU   RR  RR
DDDDDD   SSSSSS   LLLLLL   SSSSSS   DDDDDD   UUUUUU   RR  RR
DDDDD    SSSSSS   LLLLLL   SSSSSS   DDDDD    UUUUUU   RR  RR


Unload the messages from all or specified MERVA ESA queues and
write them with their queue name, QSN, .. to a sequential data set.
Use utility DSLSDLR to load the messages back to MERVA queues.

DSLSDUR_001I : DSLSDUR started by user HEG at 1. Apr. 1999 15:08:40

DSLSDUR_003I : Runtime parameters:
               1. MSGFORMAT   - Message format ..... : Q
               2. QUEUE       - Queue pattern ...... : L1*
                                                       L2*
               3. ACTIVEUSERS - Active users allowed : CMDONLY
               4. LOGLEVEL    - Log level .......... : 1
```

Figure 87. DSLSDUR (Sequential Data Set Unload) Sample Printout (Part 1 of 2)

```
DSLSDUR_005I : MERVA ID is MHEG, MERVA name is MERVAESA.

DSLSDUR_011I : The output dataset used is HEG.DSLSDU.FILE.
               Note: This info was obtained by using internal control
                     blocks.

DSLSDUR_020I : Number of queues to unload: 6
               Number of queues unloaded : 6
DSLSDUR_021I : Overview statistical data of the unload.

               E                                                    C  O
               R                                              B  H  T
               R                                              U  E  H
               O                   +--- Messages ---+  Records  S  C  E
               R  Number  Queue    Read      Written   Written  Y  K  R
               -  ------  -------- --------  --------  -------- -  -  -
                      1  L1ACK         1000      1000      4000
                      2  L1AI0         1000      1000      4000
                      3  L1DE0         1000      1000      4000
                      4  L1VE0         1000      1000      4000
                      5  L2ACK         1000      1000      4000
                      6  L2DE0         1000      1000      4000

DSLSDUR_022I : Total number of
               - messages unloaded   : 6.000
               - msg records written : 24.000
               - msg bytes written   : 8.250.345

DSLSDUR_018I : DSLSDUR ended with return code 0 - successful.

DSLSDUR_023I : DSLSDUR ended at 1. Apr. 1999 15:10:36
```

Figure 87. DSLSDUR (Sequential Data Set Unload) Sample Printout (Part 2 of 2)

Listing Fields

The 'Overview statistical data' of the listing contains the following information:

ERROR '>' indicates an error with the queue, for example, messages could not be read or messages are BUSY.

Number Running queue number.

Queue Queue name.

Messages Read
Number of messages read from the queue. The messages are read with the API function GETU.

Messages Written
Number of messages written to the output data set.

Records Written
Number of records written to the output data set. The records are written with EXECIO to the DD-name DSLSDSU. As an additional 'A' line is needed, and one record can hold only 1024 characters, the number of records is always larger than the number of messages.

BUSY 'X' indicates that one or more messages of the queue are BUSY.

CHECK 'X' indicates that one or more messages of the queue have checking errors (MSGFORMAT = S or W only).

OTHER 'X' indicates that one or more messages of the queue have other errors.

First QSN First unloaded QSN. Only printed with log level ≥ 3.

Last QSN Last unloaded QSN. Only printed with log level ≥ 3.

Messages and Codes

DSLSDUR ends with the usual return code:

0 Successful

4 Warning

8 Error

12 Severe error.

The messages have the following structure:

DSLSDUR_nnnl Message text

Where:

DSLSDUR Identifies the message as a message of the DSLSDUR utility.

_ Underscore.

nnn Is a 3-digit number used to identify the message.

l Is a single character used to show the type of message. The following characters are used:

- **I** Information message
- **W** Warning message
- **E** Error message
- **S** Severe error message.

The messages are not further documented. Note that the message numbers are subject to change at any time.

Chapter 18. Sequential Data Set Print in REXX (DSLSDYR)

DSLSDYR reads as DSLSDY a batch of messages from a MERVA ESA queue and prints them on a line (SYSOUT) printer or to a file.

Dependencies: MERVA ESA must be active.

Job Control Statements for MVS

The following figure shows the JCL to run the batch print program DSLSDYR under MVS.

```
//.......  JOB  .............
//REXXB    EXEC PGM=DSLAREXX,REGION=8M,PARM=DSLSDYR
//*
//SYSTSIN  DD *
  * Comments start with '*' and ';'
  * HELP
  * -- Required parameters --
  FUNCTION    = cccccc       ; Queue Management Function
  MSGFORMAT   = c            ; Format ID to be used, e.g., E
  QUEUE       = cccccccc     ; Queue
  * -- Optional parameters --
  BOTTOMFRAME = cccccccc     ; Bottom frame MCB, e.g. 0BOT
  CASE        = ccccc        ; ASIS or UPPER
  COMPRESSION = n            ; Compression format 0 .. 4
  FLDNAME1    = cccccc       ; MERVA field name
  FLDVALUE1   = ccccccccc..  ; MERVA field value
  FROMKEY1    = ccccccccc..  ; From Key 1
  FROMKEY2    = ccccccccc..  ; From Key 2
  FROMQSN     = nnnnnnnnnn   ; From QSN
  FROMUMR     = nnnnnnnn     ; From UMR
  KEY1        = ccccccccc..  ; Key 1
  KEY2        = ccccccccc..  ; Key 2
  LINESPP     = nnnnn        ; Lines per page
  LOGLEVEL    = n            ; Log level 1 .. 4
  MCB         = cccccccc     ; MCB name for print
  PERFORM     = ccc          ; Performance info, YES or NO
  SEPARATOR   = ccc          ; Separator page, YES or NO
  SEPLINESPP  = nnn          ; Lines per separator page
  TOKEY1      = ccccccccc..  ; To Key 1
  TOKEY2      = ccccccccc..  ; To Key 2
  TOPFRAME    = cccccccc     ; Top frame MCB, e.g. 0TOP
  TOQSN       = nnnnnnnnnn   ; To QSN
  TOUMR       = nnnnnnnn     ; To UMR
  USERPARM    = ccccccccc..  ; User specific parameter
/*
```

Figure 88. DSLSDYR (Sequential Data Set Print) Sample JCL (MVS) (Part 1 of 2)

```
//*
//*           .. MERVA ESA LOAD LIBRARY
//STEPLIB  DD DSN=loadlib,DISP=SHR
//*
//*           .. IF CURCODE=FILE SPECIFIED IN DSLPRM
//DWSCUR   DD DSN=curds,DISP=SHR
//*
//*           .. OUTPUT DATA SET (FBA133) OR SYSTEM PRINTER
//DSLSDSY  DD DSN=outputds,DISP=OLD
//*
//*           .. ON THIS PDS: DSLSDYR
//SYSEXEC  DD DSN=samplib,DISP=SHR
//*
//*           .. LISTING DATASET (VB136)
//SYSTSPRT DD DSN=listds,DISP=OLD
//
```

Figure 88. DSLSDYR (Sequential Data Set Print) Sample JCL (MVS) (Part 2 of 2)

Data Set Names

In the JCL, the lowercase data set names have the following meanings:

loadlib The name of the load library containing the MERVA ESA programs.

curds The name of the currency code file. This is required only when CURCODE=FILE is specified in your DSLPRM.

outputds The name of the output data set. Should be record format FBA, logical record length 133.

samplib The name of the library containing the program DSLSDYR.

listds The name of the listing data set. Must be preallocated, record format VB, logical record length 136 recommended.

Job Control Statements for VSE

The following figure shows the VSE JCL to run DSLSDYR.

```
// JOB DSLSDYR ...
// ASSGN SYS025,DISK,VOL=volid,SHR
// DLBL DSLSDSY,outfile,0,SD
// EXTENT SYS025,volid,extent information
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION NODUMP
// EXEC DSLAREXX,SIZE=300K,PARM='DSLSDYR'
   * Comments start with '*' and ';'
   * HELP
   * -- Required parameters --
   FUNCTION    = cccccc       ; Queue Management Function
   MSGFORMAT   = c            ; Format ID to be used, e.g., E
   QUEUE       = cccccccc     ; Queue
   * -- Optional parameters --
   BOTTOMFRAME = cccccccc     ; Bottom frame MCB, e.g. 0BOT
   CASE        = ccccc        ; ASIS or UPPER
   ...  (see MVS JCL)
/*
/&
```

Figure 89. DSLSDYR (Sequential Data Set Print) Sample JCL (VSE)

Data Set Names

In the JCL, the lowercase parameters have the following meanings:

volid The volume identification of the volume containing the output file or the program library.

outfile The data set name of the output file.

extent information
The extent information of the output file.

program library
The name of the library containing the MERVA ESA programs.

library.sublib The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibraries.

To print the generated sequential output file to SYSLST, you may use the VSE DITTO utility with parameter SFD.

Runtime Parameters

The runtime parameters are passed to DSLSDYR via SYSTSIN under MVS and via SYSIPT under VSE. They have the form KEYWORD = VALUE. Each pair must be coded on a separate line. The input is folded to uppercase with the exception of the entered KEY values and leading and trailing blanks are stripped off from the specified keyword value. Lines starting with '*' are treated as comments, ';' starts a line comment.

Notes:

1. **HELP** as the only parameter prints a description of the runtime parameters.
2. The entered values for the following keywords are not translated to uppercase: KEY1, FROMKEY1, TOKEY1, KEY2, FROMKEY2, and TOKEY2.

Required Parameters

FUNCTION
Queue Management Function:

DELete	Create a listing of the messages in the queue specified for the QUEUE parameter, print the content of these messages to the sequential data set DSLSDSY, and delete them from the queue.
KEEP	Print the messages to the sequential data set DSLSDSY, and do not delete them from the queue.
LIST	Create a listing of the messages in the queue specified for the QUEUE parameter, and do not delete them from the queue.

This parameter is required.

MSGFORMAT
Message format. Specifies the format code used to format the messages.

With COMPRESSION 0 .. 3 (PROMPT):

_	Underscore. The first system printer device format in the MCB is used.
E	English.

With COMPRESSION 4 (NOPROMPT):

P	Telex (new format)
S	SWIFT I
W	SWIFT II
X	SWIFT I with address expansion
Y	SWIFT II with address expansion
0	MERVA base format.

This parameter is required when FUNCTION = DELETE or KEEP. With FUNCTION = LIST this parameter is not required (and not used).

QUEUE
Queue name. The messages of this queue will be processed.

Optional Parameters

With the following parameters you can select the queue elements to be processed. If not specified, all are processed:

FROMQSN
From QSN. Only messages with a QSN greater than or equal to this QSN will be processed.

This parameter is optional, the default value used is FROMQSN = 0. See also TOQSN.

TOQSN
To QSN. Only messages with a QSN less than or equal to this QSN will be processed.

This parameter is optional, the default value used is that all messages are processed. See also FROMQSN.

KEY1
Key 1. Only messages with a key 1 matching the specified value will be processed. The specified string may contain the wildcards '%' and '*'.

Note: You can specify either KEY1 or FROMKEY1 .. TOKEY1, not both.

FROMKEY1
From key 1. Only messages with a key 1 greater than or equal to the specified value will be processed. The specified string must not contain the wildcards '%' and '*'. See also TOKEY1 and the note at KEY1.

TOKEY1
To key 1. Only messages with a key 1 less than or equal to the specified value will be processed. The specified string must not contain the wildcards '%' and '*'. See also FROMKEY1 and the note at KEY1.

KEY2
Key 2. Only messages with a key 2 matching the specified value will be processed. The specified string may contain the wildcards '%' and '*'.

Note: You can specify either KEY2 or FROMKEY2 .. TOKEY2, not both.

FROMKEY2
From key 2. Only messages with a key 2 greater than or equal to the specified value will be processed. The specified string must not contain the wildcards '%' and '*'. See also TOKEY2 and the note at KEY2.

TOKEY2
To key 2. Only messages with a key 2 less than or equal to the specified value will be processed. The specified string must not contain the wildcards '%' and '*'. See also FROMKEY2 and the note at KEY2.

FROMUMR
From UMR. Only messages with an UMR greater than or equal to this UMR will be processed.

This parameter is optional, the default value used is FROMUMR = 0. See also TOUMR.

TOUMR
To UMR. Only messages with an UMR less than or equal to this UMR will be processed.

This parameter is optional, the default value used is that all messages are processed. See also FROMUMR.

FLDNAME1
MERVA field name:

MSGDST	Specifies that only messages with the SWIFT master destination specified in parm FLDVALUE1 should be processed.
MSGNET	Specifies that only messages with the message type specified in parm FLDVALUE1 should be processed.
SWBHLT	Specifies that only messages with the SWIFT basic header logical terminal address specified in parm FLDVALUE1 should be processed.

This parameter is optional.

FLDVALUE1
MERVA field value:

- If parameter FLDNAME1 = MSGDST, an up to 9 character SWIFT master destination.
- If parameter FLDNAME1 = MSGNET, an up to 8 character MERVA message type. Should start with 'S', for example, S100 or S1*.
- If parameter FLDNAME1 = SWBHLT, an up to 12 character SWIFT basic header logical terminal address.

The wildcard '*' is allowed as last (or only) character.

This parameter is required when FLDNAME1 = MSGDST, MSGNET, or SWBHLT.

Other Parameters

With the following parameters you can specify how detailed the information given is and the print format:

BOTTOMFRAME
MCB name. Specifies the bottom frame MCB printed on each page (TUCFRAMB):

NONE	No bottom frame is used.
0BOT	The standard MERVA bottom frame is used: 2 empty lines at the bottom of each page.

This parameter is optional. If omitted, no bottom frame is used.

CASE
Print asis or uppercase:

ASIS	Print the messages as they are.
UPPER	Convert lowercase characters to uppercase (a .. z → A .. Z).

This parameter is optional. If omitted, the default value used is ASIS.

COMPRESSION
Compression format. The compression format used for printing (TUCBCOMP):

0	No compression. Empty fields and blank lines are printed.
1	Field compression. Only fields with data are printed (PROMPT UNIT).
2	Line compression. Empty data areas are not printed (PROMPT LINE).
3	Field and line compression. Only fields with data are printed (PROMPT UNIT LINE).
4	The message is printed in NOPROMPT format.

This parameter is optional. If omitted, the default value used is 0. Refer to the *MERVA for ESA Macro Reference* for a detailed description of the PRFORM parameter of the DSLFNT macro.

LINESPP
Lines per page. Specifies the number of lines printed on one page (TUCBROWN).

This parameter is optional. If omitted, the default value used is 55.

LOGLEVEL
Log level:

1 Only overview data is shown in the listing.

2 Detailed data for each message is shown.

3 UMR, Key 2, MSGDST, and SWBHLT are shown even if not specified as selection criterion.

4 Should be used in case of problems only.

This parameter is optional. If omitted, the default value used is 2.

MCB
MCB name. The MCB name used for printing the message. Usually it contains the message identification or the name of the cover MCB (TUCMSGID).

This parameter is optional. If omitted, the cover MCB 0COV is used. It is recommended to use this parameter only in very special cases.

PERFORM
Performance information. Indicates whether performance information about the API commands DELE, GET, PRTI, PRTL, and PRTT and the MVS or VSE EXECIO command should be gathered. If specified, the keyword value must be Yes or No.

This parameter is optional. If omitted, the default value used is NO.

SEPARATOR
Separator page. Indicates whether a separator page with a running number, the queue name, QSN, date and time of print should be printed ahead of each message.

This parameter is optional. If omitted, the default value used is NO.

SEPLINESPP
Lines per page. Specifies the number of lines printed on separator pages.

This parameter is optional. If omitted, the separator page is printed with LINESPP lines per page.

TOPFRAME
MCB name. Specifies the top frame MCB printed on each page (TUCFRAMT):

NONE
No top frame is used.

0TOP The standard MERVA top frame is used.

This parameter is optional. If omitted, no top frame is used.

With the following parameter you can pass a user-specific parameter to DSLSDYR:

USERPARM
User-specific parameter. The specified data is accepted by DSLSDYR and available in the variable *parm_userparm*.

This parameter is optional.

Required Parameters for VSE

VSEBLKSIZE
DSLSDSY BLKSIZE. BLKSIZE of output SAM file DSLSDSY, maximum 32761.

This parameter is required under VSE.

VSERECFORM
DSLSDSY RECFORM. RECFORM of output SAM file DSLSDSY: FIXUNB, FIXBLK, VARUNB, or VARBLK.

This parameter is required under VSE.

VSERECSIZE
DSLSDSY RECSIZE. RECSIZE of output SAM file DSLSDSY, maximum 32761.

This parameter is required under VSE, when VSERECFORM = FIXUNB or FIXBLK.

Customization

The following DSLPRM parameters affect DSLSDYR:

PRTNAME Your institution name as it is to appear in the printout of (most) REXX batch utilities.

SDDB2 When this parameter is set to YES, direct queue management is enabled (DB2 MVS only).

In the customization section of DSLSDYR you can set the following installation-specific variable:

valid_msgformat With FUNCTION DELETE and KEEP the entered value for the runtime parameter MSGFORMAT is checked against the variable *valid_msgformat*.

You can use the following routine in DSLSDYR to reject entered runtime parameters:

- USEREXIT_Q1 can be used to reject the entered value for runtime parameter QUEUE, queue to be printed.

Sample Printout

The following figure shows the information printed after the execution of the DSLSDYR program.

```
MERVA ESA V4.1 DSLSDYR                      13. Jan. 1999  16:59:43
(C) Copyright IBM Corp. 1999

+ ---------------------------------------------------------- +
|        S A M P L E   B A N K   B o e b l i n g e n        |
+ ---------------------------------------------------------- +


DDDDD    SSSSSS   LLL      SSSSSS   DDDDD    YY  YY   RRRRR
DDDDDD   SSSSSS   LLL      SSSSSS   DDDDDD   YY  YY   RRRRRR
DD  DD   SS       LLL      SS       DD  DD   YY  YY   RR  RR
DD  DD   SS       LLL      SS       DD  DD   YY  YY   RR  RR
DD  DD   SSSSSS   LLL      SSSSSS   DD  DD    YYYY    RRRRR
DD  DD   SSSSSS   LLL      SSSSSS   DD  DD    YYYY    RRRRRR
DD  DD       SS   LLL          SS   DD  DD     YY     RR  RR
DD  DD       SS   LLL          SS   DD  DD     YY     RR  RR
DDDDDD   SSSSSS   LLLLLL   SSSSSS   DDDDDD     YY     RR  RR
DDDDD    SSSSSS   LLLLLL   SSSSSS   DDDDD      YY     RR  RR


REXX version of DSLSDY - Sequential Data Set Print.
HELP as the only parameter prints a description.


DSLSDYR_001I : DSLSDYR started by user HEG at 13. Jan. 1999 16:59:43

DSLSDYR_003I : Runtime parameters:
                1. FUNCTION    - Queue mgmt function : KEEP
                2. MSGFORMAT   - Message format ID   : _ (blank)
                3. QUEUE       - Queue name ........ : L2DE0
                4. BOTTOMFRAME - Bottom frame MCB .. : 0BOT
                5. CASE        - ASIS or UPPER ..... : ASIS
                6. COMPRESSION - Compression format  : 1
                7. FLDNAME1    - MERVA field name .. :
                8. FLDVALUE1   - MERVA field value   :
                9. FROMKEY1    - From KEY1 ......... :
               10. FROMKEY2    - From KEY2 ......... :
               11. FROMQSN     - From QSN .......... :
               12. FROMUMR     - From UMR .......... :
               13. KEY1        - KEY1 .............. :
               14. KEY2        - KEY2 .............. :
               15. LINESPP     - Lines per page .... : 60
               16. LOGLEVEL    - Log level ......... : 3
               17. MCB         - MCB name for print  :
               18. PERFORM     - Performance info .. : YES
               19. SEPARATOR   - Separator page  ... : YES
               20. SEPLINESPP  - Lines per sep. page : 20
               21. TOKEY1      - To KEY1 ........... :
               22. TOKEY2      - To KEY2 ........... :
               23. TOPFRAME    - Top frame MCB ..... : 0TOP
               24. TOQSN       - To QSN ............ :
               25. TOUMR       - To UMR ............ :
               26. USERPARM    - User specific data  : MYDATA
```

Figure 90. DSLSDYR (Sequential Data Set Print) Sample Printout (Part 1 of 3)

```
DSLSDYR_005I : MERVA ID is MHEG, MERVA name is MERVAESA.

DSLSDYR_006I : Queue management module ..... : DSLQMCNV
               Access method / features used : VXBD

DSLSDYR_008I : There are 14 queue elements matching the specified
               From/To QSN, (From/To) KEY1, and (From/To) KEY2.

DSLSDYR_010I : The output data set used is HEG.DSLSDY.FILE.
               Note: This info was obtained by using internal control
                     blocks.

DSLSDYR_021I : Detailed statistical data

               Note: Running no. 2, UMR, Key 2, MSGDST, SWBHLT, and no.
                     of lines printed are shown in the second line, when
                     specified as selection criterion, or log level >= 3,
                     or running no. 1 and 2 are different.

               E  M                                                        B
               R  A                                                        U
               O  C                                                        S  GET  PRTL SDY  DELE
               R  H  Number  Input QSN   Key 1                   MSGNET   Y   rc   rc   rc   rc
               -  -  ------  ----------  ----------------------- --------  -   --   --   --   --
                     Number    UMR no.   Key 2                   MSGDST     SWBHLT        Lines
                     ------    --------  ----------------------- ---------  ------------  -------

                          1  0000000904  100                     S100          ok   ok   ok
                               00002559                          VNDEBET2A                    96

                          2  0000000905  105                     S105          ok   ok   ok
                               00002560                          VNDEBET2A                    70

                          3  0000000906  106                     S106          ok   ok   ok
                               00002561                          VNDEBET2A                    70

                          4  0000000907  110                     S110          ok   ok   ok
                               00002562                          VNDEBET2A                    85

                          5  0000000908  111                     S111          ok   ok   ok
                               00002563                          VNDEBET2A                    54

                          6  0000000909  112                     S112          ok   ok   ok
                               00002564                          VNDEBET2A                    54

                          7  0000000910  190                     S190          ok   ok   ok
                               00002565                          VNDEBET2A                    54

                          8  0000000911  191                     S191          ok   ok   ok
                               00002566                          VNDEBET2A                    57

                          9  0000000912  192                     S192          ok   ok   ok
                               00002567                          VNDEBET2A                    94

                         10  0000000913  195                     S195          ok   ok   ok
                               00002568                          VNDEBET2A                   111

                         11  0000000914  196                     S196          ok   ok   ok
                               00002569                          VNDEBET2A                   111
```

⋮

Figure 90. DSLSDYR (Sequential Data Set Print) Sample Printout (Part 2 of 3)

```
DSLSDYR_022I : Performance information

               PRTI time: 0.00263
               PRTT time: 0.00120

                                    GET        PRTL      EXECIO     DELE
               Number  QSN           time       time       time      time
               ------  ----------  ------    --------    ------    ------
                    1  0000000904  0.0468      1.2974    0.0949
                    2  0000000905  0.0077      0.1657    0.0130
                    3  0000000906  0.0390      0.1896    0.0116
                    4  0000000907  0.0436      0.2157    0.0242
                    5  0000000908  0.0115      0.1435    0.0100
                    6  0000000909  0.0127      0.2192    0.0106
                    7  0000000910  0.0186      0.1548    0.0108
                    8  0000000911  0.0127      0.1440    0.0107
                    9  0000000912  0.0049      0.2779    0.0143
                   10  0000000913  0.0132      0.2736    0.0186
                   11  0000000914  0.0088      0.3186    0.0174
                   12  0000000915  0.0560      2.6242    0.2115
                   13  0000000916  0.0136      0.1847    0.0154
                   14  0000000917  0.0172      0.4822    0.0108

               Sum:                0.3063      6.6911    0.4738

DSLSDYR_023I : Overview statistical data

               No. of elements matching QSN and KEY : 14

               No. of GETs with intrc ' ' ......... : 14
               No. of GETs with intrc 01 .......... : 0
               No. of GETs with intrc 02 .......... : 0
               No. of GETs with rc <= -2 .......... : 0

               PRTI ended with intrc / rc ......... : ok
               No. of PRTLs with intrc ' ' ........ : 2172
               No. of PRTLs with intrc 01 ......... : 0
               No. of PRTLs with rc <= -2 ......... : 0
               PRTT ended with intrc / rc ......... : ok

               No. of messages written to data set  : 14
               No. of records written to data set   : 2452
               No. of msg's write to data set failed: 0

DSLSDYR_024I : Number of (matching) input messages ....... : 14
               Number of messages written to DSLSDSY ..... : 14
               Number of records  written to DSLSDSY ..... : 2452

DSLSDYR_019I : DSLSDYR ended with return code 0 - successful.

DSLSDYR_025I : DSLSDYR ended at 13. Jan. 1999 16:59:57
```

Figure 90. DSLSDYR (Sequential Data Set Print) Sample Printout (Part 3 of 3)

Listing Fields

The 'Detailed statistical data' of the listing contains the following information:

Processing information for each message

First line:

ERROR	'>' indicates an error with the message, for example, the message could not be printed to the output data set, or could not be deleted.
MATCH	'N' indicates that this message does not match a specified FROMUMR, TOUMR, MSGDST, MSGNET, or SWBHLT value.
Number	Running number 1, all messages matching a specified QSN and KEY.
Input QSN	QSN of the message in the input queue.
Key 1	Key-1 value.
MSGNET	Message type.
BUSY	'X' indicates that this message is currently BUSY.
GET rc	Return code of API function GET, read a message from the input queue.
PRTL rc	Return code of API function PRTL, print a message line.
SDY rc	Return code of write with EXECIO to DSLSDSY file.
DELE rc	Return code of API function DELE, delete a message from the input queue.

Second line:

Number	Running number 2, actually printed messages. May be different from running number 1, when runtime parameter UMR and/or FLDNAME1 are specified.
UMR no.	UMR sequence number of the message in the input queue.
Key 2	Key-2 value.
MSGDST	SWIFT master destination.
SWBHLT	SWIFT basic header logical terminal address.
Lines	Number of printed lines.

Performance information:

PRTI time	Time in seconds spent in API function PRTI.
PRTT time	Time in seconds spent in API function PRTT.
Number	Running number 1, all messages matching a specified QSN and KEY.
Input QSN	QSN of the message in the input queue.
GET time	Time in seconds spent in API function GET.
PRTL time	Time in seconds spent in API function PRTL.
EXECIO time	Time in seconds spent in TSO or VSE EXECIO.

DELE time Time in seconds spent in API function DELE.

Notes:

1. Running no. 2, UMR, Key 2, MSGDST, SWBHLT, and no. of lines printed are shown in the second line, when specified as selection criterion, or log level ≥ 3, or running no. 1 and 2 are different.
2. `rc <= -2` indicates an error in DSLSDYR or the REXX host command environment. Refer to the *MERVA for ESA Application Programming Interface Guide* for a description of the return codes.

The 'Number of records written to DSLSDSY' includes the separator pages.

Messages and Codes

DSLSDYR ends with the return code:

0 Successful

4 Warning

8 Error

12 Severe error.

The messages have the following structure:

DSLSDYR_nnnl Message text

Where:

DSLSDYR Identifies the message as a message of the DSLSDYR utility.

_ Underscore.

nnn Is a 3-digit number used to identify the message.

l Is a single character used to show the type of message. The following characters are used:

- **I** Information message
- **W** Warning message
- **E** Error message
- **S** Severe error message.

The messages are not further documented. Note that the message numbers are subject to change at any time.

Part 3. Maintaining MERVA ESA

This part describes the maintenance of the MERVA ESA files and data sets using utility programs.

For information on online maintenance of the MERVA ESA files, refer to the *MERVA for ESA User's Guide*.

Chapter 19. Using the Queue Data Set Utility DSLQDSUT

The MERVA ESA queue data set and the large message cluster contain the messages of the message-processing-function queues. The queue data set utility DSLQDSUT processes the queue data set and the large message cluster. It provides the following functions:

1. FORMAT (to format the queue data set)
   - To format the space reserved for the queue data set during MERVA ESA generation and on other occasions
   - To set the last unique message reference (UMR).

   Before you format the queue data set, you should ensure that it does not contain messages that the users want to keep, as all messages are deleted when the queue data set is formatted.
2. FORMATL (to format the queue data set and reset the large message cluster)

   In addition to the functions of FORMAT, FORMATL resets the large message cluster, so that it can be reused with the formatted queue data set.
3. COPY (to copy a queue data set)
   - To create a second queue data set for the duplicate queue data set feature of MERVA ESA
   - To create a backup copy.

   The output queue data set must have the same space reservation as the input data set. If the output queue data set is smaller, the COPY function cannot be performed successfully. If the output queue data set is larger, only as much space is used as in the input queue data set, and the rest of the space cannot be accessed.
4. MODIFY (to modify a queue data set)

   Use the MODIFY function:
   - When the old queue data set is full and you want to allocate a larger one, keeping the messages contained in the old queue data set
   - When the old queue data set is too big and you want to allocate a smaller one, keeping the messages contained in the old queue data set, and reducing the number of empty blocks.

   Use the MODIFY function with the EXCLUDE FNT control statement:
   - When you have removed functions from the MERVA ESA function table that still contain messages, and you do not want to copy these messages from the old queue data set to the new queue data set.

   Use the MODIFY function with the EXCLUDE LMC control statement:
   - When the old queue data set contains references to large messages which are not in the large message cluster anymore.

   Use the MODIFY function with the EXCLUDE rbn control statement(s):
   - When the old queue data set has one or more corrupted blocks, and you want to keep as many messages as possible.

   Use the MODIFY function with the REPAIR control statement:

- When the old queue data set has one or more corrupted queue element prefixes.

Use the MODIFY function with the LASTUMR control statement:

- To adjust the last unique message reference (UMR)

The MODIFY function copies messages from the old queue data set (input) to the new queue data set (output). Messages in corrupted blocks are copied if possible. Messages can be excluded if the function they belong to is not in the MERVA ESA function table anymore or if large messages are referenced in the QDS, but their data is not in the LMC. Corrupted blocks can be excluded completely if the automatic recovery of DSLQDSUT cannot handle them. When the input queue data set has been processed completely, the rest of the output queue data set is formatted as in the FORMAT function.

The output queue data set can have a space reservation different from the input data set.

If the output queue data set is smaller than the input queue data set, the MODIFY function can be performed successfully only if all messages (except the ones excluded by EXCLUDE control statements) can be copied into the output queue data set.

When MERVA ESA is started with a queue data set created by the MODIFY function, a DSLQMGT restart is performed to create a new Queue Key Table. This restart can be shorter than a usual restart as DSLQDSUT indicates to DSLQMGT which queue data set blocks contain messages.

The MERVA ESA administrator at your installation will inform you when the queue data set utility should be run.

DSLQDSUT must not be used when MERVA ESA is running. Under CICS, the queue data set utility can be executed while CICS is running, but MERVA ESA must be stopped before the queue data set utility is started.

If your installation uses the unique message reference (UMR) option, you can use the LASTUMR control statement to set or adjust the last assigned UMR. LASTUMR is valid only with the FORMAT and MODIFY functions, and it must be the first of the control statements.

When LASTUMR is used, the MERVA ESA identifier is taken from the module DSLPRM. The UMR number is taken from the control statement, and the date and time from the current system date and time.

When LASTUMR is not used, the UMR is set to zero for FORMAT, and is taken from the input data set for MODIFY.

REPAIR is valid only with the MODIFY function. It must be the first of the control statements or, if the LASTUMR control statement is also used, the second one.

If your installation uses the duplicate queue data set feature, each queue data set must be formatted in a separate job step, or you must get an exact copy of the first queue data set using the COPY function of DSLQDSUT. This function can also be performed using the REPRO function of the VSAM Access Method Services (IDCAMS).

Job Control Statements for DSLQDSUT under MVS

FORMAT

Figure 91 shows the JCL to format the queue data set with DSLQDSUT under MVS.

```
//.......  JOB ........
//JOBCAT   DD DSN=ucat,DISP=SHR                VSAM USER CAT
//QFMT     EXEC PGM=DSLQDSUT,PARM='FORMAT'
//STEPLIB  DD DSN=loadlib,DISP=SHR
//SYSUDUMP DD SYSOUT=A
//DSLSNAP  DD SYSOUT=A
//DSLQDSO  DD DSN=qdsfile,DISP=SHR       OUTPUT QUEUE DATA SET
//DSLIN    DD *            CONTROL STATEMENT DATA SET
  LASTUMR  100
```

Figure 91. Formatting a Queue Data Set in MVS

In the JCL, the lowercase parameters have the following meanings:

ucat The VSAM user catalog where the queue data set is cataloged. In some MVS installations, this statement may not be required.

loadlib The name of the load library containing the MERVA ESA programs.

qdsfile The name of the queue data set.

During execution of the DSLQDSUT program in MVS, the following message appears on the operating system console:

```
IEC070I 104-203 ....
```

This message is caused by the formatting technique of DSLQDSUT, which uses all available blocks defined in the VSAM cluster definition for the queue data set. The message can be ignored. The MERVA ESA operator messages DSL601I and DSL602I show a successful queue data set formatting action. If the UMR option is on, or LASTUMR was specified, the message DSL621I shows the UMR status of the output queue data set.

FORMATL

Figure 92 shows the JCL to format the QDS and to reset the LMC with DSLQDSUT under MVS.

```
//.......  JOB ........
//JOBCAT   DD DSN=ucat,DISP=SHR                VSAM USER CAT
//QFMT     EXEC PGM=DSLQDSUT,PARM='FORMATL'
//STEPLIB  DD DSN=loadlib,DISP=SHR
//SYSUDUMP DD SYSOUT=A
//DSLSNAP  DD SYSOUT=A
//DSLQDSO  DD DSN=qdsfile,DISP=SHR       OUTPUT QUEUE DATA SET
//DSLQX01  DD DSN=lmcfile,DISP=SHR       LARGE MESSAGE CLUSTER
//DSLIN    DD *            CONTROL STATEMENT DATA SET
  LASTUMR  100
```

Figure 92. Formatting a Queue Data Set in MVS

In the JCL, the lowercase parameters have the following meanings:

ucat The VSAM user catalog where the queue data set is cataloged. In some MVS installations, this statement may not be required.

loadlib The name of the load library containing the MERVA ESA programs.

qdsfile The name of the queue data set.

lmcfile The name of the large message cluster.

During execution of the DSLQDSUT program in MVS, the following message appears on the operating system console:

```
IEC070I 104-203 ....
```

This message is caused by the formatting technique of DSLQDSUT, which uses all available blocks defined in the VSAM cluster definition for the queue data set. The message can be ignored. The MERVA ESA operator messages DSL601I, DSL602I, and DSL627I show a successful queue data set formatting action. If the UMR option is on, or LASTUMR was specified, the message DSL621I shows the UMR status of the output queue data set.

COPY

Figure 93 shows the JCL to copy a queue data set with DSLQDSUT under MVS.

```
//.......  JOB ........
//JOBCAT   DD DSN=ucat,DISP=SHR                VSAM USER CAT
//QFMT     EXEC PGM=DSLQDSUT,PARM='COPY'
//STEPLIB  DD DSN=loadlib,DISP=SHR
//SYSUDUMP DD SYSOUT=A
//DSLSNAP  DD SYSOUT=A
//DSLQDSI  DD DSN=qdsfile,DISP=SHR       INPUT  QUEUE DATA SET
//DSLQDSO  DD DSN=qdsfile,DISP=SHR       OUTPUT QUEUE DATA SET
```

Figure 93. Copying a Queue Data Set in MVS

In the JCL, the lowercase parameters have the following meanings:

ucat The VSAM user catalog where the queue data set is cataloged. In some MVS installations, this statement may not be required.

loadlib The name of the load library containing the MERVA ESA programs.

qdsfile The name of the queue data set. For the DSLQDSI DD statement, this is the name of the input queue data set. For the DSLQDSO DD statement, this is the name of the output queue data set.

The MERVA ESA operator messages DSL601I and DSL602I show that a queue data set has been successfully copied. If the data sets contain a UMR, it is displayed in message DSL621I.

MODIFY

Figure 94 on page 279 shows the JCL to modify a queue data set with DSLQDSUT under MVS.

```
//.......  JOB ........
//JOBCAT   DD DSN=ucat,DISP=SHR                  VSAM USER CAT
//QFMT     EXEC PGM=DSLQDSUT,PARM='MODIFY'
//STEPLIB  DD DSN=loadlib,DISP=SHR
//SYSUDUMP DD SYSOUT=A
//DSLSNAP  DD SYSOUT=A
//DSLQDSI  DD DSN=qdsfile,DISP=SHR       INPUT  QUEUE DATA SET
//DSLQDSO  DD DSN=qdsfile,DISP=SHR       OUTPUT QUEUE DATA SET
//DSLQX01  DD DSN=lmcfile,DISP=SHR       LARGE MESSAGE CLUSTER
//DSLIN    DD *                             CONTROL STATEMENTS
   LASTUMR  0
   REPAIR
   EXCLUDE  FNT
   EXCLUDE  LMC
   EXCLUDE  50
//
```

Figure 94. Modifying a Queue Data Set in MVS

In the JCL, the lowercase parameters have the following meanings:

ucat The VSAM user catalog where the queue data set is cataloged. In some MVS installations, this statement may not be required.

loadlib The name of the load library containing the MERVA ESA programs.

qdsfile The name of the queue data set. For the DSLQDSI DD statement, this is the name of the input queue data set, for the DSLQDSO DD statement, this is the name of the output queue data set.

lmcfile The name of the large message cluster.

The control statement data set DSLIN can be omitted or empty. If used and not empty, it must contain fixed length records of 80 bytes with the following contents:

- The keyword LASTUMR starting in the first byte or after any number of blanks, and followed by at least one blank.
- A number indicating the last UMR to be used. At least one blank must follow.
- Any character following the blank after the UMR is treated as comment.
- The keyword REPAIR starting in the first byte or after any number of blanks, and followed by at least one blank.
- The keyword EXCLUDE starting in the first byte or after any number of blanks, and followed by at least one blank.
- One of the following:
  - The characters FNT to indicate to exclude messages that belong to functions that are not contained in the MERVA ESA function table anymore.
  - The characters LMC to indicate to exclude message references to large messages from the queue data set which are not in the large message cluster.
  - A number indicating the relative block number (*rbn*) of the input queue data set (the relative record number in the VSAM RRDS) to be excluded when copying blocks from the input queue data set to the output queue data set. The relative block number must be followed by at least one blank, unless it ends in byte 80 of the control statement.

    EXCLUDE *rbn* control statements are only needed if the automatic recovery of corrupted blocks failed, that is, if DSLQMGT has indicated a corrupted block, DSLQDSUT should first be run without EXCLUDE *rbn* control statements.

- Any character following the blank after the characters FNT or LMC or the relative block number is treated as comment.

The following rules apply to the processing of the LASTUMR control statement:

- LASTUMR must be the first control statement.
- LASTUMR statements are ignored if the number is invalid or the control statement is out of sequence.

The following rules apply to the processing of the REPAIR control statement:

- REPAIR must be the first or, if LASTUMR is also used, the second control statement.
- REPAIR statements are ignored if they are out of sequence.

The following rules apply to the processing of the EXCLUDE control statements:

- The EXCLUDE FNT control statement must be before the first EXCLUDE *rbn* control statement.
- The EXCLUDE LMC control statement must be before the first EXCLUDE *rbn* control statement.
- The EXCLUDE *rbn* control statements must be in ascending order of the relative block number in DSLIN. Control statements out of sequence cannot be processed.
- Control statements that contain only blanks are ignored.
- Control statements referring to a system block of the input queue data set are ignored.
- EXCLUDE control statements are ignored if the relative block number is not contained in the input queue data set.
- You are informed by a message when a control statement excludes a block that contains data.
- You are not informed when a control statement excludes an empty block.

During execution of the DSLQDSUT program in MVS, the following message appears on the operating system console:

```
IEC070I 104-203 ....
```

This message is caused by the formatting technique of DSLQDSUT that uses all available blocks defined in the VSAM cluster definition for the queue data set. The message can be ignored. The MERVA ESA operator messages DSL601I and DSL602I show that a queue data set has been successfully modified. If LASTUMR was specified, or the input data set contained a UMR, the message DSL621I shows the UMR status of the output queue data set. The messages DSL622I, DSL623I, and DSL625I give information about the number of messages copied and excluded.

Job Control Statements for DSLQDSUT under VSE

FORMAT

Figure 95 on page 281 shows the JCL to format the queue data set with DSLQDSUT under VSE.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM                    VSAM USER CAT
// DLBL DSLQDSO,'qdsfile',,VSAM               OUTPUT QUEUE DATA SET
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION DUMP
// EXEC DSLQDSUT,SIZE=300K,PARM='FORMAT'
   LASTUMR 123                               CONTROL STATEMENT
/*
/&
```

Figure 95. Formatting a Queue Data Set in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

qdsfile The name of the queue data set.

program library
The name of the library containing the MERVA ESA product.

library.sublib The name of the sublibrary containing the MERVA ESA programs and JCL procedures. You can code a list of sublibrary names.

The MERVA ESA operator messages DSL601I and DSL602I show that a queue data set has been successfully formatted. If the UMR option is on, or LASTUMR was specified, the message DSL621I shows the UMR status of the output queue data set.

FORMATL

Figure 96 shows the JCL to format the QDS and to reset the LMC with DSLQDSUT under VSE.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM                    VSAM USER CAT
// DLBL DSLQDSO,'qdsfile',,VSAM               OUTPUT QUEUE DATA SET
// DLBL DSLQX01,'lmcfile',,VSAM               LARGE MESSAGE CLUSTER
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION DUMP
// EXEC DSLQDSUT,SIZE=300K,PARM='FORMATL'
   LASTUMR 123                               CONTROL STATEMENT
/*
/&
```

Figure 96. Formatting a Queue Data Set in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

qdsfile The name of the queue data set.

lmcfile The name of the large message cluster.

program library
The name of the library containing the MERVA ESA product.

library.sublib The name of the sublibrary containing the MERVA ESA programs and JCL procedures. You can code a list of sublibrary names.

The MERVA ESA operator messages DSL601I, DSL602I, and DSL627I show a successful queue data set formatting action. If the UMR option is on, or LASTUMR was specified, the message DSL621I shows the UMR status of the output queue data set.

COPY

Figure 97 shows the JCL to copy a queue data set with DSLQDSUT under VSE.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM                    VSAM USER CAT
// DLBL DSLQDSI,'qdsfile',,VSAM               INPUT  QUEUE DATA SET
// DLBL DSLQDSO,'qdsfile',,VSAM               OUTPUT QUEUE DATA SET
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION DUMP
// EXEC DSLQDSUT,SIZE=300K,PARM='COPY'
/*
/&
```

Figure 97. Copying a Queue Data Set in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

qdsfile The name of the queue data set. For the DSLQDSI DLBL statement, this is the name of the input queue data set; for the DSLQDSO DLBL statement, this is the name of the output queue data set.

program library
The name of the library containing the MERVA ESA product.

library.sublib The name of the sublibrary containing the MERVA ESA programs and JCL procedures. You can code a list of sublibrary names.

The MERVA ESA operator messages DSL601I and DSL602I show that a queue data set has been successfully copied. If the data sets contain a UMR, it is displayed in the message DSL621I.

MODIFY

Figure 98 on page 283 shows the JCL to modify a queue data set with DSLQDSUT under VSE.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM                    VSAM USER CAT
// DLBL DSLQDSI,'qdsfile',,VSAM                 INPUT  QUEUE DATA SET
// DLBL DSLQDSO,'qdsfile',,VSAM                 OUTPUT QUEUE DATA SET
// DLBL DSLQX01,'lmcfile',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION DUMP
// EXEC DSLQDSUT,SIZE=300K,PARM='MODIFY'
   LASTUMR   123                                CONTROL STATEMENT
   REPAIR
   EXCLUDE   FNT                                CONTROL STATEMENT
   EXCLUDE   LMC                                CONTROL STATEMENT
   EXCLUDE    50                                CONTROL STATEMENT
/*
/&
```

Figure 98. Modifying a Queue Data Set in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

qdsfile The name of the queue data set. For the DSLQDSI DLBL statement, this is the name of the input queue data set. For the DSLQDSO DLBL statement, this is the name of the output queue data set.

lmcfile The name of the large message cluster.

program library
The name of the library containing the MERVA ESA product.

library.sublib The name of the sublibrary containing the MERVA ESA programs and JCL procedures. You can code a list of sublibrary names.

The control statement data set SYSIPT can be empty. If used and not empty, it must contain fixed length records of 80 bytes with the following contents:

- The keyword LASTUMR starting in the first byte or after any number of blanks, and followed by at least one blank.
- A number indicating the last UMR to be used. At least one blank must follow.
- Any character following the blank after the UMR is treated as comment.
- The keyword REPAIR starting in the first byte or after any number of blanks, and followed by at least one blank.
- The keyword EXCLUDE starting in the first byte or after any number of blanks, and followed by at least one blank.
- One of the following:
  - The characters FNT to indicate to exclude messages that belong to functions that are not contained in the MERVA ESA function table anymore.
  - The characters LMC to indicate to exclude message references to large messages from the queue data set which are not in the large message cluster.
  - A number indicating the relative block number (*rbn*) of the input queue data set (the relative record number in the VSAM RRDS) to be excluded when copying blocks from the input queue data set to the output queue data set. The relative block number must be followed by at least one blank, unless it ends in byte 80 of the control statement.

EXCLUDE *rbn* control statements are only needed if the automatic recovery of corrupted blocks failed, that is, if DSLQMGT has indicated a corrupted block, DSLQDSUT should first be run without EXCLUDE *rbn* control statements.

- Any character following the blank after the characters FNT or the relative block number is treated as comment.

The following rules apply to the processing of the LASTUMR control statement:

- LASTUMR must be the first control statement.
- LASTUMR statements are ignored if the number is invalid or the control statement is out of sequence.

The following rules apply to the processing of the REPAIR control statement:

- REPAIR must be the first or, if LASTUMR is also used, the second control statement.
- REPAIR statements are ignored if they are out of sequence.

The following rules apply to the processing of the EXCLUDE control statements:

- The EXCLUDE FNT control statement must be before the first EXCLUDE *rbn* control statement.
- The EXCLUDE LMC control statement must be before the first EXCLUDE *rbn* control statement.
- The EXCLUDE *rbn* control statements must be in ascending order of the relative block number in SYSIPT. Control statements out of sequence cannot be processed.
- Control statements containing only blanks are ignored.
- Control statements referring to a system block of the input queue data set are ignored.
- EXCLUDE control statements are ignored if the relative block number is not contained in the input queue data set.
- You are informed by a message when a control statement excludes a block that contains data.
- You are not informed when a control statement excludes an empty block.

The MERVA ESA operator messages DSL601I and DSL602I show that a queue data set has been successfully modified. If LASTUMR was specified, or the input data set contained a UMR, the message DSL621I shows the UMR status of the output queue data set. The messages DSL622I, DSL623I, and DSL625I give information about the number of messages copied and excluded.

Chapter 20. Using the Large Message Cluster Maintenance Utility DSLQMNT

The large message cluster maintenance utility DSLQMNT is used for reorganizing the large message cluster (LMC). All records referenced by the MERVA ESA queue data set (QDS) are copied from the old LMC into the new LMC in ascending key sequence. LMC reorganization is necessary to:

- Free unused space in the LMC. After the next startup of MERVA ESA, the processing of the LMC is set to VSAM 'load mode' for optimum performance.
- Synchronize the LMC with the QDS, that is, do not copy large message data that is not referenced in the QDS. This makes the online information provided by the commands **dlmc** and **dlmct** more accurate.
- Get status and statistics information from the LMC.
- Get an overview of messages in QDS but not in LMC.

There are two ways to reorganize the LMC:

- Define a new LMC, run DSLQMNT, and start MERVA ESA with the new LMC. In order to do this you must terminate MERVA ESA, terminate CICS, run DSLQMNT, start CICS, and start MERVA ESA. This method is recommended as the old LMC is still available in case of errors.
- Define a new LMC, run DSLQMNT, copy the new LMC into the old one, and start MERVA ESA with the old LMC. To do this, you must terminate MERVA ESA (CICS need not be terminated), run DSLQMNT, and start MERVA ESA.

  Note: This is only applicable for CICS, there is no need to terminate IMS.

Job Control Statements under MVS

The job control statements described here are used to reorganize and copy the large message cluster under MVS.

Reorganization of the Large Message Cluster

Figure 99 shows the JCL to run the DSLQMNT utility under MVS. The new reorganized LMC is then used in the next startup of MERVA ESA. CICS must be restarted to use the new LMC.

```
//........ JOB ........
//LMCREORG EXEC PGM=DSLQMNT,REGION=4096K
//STEPLIB  DD  DSN=loadlib,DISP=SHR
//SYSUDUMP DD  SYSOUT=A
//DSLSNAP  DD  SYSOUT=A
//DSLPRINT DD  SYSOUT=A
//DSLQDS   DD  DISP=SHR,DSN=qdsfile
//DSLQX01  DD  DISP=SHR,DSN=oldlmc
//DSLQX11  DD  DISP=SHR,DSN=newlmc
```

Figure 99. Reorganization of the Large Message Cluster under MVS

In the JCL, the lowercase parameters have the following meanings:

loadlib The name of the load library containing the MERVA ESA programs.

qdsfile The queue data set belonging with the old LMC.

oldlmc The old large message cluster with possibly the following types of large message data:

- Referenced and complete
- Unreferenced and complete
- Unreferenced and incomplete.

newlmc The new large message cluster receiving referenced and complete large messages during reorganization.

Copying the New LMC into the Old LMC

Figure 100 shows the JCL to run the VSAM Access Method Services for copying the new LMC to the old LMC under MVS. MERVA ESA is restarted with the old LMC. The REUSE parameter must have been specified in the VSAM cluster definition of the old LMC.

Note: CICS may remain running during the reorganization and the copying. The old LMC remains allocated to the CICS region, and will be used in the reorganized form when MERVA ESA is restarted.

```
//........ JOB ........
//LMCCOPY  EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=A
//NEW      DD  DISP=SHR,DSN=newlmc
//OLD      DD  DISP=SHR,DSN=oldlmc
//SYSIN    DD  *
            REPRO INFILE(NEW) -
                  REUSE       -
                  OUTFILE(OLD)
```

Figure 100. Copying the New LMC into the Old LMC under MVS

In the JCL, the lowercase parameters have the following meanings:

newlmc The new reorganized large message cluster.

oldlmc The old large message cluster now receiving the reorganized LMC.

Job Control Statements under VSE

This section describes the job control statements required to reorganize and copy the large message cluster under VSE.

Reorganization of the Large Message Cluster

Figure 101 on page 287 shows the JCL to run the DSLQMNT utility under VSE. The new reorganized LMC is then used in the next startup of MERVA ESA. CICS must be restarted to use the new LMC.

```
// JOB ........
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL DSLQDS,'qdsfile',,VSAM
// DLBL DSLQX01,'oldlmc',,VSAM
// DLBL DSLQX11,'newlmc',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ...)
// OPTION DUMP
// EXEC DSLQMNT,SIZE=300K
/*
/&
```

Figure 101. Reorganization of the Large Message Cluster under VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog for the large message cluster and the queue data set.

qdsfile The queue data set belonging with the old LMC.

oldlmc The old large message cluster with possibly the following types of large message data:

- Referenced and complete
- Unreferenced and complete
- Unreferenced and incomplete.

newlmc The new large message cluster receiving referenced and complete large messages during reorganization.

program library
The name of the library containing the MERVA ESA product.

volid The volume identification of the program library.

library.sublib The name of the sublibrary containing the MERVA ESA programs and JCL procedures. You can code a list of sublibrary names.

Copying the New LMC into the Old LMC

Figure 102 on page 288 shows the JCL to run the VSAM Access Method Services for copying the new LMC to the old LMC under VSE. MERVA ESA is restarted with the old LMC. The REUSE parameter must have been specified in the VSAM cluster definition of the old LMC.

Note: CICS may remain running during the reorganization and the copying. The old LMC remains allocated to the CICS region, and will be used in the reorganized form when MERVA ESA is restarted.

```
// JOB ........
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL NEW,'newlmc',,VSAM
// DLBL OLD,'oldlmc',,VSAM
// EXEC IDCAMS,SIZE=300K
           REPRO INFILE(NEW) -
                 REUSE       -
                 OUTFILE(OLD)
/*
/&
```

Figure 102. Copying the New LMC into the Old LMC under VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog for the large message cluster

newlmc The new reorganized large message cluster

oldlmc The old large message cluster now receiving the reorganized LMC.

Statistics Report from DSLQMNT

To enable you to understand the statistics report provided by DSLQMNT, a description of the processing status and the possible contents of a large message cluster follows.

The *LMC processing mode* is either *load* or *insert*.

Reorganization is recommended to regain optimum performance when the LMC is in insert mode.

The contents of a large message cluster, which belongs to a queue data set, consist of sets of records representing large messages.

A set consists of one record if the large message is not segmented or multiple records if the large message is segmented. A large message is segmented if there are more LMC records than large messages. In Figure 103 on page 289, for example, the number of LMC Records is 196, and the Number of Large Messages is 132. The message is therefore segmented. *MERVA for ESA Installation Guide* describes how you can minimize segmentation when processing large messages and how to optimize space usage with segmentation.

The following possible sets of records, representing large messages, can exist in a large message cluster:

- **Referenced by QDS and complete**

  Sets of records representing complete messages that are referenced in the QDS.

  Only these sets of records are used in the reorganization process, that is, the records are copied in ascending key sequence from the old LMC to the new LMC.
- **Unreferenced by QDS and complete**

  Sets of records representing complete messages that are *not* referenced in the QDS. This can result from:

  – Abnormal end in DSLQMGT while storing a message, as the data is first written to the LMC and then to the QDS.

  – Abnormal end in DSLQMGT while deleting a message, as the data is first deleted in the QDS and then in the LMC.

– The LMC does not belong to the QDS.

- **Unreferenced by QDS and incomplete**

  Sets of records representing incomplete messages. These are never referenced in the QDS as they result from an incomplete store or delete operation in the LMC, for example, because of an abnormal termination of MERVA ESA.

  These records are always discarded, because the associated large messages cannot be reassembled.

Sample Statistics Report for Old and New Large Message Cluster

Figure 103 shows the information printed after the execution of the DSLQMNT utility.

```
                    Statistics for Large Message Cluster
                    ------------------------------------

Status of LMCs                      Old LMC                  New LMC
--------------------               -----------              -----------

Date/Time from LMC              19990320/114949          19990320/130352
MERVA Termination                       normal                   normal
LMC closed            no  since 19990319/174333                      yes
Processing Mode                         insert                    load

LMC Statistics
--------------------

Allocated Disk Space                  7.372.800                7.372.800
Used Disk Space                       1.474.560                  737.280
Number of Extents                            10                        2
Maximum Rec Length                      114.000                  114.000
CI Size                                  12.288                   12.288

Stored Bytes

 Total                                  545.247                  523.178
 Referenced                             523.178                  523.178
 Unref - complete                        20.676                        0
 Unref - incomplete                       1.393                        0

Number of LMC Records

 Total                                      204                      196
 Referenced                                 196                      196
 Unref - complete                             7                        0
 Unref - incomplete                           1                        0
```

Figure 103. Printout of the DSLQMNT Statistics Report (Part 1 of 2)

```
Large Message Statistics
--------------------

Limit from DSLPRM                          1.000                     1.000

Length of Large Messages

 Average                                  12.101                    12.101
 Minimum                                   1.676                     1.676
 Maximum                                  13.394                    13.394

Number of Large Messages

 Total found in LMC                         139                       132
 Referenced                                 132                       132
 Unref - Complete                             7                         0
 Unref - Incomplete                           1                         0
 In QDS-not in LMC                        2.493                     2.493
------------------------------------------------------------------------------
               OVERVIEW -- Messages in QDS, but not in LMC
------------------------------------------------------------------------------

Function          QSN Queue-Key1               Queue-Key2
L2AI2               4 00304
UMR
Length               23.910
# of Segments            1
Key of 1st seg      453.049
------------------------------------------------------------------------------
```

Figure 103. Printout of the DSLQMNT Statistics Report (Part 2 of 2)

The following statistical data is provided for the old and the new cluster.

Status of LMCs

Date/Time from LMC

old LMC: the date/time value from the Initial Load Record (ILR). It is different from the date/time value of the QDS if MERVA ESA terminated abnormally and the LMC was reused without reorganization.

new LMC: the date/time value of the QDS put into the ILR after reorganization.

MERVA Termination

normal: the LMC was closed normally, that is, MERVA ESA was normally terminated the last time the LMC was used.

abnormal: the LMC was not closed, that is, MERVA ESA terminated abnormally the last time the LMC was used.

LMC closed

yes: the LMC was always closed at normal MERVA ESA terminations without any abnormal termination in between.

no since date/time: the LMC was not closed due to an abnormal termination of MERVA ESA at the indicated date/time of the corresponding MERVA ESA startup.

After reorganization *no* always applies for the new LMC.

Processing Mode

Either *load* or *insert* for the old LMC depending on the reorganization status.

load always applies for the new LMC after reorganization.

LMC Statistics

Allocated Disk Space
The amount of space allocated (high-allocated RBA) in bytes.

Used Disk Space
The amount of space within the allocated space up to the RBA of the last byte used in the LMC.

Number of Extents
The number of extents allocated to the LMC.

Maximum Rec Length
The maximum logical record length defined for the LMC.

CI Size The control interval size defined (or default value) for the LMC.

Stored Bytes The amount of space used within allocated space by stored records including the key area of the record. VSAM overhead bytes are not included. The following details are supplied:

- **total**

  Stored bytes of all records belonging to messages that are:
  - Referenced and complete
  - Unreferenced and complete
  - Unreferenced and incomplete.
- **referenced**

  Stored bytes of all records belonging to messages that are referenced in the QDS.
- **unref - complete**

  Stored bytes of all records belonging to messages that are complete, but not referenced in the QDS.
- **unref - incomplete**

  Stored bytes of all records belonging to messages that are incomplete.

Number of LMC Records
The following statistics about LMC records are supplied:

- **referenced**

  Number of records belonging to messages that are referenced in the QDS.
- **unref - complete**

  Number of records belonging to messages that are complete, but not referenced in the QDS.
- **unref - incomplete**

  Number of records belonging to messages that are incomplete.

Large Message Statistics

Limit from DSLPRM
The value entered in the DSLPARM macro is the second subparameter of the large message parameter. The default value is 31900 bytes, which is the maximum size of a message that can be stored in the queue data set.

Length of Large Messages

The following information is supplied:

- **average**

  Accumulated length without key area divided by the number of large messages in the LMC.
- **minimum**

  The minimum length of a message in the LMC.
- **maximum**

  The maximum length of a message in the LMC.

Number of Large Messages

The following information is supplied:

- **total found in LMC**

  The number of complete messages *referenced and unreferenced* in the QDS.
- **referenced**

  The number of complete messages *referenced* in the QDS.
- **unref - complete**

  The number of complete messages *not referenced* in the QDS.
- **unref - incomplete**

  The number of segmented messages where one or more segments were found but not all segments.
- **in QDS-not in LMC**

  The number of large messages found in the QDS having no data in the large message cluster.

  The QDS has been used previously in conjunction with a different LMC or the QDS does not belong to the LMC at all.

OVERVIEW—Messages in QDS, but not in LMC

The following information is supplied for each message that is referenced in the QDS but has no data in the LMC:

- Function
- QSN
- Queue-Key1 and Queue-Key2
- UMR
- Length of message
- Number of segments in LMC
- VSAM key of first segment of message data.

Chapter 21. Maintaining the Journal Data Sets

The journal data sets contain a record of system activity. MERVA ESA has two journal data sets identified by the letters A and B. If one journal data set becomes full or inoperable, MERVA ESA switches to the other data set. By using the operator command **jswitch** you can initiate the switch at any convenient time. It is possible to reset the new data set to an empty state at the same time. This works only when the journal data sets have been defined with the REUSE option in the cluster definition. For details about the **jswitch** command, refer to "Switch the Journal Data Sets (JSWITCH)" on page 53.

This concepts allows the implementation of an automatic backup and archiving procedure for MERVA ESA journal data sets without interrupting the MERVA ESA processing.

The following steps should be executed in sequence:

Step1 Set the journal switch status to manual (operator or REXX API).

Step2 Switch journal data set to B with reset (operator or REXX API).

Step3 Archive journal data set A to a sequential file (IDCAMS REPRO utility).

Step4 Switch journal data set to A with reset (operator or REXX API).

Step5 Archive journal data set B to a sequential file (IDCAMS REPRO utility).

Step6 Reset journal data set B (MERVA ESA utility program DSLJRNR).

Step7 Set the journal switch status to the value **once** to allow emergency switching in case of errors (operator or REXX API).

The sample jobs DSLBJJ02 (for MVS) and DSLBVJ02 (for VSE) provided in the MERVA ESA sample library can be used as an example of how to implement the concept outlined above.

Printing the Journal Data Sets

This chapter contains an example of the MVS or VSE job-control language (JCL) you need to print the journal data sets. The installation library contains the JCL you need to delete the journal data sets and to allocate new space. The name of the JCL member is DSLAJOR. Depending on your installation, this job should be executed only if the old journal data sets are successfully printed or saved.

Job Control Statements under MVS

The job control statements described here are used to print a journal data set under MVS.

Figure 104 on page 294 shows the JCL to print the journal data sets under MVS.

```
//........ JOB ....
//JOBCAT   DD DSN=ucat,DISP=SHR
//RUN      EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=L
//DSLJRNA  DD DSN=jrna,DISP=SHR
//DSLJRNB  DD DSN=jrnb,DISP=SHR
//SYSIN    DD *
  VERIFY  FILE(DSLJRNA)
  PRINT INFILE(DSLJRNA)
  VERIFY  FILE(DSLJRNB)
  PRINT INFILE(DSLJRNB)
```

Figure 104. Printing the Journal Data Sets in MVS

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog for the journal data sets.

jrna The name of the journal A data set.

jrnb The name of the journal B data set.

Job Control Statements under VSE

This section describes the job control statements required to print a journal data set under VSE.

Figure 105 shows the JCL to print the journal data sets under VSE.

```
// JOB ......
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL DSLJRNA,'jrna',,VSAM
// DLBL DSLJRNB,'jrnb',,VSAM
// EXEC IDCAMS,SIZE=AUTO
  VERIFY  FILE(DSLJRNA)
  PRINT INFILE(DSLJRNA)
  VERIFY  FILE(DSLJRNB)
  PRINT INFILE(DSLJRNB)
/*
/&
```

Figure 105. Printing the Journal Data Sets in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog for the journal data sets.

jrna The name of the journal A data set.

jrnb The name of the journal B data set.

Chapter 22. Using the SPA File Initialization Program

The SPA File is used in IMS only. Its purpose is to allow a scratch-pad area (SPA) for the MERVA ESA End-User Driver that is bigger than 32760 bytes and therefore cannot be handled by IMS. IMS only needs to process a SPA of 320 bytes, and all other storage is saved in the MERVA ESA SPA File. The scratch-pad area saves the permanent storage for a user session between two conversation steps.

The SPA File initialization program (DSLEBSPA) must be run before MERVA ESA is started the first time. It must not be run when MERVA ESA is active. The MERVA ESA administrator at your installation will inform you when you should run the SPA File initialization program.

The SPA file consists of three parts:

- One index record
- Three SPA file records reserved for each active end user
- SPA file extension records to store large messages.

The maximum record length is 32760. The minimum record length is 16384.

The size of the SPA File is dependent on the MERVA ESA customizing parameter module DSLPRM that contains the DSLPARM macro.

The USER parameter defines the number of active end-users in MERVA ESA system. The size of the SPA file must be at least *1 + 3 x USER* records. When the SPA file primary allocation is larger, the remaining records are initialized as SPA file extension records for large message processing.

For a detailed description of how to calculate the space requirements of the MERVA ESA SPA file refer to the *MERVA for ESA Installation Guide*.

Figure 106 shows the job control statements for starting the SPA File initialization program.

```
//.......  JOB  ............
//INIT     EXEC PGM=DSLEBSPA
//STEPLIB  DD DISP=SHR,DSN=loadlib
//DSLSPADD DD DSN=spafile,
//            DISP=(disp),SPACE=(blksize,number),
//            DCB=(RECFM=F,BLKSIZE=blksize),UNIT=SYSDA
//SYSPRINT DD SYSOUT=*
```

Figure 106. Initializing the SPA File

In the JCL, the lowercase parameters have the following meanings:

loadlib The name of the load library containing the MERVA ESA programs.

spafile The name of the MERVA ESA SPA File.

disp The value of the MVS job control DISP parameter. When running the job for the first time, it must be NEW,CATLG. When running the job subsequently, it should be OLD or SHR.

blksize The record size (block size).

number The number of records (blocks). The minimum value required is 1 + (3 x USER). The value should not be higher than (BLKSIZE – 30)/5 – (3 x USER). If a higher value is specified the remaining space is wasted, because the index record is too small to control the additional space. The default values used in MERVA ESA are USER=20 and BLKSIZE=32700. In this case the minimum number of records required is 1 + (3 x 20) = 61. The maximum number of records which can be used by MERVA ESA is (32700 – 30)/5 – (3 x 20) = 6474.

Note: From BLKSIZE and NUMBER, MVS calculates the number of tracks or cylinders required for the space allocation. When the above job is used a second time, the SPACE and UNIT parameters can be omitted.

Chapter 23. Using the General File Utility DSLFLUT

The General File utility program DSLFLUT can process files that are used via the MERVA ESA general file services. These are:

- The MERVA ESA Nicknames File
- The Telex Correspondents File
- The SWIFT Correspondents File
- The SWIFT Currency Code File.

DSLFLUT has the following functions:

- To initialize a file
- To list the records of a file.

DSLFLUT reads control statements from a sequential input data set with a fixed record length of 80 bytes. Two control statements are required in each run of DSLFLUT. One of them specifies the function you request, the other specifies the name of the file you want to process. Information on coding control statements follows later in this chapter.

DSLFLUT provides for a report listing that, depending on the function executed, shows the list of records if requested, and the final status of the DSLFLUT processing.

On completion of DSLFLUT, register 15 contains a return code. You can see the return code in the job listing.

There are a few exceptional events when the report data set is not written or errors are not reported by diagnostic messages. These events are flagged by return codes higher than 12. The return code is then the only feedback you get from DSLFLUT.

Initializing a File

Initializing a file is necessary after the definition of the file, before the file is loaded or updated the first time: that is before MERVA ESA is started with the file. Your MERVA ESA administrator will inform you when you should initialize a file.

Listing Records of a File

In a list, the records are shown in ascending order of their keys.

All records contain a unique key. In shared files, the unique key is preceded by an 8-byte owner prefix.

You can list all records of a file or select only some of the records by owner or generic key. The following options are available:

- Owner listing: For a shared file, you can select only the common records or only the private records of one specific owner.
- Generic listing: For a shared or nonshared file, you can select only the records whose unique key starts with the specified character string.

- Generic listing for one owner: For a shared file, you can combine the above options to produce a generic listing from the common records or the private records of one specific owner.

Report Layout

The layout of the report listing is determined by a MERVA ESA Message Control Block (MCB). A default MCB name can be specified for each file in the MERVA ESA File Table.

You can override this default with DSLFLUT control statements, either by giving a new MCB name or by changing specific report characteristics, such as the number of lines per page.

Coding Control Statements

You should code your control statements in a sequential data set with records (input records) of 80 characters fixed length and include the data set in your DSLFLUT job. In MVS installations, the symbolic name of the DD statement of the data set must be DSLIN. In VSE installations, the data set must immediately follow the EXEC control statement.

Only the columns 1-72 of an input line are interpreted; the remaining columns are ignored. Blank input lines are allowed but ignored.

DSLFLUT treats statements as comments if the first nonblank character is an asterisk (*).

Each input record can contain only one control statement. Continuation records are not allowed.

Each control statement consists of a *parameter*, an *equal-sign*, and a *parameter value*. These three items can be put together in one string, or they can be separated from one another by one or more blanks.

The following list describes the parameters and their values. For each parameter, there exists a full name and an abbreviation. The parameters can appear in any sequence in your input data set, and each parameter can be specified only once.

COMMAND

CMD Specifies the function you request from DSLFLUT.

The following values are allowed:

INIT For initializing a file

LIST For listing records of a file.

This parameter is mandatory.

DATA

DAT Specifies the name of the file you want to process.

The file name must have been defined in the MERVA ESA File Table by the DAT parameter of the DSLFLT TYPE=DAT and DSLFLT TYPE=FLD macros.

This parameter is mandatory.

GENERIC

GEN Specifies a generic key for listing records of a file. It is only allowed for the LIST command, and it is optional.

The maximum length of the generic key depends on the file you want to process:

- For nonshared files, the maximum length is equal to the INFLEN parameter of the DSLFLT TYPE=FLD macro in the MERVA ESA File Table.
- For shared files, the above value must be reduced by 8 for the owner prefix.

If the GENERIC parameter is used, the list shows:

- For nonshared files, all the records whose key starts with the specified characters
- For shared files, all the records whose unique key starts with the specified characters. The OWNER control statement can further limit the records selected for the list.

If the GENERIC parameter is *not* used, the list shows all records. For shared files, the OWNER control statement can further limit the records selected for the list.

OWNER

OWN For a shared file only, specifies an owner of the records you want to list. It is only allowed for the LIST command, and it is optional.

The value can be an asterisk (*), which is the owner prefix for all common records ("common" means accessible by all users), or any user identification that is defined in the MERVA ESA user file (indicating private ownership of records).

This parameter can be used together with the GENERIC parameter described above.

LINES

LIN Specifies the number of lines per report page. The value must be in the range from 1 to 99. This parameter is optional.

When records of a file are listed, each page of the report is filled with as many records as fit on one page, according to the number of lines per page. Also, when a record of the file needs several print lines, only complete records are printed on a page. For example, when a record of the file does not fit at the end of a page, the rest of the page is skipped and the record is printed on the next page. Therefore a large value for LINES can reduce the total length of the printed output.

The default number of lines is 55.

MSGID

MID Specifies the message-ID of the MCB you want to use for formatting the report (see "Report Layout" on page 298).

The value must be defined in the MERVA ESA Message Type Table in the MTYPE parameter of the DSLMTT macro. The macro associates the MSGID value with an MCB name.

This parameter is optional.

The default for this parameter is the value specified in the MSGID parameter of the DSLFLT macro in the MERVA ESA File Table.

FORMAT

FOR Specifies the format-ID you use for formatting the report.

The value must be one alphanumeric character. The FORMAT is the language identifier of the print section in the MCB.

One MCB can contain several print sections describing different layouts and national languages, and the FORMAT parameter selects one of them.

The FORMAT parameter is optional.

If it is omitted, its default depends on the COMPRESS parameter:

- E if COMPRESS = 0, 1, 2, or 3
- X if COMPRESS = 4.

COMPRESS

CPR Specifies the compression-ID you use for formatting the report.

The value must be 0, 1, 2, 3, or 4:

0 No compression

1 Empty fields compression

2 Blank lines compression

3 Blank lines and empty fields compression

4 NOPROMPT compression.

This parameter is optional.

The default compression-ID is 0.

When processing the MERVA ESA Nicknames File or the SWIFT Link Correspondents File with the MCBs supplied, COMPRESS=4 must not be used.

Note: When neither FORMAT nor COMPRESS is specified, the defaults are COMPRESS=0 and FORMAT=E.

Environment

Running DSLFLUT requires that MERVA ESA is not started.

CICS Installations: In CICS installations, DSLFLUT runs as a batch program under control of MVS or VSE. When the CICS for MERVA ESA has been started, initializing a file with DSLFLUT requires that CICS has not opened the VSAM data set. To check the status of the data set, use the CICS transaction CEMT. This transaction also allows you to close the data set, if this is required. After DSLFLUT has completed, you can use CEMT to open the data set.

IMS Installations: In IMS installations, DSLFLUT runs as a DL/I batch program.

When the IMS control region is running, and you want to use DSLFLUT to initialize a file, the status of the DL/I DB of the file you want to process must be set to NOTOPEN.

To check the status of the DB, enter the following IMS command at an IMS terminal:

```
/DIS DB dbdname
```

where *dbdname* is the DBD-name of the DB. You can set the status to NOTOPEN by entering:

```
/DBR DB dbdname NOFEOV
```

After processing of DSLFLUT is complete, you can start the DB by entering:

```
/STA DB dbdname
```

Chapter 24. Maintaining the MERVA ESA Nicknames File

The MERVA ESA Nicknames File can be maintained in the following ways:

- By online maintenance, as described in the *MERVA for ESA User's Guide*
- By the General File Maintenance program, DSLFLUT.

Job Control Statements for MVS

The following describes the job control statements you can use under MVS to initialize and list the records in the MERVA ESA nicknames file.

Initializing the MERVA ESA Nicknames File

The following figures show the JCL to initialize the MERVA ESA Nicknames File.

```
//.......  JOB  ............
//INIT     EXEC DLIBATCH,MBR=DSLFLUT,PSB=DSLFLUTL,IMSID=imsid,RGN=2048K
//STEPLIB  DD DSN=imslib,DISP=SHR
//         DD DSN=loadlibi,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//IMS      DD DSN=psblib,DISP=SHR
//         DD DSN=dbdlib,DISP=SHR
//IEFRDER  DD DUMMY
//DFSVSAMP DD *
4096,15
/*
//SYSPRINT DD SYSOUT=L
//DSLCORN  DD DSN=namesfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=INIT
                DAT=DSLCORN
/*
```

Figure 107. Initializing the MERVA ESA Nicknames File in IMS

```
//.......  JOB  ............
//INIT     EXEC PGM=DSLFLUT,REGION=2048K
//STEPLIB  DD DSN=loadlibc,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//SYSPRINT DD SYSOUT=L
//DSLCORN  DD DSN=namesfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=INIT
                DAT=DSLCORN
/*
```

Figure 108. Initializing the MERVA ESA Nicknames File in CICS/MVS

In the JCL, the lowercase parameters have the following meanings:

imsid The identification of the IMS control region

imslib The name of the load library containing the IMS programs

loadlib The name of the load library containing the base MERVA ESA programs. This library has the low-level qualifier SDSLLODB.

loadlibi The name of the load library containing the MERVA ESA programs for IMS. This library has the low-level qualifier SDSLLODI.

loadlibc The name of the load library containing the MERVA ESA programs for CICS. This library has the low-level qualifier SDSLLODC.

psblib The name of the IMS PSB library

dbdlib The name of the IMS DBD library

namesfile The name of the MERVA ESA Nicknames File.

Listing Records of the MERVA ESA Nicknames File

The following figures show the JCL to list records of the MERVA ESA Nicknames File. A control statement with the parameter OWN=YYYYYYYY causes listing of all private records of the user YYYYYYYY.

```
//.......  JOB  ............
//LIST     EXEC DLIBATCH,MBR=DSLFLUT,PSB=DSLFLUTG,IMSID=imsid,RGN=2048K
//STEPLIB  DD DSN=imslib,DISP=SHR
//         DD DSN=loadlibi,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//IMS      DD DSN=psblib,DISP=SHR
//         DD DSN=dbdlib,DISP=SHR
//IEFRDER  DD DUMMY
//DFSVSAMP DD *
4096,15
/*
//SYSPRINT DD SYSOUT=L
//DSLCORN  DD DSN=namesfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=LIST
                DAT=DSLCORN
                OWN=YYYYYYYY
/*
```

Figure 109. Listing Records of the MERVA ESA Nicknames File in IMS

```
//.......  JOB  ............
//LIST     EXEC PGM=DSLFLUT,REGION=2048K
//STEPLIB  DD DSN=loadlibc,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//SYSPRINT DD SYSOUT=L
//DSLCORN  DD DSN=namesfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=LIST
                DAT=DSLCORN
                OWN=YYYYYYYY
/*
```

Figure 110. Listing Records of the MERVA ESA Nicknames File in CICS/MVS

In the JCL, the lowercase parameters have the following meanings:

imsid The identification of the IMS control region

imslib The name of the load library containing the IMS programs

loadlib The name of the load library containing the base MERVA ESA programs. This library has the low-level qualifier SDSLLODB.

loadlibi The name of the load library containing the MERVA ESA programs for IMS. This library has the low-level qualifier SDSLLODI.

loadlibc The name of the load library containing the MERVA ESA programs for CICS. This library has the low-level qualifier SDSLLODC.

psblib The name of the IMS PSB library

dbdlib The name of the IMS DBD library

namesfile The name of the MERVA ESA Nicknames File.

Job Control Statements for VSE

The following describes the job control statements you can use under VSE to initialize and list the records in the MERVA ESA Nicknames file.

Initializing the MERVA ESA Nicknames File

Figure 111 shows the JCL to initialize the MERVA ESA Nicknames File.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL DSLCORN,'namesfile',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// EXEC DSLFLUT,SIZE=200K
                  CMD=INIT
                  DAT=DSLCORN
/*
/&
```

Figure 111. Initializing the MERVA ESA Nicknames File in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

volid The volume identification of the particular data set.

program library
The name of the library containing the MERVA ESA product.

library.sublib The names of the libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibrary names.

namesfile The name of the MERVA ESA Nicknames File.

Listing Records of the MERVA ESA Nicknames File

Figure 112 on page 306 shows the JCL to list records of the MERVA ESA Nicknames File. A control statement with the parameter OWN=YYYYYYYY causes listing of all private records of the user YYYYYYYY.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL DSLCORN,'namesfile',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// EXEC DSLFLUT,SIZE=200K
                  CMD=LIST
                  DAT=DSLCORN
                  OWN=YYYYYYYY
/*
/&
```

Figure 112. Listing Records of the MERVA ESA Nicknames File in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

volid The volume identification of the particular data set.

program library
The name of the library containing the MERVA ESA product.

library.sublib The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibrary names.

namesfile The name of the MERVA ESA Nicknames File.

Chapter 25. Maintaining the Telex Correspondents File

The Telex Correspondents File can be maintained in the following ways:

- By online maintenance, as described in the *MERVA for ESA User's Guide*
- By the General File Maintenance program, DSLFLUT.

Job Control Statements for MVS

The following describes the job control statements you can use under MVS to initialize and list the records in the MERVA ESA Telex Correspondents File.

Initializing the Telex Correspondents File

The following figures show the JCL to initialize the Telex Correspondents File.

```
//.......  JOB  ............
//INIT     EXEC DLIBATCH,MBR=DSLFLUT,PSB=DSLFLUTL,IMSID=imsid,RGN=2048K
//STEPLIB  DD DSN=imslib,DISP=SHR
//         DD DSN=loadlibi,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//IMS      DD DSN=psblib,DISP=SHR
//         DD DSN=dbdlib,DISP=SHR
//IEFRDER  DD DUMMY
//DFSVSAMP DD *
4096,15
/*
//SYSPRINT DD SYSOUT=L
//ENLCOR   DD DSN=namesfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=INIT
                DAT=ENLCORDA
/*
```

Figure 113. Initializing the Telex Correspondents File in IMS

```
//.......  JOB  ............
//INIT     EXEC PGM=DSLFLUT,REGION=2048K
//STEPLIB  DD DSN=loadlibc,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//SYSPRINT DD SYSOUT=L
//ENLCOR   DD DSN=namesfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=INIT
                DAT=ENLCORDA
/*
```

Figure 114. Initializing the Telex Correspondents File in CICS/MVS

In the JCL, the lowercase parameters have the following meanings:

imsid The identification of the IMS control region

imslib The name of the load library containing the IMS programs

loadlib The name of the load library containing the base MERVA ESA programs. This library has the low-level qualifier SDSLLODB.

loadlibi The name of the load library containing the MERVA ESA programs for IMS. This library has the low-level qualifier SDSLLODI.

loadlibc The name of the load library containing the MERVA ESA programs for CICS. This library has the low-level qualifier SDSLLODC.

psblib The name of the IMS PSB library

dbdlib The name of the IMS DBD library

namesfile The name of the Telex Correspondents File.

Listing Records of the Telex Correspondents File

The following figures show the JCL to list records of the Telex Correspondents File.

```
//.......  JOB  ............
//LIST     EXEC DLIBATCH,MBR=DSLFLUT,PSB=DSLFLUTG,IMSID=imsid,RGN=2048K
//STEPLIB  DD DSN=imslib,DISP=SHR
//         DD DSN=loadlibi,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//IMS      DD DSN=psblib,DISP=SHR
//         DD DSN=dbdlib,DISP=SHR
//IEFRDER  DD DUMMY
//DFSVSAMP DD *
4096,15
/*
//SYSPRINT DD SYSOUT=L
//ENLCOR   DD DSN=namesfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=LIST
                DAT=ENLCORDA
/*
```

Figure 115. Listing Records of the Telex Correspondents File in IMS

```
//.......  JOB  ............
//LIST     EXEC PGM=DSLFLUT,REGION=2048K
//STEPLIB  DD DSN=loadlibc,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//SYSPRINT DD SYSOUT=L
//ENLCOR   DD DSN=namesfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=LIST
                DAT=ENLCORDA
/*
```

Figure 116. Listing Records of the Telex Correspondents File in CICS/MVS

In the JCL, the lowercase parameters have the following meanings:

imsid The identification of the IMS control region

imslib The name of the load library containing the IMS programs

loadlib The name of the load library containing the base MERVA ESA programs. This library has the low-level qualifier SDSLLODB.

loadlibi The name of the load library containing the MERVA ESA programs for IMS. This library has the low-level qualifier SDSLLODI.

loadlibc The name of the load library containing the MERVA ESA programs for CICS. This library has the low-level qualifier SDSLLODC.

psblib The name of the IMS PSB library

dbdlib The name of the IMS DBD library

namesfile The name of the Telex Correspondents File.

Job Control Statements for VSE

The following describes the job control statements you can use under VSE to initialize and list the records in the MERVA ESA Telex Correspondents File.

Initializing the Telex Correspondents File

Figure 117 shows the JCL to initialize the Telex Correspondents File.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL ENLCOR,'namesfile',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// EXEC DSLFLUT,SIZE=200K
                  CMD=INIT
                  DAT=ENLCORDA
/*
/&
```

Figure 117. Initializing the Telex Correspondents File in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

volid The volume identification of the particular data set.

program library
The name of the library containing the MERVA ESA product.

library.sublib The names of the libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibrary names.

namesfile The name of the Telex Correspondents File.

Listing Records of the Telex Correspondents File

Figure 118 shows the JCL to list records of the Telex Correspondents File.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL ENLCOR,'namesfile',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// EXEC DSLFLUT,SIZE=200K
                  CMD=LIST
                  DAT=ENLCORDA
/*
/&
```

Figure 118. Listing Records of the Telex Correspondents File in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

volid The volume identification of the particular data set.

program library
The name of the library containing the MERVA ESA product.

library.sublib The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibrary names.

namesfile The name of the Telex Correspondents File.

Chapter 26. Maintaining the SWIFT Correspondents File

The SWIFT Correspondents File can be maintained in the following ways:

- By online maintenance, as described in the *MERVA for ESA User's Guide*
- By the General File Maintenance program, DSLFLUT
- By the SWIFT Correspondents File utility, DWSCORUT.

This chapter also describes the BIC Tape Conversion utility, DWSBICCV.

Job Control Statements for MVS

The General File Maintenance program, DSLFLUT, allows you to initialize the SWIFT Correspondents File, and to list the records of it.

Initializing the SWIFT Correspondents File

The following figures show the JCL to initialize the SWIFT Correspondents File.

```
//.......  JOB  ............
//INIT     EXEC DLIBATCH,MBR=DSLFLUT,PSB=DSLFLUTL,IMSID=imsid,RGN=2048K
//STEPLIB  DD DSN=imslib,DISP=SHR
//         DD DSN=loadlibi,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//IMS      DD DSN=psblib,DISP=SHR
//         DD DSN=dbdlib,DISP=SHR
//IEFRDER  DD DUMMY
//DFSVSAMP DD *
4096,15
/*
//SYSPRINT DD SYSOUT=L
//DWSCOR   DD DSN=corrfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=INIT
                DAT=DWSCORDA
/*
```

Figure 119. Initializing the SWIFT Correspondents File in IMS

```
//.......  JOB  ............
//INIT     EXEC PGM=DSLFLUT,REGION=2048K
//STEPLIB  DD DSN=loadlibc,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//SYSPRINT DD SYSOUT=L
//DWSCOR   DD DSN=corrfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=INIT
                DAT=DWSCORDA
/*
```

Figure 120. Initializing the SWIFT Correspondents File in CICS/MVS

In the JCL, the lowercase parameters have the following meanings:

imsid The identification of the IMS control region

imslib The name of the load library containing the IMS programs

loadlib The name of the load library containing the base MERVA ESA programs. This library has the low-level qualifier SDSLLODB.

loadlibi The name of the load library containing the MERVA ESA programs for IMS. This library has the low-level qualifier SDSLLODI.

loadlibc The name of the load library containing the MERVA ESA programs for CICS. This library has the low-level qualifier SDSLLODC.

psblib The name of the IMS PSB library

dbdlib The name of the IMS DBD library

corrfile The name of the SWIFT Correspondents File.

Listing Records of the SWIFT Correspondents File

The following figures show the JCL to list records of the SWIFT Correspondents File. A control statement with the parameter GEN=XXXX causes listing of all SWIFT correspondents with the bank code XXXX.

```
//.......  JOB  ............
//LIST     EXEC DLIBATCH,MBR=DSLFLUT,PSB=DSLFLUTG,IMSID=imsid,RGN=2048K
//STEPLIB  DD DSN=imslib,DISP=SHR
//         DD DSN=loadlibi,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//IMS      DD DSN=psblib,DISP=SHR
//         DD DSN=dbdlib,DISP=SHR
//IEFRDER  DD DUMMY
//DFSVSAMP DD *
4096,15
/*
//SYSPRINT DD SYSOUT=L
//DWSCOR   DD DSN=corrfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=LIST
                DAT=DWSCORDA
                GEN=XXXX
/*
```

Figure 121. Listing Records of the SWIFT Correspondents File in IMS

```
//.......  JOB  ............
//LIST     EXEC PGM=DSLFLUT,REGION=2048K
//STEPLIB  DD DSN=loadlibc,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//SYSPRINT DD SYSOUT=L
//DWSCOR   DD DSN=corrfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=LIST
                DAT=DWSCORDA
                GEN=XXXX
/*
```

Figure 122. Listing Records of the SWIFT Correspondents File in CICS/MVS

In the JCL, the lowercase parameters have the following meanings:

imsid The identification of the IMS control region

imslib The name of the load library containing the IMS programs

loadlib The name of the load library containing the base MERVA ESA programs. This library has the low-level qualifier SDSLLODB.

loadlibi The name of the load library containing the MERVA ESA programs for IMS. This library has the low-level qualifier SDSLLODI.

loadlibc The name of the load library containing the MERVA ESA programs for CICS. This library has the low-level qualifier SDSLLODC.

psblib The name of the IMS PSB library

dbdlib The name of the IMS DBD library

corrfile The name of the SWIFT Correspondents File.

Job Control Statements for VSE

The General File Maintenance utility allows you to initialize the SWIFT Correspondents File, and to list the records in it.

Initializing the SWIFT Correspondents File

Figure 123 shows the JCL to initialize the SWIFT Correspondents File.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL DWSCOR,'corrfile',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// EXEC DSLFLUT,SIZE=200K
                  CMD=INIT
                  DAT=DWSCORDA
/*
/&
```

Figure 123. Initializing the SWIFT Correspondents File in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

volid The volume identification of the particular data set.

program library
The name of the library containing the MERVA ESA product.

library.sublib The names of the libraries containing the MERVA ESA programs and the JCL procedures. You can code a list of sublibrary names.

corrfile The name of the SWIFT Correspondents File.

Listing Records of the SWIFT Correspondents File

Figure 124 on page 314 shows the JCL to list records of the SWIFT Correspondents File. A control statement with the parameter GEN=XXXX causes listing of all SWIFT correspondents with the bank code XXXX.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL DWSCOR,'corrfile',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// EXEC DSLFLUT,SIZE=200K
                  CMD=LIST
                  DAT=DWSCORDA
                  GEN=XXXX
/*
/&
```

Figure 124. Listing Records of the SWIFT Correspondents File in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

volid The volume identification of the particular data set.

program library
The name of the library containing the MERVA ESA product.

library.sublib The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibrary names.

corrfile The name of the SWIFT Correspondents File.

Using the SWIFT Correspondents File Utility DWSCORUT

The SWIFT Correspondents File utility program DWSCORUT can import correspondents' data from

- the SWIFT Bank Identifier Code (BIC) Directory Update tape, or
- the SWIFT Bank Identifier Code (BIC) Database Plus tape

into the SWIFT Correspondents File. The SWIFT BIC tapes are created, maintained, and distributed by S.W.I.F.T.

If you do not receive either of these tapes from S.W.I.F.T., but instead receive the SWIFT BIC Bankfile tape, you must use the SWIFT Link utility DWSBICCV to convert the SWIFT BIC Bankfile tape into a format that can be used as input to the DWSCORUT utility. The utility DWSBICCV is described in "Using the Tape Conversion Utility DWSBICCV" on page 320.

DWSCORUT replaces existing records, adds new ones, and deletes records that are no longer on the tape received from SWIFT.

DWSCORUT provides a report listing that shows the updates made to the file: for example, the expansion data of records that have been added or deleted, and the old and new expansion data of records that have been changed.

The report also shows any records that were entered via online maintenance or by another program. These records remain unchanged in the SWIFT correspondents file.

The report includes feedback information such as the return code and messages for confirmation, warning, or error diagnosis.

On completion of DWSCORUT, register 15 contains a return code. You can see the return code in the job listing.

There are a few exceptional events when the report data set is not written or errors are not reported by diagnostic messages. These events are flagged by return codes higher than 12. The return code is then the only feedback you get from DWSCORUT.

DWSCORUT reads control statements from a sequential input data set with a fixed record length of 80 bytes. All control statements are optional. Details on coding control statements follow later in this chapter.

Report Layout

The layout of the report listing is determined by MERVA ESA Message Control Blocks (MCBs). You can specify in a control statement which MCB to use, or the MCB specified in the MERVA ESA File Table is taken. There are also control statements to change the report layout for a given MCB, for example, to specify the number of lines per report page.

Note: The SWIFT Link supplies the MCB DWSSCOR for formatting the report.

Coding Control Statements

For the control statements of DWSCORUT, the same rules apply as for the MERVA ESA general file utility DSLFLUT as described in section "Coding Control Statements" on page 298, except that for MVS the symbolic name of the DD statement of the input data set must be DWSIN.

DWSCORUT uses the following control statements in the same way as DSLFLUT:

- LINES
- MSGID
- FORMAT
- COMPRESS.

In addition, DWSCORUT supports the following control statements:

UPDATE
UPD Specifies if the import will be conditional or unconditional.

The following values are allowed:

UNCOND The correspondents' data is imported from the BIC input tape regardless of the applicability date.

The applicability date is shown in the report.

If the applicability date has not yet been reached, the return code is 4, and a warning message is shown in the report.

UNCOND is the default.

COND The correspondents' data is imported from the BIC input tape with the following conditions:

- If the applicability date has not yet been reached, no data is processed. A return code of 8 and an error message are shown in the report.

- If the applicability date has been reached, the correspondents' data is imported from the BIC input tape. A return code of 0 is shown in the report.

The applicability date is shown in the report.

LIST
LIS

With this parameter you can select the parts of the report listing to be produced. The report always includes:

- A list of the duplicate BIC codes that were ignored, and any records where user modifications were replaced
- A summary page which shows any error messages and gives statistical information about the file updates.

Further details can be selected with the following parameter values:

ALL All additional details are printed. The report lists all records of the SWIFT correspondents file that were added or where the expansion data was changed, all records of the BIC input tape that had no BIC code, all records of the SWIFT correspondents file that were deleted, and all records of the file that were added via online maintenance or by another program.

LIST=ALL is the default.

NONE No additional details are shown in the report listing.

CHANGES Lists all records of the SWIFT correspondents file that were added or deleted, and all records where the expansion data was changed.

USER Lists only the records of the SWIFT correspondents file that were added via online maintenance or by another program. These records were kept unchanged.

NOBIC Lists only the records of the BIC Database Plus tape that contained no BIC code. These records were ignored by DWSCORUT.

TAPE
TAP

This parameter is allowed only in VSE, and it specifies the attributes of the device from which you import the correspondents' data.

The following values are allowed:

NOLABEL The device is a nonlabeled tape (855 bytes/record).

LABEL The device is a labeled tape (855 bytes/record).

DISK The device is a disk (855 bytes/record).

BIC+NO The device is a nonlabeled tape (1248 bytes/record).

BIC+LAB The device is a labeled tape (1248 bytes/record).

BIC+DISK The device is a disk (1248 bytes/record).

The BIC Directory Update tape is a nonlabeled tape with 855 bytes/record. Use TAPE=NOLABEL.

The BIC Database Plus tape is a nonlabeled tape with 1248 bytes/record. Use TAPE=BIC+NO.

The default for this parameter is TAPE=NOLABEL.

VOLUMES
VOL
This parameter is used only in VSE for nonlabeled tapes. It specifies the number of physical tapes to be processed. You may give a number from 1 to 9.

The default is VOLUMES=1 for a BIC Directory Update tape and VOLUMES=2 for a BIC Database Plus file.

Environment

Running DWSCORUT requires that MERVA ESA has not been started.

CICS Installations

In CICS installations, DWSCORUT runs as a batch program under control of MVS or VSE. When the CICS for MERVA ESA has been started, running DWSCORUT requires that CICS has not opened the VSAM data set of the SWIFT Correspondents File. To check the status of the data set, use the CICS transaction CEMT, which also allows you to close the data set if necessary.

After completion of DWSCORUT, you can use CEMT to open the data set.

IMS Installations

In IMS installations, DWSCORUT runs as a DL/I batch program. When the IMS control region is running, and you want to use DWSCORUT, the status of the DL/I DB of the SWIFT Correspondents File must be NOTOPEN.

To check the status of the DB, enter the following IMS command from an IMS terminal:

```
/DIS DB DWSCOR
```

where *DWSCOR* is the DBD-name of the DB. You can set the status to NOTOPEN by entering:

```
/DBR DB DWSCOR NOFEOV
```

After processing of DWSCORUT is complete, you can start the DB by entering:

```
/STA DB DWSCOR
```

Job Control Statements for MVS

The following figures show sample jobs for running DWSCORUT.

In these examples, the data is imported only if the applicability date of the BIC tape has already been reached.

```
//.......  JOB  ............
//IMPORT   EXEC DLIBATCH,MBR=DWSCORUT,PSB=DWSCORUT,IMSID=imsid,RGN=2048K
//STEPLIB  DD DSN=imslib,DISP=SHR
//         DD DSN=loadlibi,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//IMS      DD DSN=psblib,DISP=SHR
//         DD DSN=dbdlib,DISP=SHR
//IEFRDER  DD DUMMY
//DFSVSAMP DD *
4096,15
/*
//SYSPRINT DD SYSOUT=L
//DWSCOR   DD DSN=corrfile,DISP=SHR
//DWSPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DWSTAPE  DD VOL=SER=volume,UNIT=TAPE,LABEL=(1,NL),DISP=SHR,
//            DCB=(RECFM=FB,LRECL=lrecl,BLKSIZE=blksize)
//DWSIN    DD *
                 UPDATE=COND
/*
```

Figure 125. Import Correspondents Data from BIC Tape in IMS

```
//.......  JOB  ............
//IMPORT   EXEC PGM=DWSCORUT,REGION=2048K
//STEPLIB  DD DSN=loadlibc,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//SYSPRINT DD SYSOUT=L
//DWSCOR   DD DSN=corrfile,DISP=SHR
//DWSPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DWSTAPE  DD VOL=SER=volume,UNIT=TAPE,LABEL=(1,NL),DISP=SHR,
//            DCB=(RECFM=FB,LRECL=lrecl,BLKSIZE=blksize)
//DWSIN    DD *
                 UPDATE=COND
/*
```

Figure 126. Import Correspondents Data from BIC Tape in CICS/MVS

In the JCL, the lowercase parameters have the following meanings:

imsid The identification of the IMS control region.

imslib The name of the load library containing the IMS programs.

loadlib The name of the load library containing the base MERVA ESA programs. This library has the low-level qualifier SDSLLODB.

loadlibi The name of the load library containing the MERVA ESA programs for IMS. This library has the low-level qualifier SDSLLODI.

loadlibc The name of the load library containing the MERVA ESA programs for CICS. This library has the low-level qualifier SDSLLODC.

psblib The name of the IMS PSB library.

dbdlib The name of the IMS DBD library.

corrfile The name of the SWIFT Correspondents File.

volume The volume identification of the BIC tape. If the BIC tape requires two volumes, specify

```
//DWSTAPE DD VOL=SER=(volume1,volume2),...
```

lrecl The record length on the BIC tape.

blksize The block size on the BIC tape.

Job Control Statements for VSE

The following figure shows the JCL for running DWSCORUT.

In these examples, the data is imported only if the applicability date of the BIC tape has already been reached.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL DWSCOR,'corrfile',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// ASSGN SYS025,nnn
// EXEC DWSCORUT,SIZE=200K
                  UPDATE=COND
/*
/&
```

Figure 127. Import Correspondents Data from BIC Tape in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

volid The volume identification of the particular data set.

program library
The name of the library containing the MERVA ESA and SWIFT Link programs.

library.sublib The names of the program libraries containing the MERVA ESA and SWIFT Link programs and JCL procedures. You can code a list of sublibraries.

corrfile The name of the SWIFT Correspondents File.

nnn The unit number of the tape unit with the BIC tape. If the tape has two volumes, you can improve the performance by using two tape units. Specify the units as follows:

```
// ASSGN SYS025,nnn
// ASSGN SYS025,mmm,ALT
```

If you have only one tape unit, after unloading the first volume and mounting the second, you must enter the response NEWTAP to continue processing.

Note: When a disk device is used instead of the magnetic tape, the statement:

```
// ASSGN SYS025,nnn
```

must be replaced by the statements:

```
// ASSGN SYS025,DISK,...
// DLBL DWSDBIC,...
// EXTENT SYS025,...
```

using appropriate parameters. You must then also use a DWSCORUT control statement:

```
TAPE=DISK
```

Using the Tape Conversion Utility DWSBICCV

The SWIFT Link tape conversion utility program DWSBICCV converts a SWIFT BIC Bankfile tape into a tape that can be used as input to the SWIFT Correspondents File utility DWSCORUT. DWSBICCV creates a sequential file that has the same record layout as the BIC Directory Update tape (855 bytes per record). In MVS, the sequential file can be on a tape or disk. In VSE, the sequential file must be on a nonlabeled tape.

Job Control Statements for MVS

The following figure shows the JCL for running DWSBICCV in MVS.

```
//.......  JOB  ............
//STEP     EXEC  PGM=DWSBICCV
//STEPLIB  DD DSN=loadlib,DISP=SHR
//SYSPRINT DD SYSOUT=L
//SYSUDUMP DD SYSOUT=L
//DWSBANKF DD VOL=SER=bankfile,
//            UNIT=TAPE,LABEL=(1,NL),
//            DCB=(RECFM=FB,LRECL=221,BLKSIZE=2652)
//DWSBIC   DD VOL=SER=bic,
//            UNIT=TAPE,LABEL=(1,NL),
//            DCB=(RECFM=FB,LRECL=855,BLKSIZE=10260)
//DWSPRINT DD SYSOUT=L,DCB=(RECFM=VBA,LRECL=133,BLKSIZE=137)
```

Figure 128. JCL to Convert the BIC Tape in MVS

In this job, lowercase letters have the following meanings:

loadlib The name of the load library containing the MERVA ESA and SWIFT Link programs.

bankfile The volume identification of the SWIFT BIC Bankfile tape.

bic The volume identification of the converted SWIFT BIC Directory Update tape. The example shows a tape, but you can also use a sequential file on disk, using the same symbolic name of the DD statement DWSBIC. This file will be input to DWSCORUT.

Job Control Statements for VSE

The following figure shows the JCL for running DWSBICCV in VSE.

```
// JOB ...
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// ASSGN SYS025,nnn
// ASSGN SYS026,mmm
// EXEC DWSBICCV,SIZE=200K
/*
/&
```

Figure 129. JCL to Convert the BIC Tape in VSE

In this JCL, lowercase letters have the following meanings:

program library
The name of the program library containing the MERVA ESA and SWIFT Link programs.

volid	The volume identification of the program library.
library.sublib	The names of the program libraries containing the MERVA ESA and SWIFT Link programs for execution.
nnn	The number of the tape unit for the SWIFT BIC Bankfile tape.
mmm	The number of the tape unit for the converted SWIFT BIC Directory Update tape. This tape must be a nonlabeled tape that will be input to DWSCORUT.

Chapter 27. Maintaining the SWIFT Currency Code File

The SWIFT Currency Code File can be maintained in the following ways:

- By online maintenance, as described in the *MERVA for ESA User's Guide*
- By the General File Maintenance program, DSLFLUT
- By the SWIFT Currency Code File utility, DWSCURUT.

Job Control Statements for MVS

The General File Maintenance program, DSLFLUT, allows you to initialize the SWIFT Currency Code File, and to list the records of it.

Initializing the SWIFT Currency Code File

The following figures show the JCL to initialize the SWIFT Currency Code File.

```
//.......  JOB  ............
//INIT     EXEC DLIBATCH,MBR=DSLFLUT,PSB=DSLFLUTL,IMSID=imsid,RGN=2048K
//STEPLIB  DD DSN=imslib,DISP=SHR
//         DD DSN=loadlibi,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//IMS      DD DSN=psblib,DISP=SHR
//         DD DSN=dbdlib,DISP=SHR
//IEFRDER  DD DUMMY
//DFSVSAMP DD *
4096,15
/*
//SYSPRINT DD SYSOUT=L
//DWSCUR   DD DSN=curfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=INIT
                DAT=DWSCURDA
/*
```

Figure 130. Initializing the SWIFT Currency Code File in IMS

```
//.......  JOB  ............
//INIT     EXEC PGM=DSLFLUT,REGION=2048K
//STEPLIB  DD DSN=loadlibc,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//SYSPRINT DD SYSOUT=L
//DWSCUR   DD DSN=curfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=INIT
                DAT=DWSCURDA
/*
```

Figure 131. Initializing the SWIFT Currency Code File in CICS/MVS

In the JCL, the lowercase parameters have the following meanings:

imsid The identification of the IMS control region

imslib The name of the load library containing the IMS programs

loadlib The name of the load library containing the base MERVA ESA programs. This library has the low-level qualifier SDSLLODB.

loadlibi The name of the load library containing the MERVA ESA programs for IMS. This library has the low-level qualifier SDSLLODI.

loadlibc The name of the load library containing the MERVA ESA programs for CICS. This library has the low-level qualifier SDSLLODC.

psblib The name of the IMS PSB library

dbdlib The name of the IMS DBD library

curfile The name of the SWIFT Currency Code File.

Listing Records of the SWIFT Currency Code File

The following figures show the JCL to list records of the SWIFT Currency Code File.

```
//.......  JOB  ............
//LIST     EXEC DLIBATCH,MBR=DSLFLUT,PSB=DSLFLUTG,IMSID=imsid,RGN=2048K
//STEPLIB  DD DSN=imslib,DISP=SHR
//         DD DSN=loadlibi,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//IMS      DD DSN=psblib,DISP=SHR
//         DD DSN=dbdlib,DISP=SHR
//IEFRDER  DD DUMMY
//DFSVSAMP DD *
4096,15
/*
//SYSPRINT DD SYSOUT=L
//DWSCUR   DD DSN=curfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=LIST
                DAT=DWSCURDA
/*
```

Figure 132. Listing Records of the SWIFT Currency Code File in IMS

```
//.......  JOB  ............
//LIST     EXEC PGM=DSLFLUT,REGION=2048K
//STEPLIB  DD DSN=loadlibc,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//SYSPRINT DD SYSOUT=L
//DWSCUR   DD DSN=curfile,DISP=SHR
//DSLPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DSLIN    DD *
                CMD=LIST
                DAT=DWSCURDA
/*
```

Figure 133. Listing Records of the SWIFT Currency Code File in CICS/MVS

In the JCL, the lowercase parameters have the following meanings:

imsid The identification of the IMS control region

imslib The name of the load library containing the IMS programs

loadlib The name of the load library containing the base MERVA ESA programs. This library has the low-level qualifier SDSLLODB.

loadlibi The name of the load library containing the MERVA ESA programs for IMS. This library has the low-level qualifier SDSLLODI.

loadlibc The name of the load library containing the MERVA ESA programs for CICS. This library has the low-level qualifier SDSLLODC.

psblib The name of the IMS PSB library

dbdlib The name of the IMS DBD library

curfile The name of the SWIFT Currency Code File.

Job Control Statements for VSE

The General File Maintenance utility allows you to initialize the SWIFT Currency Code File, and to list the records in it.

Initializing the SWIFT Currency Code File

Figure 134 shows the JCL to initialize the SWIFT Currency Code File.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL DWSCUR,'curfile',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// EXEC DSLFLUT,SIZE=200K
                  CMD=INIT
                  DAT=DWSCURDA
/*
/&
```

Figure 134. Initializing the SWIFT Currency Code File in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

curfile The name of the SWIFT Currency Code File.

program library
The name of the library containing the MERVA ESA product.

volid The volume identification of the particular data set.

library.sublib The names of the libraries containing the MERVA ESA programs and the JCL procedures. You can code a list of sublibrary names.

Listing Records of the SWIFT Currency Code File

Figure 135 on page 326 shows the JCL to list records of the SWIFT Currency Code File.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL DWSCUR,'curfile',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// EXEC DSLFLUT,SIZE=200K
                  CMD=LIST
                  DAT=DWSCURDA
/*
/&
```

Figure 135. Listing Records of the SWIFT Currency Code File in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

curfile The name of the SWIFT Currency Code File.

program library
The name of the library containing the MERVA ESA product.

volid The volume identification of the particular data set.

library.sublib The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibrary names.

Using the SWIFT Currency Code File Utility DWSCURUT

The SWIFT Currency Code File utility program DWSCURUT can import currency codes from:

- The SWIFT Bank Identifier Code (BIC) Directory Update tape, or
- The SWIFT Bank Identifier Code (BIC) Database Plus tape

into the SWIFT Currency Code File. The SWIFT BIC tapes are created, maintained, and distributed by S.W.I.F.T.

DWSCURUT replaces existing records, adds new ones, or deletes records that are no longer on the tape received from S.W.I.F.T.

DWSCURUT provides a report listing that shows the updates made to the file: for example, the contents of records that have been added, the contents of records that have been deleted, and the old and new contents of records that have been changed.

The report also includes feedback information such as the return code and messages for confirmation, warning, or error diagnosis.

On completion of DWSCURUT, register 15 contains a return code. You can see the return code in the job listing.

There are a few exceptional events when the report data set is not written or errors are not reported by diagnostic messages. These events are flagged by return codes higher than 12. The return code is then the only feedback you get from DWSCURUT.

DWSCURUT reads control statements from a sequential input data set with a fixed record length of 80 bytes. All control statements are optional. Details on coding control statements follow later in this chapter.

Report Layout

The layout of the report listing is determined by MERVA ESA Message Control Blocks (MCBs). You can specify in a control statement which MCB to use, or the MCB specified in the MERVA ESA File Table is taken. There are also control statements to change the report layout for a given MCB, for example, to specify the number of lines per report page.

Note: The SWIFT Link supplies the MCB DWSSCUR for formatting the report listing.

Coding Control Statements

For the control statements of DWSCURUT, the same rules apply as for the MERVA ESA general file utility DSLFLUT as described in "Coding Control Statements" on page 298, except that for MVS the symbolic name of the DD statement of the input data set must be DWSIN.

DWSCURUT supports the following control statements in the same way as DSLFLUT:

- LINES
- MSGID
- FORMAT
- COMPRESS.

DWSCURUT also supports the following control statements:

UPDATE

UPD Specifies if the import will be conditional or unconditional.

The following values are allowed:

UNCOND The currency codes are imported from the BIC tape regardless of the applicability date.

The applicability date is shown in the report.

If the applicability date has not yet been reached, the return code is 4, and a warning message is shown in the report.

UNCOND is the default.

COND The currency codes are imported from the BIC tape with the following conditions:

- If the applicability date has not yet been reached, no data is processed. A return code of 8 and an error message are shown in the report.
- If the applicability date has been reached, the currency codes are imported from the BIC tape. A return code of 0 is shown in the report.

The applicability date is shown in the report.

TAPE

TAP This parameter is only allowed in VSE, and it specifies the attributes of the device from which you import the currency codes.

The following values are allowed:

NOLABEL The device is a nonlabeled tape (855 bytes/record).

LABEL The device is a labeled tape (855 bytes/record).

DISK The device is a disk (855 bytes/record).

BIC+NO The device is a nonlabeled tape (1248 bytes/record).

BIC+LAB The device is a labeled tape (1248 bytes/record).

BIC+DISK The device is a disk (1248 bytes/record).

The BIC Directory Update tape is a nonlabeled tape with 855 bytes/record. Use TAPE=NOLABEL.

The BIC Database Plus tape is a nonlabeled tape with 1248 bytes/record. Use TAPE=BIC+NO.

The default for this parameter is TAPE=NOLABEL.

VOLUMES

VOL This parameter is used only in VSE for nonlabeled tapes. It specifies the number of physical tapes to be processed. You may give a number from 1 to 9.

The default is VOLUMES=1 for a BIC Directory Update tape and VOLUMES=2 for a BIC Database Plus file.

Environment

Running DWSCURUT requires that MERVA ESA has not been started.

CICS Installations

In CICS installations, DWSCURUT runs as a batch program under control of MVS or VSE. When the CICS for MERVA ESA has been started, running DWSCURUT requires that CICS has not opened the VSAM data set of the SWIFT Currency Code File. To check the status of the data set, use the CICS transaction CEMT, which also allows you to close the data set if necessary.

After completion of DWSCURUT, you can use CEMT to open the data set.

IMS Installations

In IMS installations, DWSCURUT runs as a DL/I batch program. When the IMS control region is running, and you want to use DWSCURUT, the status of the DL/I DB of the SWIFT Currency Code File must be NOTOPEN. To check the status of the DB, enter the following IMS command from an IMS terminal:

```
/DIS DB DWSCUR
```

where *DWSCUR* is the DBD-name of the DB. You can set the status to NOTOPEN by entering:

```
/DBR DB DWSCUR NOFEOV
```

After processing of DWSCURUT is complete, you can start the DB by entering:

```
/STA DB DWSCUR
```

Job Control Statements for MVS

The following figures show sample jobs for running DWSCURUT.

In these examples, the data is imported only if the applicability date of the BIC tape has already been reached.

```
//.......  JOB  ............
//IMPORT   EXEC DLIBATCH,MBR=DWSCURUT,PSB=DWSCURUT,IMSID=imsid,RGN=2048K
//STEPLIB  DD DSN=imslib,DISP=SHR
//         DD DSN=loadlibi,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//IMS      DD DSN=psblib,DISP=SHR
//         DD DSN=dbdlib,DISP=SHR
//IEFRDER  DD DUMMY
//DFSVSAMP DD *
4096,15
/*
//SYSPRINT DD SYSOUT=L
//DWSCUR   DD DSN=curfile,DISP=SHR
//DWSPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DWSTAPE  DD VOL=SER=volume,UNIT=TAPE,LABEL=(1,NL),DISP=SHR,
//            DCB=(RECFM=FB,LRECL=lrecl,BLKSIZE=blksize)
//DWSIN    DD *
                UPDATE=COND
/*
```

Figure 136. Import Currency Codes from BIC Tape in IMS

```
//.......  JOB  ............
//IMPORT   EXEC PGM=DWSCURUT,REGION=2048K
//STEPLIB  DD DSN=loadlibc,DISP=SHR
//         DD DSN=loadlib,DISP=SHR
//SYSPRINT DD SYSOUT=L
//DWSCUR   DD DSN=curfile,DISP=SHR
//DWSPRINT DD SYSOUT=L,DCB=(BLKSIZE=1370)
//DWSTAPE  DD VOL=SER=volume,UNIT=TAPE,LABEL=(1,NL),DISP=SHR,
//            DCB=(RECFM=FB,LRECL=lrecl,BLKSIZE=blksize)
//DWSIN    DD *
                UPDATE=COND
/*
```

Figure 137. Import Currency Codes from BIC Tape in CICS/MVS

In the JCL, the lowercase parameters have the following meanings:

imsid The identification of the IMS control region

imslib The name of the load library containing the IMS programs

loadlib The name of the load library containing the base MERVA ESA programs. This library has the low-level qualifier SDSLLODB.

loadlibi The name of the load library containing the MERVA ESA programs for IMS. This library has the low-level qualifier SDSLLODI.

loadlibc The name of the load library containing the MERVA ESA programs for CICS. This library has the low-level qualifier SDSLLODC.

psblib The name of the IMS PSB library

dbdlib The name of the IMS DBD library

curfile The name of the SWIFT Currency Code File

volume The volume identification of the BIC tape. If the tape requires two volumes, specify

```
//DWSTAPE DD VOL=SER=(volume1,volume2),...
```

lrecl The record length on the BIC tape

blksize The block size on the BIC tape.

Job Control Statements for VSE

The following figure shows the JCL for running DWSCURUT.

In these examples, the data is imported only if the applicability date of the BIC tape has already been reached.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL DWSCUR,'curfile',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// ASSGN SYS025,nnn
// EXEC DWSCURUT,SIZE=200K
                  UPDATE=COND
/*
/&
```

Figure 138. Import Currency Codes from BIC Tape in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

curfile The name of the SWIFT Currency Code File.

program library
The name of the library containing the MERVA ESA and SWIFT Link programs.

volid The volume identification of the particular data set.

library.sublib The names of the program libraries containing the MERVA ESA and SWIFT Link programs and JCL procedures. You can code a list of sublibraries.

nnn The unit number of the tape unit with the BIC tape. If the tape has two volumes, you can improve the performance by using two tape units. Specify the units as follows:

```
// ASSGN SYS025,nnn
// ASSGN SYS025,mmm,ALT
```

If you have only one tape unit, after unloading the first volume and mounting the second, you must enter the response NEWTAP to continue processing.

Note: When a disk device is used instead of the magnetic tape, the statement:

```
// ASSGN SYS025,nnn
```

must be replaced by the statements:

```
// ASSGN SYS025,DISK,...
// DLBL DWSDBIC,...
// EXTENT SYS025,...
```

using appropriate parameters. You must then also use a DWSCURUT control statement:

```
TAPE=DISK
```

Chapter 28. Maintaining the Authenticator-Key File

The Authenticator-Key File contains the authenticator keys that the SWIFT Link uses to authenticate the SWIFT messages.

You use the Authenticator-Key File Load program DWSAUTLD for:

- Formatting the Authenticator-Key File before it is used for the first time.
- Maintaining the Authenticator-Key File by:
  - Adding records to the Authenticator-Key File
  - Replacing existing records of the Authenticator-Key File
  - Listing the contents of the Authenticator-Key File
  - Deleting records from the Authenticator-Key File
  - Exchanging the authenticator keys in one record or a set of records, depending on the change date in the records.

  Note: Using DWSAUTLD for maintenance of the Authenticator-Key File always creates authorized records in the file. It is not possible to create unauthorized records, as it is during online maintenance, described in the *MERVA for ESA User's Guide*.
- Unloading all or part of the Authenticator-Key File to a sequential data set. You do this when you want to save the file or change the size of the file.
- Reloading the complete Authenticator-Key File, or a part of it, from a sequential data set previously created by a DWSAUTLD unload operation.

When you use DWSAUTLD, you must specify the function to be carried out in the JCL.

Calling the DWSAUTLD Program

When you call the DWSAUTLD program, you must specify what it has to do by using one of the following function keywords in the call:

LOAD
The Authenticator-Key File is initialized or updated, or both, on a DASD. You update the file with the DWSAUTLD input records.

You use this function keyword to initialize and load the Authenticator-Key File, and to maintain the records in the file.

UNLOAD
Unload all or part of the Authenticator-Key File from direct-access storage to a sequential data set.

You use this keyword to unload the Authenticator-Key File.

RELOAD
Load a sequential data set, created by the UNLOAD function, to the Authenticator-Key File.

You use this keyword to reload the Authenticator-Key File.

MVS Calls: You specify the function keyword in the PARM parameter of the JCL EXEC statement for DWSAUTLD as shown in the following:

PARM='*function* [,KEYS]'

Where:

function
Is one of the function keywords described here (LOAD, UNLOAD, RELOAD).

KEYS
Specifies that the reports from DWSAUTLD are to contain the keys. Otherwise, the keys are suppressed. A comma separates KEYS from the *function* parameter.

Note: Only paper-based keys are displayed. Keys exchanged by USE workstation (BKE) are not displayed.

VSE Calls: You specify the function keyword, and the device for unloading or reloading, in the PARM parameter of the JCL EXEC statement for DWSAUTLD. An example of the format of this statement is shown in the following:

PARM='*function*,[*device*][,KEYS]'

Where:

function
Is one of the function keywords described here (LOAD, UNLOAD, RELOAD).

You need not specify *device* if the *function* is LOAD.

device Is one of the following:

DISK The unload or reload function uses a DASD for the sequential data set.

TAPN The unload or reload function uses an unlabeled tape for the sequential data set.

TAPL The unload or reload function uses a labeled tape as the sequential data set.

KEYS Specifies that the reports from DWSAUTLD are to contain the keys. Otherwise, the keys are suppressed. If you specify KEYS but no device, you must indicate the omission by two commas: '*function*,,KEYS'.

Types of Input Record Used by DWSAUTLD

DWSAUTLD uses an input data set with records of 80 bytes fixed length. This section describes the types of input record summarized in Table 1.

Table 1. DWSAUTLD Summary

Type of Input Record	Function		
	LOAD	UNLOAD	RELOAD
FORMATx	X		X
ADDxx	X		
REPxx	X		
LIS0	X		

Table 1. DWSAUTLD Summary (continued)

Type of Input Record	Function		
	LOAD	UNLOAD	RELOAD
DEL0	X		
EXC0	X		
UNL0		X	
CHG0			X

Note: The home and correspondent LTs in the input records can be for a group and can contain two or four asterisks. For example:

- All financial institutions with the same bank code, but different country and location codes, can share a record with the address BANK****.
- All financial institutions with the same bank code and country code, but with a different location code, can share a record with the address BANKCC**.

The two forms of group address, and the form of address with a specific country and location code can be used in any combination.

FORMATx Records

The FORMATx input records cause DWSAUTLD to create a Version Record in the Authenticator-Key File with a scrambled STK value. This STK is used to scramble the bilateral keys.

The STK comes in two parts, and is generated by the User Key Management Officer on the USE workstation. A FORMAT1 input record is used for the first STK part, and a FORMAT2 input record for the second STK part.

If you are running MERVA ESA without a USE workstation, a dummy STK can be generated using the FORMAT input record.

Note: The FORMAT1 and FORMAT2 operations can be run in separate DWSAUTLD jobs, if you have two User Key Management Officers.

The layout of the records in the group is as follows:

FORMAT Record, for non-USE workstation users

Byte	Content
1-6	FORMAT keyword
7	Blank separator.

FORMAT1-FORMAT2 Records, for USE workstation users

Byte	Content
1-7	Keyword: FORMAT1 or FORMAT2
8	Leave blank
9-40	STK part (STK1 or STK2)
41	Blank or comma separator
42-45	STK part check value (STK1-KCV or STK2-KCV)

46	Blank separator.

ADDxx Records

Use the ADDxx records to add a record to the Authenticator-Key File. The ADD0 record must be followed by at least one key record (ADD1S-ADD3R). The ADD1S-ADD3R records must be in the order ADD1S, ADD2S, ADD3S, ADD1R, ADD2R, and ADD3R.

The layout of the records in the group is as follows:

ADD0 Record

Byte	Content
1-4	ADD0 keyword
5	Leave blank
6-13	Home LT
14-16	Blanks
17	Blank or comma separator
18-25	Correspondent LT
26-28	Blanks
29	Blank or comma separator
30-35	Start date
36	Blank or comma separator
37-42	End date
43	Blank or comma separator
44-49	Suspension date
50	Blank or comma separator
51	Correspondent status
52	Blank.

ADD1S-ADD3R Record

Byte	Content
1-5	Keyword: ADD1S, ADD2S, ADD3S, ADD1R, ADD2R or ADD3R.
6	Leave blank
7-38	Authenticator key
39	Blank or comma separator
40-45	From date
46	Blank or comma separator
47-50	From time
51	Blank or comma separator
52-57	To date
58	Blank or comma separator

59-62 To time

63 Blank separator.

An example of ADDxx records is shown in Figure 139.

```
ADD0 HOMEBANK    CORRBANK    <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
ADD1S 0123456789ABCDEF0123456789ABCDEF,980101,0001,980701,0001 <<<<<<<<<<<<<<<<<
ADD2S 1023456789ABCDEF0123456789ABCDEF,980701,0001,980101,0001 <<<<<<<<<<<<<<<<<
ADD3S 2013456789ABCDEF0123456789ABCDEF,980101,0001,980701,0001 <<<<<<<<<<<<<<<<<
ADD1R 0123456789ABCDEF0123456789ABCDEF,980101,0001,980701,0001 <<<<<<<<<<<<<<<<<
ADD2R 1023456789ABCDEF0123456789ABCDEF,980701,0001,980101,0001 <<<<<<<<<<<<<<<<<
ADD3R 2013456789ABCDEF0123456789ABCDEF,990101,0001,990701,0001 <<<<<<<<<<<<<<<<<
----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
```

Figure 139. Example of ADDxx Records

Note: The "<" character is used in this figure to indicate characters in the input records that are ignored.

REPxx Records

Use the REPxx records to replace a record in the Authenticator-Key File. The REP0 record must be followed by at least one key record (REP1S-REP3R). The REP1S - REP3R records must be in the order REP1S, REP2S, REP3S, REP1R, REP2R, and REP3R.

If the key record (REP1S-REP3R) is omitted, the key for that record remains unaltered in the Authenticator-Key File record. If a key record (REP1S-REP3R) is supplied without a key, the key for that record is deleted from the Authenticator-Key File record.

The layout of the REPxx records is the same as the ADDxx records, except that bytes 1 to 3 contain the keyword REP.

Note: If the old record is not found because all the records in the Authenticator-Key File have been deleted, using a DEL0 without any parameters (or by a FORMAT record), DWSAUTLD carries out an ADD.

LIS0 Record

The LIS0 record prepares a list of Authenticator-Key File records depending on the specification of the home and correspondent LTs. The authenticator keys are printed only if the KEYS parameter is used in the JCL EXEC PARM (see "Calling the DWSAUTLD Program" on page 333).

The layout of the LIS0 record is as follows:

Byte	Content
1-4	LIS0 keyword
5	Blank separator
6-13	Home LT
14-16	Blanks
17	Blank or comma separator
18-25	Correspondent LT

26-28 Blanks

29 Blank separator

30-36 Status selector

37 Blank or comma separator

38-43 Date

44 Blank or comma separator

45-46 Relationship

47 Blank or comma separator

48-54 Type

55 Blank separator.

You can specify the home LT in one of the following ways:

- Enter blanks.

  All Authenticator-Key File records that match the specified correspondent LT and record selectors are listed.
- Specify the first 4 characters of the LT (the bank code) followed by four blanks.

  The list shows all the Authenticator-Key File records with the specified bank code that match the specified correspondent LT and record selectors.
- Specify a full home LT.

  The list shows all the Authenticator-Key File records for the specified home LT that match the specified correspondent LT and record selectors.

You can specify the correspondent LT in one of the following ways:

- Enter blanks.

  The list shows all Authenticator-Key File records that match the specified home LT and record selectors.
- Specify the first 4 characters of an LT (the bank code) followed by four blanks.

  The list shows all Authenticator-Key File records with the specified bank code that match the specified home LT and record selectors.
- Specify a full correspondent LT.

  The list shows all Authenticator-Key File records for the specified correspondent LT that match the specified home LT and record selectors.

If both home and correspondent LTs are specified as blanks, the list shows all records of the Authenticator-Key File that match the specified record selectors.

The record status selector is a keyword used to limit the list to records with a particular status. It has the following values:

ADD Limits the list to records that are ADD PENDING

DELETE Limits the list to records that are DELETE PENDING

REPLACE Limits the list to records that are REPLACE PENDING

AUTH Limits the list to records that contain authorized data

UNAUTH Limits the list to records that contain unauthorized data

PENDING Limits the list to records that are ADD, DELETE or REPLACE PENDING

ALL The list shows all records matching the home and correspondent's LTs specified. This is the default parameter.

The exchange date selector keywords (*date*, *rel*, *type*) are used to limit the list to records with particular exchange dates (3rd FROM dates):

date This is the date for the selection test (*YYMMDD*).

rel This specifies a relationship between the records to be listed and the date given in this record selector:

EQ Limits the list to records whose change dates match the date specified.

LT Limits the list to records whose change dates are earlier than the date specified (excluding records with change dates of zero).

LE Limits the list to records whose change dates are earlier than or the same as the date specified (excluding records with change dates of zero).

GT Limits the list to records whose change dates are later than the date specified.

GE Limits the list to records whose change dates are later than or the same as the date specified.

type Indicates which change date(s) on a record should be examined when compiling the list to be printed. The default value is BOTH:

BOTH Causes a record to be included in the list if either the new sending key, or the new receiving key, has a change date that matches the *rel* specified.

SEND Causes a record to be included in the list if the new sending key has a change date that matches the *rel* specified.

RECEIVE Causes a record to be included in the list if the new receiving key has a change date that matches the *rel* specified.

Note: The dates are compared with the FROM date of the third keys in the authorized area, unless the **list** command also has a status selector of REPLACE or UNAUTH. In that case, the dates are compared with the unauthorized area.

DEL0 Record

The DEL0 record deletes one or more records of the Authenticator-Key File, depending on the specification of the home and correspondent LTs.

The layout of the DEL0 record is as follows:

Byte	Content
1-4	DEL0 keyword
5	Blank separator
6-13	Home LT
14-16	Blanks
17	Blank or comma separator

18-25 Correspondent LT

26-28 Blanks

29 Blank separator.

The home and correspondent LTs are specified in the same way as for the LIS0 record.

EXC0 Record

The EXC0 record replaces the first keys with the second keys, the second keys with the third keys, and erases the third keys in one or more records of the Authenticator-Key File, depending on the specification of the home and correspondent LTs and the date of the EXC0 record.

The layout of the EXC0 record is as follows:

Byte **Content**

1-4 EXC0 keyword

5 Blank separator

6-13 Home LT

14-16 Blanks

17 Blank or comma separator

18-25 Correspondent LT

26-28 Blanks

29 Blank or comma separator

30-35 Date in the format *YYMMDD* (year, month, day)

36 Blank separator

37-43 Type of exchange

44 Blank separator.

The home and correspondent LTs are specified in the same way as for the LIS0 record.

You must specify a date. DWSAUTLD carries out the exchange on an Authenticator-Key File record if the third FROM date in the record is earlier than or equal to the date specified in the EXC0 record.

You can specify a type of exchange that has the same values as the *type* in the LIS0 record, and indicates if both sending and receiving keys should be exchanged (BOTH), or only sending (SEND) or receiving (RECEIVE) keys. The default value is BOTH.

UNL0 Record

You use the UNL0 record to select, for the UNLOAD function, which Authenticator-Key File records you want to unload to the sequential file. The selection depends on the home LT specification.

The layout of the UNL0 record is as follows:

Byte **Content**

1-4	UNL0 keyword
5	Blank separator
6-13	Home LT
14-16	Blanks
17	Blank separator.

You specify the home LT as described for the LIS0 record. For each home LT you want to unload, you must provide a UNL0 record. If no UNL0 record is found, or if the home LT in the UNL0 record is blank, DWSAUTLD unloads all Authenticator-Key File records.

CHG0 Record

You use the CHG0 record to change the Home LT during the RELOAD function.

The layout of the CHG0 record is as follows:

Byte	Content
1-4	CHG0 keyword
5	Blank separator
6-13	Old Home LT
14-16	Blanks
17	Blank or comma separator
18-25	New Home LT
26-28	Blanks
29	Blank separator.

The old and new home LTs must be a full LT (8 characters).

Only one CHG0 card is allowed.

Functions from Previous Versions

The ADD, ADD1-ADD4, DEL, EXC, FORMAT, KEY, LIS, REP, REP1-REP4, and UNL records from previous versions are still supported for compatibility reasons.

Changing the STK Key

The following jobs must be run to change the STK key in the authenticator-key file:

1. Run DWSAUTLD UNLOAD.
2. Run DWSAUTLD RELOAD, with input cards FORMAT1 and FORMAT2.

Job Control Statements for MVS

The job-control statements you use to load, unload, and reload the Authenticator-Key File using MVS, are described here. Samples of printer reports that result from loading, unloading, and reloading the Authenticator-Key File can be found in Figure 146 on page 348.

Loading the Authenticator-Key File

Figure 140 shows the job-control statements you use to load the Authenticator-Key File from DWSAUTLD input data records.

```
//.......  JOB   ...........
//JOBCAT   DD DSN=ucat,DISP=SHR             VSAM USER CAT
//LOAD     EXEC PGM=DWSAUTLD,PARM='LOAD,KEYS'
//STEPLIB  DD DSN=loadlib,DISP=SHR
//SYSUDUMP DD SYSOUT=A
//DWSAUTP  DD SYSOUT=A
//DWSAUTD  DD DSN=authfile,DISP=SHR
//DWSAUTR  DD *
ADD0 ...          (refer to section
ADD1S ...          "Types of Input Record Used by DWSAUTLD" on page 334)
ADD2S ...
ADD3S ...
REP0 ...
REP2R ...
REP3R ...
LIS0 ...
 .
 .
/*
```

Figure 140. Loading the Authenticator-Key File in MVS

In the JCL, the lowercase parameters have the following meaning:

ucat The VSAM user catalog where the Authenticator-Key File is cataloged. In some MVS installations, this statement may not be required.

loadlib The name of the load library containing the MERVA ESA and SWIFT Link programs.

authfile The name of the Authenticator-Key File.

Unloading the Authenticator-Key File

You use the job-control statements shown in Figure 141 to unload the Authenticator-Key File to a sequential data set.

The unload data set has a block size of 6504 bytes.

```
//.......  JOB   ...........
//JOBCAT   DD DSN=ucat,DISP=SHR             VSAM USER CAT
//UNLOAD   EXEC PGM=DWSAUTLD,PARM='UNLOAD'
//STEPLIB  DD DSN=loadlib,DISP=SHR
//SYSUDUMP DD SYSOUT=A
//DWSAUTP  DD SYSOUT=A
//DWSAUTD  DD DSN=authfile,DISP=SHR
//DWSAUTUR DD DSN=unloadfile,DCB=BLKSIZE=6504,VOL=SER=volume,
//             DISP=OLD,UNIT=TAPE,LABEL=(1,SL)
//DWSAUTR  DD  *
UNL0 ...      (refer to section
UNL0 ...       "Types of Input Record Used by DWSAUTLD" on page 334)
/*
```

Figure 141. JCL to Unload the Authenticator-Key File in MVS

In the JCL, the lowercase parameters have the following meaning:

ucat The VSAM user catalog where the Authenticator-Key File is cataloged. In some MVS installations, this statement may not be required.

loadlib The name of the load library containing the MERVA ESA and SWIFT Link programs.

authfile The name of the Authenticator-Key File.

unloadfile The name of the unloaded Authenticator-Key File. The example shows the DD statement parameters for an unload file on a magnetic standard label tape. For unlabeled tape or a disk file, the parameters are different. However, you must specify the DCB=BLKSIZE=6504 parameter as shown.

volume The volume identification of the tape that is to contain the unloaded Authenticator-Key File.

Reloading the Authenticator-Key File

Figure 142 shows the job-control statements you use to reload the Authenticator-Key File from the sequential data set.

```
//.......  JOB   ...........
//JOBCAT   DD DSN=ucat,DISP=SHR             VSAM USER CAT
//RELOAD   EXEC PGM=DWSAUTLD,PARM='RELOAD'
//STEPLIB  DD DSN=loadlib,DISP=SHR
//SYSUDUMP DD SYSOUT=A
//DWSAUTP  DD SYSOUT=A
//DWSAUTD  DD DSN=authfile,DISP=SHR
//DWSAUTUR DD DSN=reloadfile,DCB=BLKSIZE=6504,VOL=SER=volume,
//            DISP=OLD,UNIT=TAPE,LABEL=(1,SL),
//DWSAUTR   DD *
CHG0 ...          (refer to section
/*                 "Types of Input Record Used by DWSAUTLD" on page 334)
```

Figure 142. JCL to Reload the Authenticator-Key File in MVS

In the JCL, the lowercase parameters have the following meaning:

ucat The VSAM user catalog where the Authenticator-Key File is cataloged. In some MVS installations, this statement may not be required.

loadlib The name of the load library containing the MERVA ESA and SWIFT Link programs.

authfile The name of the Authenticator-Key File.

reloadfile The name of the unloaded Authenticator-Key File. The example shows the DD statement parameters for an unload file on a magnetic standard label tape. For unlabeled tape or a disk file, the parameters are different. However, the DCB=BLKSIZE=6504 parameter must be specified as shown, except when migrating from VSE to MVS. Then the parameter DCB=BLKSIZE=6500 must be specified.

volume The volume identification of the tape that contains the unloaded Authenticator-Key File.

Job Control Statements for VSE

The job-control statements you use to load, unload, and reload the Authenticator-Key File using VSE are described here. Samples of printer reports that result from loading, unloading, and reloading the Authenticator-Key File can be found in Figure 146 on page 348.

Loading the Authenticator-Key File

Figure 143 shows the job-control statements you use to load the Authenticator-Key File from DWSAUTLD input data records.

```
// JOB     ...
// DLBL USERCAT,'ucat',,VSAM                    VSAM USER CAT
// DLBL DWSAUTD,'authfile',,VSAM                AUTHENTICATOR-KEY FILE
// EXTENT SYS005,volid
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION DUMP
// EXEC DWSAUTLD,SIZE=300K,PARM='LOAD,,KEYS'
ADD0 ...          (refer to section
ADD1S ...          "Types of Input Record Used by DWSAUTLD" on page 334)
ADD2S ...
ADD3S ...
REP0 ...
REP2R ...
REP3R ...
LIS0 ...
 .
 .
/*
/&
```

Figure 143. JCL to Load the Authenticator-Key File in VSE

In the JCL, the lowercase parameters have the following meaning:

ucat The name of the VSAM user catalog.

authfile The name of the Authenticator-Key File.

volid The volume identification of the particular data set.

program library
The name of the library containing the MERVA ESAproduct.

library.sublib The names of the program libraries containing the MERVA ESA executable programs and JCL procedures. You can code a list of sublibraries.

Unloading the Authenticator-Key File

You use the job-control statements shown in Figure 144 on page 345 to unload the Authenticator-Key File to a sequential data set.

The unload data set has a block size of 6500 bytes.

```
// JOB     ...
// DLBL IJSYSUC,'ucat',,VSAM                    VSAM USER CAT
// DLBL DWSAUTD,'authfile',,VSAM                AUTHENTICATOR-KEY FILE
// ASSGN SYS025,DISK,VOL=volid,SHR
// DLBL DWSATUD,'unloadfile',99/365,SD
// EXTENT SYS025,volid,extent information
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION DUMP
// EXEC DWSAUTLD,SIZE=300K,PARM='UNLOAD,DISK'
UNL0 ...          (refer to section
UNL0 ...           "Types of Input Record Used by DWSAUTLD" on page 334)
/*
/&
```

Figure 144. JCL to Unload the Authenticator-Key File in VSE

In the JCL, the lowercase parameters have the following meaning:

ucat The name of the VSAM user catalog.

authfile The name of the Authenticator-Key File.

volid The volume identification of the particular data set.

unloadfile The name of the unloaded Authenticator-Key File. The example shows the DLBL parameters for an unload file on a disk device. Accordingly, the EXEC PARM specifies DISK as second parameter.

extent information
The extent information necessary for the unloaded Authenticator-Key File on disk.

Note: When a magnetic tape is used instead of the disk device, the following statements

```
// ASSGN SYS025,DISK,...
// DLBL DWSATUD,...
// EXTENT SYS025,...
```

must be replaced by the statements

```
// ASSGN SYS025,nnn
// TLBL DWSAUTL,...
```

when a standard label tape is used, and the EXEC PARM must then be PARM='UNLOAD,TAPL'; *nnn* is the address of the tape unit you use. For a nonlabeled tape, no TLBL statement is required, and the EXEC PARM must then be PARM='UNLOAD,TAPN'.

The /* statement must then be followed by the following statements:

```
// MTC WTM,SYS025,2
// MTC RUN,SYS025
```

program library
The name of the library containing the MERVA ESAproduct.

library.sublib The names of the program libraries containing the MERVA ESA executable programs and JCL procedures. You can code a list of sublibraries.

Reloading the Authenticator-Key File

You use the job-control statements shown in Figure 145 to reload the Authenticator-Key File from the sequential data set.

```
// JOB     ...
// DLBL IJSYSUC,'ucat',,VSAM                    VSAM USER CAT
// DLBL DWSAUTD,'authfile',,VSAM                AUTHENTICATOR-KEY FILE
// ASSGN SYS025,DISK,VOL=volid,SHR
// DLBL DWSATSD,'reloadfile',99/365,SD
// EXTENT SYS025,volid,extent information
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION DUMP
// EXEC    DWSAUTLD,SIZE=300K,PARM='RELOAD,DISK'
CHG0              (refer to section
/*                 "Types of Input Record Used by DWSAUTLD" on page 334)
/&
```

Figure 145. JCL to Reload the Authenticator-Key File in VSE

In the JCL, the lowercase parameters have the following meaning:

ucat The name of the VSAM user catalog.

authfile The name of the Authenticator-Key File.

volid The volume identification of the particular data set.

reloadfile The name of the unloaded Authenticator-Key File that is to be reloaded. The example shows the DLBL parameters for a reload file on a disk device. Accordingly, the EXEC PARM specifies DISK as second parameter.

extent information
The extent information necessary for the unloaded Authenticator-Key File on disk.

Note: When a magnetic tape is used instead of the disk device, the following statements

```
// ASSGN SYS025,DISK,...
// DLBL DWSATSD,...
// EXTENT SYS025...
```

must be replaced by the statements

```
// ASSGN SYS025,nnn
// TLBL DWSASTL,...
```

when a standard label tape is used, and the EXEC PARM must then be PARM='RELOAD,TAPL'; *nnn* is the address of the tape unit you use. For a nonlabeled tape, no TLBL statement is required, and the EXEC PARM must then be PARM='UNLOAD,TAPN'.

The /* statement should then be followed by the following statement:

```
// MTC RUN,SYS025
```

program library
The name of the library containing the MERVA ESA product.

library.sublib The names of the program libraries containing the MERVA ESA executable programs and JCL procedure. You can code a list of sublibraries.

Unloading the Authenticator-Key File to MERVA USE Workstation programs

You perform the following steps to unload the Authenticator-Key File to MERVA USE Workstation Programs:

1. Unload the Authenticator-Key File to a VSAM ESDS (Entry Sequenced Data Set).

   If there is a DEFAULT.MODEL.ESDS.SAM file defined in the catalog (default for standard VSE/ESA installation), you can let define the file automatically in VSAM managed space. Replace the statement

   ```
   // DLBL DWSATUD,...
   ```

   in the job shown in Figure 144 on page 345 by the statement

   ```
   // DLBL DWSATUD,'unloadfile',0,VSAM,RECSIZE=6500,RECORDS=200
   ```

   and run the job.

2. Use the ICCF IUI (Interactive User Interface) to move the VSAM ESDS to the HTF (Host Transfer File).

   A CICS file control table definition of the VSAM ESDS is required. You can find a sample FCT entry in the copy book DWSFCT21:

   ```
   DFHFCT TYPE=DATASET,                                    +
         DATASET=DWSATUD,                                  +
         ACCMETH=(VSAM,ADR),                               +
         RECFORM=(FIXED,BLOCKED),                          +
         SERVREQ=BROWSE,                                   +
         STRNO=10
   ```

   Add the following statement to your ICCF CICS startup job:

   ```
   // DLBL DWSATUD,'unloadfile',0,VSAM
   ```

3. Receive the file from the HTF on the USE workstation. For example, enter the following command on the PS/2:

   ```
   RECEIVE x:AUTHKEY.DAT DWSATUD (BINARY
   ```

   where *x* is the drive where the file is to be installed.

   Note: The BINARY option is mandatory.

4. Restore the Authenticator-Key File on the USE workstation via BK Backup/Restore.

   For detailed information refer to the *MERVA USE Administration Guide*.

   You need the password for scrambling authenticator keys before they were unloaded to the VSAM ESDS. The password is specified in parameter AUTHUP of macro DWSPARM. The default password is UNLOAD*bb* (*bb* represents padding blanks).

   You find a description of DWSPARM in the *MERVA for ESA Macro Reference*.

Examples of Reports

```
A U T H E N T I C A T O R   K E Y   F I L E   BATCH PROGRAM - DATE 07 JUN 1999 TIME 18:11:01
PROGRAM FUNCTION  =  L O A D
LIST OF THE READ UPDATE RECORDS :

ADD0 VNDEBET2    BARCGB22
ADD1S 980101 0001 980701 0001
ADD1R 980101 0001 980701 0001

ADD0 VNDEBET2    COBADEFF
ADD1S 980101 0001 980701 0001
ADD2S 980701 0001 980101 0001
ADD3S 990101 0001 990701 0001
ADD1R 980101 0001 980701 0001
ADD2R 980701 0001 980101 0001
ADD3R 990101 0001 990701 0001
   DWS847I Update causes record to contain a non-recommended key            ** WARNING **

REP0 VNDEBET2    COBADEFF
REP3S 991201 0001 990701 0001
REP3R 990101 0001 990701 0001
   DWS738I FROM date/time must equal previous key's EXPIRE date/time           ** ERROR **

EXC0 VNDEBET2    COBADEFF    980301

                      PENDING  <------------ DISPLAY OF AUTHORIZED AREA ------------------>
     HOME LT  CORR LT   STATUS                BILATERAL KEY ID FROM          TO
     VNDEBET2 COBADEFF          1ST SENDING                   19980101 0001 19980701 0001
                                2ND SENDING                   19980701 0001 19990101 0001
                                3RD SENDING                   19990101 0001 19990701 0001
                                1ST RECEIVING                 19980101 0001 19980701 0001
                                2ND RECEIVING                 19980701 0001 19990101 0001
                                3RD RECEIVING                 19990101 0001 19990701 0001

LIS0 VNDEBET2

                      PENDING  <------------ DISPLAY OF AUTHORIZED AREA ----------------->
     HOME LT  CORR LT   STATUS                BILATERAL KEY ID FROM          TO
     VNDEBET2 BARCGB22          1ST SENDING                   19980101 0001 19980701 0001
                                1ST RECEIVING                 19980101 0001 19980701 0001
     VNDEBET2 COBADEFF          1ST SENDING                   19980701 0001 19990101 0001
                                2ND SENDING                   19990101 0001 19990701 0001
                                1ST RECEIVING                 19980701 0001 19990101 0001
                                2ND RECEIVING                 19990101 0001 19990701 0001

ADD0 VNDEBET2   -DEUTDEFF
   DWS810I Incorrect separator after Home LT                                 ** ERROR **
ADD1S 980101 0001
   DWS845I TO date specification missing                                     ** ERROR **
   DWS853I TO time specification missing                                     ** ERROR **
```

Figure 146. Example of a LOAD Report (Part 1 of 2)

```
S T A T I S T I C S

NUMBER OF FUNCTIONS           - TOTAL              6
                              - ADD                3
                              - EXCHANGE           1
                              - LIST               1
                              - REPLACE            1

NUMBER OF WRONG UPDATE FUNCTIONS                   2

NUMBER OF RECORDS PROCESSED   - TOTAL              5
                              - ADDED              2
                              - EXCHANGED          1
                              - LISTED             2
```

Figure 146. Example of a LOAD Report (Part 2 of 2)

```
A U T H E N T I C A T O R   K E Y   F I L E   BATCH PROGRAM - DATE 07 JUN 1999 TIME 18:16:44
PROGRAM FUNCTION  =  U N L O A D

UNL0

                       PENDING  <------------ DISPLAY OF AUTHORIZED AREA ------------------>
     HOME LT  CORR LT  STATUS                 BILATERAL KEY ID FROM          TO
     VNDEBET2 BARCGB22           1ST SENDING                  19980101 0001 19980701 0001
                                 1ST RECEIVING                19980101 0001 19980701 0001
                                 CORR STATUS : VALID
                                 START DATE  :                19980301
                                 END DATE    :                20000301
     VNDEBET2 COBADEFF           1ST SENDING                  19980701 0001 19990101 0001
                                 2ND SENDING                  19990101 0001 19990701 0001
                                 1ST RECEIVING                19980701 0001 19990101 0001
                                 2ND RECEIVING                19990101 0001 19990701 0001

S T A T I S T I C S

NUMBER OF FUNCTIONS           - TOTAL              1
                              - UNLOAD             1

NUMBER OF WRONG UPDATE FUNCTIONS                   0

NUMBER OF RECORDS PROCESSED   - TOTAL              2
NUMBER OF USED SEQ. DATA BLOCKS                    1
```

Figure 147. Example of an UNLOAD Report

```
A U T H E N T I C A T O R   K E Y   F I L E   BATCH PROGRAM - DATE 07 JUN 1999 TIME 18:21:58
PROGRAM FUNCTION  =  R E L O A D

CHG0 VNDEBET2    VNDOSYN2

 LIST OF THE PROCESSED RECORDS :
                       PENDING  <------------ DISPLAY OF AUTHORIZED AREA ------------------>
     HOME LT  CORR LT   STATUS                BILATERAL KEY ID FROM          TO
     VNDEBET2 BARCGB22           1ST SENDING                   19980101 0001 19980701 0001
                                 1ST RECEIVING                 19980101 0001 19980701 0001
   DWS803I Home SWIFT LT changed to VNDOSYN2                                 ** WARNING **
     VNDEBET2 COBADEFF           1ST SENDING                   19980701 0001 19990101 0001
                                 2ND SENDING                   19990101 0001 19990701 0001
                                 1ST RECEIVING                 19980701 0001 19990101 0001
                                 2ND RECEIVING                 19990101 0001 19990701 0001
   DWS803I Home SWIFT LT changed to VNDOSYN2                                 ** WARNING **

S T A T I S T I C S

NUMBER OF FUNCTIONS           - TOTAL              1
                              - CHANGE             1

NUMBER OF WRONG UPDATE FUNCTIONS                   0

NUMBER OF RECORDS PROCESSED   - TOTAL              2
NUMBER OF USED SEQ. DATA BLOCKS                    1
```

Figure 148. Example of a RELOAD Report

Chapter 29. Using the Message Counter Log Report Utility DSLCNTUT

The message counter log report utility DSLCNTUT shows the current status of the MERVA ESA message counter log data set. The status of all individual counters for the last 12 months is printed. In addition, the status for the current month is shown. The output is equivalent to that which is created with the MERVA ESA operator commands **dclog detail** and **dclog last**.

MERVA ESA startup is not required to run the report utility.

The output is printed in the form of MERVA ESA operator messages. The operator messages are explained in *MERVA for ESA Messages and Codes*.

DSLCNTUT runs as a batch program under control of MVS or VSE.

Job Control Statements under MVS

Figure 149 shows the JCL to create a message counter log report under MVS.

```
//........ JOB ....
//REPORT   EXEC PGM=DSLCNTUT,REGION=2048K
//STEPLIB  DD DSN=merva.SDSLLOD0,DISP=SHR
//DSLCNTA  DD DSN=merva.DSLCNTA,DISP=SHR
//SYSPRINT DD SYSOUT=*
/*
//
```

Figure 149. Creating a Message Counter Log Report in MVS

In the JCL, the lowercase parameters have the following meanings:

merva The prefix of the MERVA ESA installation library.

merva.DSLCNTA
The name of the MERVA ESA message counter log data set.

Job Control Statements under VSE

Figure 150 shows the JCL to create a message counter log report under VSE.

```
// JOB ...
// DLBL IJSYSUC,'ucat',,VSAM
// DLBL DSLCNTA,'merva.DSLCNTA',,VSAM
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// EXEC DSLCNTUT,SIZE=200K
/*
//
```

Figure 150. Creating a Message Counter Log Report in VSE

In the JCL, the lowercase parameters have the following meanings:

ucat The name of the VSAM user catalog.

merva.DSLCNTA
The name of the MERVA ESA message counter log data set.

program library
The name of the library containing the MERVA ESA product.

volid The volume identification of the particular data set.

library.sublib The names of the program libraries containing the MERVA ESA programs and JCL procedures. You can code a list of sublibrary names.

Chapter 30. Using Queue Batch Utility (DSLSQB)

This chapter describes how to use the MVS Queue Batch Utility. This utility is not available in VSE.

The purpose of the Queue Batch Utility is to provide MERVA ESA queue operations via a batch program.

The following commands are available:

DELETE Delete messages from a MERVA ESA queue

MOVE Move messages from one MERVA ESA queue to another

ROUTE Route messages within MERVA ESA.

Message selection criteria can be specified for each command to process only selected messages.

Processing is controlled via control information in the SYSIN data set.

Starting the Queue Batch Utility

The Batch Utility is started as an MVS batch job. Figure 151 shows the JCL to run the program.

```
//......   JOB ......
//SQB      EXEC PGM=DSLSQB,PARM='   '
//STEPLIB  DD DSN=loadlib,DISP=SHR    MERVA ESA Loadlib
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//SYSIN    DD *
CONSTANTS
 ...
 ...
VARIABLES
 ...
 ...
SELECTION-SEL001
 ...
 ...
SELECTION-SEL002
 ...
 ...
OPERATION
 ...
 ...
/*
```

Figure 151. Running the Queue Batch Utility

SYSIN must be a sequential, fixed block 80 data set (PS, FB, LRECL=80) containing the control information.

Controlling the Utility Program

The Queue Batch Utility is controlled via data in the SYSIN data set with records of length 80, where the columns 73-80 are ignored.

Records starting with an asterisk (*) in column 1 are treated as comment records.

The control data set is separated into sections with no predefined order:

CONSTANTS
Defines constants that are used in SELECTION expressions. Several CONSTANTS sections are allowed. The section is optional.

VARIABLES
Defines variables that are used in SELECTION expressions. Several VARIABLES sections are allowed. The section is optional.

SELECTION
Defines a SELECTION expression. Each SELECTION must have a unique name. Only messages fulfilling the selection criteria are processed. The section is optional.

OPERATION
Defines operations to be done. Several OPERATION sections are allowed. This section is mandatory.

Figure 152 illustrates the structure of the control data set.

```
*
CONSTANTS

CHRCON1  CHR 'CONSTANT'
NUMBER1  NUM -1234,5671
TODAY    TIM #DAY
YESTRDAY TIM #DAY -1DAY
LASTHOUR TIM #HOUR-1HOUR
*
VARIABLES

SWAHMT   NUM FLD=(SWAHMT,,,,,VFISRT),LEN=(3)
SW20     CHR FLD=(SW20,,,,,VFIRST)
SW57DA1  CHR FLD=(SW57,,,,,VFIRST,FIRSTDA)
CREATED  TIM FLD=(MSGTRACE,,,,,VFIRST,FIRSTDA),LEN=(6,20)
*
SELECTION-SEL001
(SWAHMT = 100) AND (SW32AMNT>99)
*
SELECTION-SEL002
(SWAHMT = 100) AND
(SW32AMNT>12)  AND
NOT(SW57DA1 =* '*PAT?ERN*')
*
SELECTION-SEL003
(CREATED < YESTRDAY)
*
OPERATION

IMRSRVI  CMD DELETE,SEL001,,,,1,1
IMRSRVO  CMD MOVE,SEL002,IMRSRVE
IMRSRVO  CMD ROUTE,SEL001
```

Figure 152. Control Statements Example

The CONSTANTS Section

A CONSTANTS section defines constants that are used in SELECTION expressions. It starts with a record containing only the word CONSTANTS in columns 1-9. A record in a CONSTANTS section defines a constant by specifying:

- The constant name in columns 1-8
- The constant type in columns 10-12, either character (CHR), numeric (NUM), or timestamp (TIM)
- The constant value in columns 14-72:
  - Character data must be enclosed in quotes, and hexadecimal character data must be preceded by a 'X' additionally.
  - Numeric values can be signed. A comma or a point are treated as decimal point. The numbers are represented internally as floating point numbers.
  - Timestamps consist of a reference timestamp and a duration expression. The reference timestamp is either explicit or relative to the current timestamp.

    Explicit definition uses a 14-digit timestamp:

    YYYYMMDDHHMMSS
    Is a 14-digit timestamp to specifying the year, month, day, hour, minute, and second.

    Relative definition of reference timestamps is done by specifying one of the following keywords:

#YEAR	Represents the start of the current year
#MONTH	Represents the start of the current month
#DAY	Represents the start of the current day
#HOUR	Represents the start of the current hour
#MIN	Represents the start of the current minute.

    The reference timestamp is optional and defaults to the current timestamp if it is not specified.

    A duration is an integer followed by a unit:

DAY	Represents one day
HOUR	Represents one hour
MIN	Represents one minute.

    Several durations can be added or subtracted from the reference timestamp to form the final timestamp.

Notes:

1. Columns 9 and 13 must be blank.
2. No more than 64 constants can be defined.
3. Character and numeric constants can be defined inline in SELECTION expressions.
4. A constant name must not start with '@'.

Examples:

Refer to the number in parenthesis for an explanation:

```
*---+----1----+----2----+----3----+----4----+----5----+----6----+----7
CHRCON1  CHR 'CONSTANT'                      (1)
HEXCON1  CHR X'0102ABCDEF'                   (2)
NUMBER1  NUM -1234,5671                      (3)
NUMBER2  NUM 0,5                             (4)
TODAY    TIM #DAY                            (5)
YESTRDAY TIM #DAY -1DAY                      (6)
HOURAGO  TIM -1HOUR                          (7)
EXAMPLE  TIM -1HOUR + 3DAY - 2SEC            (8)
```

1. Defines a character constant.
2. Defines a hexadecimal character constant.
3. Defines a negative numeric constant.
4. Defines a positive numeric constant.
5. Defines a timestamp of today (start of today).
6. Defines a timestamp of yesterday.
7. Defines a timestamp of one hour ago.
8. Defines a complex timestamp.

The VARIABLES Section

A VARIABLES section defines variables used in SELECTION expressions. It is initiated by a record containing only the word VARIABLES in columns 1-9. A record in the VARIABLES section defines one variable by specifying:

- The variable name in columns 1-8.
- The variable type in columns 10-12, either character (CHR), numeric (NUM), or timestamp (TIM).
- The TOF field reference in columns 14-72. This comprises an FLD parameter followed optionally by a LEN parameter. The FLD parameter consists of up to seven positional subparameters, where only the name of the TOF field is mandatory:
  1. Name of the TOF field
  2. Nesting level
  3. Field group
  4. Repeatable sequence index
  5. Data area index
  6. First TOF MODIF parameter
  7. Second TOF MODIF parameter.

  The FLD parameter can be followed immediately by the LEN parameter, which consists of two positional subparameters:
  1. Length of data. The default is the actual length of the data in the TOF.
  2. Offset into the data. Default is zero.

  If the actual field data is too short for the LEN specification, the variable is considered to be empty.

  If no LEN parameter is specified, the length is determined by the actual length of the data in the TOF.

Notes:

1. Columns 9 and 13 must be blank.
2. The option of a field can be accessed by specifying the keyword OPTION as a TOF MODIF subparameter.

3. Numeric data is converted to floating point numbers for processing.
4. Timestamp data is internally expanded to a 14-digit timestamp. For example, the following timestamp definition is allowed:

   ```
   CREATED  TIM FLD=(MSGTRACE,,,,,VFIRST,FIRSTDA),LEN=(6,20)
   ```

   In this case the 6-digit timestamp YYMMDD from the MSGTRACE field is expanded into a 14-digit timestamp YYYYMMDD000000.
5. No more than 64 variables can be defined.
6. A variable name must not start with '@'.

Examples:

Refer to the number in parenthesis for an explanation:

```
*---+----1----+----2----+----3----+----4----+----5----+----6----+----7
SWAHMT   NUM FLD=(SWAHMT,,,,,VFIRST),LEN=(3)              (1)
SW20     CHR FLD=(SW20,,,,,VFIRST)                        (2)
SW32AMNT NUM FLD=(SW32AMNT,,,,,VFIRST)                    (3)
SW57DA1  CHR FLD=(SW57,,,,,VFIRST,FIRSTDA)                (4)
SW57DA2  CHR FLD=(SW57,,,,,NEXTDA)                        (5)
CREATED  TIM FLD=(MSGTRACE,,,,,VFIRST,FIRSTDA),LEN=(6,20) (6)
SW32OPT  CHR FLD=(SW32,,,,,VFIRST,OPTION)                 (7)
```

1. Defines numeric SWIFT message type.
2. Defines TRN of SWIFT message.
3. Defines numeric amount in field SW32.
4. Defines first line of F57.
5. Defines second line of F57.
6. Defines timestamp of message creation.
7. Defines the option of field F32.

The SELECTION Section

A SELECTION section defines selection criteria for messages to be processed. It is initiated by a record containing:

- The word SELECTION in columns 1-9
- A minus sign '-' in column 10
- The SELECTION name in columns 11-18.

The records within a SELECTION section define in free format a boolean expression built of logical operations (OR, AND, NOT), relational expressions (for example, =, <, >), certain predicates (EMPTY, EXIST, NOTFD), and parenthesis for control of precedence.

Assume [B], [C] denote boolean expressions, [R] a relational expression, and [V], [W] variables or constants. Formally, a boolean expression can be:

([R])	Relational expression in parenthesis
([B])	Boolean expression in parenthesis
[B] AND [C]	Boolean expressions combined with the AND operator
[B] OR [C]	Boolean expressions combined with the OR operator
NOT([B])	Negation of a boolean expression
EMPTY([V])	TRUE if variable [V] is empty

NOTFD([V]) TRUE if variable [V] is not in the TOF

EXIST([V]) TRUE if variable [V] is in the TOF.

A relational expression can be:

[V] = [W] TRUE if value of [V] equals value of [W]

[V] <> [W] TRUE if value of [V] is different from value of [W]

[V] < [W] TRUE if value of [V] is less than value of [W]

[V] > [W] TRUE if value of [V] is greater than value of [W]

[V] <=[W] TRUE if value of [V] is not greater than value of [W]

[V] >= [W] TRUE if value of [V] is not less than value of [W]

[V] =* [W] TRUE if value of [V] matches pattern value of [W].

Both [V] and [W] must be of type CHR. [W] is treated as a pattern where an asterisk (*) represents any string and a question mark (?) represents any character.

Notes:

1. The types of the variables or constants in a relational expression must be the same.
2. All variables and constants used in a SELECTION must be defined previously in a VARIABLES or CONSTANTS section.
3. No more than 32 SELECTION sections can be defined.
4. A message is selected for processing only if the boolean expression is TRUE for the message.
5. Character and numeric constants can be defined inline in expressions. Timestamps must be explicitly defined in VARIABLES or CONSTANTS sections.

Examples:

Refer to the number in parenthesis for an explanation:

```
*---+----1----+----2----+----3----+----4----+----5----+----6----+----7
SELECTION-SEL001                                (1)
(SWAHMT = 100) AND (SW32AMNT>1000)

SELECTION-SEL002                                (2)
(SWAHMT = 100) AND
(SW32AMNT>500000)  AND
NOT(SW57DA1 =* '*PAT?ERN*')

SELECTION-SEL003                                (3)
(CREATED < YESTRDAY)
```

The examples use the constant and variable definitions above:

1. The SELECTION named SEL001 selects all MT 100 with an amount greater than 1000.
2. The SELECTION named SEL002 selects all MT 100 with an amount greater than 500000 and where the first line of field 57 does not match '*PAT?ERN*'.
3. The SELECTION named SEL003 selects all messages created before yesterday.

The OPERATION Section

An OPERATION section specifies commands for the Batch Utility. It starts with a record containing only the word OPERATION in columns 1-9. Each record in the OPERATION section defines an operation:

- Columns 1-8 specify a MERVA queue name or a set of queues by a queue name pattern containing question marks (?) as the wildcard character.
- Columns 10-12 contain CMD.
- Columns 14-72 define the operation to be applied to the MERVA queues. It comprises seven positional parameters and:
  1. Defines the operation, either DELETE, MOVE, or ROUTE.
  2. Names a SELECTION. If no SELECTION is specified, all messages within the QSN range defined by subparameters 4-7 are selected.
  3. Specifies a target queue for MOVE or ROUTE.
  4. Specifies the low QSN of the QSN range. Default is zero.
  5. Specifies the high QSN of the QSN range. Default is actual last QSN.
  6. Specifies the number of messages to be skipped at the beginning of the QSN range. Default is zero.
  7. Specifies the number of messages to be inspected for selection. By default, all messages in the QSN range are inspected for selection.

Notes:

1. Columns 9 and 13 must be blank.
2. Only messages with a QSN in the specified QSN range are inspected for selection.
3. If a target queue is specified for a ROUTE operation, the selected messages are routed by the routing module of the target queue. If no target queue is specified, the routing is done via the routing module of the queue containing the message.
4. If the command record specifies a set of MERVA ESA queues, the operation is applied sequentially to all source queues. The sequence is defined by the sequence of the queues in the function table.
5. No more than 32 commands can be defined.

Examples:

Refer to the number in parenthesis for an explanation:

```
*---+----1----+----2----+----3----+----4----+----5----+----6----+----7
IMRSRVI  CMD DELETE,SEL001,,,,,10          (1)
IMRSRVO  CMD MOVE,SEL002,IMRSRVE           (2)
IMRSRVO  CMD ROUTE,SEL001                  (3)
IMRSRVO  CMD ROUTE,SEL001,TARQUE,5,10      (4)
RTV????? CMD DELETE,SEL003                 (5)
```

The examples use the SELECTION definitions above:

1. Delete up to 10 messages from queue IMRSRVI. Only the first ten messages are inspected. Messages fulfilling the SELECTION criteria are deleted.
2. Move all messages selected by SELECTION SEL002 from IMRSRVO to IMRSRVE.
3. Route all messages selected by SELECTION SEL001 from IMRSRVO using the routing module of IMRSRVO.
4. Route all messages with QSNs in QSN range 5-10 selected by SELECTION SEL001 from IMRSRVO using the routing module of TARQUE.

5. Delete all messages selected by SELECTION SEL003 from all queues with a prefix RTV.

Processing Details

- If a SELECTION boolean expression is specified for a command, this expression is evaluated for the messages in the source queue. If the expression is TRUE, the message is processed according to the command. Otherwise, the message is skipped and left in the source queue.
- The evaluation of the expression is done from left to right. There is no difference in precedence between the AND and OR operators. Precedence can be controlled by parenthesis.
- Variable data is accessed when it is needed, and it is accessed only once. For example, in the expression

  ```
  (SWAHMT > 100) AND (SWAHMT < 200)
  ```

  the variable is accessed for the comparison (SWAHMT > 100). For the second comparison (SWAHMT < 200), the previously accessed value is inspected. To force a reaccess of a TOF field, another variable with the same definition values must be declared:

  ```
  VARMT    NUM FLD=(SWAHMT,,,,,VFIRST),LEN=(3)
  VARMTX   NUM FLD=(SWAHMT,,,,,VFIRST),LEN=(3)

  (VARMT > 100) AND (VARMTX < 200)
  ```

  In this case the TOF field SWAHMT is accessed twice.
- If the left operand of an AND operation is FALSE, the right operand is not evaluated any more because the result must be FALSE.

  If the left operand of an OR operation is TRUE, the right operand is not evaluated any more because the result must be TRUE.

  This optimization may result in skipping the access of TOF fields and therefore may lead to unintended access of TOF fields if relative positioning is used. The following example illustrates this:

  ```
  DA1      CHR FLD=(SW57,,,,,VFIRST,FIRSTDA)
  DA2      CHR FLD=(SW57,,,,,NEXTDA)
  DA3      CHR FLD=(SW57,,,,,NEXTDA)

  ((DA1 =* '*A*') AND (DA2 =* '*B*')) OR (DA3 =* '*C*')
  ```

  The variables DA2 and DA3 use relative positioning to access data.

  If DA1 matches '*A*', DA2 is accessed via NEXTDA. It accesses the second data area.

  If DA1 does not match '*A*', the second part is not evaluated (DA2=*'*B*'), but the third part (DA3=*'*C*'). Therefore DA3 will access data area 2 of SWIFT field SW57. This may not be the intended data area to be accessed.

Tracing the Queue Batch Utility

The processing of the Queue Batch Utility can be traced using the MERVA ESA supplied MFS and TOF traces. The trace data is written to the SYSPRINT data set.

The traces are activated by switches in the MVS PARM data:

- The MFS trace is activated by the character 'T' as first character of the PARM data.
- The TOF trace is activated by the character 'T' as second character of the PARM data.
- The TOF access trace during evaluation of SELECTION expressions is activated by the character 'T' as third character of the PARM data.

Examples:

```
EXEC PGM=DSLSQB,PARM='T'     (1)
EXEC PGM=DSLSQB,PARM=' T'    (2)
EXEC PGM=DSLSQB,PARM='TT'    (3)
EXEC PGM=DSLSQB,PARM='  T'   (4)
```

Statement (1) starts the Queue Batch Utility with MFS trace, statement (2) starts it with TOF trace, and statement (3) with MFS and TOF trace. Statement (4) requests tracing of TOF access during evaluation of SELECTION expressions.

Chapter 31. More Batch Utilities

A number of further batch utilities written in REXX are distributed with MERVA ESA.

Queue Data Set Utilities

DSLBA12R Print a specified or all queue elements of a queue.

DSLBA14R Scanning a message TOF to display the TOF structure.

DSLBA50R Print queue status list.

DSLBA51R Print queue key list.

DSLBA52R Copy or move messages from one queue to another. Optionally sort them by key value.

DSLBA53R Scan a queue for 'old' messages.

Journal Utilities

DSLBA13R Print the MERVA ESA journal.

User File Utilities

DSLBA15R Print the MERVA ESA User file.

DSLBA16R Print a cross-reference of function names and allowed user IDs from the User file.

DSLBA17R Check the User file date fields.

All these batch utilities are described in the *MERVA for ESA Application Programming Interface Guide*.

Part 4. Reacting to Abnormal Events

This part provides help in dealing with abnormal events that can occur in operating MERVA ESA and the network links.

Chapter 32. How to React to MERVA ESA Problems

This chapter describes the problems that may arise during the running of MERVA ESA.

General Problems

Whenever an abnormal situation arises, MERVA ESA displays an error message on the operating-system console. You should refer to *MERVA for ESA Messages and Codes* to see what action to take.

If a dump is available (indicated in the explanation of the error message), you should print the dump.

Under CICS, the dump is contained in the CICS dump data set. This data set can be printed using a job described in the appropriate CICS manual.

Under IMS, the dump is printed in the SYSOUT class; or is written to the device specified by the DSLSNAP DD statement, or by the SYSUDUMP or SYSABEND DD statement. These DD statements should always be present in the job stream for the MERVA ESA MPP and BMP.

Under IMS there may be dumps with user-abend codes instead of system-abend codes. These dumps are always taken by IMS. You can find the meanings of the codes in *IMS/ESA Messages and Codes*.

You should be familiar with the data security procedures for your financial institution.

If SWIFT messages are lost or partially destroyed, they can be retrieved from SWIFT as described in the *S.W.I.F.T. User Handbook*.

Problems with Signing On to MERVA ESA

When MERVA ESA runs under IMS, a problem can arise if a user terminates a MERVA ESA session by turning off the terminal without signing off. The next user to use that terminal continues the interrupted MERVA ESA session.

This problem does not arise when MERVA ESA runs under IMS/RACF, because MERVA ESA verifies all users by their IMS user identification. If one user terminates the session without signing off, the next user to use the terminal is rejected, and MERVA ESA signs off the previous user.

If a Transaction Is in a Wait State

If MERVA ESA stops while a transaction is running, the transaction may remain in a wait state because intertask communication no longer functions. The transaction must be canceled using CICS or IMS commands.

If a Batch Program Is in a Wait State

If MERVA ESA stops while a batch program is running, the program may remain in a wait state because intertask communication no longer functions. The batch program must be canceled using operating system commands.

After a restart of MERVA ESA, the batch program can be started again. No data is lost, because the batch programs determine whether the transfer of data was completed or not. If the transfer was not completed, the batch program automatically carries out a restart and completes the transfer of data.

If the Queue Data Set Is Full

If a MERVA ESA batch program detects that the queue data set is full, or that all entries in the Queue Key Table have been used, the program ends without completing its processing. You should ask users to process the messages in the queues; or you should start the hard-copy printer program, or start DSLSDO (if DSLSDO did not previously detect the condition) or DSLSDY to empty queues.

When the shortage is relieved, the interrupted batch program must be started again to complete the transfer of data.

If these conditions appear frequently, the size of the queue data set must be increased using the MODIFY function of the MERVA ESA queue data set utility DSLQDSUT. See "Chapter 19. Using the Queue Data Set Utility DSLQDSUT" on page 275 for information on this utility. Alternatively, the number of entries in the Queue Key Table must be increased before the next startup of MERVA ESA.

Queue Data Set Restart

During each MERVA ESA startup, the queue management program checks whether it has ended normally after the last run. If not, it automatically carries out a queue-management restart. If the queue data set can be read physically, queue management maintains the integrity of the data on the queue data set. If you use the duplicate queue data set feature, the queue management tells you whether they are identical, or whether you have to duplicate the best one for maintaining data integrity.

If the Journal Data Sets Are Full

Whenever the journal data set A is full, the MERVA ESA Journal program switches automatically to journal data set B. The switch is indicated to you by the message:

```
DSL040I switched from journal A to journal B.
```

MERVA ESA should be stopped as soon as possible so that you can print and clear the journal data sets before the next MERVA ESA startup. The decision to do so must be based on the size of the journal data set B. If journal data set B becomes full, MERVA ESA terminates.

The command **jswitch** is provided to switch the journal data sets under operator control. Refer to "Switch the Journal Data Sets (JSWITCH)" on page 53 for details.

MERVA ESA Restart in a Multisystem Environment

If a restart of MERVA ESA in a multisystem environment is required after a previous failure, it is possible that MERVA ESA terminates the startup because it

was not able to resolve the message sequence left on an MQI queue. In this case, the MQI receive and reply-to queues of this MERVA ESA instance have to be emptied by redefining them.

Chapter 33. How to React to Problems with the SWIFT Link

Whenever an abnormal situation arises, the SWIFT Link displays an error message on the operating system console. The MERVA ESA operator should refer to *MERVA for ESA Messages and Codes*, and take the action indicated there.

In some cases, the subtask for a line to the SWIFT network may produce a dump with a user-abend code in MVS. You can distinguish these from the user-abend codes of the IMS BMP region by looking at the name of the module that got control from MVS. If this module is:

- An IMS module, the user abend is from IMS.
- DWSNAEVV, the user abend is from the line subtask, and the user-abend codes are described in *MERVA for ESA Messages and Codes*.

If the Queue Data Set Is Full

If the SWIFT Link (DWSDGPA) detects that the queue data set is full, or that all entries in the Queue Key Table have been used, it immediately cancels the line to SWIFT. You should either ask users to process the messages in the queues, or you should start the hard-copy printer program, DSLSDO, or DSLSDY to empty queues.

If these conditions appear frequently, the size of the queue data set must be increased using the MODIFY function of the MERVA ESA queue data set utility DSLQDSUT, see "Chapter 19. Using the Queue Data Set Utility DSLQDSUT" on page 275. Alternatively, the number of entries in the Queue Key Table must be increased before the next startup of MERVA ESA.

If You Have Problems with the Connection to SWIFT

If a problem arises with the connection to the SWIFT network, refer to the *S.W.I.F.T. User Handbook* or contact your S.W.I.F.T. Support Centre to solve the problem. Before doing this, make sure that there is no problem with the modem or the connection to the telephone network.

Chapter 34. How to React to Problems with the Telex Link

There are two ways to communicate with the public telex network:

- The Telex Link via a workstation
- The Telex Link via a fault-tolerant system using the Telex Interface Program.

Telex Link via a Workstation

This section applies to the use of Telex Link via a workstation.

Telex Link via a workstation is a MERVA Link application that connects MERVA ESA and the workstation based telex functions. Problems in the MERVA Link connection should be analyzed as described in "Chapter 35. How to React to Problems with MERVA Link" on page 375. Problems that are related to the workstation should be analyzed as described in the respective MERVA Messages and Codes manuals and Diagnosis Guides.

Telex Link via a Fault-Tolerant System

This section applies to the use of Telex Link via a fault-tolerant system.

If the response to the **txdisp** command shows that the Telex Link has already waited too long for the logical acknowledgment of a telex message, you can use the command **txdisp recover** to send the message to the Telex Interface Program again.

If no logical acknowledgment is received from the Telex Interface Program for the last telex message sent during a session, this message is sent again as the first message after a restart, containing a reference to the first transmission (the original session and sequence numbers).

The Telex Link Possible Duplicate Emitted queue, TXSTPPDE, contains the last sent telex message until the Telex Link receives the logical acknowledgment for this message from the Telex Interface Program. If sending the telex message again does not help, you can do one or more of the following:

- Ensure that the communication line between the Telex Link and the Telex Interface Program is active, using the appropriate CICS, IMS, or VTAM commands.
- Ensure that the Telex Substation is working properly. Restarting the Telex Substation is described in the *Telex Interface Program: Program Description and Operations Manual*.
- Sign off the session and sign on again.
- Stop the Telex Link and start it again.

The internal status of the Telex Link can be monitored using the debugging trace of MERVA ESA. The communication with the Telex Interface Program can be journaled. (See the description of the JRN1 parameter of the ENLPARM macro instruction in the *MERVA for ESA Macro Reference*.)

Chapter 35. How to React to Problems with MERVA Link

Whenever an abnormal situation arises, MERVA Link provides error information in the MERVA Link Partner Table and in the last confirmed control message. This error information can be displayed using the MERVA Link Control Facility.

When a sending ASP is set to inoperable, and when a receiving process error is found, MERVA Link displays an error message on the operating system console. A receiving process error message is, however, not displayed on a VSE console.

When you receive an error message, refer to *MERVA for ESA Messages and Codes* for an explanation, and take the action indicated.

If Messages in the Send Queue Are Not Processed

If messages in the send queue cluster of an ASP are unexpectedly not processed, carry out the following checks:

1. Is the ASP operable?

   To start transmission of messages in the send queue cluster, the ASP must have the status ON 00 (OPEN/NOHOLD, and last message transmission confirmed).

   If the ASP is in HOLD status, you can use the **astart** command to retry message transmission. Otherwise, the **kickoff** command is sufficient to resume message transmission.

   You can ask the MERVA Link to resume message processing for an ASP in any status. The request results in the display of the most up-to-date status information.

   Note: Duplicate messages are not generated.

2. Should the ASP start automatically?

   During customization, you can define an ASP so that message processing only starts upon operator request. This means that message processing is not automatically started when a message is routed to a MERVA Link send queue.

   To start an ASP that has been defined with the parameter START=OPERATOR, you can enter a MERVA ESA **sf** (start function) command for any send queue of that ASP. Alternatively, you can enter a MERVA Link **astart** or **kickoff** command for that ASP.

3. Is there up-to-date status or error information?

   The date/time stamp in the applicable ASP list line should be updated when you have entered the **astart** or **kickoff** command and the sending process has terminated. Up-to-date status information is now displayed. Refer to *MERVA for ESA Messages and Codes* for an explanation of the status information, and take the action indicated.

4. Is there a storage dump?

   Problems in the initialization phase of a MERVA Link task, and program checks within a MERVA Link task, are not reported in the last confirmed control message and the Partner Table.

   You can find information on these errors in a storage dump of a MERVA Link task. Analyze the storage dump, and take the appropriate action.

If Receiving Process Errors Are Found

The main sources of receiving process error information are:

- The MERVA Link Control Facility displaying status information in the Partner Table
- A task dump in the CICS environment
- A console operator message
- A snap dump of the MPP in the IMS environment.

Receiving process error diagnostic information is always collected in the Partner Table. It can be displayed using the MERVA Link Control Facility using the "Display Specific Message Transfer Application" as described in "Displaying Specific ASP/MTP Parameters" on page 136.

In the IMS environment, however, this information may not be available when the MERVA Link Control Facility tries to display it. Therefore, a receiving process error in the IMS environment is always reported in a snap dump of the MPP region.

A task dump in the CICS/MVS or CICS/VSE environment is only taken in an exceptional condition or if the error could not be reported in the Partner Table.

Part 5. Appendixes

Appendix A. MERVA ESA Operator Command Reference

This appendix contains lists of the operator commands and their parameters available in the Base Functions and the network links. Each of the lists provides the full and abbreviated form of the command (in alphabetical order), the parameters that can be used, whether the command is restricted, and where in this book you can find full information about the command.

The Base Functions Operator Commands

Command	Parameters	Restricted?	Page
cancel **c**	[{DUMP | ABDUMP}]	Yes	17
cf	*function,ltname*	Note 2 on page 384	19
copy	*fromqueue,toqueue,number[,UMR]*	Note 3 on page 384	384
dclog	[{ LAST | DETAIL }]		21
delete	*queue-name,number*	Note 3 on page 384	384
delx	*queue-name,qsn*[,*number*]	Note 3 on page 384	384
df	[{ FIRST | *function*[,FIRST]}]		22
dicb	[{ TSQ | APPC | MQT [,*number*]}]		384
dlmc			26
dlmct			28
dm	[{ *prefix* }] [{ *date* [,*prefix*]}] [{ *date* ,*time* [,*prefix*]}] [{ FIRST [,*prefix*]}] [{ LAST [,*prefix*]}]		30
dns	[{ FIRST | TASK | ALL | *servername* }]		33
dp	[{ FIRST | *progname* | *pid* }]		36
dq	[*function*],[FIRST][,FILLED] SQLERROR STATUS		38
dqsorted	[*function*],[FIRST]		42
dr	*reqnum*		384
drqa	[{ *srvnum* }] [{ *srvname* }] [{ FIRST [,*srvnum*]}] [{ FIRST [,*srvname*]}] [{ *srvnum* [,FIRST]}] [{ *srvname* [,FIRST]}]		384
drr	{[FIRST,] *reqnum* } { *reqnum* [,FIRST]}		384
du	[{FIRST | *userid*[,FIRST]}]		44
force	*userid*	Yes	46
free	*queue-name,number*	Note 3 on page 384	384

Command	Parameters	Restricted?	Page
hf	{*function* | ALL}	Note 1 on page 384, Note 2 on page 384	48
journal **jrn**	[{*datetime* | STATUS}]	Note 4 on page 384	384
jset	[CYCLE | MANUAL | ONCE]	Yes	50
jstat			52
jswitch	[{A | B } [RESET]]	Yes	54
move	*fromqueue,toqueue,number*[,*UMR*]	Note 3 on page 384	384
ntrace **ntrc**	{*parm1,parm2,parm3,parm4*}		384
priority **y**	{ *progname1,priority1, ...* *,progname8,priority8* } { *pid1,priority1, ... ,pid8,priority8* }	Yes	55
qswitch **qw**	[{*state* }] [{*queue-name*[,*state*]}] [{[*state*],*queue-name*}]	Yes	57
reshut **rs**		Yes	60
rswitch **rw**	[{*state*}] [{*rt-name*[,*state*]}] [{[*state*],*rt-name*}]	Yes	61
sf	*function*	Note 2 on page 384	64
shutdown **sh**		Yes	66
start **s**	{ *progname* | *pid* } [,*parameter*]	Yes	68
stop **p**	{ *progname* | *pid* }	Yes	70
terminat **t**	[{DUMP | ABDUMP }]	Yes	17

The SWIFT Link Operator Commands

Command	Parameters	Restricted?	Page
abortap **aap**	{[*ltname* | ALL] [,*line* | **ALL**]}	Note 1 on page 384	74
abortli **ali**	{*line* | ALL}	Yes	78
abortlt **alt**	{[*ltname* | ALL] [,*line* | **ALL**]}	Note 1 on page 384	80
close **cl**	*line*[,{IMM | DUMP}]	Yes	84
dds	[*ltname*]		86
diva	*line* [,Buffers | **States** | Vtambnd]		88

Command	Parameters	Restricted?	Page
dl	[**ALL**[,FIRST]] FIRST LINES *line*[,FIRST] *ltname*		94
dla	[**ALL**[,FIRST]] FIRST *ltname*		98
login **li**	[*ltname*],[*lsn*],[*sk1*],[*sk2*][,*window*]	Yes	100
logout **lo**	{[*ltname* | ALL], [*line* | **ALL**] [,*timeday*]}	Yes	104
quit **q**	{[*ltname* | ALL], [*line* | **ALL**] [,*timeday*]}	Yes	108
select **se**	[*ltname*],[*ssn*],[*sk1*], [*sk2*],[*window*],[*state*][,*delivery*]	Yes	112
setlt **slt**	*ltname*[,*line*[,*tflag*[,*iccparm*[,*usename*]]]]	Yes	117
swiftii **sw**	{*x*},{SWIFT Link command string}		120
swa **swb** **swc**	{SWIFT Link command string}		120
xtrace	*line* [,trace flag no. [,ON | OFF]]		384

The MERVA Link ESA Operator Commands

Note: These commands can only be entered in the MSC function. They are *not* valid in the CMD function.

Command	Parameters	Restricted?	Page
aclose	*aspname*	Yes	148
aopen	*aspname*	Yes	149
astart **sa**	*aspname*	Yes	150
backward **bw**			151
disable	{*aspname* | *}	Yes	152
display **da**	[*aspname* | *generic aspname*]		153
dpth			154
dsa	[*aspname* | *generic aspname*]		155
enable	{*aspname* | * | SET}	Yes	156
explain **xpl**			157
forward **fw**			151

Command	Parameters	Restricted?	Page
hold **ha**	{*aspname* | *}	Yes	158
iprecov **ipmo**ve	{*mip msn* | *}	Yes	159
kickoff **ka**	{*aspname* | *}	Yes	161
lcreset **lcrs**	*aspname*	Yes	162
lrreset **lrrs**	*aspname*	Yes	162
lstall **la**			163
lstinop			164
nextgrp **ng**			165
node **ps**	[*partner system name* | *]	Yes	166
recover **ipco**py	*aspname*	Yes	168
refresh **ra**			169
reset	{TEST | CTRC | ACCEPT | IPRECOV | AM}	Yes	170
set	{AM | TEST | XC | CTRCF | CTRCW} | {SNAP | PSNAP | ACCEPT | IPRECOV | WSZ}	Yes	170

The MERVA Link USS Operator Commands

Note: These commands can only be entered in the ACC function of MERVA Link USS. They are *not* valid in the CMD and MSC functions of a MERVA ESA environment.

Command	Parameters	Restricted?	Page
a **aih** **ara** **arc** **asa**	 *asp_name* *partner_node_name* *asp_name*	No	174
cxf **cxfr** **cxs** **cxsr**	[*path_name*] [*path_name*] [*path_name*] [*path_name*]	Yes	175
dpa **dpc** **dph** **dsa** **dsc** **dsd** **dsh**	*{asp_name|*}* *{partner_node_name|*}* *asp_name* *partner_node_name*	No	176

Command	Parameters	Restricted?	Page
dxc **dxd** **dxmm** **dxmn** **dxmq** **dxmr** **dxmt** **dxra** **dxrm** **dxsa** **dxsm**	*partner_node_name* *diagnostic_code* *dslmmfs_rc_rs* *dslnicp_rc* *dslqmgt_rc* *dslrtnsc_rc_rs* *dsltofsv_rc_rs* *receiving_as_ecv* *receiving_mt_ecv* *sending_as_ecv* *sending_mt_ecv*	No	178
h **ha** **hc** **hcx** **hd** **hdx** **hl** **hm** **hp** **hr** **hs** **hx**		No	182
l **la** **laa** **lc** **le** **lr** **lra** **lx** **lxi** **lxo** **lxr**		No	183
re **rta** **rtc** **rtd** **rtp** **rxi**		No	185
sc **si** **srt** **sst** **sta** **stc** **std** **stp** **stx** **swa** **swc** **swp**	 *asp_name* *directory_path_name*	No	186
trm	{**EKAACD** | **daemon**}	Yes	188

The Telex Link Operator Commands

Command	Parameters	Restricted?	Page
txdisp	[RECOVER]	Yes	127
txoff		Yes	129
txon		Yes	131

Notes:

1. The command is restricted, when entered with the ALL parameter.
2. The command is restricted, when entered for a function which is defined in the MERVA ESA function table DSLFNTT using the parameter MQI=YES.
3. Whether the queue test commands are restricted depends on the setting of the parameter EXQUE in the MERVA ESA parameter module DSLPRM.
4. Whether the journal command is restricted depends on the setting of the parameter EXJRN in the MERVA ESA parameter module DSLPRM.
5. This command is for diagnostic purposes and is described in the *MERVA for ESA Diagnosis Guide*.
6. This command is for installation test purposes and is described in the *MERVA for ESA Installation Guide*.

Appendix B. Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Deutschland
Informationssysteme GmbH
Department 3982
Pascalstrasse 100

70569 Stuttgart
Germany

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement or any equivalent agreement between us.

The following paragraph does apply to the US only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States, other countries, or both:

- Advanced Peer-to-Peer Networking
- AIX
- APPN
- C/370
- CICS
- CICS/ESA
- CICS/MVS
- CICS/VSE
- DB2
- DB2 Universal Database
- Distributed Relational Database Architecture
- DRDA
- IBM
- IMS/ESA
- Language Environment
- MQSeries

- MVS
- MVS/ESA
- MVS/XA
- OS/2
- OS/390
- RACF
- VisualAge
- VSE/ESA
- VTAM

Workstation (AWS) and Directory Services Application (DSA) are trademarks of S.W.I.F.T., La Hulpe in Belgium.

Pentium is a trademark of Intel Corporation.

PC Direct is a trademark of Ziff Communications Company in the United States, other countries, or both, and is used by IBM Corporation under license.

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

Glossary of Terms and Abbreviations

This glossary defines terms as they are used in this book. If you do not find the terms you are looking for, refer to the *IBM Dictionary of Computing*, New York: McGraw-Hill, and the *S.W.I.F.T. User Handbook*.

A

ACB. Access method control block.

ACC. MERVA Link USS application control command application. It provides a means of operating MERVA Link USS in USS shell and MVS batch environments.

Access method control block (ACB). A control block that links an application program to VSAM or VTAM.

ACD. MERVA Link USS application control daemon.

ACT. MERVA Link USS application control table.

address. See *SWIFT address*.

address expansion. The process by which the full name of a financial institution is obtained using the SWIFT address, telex correspondent's address, or a nickname.

AMPDU. Application message protocol data unit, which is defined in the MERVA Link P1 protocol, and consists of an envelope and its content.

answerback. In telex, the response from the dialed correspondent to the WHO R U signal.

answerback code. A group of up to 6 letters following or contained in the answerback. It is used to check the answerback.

APC. Application control.

API. Application programming interface.

APPC. Advanced Program-to-Program Communication based on SNA LU 6.2 protocols.

APPL. A VTAM definition statement used to define a VTAM application program.

application programming interface (API). An interface that programs can use to exchange data.

application support filter (ASF). In MERVA Link, a user-written program that can control and modify any data exchanged between the Application Support Layer and the Message Transfer Layer.

application support process (ASP). An executing instance of an application support program. Each application support process is associated with an ASP entry in the partner table. An ASP that handles outgoing messages is a *sending ASP*; one that handles incoming messages is a *receiving ASP*.

application support program (ASP). In MERVA Link, a program that exchanges messages and reports with a specific remote partener ASP. These two programs must agree on which conversation protocol they are to use.

ASCII. American Standard Code for Information Interchange. The standard code, using a coded set consisting of 7-bit coded characters (8 bits including parity check), used for information interchange among data processing systems, data communication systems, and associated equipment. The ASCII set consists of control characters and graphic characters.

ASF. Application support filter.

ASP. (1) Application support process. (2) Application support program.

ASPDU. Application support protocol data unit, which is defined in the MERVA Link P2 protocol.

authentication. The SWIFT security check used to ensure that a message has not changed during transmission, and that it was sent by an authorized sender.

authenticator key. A set of alphanumeric characters used for the authentication of a message sent via the SWIFT network.

authenticator-key file. The file that stores the keys used during the authentication of a message. The file contains a record for each of your financial institution's correspondents.

B

Back-to-Back (BTB). A MERVA Link function that enables ASPs to exchange messages in the local MERVA Link node without using data communication services.

bank identifier code. A 12-character code used to identify a bank within the SWIFT network. Also called a SWIFT address. The code consists of the following subcodes:

- The bank code (4 characters)
- The ISO country code (2 characters)
- The location code (2 characters)
- The address extension (1 character)

- The branch code (3 characters) for a SWIFT user institution, or the letters "BIC" for institutions that are not SWIFT users.

Basic Security Manager (BSM). A component of VSE/ESA Version 2.4 that is invoked by the System Authorization Facility, and used to ensure signon and transaction security.

BIC. Bank identifier code.

BIC Bankfile. A tape of bank identifier codes supplied by S.W.I.F.T.

BIC Database Plus Tape. A tape of financial institutions and currency codes, supplied by S.W.I.F.T. The information is compiled from various sources and includes national, international, and cross-border identifiers.

BIC Directory Update Tape. A tape of bank identifier codes and currency codes, supplied by S.W.I.F.T., with extended information as published in the printed BIC Directory.

body. The second part of an IM-ASPDU. It contains the actual application data or the message text that the IM-AMPDU transfers.

BSC. Binary synchronous control.

BSM. Basic Security Manager.

BTB. Back-to-back.

buffer. A storage area used by MERVA programs to store a message in its internal format. A buffer has an 8-byte prefix that indicates its length.

C

CBT. SWIFT computer-based terminal.

CCSID. Coded character set identifier.

CDS. Control data set.

central service. In MERVA, a service that uses resources that either require serialization of access, or are only available in the MERVA nucleus.

CF message. Confirmed message. When a sending MERVA Link system is informed of the successful delivery of a message to the receiving application, it routes the delivered application messages as CF messages, that is, messages of class CF, to an ACK wait queue or to a complete message queue.

COA. Confirm on arrival.

COD. Confirm on delivery.

coded character set identifier (CCSID). The name of a coded set of characters and their code point assignments.

commit. In MQSeries, to commit operations is to make the changes on MQSeries queues permanent. After putting one or more messages to a queue, a commit makes them visible to other programs. After getting one or more messages from a queue, a commit permanently deletes them from the queue.

confirm-on-arrival (COA) report. An MQSeries report message type created when a message is placed on that queue. It is created by the queue manager that owns the destination queue.

confirm-on-delivery (COD) report. An MQSeries report message type created when an application retrieves a message from the queue in a way that causes the message to be deleted from the queue. It is created by the queue manager.

control fields. In MERVA Link, fields that are part of a MERVA message on the queue data set and of the message in the TOF. Control fields are written to the TOF at nesting identifier 0. Messages in SWIFT format do not contain control fields.

correspondent. An institution to which your institution sends and from which it receives messages.

correspondent identifier. The 11-character identifier of the receiver of a telex message. Used as a key to retrieve information from the Telex correspondents file.

cross-system coupling facility. See *XCF*.

coupling services. In a sysplex, the functions of XCF that transfer data and status information among the members of a group that reside in one or more of the MVS systems in the sysplex.

couple data set. See *XCF couple data set*.

CTP. MERVA Link command transfer processor.

currency code file. A file containing the currency codes, together with the name, fraction length, country code, and country names.

D

daemon. A long-lived process that runs unattended to perform continuous or periodic systemwide functions.

DASD. Direct access storage device.

data area. An area of a predefined length and format on a panel in which data can be entered or displayed. A field can consist of one or more data areas.

data element. A unit of data that, in a certain context, is considered indivisible. In MERVA Link, a data

element consists of a 2-byte data element length field, a 2-byte data-element identifier field, and a field of variable length containing the data element data.

datagram. In TCP/IP, the basic unit of information passed across the Internet environment. This type of message does not require a reply, and is the simplest type of message that MQSeries supports.

data terminal equipment. That part of a data station that serves as a data source, data link, or both, and provides for the data communication control function according to protocols.

DB2. A family of IBM licensed programs for relational database management.

dead-letter queue. A queue to which a queue manager or application sends messages that it cannot deliver. Also called *undelivered-message queue*.

dial-up number. A series of digits required to establish a connection with a remote correspondent via the public telex network.

direct service. In MERVA, a service that uses resources that are always available and that can be used by several requesters at the same time.

display mode. The mode (PROMPT or NOPROMPT) in which SWIFT messages are displayed. See *PROMPT mode* and *NOPROMPT mode.*

distributed queue management (DQM). In MQSeries message queuing, the setup and control of message channels to queue managers on other systems.

DQM. Distributed queue management.

DTE. Data terminal equipment.

E

EBCDIC. Extended Binary Coded Decimal Interchange Code. A coded character set consisting of 8-bit coded characters.

ECB. Event control block.

EDIFACT. Electronic Data Interchange for Administration, Commerce and Transport (a United Nations standard).

ESM. External security manager.

EUD. End-user driver.

exception report. An MQSeries report message type that is created by a message channel agent when a message is sent to another queue manager, but that message cannot be delivered to the specified destination queue.

external line format (ELF) messages. Messages that are not fully tokenized, but are stored in a single field in the TOF. Storing messages in ELF improves performance, because no mapping is needed, and checking is not performed.

external security manager (ESM). A security product that is invoked by the System Authorization Facility. RACF is an example of an ESM.

F

FDT. Field definition table.

field. In MERVA, a portion of a message used to enter or display a particular type of data in a predefined format. A field is located by its position in a message and by its tag. A field is made up of one or more data areas. See also *data area*.

field definition table (FDT). The field definition table describes the characteristics of a field; for example, its length and number of its data areas, and whether it is mandatory. If the characteristics of a field change depending on its use in a particular message, the definition of the field in the FDT can be overridden by the MCB specifications.

field group. One or several fields that are defined as being a group. Because a field can occur more than once in a message, field groups are used to distinguish them. A name can be assigned to the field group during message definition.

field group number. In the TOF, a number is assigned to each field group in a message in ascending order from 1 to 255. A particular field group can be accessed using its field group number.

field tag. A character string used by MERVA to identify a field in a network buffer. For example, for SWIFT field 30, the field tag is **:30:**.

FIN. Financial application.

FIN-Copy. The MERVA component used for SWIFT FIN-Copy support.

finite state machine. The theoretical base describing the rules of a service request's state and the conditions to state transitions.

FMT/ESA. MERVA-to-MERVA Financial Message Transfer/ESA.

form. A partially-filled message containing data that can be copied for a new message of the same message type.

G

GPA. General purpose application.

H

HFS. Hierarchical file system.

hierarchical file system (HFS). A system for organizing files in a hierarchy, as in a UNIX system. OS/390 UNIX System Services files are organized in an HFS. All files are members of a directory, and each directory is in turn a member of a directory at a higher level in the HFS. The highest level in the hierarchy is the root directory.

I

IAM. Interapplication messaging (a MERVA Link message exchange protocol).

IM-ASPDU. Interapplication messaging application support protocol data unit. It contains an application message and consists of a heading and a body.

incore request queue. Another name for the request queue to emphasize that the request queue is held in memory instead of on a DASD.

InetD. Internet Daemon. It provides TCP/IP communication services in the OS/390 USS environment.

initiation queue. In MQSeries, a local queue on which the queue manager puts trigger messages.

input message. A message that is input into the SWIFT network. An input message has an input header.

INTERCOPE TelexBox. This telex box supports various national conventions for telex procedures and protocols.

interservice communication. In MERVA ESA, a facility that enables communication among services if MERVA ESA is running in a multisystem environment.

intertask communication. A facility that enables application programs to communicate with the MERVA nucleus and so request a central service.

IP. Internet Protocol.

IP message. In-process message. A message that is in the process of being transferred to another application.

ISC. Intersystem communication.

ISN. Input sequence number.

ISN acknowledgment. A collective term for the various kinds of acknowledgments sent by the SWIFT network.

ISO. International Organization for Standardization.

ITC. Intertask communication.

J

JCL. Job control language.

journal. A chronological list of records detailing MERVA actions.

journal key. A key used to identify a record in the journal.

journal service. A MERVA central service that maintains the journal.

K

KB. Kilobyte (1024 bytes).

key. A character or set of characters used to identify an item or group of items. For example, the user ID is the key to identify a user file record.

key-sequenced data set (KSDS). A VSAM data set whose records are loaded in key sequence and controlled by an index.

keyword parameter. A parameter that consists of a keyword, followed by one or more values.

KSDS. Key-sequenced data set.

L

LAK. Login acknowledgment message. This message informs you that you have successfully logged in to the SWIFT network.

large message. A message that is stored in the large message cluster (LMC). The maximum length of a message to be stored in the VSAM QDS is 31900 bytes. Messages up to 2MB can be stored in the LMC. For queue management using DB2 no distinction is made between messages and large messages.

large queue element. A queue element that is larger than the smaller of:

- The limiting value specified during the customization of MERVA
- 32KB

LC message. Last confirmed control message. It contains the message-sequence number of the application or acknowledgment message that was last confirmed; that is, for which the sending MERVA Link system most recently received confirmation of a successful delivery.

LDS. Logical data stream.

LMC. Large message cluster.

LNK. Login negative acknowledgment message. This message indicates that the login to the SWIFT network has failed.

local queue. In MQSeries, a queue that belongs to a local queue manager. A local queue can contain a list of messages waiting to be processed. Contrast with *remote queue*.

local queue manager. In MQSeries, the queue manager to which the program is connected, and that provides message queuing services to that program. Queue managers to which a program is not connected are remote queue managers, even if they are running on the same system as the program.

login. To start the connection to the SWIFT network.

LR message. Last received control message, which contains the message-sequence number of the application or acknowledgment message that was last received from the partner application.

LSN. Login sequence number.

LT. See *LTERM*.

LTC. Logical terminal control.

LTERM. Logical terminal. Logical terminal names have 4 characters in CICS and up to 8 characters in IMS.

LU. A VTAM logical unit.

M

maintain system history program (MSHP). A program used for automating and controlling various installation, tailoring, and service activities for a VSE system.

MCA. Message channel agent.

MCB. Message control block.

MERVA ESA. The IBM licensed program Message Entry and Routing with Interfaces to Various Applications for ESA.

MERVA Link. A MERVA component that can be used to interconnect several MERVA systems.

message. A string of fields in a predefined form used to provide or request information. See also *SWIFT financial message.*

message body. The part of the message that contains the message text.

message category. A group of messages that are logically related within an application.

message channel. In MQSeries distributed message queuing, a mechanism for moving messages from one queue manager to another. A message channel comprises two message channel agents (a sender and a receiver) and a communication link.

message channel agent (MCA). In MQSeries, a program that transmits prepared messages from a transmission queue to a communication link, or from a communication link to a destination queue.

message control block (MCB). The definition of a message, screen panel, net format, or printer layout made during customization of MERVA.

Message Format Service (MFS). A MERVA direct service that formats a message according to the medium to be used, and checks it for formal correctness.

message header. The leading part of a message that contains the sender and receiver of the message, the message priority, and the type of message.

Message Integrity Protocol (MIP). In MERVA Link, the protocol that controls the exchange of messages between partner ASPs. This protocol ensures that any loss of a message is detected and reported, and that no message is duplicated despite system failures at any point during the transfer process.

message-processing function. The various parts of MERVA used to handle a step in the message-processing route, together with any necessary equipment.

message queue. See *queue*.

Message Queue Interface (MQI). The programming interface provided by the MQSeries queue managers. It provides a set of calls that let application programs access message queuing services such as sending messages, receiving messages, and manipulating MQSeries objects.

Message Queue Manager (MQM). An IBM licensed program that provides message queuing services. It is part of the MQSeries set of products.

message reference number (MRN). A unique 16-digit number assigned to each message for identification purposes. The message reference number consists of an 8-digit domain identifier that is followed by an 8-digit sequence number.

message sequence number (MSN). A sequence number for messages transferred by MERVA Link.

message type (MT). A number, up to 7 digits long, that identifies a message. SWIFT messages are identified by a 3-digit number; for example SWIFT message type MT S100.

MFS. Message Format Service.

MIP. Message Integrity Protocol.

MPDU. Message protocol data unit, which is defined in P1.

MPP. In IMS, message-processing program.

MQA. MQ Attachment.

MQ Attachment (MQA). A MERVA feature that provides message transfer between MERVA and a user-written MQI application.

MQH. MQSeries queue handler.

MQI. Message queue interface.

MQM. Message queue manager.

MQS. MQSeries nucleus server.

MQSeries. A family of IBM licensed programs that provides message queuing services.

MQSeries nucleus server (MQS). A MERVA component that listens for messages on an MQI queue, receives them, extracts a service request, and passes it via the request queue handler to another MERVA ESA instance for processing.

MQSeries queue handler (MQH). A MERVA component that performs service calls to the Message Queue Manager via the provided Message Queue Interface.

MRN. Message reference number.

MSC. MERVA system control facility.

MSHP. Maintain system history program.

MSN. Message sequence number.

MT. Message type.

MTP. (1) Message transfer program. (2) Message transfer process.

MTS. Message Transfer System.

MTSP. Message Transfer Service Processor.

MTT. Message type table.

multisystem application. (1) An application program that has various functions distributed across MVS systems in a multisystem environment. (2) In XCF, an authorized application that uses XCF coupling services. (3) In MERVA ESA, multiple instances of MERVA ESA that are distributed among different MVS systems in a multisystem environment.

multisystem environment. An environment in which two or more MVS systems reside on one or more processors, and programs on one system can communicate with programs on the other systems. With XCF, the environment in which XCF services are available in a defined sysplex.

multisystem sysplex. A sysplex in which one or more MVS systems can be initialized as part of the sysplex. In a multisystem sysplex, XCF provides coupling services on all systems in the sysplex and requires an XCF couple data set that is shared by all systems. See also *single-system sysplex*.

MVS/ESA. Multiple Virtual Storage/Enterprise Systems Architecture.

N

namelist. An MQSeries for MVS/ESA object that contains a list of queue names.

nested message. A message that is composed of one or more message types.

nested message type. A message type that is contained in another message type. In some cases, only part of a message type (for example, only the mandatory fields) is nested, but this “partial” nested message type is also considered to be nested. For example, SWIFT MT 195 could be used to request information about a SWIFT MT 100 (customer transfer). The SWIFT MT 100 (or at least its mandatory fields) is then nested in SWIFT MT 195.

nesting identifier. An identifier (a number from 2 to 255) that is used to access a nested message type.

network identifier. A single character that is placed before a message type to indicate which network is to be used to send the message; for example, **S** for SWIFT

network service access point (NSAP). The endpoint of a network connection used by the SWIFT transport layer.

NOPROMPT mode. One of two ways to display a message panel. NOPROMPT mode is only intended for experienced SWIFT Link users who are familiar with the structure of SWIFT messages. With NOPROMPT mode, only the SWIFT header, trailer, and pre-filled fields and their tags are displayed. Contrast with *PROMPT mode*.

NSAP. Network service access point.

nucleus server. A MERVA component that processes a service request as selected by the request queue handler. The service a nucleus server provides and the way it provides it is defined in the nucleus server table (DSLNSVT).

O

object. In MQSeries, objects define the properties of queue managers, queues, process definitions, and namelists.

occurrence. See *repeatable sequence*.

option. One or more characters added to a SWIFT field number to distinguish among different layouts for and meanings of the same field. For example, SWIFT field 60 can have an option F to identify a first opening balance, or M for an intermediate opening balance.

origin identifier (origin ID). A 34-byte field of the MERVA user file record. It indicates, in a MERVA and SWIFT Link installation that is shared by several banks, to which of these banks the user belongs. This lets the user work for that bank only.

OSN. Output sequence number.

OSN acknowledgment. A collective term for the various kinds of acknowledgments sent to the SWIFT network.

output message. A message that has been received from the SWIFT network. An output message has an output header.

P

P1. In MERVA Link, a peer-to-peer protocol used by cooperating message transfer processes (MTPs).

P2. In MERVA Link, a peer-to-peer protocol used by cooperating application support processes (ASPs).

P3. In MERVA Link, a peer-to-peer protocol used by cooperating command transfer processors (CTPs).

packet switched public data network (PSPDN). A public data network established and operated by network common carriers or telecommunication administrations for providing packet-switched data transmission.

panel. A formatted display on a display terminal. Each page of a message is displayed on a separate panel.

parallel processing. The simultaneous processing of units of work by several servers. The units of work can be either transactions or subdivisions of larger units of work.

parallel sysplex. A sysplex that uses one or more coupling facilities.

partner table (PT). In MERVA Link, the table that defines how messages are processed. It consists of a header and different entries, such as entries to specify the message-processing parameters of an ASP or MTP.

PCT. Program Control Table (of CICS).

PDE. Possible duplicate emission.

PDU. Protocol data unit.

PF key. Program-function key.

positional parameter. A parameter that must appear in a specified location relative to other parameters.

PREMIUM. The MERVA component used for SWIFT PREMIUM support.

process definition object. An MQSeries object that contains the definition of an MQSeries application. A queue manager uses the definitions contained in a process definition object when it works with trigger messages.

program-function key. A key on a display terminal keyboard to which a function (for example, a command) can be assigned. This lets you execute the function (enter the command) with a single keystroke.

PROMPT mode. One of two ways to display a message panel. PROMPT mode is intended for SWIFT Link users who are unfamiliar with the structure of SWIFT messages. With PROMPT mode, all the fields and tags are displayed for the SWIFT message. Contrast with *NOPROMPT mode*.

protocol data unit (PDU). In MERVA Link a PDU consists of a structured sequence of implicit and explicit data elements:
- Implicit data elements contain other data elements.
- Explicit data elements cannot contain any other data elements.

PSN. Public switched network.

PSPDN. Packet switched public data network.

PSTN. Public switched telephone network.

PT. Partner table.

PTT. A national post and telecommunication authority (post, telegraph, telephone).

Q

QDS. Queue data set.

QSN. Queue sequence number.

queue. (1) In MERVA, a logical subdivision of the MERVA queue data set used to store the messages associated with a MERVA message-processing function. A queue has the same name as the message-processing function with which it is associated. (2) In MQSeries, an

object onto which message queuing applications can put messages, and from which they can get messages. A queue is owned and maintained by a queue manager. See also *request queue*.

queue element. A message and its related control information stored in a data record in the MERVA ESA Queue Data Set.

queue management. A MERVA service function that handles the storing of messages in, and the retrieval of messages from, the queues of message-processing functions.

queue manager. (1) An MQSeries system program that provides queueing services to applications. It provides an application programming interface so that programs can access messages on the queues that the queue manager owns. See also *local queue manager* and *remote queue manager*. (2) The MQSeries object that defines the attributes of a particular queue manager.

queue sequence number (QSN). A sequence number that is assigned to the messages stored in a logical queue by MERVA ESA queue management in ascending order. The QSN is always unique in a queue. It is reset to zero when the queue data set is formatted, or when a queue management restart is carried out and the queue is empty.

R

RACF. Resource Access Control Facility.

RBA. Relative byte address.

RC message. Recovered message; that is, an IP message that was copied from the control queue of an inoperable or closed ASP via the **recover** command.

ready queue. A MERVA queue used by SWIFT Link to collect SWIFT messages that are ready for sending to the SWIFT network.

remote queue. In MQSeries, a queue that belongs to a remote queue manager. Programs can put messages on remote queues, but they cannot get messages from remote queues. Contrast with *local queue*.

remote queue manager. In MQSeries, a queue manager is remote to a program if it is not the queue manager to which the program is connected.

repeatable sequence. A field or a group of fields that is contained more than once in a message. For example, if the SWIFT fields 20, 32, and 72 form a sequence, and if this sequence can be repeated up to 10 times in a message, each sequence of the fields 20, 32, and 72 would be an occurrence of the repeatable sequence.

In the TOF, the occurrences of a repeatable sequence are numbered in ascending order from 1 to 32767 and can be referred to using the occurrence number.

A repeatable sequence in a message may itself contain another repeatable sequence. To identify an occurrence within such a nested repeatable sequence, more than one occurrence number is necessary.

reply message. In MQSeries, a type of message used for replies to request messages.

reply-to queue. In MQSeries, the name of a queue to which the program that issued an MQPUT call wants a reply message or report message sent.

report message. In MQSeries, a type of message that gives information about another message. A report message usually indicates that the original message cannot be processed for some reason.

request message. In MQSeries, a type of message used for requesting a reply from another program.

request queue. The queue in which a service request is stored. It resides in main storage and consists of a set of request queue elements that are chained in different queues:

- Requests waiting to be processed
- Requests currently being processed
- Requests for which processing has finished

request queue handler (RQH). A MERVA ESA component that handles the queueing and scheduling of service requests. It controls the request processing of a nucleus server according to rules defined in the finite state machine.

Resource Access Control Facility (RACF). An IBM licensed program that provides for access control by identifying and verifying users to the system, authorizing access to protected resources, logging detected unauthorized attempts to enter the system, and logging detected accesses to protected resources.

retype verification. See *verification*.

routing. In MERVA, the passing of messages from one stage in a predefined processing path to the next stage.

RP. Regional processor.

RQH. Request queue handler.

RRDS. Relative record data set.

S

SAF. System Authorization Facility.

SCS. SNA character string

SCP. System control process.

SDI. Sequential data set input. A batch utility used to import messages from a sequential data set or a tape into MERVA ESA queues.

SDO. Sequential data set output. A batch utility used to export messages from a MERVA ESA queue to a sequential data set or a tape.

SDY. Sequential data set system printer. A batch utility used to print messages from a MERVA ESA queue.

service request. A type of request that is created and passed to the request queue handler whenever a nucleus server requires a service that is not currently available.

sequence number. A number assigned to each message exchanged between two nodes. The number is increased by one for each successive message. It starts from zero each time a new session is established.

sign off. To end a session with MERVA.

sign on. To start a session with MERVA.

single-system sysplex. A sysplex in which only one MVS system can be initialized as part of the sysplex. In a single-system sysplex, XCF provides XCF services on the system, but does not provide signalling services between MVS systems. A single-system sysplex requires an XCF couple data set. See also *multisystem sysplex*.

small queue element. A queue element that is smaller than the smaller of:

- The limiting value specified during the customization of MERVA
- 32KB

SMP/E. System Modification Program Extended.

SN. Session number.

SNA. Systems network architecture.

SNA character string. In SNA, a character string composed of EBCDIC controls, optionally mixed with user data, that is carried within a request or response unit.

SPA. Scratch pad area.

SQL. Structured Query Language.

SR-ASPDU. The status report application support PDU, which is used by MERVA Link for acknowledgment messages.

SSN. Select sequence number.

subfield. A subdivision of a field with a specific meaning. For example, the SWIFT field 32 has the subfields date, currency code, and amount. A field can have several subfield layouts depending on the way the field is used in a particular message.

SVC. (1) Switched Virtual Circuit. (2) Supervisor call instruction.

S.W.I.F.T. (1) Society for Worldwide Interbank Financial Telecommunication s.c. (2) The network provided and managed by the Society for Worldwide Interbank Financial Telecommunication s.c.

SWIFT address. Synonym for *bank identifier code*.

SWIFT Correspondents File. The file containing the bank identifier code (BIC), together with the name, postal address, and zip code of each financial institution in the BIC Directory.

SWIFT financial message. A message in one of the SWIFT categories 1 to 9 that you can send or receive via the SWIFT network. See *SWIFT input message* and *SWIFT output message*.

SWIFT header. The leading part of a message that contains the sender and receiver of the message, the message priority, and the type of message.

SWIFT input message. A SWIFT message with an input header to be sent to the SWIFT network.

SWIFT link. The MERVA ESA component used to link to the SWIFT network.

SWIFT network. Refers to the SWIFT network of the Society for Worldwide Interbank Financial Telecommunication (S.W.I.F.T.).

SWIFT output message. A SWIFT message with an output header coming from the SWIFT network.

SWIFT system message. A SWIFT general purpose application (GPA) message or a financial application (FIN) message in SWIFT category 0.

switched virtual circuit (SVC). An X.25 circuit that is dynamically established when needed. It is the X.25 equivalent of a switched line.

sysplex. One or more MVS systems that communicate and cooperate via special multisystem hardware components and software services.

System Authorization Facility (SAF). An MVS or VSE facility through which MERVA ESA communicates with an external security manager such as RACF (for MVS) or the basic security manager (for VSE).

System Control Process (SCP). A MERVA Link component that handles the transfer of MERVA ESA commands to a partner MERVA ESA system, and the receipt of the command response. It is associated with a system control process entry in the partner table.

System Modification Program Extended (SMP/E). A licensed program used to install software and software changes on MVS systems.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operating sequences for transmitting information units through, and for controlling the configuration and operation of, networks.

T

tag. A field identifier.

TCP/IP. Transmission Control Protocol/Internet Protocol.

Telex Correspondents File. A file that stores data about correspondents. When the user enters the corresponding nickname in a Telex message, the corresponding information in this file is automatically retrieved and entered into the Telex header area.

telex header area. The first part of the telex message. It contains control information for the telex network.

telex interface program (TXIP). A program that runs on a Telex front-end computer and provides a communication facility to connect MERVA ESA with the Telex network.

Telex Link. The MERVA ESA component used to link to the public telex network via a Telex substation.

Telex substation. A unit comprised of the following:

- Telex Interface Program
- A Telex front-end computer
- A Telex box

Terminal User Control Block (TUCB). A control block containing terminal-specific and user-specific information used for processing messages for display devices such as screen and printers.

test key. A key added to a telex message to ensure message integrity and authorized delivery. The test key is an integer value of up to 16 digits, calculated manually or by a test-key processing program using the significant information in the message, such as amounts, currency codes, and the message date.

test-key processing program. A program that automatically calculates and verifies a test key. The Telex Link supports panels for input of test-key-related data and an interface for a test-key processing program.

TFD. Terminal feature definitions table.

TID. Terminal identification. The first 9 characters of a bank identifier code (BIC).

TOF. Originally the abbreviation of *tokenized form*, the TOF is a storage area where messages are stored so that their fields can be accessed directly by their field names and other index information.

TP. Transaction program.

transaction. A specific set of input data that triggers the running of a specific process or job; for example, a message destined for an application program.

transaction code. In IMS and CICS, an alphanumeric code that calls an IMS message processing program or a CICS transaction. Transaction codes have 4 characters in CICS and up to 8 characters in IMS.

Transmission Control Protocol/Internet Protocol (TCP/IP). A set of communication protocols that support peer-to-peer connectivity functions for both local and wide area networks.

transmission queue. In MQSeries, a local queue on which prepared messages destined for a remote queue manager are temporarily stored.

trigger event. In MQSeries, an event (such as a message arriving on a queue) that causes a queue manager to create a trigger message on an initiation queue.

trigger message. In MQSeries, a message that contains information about the program that a trigger monitor is to start.

trigger monitor. In MQSeries, a continuously-running application that serves one or more initiation queues. When a trigger message arrives on an initiation queue, the trigger monitor retrieves the message. It uses the information in the trigger message to start a process that serves the queue on which a trigger event occurred.

triggering. In MQSeries, a facility that allows a queue manager to start an application automatically when predetermined conditions are satisfied.

TUCB. Terminal User Control Block.

TXIP. Telex interface program.

U

UMR. Unique message reference.

unique message reference (UMR). An optional feature of MERVA ESA that provides each message with a unique identifier the first time it is placed in a queue. It is composed of a MERVA ESA installation name, a sequence number, and a date and time stamp.

UNIT. A group of related literals or fields of an MCB definition, or both, enclosed by a DSLLUNIT and DSLLUEND macroinstruction.

UNIX System Services (USS). A component of OS/390, formerly called OpenEdition (OE), that creates a UNIX environment that conforms to the XPG4 UNIX 1995 specifications, and provides two open systems interfaces on the OS/390 operating system:

- An application program interface (API)
- An interactive shell interface

UN/EDIFACT. United Nations Standard for Electronic Data Interchange for Administration, Commerce and Transport.

USE. S.W.I.F.T. User Security Enhancements.

user file. A file containing information about all MERVA ESA users; for example, which functions each user is allowed to access. The user file is encrypted and can only be accessed by authorized persons.

user identification and verification. The acts of identifying and verifying a RACF-defined user to the system during logon or batch job processing. RACF identifies the user by the user ID and verifies the user by the password or operator identification card supplied during logon processing or the password supplied on a batch JOB statement.

USS. UNIX System Services.

V

verification. Checking to ensure that the contents of a message are correct. Two kinds of verification are:

- Visual verification: you read the message and confirm that you have done so
- Retype verification: you reenter the data to be verified

Virtual LU. An LU defined in MERVA Extended Connectivity for communication between MERVA and MERVA Extended Connectivity.

Virtual Storage Access Method (VSAM). An access method for direct or sequential processing of fixed and variable-length records on direct access devices. The records in a VSAM data set or file can be organized in logical sequence by a key field (key sequence), in the physical sequence in which they are written on the data set or file (entry sequence), or by relative-record number.

Virtual Telecommunications Access Method (VTAM). An IBM licensed program that controls communication and the flow of data in an SNA network. It provides single-domain, multiple-domain, and interconnected network capability.

VSAM. Virtual Storage Access Method.

VTAM. Virtual Telecommunications Access Method (IBM licensed program).

W

Windows NT service. A type of Windows NT application that can run in the background of the Windows NT operating system even when no user is logged on. Typically, such a service has no user interaction and writes its output messages to the Windows NT event log.

X

X.25. An ISO standard for interface to packet switched communications services.

XCF. Abbreviation for *cross-system coupling facility*, which is a special logical partition that provides high-speed caching, list processing, and locking functions in a sysplex. XCF provides the MVS coupling services that allow authorized programs on MVS systems in a multisystem environment to communicate with (send data to and receive data from) authorized programs on other MVS systems.

XCF couple data sets. A data set that is created through the XCF couple data set format utility and, depending on its designated type, is shared by some or all of the MVS systems in a sysplex. It is accessed only by XCF and contains XCF-related data about the sysplex, systems, applications, groups, and members.

XCF group. The set of related members defined to SCF by a multisystem application in which members of the group can communicate with (send data to and receive data from) other members of the same group. All MERVA systems working together in a sysplex must pertain to the same XCF group.

XCF member. A specific function of a multisystem application that is defined to XCF and assigned to a group by the multisystem application. A member resides on one system in a sysplex and can use XCF services to communicate with other members of the same group.

Bibliography

MERVA ESA Publications

- *MERVA for ESA Version 4: Application Programming Interface Guide*, SH12-6374
- *MERVA for ESA Version 4: Advanced MERVA Link*, SH12-6390
- *MERVA for ESA Version 4: Concepts and Components*, SH12-6381
- *MERVA for ESA Version 4: Customization Guide*, SH12-6380
- *MERVA for ESA Version 4: Diagnosis Guide*, SH12-6382
- *MERVA for ESA Version 4: Installation Guide*, SH12-6378
- *MERVA for ESA Version 4: Licensed Program Specifications*, GH12-6373
- *MERVA for ESA Version 4: Macro Reference*, SH12-6377
- *MERVA for ESA Version 4: Messages and Codes*, SH12-6379
- *MERVA for ESA Version 4: Operations Guide*, SH12-6375
- *MERVA for ESA Version 4: System Programming Guide*, SH12-6366
- *MERVA for ESA Version 4: User's Guide*, SH12-6376

MERVA ESA Components Publications

- *MERVA Automatic Message Import/Export Facility: User's Guide*, SH12-6389
- *MERVA Connection/NT*, SH12-6339
- *MERVA Connection/400*, SH12-6340
- *MERVA Directory Services*, SH12-6367
- *MERVA Extended Connectivity: Installation and User's Guide*, SH12-6157
- *MERVA Message Processing Client for Windows NT: User's Guide*, SH12-6341
- *MERVA-MQI Attachment User's Guide*, SH12-6714
- *MERVA Traffic Reconciliation*, SH12-6392
- *MERVA USE: Administration Guide*, SH12-6338
- *MERVA USE & Branch for Windows NT: User's Guide*, SH12-6334
- *MERVA USE & Branch for Windows NT: Installation and Customization Guide*, SH12-6335
- *MERVA USE & Branch for Windows NT: Application Programming Guide*, SH12-6336
- *MERVA USE & Branch for Windows NT: Diagnosis Guide*, SH12-6337
- *MERVA USE & Branch for Windows NT: Migration Guide*, SH12-6393
- *MERVA USE & Branch for Windows NT: Installation and Customization Guide*, SH12-6335
- *MERVA Workstation Based Functions*, SH12-6383

Other IBM Publications

- *CICS/ESA V4 Messages and Codes*, SC33-1177
- *CICS/VSE V2 Messages and Codes*, GC33-1481
- *IMS/ESA V5 Messages and Code*, SC26-8028
- *MQSeries for MVS/ESA Messages and Codes*, GC33-0819
- *MQSeries Application Programming Reference*, SC33-1673
- *MQSeries Programming: Sysplex Services Reference*, GC28-1772
- *Telex Interface Program: Program Description and Operations Manual*, SB11-8187

S.W.I.F.T. Publications

The following are published by the Society for Worldwide Interbank Financial Telecommunication, s.c., in La Hulpe, Belgium:

- *S.W.I.F.T. User Handbook*
- *S.W.I.F.T. Dictionary*
- *S.W.I.F.T. FIN Security Guide*
- *S.W.I.F.T. Card Readers User Guide*

Index

Special Characters

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

MERVA Requirement Request

Use the form overleaf to send us requirement requests for the MERVA product. Fill in the blank lines with the information that we need to evaluate and implement your request. Provide also information about your hardware and software environments and about the MERVA release levels used in your environment.

Provide a detailed description of your requirement. If you are requesting a new function, describe in full what you want that function to do. If you are requesting that a function be changed, briefly describe how the function works currently, followed by how you are requesting that it should work.

If you are a customer, provide us with the appropriate contacts in your organization to discuss the proposal and possible implementation alternatives.

If you are an IBM employee, include at least the name of one customer who has this requirement. Add the name and telephone number of the appropriate contacts in the customer's organization to discuss the proposal and possible implementation alternatives. If possible, send this requirement online to MERVAREQ at SDFVM1.

For comments on this book, use the form provided at the back of this publication.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Send the fax to:

To: MERVA Development, Dept. 5640
Attention: Gerhard Stubbe

IBM Deutschland Entwicklung GmbH
Schoenaicher Str. 220
D-71032 Boeblingen
Germany

Fax Number: +49-7031-16-4881
Internet address:
mervareq@de.ibm.com

MERVA Requirement Request

To: MERVA Development, Dept. 5640
Attention: Gerhard Strubbe

IBM Deutschland Entwicklung GmbH
Schoenaicher Str. 220
D-71032 Boeblingen Germany

Fax Number: +49-7031-16-4881
Internet address:
mervareq@de.ibm.com

Page 1 of _____

Customer's Name _____

Customer's Address _____

_____

_____

Customer's
Telephone/Fax _____

Contact Person at _____
Customer's Location
Telephone/Fax _____

MERVA
Version/Release _____

Operating System _____
Sub-System
Version/Release _____

Hardware _____

Requirement
Description _____

_____

_____

_____

_____

_____

_____

Expected Benefits _____

_____

_____

Readers' Comments — We'd Like to Hear from You

MERVA for ESA
Operations Guide
Version 4 Release 1

Publication No. SH12-6375-01

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name _____ Address _____

Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
SH12-6375-01

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape

PLACE
POSTAGE
STAMP
HERE

IBM Deutschland Entwicklung GmbH
Information Development, Dept. 0446
Schoenaicher Strasse 220
71032 Boeblingen
Germany

Fold and Tape | **Please do not staple** | Fold and Tape

SH12-6375-01

Cut or Fold
Along Line

Program Number: 5648-B29

SH12-6375-01

Spine information:

MERVA for ESA

Operations Guide

Version 4
Release 1